

# CAPTURING TEMPORAL ASPECTS OF BIO-HEALTH ONTOLOGIES

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER  
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY  
IN THE FACULTY OF SCIENCE & ENGINEERING

2016

By  
Jared Leo  
School of Computer Science

# Contents

<b>Abstract</b>	<b>13</b>
<b>Declaration</b>	<b>14</b>
<b>Copyright</b>	<b>15</b>
<b>Acknowledgements</b>	<b>16</b>
<b>1 Introduction</b>	<b>17</b>
1.1 Temporal Information in Ontologies . . . . .	17
1.2 Temporal Extensions to DLs . . . . .	18
1.3 Research Goals . . . . .	19
1.3.1 Research Questions . . . . .	20
1.4 Outline & Research Contributions . . . . .	20
<b>2 Background &amp; Related Work</b>	<b>22</b>
2.1 Description Logics . . . . .	23
2.1.1 $\mathcal{ALC}$ . . . . .	23
2.1.2 $\mathcal{EL}$ . . . . .	26
2.1.3 Useful Terms . . . . .	26
2.1.4 OWL & $\mathcal{SROIQ}(\mathcal{D})$ . . . . .	27
2.2 Temporal Logics and other Temporal Representations . . . . .	27
2.2.1 LTL . . . . .	27
2.2.2 CTL . . . . .	29
2.2.3 Allens Interval Relations . . . . .	30
2.3 Description Logics Extensions & Design Patterns . . . . .	31
2.3.1 $\text{LTL}_{\mathcal{ALC}}$ . . . . .	32
2.3.2 Other $\text{LTL}_{DL}$ combinations . . . . .	33

2.3.3	CTL <sub>ACC</sub> . . . . .	35
2.3.4	Other CTL <sub>DL</sub> combinations . . . . .	36
2.3.5	The Fluent Ontology . . . . .	36
2.3.6	Concrete Domains $\mathcal{D}$ . . . . .	39
2.4	Reasoning in Description Logics and their Extensions . . . . .	40
2.5	Known Complexity Results . . . . .	42
2.5.1	DLs . . . . .	42
2.5.2	LTL & CTL . . . . .	42
2.5.3	LTL <sub>DL</sub> & CTL <sub>DL</sub> . . . . .	42
2.5.4	$\mathcal{AC}(\mathcal{D})$ . . . . .	43
2.6	Biological Foundations in Ontologies . . . . .	44
2.7	Summary . . . . .	52
<b>3</b>	<b>Determining the Temporal Requirements</b>	<b>53</b>
3.1	Materials: Corpus & Temporal Features . . . . .	53
3.1.1	The OBO Foundry . . . . .	53
3.1.2	The Relation Ontology . . . . .	54
3.1.3	Temporal Features . . . . .	55
3.1.4	TRO - Temporal Relation Ontology . . . . .	60
3.2	Methodology . . . . .	61
3.3	Datasets & Implementation . . . . .	63
3.4	Results . . . . .	65
3.4.1	Ontology and Relation analysis . . . . .	65
3.4.2	Feature & annotation analysis . . . . .	66
3.4.3	<i>c-c</i> results . . . . .	67
3.4.4	<i>o-o</i> relations . . . . .	70
3.4.5	<i>c-o</i> results . . . . .	70
3.4.6	<i>o-c</i> results . . . . .	72
3.4.7	<i>x-x</i> results . . . . .	73
3.5	Identifying Requirements . . . . .	74
3.5.1	<i>x-x</i> requirements . . . . .	75
3.5.2	<i>c-c</i> requirements . . . . .	76
3.5.3	<i>o-o</i> requirements . . . . .	77
3.5.4	<i>c-o</i> requirements . . . . .	77
3.5.5	<i>o-c</i> requirements . . . . .	79
3.6	Temporal Requirements . . . . .	79

3.7	Validity of the Survey . . . . .	82
3.7.1	Analysis of Importance . . . . .	82
3.7.2	Annotations . . . . .	83
3.7.3	Smart matching . . . . .	84
<b>4</b>	<b>Temporal Extensions Evaluation</b>	<b>88</b>
4.1	$\text{LTL}_{\mathcal{ALC}}$ & $\text{CTL}_{\mathcal{ALC}}$ Evaluation . . . . .	90
4.1.1	Summary . . . . .	120
4.2	$\mathcal{ALC}(\mathcal{D})$ Evaluation . . . . .	121
4.2.1	Summary . . . . .	138
4.3	Fluents Evaluation . . . . .	139
4.3.1	Summary . . . . .	151
<b>5</b>	<b>[x]- A New Family of TDLs</b>	<b>155</b>
5.1	Introduction . . . . .	155
5.2	Defining the Syntax . . . . .	156
5.2.1	$\mathcal{ALC}_{[]}$ Syntax . . . . .	158
5.2.2	$\mathcal{ALC}_{[x]}$ Syntax . . . . .	160
5.3	Defining the Semantics . . . . .	162
5.3.1	$\mathcal{ALC}_{[]}$ Semantics . . . . .	163
5.3.2	$\mathcal{ALC}_{[x]}$ Semantics . . . . .	165
5.3.3	A Brief Overview of the Semantics . . . . .	166
5.3.4	$\mathcal{ALC}_{[]}[x]$ and $\mathcal{ALC}_{[x]}[]$ Syntax . . . . .	168
5.3.5	$\mathcal{ALC}_{[x]}[]$ Semantics . . . . .	168
5.3.6	$\mathcal{ALC}_{[]}[x]$ Semantics . . . . .	169
5.3.7	Domain Constraints . . . . .	170
5.3.8	<b>[x]</b> . . . . .	170
5.4	Evaluation of <b>[x]</b> . . . . .	171
5.4.1	Summary . . . . .	191
<b>6</b>	<b>Reasoning in Fragments of [x]</b>	<b>198</b>
6.1	Extending Classical DL Reasoning Tasks . . . . .	198
6.1.1	Reasoning problems in $\mathcal{ALC}_{[]}$ . . . . .	201
6.1.2	Reasoning problems in $\mathcal{ALC}_{[]}[x]$ . . . . .	202
6.1.3	Reasoning problems in $\mathcal{ALC}_{[x]}$ . . . . .	202
6.1.4	Reasoning problems in $\mathcal{ALC}_{[x]}[]$ . . . . .	203

6.2	A Decision Procedure for Classification in $\mathcal{EL}_{[]}$ . . . . .	204
6.3	A Decision Procedure for Computing Subsumption in $\mathcal{EL}_{[]}$ . . . . .	215
6.4	A Decision Procedure For Concept Satisfiability in $\mathcal{ALC}_{[]}$ . . . . .	217
6.5	A Decision Procedure for $\mathcal{ALC}_{[]}$ Ontology Consistency . . . . .	224
6.5.1	Other reasoning problems in $\mathcal{ALC}_{[]}$ . . . . .	230
6.6	A Decision Procedure for Classification in $\mathcal{EL}_{[][x]}$ . . . . .	231
6.7	A Decision Procedure for $\mathcal{ALC}_{[][x]}$ Ontology Consistency . . . . .	234
6.8	A Decision Procedure for Concept Satisfiability in $\mathcal{ALC}_{[x]}$ . . . . .	235
6.9	A Decision Procedure for Subsumption in Restricted $\mathcal{EL}_{[x]}$ . . . . .	236
6.9.1	A Decision Procedure for Classification in Restricted $\mathcal{EL}_{[x]}$ . . . . .	248
<b>7</b>	<b>TempDL: A Reasoner for <math>DL_{[]}</math></b> . . . . .	<b>249</b>
7.1	Overview . . . . .	249
7.2	$OWL_{[]}$ - Representing <b>[x]</b> in OWL . . . . .	249
7.3	Implementation of $TEMP_{\mathcal{EL}}$ & $TEMP_{\mathcal{ALC}}$ . . . . .	252
7.3.1	$TEMP_{\mathcal{EL}}$ System Description . . . . .	252
7.3.2	$TEMP_{\mathcal{ALC}}$ System Description . . . . .	257
7.4	Evaluation . . . . .	261
7.4.1	Overview . . . . .	261
7.4.2	Experiment Design . . . . .	261
7.4.3	Hypotheses . . . . .	266
7.4.4	Experimental Setup . . . . .	266
7.4.5	Experiment 1 Results . . . . .	266
7.4.6	Experiment 2 Results . . . . .	268
7.4.7	Experiment 3 Results . . . . .	271
7.5	Summary . . . . .	273
<b>8</b>	<b>Conclusion</b> . . . . .	<b>274</b>
<b>9</b>	<b>Outlook</b> . . . . .	<b>277</b>
	<b>Bibliography</b> . . . . .	<b>281</b>

# List of Tables

2.1	Semantics of Concept Descriptions in $\mathcal{ALC}$ . . . . .	25
2.2	Semantics of Temporal Concept Descriptions in $LTL_{\mathcal{ALC}}$ . . . . .	33
2.3	Semantics of Temporal Concept Descriptions in $CTL_{\mathcal{ALC}}$ . . . . .	36
2.4	Semantics of Concept Descriptions in $\mathcal{ALC}(\mathcal{D})$ . . . . .	40
2.5	Known complexity results for DLs and their reasoning services. $\ast =$ Empty Ontology. In $\mathcal{ALC}$ satisfiability, subsumption and consistency are polynomial-time inter-reducible to one another [BCM <sup>+</sup> 03]. So the results for $\mathcal{ALC}$ hold for each reasoning problem. . . . .	42
2.6	Known Complexity Results for $LTL_{DL}$ and $CTL_{DL}$ combinations and fragments. ED = Expanding domains, CD = constant domains, RR = rigid roles, 1-RR = 1 rigid role . . . . .	43
3.1	Coverage and impact over ontology corpora in Figure 3.3 . . . . .	63
3.2	Top 10 relations used across the OBO Foundry corpus. $ \mathcal{O} $ is the number of ontologies the relation is mentioned in. . . . .	66
3.3	Coverage and Impact results for $c-c$ features used across the OBO Foundry . . . . .	67
3.4	Coverage and Impact results for $c-c$ annotations used across the OBO Foundry . . . . .	68
3.5	Coverage and Impact results for $o-o$ features used across the OBO Foundry . . . . .	69
3.6	Coverage and Impact results for $o-o$ annotations used across the OBO Foundry . . . . .	69
3.7	Coverage and Impact results for $c-o$ features used across the OBO Foundry . . . . .	70
3.8	Coverage and Impact results for $c-o$ annotations used across the OBO Foundry . . . . .	71

3.9	Coverage and Impact results for $o-c$ features used across the OBO Foundry . . . . .	72
3.10	Coverage and Impact results for $o-c$ annotations used across the OBO Foundry . . . . .	72
3.11	Coverage and Impact results for $x-x$ features used across the OBO Foundry . . . . .	73
3.12	Coverage and Impact results for $x-x$ annotations used across the OBO Foundry . . . . .	73
3.13	Average Coverage and Impact scores for each feature type . . . . .	73
3.14	Average Coverage and Impact scores for each annotation type . . . . .	74
3.15	Important $c-c$ features used and selected in the temporal requirements with their Coverage and Impact scores . . . . .	86
3.16	Important $o-o$ features used and selected in the temporal requirements with their Coverage and Impact scores . . . . .	86
3.17	Important $c-o$ features used and selected in the temporal requirements with their Coverage and Impact scores . . . . .	86
3.18	Important $o-c$ features used and selected in the temporal requirements with their Coverage and Impact scores . . . . .	86
3.19	Important $x-x$ features used and selected in the temporal requirements with their Coverage and Impact scores . . . . .	87
3.20	Important Annotations used and selected in the temporal requirements with their Coverage and Impact scores . . . . .	87
4.1	TRO important annotations with a corresponding TRO relation . . . . .	89
4.2	TR1 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	152
4.3	TR2 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	152
4.4	TR3 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	152
4.5	TR4 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.6	TR5 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.7	TR6 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.8	TR7 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.9	TR8 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.10	TR9 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.11	TR10 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	153
4.12	TR11 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	154
4.13	TR12 Scores evaluated against $CTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	154

4.14	TR13 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	154
4.15	TR14 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ and FL . . . . .	154
4.16	R15 Scores evaluated against $LTL_{\mathcal{ALC}}$ (and $CTL_{\mathcal{ALC}}$ ), $\mathcal{ALC}(\mathcal{D})$ and FL	154
5.1	TRO important annotations with a corresponding TRO relation .	171
5.2	Complexity Results (membership only) for fragments of $[x]$ with Constant Domains and unary encodings of interval boundaries. ( $\rightarrow$ =future only) . . . . .	190
5.3	TR1 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	195
5.4	TR2 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	195
5.5	TR3 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	195
5.6	TR4 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	196
5.7	TR5 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	196
5.8	TR6 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	196
5.9	TR7 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	196
5.10	TR8 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	196
5.11	TR9 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	196
5.12	TR10 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . .	196
5.13	TR11 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . .	197
5.14	TR12 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . .	197
5.15	TR13 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . .	197
5.16	TR14 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . .	197
5.17	R15 Scores evaluated against $LTL_{\mathcal{ALC}}$ , $\mathcal{ALC}(\mathcal{D})$ , FL and $[x]$ . . . .	197
6.1	Growth of an $\mathcal{EL}_{[]}$ TBox after single application of normalisation rules . . . . .	207
7.1	$\mathcal{EL}_{[]}$ normalisation rules . . . . .	255
7.2	Correlation (Spearman Coefficient) Table for all ontologies. $\mathcal{O}$ : Ontology, O-CT: Overlap/Classification Time, CT-E: Classifica- tion Time/Entailment Count O-E1: Overlap/Entailment Count, O-E1: Overlap/Entailment Count with Temporal Tautologies Re- moved, . . . . .	268



# List of Figures

2.1	Allens's Interval Relations [All83] . . . . .	31
2.2	An example usage of the Fluent Ontology . . . . .	38
2.3	A Spermatid cell transitioning through 5 stages of development, contained entirely within a Spermatocyte Cyst, making use of the relations <i>hasPart</i> and <i>developsFrom</i> . . . . .	46
2.4	An example model view of the Spermatid cell development ex- tracted from the Drosophila Ontology modelled in $LTL_{\mathcal{AC}}$ . Each Spermatid is represented by the same individual, similarly for the SpermatocyteCyst. <i>CS</i> = <i>CoalescenceSpermatid</i> , <i>AS</i> = <i>Agglomer- ationSpermatid</i> , <i>CS</i> = <i>ClewSpermatid</i> , <i>OS</i> = <i>OnionSpermatid</i> , <i>LS</i> = <i>LeafbladeSpermatid</i> . . . . .	47
2.5	An example model view of the Spermatid cell development ex- tracted from the Drosophila Ontology modelled in $\mathcal{AC}(\mathcal{D})$ . Only the first 3 stages are used in this example. A set of intervals and Allen's interval relations make up the concrete domain. <i>CS</i> = <i>CoalsecenceSpermatid</i> , <i>AS</i> = <i>AgglomerationSpermatid</i> , <i>CS</i> = <i>ClewSpermatid</i> . . . . .	49
3.1	An example of rigid relations. (1) shows an example of finite rigid- ity between continuants, (2) shows an example of infinite rigidity between continuants holding only at 3 consecutive time points, (3) shows an example of finite rigidity between occurrents during the times they exist and (4) shows an example of non rigidity between non identical continuants . . . . .	57
3.2	Illustration of the relation <i>transformation of</i> where a single con- tinuant is related to itself at some earlier time point . . . . .	57
3.3	Example of coverage and impact measures for calculating impor- tance of an <i>e</i> of two corpora of 5 ontologies. . . . .	64

3.4	Histograms of RO relation usage across OBO Foundry ontologies. Left: Proportion of axioms using RO or RO-like relations compared to the overall number of axioms (in %). Right: Proportion of RO or RO-like relations used in the ontology compared to the total number of RO relations (459). . . . .	65
3.5	A histogram showing the number of ontologies in the OBO Foundry against the % of axioms in each ontology. . . . .	85
4.1	An illustration of the rigid interpretation of elements in $LTL_{ACC}$ , and how they can be used for the effective modelling of Continuants and the modelling of change. . . . .	91
4.2	An illustration of the relation <i>part of</i> . In this example a continuant C1 is part of a continuant C2 for 3 consecutive time points, starting at time $t$ , and then is no longer related to related to C2. . . . .	98
4.3	An illustration of the relation <i>aligned with</i> . In this example the continuant $A$ is aligned with the continuant $B$ at a single time point $t$ , and the relation holds only at this single time point. . . .	101
4.4	An illustration of the relation <i>develops from</i> . In this example, the continuant $A$ develops from the continuant $B$ at times $t$ and $t'$ respectively where $t > t'$ . The continuants may be identical, although this is not necessary. . . . .	102
4.5	An illustration of the relation <i>child nucleus of</i> . In this example the continuant $C$ is a child nucleus of the continuant $D$ which existed at some previous time point $t$ when $D$ did not exist. . . . .	103
4.6	An illustration of the <i>causally downstream of</i> relation. In this example the occurrent $O1$ appears causally downstream of the occurrent $O2$ simulating Allen's interval relation <i>after</i> . . . . .	106
4.7	An illustration of the <i>happens during</i> relation. In this example the occurrent $O1$ happens during the occurrent $O2$ , simulating Allen's interval relation <i>during</i> . . . . .	107
4.8	An illustration of the <i>precedes</i> relation. In this example the occurrent $O1$ precedes the occurrent $O2$ , simulating Allen's interval relation <i>before</i> . . . . .	108
4.9	An illustration of the <i>immediately preceded by</i> relation. In this example the occurrent $O1$ is immediately preceded by the occurrent $O2$ , simulating Allen's interval relation <i>meets'</i> . . . . .	109

4.10	An illustration of the <i>immediately causally upstream of</i> relation. In this example the occurrent <i>O1</i> is immediately causally upstream of the occurrent <i>O2</i> , simulating Allen's interval relation <i>meets</i> . . .	109
4.11	An illustration of the relation <i>input of</i> , showing a continuant <i>C</i> being the input of an occurrent <i>O</i> at a single time point. . . . .	111
4.12	An illustration of the relation <i>existence starts during</i> , showing a continuant <i>C</i> 's existence starting during the life span of an occurrent <i>O</i> . . . . .	113
4.13	An illustration of the relation <i>existence starts during or after</i> , showing two continuants <i>C</i> and <i>C'</i> whose existence starts during and after the life span of an occurrent <i>O</i> respectively. . . . .	113
4.14	An illustration of relations <i>existence ends during or before</i> and <i>existence ends at point</i> . Both <i>C</i> and <i>C'</i> 's existence end during or before <i>O</i> , whereas only <i>C</i> 's existence ends at the point of <i>O</i> 's existence. . . . .	115
4.15	An illustration of the relation <i>involved in</i> , showing a continuant <i>C</i> being involved in an occurrent <i>O</i> for multiple consecutive time points over a fixed duration over the life span of <i>O</i> . . . . .	115
4.16	An illustration of the relation <i>carries out</i> . In this example, there exists a time line where the continuant <i>C</i> carries out the occurrent <i>O</i> and another time line where this does not happen. . . . .	117
4.17	An illustration of the relation <i>occurs in</i> . In this example, an occurrent <i>O</i> occurs in an continuant <i>C</i> during its entire life span. . .	118
4.18	An example of modelling continuants in $\mathcal{AC}(\mathcal{D})$ . An element <i>a</i> being an instance of a continuant <i>C</i> is related to do different time points in the first example. Then reification is used in the second example to show how it can be split in to two <i>temporal parts</i> . . . .	123
5.1	An example of localised rigidity in $[\mathbf{x}]$ , showing an element <i>A</i> being <i>R</i> related to an element <i>B</i> for 3 consecutive time points. . . . .	167
5.2	An illustration of the rigid interpretation of elements in $[\mathbf{x}]$ , and how they can be used for the effective modelling of Continuants and the modelling of change. . . . .	172
5.3	A collection of six occurrents organised along a finite time line of ten points . . . . .	185

6.1	An illustration of the tableau procedure for $\mathcal{ALC}_{[]}$ using single ABoxes and multiple sequences of ABoxes to maintain a tree structure. . . . .	219
7.1	Relationship between average interval overlap and classification time in seconds. . . . .	269
7.2	Relationship between average interval overlap and number of resulting subsumptions. . . . .	270
7.3	Relationship between average interval overlap and number of resulting subsumptions with tautologies removed. . . . .	272

# Abstract

## CAPTURING TEMPORAL ASPECTS OF BIO-HEALTH ONTOLOGIES

Jared Leo

A thesis submitted to the University of Manchester  
for the degree of Doctor of Philosophy, 2016

Extending Descriptions Logics (DLs) with a temporal dimension to aid in the ability to model meaningful temporal information is an active and popular research area that has gathered a lot of attention over recent years. DLs underpin the Web Ontology Language (OWL) which offers a way to describe ontologies for the semantic web. Representing temporal information in ontologies plays an important role, specifically for those ontologies where time information is inherently embedded in the information they describe. This is very common for ontologies in the bio-health domain, for example ontologies that describe the development of anatomies of biological entities, stage based development, evolution of diseases and so on. As expressive as DLs are, given that they are fragments of First Order Logic, they are static in nature and are limited in what they can express from a temporal view point, hence the surge in temporal extensions to DLs over recent years.

In this thesis we investigate the use of temporal extensions of DLs as suitable representations for the temporal information required for bio-health ontologies. We first set out to find out exactly what types of temporal information need to be modelled, before going on to evaluate current temporal extensions and representations to determine their suitability. We then go on to introduce several new temporal extensions to DLs and evaluate their suitability.

# Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

# Copyright

- i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the “Copyright”) and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.
- ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.
- iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.
- iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see <http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487>), in any relevant Thesis restriction declarations deposited in the University Library, The University Library’s regulations (see <http://www.manchester.ac.uk/library/aboutus/regulations>) and in The University’s policy on presentation of Theses

# Acknowledgements

Firstly, I would like to thank my supervisors Uli Sattler and Bijan Parsia. You have both been an inspiration to me and I very much appreciate all your support and wisdom. Your encouragement and motivation lead to my success and without you this would have not been possible. Thank you.

Secondly, I would like to thank my examiners Robert Stevens and Thomas Schneider for making my viva an enjoyable and memorable experience.

A much needed thank you to Nico for all the help and support given to me over the last 4 years. Whether it was programming the OWL API, setting up and running experiments, making music, recording temperatures, providing counter examples, sanity checking my ideas, prepping me for me viva or even eliminating methods with 39 parameters, you were always there to help. Thank you.

I would also like to thank Melanie Price for always leaving her door open for my unscheduled visits for the last four years. You never failed to brighten up my day.

A special thanks to Dominik and Aharron who kept my social life alive during the past four years. They reminded me that although we may be separated by thousands of miles, a Los Santos reunion is ever only an arm's reach away. And Tyanna, I urged myself to succeed if only to push you further. Thank you all.

And finally to Nastia, my untimely yet pleasant surprise. Thank you for your much needed support over the last few months.



# Chapter 1

## Introduction

### 1.1 Temporal Information in Ontologies

Description Logics (DLs) [BCM<sup>+</sup>03, BS01, KSH12] are a collection of knowledge representation languages that can be used to express the knowledge of some application domain in a precise and structured way. DLs are widely used in ontological modelling and they provide the core formalisms and structure used by OWL - the web ontology language, as standardised by the World Wide Web Consortium (W3C). The DL *SRQL* [HKS06], as the logical core for OWL 2 [GHM<sup>+</sup>08], is a widely used formalism for ontologies in the life sciences. BioPortal [NSW<sup>+</sup>09] and the OBO Foundry [SAR<sup>+</sup>07] are two examples of ontology corpora for the biomedical domain. They contain a variety of ontologies describing different types of biomedical domain knowledge, ranging from vocabularies for clinical care and medical terminology systems such as National Cancer Institute Thesaurus (NCIT) [SdCH<sup>+</sup>07] and anatomy and stage based developmental vocabularies such as the Drosophila Gross Anatomy Ontology [CRGOS13].

Different classes of temporal phenomena may generate different sorts of requirements on extensions of *SRQL*. As expressive as ontologies and their underlying DLs are, there are still limiting factors over what they can and cannot express. Many bio-health concepts involve temporal aspects. Consider an ontology describing the development of any biological entity. Any development inherently involves time: statements made in the ontology could include (1) an element developing from another, or (2) any entity taking place during a certain stage, or (3) an event occurring before, after or during another event. All involve specific types of temporal information. Take for example an ontology describing

human fetal development. Statements could be made about specific embryonic cells developing from other cells (1), organ development taking place in certain trimesters (2), and certain gestational ages occurring during specific weeks in the pregnancy (3). Each example clearly includes specific temporal information and requires this information to faithfully represent what is intended. It would be beneficial to have some sense of time encoded into the underlying logic, allowing us to represent and query knowledge in the past or present or future. OWL 2 does offer *a* way to encode some type of temporal information, through time stamping (data types), but offers no way to describe any real type of *change* since it is still a static logic, being a fragment of First Order Logic. The temporal information that is present in these ontologies is usually based on concepts changing (developing) over time, or representations of sequences of stages, both against some representation of time. Since DLs are static, they have no inherent concept of properties changing over time, i.e formulae are evaluated within only a single fixed world. Clearly if time information is needed but cannot be represented then it may be the case that many of these ontologies are currently misrepresented, or at least OWL does not have the required expressivity to meet the temporal requirements of these ontologies. It is not currently clear exactly what kind of temporal expressivity is required of OWL to meet the temporal requirements of bio-health ontologies, simply because the temporal requirements of these ontologies are not yet known.

## 1.2 Temporal Extensions to DLs

Temporal extensions to DLs have been given a lot of attention in recent years. Many proposals exist, ranging from taking classical temporal logics such as LTL, CTL or CTL\* and combining them with DLs such as  $\mathcal{EL}$  or  $\mathcal{ALC}$  [LWZ08, GJS15b] where the result can be seen as a two dimensional Temporal Description Logic (TDL). Alternatively, temporal information has been added via means of extending DLs with a concrete domain to act as a temporal referencing scheme [BH91], or even internalising temporal information by embedding it into standard OWL via means of *temporal ontologies*, for example representing Fluents via a Fluent Ontology [WF06], or a dedicated Time Ontology [HP04].

Very few of these TDLs have been investigated for a *specific* application purpose however. For example, in recent years, research on two dimensional TDLs

has been mainly focussed on complexity results rather than any specific application domain [LWZ08], similarly for DLs extended with concrete domain [Lut02]. We believe this is due to the fact that both have very interesting complexity results [GJS15b, LWZ08, Lut02]: it is very easy for these logics to enter into the undecidability realm, which is extremely undesirable for DLs and ontologies. Decidability is a must - only decidable logics can be DLs, partly for historical reasons. It can be argued that the reason for the success of OWL is due to efficiency and decidability when it comes to reasoning i.e., inferring new information from an ontology. If such extensions easily lead to undecidability, even in some of the most simplest cases, it is definitely an interesting research area to consider. One of the main culprits for undecidability in two-dimensional DLs are *rigid roles*. On a high level, a role is simply a relation between two entities in some domain of interest. A rigid role is one that remains constant throughout time. For example, if the relation *love* was rigid, then if *bob* was to *love mary* at some time point, then at any time point it would hold that *bob loves mary*. Rigid roles seem crucial for biological domains. For example, if *bob* has a blood type *A RhD positive (A+)*, then it should be the case that *bob* always has this blood type. It would not make sense for *bob* to not have the same blood type at some other time point. What seems like a seemingly harmless extension is what causes most of the two-dimensional TDL extensions to become undecidable [GJS15a, LWZ08], which is a shame since these logics are the ones most promising to describe concepts changing over time. DLs with concrete domains are useful for capturing temporal notions such as durations and interval relations, but not so much ideas such as rigidity and general change.

It may be the case that some of the proposed extensions may in fact be suitable for modelling the temporal requirements of these ontologies (rigid roles may not be as important as other temporal features). If the requirements were known, we could evaluate the current proposals, to see which were most suited, and if none were, we could set out to define a new logic based on these requirements to attempt to solve this problem.

### 1.3 Research Goals

We know that there are cases where OWL is not sufficient to model some of the necessary temporal information that may be required by ontologies but we do not

know exactly what all or any of the temporal requirements are, how important these requirements would be, what potential effect they could have on modelling, and what benefits they could have in practice. We also do not know if any of the temporal extensions that have been proposed are sufficient to *faithfully* model the temporal nature of these ontologies and which ones would be better suited for the task.

The goal of this thesis is to research exactly these problems. We wish to identify a suitable methodology to determine a clear set of Temporal Requirements for modelling the temporal patterns in bio-health ontologies, which we can then use to effectively evaluate and compare current temporal extensions to OWL, to see which are most suited to act as a temporal representation. If no such extension is suitable, then if possible, we will either attempt to extend the best current extension or create a new extension that will attempt to meet the requirements.

### 1.3.1 Research Questions

The research questions this thesis addresses are as follows:

1. *What are the temporal requirements for modelling the temporal features of Bio-Health ontologies in OWL?*
2. *Is there currently a suitable temporal extension for DLs that will allow for a faithful representation of the Temporal Requirements? And one that is decidable and of suitably low complexity?*
3. *How can we extend DLs to accommodate for the Temporal Requirements?*

## 1.4 Outline & Research Contributions

The following is a summary of our research contributions and a general outline of the thesis structure:

- **Background & Related Work (Chapter 2)** In Chapter 2 we lay down the foundations required to understand the research in this thesis. We formally introduce several DLs, temporal logics, TDLs, and other logical

representations. We also introduce several terminologies and provide a running example of how temporal information is currently modelled in OWL and how it can be modelled in TDLs.

- **Temporal Requirements Identification (*Chapter 3*)** In Chapter 3 we perform a survey on the OBO Foundry and a temporal version of the Relation Ontology [SCK<sup>+</sup>05] to determine a precise set of 15 Temporal Requirements (TRs) of biomedical ontologies.
- **Temporal Extensions & Representations Evaluation (*Chapter 4*)** In Chapter 4 we take the 15 TRs and thoroughly evaluate and compare 3 carefully selected temporal extensions and representations of OWL introduced in Chapter 3.
- **A new Family of Temporal Description Logics (*Chapter 5*)** In Chapter 5 we describe a new family of TDLs based on time point intervals and tailored towards the TRs, and evaluate it against the TRs.
- **Reasoning in New TDLs (*Chapter 6*)** In Chapter 6 we describe various new reasoning problems for our new TDL and prove various complexity results for the new TDL.
- **Reasoners (*Chapter 7*)** In Chapter 7 we introduce two new OWL Reasoners for several of the new languages introduced in Chapter 5 and provide experiments to show their practical capabilities and their benefits as to modelling temporal information in OWL.

# Chapter 2

## Background & Related Work

In this chapter we introduce and discuss background knowledge and related work associated with the research of this thesis. These include:

1. The basic concepts of Description Logics and their use as formalisms for ontologies and OWL.
2. The introduction of Temporal Logics (TLs) and their use for representing temporal information.
3. The introduction of several temporal extensions and representations of DLs.
4. The description and discussion of the common reasoning problems in DLs.
5. The description of several biological notions and terminologies in ontologies.

(1) outlines the core formalisms of two popular DLs, defining their syntax and semantics, and how they can be used to build ontologies, fixing notions such as *axioms*, *TBoxes*, *ABoxes* etc. (2) introduces several TLs, again defining their syntax and semantics. (3) outlines several extensions to classical DLs, including temporal extensions and combinations of DLs and those TLs seen in (2), as well as other extensions and temporal representations, including Fluents and concrete domains. (4) introduces the most common reasoning problems for description logics and their temporal extensions outlined in (3) along with known complexity results for their reasoning problems. (5) introduces several biological notions used throughout bio-health ontologies. We discuss notions of *identity*, *continuants* and *occurents*.

## 2.1 Description Logics

Our aim is not to provide an overview of the entirety of Description Logics (DLs), or OWL, but merely to introduce the foundation of DLs necessary for the work in this thesis. For this reason, we introduce two popular DLs,  $\mathcal{ALC}$  and  $\mathcal{EL}$ .

### 2.1.1 $\mathcal{ALC}$

DLs [BCM<sup>+</sup>03, BS01, KSH12] are a collection of knowledge representation languages that can be used to express the knowledge of some application domain in a precise and structured way. As their name suggests, DLs are logics, which means they are supplied with a precise syntax and semantics, and come equipped with the ability to reason with information in a meaningful way. One of the main aspects of DLs is to (1) provide ways to model relationships between 3 kinds of entities in a domain of interest, these being concept descriptions, roles and individual names and (2) to make more expressive terms, usually called concept expression, axioms, assertions and even ontologies. There are many varieties of DLs and they differ by what constructors, axioms and operators are allowed (see [BCM<sup>+</sup>03, KSH12] for more details).  $\mathcal{ALC}$  (*Attributive Logic with Complement*) [SSS91] is a simple yet expressive DL. Concept descriptions in  $\mathcal{ALC}$  are built according to the following definition:

**Definition 1** ( $\mathcal{ALC}$  Concept Descriptions)

Let  $N_{con}$ ,  $N_{role}$  and  $N_{ind}$  be countable infinite and disjoint sets of concept, role and individual names respectively. Let  $A \in N_{con}$ ,  $R \in N_{role}$   $C, D$  be arbitrary concept descriptions. Then concept descriptions can be formed in  $\mathcal{ALC}$  according to the following syntax rules:

$$C, D \longrightarrow A \mid \top \mid \perp \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

The operators  $\{\sqcap, \sqcup, \neg, \forall, \exists, \top, \perp\}$  are called *conjunction*, *disjunction*, *negation*, *universal restriction*, *existential restriction*, *top concept* and *bottom concept* respectively. DLs are in fact a decidable fragment of first order logic (FOL), so those familiar with FOL may wish to interpret concept, role and individuals names as unary predicates, binary predicates and constants respectively.

When representing concept names, unless using specific names of things, we will always use upper case letters to represent them, such as  $A, B, C, \dots$  possibly

sub or superscripted with indexes, for example  $A_1$  or  $A'$  etc. Concept names may also be referred to as *class names* or even *atomic classes* (we may use these terms synonymously throughout this thesis). We proceed similarly for role names. In their representation we do our best to ensure the concept and roles names are kept disjoint. We normally use  $R, S, P$  or  $Q$  to represent role names, if not using specific names in the appropriate context. Individual names will always be shown with lower case letters, for example,  $a, b, c, \dots$

Terminological axioms make statements about how concepts or roles are related to one another. They intend to constrain the domain of interest. The axioms come in the form  $C \sqsubseteq D$  or  $C \equiv D$ . The first is called a *subclass of* axiom (or a *subsumption*) and reads “ $C$  is a subclass of  $D$ ” or simply “ $C$  is a  $D$ ”. The second is called an *equivalence* axiom and reads “ $C$  is equivalent to  $D$ ”.

**Definition 2** ( $\mathcal{ALC}$  Terminological Axioms & TBoxes)

Let  $C, D$  be arbitrary  $\mathcal{ALC}$  concept descriptions. Terminological axioms are of the form

$$C \sqsubseteq D \text{ or } C \equiv D \quad (2.1)$$

An  $\mathcal{ALC}$  TBox is a finite set of these axioms.

Assertions make statements between individuals and concept and roles, providing a mechanism to assert how to relate one to the others. Assertions are of the form  $C(a)$  and  $R(a, b)$ . The first is called a concept assertion and reads “ $a$  is an instance of  $C$ ”. The second is called a role assertion and reads “ $a$  is  $R$  related to  $b$ ” or “ $R$  relates  $a$  to  $b$ ”.

**Definition 3** ( $\mathcal{ALC}$  Assertions & ABoxes)

Let  $C$  be an arbitrary  $\mathcal{ALC}$  concept description,  $R \in N_{role}$  and  $a, b \in N_{ind}$ . Assertional axioms are of the form

$$C(a) \text{ or } R(a, b) \quad (2.2)$$

An  $\mathcal{ALC}$  ABox is a finite set of these axioms.

TBoxes are seen to capture knowledge on a *conceptual* level whereas ABoxes capture knowledge on an *individual* level, using terms from the conceptual level. A TBox and an ABox together form an ontology (or a knowledge base). In the DL context, we use the term ontology to specifically refer to a set of axioms and assertions.



Name	Syntax	Semantics
Top	$\top$	$\Delta^{\mathcal{I}}$
Bottom	$\perp$	$\emptyset$
Atomic	$A$	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
Conjunction	$(C \sqcap D)$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Disjunction	$(C \sqcup D)$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Value	$(\forall R.C)$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y : (x, y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
Existential	$(\exists R.C)$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$

Table 2.1: Semantics of Concept Descriptions in  $\mathcal{ALC}$ **Definition 4** ( $\mathcal{ALC}$  Ontology)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}$  TBox, and  $\mathcal{A}$  be an  $\mathcal{ALC}$  ABox. An  $\mathcal{ALC}$  ontology  $\mathcal{O}$  is of the form  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ .  $\mathcal{O}$  can also be seen as the set  $\mathcal{O} := \mathcal{T} \cup \mathcal{A}$

We will tend to use  $\mathcal{T}$  to refer to a TBox,  $\mathcal{A}$  for an ABox, and  $\mathcal{O}$  for an ontology.

DLs have a model theoretic semantics which is given in terms of an interpretation  $\mathcal{I}$ :

**Definition 5** (Semantics of  $\mathcal{ALC}$ :  $\mathcal{I}$ )

An  $\mathcal{ALC}$  interpretation  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$  consists of a non-empty set  $\Delta^{\mathcal{I}}$  its domain, and a function  $\cdot^{\mathcal{I}}$  that maps each concept name  $A \in N_{con}$  to a subset  $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ , each role name  $R \in N_{role}$  to a subset  $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$  and each individual name  $a \in N_{ind}$  to  $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ . The function  $\cdot^{\mathcal{I}}$  is inductively extended to arbitrary concept descriptions shown in Table 2.1. Let  $C$  and  $D$  be arbitrary concept descriptions and  $A, R$  and  $e, f$  be concept, role and individual names from  $N_{con}$ ,  $N_{role}$  and  $N_{ind}$  respectively.

- $C$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C^{\mathcal{I}} \neq \emptyset$ , in which case  $\mathcal{I}$  is called a model of  $C$
- $\mathcal{I}$  satisfies a TBox axiom  $C \sqsubseteq D$  if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$  satisfies a TBox axiom  $C \equiv D$  if  $C^{\mathcal{I}} = D^{\mathcal{I}}$
- $\mathcal{I}$  is a model of a TBox  $\mathcal{T}$  if it satisfies every axiom in  $\mathcal{T}$
- $\mathcal{I}$  satisfies a concept assertion  $C(e)$  if  $e^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$  satisfies a concept assertion  $R_{\lambda}(e, f)$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_{\lambda}^{\mathcal{I}}$

- $\mathcal{I}$  is a model of an ABox  $\mathcal{A}$  if it satisfies every axiom in  $\mathcal{A}$
- $\mathcal{I}$  is a model of an ontology  $\mathcal{O}$  if it satisfies every axiom in  $\mathcal{O}$

### 2.1.2 $\mathcal{EL}$

The DL  $\mathcal{EL}$  [BBL05, Baa03], is a lightweight DL known for its limited available constructs against its impressive level of expressivity.  $\mathcal{EL}$  only allows for conjunction and existential restrictions and boasts polynomial time complexity. Although only a few operators are available, it is sufficient for many applications, particularly those in the bio-health domain. Very large and popular ontologies such as SNOMED-CT [SCS11] describing over 300,000 concepts can be almost completely represented in  $\mathcal{EL}$  and many other bio-health ontologies (for example many of those in the OBO Foundry) can be completely captured by  $\mathcal{EL}$ .  $\mathcal{EL}$  can be seen as a strict fragment of  $\mathcal{ALC}$ . The grammar of its syntax is defined as follows:

**Definition 6** ( $\mathcal{EL}$  Concept Descriptions)

Let  $N_{con}$ ,  $N_{role}$  and  $N_{ind}$  be countable infinite and disjoint sets of concept, role and individual names respectively. Let  $A \in N_{con}$ ,  $R \in N_{role}$   $C, D$  be arbitrary concept descriptions. Then concept descriptions can be formed in  $\mathcal{EL}$  according to the following syntax rules:

$$C, D \longrightarrow \top \mid A \mid C \sqcap D \mid \exists R.C$$

$\mathcal{EL}$  TBoxes, ABoxes and Ontologies are defined in the same way as in  $\mathcal{ALC}$ , with the obvious restrictions based on the allowed logical constructs. The same goes for the semantics of  $\mathcal{EL}$ .

### 2.1.3 Useful Terms

**Signature** Given an Ontology  $\mathcal{O}$ , a TBox  $\mathcal{T}$  and an ABox  $\mathcal{A}$ , the signature of each, denoted as  $\tilde{\mathcal{O}}, \tilde{\mathcal{T}}, \tilde{\mathcal{A}}$  respectively, is the set of all concept, role and individual names occurring in each set.

### 2.1.4 OWL & $\mathcal{SROIQ}(\mathcal{D})$

The Web Ontology Language (OWL) is a collection of knowledge representation languages designed for use in many applications to provide means to model information in a precise and structured way amongst the semantic web. OWL 2 [GHM<sup>+</sup>08] is the current iteration (and successor) of OWL, which has two levels of expressivity, OWL 2 DL and OWL 2 Full. The former has a DL,  $\mathcal{SROIQ}(\mathcal{D})$  [HKS06], as a logical basis.  $\mathcal{SROIQ}(\mathcal{D})$  can be seen as a logical extension to  $\mathcal{ALC}$ . Each letter in the name can be seen to roughly refer to a set of logical constructs and axiom types:

- $\mathcal{S}$  =  $\mathcal{ALC}$  extended with transitive roles.
- $\mathcal{R}$  = a set of role constraints including, role chains, inclusions, hierarchies, role characteristics such as irreflexivity and reflexivity and role disjointness
- $\mathcal{O}$  = nominals
- $\mathcal{I}$  = inverse roles
- $\mathcal{Q}$  = qualified number restrictions
- $\mathcal{D}$  = data types

When we refer to OWL, we consider only OWL 2 DL and not OWL 2 Full. Therefore, in this thesis whenever we use the term OWL, we really are referring to OWL 2 DL.

## 2.2 Temporal Logics and other Temporal Representations

### 2.2.1 LTL

Linear Temporal Logic (LTL) is a type of modal logic, first introduced by Pnueli [Pnu77]. LTL was primarily intended to model formal systems during the specification and verification phases of software building, but over the past decade its use has spread to a wider range of applications [DGFvdH07, Fis11]. LTL is based on propositional logic and uses operators from modal logic to model time, usually along a discrete and linear timeline. Formulae in LTL are constructed from the following definition:

**Definition 7**

Let  $\text{PROP}$  be a finite set of propositional symbols such as  $p, q, r, \dots$ ,  $\{\mathbf{true}, \mathbf{false}, \wedge, \vee, \neg, \Rightarrow\}$  be the set of propositional connectives and  $\{\bigcirc, \Diamond, \Box, \mathcal{U}, \mathcal{W}\}$  be the set of temporal connectives.

The set of LTL well-formed-formulae (WFF) is defined as follows:

- $\text{PROP} \subseteq \text{WFF}$
- $\{\mathbf{true}, \mathbf{false}\} \subseteq \text{WFF}$
- If  $\varphi$  and  $\psi$  are in WFF, then so are
 

$\neg \varphi$	$\varphi \Rightarrow \psi$	$\Box \varphi$
$\varphi \vee \psi$	$\Diamond \varphi$	$\varphi \mathcal{U} \psi$
$\varphi \wedge \psi$	$\bigcirc \varphi$	$\varphi \mathcal{W} \psi$

The unary modal operators  $\bigcirc, \Diamond$  and  $\Box$  are usually interpreted as “at the next time point”, “sometime in the future” and “at all future moments”, respectively. The binary modal operators  $\mathcal{U}$  and  $\mathcal{W}$  are usually interpreted as “until” and “unless”, respectively. Models of TLs are usually defined as a Kripke Structure  $\mathcal{M} = \langle \mathcal{S}, \mathcal{R}, \pi \rangle$ , where  $\mathcal{S}$  is the set of moments in time,  $\mathcal{R}$  is the (temporal) accessibility relation on  $\mathcal{S}$  and  $\pi$  is a propositional valuation function where  $\pi : \mathcal{S} \rightarrow \mathbb{P}(\text{PROP})$  which maps each moment (world) to a set of propositions that are true in that moment (world). However, since we are concerned with LTLs,  $\mathcal{R}$  is seen as a linear and discrete relation which is usually isomorphic to  $\mathbb{N}$ , so we can reduce the structure to  $\mathcal{M} = \langle \mathbb{N}, \pi \rangle$  where  $\pi : \mathbb{N} \rightarrow \mathbb{P}(\text{PROP})$ . The semantics of a temporal WFF is provided by the satisfaction relation

$$\models : (\mathcal{M} \times \mathbb{N} \times \text{WFF}) \longrightarrow \{\mathbf{true}, \mathbf{false}\}$$

s.t.  $\langle \mathcal{M}, i \rangle \models \varphi$  is true if  $\varphi$  is satisfied at time point  $i$  in the structure  $\mathcal{M}$  and false otherwise. The semantics of the propositional operators is as expected:

$$\begin{aligned} \langle \mathcal{M}, i \rangle \models \varphi \wedge \psi & \text{ iff } \langle \mathcal{M}, i \rangle \models \varphi \text{ and } \langle \mathcal{M}, i \rangle \models \psi \\ \langle \mathcal{M}, i \rangle \models \varphi \vee \psi & \text{ iff } \langle \mathcal{M}, i \rangle \models \varphi \text{ or } \langle \mathcal{M}, i \rangle \models \psi \\ \langle \mathcal{M}, i \rangle \models \varphi \Rightarrow \psi & \text{ iff if } \langle \mathcal{M}, i \rangle \models \varphi \text{ then } \langle \mathcal{M}, i \rangle \models \psi \\ \langle \mathcal{M}, i \rangle \models \neg \varphi & \text{ iff } \langle \mathcal{M}, i \rangle \not\models \varphi \end{aligned}$$

The semantics of the temporal operators is defined as follows:

$$\begin{aligned}
\langle \mathcal{M}, i \rangle &\models \bigcirc \varphi \text{ iff } \langle \mathcal{M}, i+1 \rangle \models \varphi \\
\langle \mathcal{M}, i \rangle &\models \Diamond \varphi \text{ iff there exists } j \geq i \text{ s.t. } \langle \mathcal{M}, j \rangle \models \varphi \\
\langle \mathcal{M}, i \rangle &\models \Box \varphi \text{ iff for all } j \geq i \text{ then } \langle \mathcal{M}, j \rangle \models \varphi \\
\langle \mathcal{M}, i \rangle &\models \varphi \mathcal{U} \psi \text{ iff there exists } j \geq i \text{ s.t. } \langle \mathcal{M}, j \rangle \models \psi \text{ and for all } k \text{ where} \\
&\quad i \leq k < j, \langle \mathcal{M}, k \rangle \models \varphi \\
\langle \mathcal{M}, i \rangle &\models \varphi \mathcal{W} \psi \text{ iff } \langle \mathcal{M}, i \rangle \models \varphi \mathcal{U} \psi \text{ or } \langle \mathcal{M}, i \rangle \models \Box \varphi
\end{aligned}$$

Given the semantics, the following abbreviations hold:

- $\Diamond \phi$  for  $\mathbf{true} \mathcal{U} \phi$
- $\Box \phi$  for  $\neg \Diamond \neg \phi$
- $\phi \mathcal{W} \psi$  for  $\Box \phi \vee (\phi \mathcal{U} \psi)$

### 2.2.2 CTL

CTL (Computational Tree Logic) is a branching temporal logic first introduced by Clarke and Emerson [CE82, CES86] in 1982. It is based on propositional logic, similar to LTL but allows for a branching time line instead of the standard discrete and linear time line found in LTL.

#### Definition 8

Let  $\text{PROP}$  be a finite set of propositional symbols such as  $p, q, \dots$ ,  $\{\mathbf{true}, \mathbf{false}, \wedge, \vee, \neg, \Rightarrow\}$  be the set of propositional connectives,  $\{\mathbf{A}, \mathbf{E}\}$  be the set of temporal path quantifiers and  $\{\bigcirc, \mathcal{U}\}$  be the set of temporal connectives. The set of CTL well-formed-formulae (WFF) is defined as follows:

- $\text{PROP} \subseteq \text{WFF}$
- $\{\mathbf{true}, \mathbf{false}\} \subseteq \text{WFF}$
- If  $\varphi$  and  $\psi$  are in WFF, then so are

$$\begin{array}{ll}
- \neg \varphi & - \varphi \wedge \psi \\
- \varphi \vee \psi & - \varphi \Rightarrow \psi
\end{array}$$

$$\begin{array}{ll}
 - \mathbf{A} \bigcirc \varphi & - \mathbf{A}(\varphi \mathcal{U} \psi) \\
 - \mathbf{E} \bigcirc \varphi & - \mathbf{E}(\varphi \mathcal{U} \psi)
 \end{array}$$

The temporal path quantifiers  $\mathbf{A}$  and  $\mathbf{E}$  are interpreted as “in all future paths” and “in some future path” respectively. Again, the semantics of CTL is usually defined using a Kripke structure  $\mathcal{M} = \langle \mathcal{S}, \mathcal{R}, \pi \rangle$ , where  $\mathcal{S}$  is the set of moments in time,  $\mathcal{R}$  is the temporal accessibility relation over  $\mathcal{S}$  and  $\pi$  is a propositional valuation function where  $\pi : \mathcal{S} \rightarrow \mathbb{P}(\text{PROP})$  which maps each moment (world) to a set of propositions that are true in that moment (world). The semantics of the propositional operators is as for LTL. The semantics of the temporal connectives and path quantifiers are as follows:

$$\begin{aligned}
 \langle \mathcal{M}, s \rangle \models \mathbf{E} \bigcirc \varphi & \text{ iff there exists a } s' \in \mathcal{S} \text{ where } (s, s') \in \mathcal{R} \text{ and } \langle \mathcal{M}, s' \rangle \models \varphi \\
 \langle \mathcal{M}, s \rangle \models \mathbf{A} \bigcirc \varphi & \text{ iff for all } s' \in \mathcal{S} \text{ where } (s, s') \in \mathcal{R}, \langle \mathcal{M}, s' \rangle \models \varphi \\
 \langle \mathcal{M}, s \rangle \models \mathbf{E}(\varphi \mathcal{U} \psi) & \text{ iff there exists a sequence of time points } s_0, s_1, \dots, s_n \text{ in } \mathcal{S} \\
 & \text{ where } (s_i, s_{i+1}) \in \mathcal{R} \text{ and for all } 0 \leq i < n \text{ with } s_0 = s, \langle \mathcal{M}, s_n \rangle \models \psi \text{ and} \\
 & \langle \mathcal{M}, s_i \rangle \models \varphi \\
 \langle \mathcal{M}, s \rangle \models \mathbf{A}(\varphi \mathcal{U} \psi) & \text{ iff for all sequences of time points } s_0, s_1, \dots \text{ in } \mathcal{S} \text{ where} \\
 & (s_i, s_{i+1}) \in \mathcal{R} \text{ for all } 0 \leq i \text{ with } s_0 = s, \text{ there exists an } n > i \text{ s.t. } \langle \mathcal{M}, s_n \rangle \models \psi \\
 & \text{ and } \langle \mathcal{M}, s_i \rangle \models \varphi
 \end{aligned}$$

Note that we can also utilize the  $\Diamond$  and  $\Box$  operators in CTL formulae as the following abbreviations:

- $\mathbf{A}\Diamond\psi$  for  $\mathbf{A}(\text{true}\mathcal{U}\psi)$
- $\mathbf{E}\Diamond\psi$  for  $\mathbf{E}(\text{true}\mathcal{U}\psi)$
- $\mathbf{A}\Box\psi$  for  $\neg\mathbf{E}\Diamond\neg\psi$
- $\mathbf{E}\Box\psi$  for  $\neg\mathbf{A}\Diamond\neg\psi$

### 2.2.3 Allens Interval Relations

Allen [All83] introduced an interval algebra for representing and reasoning with relations between intervals. He describes 13 relations shown in Figure 2.1. To illustrate the use of these relations, consider the following example: *bob was not in the room when alice turned on the light*. Here, we have 3 events taking place

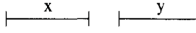
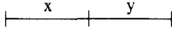
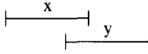
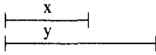
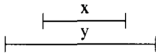
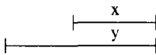

Relation	Symbol	Inverse	Meaning
x before y	b	bi	
x meets y	m	mi	
x overlaps y	o	oi	
x starts y	s	si	
x during y	d	di	
x finishes y	f	fi	
x equal y	eq	eq	

Figure 2.1: Allens's Interval Relations [All83]

- bob not being in the room (i)
- alice being in the room (ii)
- alice turning on the light (iii)

We can represent some of these events using Allen's relations as follows:

- (i)[b,m,mi,bi](ii) : (i) happened before, met, was met by, or came after (ii)
- (iii)[d](ii) : (iii) happens during (ii)

## 2.3 Description Logics Extensions & Design Patterns

Temporalising DLs is a popular and active research topic. DLs, being a subset of FOL are modelled in a static environment, and adding a temporal dimension to make the environment *dynamic* can come with many advantages, especially for ontologies modelling dynamic knowledge.

Several approaches have been proposed to adding a temporal dimension to DLs, and they differ in many ways. Whether it is the notion of time used: time points [LWZ08, GJS15b, GJS15a] or intervals [HS91, Lut01a, Lut04, Sch90], the way in which time is applied: at a concept level [LWZ08, Sch93] or axiom level [WZ98b, LWZ08], what type of time is applied: qualitative time [LWZ08] or quantitative time [LSP14], how it is applied: whether it is an extension to DLs

[LWZ08, Lut02] or embedded inside DLs [WF06], and many more (see Surveys [LWZ08, AF00, AF05] for more details).

Our research contributions focus mainly on 3 temporal representations. The first includes two extensions that include a combination of the temporal logics LTL and CTL with DLs to form  $LTL_{DL}$  and  $CTL_{DL}$ . Both are seen as extensions to DLs since they extend the classic DL syntax and semantics by combining both with additional operators and a temporal dimension from each respective temporal logic. The second is also an extension to DLs by means of adding a concrete domain to the standard DL structure, and then using the concrete domain as a temporal structure. The third is seen as an “*embedded*” temporal representation, as opposed to an explicit temporal extension. It uses the power of the expressivity of OWL to create an ontology called the Fluent Ontology [WF06], used to represent temporal information. We introduce each representation below.

### 2.3.1 $LTL_{\mathcal{ACC}}$

An active and popular suggestion for temporal extensions to DLs involves combining the well known TL LTL with DLs such as  $\mathcal{ACC}$ , where we allow the temporal operators to be applied to concept descriptions (first introduced in [Sch93]), which helps to reason over the evolution of concepts. Given the standard operators of LTL, we can encode  $LTL_{\mathcal{ACC}}$  concept descriptions according to the following definition:

#### Definition 9

*In addition to the standard concept descriptions in  $\mathcal{ACC}$ , the  $LTL_{\mathcal{ACC}}$  concept descriptions  $C$  and  $D$  can be built according to the following syntax rules:*

$$C, D \longrightarrow \bigcirc C \mid CUD$$

The semantics of  $LTL_{\mathcal{ACC}}$  is based on temporal interpretations. A temporal interpretation  $\mathcal{M}$  is a sequence  $\langle \mathcal{J}_0, \mathcal{J}_1, \mathcal{J}_2, \dots \rangle$  where each  $\mathcal{J}_i$  consists of a non-empty domain  $\Delta^{\mathcal{J}_i}$  and a function  $\cdot^{\mathcal{J}_i}$ , that maps concept names to subsets of  $\Delta^{\mathcal{J}_i}$ , roles to subsets of  $\Delta^{\mathcal{J}_i} \times \Delta^{\mathcal{J}_i}$  and named individuals to elements of  $\Delta^{\mathcal{J}_i}$  where  $i$  represents distinct time points. The notation  $x \in A^{\mathcal{J}_i}$  states that in the interpretation  $\mathcal{J}_i$ ,  $x$  is an instance of  $A$  at time point  $i$ . The semantics of  $LTL_{\mathcal{ACC}}$  concept descriptions can be seen in Table 2.2. For ease of representation, we can



Name	Syntax	Semantics
Top	$\top$	$\Delta^{\mathcal{I}_i}$
Bottom	$\perp$	$\emptyset$
Atomic	$A$	$A^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_i}$
Conjunction	$(C \sqcap D)$	$(C^{\mathcal{I}_i} \cap D^{\mathcal{I}_i})$
Disjunction	$(C \sqcup D)$	$(C^{\mathcal{I}_i} \cup D^{\mathcal{I}_i})$
Negation	$\neg C$	$(\Delta \setminus C^{\mathcal{I}_i})$
Value	$(\forall R.C)$	$\{x \in \Delta^{\mathcal{I}_i} \mid \forall x' : (x, x') \in R \rightarrow x' \in C^{\mathcal{I}_i}\}$
Existential	$(\exists R.C)$	$\{x \in \Delta^{\mathcal{I}_i} \mid \exists x' : (x, x') \in R^{\mathcal{I}_i} \wedge x' \in C^{\mathcal{I}_i}\}$
Next	$(\bigcirc C)$	$\{x \in \Delta^{\mathcal{I}_i} \mid x \in C^{\mathcal{I}_{i+1}}\}$
Until	$(CUD)$	$\{x \in \Delta^{\mathcal{I}_i} \mid \exists j \geq i \text{ s.t. } x \in D^{\mathcal{I}_j} \wedge x \in C^{\mathcal{I}_k} \text{ for } i \leq k < j\}$

Table 2.2: Semantics of Temporal Concept Descriptions in  $\text{LTL}_{\mathcal{ALC}}$ 

view the semantics as a sequence of standard  $\mathcal{ALC}$  interpretations isomorphic to the temporal structure in LTL.

There are two kinds of restrictions that are often discussed in  $\text{LTL}_{DL}$ , rigid roles and various domain constraints. We introduce both below.

**Rigid roles** A *rigid role* is a role whose interpretation remains rigid throughout the time line. Rigidity on roles is defined as a characteristic on roles. Formally, a role  $R$  has a rigid interpretation in  $\mathcal{I}$  if

$$(e, f) \in R^{\mathcal{I}_i} \Leftrightarrow (e, f) \in R^{\mathcal{I}_j} \forall i, j \in \mathbb{N}, e, f \in \Delta^{\mathcal{I}}$$

**Domain constraints** Another constraint is the ability to vary the domain of each possible world. There are four types of domain constraints available, *expanding*, *decreasing*, *constant* and *varying*. The *expanding* domain constraint enforces that domains (sequentially) can either expand or stay the same, but not shrink:  $\Delta^{\mathcal{I}_0} \subseteq \Delta^{\mathcal{I}_1} \subseteq \Delta^{\mathcal{I}_2} \subseteq \dots$ . *Decreasing* domains are the opposite:  $\Delta^{\mathcal{I}_0} \supseteq \Delta^{\mathcal{I}_1} \supseteq \Delta^{\mathcal{I}_2} \supseteq \dots$ . *Constant* domains state that each domain must be the same  $\Delta^{\mathcal{I}_0} = \Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}_2} = \dots$ . *Varying* domains have no constraints on the domains, i.e., they can either *expand*, *decrease* or remain *constant*.

### 2.3.2 Other $\text{LTL}_{DL}$ combinations

Given that  $\mathcal{EL}$  is a restriction of  $\mathcal{ALC}$ , we can define  $\text{LTL}_{\mathcal{EL}}$  as a strict sub language of  $\text{LTL}_{\mathcal{ALC}}$ , where we restrict the DL dimension to only allow the DL expressivity

up to  $\mathcal{EL}$ . The semantics is defined in the same way as for  $\text{LTL}_{\mathcal{ACC}}$ , taking into account the DL restriction.

It is also possible to further restrict  $\text{LTL}_{\mathcal{ACC}}$  in the temporal dimension to form expressively weaker variants. If  $\mathcal{X}$  was a set of the temporal constructs from standard LTL, then the logic  $\text{LTL}_{\mathcal{ACC}}^{\mathcal{X}}$  would denote  $\text{LTL}_{\mathcal{ACC}}$  with the restriction of only allowing the  $\mathcal{X}$  operators to be used in concept descriptions. For example, the syntax of the logic  $\text{LTL}_{\mathcal{ACC}}^{\circ}$  would be restricted to:

$$C, D \longrightarrow \bigcirc C$$

### Past Operators

LTL only gives the option to represent information in the present and future. This is made apparent both by the temporal operators and the time line used ( $\mathbb{N}$ ), although it is possible to still model past information. It became necessary in certain cases to combine statements about past and present in a *natural* way in LTL. LTL when extended with past operators  $\{\bigcirc^-, \Diamond^-, \Box^-, \mathcal{S}\}$  which are interpreted as *the previous time point*, *some time in the past*, *all previous time points* and *since* were shown to be useful in practice [LPZ85]. LTL with past operators does not increase the expressive power nor the complexity of LTL [GPSS80], however it is exponentially more succinct [Mar03, LPZ85, LMS02].

With regard to  $\text{LTL}_{DL}$  combinations and past operators,  $\text{LTL}_{\mathcal{ACC}}$  does not offer any past operators. Artale et al. [AFWZ02] introduce a TDL named  $\mathcal{DLR}_{US}$  that allows for both past and present temporal operators to be applied not only to DL concepts, but roles (of arity  $n$ , instead of binary) and axioms too. Concept descriptions in  $\mathcal{DLR}_{US}$  are built according to the following rules:

$$\begin{aligned} C, D \longrightarrow \top \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists^{\leq}[U_j]\mathbf{R} \mid \Diamond C \mid \\ \Diamond^- C \mid \Box C \mid \Box^- C \mid \bigcirc C \mid \bigcirc^- C \mid CUD \mid CSD \end{aligned}$$

where  $\mathbf{R}$  is a *complex* role built according to the following rules:

$$\begin{aligned} \mathbf{R} \longrightarrow \top_R \mid R \mid \neg \mathbf{R} \mid R_1 \sqcap R_2 \mid R_1 \sqcup R_2 \mid U_i/n : C \mid \\ \Diamond R \mid \Diamond^- R \mid \Box R \mid \Box^- R \mid \bigcirc R \mid \bigcirc^- R \mid R_1 \mathcal{U} R_2 \mid R_1 \mathcal{S} R_2 \end{aligned}$$

Terminological axioms are defined in the usual way, however roles can also be included in axioms, for example, in the following form  $R_1 \sqsubseteq R_2$  where each must

have the same arity. And, if  $\phi$  and  $\psi$  are axioms, then so are  $\neg\phi$ ,  $\phi \wedge \psi$ ,  $\phi\mathcal{U}\psi$  and  $\phi\mathcal{S}\psi$ . TBoxes, ABoxes and ontologies are then defined in the obvious way. The semantics is given in terms of a temporal interpretation  $\mathcal{I}$  which can be seen as a sequence of usual DL interpretations where the time line is isomorphic to  $\mathbb{Z}$ . Readers are directed towards [AFWZ02] for a full definition of the syntax and semantics.  $\mathcal{DLR}_{\mathcal{US}}$  is more expressive than  $\text{LTL}_{\mathcal{AC}}$ , and more succinct: not only do we have access to past operators, but we also have the capabilities to temporalise roles too. We can extend  $\text{LTL}_{\mathcal{AC}}$  to also use past operators to form  $\text{LTL}_{\mathcal{AC}}^-$ , in a similar way to  $\mathcal{DLR}_{\mathcal{US}}$ . We can also choose to keep the standard LTL time line (isomorphic to  $\mathbb{N}$ ) or even choose to adopt the time line isomorphic to  $\mathbb{Z}$ , similar to that of  $\mathcal{DLR}_{\mathcal{US}}$ .

### 2.3.3 $\text{CTL}_{\mathcal{AC}}$

$\text{CTL}_{\mathcal{AC}}$  is a result of the combination of  $\mathcal{AC}$  and CTL, where again, we allow the combination of  $\mathcal{AC}$  concepts with the temporal path operators and connectives from CTL.  $\text{CTL}_{\mathcal{AC}}$  concept descriptions are defined as follows.

#### Definition 10

*In addition to the standard concept descriptions in  $\mathcal{AC}$ , the  $\text{CTL}_{\mathcal{AC}}$  concept descriptions  $C$  and  $D$  can be built according to the following syntax rules:*

$$C, D \longrightarrow \mathbf{E} \bigcirc C \mid \mathbf{A} \bigcirc C \mid \mathbf{E}(CUD) \mid \mathbf{A}(CUD)$$

The semantics of  $\text{CTL}_{\mathcal{AC}}$  concept descriptions is given in terms of a temporal interpretation  $\mathcal{J} = (\mathcal{S}, \mathcal{R}, \mathcal{I})$ , where  $\mathcal{S}$  is a set of moments in time,  $\mathcal{R}$  is the temporal accessibility relation over  $\mathcal{S}$ , and  $\mathcal{I}$  is a function that maps each  $s \in \mathcal{S}$  to a standard  $\mathcal{AC}$  interpretation  $\mathcal{I}(s) = (\Delta^{\mathcal{I}(s)}, \cdot^{\mathcal{I}(s)})$ .  $\mathcal{J}$  maps each concept name  $A$  to  $A^{\mathcal{J}} = \{(s, e) \mid s \in \mathcal{S} \text{ and } e \in A^{\mathcal{I}(s)}\}$ , each role name  $R$  to  $R^{\mathcal{J}} = \{(s, e, f) \mid s \in \mathcal{S} \text{ and } (e, f) \in R^{\mathcal{I}(s)}\}$  and every individual name  $e$  to an element  $e^{\mathcal{J}}$ . As before, we assume a rigid interpretation of the individuals: for every individual  $a$ ,  $a^{\mathcal{I}(s)} = a^{\mathcal{I}(s')}$  for any  $s, s' \in \mathcal{S}$ .  $\mathcal{J}$  is inductively extended to concept descriptions shown in Table 2.3.

Name	Semantics
Top	$\top^{\mathcal{J}} := \bigcup_{s \in \mathcal{S}} \Delta^{\mathcal{I}(s)}$ where $\mathcal{I}(s) = (\Delta^{\mathcal{I}(s)}, \cdot^{\mathcal{I}(s)})$
Bottom	$\perp^{\mathcal{J}} := \emptyset$
Conjunction	$(C \sqcap D)^{\mathcal{J}} := (C^{\mathcal{J}} \cap D^{\mathcal{J}})$
Disjunction	$(C \sqcup D)^{\mathcal{J}} := (C^{\mathcal{J}} \cup D^{\mathcal{J}})$
Negation	$\neg C^{\mathcal{J}} := (\top^{\mathcal{J}} \setminus C^{\mathcal{J}})$
Value	$(\forall R.C)^{\mathcal{J}} := \{(s, d) \mid \forall d_1 \text{ where } (s, d, d_1) \in R^{\mathcal{J}} \rightarrow (s, d_1) \in C^{\mathcal{J}}\}$
Existential	$(\exists R.C)^{\mathcal{J}} := \{(s, d) \mid \exists d_1 \text{ where } (s, d, d_1) \in R^{\mathcal{J}} \wedge (s, d_1) \in C^{\mathcal{J}}\}$
Some Next	$(\mathbf{E} \bigcirc C)^{\mathcal{J}} := \{(s, d) \mid \exists s_1 \in \mathcal{S} \text{ where } (s, s_1) \in \mathcal{R} \text{ and } (s_1, d) \in C^{\mathcal{J}}\}$
All Next	$(\mathbf{A} \bigcirc C)^{\mathcal{J}} := \{(s, d) \mid \forall s_1 \in \mathcal{S} \text{ where } (s, s_1) \in \mathcal{R} \rightarrow (s_1, d) \in C^{\mathcal{J}}\}$
Some Until	$\mathbf{E}(CUD)^{\mathcal{J}} := \{(s, d) \mid \text{there exists } s_0, s_1, \dots, s_n \text{ in } \mathcal{S} \text{ where } (s_i, s_{i+1}) \in \mathcal{R} \text{ for all } 0 \leq i < n \text{ with } s_0 = s \text{ s.t. } (s_n, d) \in D^{\mathcal{J}} \text{ and } (s_i, d) \in C^{\mathcal{J}}\}$
All Until	$\mathbf{A}(CUD)^{\mathcal{J}} := \{(s, d) \mid \text{for all } s_0, s_1, \dots, s_n \text{ in } \mathcal{S} \text{ where } (s_i, s_{i+1}) \in \mathcal{R} \text{ for all } 0 \leq i < n \text{ with } s_0 = s \text{ s.t. } (s_n, d) \in D^{\mathcal{J}} \rightarrow (s_i, d) \in C^{\mathcal{J}}\}$

Table 2.3: Semantics of Temporal Concept Descriptions in  $\text{CTL}_{\mathcal{ALC}}$ 

### 2.3.4 Other $\text{CTL}_{DL}$ combinations

As before, we can vary the DL component of  $\text{CTL}_{DL}$ , for example to form the logic  $\text{CTL}_{\mathcal{EL}}$ .

As before, it is also possible to further restrict  $\text{CTL}_{\mathcal{ALC}}$  in the temporal dimension to form expressively weaker variants. Since we now have two kinds of temporal operators, temporal path operators and standard temporal operators we can make even more combinations. For example, the syntax of the logic  $\text{CTL}_{\mathcal{ALC}}^{\mathbf{E}\bigcirc}$  would be restricted to:

$$C, D \longrightarrow \mathbf{E} \bigcirc C$$

### 2.3.5 The Fluent Ontology

Welty and Fikes [WF06] introduce a reusable ontology for representing and dealing with fluents in OWL. They describe fluents as *relations that hold within certain time intervals and not in others*. Their motivation to creating the ontology was due to the general problem for dealing with relations that require time in standard OWL. Their ontology adopts the perdurantist<sup>1</sup> *four dimensional* view. In its most basic form, the perdurantist view says that everything is a perdurant, i.e., everything has a temporal part. For example the sentence “*bob sat on the chair*” should be interpreted logically as “*the temporal part of bob sat on the*

<sup>1</sup>Perdurants being events, as opposed to their counter part, endurants, being physical objects.

*temporal part of the chair*”.

The fluent ontology is described as follows (presented in Manchester syntax):

```
Ontology: FluentOntology
  Class: TimeInterval
    DisjointWith: TemporalPart
  Class: TemporalPart
    DisjointWith: TimeInterval
  ObjectProperty: temporalPartOf
    Inverse: hasTemporalPart
    Domain: TemporalPart
    Range: not TimeInterval
    Characteristics: Functional
  ObjectProperty: hasTemporalPart
    Inverse: temporalPartOf
    Domain: not TimeInterval
    Range: TemporalPart
  ObjectProperty: fluentProperty
    Domain: TemporalPart
    Range: TemporalPart
  ObjectProperty: temporalExtent
    Domain: TemporalPart
    Range: TimeInterval
    Characteristics: Functional
```

According to Welty and Fikes [WF06], the usage of the ontology should be as follows. Every instance of a standard OWL class should have at least one `hasTemporalPart` relation to an instance of `TemporalPart`, which should be associated with exactly one instance of `TimeInterval` via `temporalExtent`. Using the example above, consider its standard OWL Ontology:

```
Ontology: PersonChairOntology
  Class: Person
    DisjointWith: Chair
  Class: Chair
    DisjointWith: Person
  ObjectProperty: satOn
    Domain: Person
    Range: Chair
```

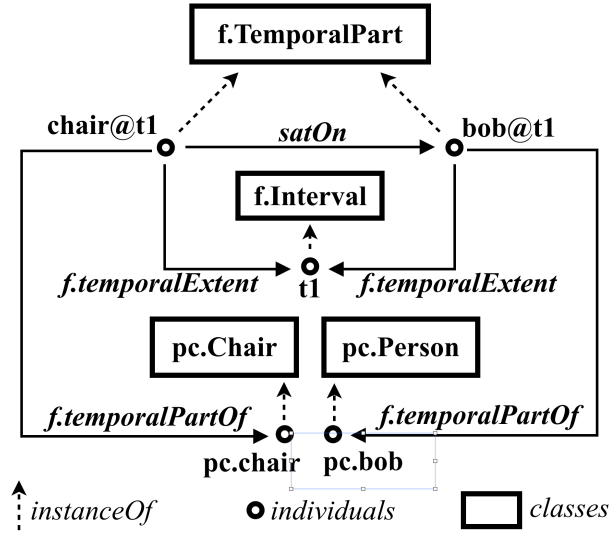


Figure 2.2: An example usage of the Fluent Ontology

```

Individual: bob
  Types: Person
  Facts: satOn chair
Individual: chair
  Types: Chair

```

Importing the `FluentOntology`, the corresponding temporal fluent ontology is constrained as follows:

```

Ontology: PersonChairFluentOntology
  Import: FluentOntology f
  Import PersonChairOntology pc
  ObjectProperty: satOn
    SubPropertyOf: f.fluentProperty
    Domain: f.temporalPartOf only Person
    Range: f.temporalPartOf only Chair

```

allowing us to represent the ontology depicted in Figure 2.2. The fluent approach is mainly focussed on representing time at an ABox level, as opposed to the TBox level. It is also important to note that this temporal approach is not an extension, but merely a means to encode temporal knowledge by defining restrictions on standard OWL concepts. The `Fluent` ontology imports its class of time intervals from the `OWL-Time` ontology [HP04].

### 2.3.6 Concrete Domains $\mathcal{D}$

Concrete domains  $\mathcal{D}$  were added to DLs to allow for the representation of concrete qualities, first formally introduced by Baader et al. in [BH91], who proposed the extension of concrete domains to  $\mathcal{ALC}$  to form  $\mathcal{ALC}(\mathcal{D})$  where the  $\mathcal{D}$  represents the concrete domain. A concrete domain  $\mathcal{D}$  is usually represented as a pair  $(\Delta_{\mathcal{D}}, \Phi_{\mathcal{D}})$ , where  $\Delta_{\mathcal{D}}$  is a set of concrete data values, such as  $\mathbb{N}$ , and  $\Phi_{\mathcal{D}}$  is a set of predicate names, such as  $\{<, >, =, \dots\}$ , where each predicate name  $p \in \Phi_{\mathcal{D}}$  is associated with arity  $n$  and a fixed interpretation  $p^{\mathcal{D}} \subseteq \Delta_{\mathcal{D}}^n$ . To access the concrete domain, we extend the standard DL with: *abstract features* which are usual object properties interpreted as functional relations, *concrete features* that act as a partial function linking elements from the DL domain to elements in the concrete domain and finally, a *new constructor* allowing us to describe a concept's association with the concrete domain.

We now go on to define the syntax of semantics of  $\mathcal{ALC}(\mathcal{D})$  by means of an extension to  $\mathcal{ALC}$ .

#### Definition 11

Let  $\mathcal{D} = (\Delta_{\mathcal{D}}, \Phi_{\mathcal{D}})$  be a concrete domain,  $f$  and  $e$  be abstract features,  $g$  and  $h$  be concrete features and  $p \in \Phi_{\mathcal{D}}$  be a predicate of arity  $n$ . A concrete feature path  $\mu$  is of the form  $f_1 \cdot \dots \cdot f_m \cdot g$  where  $m \geq 0$ . To form  $\mathcal{ALC}(\mathcal{D})$  concept descriptions,  $\mathcal{ALC}$  concept descriptions are extended by

$$C, D \longrightarrow \dots \mid \exists_p(\mu_1, \dots, \mu_n) \mid \forall_p(\mu_1, \dots, \mu_n)$$

where  $n > 0$ .

The standard  $\mathcal{ALC}$  interpretation function is extended to account for the concrete domain by mapping each abstract feature  $f$  to a partial function  $f^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow \Delta^{\mathcal{I}}$ , each concrete feature  $g$  to a partial function  $g^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow \Delta_{\mathcal{D}}$  and each concrete feature path  $\mu = f_1 \cdot \dots \cdot f_m \cdot g$  to the set  $\mu^{\mathcal{I}} = \{(x_1, y) \in \Delta^{\mathcal{I}} \times \Delta_{\mathcal{D}} \mid \exists x_2, \dots, x_m \in \Delta^{\mathcal{I}} : f_i^{\mathcal{I}}(x_i) = (x_{i+1}) \wedge 0 \leq i < m \wedge g^{\mathcal{I}}(x_m) = y\}$  and we write  $\mu^{\mathcal{I}}(x) = y$  if  $(x, y) \in \mu^{\mathcal{I}}$ . The interpretation function is extended to the additional concepts shown in Table 2.4. A survey of concrete domains in DLs can be found at [Lut02].

C	Semantics
$(\exists(\mu_1, \dots, \mu_n).p)^{\mathcal{I}}$	$\{x \in \Delta^{\mathcal{I}} \mid \exists q_1, \dots, q_n : \mu_i^{\mathcal{I}}(x) = q_i \text{ for } 0 \leq i \leq n\} \wedge (q_1, \dots, q_n) \in p^{\mathcal{I}}$
$(\forall(\mu_1, \dots, \mu_n).p)^{\mathcal{I}}$	$\{x \in \Delta^{\mathcal{I}} \mid \forall q_1, \dots, q_n : \mu_i^{\mathcal{I}}(x) = q_i \text{ for } 0 \leq i \leq n\} \rightarrow (q_1, \dots, q_n) \in p^{\mathcal{I}}$

Table 2.4: Semantics of Concept Descriptions in  $\mathcal{ALC}(\mathcal{D})$ 

## 2.4 Reasoning in Description Logics and their Extensions

Due to their precise syntax and semantics, DLs come with the ability to infer new information without having to explicitly state it. DL *reasoners* are computational systems that compute and infer new information about DL knowledge bases. Many reasoning services exist depending on what type of information is being inferred. We introduce some of these reasoning services below.

**Satisfiability** Satisfiability is usually concerned with concept descriptions and is defined as follows:

**Definition 12** (Satisfiability of a concept  $C$ )

*A concept description  $C$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C^{\mathcal{I}}$  is non empty, i.e.,  $C^{\mathcal{I}} \neq \emptyset$ .*

Satisfiability is also defined for concept descriptions in the presence of ontologies, more specifically a TBox.

**Definition 13** (Satisfiability of a concept  $C$  w.r.t a TBox  $\mathcal{T}$ )

*A concept  $C$  is satisfiable w.r.t a TBox  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C^{\mathcal{I}}$  is non empty.*

**Consistency** Ontology consistency involves determining whether or not a given ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  is consistent, i.e. determining whether there is exists a model of  $\mathcal{O}$ .

**Definition 14** (Consistency of Ontologies)

*An ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  is consistent if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$ .*



**Subsumption** Subsumption involves determining whether a concept description  $C$  is subsumed by another concept description  $D$ , w.r.t some ontology  $\mathcal{O}$ , where both  $C$  and  $D$  are built from the signature of  $\mathcal{O}$ . Formally:

**Definition 15** (Subsumption of Concept Descriptions)

*Let  $\mathcal{O}$  be an Ontology, and  $C$  and  $D$  be concept descriptions where  $C, D \in \tilde{\mathcal{O}}$ .  $C$  subsumes  $D$ , written,  $\mathcal{O} \models C \sqsubseteq D$  if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$  for all models  $\mathcal{I}$  of  $\mathcal{O}$ .*

**Instance Checking** Instance checking determines whether or not an individual is an instance of a concept description, w.r.t some ontology. Formally,

**Definition 16** (Instance Checking)

*Let  $\mathcal{O}$  be an Ontology,  $C$  and  $D$  be concept descriptions where  $C, D \in \tilde{\mathcal{O}}$  and  $a$  be an individual where  $a \in \tilde{\mathcal{O}}$ .  $a$  is an instance of  $C$  if  $a^{\mathcal{I}} \in C^{\mathcal{I}}$  for all models  $\mathcal{I}$  of  $\mathcal{O}$ .*

**Classification** Classification is usually defined as determining a partial order of subsumption relations between concept names occurring in the signature of an ontology.

**Definition 17** (Classification of an Ontology)

*Let  $\mathcal{O}$  be an ontology. A classification of an ontology  $\mathcal{O}$  is the set of all pairs  $\langle A, B \rangle$  s.t.  $A, B$  are concept names in  $\mathcal{O}$  and  $\mathcal{O} \models A \sqsubseteq B$ .*

Many other reasoning services exist, such as realisation, querying, explanation generation, module extraction, axiom pinpointing etc., but we focus only on those relevant to this thesis (specifically those related to TBox reasoning). A DL reasoner is a piece of software that implements solvers for one or many of these reasoning problems for particular DLs. For the first four reasoning tasks above, DL reasoners would usually implement decision procedures for each task, since each problem, when given the correct input has a yes or no answer. Classification on the other hand is not a decision procedure, but a computation problem, since the output of the relation is a partial order of subsumption relations between classes (however, it can normally be reduced to a feasible number of subsumptions, i.e., decision problems). There exist many (sound, complete and terminating) decision problems for many DLs and their reasoning procedures, which we discuss in more detail in the following section for the DLs and their extensions described above.

DL	Problem	Result
$\mathcal{EL}$	Subsumption	PTime [BBL05, Bra04b]
$\mathcal{ALC}$	Concept Satisfiability (*)	PSpace-Complete [SSS91]
$\mathcal{ALC}$	Ontology Consistency	ExpTime-Complete [Sch91]
$\mathcal{SROIQ}$	Ontology Consistency	N2ExpTime-Complete [Kaz08]

Table 2.5: Known complexity results for DLs and their reasoning services. \*= Empty Ontology. In  $\mathcal{ALC}$  satisfiability, subsumption and consistency are polynomial-time inter-reducible to one another [BCM<sup>+</sup>03]. So the results for  $\mathcal{ALC}$  hold for each reasoning problem.

## 2.5 Known Complexity Results

### 2.5.1 DLs

It can be argued that the success of DLs is primarily due to the fact that there exist many *efficient* implementations of decision procedures for many of its reasoning problems, even for DLs extending the expressivity well beyond that of  $\mathcal{ALC}$ . Table 2.5 shows known complexity results for  $\mathcal{EL}$  and  $\mathcal{ALC}$ , along with their reasoning problems, as well as for the DL  $\mathcal{SROIQ}$ .

### 2.5.2 LTL & CTL

The main reasoning problem in LTL is model checking and satisfiability. The satisfiability problem for LTL asks whether a given LTL formula is satisfiable. Model checking is the problem of determining whether given an LTL model and an LTL formula, whether or not the given formula holds for the model. Both of which are known to be PSpace complete [SC85]. The same problems also apply for CTL, where the complexity of model checking is PTime-complete [CES86] whilst satisfiability is ExpTime-complete [EH85]. LTL with past operators is also known to be PSpace-complete [LMS02] although exponentially more succinct than LTL without past operators.

### 2.5.3 $\text{LTL}_{DL}$ & $\text{CTL}_{DL}$

Various complexity results for  $\text{LTL}_{DL}$  and  $\text{CTL}_{DL}$  combinations have been proved in recent years. When combining temporal logics with classical DLs such as  $\mathcal{ALC}$ , they are usually computationally well behaved [WZ98a], *usually* decidable and not exceeding the complexity bounds of either of the original logics in restricted

DL	Problem	Result
$\text{LTL}_{\mathcal{ALC}}$ ED	Satisfiability	PSpace-complete
$\text{LTL}_{\mathcal{ALC}}$ ED	Satisfiability, TBoxes	ExpTime-complete
$\text{LTL}_{\mathcal{ALC}}$ CD	Satisfiability, TBoxes	ExpTime-complete
$\text{LTL}_{\mathcal{ALC}}$ CD, RR	Satisfiability, TBoxes	Undecidable
$\text{LTL}_{\mathcal{EL}}$ CD, RR	Satisfiability, TBoxes	Undecidable
$\text{LTL}_{\mathcal{ALC}}$ CD, 1-RR	Satisfiability, Acyclic TBoxes	Decidable (non-elementary)
$\text{CTL}_{\mathcal{ALC}}$ CD, Satisfiability, TBoxes	ExpTime-complete	
$\text{CTL}_{\mathcal{ALC}}$ CD, RR	Satisfiability, TBoxes	Undecidable

Table 2.6: Known Complexity Results for  $\text{LTL}_{DL}$  and  $\text{CTL}_{DL}$  combinations and fragments. ED = Expanding domains, CD = constant domains, RR = rigid roles, 1-RR = 1 rigid role

circumstances (the restrictions usually being on the terminological aspects, such as empty or acyclic terminologies). But when considering what are thought of as being desired or even required features, such as rigid roles or general terminologies, they often lead to undecidability. Table 2.6 shows some of the known results for  $\text{LTL}_{DL}$  and  $\text{CTL}_{DL}$  combinations (for more results, including various fragments of  $\text{CTL}_{DL}$  combinations studied in recent years, we refer to [LWZ08, GJS15b, GJS15b, GJL12]).

#### 2.5.4 $\mathcal{ALC}(\mathcal{D})$

Extending DLs with concrete domains usually preserves their decidability as long as the concrete domain itself satisfies the property of being *admissible* and certain restrictions hold on the DL's terminology.

**Definition 18** (Admissible Concrete Domains)

Let  $\mathcal{D}$  be a concrete domain and  $V$  a set of variables. A  $\mathcal{D}$ -conjunction is a predicate of the form

$$c = \bigwedge_{i < k} (x_0^{(i)}, \dots, x_{n_i}^{(i)}) : P_i,$$

where  $P_i$  is an  $n_i$ -ary predicate for  $i < k$  and the  $x_j^{(i)}$  are variables from  $V$ . A  $\mathcal{D}$ -conjunction  $c$  is *satisfiable* iff there exists a function  $\delta$  mapping the variables in  $c$  to elements of  $\Delta_{\mathcal{D}}$  such that  $(\delta(x_0^{(i)}), \dots, \delta(x_{n_i}^{(i)})) \in P_i^{\mathcal{D}}$  for each  $i < k$ . Such a function is called a *solution* for  $c$ . We say that the concrete domain  $\mathcal{D}$  is *admissible* iff

1. *its set of predicate names is closed under negation and contains a name  $\top_{\mathcal{D}}$  for  $\Delta_{\mathcal{D}}$*
2. *the satisfiability of  $\mathcal{D}$ -conjunctions is decidable*

In general, the reasoning problems associated with DLs extended with concrete domains are the same as with standard DLs; satisfiability, subsumption etc. Regarding satisfiability and subsumption, given a concrete domain  $\mathcal{D}$ , in  $\mathcal{ALC}(\mathcal{D})$ , it usually holds that if  $\mathcal{D}$  is admissible, then both concept satisfiability and subsumption in  $\mathcal{ALC}(\mathcal{D})$  remains decidable [BH91] (usually with empty or non general TBoxes). When considering extensions of  $\mathcal{ALC}(\mathcal{D})$ , including more modern areas of descriptions logics such as TBoxes and more specifically general TBoxes, decidability can often easily be lost. Even with very simple concrete domains, including general TBoxes can lead to undecidability. A large amount of work went into ways to regain decidability, by either restricting the concrete domain constructors available or restricting the allowed concrete domains. Examples include allowing path-free concrete constructors where only concrete features chains of size 1 are allowed [HMW01] or allowing only unary domain predicates [HS01].

Another interesting complexity result first proved in [Lut01b] shows that for concrete domains  $\mathcal{D}$  where (i)  $\mathbb{N} \subseteq \Delta_{\mathcal{D}}$  and (ii)  $\Phi_{\mathcal{D}}$  provides a unary predicate for equality with 0, a binary predicate for general equality and a binary predicate for incrementation,  $\mathcal{ALC}(\mathcal{D})$  concept satisfiability and subsumption w.r.t general TBoxes are undecidable.

## 2.6 Biological Foundations in Ontologies

Ontologies are widely used for defining and controlling vocabularies in the bio-health domain. Many bio-health ontologies exist for different purposes, ranging from ontologies describing the development of biological entities, classification of diseases, anatomy and development, life cycle stages and many more. We aim to show an example of exactly how OWL and many of the extensions discussed above can be used in practice, specifically with respect to biology and more specifically bio-health ontologies. We take an active ontology in the area, and illustrate how the knowledge can be captured in OWL, and how temporal information can be captured.

The *Drosophila* Gross Anatomy Ontology [CRGOS13] describes the anatomy and developmental stages of the life cycle of the *Drosophila Melanogaster*<sup>2</sup> (the common fruit fly). We present a small fragment of the ontology describing the development of the spermatid cell, a part of the male germline cell of the fly itself. The fragment shows temporal patterns through two of its most used properties; *developsFrom* and *partOf* and can be broken down between 4 stages shown in the following axioms:

The resulting ontology can be represented in OWL as follows (DL Syntax):

$$\begin{aligned}
\textit{AgglomerationSpermatid} &\sqsubseteq \exists \textit{developsFrom}.\textit{CoalescenceSpermatid} \\
\textit{ClewSpermatid} &\sqsubseteq \exists \textit{developsFrom}.\textit{AgglomerationSpermatid} \\
\textit{OnionSpermatid} &\sqsubseteq \exists \textit{developsFrom}.\textit{ClewSpermatid} \\
\textit{LeafbladeSpermatid} &\sqsubseteq \exists \textit{developsFrom}.\textit{OnionSpermatid} \\
\textit{LeafbladeSpermatid} &\sqsubseteq \textit{Spermatid} \\
\textit{OnionSpermatid} &\sqsubseteq \textit{Spermatid} \\
\textit{ClewSpermatid} &\sqsubseteq \textit{Spermatid} \\
\textit{AgglomerationSpermatid} &\sqsubseteq \textit{Spermatid} \\
\textit{CoalescenceSpermatid} &\sqsubseteq \textit{Spermatid} \\
\textit{Spermatid} &\sqsubseteq \exists \textit{partOf}.\textit{SpermatocyteCyst}
\end{aligned}$$

The development pattern is shown in Figure 2.3. In this example, a single Spermatid cell goes through 5 stages of development, whilst being part of a Spermatophyte Cyst. It is clear here that temporal information is present. For now we will assume that this is the entire developmental pattern and nothing occurs before or after the first and last stage. The first point to make is that we know when the developmental, or the first stage begins - with the Coalescence Spermatid, and from here it develops into a Leafblade Spermatid, spanning over 5 stages, where each stage begins immediately after the last stage ends. We can also assume that the identity of the cell remains the same - each Spermatid is the same Spermatid after each development, so any properties or mutations that hold during the original cell's stage still show up in their later developments (unless stated otherwise). Since each of the developing cells are all types of Spermatid, and the Spermatocyte Cyst has this part during its entire lifetime,

---

<sup>2</sup>[https://en.wikipedia.org/wiki/Drosophila\\_melanogaster](https://en.wikipedia.org/wiki/Drosophila_melanogaster)

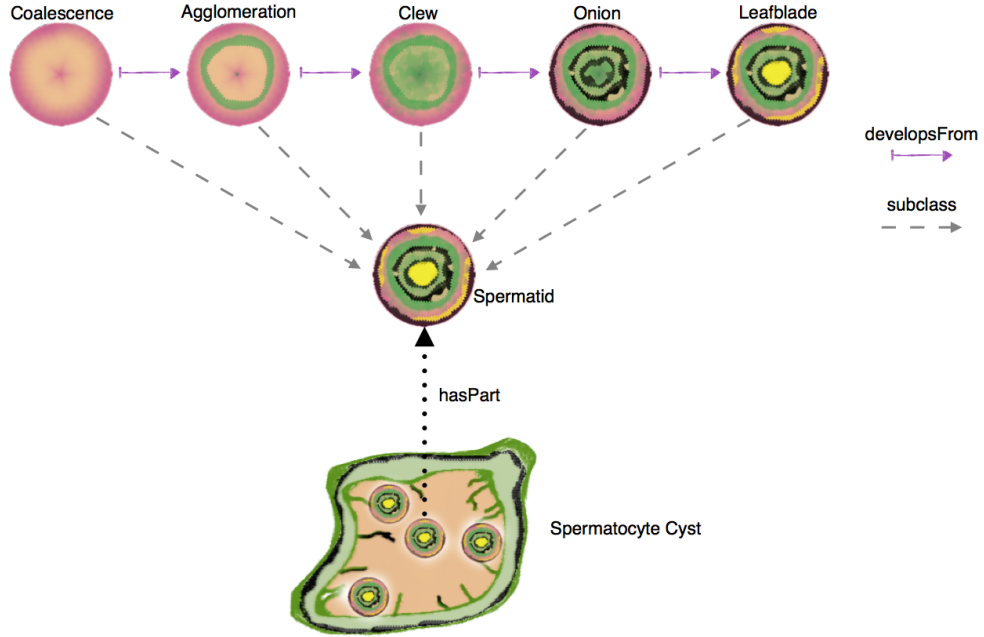


Figure 2.3: A Spermatid cell transitioning through 5 stages of development, contained entirely within a Spermatocyte Cyst, making use of the relations *hasPart* and *developsFrom*

then the Spermatocyte Cyst should have the Spermatid as a part during these 5 stages, and if they are represented by the same element, it should have the *same* element as a part for the 5 stages i.e. we want the role *hasPart* to have some kind of *rigidity* for the 5 stages. Note that it may be the case that each of the 5 Spermatids would be preferably defined as 5 distinct elements - cells with different morphologies and functions. However, in this example, and from the information present in the ontology, this is not so clear cut. Change is continuously happening in developing structures and it is difficult to categorise the exact developmental process so both scenarios may be valid. For the purpose of this example we assume each Spermatid has the same identity.

Since OWL is inherently a *static* logic, there is not much more we can say from a temporal view point to enforce these *temporal properties* that we have identified. We could of course use data properties ( $\mathcal{D}$ ) to time stamp certain concept descriptions, use nominals or transitivity to try and enforce identity constraints ( $\mathcal{R}$  and  $\mathcal{O}$ ), introduce roles to try and reify certain temporal constraints, but being in a static environment severely limits our expressivity. We have only a single world of evaluation. In this example, we are interested in modelling the change of elements or concepts over time, and this is not something OWL can

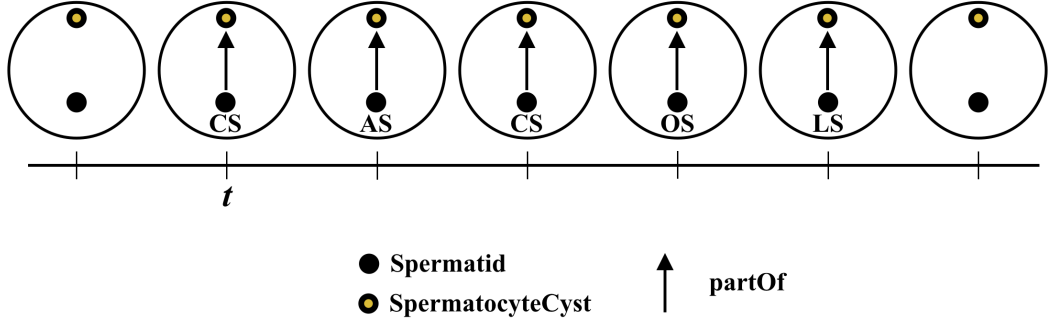


Figure 2.4: An example model view of the Spermatid cell development extracted from the *Drosophila* Ontology modelled in  $LTL_{\mathcal{ALC}}$ . Each Spermatid is represented by the same individual, similarly for the SpermatocyteCyst. *CS* = *CoalescenceSpermatid*, *AS* = *AgglomerationSpermatid*, *CS* = *ClewSpermatid*, *OS* = *OnionSpermatid*, *LS* = *LeafbladeSpermatid*.

easily offer.

This simple extraction shows that we are very limited in what we can and cannot represent in current OWL, and to a certain extent we are not faithfully representing the knowledge which *may have been* intended (more on this later).

**Modelling *Drosophila* with  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and the Fluent Ontology** We now consider briefly how  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and the Fluent Ontology can be used to model the *Drosophila* extraction above. We begin with  $LTL_{\mathcal{ALC}}$ . Consider the following axioms (a model of these axioms is shown in Figure 2.4):

$$\begin{aligned}
 CoalescenceSpermatid &\sqsubseteq \bigcirc AgglomerationSpermatid \sqcap Spermatid \\
 AgglomerationSpermatid &\sqsubseteq \bigcirc ClewSpermatid \sqcap Spermatid \\
 ClewSpermatid &\sqsubseteq \bigcirc OnionSpermatid \sqcap Spermatid \\
 OnionSpermatid &\sqsubseteq \bigcirc LeafbladeSpermatid \sqcap Spermatid \\
 Spermatid &\sqsubseteq \exists .partOf.SpermatocyteCyst
 \end{aligned}$$

We abolish the *developsFrom* relation and rely on the possible world semantics and the rigid interpretation of individuals to capture the identity constraints. For example, we state that every *CoalescenceSpermatid* is at the next time an *AgglomerationSpermatid*. If the identity constraint was not necessary, we could reintroduce the *developsFrom* relation and possibly use past operators combined with *developsFrom* (for example  $AgglomerationSpermatid \sqsubseteq \exists developsFrom. \bigcirc^-$

*CoalescenceSpermatid*). For the rigid constraint, we could make the *partOf* relation *rigid*, so we could ensure that the element representing the Spermatid, was always related to the same SpermatocyteCyst element.

We now consider  $\mathcal{ALC}(\mathcal{D})$ . Suppose we wanted to use a concrete domain of intervals, along with the predicates over Allen’s relations [All83]. We reduce the example to only three stages. Consider the following axioms:

$$\begin{aligned}
& \top \sqsubseteq \exists stage \\
& Spermatid \sqsubseteq \exists hasPart_1.CoalescenceSpermatid \\
& Spermatid \sqsubseteq \exists hasPart_2.AglomerationSpermatid \\
& Spermatid \sqsubseteq \exists hasPart_3.ClewSpermatid \\
& Spermatid \sqsubseteq \exists_{meets}(hasPart_1 \circ stage, hasPart_2 \circ stage) \sqcap \\
& \quad \exists_{meets}(hasPart_2 \circ stage, hasPart_3 \circ stage) \\
& Spermatid \sqsubseteq \exists_{during}(hasPart_1 \circ stage, stage) \sqcap \\
& \quad \exists_{during}(hasPart_2 \circ stage, stage) \sqcap \\
& \quad \exists_{during}(hasPart_3 \circ stage, stage) \\
& SpermatocyteCyst \sqsubseteq \exists hasPart.Spermatid \\
& SpermatocyteCyst \sqsubseteq \exists_{during}(hasPart \circ stage, stage)
\end{aligned}$$

where *stage* is a concrete feature and *hasPart* and *hasPart<sub>i</sub>* are abstract features.  $\mathcal{ALC}(\mathcal{D})$  offers a static environment so it is difficult to capture the identity constraints and to effectively model change: note that for example, the *CoalescenceSpermatid* and the *AgglomerationSpermatid* are different elements, so any mutations from the former would not be “*handed over*” to the latter during its development. But it does give us a powerful mechanism to model temporal constraints on entities. An example model of these axioms can be seen in Figure 2.5. The axioms state that every entity has a stage (in the concrete domain). Every Spermatid has 3 parts, and every SpermatocyteCyst has a Spermatid part. We then use feature chains and concrete features to ensure that the temporal constraints on these features all *align* correctly. For example, although we do not state it, we know the entire developmental sequence of the 3 Spermatid parts happen entirely during the stage of the Spermatocyte Cyst.



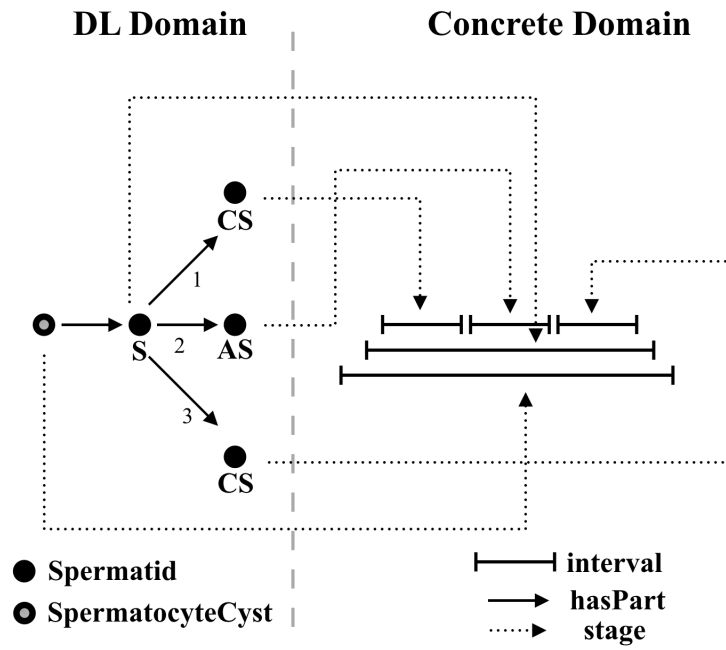


Figure 2.5: An example model view of the Spermatid cell development extracted from the Drosophila Ontology modelled in  $\mathcal{ALC}(\mathcal{D})$ . Only the first 3 stages are used in this example. A set of intervals and Allen’s interval relations make up the concrete domain. *CS* = *CoalsecenceSpermatid*, *AS* = *AgglomerationSpermatid*, *CS* = *ClewSpermatid*.

**The Fluent Ontology** Using the Fluent approach, we can represent the example as follows (again we stick to only 3 stages):

$$\begin{aligned}
CoalescenceSpermatid@t_1 &\sqsubseteq \exists partOf.Spermatid@t_1 \\
AgglomerationSpermatid@t_2 &\sqsubseteq \exists partOf.Spermatid@t_2 \sqcap \\
&\quad \exists developsFrom.CoalescenceSpermatid@t_1 \\
ClewSpermatid@t_3 &\sqsubseteq \exists partOf.Spermatid@t_3 \sqcap \\
&\quad \exists developsFrom.AgglomerationSpermatid@t_2 \\
Spermatid@t_4 &\sqsubseteq \exists partOf.SpermatocyteCyst@t_4 \\
CoalescenceSpermatid@t_1 &\sqsubseteq \exists hasTemporalPart.t_1 \sqcap \\
&\quad \exists temporalPartOf.CoalescenceSpermatid \\
AgglomerationSpermatid@t_1 &\sqsubseteq \exists hasTemporalPart.t_2 \sqcap \\
&\quad \exists temporalPartOf.AgglomerationSpermatid \\
ClewSpermatid@t_1 &\sqsubseteq \exists hasTemporalPart.t_3 \sqcap \\
&\quad \exists temporalPartOf.ClewSpermatid \\
Spermatid@t_1 &\sqsubseteq \exists hasTemporalPart.t_4 \sqcap \\
&\quad \exists temporalPartOf.Spermatid \\
SpermatocyteCyst@t_1 &\sqsubseteq \exists hasTemporalPart.t_4 \sqcap \\
&\quad \exists temporalPartOf.SpermatocyteCyst
\end{aligned}$$

Although the Fluent ontology focusses mainly on representing temporal aspects at an ABox level, our example shows how we can lift this to a terminological level by applying the notions of *temporalParts* and *temporalExtents* directly to classes themselves. Although we are again in a static environment, we can *simulate* some of the temporal features by considering that every temporal part of an entity is a specific part of an entity in time and we can make statements about it only at that time point. We can also go on to make constraints on the time intervals used to attempt to order them in the correct way. Again, we cannot model change: the *CoalescenceSpermatid* that is part of a *Spermatid* is not necessarily the same as the one that an *AgglomerationSpermatid* developsFrom - so, like for  $\mathcal{ALC}(\mathcal{D})$ , mutations are not “handed over”.

Each extension has its unique way of capturing different temporal aspects as can be seen from this example.

**Bio-Health Ontologies** There exist many large corpora of actively maintained ontologies in the bio-health domain, currently attracting and aiding scientific research, such as BioPortal [NSW<sup>+</sup>09] or the OBO Foundry [SAR<sup>+</sup>07], highlighting their importance. Many of the ontologies in these corpora have biological entities in common with each other, sharing vocabularies such as the Gene Ontology [ABB<sup>+</sup>00], an ontology of defined terms representing gene products and their properties that covers three domains including cellular components, molecular functions and biological processes. Focussing on the OBO (Open Biomedical Ontologies) Foundry, successful attempts have been made to formalise many of the biological terms used in these ontologies, to provide consistent and unambiguous definitions. The result was in the form of an ontology, called the Relation Ontology [SCK<sup>+</sup>05]. It provides a relational hierarchy including relations such as *part of* (used in our example extracted from the Drosophila Ontology above), and more biology specific relations such as *develops from* (also used above), each with a rich formal definition, and is now host to over 400 relations<sup>3</sup> being used across the OBO foundry. As well as classifying relations, it also provides a hierarchy for classes used widely amongst the foundry too. Amongst these classes are two popular entities known widely in philosophy as *Continuants* and *Occurrents*. The research in this thesis focuses heavily on these two types of entities so we proceed to give definitions and examples of each of the two entities.

**Continuants & Occurrents** In the literature, continuants are described as those entities that endure or continue to exist through time, whilst undergoing different kinds of changes. Occurrents on the other hand are described as those things that have only temporal parts, unable to undergo change, and unfold in temporal phases. In philosophy, these terms are often referred to as endurants and perdurants respectively. With regards to biology, continuants are those objects such as cells, organs, molecules and any other type of physical object that exists in full at any time. In our example above, the Spermatid cells and Cysts would indeed be continuants. Occurrents are those events, such as development, cell division, life, transportation etc. In the example above, each process of the Spermatid's development is considered to be an occurrent.

The Relation Ontology uses definitions from the Basic Formal Ontology<sup>4</sup> (BFO) for its definitions of continuants and occurrents. The formal definition

---

<sup>3</sup><http://www.obofoundry.org/ontology/ro.html>

<sup>4</sup><http://www.obofoundry.org/ontology/bfo.html>

of a continuant is as follows:

*“An entity that exists in full at any time in which it exists at all, persists through time while maintaining its identity and has no temporal parts”.*

For an occurrent, its definition is:

*“An entity that has temporal parts and that happens, unfolds or develops through time”.*

Both include statements of time and can be considered to be temporal entities.

## 2.7 Summary

We have provided an introduction to the research we will carry out in this thesis and introduced the terms and formalisms that we require to do so. We gave a brief introduction to DLs and to OWL, focussing more on the DLs that we will prominently use in this thesis, namely  $\mathcal{EL}$  and  $\mathcal{ALC}$ . We then introduced several temporal logics, LTL and CTL, and their combinations with DLs,  $LTL_{DL}$  and  $CTL_{DL}$ . We then gave a brief overview of concrete domains in DLs, specifically  $\mathcal{ALC}(\mathcal{D})$ , and we introduced the Fluent ontology for representing Fluents in OWL. We then gave an overview of the reasoning problems for each of these representations, and also complexity results for these problems. With the use of a running example (the Drosophila Development extraction), we showed briefly how temporal information was present in OWL, and how it was currently being modelled and under represented. We then showed very briefly how it could possibly be modelled in a few of the temporal extensions we introduced. Whilst it could be argued that some of the modelling attempts were *better* than the original OWL version, it would be difficult to compare each temporal version since the requirements of that specific ontology are still unclear. From the definitions given in the ontology alone, there is still room for open interpretation and ambiguous understanding. These observations drive the research presented in this thesis showing an obvious need for further investigation.

## Chapter 3

# Determining the Temporal Requirements

*What are the temporal requirements for modelling the temporal features of Bio-Health ontologies in OWL?*

To attempt to answer the first research question, we present a survey of The Relation Ontology and The OBO Foundry to determine a set of Temporal Requirements. Our goal is to first identify a suitable set of ontologies that are likely to exhibit rich and clear temporal information that we can easily extract and organise, and then go on to determine their importance in the ontologies to refine into a set of requirements.

### 3.1 Materials: Corpus & Temporal Features

#### 3.1.1 The OBO Foundry

The OBO (Open Biomedical Ontologies) foundry, first founded in 2007 [SAR<sup>+</sup>07] contains a repository for ontologies in the biomedical domain. The repository began with only 16 ontologies and now contains more than 133 ontologies<sup>1</sup>. The OBO Foundry is home to popular ontologies that range from describing developmental sequences of biological entities, such as the Drosophila Development ontology [CRGOS13], to upper level ontologies that incorporate accurate representations of biological phenomena, all containing different kinds of temporal

---

<sup>1</sup>As of November 2015

information. We found this corpus most suitable as it is widely known and actively maintained.

We then needed to determine how to identify temporal patterns amongst the ontologies. One option would be to simply iterate through each ontology in the corpus, and search for temporal information, going through each axiom, class, relation, annotation etc. in the ontology. However, given the number of ontologies in the OBO foundry, it would not be ideal or appropriate to inspect every ontology. The size of many ontologies preclude manual inspection, for example the GO ontology has over 46,000 classes, and over 500,000 axioms [ABB<sup>+</sup>00], and having to gather data manually from each would take a considerable amount of time. Not only would time be an issue, but it would also be fair to assume that some of the ontology developers would not have had an exact notion of time in mind or a general temporal mindset when creating the ontologies (due to OWL's lack of temporal expressivity), so temporal modelling choices may have been ignored or even overlooked. We would also need to solely rely on the developers including clear definitions of each class, property or axiom explaining how they should be temporally interpreted to avoid any ambiguity, to be exact in how they intended for their entities to be temporally annotated. If this information was present, then it would be theoretically possible to proceed in such a way. We inspected a few of the ontologies to see if this level of detail was present but found nothing near appropriate. To avoid manually inspecting each ontology individually and possibly (in some cases definitely) having to contact the developers of the ontologies to get a better understanding, we decided to search for a shared vocabulary, or an upper level ontology that would be likely to be used by most of the ontologies in the corpus, that would also likely contain good and clear temporal information with clear definitions to avoid any ambiguous interpretations of any temporal patterns used.

### 3.1.2 The Relation Ontology

The Relation Ontology (RO) [SCK<sup>+</sup>05] acts as a means for standardisation across ontologies in the OBO Foundry and wider OBO library. Its main focus is the classification of relations between classes that exist in the biomedical domain, but more importantly, it covers relations used in OBO Foundry ontologies. First introduced in 2007, the ontology was host to only 10 relations, including primitive biological relations such as *part of*, *derives from* and *preceded by*, where each

was equipped with a precise definition to avoid any ambiguity of their correct usage, which fortunately, in some cases also contained temporal information. The current version of RO is now host to 459 relations<sup>2</sup>, where similar levels of detail are used in the definitions for a lot of the relations. As well as modelling relations, it also provides a mildly complex class hierarchy that intends to classify the domains and ranges of the relations, most importantly *continuants* and *occurents* (imported from the BFO ontology). These classes are also important to temporal information, contain accurate and precise definitions and aid in the understanding of the relations that use these classes as their domain and range types to further enforce their own definitions. Since the RO is actively maintained and is used to organise the majority of relations used throughout the OBO Foundry, this was the obvious choice to act for our seed of temporal properties. This also aided in our choice of using the OBO Foundry as our corpus. Since RO was created to standardise the relations that ontologies use throughout the OBO Foundry, it makes complete sense to use both together since they were created for this exact purpose and they go hand in hand.

### 3.1.3 Temporal Features

Our next task is to categorise what we call *temporal features* that occur in RO. Temporal features are specific types of temporal information that represent clear temporal phenomena found in RO. Since the RO is of manageable size, we gathered the temporal features by iterating through each relation, observing their definitions, ontological constraints and annotations, and recorded all types of temporal information we came across.

As an example, consider the relation *part of*. The following are statements and definitions taken from the relation’s annotations (where the temporal considerations are emboldened):

- “A core relation that holds between a part and its whole. Part-hood requires the part and the whole to have compatible classes: only an **occurrent** can be part of an **occurrent**; only a process can be part of a process; only a **continuant** can be part of a **continuant**; only an independent continuant can be part of an independent continuant; only an immaterial entity can be part

---

<sup>2</sup>As of November 2015

*of an immaterial entity; only a specifically dependent continuant can be part of a specifically dependent continuant; only a generically dependent continuant can be part of a generically dependent continuant.”*

- *“Occurrences are not subject to **change** and so parthood between occurrences holds for **all the times that the part exists**. Many continuants are subject to change, so parthood between continuants **will only hold at certain times**, but this is difficult to specify in OWL...”*

The first type of temporal information we observe is the domain and range constraints of the relation, namely continuants and occurrences. Both continuants and occurrences are two types of distinct temporal entities that differ in nature. The relation can only hold between either continuants or occurrences. So we immediately identify two temporal features, which are two temporal types of domain and range constraints. The second type of temporal information we encounter is that the relation may only hold at certain times and not others. This is particularly interesting for the occurrence case, since the relation itself should hold between two occurrences at the time they exist. The snippet of temporal information we discover is that of *duration*. Since occurrences have limited phases of existence, then relations such as *part of* may only hold during the times they exist. Therefore relations should be able to have durations. The second piece of information is that of an identity consideration - if the relation has a duration, then it can be seen as spanning over multiple time points. And then it should hold between the same elements over these time points (highlighting the notion of diachronic identity in a consecutive manner). This type of consecutive relation is sometimes referred to as type of *rigid* relation. We identify 2 temporal features here. The first is the idea of a rigid relation, where the same individuals should be related consecutively for some duration, the second is that the relations should hold between the individuals at single time points. This is illustrated in Figure 3.1.

We now consider another type of relation, *transformation of*. The following are definitions taken from annotations of the relation in RO:

- *“ $x$  transformation of  $y$  if  $x$  is the immediate transformation of  $y$ , or is linked to  $y$  through a chain of transformation relationships”*



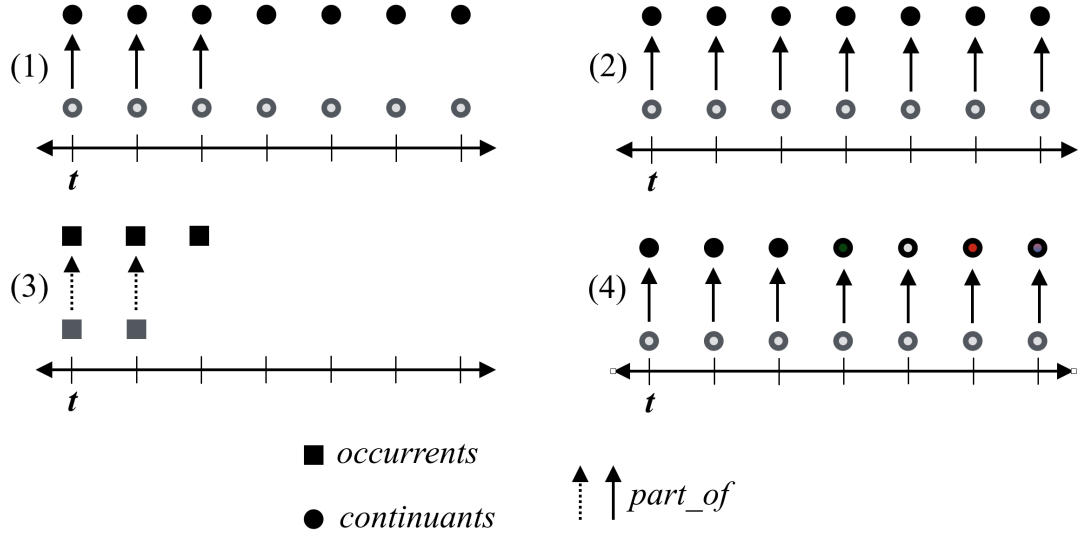


Figure 3.1: An example of rigid relations. (1) shows an example of finite rigidity between continuants, (2) shows an example of infinite rigidity between continuants holding only at 3 consecutive time points, (3) shows an example of finite rigidity between occurrents during the times they exist and (4) shows an example of non rigidity between non identical continuants

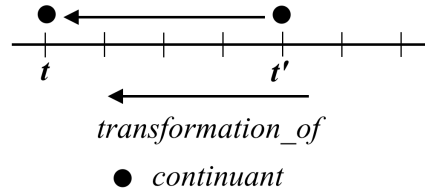


Figure 3.2: Illustration of the relation *transformation\_of* where a single continuant is related to itself at some earlier time point

- “Transformation, on the instance level is the relation of identity: each **adult** is **identical** to some **child** existing at some **earlier time**”

Transformation of is considered to be a temporal relation involving continuants that share the same *identity*, related at different time points. Consider Figure 3.2. The constraints on times  $t$  and  $t'$  are that  $t' > t$ . The temporal information that we observe is that (1) the relation is between continuants, (2) the relation must hold over two different time points (unlike *part of*), (3) the relation must hold between the same elements (again, unlike *part of*), and (4) the individual may be subject to change, i.e., at time  $t'$  the individual may have more or less properties

than it did at time  $t$ , or at a class level, can even be a member of different classes at each time. Identity (3) is an important notion in any type of biological modelling. *transformation of* requires specific identity constraints since the domain of the relation should be identical, or represented as the same individual as the range from a modelling perspective. This type of identity is quite a simple one. Of course, more complex forms exist, for example in cell division. When a cell goes through division, the identity of an original single cell is split into two new cells. So the issue of regarding the divided cells as the same cells as the original, or new independent cells, or partially identical cells etc, is difficult to model, especially in OWL. This can be seen in relations such as *child nucleus of*. We choose to ignore this type of complex identity relation as it is not specific to temporal modelling in itself, and refers more to the problem of modelling identity in ontologies in the general case. However, in the *transformation of* case, the identity constraint is simple enough and important to consider as a temporal feature. The new temporal features we identify are a relation holding over different time points, and a relation holding between identical elements over time.

After iterating through each relation in RO, we acquired 42 distinct temporal features which we grouped into the following 6 categories:

- |                     |           |             |
|---------------------|-----------|-------------|
| 1. Domain and Range | 3. States | 5. Rigidity |
| 2. Identity         | 4. Time   | 6. Possible |

**Domain and Range** contains all possible pairs of domain and range features that occurred in the relations in RO. These consisted of *continuant-continuant* (*c-c*), *occurent-occurent* (*o-o*), *continuant-occurent* (*c-o*), *occurent-continuant* (*o-c*) and *x-x*, where the first element of the pair is the domain and the second element is the range. *x-x* is a synonym for *c-c* OR *o-o*. The relation *part-of* would belong to *x-x* for example, since the relation either relates continuants or occurents, where as the relation *transformation of* would belong to *c-c*.

**Identity** contains information on the identity of each individual in the relation. Consisting of a single feature, *same*, indicating that the domain and range of the relation should be the same individual, such as *transformation of* that maintains its identity over time.

**States** contains the different states that a relation’s corresponding entities may go through during the relation. These include *domain:birth*, *domain:death*, *domain:changed*, *range:birth*, *range:death* and *range:changed*. *domain:birth* specifies that the relation indicates that the domain element came into existence, i.e there was some previous time point where it didn’t exist previously. *domain:death* is the dual of *domain:birth*, i.e, the relation indicates that the domain element goes out of existence. *domain:changed* indicates that the domain element goes through some type of development or change during, such as *transformation of*. All are defined similarly for the *range* case also. Some relations also specify that their elements come in to *and* out of existence. We also included combinations of some features in this set, for example *domain:birth&death* to reflect these.

**Time** contains all types of time relations used in RO. The first are standard time point relations: *same*, *past*, *future*, *past immediate*, *future immediate*, and the second are interval relations described as Allen’s relations : *meets*, *meets'*, *before*, *after*, *during*, *starts*, *ends*. The time point relations are mainly used for *c-c* relations, but also for *c-o* and *o-c* relations. The *same* feature indicates the property relates entities at a single time point, such as the *part of* property. *past* indicates that the relation relates entities at a present time point  $t$  and a previous time point  $t'$  where  $t' < t$ , such as the relation *transformation of*. *past immediate* indicates a *past* style relation with the added constraint that  $t' = t - 1$ , such as the relation *immediate transformation of*, which is a restricted version of *transformation of*. *future* and *future immediate* are the duals of *past* and *past immediate* respectively. Allen’s interval relations [All83] were used to capture relations between occurrents. For example, consider the relation *preceded by* which aims to capture one of Allen’s relations. The following is a definition taken from one of *preceded by*’s annotations:

*x is preceded by y if and only if the time point at which y ends is before or equivalent to the time point at which x starts. Formally: x preceded by y iff  $\phi(y) \leq \psi(x)$ , where  $\phi$  is a function that maps a process to a start point, and  $\psi$  is a function that maps a process to an end point.*

In this case, the corresponding Allen’s interval relation is *p after q*. Finally, for relations between *continuants* and *occurrents* (and *occurrents* and *continuants*), the first set of time relations are preferred. For example, consider the relation *input of*. This relation is intended to model a *continuant* being the input of a

*process* (occurrent). For this relation we use the *same* feature. As in the **States** case, combinations of features were also made to form new features when relations indicated that multiple time relations were present.

**Rigid** consists of only one feature, *rigid*, which as explained in the description of *part of* should indicate that two individuals are related consecutively during a (usually finite) time period.

**Possible** consists of only one feature, *possible* which is intended to indicate that a relation may hold somewhere in the future but is not necessary. For example, consider the relation *capable of regulating*. Its definition is:

*“Holds between  $c$  and  $p$  if and only if  $c$  is capable of some activity  $a$ , and  $a$  regulates  $p$ ”.*

We believe this relation could indicate that there may be some future where  $c$  regulates the process  $p$ , but is not necessary for this relation to hold.

### 3.1.4 TRO - Temporal Relation Ontology

After gathering and then categorising the temporal features as described above, we developed a coding scheme to annotate the relations in RO with the temporal features to form a temporally annotated version, named the Temporal Relation Ontology (TRO). Relations were annotated according to the following definition:

**Definition 19** (TRO Temporal Annotation)

*Let  $P$  be a relation from RO, and  $Y = \{\mathbf{Domain\ and\ Range}, \mathbf{Identity}, \mathbf{States}, \mathbf{Time\ Relations}, \mathbf{Rigid}, \mathbf{Possible}\}$  be the sets of temporal features described above. A temporal annotation for  $P$  is a set  $A \subset \cup Y$  where  $A$  contains*

- *a single domain and range constraint of the form  $[\text{domran:X}]$  where  $X \in \mathbf{Domain\ and\ Range}$*
- *0 or 1 identity constraints of the form  $[\text{identity:X}]$  where  $X \in \mathbf{Identity}$*
- *a single time relation constraint of the form  $[\text{time:X}]$  where  $X \in \mathbf{Time\ Relation}$*
- *0 or 1 rigidity constraints of the form  $[\text{rigid:X}]$  where  $X \in \mathbf{Rigidity}$*
- *0 or 1 possibility constraints of the form  $[\text{possible:X}]$  where  $X \in \mathbf{Possible}$*

We also reserved default TRO annotations for those properties with either no definitions or insufficient data to conclude any meaningful information from the relation. There were also several relations that were intended not to be used, described as *grouping relations* used to better organise the hierarchy, as well as several obsolete terms. We also created default annotations to annotate such relations so we knew to avoid them. We left all these relations out of the study.

The following are examples of temporal annotations for the relations *part of*, *derives from*, *preceded by* and *input of*:

1. **part of**: {[domran:x-x], [time:same], [rigid:rigid]}
2. **derives from**: {[domran:c-c], [domain:birth], [time:past]}
3. **preceded by**: {[domran:o-o], [time:after]}
4. **input of**: {[domran:c-o], [identity:same], [time:same]}

55 distinct temporal annotations cover the 459 object properties, where 18 were between continuants (*c-c*), 13 between occurrents (*o-o*), 11 between occurrents and continuants (*o-c*), 11 between continuants and occurrents (*c-o*) and 2 between either continuants or occurrents (*x-x*).

With the RO temporally annotated as TRO, we now go on to discuss the methodology behind how we discovered which of the temporal features were most important, and how we determined the temporal requirements.

## 3.2 Methodology

Since we have now identified a corpus of ontologies and a set of temporal features that will likely be used in the corpus, we now go on to describe the methodology behind how we will identify the most important features in the corpus, keeping the research questions in mind. Recall that we are interested in determining a set of temporal requirements. We could just simply take the set of features and annotations we have discovered and declare these to be our requirements. This would not be wise since not all of the annotations or features may be used in the corpus, or, certain ones may in fact be more *important* than others. Our aim is to define the notion of *importance* and develop measures of importance to be able to determine which types of relations we should consider based on their *usage*

throughout the corpus, and then go on to define the requirements based on these results.

When considering a feature or annotations usage throughout the corpus, we shift our attention towards the terminological aspects of the ontologies in the corpus. That is, we choose to investigate the explicitly asserted terminological knowledge, specifically TBox axioms. Our notion of usage is defined as follows:

**Definition 20** (Usage of an a TRO feature or TRO Annotation)

Let  $f$  be a TRO feature,  $F$  be a TRO annotation,  $P$  be a TRO relation,  $\mathcal{O}$  be an ontology occurring in the OBO Foundry and let  $\alpha$  be a terminological axiom.

- $F$  **uses**  $f$  if  $f \in F$
- $P$  **uses**  $F$  if  $P$  is annotated with  $F$
- $\alpha$  **uses**  $P$  if  $P$  occurs in  $\alpha$
- $\mathcal{O}$  **uses**  $\alpha$  if  $\alpha$  occurs in the signature of  $\mathcal{O}$

where **uses** is transitive. If  $P$  uses  $F$  or  $f$ , then any equivalent relation or sub relation  $P'$  of  $P$  also uses  $F$  or  $f$ .

We now propose 2 independent measures that will help to determine the importance of an annotation or feature amongst the corpus of ontologies.

**Definition 21** (Coverage and Impact of a TRO feature or annotation)

Let  $e$  be either a TRO feature or TRO annotation and  $\mathcal{C}$  be the OBO Foundry of ontologies.

$$coverage(e) = \frac{\# \text{ontologies that use } e}{\# \text{ ontologies in } \mathcal{C}} \quad (4.1)$$

$$impact(e) = \frac{\sum_{O \in \mathcal{C}} \left( \frac{\# \text{axioms in } O \text{ that use } e}{\# \text{axioms in } O} \right)}{\# \text{ontologies that use } e} \quad (4.3)$$

The coverage measures how many ontologies each annotation is used in at least once. The impact determines on average the percentage of axioms the annotation occurs in per ontology. Notice that both measurements are calculated using explicit axiom knowledge (TBox Axioms). The reason we opt for two individual measures instead of just one is due to the possibility of variance between each

Measure	Corpus 1	Corpus 2
Coverage	90%	100%
Impact	48.75%	22.8%

Table 3.1: Coverage and impact over ontology corpora in Figure 3.3

measure and to strengthen any measure of importance. To illustrate this, consider the examples in Figure 3.3. Both charts show the usage of a single entity ( $e$ ) over 2 corpora, each containing 5 ontologies. In the first chart, each ontology has 100 logical axioms, indicated by the red coloured bar. The blue coloured bar indicates how many of those axioms use  $e$ , which are 40, 5, 50, 0 and 100 respective to ontologies  $\mathcal{O}_1$ ,  $\mathcal{O}_2$ ,  $\mathcal{O}_3$ ,  $\mathcal{O}_4$  and  $\mathcal{O}_5$ . The coverage of  $e$  over the 5 ontologies is  $90\% = \frac{4}{5}$ , since  $e$  is used in 4 out of the 5 ontologies. Although this gives us a sense of usage over the whole corpus, it doesn't tell us much about the importance for each ontology that uses it, highlighting the need for more measures. Its impact measures  $48.75\% = \frac{195\%}{4}$ . This tells us that on average, roughly half of the axioms in each ontology use the feature. Both together indicate that  $e$  is fairly important. Consider the second chart. The first ontology has 1000 logical axioms, whilst the rest only have 50 each. The coverage of  $e$  is 100%, since  $e$  is used in each ontology at least once. This can be misleading since it has a higher coverage than the previous corpus, but it is obvious that its usage within the ontologies varies significantly more. The impact of  $e$  is only  $22.8\% = \frac{114\%}{5}$ , informing us that the average weight of axioms in ontologies that use  $e$  stands at 22.8%. So although it has a very high coverage, the impact was particularly low, prompting further investigation.

We aim to use both measures collectively to better describe the importance of  $e$ . Those  $e$ 's with high coverages and impacts will be most important, whilst those with low coverages and impacts are less important. Anything in between requires further inspection. A summary of these results can be seen in Table 3.1.

### 3.3 Datasets & Implementation

**Data** The corpus of ontologies we used in the survey is a snapshot of OBO Foundry taken in October 2015 from <http://www.obofoundry.org>. The version of the relation ontology was taken from its homepage <https://github.com/>

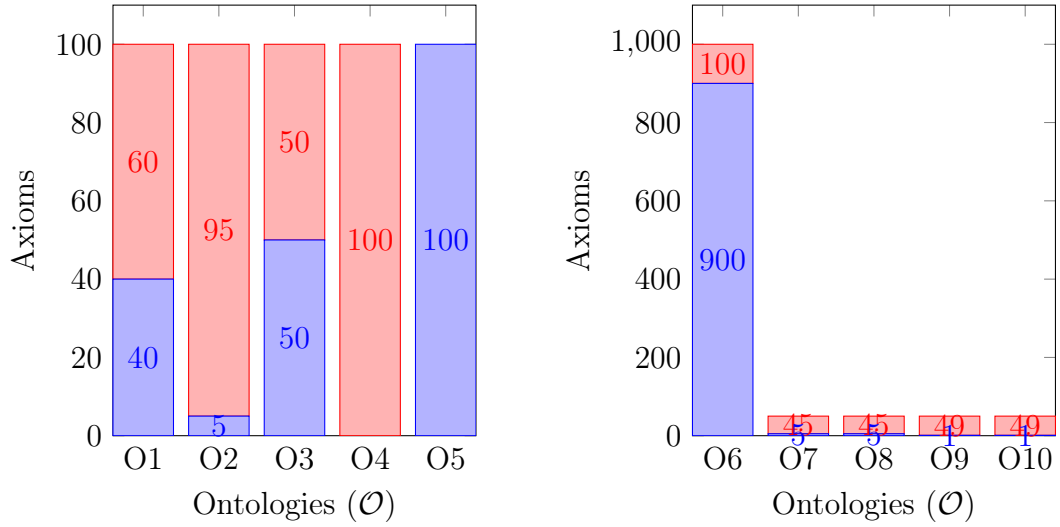


Figure 3.3: Example of coverage and impact measures for calculating importance of an  $e$  of two corpora of 5 ontologies.

`oborel/obo-relations`, which contained 459 object properties. The temporally annotated version (TRO) was created from this version.

**Implementation** We implemented the survey using the Java programming language equipped with the OWL-API libraries [HB11] for parsing and manipulating ontologies. Due to the nature of our implementation, we only accepted ontologies compatible with the OWL-API (version 3.5.2), leaving a total of 116 ontologies. Each ontology was exported into a common syntax, OWL/XML, before the survey was conducted (imports merged).

In the previous section we discussed the notion of a relation being used in an axiom (Definition 20). Ideally, to check for a relation’s usage in an axiom, one should be able to simply search the axioms signature for an occurrence of the relations IRI. However this relies heavily on ontology developers *correctly* using terms from other vocabularies, i.e. importing vocabularies. This is often not the case, since importing ontologies could result in negative side effects such as size increase or a jump in complexity. Instead developers may just create their own entity with a similar name. For this reason, we cannot simply rely on checking for matching IRIs in an axioms signature. For this reason, we adopt a *smart matching* approach, where we say a relation outside RO *matches* an RO relation if either they share the same IRI, name (`rdfs:label`), alternative term (IAO\_0000118), OBO foundry unique label (IAO\_0000589) or the same exact



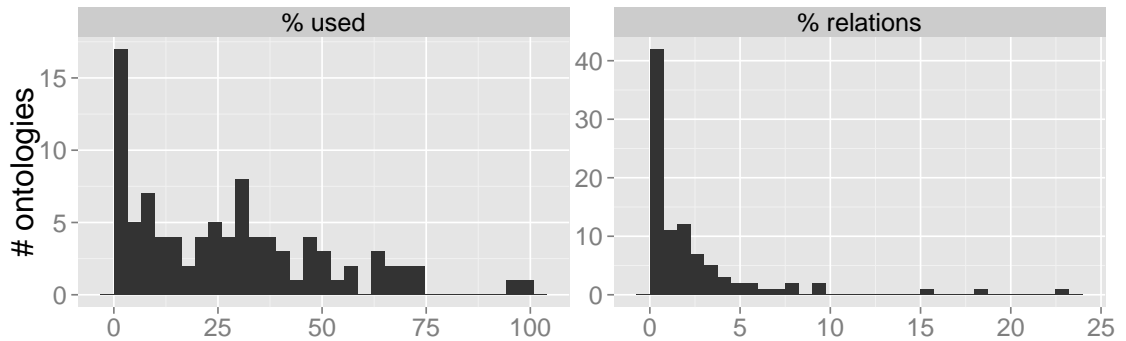


Figure 3.4: Histograms of RO relation usage across OBO Foundry ontologies. Left: Proportion of axioms using RO or RO-like relations compared to the overall number of axioms (in %). Right: Proportion of RO or RO-like relations used in the ontology compared to the total number of RO relations (459).

synonym (hasExactSynonym) to avoid any potential misses.

All data was saved and output as Java Serialised files adopting a local format, and also saved as CSV files, used later for data analysis<sup>3</sup>

## 3.4 Results

### 3.4.1 Ontology and Relation analysis

First we provide a short analysis of the relation usage across OBO Foundry ontologies. Out of the 116 ontologies in the corpus, 92 ontologies used relations from RO (either exact matches or smart-matches according to the method discussed in the previous section). In the following analysis, we only consider the 92 ontologies that made use of RO or RO-like relations. Figure 3.4 shows two histograms illustrating the prevalence and diversity of the relations used.

The left histogram (% used) shows the prevalence of axioms that make use of RO/RO-like relations as the proportion of axioms that contain at least one RO relation in their signature compared to the total number of axioms in the ontology. For example, AERO, with 1,461 axioms in total, has 88 axioms that make use of RO(-like) relations, resulting in a proportion of  $88/1461 = 0.0602$  or 6.02%. As can be seen, there are 2 ontologies that have RO(-like) relations in around 100% of their axioms (EMAP, 21,721 axioms total and WBLS, 1,384

<sup>3</sup>available to view at <http://www.cs.man.ac.uk/~leoj/thesis/>

axioms total). Most of the ontologies in the OBO foundry have RO relation prevalence in the 0%-75% range, gradually declining towards the high-proportion end. There is a large peak around the 0% region. The ontologies responsible for this peak were those with large axiom counts where RO relations were used in a low number of axioms relative to the ontology's large size. Overall the prevalence of RO(-like) relations is high. The right histogram illustrates the diversity of RO relations as the proportion of RO relations that were used in an ontology compared to the total number of RO relations (458). For example, AERO made use of 12 RO relationships (for example `part_of` and `preceded_by`), which lead to a proportion of  $12/458 = 0.026$  (2.6%). As can be seen, the vast majority of ontologies made use of less than 5% of the available RO relations. This indicates a relatively low diversity of used RO relations. At a closer look however, 182 distinct RO relations are used across the whole corpus (see Table 3.2). The high diversity across the corpus and the comparatively low within-ontology diversity can perhaps be easily explained by the purpose of the RO, as it manages all relations in the OBO foundry corpus, regardless of any particular domain.

RO relation	$ \mathcal{O} $
<code>part_of</code>	77
<code>has_part</code>	46
<code>preceded_by</code>	28
<code>inheres_in</code>	22
<code>participates_in</code>	22
<code>has_participant</code>	22
<code>develops_from</code>	21
<code>has_role</code>	20
<code>bearer_of</code>	19
<code>ends_during</code>	19

Table 3.2: Top 10 relations used across the OBO Foundry corpus.  $|\mathcal{O}|$  is the number of ontologies the relation is mentioned in.

### 3.4.2 Feature & annotation analysis

182 of the 459 temporally annotated object properties were used amongst these 92 ontologies. 106 were *c-c* relations, 19 were *o-o*, 19 were *c-o*, 10 were *o-c* and 14 were *x-x*. Out of the 55 distinct annotations that cover the relations, 42 were used in the corpus where 13 were *c-c* annotations, 12 were *o-o*, 9 were *c-o*, 6

Feature	Coverage	Impact
domran:cc	52.59	10.24
time:same	47.41	8.53
rigid	45.69	7.09
time:past	31.03	4.31
dom:changed	19.83	4.68
dom:birth	13.79	2.76
ran:death	13.79	2.76
time:future	11.21	0.55
ran:changed	10.34	0.56
time:past_imm	6.90	0.19
possible	3.45	0.41
identity:same	2.59	0.17

Table 3.3: Coverage and Impact results for *c-c* features used across the OBO Foundry

were *o-c* and 2 were *x-x*. Out of the 42 distinct features, 35 were used. For each data set we computed a Pearson correlation coefficient between the Coverage and Impact measures for feature and annotation results which we use in our analysis.

### 3.4.3 *c-c* results

Relations between continuants were very popular amongst the corpus. Overall, continuant relations had a total coverage of 52.59% and an impact of 10.24%. With regards to the individual features used in *c-c* relations, their coverage ranged from 47.41% to 2.59% with an average of 21.55%, whilst their impact ranged from 8.53% to 0.17%, with an average of 3.52%. There was a high correlation of 0.96. Out of the time relations features used, we can order them since there is a total ordering  $>$  on their coverages and impacts: *time:same*  $>$  *time:past*  $>$  *time:future*  $>$  *time:past\_imm*. Although the last two features are used in a considerable amount of ontologies, their impact in those ontologies is relatively low compared to the first two features. *time:same* had the highest coverage and impact at 47.41% and 8.53% respectively. *time:past* also had a high coverage at 31.03% but a slightly lower impact at just 4.31%. This was more predominant than its inverse *time:future* which had a coverage of 11.21% and an impact of just 0.55%. Domain constraints were more popular than range constraints in both coverage and impact. The ordering is as follows: *dom:changed*  $>$  *dom:birth* = *ran:death*  $>$  *ran:changed* (dom=domain and ran=range). The first 3 features are used in

Annotation	Coverage	Impact
domran:cc, time:same, rigid	45.69	7.09
domran:cc, dom:changed, time:past	19.83	4.65
domran:cc, time:same	15.52	7.29
domran:cc, dom:birth, ran:death, time:past	13.79	2.76
domran:cc, ran:changed, time:future	10.34	0.43
domran:cc, dom:changed, time:past_imm	6.90	0.16
domran:cc, ran:changed, time:future, possible	3.45	0.41
domran:cc, dom:changed, time:same, rigid	3.45	0.08
domran:cc, identity:same, dom:changed, time:past_imm	2.59	0.11
domran:cc, identity:same, dom:changed, time:past	2.59	0.17
domran:cc, dom:changed, ran:changed, time:past	2.59	0.01
domran:cc, time:past	1.72	1.87
domran:cc, time:future	0.86	0.47

Table 3.4: Coverage and Impact results for *c-c* annotations used across the OBO Foundry

a considerable amount of ontologies and their impacts are relatively high. The last feature, although having a reasonably important coverage score, has a very low impact. The features *possible* and *identity:same* are amongst those with the lowest coverages and impacts. The *possible* feature had a low coverage of 3.45% and a very low impact of only 0.41%. *identity:same* had a coverage of only 2.59% and the lowest impact of all at 0.17%. *rigid* was one of the most important features scoring third place for both impact and coverage at 45.69% and 7.09% respectively. With regards to the *c-c* temporal annotations, their coverage ranged from 45.69% to 0.86% and their impacts ranged from 7.09% to just 0.01% with averages of 9.95% and 1.96% respectively. There was a high correlation of 0.82. Ordered by their coverage, the first 4 annotations are amongst those with both the highest coverage and impacts. The features used in the top annotations correspond to the top features. We see *time:same*, *rigid*, *time:past* and *dom:changed* occurring in the highest ranked annotations along with *dom:birth* and *ran:death*, which also happen to be amongst the highest ranked features. The remaining annotations all have very low impact scores, although collectively, some of the features used are still considered to have high measures.

Feature	Coverage	Impact
domran:oo	40.52	15.03
time:al-before-inverse	24.14	10.28
time:al-during	20.69	14.96
time:al-before	18.10	2.69
time:al-meets-inverse	12.93	15.31
time:al-meets	7.76	3.07
time:al-starts	5.17	0.04
time:al-before/al-during	4.31	0.12
time:al-finishes	4.31	0.01
time:al-starts-inverse	2.59	0.00
time:al-finishes-inverse	2.59	0.00
time:al-before/al-equalto	0.86	0.02
time:al-equalto	0.86	0.02

Table 3.5: Coverage and Impact results for *o-o* features used across the OBO Foundry

Annotation	Coverage	Impact
domran:oo, time:al-before-inverse	24.14	10.28
domran:oo, time:al-during	20.69	14.96
domran:oo, time:al-before	18.10	2.69
domran:oo, time:al-meets-inverse	12.93	15.31
domran:oo, time:al-meets	7.76	3.07
domran:oo, time:al-starts	5.17	0.04
domran:oo, time:al-finishes	4.31	0.01
domran:oo, time:al-before/al-during	4.31	0.12
domran:oo, time:al-starts-inverse	2.59	0.00
domran:oo, time:al-finishes-inverse	2.59	0.00
domran:oo, time:al-equalto	0.86	0.02
domran:oo, time:al-before/al-equalto	0.86	0.02

Table 3.6: Coverage and Impact results for *o-o* annotations used across the OBO Foundry

Feature	Coverage	Impact
domran:co	31.90	6.11
time:same	31.90	6.11
rigid	23.28	4.98
possible	12.07	5.95
dom:birth	6.03	3.74
time:future/same	4.31	5.13
time:past/same	3.45	6.43
dom:death	3.45	6.47
dom:death&birth	2.59	0.04
time:past	1.72	0.19

Table 3.7: Coverage and Impact results for *c-o* features used across the OBO Foundry

### 3.4.4 *o-o* relations

Due to the nature of occurrent relations, the annotations only differ by their time features. We therefore only analyse the features themselves. *o-o* relations had a coverage of 40.52% of and an impact of 15.03%. Out of the 12 *o-o* annotations in TRO, the time features used were: *[before']*, *[during]*, *[before]*, *[meets']*, *[meets]*, *[starts]*, *[finishes]*, *[before/during]*, *[finishes']*, *[starts']*, *[before/equalto]*, *[equalto]* that correspond to Allen's relations ordered by their coverage in descending order. Out of the 13 basic Allen relations, 10 were used. Their coverage ranged from 24.14% to 0.86% and their impact ranged from 15.03% to 0.001%, with averages of 11.14% and 4.73% respectively. The correlation between coverage and impact was 0.81. We can see from the data that those features with the highest coverage are amongst those with the highest impacts, for example, annotations with impacts higher than 1% are the top 5 coverage annotations. *before'*, *during* and *meets'* are amongst those with the highest coverages and impacts, which are clearly the most important. *before* also had a high coverage but the impact was lower than the others at only 2.69%. *meets* was the last feature used that had an impact of over 1% at 3.07% and a coverage of 7.76%. The remaining features impact's all fell below 1% and had coverages at only 5.17% or below.

### 3.4.5 *c-o* results

*c-o* relations had a total coverage of 31.90% and an overall impact of 6.11%. The coverage of *c-o* features ranged from 31.90% to 1.72% with an average of

Annotation	Coverage	Impact
domran:co, time:same, rigid	18.97	2.34
domran:co, time:same	18.10	1.99
domran:co, time:same, rigid, possible:possible	12.07	5.95
domran:co, dom:birth, time:same	4.31	5.16
domran:co, dom:birth, time:future/same	4.31	5.13
domran:co, dom:death, time:same	3.45	6.47
domran:co, dom:death, time:past/same	3.45	6.43
domran:co, dom:death&birth, time:same	2.59	0.04
domran:co, dom:birth, time:past	1.72	0.19

Table 3.8: Coverage and Impact results for *c-o* annotations used across the OBO Foundry

12.07%, whilst their impact ranged from 6.47% to 0.04% with an average of 4.52%. The correlation between the two measures was considerably low at just 0.45. We see again that the features *time:same* and *rigid* are amongst those with the highest coverages at 31.90% and 23.28%, and above average impacts at 6.11%. The *possible* feature also had a high coverage and high impact, showing higher importance in *c-o* relations than *c-c* relations. The time features *time:past/same* and *time:future/same* (indicating a time relation of either the same time point or over two time points), both had high impacts at 6.43% and 5.13% respectively, but a lower coverages at only 3.45% and 4.31%. The remaining time feature *time:past* had the lowest coverage at only 1.72% and one of the lowest impacts at 0.19%. Only domain constraint features were used in *c-o* relations. The coverage of the domain constraint features used were all below average, although some had a high impact, such as *dom:death* at 6.47%, which was the second highest impact. The coverage of *c-o* annotations range from 18.97% to 1.72% with an average of 7.66% and their impact ranged from 6.47% to 0.04% with an average of 3.75%. The correlation between the two was -0.12. Overall the impact was low compared to the *o-o* and *c-c* relations, but the coverages of the first 3 were still relatively high. As with *c-c* relations, the annotations with the highest coverages contain the *time:same* feature and in some cases, also the *rigid* feature and the *possible* feature. Those annotations with the highest impacts had multiple time features and domain constraints, but didn't have the highest impacts, hence the low correlation. The remaining annotations had both low coverage and impacts.

Feature	Coverage	Impact
domran:oc	33.62	4.31
time:same	33.62	4.31
rigid	27.59	4.50
ran:birth	11.21	0.70
ran:changed	6.90	0.71
time:future	0.86	0.10

Table 3.9: Coverage and Impact results for *o-c* features used across the OBO Foundry

Annotation	Coverage	Impact
domran:oc, time:same, rigid	27.59	4.50
domran:oc, time:same	14.66	0.67
domran:oc, ran:birth, time:same	11.21	0.70
domran:oc, ran:changed, time:same	6.03	0.80
domran:oc, ran:changed, time:future	0.86	0.10
domran:oc, ran:birth, time:future	0.86	0.10

Table 3.10: Coverage and Impact results for *o-c* annotations used across the OBO Foundry

### 3.4.6 *o-c* results

*o-c* relations had a total coverage of 33.62% and an overall impact of 4.31%. The coverage of *o-c* features ranged from 33.62% to 0.86% with an average of 18.97%, whilst their impact ranged from just 4.5% to 0.1% with an average of 2.44%. The correlation between the two measures was very high at 0.97. Again, we see that the *time:same* and *rigid* are amongst those with the highest coverage at 33.62% and 27.59%, and some of the highest impacts at 4.31%. The range constraint features *ran:birth* and *ran:changed* had acceptably high coverages at 11.21% and 6.90%, but low impacts at only 0.70%. The only other feature used was a time feature *time:future*, but was only used in one ontology and in that ontology had an impact of only 0.1%. The *o-c* annotations had coverages ranging from 27.59% to just 0.86% with an average of 10.2%, and impacts ranging from 4.5% to 0.1% with an average of 1.15%, again with a high correlation of 0.90. The only annotation to have an impact measure over 1% involved the features *time:same* and *rigid*. The remaining annotations had coverages below impacts below 1%. There were 2 annotations with considerable coverage measures but the impact was very low.



Feature	Coverage	Impact
domran:xx	72.41	16.49
time:same	72.41	16.49
rigid	72.41	16.45
identity:same	3.45	0.78

Table 3.11: Coverage and Impact results for  $x$ - $x$  features used across the OBO Foundry

Annotation	Coverage	Impact
domran:xx, time:same, rigid	72.41	16.45
domran:xx, identity:same, time:same	3.45	0.78

Table 3.12: Coverage and Impact results for  $x$ - $x$  annotations used across the OBO Foundry

### 3.4.7 $x$ - $x$ results

$x$ - $x$  relations had a total coverage of 72.41% and an overall impact of 1.496%. The coverage of  $x$ - $x$  features ranged from 72.41% to just 3.45% with an average of 55.17%, whilst their impact ranged from just 16.49% to 0.78% with an average of 12.55%. The correlation between the two measures was very high at 0.99. There were only 2  $x$ - $x$  relations, where the most important was the annotation containing the *rigid* and *same:time* features. The coverage of this annotation was also 72.41% with an impact of 16.45%. The remaining annotation, when compared to the first was significantly unimportant, only being used in 3.45% of the ontologies and a low impact measure of just 0.78%.

Feature Type	Avg. Coverage	Avg. Impact	Cor. Cov-Imp
c-c	21.55	3.52	0.97
o-o	11.14	4.73	0.81
c-o	12.07	4.52	0.45
o-c	18.97	2.44	0.97
x-x	55.17	12.55	0.99

Table 3.13: Average Coverage and Impact scores for each feature type

Annotation Type	Avg. Coverage	Avg. Impact	Cor. Cov-Imp
c-c	9.95	1.96	0.82
o-o	8.69	3.88	0.78
c-o	7.66	3.75	-0.12
o-c	10.20	1.15	0.90
x-x	37.93	8.62	1.00

Table 3.14: Average Coverage and Impact scores for each annotation type

### 3.5 Identifying Requirements

We now go on to answer the first research question by defining the temporal requirements based on the results of the data gathered from the survey. A total of 14 requirements are defined based on these results. Each requirement contains a general description, as well as a list of sub requirements, detailing specific parts of the requirement as a whole, used to aid in our evaluation in the next chapter to get a better understanding on how well certain logics perform in meeting the requirements, and also to further describe the requirement.

The requirements are derived for each domain and range type separately for three reasons. The first is due to the fact that different domain and range types specify different *types* of temporal requirements, so it would make sense to partition these into separate groups. The second reason is that due to the pure dominance of  $x-x$  relations, evaluating important features as a whole would potentially eliminate all features apart from those present in  $x-x$  relations due to their high scoring measures of coverage and impact when compared to features and annotations in other domain and range types. The third reason is that as can be seen from the results in Tables 3.13 and 3.14, it is the case that some domain and range types have higher average coverages than others, but lower average impacts, whilst others have lower average coverages and higher average impacts. It is not clear which would be deemed *more important* in such cases.

For each domain and range type, we consider the averages of coverage and impact for both their features and annotations to filter out the important ones. Those features or annotations that have measures above the average, or features that occur in annotations that have measures above the average are considered to be the important ones for the respective domain and range type.

We begin our requirement identification with  $x-x$  relations.

### 3.5.1 *x-x* requirements

*x-x* relations are considered to be the most important relations. They have the highest coverages for annotations and second highest impacts, and also the highest feature coverages and impacts. Out of the two *x-x* relations that are used, one is responsible for its dominance. The annotation on this relation includes the features *time:same* and *rigid*. Since *x-x* is really a synonym for *c-c/o-o* we can already see the need for the ability to model both continuants and occurrents (which also becomes clearer later on). The only other feature to be used in *x-x* relations was *identity:same*, but the coverage and impact was far too low to be considered important. From the *x-x* results, it is clear that there are 3 important features, and a single important annotation, leading to our first three Temporal Requirements (TRs):

#### TR1 The ability to model continuants

TR1.1 Continuants must be traceable through time

TR1.2 Continuants must maintain their identity through time

TR1.3 Continuants must be able to undergo change

#### TR2 The ability to model occurrents

TR2.1 Occurrents can have limited phases of existence

TR2.2 Occurrents can have temporal parts

#### TR3 The ability to model same time rigid relations between occurrents or continuants

TR3.1 Each relation must hold at a single time point between the same individuals (*time:same*, *rigid*)

TR3.2 The relation can have a duration specified

Each requirement is of the form TRX, TRX.1, . . . . TRX is the main requirement, and TRX.1 etc. are its sub requirements, created from general definitions or the features or annotations the main requirement is based on.

### 3.5.2 *c-c* requirements

*c-c* relations also had high measures of impact and coverage, although lower than *x-x* relations, they were still relatively high enough to be considered important. The features used in *c-c* relations followed a similar pattern to *x-x* relations. *time:same* and *rigid* were those features with the highest impact and coverage measures. Two of the annotations with the highest measures use these features, where the more important of the two uses both, and the lesser uses only *time:same* but still has high scores. This leads us to our next requirements:

**TR4 The ability to model same time rigid relations between continuants**

TR4.1 Each relation must hold at a single time point between the same individuals (*time:same*, *rigid*)

TR4.2 The relation can have a duration specified

**TR5 The ability to model same time relations between continuants**

TR5.1 The relation must hold at a single time point (*time:same*)

The next feature with a very high coverage and impact was *time:past*. It was used in many annotations, but had dominance in the annotation *domran:cc*, *dom:changed*, *time:past* which had both a high impact and coverage. This brings us to our next requirement:

**TR6 The ability to model past time relations between continuants**

TR6.1 The relation must hold between individuals at present and past time points (*time:past*)

The three domain constraint features *dom:changed*, *dom:birth* and *ran:death* also have high enough coverages to consider important and reasonable impacts. *dom:changed* is the most prominent of the three, being used in a highly scoring annotation for both coverage and impact. *dom:birth* and *ran:death* were used in annotations with lower scores but were still high enough to be considered important. This leads us to our next requirement:

**TR7 The ability to model the changing states of continuants, specifically their birth, death and other general changing states**

TR7.1 **Model a continuant coming into existence (birth)**

TR7.2 **Model a continuant going out of existence (death)**

TR7.3 **Model general changes of continuants (change)**

The remaining features all had impacts too low to consider (<1%). For example the feature *time:future* had a high coverage but very low impact, unlike its counterpart *time:past*. The remaining features had similar patterns so we conclude they can be considered unimportant.

### 3.5.3 *o-o* requirements

*o-o* relations also had high measures of impact and coverage, although lower than *x-x* relations, they were still high enough to be considered important. They had lower coverages than *c-c* features but had higher impacts. We consider 5 of the *o-o* features to be important. The first 5 features have the highest impacts (>2.5%) and high coverages (> 7%). The time relations are *before'*, *during*, *before*, *meets* and *meets'*, leading to our next requirement:

TR8 **The ability to model the time constraints of the following Allen's interval relations between occurrents: *before'*, *during*, *before*, *meets*, *meets'***

TR8.1 **Capture the temporal constraints intended by Allen's relations**

The remaining features had very low impacts (<1%) and we conclude that they are considered to be unimportant.

### 3.5.4 *c-o* requirements

*c-o* features had both smaller coverages and impacts than both *c-c* and *o-o* features, but not so low to ignore. We consider *c-o* relations to be important due to some of the measures of the features used, leading to our next requirement:

TR9 **The ability to model relations between continuants and occurrents**

TR9.1 **Be able to make relations between the two types of distinct elements**

The coverages of the features were particularly high, especially for the *time:same* feature. In fact, this feature had the highest coverage and third highest impact. The only other time features with high enough measures to consider were *time:past/same* and *time:future/same* with the former having the highest impact and the latter not so far off. These features were used together in annotations with the features *dom:birth* and *dom:death*, which were amongst those annotations with the highest impacts. With this in mind, we describe the next requirement:

**TR10 The ability to model relations between continuants and occurrents over past, future, and present time points, each including the continuant's state constraints**

TR10.1 Relate the two individuals at a single time point (time:same)

TR10.2 Relate the two individuals at a present and past time point (time:past)

TR10.3 Relate the two individuals at a present and future time point (time:future)

TR10.4 Correctly model the continuant coming into and out of existence (birth, death)

The *rigid* feature shows up again as having some of the highest measures. It is again used only with *time:same* feature in the annotation with the highest coverage, although not the highest impact, but still notable, leading to our next requirement:

**TR11 The ability to model same time rigid relations between continuants and occurrents**

TR11.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR11.2 The relation can have a duration specified

For the first time, we also see the possible feature playing an important role. It has both high coverage and impact and appears in an annotation with the third highest coverage and impact scores, so we add this to the requirements:

**TR12 The ability to allow for multiple future time lines where relations may or may not hold**

**TR12.1 The ability to express multiple futures where relations may hold in one future and not in others**

### 3.5.5 *o-c* requirements

*o-c* relations also have high enough measures to be also considered important. There are only 2 features and one annotation however, with high enough measures to be considered important. Unsurprisingly, the features are *time:same* and *rigid*, which leads us to our final requirements:

**TR13 The ability to model relations between occurrents and continuants**

**TR13.1 Be able to make relations between the two types of distinct elements**

**TR14 The ability to model same time rigid relations between occurrents and continuants**

**TR14.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR14.2 The relation can have a duration specified**

The second research question is concerned with whether or not there is a suitable extension that can handle the requirements identified above. This gives rise to a 15th (non temporal) requirement. By *suitable*, we want the extension to be realisable for OWL.

**R15 Any extension should be decidable and of suitably low complexity, have implementations or at least decision procedures for the most common reasoning problems and be readily available for use**

Tables 3.15 to 3.20 summarise which features and annotations were most important in the selection of the requirements.

## 3.6 Temporal Requirements

The following are a list of requirements gathered in our survey responding to the research questions, along with a list of sub requirements detailing the individual parts of each requirement: The Temporal Requirements (TR) and general Requirements (R) are as follows:

**TR1 The ability to model continuants**

TR1.1 Continuants must be traceable through time

TR1.2 Continuants must maintain their identity through time

TR1.3 Continuants must be able to undergo change

**TR2 The ability to model occurrents**

TR2.1 Occurrents can have limited phases of existence

TR2.2 Occurrents can have temporal parts

**TR3 The ability to model same time rigid relations between occurrents or continuants**

TR3.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR3.2 The relation can have a duration specified

**TR4 The ability to model same time rigid relations between continuants**

TR4.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR4.2 The relation can have a duration specified

**TR5 The ability to model same time relations between continuants**

TR5.1 The relation must hold at a single time point (time:same)

**TR6 The ability to model past time relations between continuants**

TR6.1 The relation must hold between individuals at present and past time points (time:past)

**TR7 The ability to model the changing states of continuants, specifically their birth, death and other general changing states**

TR7.1 Model a continuant coming into existence (birth)

TR7.2 Model a continuant going out of existence (death)

TR7.3 Model general changes of continuants (change)



TR8 The ability to model the time constraints of the following Allen's interval relations between occurrents: *before'*, *during*, *before*, *meets*, *meets'*

TR8.1 Capture the temporal constraints intended by Allen's relations

TR9 The ability to model relations between continuants and occurrents

TR9.1 Be able to make relations between the two types of distinct elements

TR10 The ability to model relations between continuants and occurrents over past, future, and present time points, each including the continuant's state constraints

TR10.1 Relate the two individuals at a single time point (time:same)

TR10.2 Relate the two individuals at a present and past time point (time:past)

TR10.3 Relate the two individuals at a present and future time point (time:future)

TR10.4 Correctly model the continuant coming into and out of existence (birth, death)

TR11 The ability to model same time rigid relations between continuants and occurrents

TR11.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR11.2 The relation can have a duration specified

TR12 The ability to allow for multiple future time lines where relations may or may not hold

TR12.1 The ability to express multiple futures where relations may hold in one future and not in others

TR13 The ability to model relations between occurrents and continuants

- TR13.1 **Be able to make relations between the two types of distinct elements**
- TR14 **The ability to model same time rigid relations between occurrents and continuants**
- TR14.1 **Each relation must hold at a single time point between the same individuals (time:same, rigid)**
- TR14.2 **The relation can have a duration specified**
- R15 **Any extension should be decidable and of suitably low complexity, have implementations or at least decision procedures for the most common reasoning problems and be readily available for use**

## 3.7 Validity of the Survey

### 3.7.1 Analysis of Importance

It may be argued that our measurements can be seen as too simple to determine such a complex notion of importance. Indeed, our measurements are simple ones, but we do believe that they are enough to at least provide a basis for a measure of importance and enough to provide foundation to our requirements. We did have a fair number of other calculations that we tinkered with. One example was another measure that we originally included in our evaluation called the *Weight* of an entity:

$$weight(e) = \frac{\#axioms\ in\ \mathcal{C}\ that\ use\ e}{\sum_{O \in \mathcal{C}} \#axioms\ in\ O\ where\ O\ uses\ e}$$

In addition to the Coverage and Impact, the Weight measured the direct amount of axioms an entity occurred in over all ontologies that used the entity, instead of an overall average as the Impact does. We decided not to include this in our evaluation as it was very unfair since large ontologies with thousands of axioms would provide misleading and biased results. The datasets available include these metrics, but we left these out of our evaluation as no meaningful information could be concluded from the data. The Impact and Coverage were the best measures that we were left with and we do believe these are sufficient and adequate to provide a strong enough basis of a measure of importance.

We also went as far as to attempt to use Formal concept analysis (FCA) [GSW05] in our evaluation. FCA provides a method of data analysis to describe and categorise objects that have similar features. In its most basic form, the input to FCA is a matrix of relationships between objects and their features. The usual output to FCA is a concept semi lattice: a group of concepts which represent sets of features organised into a hierarchical structure (sub concept, super concept). Since our annotations were exactly sets of features (as they are defined), we were interested in performing FCA on TRO itself. The objects would be the relations from TRO and the features would be the temporal features they were annotated with. This would allow us to categorise the concepts into groups that have features in common, which we could then further analyse instead of just analysing individual entities. The FCA procedure identifies concepts by searching for maximal sets of features that are shared by one or more objects. Our intention was to take the concepts produced by FCA, and plug them directly into our evaluation system by computing the same measurements: impact and coverage. In our first attempt, (also during our first annotation schema) the FCA procedure produced a very large semi lattice with only a small number of concepts having more than one feature involved. So we could not infer any more meaningful information from the data sets that we couldn't already get simply from looking at the results from the features or annotations alone. This was mostly due to problems with our initial annotation scheme. But it did help us to correct our annotation scheme. After correcting this on several occasions, we found that there was no benefit from using FCA. We managed to get the annotation scheme to a concise representation, using only 42 features which was manageable, and when we did our final FCA analysis, no new information could be inferred after our normal analysis was conducted. The FCA results are still available and included in the original data sets.

### 3.7.2 Annotations

Although the Relation Ontology is very rich in definitions, it is still in active development, and there are some relations that do not have precise definitions or in some cases, any annotations at all. This made it difficult to be able to directly add the correct features to these relations without any uncertainty. In these few cases, we did the best we could to annotate these relations as follows. When the relation was a sub, equivalent or inverse relation of another relation that

had a definition or a TRO annotation, we would use the information from that relation to appropriately annotate the relation in question. This is sufficient due to OWL’s precise semantics. When there were no annotations, but restrictions on the domain and range, then we would rely on the relation’s name (rdfs:Label) and common sense to infer what the correct annotation should be, if it was obvious. For the cases where it was not directly obvious and not enough information was present to conclude an annotation, we used the reserved default TRO annotation to conclude it would not be used in the study as not enough information was present.

### 3.7.3 Smart matching

In the previous section we defined the notion of *smart matching*. This was where we stated that a relation local to an ontology was equivalent to a RO relation when they *looked* equivalent, i.e., they had the same IRI, *or* they had the same name, or the same label etc. The reason RO exists is that there needed to be standardisation across the OBO library for relations used, to avoid any ambiguity between similar relations used across different ontologies. Ideally, when an ontology in OBO (or any ontology to that matter) intends to use a relation defined by RO, it should, in theory, import the ontology into its signature. The problem with this is that RO is a complex ontology. Its expressivity is very high due to its complex modelling of relations (role hierarchies, role chains, size, etc) and importing such a complex ontology (w.r.t the DL expressivity required) will most likely have a direct negative effect on performance (reasoning time) of any other ontology, if not increase the complexity of the original ontology. If not importing the ontology, then at the least the same IRI of any relation used should be used in order to indicate the intention that the relation is the same relation from RO. Alas, this is not always the case, or so it seems, which is why we introduced the notion of smart matching.

We provide some analysis on how many smart matches we found and how many exact matches we found to show how *valid* our results are, and why we chose to include smart matching.

For each ontology, we iterated through each logical axiom (terminological axiom) and recorded whether or not the axiom contained an IRI match of an RO relation, or otherwise a smart match of an RO relation. We repeated this for every axiom in every ontology, for every relation in RO. When a relation matches

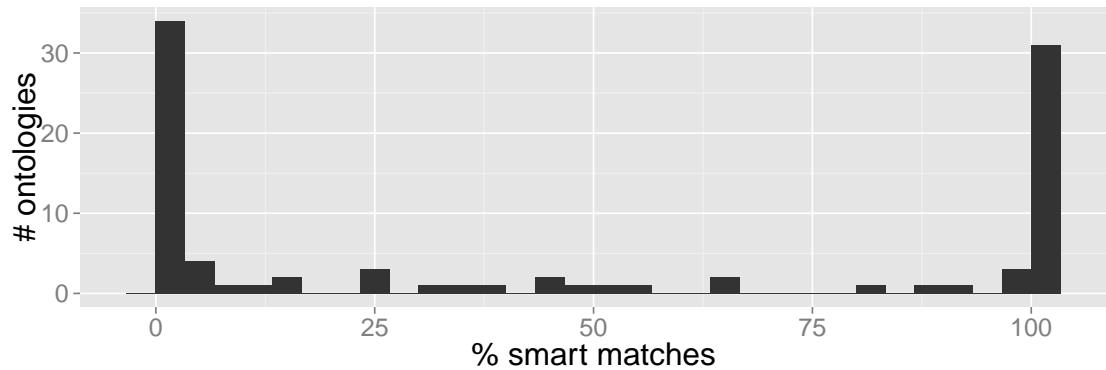


Figure 3.5: A histogram showing the number of ontologies in the OBO Foundry against the % of axioms in each ontology.

the IRI of a relation in RO, we call this an exact match. When it matches a relation based on one of the annotations we defined previously, we call this a smart match.

Out of the 116 ontologies, we found that only 61 ontologies had explicit matches, whereas 72 ontologies had smart matches. 92 ontologies overall had either smart or explicit matches. There were also a total of 31 ontologies with smart matches and no exact matches. This is shown in the histogram in Figure 3.5. If we chose to ignore the smart matches then we would be ignoring a third (33.69%) of the ontologies in the OBO Foundry that could be potentially using RO relations. In terms of the axioms the relations are used in, if we were to ignore axioms that had smart matches, we would be ignoring again roughly a third (29.29%) of all axioms that have any kind of match in the OBO Foundry. If we chose to ignore either the ontologies or the axioms, both are too large an impact to set aside, which was our reason to include them. Of course, it could be the case that all of the smart matches were incorrect matches, but we did investigate some of the matches, and it seemed at least to us that the relations should have been used correctly.

Feature c-c	Coverage	Impact
time:same	47.41	8.53
rigid	45.69	7.09
time:past	31.03	4.31
dom:changed	19.83	4.68
dom:birth	13.79	2.76
ran:death	13.79	2.76

Table 3.15: Important *c-c* features used and selected in the temporal requirements with their Coverage and Impact scores

Feature o-o	Coverage	Impact
time:al-before-inverse	24.14	10.28
time:al-during	20.69	14.96
time:al-before	18.10	2.69
time:al-meets-inverse	12.93	15.31
time:al-meets	7.76	3.07

Table 3.16: Important *o-o* features used and selected in the temporal requirements with their Coverage and Impact scores

Feature c-o	Coverage	Impact
time:same	31.90	6.11
rigid	23.28	4.98
possible	12.07	5.95
dom:birth	6.03	3.74
time:future/same	4.31	5.13
time:past/same	3.45	16.43
dom:death	3.45	6.47

Table 3.17: Important *c-o* features used and selected in the temporal requirements with their Coverage and Impact scores

Feature o-c	Coverage	Impact
time:same	33.62	4.31
rigid	27.59	4.50

Table 3.18: Important *o-c* features used and selected in the temporal requirements with their Coverage and Impact scores

Feature $x$ - $x$	Coverage	Impact
time:same	72.41	16.49
rigid	72.41	16.45

Table 3.19: Important  $x$ - $x$  features used and selected in the temporal requirements with their Coverage and Impact scores

Annotation	Coverage	Impact
domran:cc, time:same, rigid	45.69	7.09
domran:cc, dom:changed, time:past	19.83	4.65
domran:cc, time:same	15.52	7.29
domran:cc, dom:birth, time:past, ran:death	13.79	2.76
domran:oo, time:al-before-inverse	24.14	10.28
domran:oo, time:al-during	20.69	14.96
domran:oo, time:al-before	18.10	2.69
domran:oo, time:al-meets-inverse	12.93	15.31
domran:oo, time:al-meets	7.76	3.07
domran:co, time:same, rigid:rigid	18.97	2.34
domran:co, time:same	18.10	1.99
domran:co, time:same, rigid:rigid, possible:possible	12.07	5.95
domran:co, dom:birth, time:same	4.31	5.16
domran:co, dom:birth, time:future/same	4.31	5.13
domran:co, dom:death, time:same	3.45	6.47
domran:co, dom:death, time:past/same	3.45	6.43
domran:oc, time:same, rigid	27.59	4.50
domran:xx, time:same, rigid	72.41	16.45

Table 3.20: Important Annotations used and selected in the temporal requirements with their Coverage and Impact scores

## Chapter 4

# Evaluating Temporal Extensions & Representations

*Is there currently a suitable temporal extension for DLs that will allow for a faithful representation of the Temporal Requirements? And one that is decidable and of suitably low complexity?*

As we have now identified a set of Temporal Requirements (TRs), we now go on to evaluate current temporal extensions and representations against these requirements. The extensions and representations we use for our evaluation are those introduced in Chapter 2: combinations of the Temporal Logics (TLs) LTL and CTL and Descriptions Logics (DLs) ( $LTL_{DL}$  and  $CTL_{DL}$ ), the Fluent ontology and extensions of DLs with concrete domains ( $\mathcal{ALC}(\mathcal{D})$ ).

In Chapter 3, we identified a set of the most important temporal features and annotations from the Temporal Relation Ontology (TRO). TRO relations can be described by their annotations and features so we can also find important relations based on these results to use in our evaluation. Our evaluation involves choosing the most used TRO relations based on the important features and annotations identified, and we show how each relation chosen and its temporal features can be modelled in each logic.

Each annotation and feature may correspond to many relations, but due to space considerations we consider only a few per TR. The relations we use along with their corresponding annotations and features can be seen in Table 4.1. For each TR, and each relation corresponding to important features and annotations of each requirement, we evaluate each TDL's suitability by showing how well they can model these relations. Each TDL will score a ✓ for every sub requirement



Important Annotation	TRO Relation
domran:cc, time:same, rigid	connected to
domran:cc, dom:changed, time:past	develops from
domran:cc, time:same	aligned with
domran:cc, dom:birth, time:past, ran:death	child nucleus of
domran:oo, time:al-before-inverse	causally downstream of
domran:oo, time:al-during	happens during
domran:oo, time:al-before	precedes
domran:oo, time:al-meets-inverse	immediately preceded by
domran:oo, time:al-meets	immediately causally upstream of
domran:co, time:same, rigid	involved in
domran:co, time:same	input of
domran:co, time:same, rigid, uncertain	capable of
domran:co, dom:birth, time:same	existence starts during
domran:co, dom:birth, time:future/same	existence starts during or after
domran:co, dom:death, time:same	existence ends at point
domran:co, dom:death, time:past/same	existence ends during or before
domran:oc, time:same, rigid	occurs in
domran:xx, time:same, rigid	part of

Table 4.1: TRO important annotations with a corresponding TRO relation

that it meets faithfully, which together will act as a measure of how well it meets the requirement as a whole. The requirement as a whole will also receive a separate score of either  $\checkmark\checkmark$ ,  $\checkmark$  or  $\times$  where the first means the logic fully meets the requirement, the second means it partially meets the requirement, and the third means it does not meet the requirement at all or is unsuitable. This will help to compare logics against each other and determine for each logic which specific parts it performs well in and which parts it under-performs in.

For each evaluation, we opt to use  $\mathcal{ALC}$  [SSS91] as the DL fragment for each TDL.  $\mathcal{ALC}$  is a popular logic that has been widely studied. It has a suitable level of expressivity for what we need to express in our evaluation, and each TDL in question has been widely studied at least up to their  $\mathcal{ALC}$  combination, making it a suitable choice.

## 4.1 $LTL_{\mathcal{ALC}}$ & $CTL_{\mathcal{ALC}}$ Evaluation

We begin with the evaluation of both  $LTL_{\mathcal{ALC}}$  and  $CTL_{\mathcal{ALC}}$ . We will only use  $CTL_{\mathcal{ALC}}$  in our evaluation where  $LTL_{\mathcal{ALC}}$  is not sufficient and could benefit from a CTL extension. We begin by evaluating this logic against Temporal Requirement 1 (TR1).

### TR1 The ability to model continuants

TR1.1 **Continuants must be traceable through time**

TR1.2 **Continuants must maintain their identity through time**

TR1.3 **Continuants must be able to undergo change**

In both  $LTL_{\mathcal{ALC}}$  and  $CTL_{\mathcal{ALC}}$ , as with standard OWL, we only have one type of domain element, which we call individuals. We use concept descriptions, axioms and assertions to enforce restrictions on the domain to describe particular things that are of interest to us. Recall how we interpret continuants:

*“An entity that exists in full at any time in which it exists at all, persists through time while maintaining its identity and has no temporal parts.”*

Continuants will be represented in the only and obvious way; by elements in the domain. Due to the semantics of  $LTL_{\mathcal{ALC}}$ , we can view interpretations as sequences of standard DL domains (over a discrete and linear time line, isomorphic to  $\mathbb{N}$ ) which have a rigid interpretation over the individuals, i.e., we assume that  $a_i^{\mathcal{I}} = a_j^{\mathcal{I}}$  for all indices  $i, j \in \mathbb{N}$ . This bodes well for continuants since we are able to refer to any element at multiple time points (in a constant domain environment), ensuring that it is the same individual we refer to at each time point. Thus the standard elements of the interpretation domain can easily and correctly stand in as representations for continuants. We can simply introduce a class *Continuant*, meant to represent all continuants in the domain. Consider Figure 4.1. Here we have an interpretation with a single element  $a$  being an instance of class  $C_1$  at time  $t$ , and an instance of class  $C_2$  at time  $t + 1$ . Although  $a$  is an instance of different classes at different time points, it is the *same* element  $a$  due to the rigid interpretation. The change of class membership highlights how we can consider

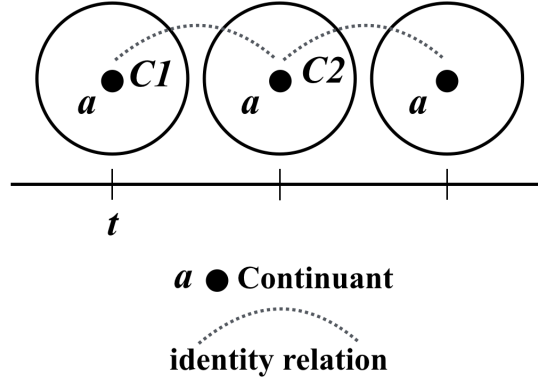


Figure 4.1: An illustration of the rigid interpretation of elements in  $LTL_{ACC}$ , and how they can be used for the effective modelling of Continuants and the modelling of change.

the changes that continuants can go through. We can capture this type of change in the axiom

$$C_1 \sqsubseteq \bigcirc C_2$$

In this axiom we specify that any instance of  $C_1$  at any time point must be an instance of  $C_2$  at the next time point. This corresponds to the example above in Figure 4.1. This shows us how we can effectively trace an individual through time, whilst maintaining its identity, and at the same time, capture some type of change by signifying a change in class. Most of the success is due to the rigid interpretation of the individuals and how we can interact with the elements via the temporal operators on class expressions. What we cannot do however is consider continuants at certain specific time points. For example, if we wanted to consider a continuant  $C$  and make a statement about it at a time point, say 1, this cannot be done since we have no operators to quantify over each of the time points individually. This is a downside to the logic which will continue to show in later requirements. The logic adopts a qualitative approach to time as opposed to a quantitative approach, meaning class expressions and axioms are all interpreted globally, and nothing can be said about classes at specific time points.

There are also other types of change that we need to consider for continuants also regarding their changing states. We saw in our survey that some features

include considering continuants coming into and out of existence. There may be points along the time line where some entities should not exist, and others where they do exist. Suppose a continuant  $c$  was to come into existence at a time point  $t$ , i.e., it was non-existent or present at any time point prior to  $t$ . Then the  $c$  should surely not be a part of any domain prior to time point  $t$ . Similarly, if the continuant was then to no longer exist after a time point  $t'$ , then the element should no longer be in any domain after  $t'$ . To account for this type of change, we believe we have two options. The first is to consider several restrictions on the semantics, specifically on the interpretation domains. The second is to introduce classes meant to stand in as representations of states of continuants as they *transition* between possible states. Focussing on the first, these types of constraints fall directly into the hands of domain constraints [WZ98a, LWZ08], as introduced in Section 2.3.1. Varying domains seem the definite choice, since elements should be able to come in to and out of existence - entities can indeed be *born* and later *die*. To interact with the varying domain, we have to make use of temporalising the  $\top$  concept operator. For example, at any time point  $t$ ,  $\top$  represents all the individuals currently existing,  $\neg\bigcirc\top$  represents all individuals who will not be existing at the next moment in time and so on. The only downside to this approach is that a non-existent individual really is *non-existent*. For want of a better phrase, a non-existent element really is a member of the empty set, i.e.,  $\perp$ , so nothing can be said about such an element. In some cases this may be a shortfall. For example, although an entity may be *dead*, or at least in an *inactive* state, statements may still wish to be made, such as future statements like the entity will eventually become active. This leads us to the second approach of instead introducing classes to stand in for these states in a constant domain environment.

We consider 3 states a continuant can be in, a *Before* state, representing continuants before they become active and therefore non-existent, an *Active* state representing continuants whilst they are active and therefore existent, and an *After* state representing continuants after they are active, and again, non-existent. By using classes we can also make constraints on them to specify how continuants can be members of these classes. For example, if we know the order of these classes are as follows  $Before < Active < After$ , and we assume that all classes were disjoint from each other (no continuant can be active and inactive

at the same time) then we can introduce axioms to constrain these as follows:

$$Continuant \sqsubseteq Before \sqcup Active \sqcup After \quad (4.1)$$

$$Before \sqsubseteq (\Diamond Active) \sqcap (\Box^- Before) \quad (4.2)$$

$$Active \sqsubseteq (\Diamond After) \sqcap (\Diamond^- Before) \quad (4.3)$$

$$After \sqsubseteq (\Box After) \sqcap (\Diamond^- Active) \quad (4.4)$$

(4.1) enforces that every continuant must be in one of the three states. (4.2)-(4.4) together with (4.1) enforce that every continuant must go through all the states in order (here we use the inverse operators (past operators)  $\Diamond^-$  and  $\Box^-$  to illustrate this - although not directly available in  $LTL_{\mathcal{AC}}$ , it saves us effort of having to increase the size of axioms to account for the same meaning by using occurrences of negations and until operators to achieve the same task. We can view this as the  $LTL_{\mathcal{AC}}^-$  fragment of  $\mathcal{DLR}_{\mathcal{US}}$  if need be, but for now we will assume past operators are available to us). Of course, sometimes this may not be the case and some modelling choices may lead to only considering always *Active* continuants that do not consider any other state for example (in which case constant domains alone would be sufficient). Clearly we can generate more types of restrictions based on certain requirements, but the axioms above capture our intention in the simple case. It is important to note however that these classes are really just classes, and they do not capture fully the essence of existence. If a continuant really was non existent then it should not be able to be instances of other classes or have relations on them for example, which is still possible in this case. If an element was an instance of the *Before* class, it could still be a member of another class, or even have relations to over individuals for example. To fully enforce this, one would have to make the restriction that the *Before* and *After* state were disjoint from every other class in the ontology. In later requirements that rely heavily on these constraints, we will go on to show which is more beneficial.

For TR1, the scores are as follows. Clearly we are able to effectively trace a continuant through time whilst maintaining its identity due to the rigid interpretation of individuals and the temporal operators, so for TR1.1 and TR1.2, we give both a score of  $\checkmark$ . For TR1.3, we do not fully manage to capture the correct type of significant change. Simple changes are possible, but changes in existence are more difficult, therefore we give it a score of  $\times$ . Finally, for an overall score, we give TR1 only a  $\checkmark$ . This is due to the lack of quantification available, and the

problems encountered with TR1.3.

## TR2 The ability to model occurrents

### TR2.1 Occurrents can have limited phases of existence

### TR2.2 Occurrents can have temporal parts

Modelling occurrents is slightly more difficult than the continuant case. Recall what is required for occurrents:

*an entity that has temporal parts, unable to undergo change, and unfold in temporal phases*

In biology, occurrents are those types of entities (often referred to as processes) that only exist at certain times and have limited temporal phases of existence, usually having different temporal parts. In the standard setting, the standard interpretation domain elements exist at all time points, and since occurrents may go through *limited temporal phases of existence*, there will definitely be points along the time line where some occurrents will exist, and others where they will not. Therefore we can see yet another use case for the issue of modelling existence, either using varying domains, or the state classes introduced previously: *Before*, *Active* and *After*. As an example, suppose we wanted to model an Occurrent  $O$  that lasted 3 consecutive time points. In the varying domain option we can model this according to the following axiom:

$$\begin{aligned} O \sqsubseteq & (\neg(\bigcirc^-\top) \sqcap \bigcirc(O \sqcap \bigcirc(O \sqcap \neg \bigcirc(\top)))) \sqcup \\ & (\bigcirc^-(O \sqcap \neg \bigcirc^-(\top)) \sqcap \bigcirc(O \sqcap \neg \bigcirc(\top))) \sqcup \\ & (\bigcirc^-(O \sqcap \bigcirc^-(O \sqcap \neg \bigcirc^-(\top))) \sqcap \neg \bigcirc(\top)) \end{aligned} \quad (4.5)$$

In (4.5) we have to specify each of the possible temporal phases the occurrent  $O$  can be in, in a lengthy and quite complex axiom. The first operand of the disjunct specifies the case where  $O$  is in the first temporal phase (each temporal phase being each distinct and sequential time point where  $O$  exists), where it has two time points to go until it ceases to exist. The operand states that  $O$  was at the previous moment in time non-existent, and will remain an instance of  $O$  for the next two time points (remaining existent) before no longer existing at the third time point in the future. The remaining operands of the disjunct capture the other cases where  $O$  is in the second or third temporal phase. The longer the

duration of  $O$ , the longer and more complex the axiom gets. Even in the constant domain case, where we use the state classes instead, we see similar results:

$$\begin{aligned} O \sqcap Act \sqsubseteq & (\bigcirc^- Bef \sqcap \bigcirc(O \sqcap Act \sqcap \bigcirc(O \sqcap Act \sqcap \bigcirc Aft))) \sqcup \\ & (\bigcirc^-(O \sqcap Act \sqcap \bigcirc^- Bef) \sqcap \bigcirc(O \sqcap Act \sqcap \bigcirc Aft)) \sqcup \\ & (\bigcirc^-(O \sqcap Act \sqcap \bigcirc^-(O \sqcap Act \sqcap \bigcirc^- Bef)) \sqcap \bigcirc Aft) \end{aligned} \quad (4.6)$$

Here, due to the fact that the domain is constant, we have to specify the state of  $O$  every time we mention it in the axiom to ensure the state of  $O$  is correct. This axiom follows a similar pattern as the varying domain axiom. The only real addition we get in the constant domain is that the entities exist at all time points, allowing possible constraints to be made on entities in the inactive states. It is difficult to say if this is a benefit or a drawback. It depends on what any ontology author wishes to say. On the one hand it may seem unfaithful to have non existing entities existing in the ontology, but on the other hand to be able to make future statements about them may be an advantage. Either way, the modelling seems very similar, and they both suffer from largely sized axioms to model something quite simple.

In both examples, we can also see identity playing a role, although not directly obvious. Although occurrents are not described as being able to *endure through time* or even *undergo change* as continuants do, due to the semantics of  $LTL_{\mathcal{AC}}$ , it seems these characteristics are undesirably inherited. The  $O$  being modelled in each example is being represented by the same element through each temporal phase. The problem lies not with the identity of the main occurrent enduring through its existence, but more with the fact that it can technically undergo change but should not be able to. This is difficult to constrain. What could it mean for an occurrent  $O$  not to change? Maybe declaring it to be an instance of  $O$  until it no longer exists would suffice, for example in the axiom:

$$O \sqsubseteq OU \neg \bigcirc \top \quad (4.7)$$

but this axiom would only state that any member of the  $O$  class remains a member of the  $O$  class, and not that it cannot be a member of any other class. For this to be enforced, it would need to be disjoint from any other class. Other changes could be changes in relations, for example, if  $O$  has a relation to a class  $X$  then it should have the relation to the class  $X$  until it no longer exists (we focus on

this problem in more detail in TR3).

Occurrents are also described as having temporal parts. The temporal parts are also considered to be occurrents, and take place entirely within the occurrent itself, usually related via common partonomy relation such as the relation *part of* and its inverse *has part*. The temporal parts are often referred to as sub processes in biology and often represented by separate individuals with different identities. Using a similar example as above, suppose during  $O$ 's lifetime, it had 2 successive temporal parts, the first,  $O_1$  lasting a single time point and occurring during the first temporal phase of  $O$ , and the second,  $O_2$  lasting two time points occurring during the last two temporal phases of  $O$ . We could represent such temporal parts in the varying domain example as follows (we assume  $O_1$  and  $O_2$  are disjoint) :

$$hasPart = p$$

$$\begin{aligned} O \sqsubseteq & (\exists p.(O_1) \sqcap \neg(\bigcirc^- \top) \sqcap \bigcirc(O \sqcap \exists p.(O_2) \sqcap \bigcirc(O \sqcap \exists p.(O_2) \sqcap \neg \bigcirc \top))) \sqcup \\ & (\exists p.(O_2) \sqcap \bigcirc^-(O \sqcap \exists p.(O_1) \sqcap \neg \bigcirc^- \top) \sqcap \bigcirc(O \sqcap \exists p.(O_2) \sqcap \neg \bigcirc \top)) \sqcup \\ & (\exists p.(O_2) \sqcap \bigcirc^-(O \sqcap \exists p.(O_2) \sqcap \bigcirc^-(O \sqcap \exists p.(O_1) \sqcap \neg \bigcirc^- \top)) \sqcap \neg \bigcirc \top) \end{aligned} \quad (4.8)$$

$$O_1 \sqsubseteq \neg \bigcirc \top \sqcap \neg \bigcirc^- \top \quad (4.9)$$

$$O_2 \sqsubseteq (\neg \bigcirc^- \top \sqcap \bigcirc(O_2 \sqcap \neg \bigcirc \top)) \sqcup (\neg \bigcirc \top \sqcap \bigcirc^-(O_2 \sqcap \neg \bigcirc^- \top)) \quad (4.10)$$

(4.8) is an extension of the original, where we also take into account the parts  $O$  must have during each of its temporal phases. For example, the first operand of the disjunct is the case where  $O$  is in the first temporal phase, and must therefore have a part that is  $O_1$  at the current time point and  $O_2$  at the next two time points. We also have to add two additional axioms that state the existent constraints on both  $O_1$  and  $O_2$  in (4.9) and (4.10). A problem exists with the identity of those temporal parts that span over a single time point. Since  $O_2$  lasts two time points, the identity of this element should remain the same. In fact, according to the third axiom, it should only last two time points. However, the individual the  $O$  is related to for its last two time points need not be the same element. It could in fact be represented by two distinct elements and therefore, the preferred constraints on  $O_2$  existing only during the time of its parent  $O$  could not be met. The problem lies with rigidity - we need to ensure the element it is related to is the same element at both time points. We focus on this constraint in more detail in the evaluation of TR3. It seems the only way to solve the problem of the existence constraints would be to embed the third axiom into the first



axiom, making it longer and more complex.

Again, each approach adopts a qualitative view, making it impossible for us to make statements about the similar occurrents that may have different constraints at different time points. We make global constraints about occurrents which can in some cases be quite detrimental. Processes are known to span different durations at different points in time but in LTL we can only make general statements about concepts making it difficult to model knowledge only at specific points in time. Again this would be different in a quantitative environment, but this is not something that can be captured in  $LTL_{ACC}$ .

It is clear that we can model occurrents in  $LTL_{ACC}$ , but we fail to capture the most important aspects of them. The rigid interpretation of individuals bodes well for occurrents again, but problems exist when attempting to capture the temporal parts and duration of occurrents. For TR2.1, we give this a score of ✗ since existence is crucial for occurrents and it is not something that can be easily captured in  $LTL_{ACC}$ . We give 2.2, a score of ✓ however. Since the rigid interpretation again works well for identity and tracking occurrents through time, they can have temporal parts, but any part spanning over multiple time points is difficult to correctly enforce. We give ✗ as an overall score for TR2. This is due to the inability to capture important features such as duration or existence faithfully which is again crucial for modelling occurrents.

### TR3 The ability to model same time rigid relations between occurrents or continuants

TR3.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR3.2 The relation can have a duration specified

The relation we use in our evaluation is *part of* (*domran:xx*, *time:same*, *rigid*). Recall the definition for *part of*:

*“A core relation that holds between a part and its whole. Part-hood requires the part and the whole to have compatible classes: only an occurrent can be part of an occurrent; only a process can be part of a process; only a continuant can be part of a continuant; only an independent continuant can be part of an independent continuant; only an immaterial entity can be part of an immaterial entity; only a*

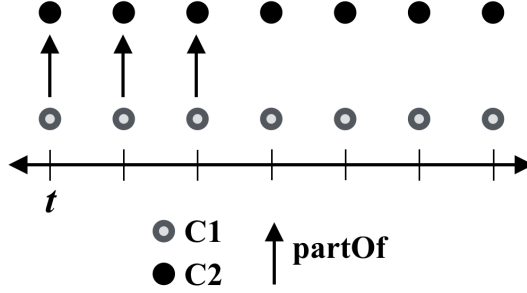


Figure 4.2: An illustration of the relation *part of*. In this example a continuant C1 is part of a continuant C2 for 3 consecutive time points, starting at time  $t$ , and then is no longer related to C2.

*specifically dependent continuant can be part of a specifically dependent continuant; only a generically dependent continuant can be part of a generically dependent continuant. Occurrents are not subject to change and so parthood between occurrents holds for all the times that the part exists. Many continuants are subject to change, so parthood between continuants will only hold at certain times, but this is difficult to specify in OWL”.*

We begin with the *continuant part of continuant* case. Our example is as follows. Given two classes  $C_1$  and  $C_2$  representing continuants, suppose we wanted to model that the  $C_1$  is part of  $C_2$  for a duration of 3 consecutive time points, as shown in Figure 4.2. The standard OWL way to model this relation would be the axiom:

$$C_1 \sqsubseteq \exists \text{partOf}.C_2 \quad (4.11)$$

Since axioms are globally interpreted in  $\text{LTL}_{\mathcal{AC}}$ , this enforces that any instance of  $C_1$  at any time  $t$  must have a *partOf* relation to an instance of  $C_2$  at the same time point  $t$ , by definition of the possible world semantics in  $\text{LTL}_{\mathcal{AC}}$ . However, this is not even close to the temporal constraints we require. The duration of 3 time points is not captured, nor is the rigidity constraint. We first focus on attempting to manage the duration:

$$C_1 \sqcap \bigcirc(C_1 \sqcap \bigcirc(C_1)) \sqsubseteq \exists \text{partOf}.C_2 \sqcap \bigcirc(\exists \text{partOf}.C_2 \sqcap \bigcirc(\exists \text{partOf}.C_2)) \quad (4.12)$$

In (4.12) we state that any element that is a  $C_1$  for three consecutive time points, must have a *partOf* relation to a  $C_2$  for the three same time points. This axiom at least enforces some kind of duration of the *partOf* relation, however the rigidity constraint is not yet satisfied. As the axiom currently stands,  $C_1$  could actually be related to 3 different instances of  $C_2$  over the 3 consecutive time points.

Another possibility could be the axiom

$$C_1 \sqcap \bigcirc(C_1 \sqcap \bigcirc(C_1)) \sqsubseteq \exists \text{partOf}.(C_2 \sqcap \bigcirc(C_2 \sqcap \bigcirc(C_2))) \quad (4.13)$$

which states that any instance of  $C_1$  for three consecutive time points must be related to a single individual that is an instance of  $C_2$  for the same three time points. Although now the individual is the same, there is now only one relation that holds only in a single time point.

It seems the only way  $\text{LTL}_{\mathcal{AC}}$  can come close to a rigid relation is to allow the *rigid* relation characteristic to be added to *partOf*. Recall from Chapter 2 that a relation  $R$  is *rigid* if in all interpretations  $\mathcal{I}$ , if there exists a pair  $(x, y) \in R^{\mathcal{I}_i}$  for some  $i \in \mathbb{N}$ , then for all  $j \in \mathbb{N}$  where  $j \neq i$ ,  $(x, y) \in R^{\mathcal{I}_j}$ . Therefore, if we declare *partOf* as being rigid, then (4.11) seems correct, since any elements that have the relation at any time will have the same relation at all times. A huge modelling problem exists with this in that it is impossible to enforce that the relation only holds during a finite duration, i.e. we cannot specify a duration on the *rigidity*. Another problem lies with the fact that declaring a relation to be rigid, declares a global constraint on the entire relation, which is not always what we want, we cannot consider only contained or partial cases. We may only want the relation to be rigid *local* to certain axioms, in this case local to  $C_1$  and  $C_2$  for this specific case. Declaring *part of* to be rigid would mean that any other elements related by *part of* would also be related infinitely. Possibly the most important downside to having rigid relations is the negative effects on complexity (outlined in Chapter 2). Usually, declaring a single rigid role leads to undecidability, even in lightweight  $\text{LTL}_{DL}$  combinations unless several restrictions are enforced on the knowledge base, such as acyclic or empty TBoxes. Another problem easy to overlook is in relation to the qualitative nature of the way we interact with the time line. Since the axioms are interpreted globally, along with all concept descriptions, we cannot make any assertions about when the part of relation should hold. For example we cannot express that the relation only holds

between certain time points.

We run into similar problems when attempting to model partonomy between occurrents, and any rigid relations in general. We saw an example of when rigid relations are required for partonomy between occurrents in TR2. As the definition suggests, partonomy can only exist between occurrents during the times they exist. We run into not only the same problems as before, but we encounter more here. Since each occurrent should not exist at all time points, it is crucial that the relation should also not exist during all time points. How can it if certain individuals may have the relation at one time point and then cease to exist at another time point? Having rigid relations would go against this principle and would therefore be an unfaithful modelling and possibly lead to an inconsistent modelling. This may suggest that in some cases varying domains may not be the ideal solution if rigid roles were required, and possibly the state constraints would be better suited since they use constant domains, and therefore no existence problems would exist in the case of rigid roles.

For TR3.1 we give  $LTL_{ACC}$  a score of ✓. When introducing rigid relations, the sub requirement holds - the same individuals are related at each consecutive time point. We give TR3.2 a score of ✗. There is no way to capture duration of a relation without it becoming non rigid. For TR3 as a whole, we give it a score of ✗. This is because of the fact that durations cannot be captured which is crucial for occurrents (since we do not want relations to hold when entities do not exist) and declaring a relation to be rigid is an overstatement of what we want. We would like a relation to be rigid only for certain axioms, and only for certain durations, not a global constraint on an entire time line, which is what rigid relations in  $LTL_{ACC}$  actually are.

#### **TR4 The ability to model same time rigid relations between continuants**

**TR4.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR4.2 The relation can have a duration specified**

We give TR4 the same scores as TR3 due to their similarity. TR4 differs only in that we are interested in the continuant case only, but the same problems still hold.

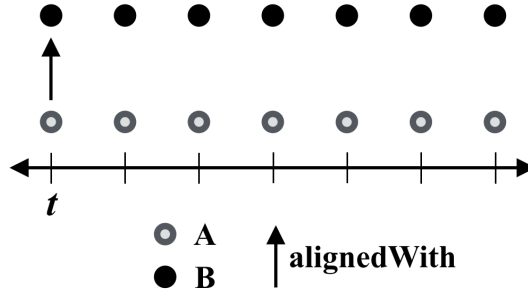


Figure 4.3: An illustration of the relation *aligned with*. In this example the continuant *A* is aligned with the continuant *B* at a single time point *t*, and the relation holds only at this single time point.

#### TR5 The ability to model same time relations between continuants

##### TR5.1 The relation must hold at a single time point (time:same)

TR5 is significantly easier to model than the rigid case in TR4. We use the relation *aligned with* in our example. *aligned with* has a generally atemporal annotation, i.e., apart from the fact that the relation relates continuants, there is no other temporal information present other than that the relation must hold at a single time point (the relation may be rigid, but it is not necessary). This makes modelling the relation particularly easy in the sense that the general OWL modelling applies. Consider Figure 4.3, showing how the relation may look along the time line. A possible axiom could be:

$$A \sqsubseteq \exists \text{alignedWith}.B \quad (4.14)$$

Since we have a global evaluation of axioms, any instance of *A* must have a *aligned with* relation to some instance of *B*. This works well in a qualitative environment, when the relation must hold at all times for any instance of *A*. For the times when the relation may only hold for a single time point or only certain durations, there is no easy way if at all to enforce this. Again this is related to the issue of not being able to quantify over certain time points. We give TR5.1 a score of ✓. We capture what is required; a relation holding in a single time point. We give TR5 a score of ✓. This is again due to the problem of quantification in a qualitative environment.

#### TR6 The ability to model past time relations between continuants

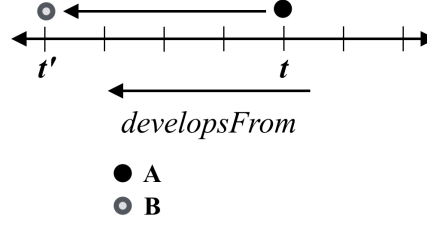


Figure 4.4: An illustration of the relation *develops from*. In this example, the continuant *A* develops from the continuant *B* at times *t* and *t'* respectively where  $t > t'$ . The continuants may be identical, although this is not necessary.

**TR6.1 The relation must hold between individuals at present and past time points (time:past)**

*develops from* is a relation that relates two individuals over a present and past time point. It is a super relation to *transformation of* (see Chapter 3). Suppose we wanted to express that *A develops from B* at time *t* and *t'*, shown in Figure 4.4. We could use the axiom:

$$A \sqsubseteq \exists \text{developsFrom} . (\Diamond^- B) \quad (4.15)$$

Here we state that any instance of *A* must have a *develops from* relation to an individual that was a *B* at some previous time point. This would be sufficient *if* the relation's start time point was the present time point and the relations end time point was time point referenced by  $\Diamond^-$ . However this is not the case since due to the restrictions on the semantics, all relations must hold between individuals in a single time point, hence the relation itself holds only in the current time point (relative to *A*). Although somewhat misleading, we believe the axiom to be sufficient to represent the relationship. As there are no constraints on the existence on each individual - we assume they exist at both time points - then regardless of when the relation holds, it does relate the correct individuals. The axiom does again however rely on past time relations, which are not available in standard  $\text{LTL}_{\mathcal{AC}}$  but are available in any extension with inverse operators. In some cases, development is often a relation involving identity, where the individuals in question are identical to each other. In fact, the relation *transformation of*, which is a sub relation of *develops from* is one of these relations, and is described as a “*relation of identity: each adult is identical to some child existing at some*

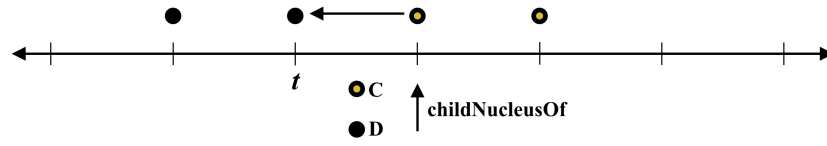


Figure 4.5: An illustration of the relation *child nucleus of*. In this example the continuant *C* is a child nucleus of the continuant *D* which existed at some previous time point *t* when *D* did not exist.

*earlier time*” In our example, *A* could be the same individual as *B*, just in a newly developed form. We saw a similar example in Chapter 2 regarding the *Drosophila* ontology [CRGOS13]. In this case, using existential operators to relate the individuals is not a faithful representation since it does not restrict the filler of the relation to the same individual. It is better to not use relations at all and opt to simply use class restrictions as follows:

$$A \sqsubseteq (\Diamond^- B) \quad (4.16)$$

Here we specify that any instance of *A* was at some previous moment and instance of *B*. This handles the identity case exactly and uses multiple time points correctly. We give TR6.1 a score of ✓. We believe our modelling attempt is sufficient to simulate the relation holding over multiple time points, and when identity is concerned, eliminating relations is sufficient. We give TR6 a score of ✓✓.

**TR7 The ability to model the changing states of continuants, specifically their birth, death and other general changing states**

TR7.1 Model a continuant coming into existence (birth)

TR7.2 Model a continuant going out of existence (death)

TR7.3 Model general changes of continuants (change)

In this example we focus on the RO relations *develops from* and *child nucleus of*. *child nucleus of* is defined as:

*“c is a child nucleus of d if and only if c and d are both nuclei and parts of cells c' and d', where c' is derived from d' by mitosis and the genetic material in c is a copy of the generic material in d”.*

We stick to a simple interpretation of the identity of both  $c$  and  $d$ . We assume that the nucleus  $c$  has a different identity to the nucleus  $d$ . Since  $d$  was split, it effectively no longer exists, hence the use of the *birth* and *death* features in the annotation. Suppose we wanted to represent *C is a child nucleus of D* at times  $t$  and  $t'$ , as shown in Figure 4.5. We first show how we could represent this using the state classes introduced above in a constant domain environment:

$$C \sqcap \text{Active} \sqsubseteq \Diamond^-(\text{Before} \sqcap \bigcirc (\text{Active} \sqcap \exists \text{childNucleusOf} . (\text{After} \sqcap \bigcirc^-(\text{Active} \sqcap D))))$$

We specify both the birth of  $C$  and the death of  $D$  using the state classes. We also enforce that the time points are correct in the sense that the transitions between the states are aligned -  $C$  becomes active after  $D$  dies.  $D$  must have existed in the past but doesn't exist now, and the opposite for  $C$ . Due to the global nature of the axioms, we also have to include the state information of the continuants in the axiom, since we would not want this axiom to hold for any active instance of  $C$ . Again, the relation should cover multiple time points, similar to *develops from* from TR6, but we believe our modelling suffices. One interesting aspect to point out is that the relation holds between an active instance of  $C$  and an *inactive* instance of  $D$ . This goes against the faithfulness of existence - if  $D$  really was in the *After* state, then it should not surely have any incoming relations. However, switching to the varying domain environment would not solve the problem here, and would actually make matters worse:

$$C \sqcap \neg \bigcirc^- \top \sqsubseteq \exists \text{childNucleusOf} . (\neg \top \sqcap \bigcirc^- D) \quad (4.17)$$

(4.17) can never be satisfied. Both entities need to exist at the same time for any relation to hold between the two entities. As we saw in TR6, all relations have to hold within a single world, but in this relation, the entities cannot exist in the same world, at least not in a faithful representation anyway. The state classes are better suited for this approach.

The annotation for *develops from* also contains the feature *changed*, indicating the entity went through some kind of change (i.e., development). An example of a possible development could be that it was a member of different classes before and after the change. We saw an example in TR6 of how change can be captured when the identity of the individuals being related are the same. We now show



how this can be done when the identity is not the same. Consider the following axiom

$$A \sqsubseteq \exists \text{developsFrom} . (\Diamond^- B) \sqcap \Diamond^-(A') \quad (4.18)$$

where  $A$  is the new class and  $A'$  was the old class which it *changed* from. Since we have a rigid interpretation of individuals, then we can be assured the same element is and was a member of both of these classes at different time points. A problem we face now is that we cannot enforce that the time point when the relation held, is the same time point when the change occurred, since we now have two  $\Diamond^-$  operators. However, we can shift the focus of the  $\Diamond^-$  operator to form the axiom:

$$A \sqsubseteq \Diamond^- ((\exists \text{developsFrom} . B) \sqcap A') \quad (4.19)$$

which would align the two time points correctly, but the relation would no longer hold in the present time point, but rather in the previous time point. We give both TR7.1 and TR7.2 a score of ✗ for the same reasons described in TR1. We give TR7.3 a score of ✓. Simple changes such as changes in classes can be easily captured in  $\text{LTL}_{\mathcal{AC}}$  as shown above, even though the relation itself relies heavily again on past operators. We give TR7 a score of ✓. If we accept that the state classes are the best we can do at modelling existence constraints, then the logic may be suitable.

**TR8 The ability to model the time constraints of the following Allen's interval relations between occurrents:** *before'*, *during*, *before*, *meets*, *meets'*

**TR8.1 Capture the temporal constraints intended by Allen's relations**

We now go on to show how well  $\text{LTL}_{\mathcal{AC}}$  can capture Allen's interval relations, seen as time features between occurrents, using their corresponding TRO relations. *before'* - *causally downstream of* does not have a definition, but its inverse, *causally upstream of*, does:

*"p is causally upstream of q if and only if p precedes q and p and q are linked in a causal chain."*

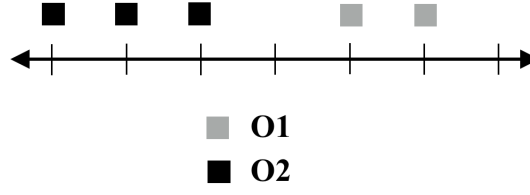


Figure 4.6: An illustration of the *causally downstream of* relation. In this example the occurrent  $O_1$  appears causally downstream of the occurrent  $O_2$  simulating Allen’s interval relation *after*

This definition also includes the relation *precedes*, which is defined as follows:

*“ $x$  precedes  $y$  if and only if the time point at which  $x$  ends is before or equivalent to the time point at which  $y$  starts. Formally  $x$  precedes  $y$  iff  $\omega(x) \leq \alpha(y)$ , where  $\alpha$  is a function that maps a process to a start point, and  $\omega$  is a function that maps a process to an end point”.*

Consider Figure 4.6. Since we are focusing on the temporal aspects of the TRO relations, we can view *causally downstream of* as being the temporal inverse of *precedes*. If  $O_1$  *causally downstream of*  $O_2$ , then the time point at which  $O_1$  starts must be equal to or after the time point in which  $O_2$  ends. We choose to model these relations using the states classes in constant domains (due to needing to model relations between *non existent* entities). We model these constraints as follows:

$$O_1 \sqcap \text{Active} \sqsubseteq \Diamond^-((\text{Before} \sqcap (\exists X.(O_2 \sqcap \text{After}))) \sqcup (\text{Active} \sqcap \bigcirc^-(\text{Before}) \sqcap (\exists X.(O_2 \sqcap \text{After} \sqcap \bigcirc^-(\text{Active})))))) \quad (4.20)$$

In (4.20), we enforce that any active instance of  $O_1$  either had an  $X$  relation to an instance of  $O_2$  that was in the *After* state whilst itself was in the *Before* state, or it has an  $X$  relation to an instance of  $O_2$  that has transitioned into the *After* state at the current time point, which is the same time point in which itself has transitioned into the *Active* state, from the *Before* state. The role of the disjunct is to capture the *or* in the *before or equal to*. We make use of the existential to enforce that there was at least one  $O_2$  for which  $O_1$  follows. However, if the constraint was a global constraint, i.e., all  $O_1$ s must come after  $O_2$ s, this cannot be captured. Notice that we also have to introduce an auxiliary relation,  $X$ , which has no temporal meaning, but merely used to assist with the

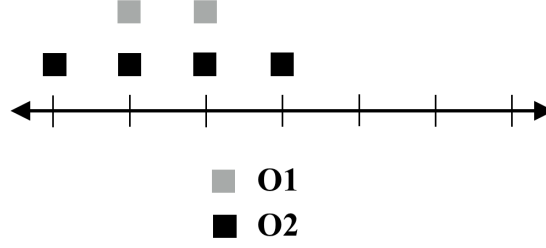


Figure 4.7: An illustration of the *happens during* relation. In this example the occurrent  $O1$  happens during the occurrent  $O2$ , simulating Allen’s interval relation *during*.

constraints on the time points. We also are presented with the same problem as before when considering what should and should not be allowed to be stated with occurrents not in an active state. However, we know varying domains would not be suitable here, so the state constraints are the best that we can achieve.

*happens during* is defined as:

“ $X$  happens during  $Y$  iff: ( $start(X)$  before or simultaneous with  $start(Y)$ ) AND ( $end(X)$  before or simultaneous with  $end(Y)$ )”.

Consider Figure 4.7. This is equivalent to Allen’s interval relation *during*, hence the use of the time feature *time:during*. Using similar techniques as the previous relation, we can model  $O_1$  happens during  $O_2$  as follows:

$$O_1 \sqcap Active \sqsubseteq \Diamond^-((Before \sqcap \bigcirc(Active \sqcap \exists X.(Active \sqcap O_2)\mathcal{U}After)) \quad (4.21)$$

(4.21) states that any *Active*  $O_1$  was sometime in the past in the *Before* state, and at the time it became *Active*, it had an  $X$  relation to an *Active* instance of  $O_2$  at every time point until it eventually becomes inactive. Again, we have to introduce an auxiliary role to temporally relate the two occurrents in time. The axiom itself is not truly faithful because of an issue with the  $X$  relation to  $O_2$ . Since we potentially have many existential relations between the two elements, it is possible that a model of the axiom could relate the first  $X$  successor of  $O_1$  to an element that is different to the second  $X$  successor at each time point. Using rigid roles would be the obvious first thought to overcome this problem. If we declare  $X$  to be rigid, then we can ensure that the successors would be the same over all time points, but this is an over-enforcement. If the two individuals were related at all time points then it defeats the purpose of the  $X$  relation. Ideally,

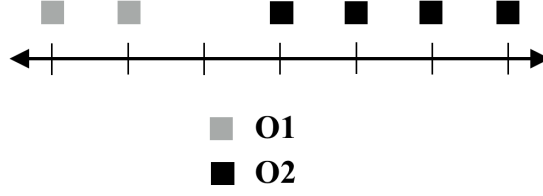


Figure 4.8: An illustration of the *precedes* relation. In this example the occurrent  $O_1$  precedes the occurrent  $O_2$ , simulating Allen's interval relation *before*.

they should only be related during the activity of  $O_1$  and not in any other time points, as depicted in Figure 4.7.

*precedes* is a relation between two occurrents defined as

"*x precedes y if and only if the time point at which x ends is before or equivalent to the time point at which y starts*".

Consider Figure 4.8. We first direct our attention towards the *before* case (the *equivalent* case is seen later in *immediately precedes*). We can model the relation  $O_1$  *precedes*  $O_2$  as follows:

$$O_1 \sqcap \text{Active} \sqsubseteq \Diamond(\text{After} \sqcap (\exists X.(\text{Before} \sqcap \Diamond(\text{Active} \sqcap O_2)))) \quad (4.22)$$

The axiom states that for any active  $O_1$  occurrent, eventually they will be in a *After* state, and there will later be an  $O_2$  for which  $O_1$  is X-related, who will be in a *Before* state but eventually *Active*. Again, we use the auxiliary role  $X$  to make a connection between the two elements to enforce the temporal relation between them.

*immediately preceded by* is similar to *causally downstream of* however the two occurrents must *meet*. Consider the following definition:

"*X immediately preceded by Y iff: start(X) simultaneous with end(Y)*".

In this case we need to ensure that the end time of  $X$  is the same as the start time of  $Y$ . Consider Figure 4.9. We can model  $O_1$  *immediately preceded by*  $O_2$  as follows:

$$O_1 \sqcap \text{Active} \sqsubseteq \Diamond^-(\text{Before} \sqcap \bigcirc(\text{Active}) \sqcap (\exists X.(O_2 \sqcap \text{Active} \sqcap \bigcirc(\text{After})))) \quad (4.23)$$

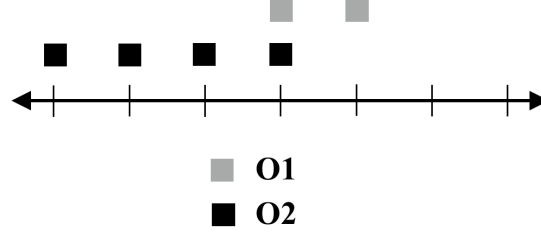


Figure 4.9: An illustration of the *immediately preceded by* relation. In this example the occurrent  $O_1$  is immediately preceded by the occurrent  $O_2$ , simulating Allen's interval relation *meets'*.

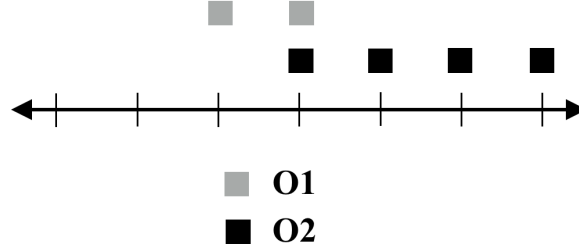


Figure 4.10: An illustration of the *immediately causally upstream of* relation. In this example the occurrent  $O_1$  is immediately causally upstream of the occurrent  $O_2$ , simulating Allen's interval relation *meets*.

(4.23) states that any *Active*  $O_1$ , at the time point which it became *Active*, is  $X$  related to an instance of  $O_2$  who has at the same time point transitioned into the *After* state.

*immediately causally upstream of* is defined as:

“ $p$  is immediately causally upstream of  $q$  iff both (a)  $p$  immediately precedes  $q$  and (b)  $p$  is causally upstream of  $q$ . In addition, the output of  $p$  must be an input of  $q$ ”

*immediately precedes* is defined as:

$X$  immediately precedes  $Y$  iff:  $\text{end}(X)$  simultaneous with  $\text{start}(Y)$

for which the relation is a sub property of. Consider Figure 4.10. We can model  $O_1$  immediately causally upstream of  $O_2$  as follows:

$$O_1 \sqcap \text{Active} \sqsubseteq \Diamond(\text{After} \sqcap \bigcirc^-(\text{Active}) \sqcap (\exists X.(O_2 \sqcap \text{Active} \sqcap \bigcirc^-(\text{Before})))) \quad (4.24)$$

Note that this axiom is really just the temporal opposite of *immediately preceded by*, which is unsurprising since they are inverses of each other in RO.

With all the *o-o* relations involving Allen's intervals, we observe that we are not defining the time relations themselves, but merely attempting to enforce the temporal constraints inside an axiom. None of the relations actually appear as relations inside the axioms. It would be unwise to use the auxiliary relations as the original relations as they do not represent them. Ideally, it would be better to be able to constrain the temporal nature of the relation itself, as was the intention of Allen's relations. Consider again the relation *precedes*. Ideally, we would like to have the subclass expression of the axiom to be something along the lines of  $\exists \textit{precedes}.O_2 \sqsubseteq \dots$ , and then go on to constrain the temporal relation between  $O_1$  and  $O_2$  in the superclass. However, such an axiom cannot exist. For example, the following axiom

$$O_1 \sqcap \textit{Active} \sqcap \exists \textit{precedes}.O_2 \sqsubseteq \Diamond(\textit{After} \sqcap (\exists X.(O_2 \sqcap \textit{Before} \sqcap \Diamond(\textit{Active})))) \quad (4.25)$$

tries to define the *precedes* relation by stating that if  $O_1$  has a *precedes* relation to  $O_2$  then the  $O_2$  must start after the death of  $O_1$ . However, the filler of the *precedes* relation in the subclass expression does not necessarily have to be the filler of the  $X$  relation in the superclass expression. We believe the first modelling attempt is better suited.

We give both TR8.1 and TR8 a score of ✓. We can model the constraints of Allen's relations, but only in a simple way. We cannot hope to gain any useful entailments or use the relations in anyway other than what is specified in the axioms.

## TR9 The ability to model relations between continuants and occurrents

### TR9.1 Be able to make relations between the two types of distinct elements

As we have now seen how to model relations between continuants and occurrents exclusively, we now move on to modelling relations between the two. Due to the two types of entities being represented in roughly the same way (all are mapped to standard DL domain elements), we can model relations between them in a similar way as in each exclusive case. We use the relation *input of* to illustrate

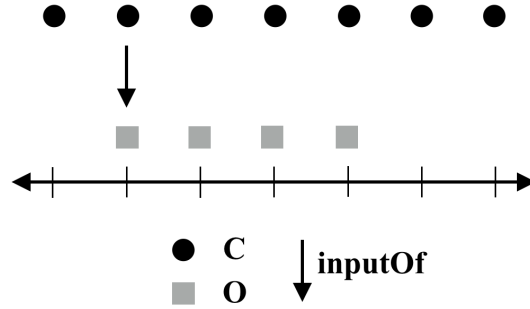


Figure 4.11: An illustration of the relation *input of*, showing a continuant  $C$  being the input of an occurrent  $O$  at a single time point.

this. Unfortunately, this relation does not have a definition in TRO, therefore we use the definition of its super property, *participates in*, to aid in the modelling:

*a relation between a continuant and a process, in which the continuant is somehow involved in the process.*

Although the definition is not very detailed, we believe the involvement is that the continuant is the input to the occurrent, upon the occurrent's activation or the start of its existence. Consider Figure 4.11. We attempt to represent *C input of O at t* as follows:

$$C \sqsubseteq \exists \text{inputOf}. (O \sqcap \neg \bigcirc^- \top) \quad (4.26)$$

The input of relation between the two entities should take place at a single time point (*time:same*) and it would be fair to assume that in some cases it should only hold at a *certain* time point. Looking again at Figure 4.11, it would not be correct to assume that  $C$  was continuously the input of  $O$  at every time point along the time line... The axiom is very similar to that expressed in TR5's *aligned with* example. It states that any instance of  $C$  must have an *inputOf* relation to an instance of an  $O$  that was previously non existent. The problem we face again is that of quantification. As it stands, any instance of  $C$  at any time point would always have the relation. But this is not what may be needed. Ideally, we would like to say something along the lines of *C is the input of O at time t, and only at time t*. This is not always the case however, and this is just one of the relations that hold this annotation. Others exist where this type of modelling is perfectly suitable. We give TR9.1 a ✓. Making relations between the two types

of elements is as easy as each exclusive case - it is not more different than any other type of relation. We give TR9 a score of ✓ also since we can model relations between the two types of elements, but not faithfully in some cases, due to the qualitative nature of the logic.

**TR10 The ability to model relations between continuants and occurrents over past, future, and present time points, each including the continuant's state constraints**

TR10.1 Relate the two individuals at a single time point (time:same)

TR10.2 Relate the two individuals at a present and past time point (time:past)

TR10.3 Relate the two individuals at a present and future time point (time:future)

TR10.4 Correctly model the continuant coming into and out of existence (birth, death)

We now move on to modelling relations between the two domain elements over single and multiple time points, also showing their domain constraints. We use the relations *existence starts during*, *existence starts during or after*, *existence ends at point* and *existence ends during or before*. *existence starts during* is defined as

*“ $x$  existence starts during  $y$  if and only if the time point at which  $x$  starts is after or equivalent to the time point at which  $y$  starts and before or equivalent to the time point at which  $y$  ends. Formally:  $x$  existence starts during  $y$  iff  $start(x) \geq start(y)$  AND  $start(x) \leq end(y)$ ”.*

Consider Figure 4.12. We can model  $C$  *existence starts during*  $O$  using the varying domain approach as follows:

$$C \sqsubseteq \Diamond^-(\exists X.(O) \sqcap \neg \bigcirc^- \top) \quad (4.27)$$

Here we specify that any instance of  $C$  at an earlier time point had an  $X$  relation to an instance of  $O$  and at the previous time was non existent. The fact that a relation exists between an  $C$  and  $O$  immediately when  $C$  came into existence



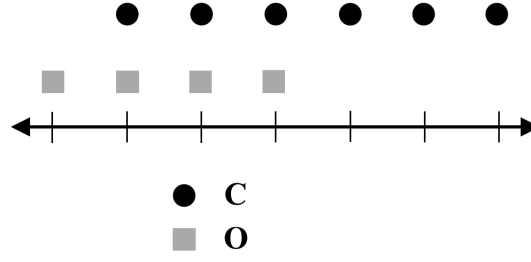


Figure 4.12: An illustration of the relation *existence starts during*, showing a continuant  $C$ 's existence starting during the life span of an occurrent  $O$ .

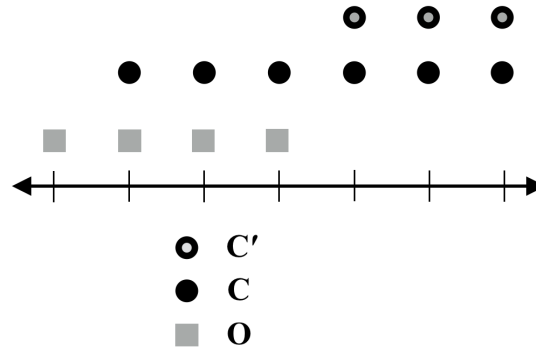


Figure 4.13: An illustration of the relation *existence starts during or after*, showing two continuants  $C$  and  $C'$  whose existence starts during and after the life span of an occurrent  $O$  respectively.

ensures that  $C$  started during  $O$ 's existence. We again have to specify another auxiliary relation to align the time points correctly. The corresponding constant domain approach could be the axiom:

$$C \sqcap \text{Active} \sqcap (\bigcirc^- \text{Before}) \sqsubseteq \exists X.(O \sqcap \text{Active}) \quad (4.28)$$

We can follow similar design patterns to model the relation *existence starts during or after*, shown in Figure 4.13 and defined as:

*“ $x$  existence starts during or after  $y$  if and only if the time point at which  $x$  starts is after or equivalent to the time point at which  $y$  starts. Formally:  $x$  existence starts during or after  $y$  iff  $\text{start}(x) \geq \text{start}(y)$ .”*

However, we have to use the constant domain in this approach. If the existence of  $C$  starts after  $O$ , then  $C$  and  $O$  will not exist at the same time, so no relations

can be made between the two. We can represent  $C$  existence starts during or after  $O$  as follows:

$$C \sqcap Active \sqcap \bigcirc^-(Before) \sqsubseteq (\exists X.(O \sqcap (Active \sqcup After))) \quad (4.29)$$

This axiom states that any instance of  $C$  at the time point that it transitions into the *Active* state must have an  $X$  relation to either an *Active* or *After* state instance of  $O$ . The *during or after* is captured in the disjunct in this case.

*existence ends at point*, is defined as follows:

*“ $x$  existence ends at point  $y$  if and only if the time point at which  $x$  ends is equivalent to the time point at which  $y$  ends”.*

An example of this relation is shown in Figure 4.14. We can model this as follows:

$$C \sqsubseteq \Diamond(\neg \bigcirc \top \sqcap \exists X.(O \sqcap \neg \bigcirc \top)) \quad (4.30)$$

Here we enforce that when an instance of  $C$  eventually ceases to exist, at the same time point it has a relation to an instance of  $O$  that also ceases to exist at the same time point, aligned via the relation  $X$ .

Finally, *existence ends during or before* is defined as

*“ $x$  existence ends during or before  $y$  if and only if the time point at which  $x$  ends is before or equivalent to the time point at which  $y$  ends.”.*

An example is shown in Figure 4.14 and represented in the following axiom:

$$C \sqcap Active \sqcap (\bigcirc After) \sqsubseteq \exists X.(O_1 \sqcap (Active \sqcup After))) \quad (4.31)$$

Interestingly, since each relation has domain and range constraints, we can capture the *time:future* and *time:past* features by embedding them into the state information, which proves to work quite well. We again see the need for the use of past operators. We give TR10.1 a score of ✓, TR10.2 a ✓ TR10.3 a ✓, and also 10.4 a ✗. The ✗ is due to the fact that in some cases we have to switch between representations as one is not sufficient. TR10 as a whole receives a score of ✓.

**TR11 The ability to model same time rigid relations between continuants and occurrents**

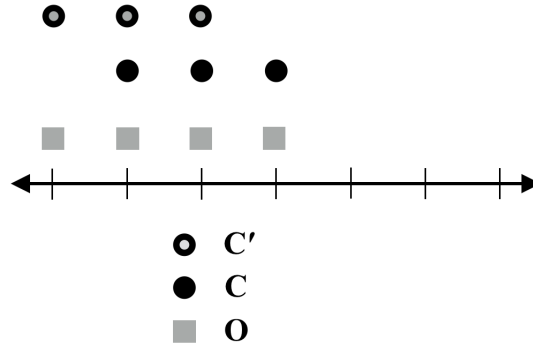


Figure 4.14: An illustration of relations *existence ends during or before* and *existence ends at point*. Both  $C$  and  $C'$ 's existence end during or before  $O$ , whereas only  $C'$ 's existence ends at the point of  $O$ 's existence.

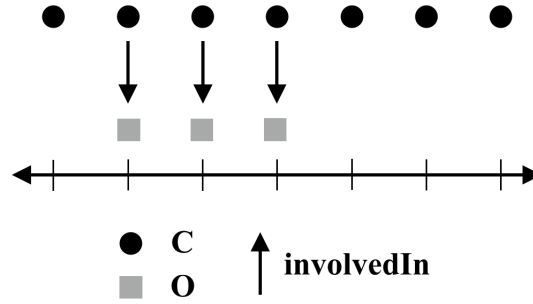


Figure 4.15: An illustration of the relation *involved in*, showing a continuant  $C$  being involved in an occurrent  $O$  for multiple consecutive time points over a fixed duration over the life span of  $O$ .

**TR11.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR11.2 The relation can have a duration specified**

*involved in* is defined as

*“a relation between a continuant and a process, in which the continuant is somehow involved in the process”*

This relation is declared to be rigid since the occurrent may span over multiple time points, or even have many temporal parts, in which the continuant should be involved in. Unfortunately, we run into the same problems as in the continuant case when wanting to enforce rigid relations. We fail to specify a duration in which we want the rigid relation to hold, or even enforce that the two elements are the same over multiple time points without introducing rigid roles, which are

undesired. However in the continuant-occurrent case, there is also another possibility to consider since an occurrent may have multiple temporal parts. Suppose a continuant  $C$  is involved in an occurrent  $O$ , which has two temporal parts,  $O_1, O_2$ , where each lasts a single time point and are in the order given:  $O_1 < O_2$ . We can represent the temporal parts in the same way as we did in TR2. We see only one way to model  $C$  involved in  $O_1$  and  $O_2$ :

$$C \sqsubseteq \exists \text{involvedIn}.O_1 \sqcap \bigcirc (\exists \text{involvedIn}.O_2) \quad (4.32)$$

This seems adequate since we ensure that  $C$  is involved in each temporal part of  $O$  in the correct order, but we cannot ensure that temporal part of  $O$  is part of the same  $O$ . If  $O$  had no temporal parts, then we run into the same obvious problems of rigidity, as in TR3 and TR4, where the only hope seems to be to introduce rigid roles, which has negative effects with both varying domains and complexity. As with TR3 and TR4, we give TR11.1 a score of ✓, since when we do introduce rigid roles, we can capture same time relations between the same individuals consecutively, but TR11.2 receives a ✗ since no duration can be captured with the rigid relations. For the same reason we give TR11 a score of ✗ as a whole.

**TR12 The ability to allow for multiple future time lines where relations may or may not hold**

**TR12.1 The ability to express multiple futures where relations may hold in one future and not in others**

Capable of is defined as

*“A relation between a material entity (such as a cell) and a process, in which the material entity has the ability to carry out the process”.*

Notice that the relation that may or may not hold is the *carrying out* relation, hence the reason for the *possible* feature. From the modelling and temporal perspective it may be interesting to know the consequences of the relation holding, similarly for it not holding. For the first time we introduce  $\text{CTL}_{ACC}$  in our modelling evaluation. We can model  $C$  capable of  $O$  as follows:

$$C \sqsubseteq \mathbf{E} \bigcirc (\Diamond (\exists \text{carriesOut}(O))) \sqcap \mathbf{E} \Box (\neg (\exists \text{carriesOut}.O)) \quad (4.33)$$

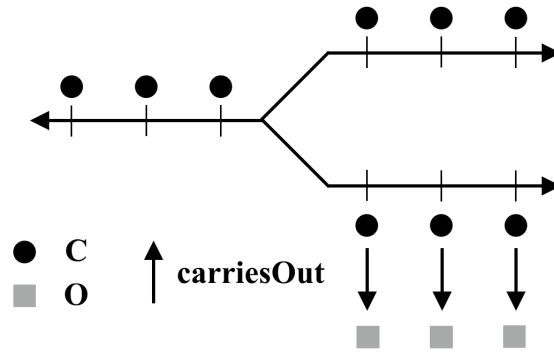


Figure 4.16: An illustration of the relation *carries out*. In this example, there exists a time line where the continuant *C* carries out the occurrent *O* and another time line where this does not happen.

Here we specify the two possible futures in each of the super classes conjunct operands. The first operand states that there is a future time line in which *C* eventually carries out the process *O*. The second operand handles the case where *C* never carries out the process, using the  $\Box$  operator and a negated role restriction. The axiom seems to capture what we intended, we have two possible futures, one where the relation holds, and one where it does not. We give TR12.1 a score of ✓ and TR12 ✓✓ as we capture exactly what was intended.

#### TR13 The ability to model relations between occurrents and continuants

TR13.1 Be able to make relations between the two types of distinct elements

Similarly to TR9, we can model relations between occurrents and continuants in the same way. There is nothing special in the order of the domain and range types, and therefore face the same problems as in the TR9 case. TR13.1 is given ✓ and TR13 is given ✓.

#### TR14 The ability to model same time rigid relations between occurrents and continuants

TR14.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR14.2 The relation can have a duration specified

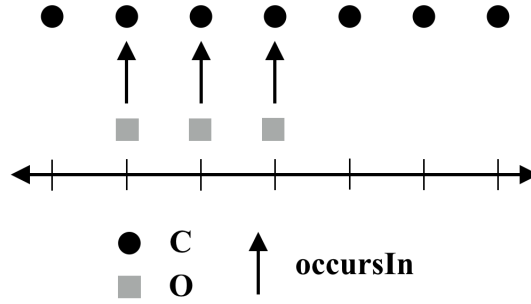


Figure 4.17: An illustration of the relation *occurs in*. In this example, an occur-rent *O* occurs in an continuant *C* during its entire life span.

TR14 is similar to TR11 - the domain and range order does not change what we can express due to the fact that they are represented by the same type of elements. We illustrate this with the relation *occurs in* show in Figure 4.17. *occurs in* is defined as follows:

*“a relation between a process and an independent continuant, in which the process takes place entirely within the independent continuant”.*

The main difference between the relations in TR11, is that here the occurrent both starts and ends (comes into and out of existence) inside the continuant. Without knowing the duration of *O*, we can only represent *O occurs in C* as follows:

$$O \sqsubseteq \exists \text{occursIn}.C \quad (4.34)$$

without losing the identity of *C*. As we have seen many times before, this is not an accurate representation - if *O* lasts more than one time point, the relation will not be correctly rigid, having multiple existential restrictions will also result in incorrect rigidity, and even declaring *occurs in* as having the *rigid* property characteristic is too strong a statement. We give TR14.1 a score of ✓, but TR14.2 receives a ✗. As before, TR14 receives a score of ✗ as a whole.

**R15 Any extension should be decidable and of suitably low complexity, have implementations or at least decision procedures for the most common reasoning problems and be readily available for use**

LTL<sub>DL</sub> and CTL<sub>DL</sub> combinations have had a lot of of attention in recent years,

with research mostly going into the complexity theory area [LWZ08]. Most current work has been spent on testing the bounds of the logic’s decidability limits when certain *desired* features are included such as rigid roles [GJS15b, GJS15a]. Little to no work has gone into the practical side of things however, such as reasoner development or work similar to what has been presented so far in this thesis. We first focus on the suitability of the logic with the expressivity required. From Chapter 2, we know that when compared to either  $\mathcal{ALC}$  or LTL,  $\text{LTL}_{\mathcal{ALC}}$  is computationally obedient. Its complexity of reasoning is no worse than either other logic. There are three, for want of a better word, *add-ons* that would be required of  $\text{LTL}_{\mathcal{ALC}}$  in order to meet some of the requirements that we saw whilst evaluating against TR1-TR14. These include, varying domains, past (or inverse) operators and rigid roles. Firstly, it is known that there is an easy polynomial reduction from varying, expanding and decreasing domains into constant domains for satisfiability in modal logics [WZ98a], so switching from constant to varying domains will not have a negative effect on complexity. We also note that varying domains require negation to interact with, and are beyond the scope of lightweight DLs such as  $\mathcal{EL}$ . So any complexity results for TLs combined with lightweight DLs such as  $\mathcal{EL}$  become irrelevant in this case.

Secondly, past operators were needed in almost all of the TRs. Adding past operators usually do not increase the complexity of TLs and they are known to usually increase succinctness [Mar03, LPZ85, LMS02]. Results are known for LTL, and decidability results exist for the more expressive  $\mathcal{DLR}_{\mathcal{US}}$  [AFWZ02].

Rigid roles generally have a negative effect on complexity in TDLs unless severe action is taken. It seems like this add-on has been given the most attention in recent years. From Chapter 2 we see that adding rigid roles usually leads to undecidability, even in fragments of  $\text{LTL}_{DL}$  and  $\text{CTL}_{DL}$ , unless restrictions are made, usually on a TBox level. For example, empty TBoxes, or acyclic ones. Restricting our knowledge bases in this way would be too much of a loss for the ontologies in question and does not seem a viable option, and rigid roles seem impossible to be incorporated.

From a practical viewpoint, no current work has gone into realising either  $\text{LTL}_{DL}$  or  $\text{CTL}_{DL}$  as a next step for temporalising OWL. From our research, no attempts have been made to create reasoners, or a language for ontology developers to use to start creating ontologies or and see how they could benefit from a temporal extension. It seems everything has been focussed mainly on the

theoretical aspect.

Although  $LTL_{ACC}$  and  $CTL_{ACC}$  have some features that are theoretically suitable, they have no practical solutions, and we therefore give R15 a score of  $\times$ .

### 4.1.1 Summary

We can see clearly that  $LTL_{ACC}$  and  $CTL_{ACC}$  work reasonably well with modelling some of the temporal requirements of bio-health ontologies but are far from a suitable representation. Only two TRs stood out as being closely faithful, TR1 and TR12. The reason  $LTL_{ACC}$  performed so well here was due to its possible world semantics, and the rigid interpretation it has on its individuals. In fact, many of the other requirements that scored well ( $\checkmark$ ) was primarily due to this feature. For example TR2, the modelling of occurrents was partially suitable, due to the possible world semantics, but lost points on being able to specify duration easily and other occurrent requirements. TR12 showed that  $CTL_{ACC}$  was sufficient to model what was required w.r.t the possible futures, and we were not limited in what we could express. Another useful feature was the option for varying domains. In some cases it worked well, and was a suitable option, but more in most cases it proved to be a problem, when relations needed to exist between entities that could not exist at the same time.

As explained in R15, the logic easily becomes undecidable even in the most simplest of cases when considering rigid roles. We also saw quite severe problems when wanting to quantify over the time line, to be able to model relations only happening at single time points.

By far the main problem that  $LTL_{ACC}$  suffered from was the inability to model rigidity, or even finite rigidity. This requirement, and its corresponding features are the most prevalent in the corpus, and  $LTL_{ACC}$  failed to capture what was required. This is also important to consider when thinking about designing another logic or an extension of  $LTL_{DL}$  to be able to handle this. There was also the problem of the large size of the axioms needed to specify quite simple properties, such as a duration of an occurrent. We believe this is due to the qualitative nature of the syntax, specifically the way in which we can interact with the time line using the temporal operators. Also, in standard  $LTL_{ACC}$  (and  $CTL_{ACC}$ ) past operators are not present. Nearly all examples actually needed the use of both past time operators and the usual future and present ones also. Although introducing past



time operators does usually not have a negative effect on the complexity of reasoning, they are not available in the logic we were evaluating. Most of the current work has gone only into  $LTL_{\mathcal{ALC}}$  and not its inverse.

Clearly  $LTL_{\mathcal{ALC}}$  and  $CTL_{\mathcal{ALC}}$  are not suitable enough to model bio-health ontologies. We can however take away a positive point when possibly considering an extension or new logic: having a possible world semantics along a time line with rigid individuals captures some of the most important aspects of biological entities and should definitely be considered.

## 4.2 $\mathcal{ALC}(\mathcal{D})$ Evaluation

Using concrete domains as an extension to DLs allows us to relate standard DL elements of the concrete domain via *concrete features*. Recall from Chapter 2, a concrete feature is an abstract (and thus functional) relation that relates DL elements to elements in a concrete domain. The intuition is that if we can relate elements to a representation of a time point in the concrete domain, then we should be able to effectively make statements about elements at particular moments in time, as well as make other constraints on both the time points and the elements, specific to the TRs. During this evaluation, we use  $\mathbb{Z}$  with the standard binary predicates  $\{\geq, >, =, <, \leq\}$  as our concrete domain  $\mathcal{D}$ . In some cases in our evaluation, we will also use the unary and binary predicate  $=_n$  and  $+_n$ .  $=_n$  is a unary predicate that holds true only on the input given if it equals  $n$ . For example,  $=_4$  holds true only for 4.  $+_n$  is a binary predicate that holds true only if the second element is exactly  $n$  greater than the first. For example,  $+_4(5, 9)$  is true, whereas  $+_4(5, 10)$  is false. We combine this concrete domain  $\mathcal{D}$  with the standard DL  $\mathcal{ALC}$ , as in the LTL case, resulting in the logic  $\mathcal{ALC}(\mathcal{D})$ .

### TR1 The ability to model continuants

#### TR1.1 Continuants must be traceable through time

#### TR1.2 Continuants must maintain their identity through time

#### TR1.3 Continuants must be able to undergo change

Unlike standard DLs, we now have two types of distinct domain elements; standard DL domain elements and also numbers in  $\mathbb{Z}$ , for which we can relate the two via means of new relations called concrete features. When modelling continuants,

unlike in  $\text{LTL}_{\mathcal{AC}}$  where there are multiple rigid instances of an element indexed by  $n \in \mathbb{N}$  we still have only a static environment, even in the presence of the concrete domain, hence we only ever have one instance of an element. However, we can link elements (and their properties, explained later) to elements in the concrete domain. As before, we create a class *Continuant* to stand as a representation for all continuants in the DL domain. We consider first the three states of continuants, *Before*, *Active* and *After*. Instead of defining classes for each, we can introduce concrete features to account for when and how an individual can *transition* between these states. Let *startTime* and *endTime* be two new concrete features. We constrain the *Continuant* class as follows:

$$\text{Continuant} \sqsubseteq \exists_{>}(\text{endTime}, \text{startTime}) \quad (4.35)$$

Here we enforce that any *Continuant* must have both a start and end time where the end time must be greater than the start time, i.e., modelling some form of its *life span*. Since they are concrete features, and therefore functional, we know that each continuant can only have exactly one start and end time, avoiding any potential conflicts, such as the problem of coming into and out of existence multiple times. Even in the event where continuants may not have start and end times, and may be considered always active, then we may prefer the universal version of the axiom:

$$\text{Continuant} \sqsubseteq \forall_{>}(\text{endTime}, \text{startTime}) \quad (4.36)$$

that states that, *if* a continuant has a start and end time, then the restriction must hold. If we knew the exact time points in which a continuant (at a concept level) *C* comes into and out of existence then we can specify this as follows:

$$C \sqsubseteq \exists_{=i}(\text{startTime}) \sqcap \exists_{=j}(\text{endTime}) \quad (4.37)$$

where  $i, j \in \mathbb{Z}$  are the known start and end times respectively.

The notion of a continuant having a start and end time is difficult to relate to the notion of *existence* in a static environment. Since there is only one world of evaluation in which all entities exist in, then having a start and end time seems to be a static approach as opposed to a dynamic approach. If we could add constraints that relations and class membership can only be made during these

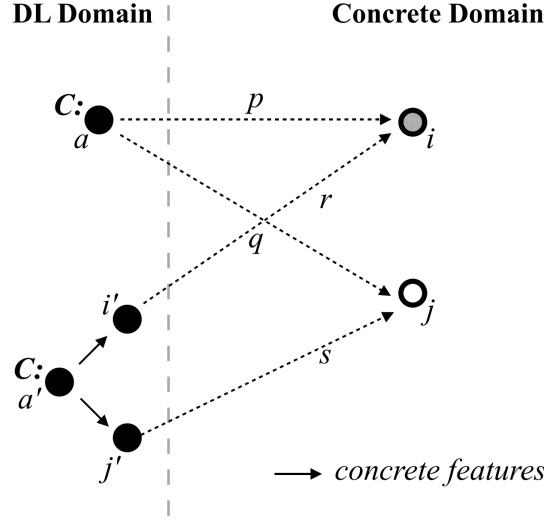


Figure 4.18: An example of modelling continuants in  $\mathcal{ALC}(\mathcal{D})$ . An element  $a$  being an instance of a continuant  $C$  is related to do different time points in the first example. Then reification is used in the second example to show how it can be split in to two *temporal parts*.

existence times, then it would make more sense, but we see later on that this is difficult to capture.

We do however run into problems regarding the identity constraints of individuals. Continuants are defined as those things that endure through time, that maintain their identity, have no temporal parts, whilst being subject to change. Maintaining identity through time is difficult to envision in a static environment. Because of the one single world of evaluation, of course all elements maintain their identity. The problems arise when we consider referring to a continuant at different time points. Suppose we wanted to model some type of change of a continuant  $C$ , by making a particular statement about  $C$  at time point  $i$  and another statement about  $C$  at time point  $j$ . Consider Figure 4.18. Distinguishing between individual  $a$  (representing  $C$ ) at time points  $i$  and  $j$  is impossible since there is only one individual  $a$  - we cannot make statements about  $a$  at  $i$  without also making the same statements for  $a$  at  $j$ . The identity of  $a$  at  $i$  should be identified by the binary relation  $p(a, i)$ , similarly for  $a$  at  $j$  by the binary relation  $q(a, j)$ . But in OWL, we cannot make statements or constraints on binary relations (ternary relations are not allowed). Instead we have to adopt the use of reification, where we introduce new individuals to act as a representation of a binary relation. This is represented in the same Figure 4.18.  $a'$  is a representation

of “*a during i*” and “*a during j*”, and is related to two reified individuals,  $i'$  and  $j'$ , which have the original concrete features related to them. Since they are now individuals, we can make statements about them, so  $i'$  and  $j'$  act as representations of the original relations. Any changes that we wanted to model can be made on the new individuals since they technically stand in place for different temporal parts of the same individual. This is not a new approach<sup>1</sup> [ASP09]. As can be seen, we have lost the identity of the original elements and any direct entailments on the element involving any change along the way.

When considering tracing continuants through time, if no reification is needed, then since we are in a static environment, no tracing needs to be done. Otherwise, many individuals may be representing the same continuant at different time points, and the tracing in time will be trying to link together each individual in a structured way that conforms to some representation of time line, for example through concrete feature chains. In the example above, to trace the concept  $C$  through time, then we need to be able to refer to each temporal part of  $C$ . We could first represent the example in the following axiom:

$$C \sqsubseteq \exists hasPart_1.C_1 \sqcap \exists hasPart_2.C_2 \quad (4.38)$$

(4.38) states that  $C$  has two parts,  $C_1$  and  $C_2$ . To then indicate that the  $C_1$  part comes before the  $C_2$  part, we can take advantage of the fact that each part is a continuant and thus has start and end times. If we consider *hasPart1* and *hasPart2* to both be *abstract features* (functional relations are allowed to be used in feature chains) we can specify this as follows:

$$C \sqsubseteq \exists_{\geq} (hasPart2 \circ startTime, hasPart1 \circ endTime) \quad (4.39)$$

which states that its  $C_1$  successor must end before its  $C_2$  successor starts. We can trace a continuant through time in this way, by using feature chains to link together its temporal parts.

We give TR1.1 a score of **X**. Although we can effectively trace continuants through time, by introducing temporal parts as shown in the example above, it is not ideal. We have to introduce a new abstract feature and a corresponding feature chain. We also give TR1.2 a score of **X** since when introducing new elements we immediately lose all and any identity constraints on the original

---

<sup>1</sup><http://www.w3.org/TR/swbp-n-aryRelations/>

element. Change can be captured if we accept the representation of the temporal parts of a single element, and we therefore give TR1.3 a ✓. Overall, we give TR1 a ✓. Although it fails to capture the essence of identity and endurance, we can still model them in the usual OWL way, and for the first time we can model continuants at real time points. Although the existence is somewhat tricky to comprehend in a static environment, making correct constraints about their start and end times is possible.

## TR2 The ability to model occurrents

### TR2.1 Occurrents can have limited phases of existence

### TR2.2 Occurrents can have temporal parts

Recall that occurrents are described as having temporal parts, limited phases of existence, and are unable to undergo change. The start and end time concrete features introduced in TR1 play well for occurrents. As before we introduce a class *Occurrent* to represent all occurrents in the DL domain, which is disjoint with the *Continuant* class.

We first show how we can model duration. Specifying duration can be achieved with the predicate  $+n$ . We can specify a duration of length 3 for an occurrent  $O$  as follows:

$$O \sqsubseteq \exists_{+2}(\text{endTime}, \text{startTime}) \quad (4.40)$$

This specifies that the end time of  $O$  must be exactly 2 larger than its start point, capturing a duration of length 3.

Occurrents are also described as having temporal parts. Consider  $O$  having two temporal parts  $O_1, O_2$ , where  $O_1$  lasts 1 time point and  $O_2$  lasts 2 time points, spanning over the entire life span of  $O$ . To enforce the correct sequence along with the correct alignment of  $O$ 's temporal phases, we are forced to introduce several abstract features: *hasTemporalPart1* and *hasTemporalPart2* as we did in

TR1:

$$O \sqsubseteq \exists hasTemporalPart1.O_1 \sqcap hasTemporalPart2.O_2 \quad (4.41)$$

$$\begin{aligned} O \sqsubseteq \exists_{=}(startTime, hasTemporalPart1 \circ startTime) \\ \sqcap \exists_{=}(endTime, hasTemporalPart2 \circ endTime) \\ \sqcap \exists_{=}(hasTemporalPart1 \circ endTime, hasTemporalPart2 \circ startTime) \end{aligned} \quad (4.42)$$

$$O_1 \sqsubseteq \exists_{=}(endTime, startTime) \quad (4.43)$$

$$O_2 \sqsubseteq \exists_{+1}(endTime, startTime) \quad (4.44)$$

(4.42) specifies that  $O$ 's start time is equal to its first temporal part's start time, whose end time is equal  $O$ 's second temporal parts start time, whose end time is equal to  $O$ 's end time. (4.43) and (4.44) specify the obvious constraints on both  $O_1$  and  $O_2$ . From these axioms we also implicitly constrain the correct duration of  $O$  without explicitly stating it.

We have to introduce an abstract feature for each temporal part and we use predicate  $=$  to align the temporal parts correctly. And of course, if specific time points are known for any occurrent we can specify this with the  $=_n$  predicate. If the identity of each temporal part was required to remain the same, then we would be faced with a similar situation as in the continuant case, where we would be forced to reify. However, this may not always be an issue since an occurrent's parts can often be seen as sub processes which certainly don't maintain the identity of its parent.

A type of duration can be captured, relatively easily, so we give TR2.1 a score of  $\checkmark$ . Temporal parts of occurrents can also be easily captured so we give TR2.2 a  $\checkmark$  overall. Again, we can see that we can quantify real time points if we know exactly when an occurrent takes place, so we give TR2 a score of  $\checkmark\checkmark$  overall.

### TR3 The ability to model same time rigid relations between occurrents or continuants

TR3.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR3.2 The relation can have a duration specified

We use *part of* again for our evaluation, starting with the continuant case. Our example is  $C_1$  *part of*  $C_2$  for 3 consecutive time points. Our best attempt involves

using reification due to the issues on temporalising a binary relation discussed above. The obvious axiom

$$C_1 \sqsubseteq \exists \text{partOf}.C_2 \quad (4.45)$$

does not suffice since we cannot link the  $\exists \text{partOf}.C_2$  (the relation in question) to a duration, let alone individual time points. We reify as follows:

$$\begin{aligned} C_1 \sqsubseteq \exists \text{hasRigidRelation}.(\exists \text{partOf}.C_2 \\ \sqcap \exists_{+2}(\text{endTime}, \text{startTime})) \end{aligned} \quad (4.46)$$

The axioms states that  $C_1$  has has a rigid relation (a new relation) to an element (a new reified element) that has a part of relation to  $C_2$  with a duration of 3. Since we cannot apply concrete features on the relation itself, we are forced to use reification to introduce a new element standing in for the relation. Although a duration seems to be captured correctly, the biggest issues lie within the *rigid* constraint. Although we are in a static environment, and technically the same elements are related for the desired duration, they are not the elements that should be related. It should be the instances of  $C_1$  and  $C_2$ , but rather the former is actually the reified element. Another problem lies with having multiple rigid relations. Suppose we also wanted the states of each continuant to be aligned, i.e., when the relation holds, both continuants would need to be in an *Active* state. Consider the following axiom

$$C_1 \sqsubseteq \exists \text{hasRigidRelation}(\exists \text{partOf}.(C_2) \sqcap \exists_{+3}(\text{endTime}, \text{startTime})) \sqcap \quad (4.47)$$

$$\exists_{\leq}(\text{startTime}, \text{hasRigidRelation} \circ \text{startTime}) \sqcap \quad (4.48)$$

$$\exists_{\geq}(\text{endTime}, \text{hasRigidRelation} \circ \text{endTime}) \sqcap \quad (4.49)$$

$$\begin{aligned} \exists_{\leq}(\text{hasRigidRelation} \circ \text{partOf} \circ \text{startTime}, \text{hasRigidRelation} \circ \text{startTime}) \sqcap \\ \exists_{\geq}(\text{hasRigidRelation} \circ \text{partOf} \circ \text{endTime}, \text{hasRigidRelation} \circ \text{endTime}) \end{aligned} \quad (4.50)$$

$$\exists_{\geq}(\text{hasRigidRelation} \circ \text{partOf} \circ \text{endTime}, \text{hasRigidRelation} \circ \text{endTime}) \quad (4.51)$$

(4.47) is as before, simply reifying the *part of* relation between  $C_1$  and  $C_2$  and capturing the duration of 3. (4.48) now specifies that the start time of  $C_1$  must be before or equal to when the relation holds and (4.49) specifies that the end time must be equal to or after the relation ends, indicating that  $C_1$  is active throughout

the relation. (4.50) and (4.51) also encode the same activity constraints for  $C_2$ . Although this seems adequate for the relation, a problem lies with the fact that *hasRigidRelation* would have to be abstract, and therefore functional. If  $C_1$  was part of another continuant, say  $C_3$ , then to capture something similar, we would need to introduce another distinct *hasRigidRelation*.

The same results hold for the occurrent case. As they are represented by the same types of elements, we also are forced to model them in the same way, leading to the same problems.

We give TR3.1 a score of ✗, because we have to introduce a new individual to make the relation hold for more than one time point, so we lose the identity of the original individual. Also there is only one relation, representing the whole rigidity, going against TR3.1. We give TR3.2 a ✓ however, since we can capture the duration on the reified relation correctly, even though there is only one relation. For TR3 as a whole we give this a score of ✗. Even though we can encode duration and specific time points which helps in the occurrent case, the relation is not rigid and we lose the important identity constraints.

#### TR4 The ability to model same time rigid relations between continuants

TR4.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR4.2 The relation can have a duration specified

As before we score the whole of TR4 the same as TR3. There is no difference in the modelling of rigidity between occurrents and continuants in  $\mathcal{ALC}(\mathcal{D})$ .

#### TR5 The ability to model same time relations between continuants

TR5.1 The relation must hold at a single time point (time:same)

Modelling same time relations in  $\mathcal{ALC}(\mathcal{D})$  can be done in one of two ways, with reification and without. We use  $C_1$  aligned with  $C_2$  as our running example. Without using reification, we can model this in the standard way as follows:

$$C \sqsubseteq \exists alignedWith.B \quad (4.52)$$



Since we are in a static environment the *time:same* feature is captured trivially, the relation holds in the only time point. Although the relation holds in a single world, there is actually no temporal information regarding the relation. Ideally, we should use reification to model the relation as in the other cases, using a new abstract feature *hasSameTimeRelation*:

$$C_1 \sqsubseteq \exists \text{sameTimeRelation}.(\exists \text{alignedWith}.C_2 \sqcap \exists_{=}(startTime, endTime)) \quad (4.53)$$

Here we specify that  $C_1$  has a relation to a reified individual, that is aligned with  $C_2$  and its start and end time are the same, signifying that the relation holds in a single time point. Using this approach would also allow us to quantify when the relation may also hold, which may be beneficial to ontologies with this information present. Having to use reification for same time relations shows that it is not only rigid relations that suffer with this problem, although it does allow us to quantify over the relation quite easily which is a positive outcome. We give TR5.1 a ✓ since the sub requirement is met. We give TR5 a ✓, since the reification problem arises again and the relation itself is not between the correct individuals.

#### TR6 The ability to model past time relations between continuants

##### TR6.1 The relation must hold between individuals at present and past time points (time:past)

*develops from* is a relation involving two time points  $t > t'$  due to the *time:past* feature, and two continuants  $C_1$  and  $C_2$ . Again we see that the temporal constraints are on the relation itself, rather than the individuals. Ideally the standard reification technique would be the first option due to the nature of the constraint, but differentiating between  $t$  and  $t'$  proves to be difficult. *developsFrom* also has the *dom:changed* feature present in its annotation, but we leave this for TR7, and focus on the time relation here. We can represent  $C_1$  at  $t$  *develops from*  $C_2$  at  $t'$

as follows:

$$(hPR = hasPastRelation)$$

$$C_1 \sqsubseteq \exists hPR. (\exists developsFrom.C_2 \sqcap \exists_{>}(startTime, endTime)) \sqcap \quad (4.54)$$

$$\exists_{=}(hPR \circ startTime, startTime) \sqcap \quad (4.55)$$

$$\exists_{=}(hPR \circ endTime, hPR \circ developsFrom \circ endTime) \quad (4.56)$$

where *hasPastRelation* is a new abstract feature used to relate  $C_1$  to the reified individual representing the relation. (4.54) simply specifies that  $C_1$  is related to a reified individual, standing in for the *developsFrom* relation to  $C_2$ , and its start point is before its end point. (4.55) specifies that the start time of the relation must be equal to the start time of  $C_1$ , to ensure the relation can only hold during  $C_1$ 's existence, and (4.56) specifies a similar constraint for the  $C_2$  individual. The modelling is very similar to that of a rigid relation. We identify two major issues with this modelling. The first is that *developsFrom* has to be an abstract feature and thus be functional. It is often the case that multiple *developsFrom* relations occur in a developmental chain and at any point in the chain, an individual has develops relations to more than one element, if not all of its predecessors. This cannot be allowed since the relation is functional indicating that further reification would be needed to capture this. The second is related to the first in that *hasPastRelation* is also an abstract feature, and if  $C_1$  had  $n$  developsFrom relations, we would need  $n$  new abstract features.

We give TR6.1 a ✓. The modelling is complicated, having to introduce new individuals and new abstract features just to model a past relation, but the time constraint itself can be captured. We give TR6 a score of ✗ as a whole, due to the fact that the relations themselves have to be abstract to meet the temporal constraints we have and maintaining identity is no longer possible.

**TR7 The ability to model the changing states of continuants, specifically their birth, death and other general changing states**

TR7.1 Model a continuant coming into existence (birth)

TR7.2 Model a continuant going out of existence (death)

TR7.3 Model general changes of continuants (change)

*developsFrom* also has the *dom:change* feature in its temporal annotation. Carrying on from the example in TR6, suppose the change was a simple change in

class. The only way to represent this would be to introduce further reified individuals, as in the example in TR1. To consider the *birth* and *death* constraints, we use the relation *child nucleus of*. This relation holds between two elements  $x$  and  $y$  where  $x$  ceases to exist when  $y$  starts to exist. In this example we can get away with not reifying since the time constraints can be captured solely on the elements themselves. We can model  $C_1$  *child nucleus of*  $C_2$  as follows:

$$\begin{aligned} C_1 &\sqsubseteq \exists \text{childNucleusOf}.C_2 \sqcap \\ &\quad \exists_{=}(startTime, \text{childNucleusOf} \circ endTime) \end{aligned} \quad (4.57)$$

In this axiom, *childNucleusOf* is an abstract feature, and we enforce that the existence of  $C_1$  happens immediately when  $C_2$  ceases to exist. If we did want to mention something about the relation itself, then we would be in a similar position as with *developsFrom*, where we would need reify the relation again. The features *dom:birth* and *dom:death* seem to be automatically captured by the concrete features. We give both TR7.1 and TR7.2 a score of ✓, since we are easily capturing their constraints correctly. TR7.3 however would receive a score of ✗. This is due to reification needed to express such a simple change. We give TR7 a score of ✓ overall.

**TR8 The ability to model the time constraints of the following Allen's interval relations between occurrents: *before'*, *during*, *before*, *meets*, *meets'***

**TR8.1 Capture the temporal constraints intended by Allen's relations**

Concrete domains give us a powerful mechanism to model some of the complex features that occurrents require, particularly Allen's interval relations. We begin by explaining the relation *causally downstream of* which has the same time relation as Allen's *before'* relation. If  $O_1$  *causally downstream of*  $O_2$  then  $O_1$  must start after  $O_2$  ends. We can model this with the following axioms:

$$\begin{aligned} (\text{causallyDownStreamOf} &= cDO) \\ \exists cDO.Occurrent &\sqsubseteq \exists_{\geq}(startTime, cDO \circ endTime) \end{aligned} \quad (4.58)$$

$$O_1 \sqsubseteq \exists cDO.O_2 \quad (4.59)$$

The relation only contains the *time:before'* feature, and the temporal constraints are not specific to individual instances or classes. Therefore in (4.58) we provide what can be seen as a definition for the *causally downstream of* relation. (4.58) states that if you have a *causallyDownstreamOf* relation to any occurrent, then your start time must come after that occurrent's end time. Then in (4.59), we simply state that  $O_1$  is causally downstream of  $O_2$ . Since  $O_2$  is an occurrent, we can be assured that the constraint between the two time points are met correctly. The only downside to this way of modelling is that *causallyDownstreamOf* becomes an abstract feature. If we wanted or needed several of these relations, we would have to introduce many distinct abstract features, or otherwise use reification again.

The remaining relations, *happens during* (*during*), *precedes* (*before*), *immediately preceded by* (*meets'*) and *immediately causally upstream of* (*meets*) can be defined as follows:

$$(happensDuring = hD)$$

$$(precedes = p)$$

$$(immediatelyPrecededBy = iPB)$$

$$(immediatelyCausallyUpstreamOf = iCUO)$$

$$\begin{aligned} \exists hD.Occurrent \sqsubseteq \exists_{\geq}(startTime, hD \circ endTime) \sqcap \\ \exists_{\leq}(endTime, hD \circ endTime) \end{aligned} \quad (4.60)$$

$$\exists p.Occurrent \sqsubseteq \exists_{\leq}(endTime, p \circ startTime) \quad (4.61)$$

$$\exists iPB.Occurrent \sqsubseteq \exists_{=}(startTime, iPB \circ endTime) \quad (4.62)$$

$$\exists iCUO.Occurrent \sqsubseteq \exists_{=}(endTime, iCUO \circ startTime) \quad (4.63)$$

$\mathcal{ALC}(\mathcal{D})$  is suited very well for occurrent relations using Allen's interval constraints, especially since we can somewhat define the relations themselves. We give TR8.1 a score of ✓. The only problem we face in all examples is that of the functionality of abstract features. Therefore, we give TR8 the same score of ✓.

## TR9 The ability to model relations between continuants and occurrents

### TR9.1 Be able to make relations between the two types of distinct elements

Relations between continuants and occurrents can be modelled similarly in the exclusive cases described above. As it currently stands, the two classes *Continuant* and *Occurrent* have similar restrictions imposed on them, making modelling relations between them no more difficult than in the cases above. We begin with the relation *input of*, and move onto more complex relations in the next requirements. *input of* is a same time relation between a continuant and an occurrent. The relation should only hold at one time point. We also assume that both entities must be active during the relation. We can represent *C input of O* as follows:

$$(hasSameTimeRelation = hSTR)$$

$$C \sqsubseteq \exists hSTR.(\exists inputOf.O)\sqcap \quad (4.64)$$

$$\exists_=(hSTR \circ startTime, hSTR \circ endTime)\sqcap \quad (4.65)$$

$$\exists_{\leq}(startTime, hSTR \circ startTime)\sqcap \quad (4.66)$$

$$\exists_=(hSTR \circ inputOf \circ startTime, hSTR \circ startTime) \quad (4.67)$$

Again, we use reification (4.64) to specify that the relation last only a single time point (4.65), and also specify that both entities must be active before or equal to when the relation itself takes place (4.66, 4.67). Again we are even able to quantify when the relation takes place if it is needed. We give both TR9.1 and TR9 a score of ✓, again losing points as a whole due to the use of reification and the functional nature of *input of*.

**TR10 The ability to model relations between continuants and occurrents over past, future, and present time points, each including the continuant's state constraints**

TR10.1 Relate the two individuals at a single time point (time:same)

TR10.2 Relate the two individuals at a present and past time point (time:past)

TR10.3 Relate the two individuals at a present and future time point (time:future)

TR10.4 Correctly model the continuant coming into and out of existence (birth, death)

We have seen in previous requirements how domain constraints can be modelled on continuants. We now show how these can be modelled when the range of the

relation is an occurrent. We begin with the relation *C existence starts during O*. As within the *o-o* relations, *existence starts during O* is not specific to particular pairs of continuants and occurrents, but merely a general relation specifying when the existence of a continuant begins w.r.t an occurrent. Therefore we define the relation as follows:

$$\begin{aligned}
 & (\textit{existenceStartsDuring} = \textit{eSD}) \\
 C \sqsubseteq \exists_{\geq}(\textit{startTime}, \textit{eSD} \circ \textit{startTime}) \sqcap \exists_{\leq}(\textit{startTime}, \textit{eSD} \circ \textit{endTime})
 \end{aligned} \tag{4.68}$$

Here we specify that *C* has an *existenceStartsDuring* relation to an *O*, and its *startTime* must occur during *O*'s life time (assuming that *O* is the filler of the *eSD* relation). Again, due to the fact that *existenceStartsDuring* is abstract, we encounter the problem of having to declare multiple *existenceStartsDuring* relations, if multiple statements are to be made. We can constrain the other 3 relations in a similar way. *existence starts during or after*, *existence ends at point* and *existence ends during or before* can be modelled as follows:

$$\begin{aligned}
 & (\textit{existenceStartsDuringOrAfter} = \textit{eSDOA}) \\
 & (\textit{existenceEndsAtPoint} = \textit{eEAP}) \\
 & (\textit{existenceEndsDuringOrBefore} = \textit{eEDOB}) \\
 C \sqsubseteq \exists_{\geq}(\textit{startTime}, \textit{eSDOA} \circ \textit{startTime}) \\
 C \sqsubseteq \exists_{=}( \textit{endTime}, \textit{eEAP} \circ \textit{endTime}) \\
 C \sqsubseteq \exists_{\leq}(\textit{endTime}, \textit{eSD} \circ \textit{endTime})
 \end{aligned} \tag{4.69}$$

Once again, the only problems we see are that the features themselves are functional. Otherwise, the constraints we can enforce are very powerful and most of all, correct. We give each sub requirement a score of ✓. The time relations here specify something different from the *c-c* case. The main temporal constraint is to do with the existence in the relation themselves. We give TR10 a score of ✓. It loses points for the functional nature of its relations, and the obvious static nature of the environment, but relations can be captured between the two, and in this case, the relations were between the same correct individuals (reification was not needed).

**TR11 The ability to model same time rigid relations between continuants and occurrents**

**TR11.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR11.2 The relation can have a duration specified**

Modelling rigid same time relations between continuants and occurrents is just as difficult as in the *c-c* rigid case, especially if the occurrent has temporal parts. When the occurrent does not have temporal parts, the relations can be modelled in a similar way as we did previously by using reification. When they do have temporal parts then the relations become even harder to model, since each temporal part is represented by its own individual. We illustrate this using the *involvedIn* relation, where a continuant *C* is involved in an occurrent *O*, in particular in each of its temporal parts *O*<sub>1</sub>, *O*<sub>2</sub> and *O*<sub>3</sub>. Without considering the constraints on how the temporal parts are ordered, we could represent this as follows:

$$C \sqsubseteq \exists \text{involvedIn}.O_1 \sqcap \exists \text{involvedIn}.O_2 \sqcap \exists \text{involvedIn}.O_3 \sqcap \text{involvedIn}.O \quad (4.70)$$

but we will not be able to use *involvedIn* as an abstract feature without all successors being the same, and even if they were, the constraints on when the relation should hold for each temporal part would be incorrect. Therefore, we would need to reify each relation individually, since each temporal part would be represented by a different individual, in the same way as in previous rigid cases. We give TR11.1 a score of ✗ for the same reasons as in each previous rigid case. As is the same for TR11.2 receiving a ✓. Overall TR11 receives a score of ✗, again for the same reasons.

**TR12 The ability to allow for multiple future time lines where relations may or may not hold**

**TR12.1 The ability to express multiple futures where relations may hold in one future and not in others**

Unfortunately in  $\mathcal{ALC}(\mathcal{D})$ , due to the static environment and linear nature of the *encoded* time line, there is no possibility for having possible futures. It is difficult to express that there are two futures, where in one future a relation holds and in

another the relation does not. The best we can get in  $\mathcal{ALC}(\mathcal{D})$  is to use disjunction, where we can say the relation holds or it does not. To express that *C capable of carrying out O*, we can use the following axiom:

$$C \sqsubseteq \exists \text{carriesOut}.O \sqcup \neg \exists \text{carriesOut}.O \quad (4.71)$$

where we can also use reification to say when the relation should hold (as in the previous examples). This is of course not what is intended, and thus not a faithful representation of possible futures (the axiom is in fact a tautology). We could further reify and have two reified elements that represent each relation, where *C* is related to both, but this again is not what needs to be modelled as it would be impossible to differentiate between the two. For these reasons, we give TR12.1 and TR12 a score of **X**.

**TR13 The ability to model relations between occurrents and continuants**

**TR13.1 Be able to make relations between the two types of distinct elements**

Similarly to TR9, continuant to occurrent relations can be modelled in the same way as occurrent to continuant relations as there is no special constraints in the order of the elements in the relation. We give TR13 the same score as TR9.

**TR14 The ability to model same time rigid relations between occurrents and continuants**

**TR14.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR14.2 The relation can have a duration specified**

As with TR13, the order in which we specify the domain and range of the relations in question have no effect on the difficulty of modelling. Modelling rigid relations is as difficult in the TR11 case, therefore we give TR14 the same scores as TR11.

**R15 Any extension should be decidable and of suitably low complexity, have implementations or at least decision procedures for the most common reasoning problems and be readily available for use**



The concrete domain  $\mathcal{D}$  uses the domain  $\mathbb{Z}$  and predicates  $\{\leq, <, =, =_n, >, \geq, +_n\}$ . Recall from Chapter 2, as a result from [Lut01b], it was shown that any concrete domain whose domain contains  $\mathbb{N}$ , has a binary predicate for equality, a unary predicate for equality with zero, and a binary predicate for incrementation, in the presence of general TBoxes is undecidable for satisfiability and subsumption. In our concrete domain  $\mathcal{D}$ , we meet all of these conditions.  $\mathbb{N}$  is contained within  $\mathbb{Z}$ ,  $=$  is our equality predicate,  $=_n$  is our predicate for equality with 0 (substituting  $n$  for 0) and  $+_n$  is our predicate for incrementation (substituting  $n$  for 1). Therefore our concrete domain can be seen to be arithmetic, meaning that in the presence of general TBoxes,  $\mathcal{ALC}(\mathcal{D})$  would be undecidable, making it unsuitable. Losing any of the predicates would be a huge loss to the language. Dropping  $+_n$  would make it difficult to capture duration. Dropping  $=_n$  would limit our quantitative expressivity and dropping  $=$  would destroy our alignment of sequencing and same time relations. Removing general TBoxes would be too much of a loss on the DL expressivity.

With regards to their usability in practice, unlike the vast amount of work gone into researching the theoretical side of concrete domains in DLs, not much work has gone into bringing them into production. From our research, *tOWL* is the most recent work that has brought concrete domains into practice. *tOWL* [MFK12, FMK07] is an OWL based temporal formalism, designed as a set of extensions built on top of OWL-DL, that enables representation and reasoning with time and temporal aspects. It extends the standard OWL-DL language with a concrete domain based on the set  $\mathbb{Q}$  and a standard set of binary predicates:  $\{<, \leq, =, \neq, \geq, >\}$ . The *tOWL* extension contains 3 layers: 1. The concrete domain layer, 2. the temporal representation layer which includes time points, intervals and Allen's interval relations [All83] (captured through syntactic sugar using the predicates described above), and 3. the timeslice/fluents layer (incorporating the Fluent ontology we go on to evaluate in the next Chapter). In 2012 they implemented an ExpTime-complete algorithm for the  $\mathcal{ALC}(\mathcal{C})$  fragment, including several well known optimisations including absorption, unfolding, backjumping and more. Even so, the underlying logic itself is not sufficient for the task at hand - more expressive predicates are needed which we know we cannot have.

For this reason we give R15 a score of **X**.

### 4.2.1 Summary

We see that  $\mathcal{ALC}(\mathcal{D})$  fails to fulfil the TRs, and in terms of the scores given, is worse off than the previous case involving  $LTL_{\mathcal{ALC}}$  and  $CTL_{\mathcal{ALC}}$ . It is fair to say that there are two main problems that this logic suffers from, the first being the static environment requiring reification and the second being the detrimental nature of functional relations (both abstract and concrete features). The static environment is a big downfall for the logic. In a static world each element only exists once, unlike in  $LTL_{\mathcal{ALC}}$  where we have several elements with rigid interpretations over the time line, possibly with different properties, which is what we desire. Due to this static nature, we immediately have to move to reification to be able to consider elements at different time points. Reification has its advantages and works to some extent, but when considering temporal phenomena in bio-health ontologies it has detrimental effects, the most prominent being the issue of identity. As soon as we start having to introduce new elements to act as temporal representations of individuals or as binary relations, we lose the identity criteria of the original elements in question. This has a knock on negative effect on other important features such as rigidity and modelling occurrents and continuants.

The usefulness of the concrete domain acting as a temporal representation of a time line is hindered by how we can interact with it. Abstract (and functional) features are necessary for the logic to be able to relate elements of the DL domain with elements in the concrete domain, but having such features is also bad for biological relations. As we saw, having some of the relations as abstract, such as *developsFrom* means that an element can only have one *developsFrom* relation to another element. But it is often the case that many *developsFrom* relations can hold between individuals so we either have to sacrifice the temporal constraints, or again use reification further becoming more and more unfaithful. The concrete domain does give the logic many advantages over  $LTL_{\mathcal{ALC}}$  however, including the strict constraints and definitions we can make on certain relations (such as Allen's relations) and also the possibility to use the concrete domain in a quantitative environment. We take away four important points from this evaluation: a static environment is not sufficient, reification is not a viable option, abstract features although powerful are very limiting to our depth of modelling, and quantification over the time line can be very useful.

### 4.3 Fluents Evaluation

The fluent approach is a static approach to a temporal representation. The approach adopts a perdurantist, or four-dimensional view, where every entity is considered to be a perdurant. That is, every element in the ontology can be seen as having temporal parts. This approach is not seen as an extension so to speak as the fluent approach is really just a fluent ontology, represented in standard OWL, and anyone wanting to adopt the approach just has to conform to the constraints in the ontology. We chose this temporal representation for evaluation because of the nature of what the ontology intends to capture. Being able to refer to an object's different temporal parts seems like a much needed feature and it relates to our TRs directly.

#### TR1 The ability to model continuants

TR1.1 **Continuants must be traceable through time**

TR1.2 **Continuants must maintain their identity through time**

TR1.3 **Continuants must be able to undergo change**

Using the fluent approach, we represent continuants in a standard OWL way; defining a class *Continuant* to constrain the domain. But, in addition to this, we also have the notion of having *temporal parts* of an element, representing different temporal parts of an element over multiple temporal extents. The Fluent approach mainly focuses on representing the temporal parts of individuals (as seen in Chapter 2). We can apply the same notion of temporal parts to the classes themselves since we are mainly focused on representing the temporal patterns at a terminological and class level. Time is presented by classes of Time Intervals [HP04] for which each temporal part is associated with an interval via a relation called *temporalExtent*.

Suppose we had a continuant  $C$  and three time intervals  $t_1$ ,  $t_2$ , and  $t_3$ , ordered consecutively, for which we wanted to make statements about  $C$  for over these time intervals. The fluent approach suggests to use the relations *temporalPartOf*

and *temporalExtent* to relate temporal parts to standard OWL classes, and temporal parts to their respective time intervals. Consider the following axioms.

$$\begin{aligned}
C@t_1 &\sqsubseteq \exists \text{temporalPartOf}.C \\
C@t_2 &\sqsubseteq \exists \text{temporalPartOf}.C \\
C@t_3 &\sqsubseteq \exists \text{temporalPartOf}.C \\
C@t_1 &\sqsubseteq \exists \text{temporalExtent}.t_1 \\
C@t_2 &\sqsubseteq \exists \text{temporalExtent}.t_2 \\
C@t_3 &\sqsubseteq \exists \text{temporalExtent}.t_3
\end{aligned} \tag{4.72}$$

Here we introduce three temporal parts,  $C@t_1$ ,  $C@t_2$  and  $C@t_3$ , which are temporal parts of  $C$  at the time intervals,  $t_1$ ,  $t_2$  and  $t_3$  respectively. The idea behind what we want to represent is actually quite promising: each temporal part is meant to represent a unique temporal slice of the global instance of  $C$ . This enables us to make statements about each different part individually, for example,  $C@t_1$  may be when  $C$  was inactive,  $C@t_2$  may be when  $C$  was active and  $C@t_3$  may be when  $C$  was inactive again. Of course we could represent this in the following axioms:

$$\begin{aligned}
C@t_1 &\sqsubseteq \text{Before} \\
C@t_2 &\sqsubseteq \text{Active} \\
C@t_3 &\sqsubseteq \text{After}
\end{aligned} \tag{4.73}$$

where we reuse the state classes introduced previously (without the obvious temporal constraints) to account for the activity. We could even capture more basic types of change in a similar way by simply declaring each temporal part to be members of different classes, or have relations to different objects. Of course, since we are again in a static environment, we are forced to accept that the existence we model is really the best we can do since we only have one world of evaluation in which all entities exist at the only time point.

Since the fluent ontology incorporates OWL-Time, there are also possibilities to define the intervals  $t_1$ ,  $t_2$  and  $t_3$  in a structured way, for example to state that  $t_1$  comes before  $t_2$  etc. OWL-Time contains a set of object properties intended to simply model Allen's interval relations between its interval classes it defines. It

has relations such as *intervalEquals*, *intervalBefore*, *intervalMeets* etc., used to relate instances of intervals. For example, *intervalEquals* is defined as (Manchester Syntax)

```
ObjectProperty: intervalEquals
    Domain: ProperInterval
    Range: ProperInterval
```

we could then go on to constrain the intervals  $t_1, t_2$  and  $t_3$  as follows:

$$\begin{aligned} t_1 &\sqsubseteq \exists \text{intervalBefore}.t_2 \\ t_2 &\sqsubseteq \exists \text{intervalBefore}.t_3 \end{aligned} \tag{4.74}$$

The Time-Ontology does not have the expressivity to fully capture Allen's relations however. For example, it would be perfectly legal to also add that  $t_2$  comes before  $t_1$ , creating a contradiction according to Allen's calculus, highlighting some of the temporal inadequacies of this ontology.

Any change that could need to be captured could be simply stated by declaring constraints on each temporal part. For example, declaring  $C@t_1$  and  $C@t_2$  to be members of different classes could represent this type of change.

Other big problems lie with the semantics and the fact that we are simply again using reification, but rather on a class level than a relation level. We are introducing new elements for every temporal part of a class and we immediately lose any identity of any continuant, and therefore cannot trace an individual through time easily. This means that capturing change and so on is difficult to faithfully model. We give TR1.1, TR1.2 and TR1.3 each of score of ✗. We give TR1 however a score of ✓. This is because although the semantic nature fails to capture the requirements of continuants, the syntax of the temporal classes themselves were actually quite useful.

## TR2 The ability to model occurrents

### TR2.1 Occurrents can have limited phases of existence

### TR2.2 Occurrents can have temporal parts

From a static viewpoint, using the fluent approach should work well when modelling occurrents since occurrents are perdurants. Since each entity is seen to have a temporal part, referenced by relations to time intervals, it becomes quite

easy to model each temporal part and also some type of duration of an occurrent. As in the continuant case, we can reuse the classes and make statements about parts of an occurrent  $O$  to determine its state:

$$\begin{aligned}
 O@t_1 &\sqsubseteq \textit{Before} \\
 O@t_2 &\sqsubseteq \textit{Before} \\
 O@t_3 &\sqsubseteq \textit{Active} \\
 O@t_4 &\sqsubseteq \textit{Active} \\
 O@t_5 &\sqsubseteq \textit{After}
 \end{aligned}
 \tag{4.75}$$

These states are crucial for occurrents as they are supposed to have limited phases of existence. As before, it is difficult to specify the order of these states, for example the following axioms are still valid even though the order of the occurrent states are incorrect:

$$\begin{aligned}
 O@t_1 &\sqsubseteq \textit{Active} \\
 O@t_2 &\sqsubseteq \textit{Before}
 \end{aligned}
 \tag{4.76}$$

We can again use the relations that are available in OWL-Time that simulate Allen's interval relations to try to structure the intervals correctly, but this will not prevent us from making incorrect statements, which shows the temporal weakness of the logic. However we can capture some type of duration this way, but this refers back to the OWL-Time ontology, where durations refer to date-times. When considering temporal parts of an occurrent, we can get this for free as a consequence of the *hasTemporalPart* relation, since it models exactly what we need to capture directly. We give TR2.1 a score of ✗ since we cannot fully constrain what it means for an entity to exist in a static world with nothing other than class membership to state existence. We give TR2.2 a score of ✓ however since when identity is not a concern it is possible to capture what is intended. We give TR2 a score of ✗ overall.

### TR3 The ability to model same time rigid relations between occurrents or continuants

TR3.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR3.2 The relation can have a duration specified

Suppose we want to model that a continuant  $C_1$  was *part of* another continuant  $C_2$  for 3 time intervals,  $t_1, t_2$  and  $t_3$ . When adopting the fluent approach, we should be referring to each temporal part of the continuants and make the relations on them as follows:

$$\begin{aligned} C_1@t_1 &\sqsubseteq \exists \text{partOf}.C_2@t_1 \\ C_1@t_2 &\sqsubseteq \exists \text{partOf}.C_2@t_2 \\ C_1@t_3 &\sqsubseteq \exists \text{partOf}.C_2@t_3 \end{aligned} \tag{4.77}$$

Here we are stating that each temporal part of  $C_1$  at the time points in question are related the corresponding  $C_2$  temporal parts. It does model a type of rigidity, if we accept that the reified elements (the temporal parts) are in fact simulating the original individual. And we also manage to capture some type of duration on the relation itself based on what we state explicitly, which is a positive sign. The downsides are that we do not manage to faithfully capture what is intended by rigidity - the same elements are not related, and the relations are in no way consecutive in time - we rely on the ordering of the time intervals through standard OWL relations to structure these. It is also important to note that the fluent approach does not provide any insight behind the temporalisation of the relations themselves. For example, what if *part of* was to be considered a perdurant, and it too could have its own temporal parts, such as  $\text{partOf}@t_1$  standing in for the *part of* relation at the time slice  $t_1$ . This would not directly solve our problem but would be interesting to consider since no other logic that we have evaluated allows the temporalisation of relations. We see similar results for the occurrent case also. We give TR3.1 a score a ✗. We give TR3.2 a score of ✓ however since some kind of duration is possible when considering which time intervals we state the relations hold for. But overall, we give TR3 a score of ✗.

**TR4 The ability to model same time rigid relations between continuants**

**TR4.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR4.2 The relation can have a duration specified**

As before we score the whole of TR4 the same as TR3. There is no difference in between the modelling of rigidity between occurrents and continuants in the fluent approach.

**TR5 The ability to model same time relations between continuants****TR5.1 The relation must hold at a single time point (time:same)**

Same time relations are particularly easy to model using the fluent approach. If we wanted to state that a continuant  $C_1$  was *aligned with* another continuant  $C_2$  at a single time point  $t_1$ , then the following axiom would suffice:

$$C_1@t_1 \sqsubseteq \exists \text{alignedWith}.C_2@t_1 \quad (4.78)$$

We only need to ensure that the time interval  $t_1$  lasts a single time point, which can be done quite easily since the OWL-Time ontology also has classes for this such as Time Instances. The axiom can also be seen to state that the relation holds only at a single time slice too, since the relation holds only between those temporal parts. This may be important for when relations need to only hold in certain time slices and not in others. For that reason, we give TR5.1 a score of ✓, and TR5 a score of ✓✓.

**TR6 The ability to model past time relations between continuants****TR6.1 The relation must hold between individuals at present and past time points (time:past)**

Modelling relations seen to take place over multiple time points is easy to envision using the fluent approach and is also quite easy to model. For representing  $C_1$  *develops from*  $C_2$  at time points  $t_1$  and  $t_2$  where  $t_1 > t_2$ , we can simply use the following axiom:

$$C_1@t_1 \sqsubseteq \exists \text{developsFrom}.C_2@t_2 \quad (4.79)$$

Here we state the temporal part of  $C_1$  at  $t_1$  has a *develops from* relation to the temporal part of  $C_2$  at  $t_2$ . Although in a static environment, and the relation itself being static, the relation can be seen to have different start and end points, since the time slices  $t_1$  and  $t_2$  are different, and therefore, the individuals it relates belong to different time slices. If all time intervals have been set up correctly in the ontology, then it would also be possible to add a type of eventuality to, for the cases when exact time intervals are unknown. As an example, consider the



following axiom:

$$C_1@t_1 \sqsubseteq \exists \textit{developsFrom}.(\exists \textit{temporalPartOf}.C_2 \sqcap \exists \textit{hasTemporalExtent}.(\exists \textit{before}.t_1)) \quad (4.80)$$

Here we state in the right hand side of the axiom that the *develops from* successor has to be a temporal part of  $C_2$  and has a temporal extent that comes before  $t_1$ . Again, we rely fully on the intervals being structured correctly, but also on the fact that the axiom is correct - one needs to ensure the correct interval is used in the *before* successor, which should match the interval on the left hand side's temporal part.

The *develops from* relation also has a *change* feature present but we leave this for TR7 which focuses more on the states of continuants. We give TR6.1 a score of ✓ and TR6 as a whole a score of ✓ also. Although the idea behind the relation seems plausible, and the syntax - specifically being able to simulate a relation over multiple time points - seems ideal, there is still room for improvement.

**TR7 The ability to model the changing states of continuants, specifically their birth, death and other general changing states**

**TR7.1 Model a continuant coming into existence (birth)**

**TR7.2 Model a continuant going out of existence (death)**

**TR7.3 Model general changes of continuants (change)**

TR7 involves the relations *child nucleus of* and *develops from*, which includes the features *birth*, *death*, *time:past* and *changed*. We saw in TR6 how the *time:past* feature could be captured so we focus mainly on the state change features here. Starting first with the simple change seen in *develops from*, suppose we use the same example where  $C_1@t_1$  *develops from*  $C_2@t_2$ . The development could have been a simple change in class where  $C_1$  could have originally been a  $C_3$  at  $t_2$ . Development is usually a relation where the identity of the elements in the relation are the same, although this is not always the case. To maintain identity, ideally the class  $C_1@t_2$  should be a temporal part of  $C_3$  at  $t_2$ . If this was the case, then this should be captured in the axiom

$$C_1@t_1 \sqsubseteq \exists \textit{temporalPartOf}.C_3 \quad (4.81)$$

However, to correctly conform to the fluent ontology,  $C_1@t_1$  should have the temporal extent  $t_2$ , but it has the temporal extent  $t_1$ . This is obviously a problem, and shows that it is impossible to maintain the identity when considering a change of this nature. Instead, we are forced to further reify and introduce several new individuals to account for simple types of change. Therefore the change can simply be captured in the axiom:

$$C_2@t_2 \sqsubseteq C_3 \quad (4.82)$$

*child nucleus of* holds two more serious types of change involving existence. These are of particular importance as outlined in TR1 and TR2, but here they are specific to the relation at hand. If we want to specify that  $C_1@t_1$  was a *child nucleus of*  $C_2@t_2$  we could do so as follows:

$$C_1@t_1 \sqsubseteq \exists childNucleusOf.C_2@t_2 \quad (4.83)$$

$$C_1@t_i \sqsubseteq Before \quad (4.84)$$

$$C_2@t_j \sqsubseteq After \quad (4.85)$$

In (4.83) we state simply that the temporal part of  $C_1$  at the time interval  $t_1$  has a *child nucleus of* relation to an instance of a temporal part of  $C_2$  at  $t_2$ . (4.84) holds for all time intervals  $t_i$  that are declared to occur before  $t_1$  to capture that no temporal part of  $C_1$  existed before  $t_1$  and (4.85) holds for all time intervals  $t_j$  that are declared to occur after  $t_2$  to capture that no temporal part of can exist after the relation holds. Again, we rely on making sure the time intervals are declared and structured correctly in order for this to work and it would still be valid to make any statement about a class that is meant to be non existent. This again highlights the weakness of the fluent approach. It is still legal to make any statement about a class in the before or after state. We give TR7.1, TR7.2 and TR7.3 a score of **X**, due to the problems with identity and existence. And because of these reasons we give TR7 a score of **X**.

**TR8 The ability to model the time constraints of the following Allen's interval relations between occurrents: *before'*, *during*, *before*, *meets*, *meets'***

**TR8.1 Capture the temporal constraints intended by Allen's relations**

Since the fluent ontology incorporates the OWL-Time ontology, which itself has encodings of Allen's relations as object properties, if we wanted to model axioms using Allen's relations we can do so quite easily. For example, suppose we had two continuants,  $O_1$  and  $O_2$  where  $O_1$  was present at the time intervals  $t_1$  and was *immediately causally upstream of* (meets)  $O_2$  which was present at the time interval  $t_2$ . The meets relation itself should hold on the time intervals, so we declare the axiom:

$$t_1 \sqsubseteq \exists \text{meets}.t_2 \quad (4.86)$$

Then by declaring that  $O_1@t_1$  as the temporal part of  $O_1$  and a temporal extent of  $t_1$ , (similarly for  $O_2$  and  $t_2$ ), the meets relation indirectly holds between the two temporal parts. As before, there is nothing stopping us from making temporally incorrect statements, for example declaring that  $t_2$  happens before  $t_1$ , to try and force a contradiction - the relations are just simple object properties and nothing more, they contain no temporal information. And the actual relation does not hold on the temporal parts themselves but on their temporal extents limiting the use of standard OWL reasoning. We can model the other relations in a similar way. We give TR8.1 and TR8 both a score of **X**, due to their limitations in temporal expressiveness.

#### TR9 The ability to model relations between continuants and occurrents

##### TR9.1 Be able to make relations between the two types of distinct elements

Once again, continuants and occurrents are fundamentally the same type of elements when modelling using the fluent approach. They differ only in what class they belong to. Therefore, modelling relations between the two should not be more difficult as in either exclusive case. We show this using the relation *input of*. *input of* is a same time relation that specifies that a continuant is the input of some process. This is an example of a relation that should only hold at a single time point and not in all time points. This works well with the fluents interpretation of time, since we have already seen that making relations between the temporal parts of classes can be interpreted as only holding at the times involved in the temporal parts. Suppose we want to model that the continuant  $C$  at  $t_1$

was the *input of* the occurrent  $O$  at  $t_1$ . Then the axiom:

$$C@t_1 \sqsubseteq \exists \text{inputOf}.O@t_1 \quad (4.87)$$

Here we specify that the temporal part of  $C$  at  $t_1$  is a part of temporal part of  $O$  at  $t_1$ . Since we are referring to each class's temporal part then we can view this relation as only holding between each temporal part, and therefore only at the time interval  $t_1$ . And again, we can add constraints to ensure that  $t_1$  only lasts a single time point. Since occurrents often span over multiple time points, then we will again face problems with identity if we were relating a single continuant to several parts of the same occurrent (as seen previously). We give TR9.1 a score of ✓ since relations between the two types of entities can obviously be captured, and in some cases reasonably accurately. We give TR9 a score of ✓ also, losing points again with the issues of identity.

**TR10 The ability to model relations between continuants and occurrents over past, future, and present time points, each including the continuant's state constraints**

TR10.1 Relate the two individuals at a single time point (time:same)

TR10.2 Relate the two individuals at a present and past time point (time:past)

TR10.3 Relate the two individuals at a present and future time point (time:future)

TR10.4 Correctly model the continuant coming into and out of existence (birth, death)

Due to the general simplicity of the fluent approach, the requirements for TR10 can be generalised over those we have seen in previous TRs, specifically, TR7, TR6, TR2 and TR1. We therefore skip the detailed evaluation as it will be a repeat of those already seen. We give TR10.1-10.4 each a score of ✓ and TR10 a score of ✓ overall.

**TR11 The ability to model same time rigid relations between continuants and occurrents**

TR11.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

### TR11.2 The relation can have a duration specified

As in TR3 and TR4, rigid relations prove difficult to capture, even between continuants and occurrents. The same problem exists - identity. Suppose we wanted to model that a continuant  $C$  was *involved in* and occurrent  $O$  for its entire life time of the three time intervals,  $t_1, t_2$  and  $t_3$ . The obvious axioms would be as follows:

$$\begin{aligned} C@t_1 &\sqsubseteq \exists \text{involvedIn}.O@t_1 \\ C@t_2 &\sqsubseteq \exists \text{involvedIn}.O@t_2 \\ C@t_3 &\sqsubseteq \exists \text{involvedIn}.O@t_3 \end{aligned} \tag{4.88}$$

As mentioned above, the same problems exist involving the identity of the individuals that are being related (see TR3 for further explanation). We can capture some type of duration, based on what we state in the axioms, but the duration is not on the relation, but rather the axioms as a whole. It is possible to introduce a new occurrent, say  $O@t'$ , where  $t'$  can be seen as the time interval totally covering  $t_1, t_2$  and  $t_3$ , i.e.,  $t' = t_1 + t_2 + t_3$ . We could then state that  $C@t' \sqsubseteq \exists \text{involvedIn}.O@t'_t$ . We would then have use OWL-Time to relate each interval in the right way. For the reasons outlined above, we give TR11.1 a score a ✗ and we give TR11.2 a score of ✓. Overall, we give TR11 a score of ✗.

### TR12 The ability to allow for multiple future time lines where relations may or may not hold

#### TR12.1 The ability to express multiple futures where relations may hold in one future and not in others

As we are in a static environment, there is no option for having multiple future time lines, since technically there is no time line to start with. The time line that we consider is implied by the intervals declared, i.e., all subclasses of temporal extents, which can be structured in some way that simulates a time line. But in OWL-Time, there is only one of these time lines. There is nothing stopping us from extending this however, to allow the possibility of having multiple time lines, at least at a very high level. For example, we could declare sets of disjoint intervals that are meant to represent different temporal structures to be used as potentially different time lines. For example, the relation *capable of* is described

as a relation between a continuant and an occurrent in which the continuant is capable of *carrying out* the occurrent. Using *carriesOut* as a standard relation, we could model this as follows:

$$C@t_1 \sqsubseteq \exists \text{carriesOut}.O@t_1 \quad (4.89)$$

$$C@t_1' \sqsubseteq \neg(\exists \text{carriesOut}.O@t_1') \quad (4.90)$$

(4.89) refers to the temporal parts of both  $C$  and  $O$  at the time intervals  $t_1$  and (4.90) refers to the time intervals  $t_1'$ . The idea is that both  $t_1$  and  $t_1'$  represent the same time interval, but in separate temporal structures, and therefore different time lines. In the  $t$  time line, we declare that  $C@t_1$  has a *carriesOut* relation to  $O@t_1$  and in the  $t'$  time line, we state that no such relation exists between the two classes. Our approach is clearly not entirely faithful or adequate - we will eventually have to reproduce all time intervals for every separate time line we need, along with their structures leading to a large increase in the ontology, but it still captures some type of possible future. Therefore we score TR12.1 with ✓, but TR12 ✗.

### TR13 The ability to model relations between occurrents and continuants

#### TR13.1 Be able to make relations between the two types of distinct elements

Similarly to TR9, continuant to occurrent relations can be modelled in the same way as occurrent to continuant relations as there is no special constraints in the order of the elements in the relation. We give it the same score as TR9.

### TR14 The ability to model same time rigid relations between occurrents and continuants

#### TR14.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

#### TR14.2 The relation can have a duration specified

As with TR13 the order in which we specify the domain and range of the relations in question have no positive change. Modelling rigid relations is as difficult in the TR11 case, therefore we give TR14 the same scores.

**R15 Any extension should be decidable and of suitably low complexity, have implementations or at least decision procedures for the most common reasoning problems and be readily available for use**

The fluent ontology is an OWL ontology. It is not an extension to OWL by any means. This makes its suitability adequate. Its suitability is as good as the suitability of OWL is for ontology languages in general. Since it is encoded in OWL, there are many existing tools and support which are actively maintained and readily available. Anyone wishing to use the ontology can simply just import it and confine to its rules. We therefore give R15 a score of ✓✓.

### 4.3.1 Summary

We see that this approach fails to meet most of the temporal requirements. The only TR that the logic met completely was TR5, which was considered to be the only TR that needed minimal temporal features to satisfy. This result was not surprising however as we are still in OWL with no real temporal extension, only an ontology that describes temporal patterns. The main issues the approach suffered from was the problem of using reification to model temporal parts of an entity. Introducing elements to account for temporal parts of a temporal entity leads to several problems involving identity and existence, as was evident in the majority of the temporal requirements that held these features. These issues were also apparent in the original Fluent ontology paper [WF06] with regard to their original use case, as what they describe as *Redundant Objects*, again referring to the problem of reification leading to problems with identity conditions. When it comes to some of the most important temporal features, such as rigidity, it was clear that the approach was a good attempt to model what was needed, in terms of the syntax and the intended meaning of the relations. Being able to refer to the temporal part of an entity directly seems like a great idea, and it provides a great insight into considering the possibility of the temporal parts of relations also, but the problem lies with how the logic interprets those temporal parts and the fact that reification was used, leading to an unfaithful representation of the temporal features required. The underlying problem was that the environment was wholly static. Every entity we could have would be entirely static, and there was no way it could gain any temporal advantage when compared to previous extensions, which had some type of dynamic or non static entities.

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR1	✓	✓	✓
TR1.1	✓	✗	✗
TR1.2	✓	✗	✗
TR1.3	✗	✓	✗

Table 4.2: TR1 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR2	✗	✓✓	✗
TR2.1	✗	✓	✗
TR2.2	✓	✓	✓

Table 4.3: TR2 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

The positive aspects we take away from this evaluation are that being able to refer to an element's temporal parts comes with many benefits towards meeting the temporal requirements and the possibility of considering temporal parts of relations could also be a positive outcome.

We present a list of tables, Tables 4.2-4.16, summarising the results of the evaluation of  $LTL_{\mathcal{ALC}}$  &  $CTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and the Fluent ontology (FL) against the TRs, as evaluated above. We will use these results as a means to guide the design and investigation of a new TDL in the next chapter.

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR3	✗	✗	✗
TR3.1	✓	✗	✗
TR3.2	✗	✓	✓

Table 4.4: TR3 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL



	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR4	✗	✗	✗
TR4.1	✓	✗	✗
TR4.2	✗	✓	✓

Table 4.5: TR4 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR5	✓	✗	✓✓
TR5.1	✓	✓	✓

Table 4.6: TR5 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR6	✓✓	✗	✓
TR6.1	✓	✓	✓

Table 4.7: TR6 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR7	✓	✓	✗
TR7.1	✗	✓	✗
TR7.2	✗	✓	✗
TR7.3	✓	✗	✗

Table 4.8: TR7 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR8	✓	✓	✗
TR8.1	✓	✓	✗

Table 4.9: TR8 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR9	✓	✓	✓
TR9.1	✓	✓	✓

Table 4.10: TR9 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR10	✓	✓	✓
TR10.1	✓	✓	✓
TR10.2	✓	✓	✓
TR10.3	✓	✓	✓
TR10.4	✗	✓	✓

Table 4.11: TR10 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR11	$\times$	$\times$	$\times$
TR11.1	$\checkmark$	$\times$	$\times$
TR11.2	$\times$	$\checkmark$	$\checkmark$

Table 4.12: TR11 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$CTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR12	$\checkmark\checkmark$	$\times$	$\times$
TR12.1	$\checkmark$	$\times$	$\checkmark$

Table 4.13: TR12 Scores evaluated against  $CTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR13	$\checkmark$	$\checkmark$	$\checkmark$
TR13.1	$\checkmark$	$\checkmark$	$\checkmark$

Table 4.14: TR13 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
TR14	$\times$	$\times$	$\times$
TR14.1	$\checkmark$	$\times$	$\times$
TR14.2	$\times$	$\checkmark$	$\checkmark$

Table 4.15: TR14 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL
R15	$\times$	$\times$	$\checkmark\checkmark$

Table 4.16: R15 Scores evaluated against  $LTL_{\mathcal{ALC}}$  (and  $CTL_{\mathcal{ALC}}$ ),  $\mathcal{ALC}(\mathcal{D})$  and FL

# Chapter 5

## [x]- A New Family of TDLs

*How can we extend DLs to accommodate for the Temporal Requirements?*

### 5.1 Introduction

As we saw in the previous Chapter, the three temporal extensions we evaluated,  $LTL_{\mathcal{ALC}}$  (and  $CTL_{\mathcal{ALC}}$ ),  $\mathcal{ALC}(\mathcal{D})$  and the Fluent ontology, were not fully suitable for modelling the temporal nature of bio-health ontologies, w.r.t the Temporal Requirements (TRs) identified in Chapter 3, and some were even computationally inadequate to do so. This leaves us with two options: to further develop one of the current extensions to meet the TRs, or to design a new extension to meet the TRs. Although an extension may seem easier and more sensible, we believe that the second option is more suitable. The first thing to take into account is that the evaluated extensions were not created or investigated specifically for this problem at hand, and there is no guarantee that the logics can be *extended* at all. We have already seen that some features, such as *rigidity*, immediately makes some of the logics undecidable. Therefore, an extension could rather be seen as both a *restriction* and an extension to attempt to make it more suitable. Instead, we adopt a bottom up approach, by designing a new TDL based on what we learnt in the previous evaluation section.

We proceed as follows. Whilst considering the TRs along with the scores each logic received during the evaluation, we plan to pick and choose certain aspects of each logic that gave it an advantage or performed well on certain TRs, whilst at the same time, keeping the logic as simple as possible, trying not to push the

boundaries on decidability. We will then go on to define the syntax and semantics of the new logic. We will then attempt to show the suitability of the logic by evaluating it against the TR1-TR14, as we did previously. For R15, we will show decidability results and complexity analyses of the common reasoning problems of the new logic in Chapter 6. We will then go on to design and implement reasoners for the new logic to show its usability and practicality in Chapter 7.

## 5.2 Defining the Syntax

Based on the evaluation in the previous section, we saw that all logics have useful features in their syntax. In  $LTL_{\mathcal{ALC}}$  and  $CTL_{\mathcal{ALC}}$ , using temporal operators on class expressions proved useful in a qualitative environment, such as being able to relate a class to points on a time line, such as the *next moment in time* ( $\bigcirc$ ) or *some time in the future* ( $\Diamond$ ). However the logic lacked the ability to consider specific time points along a time line. The problem was that there was no way to consider class expressions or axioms to only hold at certain points in time. There was also no possibility to use the temporal operators on the relations, which were needed in some cases.  $\mathcal{ALC}(\mathcal{D})$  also had a nice syntax w.r.t the predicates available and how they interact with the concrete domain. Being able to relate start and end times of elements to specific points on the time line was hugely beneficial during its evaluation. The problems it faced were the functional nature of the abstract and concrete features. The abstract features were both part of the logic's success and failure. Their functionality characteristics allowed us to express powerful temporal constraints, but the fact that they were functional limited exactly how much you could say for one given element. The other major problem was again that relations themselves could not be temporalised, without the use of reification. A type of duration could also be captured with the concrete features and predicates given, which was also beneficial. The fluent approach had a surprisingly good syntax. Being able to refer to a class at different time slices (or intervals) was a good advantage. This approach could technically be applied to roles themselves too. The problems it faced were not issues directly with the syntax, but rather the semantics. A type of reification was again used, and we needed many axioms to relate temporal parts of a class to the whole part itself.

In an ideal setting, we would like to easily be able to relate both classes and roles (the building blocks of class expressions) to particular points or durations

on a given time line, whether they be specific or equivocal ones. Our first choice was to decide how to access points on a given time line. We decided to use time point intervals, either of the form  $[i, j]$  or  $[i, j]_x$  where  $i, j \in \mathbb{Z}$  and  $i \leq j$  and  $x$  is a variable. The first interval is a quantitative interval where we consider real time points. For example  $[0, 2]$  corresponds to the interval made up of the sequence of three time points 0, 1 and 2. The second can be seen as a qualitative interval, where the variable  $x$  is quantified over all time points in  $\mathbb{Z}$ . The interval  $[0, 2]_x$  represents all sequences of consecutive time points of length three over  $\mathbb{Z}$ :  $\{\dots, [-1, 0, 1], [0, 1, 2], [1, 2, 3], [2, 3, 4], \dots\}$ .

**Definition 22** (Interval)

*An interval  $\lambda$  is of the form  $[i, j]$  where  $i, j \in \mathbb{Z}$  and  $i \leq j$ . We call  $i$  the start point of  $\lambda$  and is denoted as  $\lambda^s$ . Similarly  $j$  is the end point of  $\lambda$  and is denoted as  $\lambda^e$ .*

*A variable interval (v-interval)  $\lambda_x$  is of the form  $[i, j]_x$  where  $i, j \in \mathbb{Z}$ ,  $i \leq j$  and  $x$  is a variable. We call  $i$  the start point of  $\lambda_x$  and is denoted as  $\lambda_x^s$  and similarly  $j$  is the end point of  $\lambda_x$  and is denoted as  $\lambda_x^e$ .*

We use the notation  $[i]$  and  $[i]_x$  as a syntactic sugar for an interval whose start and end points are the same, i.e.  $[i, i]$  and  $[i, i]_x$ .

The intuition behind the intervals is that classes *and* roles can be annotated with intervals where one should be able to refer to that particular class or role at different time points or durations on the time line, without the need for additional properties. For example, the annotated class  $A_{[0,1]}$  would refer to the class  $A$  at time points 0 and 1. Similarly the annotated role  $R_{[0,2]}$  would correspond to the role  $R$  at time points 0, 1 and 2. In the class example, it looks very similar to both the  $\mathcal{ALC}(\mathcal{D})$  and Fluent approach. For example, the  $\mathcal{ALC}(\mathcal{D})$  class expression  $A \sqcap \exists_{=0}(\text{startTime}) \sqcap \exists_{=1}(\text{endTime})$  could be seen as similar to  $A_{[0,1]}$ , as well as the Fluent class  $A@t_1$  where  $t_1$  was the interval  $[0, 1]$  and  $A@t_1$  was a temporal slice of another static class  $A$ . The differences will be that we do not need additional relations to specify when the start or end points of the class in question are, or that it is a temporal part of another more general class (although this will be later explained in the semantics).  $\text{LTL}_{\mathcal{ALC}}$  can not capture anything like this due to its qualitative nature. The annotated role  $R_{[0,2]}$  has no counterpart in either logic without reification and even then does not come close to a similar representation. For the v-interval examples,  $\mathcal{ALC}(\mathcal{D})$  and  $\text{LTL}_{\mathcal{ALC}}$  have similar representations. The class  $A_{[0,1]_x}$  stands for the class  $A$  at any

two consecutive time points along the time line. This is similar to the  $\mathcal{ALC}(\mathcal{D})$  class expression  $A \sqcap \exists_{+1}(startTime, endTime)$ . It is also more similar to the  $LTL_{\mathcal{ALC}}$  class expression  $A \sqcap \bigcirc A$ . Again, the role  $R_{[0,1]_x}$  can not be captured in  $LTL_{\mathcal{ALC}}$  (unless temporal constraints can be added on roles as studied in [LWZ08, AFWZ02]), and not without reification in  $\mathcal{ALC}(\mathcal{D})$ . The intuition behind the duration is that the intervals themselves encode a duration, seen as the count of an interval's time points. From a user perspective, this saves us having to create large class expressions as we saw in  $LTL_{\mathcal{ALC}}$ . It is also important to note at this stage that the intervals are only applied to classes and roles, not any complex class expression, as can be done elsewhere. We need to be careful in our extension to avoid undesirable effects (such as undecidability), so we plan to keep the logic as expressive as possible whilst being as simple as possible, limiting what we will consider a requirement over a desire.

We now provide a definition of the syntax of the new temporal extension. We combine the extension, which we denote as  $[x]$ , with the standard DL  $\mathcal{ALC}$ . We use  $\mathcal{ALC}$  to make the evaluation as fair as possible when comparing with the previously evaluated logics, and also due to its popularity and its many known complexity results. For now, we introduce two separate extensions. The first is  $\mathcal{ALC}_{[]}$ , which uses only the standard intervals in its syntax. The second, called  $\mathcal{ALC}_{[x]}$  uses only the variable intervals. After defining the semantics, we will go on to explain how the two can be combined in two distinct ways to form two variants,  $\mathcal{ALC}_{[][x]}$  and  $\mathcal{ALC}_{[x][]}$ .

We now go on to define the syntax of each extension.

### 5.2.1 $\mathcal{ALC}_{[]}$ Syntax

The syntax of  $\mathcal{ALC}_{[]}$  concept descriptions are an extension of  $\mathcal{ALC}$  concepts with the addition of intervals occurring on concept and role names. In  $\mathcal{ALC}$ , concept descriptions are built from countably infinite sets  $N_{con}$  and  $N_{role}$  of concept and roles names respectively. Since we now allow for intervals to appear on these names, we define  $N_{con}^{[]}$  and  $N_{role}^{[]}$  to be countably infinite and mutually disjoint sets of concept names and role names annotated with intervals respectively. Formally,  $N_{con}^{[]} = N_{con} \times \Lambda$  and  $N_{role}^{[]} = N_{role} \times \Lambda$ , where  $\Lambda$  is the set of all intervals.

Concept descriptions in  $\mathcal{ALC}_{[]}$  are inductively defined with the constructors  $\{\sqcap, \sqcup, \exists, \forall, \perp, \top, \neg\}$  according to the following definition:

**Definition 23** ( $\mathcal{ALC}_{[\ ]}$  Concept Descriptions)

Let  $A_\lambda \in N_{con}^{[\ ]}$  be an annotated atomic concept,  $R_\lambda \in N_{role}^{[\ ]}$  be an annotated atomic role (role name),  $C, D$  be arbitrary concept descriptions and  $\lambda \in \Lambda$  be an arbitrary interval. Then  $\mathcal{ALC}_{[\ ]}$  concept descriptions can be formed according to the following syntax rules:

$$C, D \longrightarrow \top_\lambda \mid \perp_\lambda \mid \neg C \mid A_\lambda \mid C \sqcap D \mid C \sqcup D \mid \exists R_\lambda.C \mid \forall R_\lambda.C$$

TBoxes in  $\mathcal{ALC}_{[\ ]}$  are defined in the usual way:

**Definition 24** ( $\mathcal{ALC}_{[\ ]}$  Terminological Axioms & TBoxes)

Let  $C, D$  be arbitrary  $\mathcal{ALC}_{[\ ]}$  concept descriptions. Terminological axioms are of the form

$$C \sqsubseteq D \text{ or } C \equiv D \quad (5.1)$$

An  $\mathcal{ALC}_{[\ ]}$  TBox is a finite set of these axioms.

When considering ABoxes, since we want to keep a rigid interpretation of individuals, there is no need to also annotate individuals with intervals as we did with concept and role names. If we wanted to keep to standard then  $N_{ind}^{[\ ]} = N_{ind}$ , where  $N_{ind}$  is the standard countably infinite set of individual names, also mutually disjoint with  $N_{con}$  and  $N_{role}$ .

**Definition 25** ( $\mathcal{ALC}_{[\ ]}$  Assertions & ABoxes)

Let  $C$  be an arbitrary  $\mathcal{ALC}_{[\ ]}$  concept description,  $R_\lambda$  be a role name from  $N_{role}^{[\ ]}$  and  $a$  and  $b$  be individual names from  $N_{ind}^{[\ ]}$ . Assertion axioms are of the form

$$C(a) \text{ or } R_\lambda(a, b) \quad (5.2)$$

An  $\mathcal{ALC}_{[\ ]}$  ABox is a finite set of these axioms.

**Definition 26** ( $\mathcal{ALC}_{[\ ]}$  Ontology)

An  $\mathcal{ALC}_{[\ ]}$  ontology (sometimes referred to as an  $\mathcal{ALC}_{[\ ]}$  knowledge base  $\mathcal{K}$ )  $\mathcal{O}$  is a pair  $(\mathcal{T}, \mathcal{A})$  where  $\mathcal{T}$  is an  $\mathcal{ALC}_{[\ ]}$  TBox and  $\mathcal{A}$  is an  $\mathcal{ALC}_{[\ ]}$  ABox.

Next, we introduce some notations for *interval occurrence* that will be useful for defining several functions on  $\mathcal{ALC}_{[\ ]}$  ontologies required later on in this chapter.

**Definition 27** (Interval Occurrence)

Let  $\lambda$  be an interval.  $\lambda$  occurs in:

- a concept description  $C$  if there is some annotated concept or role name for which  $\lambda$  is the interval for
- a TBox  $\mathcal{T}$  if  $\lambda$  occurs in some concept description  $C$  where  $C \in \alpha$  and  $\alpha \in \mathcal{T}$
- a ABox  $\mathcal{A}$  if  $\lambda$  occurs in some concept description  $C$  where  $C \in \alpha$  and  $\alpha \in \mathcal{A}$
- an Ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  if  $\lambda$  occurs in  $\mathcal{T}$  or  $\mathcal{A}$

**Definition 28** (Minimum and Maximum points of a Concept Description, TBox, ABox and Ontology)

Let  $\text{MIN}$  and  $\text{MAX}$  be functions that return the minimum and maximum time points of intervals occurring in concept descriptions, TBoxes, ABoxes and Ontologies. Let  $C$  be a concept description,  $\mathcal{T}$  a TBox,  $\mathcal{A}$  and ABox and  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  and Ontology. Let **minimum** and **maximum** be standard arithmetic functions of arity  $n$  that return minimum and maximum integers from a list of  $n$  integers respectively.

- $\text{MIN}(C) = \text{minimum}\{\lambda^s \mid \text{for all } \lambda \text{ that occur in } C\}$
- $\text{MAX}(C) = \text{maximum}\{\lambda^e \mid \text{for all } \lambda \text{ that occur in } C\}$
- $\text{MIN}(\mathcal{T}) = \text{minimum}\{\text{MIN}(C) \mid \text{for all } C \text{ that occur in } \mathcal{T}\}$
- $\text{MAX}(\mathcal{T}) = \text{maximum}\{\text{MAX}(C) \mid \text{for all } C \text{ that occur in } \mathcal{T}\}$
- $\text{MIN}(\mathcal{A}) = \text{minimum}\{\text{MIN}(C) \mid \text{for all } C \text{ that occur in } \mathcal{A}\}$
- $\text{MAX}(\mathcal{A}) = \text{maximum}\{\text{MAX}(C) \mid \text{for all } C \text{ that occur in } \mathcal{A}\}$
- $\text{MIN}(\mathcal{O}) = \text{minimum}\{\text{MIN}(\mathcal{T}), \text{MIN}(\mathcal{A})\}$
- $\text{MAX}(\mathcal{O}) = \text{maximum}\{\text{MAX}(\mathcal{T}), \text{MAX}(\mathcal{A})\}$

### 5.2.2 $\mathcal{ALC}_{[x]}$ Syntax

The syntax of  $\mathcal{ALC}_{[x]}$  concept descriptions are defined in the same way as  $\mathcal{ALC}_{[]}$  concept descriptions substituting the standard intervals for v-intervals. Therefore we introduce the sets  $N_{con}^{[x]}$  and  $N_{role}^{[x]}$  to be countably infinite sets of class names



annotated with variable intervals and role names annotated with variable intervals respectively. Formally  $N_{con}^{[x]} = N_{con} \times \Lambda_x$  and  $N_{role}^{[x]} = N_{role} \times \Lambda_x$  where  $\Lambda_x$  is the set of all variable intervals over the variable  $x$ .

Concept descriptions in  $\mathcal{ALC}_{[x]}$  are inductively defined with the constructors  $\{\sqcap, \sqcup, \exists, \forall, \perp, \top, \neg\}$  according to the following definition:

**Definition 29** ( $\mathcal{ALC}_{[x]}$  Concept Descriptions)

Let  $A_{\lambda_x} \in N_{con}^{[x]}$  be an annotated atomic concept,  $R_{\lambda_x} \in N_{role}^{[x]}$  be an annotated atomic role (role name),  $C_x, D_x$  be arbitrary concept descriptions where all class and role names use the same variable  $x$  and  $\lambda_x \in \Lambda_x$  be an arbitrary interval. Then concept descriptions can be formed in  $\mathcal{ALC}_{[x]}$  according to the following syntax rules:

$$C_x, D_x \longrightarrow \top_{\lambda_x} \mid \perp_{\lambda_x} \mid \neg C_x \mid A_{\lambda_x} \mid C_x \sqcap D_x \mid C_x \sqcup D_x \mid \exists R_{\lambda_x}.C_x \mid \forall R_{\lambda_x}.C_x$$

TBoxes in  $\mathcal{ALC}_{[x]}$  are defined in the usual way with the addition that every v-interval used in a terminological axiom must use the same variable.

**Definition 30** ( $\mathcal{ALC}_{[x]}$  Terminological Axioms & TBoxes)

Let  $C_x, D_x$  be arbitrary  $\mathcal{ALC}_{[x]}$  concept descriptions. Terminological axioms are of the form

$$C_x \sqsubseteq D_x \text{ or } C_x \equiv D_x \quad (5.3)$$

An  $\mathcal{ALC}_{[x]}$  TBox is a finite set of these axioms.

Assertions are also defined in the usual way, with the addition of indexes appearing on each assertion.

**Definition 31** ( $\mathcal{ALC}_{[x]}$  Assertions & ABoxes)

Let  $C_x$  be an arbitrary  $\mathcal{ALC}_{[x]}$  concept description,  $R_{\lambda_x}$  be a role name from  $N_{role}^{[x]}$ ,  $a$  and  $b$  be individual names from  $N_{ind}^{[x]}$  (defined in the obvious way) and  $i \in \mathbb{Z}$ . Assertional axioms are of the form

$$C_x(a)\langle i \rangle \text{ or } R_{\lambda_x}(a, b)\langle i \rangle \quad (5.4)$$

An  $\mathcal{ALC}_{[x]}$  ABox is a finite set of these axioms.

## Ontology

### Definition 32 ( $\mathcal{ALC}_{[x]}$ Ontology)

An  $\mathcal{ALC}_{[x]}$  ontology  $\mathcal{O}$  is a pair  $(\mathcal{T}, \mathcal{A})$  where  $\mathcal{T}$  is a *TBox* and  $\mathcal{A}$  is an *ABox*.

### Other Definitions

MIN and MAX have slight changes in their definitions due to the indexes used in an ABox. When defining  $\text{MIN}(\mathcal{A})$  and  $\text{MAX}(\mathcal{A})$ , each index  $i$  of each assertion is added to each  $\lambda_s$  and  $\lambda_e$  index in the assertion to account for the shift. Otherwise, the MIN and MAX are defined as usual.

## 5.3 Defining the Semantics

So far we have introduced the syntax of the new extensions involving intervals and v-intervals. We now go on to investigate and design the semantics of each extension.

We discussed in the previous section using integers as the representation of our time line, but not the precise way in which we intend to embed these into the semantics. Our first decision was to choose the discrete and linear structure of  $\mathbb{Z}$  as our temporal dimension.  $\mathbb{Z}$ , being discrete and unbounded in both directions, other than the issue of being non branching (for TR12), we believe would be the best temporal structure, leaving us with no limitations on representing information in the past or future. The intervals we defined previously are exactly interpreted as sequences of time points. We found that in both  $\text{LTL}_{\mathcal{ALC}}$  and  $\mathcal{ALC}(\mathcal{D})$  when we used  $\mathbb{N}$  or  $\mathbb{Z}$  respectively as our temporal dimension, we did not encounter any problems specifically with each representation (although we did encounter problems with wanting to discuss previous time points in  $\text{LTL}_{\mathcal{ALC}}$  not only with the time line but also the operators allowed). We opt to use  $\mathbb{Z}$  to allow us as much flexibility as possible, in both temporal directions.

$\text{LTL}_{\mathcal{ALC}}$  has the most beneficial semantics, specifically to do with their possible world semantics. The semantics allows for the rigid interpretation of individuals which proved very useful when considering notions such as identity and allowing elements to undergo change, which is crucial in biological modelling. It also allowed us to view rigidity on relations in almost the desired way, despite its infinite

and often undecidable nature. The problems we encountered with this representation was that we couldn't quantify the time line. The concrete domain approach had a nice way to interact with the time points by means of the predicates we defined but the failures of the logic were mainly related to its static environment. Since we want to avoid the necessity for reification which was a consequence of the static environment, and overcome the difficulties when it comes to modelling features such as change, we choose to embed the time line into a possible world semantics.

For the  $\mathcal{ALC}_{[]}$  case, the time line will be mapped to a subsequence of  $\mathbb{Z}$ . In the qualitative case, i.e.  $\mathcal{ALC}_{[x]}$ , the time line will simply be isomorphic to  $\mathbb{Z}$ . The first thing we will ensure will be the rigid interpretation of individuals, similar to what we saw in  $\text{LTL}_{\mathcal{ALC}}$ . This will also aid in our design of capturing some kind of rigid roles. We first begin with defining the semantics of annotated classes. Recall that the class  $A_{[0,2]}$  informally reads as

The class  $A$  at time points 0, 1 and 2

the keyword here being “*and*”. We plan to interpret the intervals with conjunctive readings, i.e. elements belonging to this class should be instances of  $A$  at times zero one and two. Or put another way, this class defines *a set of all elements that are instances of  $A$  at times zero, one and two*. We plan to interpret roles the same way.  $R_{[0,2]}$  should be read as *all pairs of elements that are  $R$  related at times zero, one and two*. If we were to also interpret this in a conjunctive reading then this would really be all individuals who were consecutively related to the same individual for the desired time. This in itself encodes a type of rigidity, which we define as *localised rigidity* (which will become clearer later on) as the rigidity is local to the interval itself. Before discussing this in more detail, we go on to formally define semantics of both  $\mathcal{ALC}_{[]}$  and  $\mathcal{ALC}_{[x]}$ .

### 5.3.1 $\mathcal{ALC}_{[]}$ Semantics

The semantics of concept descriptions is defined in terms of a temporal interpretation with possible worlds. The possible worlds can be seen as a finite sequence of standard  $\mathcal{ALC}$  interpretations.

**Definition 33** ( $\mathcal{ALC}_{[]}$  Semantics)

An  $\mathcal{ALC}_{[]}$  interpretation  $\mathcal{I} = ((\Delta^{\mathcal{I}_i}, \cdot^{\mathcal{I}_i})_{i \in \mathbb{Z}}, \cdot^{\mathcal{I}})$  consists of a sequence of non-empty

domains  $\Delta^{\mathcal{I}_i}$  and functions  $\cdot^{\mathcal{I}_i}$  that map each concept name  $A$  to a subset  $A^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_i}$ , each role name  $R$  to a subset  $R^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_i}$  and each individual name  $a$  to  $a^{\mathcal{I}_i} \in \Delta^{\mathcal{I}_i}$  where  $i$  is an index which represents a possible world. To ensure a rigid interpretation of individuals, we require that  $a^{\mathcal{I}_i} = a^{\mathcal{I}_j}$  for any  $i, j \in \mathbb{Z}$ . Therefore we simply use  $a^{\mathcal{I}}$  to refer to the interpretation of the individual  $a$  at any time point. We define  $\cdot^{\mathcal{I}}$  as a **global** function which maps each annotated concept name  $A_{[i,j]}$  to  $A_{[i,j]}^{\mathcal{I}} = \bigcap_{i \leq \ell \leq j} A^{\mathcal{I}_\ell}$  and each annotated role  $R_{[i,j]}$  to  $R_{[i,j]}^{\mathcal{I}} = \bigcap_{i \leq \ell \leq j} R^{\mathcal{I}_\ell}$  for all  $i, j, \ell \in \mathbb{Z}$ . The function  $\cdot^{\mathcal{I}}$  is inductively extended to arbitrary concepts by setting

- $\Delta^{\mathcal{I}} := \bigcup_{i \in \mathbb{Z}} \Delta^{\mathcal{I}_i}$
- $\top_{[i,j]}^{\mathcal{I}} := \bigcap_{i \leq \ell \leq j} \Delta^{\mathcal{I}_\ell}$
- $\perp_{[i,j]}^{\mathcal{I}} := (\neg \top_{[i,j]}^{\mathcal{I}})$
- $(\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} := C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\exists R_\lambda.C)^{\mathcal{I}} := \{e \in \Delta^{\mathcal{I}} \mid \exists f \in C^{\mathcal{I}} \wedge (e, f) \in R_\lambda^{\mathcal{I}}\}$
- $(\forall R_\lambda.C)^{\mathcal{I}} := \{e \in \Delta^{\mathcal{I}} \mid \exists f: (e, f) \in R_\lambda^{\mathcal{I}} \rightarrow f \in C^{\mathcal{I}}\}$

$C$  and  $D$  are arbitrary concept descriptions and  $R_\lambda$  is a role name from  $N_{role}^{[]}$ .

**Definition 34** ( $\mathcal{ALC}_{[]}^{\mathcal{I}}$  Satisfaction)

Let  $C$  and  $D$  be arbitrary concept descriptions and  $A_\lambda$ ,  $R_\lambda$  and  $e, f$  be concept, role and individual names from  $N_{con}^{[]}$ ,  $N_{role}^{[]}$  and  $N_{ind}^{[]}$  respectively.

- $C$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C^{\mathcal{I}} \neq \emptyset$
- $\mathcal{I}$  satisfies a TBox axiom  $C \sqsubseteq D$  if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$  satisfies a TBox axiom  $C \equiv D$  if  $C^{\mathcal{I}} = D^{\mathcal{I}}$
- $\mathcal{I}$  is a model of a TBox  $\mathcal{T}$  if it satisfies every axiom in  $\mathcal{T}$
- $\mathcal{I}$  satisfies a concept assertion  $C(e)$  if  $e^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$  satisfies a concept assertion  $R_\lambda(e, f)$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_\lambda^{\mathcal{I}}$
- $\mathcal{I}$  is a model of an ABox  $\mathcal{A}$  if it satisfies every axiom in  $\mathcal{A}$
- $\mathcal{I}$  is a model of an ontology  $\mathcal{O}$  if it satisfies every axiom in  $\mathcal{O}$

### 5.3.2 $\mathcal{ALC}_{[x]}$ Semantics

#### Definition 35 ( $\mathcal{ALC}_{[x]}$ Semantics)

The semantics is given by means of a temporal interpretation  $\mathcal{I} = ((\Delta^{\mathcal{I}_i}, \cdot^{\mathcal{I}_i})_{i \in \mathbb{Z}}, \cdot^{\mathcal{I}^k})$  where  $\Delta^{\mathcal{I}_i}$  is a sequence of non empty domains and  $\cdot^{\mathcal{I}_i}$  is a sequence of functions that map every concept name  $A \in N_{con}$  to a subset  $A^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_i}$ , every role name  $R \in N_{role}$  to a subset  $R^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_i}$  and every individual name  $a \in N_{ind}$  to an element  $a^{\mathcal{I}_i} \in \Delta^{\mathcal{I}_i}$ . Again, to ensure a rigid interpretation of individuals, it holds that  $a^{\mathcal{I}_i} = a^{\mathcal{I}_j}$  for any  $i, j \in \mathbb{Z}$ . Therefore we simply use  $a^{\mathcal{I}}$  to refer to the interpretation of the individual  $a$  at any time point.  $\cdot^{\mathcal{I}^k}$  is a global function that for every  $k \in \mathbb{Z}$ , maps every annotated concept name  $A_{[i,j]_x} \in N_{con}^{[x]}$  to  $A_{[i,j]_x}^{\mathcal{I}^k} = \bigcap_{k+i \leq l \leq k+j} A^{\mathcal{I}_l}$ , every annotated role name  $R_{[i,j]_x} \in N_{role}^{[x]}$  to  $R_{[i,j]_x}^{\mathcal{I}^k} = \bigcap_{k+i \leq l \leq k+j} R^{\mathcal{I}_l}$ .

The function  $\cdot^{\mathcal{I}^k}$  is inductively extended to arbitrary concepts by setting

- $\Delta^{\mathcal{I}} := \bigcup_{i \in \mathbb{Z}} \Delta^{\mathcal{I}_i}$
- $\top_{[i,j]_x}^{\mathcal{I}^k} := \bigcap_{k+i \leq l \leq k+j} \Delta^{\mathcal{I}_l}$
- $\perp_{[i,j]_x}^{\mathcal{I}^k} := (\neg \top_{[i,j]_x}^{\mathcal{I}^k})$
- $(\neg C)^{\mathcal{I}^k} := \Delta^{\mathcal{I}^k} \setminus C^{\mathcal{I}^k}$
- $(C \sqcap D)^{\mathcal{I}^k} := C^{\mathcal{I}^k} \cap D^{\mathcal{I}^k}$
- $(C \sqcup D)^{\mathcal{I}^k} := C^{\mathcal{I}^k} \cup D^{\mathcal{I}^k}$
- $(\exists R_{\lambda_x}. C)^{\mathcal{I}^k} := \{e \in \Delta^{\mathcal{I}^k} \mid \exists f \in C^{\mathcal{I}^k} \wedge (e, f) \in R_{\lambda_x}^{\mathcal{I}^k}\}$
- $(\forall R_{\lambda_x}. C)^{\mathcal{I}^k} := \{e \in \Delta^{\mathcal{I}^k} \mid \exists f: (e, f) \in R_{\lambda_x}^{\mathcal{I}^k} \rightarrow f \in C^{\mathcal{I}^k}\}$

for all  $k \in \mathbb{N}$ .

#### Definition 36 ( $\mathcal{ALC}_{[x]}$ Satisfaction)

Let  $C_x$  and  $D_x$  be arbitrary concept descriptions and  $A_{\lambda_x}$ ,  $R_{\lambda_x}$  and  $e, f$  be concept, role and individual names from  $N_{con}^{[x]}$ ,  $N_{role}^{[x]}$  and  $N_{ind}^{[x]}$  respectively.

- $C_x$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C^{\mathcal{I}^0} \neq \emptyset$
- $\mathcal{I}$  satisfies a TBox axiom  $C_x \sqsubseteq D_x$  if  $C_x^{\mathcal{I}^k} \subseteq D_x^{\mathcal{I}^k}$  for all  $k \in \mathbb{Z}$
- $\mathcal{I}$  satisfies a TBox axiom  $C_x \equiv D_x$  if  $C_x^{\mathcal{I}^k} = D_x^{\mathcal{I}^k}$  for all  $k \in \mathbb{Z}$

- $\mathcal{I}$  is a model of a TBox  $\mathcal{T}$  if it satisfies every axiom in  $\mathcal{T}$
- $\mathcal{I}$  satisfies a concept assertion  $C_x(e)\langle i \rangle$  if  $e^{\mathcal{I}} \in C_x^{\mathcal{I}^i}$
- $\mathcal{I}$  satisfies a concept assertion  $R_{\lambda_x}(e, f)\langle i \rangle$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_{\lambda_x}^{\mathcal{I}^i}$
- $\mathcal{I}$  is a model of a ABox  $\mathcal{A}$  if it satisfies every assertion in  $\mathcal{A}$
- $\mathcal{I}$  is a model of an ontology  $\mathcal{O}$  if it satisfies every axiom in  $\mathcal{O}$ .

### 5.3.3 A Brief Overview of the Semantics

The defined semantics give rise to powerful temporal features. We begin first by explaining the relations between classes. Recall from the fluent approach when we had two similar classes  $A@t_1$  and  $A@t_2$ . Both were meant to represent temporal parts of the same class  $A$ . However the only relation both classes have to the class  $A$  is by means of a relation called *temporalExtent*. This in itself lacks temporal information as we saw in its evaluation. For example suppose the time slices  $t_1$  and  $t_2$  are intervals both lasting two time points, where the former immediately precedes the latter (meets). Then any element who was an instance of both  $A@t_1$  and  $A@t_2$  should in theory be an instance of  $A$  at 4 consecutive time points, or at least an instance of  $A@t_3$  where  $t_3$  covered four time points that strictly contained  $t_1$  and  $t_2$ . Although each temporal extent had  $A$  in common, they were not related to each other temporally in any way other than what the user specified. Clearly this is temporal information that is needed but cannot be modelled. The semantics of [x] solve this problem directly.  $A_{[0,1]}$  and  $A_{[2,3]}$  are more than just distinct class names. Both of the  $A$ s in each class are referring to the same static  $A$  at multiple time points. Because of this, any instance that belongs to both classes, by definition also belongs to the class  $A_{[0,3]}$  solving the problem and including the relevant temporal information. This also holds true for the qualitative intervals. For example, consider the following axioms:

$$C_{[0]} \sqsubseteq A_{[0,1]} \sqcap A_{[2,3]} \quad (5.5)$$

$$A_{[0,3]} \sqsubseteq B_{[0]} \quad (5.6)$$

$$C \sqsubseteq A@t_1 \sqcap A@t_2 \quad (5.7)$$

$$A@t_3 \sqsubseteq B \quad (5.8)$$

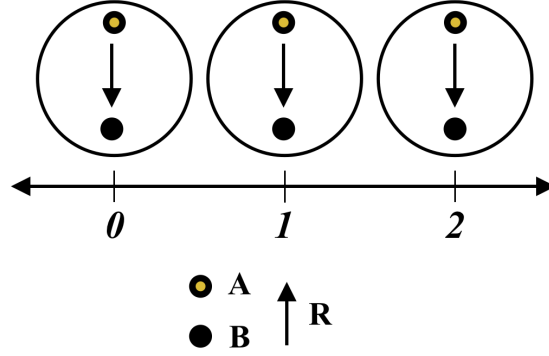


Figure 5.1: An example of localised rigidity in  $[x]$ , showing an element  $A$  being  $R$  related to an element  $B$  for 3 consecutive time points.

In the  $\mathcal{ALC}_{[]}$  axioms, it would follow that  $C_{[0,0]} \sqsubseteq B_{[0]}$ , but it would not follow from the fluent example that  $C \sqsubseteq B$ .

The semantics also works well for our localised rigidity. Suppose we had an annotated relation such as  $R_{[0,1]}$ .  $R_{[0,1]}$  contains all pairs of individuals which are  $R$  related for the consecutive time points 0 and 1. To illustrate this further, consider the following axioms:

$$A_{[0,1]} \sqsubseteq \exists R_{[0,1]}. B_{[0,1]} \quad (\mathcal{ALC}_{[]}) \quad (5.9)$$

$$A_{[0,1]_x} \sqsubseteq \exists R_{[0,1]_x}. B_{[0,1]_x} \quad (\mathcal{ALC}_{[x]}) \quad (5.10)$$

(5.9) states that all instances of  $A$  at 0 and 1 must be  $R$  related for two consecutive time points, also 0 and 1, to an individual that is an instance of  $B$  at 0 and 1. (5.10) states that any individual who is an instance of  $A$  at any two consecutive time points must have an  $R$  relation over the same two consecutive time points to an instance of  $B$ , also at the same two time points. This clearly defines a type of rigidity, local to the annotated role itself, hence the term *localised rigidity*. An illustration of this relation can be seen in Figure 5.1.

We mentioned previously the possibility to combine the two logics as it may be the case that in some ontologies both the qualitative and quantitative approach may be necessary. Combining the two logics can be done in two different ways, depending on which is the underlying logic. This is due to the different time lines in each logic.  $\mathcal{ALC}_{[][x]}$  is the result of adding  $\mathcal{ALC}_{[x]}$  to  $\mathcal{ALC}_{[]}$ , and  $\mathcal{ALC}_{[x][]}$  is the result of adding  $\mathcal{ALC}_{[]}$  to  $\mathcal{ALC}_{[x]}$ . The difference between the two is in direct

relation to the time line used. For the former, the time line used will still map to a subsequence of  $\mathbb{Z}$ , the addition is that qualitative axioms, specifically  $\mathcal{ALC}_{[x]}$  terminological axioms, can be used in the finite subsequence to easily quantify over all of the available time points. For the latter, the time line will remain isomorphic to  $\mathbb{Z}$ , however in addition, quantitative  $\mathcal{ALC}_{[]}$  axioms will be allowed. The syntax in each logic is the same, which we will now go on to define.

### 5.3.4 $\mathcal{ALC}_{[][x]}$ and $\mathcal{ALC}_{[x][]}$ Syntax

The syntax of  $\mathcal{ALC}_{[][x]}$  and  $\mathcal{ALC}_{[x][]}$  are the same, and can be defined as combinations of  $\mathcal{ALC}_{[x]}$  and  $\mathcal{ALC}_{[]}$ . Concept descriptions are simply built as the disjoint union of  $\mathcal{ALC}_{[]}$  and  $\mathcal{ALC}_{[x]}$  concept descriptions.

As with concept descriptions, terminological axioms and TBoxes are simply defined as the disjoint union of each previous one, and assertional axioms ABoxes are also the same.

An  $\mathcal{ALC}_{[][x]}$  or  $\mathcal{ALC}_{[x][]}$  ontology  $\mathcal{O}$  is a pair  $(\mathcal{T}, \mathcal{A})$  where  $\mathcal{T}$  is a TBox and  $\mathcal{A}$  is an ABox. MIN and MAX are defined in the usual way, and are extended to include both qualitative and quantitative input by combining both previous definitions.

We now go on to define the semantics which differ slightly for each extension.

### 5.3.5 $\mathcal{ALC}_{[x][]}$ Semantics

The semantics of  $\mathcal{ALC}_{[x][]}$  can be seen as an extension of  $\mathcal{ALC}_{[x]}$  to account for the additional standard (non variable) intervals. We extend  $\mathcal{I}$  to also include  $\cdot^{\mathcal{I}}$ .

#### Definition 37

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x][]}$  TBox,  $\mathcal{A}$  be an  $\mathcal{ALC}_{[x][]}$  ABox,  $\mathcal{O}$  be an  $\mathcal{ALC}_{[x][]}$  Ontology,  $C$  and  $D$  be arbitrary  $\mathcal{ALC}_{[]}$  concept descriptions,  $C_x$  and  $D_x$  be arbitrary  $\mathcal{ALC}_{[x]}$  concept descriptions,  $\alpha \in \mathcal{T}$  be an  $\mathcal{ALC}_{[]}$  TBox axiom of the form  $C \sqsubseteq D$ ,  $\alpha_x \in \mathcal{T}$  be an  $\mathcal{ALC}_{[x]}$  TBox axiom of the form  $C_x \sqsubseteq D_x$ ,  $R_\lambda$  be a role name in  $N_{role}^{[]}$ ,  $R_{\lambda_x}$  be a role name in  $N_{role}^{[x]}$  and  $e, f$  be individuals.

- $C$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C^{\mathcal{I}} \neq \emptyset$
- $C_x$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C_x^{\mathcal{I}^0} \neq \emptyset$
- $\mathcal{I}$  satisfies  $\alpha$  if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$



- $\mathcal{I}$  satisfies  $\alpha_x$ , if  $C_x^{\mathcal{I}^k} \subseteq D_x^{\mathcal{I}^k}$  for all  $k \in \mathbb{Z}$
- $\mathcal{I}$  is a model of  $\mathcal{T}$  if it satisfies every axiom in  $\mathcal{T}$
- $\mathcal{I}$  satisfies an assertion  $C(e)$  if  $e^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$  satisfies an assertion  $R_\lambda(e, f)$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_\lambda^{\mathcal{I}}$
- $\mathcal{I}$  is satisfies an assertion  $C_x(e)\langle i \rangle$  if  $e^{\mathcal{I}} \in C^{\mathcal{I}^i}$
- $\mathcal{I}$  is satisfies an assertion  $R_{\lambda_x}(e, f)\langle i \rangle$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_{\lambda_x}^{\mathcal{I}^i}$
- $\mathcal{I}$  is a model of  $\mathcal{A}$  if it satisfies every axiom in  $\mathcal{A}$
- $\mathcal{I}$  is a model of  $\mathcal{T}$  if it is a model of every axiom in  $\mathcal{T}$
- $\mathcal{I}$  is a model of  $\mathcal{O}$  if it models every axiom in  $\mathcal{O}$

### 5.3.6 $\mathcal{ALC}_{[]}[x]$ Semantics

The semantics of  $\mathcal{ALC}_{[]}[x]$  can be seen as an extension of  $\mathcal{ALC}_{[]}$  to also account for the additional v-interval usage. We extend  $\mathcal{I}$  to also include  $\cdot^{\mathcal{I}^k}$ .

#### Definition 38

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[]}[x]$  TBox,  $\mathcal{A}$  be an  $\mathcal{ALC}_{[]}[x]$  ABox,  $\mathcal{O}$  be an  $\mathcal{ALC}_{[]}[x]$  Ontology,  $C$  and  $D$  be arbitrary  $\mathcal{ALC}_{[]}$  concept descriptions,  $C_x$  and  $D_x$  be arbitrary  $\mathcal{ALC}_{[x]}$  concept descriptions,  $\alpha \in \mathcal{T}$  be an  $\mathcal{ALC}_{[]}$  TBox axiom of the form  $C \sqsubseteq D$ ,  $\alpha_x \in \mathcal{T}$  be an  $\mathcal{ALC}_{[x]}$  TBox axiom of the form  $C_x \sqsubseteq D_x$ ,  $R_\lambda$  be a role name in  $N_{role}^{[]}$ ,  $R_{\lambda_x}$  be a role name in  $N_{role}^{[x]}$ ,  $e, f$  be individuals, and MIN and MAX be defined as usual.

- $C$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C^{\mathcal{I}} \neq \emptyset$
- $C_x$  is satisfiable if there is an interpretation  $\mathcal{I}$  where  $C_x^{\mathcal{I}^0} \neq \emptyset$
- $\mathcal{I}$  satisfies  $\alpha$  if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$  satisfies  $\alpha_x$  if  $C_x^{\mathcal{I}^k} \subseteq D_x^{\mathcal{I}^k}$  for all  $k$  where  $\text{MIN}(\mathcal{T}) \leq k \leq \text{MAX}(\mathcal{T})$
- $\mathcal{I}$  is a model of  $\mathcal{T}$  if it satisfies every axiom in  $\mathcal{T}$
- $\mathcal{I}$  satisfies an assertion  $C(e)$  if  $e^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$  satisfies an assertion  $R_\lambda(e, f)$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_\lambda^{\mathcal{I}}$

- $\mathcal{I}$  is satisfies an assertion  $e : C_x \langle i \rangle$  if  $e^{\mathcal{I}} \in C_x^{\mathcal{I}_i}$
- $\mathcal{I}$  is satisfies an assertion  $(e, f) : R_{\lambda_x} \langle i \rangle$  if  $(e^{\mathcal{I}}, f^{\mathcal{I}}) \in R_{\lambda_x}^{\mathcal{I}_i}$
- $\mathcal{I}$  is a model of  $\mathcal{A}$  if it satisfies every assertion in  $\mathcal{A}$
- $\mathcal{I}$  is a model of  $\mathcal{T}$  if it is a model of every axiom in  $\mathcal{T}$
- $\mathcal{I}$  is a model of  $\mathcal{O}$  if it models every axiom in  $\mathcal{O}$

In the quantitative environments,  $\mathcal{ALC}_{[]}$  and  $\mathcal{ALC}_{[][x]}$ , MIN and MAX define the bounds of the time line for an ontology. In the qualitative environments, the time line remains unbounded, and potentially infinite in both directions.

### 5.3.7 Domain Constraints

In each extension,  $\Delta$  is indexed by each available time point in  $\mathbb{Z}$  (for each possible world). This allows us to consider various domain constraints; expanding, decreasing, constant or varying domains. These will be defined in the same way as in the  $\text{LTL}_{\mathcal{ALC}}$  case. Expanding domains are where  $\Delta^{\mathcal{I}_i}$  can only ever *grow* or remain unchanged as  $i$  increases. Decreasing domains are where  $\Delta^{\mathcal{I}_i}$  can only ever *shrink* or remain unchanged as  $i$  increases. Constant domains are where  $\Delta^{\mathcal{I}_i}$  can only remain unchanged as  $i$  increases and varying domains impose no constraints on  $\Delta^{\mathcal{I}_i}$  as  $i$  increases (the obvious constraints hold for decreasing values of  $i$ ).

**Definition 39** (Domain Constraints)

For all  $i \in \mathbb{Z}$

- *Expanding Domains:*  $\dots \subseteq \Delta^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_{i+1}} \subseteq \dots$
- *Decreasing Domains:*  $\dots \supseteq \Delta^{\mathcal{I}_i} \supseteq \Delta^{\mathcal{I}_{i+1}} \supseteq \dots$
- *Constant Domains:*  $\dots = \Delta^{\mathcal{I}_i} = \Delta^{\mathcal{I}_{i+1}} = \dots$
- *Varying Domains:* No constraints

### 5.3.8 **[x]**

We call the set containing each temporal extension the **[x]** family of Temporal Description Logics, i.e.,  $\mathbf{[x]} = \{[], [x], [[]x], [x][[]]\}$ .

Important Annotation	TRO Relation
domran:cc, time:same, rigid	connected to
domran:cc, dom:changed, time:past	develops from
domran:cc, time:same	aligned with
domran:cc, dom:birth, time:past, ran:death	child nucleus of
domran:oo, time:al-before-inverse	causally downstream of
domran:oo, time:al-during	happens during
domran:oo, time:al-before	precedes
domran:oo, time:al-meets-inverse	immediately preceded by
domran:oo, time:al-meets	immediately causally upstream of
domran:co, time:same, rigid	involved in
domran:co, time:same	input of
domran:co, time:same, rigid, uncertain	capable of
domran:co, dom:birth, time:same	existence starts during
domran:co, dom:birth, time:future/same	existence starts during or after
domran:co, dom:death, time:same	existence ends at point
domran:co, dom:death, time:past/same	existence ends during or before
domran:oc, time:same, rigid	occurs in
domran:xx, time:same, rigid	part of

Table 5.1: TRO important annotations with a corresponding TRO relation

## 5.4 Evaluation of $[x]$

We will now go on to evaluate  $[x]$ . We plan to evaluate  $[x]$  as a whole language and when necessary discuss each individual extensions. We again evaluate against the TRs, starting first with TR1-TR14. In addition, where possible, we also compare and discuss entailment differences between each logic, as we saw briefly in Section 5.3.3.

The relations we use in our evaluation can be seen again in Table 5.1.

### TR1 The ability to model continuants

#### TR1.1 Continuants must be traceable through time

#### TR1.2 Continuants must maintain their identity through time

#### TR1.3 Continuants must be able to undergo change

Continuants can be modelled fairly well in  $[x]$  and in most cases very similar to  $LTL_{AC}$  due to the similarities in their semantics. Each extension in  $[x]$  has a rigid interpretation of individuals coinciding with each possible world along the time

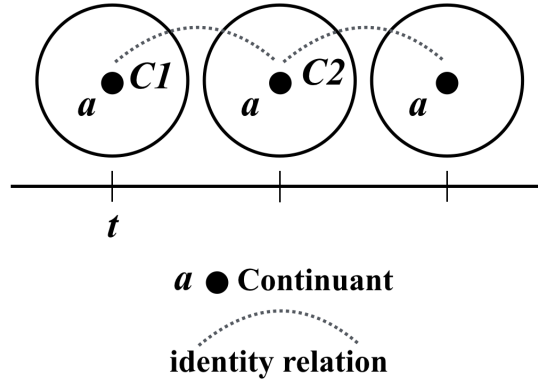


Figure 5.2: An illustration of the rigid interpretation of elements in  $[x]$ , and how they can be used for the effective modelling of Continuants and the modelling of change.

line, so we can be assured that any element  $a$  at a certain time point is that same  $a$  in any other time point where it exists. This can be seen in Figure 5.2, where although the individual  $a$  in each world belongs to a different domain, it holds true that they are interpreted the same. This bodes well for continuants since the elements are effectively maintaining their identity through time (TR1.2). To represent continuants, in the same way as before, we simply introduce a class  $\text{Continuant}_\lambda$ , for any interval  $\lambda$  when statements about continuants are needed. Continuants may undergo change and in some cases this is quite difficult to capture. Based on our survey results the changes needed include general changes such as changes in classes, related to some type of development, or a more specific type of change in the state of the continuant itself, for example a continuant coming into and out of existence. For the first case, we can capture general changes quite easily. For example suppose a continuant of type  $C$  at a time period, say  $0 - 2$ , changed into a  $C'$  at time 3. Using  $\mathcal{ALC}_{[]}$ , we can capture this as follows:

$$C_{[0,2]} \sqsubseteq C'_{[3]} \quad (5.11)$$

Here we state that any element which is an instance of  $C$  at times 0, 1 and 2 is a  $C'$  at time point 3. The identity of any continuant  $C$  at 0-2 maintains its identity into  $C'$  at 3, which is exactly what we need to capture. This is similar to the Fluent approach but with a richer semantics. If  $C'_{[3]}$  was to go on to change

again, for example,  $C'_{[3]} \sqsubseteq C''_{[4]}$ , then an entailment would follow that  $C_{[0,2]} \sqsubseteq C''_{[4]}$ . Such an entailment would not follow from either the fluent approach, nor the  $\mathcal{ALC}(\mathcal{D})$  approach, but it would follow from when modelling something similar in  $LTL_{\mathcal{ALC}}$ . This entailment follows directly from the possible world semantics and the rigid interpretation of individuals. Notice as well that when compared to  $LTL_{\mathcal{ALC}}$ , we can talk about specific time points. This is very important since the axioms are no longer global constraints, so the same axiom wouldn't apply for an instance of  $C$  at say times 2-4 ( $C_{[2,4]}$ ) even though the duration of the  $C$  remains the same. Therefore it is possible in  $LTL_{\mathcal{ALC}}$  to get additional entailments that may sometimes not be wanted. This is an advantage when certain facts are only supposed to hold only at certain times and not in others. Of course it also had disadvantages, for example if the axiom was to hold true for any individual that was an instance of  $C$  for two consecutive time points, in  $\mathcal{ALC}_{[ ]}$  we would have to quantify over all the time points as follows

$$C_{[\text{MIN}, \text{MIN}+2]} \sqsubseteq C'_{[\text{MIN}+3]} \quad (5.12)$$

$$C_{[\text{MIN}+1, \text{MIN}+3]} \sqsubseteq C'_{[\text{MIN}+4]} \quad (5.13)$$

...

$$C_{[0,2]} \sqsubseteq C'_{[3]} \quad (5.14)$$

$$C_{[0,2]} \sqsubseteq C'_{[3]} \quad (5.15)$$

...

$$C_{[\text{MAX}-3, \text{max}-1]} \sqsubseteq C'_{[\text{MAX}]} \quad (5.16)$$

$$(5.17)$$

which is obviously a problem especially considering the size increase of ontologies that will happen when the time line becomes very large. Here MIN and MAX would refer to the minimum and maximum points in the ontology which we can use correctly in a quantitative environment, since a finite time line is implied. We did anticipate this problem during the design of the new logics, which is why we also introduced variable intervals, more specifically  $\mathcal{ALC}_{[x]}$ . To capture the same axiom in  $\mathcal{ALC}_{[x]}$  we need only the axiom

$$C_{[0,2]_x} \sqsubseteq C'_{[3]_x} \quad (5.18)$$

which states that any instance of  $C$  for two consecutive time points will be an

instance of  $C'$  at the following time point. This is obviously very similar to  $\text{LTL}_{\mathcal{ACC}}$  but arguably more concise. Obviously this axiom is really the opposite of the first, i.e. this is a global constraint, whereas the first was a local constraint. Depending on what type of environment is needed, one can choose between the two extensions or if necessary even combine them for example in  $\mathcal{ACC}_{[][x]}$  or  $\mathcal{ACC}_{[x][]}$ .

The more specific type of change i.e. handling the changing states of continuants are more tricky to capture. From the survey we found two particular states of interest, existing and not existing, for the case where certain continuants may come into and out of existence at certain time points. Because of our possible world semantics with multiple domains, ideally we would like to be able to specify that elements may occur in some domains and not in others, which seems to be a perfect way to capture this situation. So we are left again with the same two options. The first is to use constant domains and introduce concepts to model the states continuants can go through and attempt to temporally constrain the order of those classes. The second is to use a possibly more faithful approach by adopting varying domains. For the constant domain case, we again introduce the three classes *Before*, *Active* and *After*. In the  $\text{LTL}_{\mathcal{ACC}}$  case, we can make global constraints, such as anything *Active* will eventually become *Inactive*, for example in the axiom  $\text{Active} \sqsubseteq \Diamond \text{After}$ . This isn't possible in  $\mathcal{ACC}_{[]}$  since we only have access to time points and no qualitative information, and more specifically we have no type of eventuality, hence we can only model specific information. For example suppose a continuant  $C$  came into existence at time 2 and went out of existence at time 4. The axiom

$$C_{[2,4]} \sqsubseteq \text{Active}_{[2,4]} \sqcap \text{Before}_{[\min,1]} \sqcap \text{After}_{[5,\max]} \quad (5.19)$$

states that any instance of  $C$  at times 2, 3 and 4 is *Active* at those time points and in the *Before* state in every time point before 2 and is in the *After* state any time point after 4. This would have to be stated for every class, varying depending on its interval. This is not so simple in the qualitative case. In the quantitative case we have direct access to the minimum and maximum time points as we used in the axiom to ensure that the correct constraints were captured. In the qualitative case we do not have minimum and maximum time points since the time line itself is potentially infinite. Therefore modelling the qualitative version is slightly more tricky. Suppose we were in  $\mathcal{ACC}_{[x][]}$  and we wanted to capture the

same constraints as above. We could do so in the following axioms:

$$C_{[2,4]} \sqsubseteq \text{Active}_{[2,4]} \sqcap \text{Before}_{[1]} \sqcap \text{After}_{[5]} \quad (5.20)$$

$$\text{Before}_{[0]_x} \sqsubseteq \text{Before}_{[-1]_x} \quad (5.21)$$

$$\text{After}_{[0]_x} \sqsubseteq \text{After}_{[1]_x} \quad (5.22)$$

In (5.20) we specify the same constraints on the continuant  $C$  at times 2-4, declaring it *Active*, and we also state that the continuant is in a *Before* state at 1 and then in an *After* state at 5. (5.21) and (5.22) state that any element in a *Before* state must be in a before state at the previous moment in time (which initiates an infinite descending sequence of *Before* states) and any element in the *After* state will be in the *After* state at the next moment in time (initiating an infinite ascending sequence of *After* states). Both axioms show similarities with LTL's  $\Box$  and  $\Box^-$  operators. This shows the benefit of having both a qualitative and quantitative approach combined.

Obviously a continuant or any entity for that matter should not be in two states at the same time. Therefore we should make the three states disjoint. However, simply stating that  $\text{Active}_{[2,4]}$  is disjoint from  $\text{Before}_{[2,4]}$  only makes these specific two classes disjoint, and not other temporal forms of the classes for example. We would therefore need to quantify over all of the time intervals and make them all disjoint. This is another key example where the combination of extensions becomes useful. For example as well as the axiom above we can also state the following disjoint axioms:

$$\text{Active}_{[0]_x} \sqsubseteq \neg \text{Before}_{[0]_x} \quad (5.23)$$

$$\text{Active}_{[0]_x} \sqsubseteq \neg \text{After}_{[0]_x} \quad (5.24)$$

$$\text{Before}_{[0]_x} \sqsubseteq \neg \text{After}_{[0]_x} \quad (5.25)$$

that states that any element that is *Active* at any time point cannot be also *Before* or *After*. Also in the qualitative environment, it is difficult to enforce some of the possible constraints between the states. For example, suppose we wanted to say any individual in the *Before* state would eventually be in an *Active* state. The axiom:

$$\text{Before}_{[0]_x} \sqsubseteq \text{Active}_{[0]_y} : y > x \quad (5.26)$$

would suffice but since all intervals in the axiom must be quantified over the same variable it diminishes any possibility of eventuality. This is one of the main reasons this logic differs from  $LTL_{\mathcal{AC}}$ . If we recall the constraints we made on continuants in  $LTL_{\mathcal{AC}}$ , then declaring any class  $C$  to be a continuant, then we immediately know certain entailments to follow such as  $C \sqsubseteq \Diamond After$  or  $C \sqsubseteq Active\mathcal{U}After$ . These are entailments that  $\mathcal{AC}_{[x]}$  lacks the power to express. It is even possible to get a somewhat *equivalent* entailment of eventuality in  $\mathcal{AC}(\mathcal{D})$ . If we recall that we declared continuants to having an *end time* concrete feature, then we know that any continuant  $C$  will have an end time which is larger than its *start time*. Although static, we still get some type of eventuality.

When considering varying domains instead of constant domains, we can use the  $\top$  operator to interact directly with the domain. Using the same example, suppose a continuant  $C$  came into existence at time 2 and came out of existence at time 4. In  $\mathcal{AC}_{[]}$  we could model this as follows:

$$C_{[2,4]} \sqsubseteq \neg \top_{[1]} \sqcap \neg \top_{[5]} \quad (5.27)$$

Here we specify that any instance of  $C$  at times 2-4 was not present in the domain at the previous and post time point before and after its existence. One has to be careful however since this axiom applies to all elements that are instances of  $C$  at 2-4. If there was a class  $C_{[2,5]}$ , it would also apply since  $C_{[2,5]}^{\mathcal{I}} \subseteq C_{[2,4]}^{\mathcal{I}}$  in all models  $\mathcal{I}$ . In a qualitative environment, since we can not consider exactly when an element may have come into and out of existence, we have to model it in a different way, similar to how we modelled existence in LTL. If we knew the duration of any  $C$  was 3 time points, then this could be modelled as follows:

$$C_{[0]_x} \sqsubseteq (\neg \top_{[-1]_x} \sqcap C_{[1]_x} \sqcap C_{[2]_x} \sqcap \neg \top_{[3]_x}) \sqcup \quad (5.28)$$

$$(C_{[-1]_x} \sqcap \neg \top_{[-2]_x} \sqcap C_{[1]_x} \sqcap \neg \top_{[2]_x}) \sqcup \quad (5.29)$$

$$(C_{[-1]_x} \sqcap C_{[-2]_x} \sqcap \neg \top_{[-3]_x} \sqcap \neg \top_{[1]_x}) \quad (5.30)$$

Here we again have to specify each of the possible phases  $C$  can be in during its lifetime.

We give TR1.1 and TR1.2 a score of  $\checkmark$  due to the advantages of the rigid interpretation of individuals. We give TR1.3 however a score of  $\times$  due to the weakness the logics has when it comes to modelling existence. We give TR1 a



score of ✓ overall.

## TR2 The ability to model occurrents

### TR2.1 Occurrents can have limited phases of existence

### TR2.2 Occurrents can have temporal parts

Occurrents are modelled in a similar way to continuants. We can introduce a class  $Occurrent_\lambda$  to represent all occurrents at any interval  $\lambda$ . Occurrents are described as having limited phases of existence, often durations and in some cases temporal parts. Existence of occurrents is crucial to model, and can be modelled in a similar way as in the continuant case, either using constant domains with state constraints, or using varying domains.

Occurrents also are described as having temporal parts, which are also occurrents that are separate entities. Suppose an occurrent  $O$  whose life time lasted 3 time points, had two temporal parts,  $O_1$  and  $O_2$ , whose life times last 1 time point and 2 time points respectively, spanning the whole of  $O$ 's life time. In  $\mathcal{ALC}_{[ ]}$ , suppose we know that  $O$  started at time 0. Then this could be represented using the *hasPart* relation and varying domains:

$$\begin{aligned} & hasPart = p \\ O_{[0,2]} & \sqsubseteq \neg \top_{[-1]} \sqcap \exists p_{[0]}.O_{1[0]} \sqcap \exists p_{[1,2]}.O_{2[1,2]} \sqcap \neg \top_{[3]} \end{aligned} \quad (5.31)$$

Here we state that any instance of  $O$  at times 0-2, must have a *hasPart* relation at time 0 to an instance of  $O_1$ , and must have a *hasPart* relation at times 1 and 2 to an instance of  $O_2$ , and also due to the localised rigidity, we have no issue ensuring that the instance of  $O_2$  is the *same* instance over the time points in which the relation holds. In  $\mathcal{ALC}_{[x]}$ , we follow a similar pattern to modelling in  $LTL_{\mathcal{ALC}}$ , but the localised rigidity still holds:

$$\begin{aligned} & hasPart = p \\ O_{[0]_x} & \sqsubseteq (O_{[1,2]_x} \sqcap \exists p_{[0]_x}.O_{1[0]_x} \sqcap \exists p_{[1,2]_x}.O_{2[1,2]_x} \sqcap \neg \top_{[-1]_x} \sqcap \neg \top_{[3]_x}) \sqcup \\ & (O_{[-1]_x} \sqcap O_{[1]_x} \sqcap \exists p_{[-1]_x}.O_{1[-1]_x} \sqcap \exists p_{[0,1]_x}.O_{2[0,1]_x} \sqcap \neg \top_{[-2]_x} \sqcap \neg \top_{[2]_x}) \sqcup \\ & (O_{[-2,-1]_x} \sqcap \exists p_{[-2]_x}.O_{1[2]_x} \sqcap \exists p_{[-1,0]_x}.O_{2[-1,0]_x} \sqcap \neg \top_{[-3]_x} \sqcap \neg \top_{[1]_x}) \\ & O_{1[0]_x} \sqsubseteq \neg \top_{[-1]_x} \sqcap \neg \bigcirc^- \top_{[1]_x} \\ & O_{2[0]_x} \sqsubseteq (\neg \top_{[-1]_x} \sqcap O_{2[1]_x} \sqcap \neg \top_{[2]_x}) \sqcup (\neg \top_{[1]_x} \sqcap O_{2[-1]_x} \sqcap \neg \top_{[-2]_x}) \end{aligned} \quad (5.32)$$

The modelling resembles that which we saw in the  $LTL_{\mathcal{ALC}}$  case. We use disjunctions to take into account the possible temporal phases that  $O$  could be in but the biggest difference is that when the temporal parts are concerned, specifically those that span over multiple time points, the localised rigidity suffices to meet the requirements.

Many ontologies in the OBO-Foundry, often those that discuss stage based development, have large sequences of occurrents which they call stages, organised in a sequential way that usually incorporate Allen's relations in some way. This is also very important for occurrents but focuses more on the relations between them rather than the modelling of the occurrents themselves, so we save this evaluation for TR5 and TR8.

We give TR2.1 a score of **X**. We still suffer with problems involving existence. Although duration is easy to capture in both quantitative environments, it is still difficult in the qualitative one. We give TR2.2 a **✓** since the ability to temporalise roles helps to capture important temporal features, involving maintaining identity with temporal parts. We give TR2 a score of **✓** overall.

### TR3 The ability to model same time rigid relations between occurrents or continuants

TR3.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)

TR3.2 The relation can have a duration specified

As we saw already in TR2, capturing rigid relations in **[x]** is now possible. The relation we use in our example is in fact the most important relation that was used across the OBO Foundry, *part of*. We begin first with the continuant case. In a quantitative environment if we wanted to express that a continuant  $C_1$  was part of another continuant  $C_2$  for the duration 0-3 we could model this in  $\mathcal{ALC}_{[]}$  as follows:

$$C_{1[0,3]} \sqsubseteq \exists partOf_{[0,3]}.C_{2[0,3]} \quad (5.33)$$

Here we specify that any instance of  $C_1$  at time 0-3 must have a *part of* relation also at time points 0-3 to an individual that is an instance of  $C_2$  at 0-3. This seems to be as *rigid* as the definition requires. We can specify a fixed rigid duration in which a relation should hold between the same elements. Even in a quantitative

environment, if we wanted to state that the relation holds for all time points then we could easily substitute the  $[0, 3]$  interval for the  $[\text{MIN}, \text{MAX}]$  interval. Clearly we capture both cases of rigidity and meet the requirements. If either continuant was to change i.e. become a member of a new class, the rigidity is actually local to the relation itself and not the classes, so no matter what the constraints were on the classes themselves the relation would still hold rigid between the same elements. For example, the axiom

$$C_{1\lambda_1} \sqsubseteq \exists \text{partOf}_{[0,3]}. C_{2\lambda_2} \quad (5.34)$$

where  $\lambda_1$  and  $\lambda_2$  were any interval, the relation would still be rigid at time 0 to 3, leaving the option open to constrain the classes.

Even in a qualitative environment, if we knew the duration that the relation should hold for, we can simply substitute the intervals for v-intervals as follows:

$$C_{1[0,3]_x} \sqsubseteq \exists \text{partOf}_{[0,3]_x}. C_{2[0,3]_x} \quad (5.35)$$

which states that any instance of  $C_1$  for four consecutive time points must have a *part of* relation for the same four time points to an individual that is also a  $C_2$  for the same four time points. However what we cannot do now is declare the relation to be globally rigid since we no longer have access to a MIN and MAX in a qualitative environment and therefore an infinite time line. This is the only downside to our form of rigidity but we know that having a global form of rigidity easily leads to undecidability [LWZ08].

Obviously the most important aspect of the rigidity is that the relation holds between the same elements. We saw in the  $\text{LTL}_{\mathcal{ALC}}$  attempt, that using conjunctions between existentials is not sufficient to ensure this. We now explain why this is the case. Consider the following  $\mathcal{ALC}_{[x]}$  and  $\text{LTL}_{\mathcal{ALC}}$  axioms:

$$C_1 \sqsubseteq \exists \text{partOf}. C_2 \sqcap \bigcirc \exists \text{partOf}. C_2 \sqcap \bigcirc \bigcirc \exists \text{partOf}. C_2 \sqcap \bigcirc \bigcirc \bigcirc \exists \text{partOf}. C_2 \quad (5.36)$$

Now suppose in addition we had the following (equivalent) axioms also present:

$$\begin{aligned} C_{2[0]_x} &\sqsubseteq \neg C_{2[1]_x} \quad (\mathcal{ALC}_{[x]}) \\ C_2 &\sqsubseteq \bigcirc \neg C_2 \quad (\text{LTL}_{\mathcal{ALC}}) \end{aligned} \quad (5.37)$$

In  $\mathcal{ACC}_{[x]}$  this would cause an inconsistency, but this would not be the case in  $\mathcal{ACC}_{[]}$ . The same also holds for  $\mathcal{ACC}(\mathcal{D})$  and the fluent case. In those examples, reification is often used, and therefore different elements are related. In each case we would have to impose more restrictions in order to catch the inconsistency, highlighting the benefits of  $\mathcal{ACC}_{[x]}$ .

We saw how to model rigid relations between occurrents in TR2 briefly. In fact, we can model rigid relations between them in the same way as we did with continuants above, since they are fundamentally the same type of elements in [x]. Since we can capture both the duration of the rigidity and ensure that the relations holds between the same elements we give TR3.1 and TR3.2 a score of ✓, and we give TR3 a score of ✓✓.

**TR4 The ability to model same time rigid relations between continuants**

**TR4.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR4.2 The relation can have a duration specified**

We give TR4 the same scores as TR3 due to their similarity. TR4 differs only in that we are interested in the continuant case only.

**TR5 The ability to model same time relations between continuants**

**TR5.1 The relation must hold at a single time point (time:same)**

Same time relations between continuants can be modelled fairly easily. To capture the same time feature we need only specify that the relation lasts a single time point, which we can do by annotating the desired relation within interval of duration of size one. This can be done in both the quantitative and qualitative environment very easily. We use the relation *aligned with* to aid our example In  $\mathcal{ACC}_{[]}$ , suppose we wanted to model that  $C_1$  was aligned with  $C_2$  at time  $i$ . We can model this as follows:

$$C_{1[i]} \sqsubseteq \exists alignedWith_{[i]}. C_{2[i]} \quad (5.38)$$

Here we specify that any instance of  $C_1$  at time  $i$  must have an *aligned with* relation at time  $i$  to an instance of  $C_2$ , also at time  $i$ . The relation only holds at

the desired time point and not anywhere else. From a qualitative viewpoint, in  $\mathcal{ACC}_{[x]}$  we can do exactly the same thing where we simply replace each interval with the variable interval  $[0]_x$ :

$$C_{1[0]_x} \sqsubseteq \exists \text{alignedWith}_{[0]_x}. C_{2[0]_x} \quad (5.39)$$

Here we specify that any instance of  $C_1$  in any single time point must be *aligned with* an instance of  $C_2$  also in the same time point as before. This is identical to  $\text{LTL}_{\mathcal{ACC}}$  version of the axiom. Obviously depending on the environment the user wishes to be in, we can choose whether the qualitative or quantitative or even both suffice. Again, as we saw in TR1, we can restrict our entailments to only quantitative ones, which is something we cannot do in  $\text{LTL}_{\mathcal{ACC}}$ . The requirement is clearly captured, therefore we give TR5.1 a score of ✓ and TR5 a score of ✓✓.

#### TR6 The ability to model past time relations between continuants

##### TR6.1 The relation must hold between individuals at present and past time points (time:past)

Modelling past time relations are more tricky than same time relations. The *time:past* feature indicates that a relation starts at one time point and effectively ends in another previous time point. In **[x]** all relations can only hold between elements at single time points which makes this difficult to capture, and requires a work around - the same work around we used in  $\text{LTL}_{\mathcal{ACC}}$ . Our example includes the relation *develops from*. Suppose we wanted to model that  $C_1$  at  $i$  *develops from*  $C_2$  at  $j$  where  $i > j$ . If time points  $i$  and  $j$  are known then in  $\mathcal{ACC}_{[ ]}$  we can model this as follows:

$$C_{1[i]} \sqsubseteq \exists \text{developsFrom}_{[i]}. C_{2[j]} \quad (5.40)$$

Here we specify that any instance of  $C_1$  at time  $i$  must have a *developsFrom* relation at time  $i$  to some individual that was a  $C_2$  at  $j$ . The relation itself only holds in the world  $i$ . Although this may seem misleading, consider the identity of the filler  $C_2$ . The rigid interpretation can ensure that the individual we relate to was definitely the same individual that was a  $C_2$  at  $j$ , so although the relation holds at  $i$ , it does relate to the correct individual.

From a qualitative view point, the same relation can be captured when exact time points are known, i.e we know the value of  $i-j$ . We could simply specify

$$C_{1[0]_x} \sqsubseteq \exists \text{developsFrom}_{[0]_x}. C_{2[0-(i-j)]_x} \quad (5.41)$$

that states that every instance of  $C_1$  must have a *develops from* relation to an instance that was a  $C_2$ ,  $i - j$  time points ago. Notice that we need no additional constructors to specify past time points, unlike in  $\text{LTL}_{\mathcal{AC}}$  where we need past operators. If we did not know the exact values of  $i$  and  $j$ , then this relation can not be captured. This again relates back to the problem of not being able to capture any type of eventuality as we saw in TR1. The relation also contains a *changed* feature, indicating that the domain goes through some kind of change. We discuss this in more detail in TR7 which focusses more the the domain constraints rather than the time constraints.

We give TR6.1 a score of ✓. We believe our modelling is sufficient, as was the case in  $\text{LTL}_{\mathcal{AC}}$ - although the relation holds in a single time point, it the constraints on the individuals are correct for the past and present time points as they should be. We give TR6 a score of ✓ due to its lack of eventuality.

#### TR7 The ability to model the changing states of continuants, specifically their birth, death and other general changing states

TR7.1 Model a continuant coming into existence (birth)

TR7.2 Model a continuant going out of existence (death)

TR7.3 Model general changes of continuants (change)

We saw in TR1 that the states of continuants can be captured by introducing classes in a constant domain environment, or by using varying domains. We begin first with the easier type of change using the relation *developsFrom*, as used in TR6. This relation has the feature *dom:changed*, since the domain went through some change, such as a type of development. For example  $C_1$  may have been an instance of another class  $C_3$  before the development into  $C_1$ . We can capture this as follows

$$C_{1[i]} \sqsubseteq \exists \text{developsFrom}_{[i]}. C_{2[i]} \sqcap C_{3[j]} \quad (5.42)$$

Here we specify that any instance of  $C_1$  at  $i$  has a *develops from* relation to an instance of  $C_2$  at  $j$ , and at  $j$ ,  $C_1$  was an instance of  $C_3$ . Again because the rigid interpretation of individuals we can be sure that the same instance of  $C_1$  was an instance of  $C_3$  at the previous time point  $j$ . This can also be modelled in the same way as in the qualitative version. But once again we do run into issues when considering eventuality.

Relations like *developsFrom*, often but not always involve identity constraints. As we saw before in our initial evaluation of TR7, *transformationOf* is a type of *developsFrom* relation where identity is maintained between the elements involved in the relation. In this case, it would be better to instead eliminate the use of a relation and just focus on class membership as follows:

$$C_{1[i]} \sqsubseteq C_{2[j]} \quad (5.43)$$

where we maintain the identity of the elements involved. We state that every  $C_1$  at time  $i$  was a  $C_2$  at time  $j$ . Of course, in the qualitative case, we will still need some kind of eventuality if the distance between  $i$  and  $j$  is unknown.

The second type of change we wish to consider is when entities come into and out of existence. The relation we use is *child nucleus of* which has the features *dom:birth* and *ran:death*. We identified two ways to capture this by using the state classes introduced in TR1. Suppose first we wanted to model  $C_1$  at  $i$  is a *child nucleus of*  $C_2$  at  $j$ .  $C_1$ 's existence should effectively start at  $i$  and  $C_2$ 's existence should end at  $j$  where  $i$  is one greater than  $j$ . As we saw in  $LTL_{\mathcal{ALC}}$ , due to the fact that relations can only hold between elements in single worlds, then varying domains are not a viable option here, so we opt to use the constant domain approach. In  $\mathcal{ALC}_{[]}$ , since we have access to MIN and MAX, we can capture this as follows:

$$C_{1[i]} \sqsubseteq Active_{[i]} \sqcap Before_{[MIN,j]} \sqcap \exists childNucleusOf_{[i]}.(C_{2[j]}) \quad (5.44)$$

$$C_{2[j]} \sqsubseteq Active_{[j]} \sqcap After_{[i,MAX]} \quad (5.45)$$

(5.44) states that any instance of  $C_1$  at time  $i$  is *Active* at time point  $i$ , was in the *Before* state until time  $j$  and at time point  $i$  has a *child nucleus of* relation to an instance of  $C_2$  that at  $j$ . (5.45) states that any instance of  $C_2$  at  $j$  was in the *After* state from  $i$  until the end of the time line and was in an *Active* state at  $j$ . Therefore the individual that  $C_1$  is related to will definitely be in the *After*

state. Here we capture the birth and death constraints quite easily since we can quantify over the time points. Of course we can still make statements about when  $C_2$  transitioned into the active state as well. In  $\mathcal{ALC}_{[x]}$  we do not have access to MIN and MAX, so the modelling changes slightly. Consider the following axiom:

$$C_{1[0]_x} \sqcap Active_{[0]_x} \sqcap Before_{[-1]_x} \sqsubseteq \exists childNucleusOf_{[0]_x}. (C_{2[-1]_x} \sqcap After_{[0]_x} \sqcap Active_{[-1]_x}) \quad (5.46)$$

(5.46) states that any instance of  $C_1$  that is *Active* and was previously in the *Before* state (pinpointing the transition), must have a *child nucleus of* relation to an instance of  $C_2$  at the previous time point that is now in the *After* state but was previously *Active*. This is equivalent to how we modelled the same relation in  $LTL_{\mathcal{ALC}}$ . Here we can only specify the exact point  $C_1$  transitions since we don't have access to any kind of eventuality, but since we pinpoint when it transitions all constraints seem to be met and no eventuality is needed. We again see that modelling attempts are near identical to those of  $LTL_{\mathcal{ALC}}$ , with only very slight differences.

We give TR7.1 and TR7.2 a ✖. The simple types of change are easier to capture so TR7.3 receives a score of ✓. We give TR7 a score a ✓ overall.

**TR8 The ability to model the time constraints of the following Allen's interval relations between occurrents:** *before'*, *during*, *before*, *meets*, *meets'*

**TR8.1 Capture the temporal constraints intended by Allen's relations**

In a quantitative environment, we are given another way to model temporal relations between occurrents. As we now have direct access to a time line we can structure occurrents on the time line in the way they are constrained by Allen's relations. It is often the case that ontologies describing stage based development have sequences of stages where durations and start and end points are known. In Figure 5.3 we give an example of a sequence of stages. Since the exact time points are known (or at least implied) then simply declaring the classes as follows

$$O_{1[1,2]}, O_{2[2,3]}, O_{3[0,4]}, O_{4[4,7]}, O_{5[5,6]}, O_{6[8,9]}, \quad (5.47)$$



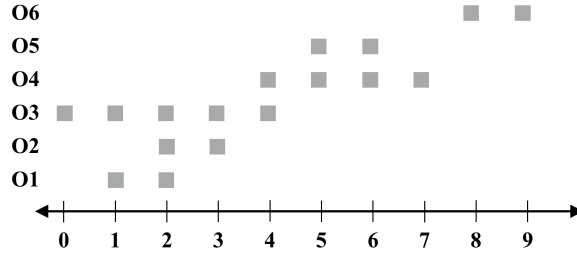


Figure 5.3: A collection of six occurrents organised along a finite time line of ten points

along with axioms describing their active and inactive states would be enough to simulate the constraints on the occurrents themselves. This can be seen as a way to capture the constraints of Allen's relations, but this is certainly not a way we could use the constraints to for example infer new information. For example there is no entailment that states  $O_1$  is *during*  $O_3$ . It would be possible to add in the appropriate relations, for example to state that  $O_1$  *happens during* (during)  $O_3$  and  $O_1$  *immediately causally upstream of* (meets)  $O_2$  we could do this in the following axioms:

$$O_{1[1,2]} \sqsubseteq \exists \text{happensDuring}_{[1,2]}. O_{3[0,4]} \quad (5.48)$$

$$O_{1[1,2]} \sqsubseteq \exists \text{immediatelyCausallyUpstreamOf}_{[2,2]}. O_{2[2,3]} \quad (5.49)$$

In (5.48) we state that any instance of  $O_1$  at times 1 and 2 has a *happens during* relation to an instance of  $O_3$  at times 1 and 2 (locally rigid) and in (5.49) we state that it also has a *immediately casually upstream of* relation to an instance of  $O_2$  at 2 and 3. Even with these relations added, other than the temporal information we capture, for example, the meets relation only holding at a certain time or the during relation being rigid between time points 1 and 2, there is nothing more we can get from Allen's relations themselves including inferring new information. Unlike  $\mathcal{ALC}(\mathcal{D})$ , we are not defining the relations, we are merely defining their constraints inside an axiom, which goes against the nature of Allen's relations. This is further illustrated in the fact that we could even add inaccurate but legal statements regarding Allen's relations. For example, we can add the following axiom

$$O_{1[1,2]} \sqsubseteq \exists \text{happensDuring}_{[1,2]}. O_{4[4,7]} \quad (5.50)$$

which although is incorrect and invalid from the intended viewpoint of Allen's interval relations, it is perfectly legal to have in [x], again, the same as what we saw in the Fluent approach.

In a qualitative environment, the modelling is again similar to the  $LTL_{ACC}$  approach, but in most cases is actually more difficult to model due to the lack of eventuality in [x]. Suppose we wanted to model that  $O_1$  happens during  $O_2$ .  $O_1$ 's start time must be greater than or equal to  $O_2$ 's start time, and its end time must be less than or equal to  $O_2$ 's end time. Without knowing the duration of either occurrent, we cannot model this relation since there is no way of pinpointing *some* previous time point when  $O_1$  came into existence, similarly for *some* future time point when it will cease to exist. If the duration is known, then we have to take each temporal phase into account similar to how we modelled the axioms in TR2. Each relation suffers from this fact in the qualitative approach. We give TR8 and TR8.1 both a score of ✗.

#### TR9 The ability to model relations between continuants and occurrents

##### TR9.1 Be able to make relations between the two types of distinct elements

The only difference between continuants and occurrents is simply just the classes that they belong to when modelling them in [x]. Modelling relations between the continuants and occurrents is just as easy as in either exclusive case as we saw in every other evaluation. The example we use is the relation *input of*. This relation uses the feature *time:same*. More specifically this is an example where the relation should hold only at a specific time point. Suppose a continuant  $C_1$  was the *input of* an occurrent  $O$  at the time point  $i$ . We can capture this in the following axiom:

$$C_{[i]} \sqsubseteq \exists inputOf_{[i]}.O_{[i]} \quad (5.51)$$

Here we state that any instance of the continuant  $C$  at time  $i$  must have an *input of* relation to an occurrent  $O$  at the same time point. Once again the quantification is an advantage here as it is possible that the relation may and in some cases should only hold at one time point. Even if the relation was to be

interpreted globally, then we could switch the axiom for the qualitative version:

$$C_{[0]_x} \sqsubseteq \exists inputOf_{[0]_x}.O_{[0]_x} \quad (5.52)$$

that states that any instance of  $C$  at any time point must have an *input of* relation to an instance of  $O$  at the same time point. We can clearly model relations between the two types of elements just as easy as we could in each exclusive case. And in our example, we can faithfully represent what was temporally intended from the relation itself. Therefore, we give TR9.1 a score of ✓ and TR9 a score of ✓✓.

**TR10 The ability to model relations between continuants and occurrents over past, future, and present time points, each including the continuant's state constraints**

TR10.1 **Relate the two individuals at a single time point (time:same)**

TR10.2 **Relate the two individuals at a present and past time point (time:past)**

TR10.3 **Relate the two individuals at a present and future time point (time:future)**

TR10.4 **Correctly model the continuant coming into and out of existence (birth, death)**

TR10 focuses on modelling relations between continuants and occurrents involving particular state constraints. The relations we use are *existence starts during*, *existence starts during or after*, *existence ends at point* and *existence ends during or before*.

In a quantitative environment, we are given another option to modelling at least some of the relations above, which differ heavily from the qualitative approach. For *existence starts during*, suppose we wanted to model that  $C$  *existence starts during*  $O$  in a quantitative environment where  $C$  was active over time points 0-2 and  $O$  was active from 0-3. Then the axioms:

$$C_{[0,2]} \sqsubseteq \neg \top_{[-1]} \sqcap \neg \top_{[3]} \quad (5.53)$$

$$O_{[0,3]} \sqsubseteq \neg \top_{[-1]} \sqcap \neg \top_{[4]} \quad (5.54)$$

are sufficient to state the correct constraints on the existence the relation implies. The problem lies in the fact that no relation between the two elements exist, so any entailment we wish to gain between the two elements is lost. Of course we can always add a relation between them, but as we have seen previously this would not gain any advantage. This problem was also present in TR8 when attempting to model Allen's relations in this way.

From a qualitative view point, suppose we wanted to model *C existence starts during O* without knowing the exact time points. Without knowing the duration of each of the elements lifetime, we struggle again with the lack of eventuality. Ideally, we would like to model something similar to the  $LTL_{ACC}$  case, where we used the axiom  $C \sqsubseteq \Diamond^-(\exists X.(O) \sqcap \neg \bigcirc^+ \top)$ . But since we don't have access to something that resembles a  $\Diamond$  operator, we have to change the way we model the axiom:

$$C_{[0]_x} \sqcap \neg \top_{[-1]_x} \sqsubseteq \exists X_{[0]_x}. O_{[0]_x} \quad (5.55)$$

Here we state than any instance of *C* that has just transitioned into the active state, must have done so whilst being related to an instance of an active *O*, hence it happened *during O*.

The remaining relations all suffer the same disadvantages as the first, and are all modelled in a similar fashion to the first also, and similar to those seen in  $LTL_{ACC}$ , so we leave them out of the evaluation. We give TR10.1-10.3 a score of ✓, but TR10.4 a score of ✗. We believe the problems lie mainly with the domain constraints and the modelling on the elements themselves, rather than with the way the time relations are modelled, since the time relations are embedded into the domain constraints. We give TR10 a generous ✓.

**TR11 The ability to model same time rigid relations between continuants and occurrents**

**TR11.1 Each relation must hold at a single time point between the same individuals (time:same, rigid)**

**TR11.2 The relation can have a duration specified**

Rigid relations between continuants and occurrents are just as easy and effective as in the exclusive case. *involved in* is the relation we use in our example. Suppose we wanted to model that a continuant *C* was *involved in* an occurrent *O* for its

duration of 4 time points. And suppose those 4 time points were 0-3. Without considering the states of each entity (since we have already seen how to capture these), consider the following axiom:

$$C_{[0,3]} \sqsubseteq \exists \text{involvedIn}_{[0,3]}.O_{[0,3]} \quad (5.56)$$

Here we state that the continuant  $C$  at times 0-3 must be involved in the occurrent for those same time points. Once again the relation holds between the same elements for a fixed duration. This also holds with the qualitative case to:

$$C_{[0,3]_x} \sqsubseteq \exists \text{involvedIn}_{[0,3]_x}.O_{[0,3]_x} \quad (5.57)$$

We run into the same problems as in the exclusive case, the only one being the global rigidity, but this is only present in an infinite time line case. Therefore we give TR11.1 and TR11.2 both a score of ✓, since clearly the identity constraints can be captured and also duration of the rigidity which is important for occurrents. We give TR11 a score of ✓✓.

**TR12 The ability to allow for multiple future time lines where relations may or may not hold**

**TR12.1 The ability to express multiple futures where relations may hold in one future and not in others**

Unfortunately in any extension in [x] there is no possibility for multiple time lines as each time line is discrete and linear (isomorphic to  $\mathbb{Z}$  or strict subsets of  $\mathbb{Z}$ ). This eliminates the possibility for a branching time line and possibility of capturing those relations that may or may not hold. We are left only with the option to use a disjunction as was the case in the  $\mathcal{ALC}(\mathcal{D})$  evaluation. Unfortunately this does not capture the requirements, proving to be insufficient. Therefore we give TR12.1 and TR12 a score of ✗.

**TR13 The ability to model relations between occurrents and continuants**

**TR13.1 Be able to make relations between the two types of distinct elements**

Similarly to TR9 we can model relations between occurrents and continuants in the same way. There is nothing specific in the order of the domain and range and

TDL	Problem	Result
$\mathcal{EL}_{[], \mathcal{EL}_{[]}[x]}$	Subsumption	PTime
$\mathcal{EL}_{[], \mathcal{EL}_{[]}[x]}$	Classification	PTime
$\mathcal{ALC}_{[], \mathcal{ALC}_{[]}[x]}$	Satisfiability	ExpSpace
$\mathcal{ALC}_{[], \mathcal{ALC}_{[]}[x]}$	Ontology Consistency	2ExpSpace
$\mathcal{EL}_{[x]} \rightarrow$	Subsumption	PTime
$\mathcal{EL}_{[x]} \rightarrow$	Classification	PTime

Table 5.2: Complexity Results (membership only) for fragments of  $[x]$  with Constant Domains and unary encodings of interval boundaries. ( $\rightarrow$ =future only)

it actually proves to be as easy as in the exclusive case. We score TR13 the same as TR9.

**TR14 The ability to model same time rigid relations between occurments and continuants**

TR14.1 **Each relation must hold at a single time point between the same individuals (time:same, rigid)**

TR14.2 **The relation can have a duration specified**

Once again, capturing rigid relations between occurments and continuants can be done in the same way as the continuant case and of course, either exclusive case. Therefore we score TR14 the same as TR11.

**R15 Any extension should be decidable and of suitably low complexity, have implementations or at least decision procedures for the most common reasoning problems and be readily available for use**

$[x]$  is an entirely new logic and the entirety of its research is spread across this thesis. In Chapter 6 we introduce several decision procedures for fragments of  $[x]$  when combined with the DLs  $\mathcal{EL}$  and  $\mathcal{ALC}$ . Although these are only small subsets of the DL underlying OWL ( $\mathcal{SROIQ}(\mathcal{D})$ ), the results we saw were positive, outlined in Table 5.2. We introduced extensions to the common reasoning problems, such as *temporal classifications* and showed that they remain decidable, even with localised rigidity and no restrictions on TBoxes. Although not all of the languages in  $[x]$  were proved to be decidable, none were yet shown to be *undecidable* and there was no obvious sign (as there is in the  $LTL_{\mathcal{ALC}}$  cases) that suggests this. In Chapter 7, we have designed two OWL Reasoners, called  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$ ,

that implement decision procedures for the common reasoning problems for the logics  $\mathcal{EL}_{[]}$  and  $\mathcal{ALC}_{[]}$ , which are compatible with the OWL API [HB11], and are able to solve classification, satisfiability, subsumptions and ontology consistency for valid ontologies in either language. Both reasoners are readily available for use. We have conducted experiments to prove their correctness, their practicality and their benefit over standard OWL representations, each showing correct and positive results.

Although  $[x]$  has only been applied to small DLs such as  $\mathcal{EL}$  and  $\mathcal{ALC}$ , they do have positive results, mainly their decidability, and our results provide enough insight into taking  $[x]$  further towards the expressivity of  $\mathcal{SROIQ}(\mathcal{D})$ . Our preliminary results on our implementations show that they can be used effectively in practice and to good use. Since this is ongoing work, we give R15 a score of  $\checkmark$ .

### 5.4.1 Summary

The overall scores for the evaluation of  $[x]$  against the TRs can be seen in Tables 5.3-5.17. We can see from the evaluation that  $[x]$  has three strong points which aids in its strong performance. The first is that its possible world semantics and rigid interpretation helps with identity constraints and modelling change, the second is that the ability to use global and specific time point intervals is a good advantage and the third is the ability to temporalise relations, specifically the localised rigidity we gain helps to capture the most important temporal features. It also has some major downsides, the most prominent being the inability to capture any type of eventuality which makes it suffer heavily when this is needed, for example when modelling Allen's relations, as we saw in TR8 or the loss of entailments regarding continuants or occurrents existence in qualitative environments, as we saw in TR1 and TR2.

In terms of how it compared directly with the other logics we evaluated previously, we consider each TR individually.

TR1 It performed equally as well as  $LTL_{\mathcal{ALC}}$ , out performing both  $\mathcal{ALC}(\mathcal{D})$  and FL. Its equal performance with  $LTL_{\mathcal{ALC}}$  was not surprising since the logics are closely related sharing a similar semantics. Where  $LTL_{\mathcal{ALC}}$  failed with quantification,  $[x]$  gained an advantage. Similarly, where  $[x]$  failed with

eventuality,  $LTL_{\mathcal{AC}}$  gained an advantage. Both had advantages and disadvantages, but we believe they were roughly equal. Neither was perfect however, but both shed lights on different aspects. The high scores were mainly down to the possible world semantics and the rigid interpretation of individuals, paying kindly to the requirements of continuants, which is mainly why  $\mathcal{AC}(\mathcal{D})$  and FL did not score so highly.

TR2 It performed second best for TR2, being outscored by  $\mathcal{AC}(\mathcal{D})$ . The reason it performed better than  $LTL_{\mathcal{AC}}$  was solely because of its ability to better capture the temporal parts, mainly due to its localised rigidity constraints. It still lacked entailments involving eventuality that  $LTL_{\mathcal{AC}}$  does have, but its ability to model temporal parts were seen to be far more important. It lost to  $\mathcal{AC}(\mathcal{D})$  due to its inability to easily capture its existence, for which was the reason  $LTL_{\mathcal{AC}}$  and FL also lost out to  $\mathcal{AC}(\mathcal{D})$ .

TR3 It outperformed each other logic in TR3, getting a strong result directly because of the localised rigidity being able to be captured in the logic. This was arguably the most positive outcome of the logic since this requirement held the most important temporal feature and temporal relation.

TR4 See TR3.

TR5 It outperformed both  $LTL_{\mathcal{AC}}$  and  $\mathcal{AC}(\mathcal{D})$ , and performed equally as well as FL. It outscored  $LTL_{\mathcal{AC}}$  due its ability for quantification and the additional entailments it could gain when combining both the quantitative and qualitative environments, and outscored  $\mathcal{AC}(\mathcal{D})$  due to not needing to reify any relations since we could directly temporalise a relation.

TR6 It came joint second, losing out to  $LTL_{\mathcal{AC}}$ , but out performing  $\mathcal{AC}(\mathcal{D})$ . It lost points primarily due its inability to represent any type of eventuality. This was the only downside when compared to  $LTL_{\mathcal{AC}}$ , but it affected the logic only in the qualitative environment. This was the only downside for which the logic suffered when compared to  $LTL_{\mathcal{AC}}$ .

TR7 Overall, the logic came joint first with  $\mathcal{AC}(\mathcal{D})$ , although more sub requirements were met in  $\mathcal{AC}(\mathcal{D})$  than in  $[x]$ . The reason it outperformed  $LTL_{\mathcal{AC}}$  was primarily due to the fact that it was possible to consider quantification



as well as a qualitative environment. Modelling changes was equally expressible in  $LTL_{\mathcal{ALC}}$  and  $[x]$ , but in  $\mathcal{ALC}(\mathcal{D})$  they were far more manageable which is why the sub requirements outperformed every other logic, but due to the static nature it did not score full marks, which is why overall we gave them equal marks.

TR8 It came joint last with FL. The problem here was the underlying problem of the difficulty of modelling existence which is present in every other requirement and also the problem of eventuality. It could not match the expressive power of  $LTL_{\mathcal{ALC}}$  which had eventuality, let alone the expressive power of  $\mathcal{ALC}(\mathcal{D})$  which had the concrete features. Although in the quantitative setting some of Allen's relations could be simulated, we couldn't see any advantage over the modelling, such as generating new information from our encoding which is why it scored so poorly.

TR9 It outperformed every other logic scoring the highest possible points. Every other logic performed equally well. A combination of three things: the possible world semantics, the ability for quantification and the ability to temporalise relations, gave it an advantage over every other logic.

TR10 It scored equally well with  $LTL_{\mathcal{ALC}}$ , but when compared to the other logics,  $LTL_{\mathcal{ALC}}$  was better off. The quantitative approach could add a nice temporal structure but it lacked in terms of what it could offer in terms of new inferred information. Also, the lack of eventuality played another important role here and it was needed in most scenarios yet again.

TR11 As with TR3 and TR4, it outperformed every other logic and scored full points. This was solely because of the ability to capture localised rigidity, the most important feature.

TR12 It came joint last but this time with  $\mathcal{ALC}(\mathcal{D})$ . The problem here was specifically a direct problem with the semantics due to the fact that a branching time line was simply not something that could be expressed, due to the fact that the logic encoded a discrete linear time line in its semantics. It was outperformed by  $CTL_{\mathcal{ALC}}$  whose semantics allowed for this exact time line structure so this result was unsurprising.

TR13 See TR9.

TR14 See TR11.

R15 It came second, being outscored by FL. FL wins outright, since it will always be as powerful as OWL since it is encoded in OWL as an ontology. Although  $[x]$  has only been evaluated against certain fragments of OWL for which good results have been proved, reasoners have been created which are in active development and the rigidity is not immediately undecidable.

We believe the approach that we took in creating the logic also aided in its success. By borrowing the concept of temporal parts of a single entity from the fluent ontology, adopting the style of semantics from  $LTL_{\mathcal{ALC}}$  and considering the quantitative environment from  $\mathcal{ALC}(\mathcal{D})$ , we were able to capture most of the positive aspects of each logic and combine them into one. By far the most important aspect was that rigid relations could be captured, and more specifically finite durations of rigid relations. Since the relations themselves could now be annotated with intervals, then unlike other approaches we could make the rigidity belong to the axiom in question and not the relation, which gave a lot more flexibility but also controlled restrictions. Also, the idea to use intervals as a representation of sequences of time points, helped to cut down on lengthy axioms and also aided in the representation of durations. We also found that some previously evaluated logics were useful in a qualitative environment and others were useful in a quantitative environment, but there wasn't any option for both other than what we saw in  $\mathcal{ALC}(\mathcal{D})$ . We decided to include both in our logic as we saw that this was needed and it proved to be very useful.

Unfortunately we did not manage to embed any eventuality into the logic at this stage. We were more focused on solving the more important problem of rigidity. Another place the logic suffered was with regards to Allen's relations, particularly with occurrents.  $\mathcal{ALC}(\mathcal{D})$  was really the only approach that could define the constraints of Allen's relations correctly due to the power of concrete domains and their predicates. But when considering the combination of a possible world semantics and a concrete domain, it is difficult to comprehend how they would work together since each temporal dimension is fundamentally different and they do not easily coincide.

Although many issues arise, our goal was to try and keep the logic as simple as possible, because we know that even some of the most simplest temporal features can make a TDL immediately undecidable, and this is something which we are desperately trying to avoid.

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR1	✓	✓	✓	✓
TR1.1	✓	✗	✗	✓
TR1.2	✓	✗	✗	✓
TR1.3	✗	✓	✗	✗

Table 5.3: TR1 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR2	✗	✓✓	✗	✓
TR2.1	✗	✓	✗	✗
TR2.2	✓	✓	✓	✓

Table 5.4: TR2 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

Although  $[x]$  performs better overall than any other temporal extension w.r.t the TRs, it is by no means considered the best TDL. It still suffers from severe problems, that other logics easily solve, these being the lack of eventuality, the difficulties in modelling existence, the inability to constrain Allen's relations and the unknown complexity results. They are not however, *known unsolvable problems*. We do intend to extend the  $[x]$  logics to attempt solve these problems, and even extend the original survey to discover more requirements, pushing the limits of  $[x]$  to see how expressive it can be whilst remaining decidable. We continue this discussion in Chapter 9.

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR3	✗	✗	✗	✓✓
TR3.1	✓	✗	✗	✓
TR3.2	✗	✓	✓	✓

Table 5.5: TR3 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR4	$\times$	$\times$	$\times$	$\checkmark\checkmark$
TR4.1	$\checkmark$	$\times$	$\times$	$\checkmark$
TR4.2	$\times$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.6: TR4 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR5	$\checkmark$	$\times$	$\checkmark\checkmark$	$\checkmark\checkmark$
TR5.1	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.7: TR5 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR6	$\checkmark\checkmark$	$\times$	$\checkmark$	$\checkmark$
TR6.1	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.8: TR6 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR7	$\times$	$\checkmark$	$\times$	$\checkmark$
TR7.1	$\times$	$\checkmark$	$\times$	$\times$
TR7.2	$\times$	$\checkmark$	$\times$	$\times$
TR7.3	$\checkmark$	$\times$	$\times$	$\checkmark$

Table 5.9: TR7 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR8	$\checkmark$	$\checkmark$	$\times$	$\times$
TR8.1	$\checkmark$	$\checkmark$	$\times$	$\times$

Table 5.10: TR8 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR9	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark\checkmark$
TR9.1	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.11: TR9 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR10	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
TR10.1	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
TR10.2	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
TR10.3	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
TR10.4	$\times$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.12: TR10 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR11	$\times$	$\times$	$\times$	$\checkmark\checkmark$
TR11.1	$\checkmark$	$\times$	$\times$	$\checkmark$
TR11.2	$\times$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.13: TR11 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$CTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR12	$\checkmark\checkmark$	$\times$	$\times$	$\times$
TR12.1	$\checkmark$	$\times$	$\checkmark$	$\times$

Table 5.14: TR12 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR13	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark\checkmark$
TR13.1	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.15: TR13 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
TR14	$\times$	$\times$	$\times$	$\checkmark\checkmark$
TR14.1	$\checkmark$	$\times$	$\times$	$\checkmark$
TR14.2	$\times$	$\checkmark$	$\checkmark$	$\checkmark$

Table 5.16: TR14 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$ 

	$LTL_{\mathcal{ALC}}$	$\mathcal{ALC}(\mathcal{D})$	FL	$[x]$
R15	$\times$	$\times$	$\checkmark\checkmark$	$\checkmark$

Table 5.17: R15 Scores evaluated against  $LTL_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL and  $[x]$

# Chapter 6

## Reasoning in Fragments of $[x]$

In this chapter we discuss the reasoning problems and challenges we face in different fragments of  $[x]$ . We discuss how standard DL reasoning tasks can be seen as insufficient for temporal reasoning in certain fragments in  $[x]$  and discuss how new reasoning tasks are needed. We then go on to prove decidability and complexity results for interesting fragments of  $[x]$ .

### 6.1 Extending Classical DL Reasoning Tasks

The standard DL reasoning tasks are satisfiability testing, subsumption, classification and consistency checking [BCM<sup>+</sup>03]. Other reasoning task exists (for example, query answering, realisation, data access, axiom pinpointing, explanation generation etc), but we set these aside for now.

In standard DLs, ontologies are considered to be finite sets of axioms and assertions, and thus there exist only finitely many concept descriptions, and atomic concepts that occur in an ontology. This works well for reasoning procedures such as classification, since their can only be finitely many pairs of atomic subsumptions. A classification usually involves a *finite* number of entailment tests and the inferred concept hierarchy can be represented as a directed acyclic graph. For reasoning procedures such as satisfiability, there is a clear understanding over what it means for a concept to be satisfiable, which happens to be inter-reducible to many of the other reasoning procedures [BCM<sup>+</sup>03], further making it more clear what the purpose of each task is. For some temporal logics this is not necessarily the case. For example consider the class  $A_{[0]_x}$  from  $\mathcal{ALC}_{[x]}$ . If this class

was considered to be satisfiable, since we now have an infinite amount of possible worlds, should satisfiability be interpreted as there being an interpretation  $\mathcal{I}$  where  $A_{[0]_x}^{\mathcal{I}^i}$  is non empty for every  $i \in \mathbb{Z}$ , or is it sufficient to quantify over just a single  $i$  instead? We made the decision when defining the semantics by defining satisfiability of concepts in the qualitative environments to be satisfiable if they have a model at time 0. This was a very important decision to make and it was based mostly on the results of the evaluation in Chapter 4. Generally, we believe that a concept or class need not have a non empty interpretation in *all* worlds for it to be considered satisfiable, especially in cases where we have entities coming into and out of existence.

Classification of  $[x]$  ontologies causes the biggest problem in unbounded infinite environments. In the bounded finite environments, such as  $\mathcal{ALC}_{[]}$ , there are again only finitely many atomic concepts that may occur inside a given ontology, so in terms of a classification, the inferred concept hierarchy will always be of finite size since the number of atomic entailments are also of finite size. But when we consider the unbounded infinite environments, such as  $\mathcal{ALC}_{[x]}$ , a classification is not so simple. We explain why with an example. Consider the following  $\mathcal{ALC}_{[x]}$  ontology:

$$\begin{aligned} \mathcal{O}_1 := \{ & A_{[0]_x} \sqsubseteq B_{[1]_x} \\ & \{B_{[0]_x} \sqsubseteq A_{[1]_x}\} \end{aligned} \quad (6.1)$$

The standard signature of  $\mathcal{O}_1$  is simply the concept names that occur in  $\mathcal{O}_1$ ,  $\{A_{[0]_x}, B_{[1]_x}, B_{[0]_x}, A_{[1]_x}\}$ . Since classical classification is normally only concerned with the concept names occurring directly in the knowledge base, then other than the axioms that appear in  $\mathcal{O}$ , there are not any extra entailments that a classification could provide. However, if we direct our attention away from the classical signature of  $\mathcal{O}_1$ , there is in fact a number of atomic entailments when we focus on the temporal nature of the concepts that appear in  $\mathcal{O}_1$ . Considering the concept  $A_{[0]_x}$ . Its entailed super concepts would in fact include  $\{B_{[1]_x}, A_{[2]_x}, B_{[3]_x}, A_{[4]_x}, B_{[5]_x}, A_{[6]_x} \dots\}$ . Although the concept  $A_{[2]_x}$  does not appear in the ontology itself, the variable  $x$  quantifies over all time points in  $\mathbb{Z}$ , so it can be argued that it still should be considered in a classification since it is a *relevant* temporal entailment of the ontology. It is clear that the standard reasoning procedures are not sufficient for this level of temporal reasoning in  $[x]$ .

For this reason, we introduce several new notions to account for this added temporal expressivity. We redefine what the *signature* of temporal ontologies is, as well as introducing the notion of bounded classification for the unbounded infinite fragments of  $[X]$ , to ensure inferred concept hierarchies remain finite under the notion of what we call a  $k$ -bound classification, where given each atomic concept  $A_{\lambda_x}$  that occurs in an ontology, a  $k$ -bound classification would include all super concepts of  $A_{\lambda_x}$  that were no more than a *temporal distance* of  $k$  away from  $A_{\lambda_x}$ . In the example above, a 4-bound classification would restrict the super concepts of  $A_{[0]_x}$  to be  $\{B_{[1]_x}, A_{[2]_x}, B_{[3]_x}, A_{[4]_x}\}$ . This would ensure that the inferred class hierarchy of an ontology remains finite.

Before defining the signatures of languages of  $[X]$  and then the  $k$ -bound classification, we introduce the notion of a *base version* of a v-interval.

**Definition 40** (V-Interval Base Versions)

Let  $\lambda_x$  be any interval in  $\Lambda_x$ . The interval  $[0, \lambda_x^e - \lambda_x^s]$  is called the **base version** of  $\lambda_x$  and is denoted as  $\dot{\lambda}_x$ .

The base version of a v-interval is the v-interval *shifted to zero*. For example, the base version of the v-interval  $[2, 4]_x$  is  $[0, 2]_x$ , and the base version of  $[-2, -1]_x$  is  $[0, 1]_x$ .

In most of our definitions, we use  $\mathcal{ALC}$  as the base DL to aid our examples. We also do not exceed the expressivity of  $\mathcal{ALC}$  when we go on to prove certain complexity results for  $[X]$ . We could replace  $\mathcal{ALC}$  with another DL, but for the purpose of this thesis, this is not necessary and  $\mathcal{ALC}$  is most suitable.

**Definition 41** (Signatures of languages of  $[X]$ )

- $\mathcal{ALC}_{[]}$  Given an  $\mathcal{ALC}_{[]}$  Ontology  $\mathcal{O}$ , an  $\mathcal{ALC}_{[]} TBox \mathcal{T}$  and an  $\mathcal{ALC}_{[]} ABox \mathcal{A}$ , the signature of each, denoted as  $\tilde{\mathcal{O}}, \tilde{\mathcal{T}}, \tilde{\mathcal{A}}$  respectively, is the set of all concept, role and individual names occurring in each set.
- $\mathcal{ALC}_{[x]}$  Given an  $\mathcal{ALC}_{[x]}$  Ontology  $\mathcal{O}$ , an  $\mathcal{ALC}_{[x]} TBox \mathcal{T}$  and an  $\mathcal{ALC}_{[x]} ABox \mathcal{A}$ , the signature of each, denoted as  $\tilde{\mathcal{O}}, \tilde{\mathcal{T}}, \tilde{\mathcal{A}}$  respectively, is the set of all concept, role and individual names occurring in each set, where for each concept and role name, each v-interval is converted to its base version.
- $\mathcal{ALC}_{[][x]}$  Given an  $\mathcal{ALC}_{[][x]}$  Ontology  $\mathcal{O}$ , an  $\mathcal{ALC}_{[][x]} TBox \mathcal{T}$  and an  $\mathcal{ALC}_{[][x]} ABox \mathcal{A}$ , the signature of each, denoted as  $\tilde{\mathcal{O}}, \tilde{\mathcal{T}}, \tilde{\mathcal{A}}$  respectively, is the set of all concept, role and individual names occurring in each set, where each



variable concept and role name is quantified over each time point between  $\text{MIN}(\mathcal{O})$  and  $\text{MAX}(\mathcal{O})$ .

$\mathcal{ALC}_{[x][\ ]}$  Given an  $\mathcal{ALC}_{[x][\ ]}$  Ontology  $\mathcal{O}$ , an  $\mathcal{ALC}_{[x][\ ]}$  TBox  $\mathcal{T}$  and an  $\mathcal{ALC}_{[x][\ ]}$  ABox  $\mathcal{A}$ , the signature of each, denoted as  $\tilde{\mathcal{O}}, \tilde{\mathcal{T}}, \tilde{\mathcal{A}}$  respectively, is the set of all concept, role and individual names occurring in each set, where for each variable concept and role name, each  $v$ -interval must be converted to its base version.

As an example, the signature of  $\mathcal{O}_1$  from above would be  $\{A_{[0]_x}, B_{[0]_x}\}$

We now go on to define a  $k$ -bound classification.

**Definition 42** ( $k$ -bound classification)

Let  $\mathcal{T}$  be a TBox in a language of  $[x]$  with an infinite time line. A  $k$ -bound classification of  $\mathcal{T}$  is the set of all pairs  $\langle A_{\lambda_x}, B_{\lambda_{x+i}} \rangle$  s.t.  $A_{\lambda_x}, B_{\lambda_x}$  are in  $\tilde{\mathcal{T}}$  and  $\mathcal{T} \models A_{\lambda_x} \sqsubseteq B_{\lambda_{x+i}}$  where  $-k \leq i \leq k$ .

We now go on to define several reasoning problems for each language of  $[x]$ .

### 6.1.1 Reasoning problems in $\mathcal{ALC}_{[\ ]}$

**Definition 43** (Satisfiability)

Let  $C$  be an  $\mathcal{ALC}_{[\ ]}$  concept,  $\mathcal{T}$  an  $\mathcal{ALC}_{[\ ]}$  TBox,  $\mathcal{A}$  an  $\mathcal{ALC}_{[\ ]}$  ABox and let  $\mathcal{O}$  and an  $\mathcal{ALC}_{[\ ]}$  ontology of the form  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ .

$C$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C^{\mathcal{I}}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C^{\mathcal{I}}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{O}$  if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$  where  $C^{\mathcal{I}}$  is non empty.

**Definition 44** (Ontology Consistency)

An  $\mathcal{ALC}_{[\ ]}$  ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  is consistent if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$ .

**Definition 45** (Subsumption)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[\ ]}$  TBox and  $C$  and  $D$  be  $\mathcal{ALC}_{[\ ]}$  concept descriptions.  $C$  is subsumed by  $D$  written  $\mathcal{T} \models C \sqsubseteq D$  if for all models  $\mathcal{I}$  of  $\mathcal{T}$  it holds that  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ .

**Definition 46** (Classification)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[\ ]}$  TBox. A classification of  $\mathcal{T}$  is the set of all pairs  $\langle A_{\lambda}, B_{\lambda} \rangle$  s.t.  $A_{\lambda}, B_{\lambda}$  are in  $\tilde{\mathcal{T}}$  and  $\mathcal{T} \models A_{\lambda} \sqsubseteq B_{\lambda}$ .

### 6.1.2 Reasoning problems in $\mathcal{ALC}_{[]}[x]$

**Definition 47** (Satisfiability)

Let  $C$  be an  $\mathcal{ALC}_{[]}[x]$  concept involving intervals,  $C_x$  be an  $\mathcal{ALC}_{[]}[x]$  concept involving  $v$ -intervals,  $\mathcal{T}$  be an  $\mathcal{ALC}_{[]}[x]$  TBox,  $\mathcal{A}$  an  $\mathcal{ALC}_{[]}[x]$  ABox and let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[]}[x]$  ontology of the form  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ .

$C$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C^{\mathcal{I}}$  is non empty.

$C_x$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C_x^{\mathcal{I}^0}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C^{\mathcal{I}}$  is non empty.

$C_x$  is satisfiable w.r.t  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C_x^{\mathcal{I}^0}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{O}$  if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$  where  $C^{\mathcal{I}}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{O}$  if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$  where  $C_x^{\mathcal{I}^0}$  is non empty.

**Definition 48** (Ontology Consistency)

An  $\mathcal{ALC}_{[]}[x]$  ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  is consistent if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$ .

**Definition 49** (Subsumption)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[]}[x]$  TBox and  $C$  and  $D$  be  $\mathcal{ALC}_{[]}[]$  concept descriptions.  $C$  is subsumed by  $D$  written  $\mathcal{T} \models C \sqsubseteq D$  if for all models  $\mathcal{I}$  of  $\mathcal{T}$  it holds that  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ .

**Definition 50** (Classification)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[]}[x]$  TBox. A classification of  $\mathcal{T}$  is the set of all pairs  $\langle A_\lambda, B_\lambda \rangle$  s.t.  $A_\lambda, B_\lambda$  are named classes over standard intervals in  $\tilde{\mathcal{T}}$  and  $\mathcal{T} \models A_\lambda \sqsubseteq B_\lambda$ .

### 6.1.3 Reasoning problems in $\mathcal{ALC}_{[x]}$

**Definition 51** (Satisfiability)

Let  $C$  be an  $\mathcal{ALC}_{[x]}$  concept,  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x]}$  TBox,  $\mathcal{A}$  an  $\mathcal{ALC}_{[x]}$  ABox and let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[x]}$  ontology of the form  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ .

$C$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C^{\mathcal{I}}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C^{\mathcal{I}^0}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{O}$  if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$  where  $C^{\mathcal{I}^0}$  is non empty.

**Definition 52** (Ontology Consistency)

An  $\mathcal{ALC}_{[x]}$  ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  is consistent if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$ .

**Definition 53** (Subsumption)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x]}$  TBox and  $C_x$  and  $D_x$  be  $\mathcal{ALC}_{[x]}$  concept descriptions.  $C_x$  is subsumed by  $D_x$  written  $\mathcal{T} \models C_x \sqsubseteq D_x$  if for all models  $\mathcal{I}$  of  $\mathcal{T}$ , it holds that  $C^{\mathcal{I}^k} \subseteq D^{\mathcal{I}^k}$  for all  $k \in \mathbb{Z}$ .

**Definition 54** (Classification)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x]}$  TBox and let  $k \geq \max(|\text{MIN}(\mathcal{T})|, |\text{MAX}(\mathcal{T})|)$ . A  $k$ -bound classification of  $\mathcal{T}$  is the set of all pairs  $\langle A_{\lambda_x}, B_{\lambda_{x+i}} \rangle$  s.t.  $A_{\lambda_x}, B_{\lambda_x}$  are in  $\tilde{\mathcal{T}}$  and  $\mathcal{T} \models A_{\lambda_x} \sqsubseteq B_{\lambda_{x+i}}$  where  $-k \leq i \leq k$ .

**6.1.4 Reasoning problems in  $\mathcal{ALC}_{[x][\ ]}$** **Definition 55** (Satisfiability)

Let  $C$  be an  $\mathcal{ALC}_{[x][\ ]}$  concept involving standard intervals,  $C_x$  be an  $\mathcal{ALC}_{[x][\ ]}$  concept involved  $v$ -intervals,  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x][\ ]}$  TBox,  $\mathcal{A}$  an  $\mathcal{ALC}_{[x][\ ]}$  ABox and let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[x][\ ]}$  ontology of the form  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ .

$C$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C^{\mathcal{I}}$  is non empty.

$C_x$  is satisfiable if there exists a model  $\mathcal{I}$  where  $C_x^{\mathcal{I}^0}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C^{\mathcal{I}}$  is non empty.

$C_x$  is satisfiable w.r.t  $\mathcal{T}$  if there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  where  $C_x^{\mathcal{I}^0}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{O}$  if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$  where  $C^{\mathcal{I}}$  is non empty.

$C$  is satisfiable w.r.t  $\mathcal{O}$  if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$  where  $C_x^{\mathcal{I}^0}$  is non empty.

**Definition 56** (Ontology Consistency)

An  $\mathcal{ALC}_{[x][\ ]}$  ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  is consistent if there exists a model  $\mathcal{I}$  of both  $\mathcal{T}$  and  $\mathcal{A}$ .

**Definition 57** (Subsumption)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x][\ ]}$  TBox,  $C_x$  and  $D_x$  be  $\mathcal{ALC}_{[x]}$  concept descriptions and  $C$  and  $D$  be  $\mathcal{ALC}_{[\ ]}$  concept descriptions.  $C_x$  is subsumed by  $D_x$  written  $\mathcal{T} \models C_x \sqsubseteq D_x$  if for all models  $\mathcal{I}$  of  $\mathcal{T}$ , it holds that  $C^{\mathcal{I}^k} \subseteq D^{\mathcal{I}^k}$  for all  $k \in \mathbb{Z}$ .  $C$  is subsumed by  $D$  written  $\mathcal{T} \models C \sqsubseteq D$  if for all models  $\mathcal{I}$  of  $\mathcal{T}$  it holds that  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ .

**Definition 58** (Classification)

Let  $\mathcal{T}$  be an  $\mathcal{ALC}_{[x][\ ]}$  TBox and let  $k \geq \max(|\text{MIN}(\mathcal{T})|, |\text{MAX}(\mathcal{T})|)$ . A  $k$ -bound classification of  $\mathcal{T}$  is the set of all pairs  $\langle A_{\lambda_x}, B_{\lambda_{x+i}} \rangle$  s.t.  $A_{\lambda_x}, B_{\lambda_x}$  are in  $\tilde{\mathcal{T}}$  and  $\mathcal{T} \models A_{\lambda_x} \sqsubseteq B_{\lambda_{x+i}}$  where  $-k \leq i \leq k$ .

We now go on to show decidability and upper complexity bounds for various of the reasoning problems introduced above for languages of  $[x]$  when combined with the DLs  $\mathcal{EL}$  and  $\mathcal{ALC}$ . We show results for  $\mathcal{EL}_{[\ ]}$ ,  $\mathcal{EL}_{[\ ]}[x]$ ,  $\mathcal{ALC}_{[\ ]}$ ,  $\mathcal{ALC}_{[x]}$ ,  $\mathcal{ALC}_{[\ ]}[x]$  and a restricted form of  $\mathcal{EL}_{[x]}$ . All decision procedures assume a constant domain restriction. The decidability of the remaining languages remains unknown.

## 6.2 A Decision Procedure for Classification in $\mathcal{EL}_{[\ ]}$

$\mathcal{EL}$  is a light weight DL, well known for its impressive expressiveness with its few logics operators and polynomial time complexity [Bra04a, BBL05].  $\mathcal{EL}$  was formally introduced in Chapter 2. We go on to define the syntax and semantics of  $\mathcal{EL}_{[\ ]}$  before describing a decision procedure for classification.

### $\mathcal{EL}_{[\ ]}$ Syntax

$\mathcal{EL}_{[\ ]}$  concept descriptions are a restriction of  $\mathcal{ALC}_{[\ ]}$  concept descriptions where we only allow for the  $\sqcap, \sqcup$  logical operators.

**Definition 59** ( $\mathcal{EL}_{[\ ]}$  Concept Descriptions)

Let  $\lambda \in \Lambda$  be an arbitrary interval,  $A_\lambda \in N_{con}^{[\ ]}$  an atomic concept,  $R_\lambda \in N_{role}^{[\ ]}$  an atomic role and  $C, D$  arbitrary concept descriptions. Then concept descriptions are formed in  $\mathcal{EL}_{[\ ]}$  according to the following syntax rule:

$$C, D \longrightarrow \top \mid A_\lambda \mid C \sqcap D \mid \exists R_\lambda.C$$

### $\mathcal{EL}_{[\ ]}$ Semantics

The semantics is defined in the usual way.

### $\mathcal{EL}_{[\ ]}$ TBox, ABox & Ontology

TBoxes, ABoxes and Ontologies are defined in the usual way.

## Reasoning

In standard  $\mathcal{EL}$ , the standard reasoning problem is classification, i.e. deciding for a given ontology  $\mathcal{O}$ , whether  $\mathcal{O} \models A \sqsubseteq B$  for all concept names  $A, B$  occurring in  $\mathcal{O}$ . Satisfiability and ontology consistency are trivial in  $\mathcal{EL}$  and thus  $\mathcal{EL}_{[]}$  since  $\mathcal{EL}$  does not allow any negation, and also not  $\perp$ . Therefore every concept in  $\mathcal{EL}_{[]}$  is automatically satisfiable and every ontology is also automatically consistent since one can easily find a trivial model by setting the interpretation of each concept and role name to the full domain in every time point. So we focus only on the reasoning problem of classification.

Before showing a decision procedure to compute a classification, we rely on an  $\mathcal{EL}_{[]}$  TBox to be in normal form. We first show how this can be computed in both polynomial time and space. This normal form is similar to those seen in [BBL05, Bra04b].

## Normalisation

**Definition 60** (Normalised  $\mathcal{EL}_{[]}$  TBox)

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}$  TBox over the set  $N_{con}^{[]}$  and  $N_{role}^{[]}$ .  $\mathcal{T}$  is normalised if  $\mathcal{T}$  contains only axioms of the form:

$$\begin{aligned} A_{\lambda^1} &\sqsubseteq B_{\lambda^2} \\ A_{\lambda^1} \sqcap B_{\lambda^2} &\sqsubseteq C_{\lambda^3} \\ A_{\lambda^1} &\sqsubseteq \exists R_{\lambda^2}. B_{\lambda^3} \\ \exists R_{\lambda^1}. A_{\lambda^2} &\sqsubseteq B_{\lambda^3} \end{aligned}$$

where  $A_{\lambda^i}, B_{\lambda^i}, C_{\lambda^i}$  are atomic concepts from  $N_{con}^{[]}$  and  $R_{\lambda^i}$  is a role name from  $N_{role}^{[]}$ .

The following definition shows how any  $\mathcal{EL}_{[]}$  TBox can be transformed into a normalised version using a set of normalisation rules.

**Definition 61** (Normalisation Rules for an  $\mathcal{EL}_{[]}$  TBox)

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}$  TBox over the set  $N_{con}^{[]}$  and  $N_{role}^{[]}$ . For any concept descriptions  $C, D, E$  over  $N_{con}^{[]}$ , any role  $R_{\lambda^2}$  over  $N_{role}^{[]}$  and any complex concept descriptions  $\hat{C}$  and  $\hat{D}$  (that are not concept names) over  $N_{con}^{[]}$  and  $N_{role}^{[]}$ , the rules are defined as follows:

$R$	$G \longrightarrow G'$
$NF1$	$C \equiv D \longrightarrow \{C \sqsubseteq D, D \sqsubseteq C\}$
$NF2$	$\hat{C} \sqcap D \sqsubseteq E \longrightarrow \{ \hat{C} \sqsubseteq A_{\lambda^1}, A_{\lambda^1} \sqcap D \sqsubseteq E \}$
$NF3$	$C \sqcap \hat{D} \sqsubseteq E \longrightarrow \{ \hat{D} \sqsubseteq A_{\lambda^1}, A_{\lambda^1} \sqcap C \sqsubseteq E \}$
$NF4$	$\exists R_{\lambda^2}.\hat{C} \sqsubseteq D \longrightarrow \{ \hat{C} \sqsubseteq A_{\lambda^1}, \exists R_{\lambda^2}.A_{\lambda^1} \sqsubseteq D \}$
$NF5$	$\hat{C} \sqsubseteq \hat{D} \longrightarrow \{ \hat{C} \sqsubseteq A_{\lambda^1}, A_{\lambda^1} \sqsubseteq \hat{D} \}$
$NF6$	$C \sqsubseteq \exists R_{\lambda^2}.\hat{C} \longrightarrow \{C \sqsubseteq \exists R_{\lambda^2}.A_{\lambda^1}, A_{\lambda^1} \sqsubseteq \hat{C} \}$
$NF7$	$C \sqsubseteq D \sqcap E \longrightarrow \{C \sqsubseteq D, C \sqsubseteq E\}$

where  $A_{\lambda^1}$  is a **fresh** concept name not occurring in  $\mathcal{T}$  and  $\lambda^1 = [0, 0]$ . Each rule has the form  $R : G \longrightarrow G'$ , where  $R$  is the rule name,  $G$  (LHS) is the input axiom and  $G'$  (RHS) are the resulting axioms.

When a rule  $R$  is applied to  $\mathcal{T}$ , it is changed to  $\mathcal{T}' := \mathcal{T} \setminus \{G\} \cup G'$ . The normalised TBox  $NF(\mathcal{T})$  is created by first exhaustively applying rules  $NF1$  -  $NF4$  (Step 1) in ascending order to any GCI conforming to the structure of the rule, and then applying rules  $NF5$  -  $NF7$  (Step 2) in a similar manner. It is important to note that when moving from Step 1 to Step 2, no rules in Step 2 can generate new axioms that could be applicable to rules in the previous step. This can easily be seen by observing the RHS of the rules.

**Theorem 1** (Computing  $NF(\mathcal{T})$  takes polynomial time)

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}$  TBox.  $NF(\mathcal{T})$  can be computed in polynomial time w.r.t the size of  $\mathcal{T}$  and  $NF(\mathcal{T})$  is of polynomial size w.r.t the size of  $\mathcal{T}$ .

**Proof** Consider the first step involving rules  $NF1$ - $NF4$ . Applying any of these rules once to an appropriate axiom only increases the size of  $\mathcal{T}$  linearly. Consider  $NF1$ . The input axiom (LHS) has length

$$|C| + |D| + 1 \quad (6.2)$$

where 1 represents the number of additional symbols needed to encode the rule, i.e.  $\{\equiv\}$ . When the rule fires, the new axioms (RHS) have length

$$(2 \cdot |C|) + (2 \cdot |D|) + 2 \quad (6.3)$$

showing that  $\mathcal{T}$  is increased only linearly by single application of rule  $NF1$ . Notice that this rule can not be triggered by a result of any other rule, as no other rule

Rule	LHS	RHS	Remarks
<i>NF1</i>	$ C  +  D  + 1$	$(2 \cdot  C ) + (2 \cdot  D ) + 2$	Linear
<i>NF2</i>	$ \hat{C}  +  D  +  E  + 2$	$ \hat{C}  +  D  +  E  + 5$	Constant
<i>NF3</i>	$ \hat{D}  +  C  +  E  + 2$	$ \hat{D}  +  C  +  E  + 5$	Constant
<i>NF4</i>	$ \hat{C}  +  D  +  R_{\lambda^2}  + 2$	$ \hat{C}  +  D  +  R_{\lambda^2}  + 8$	Constant
<i>NF5</i>	$ \hat{C}  +  \hat{D}  + 1$	$ \hat{C}  +  \hat{D}  + 4$	Constant
<i>NF6</i>	$ C  +  \hat{C}  +  R_{\lambda^2}  + 3$	$ \hat{C}  +  C  +  R_{\lambda^2}  + 8$	Constant
<i>NF7</i>	$ C  +  D  +  E  + 2$	$(2 \cdot  C ) +  D  +  E  + 2$	Constant

Table 6.1: Growth of an  $\mathcal{EL}_{[]}$  TBox after single application of normalisation rules

adds axioms of this form, hence *NF1* can only be fired once per axiom in  $\mathcal{T}$ , thus exhaustive application of this rule only requires linear time. The affected size of the TBox after application of the rules can be seen in Table 6.1. Consider *NF2*. A single application of this rule only increases the size of the  $\mathcal{T}$  by a constant (see Table 6.1) and splits the input axiom  $G$  into two new axioms contained in  $G'$ , which contains axioms of the form  $\{\hat{C} \sqsubseteq A, A \sqcap D \sqsubseteq E\}$ . The first axiom may be subject to another application of *NF2* as it could have the form  $\hat{C} \sqcap D \sqsubseteq E$ , hence the number of times *NF2* is applicable as result of the rule firing is dependant on the number of ' $\sqcap$ 's in the input axiom. The same reasoning applies to *NF3*, and also *NF4* where the number of times the rule is applicable is dependant on number of ' $\exists$ 's in the input axiom. Exhaustive application of the rules from *NF2* to *NF4* only increases the size of the TBox by a constant and the number of times these rules can fire is limited by the occurrence of specific constructs on the LHS of these axioms, hence exhaustive application of the rules in Step 1 takes linear time and requires linear space w.r.t to the size of the axioms, which is bounded by the size of  $\mathcal{T}$ . Consider the second step involving rules *NF5* to *NF7*. A single application of *NF5* only increases the size of  $\mathcal{T}$  by a constant (see Table 6.1) and is applicable at most once per resulting axiom. A single application of rule *NF6* also increases the size of  $\mathcal{T}$  by a constant and similar to *NF4*, the number of times the rule is applicable is dependant on number of ' $\exists$ 's in the input axiom. A single application of rule *NF7* also increases the size of  $\mathcal{T}$  by a constant and similar to *NF2* and *NF3*, the number of times the rule is applicable is dependant on number of ' $\sqcap$ 's in the input axiom. Exhaustive application of the rules from *NF5* to *NF7* only increase the size of the TBox by a constant and the number of times these rules can fire is limited by the occurrence of specific constructs on the LHS of these axioms, hence exhaustive application of the rules in Step 2 also

takes linear time and requires linear space w.r.t to the size of  $\mathcal{T}$ .  $\square$

**Proposition 1**

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}$  TBox and let  $NF(\mathcal{T})$  be its normalised version. Then

- (i)  $\tilde{\mathcal{T}} \subseteq \widetilde{NF(\mathcal{T})}$
- (ii) For any model  $\mathcal{I}$  of  $NF(\mathcal{T})$ ,  $\mathcal{I}$  is a model of  $\mathcal{T}$
- (iii) For any model  $\mathcal{I}$  of  $\mathcal{T}$ , there is a model  $\mathcal{I}'$  of  $NF(\mathcal{T})$  s.t.  $\mathcal{I} \equiv \mathcal{I}'_{\tilde{\mathcal{T}}}$

**Proof** (i) Trivial since no rule ever removes concept names - concept names are only added and addition/removal of role names does not exist. By simple inspection of the rules, one can see that for any axiom involving existential constructs, the role name including the construct is added back as a resulting GCI without any change of the role name.

(ii) Suppose  $\mathcal{I}$  is a model on  $NF(\mathcal{T})$ . Consider how  $NF(\mathcal{T})$  was built. Every complex concept description occurring in an axiom was replaced with a fresh concept name, and the axiom was split into two. The fresh concept name acted as a *concept filler* for the two new axioms, and because of the transitivity of subsumption, no information was lost - it was only added. Since (i) holds, it also holds that  $\mathcal{I}$  is a model of  $\mathcal{T}$  where we simply ignore the sets  $A_{\lambda}^{\mathcal{I}}$  where  $A'_{\lambda}$  is a fresh concept.

(iii) Let  $\mathcal{I}$  be a model of  $\mathcal{T}$ . We can build a model  $\mathcal{I}'$  for  $NF(\mathcal{T})$  that is equivalent to  $\mathcal{I}$  when restricted to  $\tilde{\mathcal{T}}$  by setting  $\Delta^{\mathcal{I}} = \Delta^{\mathcal{I}'}$ ,  $C^{\mathcal{I}} = C^{\mathcal{I}'}$  for all  $C \in N_{con}^{\mathcal{T}}$ , and  $A_{\lambda}^{\mathcal{I}} = C^{\mathcal{I}}$  where  $A'_{\lambda}$  is a fresh concept occurring in  $NF(\mathcal{T})$  and  $C$  is the concept that was replaced in the original axiom by  $A'$ . By using similar reasoning in (ii), as the subsumption relationships are preserved with the addition of the fresh axioms,  $\mathcal{I}'$  is a model of  $NF(\mathcal{T})$ , and is equivalent to  $\mathcal{I}$  when restricted over  $\tilde{\mathcal{T}}$ .  $\square$

**Corollary 1**

$NF(\mathcal{T})$  is entailment preserving.

From this point onwards, we assume any  $\mathcal{EL}_{[]}$  TBox is in normal form.

**Some Useful Definitions**

Let  $\lambda^1, \lambda^2$  be intervals.



- $\lambda^1$  is contained within  $\lambda^2$ , written as  $\lambda^1 \rightsquigarrow_c \lambda^2$  if  $\lambda_s^2 \leq \lambda_s^1 \leq \lambda_e^1 \leq \lambda_e^2$ .
- $\lambda^1$  touches  $\lambda^2$ , written as  $\lambda^1 \rightsquigarrow_t \lambda^2$  if  $\lambda_s^2 \leq \lambda_s^1 \leq \lambda_e^2$  or  $\lambda_s^2 \leq \lambda_e^1 \leq \lambda_e^2$  or  $\lambda^2 \rightsquigarrow_c \lambda^1$ .

Note that  $\lambda_1 \rightsquigarrow_c \lambda_2 \Rightarrow \lambda_1 \rightsquigarrow_t \lambda_2$  and  $\lambda_1 \rightsquigarrow_t \lambda_2 \Rightarrow \lambda_2 \rightsquigarrow_t \lambda_1$ .

### $\mathfrak{D}$ - Decision Procedure

Having shown that normalisation of any  $\mathcal{EL}_{[]} \text{ TBox}$  can be computed in both polynomial time and space, we now show how we can compute a classification for an  $\mathcal{EL}_{[]} \text{ TBox}$ . Using a similar approach to [Bra04b], we compute a classification by building a set  $S_*(A_\lambda)$  (a subsumer set for  $A_\lambda$ ) for every  $A_\lambda \in \tilde{\mathcal{T}}$  for some given TBox  $\mathcal{T}$ , for which each set contains all concept names in  $\tilde{\mathcal{T}}$  that are subsumers of  $A_\lambda$ , i.e., the subsumer sets encode the classification. The subsumer sets are defined as follows:

**Definition 62** (Subsumer Sets  $S_*(A_\lambda)$ )

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]} \text{ TBox}$  (normalized). For every  $A_\lambda \in \tilde{\mathcal{T}}$ , and every  $i \in \mathbb{N}$ , the subsumer set  $S_i(A_\lambda)$  is defined inductively by first applying rule **INIT0**, then for every  $i \geq 0$ ,  $S_{i+1}(A_\lambda)$  is defined by extending  $S_i(A_\lambda)$  by exhaustive application of rules **CR0-CR4**. The subsumer set  $S_*(A_\lambda)$ , is defined as the union of  $\bigcup_{i \geq 0} S_i(A_\lambda)$ .  $S_i(\cdot)$  is complete if no more rules are applicable for any subsumer set.

- **INIT0**  $S_0(A_{\lambda^1}) := \{\top, A_{\lambda^1}\}$  for every  $A_{\lambda^1} \in \tilde{\mathcal{T}}$
- **CR0** If  $A_{\lambda^3} \in S_i(C_{\lambda^1})$  and  $D_{\lambda^2} \in S_i(A_{\lambda^4})$  where  $\lambda^4 \rightsquigarrow_c \lambda^3$  and  $D_{\lambda^2} \notin S_i(C_{\lambda^1})$  then  $S_{i+1}(C_{\lambda^1}) := S_i(C_{\lambda^1}) \cup \{D_{\lambda^2}\}$
- **CR1** If  $A_{\lambda^1} \sqsubseteq B_{\lambda^2} \in \mathcal{T}$  and  $A_{\lambda^1} \in S_i(C_{\lambda^3})$  and  $B_{\lambda^2} \notin S_i(C_{\lambda^3})$  then  $S_{i+1}(C_{\lambda^3}) := S_i(C_{\lambda^3}) \cup \{B_{\lambda^2}\}$ .
- **CR2** If  $A_{\lambda^1} \sqcap B_{\lambda^2} \sqsubseteq C_{\lambda^3} \in \mathcal{T}$  and  $A_{\lambda^1}, B_{\lambda^2} \in S_i(D_{\lambda^4})$  and  $C_{\lambda^3} \notin S_i(D_{\lambda^4})$  then  $S_{i+1}(D_{\lambda^4}) := S_i(D_{\lambda^4}) \cup \{C_{\lambda^3}\}$ .
- **CR3** If  $A_{\lambda^1} \in S_i(B_{\lambda^2})$  and  $A_{\lambda^1} \sqsubseteq \exists R_{\lambda^3}. C_{\lambda^4} \in \mathcal{T}$  and  $D_{\lambda^5} \in S_i(C_{\lambda^4})$  and  $\exists R_{\lambda^6}. D_{\lambda^5} \sqsubseteq E_{\lambda^7} \in \mathcal{T}$  and  $\lambda^6 \rightsquigarrow_c \lambda^3$  and  $E_{\lambda^7} \notin S_i(B_{\lambda^2})$  then  $S_{i+1}(B_{\lambda^2}) := S_i(B_{\lambda^2}) \cup \{E_{\lambda^7}\}$ .

- CR4 If  $B_{\lambda^1} \in S_i(C_{\lambda'})$  and  $C_{\lambda^3} \in S_i(A_{\lambda^2})$ ,  $C_{\lambda^4} \in S_i(A_{\lambda^2})$ , ...,  $C_{\lambda^n} \in S_i(A_{\lambda^2})$ , and  $\lambda^3 \rightsquigarrow_t \lambda^4 \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^n$  and  $\lambda' \rightsquigarrow_c [\lambda_s^3, \lambda_e^n]$  and  $B_{\lambda^1} \notin S_i(A_{\lambda^2})$  then  $S_{i+1}(A_{\lambda^2}) = S_i(A_{\lambda^2}) \cup \{B_{\lambda^1}\}$ .

### Theorem 2

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}$  TBox (normalized). For every  $F_{\lambda^1}, G_{\lambda^2} \in \tilde{\mathcal{T}}$ , it holds that  $G_{\lambda^2} \in S_*(F_{\lambda^1})$  iff  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$ .

**Proof** We first show the  $\Rightarrow$  direction by induction on  $n$  over  $S_n$ .

**Claim:**  $G_{\lambda^2} \in S_*(F_{\lambda^1}) \Rightarrow \mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$

**n = 0:**

- INIT0: If INIT0 added  $G_{\lambda^2}$  to  $S_n(F_{\lambda^1})$ , then either  $G_{\lambda^2} = F_{\lambda^1}$  or  $G_{\lambda^2} = \top$ . Clearly, any TBox  $\models A_{\lambda^1} \sqsubseteq \top$  and  $\models A_{\lambda^1} \sqsubseteq A_{\lambda^1}$ , proving the claim holds.

**n > 0:**

- CR0: If CR0 added  $G_{\lambda^2}$  to  $S_n(F_{\lambda^1})$  then there must exist a concept  $A_{\lambda^3} \in S_{n-1}(F_{\lambda^1})$  and  $G_{\lambda^2} \in S_{n-1}(A_{\lambda^4})$  where  $\lambda^4 \rightsquigarrow_c \lambda^3$  and  $G_{\lambda^2} \notin (F_{\lambda^1})$ . By induction hypothesis it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq A_{\lambda^3}$  and  $\mathcal{T} \models A_{\lambda^4} \sqsubseteq G_{\lambda^2}$ . By definition of the semantics, we know that  $\mathcal{T} \models A_{\lambda^3} \sqsubseteq A_{\lambda^4}$ . By transitivity of subsumption it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$ , proving the claim holds.
- CR1: If CR1 added  $G_{\lambda^2}$  to  $S_n(F_{\lambda^1})$ , then there exists a concept name  $A_{\lambda} \in S_{n-1}(F_{\lambda^1})$ , an axiom  $A_{\lambda} \sqsubseteq G_{\lambda^2} \in \mathcal{T}$  and  $G_{\lambda^2} \notin S_{n-1}(F_{\lambda^1})$ . By induction hypothesis it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq A_{\lambda}$ . By transitivity of subsumption it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$ , proving the claim holds.
- CR2: If CR2 added  $G_{\lambda^2}$  to  $S_n(F_{\lambda^1})$ , then there exists two concept names  $\{A_{\lambda^3}, B_{\lambda^4}\} \in S_{n-1}(F_{\lambda^1})$ , a GCI axiom  $A_{\lambda^3} \sqcap B_{\lambda^4} \sqsubseteq G_{\lambda^2} \in \mathcal{T}$  and  $G_{\lambda^2} \notin S_{n-1}(F_{\lambda^1})$ . By induction hypothesis it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq A_{\lambda^3}$  and  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq B_{\lambda^4}$ . By transitivity of subsumption it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$  proving the claim.
- CR3: If CR3 added  $G_{\lambda^2}$  to  $S_n(F_{\lambda^1})$ , then there exists 2 concept names  $A_{\lambda^3} \in S_{n-1}(F_{\lambda^1})$  and  $B_{\lambda^5} \in S_{n-1}(C_{\lambda^4})$ , two GCI axioms  $A_{\lambda^3} \sqsubseteq \exists R_{\lambda^7}.C_{\lambda^4} \in \mathcal{T}$  and  $\exists R_{\lambda^6}.B_{\lambda^5} \sqsubseteq G_{\lambda^2} \in \mathcal{T}$  and  $\lambda^6 \rightsquigarrow_c \lambda^7$  and finally  $G_{\lambda^2} \notin S_{n-1}(F_{\lambda^1})$ . By definition of the semantics it holds that  $\mathcal{T} \models R_{\lambda^7} \sqsubseteq R_{\lambda^6}$  and by induction hypothesis it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq A_{\lambda^3}$  and  $\mathcal{T} \models C_{\lambda^4} \sqsubseteq B_{\lambda^5}$ . Since  $\mathcal{T} \models$

$A_{\lambda^3} \sqsubseteq \exists R_{\lambda^7}.C_{\lambda^4}$  and  $\mathcal{T} \models \exists R_{\lambda^6}.B_{\lambda^5} \sqsubseteq G_{\lambda^2}$ , by transitivity of subsumption (and abuse of notation) we have  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq \exists R_{\lambda^6}.B_{\lambda^5} \sqsubseteq G_{\lambda^2}$ , showing  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$ , and proving the claim holds.

- **CR4:** If CR4 added  $G_{\lambda^2}$  to  $S_n(F_{\lambda^1})$  then there exists a concept  $C_{\lambda'}$  where  $G_{\lambda^2} \in S_{n-1}(C_{\lambda'})$  and there exists concept  $C_{\lambda^3}, C_{\lambda^4}, \dots, C_{\lambda^n}$  where  $\lambda^3 \rightsquigarrow_t \lambda^4 \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^n$  and  $\lambda' \rightsquigarrow_c [\lambda_s^3, \lambda_e^n]$  and  $C_{\lambda^3} \in S_{n-1}(F_{\lambda^1})$ ,  $C_{\lambda^4} \in S_{n-1}(F_{\lambda^1})$ , ...,  $C_{\lambda^n} \in S_{n-1}(F_{\lambda^1})$ . By IH, it holds that,  $\mathcal{T} \models C_{\lambda'} \sqsubseteq G_{\lambda^2}$ ,  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq C_{\lambda^3}$ ,  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq C_{\lambda^4}$ ,  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq \dots$ ,  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq C_{\lambda^n}$ . By definition of the semantics it holds that  $\mathcal{T} \models C_{\lambda^3} \sqcap C_{\lambda^4} \sqcap \dots \sqcap C_{\lambda^n} \sqsubseteq C_{\lambda'}$  and thus by transitivity of subsumption it holds that  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$ , proving the claim holds.

Next, we show  $\mathcal{T} \models F_{\lambda^1} \sqsubseteq G_{\lambda^2} \Rightarrow G_{\lambda^2} \in S_*(F_{\lambda^1})$  by proving the contraposition:

**Claim:**  $G_{\lambda^2} \notin S_*(F_{\lambda^1}) \Rightarrow \mathcal{T} \not\models F_{\lambda^1} \sqsubseteq G_{\lambda^2}$  where we build a canonical model  $\mathcal{I}$  of  $\mathcal{T}$  with a witness  $x \in F_{\lambda^1}^{\mathcal{I}} \setminus G_{\lambda^2}^{\mathcal{I}}$ . We construct the canonical model  $\mathcal{I}$  according to the following definition: Let  $\text{MIN}(\mathcal{T})$  and  $\text{MAX}(\mathcal{T})$  be defined as usual. Let  $\mathcal{I}^0$  be  $\Delta^{\mathcal{I}^0} = \{a_\lambda | A_\lambda \in \tilde{\mathcal{T}}\}$  and  $A_{\lambda^j}^{\mathcal{I}^0} = \{a_\lambda | \lambda_s \leq j \leq \lambda_e\}$  for all  $l$  where  $\text{MIN}(\mathcal{T}) \leq l \leq \text{MAX}(\mathcal{T})$ . For each  $i \in \mathbb{N}$ , let  $\mathcal{I}^{i+1}$  be the result of exhaustive application of the following rules:

- I1: If  $A_{\lambda^1} \sqsubseteq B_{\lambda^2} \in \mathcal{T}$  then for every individual  $x \in A_{\lambda^1}^{\mathcal{I}^i}$  where  $x \notin B_{\lambda^2}^{\mathcal{I}^i}$ , add  $x$  to  $B_{\lambda^2}^{\mathcal{I}^{i+1}}$
- I2: If  $A_{\lambda^1} \sqcap B_{\lambda^2} \sqsubseteq C_{\lambda^3} \in \mathcal{T}$  then for every individual  $x \in A_{\lambda^1}^{\mathcal{I}^i} \cap B_{\lambda^2}^{\mathcal{I}^i}$  where  $x \notin C_{\lambda^3}^{\mathcal{I}^i}$ , add  $x$  to  $C_{\lambda^3}^{\mathcal{I}^{i+1}}$
- I3: If  $A_{\lambda^1} \sqsubseteq \exists R_{\lambda^2}.B_{\lambda^3} \in \mathcal{T}$  then for every individual  $x \in A_{\lambda^1}^{\mathcal{I}^i}$  where there is no individual  $y \in B_{\lambda^3}^{\mathcal{I}^i}$  where  $(x, y) \in R_{\lambda^2}^{\mathcal{I}^i}$ , add  $(x, y)$  to  $R_{\lambda^2}^{\mathcal{I}^{i+1}}$  where  $y \in B_{\lambda^3}^{\mathcal{I}^i}$ .
- I4: If  $\exists R_{\lambda^1}.A_{\lambda^2} \sqsubseteq B_{\lambda^3} \in \mathcal{T}$  then for every individual  $x \in R_{\lambda^1}.A_{\lambda^2}^{\mathcal{I}^i}$  where  $x \notin B_{\lambda^3}^{\mathcal{I}^i}$ , add  $x$  to  $B_{\lambda^3}^{\mathcal{I}^{i+1}}$ .

The rules are applied exhaustively.  $\mathcal{I}$  is then defined as the infinite union of each  $\mathcal{I}^i$ , i.e.  $\mathcal{I} := \bigcup_{i=0}^{\infty} \mathcal{I}^i$ . We first show that  $\mathcal{I}$  is in fact a model of  $\mathcal{T}$ . Since  $\mathcal{T}$  is normalised, it suffices to show that  $\mathcal{I}$  is a model for each of the four possible axioms in  $\mathcal{T}$ .

- If  $A_{\lambda^1} \sqsubseteq B_{\lambda^2} \in \mathcal{T}$  then  $A_{\lambda^1}^{\mathcal{I}} \subseteq B_{\lambda^2}^{\mathcal{I}}$ . Suppose  $x \in A_{\lambda^1}^{\mathcal{I}^m}$  and  $x \notin B_{\lambda^2}^{\mathcal{I}^n}$  for some  $0 \leq m \leq n-1$ . By definition of  $I1$ ,  $x \in B_{\lambda^2}^{\mathcal{I}^{m+1}}$ , disproving the assumption, and thus proving the claim holds.
- If  $A_{\lambda^1} \sqcap B_{\lambda^2} \sqsubseteq C_{\lambda^3} \in \mathcal{T}$  then  $A_{\lambda^1}^{\mathcal{I}} \cap B_{\lambda^2}^{\mathcal{I}} \subseteq C_{\lambda^3}^{\mathcal{I}}$ . Suppose  $x \in A_{\lambda^1}^{\mathcal{I}^m}$  and  $x \in B_{\lambda^2}^{\mathcal{I}^m}$  and  $x \notin C_{\lambda^3}^{\mathcal{I}^n}$  for some  $0 \leq m \leq n-1$ . By definition of  $I2$ ,  $x \in C_{\lambda^3}^{\mathcal{I}^{m+1}}$ , disproving the assumption, and thus proving the claim holds.
- If  $A_{\lambda^1} \sqsubseteq \exists R_{\lambda^2}.B_{\lambda^3} \in \mathcal{T}$  then  $A_{\lambda^1}^{\mathcal{I}} \subseteq (\exists R_{\lambda^2}.B_{\lambda^3})^{\mathcal{I}}$ . Suppose  $x \in A_{\lambda^1}^{\mathcal{I}^m}$  and there does not exist a  $y$  where  $(x, y) \in R_{\lambda^2}^{\mathcal{I}^n}$  and  $y \in B_{\lambda^3}^{\mathcal{I}^n}$  for some  $0 \leq m \leq n-1$ . By definition of  $I3$  and initialisation of  $\mathcal{I}$ ,  $x \in (\exists R_{\lambda^2}.B_{\lambda^3})^{\mathcal{I}^{m+1}}$ , disproving the assumption, and thus proving the claim holds.
- If  $\exists R_{\lambda^1}.A_{\lambda^2} \sqsubseteq B_{\lambda^3}$  then,  $(\exists R_{\lambda^1}.A_{\lambda^2})^{\mathcal{I}^n} \subseteq B_{\lambda^3}^{\mathcal{I}^n}$ . Suppose  $x \in (\exists R_{\lambda^1}.A_{\lambda^2})^{\mathcal{I}^m}$  and  $x \notin B_{\lambda^3}^{\mathcal{I}^n}$  for some  $0 \leq m \leq n-1$ . By definition of  $I3$ ,  $x \in B_{\lambda^3}^{\mathcal{I}^{m+1}}$ , disproving the assumption, and thus proving the claim holds.

Having shown  $\mathcal{I}$  to be a model of  $\mathcal{T}$ , it remains to show that  $G \notin S_*(F) \rightarrow F^{\mathcal{I}} \not\subseteq G^{\mathcal{I}}$ . To this end we show that for every  $n \in \mathbb{N}$  and every  $A_{\lambda^1}, B_{\lambda^2} \in \tilde{\mathcal{T}}$  where  $A_{\lambda^1} \neq B_{\lambda^2}$ ,  $\forall x_{\lambda} \in A_{\lambda^1}^{\mathcal{I}^n}$ , if  $x_{\lambda}$  was born for  $A_{\lambda^1}$  at  $t$ , i.e.  $x_{\lambda} = a_{\lambda^1}$ , and  $x_{\lambda} \in B_{\lambda^2}^{\mathcal{I}^n}$  then  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ . We show this by proof of induction over  $n - t$ .

**n - t = 0:**

Since  $A_{\lambda^1} \neq B_{\lambda^2}$  then there are only two cases:

- $B_{\lambda^2} = \top$ : By non-applicability (NA) of the **INIT0** rule,  $\top \in S_*(A_{\lambda^1})$ , proving the claim holds.
- $B = A$  where  $\lambda^2 \rightsquigarrow_c \lambda^1$ . By NA of **INIT0** and **CR0**, it holds that  $B_{\lambda^2}^2 \in S_*(A_{\lambda^1})$ , proving the claim holds.

**n - t > 0:**

Let  $x_{\lambda} \in B_{\lambda^2}^{\mathcal{I}^n} \setminus B_{\lambda^2}^{\mathcal{I}^{n-1}}$ . Then one of the rules  $I1$ ,  $I2$  or  $I4$  was responsible for this:

- $I1$ : Then there is a GCI of the form  $C_{\lambda^3} \sqsubseteq B_{\lambda^4}$  where  $x_{\lambda} \in C_{\lambda^3}^{\mathcal{I}^{n-1}}$ . By IH it holds that  $C_{\lambda^3} \in S_*(A_{\lambda^1})$ . By NA of **CR1** it holds that  $B_{\lambda^4} \in S_*(A_{\lambda^1})$ .  $x_{\lambda}$  is in  $B_{\lambda^2}^{\mathcal{I}^n}$  by one of three scenarios:

i)  $B_{\lambda^4} = B_{\lambda^2}$ : then  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.

- ii)  $\lambda^2 \rightsquigarrow_c \lambda^4$ : then by NA of CR0, it holds that  $B_{\lambda^2} \in S_*(B_{\lambda^4})$ , and by NA of CR0, it holds that  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
  - iii)  $\lambda^2 \rightsquigarrow_c [i, j]$ : where there exists  $B_{\lambda'}, B_{\lambda''}, \dots, B_{\lambda^*}$  and  $\lambda' \rightsquigarrow_t \lambda'' \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^4 \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^*$  and  $x_\lambda \in B_{\lambda'}^{\mathcal{I}^{n-1}} \cap B_{\lambda''}^{\mathcal{I}^{n-1}} \cap \dots \cap B_{\lambda^4}^{\mathcal{I}^n} \cap \dots \cap B_{\lambda^*}^{\mathcal{I}^{n-1}}$  and  $i = \lambda'_S$  and  $j = \lambda_e^*$ . By IH,  $B_{\lambda'} \in S_*(A_{\lambda^1})$ ,  $B_{\lambda''} \in S_*(A_{\lambda^1})$ ,  $\dots$ ,  $B_{\lambda^*} \in S_*(A_{\lambda^1})$ . By NA of CR4,  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
- I2: Then there exists an axiom of the form  $C_{\lambda^3} \sqcap D_{\lambda^5} \sqsubseteq B_{\lambda^4}$  where  $x_\lambda \in C_{\lambda^3}^{\mathcal{I}^{n-1}}$  and  $x_\lambda \in B_{\lambda^5}^{\mathcal{I}^{n-1}}$ . By IH it holds that  $C_{\lambda^3} \in S(A_{\lambda^1})$ . By NA of CR2 it holds that  $B_{\lambda^4} \in S(A_{\lambda^1})$ .  $x_\lambda$  is in  $B_{\lambda^2}^{\mathcal{I}^n}$  by one of three scenarios:
    - i)  $B_{\lambda^4} = B_{\lambda^2}$ : then  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
    - ii)  $\lambda^2 \rightsquigarrow_c \lambda^4$ : then by NA of CR0, it holds that  $B_{\lambda^2} \in S_*(B_{\lambda^4})$ , and by NA of CR0, it holds that  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
    - iii)  $\lambda^2 \rightsquigarrow_c [i, j]$ : where there exists  $B_{\lambda'}, B_{\lambda''}, \dots, B_{\lambda^*}$  and  $\lambda' \rightsquigarrow_t \lambda'' \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^4 \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^*$  and  $x_\lambda \in B_{\lambda'}^{\mathcal{I}^{n-1}} \cap B_{\lambda''}^{\mathcal{I}^{n-1}} \cap \dots \cap B_{\lambda^4}^{\mathcal{I}^n} \cap \dots \cap B_{\lambda^*}^{\mathcal{I}^{n-1}}$  and  $i = \lambda'_S$  and  $j = \lambda_e^*$ . By IH,  $B_{\lambda'} \in S_*(A_{\lambda^1})$ ,  $B_{\lambda''} \in S_*(A_{\lambda^1})$ ,  $\dots$ ,  $B_{\lambda^*} \in S_*(A_{\lambda^1})$ . By NA of CR4,  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
  - I4: Then there exists an axiom of the form  $\exists R_{\lambda^3}. C_{\lambda^5} \sqsubseteq B_{\lambda^4}$  where  $x_\lambda \in (\exists R_{\lambda^2}. C_{\lambda^5} \sqsubseteq B_{\lambda^4})^{\mathcal{I}^{n-1}}$ . Then there exists a  $y$  where  $(x, y) \in R_{\lambda^3}^{\mathcal{I}^{n-1}}$  where  $y \in C_{\lambda^5}^{\mathcal{I}^{n-1}}$ . The only rule that ever adds pairs is I3, therefore there must be an axiom of the form  $E_{\lambda^6} \sqsubseteq \exists R_{\lambda^7}. F_{\lambda^8}$  where  $x_\lambda \in E_{\lambda^6}^{\mathcal{I}^{n-1-i}}$  and  $y$  was born for  $F_{\lambda^8}$  at  $t'$  where  $n - 1 - i \leq t' \leq n - 1$  for some  $i \geq 1$ . By IH it holds that  $E_{\lambda^6} \in S_*(A_{\lambda^1})$  and  $C_{\lambda^5} \in S_*(F_{\lambda^8})$ . If  $R_{\lambda^3} \neq R_{\lambda^7}$  then the only way for the same pair  $(x, y)$  to belong to two separate roles is for  $\lambda^3 \rightsquigarrow_c \lambda^7$ . By NA of CR3,  $B_{\lambda^4} \in S_*(A_{\lambda^1})$ .  $x_\lambda$  is in  $B_{\lambda^2}^{\mathcal{I}^n}$  by one of three scenarios:
    - i)  $B_{\lambda^4} = B_{\lambda^2}$ : then  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
    - ii)  $\lambda^2 \rightsquigarrow_c \lambda^4$ : then by NA of CR0, it holds that  $B_{\lambda^2} \in S_*(B_{\lambda^4})$ , and by NA of CR0, it holds that  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.
    - iii)  $\lambda^2 \rightsquigarrow_c [i, j]$ : where there exists  $B_{\lambda'}, B_{\lambda''}, \dots, B_{\lambda^*}$  and  $\lambda' \rightsquigarrow_t \lambda'' \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^4 \rightsquigarrow_t \dots \rightsquigarrow_t \lambda^*$  and  $x_\lambda \in B_{\lambda'}^{\mathcal{I}^{n-1}} \cap B_{\lambda''}^{\mathcal{I}^{n-1}} \cap \dots \cap B_{\lambda^4}^{\mathcal{I}^n} \cap \dots \cap B_{\lambda^*}^{\mathcal{I}^{n-1}}$  and  $i = \lambda'_S$  and  $j = \lambda_e^*$ . By IH,  $B_{\lambda'} \in S_*(A_{\lambda^1})$ ,  $B_{\lambda''} \in S_*(A_{\lambda^1})$ ,  $\dots$ ,

$B_{\lambda^*} \in S_*(A_{\lambda^1})$ . By NA of CR4,  $B_{\lambda^2} \in S_*(A_{\lambda^1})$ , proving the claim holds.

□

### Lemma 1

Given an  $\mathcal{EL}_{[]}$  TBox  $\mathcal{T}$ , computing  $S_*(A_\lambda)$  for every  $A_\lambda \in \tilde{\mathcal{T}}$  using  $\mathfrak{D}$  terminates.

### Termination

Termination is a consequence of the following observations:

- Monotonic: the rules work in a monotonic way; each rule only adds information to subsumer sets. No information is ever removed.
- Size: the size of each subsumer set is limited by  $|\tilde{\mathcal{T}}|$ .
- Steps: the algorithm stops when no more information can be added, i.e.,  $S_n(A_\lambda) = S_{n-1}(A_\lambda)$  for all  $A_\lambda \in \tilde{\mathcal{T}}$

### Corollary 2

$\mathfrak{D}$  is a decision procedure for classification of  $\mathcal{EL}_{[]}$  TBoxes.

### Time and Space Complexity

Since we have shown the correctness and termination of the subsumer sets, we now show that the classification produced by  $\mathfrak{D}$  can be computed in polynomial time w.r.t the size of  $\mathcal{T}$  and the number of concepts occurring in  $\tilde{\mathcal{T}}$ .

$\mathcal{T}$  can be normalised in polynomial time. It remains to show that sets  $S_*(A)$  can be computed in polynomial time, w.r.t. the size of  $\mathcal{T}$ . We assume a unary encoding of the numbers in the intervals in  $\mathcal{T}$ . Let  $n = |\mathcal{T}|$  and  $m = |\tilde{\mathcal{T}}|$ . The number of sets  $S$  is bounded by  $m$ , and the size of each set is also bounded by  $m$ . Since each set relies only on previous versions, each of the  $\ell$  possible rules can only be applied  $m$  times to each set. The premises of rules INIT0 – CR3 rely only on TBox axioms and the current subsumer sets, both of which are bounded by  $n$  and  $m$  respectively. Rule CR4 however includes a premise that must determine whether a given set of intervals form a *touching sequence* ( $\leadsto_t$ ). Given a certain set of intervals, there are an exponential number of possible sequences in which they may be ordered. Under a naive implementation, searching through all of these sequences to determine whether or not they *touch* in the correct way would

result in an exponential time step. We present an algorithm **GAP-ANALYSIS** (Algorithm 1) that applies **CR4** to  $C_{\lambda'}$  and all  $B_{\lambda^1} \in S_i(C_{\lambda'})$  in polynomial time. The sequencing of intervals begins in **CR4** on line 4.  $X$  represents the set of intervals for which a touching sequence may be formed. We begin by ordering  $X$  in the **ORDER** procedure by their intervals. Each concept is ordered by their intervals start time in ascending order, and for those intervals whose start times are equal, they are then ordered again in ascending order by their end times. Any efficient ordering algorithm can be chosen. The algorithm illustrated uses a well known *bubble sort* technique, known to require only polynomial time. After the sorting is complete, concepts are then removed from  $X$  that will have no effect on the eventual containment of  $C_{\lambda'}$ , which is achieved in the **REMOVAL** procedure. This procedure simply iterates over each (sorted) interval in  $X$  and removes all concepts that end before  $C_{\lambda'}$  starts and start after  $C_{\lambda'}$  ends. This effectively removes all redundant concepts from  $X$  that will not be required in the final premise. This procedure takes only linear time. Finally a gap analysis is performed that simply iterates through  $X$  and checks that there are no gaps in the set by building an array bounded by the minimum and maximum time points in the set and records whether or not all time points in this array have been covered by the intervals. This uses the procedure **NO-GAPS**. Again, this procedure takes only polynomial time. After checking for containment, only then does the rule fire since all premises are now met. It is clear that this rule takes only polynomial time, provided a sorting algorithm of no more than polynomial time is chosen. Since  $m$  is bounded by  $n$ , all sets  $m$  can be computed in polynomial time w.r.t.  $n$ .

### Lemma 2

*Given an  $\mathcal{EL}_{[]}$  TBox  $\mathcal{T}$ , computing  $S_*(A_\lambda)$  using  $\mathfrak{D}$  can be done in polynomial time.*

## 6.3 A Decision Procedure for Computing Subsumption in $\mathcal{EL}_{[]}$

The classification algorithm  $\mathfrak{D}$  presented above is used as a basis for computing subsumption relations between *atomic* concepts (class names, as opposed to

**Algorithm 1** GAP-ANALYSIS for CR4

---

```

1: CR4( $C_{\lambda'}$ )
2: for  $B_{\lambda^1} \in S_i(C_{\lambda'})$  do
3:   for  $A_{\lambda^2}$  do
4:      $X := \{C_{\lambda^i} \mid C_{\lambda^i} \in S_i(A_{\lambda^2})\}$ 
5:      $ORDER(X)$ 
6:      $REMOVAL(X, C_{\lambda'})$ 
7:     if  $NO-GAPS(X)$  then
8:        $n = |X|$ 
9:       if  $start(C_{\lambda'}) \geq start(X[0])$  and  $end(C_{\lambda'}) \leq end(X[n-1])$  then
10:         $S_{i+1}(A_{\lambda^2}) := S_i(A_{\lambda^2}) \cup \{B_{\lambda^1}\}$ 
11:      end if
12:    end if
13:  end for
14: end for

1:  $ORDER(X)$ 
2:  $swapped = \text{true}$ 
3:  $n = |X|$ 
4: while  $swapped$  do
5:    $swapped = \text{false}$ 
6:   for  $\text{int } i = 0$  to  $n - 1$  do
7:     if  $start(X[i+1]) < start(X[i])$  then
8:        $swap(X[i], X[i+1])$ 
9:     else if  $start(X[i+1]) == start(X[i])$  and  $end(X[i+1]) < end(X[i])$  then
10:       $swap(X[i], X[i+1])$ 
11:    end if
12:  end for
13: end while

1:  $REMOVAL(X, C_{\lambda'})$ 
2:  $n = |X|$ 
3: for  $\text{int } i = 0$  to  $n - 1$  do
4:   if  $end(X[i]) < start(C_{\lambda'})$  or  $start(X[i]) > end(C_{\lambda'})$  then
5:      $remove(X[i], X)$ 
6:   end if
7: end for

1:  $NOGAPS(X)$ 
2:  $n = |X|, min = start(X[0]), max = max(X), gaps[] = \text{boolean}$ 
3: for  $\text{int } i = 0$  to  $n - 1$  do
4:   for  $\text{int } j = start(X[i])$  to  $end(X[i])$  do
5:      $gaps[j] = \text{true}$ 
6:   end for
7: end for
8: for  $\text{int } i = min$  to  $max$  do
9:   if  $gaps[i] == \text{false}$  then
10:    return false
11:  end if
12: end for
13: return true

```

---



complex expressions). We can utilise this decision procedure for computing subsumption between complex expressions. Given two possibly complex  $\mathcal{EL}_{[]}$  class expressions  $C$  and  $D$ , and a possibly empty  $\mathcal{EL}_{[]}$  TBox  $\mathcal{T}$ , to decide whether  $\mathcal{T} \models C \sqsubseteq D$ , we can simply introduce two new class names  $A_{[0]}^1$  and  $A_{[0]}^2$  not already occurring in  $\tilde{\mathcal{T}}$ , and use them as definitions for  $C$  and  $D$ :

$$\mathcal{T} := \mathcal{T} \cup \{A_{[0]}^1 \equiv C, A_{[0]}^2 \equiv D\}$$

We could then use  $\mathfrak{D}$  to see whether  $A_{[0]}^2 \in S_*(A_{[0]}^1)$  by checking the resulting classification which would take only linear time using set operations. Adding two classes to  $\mathcal{T} \cup C \cup D$  only increases the size of  $\mathcal{T} \cup C \cup D$  by a constant, and will have no effect on the complexity of reasoning. The size of the intervals are also the smallest they can be (zero intervals as they will be encoded), therefore we achieve the following results:

**Theorem 3**

*$\mathfrak{D}$  can be used as a decision procedure for subsumption in  $\mathcal{EL}_{[]}$ .*

**Theorem 4**

*Subsumption in  $\mathcal{EL}_{[]}$  can be computed in polynomial time.*

## 6.4 A Decision Procedure For Concept Satisfiability in $\mathcal{ALC}_{[]}$

We present a tableau algorithms for testing the satisfiability of an  $\mathcal{ALC}_{[]}$  concept description  $C$ . We assume that  $C$  is in negation normal form (NNF), i.e, negation appears only on atomic formulae. It is well know that any  $C$  can be converted into NNF in linear time.

**Definition 63** (Clash Free ABox)

*An  $\mathcal{ALC}_{[]}$  ABox  $\mathcal{A}$  is said to contain a clash if*

- $\{a : A_{[i]}, a : \neg A_{[i]}\} \subseteq \mathcal{A}$  for any individual  $a$  and any index  $i \in \mathbb{N}$
- $\{a : \perp\} \in \mathcal{A}$  for any individual  $a$ .

*Otherwise,  $\mathcal{A}$  is said to be clash free.*

The general intuition behind the algorithm is to build a set of ABoxes based on a concept description  $C$  using a set of completion rules which terminates when no more rules are applicable. If the set of ABoxes contains an ABox that is *clash free* then  $C$  is satisfiable, otherwise it is unsatisfiable. This is analogous to the standard  $\mathcal{ALC}$  tableau.

We begin with the case where there is no ontology present.

### $\mathfrak{T}_1$ - A Tableau Algorithm for the Satisfiability of $\mathcal{ALC}_{[]}$ Concept Descriptions

The set of ABoxes is built according to the following tableau rules  $\mathfrak{T}_1$ :

#### Definition 64 ( $\mathfrak{T}_1$ )

Let  $\mathfrak{S}$  be a set of  $\mathcal{ALC}_{[]}$  ABoxes, initially empty and  $C$  be an  $\mathcal{ALC}_{[]}$  concept description that will be tested for satisfiability.

*Init* • Set  $\mathfrak{S} := \{\mathcal{A}\}$  where  $\mathcal{A} := \{a : C\}$  and  $a$  is a fresh individual.

$\sqcap$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : C_1 \sqcap C_2 \in \mathcal{A}$  and  $\{a : C_1, a : C_2\} \not\subseteq \mathcal{A}$ , then set  $\mathcal{A} := \mathcal{A} \cup \{a : C_1, a : C_2\}$ .

$\sqcup$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : C_1 \sqcup C_2 \in \mathcal{A}$  and  $\{a : C_1, a : C_2\} \cap \mathcal{A} = \emptyset$ , then create a new ABox  $\mathcal{A}' := \mathcal{A} \cup \{a : C_1\}$ , set  $\mathcal{A} := \mathcal{A} \cup \{a : C_2\}$  and set  $\mathfrak{S} := \mathfrak{S} \cup \mathcal{A}'$ .

$\exists$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : \exists R_{\lambda}. C_1 \in \mathcal{A}$  and there is no  $b$  where  $\{(a, b) : R_{[\ell]}, b : C_1 \mid \lambda_s \leq \ell \leq \lambda_e\} \subseteq \mathcal{A}$ , then select a new individual  $c$  and update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{(a, c) : R_{[\ell]}, c : C_1 \mid \lambda_s \leq \ell \leq \lambda_e\}$ .

$\forall$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $\{a : \forall R_{[i,j]}. C_1, \} \cup \{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\} \subseteq \mathcal{A}$  and  $b : C_1 \notin \mathcal{A}$ , then update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{b : C_1\}$ .

$A_{[]}$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : A_{[i,j]} \in \mathcal{A}$  and  $\{a : A_{[\ell]} \mid i \leq \ell \leq j\} \not\subseteq \mathcal{A}$  then update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{a : A_{[\ell]} \mid i \leq \ell \leq j\}$ .

$\neg A_{[]}$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : \neg A_{[i,j]} \in \mathcal{A}$  and  $\{a : \neg A_{[\ell]} \mid i \leq \ell \leq j\} \cap \mathcal{A} = \emptyset$  then create new ABoxes  $\mathcal{A}^k = \mathcal{A} \cup \{a : \neg A_{[k]}\}$  for  $i \leq k \leq j-1$  and update  $\mathcal{A} := \mathcal{A} \cup \{a : \neg A_{[j]}\}$  and update  $\mathfrak{S} := \mathfrak{S} \cup \mathcal{A}^k$  for  $i \leq k \leq j-1$ .

$R_{[]}$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $(a, b) : R_{[i,j]} \in \mathcal{A}$  and  $\{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\} \not\subseteq \mathcal{A}$  then update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\}$ .

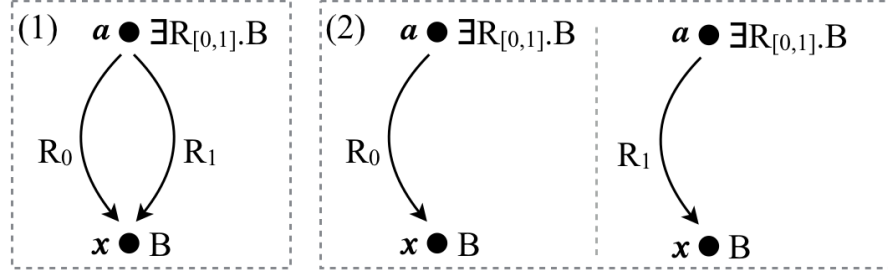


Figure 6.1: An illustration of the tableau procedure for  $\mathcal{ALCC}[\cdot]$  using single ABoxes and multiple sequences of ABoxes to maintain a tree structure.

An ABox  $\mathcal{A}$  is said to be complete if no rules are applicable to  $\mathcal{A}$ . The rules are applied to each ABox  $\mathcal{A}$  in  $\mathfrak{S}$  until  $\mathcal{A}$  is complete or  $\mathcal{A}$  contains a clash.

### Termination of $\mathfrak{T}_1$

#### Lemma 3

Let  $C$  be an  $\mathcal{ALCC}[\cdot]$  concept description.  $\mathfrak{T}_1(C)$  terminates.

As in the usual  $\mathcal{ALC}$  tableaux, we can view each ABox as a tree like structure where individuals are nodes in the tree, and relations between individuals act as edges between the nodes. As an observation of the tableau rules, there is only one root node, which is the individual created upon initialisation, and there are no cycles, conforming to the structure of a tree. However, since we now allow roles to be labelled with intervals, it is possible that multiple edges may exist between two individuals. Suppose we had an assertion  $a : \exists R_{[0,1]}.B_{[0]}$ . An illustration of the tableau rules is shown in Figure 6.1 (1). We can choose to represent this as another dimension, also shown in Figure 6.1 (2), by introducing multiple ABoxes for each index  $i$  in  $\text{MIN}(C) \leq i \leq \text{MAX}(C)$  that contains each single time point assertion of the form  $a : A_{[i]}$  and  $(a, b) : R_{[i]}$ , as well as the usual assertions. This would maintain the tree structure. We can easily transform any ABox into a sequence of these ABoxes.

Let  $m$  be the number of sub-concepts occurring in  $C$  and  $k = \text{MAX}(C) - \text{MIN}(C)$ . Termination of  $\mathfrak{T}_1$  is a consequence of the following observations:

1. The tableaux rules work in a monotonic way. Sub concepts are only ever added to individuals  $a$  in  $\mathcal{A}$  of the form  $a : D$  where  $D$  is a subconcept of  $C$ . No assertions are ever removed. There is also obviously a limit on

the number of sub concepts an individual can be asserted to, bounded by  $m \times k$ .

2. A new individual is added to  $\mathcal{A}$  only when the ' $\exists$ ' rule is applied to an individual  $a$  that has an assertion of the form  $a : \exists R.D$ , and only one individual is created when the rule is applied. Since the number of sub concepts of any individual is bounded by  $m$ , the out degree any node (and thus the amount of any edges to a sequence of ABoxes) is bounded again by  $m \times k$ .
3. Whenever a new individual is added, the number of sub concepts it can have is dependent solely on its parent, specifically they can either come directly from the  $D$  in  $\exists R.D$  from where the individual was created, or from various  $\forall R.D$ s. The only concepts passed down are  $D$ s, which are collectively strictly smaller than the concepts belonging to the parent. This ensures that the depth of each the tree of each ABox does not exceed the size of the length of  $C$  ( $m$ ).
4. A new ABox is introduced for every individual containing a disjunction of the form  $a : C \sqcup D$ , or a negated concept of the form  $a : \neg A_{[i,j]}$ . For each disjunction, only one new ABox is introduced and then the rule is no longer applicable. Since, we know each ABox is of bounded size, then the number of ABoxes is also bounded. For negated concepts, a number of new ABoxes are introduced bounded by the size of the interval on the negated concept. This is also of bounded size.

Together, these properties ensure that there is a bound on the size of each ABox, and the number of ABoxes.  $\square$

**Lemma 4**

*Let  $C$  be an  $\mathcal{ACC}_{[]}$  concept description.  $\mathfrak{T}_1(C)$  produces a complete and clash free ABox iff  $C$  is satisfiable.*

**Proof** We first show the  $\Rightarrow$  direction.

Suppose  $\mathcal{A}$  is a complete and clash free ABox produced by  $\mathfrak{T}_1(C)$ . Our aim is to build an interpretation  $\mathcal{I}$  based on  $\mathcal{A}$  and show that  $\mathcal{I}$  is a witness model of  $C$  by induction on the structure of concepts.

We build the interpretation according to the following rules

**Definition 65** •  $\Delta^{\mathcal{I}_i} := \{x \mid x \text{ is an individual name that occurs in } \mathcal{A}\}$  for each  $i \in \mathbb{Z}$

- $a^{\mathcal{I}} = a$  for each  $a$  occurring in  $\mathcal{A}$
- $A^{\mathcal{I}_i} := \{x \mid x : A_{[i]} \in \mathcal{A}\}$
- $R^{\mathcal{I}_i} := \{(x, y) \mid (x, y) : R_{[i]} \in \mathcal{A}\}$

We now show by induction over the structure of concepts that  $\mathcal{I}$  is a model of  $C$ .

**Claim** (1) For any sub concept  $D$  of  $C$ , and any individual  $x$  that occurs in  $\mathcal{A}$ , if  $x : D \in \mathcal{A} \rightarrow x \in D^{\mathcal{I}}$ .

- Base case:
  - Suppose  $D$  is of the form  $A_{[i,j]}$  where  $i \neq j$ . Then (1) holds by NA of the  $A_{[\ ]}$ -rule and definition of  $\mathcal{I}$ .
  - Suppose  $D$  is of the form  $A_{[i]}$ . Then (1) holds by definition of  $\mathcal{I}$ .
- For any complex concept  $D$ :
  - Suppose  $D$  is of the form  $\neg A_{[i,j]}$ . By NA of the  $\neg A_{[\ ]}$ -rule it holds that  $x : \neg A_{[\ell]}$  for some  $\ell$  where  $i \leq \ell \leq j$ . Since  $\mathcal{A}$  is clash free, then it holds that  $x : A_{[\ell]} \notin \mathcal{A}$ . By definition of  $\mathcal{I}$ ,  $x \notin A^{\mathcal{I}_\ell}$ , therefore  $x \in (\neg A_{[i,j]})^{\mathcal{I}}$ , proving (1) holds.
  - Suppose  $D$  is of the form  $C_1 \sqcap C_2$ . By NA of the  $\sqcap$ -rule, it holds that  $\{x : C_1, x : C_2\} \subseteq \mathcal{A}$ . By IH it holds that  $x \in C_1^{\mathcal{I}}$  and  $x \in C_2^{\mathcal{I}}$ , and therefore  $x \in (C_1 \sqcap C_2)^{\mathcal{I}}$ , proving (1) holds.
  - Suppose  $D$  is of the form  $C_1 \sqcup C_2$ . By NA of the  $\sqcup$ -rule, it holds that  $\{x : C_1\} \in \mathcal{A}$  or  $\{x : C_2\} \in \mathcal{A}$ . By IH it holds that  $x \in C_1^{\mathcal{I}}$  or  $x \in C_2^{\mathcal{I}}$ , and therefore  $x \in (C_1 \sqcup C_2)^{\mathcal{I}}$ , proving (1) holds.
  - Suppose  $D$  is of the form  $\exists R_{[i,j]}.C_1$ . By NA of the  $\exists$ -rule, there is some  $y$  where  $\{(x, y) : R_{[i,j]}, y : C_1\} \subseteq \mathcal{A}$ . By IH it holds that  $y \in C_1^{\mathcal{I}}$ . By NA of the  $R_{[\ ]}$ -rule it holds that  $\{(x, y) : R_{[\ell]} \mid i \leq \ell \leq j\} \subseteq \mathcal{A}$ . By NA of  $I2$  it holds that  $(x, y) \in R_{[i,j]}^{\mathcal{I}}$ . Therefore  $x \in (\exists R_{[i,j]}.C_1)^{\mathcal{I}}$ , proving (1) holds.

- Suppose  $D$  is of the form  $\forall R_{[i,j]}.C_1$ . Let  $y$  be any individual s.t.  $(x, y) \in R_{[i,j]}^{\mathcal{I}}$ . By definition of  $\mathcal{I}$ , it holds that  $\{(x, y) : R_{[\ell]} \mid i \leq \ell \leq j\} \in \mathcal{A}$ . By NA of the  $\forall$ -rule,  $y : C_1 \in \mathcal{A}$ . By IH it holds that  $y \in C_1^{\mathcal{I}}$ . Therefore  $x \in (\forall R_{[i,j]}.C_1)^{\mathcal{I}}$ , proving (1) holds.

□

It suffices to show the  $\Leftarrow$  direction. Note that  $C$  is satisfiable iff the ABox  $\{a : C\}$  is consistent and then  $\{a : C\}$  is obviously clash free. We show that when applying  $\mathfrak{T}_1$  to  $C$ , after each rule application there will be a consistent ABox in  $\mathfrak{S}$ . Since a consistent ABox implies a clash free ABox, together with termination, this proves completeness.

**Claim(2)** There is an ABox  $\mathcal{A}'$  resulting from a possible rule application of  $\mathfrak{T}_1$  to a consistent ABox  $\mathcal{A}$ , that is consistent.

We begin with the ‘*Init*’-rule which produces a single ABox  $\mathcal{A}' := \{a : C\}$  from an empty ABox  $\mathcal{A}$ . Since  $C$  is satisfiable, then the claim holds trivially. If  $\mathcal{A}'$  is not complete, then one of the remaining rules will be fired. Since  $\mathcal{A}'$  is consistent, let  $\mathcal{I}$  be any model of  $\mathcal{A}'$ .

‘ $\sqcap$ ’-rule Let  $a : C_1 \sqcap C_2 \in \mathcal{A}$ . Then  $a^{\mathcal{I}} \in (C_1 \sqcap C_2)^{\mathcal{I}}$  and thus  $a^{\mathcal{I}} \in C_1^{\mathcal{I}}$  and  $a^{\mathcal{I}} \in C_2^{\mathcal{I}}$ .

When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{a : C_1, a : C_2\}$ . It holds that  $\mathcal{I}$  is still a model of  $\mathcal{A}''$ , and thus  $\mathcal{A}''$  is consistent, proving the claim holds.

‘ $\sqcup$ ’-rule Let  $a : C_1 \sqcup C_2 \in \mathcal{A}'$ . Then  $a^{\mathcal{I}} \in (C_1 \sqcup C_2)^{\mathcal{I}}$  and thus either  $a^{\mathcal{I}} \in C_1^{\mathcal{I}}$  or  $a^{\mathcal{I}} \in C_2^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}_1''$  and  $\mathcal{A}_2''$  are produced where  $\mathcal{A}_1'' = \mathcal{A}' \cup \{a : C_1\}$  and  $\mathcal{A}_2'' = \mathcal{A}' \cup \{a : C_2\}$ . It holds that  $\mathcal{I}$  is either a model of  $\mathcal{A}_1''$  or  $\mathcal{A}_2''$ , and thus either  $\mathcal{A}_1''$  is consistent or  $\mathcal{A}_2''$  is consistent, proving the claim holds.

‘ $\exists$ ’-rule Let  $a : \exists R_{[i,j]}.C_1 \in \mathcal{A}'$ . Then  $a^{\mathcal{I}} \in (\exists R_{[i,j]}.C_1)^{\mathcal{I}}$ . By definition of the semantics, there exists an individual  $b \in \Delta^{\mathcal{I}}$  where  $b \in C_1^{\mathcal{I}}$  and  $(a^{\mathcal{I}}, b) \in R_{[i,j]}^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{c : C_1, (a, c) : R_{[\ell]} \mid i \leq \ell \leq j\}$ . Since  $c$  is a new individual then we can set  $b = c^{\mathcal{I}}$  and thus  $\mathcal{I}$  is also a model of  $\mathcal{A}''$ , proving the claim holds.

- ‘ $\forall$ ’-rule Let  $\{a : \forall R_{[i,j]}.C_1\} \cup \{(a, b) : R_{[\ell]} | i \leq \ell \leq j\} \subseteq \mathcal{A}'$ . Then  $a^{\mathcal{I}} \in (\forall R_{[i,j]}.C_1)^{\mathcal{I}}$  and  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_{[i,j]}^{\mathcal{I}}$ . By definition of  $\mathcal{I}$ ,  $b^{\mathcal{I}} \in C_1^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{b : C_1\}$ . Then it holds that  $\mathcal{I}$  is still a model of  $\mathcal{A}''$ , proving the claim holds.
- ‘ $A'_{[\ ]}$ ’-rule Let  $a : A_{[i,j]} \in \mathcal{A}'$ . Then  $a^{\mathcal{I}} \in A_{[i]}^{\mathcal{I}}, \dots, a^{\mathcal{I}} \in A_{[j]}^{\mathcal{I}}$  (since  $A_{[i]}^{\mathcal{I}} = A^{\mathcal{I}_i}$  by definition of  $\mathcal{I}$ ). Since  $\mathcal{A}' = \mathcal{A} \cup \{a : A_{[\ell]} | i \leq \ell \leq j\}$ ,  $\mathcal{I}$  is still a model of  $\mathcal{A}''$  proving the claim holds.
- ‘ $R'_{[\ ]}$ ’-rule Let  $(a, b) : R_{[i,j]} \in \mathcal{A}'$ . Then  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_{[i]}^{\mathcal{I}}, \dots, (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_{[j]}^{\mathcal{I}}$ . Since  $\mathcal{A}'' = \mathcal{A}' \cup \{(a, b) : R_{[\ell]} | i \leq \ell \leq j\}$ , then  $\mathcal{I}$  is still a model of  $\mathcal{A}''$ , proving the claim holds.
- ‘ $\neg A'_{[\ ]}$ ’-rule Let  $a : \neg A_{[i,j]} \in \mathcal{A}'$ . Then  $a^{\mathcal{I}} \in \neg A_{[i]}^{\mathcal{I}}$  or , ..., or  $a^{\mathcal{I}} \in \neg A_{[j]}^{\mathcal{I}}$  (since  $A_{[i]}^{\mathcal{I}} = A^{\mathcal{I}_i}$  by definition of  $\mathcal{I}$ ). When applying rule ‘ $\neg A'_{[\ ]}$ ’ there exists at least one resulting ABox  $\mathcal{A}''$  of the form  $\mathcal{A}'' := \mathcal{A}' \cup \{a : \neg A_{[\ell]}\}$  for some  $i \leq \ell \leq j$ , which  $\mathcal{I}$  will be a model of, and thus  $\mathcal{A}''$  is still consistent, proving the claim holds.

□

**Corollary 3**

$\mathfrak{T}_1$  is a decision procedure for concept satisfiability for  $\mathcal{ALC}_{[\ ]}$ .

**Complexity**

If we again assume a unary encoding on the numbers used in the intervals, then the algorithm requires at most exponential space, w.r.t the size of a concept  $C$ . The algorithm builds a set of ABoxes, each with a tree-like structure. Each tree is bounded in depth by the size of  $C$ , in breadth by an exponential in the size of  $C$ , and the size of the sequence of ABoxes is bounded by exactly  $\text{MIN}(C)$  and  $\text{MAX}(C)$ . For every interval that occurs on a concept name the algorithm can create an ABox for every time point belonging to that interval, depending on the usage of negations, resulting in a exponential blow up.

The algorithm itself is clearly non-optimal however. Consider the algorithm running on the concept  $A_{[0,1]} \sqcap B_{[0,100]}$ . The number of ABoxes the algorithm would create would be 101, (0-100), making an individual  $x$  (by initialisation)

an instance of  $B_{[0]}, B_{[1]}, \dots, B_{[100]}$ . Clearly the only ABoxes of interest would be those at index 0, 1, and 2 since those are the only ones that have interactions with multiple concepts. There is obviously a lot of redundancy by creating instances of concepts that have no interaction with other concepts. Another point worth noting is that if the concept would be instead  $A_{[0,1]} \sqcap \neg B_{0,100}$ , then not only would we again have many redundant declarations, but we would now also introduce a large amount of additional ABoxes (via disjunctions) due to the negated concept.

We leave the task of proving tighter complexity bounds for future work.

## 6.5 A Decision Procedure for $\mathcal{ALC}_{[]}$ Ontology Consistency

We now present a tableau algorithm for deciding  $\mathcal{ALC}_{[]}$  ontology consistency. The intuition behind the algorithm is as before - to build a set of ABoxes using a set of tableau rules and checking for a clash free ABox. However, since we now have a TBox and an ABox, we add more rules to the tableau to handle the additions. As usual, in the presence on a general TBox, to ensure termination, we define the notion of *blocking* w.r.t the new tableau rules (again, analogous to the usual  $\mathcal{ALC}$  tableau algorithm for ontology consistency):

**Definition 66** (Blocking in an ABox)

Given an ABox  $\mathcal{A}$ , and individuals  $a, b \in \mathcal{A}$ ,  $a$  is blocked by  $b$  (or  $b$  blocks  $a$ ) if

- $a$  is younger than  $b$  and
- $\{C \mid a : C \in \mathcal{A}\} \subseteq \{C \mid b : C \in \mathcal{A}\}$

Given an ontology  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ , we assume that every concept occurring in  $\mathcal{O}$  is in NNF and  $\mathcal{A}$  is non empty, and we use  $\dot{\neg}C$  to denote the NNF of  $\neg C$ .

### $\mathfrak{T}_2$ - A Tableau Algorithm for $\mathcal{ALC}_{[]}$ Ontology Consistency

The rules of the tableau  $\mathfrak{T}_2$  are defined as follows:

**Definition 67** ( $\mathfrak{T}_2$ )

Let  $\mathcal{O} = (\mathcal{T}, \mathcal{A}_0)$  be an ontology to be checked for consistency. Let  $\mathfrak{S}$  be a set of ABoxes, initially empty.

*Init* • Set  $\mathfrak{S} := \{\mathcal{A}_0\}$ .



- $\sqcap$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : C_1 \sqcap C_2 \in \mathcal{A}$ ,  $\{a : C_1, a : C_2\} \not\subseteq \mathcal{A}$  and  $a$  is not blocked, then set  $\mathcal{A} := \mathcal{A} \cup \{a : C_1, a : C_2\}$ .
- $\sqcup$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : C_1 \sqcup C_2 \in \mathcal{A}$ ,  $\{a : C_1, a : C_2\} \cap \mathcal{A} = \emptyset$  and  $a$  is not blocked, then create a new ABox  $\mathcal{A}' := \mathcal{A} \cup \{a : C_1\}$ , set  $\mathcal{A} := \mathcal{A} \cup \{a : C_2\}$  and set  $\mathfrak{S} := \mathfrak{S} \cup \mathcal{A}'$ .
- $\exists$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : \exists R_{\lambda_s} C_1 \in \mathcal{A}$  and there is no  $b$  where  $\{(a, b) : R_{[\ell]}, b : C_1 \mid \lambda_s \leq \ell \leq \lambda_e\} \subseteq \mathcal{A}$  and  $a$  is not blocked, then select a new individual  $c$  and update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{(a, c) : R_{[\ell]}, c : C_1 \mid \lambda_s \leq \ell \leq \lambda_e\}$ .
- $\forall$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $\{a : \forall R_{[i,j]} C_1, \} \cup \{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\} \subseteq \mathcal{A}$ ,  $b : C_1 \notin \mathcal{A}$  and  $a$  is not blocked, then update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{b : C_1\}$ .
- $A_{[\ell]}$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : A_{[i,j]} \in \mathcal{A}$  and  $\{a : A_{[\ell]} \mid i \leq \ell \leq j\} \not\subseteq \mathcal{A}$  then update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{a : (\neg)A_{[\ell]} \mid i \leq \ell \leq j\}$ .
- $R_{[\ell]}$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $(a, b) : R_{[i,j]} \in \mathcal{A}$  and  $\{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\} \not\subseteq \mathcal{A}$  then update  $\mathcal{A}$  with  $\mathcal{A} := \mathcal{A} \cup \{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\}$ .
- $\neg A_{[\ell]}$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $a : \neg A_{[i,j]} \in \mathcal{A}$  and  $\{a : \neg A_{[\ell]} \mid i \leq \ell \leq j\} \cap \mathcal{A} = \emptyset$  then create new ABoxes  $\mathcal{A}^k = \mathcal{A} \cup \{a : \neg A_{[k]}\}$  for  $i \leq k \leq j-1$  and update  $\mathcal{A} := \mathcal{A} \cup \{a : \neg A_{[j]}\}$  and update  $\mathfrak{S} := \mathfrak{S} \cup \mathcal{A}^k$  for  $i \leq k \leq j-1$ .
- $\sqsubseteq$  • For any ABox  $\mathcal{A} \in \mathfrak{S}$ , if  $C \sqsubseteq D \in \mathcal{T}$ ,  $b : (\dot{\neg}C \sqcup D) \notin \mathcal{A}$  for any  $b \in \mathcal{A}$  and  $b$  is not blocked, then update  $\mathcal{A}$  with  $\mathcal{A} \cup \{b : (\dot{\neg}C \sqcup D)\}$ .

As before, an ABox  $\mathcal{A}$  is said to be complete if no rules are applicable to  $\mathcal{A}$ . The rules are applied to each ABox  $\mathcal{A}$  in  $\mathfrak{S}$  until  $\mathcal{A}$  is complete or  $\mathcal{A}$  contains a clash.

### Termination of $\mathfrak{T}_2$

#### Lemma 5

Let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[\ell]}$  ontology.  $\mathfrak{T}_2(\mathcal{O})$  terminates.

As before, we can still view any resulting ABox as a sequence of ABoxes each with a tree like structure (See figure 6.1). Proof of termination is very similar to that seen in Section 6.4. Let  $m$  be the number of sub concepts occurring in  $\mathcal{O}$  and  $k = \text{MAX}(\mathcal{O}) - \text{MIN}(\mathcal{O})$ . Termination of  $\mathfrak{T}_2$  is a consequence of the following observations:

1. The rules are again applied in a monotonic way and there can again be at most only  $m \times k$  rule applications for any individual
2. The out degree of any individual is still bounded by  $m \times k$
3. The number of sub concepts for any individual is bounded by  $m \times k$ . In the presence of TBox axioms, due to blocking the number of individuals can not exceed  $2^{m \times k}$ .
4. The number of ABoxes is again bounded by  $m$  and  $k$ .

Together with the arguments shown in Section 6.4, these four properties show termination.  $\square$

### Theorem 5

Let  $\mathcal{O} = (\mathcal{T}, \mathcal{A}_0)$  be a  $\mathcal{ALC}_{[]}$  ontology.  $\mathfrak{T}_2(\mathcal{O})$  produces a complete and clash free ABox iff  $\mathcal{O}$  is consistent.

**Proof** We first show the  $\Rightarrow$  direction.

Suppose  $\mathcal{A}'$  is such a complete and clash free ABox. Our aim is to build an interpretation  $\mathcal{I}$  based on  $\mathcal{A}'$  and show that  $\mathcal{I}$  is a witness model of  $\mathcal{O}$ , by induction on the structure of concepts by showing:

#### Claim(1)

(1.1)  $x : D \in \mathcal{A}' \rightarrow x \in D^{\mathcal{I}}$  where  $x$  is not blocked.

(1.2)  $C \sqsubseteq D \in \mathcal{T} \rightarrow C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$  for any non blocked individuals

We build the interpretation according to the following rules

### Definition 68

Let  $\mathcal{A}$  be a complete and clash free ABox. Fix any mapping  $bl(\cdot)$  that associates with each blocked individual  $y'$ , a unique individual  $bl(y')$  that blocks  $y'$ , and that is not blocked itself.

- $\Delta^{\mathcal{I}_i} := \{x \mid x \text{ occurs in } \mathcal{A} \text{ where } x \text{ is not blocked}\}$  for each  $i \in \mathbb{Z}$
- $a^{\mathcal{I}} = a$  for each  $a$  occurring in  $\mathcal{A}$  where  $a$  is not blocked

- $A^{\mathcal{I}_i} := \{x \mid x : A_{[i]} \in \mathcal{A}\}$  where  $x$  is not blocked
- $R^{\mathcal{I}_i} := \{(x, y) \mid (x, y) : R_{[i]} \in \mathcal{A} \text{ where } x \text{ and } y \text{ are not blocked, or } (x, y') : R_{[i]} \in \mathcal{A} \text{ and } y = bl(y')\}$ .

As usual, we only build the model based on individuals that are not blocked. We show (1.1) by induction on the structure of  $D$ .

Base case:

- Suppose  $D$  is of the form  $A_{[i,j]}$  where  $i \neq j$ . (1.1) holds by NA of the  $A_{[\ ]}$ -rule and definition of  $\mathcal{I}$ .
- Suppose  $D$  is of the form  $A_{[i]}$ . (1.1) holds by definition of  $\mathcal{I}$ .

For any complex concept  $D$ :

- Suppose  $D$  is of the form  $\neg A_{[i,j]}$ . By NA of the  $A_{[\ ]}$ -rule it holds that  $x : \neg A_{[\ell]}$  for some  $\ell$  where  $i \leq \ell \leq j$ . Since  $\mathcal{A}$  is clash free, it holds that  $x : A_{[\ell]} \notin \mathcal{A}$ . By definition of  $\mathcal{I}$ ,  $x \notin A^{\mathcal{I}_i}$ , therefore  $x \in (\neg A_{[i,j]})^{\mathcal{I}}$ , proving (1.1) holds.
- Suppose  $D$  is of the form  $C_1 \sqcap C_2$ . By NA of the  $\sqcap$ -rule, it holds that  $\{x : C_1, x : C_2\} \subseteq \mathcal{A}$ . By induction hypothesis (IH) it holds that  $x \in C_1^{\mathcal{I}}$  and  $x \in C_2^{\mathcal{I}}$ , and therefore  $x \in (C_1 \sqcap C_2)^{\mathcal{I}}$ , proving (1.1) holds.
- Suppose  $D$  is of the form  $C_1 \sqcup C_2$ . By NA of the  $\sqcup$ -rule, it holds that  $\{x : C_1\} \in \mathcal{A}$  or  $\{x : C_2\} \in \mathcal{A}$ . By IH it holds that  $x \in C_1^{\mathcal{I}}$  or  $x \in C_2^{\mathcal{I}}$ , and therefore  $x \in (C_1 \sqcup C_2)^{\mathcal{I}}$ , proving (1.1) holds.
- Suppose  $D$  is of the form  $\exists R_{[i,j]}.C_1$ . By NA of the  $\exists$ -rule, there is some  $y$  where  $\{(x, y) : R_{[i,j]}, y : C_1\} \subseteq \mathcal{A}$ . If  $y$  is not blocked then by IH it holds that  $y \in C_1^{\mathcal{I}}$ . By NA of the  $R_{[\ ]}$ -rule it holds that  $\{(x, y) : R_{[\ell]} \mid i \leq \ell \leq j\} \subseteq \mathcal{A}$ . By definition of  $\mathcal{I}$  it holds that  $(x, y) \in R_{[i,j]}^{\mathcal{I}}$ . Therefore  $x \in (\exists R_{[i,j]}.C_1)^{\mathcal{I}}$ , proving (1.1) holds. If  $y$  is blocked, then we know there is some individual  $y'$  in  $\mathcal{A}$  where  $y' : C_1$  and  $y'$  is not blocked. By NA of the  $R_{[\ ]}$ -rule it holds that  $\{(x, y) : R_{[\ell]} \mid i \leq \ell \leq j\} \subseteq \mathcal{A}$ . By definition of  $\mathcal{I}$  it holds that  $(x, y') \in R_{[i,j]}^{\mathcal{I}}$ . Therefore  $x \in (\exists R_{[i,j]}.C_1)^{\mathcal{I}}$ , proving (1.1) holds.

- Suppose  $D$  is of the form  $\forall R_{[i,j]}.C_1$ . Let  $y$  be any individual s.t  $(x, y) \in R_{[i,j]}^{\mathcal{I}}$ . By definition of  $\mathcal{I}$ , it holds that  $\{(x, y) : R_{[\ell]} \mid i \leq \ell \leq j\} \in \mathcal{A}$ . By NA of the  $\forall$ -rule,  $y : C_1 \in \mathcal{A}$ . By IH it holds that  $y \in C_1^{\mathcal{I}}$ . Therefore  $x \in (\forall R_{[i,j]}.C_1)^{\mathcal{I}}$ , proving (11.1) holds.

$\mathcal{I}$  satisfies all concept and role assertions in  $\mathcal{A}'$ , and thus satisfies all assertions in  $\mathcal{A}$ . Thus  $\mathcal{A}$  has a model and is therefore consistent.

Next we prove (1.2).

Since  $\mathcal{A}'$  is complete, the ' $\sqsubseteq$ ' rule is no longer applicable for any individual in  $\mathcal{A}'$ . Therefore, for any TBox axiom  $C \sqsubseteq D \in \mathcal{T}$ ,  $x : \neg C \sqcup D \in \mathcal{A}'$  for every individual  $x$  that is not blocked. By (1.1) and definition of  $\mathcal{I}$ ,  $x \in (\neg C \sqcup D)^{\mathcal{I}}$ , and thus for every  $x \in C^{\mathcal{I}}$ , it holds that  $x \in D^{\mathcal{I}}$ .

It suffices to show the  $\Leftarrow$  direction. Note that  $\mathcal{O}$  is consistent then by definition  $\mathcal{A}_0$  is clash free. To this end, we show that when applying  $\mathfrak{T}_2$  to  $\mathcal{O}$ , after each rule application there will be a consistent ABox in  $\mathfrak{S}$ . Since a consistent ABox implies a clash free ABox, together with termination, completeness will hold.

**Claim(2)** There is an ABox  $\mathcal{A}'$  resulting from a possible rule application of  $\mathfrak{T}_2$  when applied to an existing ABox  $\mathcal{A}$ , that is consistent.

We begin with the ' $Init$ '-rule which produces a single ABox  $\mathcal{A}' = \mathcal{A}_0$ . Since  $\mathcal{A}_0$  is consistent, then the claim holds trivially. If  $\mathcal{A}'$  is not complete, then one of the remaining rules will be fired. Since  $\mathcal{A}'$  is consistent, let  $\mathcal{I}$  be any model of  $\mathcal{A}'$ .

' $\sqcap$ '-rule Then  $a : C_1 \sqcap C_2 \in \mathcal{A}'$ . Since  $\mathcal{A}'$  is consistent then  $a^{\mathcal{I}} \in (C_1 \sqcap C_2)^{\mathcal{I}}$  and thus  $a^{\mathcal{I}} \in C_1^{\mathcal{I}}$  and  $a^{\mathcal{I}} \in C_2^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{a : C_1, a : C_2\}$ . It holds that  $\mathcal{I}$  is still a model of  $\mathcal{A}''$ , and thus  $\mathcal{A}''$  is consistent, proving the claim holds.

' $\sqcup$ '-rule Then  $a : C_1 \sqcup C_2 \in \mathcal{A}'$ . Since  $\mathcal{A}'$  is consistent then  $a^{\mathcal{I}} \in (C_1 \sqcup C_2)^{\mathcal{I}}$  and thus either  $a^{\mathcal{I}} \in C_1^{\mathcal{I}}$  or  $a^{\mathcal{I}} \in C_2^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}_1''$  and  $\mathcal{A}_2''$  are produced where  $\mathcal{A}_1'' = \mathcal{A}' \cup \{a : C_1\}$  and  $\mathcal{A}_2'' = \mathcal{A}' \cup \{a : C_2\}$ . It holds that  $\mathcal{I}$  is either a model of  $\mathcal{A}_1''$  or  $\mathcal{A}_2''$ , and thus either  $\mathcal{A}_1''$  is consistent or  $\mathcal{A}_2''$  is consistent, proving the claim holds.

' $\exists$ '-rule Since  $\mathcal{A}'$  is consistent then  $a : \exists R_{[i,j]}.C_1 \in \mathcal{A}'$ . Since  $\mathcal{A}'$  is consistent then  $a^{\mathcal{I}} \in (\exists R_{[i,j]}.C_1)^{\mathcal{I}}$ . By definition of  $\mathcal{I}$ , there exists an individual  $b \in \Delta^{\mathcal{I}}$

where  $b \in C_1^{\mathcal{I}}$  and  $(a^{\mathcal{I}}, b) \in R_{[i,j]}^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{c : C_1, (a, c) : R_{[\ell]} \mid i \leq \ell \leq j\}$ . Since  $c$  is a new individual then we can set  $b = c^{\mathcal{I}}$  and thus  $\mathcal{I}$  is also a model of  $\mathcal{A}''$ , which is then consistent, proving the claim holds.

‘ $\forall$ ’-rule Then  $\{a : \forall R_{[i,j]}.C_1\} \cup \{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\} \subseteq \mathcal{A}'$ . Since  $\mathcal{A}'$  is consistent then  $a^{\mathcal{I}} \in (\forall R_{[i,j]}.C_1)^{\mathcal{I}}$  and  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_{[i,j]}^{\mathcal{I}}$ . By definition of  $\mathcal{I}$ ,  $b^{\mathcal{I}} \in C_1^{\mathcal{I}}$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{b : C_1\}$ . Then it holds that  $\mathcal{I}$  is still a model of  $\mathcal{A}''$ , and thus  $\mathcal{A}''$  is consistent, proving the claim holds.

‘ $A'_{[i]}$ ’-rule Then  $a : A_{[i,j]} \in \mathcal{A}'$ . Since  $\mathcal{A}'$  is consistent then  $a^{\mathcal{I}} \in A_{[i]}^{\mathcal{I}}, \dots, a^{\mathcal{I}} \in A_{[j]}^{\mathcal{I}}$  (since  $A_{[i]}^{\mathcal{I}} = A^{\mathcal{I}_i}$  by definition of  $\mathcal{I}$ ). Since  $\mathcal{A}'' = \mathcal{A}' \cup \{a : A_{[\ell]} \mid i \leq \ell \leq j\}$ , then  $\mathcal{I}$  is still a model  $\mathcal{A}''$  and thus  $\mathcal{A}''$  is consistent, proving the claim holds.

‘ $R'_{[i]}$ ’-rule Then  $(a, b) : R_{[i,j]} \in \mathcal{A}$ . Since  $\mathcal{A}'$  is consistent then  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_{[i]}^{\mathcal{I}}, \dots, (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_{[j]}^{\mathcal{I}}$ . Since  $\mathcal{A}'' = \mathcal{A}' \cup \{(a, b) : R_{[\ell]} \mid i \leq \ell \leq j\}$ , then  $\mathcal{I}$  is still a model  $\mathcal{A}''$  and thus  $\mathcal{A}''$  is consistent, proving the claim holds.

‘ $\neg A'_{[i]}$ ’-rule Then  $a : \neg A_{[i,j]} \in \mathcal{A}'$ . Since  $\mathcal{A}'$  is consistent then  $a^{\mathcal{I}} \in \neg A_{[i]}^{\mathcal{I}}$  or  $\dots$ , or  $\neg a^{\mathcal{I}} \in A_{[j]}^{\mathcal{I}}$  (since  $A_{[i]}^{\mathcal{I}} = A^{\mathcal{I}_i}$  by definition of  $\mathcal{I}$ ). When applying rule ‘ $\neg A'_{[i]}$ ’ there exists at least one resulting ABox  $\mathcal{A}''$  of the form  $\mathcal{A}'' := \mathcal{A}' \cup \{a : \neg A_{[\ell]}\}$  for some  $i \leq \ell \leq j$ , for which  $\mathcal{I}$  will be a model of, and thus  $\mathcal{A}''$  is still consistent, proving the claim holds.

‘ $\sqsubseteq'$ ’-rule Then there is an individual  $a \in \mathcal{A}'$  and a TBox axiom  $C \sqsubseteq D$ . By definition of  $\mathcal{I}$ , if  $a \in C^{\mathcal{I}}$  then  $a \in D^{\mathcal{I}}$ . Since  $\mathcal{A}'$  is consistent then  $a : C \in \mathcal{A}' \rightarrow a : D \in \mathcal{A}'$ . When applying the rule to  $\mathcal{A}'$ ,  $\mathcal{A}''$  is produced where  $\mathcal{A}'' = \mathcal{A}' \cup \{a : \neg C \sqcup D\}$ . It holds that  $\mathcal{I}$  is still a model of  $\mathcal{A}''$ , and thus  $\mathcal{A}''$  is consistent, proving the claim holds.

□

### Theorem 6

$\mathfrak{T}_2$  is a decision procedure for  $\mathcal{ALC}_{[]}$  ontology consistency.

### Complexity

Observations regarding the complexity follows a similar pattern to that seen in Section 6.4. Like the naive  $\mathcal{ALC}$  tableau algorithm for ontology consistency, the algorithm requires at least exponential space w.r.t the size of the ontology. In its current form the algorithm can run in double exponential space. Under the naive implementation, the algorithm constructs a set of ABoxes, each of which is of possibly exponential size, i.e., with at most exponentially many individuals, w.r.t  $m$  and  $k$  ( $2^{m \times k}$  where  $m$  is the size of the ontology and  $k$  is the temporal distance of the ontology seen in the termination proof). W.r.t. TBox axioms and the exponential amount of individuals, the algorithm may generate double exponentially many ABoxes, analogous to the standard naive  $\mathcal{ALC}$  tableaux algorithm for ontology consistency. The set of ABoxes can be generated/searched in a non-deterministic fashion, but we leave this for future work.

The algorithm is again non optimal. For every TBox axiom  $A_{\lambda_1} \sqsubseteq B_{\lambda_2}$ , the algorithm introduces a disjunction for every individual in the TBox. This may sometimes be unnecessary. For example, suppose if in an ABox  $a : A_{\lambda_1}$ , adding a disjunction of the form  $a : \neg(A_{\lambda_1} \sqcup B_{\lambda_2})$  would be unnecessary, and a more sensible action would be rather to simply make  $a$  an instance of  $B_{\lambda_2}$ . Again it is clear that several optimisations can be made.

#### 6.5.1 Other reasoning problems in $\mathcal{ALC}_{[]}$

In standard DLs, many reasoning problems are inter-reducible to one another. Consider  $\mathcal{ALC}$ . The reasoning problems of satisfiability and subsumption (and thus classification) w.r.t an ontology  $\mathcal{O}$ , can be reduced to the problem of ontology consistency [BCM<sup>+</sup>03]. Consider the following theorem:

##### Theorem 7

*Let  $\mathcal{O}$  be an  $\mathcal{ALC}$  ontology and  $a$  an individual name not in  $\mathcal{O}$ . Then*

1.  *$C$  is satisfiable w.r.t.  $\mathcal{O}$  iff  $\mathcal{O} \cup \{a : C\}$  is consistent*
2.  *$\mathcal{O} \models A_{\lambda_1} \sqsubseteq B_{\lambda_2}$  iff  $\mathcal{O} \cup \{a : (A_{\lambda_1} \sqcap \neg B_{\lambda_2})\}$  is not consistent*

The same also holds for  $\mathcal{ALC}_{[]}$  as can easily be seen by definition of the semantics in Chapter 5. Therefore, we can use the tableau algorithm  $\mathfrak{T}_2$  to decide both subsumption and a classification in  $\mathcal{ALC}_{[]}$ .

**Theorem 8**

$\mathfrak{T}_2$  can be used as a decision procedure to decide subsumption and classification in  $\mathcal{ALC}_{[]}$ .

## 6.6 A Decision Procedure for Classification in $\mathcal{EL}_{[]}[x]$

We present a decision procedure for a fragment of  $\mathcal{ALC}_{[]}[x]$ ,  $\mathcal{EL}_{[]}[x]$ . Since  $\mathcal{EL}_{[]}[x]$  is closely related to  $\mathcal{EL}_{[]}$ , we utilise the decision procedure for  $\mathcal{EL}_{[]}$ , by providing a transformation from  $\mathcal{EL}_{[]}[x]$  TBoxes into  $\mathcal{EL}_{[]}$  TBoxes, showing that we can use the decision procedure for  $\mathcal{EL}_{[]}$  to classifying  $\mathcal{EL}_{[]}[x]$  TBoxes.

### $\mathcal{EL}_{[]}[x]$ Syntax and Semantics

$\mathcal{EL}_{[]}[x]$  can either be seen as a restriction of  $\mathcal{ALC}_{[]}[x]$ , where we restrict the available DL logical operators to those allowed in  $\mathcal{EL}$ , specifically,  $\top, \sqcap$  and  $\exists$ , or even as an extension to  $\mathcal{EL}_{[]}$ , with the addition of  $[x]$  variables being allowed to be used in axioms.

**Normalisation**

We begin again by transforming our knowledge base into the same normal form seen in Section 6.2. We can use the same procedure as before. The only difference is that we now have to account for the axioms using variables intervals also. Since the normalisation rules have no preference over what kind of intervals are being used in the axiom, the procedure needs only minor changes for the normalisation to work, and is still bounded by polynomial time. We also present a normalisation procedure solely for  $\mathcal{EL}_{[x]}$  in Section 6.9. Both can easily turned into a normalisation procedure for  $\mathcal{EL}_{[]}[x]$ .

**Lemma 6**

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}[x]$ -TBox.  $NF(\mathcal{T})$  can be computed in polynomial time w.r.t the size of  $\mathcal{T}$  and  $NF(\mathcal{T})$  is of polynomial size w.r.t the size of  $\mathcal{T}$ .

All TBoxes mentioned hereinafter in this section are assumed to be in normal form.

### Reduction

We provide a reduction  $\mathfrak{d}$  that takes any  $\mathcal{EL}_{[]}[x]$  TBox  $\mathcal{T}$  and transforms it into an  $\mathcal{EL}_{[]}$  TBox  $\mathcal{T}'$ , such that  $\mathcal{T}$  and  $\mathcal{T}'$  are equivalent, i.e., they have the same models. The method we use is a form of *grounding*.

$\mathfrak{d}$  is defined as follows:

**Definition 69** ( $\mathfrak{d}$  - Reducing  $\mathcal{EL}_{[]}[x]$  TBoxes to  $\mathcal{EL}_{[]}$  TBoxes)

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}[x]$  TBox and Let  $i = \text{MIN}(\mathcal{T})$  and  $j = \text{MAX}(\mathcal{T})$ . For each axiom  $\alpha_x$  in  $\mathcal{T}$  over  $v$ -intervals, replace  $\alpha_x \in \mathcal{T}$  with  $\alpha[x \rightarrow \ell]$  for all  $\ell : i \leq \ell \leq j$ , where no interval index in  $\alpha[x \rightarrow \ell]$  exceeds  $i$  or  $j$  to form  $\mathcal{T}'$ . Then  $\mathfrak{d}(\mathcal{T}) = \mathcal{T}'$ .

The reductions works by unfolding the variable based axioms into their full quantitative versions.

As an example consider the following  $\mathcal{EL}_{[]}[x]$  TBox  $\mathcal{T}$ :

$$\mathcal{T} := \{A_{[0]_x} \sqsubseteq B_{[1]_x} \\ C_{[0]} \sqsubseteq D_{[10]}\}$$

$\text{MIN}(\mathcal{T}) = 0$  and  $\text{MAX}(\mathcal{T}) = 10$  Currently, the atomic entailments are as follows:  
 $\mathcal{T} \models \{C_{[0]} \sqsubseteq D_{[10]}, A_{[0]_x} \sqsubseteq B_{[1]_x}[x \rightarrow k] \mid \forall k : \text{MIN}(\mathcal{T}) \leq k \leq \text{MAX}(\mathcal{T})\}$ .

After performing the reduction  $\mathfrak{d}$  on  $\mathcal{T}$ , we are left with the following  $\mathcal{EL}_{[]}$  TBox  $\mathfrak{d}(\mathcal{T}) := \{C_{[0]} \sqsubseteq D_{[10]}, A_{[0]} \sqsubseteq B_{[1]}, A_{[1]} \sqsubseteq B_{[2]}, A_{[2]} \sqsubseteq B_{[3]}, A_{[3]} \sqsubseteq B_{[4]}, A_{[4]} \sqsubseteq B_{[5]}, A_{[5]} \sqsubseteq B_{[6]}, A_{[6]} \sqsubseteq B_{[7]}, A_{[7]} \sqsubseteq B_{[8]}, A_{[8]} \sqsubseteq B_{[9]}, A_{[9]} \sqsubseteq B_{[10]}\}$ , which clearly has the same interval entailments.

### Lemma 7

$\mathfrak{d}$  is entailment preserving.

We aim to show that  $\mathfrak{d}$  is entailment preserving, i.e.,  $\mathcal{T} \models \alpha$  iff  $\mathfrak{d}(\mathcal{T}) \models \alpha$ .

To this end, we show that for any model  $\mathcal{I}$  of  $\mathcal{T}$ , for all variable axioms  $\alpha_x \in \mathcal{T}$ ,  $\mathcal{I} \models \alpha_x$  iff  $\mathcal{I} \models \mathfrak{d}(\alpha_x)$

We first show the  $\Rightarrow$  direction.

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[]}[x]$  TBox and let  $\alpha_x \in \mathcal{T}$ . Suppose  $\mathcal{I}$  is a model of  $\alpha_x$ . Then by definition of  $\mathcal{I}$ , it holds that  $\mathcal{I} \models \alpha_x[x \rightarrow k]$ , for all  $k$  where  $\text{min}(\mathcal{T}) \leq k \leq \text{max}(\mathcal{T})$ . Since  $\mathfrak{d}$  adds each axiom within this bound, then clearly it holds that  $\mathcal{I}$  is a model of each axiom in  $\alpha_{\mathfrak{f}}$ , proving the claim holds.



For the  $\Leftarrow$  case, suppose  $\mathcal{I}$  is a model  $\mathfrak{d}(\alpha_x)$ . Then  $\mathcal{I} \models \alpha_x[x \rightarrow k]$ , for each  $k$  where  $\min(\mathcal{T}) \leq k \leq \max(\mathcal{T})$ . Then  $\mathcal{I}$  is a model of  $\alpha_x$  by definition of  $\mathcal{I}$ , proving the claim holds.

The reduction does not affect any non variable based interval axiom, thus preserving all standard entailments, hence  $\mathcal{I} \models \alpha$  iff  $\mathcal{I} \models \mathfrak{d}(\alpha)$  holds trivially.  $\square$

### Complexity

Let  $n$  be the size of the time line encoded in a  $\mathcal{EL}_{[ ] [x]}$  TBox  $\mathcal{T}$ , i.e.,  $n = \max(\mathcal{T}) - \min(\mathcal{T})$ . Let  $m$  be the number of axioms occurring in  $\mathcal{T}$ . For each variable axiom  $\alpha_x$  in  $\mathcal{T}$ , we make at most  $n$  copies of  $\alpha_x$  and add them to  $\mathcal{T}$ . The size increase of  $\mathcal{T}$  is bounded by  $n \times m$ , and thus  $\mathfrak{d}(\mathcal{T})$  takes only a polynomial number of steps to compute, and thus can be computed in both polynomial time and space with respect to the size of  $\mathcal{T}$  and the intervals occurring in  $\mathcal{T}$  (assuming a unary encoding of the intervals).

### Theorem 9

$\mathfrak{d}(\mathcal{T})$  can be computed in polynomial time.

### $\mathfrak{D}_2$ A Decision Procedure for Classification

We describe a decision procedure  $\mathfrak{D}_2$  which utilises the decision procedure  $\mathfrak{D}$  from  $\mathcal{EL}_{[ ]}$  as follows. It takes as input a normalised  $\mathcal{EL}_{[ ] [x]}$  TBox and performs the  $\mathfrak{d}$  reduction to produce an equivalent  $\mathcal{EL}_{[ ]}$  TBox  $\mathfrak{d}(\mathcal{T})$ . Since  $\mathfrak{D}$  is a decision procedure for  $\mathcal{EL}_{[ ]}$ , then we simply use  $\mathfrak{D}$  to perform classification on  $\mathfrak{d}(\mathcal{T})$ , using its output as the output for  $\mathfrak{D}_2$ .

### Theorem 10

Classification in  $\mathcal{EL}_{[ ] [x]}$  is decidable.

### Complexity

Since  $\mathfrak{d}(\mathcal{T})$  can be computed in polynomial time and is of polynomial size, and  $\mathfrak{D}$  can be decided in polynomial time, then  $\mathfrak{D}_2$  can also be computed in polynomial time, if we again assume a unary encoding of numbers.

### Theorem 11

Classification in  $\mathcal{EL}_{[ ] [x]}$  can be computed in polynomial time.

As with  $\mathcal{EL}_{[]}$ , subsumption can easily be mapped into to classification (see Section 6.3) in  $\mathcal{EL}_{[][x]}$ .

**Corollary 4**

*Subsumption in  $\mathcal{EL}_{[][x]}$  can be decided in polynomial time.*

## 6.7 A Decision Procedure for $\mathcal{ALC}_{[][x]}$ Ontology Consistency

We present a decision procedure for  $\mathcal{ALC}_{[][x]}$  ontology consistency. As in  $\mathcal{EL}_{[][x]}$ , we again utilise the decision procedures from  $\mathcal{ALC}_{[]}$  by providing a reduction from  $\mathcal{ALC}_{[][x]}$  ontologies into  $\mathcal{ALC}_{[]}$  ontologies.

**Reduction**

We again provide a reduction  $\mathfrak{d}$  that takes any  $\mathcal{ALC}_{[][x]}$  ontology  $\mathcal{O}$  and grounds it into an  $\mathcal{ALC}_{[]}$  ontology  $\mathcal{O}'$ , such that  $\mathcal{O}$  and  $\mathcal{O}'$  are equivalent. In  $\mathcal{ALC}_{[][x]}$  ontologies, variables only occur in TBox axioms, so the reduction remains the same, accounting for the additional ABox:

**Definition 70** ( $\mathfrak{d}$  - Reducing  $\mathcal{ALC}_{[][x]}$  Ontologies to  $\mathcal{ALC}_{[]}$  Ontologies)

*Let  $\mathcal{O} = (\mathcal{T}, \mathcal{A})$  be an  $\mathcal{ALC}_{[][x]}$  ontology and Let  $i = \text{MIN}(\mathcal{O})$  and  $j = \text{MAX}(\mathcal{O})$ . For each axiom  $\alpha_x$  in  $\mathcal{T}$  over  $v$ -intervals, replace  $\alpha_x \in \mathcal{T}$  with  $\alpha[x \rightarrow \ell]$  for all  $\ell : i \leq \ell \leq j$ , where no interval index in  $\alpha[x \rightarrow \ell]$  exceeds  $i$  or  $j$ , to form  $\mathcal{T}'$ . Then  $\mathfrak{d}(\mathcal{O}) = (\mathcal{T}', \mathcal{A})$ .*

**Lemma 8**

*Let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[][x]}$  ontology, and let  $\mathcal{O}' = \mathfrak{d}(\mathcal{O})$  be a  $\mathcal{ALC}_{[]}$  ontology.  $\mathcal{O}$  and  $\mathcal{O}'$  are equivalent.*

We aim to show  $\mathcal{O}$  and  $\mathfrak{d}(\mathcal{O})$  are equivalent by showing that:

1. For any model  $\mathcal{I}$  of  $\mathcal{O}$ ,  $\mathcal{I}$  is also a model of  $\mathcal{O}'$
2. For any model  $\mathcal{I}$  of  $\mathcal{O}'$ ,  $\mathcal{I}$  is also a model of  $\mathcal{O}$

Let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[][x]}$  ontology and  $\mathcal{O}' = \mathfrak{d}(\mathcal{O})$  (1) Let  $\mathcal{I}$  be a model of  $\mathcal{O}$ .  $\mathcal{I}$  satisfies every axiom in both  $\mathcal{T}$  and  $\mathcal{A}$ . For every variable axiom  $\alpha_x \in \mathcal{T}$  of the form  $C_x \sqsubseteq D_x$ , it holds that  $\mathcal{I} \models C_x^{\mathcal{I}^k} \subseteq D_x^{\mathcal{I}^k}$  for all  $k$  where  $\text{MIN}(\mathcal{T}) \leq$

$k \leq \text{MAX}(\mathcal{T})$ , and therefore  $\mathcal{I} \models \alpha_x[x \rightarrow k]$ .  $\mathcal{O}'$  is created by replacing only the variable axioms  $\alpha_x$  in  $\mathcal{T}$  by their grounded versions  $\alpha[x \rightarrow k]$ , thus  $\mathcal{I}$  remains a model of each  $\alpha$ . Since  $\mathcal{I}$  remains a model of every other axiom in  $\mathcal{O}'$ , then  $\mathcal{I}$  is a model of  $\mathcal{O}'$ .

(2) Let  $\mathcal{I}$  be a model of  $\mathcal{O}'$ . Then  $\mathcal{I}$  satisfies every axiom  $\alpha \in \mathcal{T}'$  that was introduced for a variable axiom  $\alpha_x \in \mathcal{T}$ . By definition of  $\mathcal{I}$ ,  $\mathcal{I}$  also satisfies  $\alpha_x$ . Since  $\mathcal{I}$  satisfies every other axiom in  $\mathcal{O}'$  and thus  $\mathcal{O}$ ,  $\mathcal{I}$  is also a model of  $\mathcal{O}$ .  $\square$

### Complexity

#### Theorem 12

Let  $\mathcal{O}$  be an  $\mathcal{ALC}_{[x]}$  ontology.  $\mathfrak{d}(\mathcal{O})$  can be computed in polynomial time.

The complexity of the reduction remains the same as in the  $\mathcal{EL}_{[x]}$  case. Although in addition we have a ABox in this case, our operations only work on the TBox, and we are again bounded by the number of axioms in the TBox, and the time line, and the intervals occurring in the TBox.  $\mathfrak{d}(\mathcal{O})$  can be computed in polynomial time w.r.t the size of  $\mathcal{O}$  and the intervals occurring in  $\mathcal{O}$ , again assuming a unary encoding of the intervals.

#### $\mathfrak{D}_3$ - A Decision Procedure for Ontology Consistency in $\mathcal{ALC}_{[x]}$

Since we have shown that we can reduce any  $\mathcal{ALC}_{[x]}$  ontology into an equivalent  $\mathcal{ALC}_{[]}$  ontology, we can utilize the decision procedure for  $\mathcal{ALC}_{[]}$  as a decision procedure for  $\mathcal{ALC}_{[x]}$ . The following decision procedure,  $\mathfrak{D}_3$ , which decides ontology consistency for any  $\mathcal{ALC}_{[x]}$  ontology: it takes as input a  $\mathcal{ALC}_{[x]}$  ontology  $\mathcal{O}$ , and performs the reduction  $\mathfrak{d}$  to convert it into a  $\mathcal{ALC}_{[]}$  ontology  $\mathcal{O}'$ . It then uses the tableau algorithm  $\mathfrak{T}_2$  to decide ontology consistency of  $\mathcal{O}'$ . Its output is then the output of  $\mathfrak{T}_2(\mathcal{O}')$ .

#### Theorem 13

$\mathfrak{D}_3$  is a decision procedure for ontology consistency in  $\mathcal{ALC}_{[x]}$ .

## 6.8 A Decision Procedure for Concept Satisfiability in $\mathcal{ALC}_{[x]}$

The tableau algorithm  $\mathfrak{T}_1$  for concept satisfiability of  $\mathcal{ALC}_{[]}$  concept descriptions w.r.t an empty ontology can also be applied to testing satisfiability of  $\mathcal{ALC}_{[x]}$

concept descriptions w.r.t an empty ontology. In  $\mathcal{ALC}_{[x]}$ , a concept  $C_x$  is satisfiable if it is satisfiable at time 0, i.e. if there is some model  $\mathcal{I}$  where  $C_x^{\mathcal{I}^0} \neq \emptyset$  which is equivalent to determining whether there is an  $\mathcal{ALC}_{[]}$  model  $\mathcal{I}$  of the  $\mathcal{ALC}_{[]}$  concept  $C_x[x \rightarrow 0]$  where  $C^{\mathcal{I}} \neq \emptyset$ . This *reduction* is possible since there is no TBox to take into account. Therefore, as in the previous cases with  $\mathcal{EL}_{[][x]}$  and  $\mathcal{ALC}_{[][x]}$ , we can reduce the problem of concept satisfiability of  $\mathcal{ALC}_{[x]}$  concept descriptions w.r.t the empty ontology into concept satisfiability of  $\mathcal{ALC}_{[]}$  concept descriptions w.r.t an empty ontology. The reduction is simply to ground each variable occurring in an interval on  $C_x$  with 0, which converts  $C_x$  into the equivalent  $\mathcal{ALC}_{[]}$  concept,  $C_0$ , and run the tableau algorithm  $\mathfrak{T}_1$  on the new concept  $C_0$ . Let this reduction be called  $\mathfrak{d}$ .

The reduction clearly can be done in linear time w.r.t the size of the concept, and regardless on how the concept is encoded, the size of the output concept is bounded by the size of the input concept.

Therefore we achieve the following results:

**Theorem 14**

$\mathfrak{d}(C_x)$  can be computed in linear time w.r.t the size of  $C_x$ .

**Theorem 15**

$\mathfrak{T}_1$  can be used as a decision procedure for satisfiability of  $\mathcal{ALC}_{[x]}$  concept description.

## 6.9 A Decision Procedure for Subsumption in Restricted $\mathcal{EL}_{[x]}$

We present a decision procedure for testing subsumption between two named atomic  $\mathcal{EL}_{[x]}$  concepts w.r.t. restricted version of  $\mathcal{EL}_{[x]}$  TBoxes, called *future-only* TBoxes, which we will explain in more detail later on in this section.

### $\mathcal{EL}_{[x]}$ Syntax & Semantics

The syntax and semantics of  $\mathcal{EL}_{[x]}$  can be seen as a restriction of  $\mathcal{ALC}_{[x]}$ , allowing only for the logical operators  $\top, \sqcap$  and  $\exists$ , and adjusting the semantics to account for this. TBoxes, ABoxes and Ontologies are defines as usual.

Since we are focussed on the v-intervals only, we assume all intervals in this section are v-intervals, so we will often use  $\lambda$  as a representation for v-intervals instead of the usual  $\lambda_x$  (similarly for axioms  $\alpha$ ).

### Some Useful Definitions

Let  $\lambda_1, \lambda_2$  be v-intervals and  $\alpha$  be an  $\mathcal{EL}_{[x]}$  TBox axiom.

- $\_ + \_ : \lambda \times \mathbb{Z} \longrightarrow \lambda'$  where  $[x + i, x + j] + n = [x + i + n, x + j + n]$ .
- $\text{BASE} : \alpha \mapsto \alpha'$ . Let  $\lambda_1$  be in  $\alpha$  s.t there does not exist  $\lambda_2$  in  $\alpha$  where  $\lambda_2^s < \lambda_1^s$ . For all  $\lambda'$  in  $\alpha$ , change  $\lambda'$  to  $\lambda'' = \lambda' + (-\lambda_1^s)$
- $\text{SHIFT}_i : \alpha \mapsto \alpha'$ .  $\text{SHIFT}_i(\alpha) = \alpha'$  is the result of adding  $i \in \mathbb{Z}$  to each  $\lambda$  occurring in  $\alpha$ .

### Future Only TBox

We introduce the notion of a future only TBox, to which we restrict our attention in this section.

**Definition 71** (Future Only  $\mathcal{EL}_{[x]}$  TBox)

A future only TBox axiom  $\alpha(x) = C_x \sqsubseteq D_x$  is an  $\mathcal{EL}_{[x]}$  TBox axiom with the following restrictions:

1. for all concept descriptions  $\exists R_{\lambda^1}.E \in \text{sub}(C_x) \cup \text{sub}(D_x)$ , there is no  $\lambda_2$  in  $E$  where  $\lambda_2^s < \lambda_1^s$
2. for all  $\lambda_1 \in C$ , there is no  $\lambda_2 \in D$  where  $\lambda_2^s < \lambda_1^s$

A future only TBox is a finite set of future only axioms.

The idea behind a future only TBox is that the TBox can only express future information.

Before showing a decision procedure to compute subsumption between concepts in an  $\mathcal{EL}_{[x]}$  future only TBox  $\mathcal{T}$ , we rely on  $\mathcal{T}$  to be in normal form so we first show how this can be computed in polynomial time. We use a similar approach as in Section 6.2 and [Bra04a].

### Normalisation

**Definition 72** (Normal Form of  $\mathcal{EL}_{[x]}$  Future Only TBox)

Let  $\mathcal{T}$  be a future only  $\mathcal{EL}_{[x]}$  TBox.  $\mathcal{T}$  is normalised iff  $\mathcal{T}$  contains only axioms of the form:

$$\begin{aligned} A_{\lambda_1} &\sqsubseteq B_{\lambda_2} \\ A_{\lambda_1} \sqcap B_{\lambda_2} &\sqsubseteq C_{\lambda_3} \\ A_{\lambda_1} &\sqsubseteq \exists R_{\lambda_2}.B_{\lambda_3} \\ \exists R_{\lambda_1}.A_{\lambda_2} &\sqsubseteq B_{\lambda_3} \end{aligned}$$

The following definition shows how any future only  $\mathcal{EL}_{[x]}$  TBox can be transformed into a normalised version using a set of normalisation rules.

**Definition 73** (Normalisation rules of an  $\mathcal{EL}_{[x]}$  future only TBox)

Let  $\mathcal{T}$  be a future only  $\mathcal{EL}_{[x]}$  TBox and  $G$  be an axiom in  $\mathcal{T}$ . For any concept descriptions  $C, D, E$  in  $\tilde{\mathcal{T}}$ , any role  $R_{\lambda_2}$  in  $\tilde{\mathcal{T}}$  and any non-atomic concept descriptions  $\hat{C}$  and  $\hat{D}$  in  $\tilde{\mathcal{T}}$ , the rules are defined as follows:

$R$	$G \longrightarrow G'$
$NFX$	$C \sqsubseteq D \longrightarrow \text{BASE}(C \sqsubseteq D)$
$NF1$	$\hat{C} \sqcap D \sqsubseteq E \longrightarrow \{\hat{C} \sqsubseteq A_{[y]}, A_{[y]} \sqcap D \sqsubseteq E\}$ where $y = \lambda^s : \lambda$ is in $\hat{C}$ and $\nexists \lambda_1$ in $\hat{C}$ s.t. $\lambda_1^s < \lambda^s$
$NF2$	$C \sqcap \hat{D} \sqsubseteq E \longrightarrow \{\hat{D} \sqsubseteq A_{[y]}, A_{[y]} \sqcap C \sqsubseteq E\}$ where $y = \lambda_s : \lambda$ is in $\hat{D}$ and $\nexists \lambda_1$ in $\hat{D}$ s.t. $\lambda_1^s < \lambda^s$
$NF3$	$\exists R_{\lambda_1}.\hat{C} \sqsubseteq D \longrightarrow \{\hat{C} \sqsubseteq A_{[y]}, \exists R_{\lambda_1}.A_{[y]} \sqsubseteq D\}$ where $y = \lambda_s : \lambda$ is in $\hat{C}$ and $\nexists \lambda_2$ in $\hat{C}$ s.t. $\lambda_2^s < \lambda^s$
$NF4$	$C_{\lambda_1} \sqcap D_{\lambda_2} \sqsubseteq E \longrightarrow \{D_{\lambda_2} \sqcap C_{\lambda_1} \sqsubseteq E\}$ where $\lambda_2^s < \lambda_1^s$
$NF5$	$\hat{C} \sqsubseteq \hat{D} \longrightarrow \{\hat{C} \sqsubseteq A_{[y]}, A_{[y]} \sqsubseteq \hat{D}\}$ where $y = \lambda^s : \lambda$ is in $\hat{C}$ and $\nexists \lambda_1$ in $\hat{C}$ s.t. $\lambda_1^s < \lambda^s$
$NF6$	$C \sqsubseteq \exists R_{\lambda_1}.\hat{D} \longrightarrow \{C \sqsubseteq \exists R_{\lambda_1}.A_{[y]}, A_{[y]} \sqsubseteq \hat{D}\}$ where $y = \lambda^s : \lambda$ is in $\hat{D}$ and $\nexists \lambda_1$ in $\hat{D}$ s.t. $\lambda_1^s < \lambda^s$
$NF7$	$C \sqsubseteq D \sqcap E \longrightarrow \{C \sqsubseteq D, C \sqsubseteq E\}$

where  $A$  is a **fresh** concept name not occurring in  $\tilde{\mathcal{T}}$ .

The normalised TBox  $NF(\mathcal{T})$  is created by first applying  $NFX$  to all axioms in  $\mathcal{T}$ , followed by exhaustive application of rules  $NF1$  -  $NF4$  (Step 1) in ascending

order to any axiom conforming to the structure of the rule, then applying rules  $NF5 - NF7$  (Step 2) in a similar manner and finally applying  $NFX$  exhaustively to each axiom. After each application of rules  $NF1 - NF7$ ,  $NFX$  is applied to the resulting axioms to ensure they are base-shifted.

### Proof of complexity

#### Theorem 16

$NF(\mathcal{T})$  can be computed in polynomial time w.r.t the size of  $\mathcal{T}$ .

The proof is similar to the one seen in Section 6.2, so we only sketch the proof here. The number of possible rule applications is limited linearly in the number of sub concepts in  $\mathcal{T}$ . Each of the rules increases the size of  $\mathcal{T}$  only by a constant plus a bound on the original axiom size due to the introduction of a new interval which is no larger than an interval occurring in the current axiom each time, and since each rule can be only applied to a new axiom an amount of times restricted to the size of  $\mathcal{T}$  linearly, it is easy to see that normalisation is bounded polynomially in time to the size of  $\mathcal{T}$ .

#### Lemma 9

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[x]}$ -TBox and let  $NF(\mathcal{T})$  be its normalised version. Then

- (i)  $\widetilde{\mathcal{T}} \subseteq \widetilde{NF(\mathcal{T})}$
- (ii) For any axiom  $G \in NF(\mathcal{T})$ ,  $G$  is a future only axiom
- (iii) For any model  $\mathcal{I}$  of  $NF(\mathcal{T})$ ,  $\mathcal{I}$  is a model of  $\mathcal{T}$
- (iv) For any model  $\mathcal{I}$  of  $\mathcal{T}$ , there is a model  $\mathcal{I}'$  of  $NF(\mathcal{T})$  s.t.  $\mathcal{I} \equiv_{/\mathcal{T}} \mathcal{I}'$

The normalised TBox is entailment preserving. The normalisation procedure works in the same way as in Section 6.2, only with extra massaging for future only axioms and variable intervals. Therefore we show only (ii). Consider every axiom  $G'$  being added to  $NF(\mathcal{T})$ . Every input axiom was already in correct syntactic form, so violation can only occur when introducing new intervals into an axiom due to the nature of the rules. We show an example for  $NF1$ , but the same reasoning holds for every other rule that introduces new intervals. In  $NF1$ , two new axioms are produced for an input axiom  $\hat{C} \sqcap D \sqsubseteq E$ , and a new concept name  $A_{[y]}$  is produced.  $y$  is the smallest start index of an interval occurring in  $\hat{C}$ , so the first

axiom  $\hat{C} \sqsubseteq A_{[y]}$  does not violate either restriction of future only axioms. The second axiom is of the form  $A_{[y]} \sqcap D \sqsubseteq E$ .  $y$  was the smallest start index, so every start index in an interval in  $E$  must be greater than or equal to  $y$ . From the original axiom this was already the case and therefore still holds, proving the claim.

### Corollary 5

$NF(\mathcal{T})$  is entailment preserving.

From this point onwards we assume that all  $\mathcal{EL}_{[x]}$  TBoxes are future only TBoxes and in normal form.

### A Decision Procedure for Atomic Subsumption in $\mathcal{EL}_{[x]}$

We now provide a decision procedure to compute atomic subsumption w.r.t a future only  $\mathcal{EL}_{[x]}$  TBox  $\mathcal{T}$ . We again assume we have a constant domain. Suppose we have two concepts  $F_{\dot{\lambda}_1}$  and  $G_{\dot{\lambda}_2}$ , an  $\mathcal{EL}_{[x]}$  TBox  $\mathcal{T}$ , and we wish to know whether  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+\omega}$  for some  $\omega \in \mathbb{N}$ . We do this by building a directed weighted multi graph  $G = (V, E)$ , where  $V$  is a set of nodes representing base versions of atomic concepts occurring in  $\tilde{\mathcal{T}}$ ,  $E$  is a set of weighted edges over  $V$  and  $\mathbb{N}$  where an edge between two concepts  $F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}$  with weight  $\omega_1$  signifies the subsumption relation  $F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+\omega_1}$ . We build the graph  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  according to the following definition:

#### Definition 74 ( $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$ )

Let  $\mathcal{T}$  be an  $\mathcal{EL}_{[x]}$  TBox,  $F_{\dot{\lambda}_1}$  and  $G_{\dot{\lambda}_2} \in \tilde{\mathcal{T}}$ ,  $\omega \in \mathbb{N}$ ,  $k \in \mathbb{N}$  be the largest  $\lambda^e$  occurring in  $\mathcal{T} \cup \{\dot{\lambda}_1, \dot{\lambda}_2\}$  and let  $k' = k + \omega$ . Initialise  $G$  with

- $V = \{\top_{[0]}\} \cup \{A_{\dot{\lambda}} \mid A_{\lambda} \in \tilde{\mathcal{T}}\}$
- $E = \{(\top_{[0]}, \top_{[0]}, \{0, 1\})\}$ 
  - $\cup \{(A_{\dot{\lambda}}, A_{\dot{\lambda}}, 0), (A_{\dot{\lambda}}, \top_{[0]}, 0)\}$
  - $\cup \{(A_{\dot{\lambda}_1}, A_{\dot{\lambda}_2}, \gamma) \mid \dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_1 \wedge 0 \leq \gamma \leq \dot{\lambda}_1^e - \dot{\lambda}_2^e\}$
  - $\cup \{(A_{\dot{\lambda}_1}, B_{\dot{\lambda}_2}, \gamma) \mid A_{\dot{\lambda}_1} \sqsubseteq B_{\dot{\lambda}_2} \in \mathcal{T} \wedge \gamma = \lambda_s^2 - \lambda_s^1\}$

Next apply the following rules R0 – R3:

$$\begin{aligned} R0: E := E \cup \{ & (A_{\dot{\lambda}_1}, B_{\dot{\lambda}_2}, \gamma) \mid C_{\dot{\lambda}_3} \sqcap D_{\dot{\lambda}_4} \sqsubseteq B_{\dot{\lambda}_2} \in \mathcal{T} \wedge \exists (A_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma_1) \in E \wedge \\ & \exists (A_{\dot{\lambda}_1}, D_{\dot{\lambda}_4}, \gamma_2) \in E \wedge \gamma_2 = \gamma_1 + \lambda_4^s \wedge \gamma = \gamma_1 + \lambda_2^s \wedge \dot{\lambda}_2^e + \gamma \leq k' \} \end{aligned}$$



$$R1: E := E \cup \{(A_{\dot{\lambda}_1}, B_{\dot{\lambda}_2}, \gamma) \mid A_{\dot{\lambda}_1} \sqsubseteq \exists R_{\lambda_3}. C_{\lambda_4} \in \mathcal{T} \wedge \exists R_{\dot{\lambda}_5}. D_{\lambda_6} \sqsubseteq B_{\lambda_2} \in \mathcal{T} \wedge \dot{\lambda}_5 \rightsquigarrow_c \dot{\lambda}_3 \wedge 0 \leq \gamma_1 \leq \dot{\lambda}_3^e - \dot{\lambda}_5^e \wedge (C_{\dot{\lambda}_4}, D_{\dot{\lambda}_6}, \lambda_6^s - (\lambda_4^s - \lambda_3^s) + \gamma_1) \in E \wedge \gamma = \lambda_3^s + \lambda_2^s + \gamma_1 \wedge \dot{\lambda}_2^e + \gamma \leq k'\}$$

$$R2: E := E \cup \{(A_{\dot{\lambda}_1}, B_{\dot{\lambda}_2}, \gamma) \mid \{(A_{\dot{\lambda}_1}, B_{\dot{\lambda}_3}, \gamma_1) \dots (A_{\dot{\lambda}_1}, B_{\dot{\lambda}_n}, \gamma_n)\} \subseteq E \wedge \gamma_1 < \gamma_2 < \dots < \gamma_{n-1} < \gamma_n \wedge \dot{\lambda}_3 + \gamma_i \rightsquigarrow_t \dot{\lambda}_3 + \gamma_j \forall i, j \in \{0, \dots, n\} \text{ where } j = i+1 \wedge \dot{\lambda}_2 \rightsquigarrow_c [0, \gamma_n + \dot{\lambda}_3^e - \gamma_1] \wedge \gamma_1 \leq \gamma \leq \dot{\lambda}_3^e + \gamma_n - \dot{\lambda}_2^e \wedge \dot{\lambda}_2^e + \gamma \leq k'\}$$

$$R3: E := E \cup \{(A_{\dot{\lambda}_1}, B_{\dot{\lambda}_2}, \gamma) \mid \exists (A_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma_1) \in E \wedge \exists (C_{\dot{\lambda}_3}, B_{\dot{\lambda}_2}, \gamma_2) \in E \wedge \gamma = \gamma_1 + \gamma_2 \wedge \gamma + \dot{\lambda}_2^e \leq k'\}$$

**Theorem 17**

Let  $\mathcal{T}$  be a  $\mathcal{EL}_{[x]}$  TBox,  $F_{\dot{\lambda}_1}$  and  $G_{\dot{\lambda}_2} \in \tilde{\mathcal{T}}$ ,  $(V, E) = G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  and let  $\omega \in \mathbb{N}$ . Then  $e = (F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega) \in E$  iff  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2 + \omega}$ .

**Proof** We first show  $\Rightarrow$  by proof of induction on the number of rule applications.

**Claim**  $e = (F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega) \in E \Rightarrow \mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2 + \omega}$

**n=0** Then according to the initialisation of  $G$ , there are 4 possible cases:

- $A = B = \top$ ,  $\omega = \{0, 1\}$  and  $\dot{\lambda}_1 = \dot{\lambda}_2 = [0]$ . If  $\omega = 0$  then it is trivial to see that  $\mathcal{T} \models \top_{[0]} \sqsubseteq \top_{[0]}$ , proving the claim holds. If  $\omega = 1$  then it is also trivial to see that  $\mathcal{T} \models \top_{[0]} \sqsubseteq \top_{[1]}$ , since we assume a constant domain.
- $\omega = 0$  and either (1)  $B = A$  and  $\dot{\lambda}_1 = \lambda_2$  or (2)  $B = \top$  and  $\dot{\lambda}_2 = [0]$ . If (1) then it holds trivially that  $\mathcal{T} \models A_{\dot{\lambda}_1} \sqsubseteq A_{\dot{\lambda}_1 + 0}$ . If (2) then it also holds trivially that  $\mathcal{T} \models A_{\dot{\lambda}_1} \sqsubseteq \top_{\dot{\lambda}_2 + 0}$  where  $\dot{\lambda}_2 = [0]$ .
- $B = A$ ,  $\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_1$  and  $k' \in \{0, 1, \dots, \dot{\lambda}_1^e - \dot{\lambda}_2^e\}$ . For all models  $\mathcal{I}$  of  $\mathcal{T}$ , it holds by definition of  $\mathcal{I}$  that  $A_{\dot{\lambda}_1}^{\mathcal{I}} \subseteq A_{\dot{\lambda}_2 + \omega}^{\mathcal{I}}$  since  $\dot{\lambda}_2 + k' \rightsquigarrow_c \dot{\lambda}_1$ , thus  $\mathcal{T} \models A_{\dot{\lambda}_1} \sqsubseteq A_{\dot{\lambda}_2 + \omega}$ .
- There exists an axiom  $\alpha_x = A_{\dot{\lambda}_1} \sqsubseteq B_{\lambda_2} \in \mathcal{T}$  and  $\omega = \lambda_2^s - \dot{\lambda}_1^s$ . Since  $\dot{\lambda}_1^s = 0$  then  $\lambda_2 = \dot{\lambda}_2 + \omega$ , therefore  $\mathcal{T} \models A_{\dot{\lambda}_1} \sqsubseteq B_{\dot{\lambda}_2 + \omega}$ .

**n > 0**

- Suppose  $R0$  added  $e$  to  $E$ . There exists an axiom  $\alpha_x = C_{\dot{\lambda}_3} \sqcap D_{\lambda_4} \sqsubseteq G_{\lambda_2} \in \mathcal{T}$  and edges  $\{(F_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma_1), (F_{\dot{\lambda}_1}, D_{\dot{\lambda}_4}, \gamma_2)\} \in E$  where  $\gamma_2 = \gamma_1 + \lambda_4^s$  and  $\omega = \gamma_1 + \lambda_2^s$ .

By *IH*, it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq C_{\dot{\lambda}_3+\gamma_1}$  and  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq D_{\dot{\lambda}_4+\gamma_2}$ . Therefore  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq C_{\dot{\lambda}_3+\gamma_1} \sqcap D_{\dot{\lambda}_4+\gamma_2}$ .

Since  $\gamma_2 - \gamma_1 = \lambda_4^s - \dot{\lambda}_3^s$ , it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+\gamma_1}$ . Since  $\omega = \lambda_2^s + \gamma_1$  it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+\omega}$ .

- Suppose *R1* added  $e$  to  $\mathbf{E}$ . There exist axioms  $\alpha_x^1 = F_{\dot{\lambda}_1} \sqsubseteq \exists R_{\dot{\lambda}_3}.C_{\dot{\lambda}_4} \in \mathcal{T}$  and  $\alpha_x^2 = \exists R_{\dot{\lambda}_5}.D_{\dot{\lambda}_6} \sqsubseteq G_{\dot{\lambda}_2} \in \mathcal{T}$  and an edge  $(C_{\dot{\lambda}_4}, D_{\dot{\lambda}_6}, \lambda_6^s - (\lambda_4^s - \lambda_3^s) + \gamma_1) \in \mathbf{E}$  where  $\dot{\lambda}_5 \rightsquigarrow_c \dot{\lambda}_3$  and  $0 \leq \gamma_1 \leq \dot{\lambda}_3^e - \dot{\lambda}_5^e$  and  $\omega = \lambda_3^s + \lambda_2^s + \gamma_1$ .

By *IH* it holds that  $\mathcal{T} \models C_{\dot{\lambda}_4} \sqsubseteq D_{\dot{\lambda}_6+(\lambda_6^s-(\lambda_4^s-\lambda_3^s)+\gamma_1)}$  and therefore  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq \exists R_{\dot{\lambda}_3}.D_{\dot{\lambda}_6+(\lambda_6^s+\lambda_3^s+\gamma_1)}$ .

Since  $\lambda_6^s - \dot{\lambda}_5^s + \gamma_1 = (\lambda_6^s + \lambda_6^s + \lambda_3^s) - \lambda_3^s + \gamma_1$  it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq \exists R_{\dot{\lambda}_5+\lambda_3^s+\gamma_1}.D_{\dot{\lambda}_6+\lambda_3^s+\gamma_1}$ . Finally, it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+(\lambda_2^s+\lambda_3^s+\gamma_1)}$ .

- Suppose *R2* added  $e$  to  $\mathbf{E}$ . There exists edges  $\{(F_{\dot{\lambda}_1}, G_{\dot{\lambda}_3}, \gamma_1) \dots (F_{\dot{\lambda}_1}, G_{\dot{\lambda}_3}, \gamma_n)\} \subseteq \mathbf{E}$  where  $\gamma_1 < \dots < \gamma_n$  and  $\dot{\lambda}_3 + \gamma_i \rightsquigarrow_t \dot{\lambda}_3 + \gamma_j$  where  $i, j \in 0, \dots, n$  and  $j = i + 1$  and  $\dot{\lambda}_2 \rightsquigarrow_c [0, \gamma_n + \dot{\lambda}_3^e - \gamma_1]$  and  $\gamma_1 \leq \omega \leq \dot{\lambda}_3^e + \gamma_n - \dot{\lambda}_2^e$ . By *IH*, it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_3+\gamma_1} \sqcap \dots \sqcap G_{\dot{\lambda}_3+\gamma_n}$ . It holds by definition of the semantics of  $\mathcal{T}$  that  $\mathcal{T} \models G_{\dot{\lambda}_3+\gamma_1} \sqcap \dots \sqcap G_{\dot{\lambda}_3+\gamma_n} \sqsubseteq G_{\dot{\lambda}_2+\gamma'}$  where  $\gamma_1 \leq \gamma' \leq \dot{\lambda}_3^e + \gamma_n - \dot{\lambda}_2^e$ . By transitivity of subsumption, it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+\omega}$ .
- Suppose *R3* added  $e$  to  $\mathbf{E}$ . There exists edges  $\{(F_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma_1), (C_{\dot{\lambda}_3}, G_{\dot{\lambda}_2}, \gamma_2)\} \in \mathbf{E}$  and  $\omega = \gamma_1 + \gamma_2$ . By *IH* it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq C_{\dot{\lambda}_3+\omega_1}$  and  $\mathcal{T} \models C_{\dot{\lambda}_3} \sqsubseteq G_{\dot{\lambda}_2+\gamma_2}$ . Then it holds that  $\mathcal{T} \models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+(\gamma_1+\gamma_2)}$ .

Since we have shown soundness,  $(\Rightarrow)$ , it suffices to completeness,  $(\Leftarrow)$ . We approach this by proving the contraposition: **Claim:**  $e = (F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega) \notin \mathbf{E} \Rightarrow \mathcal{T} \not\models F_{\dot{\lambda}_1} \sqsubseteq G_{\dot{\lambda}_2+\omega}$ . We build a canonical model  $\mathcal{I}$  of  $\mathcal{T}$  with a witness  $y$ , so that there exists a  $j \in \mathbb{Z}$  where  $y \in F_{\dot{\lambda}_1}^{\mathcal{I}}[x \rightarrow j] \setminus G_{\dot{\lambda}_2+\omega}^{\mathcal{I}}[x \rightarrow j]$ . This is similar as in the case of  $\mathcal{EL}_{[]}$  6.2. We construct the canonical model  $\mathcal{I}$  according to the following definition:

**Definition 75** (Building a Model  $\mathcal{I}$  of  $\mathcal{T}$ )

Given an edge  $(A_{\dot{\lambda}_1}, B_{\dot{\lambda}_2}, \omega) \notin \mathbf{G}$  where  $\omega \leq k'$ , let  $\mathcal{I}^0$  be  $\Delta^{\mathcal{I}^0} = \{w\}$  and  $A^{\mathcal{I}^0} = \{w\}$  for  $0 \leq i \leq \dot{\lambda}_e^1$ , and for each  $i \in \mathbb{N}$ , let  $\mathcal{I}^{i+1}$  be the extension of  $\mathcal{I}^i$  obtained by an application of one of the following rules:

- I1:* If  $C_{\dot{\lambda}_3} \sqsubseteq D_{\dot{\lambda}_4} \in \mathcal{T}$  then for all  $j \in \mathbb{N}$ , for every individual  $x \in C_{\dot{\lambda}_3}^{\mathcal{I}^i}[x \rightarrow j]$  where  $x \notin D_{\dot{\lambda}_4}^{\mathcal{I}^i}[x \rightarrow j]$ , add  $x$  to  $D_{\dot{\lambda}_4}^{\mathcal{I}^{i+1}}[x \rightarrow j]$ .

*I2:* If  $C_{\lambda^3} \sqcap D_{\lambda^4} \sqsubseteq E_{\lambda^5} \in \mathcal{T}$  then for all  $j \in \mathbb{N}$ , for every individual  $x \in C_{\lambda^3}^{\mathcal{T}^i}[x \rightarrow j] \cap D_{\lambda^4}^{\mathcal{T}^n}[x \rightarrow j]$  where  $x \notin E_{\lambda^5}^{\mathcal{T}^i}[x \rightarrow j]$ , add  $x$  to  $E_{\lambda^5}^{\mathcal{T}^{i+1}}[x \rightarrow j]$ .

*I3:* If  $C_{\lambda^3} \sqsubseteq \exists R_{\lambda^4}. D_{\lambda^5} \in \mathcal{T}$  then for all  $j \in \mathbb{N}$ , for every individual  $x \in C_{\lambda^3}^{\mathcal{T}^i}[x \rightarrow j]$  where there is no individual  $y \in D_{\lambda^5}^{\mathcal{T}^i}[x \rightarrow j]$  where  $(x, y) \in R_{\lambda^4}^{\mathcal{T}^i}[x \rightarrow j]$ , if  $y \in D_{\lambda^5}^{\mathcal{T}^i}$  then add  $(x, y)$  to  $R_{\lambda^4}^{\mathcal{T}^{i+1}}[x \rightarrow j]$ . If such a  $y$  does not exist, then create  $y$ , add  $y$  to  $D_{\lambda^5}^{\mathcal{T}^{i+1}}[x \rightarrow j]$  and then add  $(x, y)$  to  $R_{\lambda^4}^{\mathcal{T}^{i+1}}[x \rightarrow j]$

*I4:* If  $\exists R_{\lambda^3}. C_{\lambda^4} \sqsubseteq D_{\lambda^5} \in \mathcal{T}$  then for all  $j \in \mathbb{N}$ , for every individual  $x \in R_{\lambda^3}. C_{\lambda^4}^{\mathcal{T}^i}[x \rightarrow j]$  where  $x \notin D_{\lambda^5}^{\mathcal{T}^i}[x \rightarrow j]$ , add  $x$  to  $D_{\lambda^5}^{\mathcal{T}^{i+1}}[x \rightarrow j]$ .

$\mathcal{I}$  is defined as :  $\mathcal{I} := \bigcup_{i=0}^{\infty} \mathcal{I}^i$ .

### Lemma 10

$\mathcal{I}$  is a model of  $\mathcal{T}$

We first show that  $\mathcal{I}$  is in fact a valid model of  $\mathcal{T}$ . Since  $\mathcal{T}$  is normalised, we show that  $\mathcal{I}$  is a model of each of the four possible axioms in  $\mathcal{T}$ .

1. If  $\alpha = A_{\lambda^1} \sqsubseteq B_{\lambda^2} \in \mathcal{T}$ , then for  $\mathcal{I}$  to be a model of  $\alpha$ , it has to be the case that  $\forall j \in \mathbb{Z}, A_{\lambda^1}^{\mathcal{I}}[x \rightarrow j] \subseteq B_{\lambda^2}^{\mathcal{I}}[x \rightarrow j]$ . This holds by definition of *I1*.
2. If  $\alpha = A_{\lambda^1} \sqcap B_{\lambda^2} \sqsubseteq C_{\lambda^3} \in \mathcal{T}$ , then for  $\mathcal{I}$  to be a model of  $\alpha$ , it has to be the case that  $\forall j \in \mathbb{Z}, A_{\lambda^1}^{\mathcal{I}}[x \rightarrow j] \cap B_{\lambda^2}^{\mathcal{I}}[x \rightarrow j] \subseteq C_{\lambda^3}^{\mathcal{I}}[x \rightarrow j]$ . This holds by definition of *I2*.
3. If  $\alpha = A_{\lambda^1} \sqsubseteq \exists R_{\lambda^2}. B_{\lambda^3}$ , then for  $\mathcal{I}$  to be a model of  $\alpha$ , it has to be the case that  $\forall j \in \mathbb{Z}, A_{\lambda^1}^{\mathcal{I}}[x \rightarrow j] \subseteq (\exists R_{\lambda^2}. B_{\lambda^3})^{\mathcal{I}}[x \rightarrow j]$ . This holds by definition of *I3*.
4. If  $\alpha = \exists R_{\lambda^1}. A_{\lambda^2} \sqsubseteq B_{\lambda^3}$ , then for  $\mathcal{I}$  to be a model of  $\alpha$ , it has to be the case that,  $\forall j \in \mathbb{Z}, (\exists R_{\lambda^1}. A_{\lambda^2})^{\mathcal{I}}[x \rightarrow j] \subseteq B_{\lambda^3}^{\mathcal{I}}[x \rightarrow j]$ . This holds by definition of *I4*.

We now show that, for every  $n \in \mathbb{N}$ , for each  $X_{\lambda_1}, Y_{\lambda_2} \in \tilde{\mathcal{T}}$  where  $X_{\lambda_1} \neq Y_{\lambda_2}$ , for all  $y \in X_{\lambda_1}^{\mathcal{T}^n}[x \rightarrow j]$  for all  $j \in \mathbb{N}$ , if  $y$  was born for  $X_{\lambda_1}^{\mathcal{T}^t}[x \rightarrow j]$  and  $y \in Y_{\lambda_2}^{\mathcal{T}^n}[x \rightarrow l]$  where  $t \leq n$ ,  $l \geq j$  and  $l - j \leq k' + \lambda_2^e$ , then  $(X_{\lambda_1}, Y_{\lambda_2}, l - j) \in \mathbf{G}$ . We show this by proof of induction over  $n - t$ .

- $n - t = 0$ : Then  $y$  was born at  $t$  for  $X_{\dot{\lambda}_1}^{\mathcal{I}^t}[x \rightarrow j]$ . If  $y \in Y_{\dot{\lambda}_2}^{\mathcal{I}^t}[x \rightarrow l]$  and  $X_{\dot{\lambda}_1} \neq$  then only two cases can apply:
  - $Y_{\dot{\lambda}_2} = \top_{[0]}$ : By initialisation of  $\mathbf{G}$  and non applicability (NA) of  $R3$ , edges between all concepts and  $\top$  exist with weights from  $0-k'$ , proving the claim holds.
  - $Y = X$ : Then by definition of  $\mathcal{I}$ ,  $\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_1$  and  $l \leq \dot{\lambda}_1^e - \dot{\lambda}_2^e$ . By initialisation of  $\mathbf{G}$ ,  $(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j) \in \mathbf{G}$ , proving the claim holds.
- $n - t > 0$ : Suppose  $y \in Y_{\dot{\lambda}_2}^{\mathcal{I}^n}[x \rightarrow l]$  and  $y \notin Y_{\dot{\lambda}_2}^{\mathcal{I}^{n-1}}[x \rightarrow l]$ . In the definition of  $\mathcal{I}$ , 3 rules could added  $y$  to  $Y_{\dot{\lambda}_2}^{\mathcal{I}^n}[x \rightarrow l]$ :

*I1*: Then there is an axiom of the form  $C_{\dot{\lambda}_3} \sqsubseteq Y_{\dot{\lambda}_4 + \gamma}$  and  $y \in C_{\dot{\lambda}_3}^{\mathcal{I}^{n-1}}[x \rightarrow \gamma']$  where  $j < \gamma'$ .  $y$  is in  $Y_{\dot{\lambda}_2}^{\mathcal{I}^n}[x \rightarrow l]$  under 3 possible scenarios. The first is a **Direct** case where  $y$  is added to the direct extension of  $Y_{\dot{\lambda}_2}$ . The second is a **Contained** case where  $y$  is added to the extension of a  $Y$  that contains  $Y_{\dot{\lambda}_2}$ , and by definition of  $\mathcal{I}$ , ends up in the extension of  $Y_{\dot{\lambda}_2}$ . The third is an **Overlap** case where  $y$  is added to the extension of a  $Y$  that makes up a larger sequence of  $Y$ s, which then contains  $Y_{\dot{\lambda}_2}$ , and by definition of  $\mathcal{I}$ , ends up in ends up in the extension of  $Y_{\dot{\lambda}_2}$ , again similar to the results in Section 6.2.

1. **Direct**:  $Y_{\dot{\lambda}_2} = Y_{\dot{\lambda}_4}$ . By IH there is an edge  $(X_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma' - j) \in \mathbf{E}$  and by initialisation of  $\mathbf{G}$ , there is an edge  $(C_{\dot{\lambda}_3}, Y_{\dot{\lambda}_4}, \gamma) \in \mathbf{E}$ . Then  $l = \gamma + \gamma'$  and by NA of  $R3$ , there is an edge  $(X_{\dot{\lambda}_3}, Y_{\dot{\lambda}_2}, l - j) \in \mathbf{E}$ , proving the claim holds.
2. **Contained**:  $\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_4$  and  $l = \gamma + \gamma' + \gamma''$  for  $0 \leq \gamma'' \leq \dot{\lambda}_4^e - \dot{\lambda}_2^e$ . By IH, there is an edge  $(X_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma' - j) \in \mathbf{E}$  and by initialisation of  $\mathbf{G}$ , there is an edge  $(C_{\dot{\lambda}_3}, Y_{\dot{\lambda}_4}, \gamma) \in \mathbf{E}$ . By initialisation of  $\mathbf{G}$ , there is an edge  $(Y_{\dot{\lambda}_4}, Y_{\dot{\lambda}_2}, \gamma'') \in \mathbf{E}$ . By NA of  $R3$ , there is an edge  $(X_{\dot{\lambda}_3}, Y_{\dot{\lambda}_2}, l - j)$ , proving the claim holds.
3. **Overlap** Let  $\dot{\lambda}_*$  be the smallest interval occurring on a concept  $Y$  occurring in  $\mathcal{T}$ .

Then there is a sequence of  $Y_{\dot{\lambda}_*}$  where

$$y \in Y_{\dot{\lambda}_*}^{\mathcal{I}^{n-1}}[x \rightarrow \gamma_1] \cap Y_{\dot{\lambda}_*}^{\mathcal{I}^{n-1}}[x \rightarrow \gamma_2] \cap \dots \cap Y_{\dot{\lambda}_*}^{\mathcal{I}^{n-1}}[x \rightarrow \gamma_n] \text{ where}$$

$$\gamma_1 \leq \gamma_2 \leq \dots \leq \gamma_n \text{ and}$$

$$\dot{\lambda}_4 + \gamma \rightsquigarrow_t \dot{\lambda}_* + \gamma_i \text{ for some } i \in \{0, \dots, n\} \text{ and}$$

$j \leq \gamma_1$  and

$\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_0$  where  $\dot{\lambda}_0 = [0, \max(\dot{\lambda}_*^e + \gamma_n, \dot{\lambda}_4^e + \gamma)]$ .

Since  $\dot{\lambda}_*$  is minimal, then  $\dot{\lambda}_0$  can be seen as a sequence of consecutive (possibly overlapping)  $\dot{\lambda}_*$  intervals.

Let  $\gamma_{min} = \min(\gamma_1, \gamma)$  and  $\gamma_{max} = \max(\gamma_n + \dot{\lambda}_*^e, \gamma + \dot{\lambda}_4^e)$ .

Then  $\gamma_{min} \leq l \leq \gamma_{max}$ .

Since  $\dot{\lambda}_*$  is minimal, then  $\dot{\lambda}_0$  can be reduced in size to that of  $\dot{\lambda}_2$ , by NA of R2.

By IH there are edges  $\{(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_*}, \gamma' + \gamma), \dots, ((X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_*}, \gamma' + \gamma + \dot{\lambda}_4^e))\} \subseteq E$ .

Then by NA of R2, there is an edge  $(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j)$ .

I2: Then there is an axiom of the form  $C_{\dot{\lambda}_3} \sqcap D_{\dot{\lambda}_4 + \gamma_1} \sqsubseteq Y_{\dot{\lambda}_5 + \gamma}$  and  $y \in C_{\dot{\lambda}_3}^{T^{n-1}}[x \rightarrow \gamma']$  and  $y \in D_{\dot{\lambda}_4}^{T^{n-1}}[x \rightarrow \gamma' + \gamma_1]$  where  $j < \gamma'$ .  $y$  is in  $Y_{\dot{\lambda}_2}^{T^n}[x \rightarrow l]$  under 3 possible scenarios:

1. **Direct:**  $Y_{\dot{\lambda}_2} = Y_{\dot{\lambda}_5}$ . Then  $l = \gamma + \gamma'$ . By IH there are edges  $\{(X_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma' - j), (X_{\dot{\lambda}_1}, D_{\dot{\lambda}_4}, \gamma' + \gamma_1 - j)\} \subseteq E$ . By NA of R0 and R3 there is an edge  $(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j)$ , proving the claim holds.
2. **Contained:**  $\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_5$  and  $l = \gamma + \gamma' + \gamma''$  for  $0 \leq \gamma'' \leq \dot{\lambda}_5^e - \dot{\lambda}_2^e$ . By IH there are edges  $\{(X_{\dot{\lambda}_1}, C_{\dot{\lambda}_3}, \gamma' - j), (X_{\dot{\lambda}_1}, D_{\dot{\lambda}_4}, \gamma' + \gamma_1 - j)\} \subseteq E$ . By initialisation of  $G$ , there is an edge  $(Y_{\dot{\lambda}_5}, Y_{\dot{\lambda}_2}, \gamma'') \in E$ . By NA of R0 there is an edge  $(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j)$ , proving the claim holds.
3. **Overlap** Let  $\dot{\lambda}_*$  be the smallest interval occurring on a concept  $Y$  occurring in  $\mathcal{T}$ .

Then there is a sequence of  $Y_{\dot{\lambda}_*}$  where

$y \in Y_{\dot{\lambda}_*}^{T^{n-1}}[x \rightarrow \gamma_1] \cap Y_{\dot{\lambda}_*}^{T^{n-1}}[x \rightarrow \gamma_2] \cap \dots \cap Y_{\dot{\lambda}_*}^{T^{n-1}}[x \rightarrow \gamma_n]$  where

$\gamma_1 \leq \gamma_2 \leq \dots \leq \gamma_n$  and

$\dot{\lambda}_5 + \gamma \rightsquigarrow_t \dot{\lambda}_* + \gamma_i$  for some  $i \in \{0, \dots, n\}$  and

$j \leq \gamma_1$  and

$\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_0$  where  $\dot{\lambda}_0 = [0, \max(\dot{\lambda}_*^e + \gamma_n, \dot{\lambda}_5^e + \gamma)]$ .

Since  $\dot{\lambda}_*$  is minimal, then  $\dot{\lambda}_0$  can be seen as a sequence of consecutive (possibly overlapping)  $\dot{\lambda}_*$  intervals.

Let  $\gamma_{min} = \min(\gamma_1, \gamma)$  and  $\gamma_{max} = \max(\gamma_n + \dot{\lambda}_*^e, \gamma + \dot{\lambda}_5^e)$ .

Then  $\gamma_{min} \leq l \leq \gamma_{max}$ .

Since  $\dot{\lambda}_*$  is minimal, then  $\dot{\lambda}_0$  can be reduced in size to that of  $\dot{\lambda}_2$ , by NA of R2.

By IH there are edges  $\{(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_*}, \gamma' + \gamma), \dots, ((X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_*}, \gamma' + \gamma + \dot{\lambda}_5^e))\} \subseteq E$ .

Then by NA of R2, there is an edge  $(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j)$ .

I4: There there is an axiom of the form  $\exists R_{\dot{\lambda}_3}. C_{\dot{\lambda}_4 + \gamma_1} \sqsubseteq Y_{\dot{\lambda}_5 + \gamma} \in \mathcal{T}$  where  $y \in (\exists R_{\dot{\lambda}_3}. C_{\dot{\lambda}_4 + \gamma_1}^{T^{n-1}}[x \rightarrow \gamma']$ . There must exist a  $z \in C_{\dot{\lambda}_4 + \gamma_1}^{T^{n-1}}[x \rightarrow \gamma' + \gamma_1]$  where  $(y, z) \in R_{\dot{\lambda}_3}^{T^{n-1}}[x \rightarrow \gamma']$ . By definition of I3, there exists an axiom  $D_{\dot{\lambda}_7} \sqsubseteq \exists R_{\dot{\lambda}_7 + \gamma_2}. E_{\dot{\lambda}_8 + \gamma_3}$  and  $\dot{\lambda}_3 \rightsquigarrow_c \dot{\lambda}_7$ ,  $y \in D_{\dot{\lambda}_6}^{T^{n-1}}[x \rightarrow \gamma'']$ ,  $z \in E_{\dot{\lambda}_8}^{T^{n-1}}[x \rightarrow \gamma'' + \gamma_3]$  and  $(y, z) \in R_{\dot{\lambda}_7}^{T^{n-1}}[x \rightarrow \gamma'' + \gamma_2]$  where  $j \leq \gamma'' \leq \gamma'$ . Since  $\dot{\lambda}_3 \rightsquigarrow_c \dot{\lambda}_7$  then  $\gamma' = \gamma'' + \gamma_2 + \gamma'''$  where  $0 \leq \gamma''' \leq \dot{\lambda}_7^e - \dot{\lambda}_3^e$ .  $y$  is in  $Y_{\dot{\lambda}_2}^{T^n}[x \rightarrow l]$  under 3 possible scenarios:

1. **Direct:**  $Y_{\dot{\lambda}_2} = Y_{\dot{\lambda}_5}$ . Then  $l = \gamma + \gamma'$ . By IH there are edges  $\{(X_{\dot{\lambda}_1}, D_{\dot{\lambda}_6}, \gamma'' - j), (E_{\dot{\lambda}_8}, C_{\dot{\lambda}_4}, \gamma' + \gamma_1 - (\gamma'' + \gamma_3) + \gamma''')\} \subseteq E$ . By NA of R1 there is an edge  $\{(D_{\dot{\lambda}_6}, Y_{\dot{\lambda}_2}, \gamma_2 + \gamma + \gamma''') \in E$ . By NA of R3 there is an edge  $\{(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j) \in E$  since  $l - j = \gamma'' - j + \gamma^2 + \gamma + \gamma'''$ , proving the claim holds.
2. **Contained:**  $\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_5$  and  $l = \gamma + \gamma' + \gamma''''$  for  $0 \leq \gamma'''' \leq \dot{\lambda}_5^e - \dot{\lambda}_2^e$ . By IH there are edges  $\{(X_{\dot{\lambda}_1}, D_{\dot{\lambda}_6}, \gamma'' - j), (E_{\dot{\lambda}_8}, C_{\dot{\lambda}_4}, \gamma' + \gamma_1 - (\gamma'' + \gamma_3) + \gamma''')\} \subseteq E$ . By NA of R1 there is an edge  $\{(D_{\dot{\lambda}_6}, Y_{\dot{\lambda}_5}, \gamma_2 + \gamma + \gamma''') \in E$ . By initialisation of  $G$ , there is an edge  $(Y_{\dot{\lambda}_5}, Y_{\dot{\lambda}_2}, \gamma''') \in E$ . By NA of R3 there is an edge  $\{(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j) \in E$  since  $l - j = \gamma'' - j + \gamma^2 + \gamma + \gamma'''$ , proving the claim holds.
3. **Overlap** Let  $\dot{\lambda}_*$  be the smallest interval occurring on a concept  $Y$  occurring in  $\mathcal{T}$ .

Then there is a sequence of  $Y_{\dot{\lambda}_*}$  where

$y \in Y_{\dot{\lambda}_*}^{T^{n-1}}[x \rightarrow \gamma_1] \cap Y_{\dot{\lambda}_*}^{T^{n-1}}[x \rightarrow \gamma_2] \cap \dots \cap Y_{\dot{\lambda}_*}^{T^{n-1}}[x \rightarrow \gamma_n]$  where

$\gamma_1 \leq \gamma_2 \leq \dots \leq \gamma_n$  and

$\dot{\lambda}_5 + \gamma \rightsquigarrow_t \dot{\lambda}_* + \gamma_i$  for some  $i \in \{0, \dots, n\}$  and

$j \leq \gamma_1$  and

$\dot{\lambda}_2 \rightsquigarrow_c \dot{\lambda}_0$  where  $\dot{\lambda}_0 = [0, \max(\dot{\lambda}_*^e + \gamma_n, \dot{\lambda}_5^e + \gamma)]$ .

Since  $\dot{\lambda}_*$  is minimal, then  $\dot{\lambda}_0$  can be seen as a sequence of consecutive (possibly overlapping)  $\dot{\lambda}_*$  intervals.

Let  $\gamma_{min} = \min(\gamma_1, \gamma)$  and  $\gamma_{max} = \max(\gamma_n + \dot{\lambda}_*^e, \gamma + \dot{\lambda}_5^e)$ .

Then  $\gamma_{min} \leq l \leq \gamma_{max}$ .

Since  $\dot{\lambda}_*$  is minimal, then  $\dot{\lambda}_0$  can be reduced in size to that of  $\dot{\lambda}_2$ , by NA of R2.

By IH there are edges  $\{(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_*}, \gamma' + \gamma), \dots, ((X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_*}, \gamma' + \gamma + \dot{\lambda}_5^e))\} \subseteq E$ .

Then by NA of R2, there is an edge  $(X_{\dot{\lambda}_1}, Y_{\dot{\lambda}_2}, l - j)$ .

□

## Termination

### Lemma 11

Given an  $\mathcal{EL}_{[x]}$  TBox  $\mathcal{T}$ , computing  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  terminates.

To show that  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  terminates, we have to show that (1) initialisation terminates and (2) exhaustive application of rules  $R0 - R3$  terminates. For (1), there are 5 steps where edges and vertices are added to  $G$ .

1. Step 1 involves adding all vertices to the graph. Vertices represent base version of concepts occurring in  $\mathcal{T}$ . Since there are only finitely many, this step terminates after finitely many steps.
2. Step 2 involves adding two single edges between the top concept.
3. Step 3 adds edges a single self edge to every concept and a single edge to every concept and the top concept, after which this step terminates.
4. Step 4 adds edges representing containment edges between concepts with the same name. There are only finitely many containment edges that can be added.
5. Step 5 adds edges based atomic subsumptions present in the TBox. Since the TBox contains only finitely many axioms this is guaranteed to terminate.

For (2), since  $k'$  is the temporal bound for which we compute  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$ , the number of possible edges between 2 nodes in  $G$  is limited by  $k'$ . Next, the algorithm stops when no more rules are applicable, i.e either we have reached the temporal bound for all possible edges or no more information can be inferred. Note that the algorithm works in a monotonic way - information is always only added and never removed. □

### Theorem 18

$G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  is a decision procedure for subsumption in  $\mathcal{EL}_{[x]}$ .

### Complexity

We assume a unary encoding of intervals. Let  $n$  be the number of distinct base concepts names occurring in  $\mathcal{T}$ . The number of nodes in  $G$  is at most  $n + 1$ , and the number of possible edges between any 2 nodes is bounded by  $k'$ . Then initialisation can add  $m$  edges where  $m < k'$ . Each rule application ( $R0 - R3$ ) can then add at most  $l < k' - m$  edges and during each rule application we may have to observe each edge in the current graph. As in the  $\mathcal{EL}_{[]}$  case, rule  $R2$  (CR4 in  $\mathcal{EL}_{[]}$ ) needs to be applied in a *smart* way as to avoid an exponential number of tests for the *touching sequence*. We can reuse Algorithm 1 to achieve an optimised rule application in polynomial time. It is clear that the number of possible steps of creating  $G$  is bounded polynomially by  $k'$  and  $n$ .  $\square$

### Theorem 19

*Subsumption in  $\mathcal{EL}_{[x]}$  can be decided in polynomial time.*

## 6.9.1 A Decision Procedure for Classification in Restricted

### $\mathcal{EL}_{[x]}$

As is usually the case, we can utilise the decision procedure  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  for other reasoning procedures, specifically for a  $k$ -bound classification in future only  $\mathcal{EL}_{[x]}$  TBoxes. If  $\omega$  was our temporal bound ( $k$ ) for which we wished to compute our classification, we could run  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  with the temporal bound  $\omega$  for each pair of base atomic concepts  $F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}$  occurring in a  $\mathcal{EL}_{[x]}$  TBox  $\mathcal{T}$ , and use the result to build a classification. Since the number of times we would need to run  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  is only quadratic in the size of  $\mathcal{T}$ ,  $G(\mathcal{T}, F_{\dot{\lambda}_1}, G_{\dot{\lambda}_2}, \omega)$  can be used as a decision procedure for classification for  $\mathcal{EL}_{[x]}$ .

### Theorem 20

*Classification of future only  $\mathcal{EL}_{[x]}$  TBoxes can be decided in polynomial time.*



# Chapter 7

## TempDL: A Reasoner for $DL_{[]}$

In this chapter we present the design and implementation of two TDL reasoners,  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$ , and an OWL-based syntax for  $[x]$ .

We then go on to evaluate the reasoners with respect to correctness and performance.

### 7.1 Overview

$TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$  are both reasoners for sub languages of  $[x]$  implemented using the OWL API [HB11] in the Java programming language. Both are reasoners that implement the decision procedures introduced in Chapter 6 for the logics  $\mathcal{EL}_{[]}$  and  $\mathcal{ALC}_{[]}$ .  $TEMP_{\mathcal{EL}}$  computes a classification for  $\mathcal{EL}_{[]}$  TBoxes.  $\mathcal{ALC}_{[]}$  also computes a classification as well as satisfiability testing for  $\mathcal{ALC}_{[]}$  concept expressions and  $\mathcal{ALC}_{[]}$  ontology consistency. Both reasoners are meant to act only as a *proof of concept* prototype implementation for each respective  $[x]$  fragment. In the following we describe their implementations, including normalisation techniques and their evaluation.

We first describe our representation of  $[x]$  knowledge bases in OWL using annotation properties for encoding time intervals.

### 7.2 $OWL_{[]}$ - Representing $[x]$ in OWL

$OWL_{[]}$  is a subset of OWL amended with predefined annotation properties acting as a representation of  $[x]$  time intervals (as described in Chapter 5) occurring on each `OWLClass` and `OWLObjectProperty`. Recall that any in sub language

of  $[x]$ , specifically in the  $[]$  fragments,  $N_{con}$  and  $N_{role}$ , are replaced with the sets  $N_{con}^{[]}$  and  $N_{role}^{[]}$  where  $N_{con}^{[]} = N_{con} \times \Lambda$  and  $N_{role}^{[]} = N_{role} \times \Lambda$ , where  $\Lambda$  is the set of all intervals. In order to represent the time intervals, we need three annotation properties: one representing the start point of an interval, another representing the end point of an interval, and for every class or role, one property representing the fully qualified name. The latter is necessary because the same non temporal entity (e.g.  $A$ ) can exist in multiple temporal phases (e.g.  $A_{[0,1]}$  and  $A_{[2,3]}$ ). The annotation properties for representing the start and end points have a range of  $xs:int$ . This ensures that the time points correspond directly to the grammar presented in Chapter 5. Suppose we had the following three annotation properties: `hasStartIndex`, `hasEndIndex` and `hasName`. The classes  $A_{[0,1]}$  and  $A_{[2,3]}$  are represented as follows (using Manchester Syntax):

```

Class: A01
  Annotations:
    hasName "A",
    hasStartIndex 0,
    hasEndIndex 1
Class: A23
  Annotations:
    hasName "A",
    hasStartIndex 2,
    hasEndIndex 3

```

The following definition ensures that in a ‘legal’  $OWL_{[]}$  ontology, each class and each object property value has suitable annotations - in particular that their intervals are correct.

### Definition 76

*Given an OWL 2 DL ontology  $\mathcal{O}$ , an annotation property  $a_s$  with range  $xs:int$  called the *startIndex*, an annotation property  $a_e$  with range  $xs:int$  called the *endIndex*, an annotation property  $a_n$  with range  $rdfs:Literal$  called the *baseName*, an *OWLClass* or *OWLObjectProperty*  $X$  and an annotation assertion  $a(X, v)$  annotating  $X$  with property  $a$  the value  $v$ , we say  $\mathcal{O}$  is  $OWL_{[]}$  legal if the following holds: for every class or object property name  $X \in \tilde{\mathcal{O}}$ ,  $a_s(X, v_1) \in \mathcal{O}$ ,  $a_e(X, v_2) \in \mathcal{O}$  and  $a_n(X, v) \in \mathcal{O}$  and  $v_2 \geq v_1$ .*

We have implemented an  $OWL_{[]}$  validator which only accepts legal  $OWL_{[]}$  ontologies according to Definition 76. The  $TEMP_{\mathcal{EL}}$  validator accepts ontologies that are both in  $\mathcal{EL}$  and are legal  $OWL_{[]}$ .  $TEMP_{\mathcal{ACC}}$  accepts ontologies that are both in  $\mathcal{ACC}$  and are legal  $OWL_{[]}$ .

**Implementation** As mentioned previously, we have implemented an  $OWL_{[]}$  validator used by both  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ACC}}$ , configured to accept either  $\mathcal{EL}_{[]}$  and  $\mathcal{ACC}_{[]}$  ontologies respectively.

The validator has three class methods to check for expressivity levels of given ontologies and to check for  $OWL_{[]}$  membership.

```
public class Validator {
    public static boolean
        isOntologyInALC(OWLOntology ontology)

    public static boolean
        isOntologyInEL(OWLOntology ontology)

    public static boolean
        isOntologyOWLCON(OWLOntology ontology,
            OWLAnnotationProperty startIndex,
            OWLAnnotationProperty endIndex,
            OWLAnnotationProperty basename)
}
```

When checking for DL expressivity levels, each method contains a set of legal axiom types (using the OWL API's `AxiomType`) and legal class expression types (using the OWL API's `ClassExpressionType`), and then parses the ontology's signature, including all logical axioms to ensure that each entity and axiom in the signature conforms to the requirements of each DL. The method returns `true` if the ontology is legal, and `false` otherwise.

When checking for  $OWL_{[]}$  membership, the method is configured with the three annotation properties given by the user. It then proceeds to parse through class and role names in the signature (excluding  $\top$  and  $\perp$ ) and checks each entity has the correct annotation assertion axioms. The method returns `true` if the ontology is valid  $OWL_{[]}$ , and `false` otherwise.

### 7.3 Implementation of $TEMP_{\mathcal{EL}}$ & $TEMP_{\mathcal{ALC}}$

We now go on to provide descriptions of the implementations of both reasoners<sup>1</sup>. Both  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$  are implemented in the Java programming language, and make use of the OWL API [HB11]. Both reasoners partially implement the `OWLReasoner` interface from the OWL API, allowing each of them to be used by any application based on the OWL API. We begin with a system description of  $TEMP_{\mathcal{EL}}$ .

#### 7.3.1 $TEMP_{\mathcal{EL}}$ System Description

$TEMP_{\mathcal{EL}}$  implements the `OWLReasoner` interface.  $TEMP_{\mathcal{EL}}$ 's main function is to compute a classification of a given  $\mathcal{EL}_{[]}$  ontology, and does not implement all the methods given in the interface. We only implement those methods that are directly relevant to the classification and the expressivity limits of  $\mathcal{EL}_{[]}$ . For example, any method relating to data properties or ontology consistency are irrelevant since both are outside the scope of  $\mathcal{EL}_{[]}$ . The functional methods it does implement are as follows:

- `public void precomputeInferences(InferenceType... arg0) :` *Asks the reasoner to precompute certain types of inferences.* This is the main function of the reasoner, which is called to initially compute the classification on a given  $\mathcal{EL}_{[]}$  ontology, stored in an internal data structure.
- `public Set<InferenceType> getPreComputableInferenceTypes() :` *Returns the set of InferenceTypes that are precomputable by reasoner.* Since classification is the only *inference* the reasoner performs, this returns a singleton set, containing the `InferenceType CLASS_HIERARCHY`.

Other non functional methods are implemented, but are not relevant to the reasoning task (such as `getReasonerName()` etc).

Some methods are partially implemented such as

- `public NodeSet<OWLClass> getSubClasses(OWLClassExpression arg0, boolean direct) :` *Gets the set of named classes that are the strict (potentially direct) subclasses of the specified class expression with respect to the reasoner axioms.*

---

<sup>1</sup>both are available to download at <http://www.cs.man.ac.uk/~leoj/thesis/tempdl.jar>

They are *partially* implemented in the sense that they are constrained to specific entities over the signature of the ontology. Specifically, the method works only for `public NodeSet<OWLClass> getSubClasses(OWLClass arg0)` where `arg0` is in the signature of the ontology. For example, calling `getSubClasses` on an `OWLClass` present in the ontology would return the correct subClasses, but calling the method with a complex class expression would return an empty `Set`.

Partially implemented methods include:

```
public NodeSet<OWLClass> getSubClasses
    (OWLClassExpression arg0, boolean direct)

public NodeSet<OWLClass> getSuperClasses
    (OWLClassExpression arg0, boolean direct)

public NodeSet<OWLClass> getEquivalentClasses
    (OWLClassExpression arg0, boolean direct)

public NodeSet<OWLObjectPropertyExpression>
    getSubObjectProperties
        (OWLObjectPropertyExpression arg0,
         boolean direct)

public NodeSet<OWLObjectPropertyExpression>
    getSuperObjectProperties
        (OWLObjectPropertyExpression arg0,
         boolean direct)
```

### Using $TEMP_{\mathcal{EL}}$

**Setup - TempELReasonerFactory** To set up an instance of the  $TEMP_{\mathcal{EL}}$  reasoner, we have implemented the `OWLReasonerFactory` interface available in the OWLAPI. This is the default point of access for creating instances of `OWLReasoner` objects. In the interface are instance methods for returning an instance of a reasoner. The class `TempELReasonerFactory` is described as follows:

```
public class TempELReasonerFactory implements OWLReasonerFactory {
```

```

    public TempELReasonerFactory(OWLAnnotationProperty startIndex,
                                OWLAnnotationProperty endIndex,
                                OWLAnnotationProperty basename,
                                String prefix) {...}

    public OWLReasoner createReasoner(OWLOntology ontology) {...}
}

```

The constructor is passed in the four arguments that define the annotation properties used to encode the temporal information of the intervals in the desired ontology, along with a String to act as a new entity namer when the reasoner may wish to create new non conflicting entities (more on this later). The method `createReasoner` returns an instance of a `TempELReasoner` passing in the three annotation properties and the String prefix which are stored in the instance of the reasoner.

The following is an example of how to return an instance of an `TempELReasoner`.

```

...
TempELReasonerFactory rf =
    new TempELReasonerFactory(start,end,name,prefix);
TempELReasoner r = (TempELReasoner) rf.createReasoner(ont);
...

```

**Initialisation - TempELReasoner** Upon creation, the reasoner stores the three annotation properties, the prefix and the ontology locally in its constructor method. When the user wishes to perform classification a call to the `precomputeInferences` method is considered the first point of access and is performed as follows:

```

...
r.precomputeInferences(InferenceType.CLASS\_HIERARCHY);
...

```

which is where the process of classification begins.

**Validation** The first step of the process is to determine whether or not the ontology is in fact in  $\mathcal{EL}$  and valid  $OWL_{[]}$ . It calls both methods in the `Validator` class to ensure this is the case, throwing a custom exception, `NONELCONOntology`

R	G	$\longrightarrow$	G'
NF1	$C \equiv D$	$\longrightarrow$	$\{C \sqsubseteq D, D \sqsubseteq C\}$
NF2	$\hat{C} \sqcap D \sqsubseteq E$	$\longrightarrow$	$\{ \hat{C} \sqsubseteq A_{\lambda^1}, A_{\lambda^1} \sqcap D \sqsubseteq E \}$
NF3	$C \sqcap \hat{D} \sqsubseteq E$	$\longrightarrow$	$\{ \hat{D} \sqsubseteq A_{\lambda^1}, A_{\lambda^1} \sqcap C \sqsubseteq E \}$
NF4	$\exists R_{\lambda^2}.\hat{C} \sqsubseteq D$	$\longrightarrow$	$\{ \hat{C} \sqsubseteq A_{\lambda^1}, \exists R_{\lambda^2}.A_{\lambda^1} \sqsubseteq D \}$
NF5	$\hat{C} \sqsubseteq \hat{D}$	$\longrightarrow$	$\{ \hat{C} \sqsubseteq A_{\lambda^1}, A_{\lambda^1} \sqsubseteq \hat{D} \}$
NF6	$C \sqsubseteq \exists R_{\lambda^2}.\hat{C}$	$\longrightarrow$	$\{ C \sqsubseteq \exists R_{\lambda^2}.A_{\lambda^1}, A_{\lambda^1} \sqsubseteq \hat{C} \}$
NF7	$C \sqsubseteq D \sqcap E$	$\longrightarrow$	$\{ C \sqsubseteq D, C \sqsubseteq E \}$

Table 7.1:  $\mathcal{EL}_{[]}$  normalisation rules

**Exception**, if the ontology is not valid and terminates. Otherwise, at this point the ontology is known to be in  $\mathcal{EL}$  and correctly temporal (valid  $OWL_{[]}$ ), so it passes through the validation step and proceeds onto the next step of normalisation.

**Normalisation** The intention of the normalisation step is to *convert* the ontology into a normal form, to aid in simplifying the reasoning steps. The normal form is that presented in Chapter 6. Recall, an  $\mathcal{EL}_{[]}$  ontology is said to be in normal form if its TBox contains only axioms of the form:

- $A_{\lambda^1} \sqsubseteq B_{\lambda^2}$
- $A_{\lambda^1} \sqcap B_{\lambda^2} \sqsubseteq C_{\lambda^3}$
- $A_{\lambda^1} \sqsubseteq \exists R_{\lambda^2}.B_{\lambda^3}$
- $\exists R_{\lambda^1}.A_{\lambda^2} \sqsubseteq B_{\lambda^3}$

The normalisation phase takes the input ontology and returns a Set **OWLAxioms** representing the original ontology in its normal form version. We have created a bespoke class **ELCONormalizer**, with a method **getNormalFormELCONTBox** that takes in an  $\mathcal{EL}_{[]}$  ontology and returns a **Set<OWLSubClassOfAxiom>** object, representing the TBox of the ontology in its normal form. The class has several methods which implement the normalizer rules introduced in Chapter 6 and shown again in Table 7.1, with slight variations based on helper methods in the OWL API to optimise certain rules. As an example, consider the rule NF1. The left hand side (LHS) of the rule states that if there is an axiom of the form  $C \equiv D$ , then replace this axiom with two new axioms  $C \sqsubseteq D$  and  $D \sqsubseteq C$ . However,

in the OWL API, equivalent class axioms are not restricted to be binary, and are in fact n-ary. So the implementation of the rule is modified to take this into account:

```
private void NF1(OWLEquivalentClassesAxiom ax) {
    for (OWLClassExpression ce1 : ax.getClassExpressionsAsList()) {
        for (OWLClassExpression ce2 : ax.getClassExpressionsAsList()) {
            if (!ce1.equals(ce2)) {
                OWLSubClassOfAxiom newAxiom1 = getOWLSubClassOfAxiom(ce1, ce2);
                addNewAxiomToOntology(newAxiom1);
            }
        }
    }
}
```

The normalisation procedure may also need to introduce new class names into the signature of the ontology. The *prefix* String passed into the constructor of the reasoner acts as a *new entity namer* system to prevent any internal inconsistencies, for example to prevent reusing a class name already in the ontology. The constraints on the prefix is that it should not already be used as the name of an entity already in the core ontology, hence leaving this as a task for the user to specify. The normaliser keeps an internal *counter* and constructs new names based on this counter and the prefix for each new entity needed. After the normalisation procedure is complete, the reasoner then proceeds on to the classification phase.

**Classification** The classification is based on the subsumer set procedure for classifying  $\mathcal{EL}_{[]} \text{ TBoxes}$  introduced in Chapter 6. The reasoner maintains a `HashMap` called `hierarchy` of the form `HashMap<OWLClass, Set<OWLClass>>` representing the subsumer sets of the classes over the signature of the normalised  $\mathcal{EL}_{[]} \text{ TBox}$ . This acts as a representation the sets  $S_*(A_\lambda)$ . Therefore, the value of each `OWLClass` key is a set containing all subsumers of the `OWLClass`. The reasoner implements methods for each of the rules in the algorithm, *INIT0*, *CR0*, *CR1*, *CR2*, *CR3* and *CR4*, where each rule updates 1 or more values in `hierarchy`, and terminates when `hierarchy` remains unchanged. Upon completion, `hierarchy` represents the completed class hierarchy of the original ontology. To access the information, users can proceed to use the methods described above to extract certain entailments such as

...



```

    r.getSubClasses(class, false)
    ...

```

### 7.3.2 $TEMP_{\mathcal{ALC}}$ System Description

$TEMP_{\mathcal{ALC}}$  is set up in a similar way to  $TEMP_{\mathcal{EL}}$ . It too implements the `OWLReasoner` interface along with many and more of its methods to carry out reasoning tasks. Along with computing classification of  $\mathcal{ALC}_{[]}$  ontologies,  $TEMP_{\mathcal{ALC}}$  also comes with the options to check for ontology consistency, and check for satisfiability of  $\mathcal{ALC}_{[]}$  concept descriptions. Along with all methods that  $TEMP_{\mathcal{EL}}$  (partially) implements,  $TEMP_{\mathcal{ALC}}$  also implements:

- `public boolean isConsistent()` : *Determines if the set of reasoner axioms is consistent.* Returns `true` if the ontology has a model, and `false` otherwise
- `public boolean isSatisfiable(OWLClassExpression classExpression)` : *A convenience method that determines if the specified class expression is satisfiable with respect to the axioms in the ontology.* Returns `true` if `classExpression` is satisfiable w.r.t the ontology, and `false` otherwise.

#### Using $TEMP_{\mathcal{ALC}}$

**Setup - TempALCReasonerFactory** As before we have implemented the `OWLReasonerFactory` interface to create a class `TempALCReasonerFactory` as follows:

```

public class TempALCReasonerFactory implements OWLReasonerFactory {

    public TempALCReasonerFactory(OWLAnnotationProperty startIndex,
        OWLAnnotationProperty endIndex,
        OWLAnnotationProperty basename,
        String prefix) {...}

    public OWLReasoner createReasoner(OWLOntology ontology) {...}
}

```

The reasoner factory acts in the same way as before, taking in the 3 annotation properties representing the interval and base name information, along with a String prefix for new entity names.

Retrieving an instance of an `TempALCReasoner` is done in the same way:

```

...
TempALCReasonerFactory rf =
    new TempALCReasonerFactory(start,end,name,prefix);
TempALCReasoner r = (TempALCReasoner) rf.createReasoner(ont);
...

```

**Initialisation - TempALCReasoner** Upon creation, the reasoner stores the three annotation properties, the new entity prefix and the ontology locally in its constructor method. When the user wishes to perform classification, satisfiability or consistency, a call to the `precomputeInferences(...)` method must first be performed as follows:

```

...
r.precomputeInferences(null);
...

```

**Validation** The first step of the process is to determine whether or not the ontology is in fact in  $\mathcal{ALC}$  and valid  $OWL_{[]}$ . It calls both methods in the `Validator` class to ensure this is the case, throwing a custom exception, `NONALCCONontologyException`, if the ontology is not legal and terminates the process. Otherwise, at this point the ontology is known to be in  $\mathcal{ALC}$  and correctly temporal, so it passes through the validation step and proceeds onto the next step of normalisation.

**Normalisation** The normalisation procedure is simpler than the  $TEMP_{\mathcal{EL}}$  normalisation procedure. The requirement is that the TBox needs only to be a set of subclass axioms and that all concept expression be represented in negation normal form (NNF). This can be done using standard methods available in the OWL API. No new concept or role names are introduced during the normalisation phase. After this phase is complete, the reasoner then begins to check for ontology consistency.

### Ontology Consistency

The ontology consistency phase consists of a building an internal ABox representation of the ontology's assertional axioms, and executing an implementation of the tableau algorithm for  $\mathcal{ALC}_{[]}$  ontology consistency introduced in Chapter 6.

**ALCABox** We have implemented a class **ALCABox**, to act as a representation of an  $\mathcal{ALC}_{[]}$  ABox, compatible with the OWL API. The **ALCABox** contains 5 **Maps**, storing mappings between individuals and five types of assertions:

1. Static class assertions of the form  $A_i(a)$  which reads as *the individual  $a$  is an instance of the class  $A$  in world  $i$*
2. Negative static class assertions of the form  $\neg A_i(a)$  which reads as *the individual  $a$  is not an instance of the class  $A$  in world  $i$*
3. Property assertions of the form  $R_{[i,j]}(a, b)$  which reads as *the individual  $a$  is  $R$  related to the individual  $b$  in worlds  $i$  to  $j$*
4. Static property assertions of the form  $R_i(a, b)$  which reads as *the individual  $a$  is  $R$  related to the individual  $b$  in world  $i$*
5. Class assertions of the form  $C(a)$  which reads as *the individual  $a$  is an instance of the class expression  $C$*

The **ALCABox** has methods to check for clashes, create new individuals and check for blocking. A method `public boolean isABoxClashFree()` has been implemented that checks the **Maps** above for individuals either having  $\perp$  in their negative static class assertions, or the same classes in both their negative static class assertions and static class assertions. A method has also been implemented `public boolean isIndividualBlocked(OWLNamedIndividual)` which determines whether an individual has been blocked according to the definition outlined in the tableau definition. A method also exists to create new individuals when necessary which uses the *prefix* String passed into the reasoner to ensure internal consistencies between previously existing individuals.

When ontology consistency testing begins, the **ALCABox** is initialised with all assertions in the ontology (converting to the internal format as necessary). If no assertions exist in the ontology, then a new individual  $x$  is created, and an assertion is added to the ABox of the form  $\top(x)$ . The tableau algorithm then proceeds to execute on the ABox.

**Tableau Algorithm** The tableau algorithm is an exact implementation of the decision procedure introduced in Chapter 6 for  $\mathcal{ALC}_{[]}$ . There exists implementations of each of the 8 tableau rules in **TempALCReasoner** class, which work on a set of **ALCABoxes**, and expand and possibly introduce new **ALCABoxes** until completeness or a clash is found. At any stage during the procedure, if an ABox is

found to be complete (no more rules are applicable) and it does not contain any clash, then the algorithm terminates, and the ontology is deemed consistent since a model has been found (represented in the clash free and complete `ALCABox`). If such an `ALCABox` does not exist, then a `InconsistentOntologyException` is thrown and the method `precomputeInferences` terminates without successful completion.

If the ontology was consistent, then the method continues to perform the classification.

### Classification

After the ontology has been found consistent, the reasoner then proceeds to perform the classification on the ontology. Since subsumption w.r.t an ontology can be reduced to ontology consistency, it reuses the tableau method used for consistency with a slight tweak on how the initial `ALCABox` was constructed. The algorithm parses through each pair of concept names  $C_1, C_2$  occurring in the ontology and constructs a new `ALCABox` with the assertion  $C_1 \sqcap \neg C_2(x)$ . It then runs the algorithm on this `ALCABox` and checks for consistency. If there is no resulting complete and clash free `ALCABox`, then  $C_1$  is a subclass of  $C_2$ , and is added to an internal representation of the class hierarchy, the same as that seen in `TempELReasoner`. The algorithm terminates once all pairs have been checked. A few optimisations are made, for example not checking entailments that are asserted tautologies such as  $A_{[0,1]} \sqsubseteq A_{[0,0]}$ .

To access the information in the hierarchy, users can proceed to use the methods described above to extract certain entailments such as

```
...
    r.getSubClasses(class, false)
...
```

**Satisfiability** As mentioned above, the reasoner also implements satisfiability checking of class expressions via the method

```
...
    r.isSatisfiable(OWLClassExpression ce)
...
```

This method again reuses the tableau algorithm with another slight modification on how the initial **ALCABox** is set up. The **ALCABox** is initialised with the **TBox** axioms in the ontology along with a new individual being an instance of the class expression given. After running the tableau algorithm on the **ALCABox**, if there is a complete and clash free **ALCABox** present, then the method returns **true**, otherwise it returns **false**.

## 7.4 Evaluation

### 7.4.1 Overview

The goal of the evaluation is to test the correctness of the implementation of the classification algorithm, and, for  $TEMP_{\mathcal{EL}}$ , to provide an evaluation of its performance w.r.t different temporal ontologies and to observe how the overlap of intervals on entities affect the number of resulting inferred subsumptions. We have conducted three experiments to perform these evaluations using temporal versions of selected ontologies from the same snapshot of the OBO foundry used in our survey from Chapter 3.

Both  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$  are proof of concept implementations that aim to demonstrate that temporal reasoning is possible, and show some of the benefits specifically the difference in temporal entailments when using a Temporal Description Logic (TDL) as a representation for bio-health ontologies compared to the standard OWL 2 representation.

### 7.4.2 Experiment Design

The first experiment involves determining the correctness of both  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$ . The second involves evaluating the performance of  $TEMP_{\mathcal{EL}}$  and observing the effects that the temporal aspects of an ontology have on its reasoning time, whilst the third involves observing the additional entailments of temporal ontologies whilst varying temporal aspects.

#### Experiment 1. Correctness

We test the correctness of the reasoners in two different ways. Due to the fact that  $[x]$  is a new TDL, there are no *real* ontologies that are encoded in  $[x]$ , let alone any existing reasoners for  $[x]$  that we can compare either  $TEMP_{\mathcal{EL}}$  or

TEMP<sub>ACC</sub> against. Therefore we are left with two options. The first is to create new ontologies that meet the specification of  $\mathcal{EL}_{[]}$ ,  $\mathcal{ACC}_{[]}$  and  $OWL_{[]}$ , making them suitable for TEMP<sub>EL</sub> and TEMP<sub>ACC</sub>, manually verifying the ontology's entailments and compare them with results of each reasoner. The second is to convert existing ontologies into temporal versions which could then be verified manually or automatically, depending on how they were generated. In either case, the manual verification would be human verification, and would prove to be the biggest problem since this can take a considerable amount of time. In the first case, the ontologies created would have to be considered *complex* enough to test the reasoners' correctness, pushing the limits of the temporal representations, but at the same time be reasonably small in size to make it easy enough for human verification. In the second case, if human verification was needed, the converted ontologies would have to be small enough to reduce time spent manually inspecting each ontology. If we could convert an ontology into a temporal one such that it was entailment preserving, then the size of the ontology would not have an effect on the difficulty of verification, since this would be an automatic process, where we could use the OWL API to verify that they have the same entailments.

For TEMP<sub>EL</sub>, we aim to adopt both approaches to prove correctness. We aim to perform classification on *basic* temporal versions of current  $\mathcal{EL}$  ontologies that would *preserve* the original entailments of the ontologies, and then go on to compare the entailments that TEMP<sub>EL</sub> produces, with the entailments of the original ontology against a current OWL reasoner. For this to be possible, an  $\mathcal{EL}$  ontology would need to be first converted into a temporal version, i.e., legal  $OWL_{[]}$ , whilst at the same time somehow preserving atomic entailments. A straightforward way to achieve this task is to give each `OWLClass` and `OWLObjectProperty` in the signature of an  $\mathcal{EL}$  ontology a *zero-interval* of the form  $[0, 0]$  ( $[0]$ ). As an example, consider the following ontology  $\mathcal{O} := \{A \sqsubseteq B, B \sqsubseteq C\}$ . As well as the entailments already asserted, an inferred entailment holds:  $A \sqsubseteq C$ . The zero-interval version of  $\mathcal{O}$ , written  $\mathcal{O}_{[0]}$ , is of the form  $\mathcal{O}_{[0]} := \{A_{[0]} \sqsubseteq B_{[0]}, B_{[0]} \sqsubseteq C_{[0]}\}$ . Clearly the corresponding inferred entailment holds,  $A_{[0]} \sqsubseteq C_{[0]}$ . As the only interval we use in  $\mathcal{O}_{[0]}$  is  $[0]$ , any model of the temporal version is in fact a model of the standard version since only one possible world is considered. Finally, since we embed the intervals into the standard OWL syntax, by means of annotation properties, determining correctness becomes trivial. We can simply create a zero-interval

ontology from any  $\mathcal{EL}$  ontology, and run both  $\text{TEMP}_{\mathcal{EL}}$  and a standard OWL reasoner, and compare their entailments.  $\text{TEMP}_{\mathcal{EL}}$  will be considered correct if a standard OWL reasoner and  $\text{TEMP}_{\mathcal{EL}}$  produce the same atomic entailments for every class in the signature of an ontology. The OWL reasoner we will use in our evaluation is ELK [KKS12]. ELK is a specialized reasoner for the lightweight ontology language OWL EL, which extends the DL  $\mathcal{EL}$ . ELK is actively maintained and focuses on practical subsets of OWL. ELK was also the winner of the ORE 2015 EL classification challenge [PMG<sup>+</sup>15], so it seemed to be an obvious choice. The set of  $\mathcal{EL}$  ontologies we use in our experiment are those ontologies in the OBO foundry found to be in  $\mathcal{EL}$  (the same snapshot as used in Chapter 3). We reuse our `Validator` class to determine which of those ontologies are in fact valid  $\mathcal{EL}$  ontologies and go on to convert those into valid  $OWL_{[]}$  ontologies with  $[0]$  intervals on each concept and role name. The resulting ontologies, although being valid  $OWL_{[]}$ , and thus members of the  $\mathcal{EL}_{[]}$  ontologies, do not carry any temporal information (hence why they have the same entailments as the non temporal versions). This experiment is only aimed at testing the basic correctness of the reasoner. To extend the tested ontologies to also contain some with temporal information, we also opt to include a bespoke set of small but *difficult*  $\mathcal{EL}_{[]}$  ontologies into our corpus and manually verify the entailments. These ontologies include difficult temporal patterns including many overlapping intervals, many contained intervals and many roles to test the localised rigidity. The ontologies are available to view and download at <http://www.cs.man.ac.uk/~leoj/thesis/>. Human verification will be carried out by three individuals, considered to be experts in the field of DLs. Each individual will be presented with each ontology and asked to compute any and all atomic entailments that they believe to hold. Their results will then be checked against the results of  $\text{TEMP}_{\mathcal{EL}}$  for the same ontologies and the results recorded.

For  $\text{TEMP}_{\mathcal{ALC}}$ , we test correctness only on a set of bespoke  $\mathcal{ALC}_{[]}$  ontologies. Again, we use the same 3 individuals and method to determine the correctness. As well as providing a classification,  $\text{TEMP}_{\mathcal{ALC}}$  also provides satisfiability testing and ontology consistency checking. We aim to incorporate these into our test suite in a similar way. The participants will be presented with an  $\mathcal{ALC}_{[]}$  ontology and asked whether certain classes are satisfiable or whether the ontology is consistent.

## Experiment 2. Performance of $TEMP_{\mathcal{EL}}$

We have found that in preliminary tests,  $TEMP_{\mathcal{EL}}$  performs reasonably well on real ontologies (particularly those zero ontologies created in Experiment 1). We intend to observe the effect that various kinds of temporal aspects of ontologies have on reasoning time.

We take each zero-interval  $\mathcal{EL}_{[]}$  ontology from Experiment 1, and generate a number of different temporal versions. We choose to vary each ontology depending on what types of intervals are used in the ontology on each class and role name, which will also define the time line the ontology has. We will then run  $TEMP_{\mathcal{EL}}$  on each of these new ontologies and see how reasoning time is effected. For example, given an ontology, it would be interesting to see if ontologies containing entities with many overlapping intervals have a direct effect on reasoning time.

To generate varying temporal ontologies, we created several versions of each ontology varying the time line of each with the following ranges:  $\{0-20, 0-30, 0-40, 0-50\}$  and randomly varying the interval sizes for each class and role name appearing in each axiom according to the following set of durations  $\{1, 2, 3, 4, 5, 10, 15, 20\}$ . An interval would be chosen randomly for each concept and role name occurring in each axiom in an ontology w.r.t those bounds. An ontology is generated for each time line size and each interval size pair. For each ontology, a total of 32 different temporal versions of the ontology are generated. These include ontologies with both small and large time lines with many overlapping intervals, both small and large time lines with minimal overlapping intervals and ontologies with intervals spaced out along the time line, amongst others. Reasoning time will be recorded using Java's built in function `System.currentTimeMillis()` which will be called before and after classification. For each ontology we will run the reasoner three times and take an average over the times recorded, unless there is a significant variation between their times, in which case the experiment will be run again.

We intend to analyse the effect that varying temporal patterns have on reasoning time. For example, simple patterns such as whether increasing the time line bounds have an effect on reasoning time, or on a more complex level, whether or not having classes with many intervals that *overlap* will have an effect on reasoning time. An entity overlaps another if it has the same name and their time points in their intervals intersect in some way (this is similar to the notions we defined previously as interval *touching* and interval *containment*). We therefore



introduce the notion of overlap *density*, which will provide an average measure of how much of an ontology's concepts and roles have overlapping intervals.

Two entities with the same name overlap if their intervals intersect. The amount they overlap is the size of the intersection of the two intervals. For example, given two classes  $A_{[0,3]}$  and  $A_{[0,2]}$ , their overlap is 3 since 3 of their time points intersect, the overlap of  $A_{[0,3]}$  and  $A_{[2,4]}$  would be 2, the overlap of  $A_{[0,3]}$  and  $A_{[3,3]}$  is 1 and the overlap of  $A_{[0,3]}$  and  $B_{[0,3]}$  is 0 since they do not have the same name. For each ontology we compute pairwise overlap between entities in an ontology with the same name. We then take an average over each pairwise overlap value and multiply the result by the number of times an entity with the same name appears in axioms, to account for the uneven usage of entities throughout an ontology. This gives overlap values for each name of an entity in an ontology. We then take an average over these values to act as a measure for the *overlap density* of an ontology. We expect that those ontologies with large time lines and small interval sizes will have a generally small overlap density, and those with small time lines but large intervals will have a large overlap density. Overlap density will be used as the varying measure to observe the effect on reasoning time for  $TEMP_{\mathcal{EL}}$ . It incorporates both the time line and the intervals and provides us with an overall insight into the temporal nature of an ontology.

No performance evaluation is performed for  $TEMP_{\mathcal{ALC}}$ . Upon preliminary testing we found  $TEMP_{\mathcal{ALC}}$  performed very slowly on real ontologies, often running out of memory. As previously mentioned  $TEMP_{\mathcal{ALC}}$  is only meant to act as a proof of concept reasoner. No optimisations have been implemented, and therefore this result is unsurprising.

### Experiment 3. Additional Entailments of $TEMP_{\mathcal{EL}}$

We are also interested in seeing whether or not having temporal versions of ontologies actually results in additional information being entailed. With the temporal ontologies generated in Experiment 2, we plan to compare the number of additional atomic entailments against their standard non temporal versions. We aim to see which of the temporal versions, if any, offer more entailments and what specific role the intervals and time line together play on the amount of entailments inferred. We again plan to use the overall density as a measurement in this experiment.

### 7.4.3 Hypotheses

**Hypothesis 1:**  $TEMP_{\mathcal{EL}}$  and  $TEMP_{\mathcal{ALC}}$  are correct implementations of the reasoning problems for  $\mathcal{EL}_{[]}$  and  $\mathcal{ALC}_{[]}$  ontologies, specifically, classification for  $\mathcal{EL}_{[]}$ , and classification, satisfiability and consistency for  $\mathcal{ALC}_{[]}$

**Hypothesis 2:** Average entity interval overlap density correlates positively with classification time of  $TEMP_{\mathcal{EL}}$

**Hypothesis 3:** Average entity interval overlap density correlates positively with ontology entailment count

### 7.4.4 Experimental Setup

Each experiment was conducted on a single Linux based machine (64 bit) with a 2.7 GHz i7 processor. Each experiment was given a dedicated 16GB of DDR3 RAM from an available 64GB. All experiments were run using the Java 8 virtual machine with OWL API version 3.5.2.

### 7.4.5 Experiment 1 Results

**$\mathcal{EL}$  Ontologies in The OBO Foundry** We used the `Validator` class, specifically the method `boolean isOntologyEL(OWLOntology o)` introduced previously to identify those ontologies in the OBO foundry snapshot that were inside the DL  $\mathcal{EL}$ . 22 were found to be inside  $\mathcal{EL}$ .

**Zero-Interval Ontologies** We created a corpus of 22 valid  $OWL_{[]}$  ontologies, by taking each of the 22 OBO Foundry ontologies and adding annotation properties which encoded zero-intervals onto each class and role name. The start index annotation and end index annotation properties have the following IRIs respectively: `<tempEL.startIndex>`, `<tempEL.endIndex>`. For representing the base name of each entity, the original IRI was used. This has the annotation property with the following IRI: `<tempEL.baseName>`. We verified that each ontology was in fact valid  $OWL_{[]}$  by using the `Validator`'s method `isOntologyCorrectlyTemporal(OWLOntology o, ...)`. All ontologies were valid  $OWL_{[]}$ .

**Atomic Entailment Correctness:  $\text{TEMP}_{\mathcal{EL}}$  &  $\text{ELK}$**  We ran classification for both  $\text{TEMP}_{\mathcal{EL}}$  and  $\text{ELK}$  on each ontology in  $\mathcal{O}_0$ . For  $\text{TEMP}_{\mathcal{EL}}$ , we set a time out of 60 minutes to classify each ontology. Out of 22 ontologies, only 13 successfully finished within this time. For the 13 that did terminate within the time out bound, all inferred atomic entailments that  $\text{TEMP}_{\mathcal{EL}}$  produced were also present in  $\text{ELK}$ , and all inferred atomic entailments that  $\text{ELK}$  produced were also present in  $\text{TEMP}_{\mathcal{EL}}$ , proving the correctness of standard entailments in  $\text{TEMP}_{\mathcal{EL}}$ .

**Manual Verification of  $\text{TEMP}_{\mathcal{EL}}$  &  $\text{TEMP}_{\mathcal{ALC}}$**  10  $\mathcal{EL}_{[]}$  and 10  $\mathcal{ALC}_{[]}$  ontologies were manually created. The following is an example of one of the  $\mathcal{EL}_{[]}$  ontologies manually created and used in our experiment:

$$\{A_{[3]} \sqsubseteq B_{[3,7]}, A_{[0,1]} \sqsubseteq B_{[0,3]}, A_{[0,1]} \sqsubseteq \exists R_{[3,5]}.D_{[3,6]}, \\ D_{[3]} \sqsubseteq E_{[3]}, \exists R_{[4,5]}.(D_{[4,6]} \sqcap E_{[3]}) \sqsubseteq B_{[4,7]}, B_{[2,6]} \sqsubseteq C_{[0]}\}$$

$\text{TEMP}_{\mathcal{EL}}$  produced the following entailments (not involving  $\top_\lambda$  and tautologies such as  $A \sqsubseteq A$ )

$$D_{[3,6]} \sqsubseteq D_{[4,6]}, D_{[3,6]} \sqsubseteq E_{[3,3]}, D_{[3,6]} \sqsubseteq D_{[3,3]}, \\ B_{[2,6]} \sqsubseteq C_{[0,0]}, B_{[3,7]} \sqsubseteq B_{[4,7]}, \\ A_{[3]} \sqsubseteq B_{[3,7]}, A_{[3]} \sqsubseteq B_{[4,7]}, D_{[3]} \sqsubseteq E_{[3,3]}, \\ A_{[0,1]} \sqsubseteq B_{[2,6]}, A_{[0,1]} \sqsubseteq B_{[3,7]}, A_{[0,1]} \sqsubseteq B_{[0,3]}, \\ A_{[0,1]} \sqsubseteq B_{[4,7]}, A_{[0,1]} \sqsubseteq C_{[0,0]}$$

Each of the three participants were fully briefed on both the syntax and semantics of each  $\mathcal{EL}_{[]}$  and  $\mathcal{ALC}_{[]}$ , and were shown examples of the features of each logic, explaining important properties such as the possible world semantics, interval overlapping properties and rigidity. Each participant was familiar and considered to be well versed in DLs and understood advanced notions such as atomic entailments, reasoning, classification, etc. They were then presented with each of the sample ontologies and were asked to record any atomic subsumptions that they thought the ontology entailed. For the  $\mathcal{EL}$  case, all participants were in agreement with the entailments of all ontologies. They also exactly matched the entailments the reasoner produced. The same results held for the  $\mathcal{ALC}$  case. All three participants matched all atomic subsumption entailments that  $\text{TEMP}_{\mathcal{ALC}}$  produced, all participants matched satisfiability checks with  $\text{TEMP}_{\mathcal{ALC}}$ . For consistency checks,

again, all three participants matched  $TEMP_{ACC}$ 's output.

These results support Hypothesis 1.

All ontologies used in this experiment are available at <http://www.cs.man.ac.uk/~leoj/thesis/>.

O	O-CT	CT-E	O-E1	O-E2
aeo	0.96	0.85	0.89	0.87
apo	0.99	0.87	0.91	0.90
ddpheno	0.99	0.88	0.90	0.88
eo	0.99	0.92	0.89	0.89
fix	0.98	0.88	0.91	0.91
mmo	0.98	0.87	0.92	0.91
pw	0.91	0.95	0.92	0.92
rex	0.98	0.86	0.92	0.92
symp	0.99	0.93	0.92	0.90
taxrank	0.97	0.97	0.93	0.93
trans	0.77	0.52	0.79	0.67
uo	0.99	0.88	0.88	0.88
wbphenotype	0.99	0.89	0.91	0.91

Table 7.2: Correlation (Spearman Coefficient) Table for all ontologies.  $\mathcal{O}$ : Ontology, O-CT: Overlap/Classification Time, CT-E: Classification Time/Entailment Count O-E1: Overlap/Entailment Count, O-E1: Overlap/Entailment Count with Temporal Tautologies Removed,

### 7.4.6 Experiment 2 Results

The relationship of classification time to average interval overlap can be observed in Figure 7.1. A Spearman coefficient was computed to assess the relationship. For all ontologies, the relationship is positive: the higher the average overlap, the longer it takes to classify the ontology. The slightly higher variance for the TRANS ontology can be simply explained by measurement error, which is higher for measurements in the sub second area. The correlation of the two metrics ranges from 0.99 (nearly perfect correlation) to 0.77 for TRANS, which was also the only ontology for which the correlation was less than 0.9. We can conclude that Hypothesis 2 holds. A deeper analysis of the nature of the relationship is beyond the scope of this thesis. For most of the ontologies, the effect appears to be roughly linear. Some artefacts, most importantly the outliers in AEO, require further investigation.

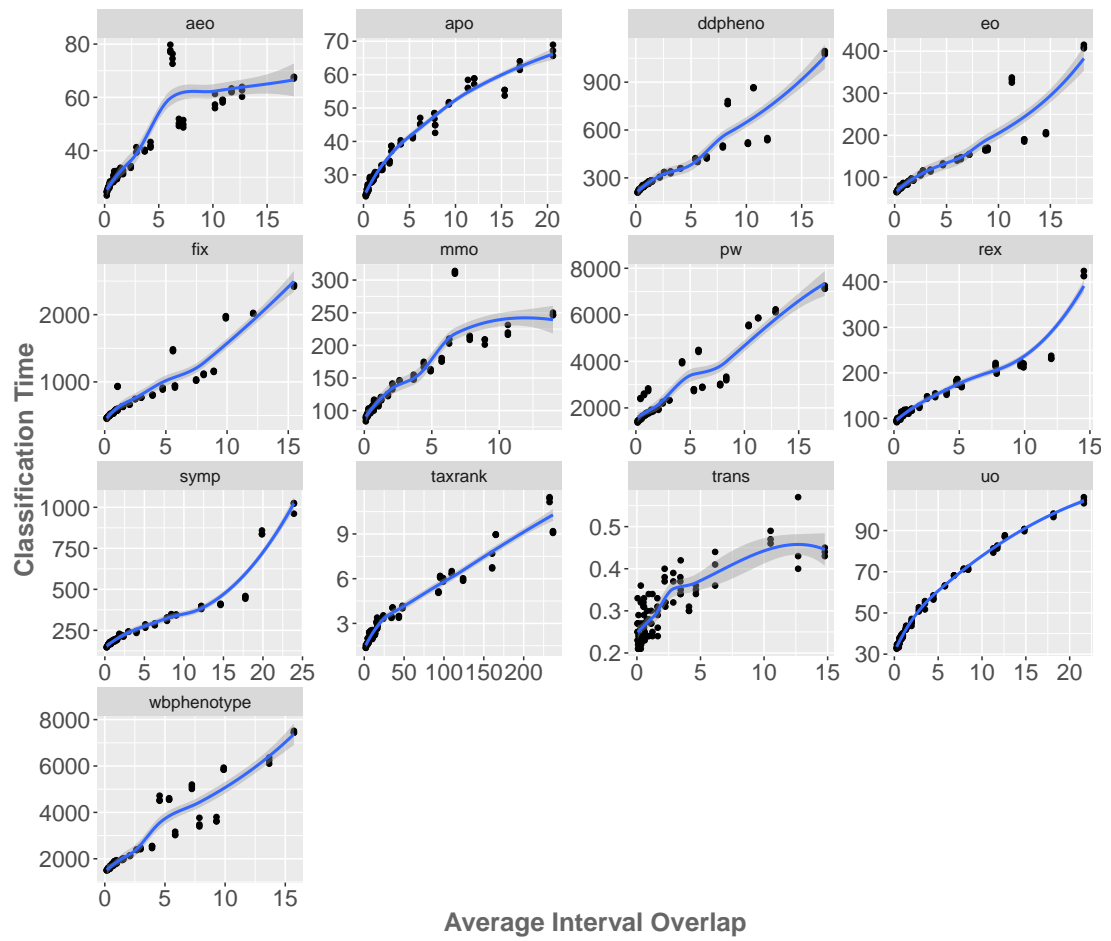


Figure 7.1: Relationship between average interval overlap and classification time in seconds.

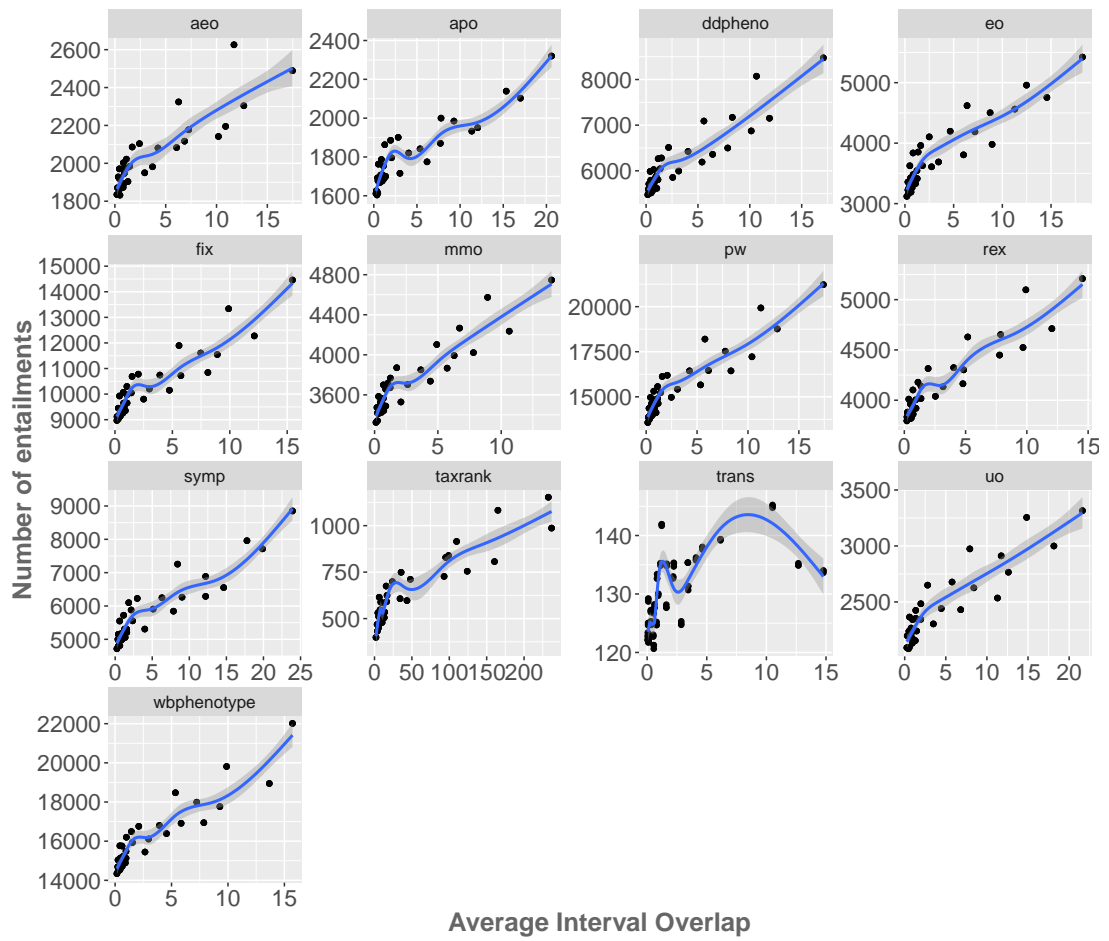


Figure 7.2: Relationship between average interval overlap and number of resulting subsumptions.

### 7.4.7 Experiment 3 Results

The relationship between the average overlap density and the number of resulting entailments can be seen in Figure 7.2. The relationship appears very similar to the one between overlap and classification time. This suggests that the computation time needed to produce additional entailments has an effect on reasoning time. Again, the TRANS ontology may be simply too small to yield any significant insight. The correlation coefficient ranges from 0.79 (TRANS) to 0.93 (PW), with most of the ontologies being around 0.9 (strong correlation). This means that the higher the overlap density, the more atomic subsumptions are computed. This follows from the fact that when the overlap density is high, then there are more concepts and roles in the ontology with the same names with intervals that are contained or overlap with each other and form more overlapping chains which contain more concepts and roles. This implies that the underlying rules in the algorithm will be fired more often than in the normal (zero-ontology) case, since more entailments now follow. Of course, by simply creating a random interval version of an ontology, we increase the signature size by introducing a new concept and role for every entity in the signature of every logical axiom, which will increase the size of a classification by default, making comparisons with the zero interval ontologies and the random interval ontologies unfair. However the signature size of every random interval ontology remains consistent so comparisons remain fair between these.

It is also the case that many of the ontologies have additional entailments of the form of *temporal tautologies* due to the way they were temporalised. A temporal tautology is one of the form  $A_{\lambda_1} \sqsubseteq A_{\lambda_2}$  where  $\lambda_2$  is contained within  $\lambda_1$ . If we temporalise an ontology with a large overlap density then we get many entailments of this form. We decide to investigate this further to see exactly how many temporal tautologies were responsible for the additional entailments. We removed all atomic entailments from the datasets that had the same name (of course this is slight overkill - an entailment of the form  $A_{[0]} \sqsubseteq A_{[1]}$  is certainly not a tautology but is still eliminated under this removal scheme). The resulting plots can be seen in Figure 7.3 and the corresponding correlations are again shown in Table 7.2. Compared to the original data sets including all entailments, there was on average a loss of 39.98% of entailments. We are currently unsure how many of these were tautologies as this would require further reasoning. However we can see there is still an obvious growth in entailments when the overlap density

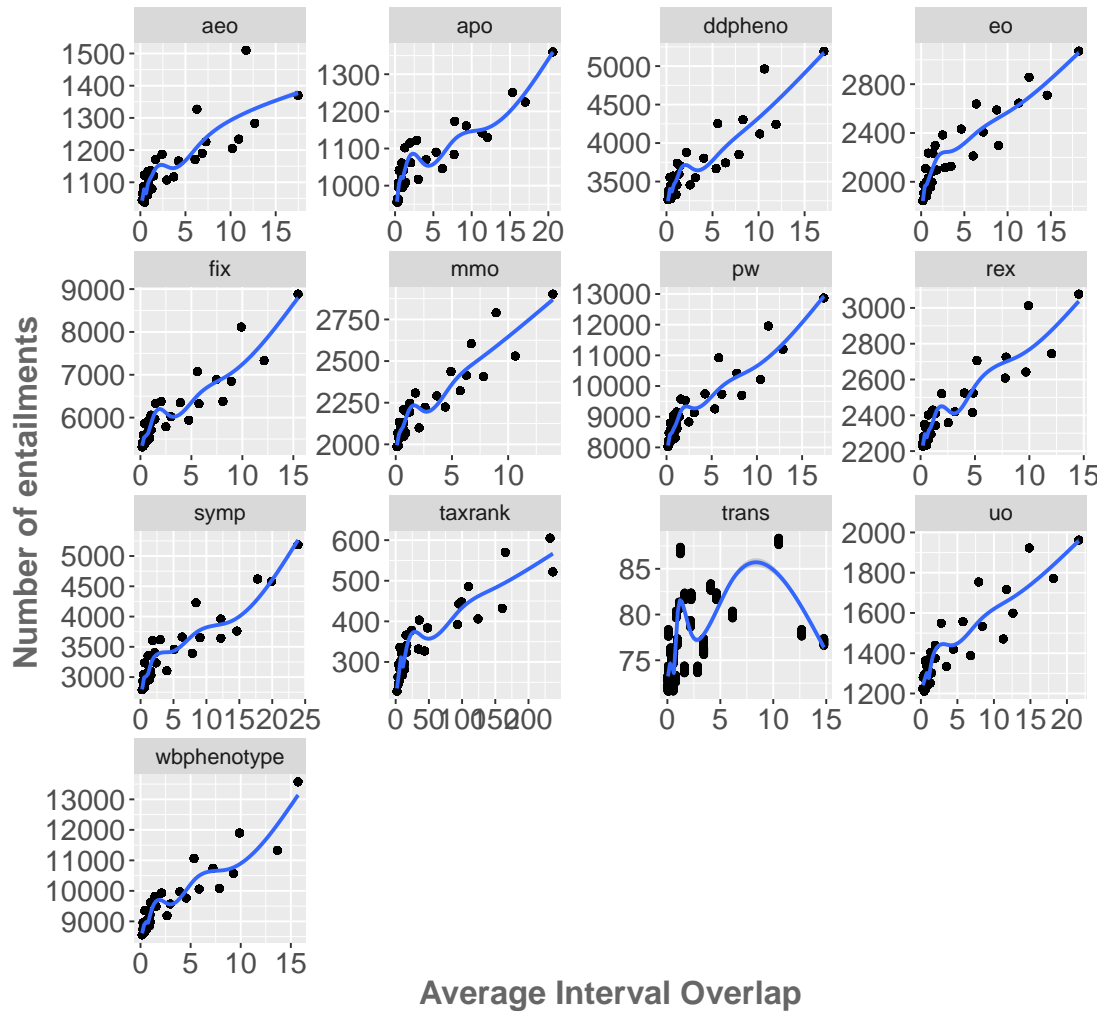


Figure 7.3: Relationship between average interval overlap and number of resulting subsumptions with tautologies removed.



increases in each ontology, again with the exception of the TRANS ontology. As can be seen comparing Figure 7.2 and 7.3 between the two data sets, the trend lines appear roughly the same, with near exact correlations (Table 7.2). The average number of entailments across all ontologies was 5,634 compared to 3,381 adopting the removal scheme. We confirm Hypothesis 3.

## 7.5 Summary

The experiments provide supporting evidence for our hypotheses. Both  $\text{TEMP}_{\mathcal{EL}}$  and  $\text{TEMP}_{\mathcal{ALC}}$  are correct proof of concept reasoners for  $\mathcal{EL}_{[]}$  and  $\mathcal{ALC}_{[]}$ . We can see that, by adding a temporal dimension there can be an advantage gained in terms of additional entailments from a temporal knowledge base when compared to an atemporal (static) knowledge base, and overlap density is a key factor here, although increasing reasoning time may likely play a crucial role. Of course, not all temporal ontologies may have a dense overlap structure, in which case the number of additional entailments may not have a significant impact - we know that having a simple temporal dimension (zero-ontologies) has no effect of additional entailments - but may simply gain an advantage in terms of a modelling perspective. In the end, this will likely be solely dependent on the temporal information present in ontologies, but it is clear that an advantage can be gained.

# Chapter 8

## Conclusion

We have investigated the requirements for temporal modelling of bio-health ontologies and introduced several new extensions of Description Logics (DLs) and introduced implementations of OWL reasoners to capture the temporal aspects that a collection of bio-health ontologies exhibit. More specifically:

- Amongst a vast set of temporal features we identified in a survey of the OBO Foundry, there were clear distinctions between important and unimportant features, the most important being the *rigid* feature.
- Out of the current temporal extensions and representations we evaluated ( $LTL_{DL}$  &  $CTL_{DL}$ ,  $\mathcal{ALC}(\mathcal{D})$ , FL) against the TRs we extracted, none were sufficient, although  $LTL_{\mathcal{ALC}}$  has a clear advantage although suffered greatly when it came to modelling the most important feature (*rigidity*) and was not suitable in practice.
- The new set of logics we created,  $[x]$ , tailored to the TRs, were in some cases better than all other logics evaluated, but did have some major drawbacks. However it did succeed at meeting some of the most important temporal features.
- The implementations of the Reasoners for  $[x]$  were shown to be at least practical in the prototype phase and showed promising results for the benefits of using a temporal logic for temporal modelling in ontologies.

The following is a summary of our research contributions:

- **Temporal Requirements Identification** (*Chapter 3*)

- We temporally annotated the Relation Ontology [SCK<sup>+</sup>05], an upper level ontology of relations used amongst ontologies in the OBO Foundry [SAR<sup>+</sup>07], by annotating all relations with their temporal information which we called *temporal features*, in a precise and structured way, to form the Temporal Relation Ontology (TRO).
- We then developed an importance measuring scheme which used TRO to parse the OBO Foundry to determine which temporal features were most important, which temporal features were always used together, which annotations were most important and so on. This allowed us to develop 15 TRs to act as a basis for the evaluation of temporal extensions to DLs to determine their suitability.

- **Temporal Representations Evaluation (*Chapter 4*)**

- We then evaluated 3 current Temporal Description Logics (TDLs),  $\text{LTL}_{\mathcal{ALC}}$  &  $\text{CTL}_{\mathcal{ALC}}$ ,  $\mathcal{ALC}(\mathcal{D})$  and FL against the 15 TRs to see which if any was most suited to modelling the temporal patterns found in bio-health ontologies. We made clear comparisons between each logic, showing the advantages and disadvantages of each. When it was clear that no logic was suited for the modelling, we set out to introduce a new TDL.

- **[x]- Defining a new TDL (*Chapter 5*)**

- We then introduced 4 new TDLs, contained within [x], each allowing for a different combination of syntax and semantics. The logics were based on annotating standard DL concepts and roles with time point intervals or variable based time point intervals. The logics were tailored to the TRs, unlike each previous logic evaluated, and were partially designed based on the positive aspects of each previously evaluated logic where possible.
- We then went on to evaluate each new logic against the TRs, comparing with each previously evaluated logic.

- **[x] Reasoning problems (*Chapter 6*)**

- We then went on to define several of the reasoning problems of the [x] logics. Since classification relies on a fixed set of concept names which

is not so easily defined in  $[\mathbf{x}]$ , we had to adapt classification (and other reasoning problems) to the case on concepts involving intervals.

- We proved decidability results for fragments of  $[\mathbf{x}]$ , whilst leaving several undecided.
- We showed that in several fragments, one of the most desired features, rigidity, was still decidable, which was one of the most positive results.

- **$[\mathbf{x}]$  Implementations of  $[\mathbf{x}]$  Reasoners (*Chapter 7*)**

- We then introduced two OWL Reasoners that we implemented,  $\text{TEMP}_{\mathcal{EL}}$  and  $\text{TEMP}_{\mathcal{ALC}}$ . Each reasoner implements all the reasoning procedures for the TDLs of  $\mathcal{EL}_{[\ ]}$  and  $\mathcal{ALC}_{[\ ]}$ , which are fragments of  $[\mathbf{x}]$ . We introduce them only as proof of concept reasoners, or prototypes, but also show they have sufficient use in practice.
- We conducted several experiments to show their correctness, their use in practice, and the benefits of their respective logics as substitutes for standard DLs.

# Chapter 9

## Outlook

Although we have covered a wide range of concepts in this thesis, our research has left a lot of open doors for further exploration. We give a summary of prospective extensions of our work that we have considered which can be seen as *next steps* to continue our research.

**Extending the Temporal Requirements** Our first point of interest would be to extend the temporal requirements, or further constrain them to make them more *fine-grained*, so to speak. Our first point of contact would be to solve the issue of our smart matching problem we identified. This was the problem of determining whether a relation being used in an ontology is in fact a relation from the Relation Ontology (RO) if their IRIs did not match. This will enable us to get more accurate results and give us clearer and more accurate temporal requirements. It seems the most appropriate way to do this would be to contact the ontology developers. Although this might be a difficult and time-consuming task, there seems no other way to solve this problem. A single ontology does not use many of the relations from RO, and a simple questionnaire sent out to the the developers of the ontology asking them to verify if the relations being used were meant to be interpreted as the relations from RO would certainly not be a difficult task. However in the process of contacting the developers we could take this opportunity to also conduct a survey where we could reaffirm our decision on the annotating scheme of the Temporal RO (TRO) to see if the temporal features we annotated the relations with were the correct ones or whether ontology developers would have another opinion for example.

The next thing would be to consider a wider range of ontologies, for example

using an additional corpus, such as BioPortal. BioPortal [NSW<sup>+</sup>09] is another repository for ontologies in the biomedical domain, containing not only many of the ontologies in the OBO Foundry, but many more. The more ontologies we use in our survey, the more results we will have to base our requirements upon.

During our evaluation of not only our newly introduced logics but also the three logics evaluated previously, we found that many ontologies had different options for the temporal dimensions that they encoded (mainly deciding whether to use quantitative vs qualitative time lines), and this is something that the temporal requirements did not capture quite so easily. This was not the fault of the requirements themselves or the way we derived requirements, but more in relation to *where* we derived the requirements from. The relation ontology did not encode this information since this type of information is specific to an ontology itself. So it would be wise to check each individual ontology for this information, which again could be done whilst contacting ontology developers. This information could then be added as a separate requirement.

It is obvious that the more we investigate individual ontologies, the more fine grained we can make our temporal requirements. This was not in our initial plan, due not only to our time constraints but also to not knowing the amount of variation in the ontologies we were inspecting. But with the survey we have conducted, we have gathered enough information to be able to now make these decisions and proceed further.

**A Framework for Entity Importance** Whilst extracting the temporal requirements, we developed a simple scheme to measure the importance of both temporal features and temporal annotations based on two metrics called *coverage* and *impact*. Together these two metrics formed a measure of *importance*. Both the coverage and impact metrics were calculated using explicit information in an ontology's signature, specifically their usage in axioms. We developed this novel approach since no research had gone into ways to measure the importance of entities in ontology's. It would certainly be interesting to see if this system we developed could be used on a more *general* basis. Our importance measuring system could easily be turned into a framework where instead of measuring the importance of individual temporal features, we can easily measure the importance of specific OWL entities, for example OWL classes or OWL relations in an ontology or even entire corpus. As mentioned in Chapter 3, we also did

attempt to use formal concept analysis (FCA) when measuring importance, but it just so happened that in this particular survey it bore no particular benefit. It may be the case that when using more ontologies or even a different use case, for example, if we were to repeat the experiment with BioPortal, it could play a more important role. Developing a framework that also incorporated FCA may have huge benefits not only for this survey but for more general applications that require entity importance in ontologies. This framework is certainly something to be considered as it could not only have benefits in our applications but others also.

**Logical extensions and Similarities** Unfortunately, we only managed to prove certain decidability results for small fragments of  $[\mathbf{x}]$ . Although these results were positive, some of the more important fragments were particularly difficult to solve. Our primary goal was only to prove decidability in the presence of some form of rigidity without having severe restrictions in the DL expressiveness, which is what we achieved, to some extent. The first steps of pushing the logic further would be to first prove the decidability or undecidability of the unrestricted version of  $\mathcal{EL}_{[x]}$ , i.e. one without the “*future only*” restriction. We believe that a result in this logic would enable us to immediately gain decidability (or undecidability) results in the logics left unproven in this thesis, namely  $\mathcal{EL}_{[x][\ ]}$ ,  $\mathcal{ALC}_{[x]}$  and  $\mathcal{ALC}_{[x][\ ]}$ . Depending on the result, if it was indeed decidable then there are two steps we can take. We could either extend the logic in the DL dimension, or in the temporal dimension. For the DL dimension, we would hope to get as close to *SROIQ* as possible, or at least to an OWL profile, such as the OWL 2 EL profile, whilst remaining decidable. We would first hope to see what effect adding property characteristics to fragments of  $[\mathbf{x}]$  would have on complexity. Many of the relations we evaluate do indeed exhibit characteristics such as transitivity, reflexivity, role chains etc, and this would be something immediately important to further explore. For the temporal dimension, we could possibly try to add in some of the features that were missing, such as eventuality. If the unrestricted versions of the logic were indeed undecidable then our focus would be solely on the quantitative versions, pushing more in the DL dimension than the temporal dimensions. We believe that  $[\mathbf{x}]$  is already a very simple TDL, restricting it further to maintain decidability is not something we desire and we believe would only detriment its performance w.r.t. the TRs.

We have also only shown loose complexity bounds for each logic we have proved so far. We plan to tighten these bounds and show complete complexity class membership to shed more light on the true complexity of these logics and their relationships to other logics.

During the evaluation we also found that the logic was very similar to the logics of  $LTL_{DL}$  combinations. This was unsurprising since the logic's semantics was built on the same structure, aided by the evaluation results. There is no doubt that if we possibly extended the logics, for example by adding eventuality, it may be possible that the logics will eventually be equivalent, or one being sub languages of the other, most likely  $[x]$  being a sub language of  $LTL_{DL}$  with role conjunctions for example. There is also some speculation that the quantitative versions of  $[x]$  may even be reducible to standard OWL. Of course these are important issues that should certainly be looked in to.

**Reasoner Optimisations and Extensions** The reasoners we introduced in Chapter 7 have been presented only as proof of concept, and they are there to act only as prototypes to show that reasoning is possible in TDLs and benefits can be gained by switching to a TDL. The reasoners can be used, are readily available and from our experiment results we have shown they can actually be used with numerous ontologies in the OBO Foundry, at least for the  $TEMP_{\mathcal{EL}}$  case. In terms of next steps, our first point of contact would be to develop these reasoners enough to the point where they would be comparable with current state-of-the-art OWL reasoners. This would include implementing several optimisations to speed up the reasoning time which currently is its only drawback. It would also be beneficial to develop a plug-in for the Protégé ontology editor to support the annotation of classes and properties with intervals. This would enable ontology developers to interact directly with temporal ontologies through a dedicated interface, instead of through means of annotations in a programming environment, as is currently the case.

Since we have decidability results for other sub languages of  $[x]$ , namely  $\mathcal{EL}_{[][x]}$ ,  $\mathcal{ALC}_{[][x]}$ ,  $\mathcal{EL}_{[x]}$  and  $\mathcal{ALC}_{[x]}$ , implementations of reasoning services for these logics will also be made available (they are currently in development).



# Bibliography

- [ABB<sup>+</sup>00] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene Ontology: Tool for the Unification of Biology. *Nat Genet*, 25(1):25–29, 05 2000.
- [AF00] Alessandro Artale and Enrico Franconi. A Survey of Temporal Extensions of Description Logics. *Ann. Math. Artif. Intell.*, 30(1-4):171–210, 2000.
- [AF05] Alessandro Artale and Enrico Franconi. Temporal Description Logics. In Michael Fisher, Dov M. Gabbay, and Lluís Vila, editors, *Handbook of Temporal Reasoning in Artificial Intelligence*, volume 1 of *Foundations of Artificial Intelligence*, pages 375–388. Elsevier, 2005.
- [AFWZ02] Alessandro Artale, Enrico Franconi, Frank Wolter, and Michael Zharkaryashev. A Temporal Description Logic for Reasoning over Conceptual Schemas and Queries. In Sergio Flesca, Sergio Greco, Nicola Leone, and Giovambattista Ianni, editors, *Logics in Artificial Intelligence, European Conference, JELIA 2002, Cosenza, Italy, September, 23-26, Proceedings*, volume 2424 of *Lecture Notes in Computer Science*, pages 98–110. Springer, 2002.
- [All83] James F. Allen. Maintaining Knowledge about Temporal Intervals. *Commun. ACM*, 26(11):832–843, November 1983.

- [ASP09] Theodore Andronikos, Michalis Stefanidakis, and Ioannis Papadakis. Adding Temporal Dimension to Ontologies via OWL Reification. In Vassilios Chrissikopoulos, Nikolaos Alexandris, Christos Douligeris, and Spyros Sioutas, editors, *PCI 2009, 13th Panhellenic Conference on Informatics, 10-12 September 2009, Corfu, Greece*, pages 19–22. IEEE Computer Society, 2009.
- [Baa03] Franz Baader. Terminological Cycles in a Description Logic with Existential Restrictions. In Georg Gottlob and Toby Walsh, editors, *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pages 325–330. Morgan Kaufmann, 2003.
- [BBL05] Franz Baader, Sebastian Brand, and Carsten Lutz. Pushing the EL Envelope. In *Proc. of IJCAI 2005*, pages 364–369. Morgan-Kaufmann Publishers, 2005.
- [BCM<sup>+</sup>03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
- [BH91] Franz Baader and Philipp Hanschke. A Scheme for Integrating Concrete Domains into Concept Languages. In John Mylopoulos and Raymond Reiter, editors, *IJCAI*, pages 452–457. Morgan Kaufmann, 1991.
- [Bra04a] Sebastian Brandt. Polynomial Time Reasoning in a Description Logic with Existential Restrictions, GCI Axioms, and - What Else? In Ramon López de Mántaras and Lorenza Saitta, editors, *Proceedings of the 16th European Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, pages 298–302. IOS Press, 2004.
- [Bra04b] Sebastian Brandt. Reasoning in ELH w.r.t. General Concept Inclusion Axioms. Technical report, 2004.

- [BS01] Franz Baader and Ulrike Sattler. An Overview of Tableau Algorithms for Description Logics. *Studia Logica*, 69(1):5–40, 2001.
- [CE82] Edmund M. Clarke and E. Allen Emerson. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In *Logic of Programs, Workshop*, pages 52–71, London, UK, UK, 1982. Springer-Verlag.
- [CES86] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic Verification of Finite-State Concurrent Systems using Temporal Logic Specifications. *ACM Trans. Program. Lang. Syst.*, 8(2):244–263, April 1986.
- [CRGOS13] Marta Costa, Simon Reeve, Gary Grumbling, and David Osumi-Sutherland. The Drosophila Anatomy Ontology. *Journal of Biomedical Semantics*, 4(1):32, 2013.
- [DGFvdH07] Clare Dixon, M. Carmen Fernández Gago, Michael Fisher, and Wiebe van der Hoek. Temporal Logics of Knowledge and their Applications in Security. *Electr. Notes Theor. Comput. Sci.*, 186:27–42, 2007.
- [EH85] E. Allen Emerson and Joseph Y. Halpern. Decision Procedures and Expressiveness in the Temporal Logic of Branching Time. *J. Comput. Syst. Sci.*, 30(1):1–24, 1985.
- [Fis11] M. Fisher. *An Introduction to Practical Formal Methods Using Temporal Logic*. Wiley, 2011.
- [FMK07] Flavius Frasincar, Viorel Milea, and Uzay Kaymak. tOWL: Integrating Time in OWL. In Roberto De Virgilio, Fausto Giunchiglia, and Letizia Tanca, editors, *Semantic Web Information Management*, pages 225–246. Springer, 2007.
- [GHM<sup>+</sup>08] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter F. Patel-Schneider, and Ulrike Sattler. OWL 2: The Next Step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.
- [GJL12] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Carsten Lutz. Complexity of Branching Temporal Description Logics. In Luc De

- Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas, editors, *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31, 2012*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 390–395. IOS Press, 2012.
- [GJS15a] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Thomas Schneider. Lightweight Temporal Description Logics with Rigid Roles and Restricted TBoxes. In Qiang Yang and Michael Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3015–3021. AAAI Press, 2015.
- [GJS15b] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Thomas Schneider. The Complexity of Temporal Description Logics with Rigid Roles and Restricted TBoxes: In Quest of Saving a Troublesome Marriage. In Diego Calvanese and Boris Konev, editors, *Proceedings of the 28th International Workshop on Description Logics, Athens, Greece, June 7-10, 2015.*, volume 1350 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.
- [GLN01] Rajeev Goré, Alexander Leitsch, and Tobias Nipkow, editors. *Automated Reasoning, First International Joint Conference, IJCAR 2001, Siena, Italy, June 18-23, 2001, Proceedings*, volume 2083 of *Lecture Notes in Computer Science*. Springer, 2001.
- [GPSS80] Dov M. Gabbay, Amir Pnueli, Saharon Shelah, and Jonathan Stavi. On the Temporal Basis of Fairness. In Paul W. Abrahams, Richard J. Lipton, and Stephen R. Bourne, editors, *Conference Record of the Seventh Annual ACM Symposium on Principles of Programming Languages, Las Vegas, Nevada, USA, January 1980*, pages 163–173. ACM Press, 1980.
- [GSW05] Bernhard Ganter, Gerd Stumme, and Rudolf Wille, editors. *Formal*

- Concept Analysis, Foundations and Applications*, volume 3626 of *Lecture Notes in Computer Science*. Springer, 2005.
- [HB11] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL Ontologies. *Semantic Web*, 2(1):11–21, 2011.
- [HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible SROIQ. In *KR*, pages 57–67, 2006.
- [HMW01] Volker Haarslev, Ralf Möller, and Michael Wessel. The Description Logic ALCNHR+ Extended with Concrete Domains: A Practically Motivated Approach. In Goré et al. [GLN01], pages 29–44.
- [HP04] Jerry R. Hobbs and Feng Pan. An Ontology of Time for the Semantic Web. *ACM Trans. Asian Lang. Inf. Process.*, 3(1):66–85, 2004.
- [HS91] Joseph Y. Halpern and Yoav Shoham. A Propositional Modal Logic of Time Intervals. *J. ACM*, 38(4):935–962, 1991.
- [HS01] Ian Horrocks and Ulrike Sattler. Ontology Reasoning in the SHOQ(D) Description Logic. In Nebel [Neb01], pages 199–204.
- [Kaz08] Yevgeny Kazakov. SRIQ and SROIQ are Harder than SHOIQ. In Franz Baader, Carsten Lutz, and Boris Motik, editors, *Proceedings of the 21st International Workshop on Description Logics (DL2008), Dresden, Germany, May 13-16, 2008*, volume 353 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
- [KKS12] Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. ELK Reasoner: Architecture and Evaluation. In Ian Horrocks, Mikalai Yatskevich, and Ernesto Jiménez-Ruiz, editors, *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012*, volume 858 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.
- [KSH12] Markus Krötzsch, Frantisek Simancik, and Ian Horrocks. A Description Logic Primer. *CoRR*, abs/1201.4089, 2012.

- [LMS02] François Laroussinie, Nicolas Markey, and Philippe Schnoebelen. Temporal Logic with Forgettable Past. In *17th IEEE Symposium on Logic in Computer Science (LICS 2002), 22-25 July 2002, Copenhagen, Denmark, Proceedings*, pages 383–392. IEEE Computer Society, 2002.
- [LPZ85] Orna Lichtenstein, Amir Pnueli, and Lenore D. Zuck. The Glory of the Past. In Rohit Parikh, editor, *Logics of Programs, Conference, Brooklyn College, June 17-19, 1985, Proceedings*, volume 193 of *Lecture Notes in Computer Science*, pages 196–218. Springer, 1985.
- [LSP14] Jared Leo, Ulrike Sattler, and Bijan Parsia. Temporalising EL Concepts with Time Intervals. In *Informal Proceedings of the 27th International Workshop on Description Logics, Vienna, Austria, July 17-20, 2014.*, pages 620–632, 2014.
- [Lut01a] Carsten Lutz. Interval-Based Temporal Reasoning with General TBoxes. In Nebel [Neb01], pages 89–96.
- [Lut01b] Carsten Lutz. NEXPTIME-Complete Description Logics with Concrete Domains. In Goré et al. [GLN01], pages 45–60.
- [Lut02] Carsten Lutz. Description Logics with Concrete Domains-A Survey. In Philippe Balbiani, Nobu-Yuki Suzuki, Frank Wolter, and Michael Zakharyashev, editors, *Advances in Modal Logic*, pages 265–296. King’s College Publications, 2002.
- [Lut04] Carsten Lutz. Combining Interval-Based Temporal Reasoning with General TBoxes. *Artif. Intell.*, 152(2):235–274, 2004.
- [LWZ08] Carsten Lutz, Frank Wolter, and Michael Zakharyashev. Temporal Description Logics: A Survey. In Stéphane Demri and Christian S. Jensen, editors, *TIME*, pages 3–14. IEEE Computer Society, 2008.
- [Mar03] Nicolas Markey. Temporal Logic with Past is Exponentially more Succinct, Concurrency Column. *Bulletin of the EATCS*, 79:122–128, 2003.

- [MFK12] Viorel Milea, Flavius Frasincar, and Uzay Kaymak. tOWL: A Temporal Web Ontology Language. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 42(1):268–281, 2012.
- [Neb01] Bernhard Nebel, editor. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*. Morgan Kaufmann, 2001.
- [NSW<sup>+</sup>09] Natalya Fridman Noy, Nigam H. Shah, Patricia L. Whetzel, Benjamin Dai, Michael Dorf, Nicholas Griffith, Clement Jonquet, Daniel L. Rubin, Margaret-Anne D. Storey, Christopher G. Chute, and Mark A. Musen. BioPortal: Ontologies and Integrated Data Resources at the Click of a Mouse. *Nucleic Acids Research*, 37(Web-Server-Issue):170–173, 2009.
- [PMG<sup>+</sup>15] Bijan Parsia, Nicolas Matentzoglou, Rafael S. Gonçalves, Birte Glimm, and Andreas Steigmiller. The OWL Reasoner Evaluation (ORE) 2015 Competition Report. In Thorsten Liebig and Achille Fokoue, editors, *Proceedings of the 11th International Workshop on Scalable Semantic Web Knowledge Base Systems co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, October 11, 2015.*, volume 1457 of *CEUR Workshop Proceedings*, pages 2–15. CEUR-WS.org, 2015.
- [Pnu77] Amir Pnueli. The Temporal Logic of Programs. In *FOCS*, pages 46–57. IEEE Computer Society, 1977.
- [SAR<sup>+</sup>07] Barry Smith, Michael Ashburner, Cornelius Rosse, Jonathan Bard, William Bug, Werner Ceusters, Louis J. Goldberg, Karen Eilbeck, Amelia Ireland, Christopher J. Mungall, OBI Consortium, Neocles Leontis, Philippe Rocca-Serra, Alan Ruttenberg, Susanna-Assunta A. Sansone, Richard H. Scheuermann, Nigam Shah, Patricia L. Whetzel, and Suzanna Lewis. The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration. *Nature biotechnology*, 25(11):1251–1255, November 2007.
- [SC85] A. P. Sistla and E. M. Clarke. The Complexity of Propositional Linear Temporal Logics. *J. ACM*, 32(3):733–749, July 1985.

- [Sch90] Albrecht Schmiedel. Temporal Terminological Logic. In Howard E. Shrobe, Thomas G. Dietterich, and William R. Swartout, editors, *Proceedings of the 8th National Conference on Artificial Intelligence. Boston, Massachusetts, July 29 - August 3, 1990, 2 Volumes.*, pages 640–645. AAAI Press / The MIT Press, 1990.
- [Sch91] Klaus Schild. A correspondence theory for terminological logics: Preliminary report. In *In Proc. of IJCAI-91*, pages 466–471, 1991.
- [Sch93] Klaus Schild. Combining terminological logics with tense logic. In Miguel Filgueiras and Luís Damas, editors, *EPIA*, volume 727 of *Lecture Notes in Computer Science*, pages 105–120. Springer, 1993.
- [SCK<sup>+</sup>05] Barry Smith, Werner Ceusters, Bert Klagges, Jacob Köhler, Anand Kuma, Jane Lomax, Chris Mungall, Fabian Neuhaus, Alan Rector, and Cornelius Rosse. Relations in Biomedical Ontologies. *Genome Biology*, 6(5):R46, 2005.
- [SCS11] Stefan Schulz, Ronald Cornet, and Kent A. Spackman. Consolidating SNOMED CT’s Ontological Commitment. *Applied Ontology*, 6(1):1–11, 2011.
- [SdCH<sup>+</sup>07] Nicholas Sioutos, Sherri de Coronado, Margaret W. Haber, Frank W. Hartel, Wen-Ling Shaiu, and Lawrence W. Wright. NCI Thesaurus: A Semantic Model Integrating Cancer-related Clinical and Molecular Information. *Journal of Biomedical Informatics*, 40(1):30–43, 2007.
- [SSS91] Manfred Schmidt-Schauß and Gert Smolka. Attributive Concept Descriptions with Complements. *Artif. Intell.*, 48(1):1–26, 1991.
- [WF06] Christopher A. Welty and Richard Fikes. A Reusable Ontology for Fluents in OWL. In *Formal Ontology in Information Systems, Proceedings of the Fourth International Conference, FOIS 2006, Baltimore, Maryland, USA, November 9-11, 2006*, pages 226–236, 2006.
- [WZ98a] Frank Wolter and Michael Zakharyashev. On the Decidability of Description Logics with Modal Operators. In Anthony G. Cohn,



- Lenhart K. Schubert, and Stuart C. Shapiro, editors, *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, June 2-5, 1998.*, pages 512–523. Morgan Kaufmann, 1998.
- [WZ98b] Frank Wolter and Michael Zakharyashev. Temporalizing Description Logics. In *In Proceedings of FroCoS'98*, pages 104–109, 1998.