TECHNICAL UNIVERSITY OF CRETE

SCHOOL OF ELECTRONIC AND COMPUTER
ENGINEERING

CHRONOS: A Reasoning Engine for Qualitative Temporal Information in OWL

Anagnostopoulos Eleftherios

eanagnostopoulos@hotmail.com

Diploma Thesis

Committee

Euripides G.M Petrakis, Professor and Laboratory Director (Supervisor)

Stavros Christodoulakis, Professor and Laboratory Director

Michail G. Lagoudakis, Associate Professor

November 3, 2013

Acknowledgements

Abstract

We propose CHRONOS, a system for reasoning over temporal information in OWL ontologies. Representing both qualitative temporal (i.e., information whose temporal extends are unknown, such as *"before"* or *"after"* for temporal relations) in addition to quantitative information (i.e., where temporal assertions are defined precisely e.g., using dates) is a distinctive feature of the proposed approach. Qualitative representations are very common in natural language expressions such as in free text or speech and can be proven to be valuable in the Semantic Web. CHRONOS reasoner applies over temporal relations, infers implied relations, detects inconsistencies and retains soundness, completeness and tractability over the supported relations using path consistency. Experimental results have demonstrated that CHRONOS runs up to several times faster compared to SOWL, a temporal reasoner for qualitative temporal information which is implemented in SWRL, and scales-up well for large data sets.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Typical use of the World Wide Web today, like search and use of information, has generated the need for intelligent tools and mechanisms, which automatically handle tasks that are commonly handled manually by users. These services must be capable of understanding the meaning of the content of Web pages and reason over their content in a way similar to humans. Semantic Web is a solution to this need by representing Web content in a form that is processable and understandable by both, machines and humans.

Semantic Web has motivated research on tools and methodologies capable of extracting, processing and representing the meaning of Web pages as well as, for reasoning and querying using this content. Ontologies, in particular, offer the means for representing high level concepts, their properties and interrelationships. Ontologies contain definitions of concepts and of their properties by means of binary relations between concepts or, between concepts and numerical (concrete) domain. Dynamic ontologies will in addition enable representation of information evolving time. Representation of dynamic features calls for mechanisms allowing for the representation of the notions of time (and of properties varying in time) within an ontology. Methods for achieving this goal include (among others) N-ary relations [1] and the 4D-fluents (perdurantist) approach [2].

Reasoning is the process of making implicit logical inferences from the explicit set of facts or statements, which constitute the core of any knowledge base. Qualitative reasoning, in particular, is the process of reasoning over symbolic information (e.g., concepts such as "before", "after") without making numerical computations. In this case, knowledge is represented using a limited vocabulary for expressing qualitative relationships between entities or qualitative categories of numerical values. The motivation for using a qualitative approach is that it is closer to the way humans represent and reason about commonsense knowledge. Another motivation is that it is possible to deal with incomplete knowledge.

1.1 Problem Definition

Representing temporal information in ontologies resorts to OWL. However, the syntactic restriction of OWL to binary relations complicates the representation of temporal properties: A property holding for a specific time instant or interval is in fact a ternary relation involving three objects (an object, a subject and a time instant or interval) which cannot be represented directly by a single OWL statement. A standard fix to this problem is to represent the temporal relation by a set of binary ones.

Approaches for dealing with ternary relations include (among others) N-ary relations [1] and the 4D-fluents (perdurantist) approach [2]. However, these are rather complicated representations that cannot be handled directly by a standard OWL reasoner like Pellet [3]. For example, Pellet might not be able to infer new relations or detect inconsistencies in an N-ary representation. Reasoning and query answering over qualitative temporal information is the problem this work is dealing with.

In previous works [4], both, the N-ary relations and the 4D-fluents approaches are enhanced with qualitative (in addition to quantitative) temporal expressions allowing for the representation of temporal intervals with unknown starting and ending points by means of their relation (e.g., "before", "after") to other time intervals. Reasoning is realized by means of SWRL rules implementing the allowable compositions over the supported (tractable) relation set combined with OWL 2.0 constructs (e.g., for declaring disjoint properties) ensuring path consistency while being sound and complete [5]. Reasoning is embedded within the ontology and can be executed by a general purpose reasoner such as Pellet[1].

Although fast and intuitive, this approach still suffers from the following disadvantages: (a) Relying in a general purpose reasoner, it is not possible to accommodate any optimizations within the implementation of path consistency (resorting entirely to the performance of the SWRL interpreter at hand) and, (b) Implementing path consistency in SWRL calls for additional temporal relations thus complicating the representation. The proposed approach tackles both these problems. CHRONOS is independent of temporal model representing the evolution of concepts in time (i.e., works with both the N-ary and the 4D-fluents representations).

1.2 Proposed Solution

We propose CHRONOS, a system for reasoning over temporal information in OWL ontologies. Following the example of Pellet-Spatial [6] and CHOROS [7], CHRONOS is a dedicated temporal reasoning engine implemented in Java which allows for incorporating certain optimizations in implementing path consistency. The proposed solution extends the OWL-DL reasoning features of Pellet [3] reasoner with qualitative temporal reasoning capabilities. It supports checking of

---

[1]http://clarkparsia.com/pellet/

consistency and computation of new temporal inferences resulting from a set of asserted relations. Furthermore, answering SPARQL queries over a set of temporal and non-temporal relations is also supported. The last ones are expressed in RDF/OWL and can be applied on arbitrary domain ontologies.

For assessing the performance of CHRONOS we carried-out a series of runtime experiments involving comparisons of CHRONOS against SOWL (its SWRL counterpart) in both the average and worst cases. The experimental results demonstrate that CHRONOS runs several times faster than SOWL and scales-up well for large relation sets.

1.3   Thesis Outline

Related work in the field of knowledge representation is discussed in chapter 2. This includes issues related to Semantic Web such as representing information of time and space in ontologies, as well as reasoning using this information. Earlier work related to representation, reasoning and querying over temporal and spatial information are also presented and discussed. CHRONOS is presented in chapter 3. Experimental results are presented in chapter 4, followed by conclusions and issues for future work in chapter 5.

Chapter 2

Background and Related Work

2.1 Semantic Web

The Semantic Web[1] is a vision for the future of the Web in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available in Web pages. One of the most significant drawbacks of the Web, in its current form, is that Web pages are designed to be read by people, not by machines. Thus, machines cannot accomplish complex tasks, such as on-line shopping, travel planning or ticket reservations without human intervention. Automating these tasks require that machine understandable semantics of Web pages and data associated with them become also available in a machine readable form. The requirement of machine interpreted semantics is the core idea of the Semantic Web vision.

2.1.1 Ontologies

Ontology[2] is a means for expressing, in a machine readable form, concepts, properties of concepts and their relations that describe knowledge on a subject or an area of concern. Ontologies can also be used to classify the concepts (e.g., in hierarchies of concepts) and define possible constraints on using these concepts. In practice, ontologies can be very complex (with several thousands of terms) or very simple (describing one or two concepts only). In recent years, ontologies have been adopted in many business and scientific communities as a way to share, reuse and process domain knowledge. In nowadays, they are used in applications such as scientific knowledge portals, information management and integration systems, electronic commerce and semantic web services.

Ontologies are the basic building blocks for inference in the Semantic Web. Their role is to help data integration. Consider, for example, the application of ontologies in the field of health care. In medicine, ontologies are used to represent

---

[1] http://www.w3.org/standards/semanticweb/
[2] http://www.w3.org/standards/semanticweb/ontology

knowledge about symptoms, diseases, and treatments. Pharmaceutical companies use them to represent information about drugs, dosages, and allergies. Combining this knowledge from the medical and pharmaceutical communities with patient data enables a whole range of intelligent applications such as decision support tools that search for possible treatments; systems that monitor drug efficacy and possible side effects; and tools that support epidemiological research. Ontologies also provide the means for better knowledge management in a domain in general. Libraries, museums, newspapers, government portals, enterprises, social networking applications, and other communities that manage large collections of books, historical artifacts, news reports, business glossaries, blog entries and other items can now use ontologies, using standard formalisms, to leverage the power of linked data.

Common components of ontologies include:

Individuals: instances or objects.

Classes: sets, collections, concepts or types of objects.

Attributes: aspects, properties, features, characteristics, or parameters that objects and classes can have.

Relations: ways in which classes and individuals can be related to one another.

Function terms: complex structures formed from certain relations that can be used in place of an individual term in a statement.

Restrictions: formally stated descriptions of what must be true in order for some assertion to be accepted as input.

Rules: statements that describe the logical inferences that can be drawn from an assertion in a particular form.

Axioms: assertions (including rules) in a logical form that together comprise the overall theory that the ontology describes in its domain of application.

Events: the changing of attributes or relations.

2.1.2   Semantic Web Technologies

Resource Description Framework (RDF)

The *Resource Description Framework (RDF)*[3] is a language for representing in-
formation about resources in the World Wide Web. It is particularly intended for
representing metadata about Web resources, such as the title, author, and mod-
ification date of a Web page, copyright and licensing information about a Web
document, or the availability schedule for some shared resource. However, by
generalizing the concept of a "Web resource", RDF can also be used to represent
information about things that can be identified on the Web, even when they can-
not be directly retrieved on the Web. Examples include information about items
available from on-line shopping facilities (e.g., information about specifications,
prices, and availability), or the description of a Web user's preferences for infor-
mation delivery. RDF is intended for situations in which this information needs to
be processed by applications, rather than being only displayed to people. RDF pro-
vides a common framework for expressing this information so it can be exchanged
between applications without loss of meaning. Since it is a common framework
application, designers can leverage the availability of common RDF parsers and
processing tools. The ability to exchange information between different applica-
tions means that the information may be made available to applications other than
those for which it was originally created.

RDF is based on the idea of identifying things using Web identifiers (called
Uniform Resource Identifiers, or *URIs*), and describing resources in terms of simple
properties and property values. This enables RDF to represent simple statements
about resources as a graph of nodes and arcs representing the resources, and their
properties and values. RDF's vocabulary description language, **RDF Schema**, is
a semantic extension of RDF. It provides mechanisms for describing groups of
related resources and the relationships between these resources.

Web Ontology Language (OWL)

The objective of Semantic Web standards is to offer means for formal machine
understandable semantics of application domains. OWL[4] is an ontology language
for the Semantic Web which can be used to explicitly represent the meaning of
terms in ontologies and the relationships between those terms. OWL has more
facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and
thus it goes beyond these languages in its ability to represent machine interpretable
content on the Web.

OWL is part of the growing stack of W3C recommendations related to the Se-
mantic Web. It adds more vocabulary for describing properties and classes such as
cardinality (e.g. *"exactly one"*), equality, relations between classes (e.g. *disjoint-*

---

[3]http://www.w3.org/RDF/
[4]http://www.w3.org/TR/owl-features/

ness), richer typing of properties, characteristics of properties (e.g. *symmetry*), and enumerated classes.

OWL provides three increasingly expressive sublanguages:

> ***OWL-Full***, which provides maximum expressiveness and the syntactic freedom of RDF, but without computational guarantees.

> ***OWL-DL***, which is based on Description Logics and supports maximum expressiveness without loosing computational completeness and decidability.

> ***OWL-Lite***, a subset of OWL-DL, which supports classification hierarchy and simple constraint features.

In October 2007, a new W3C working group was started to extend OWL with several new features. W3C announced the new version of OWL on 27 October 2009. This new version, called ***OWL 2 Web Ontology Language***[5], adds new functionality with respect to OWL 1. Some of the new features are syntactic sugar (e.g., *disjoint union* of classes) while others offer new expressivity, including keys, property chains, richer datatypes, data ranges, qualified cardinality restrictions, asymmetric, reflexive and disjoint properties and enhanced annotation capabilities.

Semantic Web Rule Language (SWRL)

The Semantic Web Rule Language (SWRL)[6] is an expressive OWL-based rule language. SWRL allows users to write rules that can be expressed in terms of OWL concepts to provide more powerful deductive reasoning capabilities than OWL alone. Semantically, SWRL is built on the same description logic foundation as OWL does and provides similar strong formal guarantees when performing inference.

SPARQL Query Language

"Query" in the Semantic Web context means technologies and protocols that can programmatically retrieve information from the Web of Data.

RDF provides the foundation for publishing and linking your data. However, just as relational databases or XML need specific query languages (SQL and XQuery respectively), the Web of Data, typically represented using RDF as a data format, needs its own, RDF-specific query language and facilities. This is provided by the SPARQL query language[7].

---

[5]http://www.w3.org/TR/owl2-overview/
[6]http://www.w3.org/Submission/SWRL/
[7]http://www.w3.org/TR/rdf-sparql-query/

Technically, SPARQL queries are based on (triple) patterns. RDF can be seen as a set of relationships among resources (i.e., RDF triples); SPARQL queries provide one or more patterns against such relationships. These triple patterns are similar to RDF triples, except that one or more of the constituent resource references are variables. A SPARQL engine would returns the resources for all triples that match these patterns.

Using SPARQL, consumers of the Web of Data can extract possibly complex information (i.e., existing resource references and their relationships) which are returned, for example, in a table format. This table can be incorporated into another Web page; using this approach SPARQL provides a powerful tool to build, for example, complex mash-up sites or search engines that include data stemming from the Semantic Web.

2.1.3 Semantic Web Tools

Protégé

Protégé[8] is a free, open-source platform that provides a growing user community with a suite of tools to construct domain models and knowledge-based applications with ontologies. At its core, Protégé implements a rich set of knowledge-modeling structures and actions that support the creation, visualization and manipulation of ontologies in various representation formats. Protégé can be customized to provide domain-friendly support for creating knowledge models and entering data. Furthermore, Protégé can be extended by way of a plug-in architecture and a Java-based Application Programming Interface (API) for building knowledge-based tools and applications.
The Protégé platform supports two main ways of modeling ontologies, the Protégé-Frames editor and the Protégé-OWL editor.

The ***Protégé-OWL editor*** is an extension of Protégé that supports the Web Ontology Language (OWL). It enables users to:

Load and save OWL and RDF ontologies.

Edit and visualize classes, properties, and SWRL rules.

Define logical class characteristics as OWL expressions.

Execute reasoners such as description logic classifiers.

Edit OWL individuals for Semantic Web markup.

The ***Protégé-Frames editor*** enables users to build and populate ontologies that are

---

[8]http://protege.stanford.edu/

frame-based, in accordance with the *Open Knowledge Base Connectivity protocol (OKBC)*. In this model, an ontology consists of a set of classes organized in a subsumption hierarchy to represent a domain's salient concepts, a set of slots associated with classes to describe their properties and relationships, and a set of instances of those classes - individual exemplars of the concepts that hold specific values for their properties.

Jena

Jena[9] is a Java framework for building Semantic Web applications. It provides a extensive Java libraries for helping developers develop code that handles RDF, RDF-S, OWL and SPARQL in line with published W3C recommendations. Jena includes a rule-based inference engine to perform reasoning based on OWL and RDF-S ontologies, and a variety of storage strategies to store RDF triples in memory or on disk.

2.2   Reasoning

Reasoning is the process of making implicit logical inferences from the explicit set of facts or statements, which constitute the core of any knowledge base.

2.2.1   Pellet

Pellet [3] is a complete and capable OWL-DL reasoner with very good performance and a number of unique features. It is written in Java and is open source under a very liberal license. It is used in a number of projects, from pure research to industrial settings. Pellet is the first implementation of the full decision procedure for OWL-DL (including instances) and has extensive support for reasoning with individuals (including conjunctive query over assertions), user-defined datatypes and debugging ontologies. It implements several extensions to OWL-DL including a combination formalism for OWL-DL ontologies, a non-monotonic operator and preliminary support for OWL/Rule hybrid reasoning. It has proven to be a reliable tool for working with OWL-DL ontologies and experimenting with OWL extensions.

Pellet provides all the standard inference services that are traditionally provided by DL reasoners:

> *Consistency checking*: ensures that an ontology does not contain any contradictory facts. The OWL 2 Direct Semantics provides the formal definition of ontology consistency used by Pellet.

---

[9]http://jena.apache.org/

Concept satisfiability: determines whether it is possible for a class to have any instances. If a class is unsatisfiable, then defining an instance of that class makes the entire ontology inconsistent.

Classification: computes the subclass relations between every named class to create the complete class hierarchy. The class hierarchy can be used to answer queries such as getting all or only the direct subclasses of a class.

Realization: finds the most specific classes that an individual belongs to. Realization can only be performed after classification since direct types are defined with respect to a class hierarchy. Using the classification hierarchy, it is also possible to get all the types for each individual.

Pellet relies on an implementation of a direct tableau algorithm for a DL-safe rules extension to OWL-DL. This implementation allows one to load and reason with DL-safe rules encoded in SWRL and includes support for some SWRL built-ins.

2.2.2 Pellet-Spatial

Pellet-Spatial [6] extends Pellet with qualitative spatial reasoning capabilities. It supports consistency checking of spatial relations expressed using Region Connection Calculus (RCC) and computes new spatial inferences from asserted relations. The spatial relations are expressed in RDF/OWL and can be combined with arbitrary domain ontologies. Pellet-Spatial can also answer SPARQL queries that mix spatial relations with arbitrary RDF/OWL relations. Pellet-Spatial implements two RCC reasoners: (a) A reasoner based on the semantics preserving translation of RCC relations to OWL-DL class axioms and (b) a reasoner based on the RCC composition table that implements a path-consistency algorithm.

Results show that, without further optimizations, the first reasoner based on the translation of RCC relations to OWL-DL class axioms lacks practicability even for small datasets. In addition to translation RCC relations to OWL class axioms, one axiom is defined for each region to satisfy the regularity condition of region (to be a non-empty concept and to contain all of the regions interior points). This axiom significantly affects non-determinism as well as the number of qualified existential quantifiers in the ontology. Qualified existential and universal quantifiers, is one of the source of complexity (AND-branching) in DL reasoning. The second spatial reasoner based on a path-consistency algorithm and the RCC-8 composition table has been shown to be more promising with regards to performance.

Following the example of Pellet-Spatial, CHRONOS implements the same path-consistency algorithm (Algorithm 1), incorporating certain optimizations for temporal representations expressed using the Temporal Relation Model of Allen (Table 2.1).

2.2.3 CHOROS

CHOROS [7] is a reasoner which supports consistency checking and query answering over spatial information in OWL expressed using the Region Connection Calculus (RCC) or the Cone-Shaped Directional logic formalism (CSD). Choosing either representation is a design decision that depends mainly on the application. However, both RCC and CSD expressions in OWL may co-exist within the same ontology together with standard OWL semantic relations. In that respect, CHOROS extends Pellet-Spatial to support CSD-9 relations in addition to RCC-8 relations. As such, it can answer mixed SPARQL queries over all spatial and non spatial relation types.

2.2.4 SOWL

SOWL [8] is an ontology for representing and reasoning over spatiotemporal information in OWL. Building upon well established standards of the semantic web (OWL 2.0, SWRL) SOWL enables representation of static as well as of dynamic information based on the 4D-fluents [2] or, equivalently, on the N-ary [1] approach.

Representing qualitative temporal information (i.e., information whose temporal extents are unknown) in addition to quantitative information (i.e., where temporal information is defined precisely) is a distinctive feature of SOWL. Temporal representation is based on the 4D-fluents approach enhanced with Allen relations which are defined as object properties between intervals. The N-ary version of the SOWL ontology introduces one additional object for representing a temporal property. This object is an individual of class Event. In SOWL, the temporal property remains a property relating the additional object with both the objects (e.g., an Employee and a Company) involved in a temporal relation.

Temporal reasoning in SOWL is realized by introducing a set SWRL rules for asserting inferred temporal relations of Allen. Reasoners that support DL-safe rules (i.e., rules that apply only on named individuals in the knowledge base) such as Pellet [3] can be used for inference and consistency checking over temporal relations. Alternatively, OWL axioms on temporal properties can be used instead of SWRL. However, this approach cannot guarantee decidability and is therefore not compatible with W3C specifications. Reasoning is applied either on temporal intervals directly, or by applying point-based reasoning operating on representations of intervals involving their starting and ending points.

2.3 Representation of Temporal Knowledge

The OWL-Time temporal ontology[10] describes the temporal content of Web pages and the temporal properties of Web services. Apart from language constructs for the representation of time in ontologies, there is still a need for mechanisms for the representation of the evolution of concepts (e.g., events) in time. Temporal relations are in fact ternary (i.e., properties of objects that change in time involve also a temporal value in addition to the object and the subject) and cannot be expressed directly in OWL. Representing temporal relations in OWL calls for specific methods such as N-ary relations [1] or 4D-fluents [2].

2.3.1 The N-ary Relations Approach

The N-ary relations approach [1] suggests representing an n-ary relation as two properties each related with a new object, which in turn is related with the temporal interval that this relation holds. This approach requires only one additional object for every temporal relation and maintains property semantics. A temporal property between two individuals (e.g., Employee works for Company) holds as long as that event endures. The N-ary property is represented as a class rather than as a property. Instances of such classes correspond to instances of the relation. Additional properties introduce additional binary links to each argument of the relation. For properties that change in time, their domains and ranges have to be adjusted taking into account the classes of intermediate objects representing the relation. For example, the *"worksFor"* relation in Figure 2.1 is no longer a relation having as object an individual of class "Company" and subject of class "Employee" as they are now related to the new object *"EmploymentEvent"*. The new domain is the union of the old domain with the class that represents the N-ary property (Event class). Likewise, the new range is a union of the old one with the Event class.
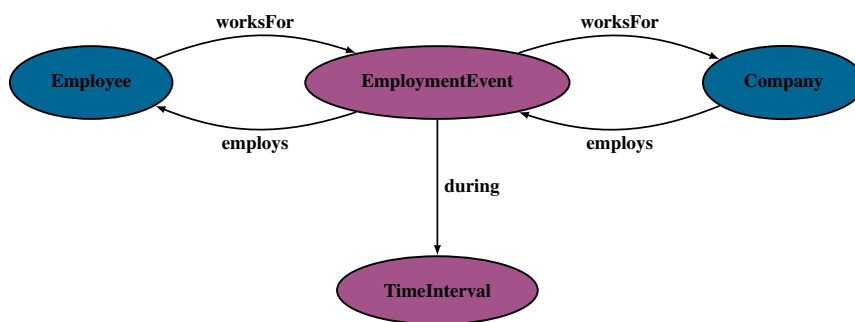

Figure 2.1: Representation of the temporal property *"Employee WorksFor Company During TimeInterval"* using the N-ary approach.

In 4D-fluents approach, for each object participating in a temporal property a new

---

[10]http://www.w3.org/TR/owl-time/

intermediate object is asserted, in contrast to a single object for the entire relation of the N-ary approach. Overall, the N-ary approach requires fewer objects and is the preferred approach in our work.

2.3.2 Temporal Relations of Allen

Following the N-ary relations approach, to add the time dimension to an ontology, classes *Event* and *TimeInterval* are introduced. Class *Event* represents temporal relations (which can be specialized subclasses of *Event* such as *EmploymentEvent* and class *TimeInterval* is the domain class of time intervals. Properties having a temporal dimension are called *fluent properties* and connect initial objects with instances of class *Event* (as in Figure 2.1).

In [8] the temporal representation was enhanced with qualitative temporal relations (i.e., relations holding between time intervals whose starting and ending points are not specified) by introducing temporal relationships as object relations between time intervals. A temporal relation can be one of the thirteen pairwise disjoint relations of Allen [9] of Figure 2.2.
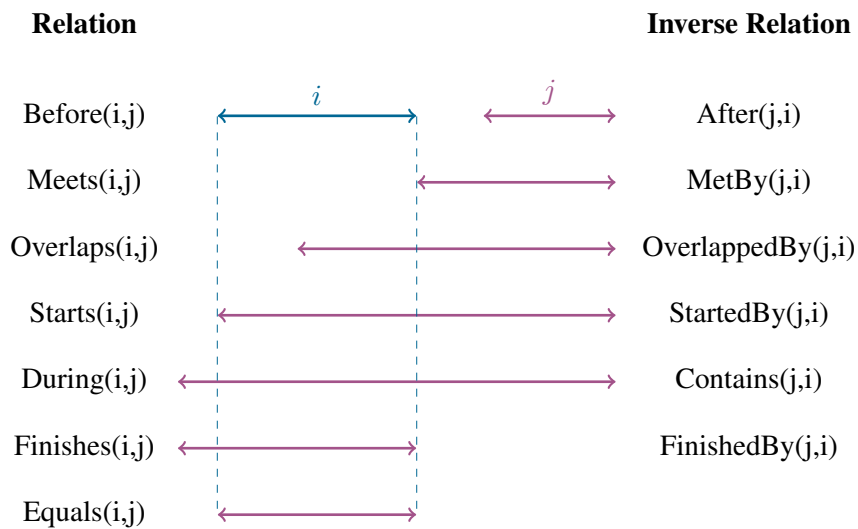
Relation   **Inverse Relation**

Before(i,j)	After(j,i)
Meets(i,j)	MetBy(j,i)
Overlaps(i,j)	OverlappedBy(j,i)
Starts(i,j)	StartedBy(j,i)
During(i,j)	Contains(j,i)
Finishes(i,j)	FinishedBy(j,i)
Equals(i,j)	

Figure 2.2: Temporal Relations of Allen

Relations between intervals are expressed as relations between their starting and ending points which in turn are expressed as a function of the three possible relations between points (time instants) namely *equals*, *before* and *after* denoted by "=", "<" and ">" respectively, forming the so called *"point algebra"* [5].

Let $i_1 = [s_1, e_1]$ and $i_2 = [s_2, e_2]$ be two intervals with starting and ending points s_1, s_2 and e_1, e_2 respectively; then, the thirteen relations of Allen of Figure 2.2 are rewritten as follows in Table 2.1:

Relation		Inverse Relation
i_1 **Before** i_2	$e_1 < s_2$	**After**
i_1 **Meets** i_2	$e_1 = s_2$	**MetBy**
i_1 **Overlaps** i_2	$(s_1 < s_2) \cap (e_1 < e_2) \cap (e_1 > s_2)$	**OverlappedBy**
i_1 **Starts** i_2	$(s_1 = s_2) \cap (e_1 < e_2)$	**StartedBy**
i_1 **During** i_2	$(s_1 > s_2) \cap (e_1 < e_2)$	**Contains**
i_1 **Finishes** i_2	$(s_1 > s_2) \cap (e_1 = e_2)$	**FinishedBy**
i_1 **Equals** i_2	$(s_1 = s_2) \cap (e_1 = e_2)$	

Table 2.1: Temporal Relations of Allen

The relations *after, metby, overlappedby, startedby, contains* and *finishedby* are the inverse of *before, meets, overlaps, starts, during* and *finishes* and are defined accordingly (by interchanging s_1, s_2 and e_1, e_2 in their respective definitions). Basic properties are that the relation *equals* is symmetric and relations *before* and *after* are transitive. These temporal relations and the corresponding reasoning mechanism are integrated both in the 4D-fluents and the N-ary based ontologies.

Qualitative relations can be asserted, even when the specific time points or the temporal extends of intervals are unknown but their relative position is known; thus, the expressiveness of the representation increases. The qualitative approach to temporal information is popular in Artificial Intelligence mainly because precise numerical information is often unavailable or not necessary in many real world applications. Whenever a temporal relation between two events is uncertain, disjunctions of some of the thirteen basic relations can be used to represent what is known.

Chapter 3

CHRONOS

CHRONOS is a system for reasoning over temporal information in OWL ontologies. Following the example of Pellet-Spatial [6] and CHOROS [7], CHRONOS is a dedicated temporal reasoning engine implemented in Java which allows for incorporating certain optimizations in implementing path consistency. The proposed solution extends the OWL-DL reasoning features of Pellet [3] reasoner with qualitative temporal reasoning capabilities. It supports checking of consistency and computation of new temporal inferences resulting from asserted relations. Furthermore, answering of SPARQL queries over a set of temporal and nontemporal relations is also supported. The last ones are expressed in RDF/OWL and can be applied on arbitrary domain ontologies

3.1 Temporal Relations in CHRONOS

CHRONOS defines an RDF/OWL vocabulary for expressing qualitative temporal relations between time intervals, using the Temporal Relation Model of Allen.
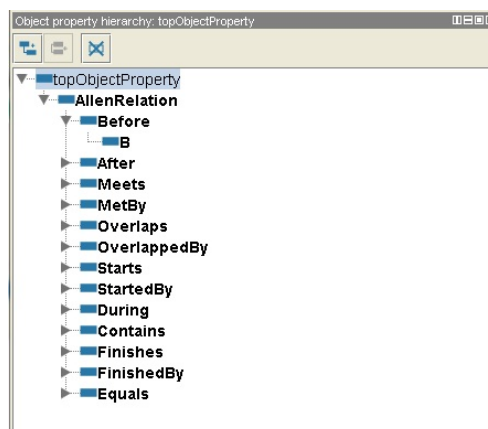

Figure 3.1: Temporal Relations of Allen, as object properties, in Protégé.

Specifically, an interval is represented as an OWL individual (e.g., *Individual : interval*$_1$ *oftype Class : Interval*) and a temporal relation between two intervals is represented as an OWL object property assertion (e.g., *Individual : interval*$_1$ *ObjectProperty : Before Individual : interval*$_2$).

As illustrated in Figure 3.1, the temporal relations of Allen are defined as simple object properties with no extra characteristics (e.g., *inverse, transitive*). One can either use the vocabulary provided, or use his own by defining sub-property axioms (e.g., *ObjectProperty : B subPropertyOf : Before*).

Non-temporal relations are represented as ordinary OWL assertions. This approach supports reasoning and querying for both temporal relations and standard RDF semantic relations by setting apart each problem. Thus, CHRONOS strictly separates temporal reasoning from semantic DL reasoning by using an exclusive temporal reasoner component.
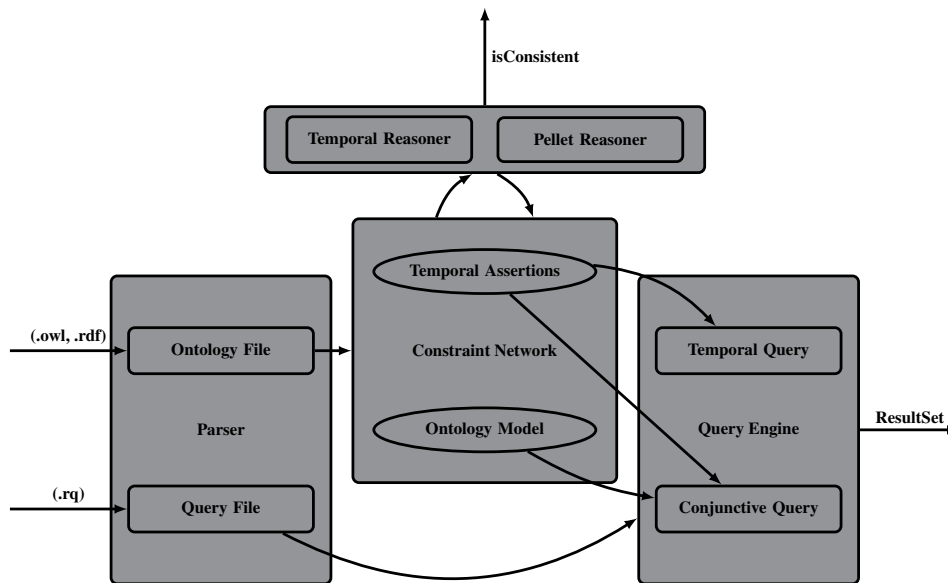
3.2 Architecture of CHRONOS

Figure 3.2: Architecture of CHRONOS

The architecture of CHRONOS is illustrated in Figure 3.2. It consists of several modules, the most important of them being the following:

Parser, which is used for loading and decoding of ontologies and queries.

Constraint Network, which is used for storing of temporal property assertions and non-temporal standard OWL assertions.

Reasoner, which consists of the Temporal Reasoner and Pellet, and is used for temporal and semantic DL reasoning.

Query Engine, which is used for answering of temporal and conjunctive temporal and non-temporal semantic queries.

3.2.1 Parser

Parser consists of two separate components, the Ontology Parser and the Query Parser, which implement an RDF[1] and an ARQ[2] parser , for parsing of ontologies and queries respectively:

Ontology Parser: Jena API is used in ontology parsing, together with internal temporal vocabularies, in order to extract data from the RDF graph. These vocabularies have been created, as model properties of Jena, to define the thirteen basic relations of Allen. In this way, temporal triples can be tracked and removed from the graph, in order to store them into the constraint network as *temporal assertions*. The rest of the graph, which contains non-temporal standard OWL assertions, is stored into the Knowledge Base of Pellet, forming the *ontology model*.

Query Parser: Jena API is also used in query parsing, together with internal query formats, in order to divide query atoms into temporal and non-temporal, as well as to map triple predicates to temporal relations. These techniques enable CHRONOS to process queries containing exclusively temporal information, in addition to conjunctive temporal and non-temporal semantic queries.

As already mentioned, Ontology Parser implements an RDF parser, therefore input files must be either in RDF (*.rdf*), or in OWL (*.owl*) format. Similarly, the ARQ parser of CHRONOS, which is the Query Parser, accepts either SPARQL (*.rq*) or exclusively temporal (*.tmp*) files as inputs. An example of an ontology file, in rdf format, as well as an example of a query in SPARQL, are illustrated in Appendix C.

3.2.2 Constraint Network

CHRONOS parses the ontology and temporal relations between intervals are detected and extracted, forming a Constraint Network.

A *Constraint Network (CN)* is a set of variables together with a set of constraints defined on these variables. The constraint network of CHRONOS comprises of the temporal triples in the ontology graph (extracted by the Ontology Parser), as well as, of non-temporal OWL assertions, which are stored into the

---

[1]http://www.w3.org/RDF
[2]http://jena.apache.org/documentation/query

Knowledge Base of Pellet, as it is typical in standard (i.e., non-temporal) reasoning.

A *Knowledge Base (KB)* is a special kind of database for knowledge management, providing the means for the computerized collection, organization, and retrieval of knowledge. The Knowledge Base of Pellet is a combination of an assertional box (i.e., component that contains assertions about individuals) and a terminological box (i.e., component that contains axioms about classes) which provides consistency checking and query services.

CHRONOS allows user to search for relations (by specifying the temporal interval of interest), count the number of temporal and non-temporal relations, check the temporal network for consistency using the composition table of the temporal reasoner, and also check the consistency of the non-temporal knowledge base by using Pellet [3].

3.2.3 Reasoning in CHRONOS

CHRONOS separates temporal from semantic DL reasoning, which is managed by Pellet, and uses an exclusive reasoner for temporal calculus. Temporal relations, expressed by a set of jointly exclusive and pairwise disjoint basic relations which is closed under several operations, are handled separately by their respective network. Temporal reasoning (i.e., inferring implied relations and detecting inconsistencies) can be viewed as a form of a constraint satisfactions problem which is NP in the general case. CHRONOS handles tractable cases of such problems by limiting asserted temporal relations to the basic Allen relations of Figure 2.2. Temporal reasoning is then achieved by applying the path consistency algorithm [Algorithm 1]. Path consistency computes all inferred relations using compositions of existing relations defined, until a fixed point is reached (i.e., algorithm does not yield new inferences) or until an inconsistency is detected (i.e., yield the empty relation $\emptyset$ as a result). Path consistency, when applied on a set of assertions containing only basic relations, retains tractability and guarantees soundness and completeness of reasoning [10].

Composition Table of Basic Allen Temporal Relations

The possible compositions of basic Allen temporal relations are defined in the composition table Table 3.1 [9]. The composition table represents the result of the composition of two temporal relations. For example, if relation R_1 holds between $interval_x$ and $interval_y$ and relation R_2 holds between $interval_y$ and $interval_z$, then the entry of the composition table corresponding to row R_1 and column R_2 denotes the possible relation(s) holding between $interval_x$ and $interval_z$:

$$R_1(x, y) \circ R_2(y, z) \rightarrow R_3(x, z)$$

	B	A	M	Mi	O	Oi	S	Si	D	Di	F	Fi	Eq
B	B	B, A, M, Mi, O, Oi, S, Si, D, Di, F, Fi, Eq	B	B, M, O, S, D	B	B, M, O, S, D	B	B	B, M, O, S, D	B	B, M, O, S, D	B	B
A	B, A, M, Mi, O, Oi, S, Si, D, Di, F, Fi, Eq	A	A, Mi, Oi, D, F	A	A, Mi, Oi, D, F	A	A, Mi, Oi, D, F	A	A, Mi, Oi, D, F	A	A	A	A
M	B	A, Mi, Oi, Si, Di	B	F, Fi, Eq	B	O, S, D	M	M	O, S, D	B	O, S, D	B	M
Mi	B, M, O, Di, Fi	A	S, Si, Eq	A	Oi, D, F	A	Oi, D, F	A	Oi, D, F	A	Mi	Mi	Mi
O	B	A, Mi, Oi, Si, Di	B	Oi, Si, Di	B, M, O	O, Oi, S, Si, D, Di, F, Fi, Eq	O	O, Di, Fi	O, S, D	B, M, O, Di, Fi	O, S, D	B, M, O	O
Oi	B, M, O, Di, Fi	A	O, Di, Fi	A	O, Oi, S, Si, D, Di, F, Fi, Eq	A, Mi, Oi	Oi, D, F	A, Mi, Oi	Oi, D, F	A, Mi, Oi, Si, Di	Oi	Oi, Si, Di	Oi
S	B	A	B	Mi	B, M, O	Oi, D, F	S	S, Si, Eq	D	B, M, O, Di, Fi	D	B, M, O	S
Si	B, M, O, Di, Fi	A	O, Di, Fi	Mi	O, Di, Fi	Oi	S, Si, Eq	Si	Oi, D, F	Di	Oi	Di	Si
D	B	A	B	A	B, M, O, S, D	A, Mi, Oi, D, F	D	A, Mi, Oi, D, F	D	B, A, M, Mi, O, Oi, S, Si, D, Di, F, Fi, Eq	D	B, M, O, S, D	D
Di	B, M, O, Di, Fi	A, Mi, Oi, Si, Di	O, Di, Fi	Oi, Di, Si	O, Di, Fi	Oi, Di, Si	O, Di, Fi	Di	O, Oi, S, Si, D, Di, F, Fi, Eq	Di	Oi, Di, Si	Di	Di
F	B	A	M	A	O, S, D	A, Mi, Oi	D	A, Mi, Oi	D	A, Mi, Oi, Si, Di	F	F, Fi, Eq	F
Fi	B	A, Mi, Oi, Si, Di	M	Oi, Si, Di	O	Oi, Si, Di	O	Di	O, S, D	C	F, Fi, Eq	Fi	Fi
Eq	B	A	M	Mi	O	Oi	S	Si	D	Di	F	Fi	Eq

Table 3.1: Composition Table of Basic Temporal Relations of Allen

Relations *before, after, meets, metby, overlaps, overlappedby, starts, startedby, during, contains, finishes, finishedby* and *equals* are represented using symbols *B, A, M, Mi, O, Oi, S, Si, D, Di, F, Fi* and *Eq* respectively. As we can observe, compositions with *equals* keep initial relations unchanged. The following is an example of such a temporal temporal inference rule:

$$Before(x, y) \circ Equals(y, z) \rightarrow Before(x, z)$$

Path-Consistency Algorithm

A series of compositions of relations may imply relations which are inconsistent with existing ones (for example the rule referred to above will yield a contradiction if *after(x,z)* has been asserted into the ontology for specific values of *x,y,z*). In CHRONOS, path consistency is implemented by applying the path consistency algorithm [5], which computes all inferred relations using compositions of existing relations defined, until a fixed point is reached (i.e., application of the algorithm does not yield new inferences) or until an inconsistency is detected (i.e., yield the empty relation $\emptyset$ as a result).

Algorithm 1 Path-Consistency Algorithm

```
 1: procedure PATHCONSISTENCY(N)
 2:     if N = ∅ then
 3:         return true
 4:     end if
 5:     complete(N)
 6:     Q ← {R_ij\R_ij ∈ N}
 7:     while Q ≠ ∅ do
 8:         R_ab ← remove(Q)
 9:         if !isConsistent(N, Q, R_ab) then
10:             return false
11:         end if
12:     end while
13:     return true
14: end procedure


15: procedure ISCONSISTENT(N, Q, R_ab)
16:     for S_bc ∈ N do
17:         T_bc ← R_ab ∘ S_bc
18:         add(N, Q, T_ac)
19:         if !isConsistent then
20:             return false
21:         end if
22:     end for
23:     return true
24: end procedure
```

```
25: procedure ADD(N, Q, T_ac)
26:     if T = ⊤ then
27:         return
28:     end if
29:     U_ac ← {R_ij\i = a, j = c, R_ij ∈ N}
30:     if ∄U_ac then
31:         V_ac ← T_ac
32:     else
33:         V_ac ← T_ac ∩ U_ac
34:         if V = ∅ then
35:             isConsistent = false
36:             return
37:         end if
38:         if U = V then
39:             return
40:         end if
41:         N ← N\{U_ac}
42:     end if
43:     N ← N ∪ {V_ac}
44:     Q ← Q ∪ {V_ac}
45:     add(N, Q, {V_ĉa})
46: end procedure
```

Given a constraint network *N* (i.e., a set of defined Allen temporal relations), *N* is consistent if it is either empty, or if every relation in the network is consistent. The requirement for the relations in *N* to be defined is relevant to the tractability of a sound and complete path-consistency procedure. As it is noticed in [5], path consistency is tractable for temporal relations of Allen.

A queue (Q) is used as a structure to keep track of relations that have to be processed. The algorithm runs until $Q = \emptyset$, or an inconsistency is detected. Q is initialized with all the defined relations $R_{ij} \in N$.

Complete step (line 5) processes N to infer all the inverse and equals relations:

> ***Inverse complete***: For every defined relation $R_{ij} \in N$, we ensure that $R_{\widehat{ji}} \in N$, for example, for $meets(x, y) \in N$ we ensure that $metby(y, x) \in N$.

> ***Equals complete***: For every region $x \in N$, we ensure that relation $equals(x, x) \in N$.

A relation R_{ab} (line 15) is path-consistent if the rule for combining a compositional inference with existing information,

$$V_{ac} \leftarrow U_{ac} \cap R_{ab} \circ S_{bc}$$

results in a non-empty set ($V \neq \emptyset$) for regions a and c, where $S_{bc} \in N$ relations with a transitive path with R_{ab} from a, through b, to c and U_{ac} a relation possibly $\in N$ as existing information.

The compositional inference $T_{ac} \leftarrow R_{ab} \circ S_{bc}$ (line 17), is computed for regions a and c as the union set T for the composition of each pair (r,s) in the set $R \times S$, $r \in R$, $s \in S$. The composition of a pair (r,s) consists in a lookup for the composition table of Allen given that r and s are elements of the set of thirteen defined relations.
If $U_{ac} \in N$ (i.e., there is information for the pair (a,c)), we complete the rule by computing the intersection $V_{ac} \leftarrow T_{ac} \cap U_{ac}$ (line 33), where V is the intersection set of relations $v \in T \cap U$. This step refines the already existing relation $U_{ac} \in N$ and is essential for the path-consistency algorithm as it defines the inconsistent state: *if ($V = \emptyset$) then network is inconsistent*.

The state, *if (U = V) then return* (line 38), defines that the step $V_{ac} \leftarrow T_{ac} \cap U_{ac}$ (line 33), could not refine relation U_{ac}. Hence, combining compositional inference T_{ac} with existing information U_{ac} does not add new information. In this case, we can return. Otherwise, U_{ac} is removed from N, the refined V_{ac} is added to N and Q, and the inverse $V_{\widehat{ca}}$ is processed.

In conclusion, path consistency is implemented by consecutive application of the formula:

$$R_s(x, y) \leftarrow R_i(x, y) \cap (R_j(x, k) \circ R_k(k, y)), \forall x, y, k$$

representing intersection of compositions of relations with existing relations. This formula is applied until there are no new inferences, or until the empty set is reached, implying that the network is inconsistent.

Consistent Example

Consider the following intervals: $i_1=[s_1, e_1]$, $i_2=[s_2, e_2]$, $i_3=[s_3, e_3]$ and $i_4=[s_4, e_4]$.
The first interval is *before* the second and *overlaps* the fourth.
The second interval *meets* the third.
The fourth interval is *finishedby* the third.
What can be inferred about the first and the third interval?

- i_1 **Before** i_2: $(e_1 < s_2)$

- i_2 **Meets** i_3: $(s_1 < s_4) \cap (e_1 < e_4) \cap (e_1 > s_4)$

- i_1 **Overlaps** i_4: $(e_2 = s_3)$

- i_4 **FinishedBy** i_3: $(s_3 > s_4) \cap (e_3 = e_4)$

Using the composition table of Allen (Table 3.2 and Table 3.3) and the path-consistency algorithm (Algorithm 1), we can refine the network in the following way:

$$(i_1 \textbf{ Before } i_2) \circ (i_2 \textbf{ Meets } i_3)$$

$$\rightarrow i_1 \textbf{ Before } i_3$$

	B	A	**M**	Mi	O	Oi	S	Si	D	Di	F	Fi	Eq
B			B										

Table 3.2: Composition of *before* and *meets*.

$$(i_1 \textbf{ Overlaps } i_4) \circ (i_4 \textbf{ FinishedBy } i_3)$$

$$\rightarrow i_1 \{\textbf{Before} \cup \textbf{Meets} \cup \textbf{Overlaps}\} i_3$$

	B	A	M	Mi	O	Oi	S	Si	D	Di	F	**Fi**	Eq
O												B, M, O	

Table 3.3: Composition of *overlaps* and *finishedby*.

$$(i_1 \textbf{ Before } i_3) \cap (i_1 \{\textbf{Before} \cup \textbf{Meets} \cup \textbf{Overlaps}\} i_3)$$

$$\rightarrow i_1 \textbf{ Before } i_3$$

Inconsistent Example

Consider the following intervals: $i_1=[s_1, e_1]$, $i_2=[s_2, e_2]$, $i_3=[s_3, e_3]$ and $i_4=[s_4, e_4]$.
The first interval *starts* the second and *finishes* the fourth.
The second interval *meets* the third.
The fourth interval is *metby* the third.
What can be inferred about the first and the third interval?

- i_1 **Starts** i_2: $(s_1 = s_2) \cap (e_1 < e_2)$

- i_2 **Meets** i_3: $e_2 = s_3$)

- i_1 **Finishes** i_4: $(s_1 > s_4) \cap (e_1 = e_4)$

- i_4 **MetBy** i_3: $(e_3 = s_4)$

Using the composition table of Allen (Table 3.4 and Table 3.5) and the path-consistency algorithm (Algorithm 1), we can refine the network in the following way:

$$(i_1 \ \textbf{Starts} \ i_2) \circ (i_2 \ \textbf{Meets} \ i_3)$$

$$\rightarrow i_1 \ \textbf{Before} \ i_3$$

	B	A	M	Mi	O	Oi	S	Si	D	Di	F	Fi	Eq
S			B										

Table 3.4: Composition of *starts* and *meets*.

$$(i_1 \ \textbf{Finishes} \ i_4) \circ (i_4 \ \textbf{MetBy} \ i_3)$$

$$\rightarrow i_1 \ \textbf{After} \ i_3$$

	B	A	M	Mi	O	Oi	S	Si	D	Di	F	Fi	Eq
F				A									

Table 3.5: Composition of *finishes* and *metby*.

$$(i_1 \ \textbf{Before} \ i_3) \cap (i_1 \ \textbf{After} \ i_3)$$

$$\rightarrow \emptyset, \quad \textbf{inconsistency}$$

Handling Disjunctions of Basic Relations

The composition of basic relations may infer disjunctions of such relations because disjunctive entries exist in the composition table of Allen relations. For example, the composition of *overlaps* and *during* yields disjunction of three possible relations (*starts, overlaps* and *during*) as a result:

$$Overlaps(x, y) \circ During(y, z) \rightarrow \{Overlaps \cup Starts \cup During\}(x, z)$$

Disjunctions of relations are represented using new relations, whose compositions must also be defined and asserted into the knowledge base. Composing disjunctions of relations can be achieved in two different ways:

  a) By pre-computing the composition of disjunctions and storing the result in a structure, which contains all possible compositions between basic and disjunctive relations.

  b) By decomposing disjunctive relations into basic ones and computing each composition "on the fly".

The first approach has been adopted in CHOROS [7] spatial reasoner, an earlier work of our lab, which is feasible for the set of 8 basic RCC-8 topological or for the set of 9 Directional relations. For topological relations the composition table has $2^8 \times 2^8$ entries (i.e., up to 2^8 disjunctions can appear). Similarly, for directional relations, the composition table has $2^8 \times 2^8$ entries (i.e., spatial relation "identical to" is replaced by the owl axiom "sameAs").

However, this approach is not preferred in CHRONOS, because with the set of 13 basic Allen relations, the table must have $2^{13} \times 2^{13}$ entries (i.e., up to 2^{13} disjunctions can appear) which requires more memory to store and more time to compute.

CHRONOS$_{x13}$: The first implementation of CHRONOS follows the later approach. Computing compositions on the fly results in less than $2^{13} \times 2^{13}$ combinations of disjunctions of basic relations. Disjunctive relations are divided into basic ones and each composition involves a simple look-up operation in the 13×13 composition table of Allen (Table 3.1):

$$(relation_1 \cup relation_2) \circ (relation_3 \cup relation_4)$$

$$\rightarrow (relation_1 \circ relation_3) \cup (relation_1 \circ relation_4) \cup (relation_2 \circ relation_3) \cup$$

$$(relation_2 \circ relation_4)$$

For example, the composition of {*overlaps, starts, during*} and *during* results as follows:

$$(Overlaps \cup Starts \cup During) \circ During$$

$$\rightarrow (Overlaps \circ During) \cup (Starts \circ During) \cup (During \circ During)$$

$$\rightarrow (Overlaps \cup Starts \cup During) \cup (During) \cup (During)$$

$$\rightarrow Overlaps \cup Starts \cup During \cup During \cup During$$

$$\rightarrow Overlaps \cup Starts \cup During$$

Advantages: Disjunctive relations are decomposed into basic ones and thus, the composition table contains only 13×13 entries (instead of $2^{13} \times 2^{13}$), which requires less memory to be stored and less time to be initialized.

Disadvantages: The division of disjunctive relations into simple ones, increases the number of required compositions.

CHRONOS$_{x28}$: The second implementation of CHRONOS is an optimization of CHRONOS$_{x13}$. As demonstrated in Appendix A and Appendix B, a reasoning mechanism over the Allen relations can be based on a tractable set of the following twenty-eight relations [8]:

{Before}, {After}, {Meets}, {MetBy}, {Overlaps}, {OverlappedBy}, {Starts}, {StartedBy}, {During}, {Contains}, {Finishes}, {FinishedBy}, {Equals}, {Finishes, FinishedBy, Equals}, {Overlaps, Starts, During}, {Starts, StartedBy, Equals}, {OverlappedBy, During, Finishes}, {OverlappedBy, StartedBy, Contains}, {Before, Meets, Overlaps}, {Overlaps, Contains, FinishedBy}, {After, MetBy, OverlappedBy}, {Before, Meets, Overlaps, Starts, During}, {After, MetBy, OverlappedBy, During, Finishes}, {After, MetBy, OverlappedBy, StartedBy, Contains}, {Before, Meets, Overlaps, Contains, FinishedBy}, {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}, {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}, {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}.

Based on this property, CHRONOS$_{x28}$ uses the total of 28×28 possible compositions, which have been pre-computed, as presented at Appendix B. In this way, the time required to calculate new compositions is eliminated, since this function has already been implemented. Furthermore, at odds with CHRONOS$_{x13}$,

CHRONOS$_{x28}$ does not contain any static structure for storing of compositions. Pre-computed compositions are used at run-time through functions, which simulate the storing process, in order to save the memory that would have been used by the structure.

Inverse of Disjunctions: The inverse of a disjunctive relation, is the disjunction of the inverse relations. For example, the inverse of the disjunctive relation {*before, meets*} is the disjunction of their inverse relations {*after, metby*} respectively. Using the tractable set of CHRONOS$_{x28}$, the possible twenty-eight inverse relations have been pre-computed, in order to be used at run-time, through functions which simulate the storing process, in the *"inverse complete"* step of the path consistency algorithm (Algorithm 1).

Table 3.6 presents a brief comparison between CHRONOS$_{x13}$ and CHRONOS$_{x28}$.

	CHRONOS$_{x13}$	**CHRONOS$_{x28}$**
Relational Model	Temporal relations of Allen, with 13 basic relations and 2^{13} possible disjunctions.	
Compositions of Disjunctions	Disjunctive relations are decomposed into basic ones and computation of compositions involves a simple look-up operation in the 13×13 composition table of Allen (Table 3.1). In practice, results in less than $2^{13} \times 2^{13}$ combinations of disjunctions of basic relations.	A tractable set of 28 basic and disjunctive relations is used. All possible compositions, totally 28×28, have been pre-computed in order to be used by the path consistency algorithm (Algorithm 1).
Storing Structure	A composition table with 13×13 entries, which contains the compositions of the basic Allen relations. Compositions of relations are not stored but are computed on the fly at run time (using simple look-up operations in the 13×13 table of basic relations).	CHRONOS$_{x28}$ does not contain any static structure for storing of compositions. Pre-computed compositions are used at run-time through functions, which simulate the storing process, in order to save the memory that would have been used by the structure.
Inverse of Disjunctions:	CHRONOS$_{x13}$ computes the inverse of a disjunctive relation at run-time, using the inverse of the basic relations which making up the composition.	Based on the tractable set of CHRONOS$_{x28}$, the possible twenty-eight inverse relations have been pre-computed, in order to be used at run-time, through functions which simulate the storing process, in the *"inverse complete"* step of the path consistency algorithm (Algorithm 1).

Table 3.6: CHRONOS$_{x13}$ vs CHRONOS$_{x28}$.

3.2.4 Query Engine

Both implementations of CHRONOS can answer conjunctive queries specifying temporal and non-temporal patterns (i.e., triple patterns for temporal relations joined with triple patterns for semantic RDF/OWL relations). More specifically, this module can process queries written in SPARQL satisfying the following conditions:

No variable is used in the predicate position,

Each property used in the predicate position is either an (object or datatype) property defined in the ontology or one of the following built-in properties: {*rdf : type, owl : sameIndividualAs, owl : differentFrom*} and,

At least one of the triples must contain a temporal object property in the predicate position (otherwise it is a non-temporal query).

The query engine of CHRONOS can process temporal queries specifying temporal properties in an RDF graph or Allen relations between temporal intervals, by implementing a dual stage query answering technique: The set of query answers returned by the first stage in response to a given a temporal query, are fed to the second stage whose purpose is to guarantee that the non-temporal part of the query is satisfied as well.

Chapter 4

Evaluation

In the following, the efficiency of our reasoning approach is assessed both, theoretically and experimentally. We carried-out several groups of experiments using synthetic (but realistic) data sets, demonstrating the run-time efficiency of our reasoning engine as a function of the size of the data set, in order to compare CHRONOS with other reasoning engines which also support reasoning over qualitative data, such as SOWL and CHOROS.

4.1   Theoretical Evaluation

Reasoning in CHRONOS calls for a path consistency algorithm (Algorithm 1) applying over a set of temporal relations, whose purpose is to infer implied relations, detect inconsistencies and ensure path consistency. If only the basic Allen relations are supported [10], [5] (as it is the case in our present work), reasoning has polynomial time complexity.

Since, within *n* intervals, any time interval can be related with every other interval with one basic Allen relation (basic Allen relations are mutually exclusive), at most $O(n^2)$ relations can be inferred. In average case, the number of temporal relations is in the order of *n* and path consistency has $O(n)$ time complexity for both implementations of CHRONOS.

In worst case, n^2 relations are asserted into the knowledge base for n individuals. Path consistency has $O(n^3)$ time complexity [11], for both implementations of CHRONOS, and is sound and complete for the supported sets of relations. This upper bound is pessimistic, since an inconsistency detection may terminate the reasoning process early, or the asserted relations may yield a small number of inferences.

In the most general case where disjunctive relations are supported in addition to the basic ones, any time interval can be related with every other interval by at most *k* relations, where *k* is the size of the set of supported relations (max$\{k\}$ = 13 for temporal relations of Allen). Therefore, for n intervals or instants, using $O(k^2)$ rules, at most $O(kn^2)$ relations can be asserted into the knowledge base.

4.2 Experimental Evaluation

The purpose of the following set of experiments is to evaluate the two implementations of CHRONOS reasoning engine (i.e., CHRONOS$_{x13}$ and CHRONOS$_{x28}$). We carried out two different experiments, corresponding to measurements of performance in the average and in the worst case, in order to demonstrate the run-time efficiency of reasoning of the two competing implementations (CHRONOS$_{x13}$ and CHRONOS$_{x28}$) as a function of the size of input data set. All experiments were run on a PC with 4 GB RAM running Windows 7 at 1.90 GHz.

4.2.1 Average Case Experiments

The average case performance is encountered when less than n^2 temporal relations are inferred from an input set of *n* temporal intervals. In the following experiments, these are in the order of *n*.

To study the run-time efficiency of reasoning in the average case, we used data sets comprising between 500 and 10,000 assertions. These assertions are used to populate 10 ontologies with random instances. Reasoning times are computed as averages over 10 iterations.

Figure 4.1 presents measurements of reasoning time, obtained by the two competing implementations, in average case.

Figure 4.2 presents measurements of the number of inferred relations of the two competing implementations, in average case.

Figure 4.3 presents measurements of the efficiency rate of the two implementations, in average case. This measure represents the time needed to infer each new relation:

$$Efficiency\ Rate = \frac{Reasoning\ Time\ (ms)}{Number\ of\ Inferred\ Relations}$$

Table 4.1 presents the entire set of measurements of CHRONOS$_{x13}$ and CHRONOS$_{x28}$, regarding their reasoning performance in the average case.
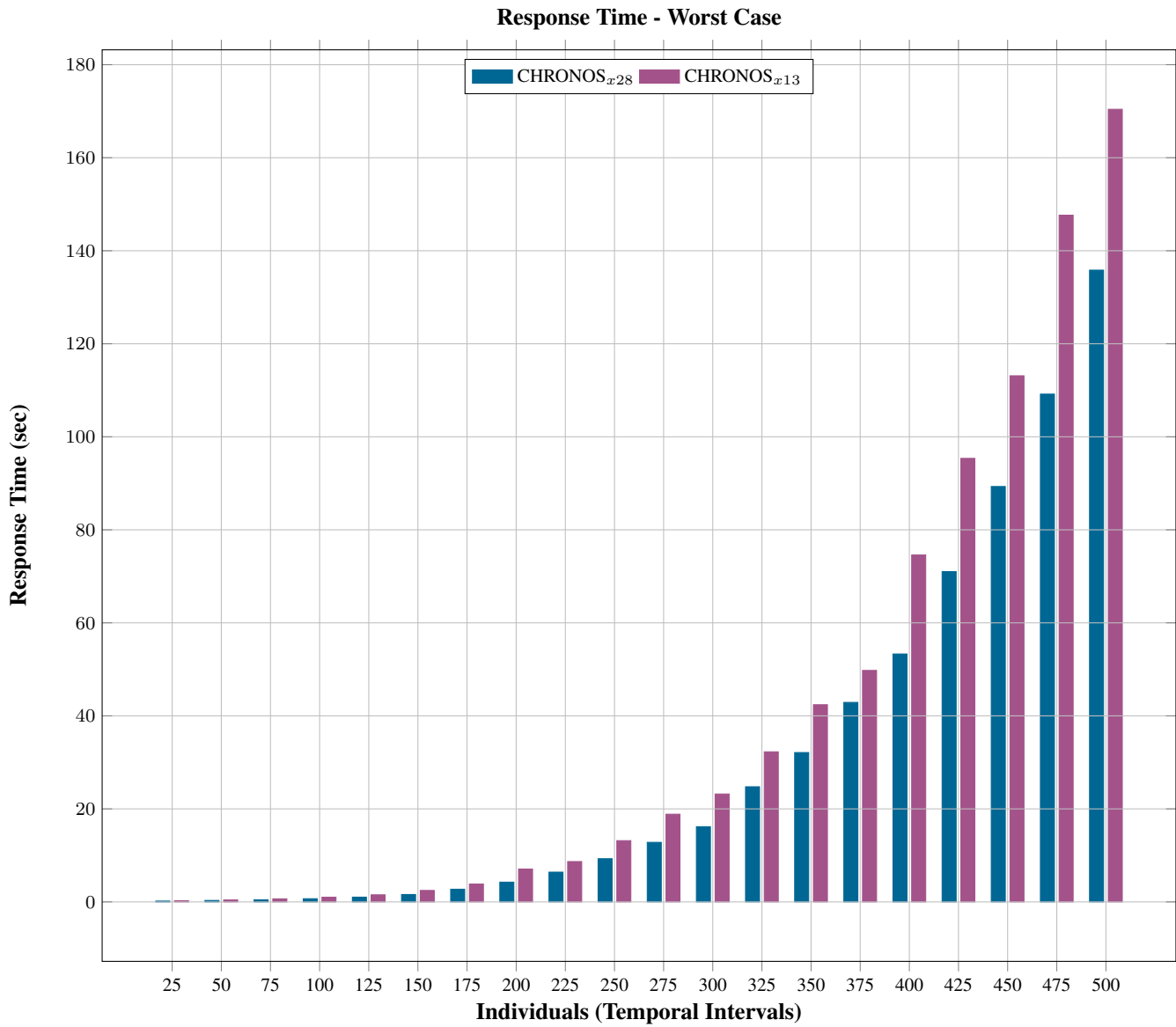
Response Time - Average Case (10 iterations)
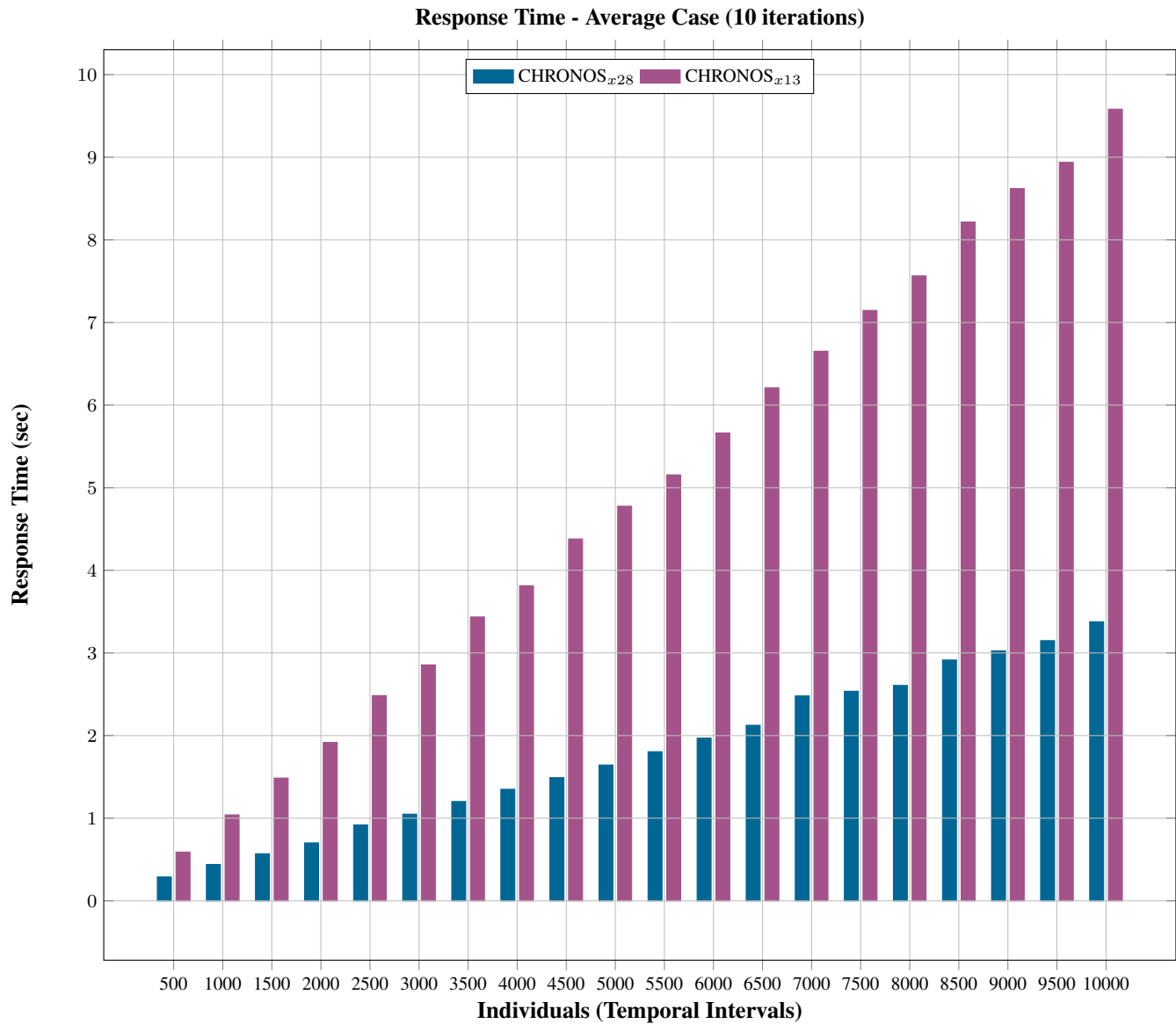
Figure 4.1: Average response time of temporal reasoning, over 10 iterations, as a function of the number of individuals (temporal intervals). **CHRONOS$_{x28}$ vs CHRONOS$_{x13}$**

Observations: Both implementations exhibit linear response time ($O(n)$ response time complexity), with CHRONOS$_{x28}$ performing almost three times faster than CHRONOS$_{x13}$.
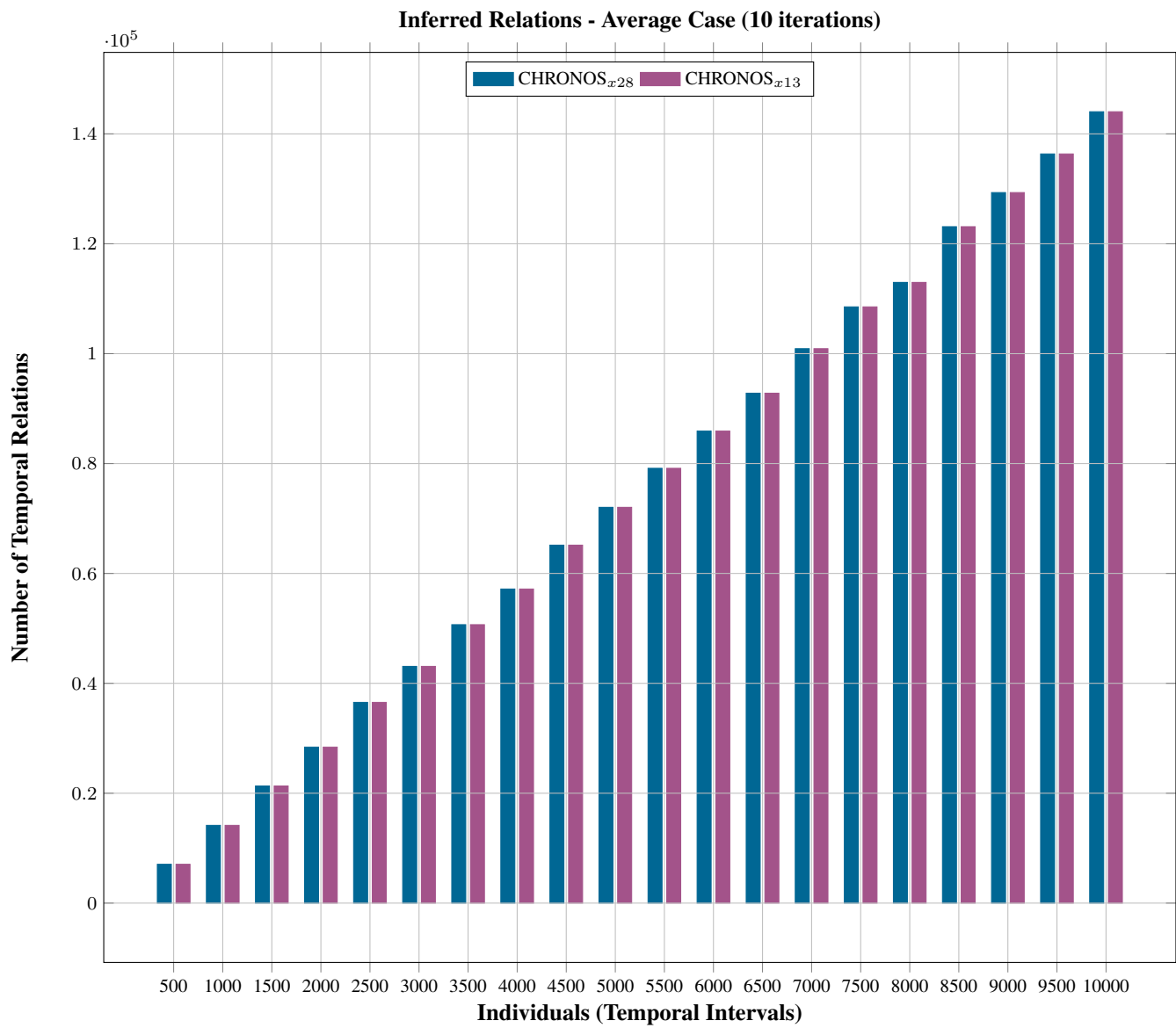
Figure 4.2: Average number of inferred relations, over 10 iterations, as a function of the number of individuals (temporal intervals). **CHRONOS**$_{x28}$ **vs CHRONOS**$_{x13}$

Observations: Both implementations infer the same number of temporal relations, which is in the order of n.
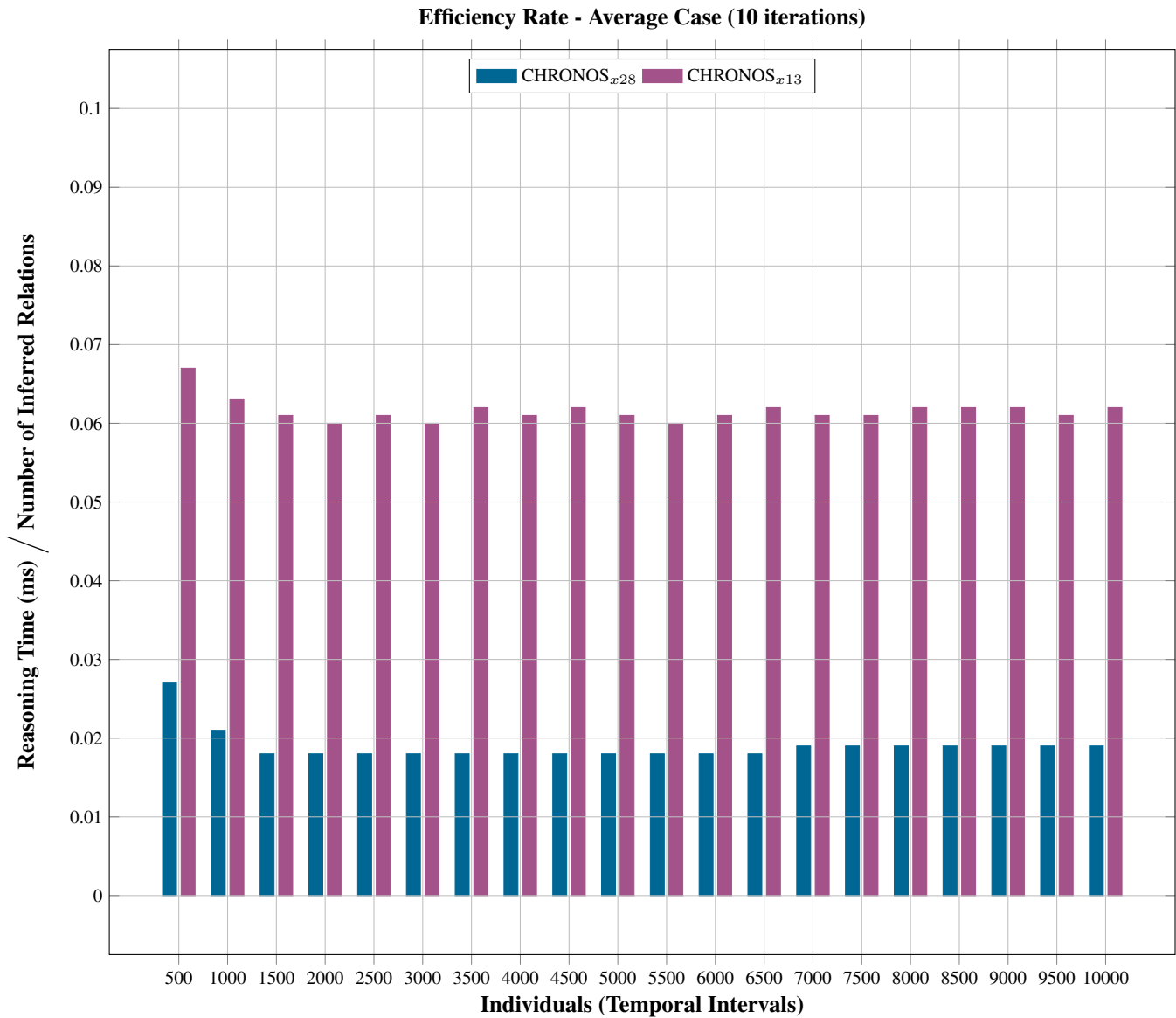
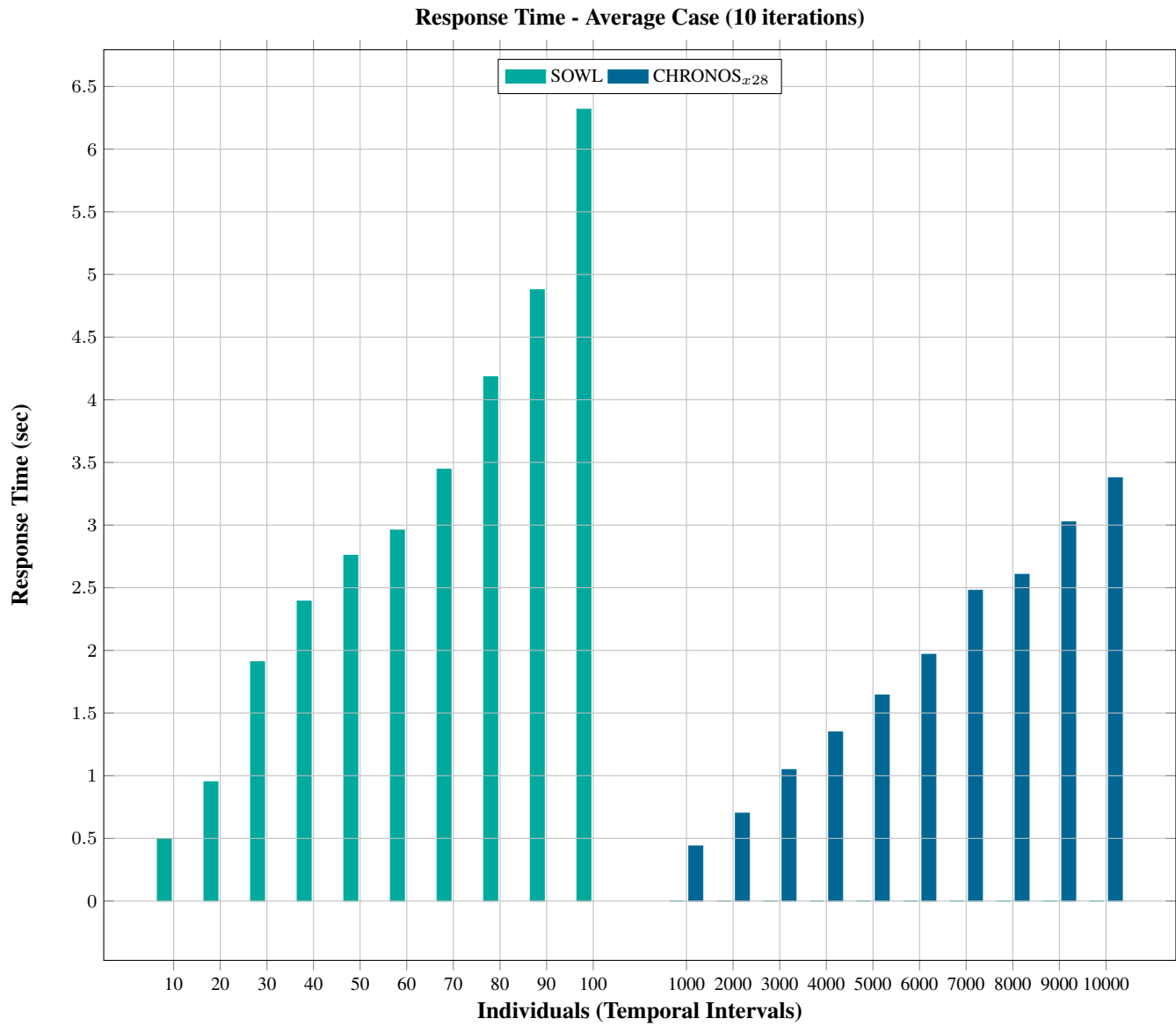**Figure 4.3:** Average efficiency rate of temporal reasoning, over 10 iterations, as a function of the number of individuals (temporal intervals). **CHRONOS**$_{x28}$ **vs CHRONOS**$_{x13}$

Observations: Both implementations demonstrate stable performance, independent of the amount of individuals. Moreover, it is also confirmed that CHRONOS$_{x28}$ performs almost three times faster than CHRONOS$_{x13}$.

	CHRONOS$_{x28}$					CHRONOS$_{x13}$			
Initial Relations	Final Relations	Parser (ms)	Reasoner (ms)	Total (ms)	EfRate	Final Relations	Parser (ms)	Reasoner (ms)	Total (ms)	EfRate
500	7067	101	188	289	0.027	7067	117	473	590	0.067
1000	14148	142	298	440	0.021	14148	149	891	1040	0.063
1500	21317	178	392	570	0.018	21317	185	1301	1486	0.061
2000	28392	201	500	701	0.018	28392	214	1703	1917	0.060
2500	36528	253	666	919	0.018	36528	256	2228	2484	0.061
3000	43083	262	787	1049	0.018	43083	272	2584	2856	0.060
3500	50675	291	911	1202	0.018	50675	294	3142	3436	0.062
4000	57165	321	1029	1350	0.018	57165	327	3487	3814	0.061
4500	65150	333	1158	1491	0.018	65150	341	4039	4380	0.062
5000	72036	375	1269	1644	0.018	72036	382	4394	4776	0.061
5500	79136	380	1425	1805	0.018	79136	408	4748	5156	0.060
6000	85907	415	1554	1969	0.018	85907	421	5241	5662	0.061
6500	92805	454	1671	2125	0.018	92805	456	5754	6210	0.062
7000	100910	496	1884	2480	0.019	100910	497	6155	6652	0.061
7500	108515	527	2010	2537	0.019	108515	527	6619	7146	0.061
8000	112972	556	2151	2607	0.019	112972	560	7004	7564	0.062
8500	123108	580	2335	2915	0.019	123108	583	7633	8216	0.062
9000	129317	599	2467	3026	0.019	129317	603	8018	8621	0.062
9500	136347	619	2530	3149	0.019	136347	622	8317	8939	0.061
10000	144029	648	2730	3378	0.019	144029	653	8929	9582	0.062

Table 4.1: The entire set of experimental measurements, in average case, as functions of the number of individuals (temporal intervals). **CHRONOS$_{x28}$ vs CHRONOS$_{x13}$**

Observations: As it is indicated by the measurements, CHRONOS$_{x28}$ clearly outperforms CHRONOS$_{x13}$ in average case, since it is almost three time faster than its counterpart.

4.2.2   Worst Case Experiments

Worst case performance is encountered when the number of inferred relations is in the order of n^2. Worst case experiments present the longest response time of CHRONOS$_{x13}$ and CHRONOS$_{x28}$, given a hard input of n individuals (range 25 to 500). Each individual is associated with others, with relations *"before"* or *"after"*. In this way, $O(n^2)$ relations are asserted into the knowledge base for n individuals.

Figure 4.4 presents measurements of reasoning time obtained by the two competing implementations, in worst case.

Figure 4.5 presents measurements of the number of inferred relations of the two competing implementations, in worst case.

Figure 4.6 presents measurements of the efficiency rate of the two competing implementations, in worst case.

Finally, Table 4.2 presents the entire set of measurements of CHRONOS$_{x13}$ and CHRONOS$_{x28}$, regarding their reasoning performance in worst case.

Figure 4.4: Response time of temporal reasoning, in worst case, as a function of the number of individuals (temporal intervals). **CHRONOS$_{x28}$ vs CHRONOS$_{x13}$**

Observations: Both implementations exhibit $O(n^3)$ response time complexity, with CHRONOS$_{x28}$ being the faster implementation.
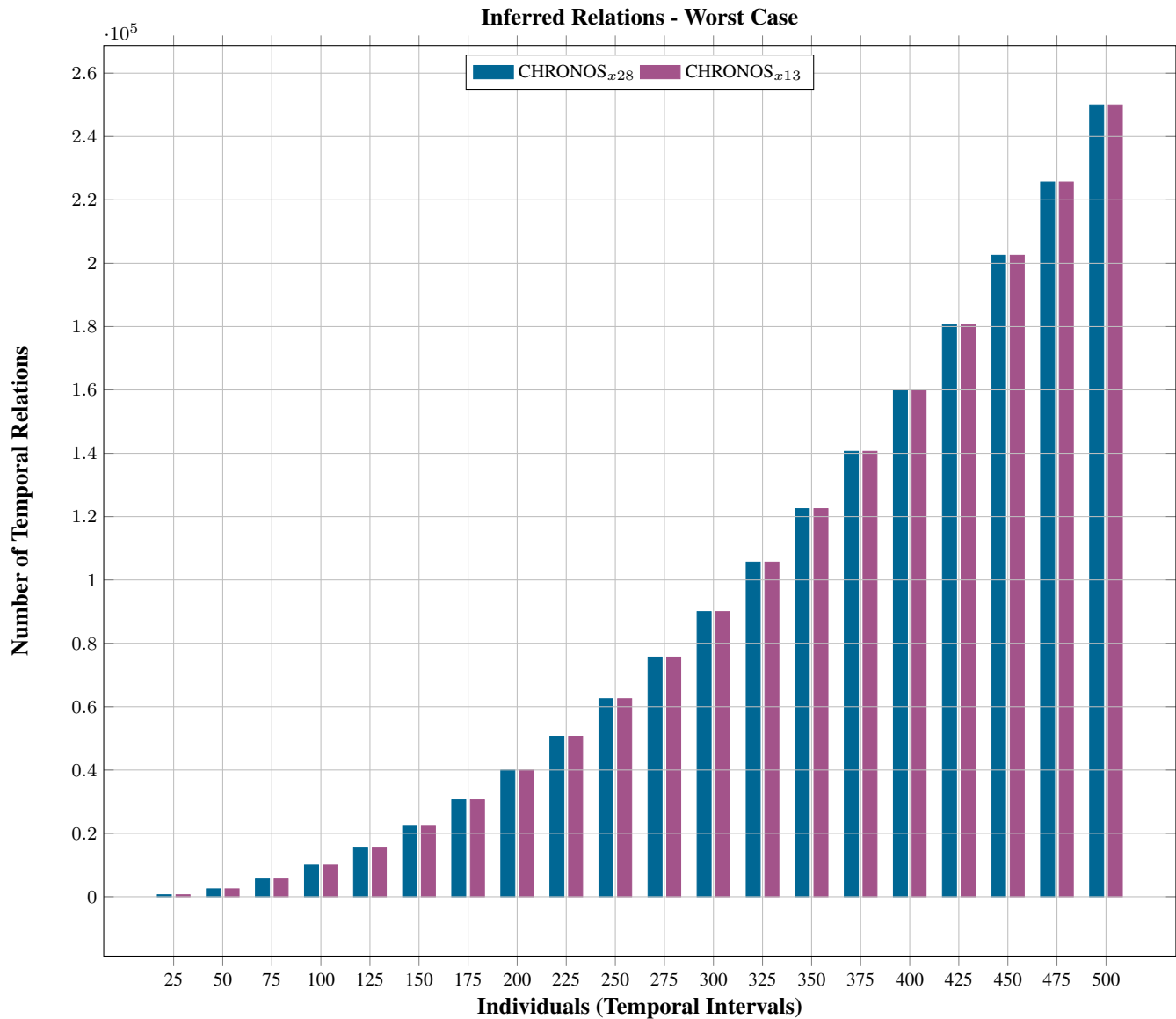
Figure 4.5: Inferred relations, in worst case, as function of the number of individuals (temporal intervals). **CHRONOS**$_{x28}$ **vs CHRONOS**$_{x13}$

Observations: Both implementations infer n^2 temporal relations for *n* individuals, proving that our data sets have been defined correctly through our experiments.

Efficiency Rate - Worst Case
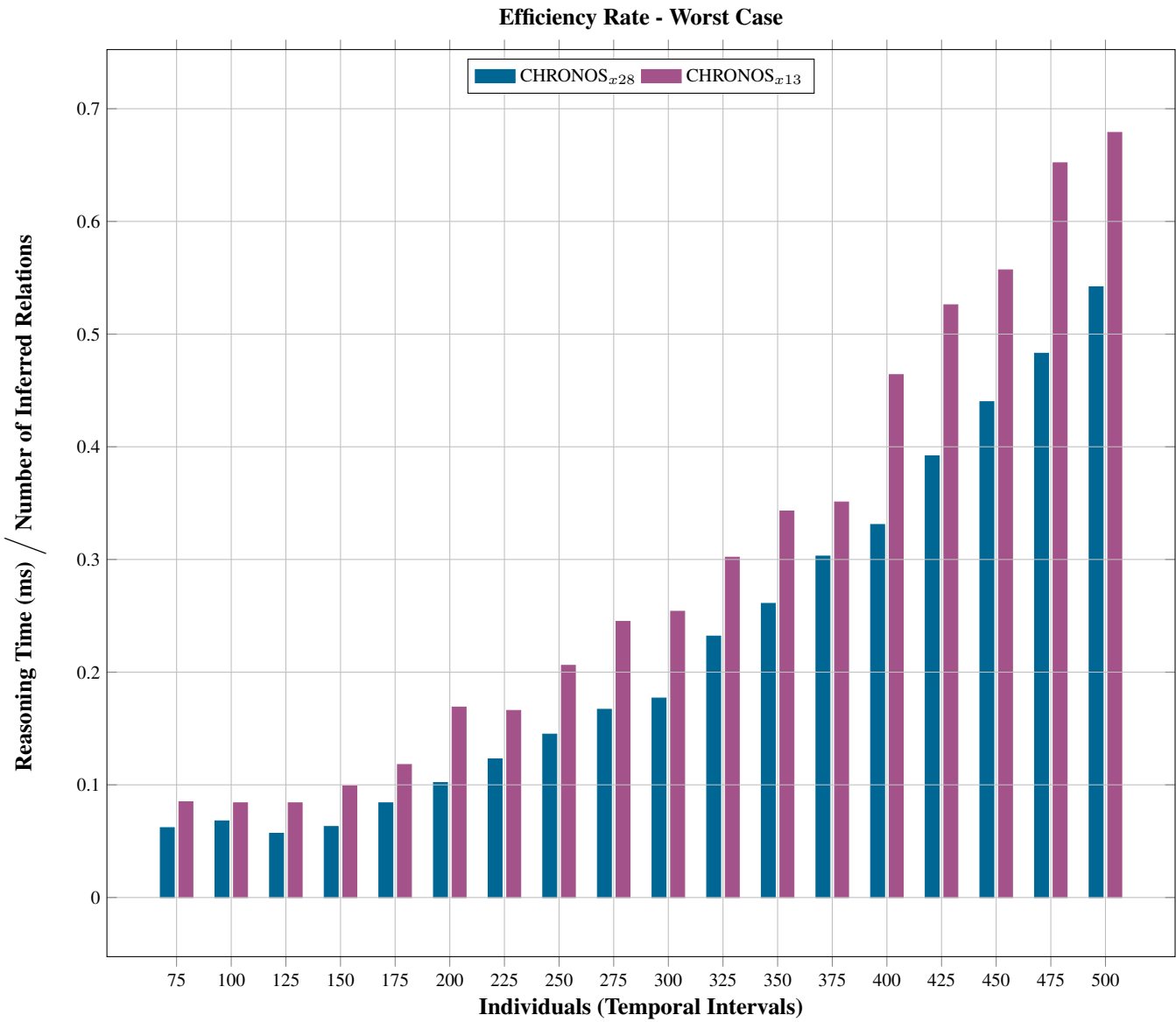
Figure 4.6: Efficiency rate of temporal reasoning, in worst case, as a function of the number of individuals (temporal intervals). **CHRONOS$_{x28}$ vs CHRONOS$_{x13}$**

Observations: Both implementations exhibit nearly linear efficiency rate, as the number of individuals increases, with CHRONOS$_{x28}$ being the faster implementation.

	CHRONOS$_{x28}$					CHRONOS$_{x13}$			
Initial Relations	Final Relations	Parser (ms)	Reasoner (ms)	Total (ms)	EfRate	Final Relations	Parser (ms)	Reasoner (ms)	Total (ms)	EfRate
25	625	78	100	178	0.160	625	62	112	252	0.179
50	2500	78	236	314	0.094	2500	78	280	420	0.112
75	5625	109	349	458	0.062	5625	125	477	664	0.085
100	10000	109	574	683	0.068	10000	109	842	1014	0.084
125	15625	125	898	1023	0.057	15625	125	1319	1522	0.084
150	22500	156	1435	1591	0.063	22500	140	2246	2464	0.099
175	30625	141	2584	2725	0.084	30625	156	3609	3843	0.118
200	40000	156	4092	4248	0.102	40000	187	6754	7066	0.169
225	50625	187	6215	6402	0.123	50625	172	8410	8675	0.166
250	62500	202	9085	9287	0.145	62500	203	12874	13171	0.206
275	75625	203	12604	12807	0.167	75625	234	18532	18859	0.245
300	90000	234	15936	16170	0.177	90000	250	22857	23200	0.254
325	105625	234	24523	24757	0.232	105625	203	31941	32269	0.302
350	122500	250	31861	32111	0.261	122500	265	42021	42395	0.343
375	140625	265	42631	42896	0.303	140625	280	49378	49768	0.351
400	160000	281	53027	53308	0.331	160000	234	74229	74588	0.464
425	180625	266	70751	71017	0.392	180625	281	94956	95361	0.526
450	202500	266	89045	89311	0.440	202500	266	112731	113121	0.557
475	225625	297	108877	109174	0.483	225625	343	147180	147648	0.652
500	250000	297	135532	135829	0.542	250000	297	169996	170418	0.679

Table 4.2: The entire set of experimental measurements, in worst case, as functions of the number of individuals (temporal intervals). **CHRONOS$_{x28}$ vs CHRONOS$_{x13}$**

Observations: As mentioned in the theoretical evaluation of CHRONOS, path consistency has $O(n^3)$ worst case complexity. Our experiments confirm our theoretical evaluation, proving that n^2 relations are inferred in worst case scenario (Figure 4.5), and that temporal reasoning exhibits nearly linear efficiency rate in both implementations of CHRONOS (Figure 4.6). Hence, we conclude that CHRONOS exhibits $O(n^3)$ response time complexity in worst case (Figure 4.4).

4.3 Run-Time Efficiency Comparisons

The purpose of the following set of experiments is to demonstrate the run-time efficiency of CHRONOS over SOWL [8], a temporal reasoner implemented in SWRL, and CHOROS [7], a spatial reasoner implemented in Java, which also support reasoning over qualitative data.

4.3.1 CHRONOS vs SOWL

SOWL is an ontology for representing and reasoning over spatiotemporal information in OWL. Temporal reasoning in SOWL is realized by introducing a set SWRL rules for asserting inferred temporal Allen relations.

In our comparisons, the running time of both implementations of CHRONOS and SOWL is demonstrated, as function of the number of temporal assertions in the input ontology. In the average case, the number of inferred relations is in the order of n while in the worst case, n^2 temporal relations are inferred for n individuals.

Both implementations of CHRONOS clearly outperform their SWRL competitor in all cases. The run-time performance of SOWL declines drastically for large data sets as the large number of inferred relations caused the memory to overflow. Although the SOWL reasoner may perform better on computers with more memory, CHRONOS scales-up much better than SOWL with the size of the input (i.e., the performance gap between the two reasoner implementations increases with the size of the data set) and will run much faster than SOWL for large data sets even on average computers.

Response Time (ms)	Average Case			Worst Case		
Individuals	CHRONOS$_{x28}$	CHRONOS$_{x13}$	SOWL	CHRONOS$_{x28}$	CHRONOS$_{x13}$	SOWL
10	40	52	500	87	103	1142
20	58	73	952	143	156	2185
30	72	86	1909	202	213	2887
40	85	99	2393	264	304	5398
50	97	113	2759	314	420	7443
60	108	122	2960	368	531	11048
70	117	135	3446	425	645	16443
80	126	148	4184	503	773	-
90	135	162	4878	598	889	-
100	142	187	6319	683	1014	-

Table 4.3: Response Time, **CHRONOS vs SOWL.**

Figure 4.7: Average response time of temporal reasoning, over 10 iterations, as a function of the number of individuals (temporal intervals). **CHRONOS**$_{x28}$ **vs SOWL**
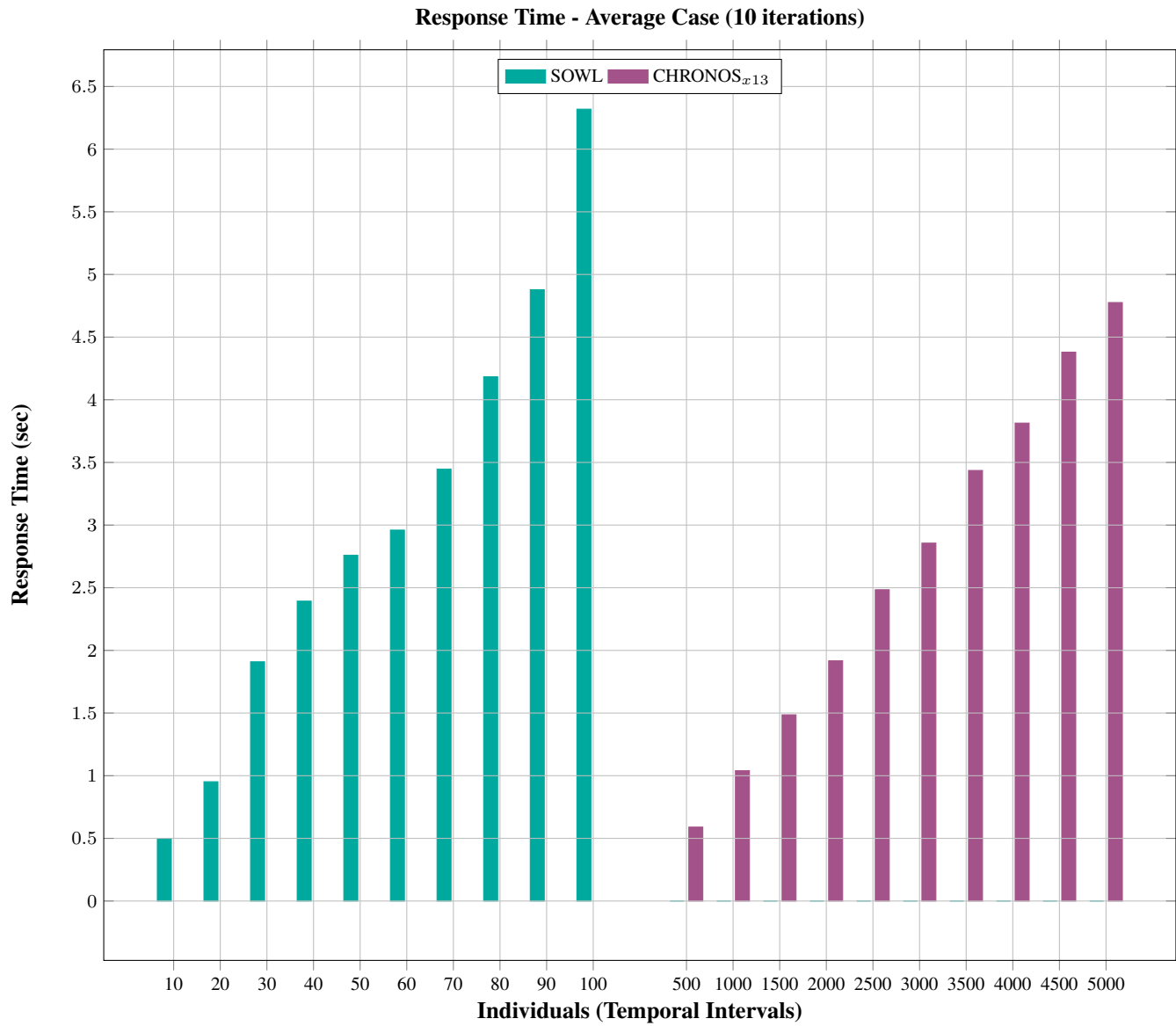
Figure 4.8: Average response time of temporal reasoning, over 10 iterations, as a function of the number of individuals (temporal intervals). **CHRONOS$_{x13}$ vs SOWL**

Response Time - Worst Case (10 iterations)
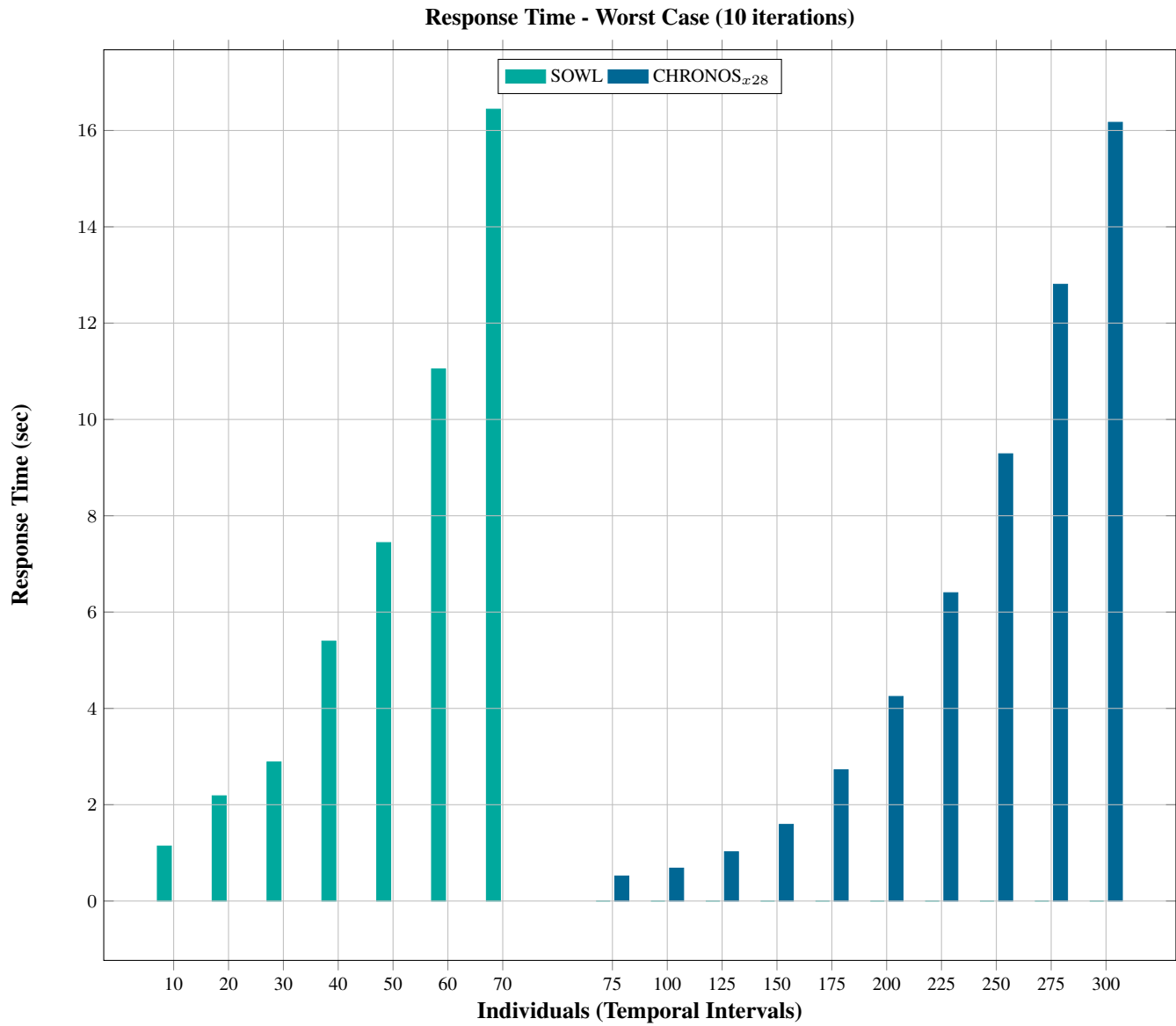
Figure 4.9: Response time of temporal reasoning, in worst case, as a function of the number of individuals (temporal intervals). **CHRONOS$_{x28}$ vs SOWL**
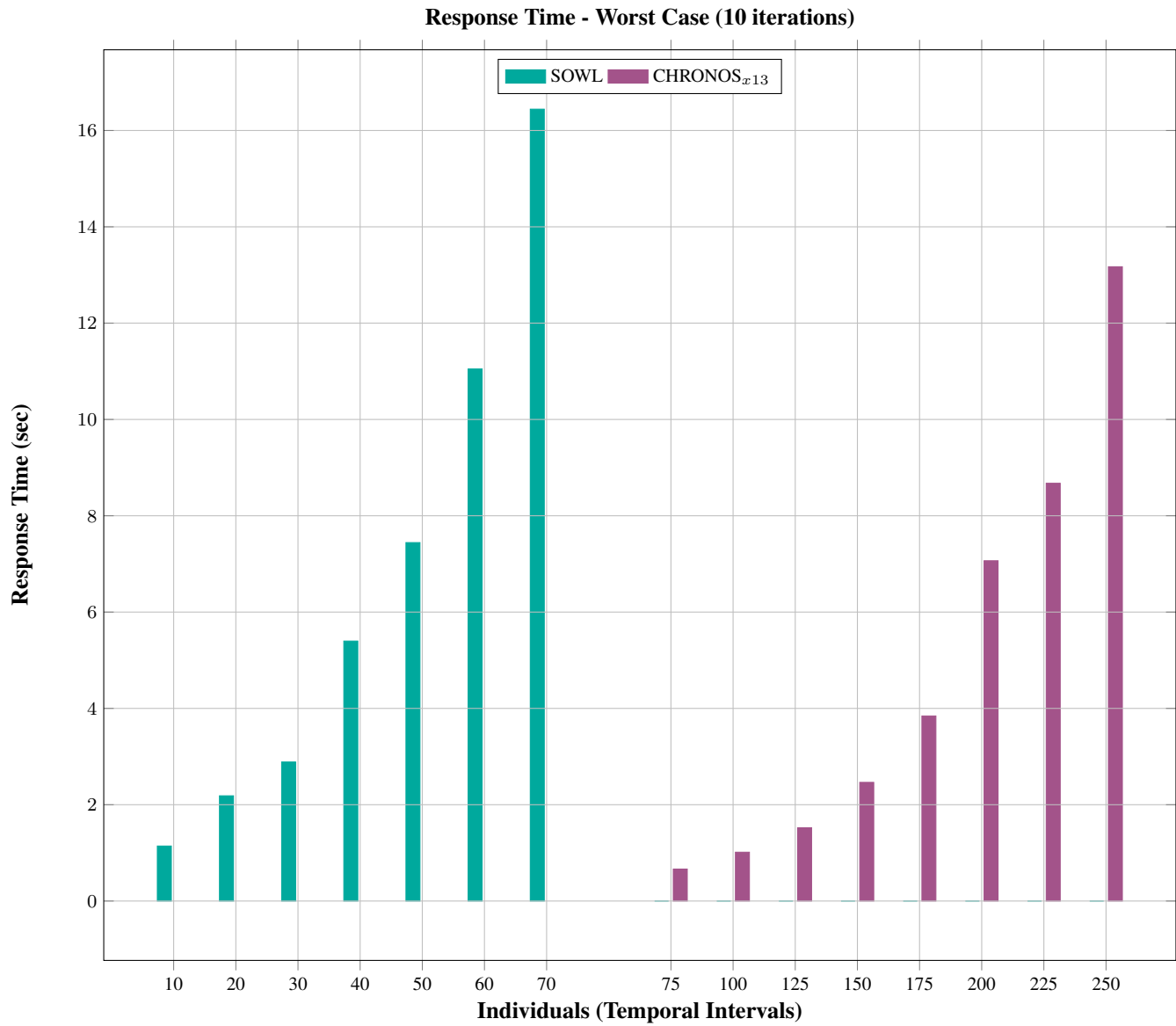
Figure 4.10: Response time of temporal reasoning, in worst case, as a function of the number of individuals (temporal intervals). **CHRONOS$_{x13}$ vs SOWL**

4.3.2 CHRONOS$_{x28}$ vs CHOROS

CHOROS is a reasoner which supports consistency checking over spatial information in OWL, expressed using the Region Connection Calculus (RCC) or the Cone-Shaped Directional logic formalism (CSD). Choosing either representation is a design decision that depends mainly on the application. However, both RCC and CSD expressions in OWL may co-exist within the same ontology together with standard OWL semantic relations. In that respect, CHOROS extends Pellet-Spatial to support CSD-9 relations in addition to RCC-8 relations.

Our experiments focused solely on the CSD-9 part of CHOROS, in order to examine the performance of the static composition table of CHRONOS, which stores all the possible compositions over basic and disjunctive CSD relations, versus our reasoning approach used in CHRONOS$_{x28}$:

	CHRONOS$_{x28}$	CHOROS
Relational Model	Temporal Relations of Allen	CSD-9 Spatial Relations
Basic Relations	13	8
Possible Disjunctions	$2^{13} = 65,536$ relations	$2^8 = 256$ relations
Compositions of Disjunctions	CHRONOS$_{x28}$ uses a tractable set of 28 basic and disjunctive relations and pre-computes totally 28×28 possible compositions of them. CHRONOS$_{x28}$ does not contain any static structure for storing of compositions. Pre-computed compositions are used at run-time through functions, which simulate the storing process, in order to save the memory that would have been used by the structure.	Following the example of Pellet-Spatial [11], CHOROS pre-computes all the possible compositions over basic and disjunctive CSD relations, total $2^8 \times 2^8$ relations, and stores them into a static full composition table which contains $2^{16} = 65,536$ relations.

Table 4.4: CHRONOS$_{x28}$ vs CHOROS.

Although CSD relational model comprises of 9 basic relations, CHOROS applies to consistency checking over a CSD calculus of 8 basic relations. Spatial relation *"identical to"* is replaced by the owl axiom *"sameAs"*.

In our experiments, we demonstrate comparisons regarding the running time of the two competitor implementations as a function of the number of temporal assertions in the input ontology. The charts below illustrate comparisons between the two reasoners regarding their reasoning performance in both average and worst case.
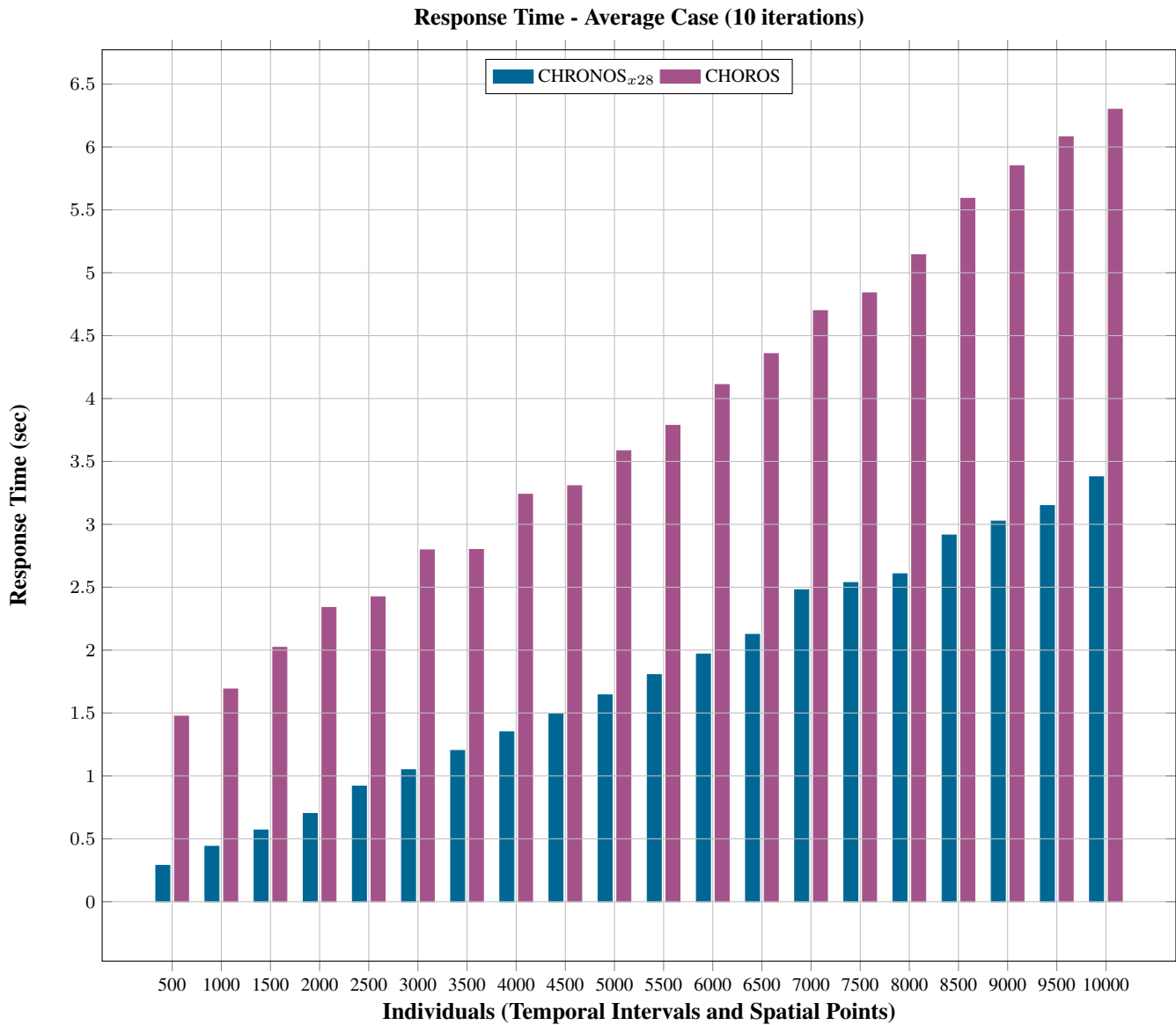
Figure 4.11: Average response time of temporal and spatial reasoning, over 10 iterations, as a function of the number of individuals (temporal intervals and spatial points). **CHRONOS$_{x28}$ vs CHOROS**

Observations: Both implementations exhibit linear response time (O(n) response time complexity) in average case. CHRONOS$_{x28}$ is almost two times faster than its counterpart, although CHRONOS processes fewer relations.
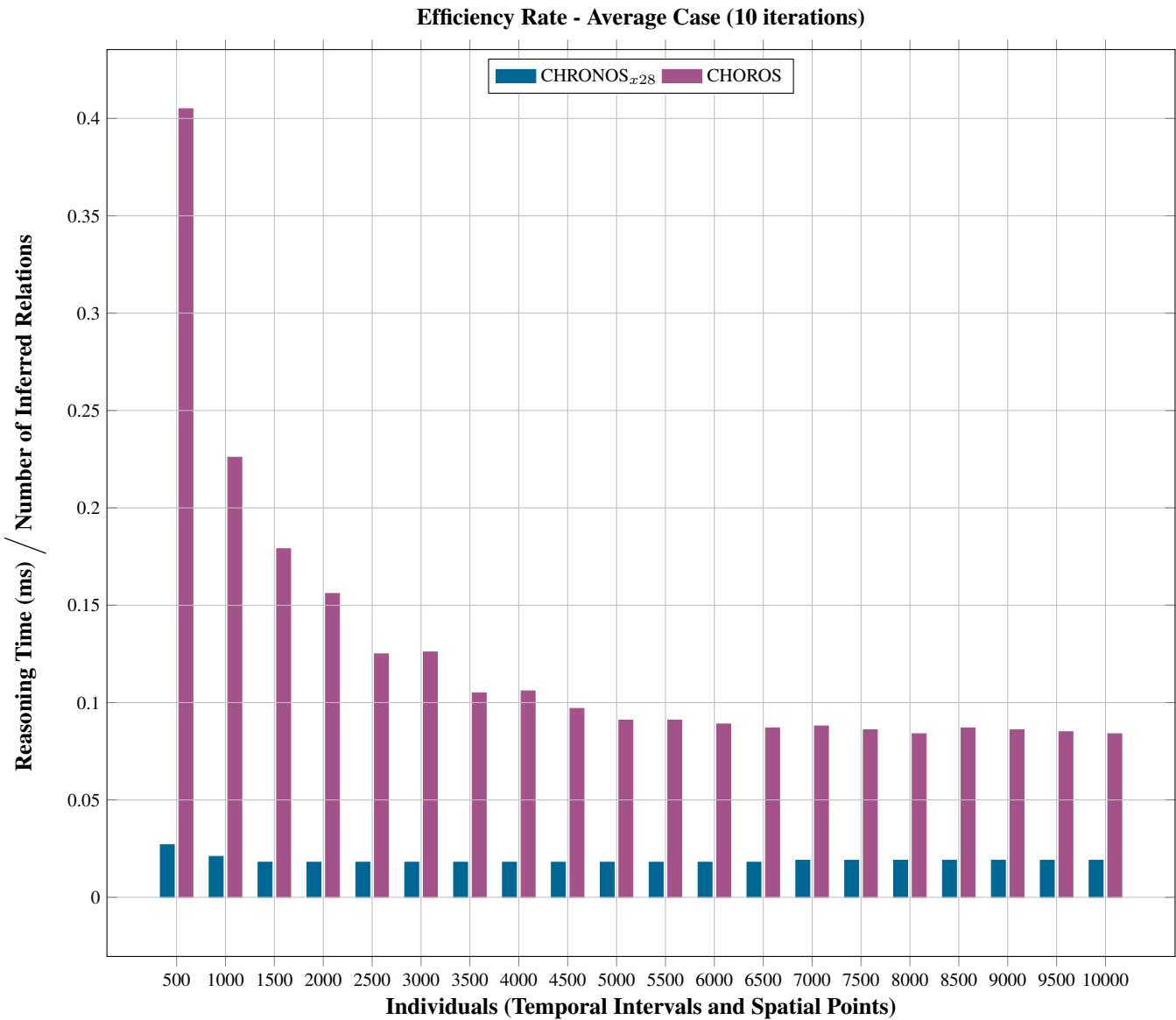
Figure 4.12: Average efficiency rate of temporal and spatial reasoning, over 10 iterations, as a function of the number of individuals (temporal intervals and spatial points). **CHRONOS$_{x28}$ vs CHOROS**

Observations: This measurement helps us evaluate the performance of the two implementations, regardless the number of relations being processed. CHRONOS$_{x28}$ clearly outperforms CHOROS, featuring stable performance in all cases. On the other hand, it is worth to observe the variation in performance of CHOROS for smaller data sets, because of the time required to initialize the composition table.

Response Time - Worst Case
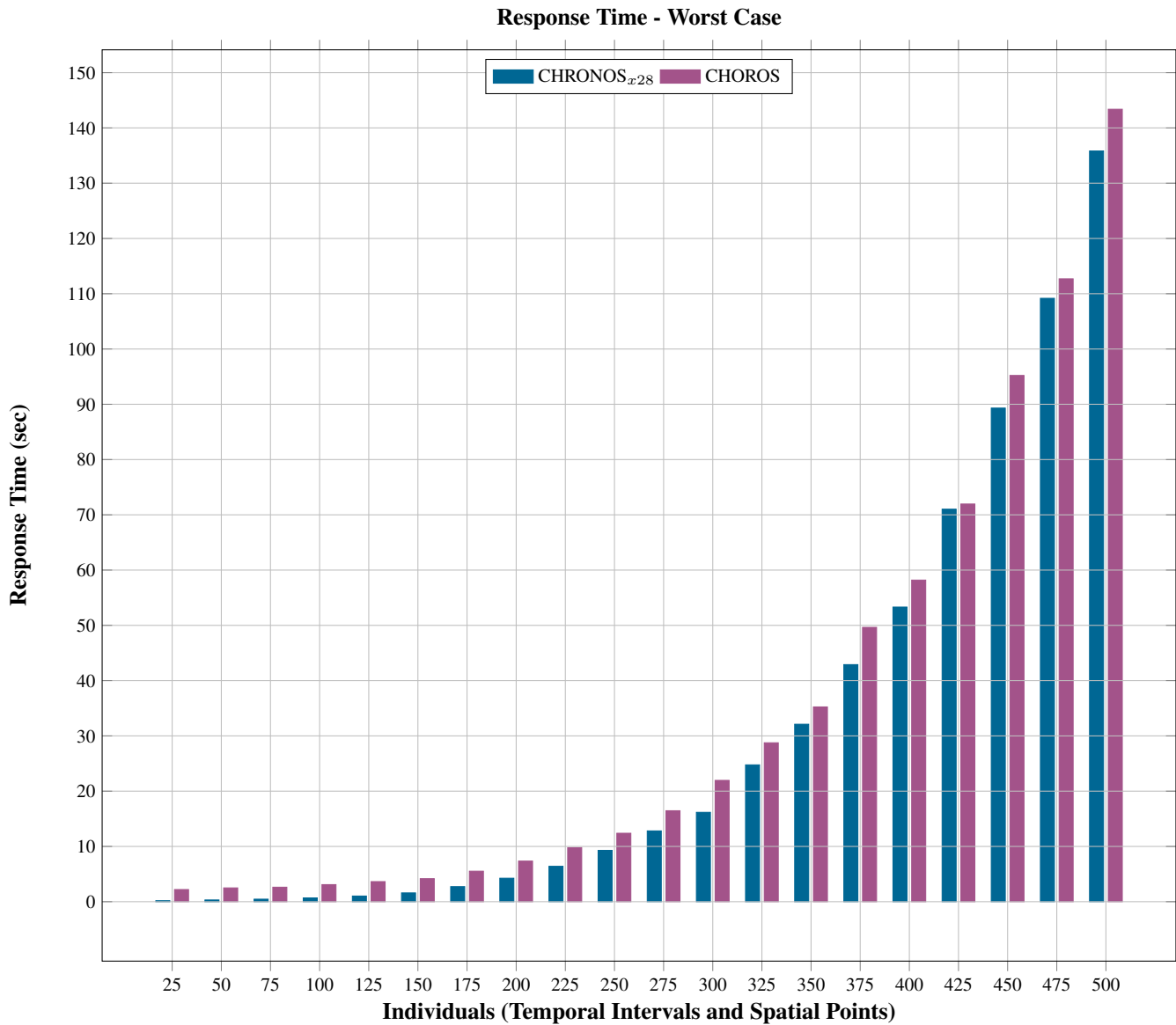
Figure 4.13: Response time of temporal and spatial reasoning, in worst case, as a function of the number of individuals (temporal intervals and spatial points). **CHRONOS$_{x28}$ vs CHOROS**

Observations: As already mentioned, both implementations infer n^2 relations for n individuals in wort case. CHRONOS$_{x28}$ is consistently faster than its counterpart, while both implementations exhibit $O(n^3)$ response time complexity.
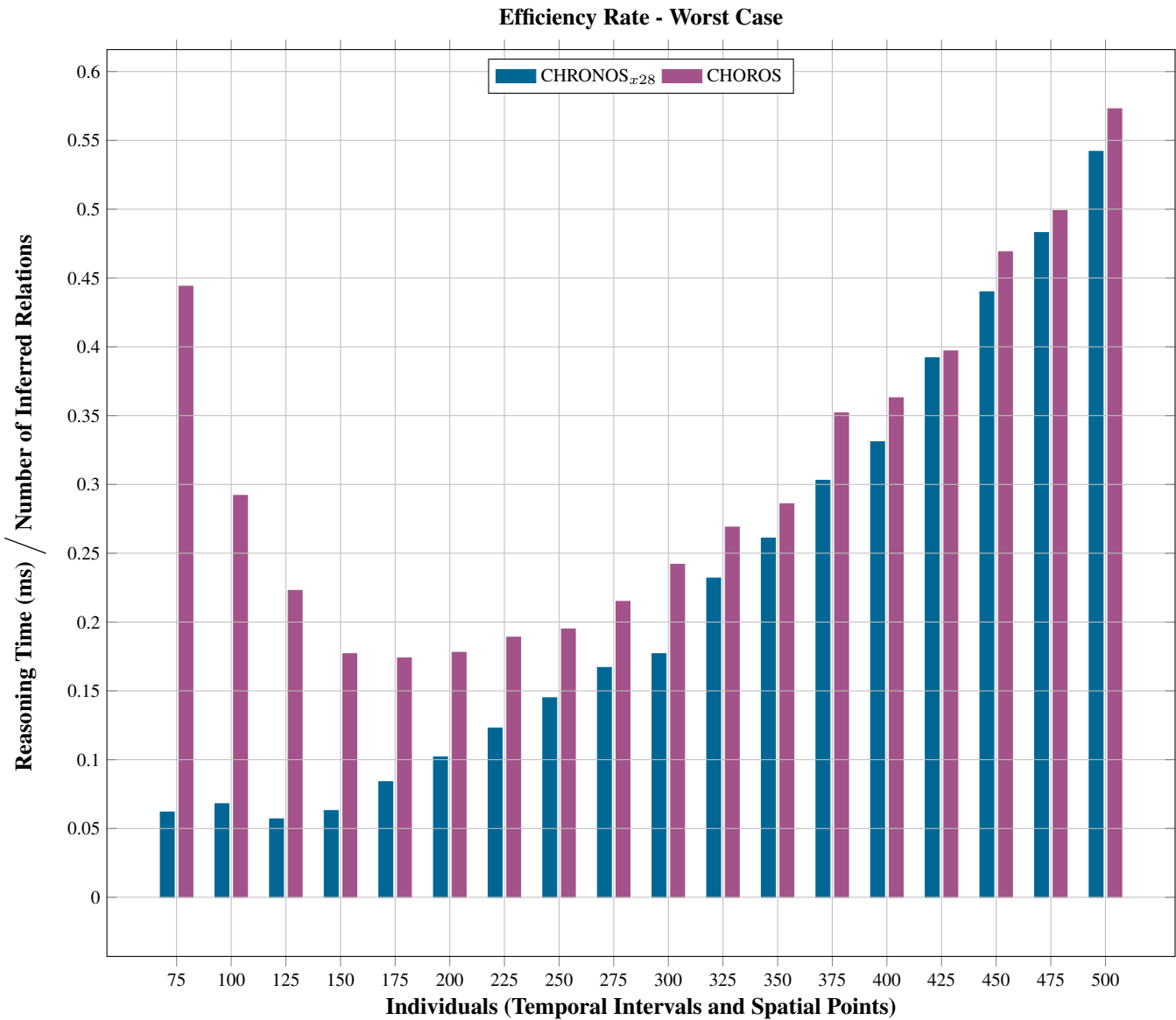
Figure 4.14: Efficiency rate of temporal and spatial reasoning, in worst case, as a function of the number of individuals (temporal intervals and spatial points). **CHRONOS$_{x28}$ vs CHOROS**

Observations: As mentioned in the previous figure, CHRONOS$_{x28}$ is the faster implementation featuring linear efficiency rate, while CHOROS exhibits variation in its performance for smaller data sets, due to the time required to initialize its composition table.

Chapter 5

Conclusions and Future Work

We propose CHRONOS, a stand-alone temporal OWL reasoner following the example of Pellet-Spatial and CHOROS for reasoning over temporal knowledge in ontologies using OWL. CHRONOS implements certain optimizations and exhibited increased performance improvements over SOWL in both the average and the worst cases. Extending CHRONOS to handle both temporal instants in addition to temporal intervals (as SOWL does) is a direction of future work.

Reasoning over Point-Based Representations

Point-based representations assume linear ordering of time points with three possible relations the "$<$", "$>$" and "$=$" often referred to as *Before, After* and *Equals* respectively.

	Before	**After**	**Equals**
Before	Before	{Before, After, Equals}	Before
After	{Before, After, Equals}	After	After
Equals	Before	After	Equals

Table 5.1: Composition Table for point-based temporal relations.

The composition table represents the result of the composition of two temporal relations. For example, if relation R_1 holds between $instant_1$ and $instant_2$ and relation R_2 holds between $instant_2$ and $instant_3$ then, the entry of Table 5.1 corresponding to row R_1 and column R_2 denotes the possible relation(s) holding between $instant1$ and $instant3$.

As shown in Table 5.1, compositions of relations may yield one of the following four relations: Before, After, Equals and the disjunction of these three relations. Intersecting the disjunction of all three relations with any of these leaves existing

relations unchanged. Intersecting any one of the tree basic relations with itself also leaves existing relations unaffected. Only compositions of pairs of different basic relations affect the ontology by yielding the empty relation as a result, thus detecting an inconsistency. We can define all intersections that can affect the ontology, by declaring the three basic relations as pairwise disjoint, since they cannot simultaneously hold between two instants.

As shown in Table 5.2, the temporal relations of Allen can be represented by means of point relations between their endpoints, since intervals can be defined as ordered pairs of points s and e with $s < e$, often referred as start and end of an interval respectively:

Relation		Inverse Relation
Before(i_1, i_2)	$Before(e_1, s_2)$	**After**(i_2, i_1)
Meets(i_1, i_2)	$Equals(e_1, s_2)$	**MetBy**(i_2, i_1)
Overlaps(i_1, i_2)	$Before(s_1, s_2) \cap Before(e_1, e_2) \cap After(e_1, s_2)$	**OverlappedBy**(i_2, i_1)
Starts(i_1, i_2)	$Equals(s_1, s_2) \cap Before(e_1, e_2)$	**StartedBy**(i_2, i_1)
During(i_1, i_2)	$After(s_1, s_2) \cap Before(e_1, e_2)$	**Contains**(i_2, i_1)
Finishes(i_1, i_2)	$After(s_1, s_2) \cap Equals(e_1, e_2)$	**FinishedBy**(i_2, i_1)
Equals(i_1, i_2)	$Equals(s_1, s_2) \cap Equals(e_1, e_2)$	

Table 5.2: The temporal relations of Allen, represented by means of point relations between their endpoints.

It is our belief that a reasoning engine based on the composition table for point-based relations would be more efficient, since there will be no disjunctive relations. On the other hand, there is no guarantee for better overall performance, since there will be delays in the conversion of intervals to points and the other way around.

Appendix A

The Tractable Set of Allen Relations

The following section sets out to prove that a reasoning mechanism over the Allen relations, can be based on a tractable set of relations. These relations are the thirteen basic relations of Allen and sixteen possible disjunctions of them represented as sets of relations.

As already mentioned, there are thirteen basic relations of Allen: {**Before**}, {**After**}, {**Meets**}, {**MetBy**}, {**Overlaps**}, {**OverlappedBy**}, {**Starts**}, {**StartedBy**}, {**During**}, {**Contains**}, {**Finishes**}, {**FinishedBy**} and {**Equals**}. Consequently there are $13 \times 13 = 169$ possible compositions of them, as presented bellow:

1) Compositions of {Before}:

{Before} ∘ {Before} → {Before}

{Before} ∘ {After} → {**Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}

{Before} ∘ {Meets} → {Before}

{Before} ∘ {MetBy} → {**Before, Meets, Overlaps, Starts, During**}

{Before} ∘ {Overlaps} → {Before}

{Before} ∘ {OverlappedBy} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Starts} → {Before}

{Before} ∘ {StartedBy} → {Before}

{Before} ∘ {During} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Contains} → {Before}

{Before} ∘ {Finishes} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {FinishedBy} → {Before}

56

{Before} ∘ {Equals} → {Before}

2) Compositions of {After}:

{After} ∘ {Before} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After} ∘ {After} → {After}

{After} ∘ {Meets} → {**After, MetBy, OverlappedBy, During, Finishes**}

{After} ∘ {MetBy} → {After}

{After} ∘ {Overlaps} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {OverlappedBy} → {After}

{After} ∘ {Starts} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {StartedBy} → {After}

{After} ∘ {During} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {Contains} → {After}

{After} ∘ {Finishes} → {After}

{After} ∘ {FinishedBy} → {After}

{After} ∘ {Equals} → {After}

3) Compositions of {Meets}:

{Meets} ∘ {Before} → {Before}

{Meets} ∘ {After} → {**After, MetBy, OverlappedBy, StartedBy, Contains**}

{Meets} ∘ {Meets} → {Before}

{Meets} ∘ {MetBy} → {**Finishes, FinishedBy, Equals**}

{Meets} ∘ {Overlaps} → {Before}

{Meets} ∘ {OverlappedBy} → {**Overlaps, Starts, During**}

{Meets} ∘ {Starts} → {Meets}

{Meets} ∘ {StartedBy} → {Meets}

{Meets} ∘ {During} → {Overlaps, Starts, During}

{Meets} ∘ {Contains} → {Before}

{Meets} ∘ {Finishes} → {Overlaps, Starts, During}

{Meets} ∘ {FinishedBy} → {Before}

{Meets} ∘ {Equals} → {Meets}

4) Compositions of {MetBy}:

{MetBy} ∘ {Before} → {**Before, Meets, Overlaps, Contains, FinishedBy**}

{MetBy} ∘ {After} → {After}

{MetBy} ∘ {Meets} → {**Starts, StartedBy, Equals**}

{MetBy} ∘ {MetBy} → {After}

{MetBy} ∘ {Overlaps} → {**OverlappedBy, During, Finishes**}

{MetBy} ∘ {OverlappedBy} → {After}

{MetBy} ∘ {Starts} → {OverlappedBy, During, Finishes}

{MetBy} ∘ {StartedBy} → {After}

{MetBy} ∘ {During} → {OverlappedBy, During, Finishes}

{MetBy} ∘ {Contains} → {After}

{MetBy} ∘ {Finishes} → {MetBy}

{MetBy} ∘ {FinishedBy} → {MetBy}

{MetBy} ∘ {Equals} → {MetBy}

5) Compositions of {Overlaps}:

{Overlaps} ∘ {Before} → {Before}

{Overlaps} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps} ∘ {Meets} → {Before}

{Overlaps} ∘ {MetBy} → {**OverlappedBy, StartedBy, Contains**}

{Overlaps} ∘ {Overlaps} → {**Before, Meets, Overlaps**}

{Overlaps} ∘ {OverlappedBy} → {**Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}

{Overlaps} ∘ {Starts} → {Overlaps}

{Overlaps} ∘ {StartedBy} → {**Overlaps, Contains, FinishedBy**}

{Overlaps} ∘ {During} → {Overlaps, Starts, During}

{Overlaps} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps} ∘ {Finishes} → {Overlaps, Starts, During}

{Overlaps} ∘ {FinishedBy} → {Before, Meets, Overlaps}

{Overlaps} ∘ {Equals} → {Overlaps}

6) Compositions of {OverlappedBy}:

{OverlappedBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{OverlappedBy} ∘ {After} → {After}

{OverlappedBy} ∘ {Meets} → {Overlaps, Contains, FinishedBy}

{OverlappedBy} ∘ {MetBy} → {After}

{OverlappedBy} ∘ {Overlaps} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {OverlappedBy} → {**After, MetBy, OverlappedBy**}

{OverlappedBy} ∘ {Starts} → {OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {StartedBy} → {After, MetBy, OverlappedBy}

{OverlappedBy} ∘ {During} → {OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Finishes} → {OverlappedBy}

{OverlappedBy} ∘ {FinishedBy} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Equals} → {OverlappedBy}

7) Compositions of {**Starts**}:

{Starts} ∘ {Before} → {Before}

{Starts} ∘ {After} → {After}

{Starts} ∘ {Meets} → {Before}

{Starts} ∘ {MetBy} → {MetBy}

{Starts} ∘ {Overlaps} → {Before, Meets, Overlaps}

{Starts} ∘ {OverlappedBy} → {OverlappedBy, During, Finishes}

{Starts} ∘ {Starts} → {Starts}

{Starts} ∘ {StartedBy} → {Starts, StartedBy, Equals}

{Starts} ∘ {During} → {During}

{Starts} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts} ∘ {Finishes} → {During}

{Starts} ∘ {FinishedBy} → {Before, Meets, Overlaps}

{Starts} ∘ {Equals} → {Starts}

8) Compositions of {**StartedBy**}:

{StartedBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{StartedBy} ∘ {After} → {After}

{StartedBy} ∘ {Meets} → {Overlaps, Contains, FinishedBy}

{StartedBy} ∘ {MetBy} → {MetBy}

{StartedBy} ∘ {Overlaps} → {Overlaps, Contains, FinishedBy}

{StartedBy} ∘ {OverlappedBy} → {OverlappedBy}

{StartedBy} ∘ {Starts} → {Starts, StartedBy, Equals}

{StartedBy} ∘ {StartedBy} → {StartedBy}

{StartedBy} ∘ {During} → {OverlappedBy, During, Finishes}

{StartedBy} ∘ {Contains} → {Contains}

{StartedBy} ∘ {Finishes} → {OverlappedBy}

{StartedBy} ∘ {FinishedBy} → {Contains}

{StartedBy} ∘ {Equals} → {StartedBy}

9) Compositions of {During}:

{During} ∘ {Before} → {Before}

{During} ∘ {After} → {After}

{During} ∘ {Meets} → {Before}

{During} ∘ {MetBy} → {After}

{During} ∘ {Overlaps} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {Starts} → {During}

{During} ∘ {StartedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {During} → {During}

{During} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {Finishes} → {During}

{During} ∘ {FinishedBy} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {Equals} → {During}

10) Compositions of {Contains}:

{Contains} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Contains} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Meets} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {MetBy} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Overlaps} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {OverlappedBy} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Starts} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {StartedBy} → {Contains}

{Contains} ∘ {During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {Contains} → {Contains}

{Contains} ∘ {Finishes} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {FinishedBy} → {Contains}

{Contains} ∘ {Equals} → {Contains}

11) Compositions of {Finishes}:

{Finishes} ∘ {Before} → {Before}

{Finishes} ∘ {After} → {After}

{Finishes} ∘ {Meets} → {Meets}

{Finishes} ∘ {MetBy} → {After}

{Finishes} ∘ {Overlaps} → {Overlaps, Starts, During}

{Finishes} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy}

{Finishes} ∘ {Starts} → {During}

{Finishes} ∘ {StartedBy} → {After, MetBy, OverlappedBy}

{Finishes} ∘ {During} → {During}

{Finishes} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes} ∘ {Finishes} → {Finishes}

{Finishes} ∘ {FinishedBy} → {Finishes, FinishedBy, Equals}

{Finishes} ∘ {Equals} → {Finishes}

12) Compositions of {FinishedBy}:

{FinishedBy} ∘ {Before} → {Before}

{FinishedBy} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{FinishedBy} ∘ {Meets} → {Meets}

{FinishedBy} ∘ {MetBy} → {OverlappedBy, StartedBy, Contains}

{FinishedBy} ∘ {Overlaps} → {Overlaps}

{FinishedBy} ∘ {OverlappedBy} → {OverlappedBy, StartedBy, Contains}

{FinishedBy} ∘ {Starts} → {Overlaps}

{FinishedBy} ∘ {StartedBy} → {Contains}

{FinishedBy} ∘ {During} → {Overlaps, Starts, During}

{FinishedBy} ∘ {Contains} → {Contains}

{FinishedBy} ∘ {Finishes} → {Finishes, FinishedBy, Equals}

{FinishedBy} ∘ {FinishedBy} → {FinishedBy}

{FinishedBy} ∘ {Equals} → {FinishedBy}

13) Compositions of {Equals}:

{Equals} ∘ {Before} → {Before}

{Equals} ∘ {After} → {After}

{Equals} ∘ {Meets} → {Meets}

{Equals} ∘ {MetBy} → {MetBy}

{Equals} ∘ {Overlaps} → {Overlaps}

{Equals} ∘ {OverlappedBy} → {OverlappedBy}

{Equals} ∘ {Starts} → {Starts}

{Equals} ∘ {StartedBy} → {StartedBy}

{Equals} ∘ {During} → {During}

{Equals} ∘ {Contains} → {Contains}

{Equals} ∘ {Finishes} → {Finishes}

{Equals} ∘ {FinishedBy} → {FinishedBy}

{Equals} ∘ {Equals} → {Equals}

As we can deduce, fourteen new relationships resulting from these compositions and are the following:

{**Finishes, FinishedBy, Equals**},
{**Overlaps, Starts, During**},
{**Starts, StartedBy, Equals**},
{**OverlappedBy, During, Finishes**},
{**OverlappedBy, StartedBy, Contains**},
{**Before, Meets, Overlaps**},
{**Overlaps, Contains, FinishedBy**},
{**After, MetBy, OverlappedBy**},
{**Before, Meets, Overlaps, Starts, During**},
{**After, MetBy, OverlappedBy, During, Finishes**},

{**After, MetBy, OverlappedBy, StartedBy, Contains**},
{**Before, Meets, Overlaps, Contains, FinishedBy**},
{**Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**},
{**Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}

Therefore, the total number of temporal relations is now twenty-seven relations. If we continue now with the calculation of the $27 \times 27 = 729$ new compositions, we will find out that only two compositions generating new relationships and are the following:

{Meets} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {**After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}

{MetBy} ∘ {Before, Meets, Overlaps, Starts, During} → {**Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}

Hence, the total number of temporal relationships is now twenty-nine, which is the final tractable set, because as we continue with our compositions no new relations are generated:

{**Before**}, {**After**}, {**Meets**}, {**MetBy**}, {**Overlaps**}, {**OverlappedBy**}, {**Starts**},
{**StartedBy**}, {**During**}, {**Contains**}, {**Finishes**}, {**FinishedBy**}, {**Equals**},
{**Finishes, FinishedBy, Equals**}, {**Overlaps, Starts, During**},
{**Starts, StartedBy, Equals**}, {**OverlappedBy, During, Finishes**},
{**OverlappedBy, StartedBy, Contains**}, {**Before, Meets, Overlaps**},
{**Overlaps, Contains, FinishedBy**}, {**After, MetBy, OverlappedBy**},
{**Before, Meets, Overlaps, Starts, During**},
{**After, MetBy, OverlappedBy, During, Finishes**},
{**After, MetBy, OverlappedBy, StartedBy, Contains**},
{**Before, Meets, Overlaps, Contains, FinishedBy**},
{**Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**},
{**Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**},
{**After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**},
{**Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**},

Appendix B

Compositions of CHRONOS$_{x28}$

A reasoning mechanism over the temporal Allen relations, can be based on a tractable set of the following twenty-nine relations (Appendix A):

{**Before**}, {**After**}, {**Meets**}, {**MetBy**}, {**Overlaps**}, {**OverlappedBy**}, {**Starts**}, {**StartedBy**}, {**During**}, {**Contains**}, {**Finishes**}, {**FinishedBy**}, {**Equals**}, {**Finishes, FinishedBy, Equals**}, {**Overlaps, Starts, During**}, {**Starts, StartedBy, Equals**}, {**OverlappedBy, During, Finishes**}, {**OverlappedBy, StartedBy, Contains**}, {**Before, Meets, Overlaps**}, {**Overlaps, Contains, FinishedBy**}, {**After, MetBy, OverlappedBy**}, {**Before, Meets, Overlaps, Starts, During**}, {**After, MetBy, OverlappedBy, During, Finishes**}, {**After, MetBy, OverlappedBy, StartedBy, Contains**}, {**Before, Meets, Overlaps, Contains, FinishedBy**}, {**Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}, {**Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}, {**After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**}, {**Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals**},

We can improve the performance of our reasoning mechanism, by observing that the disjunction of all basic Allen relations when composed with other relations yields the same relation:

Compositions of {All} = {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}:

{All} ∘ {Before} → {All}

{All} ∘ {After} → {All}

{All} ∘ {Meets} → {All}

{All} ∘ {MetBy} → {All}

{All} ∘ {Overlaps} → {All}

{All} ∘ {OverlappedBy} → {All}

{All} ∘ {Starts} → {All}

{All} ∘ {StartedBy} → {All}

{All} ∘ {During} → {All}

{All} ∘ {Contains} → {All}

{All} ∘ {Finishes} → {All}

{All} ∘ {FinishedBy} → {All}

{All} ∘ {Equals} → {All}

Therefore, based on this property, we can use a composition table which contains $28 \times 28 = 784$ possible compositions, as presented below:

1) Compositions of {Before}:

{Before} ∘ {Before} → {Before}

{Before} ∘ {After} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before} ∘ {Meets} → {Before}

{Before} ∘ {MetBy} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Overlaps} → {Before}

{Before} ∘ {OverlappedBy} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Starts} → {Before}

{Before} ∘ {StartedBy} → {Before}

{Before} ∘ {During} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Contains} → {Before}

{Before} ∘ {Finishes} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {FinishedBy} → {Before}

{Before} ∘ {Equals} → {Before}

{Before} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, Dur-

ing}

{Before} ∘ {Starts, StartedBy, Equals} → {Before}

{Before} ∘ {OverlappedBy, During, Finishes} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Before, Meets, Overlaps} → {Before}

{Before} ∘ {Overlaps, Contains, FinishedBy} → {Before}

{Before} ∘ {After, MetBy, OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before}

{Before} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Before} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

2) Compositions of {**After**}:

{After} ∘ {Before} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After} ∘ {After} → {After}

{After} ∘ {Meets} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {MetBy} → {After}

{After} ∘ {Overlaps} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {OverlappedBy} → {After}

{After} ∘ {Starts} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {StartedBy} → {After}

{After} ∘ {During} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {Contains} → {After}

{After} ∘ {Finishes} → {After}

{After} ∘ {FinishedBy} → {After}

{After} ∘ {Equals} → {After}

{After} ∘ {Finishes, FinishedBy, Equals} → {After}

{After} ∘ {Overlaps, Starts, During} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {OverlappedBy, StartedBy, Contains} → {After}

{After} ∘ {Before, Meets, Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {After, MetBy, OverlappedBy} → {After}

{After} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After}

{After} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{After} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

3) Compositions of {Meets}:

{Meets} ∘ {Before} → {Before}

{Meets} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Meets} ∘ {Meets} → {Before}

{Meets} ∘ {MetBy} → {Finishes, FinishedBy, Equals}

{Meets} ∘ {Overlaps} → {Before}

{Meets} ∘ {OverlappedBy} → {Overlaps, Starts, During}

{Meets} ∘ {Starts} → {Meets}

{Meets} ∘ {StartedBy} → {Meets}

{Meets} ∘ {During} → {Overlaps, Starts, During}

{Meets} ∘ {Contains} → {Before}

{Meets} ∘ {Finishes} → {Overlaps, Starts, During}

{Meets} ∘ {FinishedBy} → {Before}

{Meets} ∘ {Equals} → {Meets}

{Meets} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Meets} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Meets} ∘ {Starts, StartedBy, Equals} → {Meets}

{Meets} ∘ {OverlappedBy, During, Finishes} → {Overlaps, Starts, During}

{Meets} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, Starts, During}

{Meets} ∘ {Before, Meets, Overlaps} → {Before}

{Meets} ∘ {Overlaps, Contains, FinishedBy} → {Before}

{Meets} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Meets} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Meets} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Meets} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Meets} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before}

{Meets} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Meets} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Meets} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

4) Compositions of {MetBy}:

{MetBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{MetBy} ∘ {After} → {After}

{MetBy} ∘ {Meets} → {Starts, StartedBy, Equals}

{MetBy} ∘ {MetBy} → {After}

{MetBy} ∘ {Overlaps} → {OverlappedBy, During, Finishes}

{MetBy} ∘ {OverlappedBy} → {After}

{MetBy} ∘ {Starts} → {OverlappedBy, During, Finishes}

{MetBy} ∘ {StartedBy} → {After}

{MetBy} ∘ {During} → {OverlappedBy, During, Finishes}

{MetBy} ∘ {Contains} → {After}

{MetBy} ∘ {Finishes} → {MetBy}

{MetBy} ∘ {FinishedBy} → {MetBy}

{MetBy} ∘ {Equals} → {MetBy}

{MetBy} ∘ {Finishes, FinishedBy, Equals} → {MetBy}

{MetBy} ∘ {Overlaps, Starts, During} → {OverlappedBy, During, Finishes}

{MetBy} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{MetBy} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{MetBy} ∘ {OverlappedBy, StartedBy, Contains} → {After}

{MetBy} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{MetBy} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{MetBy} ∘ {After, MetBy, OverlappedBy} → {After}

{MetBy} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{MetBy} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{MetBy} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After}

{MetBy} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{MetBy} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{MetBy} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{MetBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

5) Compositions of {Overlaps}:

{Overlaps} ∘ {Before} → {Before}

{Overlaps} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps} ∘ {Meets} → {Before}

{Overlaps} ∘ {MetBy} → {OverlappedBy, StartedBy, Contains}

{Overlaps} ∘ {Overlaps} → {Before, Meets, Overlaps}

{Overlaps} ∘ {OverlappedBy} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {Starts} → {Overlaps}

{Overlaps} ∘ {StartedBy} → {Overlaps, Contains, FinishedBy}

{Overlaps} ∘ {During} → {Overlaps, Starts, During}

{Overlaps} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps} ∘ {Finishes} → {Overlaps, Starts, During}

{Overlaps} ∘ {FinishedBy} → {Before, Meets, Overlaps}

{Overlaps} ∘ {Equals} → {Overlaps}

{Overlaps} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Overlaps} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Overlaps} ∘ {Starts, StartedBy, Equals} → {Overlaps, Contains, FinishedBy}

{Overlaps} ∘ {OverlappedBy, During, Finishes} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps}

{Overlaps} ∘ {Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Overlaps} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

6) Compositions of {OverlappedBy}:

{OverlappedBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{OverlappedBy} ∘ {After} → {After}

{OverlappedBy} ∘ {Meets} → {Overlaps, Contains, FinishedBy}

{OverlappedBy} ∘ {MetBy} → {After}

{OverlappedBy} ∘ {Overlaps} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy}

{OverlappedBy} ∘ {Starts} → {OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {StartedBy} → {After, MetBy, OverlappedBy}

{OverlappedBy} ∘ {During} → {OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Finishes} → {OverlappedBy}

{OverlappedBy} ∘ {FinishedBy} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Equals} → {OverlappedBy}

{OverlappedBy} ∘ {Finishes, FinishedBy, Equals} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Overlaps, Starts, During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy}

{OverlappedBy} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

7) Compositions of {Starts}:

{Starts} ∘ {Before} → {Before}

{Starts} ∘ {After} → {After}

{Starts} ∘ {Meets} → {Before}

{Starts} ∘ {MetBy} → {MetBy}

{Starts} ∘ {Overlaps} → {Before, Meets, Overlaps}

{Starts} ∘ {OverlappedBy} → {OverlappedBy, During, Finishes}

{Starts} ∘ {Starts} → {Starts}

{Starts} ∘ {StartedBy} → {Starts, StartedBy, Equals}

{Starts} ∘ {During} → {During}

{Starts} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts} ∘ {Finishes} → {During}

{Starts} ∘ {FinishedBy} → {Before, Meets, Overlaps}

{Starts} ∘ {Equals} → {Starts}

{Starts} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Starts} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Starts} ∘ {Starts, StartedBy, Equals} → {Starts, StartedBy, Equals}

{Starts} ∘ {OverlappedBy, During, Finishes} → {OverlappedBy, During, Finishes}

{Starts} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps}

{Starts} ∘ {Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{Starts} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Starts} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{Starts} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

8) Compositions of {StartedBy}:

{StartedBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{StartedBy} ∘ {After} → {After}

{StartedBy} ∘ {Meets} → {Overlaps, Contains, FinishedBy}

{StartedBy} ∘ {MetBy} → {MetBy}

{StartedBy} ∘ {Overlaps} → {Overlaps, Contains, FinishedBy}

{StartedBy} ∘ {OverlappedBy} → {OverlappedBy}

{StartedBy} ○ {Starts} → {Starts, StartedBy, Equals}

{StartedBy} ○ {StartedBy} → {StartedBy}

{StartedBy} ○ {During} → {OverlappedBy, During, Finishes}

{StartedBy} ○ {Contains} → {Contains}

{StartedBy} ○ {Finishes} → {OverlappedBy}

{StartedBy} ○ {FinishedBy} → {Contains}

{StartedBy} ○ {Equals} → {StartedBy}

{StartedBy} ○ {Finishes, FinishedBy, Equals} → {OverlappedBy, StartedBy, Contains}

{StartedBy} ○ {Overlaps, Starts, During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{StartedBy} ○ {Starts, StartedBy, Equals} → {Starts, StartedBy, Equals}

{StartedBy} ○ {OverlappedBy, During, Finishes} → {OverlappedBy, During, Finishes}

{StartedBy} ○ {OverlappedBy, StartedBy, Contains} → {OverlappedBy, StartedBy, Contains}

{StartedBy} ○ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{StartedBy} ○ {Overlaps, Contains, FinishedBy} → {Overlaps, Contains, FinishedBy}

{StartedBy} ○ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy}

{StartedBy} ○ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{StartedBy} ○ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{StartedBy} ○ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{StartedBy} ○ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{StartedBy} ○ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{StartedBy} ○ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{StartedBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

9) Compositions of {During}:

{During} ∘ {Before} → {Before}

{During} ∘ {After} → {After}

{During} ∘ {Meets} → {Before}

{During} ∘ {MetBy} → {After}

{During} ∘ {Overlaps} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {Starts} → {During}

{During} ∘ {StartedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {During} → {During}

{During} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {Finishes} → {During}

{During} ∘ {FinishedBy} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {Equals} → {During}

{During} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, Dur-

ing, Finishes}

{During} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{During} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{During} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{During} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

10) Compositions of {Contains}:

{Contains} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Contains} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Meets} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {MetBy} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Overlaps} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {OverlappedBy} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Starts} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {StartedBy} → {Contains}

{Contains} ∘ {During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {Contains} → {Contains}

{Contains} ∘ {Finishes} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {FinishedBy} → {Contains}

{Contains} ∘ {Equals} → {Contains}

{Contains} ∘ {Finishes, FinishedBy, Equals} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Overlaps, Starts, During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {Starts, StartedBy, Equals} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {OverlappedBy, During, Finishes} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {OverlappedBy, StartedBy, Contains} → {OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Contains} ∘ {Overlaps, Contains, FinishedBy} → {Overlaps, Contains, FinishedBy}

{Contains} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Contains} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Contains} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Contains} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

11) Compositions of {Finishes}:

{Finishes} ∘ {Before} → {Before}

{Finishes} ∘ {After} → {After}

{Finishes} ∘ {Meets} → {Meets}

{Finishes} ∘ {MetBy} → {After}

{Finishes} ∘ {Overlaps} → {Overlaps, Starts, During}

{Finishes} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy}

{Finishes} ∘ {Starts} → {During}

{Finishes} ∘ {StartedBy} → {After, MetBy, OverlappedBy}

{Finishes} ∘ {During} → {During}

{Finishes} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes} ∘ {Finishes} → {Finishes}

{Finishes} ∘ {FinishedBy} → {Finishes, FinishedBy, Equals}

{Finishes} ∘ {Equals} → {Finishes}

{Finishes} ∘ {Finishes, FinishedBy, Equals} → {Finishes, FinishedBy, Equals}

{Finishes} ∘ {Overlaps, Starts, During} → {Overlaps, Starts, During}

{Finishes} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{Finishes} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{Finishes} ∘ {OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Starts, During}

{Finishes} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy}

{Finishes} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Finishes} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{Finishes} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes} ○ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes} ○ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes} ○ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

12) Compositions of {FinishedBy}:

{FinishedBy} ○ {Before} → {Before}

{FinishedBy} ○ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{FinishedBy} ○ {Meets} → {Meets}

{FinishedBy} ○ {MetBy} → {OverlappedBy, StartedBy, Contains}

{FinishedBy} ○ {Overlaps} → {Overlaps}

{FinishedBy} ○ {OverlappedBy} → {OverlappedBy, StartedBy, Contains}

{FinishedBy} ○ {Starts} → {Overlaps}

{FinishedBy} ○ {StartedBy} → {Contains}

{FinishedBy} ○ {During} → {Overlaps, Starts, During}

{FinishedBy} ○ {Contains} → {Contains}

{FinishedBy} ○ {Finishes} → {Finishes, FinishedBy, Equals}

{FinishedBy} ○ {FinishedBy} → {FinishedBy}

{FinishedBy} ○ {Equals} → {FinishedBy}

{FinishedBy} ○ {Finishes, FinishedBy, Equals} → {Finishes, FinishedBy, Equals}

{FinishedBy} ○ {Overlaps, Starts, During} → {Overlaps, Starts, During}

{FinishedBy} ○ {Starts, StartedBy, Equals} → {Overlaps, Contains, FinishedBy}

{FinishedBy} ○ {OverlappedBy, During, Finishes} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{FinishedBy} ○ {OverlappedBy, StartedBy, Contains} → {OverlappedBy, StartedBy, Contains}

{FinishedBy} ○ {Before, Meets, Overlaps} → {Before, Meets, Overlaps}

{FinishedBy} ○ {Overlaps, Contains, FinishedBy} → {Overlaps, Contains, FinishedBy}

{FinishedBy} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{FinishedBy} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{FinishedBy} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{FinishedBy} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{FinishedBy} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{FinishedBy} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{FinishedBy} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{FinishedBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

13) Compositions of {Equals}:

{Equals} ∘ {Before} → {Before}

{Equals} ∘ {After} → {After}

{Equals} ∘ {Meets} → {Meets}

{Equals} ∘ {MetBy} → {MetBy}

{Equals} ∘ {Overlaps} → {Overlaps}

{Equals} ∘ {OverlappedBy} → {OverlappedBy}

{Equals} ∘ {Starts} → {Starts}

{Equals} ∘ {StartedBy} → {StartedBy}

{Equals} ∘ {During} → {During}

{Equals} ∘ {Contains} → {Contains}

{Equals} ∘ {Finishes} → {Finishes}

{Equals} ∘ {FinishedBy} → {FinishedBy}

{Equals} ∘ {Equals} → {Equals}

{Equals} ∘ {Finishes, FinishedBy, Equals} → {Finishes, FinishedBy, Equals}

{Equals} ∘ {Overlaps, Starts, During} → {Overlaps, Starts, During}

{Equals} ∘ {Starts, StartedBy, Equals} → {Starts, StartedBy, Equals}

{Equals} ∘ {OverlappedBy, During, Finishes} → {OverlappedBy, During, Finishes}

{Equals} ∘ {OverlappedBy, StartedBy, Contains} → {OverlappedBy, StartedBy, Contains}

{Equals} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps}

{Equals} ∘ {Overlaps, Contains, FinishedBy} → {Overlaps, Contains, FinishedBy}

{Equals} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy}

{Equals} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Equals} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{Equals} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Equals} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Equals} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Equals} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Equals} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

14) Compositions of {Finishes, FinishedBy, Equals}:

{Finishes, FinishedBy, Equals} ∘ {Before} → {Before}

{Finishes, FinishedBy, Equals} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Meets} → {Meets}

{Finishes, FinishedBy, Equals} ∘ {MetBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Overlaps} → {Overlaps, Starts, During}

{Finishes, FinishedBy, Equals} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy,

StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Starts} → {Overlaps, Starts, During}

{Finishes, FinishedBy, Equals} ∘ {StartedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {During} → {Overlaps, Starts, During}

{Finishes, FinishedBy, Equals} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Finishes} → {Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {FinishedBy} → {Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {Equals} → {Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {Finishes, FinishedBy, Equals} → {Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {Overlaps, Starts, During} → {Overlaps, Starts, During}

{Finishes, FinishedBy, Equals} ∘ {Starts, StartedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Starts, During}

{Finishes, FinishedBy, Equals} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Contains, FinishedBy}

→ {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Finishes, FinishedBy, Equals} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

15) Compositions of {Overlaps, Starts, During}:

{Overlaps, Starts, During} ∘ {Before} → {Before}

{Overlaps, Starts, During} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps, Starts, During} ∘ {Meets} → {Before}

{Overlaps, Starts, During} ∘ {MetBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps, Starts, During} ∘ {Overlaps} → {Before, Meets, Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Starts} → {Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {StartedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {During} → {Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Finishes} → {Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {FinishedBy} → {Before, Meets, Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {Equals} → {Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets,

Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {Starts, StartedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Overlaps, Starts, During} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Starts, During} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

16) Compositions of {Starts, StartedBy, Equals}:

{Starts, StartedBy, Equals} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {After} → {After}

{Starts, StartedBy, Equals} ∘ {Meets} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {MetBy} → {MetBy}

{Starts, StartedBy, Equals} ∘ {Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {OverlappedBy} → {OverlappedBy, During, Finishes}

{Starts, StartedBy, Equals} ∘ {Starts} → {Starts, StartedBy, Equals}

{Starts, StartedBy, Equals} ∘ {StartedBy} → {Starts, StartedBy, Equals}

{Starts, StartedBy, Equals} ∘ {During} → {OverlappedBy, During, Finishes}

{Starts, StartedBy, Equals} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {Finishes} → {OverlappedBy, During, Finishes}

{Starts, StartedBy, Equals} ∘ {FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {Equals} → {Starts, StartedBy, Equals}

{Starts, StartedBy, Equals} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {Starts, StartedBy, Equals} → {Starts, StartedBy, Equals}

{Starts, StartedBy, Equals} ∘ {OverlappedBy, During, Finishes} → {OverlappedBy, During, Finishes}

{Starts, StartedBy, Equals} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes,

FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{Starts, StartedBy, Equals} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{Starts, StartedBy, Equals} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Starts, StartedBy, Equals} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Starts, StartedBy, Equals} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

17) Compositions of {OverlappedBy, During, Finishes}:

{OverlappedBy, During, Finishes} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{OverlappedBy, During, Finishes} ∘ {After} → {After}

{OverlappedBy, During, Finishes} ∘ {Meets} → {Before, Meets, Overlaps, Contains, FinishedBy}

{OverlappedBy, During, Finishes} ∘ {MetBy} → {After}

{OverlappedBy, During, Finishes} ∘ {Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {Starts} → {OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {StartedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {During} → {OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {Finishes} → {OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {FinishedBy} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {Equals} → {OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ○ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ○ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{OverlappedBy, During, Finishes} ○ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ○ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ○ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ○ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, During, Finishes} ○ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

18) Compositions of {OverlappedBy, StartedBy, Contains}:

{OverlappedBy, StartedBy, Contains} ○ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{OverlappedBy, StartedBy, Contains} ○ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ○ {Meets} → {Overlaps, Contains, FinishedBy}

{OverlappedBy, StartedBy, Contains} ○ {MetBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ○ {Overlaps} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ○ {OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ○ {Starts} → {Overlaps, OverlappedBy,

Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {StartedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Finishes} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {FinishedBy} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Equals} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Finishes, FinishedBy, Equals} → {OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Overlaps, Starts, During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {Starts, StartedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During,

Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

19) Compositions of {Before, Meets, Overlaps}:

{Before, Meets, Overlaps} ∘ {Before} → {Before}

{Before, Meets, Overlaps} ∘ {After} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Meets} → {Before}

{Before, Meets, Overlaps} ∘ {MetBy} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Overlaps} → {Before, Meets, Overlaps}

{Before, Meets, Overlaps} ∘ {OverlappedBy} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Starts} → {Before, Meets, Overlaps}

{Before, Meets, Overlaps} ∘ {StartedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps} ∘ {During} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps} ∘ {Finishes} → {Before, Meets, Overlaps, Starts, Dur-

ing}

{Before, Meets, Overlaps} ∘ {FinishedBy} → {Before, Meets, Overlaps}

{Before, Meets, Overlaps} ∘ {Equals} → {Before, Meets, Overlaps}

{Before, Meets, Overlaps} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps} ∘ {Starts, StartedBy, Equals} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps} ∘ {OverlappedBy, During, Finishes} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps}

{Before, Meets, Overlaps} ∘ {Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps} ∘ {After, MetBy, OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy,

Equals}

{Before, Meets, Overlaps} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

20) Compositions of {Overlaps, Contains, FinishedBy}:

{Overlaps, Contains, FinishedBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps, Contains, FinishedBy} ∘ {Meets} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {MetBy} → {OverlappedBy, StartedBy, Contains}

{Overlaps, Contains, FinishedBy} ∘ {Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {OverlappedBy} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Starts} → {Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {StartedBy} → {Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {Finishes} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {Equals} → {Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Starts, StartedBy, Equals} → {Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {OverlappedBy, During, Finishes} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, Contains, FinishedBy} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, Contains, FinishedBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

21) Compositions of {After, MetBy, OverlappedBy}:

{After, MetBy, OverlappedBy} ∘ {Before} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {After} → {After}

{After, MetBy, OverlappedBy} ∘ {Meets} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {MetBy} → {After}

{After, MetBy, OverlappedBy} ∘ {Overlaps} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy}

{After, MetBy, OverlappedBy} ∘ {Starts} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy} ∘ {StartedBy} → {After, MetBy, OverlappedBy}

{After, MetBy, OverlappedBy} ∘ {During} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy} ∘ {Finishes} → {After, MetBy, OverlappedBy}

{After, MetBy, OverlappedBy} ∘ {FinishedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy} ∘ {Equals} → {After, MetBy, OverlappedBy}

{After, MetBy, OverlappedBy} ∘ {Finishes, FinishedBy, Equals} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy} ∘ {Overlaps, Starts, During} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy} ∘ {OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy} ∘ {Before, Meets, Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes,

FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy}

{After, MetBy, OverlappedBy} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

22) Compositions of {Before, Meets, Overlaps, Starts, During}:

{Before, Meets, Overlaps, Starts, During} ∘ {Before} → {Before}

{Before, Meets, Overlaps, Starts, During} ∘ {After} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Meets} → {Before}

{Before, Meets, Overlaps, Starts, During} ∘ {MetBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Overlaps} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Fin-

ishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Starts} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {StartedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {During} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Finishes} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {FinishedBy} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {Equals} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {Starts, StartedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {After, MetBy, OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, Starts, During}

{Before, Meets, Overlaps, Starts, During} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Starts, During} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

23) Compositions of {After, MetBy, OverlappedBy, During, Finishes}:

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Before} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {After} → {After}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Meets} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {MetBy} → {After}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Starts} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {StartedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {During} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Overlaps, Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Starts, StartedBy, Equals} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Before, Meets, Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ∘ {Before, Meets, Overlaps,

Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ○ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, OverlappedBy, During, Finishes}

{After, MetBy, OverlappedBy, During, Finishes} ○ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ○ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ○ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ○ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, During, Finishes} ○ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

24) Compositions of {**After, MetBy, OverlappedBy, StartedBy, Contains**}:

{After, MetBy, OverlappedBy, StartedBy, Contains} ○ {Before} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ○ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ○ {Meets} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ○ {MetBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ○ {Overlaps} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ○ {OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Starts} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {StartedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {During} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Finishes} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {FinishedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Equals} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Finishes, FinishedBy, Equals} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Overlaps, Starts, During} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Starts, StartedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Overlaps, Contains, FinishedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps,

Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, OverlappedBy, StartedBy, Contains} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

25) Compositions of {Before, Meets, Overlaps, Contains, FinishedBy}:

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {After} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Meets} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {MetBy} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {OverlappedBy} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Starts} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {StartedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Contains} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Finishes} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Equals} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Starts, StartedBy, Equals} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {OverlappedBy, During, Finishes} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {OverlappedBy, StartedBy, Contains} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {After, MetBy, OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps,

Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, Contains, FinishedBy} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

26) Compositions of {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}:

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Meets} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {MetBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Starts} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {StartedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {During} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Finishes} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {FinishedBy} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Equals} → {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Starts, StartedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, OverlappedBy,

Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

27) Compositions of {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}:

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Meets} → {Before, Meets, Overlaps, Contains, FinishedBy}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {MetBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Starts} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {StartedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Finishes} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {FinishedBy} → {Before, Meets, Overlaps, Over-

lappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Finishes, FinishedBy, Equals} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Starts, StartedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

28) Compositions of {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}:

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Meets} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Fin-

ishes, FinishedBy, Equals} ∘ {MetBy} → {After, MetBy, OverlappedBy, StartedBy, Contains}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Starts} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {StartedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {During} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Starts, StartedBy, Equals} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Starts, During} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, During, Finishes} → {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, OverlappedBy, StartedBy, Contains} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, Contains, FinishedBy} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {Before, Meets, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

{After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} ∘ {After, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals} → {Before, After, Meets, MetBy, Overlaps, OverlappedBy, Starts, StartedBy, During, Contains, Finishes, FinishedBy, Equals}

Appendix C

Appendix C

Ontology Files

As already mentioned in subsection 3.2.1, Ontology Parser accepts either RDF (*.rdf*) or OWL (*.owl*) as input files. The following example describes an ontology file, in rdf format, which contains fourteen intervals ($i_0 - i_{13}$) which are connected with the basic relations of Allen:

<rdf:RDF

xmlns="http://www.intelligence.tuc.gr/Temporal-IntervalBased#"

xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

xmlns:owl="http://www.w3.org/2002/07/owl#"

xmlns:xsd="http://www.w3.org/2001/XMLSchema#"

xmlns:swrl="http://www.w3.org/2003/11/swrl#"

xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"

xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased">

<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Ontology"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object">

<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Null">

<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation">

⟨owl:propertyDisjointWith rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Null"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Before"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#After"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Meets"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#MetBy"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Overlaps"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#OverlappedBy"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Starts"⟩
⟨rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/⟩
⟨rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/⟩
⟨/rdf:Description⟩

⟨rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#StartedBy"⟩

```
<rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#During">
<rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Contains">
<rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Finishes">
<rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#FinishedBy">
<rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Equals">
<rdfs:subPropertyOf rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#AllenRelation"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/i0">
<rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/>
<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/i1">
<Before rdf:resource="http://www.intelligence.tuc.gr/i0"/>
<rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.intelligence.tuc.gr/i2">
```

‹After rdf:resource="http://www.intelligence.tuc.gr/i1"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i3"›

‹Meets rdf:resource="http://www.intelligence.tuc.gr/i2"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i4"›

‹MetBy rdf:resource="http://www.intelligence.tuc.gr/i3"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i5"›

‹Overlaps rdf:resource="http://www.intelligence.tuc.gr/i4"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i6"›

‹OverlappedBy rdf:resource="http://www.intelligence.tuc.gr/i5"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i7"›

‹Starts rdf:resource="http://www.intelligence.tuc.gr/i6"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i8"›

‹StartedBy rdf:resource="http://www.intelligence.tuc.gr/i7"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i9"›

‹During rdf:resource="http://www.intelligence.tuc.gr/i8"/›

‹rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/›

‹/rdf:Description›

‹rdf:Description rdf:about="http://www.intelligence.tuc.gr/i10"›

<Contains rdf:resource="http://www.intelligence.tuc.gr/i9"/>

<rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/i11">

<Finishes rdf:resource="http://www.intelligence.tuc.gr/i10"/>

<rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/i12">

<FinishedBy rdf:resource="http://www.intelligence.tuc.gr/i11"/>

<rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.intelligence.tuc.gr/i13">

<Equals rdf:resource="http://www.intelligence.tuc.gr/i12"/>

<rdf:type rdf:resource="http://www.intelligence.tuc.gr/Temporal-IntervalBased#Object"/>

</rdf:Description>

</rdf:RDF>

Query Files

Query Parser accepts either SPARQL (*.rq*) or exclusively temporal (*.tmp*) as input files. The following example is a SPARQL query, which returns all the triples of the network containing relation "*equals*":

PREFIX temp: <http://www.intelligence.tuc.gr/Temporal-IntervalBased#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX owl: <http://www.w3.org/2002/07/owl#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?x ?y

WHERE {

?x temp:Equals ?y

}

References

[1] Natasha Noy and Alan Rector. *"Defining N-ary Relations on the Semantic Web"*. W3C Working Group Note 12, April 2006. Available at: http://www.w3.org/TR/swbp-n-aryRelations/.

[2] Christopher A. Welty and Richard Fikes. *"A Reusable Ontology for Fluents in OWL"*. Frontiers in Artificial Intelligence and Applications, 150:226-236, 2006.

[3] Bijan Parsia and Evren Sirin. *"Pellet: An OWL DL reasoner"*. Third International Semantic Web Conference - Poster (2004).

[4] Sotirios Batsakis, Kostas Stravoskoufos, and Euripides G.M. Petrakis. *"Temporal Reasoning for Supporting Temporal Queries in OWL 2.0"*. 15th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES2011), 6881:558-567, 12-14 September 2011, Kaiserslautern, Germany. Available at: http://www.intelligence.tuc.gr/~petrakis/publications/KES2011.pdf.

[5] Peter van Beek and Robin Cohen. *"Exact and Approximate Reasoning about Temporal Relations"*. Computational intelligence, Vol 6(3), pp. 132-147, 1990.

[6] Markus Stocker and Evren Sirin. *"Pellet-Spatial: A Hybrid RCC-8 and RDF/OWL Reasoning and Query Engine"*. In CEUR Workshop Proceedings, vol. 529-OWLED 2009, pp.2-31, 2009.

[7] George Christodoulou, Euripides G.M. Petrakis, and Sotirios Batsakis. *"Qualitative Spatial Reasoning using Topological and Directional Information in OWL"*. 24th International Conference on Tools with Artificial Intelligence (ICTAI 2012), November 7-9, 2012, Athens, Greece. Available at: http://www.intelligence.tuc.gr/~petrakis/publications/ICTAI2012-CHOROS.pdf.

[8] Sotirios Batsakis and Euripides G.M. Petrakis. *"SOWL: Spatio-temporal Representation, Reasoning and Querying over the Semantic Web"*. 6th International Conference on Semantic Systems (I-

SEMANTICS 2010), 1-3 September 2010, Graz, Austria. Available at: http://www.intelligence.tuc.gr/˜petrakis/ publications/I-SEM10.pdf.

[9] James F. Allen. *"Maintaining Knowledge About Temporal Intervals"*. Communications of the ACM, 26:832-843, 1983.

[10] Bernhard Nebel and Hans-Jurgen Burckert. *"Reasoning about Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra"*. Journal of the ACM (JACM), Vol.42(1), pages:43-66, 1995.

[11] Jochen Renz and Bernhard Nebel. *"Qualitative Spatial Reasoning using Constraint Calculi"*. In Handbook of Spatial Logics, Springer, Netherlands , pp. 161-215.

[12] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. *"SWRL: A Semantic Web Rule Language Combining OWL and RuleML"*. W3C Member Submission 21 May 2004. Available at: http://www.w3.org/Submission/SWRL/.