Algebraic Tableau Reasoning for the Description Logic $\mathcal{SHOQ}$

Jocelyne Faddoul and Volker Haarslev

Department of Computer Science and Software Engineering, Concordia University, 1455 de Maisonneuve Blvd. W. Montreal, Quebec, H3G 1M8, Canada
E-mail: {j_faddou,haarslev}@cse.concordia.ca

Abstract

Semantic web applications based on the web ontology language (OWL) often require the use of numbers in class descriptions for expressing cardinality restrictions on properties or even classes. Some of these cardinalities are specified explicitly but quite a few are entailed and need to be discovered by reasoning procedures. Due to the description logic (DL) foundation of OWL those reasoning services are offered by DL reasoners which employ reasoning procedures that are arithmetically uninformed and substitute arithmetic reasoning by "don't know" non-determinism in order to cover all possible cases. This lack of information about arithmetic problems dramatically degrades the performance of DL reasoners in many cases, especially with ontologies relying on the use of nominals ($\mathcal{O}$) and qualified cardinality restrictions ($\mathcal{Q}$). In this article we present a new algebraic tableau reasoning procedure for the DL $\mathcal{SHOQ}$ that combines tableau procedures and algebraic methods, namely linear integer programming, to ensure arithmetically better informed reasoning procedures. $\mathcal{SHOQ}$ extends the standard DL $\mathcal{ALC}$ (which is equivalent to the multi-modal logic K_m) with transitive roles, role hierarchies, qualified cardinality restrictions, and nominals, and forms an expressive subset of the web ontology language OWL 2. Although the proposed algebraic tableau (in analogy to standard tableau) is still double exponential in the worst case, it deals with cardinalities in a very informed way due to its arithmetic component and can be considered as a novel foundation for informed reasoning procedures addressing cardinality restrictions.

Keywords: Description Logics, Tableau Reasoning, Algebraic Reasoning

1 Introduction

Description Logics (DLs) [3] are a family of knowledge representation formalisms used to represent and reason about an application's domain elements. DLs are distinguished by their terminological orientation, their well defined logic-based semantics, and their deductive inference capabilities. They are interestingly applicable in the semantic web as they provide a basis for the Web Ontology Language (OWL). In this article we focus on the DL $\mathcal{SHOQ}$, which extends the standard DL $\mathcal{ALC}$ with transitive roles ($\mathcal{S}$), role hierarchies ($\mathcal{H}$), nominals ($\mathcal{O}$) and qualified cardinality restrictions ($\mathcal{Q}$). Nominals are named individuals studied in the area of DLs [1,27,4] as well as in the area of hybrid logics [5]. In DLs, nominals play an important role as they allow one to express the notion of uniqueness and identity; nominals must be interpreted as singleton sets.

It is known that DLs offering *nominals* and *qualified cardinality restrictions (QCRs)* enjoy additional expressive power. There exist no other way in $\mathcal{SHOQ}$ to close a concept or domain with a finite number of elements except using nominals, which can also emulate

concept cardinalities [2] (as was shown in [28]). QCRs have become part of OWL 2 [22] as they are needed in many ontologies [15] especially in the medical domain.

In order to illustrate the interaction between QCRs and nominals let us consider the following two axioms expressed in standard DL notation (see Section 2.1 for more details on syntax and semantics).

$$EU\_MemberState \equiv Austria \sqcup \ldots \sqcup UK \tag{1}$$

$$Future\_EU \sqsubseteq \geq 30\, MemberOf.EU\_MemberState \tag{2}$$

The notion of a concept (class, unary predicate) is used to denote a set of individuals with common characteristics (*EU_MemberState*, *Future_EU*), and the notion of a role (object property, binary predicate) is used to denote a binary relationship between individuals (*MemberOf*). Concepts and roles are the building blocks of a DL language that also comes equipped with a set of operators ($\sqcap, \sqcup, \ldots$); complex concepts and roles can be built from atomic ones by the application of available operators. Axiom (1) defines the concept *EU_MemberState* by enumerating all 27 EU member states as nominals (*Austria*, ..., *UK*). Let us additionally assume that all the 27 nominals are mutually disjoint.

Axiom (2) states as necessary condition for the concept *Future_EU* that a future EU must have at least 30 member states (using the operator $\geq$, the role *MemberOf*, and the qualifying concept *EU_MemberState*). This is an example for a QCR specifying a lower (≥ 30) bound on the number of elements related via the role *MemberOf* with additionally specifying qualities on the related elements. A typical inference service answered by a DL reasoner would be to test whether the concept *Future_EU* is satisfiable (see Section 2.1 for a definition of a satisfiable concept).

Informally speaking, it turns out that axiom (1) and the disjointness of all 27 nominals express an implicit cardinality restriction of 27 for the concept *EU_MemberState*. However, axiom (2) states that *Future_EU* needs to be related to at least 30 different individuals of *EU_MemberState*, which is not possible because the cardinality of *EU_MemberState* is restricted to 27. So, it is easy to see that *EU_MemberState* is not satisfiable. [1]

An ontology containing the axioms 1 and 2, and the disjointness declaration for the 27 nominals was modeled and tested with Protege 4.0 [2], and neither of the highly optimized tableau reasoners FaCT++ [3] or Pellet [4] nor the highly optimized hypertableau reasoner HermiT [5] could decide the satisfiability of *EU_MemberState* in this small OWL ontology within 2 hours of CPU time (using a PC with an AMD 64*2 Dual Core Processor 5200, 2.70 GHZ and 3GB of RAM). Most reasoning procedures employed by standard DL reasoners deal with the problem to decide whether *Future_EU* is satisfiable in a very inefficient way because they blindly try to satisfy the numerical restrictions implied by such a concept description. For instance, in the case of *Future_EU*, a standard tableau algorithm creates 30 different but anonymous individuals of *EU_MemberState* and then non-deterministically tries to merge them with the 27 nominals until all possibilities are exhausted and the unsatisfiability of *Future_EU* is returned. The overall fact that 27 nominals can never be distributed over 30 different individuals is lost.

[1] The disjointness of the nominals is not even needed to make *EU_MemberState* unsatisfiable.
[2] http://www.co-ode.org/downloads/protege-x/
[3] http://owl.man.ac.uk/factplusplus/, version 1.2
[4] http://pellet.owldl.org/, version 2.0
[5] http://www.hermit-reasoner.com/, version 1.1

There do not exist many tableau approaches that address reasoning about nominals or QCRs in a more informed way. First performance improvements with QCRs and algebraic reasoning have been reported for the DL $\mathcal{SHQ}$ in [13,14] and more recently in [12]. Decision procedures for expressive DLs enabling both nominals and QCRs were published in [18] with very weak optimizations if any (no DL reasoner was able to classify the WINE [6] ontology until recent efforts [26]). The optimization techniques for nominals proposed in [26] do not address the interaction between nominals and QCRs. Another challenge is that ontologies that use nominals no longer enjoy the quasi-tree model property which has always been advantageous for tableau. Recent efforts in [21] address inefficient reasoning due to the creation of large tableau models and the presence of nominals. Other major inefficiency sources can be (i) the high degree of non-determinism introduced by the use of GCIs or when merging of domain elements is necessary, (ii) the construction of large pre-models, or (iii) the interaction between constructors. A high worst case complexity (NEXPTIME) occurs when *nominals* interact with *inverse roles* ($\mathcal{I}$) and *QCRs*. Each of these constructs alone is challenging to reason with and needs special optimization techniques. Resolution-based reasoning procedures were proposed in [20] and were proven to be weak in dealing with large numbers in QCRs. Hypertableaux [23] were recently studied to minimize non-determinism in DL reasoning with no special treatment for QCRs. To the best of our knowledge no arithmetically informed approaches have been reported for ontologies that rely on the use and interaction of both nominals and QCRs.

In this paper, we propose a hybrid calculus extending the one reported in [10] such that at-least and at-most cardinality restrictions (and their interaction) on nominals and roles are represented as a system of linear inequations. The solvability of such a system can be decided by standard linear integer programming algorithms. In the case of our *Future_EU* example the corresponding system of linear inequations can easily be recognized as unsolvable (see also Section 8.6).

This article extends our work in [10] on $\mathcal{ALCOQ}$ to include GCIs, transitive roles and role hierarchies and demonstrates how a standard tableau reasoning algorithm for DLs can be extended with an arithmetic component while maintaining soundness, completeness and termination. The result is a hybrid reasoning algorithm which is more informed about arithmetic constraints imposed by concept descriptions. In particular, a better handling of numerical restrictions implied by concept operators such as nominals and QCRs is ensured. Such a reasoning algorithm can form a basis to provide more efficient reasoning support for ontologies using nominals or QCRs. A naïve implementation of our approach is by no means better than any other naïve implementation of DL reasoning algorithms proposed so far. However, the algebraic method enjoys an additional level of information and properties that make the calculus amenable and well suited for optimization techniques that address the sources of complexity related to nominals and QCRs.

The contributions of this paper can be summarized as follows.

(i) The numerical restrictions imposed by nominals are handled by algebraic reasoning. The key intuition to consider the interaction between different numerical restrictions is that one can think of a model as sets of individuals. Numerical restrictions on these sets can be handled together using a linear inequation solver.

(ii) By setting the sum of the cardinalities of the sets of nominals and role fillers as an ob-

[6] http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine.rdf

jective function to be minimized, one can ensure that a minimum number of nominals and role fillers satisfies all relevant at-least/at-most restrictions.

(iii) One can use a proxy individual as a representative of individuals satisfying common restrictions. Allowing the re-use of proxy individuals also helps in reducing the number of nodes in a completion graph (see Section 6 for the notion of completion graphs).

(iv) One can extend a tableau reasoning algorithm with an algebraic component and still maintain soundness, completeness, and termination.

2 Preliminaries

We introduce the syntax, semantics and inference problems of $\mathcal{SHOQ}$ which is the DL obtained by extending the basic DL $\mathcal{ALC}$ with transitive roles (leading to the DL $\mathcal{S}$), role hierarchies ($\mathcal{H}$), nominals ($\mathcal{O}$), and qualified cardinality restrictions (QCRs) ($\mathcal{Q}$). We also illustrate the preprocessing needed to combine the algebraic method with tableau algorithms.

2.1 Syntax and Semantics

Let N_C, N_R be non-empty and disjoint sets of concept names and roles respectively. Let $N_o \subseteq N_C$ be the set of nominals, and $N_{R^+} \subseteq N_R$ the set of transitive roles.

A $\mathcal{SHOQ}$ RBox $\mathcal{R}$ is a finite set of *role inclusion axioms* (RIAs) of the form $R \sqsubseteq S$, where R, S are roles in N_R and R is called a *sub-role* of S while S is called a *super-role* of R. We define $\sqsubseteq_*$ as the reflexive transitive closure of $\sqsubseteq$ on $\mathcal{R}$, and $R \equiv_* S$ as an abbreviation for $R \sqsubseteq_* S$ and $S \sqsubseteq_* R$ in $\mathcal{R}$. A role R is called *simple* if it is neither transitive nor has a transitive sub-role S ($S \sqsubseteq_* R$), see [16] for a formal definition of *simple* roles.

A $\mathcal{SHOQ}$ TBox $\mathcal{T}$ is a finite set of *general concept inclusion axioms* (GCIs) of the form $C \sqsubseteq D$, where C, D are $\mathcal{SHOQ}$ concepts, and $C \equiv D$ abbreviates $\{C \sqsubseteq D,\ D \sqsubseteq C\}$. The set of $\mathcal{SHOQ}$ concepts is the smallest set such that: (i) every concept name $A \in N_C$ is a concept, and (ii) if C, D are concepts, R is a role in N_R, and S is a *simple* role in N_R, then $\neg C$, $(C \sqcup D)$, $(C \sqcap D)$, $(\exists R.C)$, $(\forall R.C)$, $(\geq nS.C)$, $(\leq nS.C)$ with $n \in \mathbb{N}$ are also concepts. In the following we use $\top$ ($\bot$) as an abbreviation for $A \sqcup \neg A$ ($A \sqcap \neg A$) and $\geq nS$ ($\leq nS$) for $\geq nS.\top$ ($\leq nS.\top$). We do not consider descriptions of the form $\exists R.C$ as they can be converted to $\geq 1R.C$ without imposing the simple role restriction and $\leq 0R.C$ as they can be converted to $\forall R.\neg C$.

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of $\Delta^{\mathcal{I}}$, a non-empty set of individuals, called the domain of the interpretation, and $\cdot^{\mathcal{I}}$, an interpretation function. The interpretation function $\cdot^{\mathcal{I}}$ maps atomic concepts $A \in N_C$ to subsets of $\Delta^{\mathcal{I}}$, roles $R \in N_R$ to subsets of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and for a transitive role $R \in N_{R^+}$ it holds that $R^{\mathcal{I}} = (R^{\mathcal{I}})^+$.

Using # to denote the cardinality of a set, we define the set of *R-fillers* of an individual $s \in \Delta^{\mathcal{I}}$ given a role $R \in N_R$ as $FIL(R, s) = \{t \in \Delta^{\mathcal{I}} \mid \langle s, t\rangle \in R^{\mathcal{I}}\}$ and the set of all *R-fillers* as: $FIL(R) = \bigcup_{s \in \Delta^{\mathcal{I}}} FIL(R, s)$. Given a $\mathcal{SHOQ}$ concept, the following must hold.

$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$, $\#o^{\mathcal{I}} = 1$ for all $o \in N_o$

$(\forall R.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \,|\, for\ all\ t \in \Delta^{\mathcal{I}}, if\ \langle s,t\rangle \in R^{\mathcal{I}}\ then\ t \in C^{\mathcal{I}}\}$

$(\geq nS.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \,|\, the\ set\ of\ S\text{-}fillers\ of\ s\ satisfies\ \#(FIL(S,s) \cap C^{\mathcal{I}}) \geq n\}$

$(\leq nS.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \,|\, the\ set\ of\ S\text{-}fillers\ of\ s\ satisfies\ \#(FIL(S,s) \cap C^{\mathcal{I}}) \leq n\}$

We denote a $\mathcal{SHOQ}$ knowledge base (KB) consisting of a TBox $\mathcal{T}$ and an RBox $\mathcal{R}$ by KB$(\mathcal{T},\mathcal{R})$. The TBox $\mathcal{T}$ is said to be consistent iff there exists an interpretation $\mathcal{I}$ satisfying $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for each $C \sqsubseteq D \in \mathcal{T}$. Such an $\mathcal{I}$ is called a model of $\mathcal{T}$. An interpretation $\mathcal{I}$ satisfies an RBox $\mathcal{R}$ iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ is satisfied for each $R \sqsubseteq S \in \mathcal{R}$ and $\mathcal{I}$ is said to be a model of $\mathcal{R}$. A KB$(\mathcal{T},\mathcal{R})$ is said to be consistent iff there exists a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{R}$.

A concept C is said to be satisfiable w.r.t. KB$(\mathcal{T},\mathcal{R})$ iff there exists a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{R}$ with $C^{\mathcal{I}} \neq \emptyset$, i.e., there exists an individual $s \in \Delta^{\mathcal{I}}$ as an instance of C, $s \in C^{\mathcal{I}}$. $\mathcal{I}$ is called a model of C w.r.t. $\mathcal{R}$ and $\mathcal{T}$.

A $\mathcal{SHOQ}$ ABox $\mathcal{A}$ is a finite set of *concept membership assertions* of the form $a:C$ or *role membership assertions* of the form $(a,b):R$ with a,b two individual names. An Abox $\mathcal{A}$ is said to be consistent w.r.t. KB$(\mathcal{T},\mathcal{R})$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{R}$ such that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ is satisfied for each $a:C$ in $\mathcal{A}$ and $(a^{\mathcal{I}},b^{\mathcal{I}}) \in R^{\mathcal{I}}$ is also satisfied for each $(a,b):R$ in $\mathcal{A}$. Using nominals, concept satisfiability and ABox consistency can be reduced to KB consistency; a concept C is satisfiable w.r.t. KB$(\mathcal{T},\mathcal{R})$ iff KB$(\mathcal{T} \cup \{o \sqsubseteq C\},\mathcal{R})$ is consistent and $o \in N_{\mathrm{o}}$ new in $\mathcal{T}$, an ABox $\mathcal{A}$ is consistent w.r.t. KB$(\mathcal{T},\mathcal{R})$ iff KB$((\mathcal{T} \cup \bigcup_{(a:C)\in\mathcal{A}}\{a \sqsubseteq C\} \cup \bigcup_{((a,b):R)\in\mathcal{A}}\{a \sqsubseteq \exists R.b\}),\mathcal{R})$ is consistent. Hence, without loss of generality we restrict our attention to KB consistency in this article. In the following, we assume all concepts to be in their *negation normal form* (NNF), i.e., negation appears in front of concept names only. We use $\dot{\neg}C$ to denote the NNF of $\neg C$ and $nnf(C)$ to denote the NNF of C.

2.2 *Preprocessing*

When checking a KB$(\mathcal{T},\mathcal{R})$ consistency, the concept axioms in $\mathcal{T}$ can be reduced to a single axiom $\top \sqsubseteq C_{\mathcal{T}}$ such that $C_{\mathcal{T}}$ abbreviates $\sqcap_{C\sqsubseteq D\in\mathcal{T}} nnf(\neg C \sqcup D)$ [18]. A TBox consistency test can be checked by testing the consistency of $o \sqsubseteq C_{\mathcal{T}}$ with $o \in N_{\mathrm{o}}$ new in $\mathcal{T}$, which means that at least $o^{\mathcal{I}} \in {C_{\mathcal{T}}}^{\mathcal{I}}$ and ${C_{\mathcal{T}}}^{\mathcal{I}} \neq \emptyset$. Moreover, since $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ then every domain element must also satisfy $C_{\mathcal{T}}$ (every domain element is a member of $C_{\mathcal{T}}$).

In order to allow the applicability of the algebraic method, concept expressions occurring in $C_{\mathcal{T}}$ are rewritten according to Algorithm 1 similarly to [25,8,12,11], which allows a separation between numerical restrictions and their qualifications. Algorithm 1 also allows a bookkeeping of negated *qualifying concepts* in their preprocessed NNF into the set $\mathcal{Q}_{\mathrm{C}}^{\neg}$.

Definition 2.1 [Qualifying concept] A qualifying concept D is a concept used to impose a qualification, D, on the set of R-fillers for some role $R \in N_R$. We define $\mathcal{Q}_{\mathrm{C}}(R) = \{D \,|\, \forall S.D$ occurs in $C_{\mathcal{T}}$ with $R \sqsubseteq_* S \in \mathcal{R}\}$ as the set of qualifying concepts for $R \in N_{\mathrm{R}}$, and $\mathcal{Q}_{\mathrm{C}} = \bigcup_{R\in N_{\mathrm{R}}} \mathcal{Q}_{\mathrm{C}}(R)$ as the set of qualifying concepts in $C_{\mathcal{T}}$. We also define $\mathcal{Q}_{\mathrm{C}}^{\neg} = \{\dot{\neg}D \,|\, D \in \mathcal{Q}_{\mathrm{C}}\}$ as the set of negated qualifying concepts in their NNF. A mapping $\dot{\neg}_Q$ is maintained between $\mathcal{Q}_{\mathrm{C}}$ and $\mathcal{Q}_{\mathrm{C}}^{\neg}$ such that given a qualifying concept $D \in \mathcal{Q}_{\mathrm{C}}$, $\dot{\neg}_Q(D) = \dot{\neg}D$ with $\dot{\neg}D \in \mathcal{Q}_{\mathrm{C}}^{\neg}$.

Given $C_{\mathcal{T}}$, a set N_{R} of roles, $\mathcal{R}$ a set of RIAs, and $\mathcal{Q}_{\mathrm{C}}$ a set of qualifying concepts, we define a new concept operator $\forall_{\backslash}$, the *role-set difference* operator, used for descriptions like $\forall(R\backslash S).D$ such that R,S in N_{R} and D a $\mathcal{SHOQ}$ concept. The $\forall_{\backslash}$ operator is based on

set semantics such that given an interpretation $\mathcal{I}$, then $(\forall(R\backslash S).D)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \langle s,t\rangle \in R^{\mathcal{I}} \wedge \langle s,t\rangle \notin S^{\mathcal{I}} \Rightarrow t \in D^{\mathcal{I}}\}$ is satisfied. Let $\mathcal{SHON}_{\mathcal{R}\backslash}$ denote the DL $\mathcal{SHO}$ extended with unqualified cardinality restrictions ($\mathcal{N}$) and the role-set difference operator ($_{\mathcal{R}\backslash}$).

Algorithm 1 is applied to $C_{\mathcal{T}}$ such that $rw(C_{\mathcal{T}}, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg})$ returns an equivalent concept expression ($C'_{\mathcal{T}}$) in the DL $\mathcal{SHON}_{\mathcal{R}\backslash}$. $Q_{\mathrm{C}}^{\neg}$ is initially empty and is extended with the preprocessed form of $\doteq\!\neg C$ every time rw is applied to a concept of the form $\forall R.C$. This bookkeeping of the preprocessed form of $\doteq\!\neg C$ is required because $\mathcal{SHON}_{\mathcal{R}\backslash}$ is not closed under negation. For example, with $\forall R.(\forall S.C)$ the negation of the qualifying concept for R, $\doteq\!\neg(\forall S.C) = \geq 1S.\neg C$, is not in $\mathcal{SHON}_{\mathcal{R}\backslash}$. However after applying rw to $\forall R.(\forall S.C)$ we have $\geq 1S_1 \sqcap \forall S_1.\neg C$ added to $Q_{\mathrm{C}}^{\neg}$ such that $\doteq\!\neg_Q(\forall S.C) = \geq 1S_1 \sqcap \forall S_1.\neg C$ which is in $\mathcal{SHON}_{\mathcal{R}\backslash}$. Also, N_{R} and $\mathcal{R}$ are extended with a fresh role R' new in $\mathcal{T}$ every time the conditions in lines 7 and 8 are applicable. It can be shown that Lemma 2.2 holds.

Lemma 2.2 (Preserving Satisfiability) *Rewriting $C_{\mathcal{T}}$ according to Algorithm 1 preserves satisfiability. Satisfying $C_{\mathcal{T}}$ w.r.t. $\mathcal{R}$ consists of satisfying $C'_{\mathcal{T}}$ w.r.t. $\mathcal{R}$.*

Proof. The proof found in [10] for the DL $\mathcal{ALCOQ}$ also holds for the DL $\mathcal{SHOQ}$. Since transitive roles are not used in at-least restrictions $\geq nR.C$ with $n > 1$, nor in at most restrictions with $n > 0$, preserving the satisfiability of $rw(\geq 1R.C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg})$ is an easy consequence of the proof in [10]. With the DL $\mathcal{ALCOQ}$ the rewriting starts with an empty set $\mathcal{R}$ of RIAs that is extended every time cases 7 and 8 are applicable. Which means that after the first time case 7 or 8 had been applied, $\mathcal{R}$ is no longer empty and the conditions for (rw) as presented in this article are met. Hence, the proofs are also applicable. □

Algorithm 1 *rw*: Given $\mathcal{SHOQ}$ concepts $A \in N_{\mathrm{C}}$, C, D, and $R \in N_{\mathrm{R}}$, $\mathcal{R}$ the set of RIAs, and $Q_{\mathrm{C}}^{\neg}$ the set of negated qualifying concepts the following rewriting holds:

1: $rw(A, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow A$
2: $rw(\neg A, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow \neg A$
3: $rw((C \sqcap D), N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow (rw(C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \sqcap rw(D, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}))$
4: $rw((C \sqcup D), N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow (rw(C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \sqcup rw(D, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}))$
5: $rw((\neg C), N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow rw(\doteq\!\neg C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg})$
6: $rw(\forall R.C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow \forall R.rw(C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg} \cup rw(\doteq\!\neg C, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}))$
7: $rw((\geq nR.C), N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow (\geq nR' \sqcap \forall R'.rw(C, N_{\mathrm{R}} \cup \{R'\}, \mathcal{R} \cup \{R' \sqsubseteq R\}, Q_{\mathrm{C}}^{\neg}))$
//same with $\geq nR.\top$
8: $rw((\leq nR.C), N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \longrightarrow (\leq nR' \sqcap \forall R'.rw(C, N_{\mathrm{R}} \cup \{R'\}, \mathcal{R}, Q_{\mathrm{C}}^{\neg}) \sqcap \forall(R\backslash R').$
$rw(\doteq\!\neg C, N_{\mathrm{R}} \cup \{R'\}, \mathcal{R} \cup \{R' \sqsubseteq R\}, Q_{\mathrm{C}}^{\neg}))$ *//same with $\leq nR.\top$*

Example 2.3 Applying rw to $(\geq 1\,R.(o \sqcap \leq 1\,R.o))$ with $N_R = \{R\}$ and $\mathcal{R} = \emptyset$, and $Q_{\mathrm{C}}^{\neg} = \emptyset$ gives the concept expression (3) with $\mathcal{R} = \{R_1 \sqsubseteq R,\ R_2 \sqsubseteq R\}$, $Q_{\mathrm{C}}^{\neg} = \emptyset$ and $N_R = \{R, R_1, R_2\}$.

$$\geq 1R_1 \sqcap \forall R_1.(o \sqcap \leq 1R_2 \sqcap \forall R_2.o \sqcap \forall R\backslash R_2.\neg o) \tag{3}$$

Example 2.4 Let the TBox $\mathcal{T}$ consist of the axioms (2) and (1) in Section 1 then

$$C_{\mathcal{T}} = \begin{pmatrix} (\neg EU\_MemberState \sqcup Austria \sqcup \ldots \sqcup UK) \sqcap \\ ((\neg Austria \sqcap \ldots \sqcap \neg UK) \sqcup EU\_MemberState) \sqcap \\ (\neg Future\_EU \sqcup \geq 30\, MemberOf.EU\_MemberState) \end{pmatrix}$$

Additionally we have $N_{\mathrm{R}} = \{MemberOf\}$, $\mathcal{R} = \emptyset$, $Q_{\mathrm{C}}^{\neg} = \emptyset$ and $rw(C_{\mathcal{T}}, N_{\mathrm{R}}, \mathcal{R}, Q_{\mathrm{C}}^{\neg})$ extends N_{R} to $N_{\mathrm{R}} = \{M', MemberOf\}$, and $\mathcal{R}$ to $\mathcal{R} = \{M' \sqsubseteq MemberOf\}$, and $C_{\mathcal{T}}$ to:

$$C'_{\mathcal{T}} = \begin{pmatrix} (\neg EU\_MemberState \sqcup Austria \sqcup \ldots \sqcup UK) \sqcap \\ ((\neg Austria \sqcap \ldots \sqcap \neg UK) \sqcup EU\_MemberState) \sqcap \\ (\neg Future\_EU \sqcup \geq 30\, M' \sqcap \forall M'.EU\_MemberState) \end{pmatrix}$$

3 Algebraic Tableau Reasoning and $\mathcal{SHOQ}$

Extending our work in [8] to reach the expressiveness of $\mathcal{SHOQ}$ (with GCIs) is not straight forward, each added language element brings its own challenges.

3.1 Nominals impose global numerical restrictions

The numerical restrictions imposed by nominals ($\mathcal{O}$) are global restrictions that affect domain elements as a whole. These restrictions could interact with the numerical restrictions imposed by QCRs as it is the case with the definition of *Future_EU* in axiom (2), which implies a numerical restriction that is local to *MemberOf*-fillers, and the definition of *EU_MemberState* in axiom (1), which implies a numerical restriction that is global and affects all elements in the domain. This means that applying the algebraic method locally to each individual as in [8,12] can no longer ensure soundness, the algebraic reasoner may satisfy local numerical restrictions imposed by QCRs without necessarily satisfying the global ones imposed by nominals. We addressed this challenge in [11], where we apply algebraic reasoning globally, which means that the system of linear inequations captures and resolves the numerical restrictions imposed by QCRs as well as those imposed by nominals to decide $\mathcal{ALCOQ}$ concept satisfiability. We do this by extending the *atomic decomposition technique* [25], which is a key technique to encode numerical restrictions into inequations, to consider the sets of nominals and their interaction with the sets of role fillers.

3.2 Transitive roles and GCIs introduce cycles

With $\mathcal{ALCOQ}$ and unfoldable TBoxes, the algebraic tableau algorithm in [10] terminates naturally because when a concept label is expanded using completion rules, the introduced concept labels are of size smaller than the original concept $C_{\mathcal{T}}$. Eventually this leads to a natural halt when labels can no longer be expanded. When transitive roles and/or GCIs are allowed, as is the case with the DL $\mathcal{SHOQ}$, one must keep in mind that termination of the algorithm may no longer be naturally handled because completion rules can repeat concept labels through nodes. For example, when we have the concept description (4) in the label of a node x and R is a transitive role then completion rules introduce a node y having the

same label as x. One might think that an algorithm needs to implement blocking strategies and disallow completion rules on a node y repeating the label of an existing node x, as it is done in traditional tableau algorithms [16] for DLs with transitive roles and GCIs. We show in this article how the cases of repeated labels are handled with our re-use strategy.

$$A \sqcap \geq 1R \sqcap \forall R.A \sqcap \forall R.(\geq 1R \sqcap \forall R.A) \tag{4}$$

3.3 GCIs and role hierarchies encapsulate qualifications on role fillers

Example 3.1 Assuming we want to test the satisfiability of the concept (5) w.r.t. the TBox $\mathcal{T}$ given in (6).

$$\geq 1S_1.(A \sqcap B_1) \sqcap \geq 1S_2.(A \sqcap B_2) \tag{5}$$

$$\mathcal{T} = \{A \sqsubseteq \geq 1R.B,\ B_1 \sqsubseteq \forall R.C,\ B_2 \sqsubseteq \forall R.\neg C\} \tag{6}$$

The numerical restriction $(\geq 1R.B)$ encapsulated in A is common to S_1-fillers and S_2-fillers which both require an R-filler being a member of B. On the other hand, S_1-fillers and S_2-fillers have different qualifying concepts for their R-fillers due to the axioms for concepts B_1, B_2. S_1-fillers which are members of B_1 must have R-fillers being members of C, and S_2-fillers which are members of B_2 must have R-fillers being members of $\neg C$. To check the satisfiability of the concept (5) we add the axiom (7) to $\mathcal{T}$ with $o \in N_o$ a nominal new in $\mathcal{T}$.

$$o \sqsubseteq \geq 1S_1.(A \sqcap B_1) \sqcap \geq 1S_2.(A \sqcap B_2) \tag{7}$$

In principle, when preprocessing a KB by applying the rewriting algorithm, one has two choices: Case (1) or Case (2). When the TBox is unfoldable one can opt for case (1) and otherwise one has to consider case (2).

- Case (1): Unfolding $\mathcal{T}$ would make all $(\geq nR.C)$ restrictions explicit as in [10], and $C_{\mathcal{T}}$ is of the form: $C_{\mathcal{T}} = \{\neg o \sqcup (\geq 1S_1.(\geq 1R.B \sqcap \forall R.C) \sqcap \geq 1S_2.(\geq 1R.B \sqcap \forall R.\neg C))\}$.

  A distinction can be made between R-fillers of S_1-fillers and those of S_2-fillers because *rw* uses a different role for each occurrence of $\geq 1R.B$. Rewriting $C_{\mathcal{T}}$ gives

  $$C'_{\mathcal{T}} = \begin{pmatrix} \neg o \sqcup (\geq 1S_{11} \sqcap \forall S_{11}.(\geq 1R_1 \sqcap \forall R_1.B \sqcap \forall R.C) \sqcap \\ \geq 1S_{21} \sqcap \forall S_{21}.(\geq 1R_2 \sqcap \forall R_2.B \sqcap \forall R.\neg C)) \end{pmatrix}$$

  In this case, the algebraic method will automatically consider the cases when R-fillers have different qualifications due to R_1 and R_2 which are sub-roles of R.
- Case (2): When the TBox is not unfolded or cannot be unfolded then $C_{\mathcal{T}}$ is of the form

  $$C_{\mathcal{T}} = \begin{pmatrix} (\neg A \sqcup \geq 1R.B) \sqcap (\neg B_1 \sqcup \forall R.C) \sqcap (\neg B_2 \sqcup \forall R.\neg C) \sqcap \\ (\neg o \sqcup (\geq 1S_1.(A \sqcap B_1) \sqcap \geq 1S_2.(A \sqcap B_2))) \end{pmatrix}$$

  Rewriting $C_{\mathcal{T}}$ gives

  $$C'_{\mathcal{T}} = \begin{pmatrix} (\neg A \sqcup \geq 1R_1 \sqcap \forall R_1.B) \sqcap (\neg B_1 \sqcup \forall R.C) \sqcap (\neg B_2 \sqcup \forall R.\neg C) \sqcap \\ (\neg o \sqcup (\geq 1S_{11} \sqcap \forall S_{11}.(A \sqcap B_1) \sqcap \geq 1S_{21} \sqcap \forall S_{21}.(A \sqcap B_2))) \end{pmatrix}$$

  In this case, the qualifications differentiating R-fillers are still encapsulated in B_1 and B_2; S_{11}-fillers and S_{21}-fillers have different qualifying concepts for their R_1-fillers. R_1-fillers of S_{11}-fillers must also be members of C and this is encapsulated in B_1, and R_1-fillers of S_{21}-fillers must be members of $\neg C$ and this is encapsulated in B_2.

These cases lead to the following conclusion. When algebraic reasoning is applied locally for each domain element as in [8,12] one does not need to distinguish the cases when role fillers have different qualifying concepts because the algebraic reasoning is done locally and separately for each domain element even if role fillers for different domain elements end up having common restrictions. In the presence of nominals, the algebraic reasoning can no longer be done locally and separately. The approach used in [10,11] to handle global algebraic reasoning in the presence of nominals still cannot deal with encapsulated qualifications in the case when a TBox cannot be unfolded. These encapsulated qualifications could also be inherent due to a role hierarchy or role transitivity, and if not taken into consideration make the algebraic method incomplete. In the next section, we show how we can address the challenge of encapsulated qualifications by allowing the atomic decomposition to consider the qualifications on roles using qualifying concepts.

4 A Tableau for the DL $\mathcal{SHON}_{\mathcal{R}\setminus}$

Before illustrating the algebraic method, we define a tableau for the DL $\mathcal{SHON}_{\mathcal{R}\setminus}$. It is important to note that $\mathcal{SHON}_{\mathcal{R}\setminus}$ is not closed under negation due to the preprocessing step described in Algorithm 7 but this does not cause a problem because our proposed calculus never negates a preprocessed concept [7] .

We define *clos(C)*, for a given concept description C (in the DL $\mathcal{SHON}_{\mathcal{R}\setminus}$), to be the smallest set of concepts such that: (a) $C \in clos(C)$, (b) if $A \in N_R$ and $A \in clos(C)$ then $\neg A \in clos(C)$, (c) if $(E \sqcap D)$ or $(E \sqcup D) \in clos(C)$ then $E, D \in clos(C)$, (d) if $\forall R.D$ or $\forall R\backslash S.D \in clos(C)$ then $D \in clos(C)$. The size of $clos(C)$ is bounded by the size of C. The set of relevant sub-concepts of a TBox $\mathcal{T}$ is then defined as $clos(\mathcal{T}) = clos(C'_{\mathcal{T}})$.

Definition 4.1 [Tableau] Given a $\mathcal{SHOQ}$ KB$(\mathcal{T},\mathcal{R})$ which has been preprocessed into a $\mathcal{SHON}_{\mathcal{R}\setminus}$ KB$(\mathcal{T},\mathcal{R})$, we define a tableau T for $(\mathcal{T},\mathcal{R})$ as T = $(\mathbf{S},\mathcal{L},\mathcal{E})$ as an abstraction of a model with $\mathbf{S}$ a non-empty set of individuals, $\mathcal{L} : \mathbf{S} \rightarrow 2^{clos(\mathcal{T})}$ a mapping between each individual and a set of concepts, and $\mathcal{E}$: $N_{\mathrm{R}} \rightarrow 2^{\mathrm{S}\times\mathrm{S}}$ a mapping between each role and a set of pairs of individuals in $\mathbf{S}$. For all $s,t \in \mathbf{S}$, $A \in N_{\mathrm{C}}$, $C, D \in clos(\mathcal{T})$, $o \in N_{\mathrm{o}}$, $R, S \in N_{\mathrm{R}}$, and given the definition $R^T(s) = \{t \in \mathbf{S} \mid \langle s,t\rangle \in \mathcal{E}(R)\}$, properties (i) - (xi) must always hold:

(i) $C'_{\mathcal{T}} \in \mathcal{L}(s)$

(ii) If $A \in \mathcal{L}$(s) then $\neg A \notin \mathcal{L}$(s).

(iii) If $C \sqcap D \in \mathcal{L}$(s) then $C \in \mathcal{L}(s)$ and $D \in \mathcal{L}(s)$.

(iv) If $C \sqcup D \in \mathcal{L}$(s) then $C \in \mathcal{L}(s)$ or $D \in \mathcal{L}(s)$.

(v) If $\forall S.C \in \mathcal{L}$(s) and $\langle s,t\rangle \in \mathcal{E}(S)$ then $C \in \mathcal{L}(t)$.

(vi) If $\forall S.C \in \mathcal{L}$(s) and $\langle s,t\rangle \in \mathcal{E}(R)$ with $R \sqsubseteq_* S$ and R is transitive then $\forall R.C \in \mathcal{L}(t)$.

(vii) If $\forall(R\backslash S).C \in \mathcal{L}$(s) and $\langle s,t\rangle \in \mathcal{E}(R)$, and $\langle s,t\rangle \notin \mathcal{E}(S)$ then $C \in \mathcal{L}(t)$.

(viii) If $(\geq nR) \in \mathcal{L}$(s) then $\#R^T(s) \geq n$.

(ix) If $(\leq mR) \in \mathcal{L}$(s) then $\#R^T(s) \leq m$.

(x) If $\langle s,t\rangle \in \mathcal{E}(R)$ and $R \sqsubseteq_* S \in \mathcal{R}$, then $\langle s,t\rangle \in \mathcal{E}(S)$.

(xi) For each $o \in N_{\mathrm{o}}, \#\{s \in \mathbf{S} \mid o \in \mathcal{L}(s)\} = 1$.

[7] The negations of qualifying concepts are computed using $\dot{\neg}_Q$ which returns $\mathcal{SHON}_{\mathcal{R}\setminus}$ concepts (See Section 2.2).

Lemma 4.2 *A $\mathcal{SHOQ}$ knowledge base KB($\mathcal{T},\mathcal{R}$) is consistent iff there exists a tableau T for ($\mathcal{T},\mathcal{R}$).*

Proof. The proof is similar to the one found in [10]. Property (vi) ensures that the qualification restrictions due to role transitivity are enforced while taking into consideration role hierarchies. Property (x) ensures that the role hierarchy expressed in $\mathcal{R}$ is preserved. □

5 The Algebraic Method

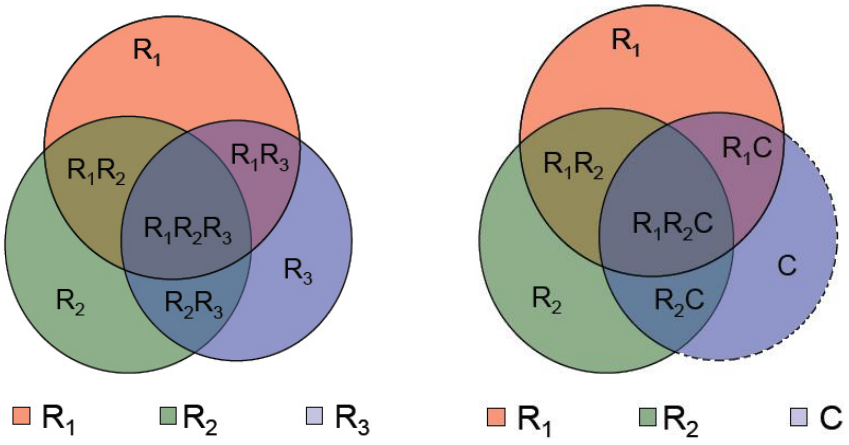
The algebraic reasoning approach for set description languages including DL was first introduced in [24] and later in [25] where it was investigated how a concept satisfiability check can be reduced to a pure inequation solving problem. The DL operators discussed handle only the expressiveness of $\mathcal{ALCQ}$ with empty TBoxes and no formal calculus was proposed until the recent efforts in [8,12,6]. In [8,12,6] the algebraic method is combined with tableau reasoning; it is applied locally and separately for each domain element to reduce the satisfiability of concept descriptions using QCRs into inequation solving. In [10,11] the algebraic method is combined with tableau reasoning and is applied globally to reduce the satisfiability of concept descriptions using QCRs and/or nominals into inequation solving. A key technique to enable the algebraic method is the atomic decomposition [24] which allows the decomposition of a set of elements into mutually disjoints subsets. We illustrate how this technique can enable the algebraic method for the DL $\mathcal{SHON}_{\mathcal{R}\setminus}$ with GCIs by using the appropriate *decomposition set*. Unlike the other approaches, the *decomposition set* includes roles, qualifying concepts and nominals.

5.1 The Atomic Decomposition

Let $H(R)$ denote the set of all sub-roles of R: $H(R) = \{R' \mid R' \sqsubseteq_* R,\ R' \neq R\}$. For technical reasons we do not add R to $H(R)$ since R is a super-role for elements in $H(R)$ and R does not occur in QCRs anymore after preprocessing. For every role $R' \in H(R)$, the set of *R'-fillers* forms a subset of the set of *R-fillers* ($FIL(R') \subseteq FIL(R)$). We define $\overline{R'}$ to be the complement of R' relative to $H(R)$, the set of *$\overline{R'}$-fillers* is then defined as $FIL(\overline{R'}) = (FIL(R) \setminus FIL(R'))$.

In order to distinguish the cases when role fillers have different qualifications, as was discussed in Example 3.1, the atomic decomposition must also consider when $FIL(R)$ intersects with the interpretation of a qualifying concept. For this purpose, we use the set of qualifying concepts of R, $Q_C(R)$ as defined in Definition 2.1. Since $D \in Q_C(R)$ could be a complex expression or a nominal, and for ease of presentation, we assign a unique qualification name q for each $D \in Q_C(R)$. Let $\mathcal{Q}_N$ be the set of all qualification names assigned. We maintain a mapping between qualification names and their corresponding concept expressions using a bijection $\theta : \mathcal{Q}_N \rightarrow \mathcal{Q}_C$; in case a nominal $o \in N_o$ has been used as a qualifying concept expression then o is also used as the qualification name and $\theta(o) = o$. Let $Q_N(R)$ denote the set of qualification names for a role ($R \in N_R$) then $Q_N(R)$ is defined as $Q_N(R) = \{q \in \mathcal{Q}_N \mid \theta(q) \in Q_C(R)\}$.

Definition 5.1 [Decomposition Set] Let $\mathcal{D}_R = H(R) \cup Q_N(R)$ define the decomposition set for R-fillers. $\mathcal{D}_R$ is a decomposition set since each subset P of $\mathcal{D}_R$ ($P \subseteq \mathcal{D}_R$) defines a unique set of roles and/or qualification names that admits an interpretation $P^{\mathcal{I}}$ corresponding to the unique intersection of role fillers and interpretation of qualifying concepts for



(a) $\mathcal{D}_R = \{R_1, R_2, R_3\}$. (b) $\mathcal{D}_R = \{R_1, R_2, \theta^{-}(C)\}$.

Fig. 1. Atomic Decomposition of $\mathcal{D}_R$ (assuming that $\theta^{-}(C) = C$)

the roles and qualification names in P: $P^{\mathcal{I}} = \bigcap_{\mathrm{R}' \in P \cap H(R)} FIL(R') \cap \bigcap_{\mathrm{R}'' \in (H(R) \setminus P)} FIL(\overline{R''}) \cap \bigcap_{p \in P \cap Q_{\mathrm{N}}} \theta(p)^{\mathcal{I}} \cap \bigcap_{q \in (Q_{\mathrm{N}} \setminus P)} (\dot{\neg}_Q \theta(q))^{\mathcal{I}}$. $P^{\mathcal{I}}$ cannot overlap with role fillers for roles that do not appear in P since it is assumed to overlap with their complement. Similarly, in the case when $Q_{\mathrm{N}}(R) \neq \emptyset$, $P^{\mathcal{I}}$ cannot overlap with the interpretation of a qualifying concept whose corresponding qualification name is not in P because it overlaps with the interpretation of its complement. This makes all $P^{\mathcal{I}}$ disjoint as in [25] and the set of all $P \subseteq D_R$ defines a partitioning of D_R.

Example 5.2 If we have a decomposition set $\mathcal{D}_R = \{R_1, R_2, R_3\}$ with $H(R) = \{R_1, R_2, R_3\}$ and $Q_{\mathrm{N}}(R) = \emptyset$ and the decomposition is as shown in Fig.1(a), then if $P_1 = \{R_1, R_2\}$ and $P_2 = \{R_2, R_3\}$ this means that P_1 is the partition name for $FIL(R_1) \cap FIL(R_2) \cap FIL(\overline{R_3})$ which is equal to $P_1^{\mathcal{I}}$ and P_2 is the partition name for $FIL(R_2) \cap FIL(R_3) \cap FIL(\overline{R_1})$, and therefore, although $P_1 \cap P_2 = \{R_2\}$ we have $P_1^{\mathcal{I}} \cap P_2^{\mathcal{I}} = \emptyset$.

Since $\mathcal{SHON}_{\mathcal{R}\setminus}$ does not allow $\geq nR$ or $\leq nR$ concept expressions using role complements, no role complement will be explicitly used. For ease of presentation, we do not list the role complements in a partition name. Also, since a qualification is not applicable unless there exists a corresponding role filler, we do not consider a partition P, $P \subseteq Q_{\mathrm{N}}(R)$, if P includes a qualification name without including a role. For example, Fig. 1(b) shows the decomposition of $\mathcal{D}_R = \{R_1, R_2, \theta^{-}(C)\}$ and the dashed part corresponds to the partition $P = \{\theta^{-}(C)\}$ which does not need to be considered. [8]

Interaction with Nominals For each nominal $o \in N_{\mathrm{o}}$, $o^{\mathcal{I}}$ can interact with *R-fillers* for some R in N_{R} such that ($o^{\mathcal{I}} \subseteq FIL(R)$) as is the case with Example 2.3. Also the same nominal o can interact with *R-fillers* and *S-fillers* for $R, S \in N_{\mathrm{R}}$ such that R, S do not necessarily share sub-roles or super-roles in $\mathcal{R}$. This means that *R-fillers* and *S-fillers* could interact with each other due to their common interaction with the same nominal o. These interactions lead to the following definition of a *Global Decomposition Set* (GDS).

Definition 5.3 [Global Decomposition Set] We define the set of all roles, qualifications and nominals occurring in $C'_{\mathcal{T}}$ as $\mathcal{DS} = (\bigcup_{R \in N_{\mathrm{R}}} \mathcal{D}_R \cup N_{\mathrm{o}}) \setminus \{\dot{\neg}C \mid \{C, \dot{\neg}C\} \subseteq Q_{\mathrm{C}}\}$. When

[8] For sake of simplicity we assume that $\theta^{-}(C) = C$.

C and $\dot{\neg}C$ are both used as qualifying concepts, we only include C in $\mathcal{DS}$. Applying the decomposition technique on $\mathcal{DS}$ defines a *global partitioning* of domain elements.

Definition 5.4 [Global Partitioning] We define a global partitioning on domain elements as follows. Let $\mathcal{P}$ be the set of the disjoint partition names defined for the decomposition of $\mathcal{DS}$: $\mathcal{P} = \{P \mid P \subseteq \mathcal{DS}\}$. Then $\mathcal{P}^{\mathcal{I}} = \Delta^{\mathcal{I}}$ because it includes all possible domain elements which correspond to a nominal and/or a role filler; $\mathcal{P}^{\mathcal{I}} = \bigcup_{P \subseteq \mathcal{DS}} P^{\mathcal{I}}$.

5.2 Partitions are Signatures

A given model $\mathcal{I}$ of a TBox $\mathcal{T}$ consists of domain elements grouped into mutually disjoint partitions. Each partition represents a signature of concept descriptions that is common to all elements in the partition. A model $\mathcal{I}$ of $\mathcal{T}$ satisfies a signature $F \subseteq clos(\mathcal{T})$ iff $F^{\mathcal{I}} \neq \emptyset$ with $F^{\mathcal{I}} = \bigcap_{E \in F} E^{\mathcal{I}}$.

Lemma 5.5 *Given a model $\mathcal{I}$ of $\mathcal{T}$, for each non-empty partition $p^{\mathcal{I}} \subseteq \mathcal{P}^{\mathcal{I}}$, and two domain elements $i, j \in p^{\mathcal{I}}$, if $i \in F^{\mathcal{I}}$ (F is the signature of p) then: (1) $j \in F^{\mathcal{I}}$ and, (2) there exists no other domain element $i' \in \Delta^{\mathcal{I}}$ such that $i' \in p^{\mathcal{I}} \cap F^{\mathcal{I}} \cap p'^{\mathcal{I}}$ for some partition $p'^{\mathcal{I}} \subseteq \mathcal{P}^{\mathcal{I}}$ different from $p^{\mathcal{I}}$.*

Proof. It is easy to prove (2) since all partitions are disjoint by definition. For (1), given $R_1, \ldots R_n \in N_{\mathrm{R}}$, $o_1, \ldots, o_n \in N_{\mathrm{o}}$, $q_1 \ldots q_n \in Q_{\mathrm{N}}$, and $i, j \in \Delta^{\mathcal{I}}$ we consider Cases 1-5.

- Case 1 - Nominals partition: $p^{\mathcal{I}}$ is a nominals partition, then it corresponds to some partition name $p \in \mathcal{P}$ of the form $p = \{o_1, \ldots, o_n\}$ and individuals in $p^{\mathcal{I}}$ satisfy the signature F such that $F^{\mathcal{I}} = p^{\mathcal{I}} = (o_1^{\mathcal{I}} \cap \ldots \cap o_n^{\mathcal{I}})$. Given the nominals semantics, $i \in F^{\mathcal{I}}$ and if there exists $j \in p^{\mathcal{I}}$ then $j \in F^{\mathcal{I}}$ since $i = j$; there can only be one element in $p^{\mathcal{I}}$.
- Case 2 - Role fillers partition: $p^{\mathcal{I}}$ is a role fillers partition, then it corresponds to some partition name $p \in \mathcal{P}$ of the form $p = \{R_1, \ldots, R_n\}$ and individuals in $p^{\mathcal{I}}$ satisfy $p^{\mathcal{I}} = (FIL(R_1) \cap \ldots \cap FIL(R_n))$. If $i, j \in p^{\mathcal{I}}$ then $i, j \in (FIL(R_1) \cap \ldots \cap FIL(R_n))$; assume $i \notin (FIL(R_1) \cap \ldots \cap FIL(R_n))$ then i is a nominal or an R_x-filler for some $x > n$. However i cannot be a nominal since $p \cap N_{\mathrm{o}} = \emptyset$. Without loss of generality, assuming $i \in FIL(R_1)$ but $i \notin (FIL(R_2) \cap \ldots \cap FIL(R_n))$ this means that i belongs to a partition $p'^{\mathcal{I}}$ corresponding to some partition name $p' \in \mathcal{P}$ such that $R_1 \in p'$ and $\{R_2, \ldots, R_n\} \nsubseteq p'$. Now we have p' different from p with $i \in (p \cap p')$, this is a contradiction since partitions are disjoint. Therefore, $i \in (FIL(R_1) \cap \ldots \cap FIL(R_n))$, and by analogy we prove that $j \in (FIL(R_1) \cap \ldots \cap FIL(R_n))$. Therefore both i and j must satisfy the signature F such that $F^{\mathcal{I}} = \cap_{\forall R_1.C_1 \in \mathcal{T}} C_1^{\mathcal{I}} \ldots \cap_{\forall R_n.C_n \in \mathcal{T}} C_n^{\mathcal{I}}$.
- Case 3 - Role fillers with qualifications partition: $p^{\mathcal{I}}$ is a role fillers partition with qualifications, then it corresponds to some partition name $p \in \mathcal{P}$ of the form $p = \{q_k, R_l\}$ for some k, l, $1 \leq k, l \leq n$, and individuals in $p^{\mathcal{I}}$ satisfy $p^{\mathcal{I}} = (\bigcap_{1 \leq k \leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q \in (Q_{\mathrm{N}} \setminus \{q_1, \ldots, q_k\})} \dot{\neg}_Q \theta(q)^{\mathcal{I}} \cap \bigcap_{1 \leq l \leq n} FIL(R_l))$. If $i, j \in p^{\mathcal{I}}$ then $i, j \in (FIL(R_1) \cap \ldots \cap FIL(R_n)) \cap \bigcap_{1 \leq k \leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q \in (Q_{\mathrm{N}} \setminus \{q_1, \ldots, q_k\})} \dot{\neg}_Q \theta(q)^{\mathcal{I}}$. Similar to Case 2, we can prove that $i, j \in (FIL(R_1) \cap \ldots \cap FIL(R_n))$ and i cannot be a nominal since $p \cap N_{\mathrm{o}} = \emptyset$. Now we need to prove that $i, j \in \bigcap_{1 \leq k \leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q \in (Q_{\mathrm{N}} \setminus \{q_1, \ldots, q_k\})} \dot{\neg}_Q \theta(q)^{\mathcal{I}}$.

  Let us assume that $i \notin (\bigcap_{1 \leq k \leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q \in (Q_{\mathrm{N}} \setminus \{q_1, \ldots, q_k\})} \dot{\neg}_Q \theta(q)^{\mathcal{I}})$ and without loss of generality, let $i \in \theta(q_1)^{\mathcal{I}}$ but $i \notin (\theta(q_2)^{\mathcal{I}} \cap \ldots \cap \theta(q_n)^{\mathcal{I}})$ this means that i belongs to a partition $p'^{\mathcal{I}}$ corresponding to some partition name $p' \in \mathcal{P}$ such that $q_1 \in p'$ and $\{q_2, \ldots, q_n\} \nsubseteq$

p'. Now we have p' different from p with $i \in (p \cap p')$, this is a contradiction since partitions are disjoint. Therefore, $i \in (\bigcap_{1\leq k\leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q\in(\mathcal{Q}_{\mathrm{N}}\setminus\{q_1,\ldots,q_k\})} \doteq_{\mathcal{Q}} \theta(q)^{\mathcal{I}})$, and by analogy we prove that $j \in (\bigcap_{1\leq k\leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q\in(\mathcal{Q}_{\mathrm{N}}\setminus\{q_1,\ldots,q_k\})} \doteq_{\mathcal{Q}} \theta(q)^{\mathcal{I}})$. Hence, both i and j must satisfy the signature F such that $F^{\mathcal{I}} = \cap_{\forall R_1.C_1\in\mathcal{T}} C_1^{\mathcal{I}} \ldots \cap_{\forall R_n.C_n\in\mathcal{T}} C_n^{\mathcal{I}} \cap (\bigcap_{1\leq k\leq n} \theta(q_k)^{\mathcal{I}} \cap \bigcap_{q\in(\mathcal{Q}_{\mathrm{N}}\setminus\{q_1,\ldots,q_k\})} \doteq_{\mathcal{Q}} \theta(q)^{\mathcal{I}})$.

- Case 4 - Nominals and role fillers partition: $p^{\mathcal{I}}$ is a role filler partition of nominals, then it corresponds to some partition name $p \in \mathcal{P}$ of the form $p = \{o_k, R_l\}$ for some k, l, $1 \leq k, l \leq n$, and individuals in $p^{\mathcal{I}}$ satisfy $p^{\mathcal{I}} = (\bigcap_{1\leq k\leq n} o_k^{\mathcal{I}} \cap \bigcap_{1\leq l\leq n} FIL(R_l))$. Given the nominals semantics and similarly to case 1 if there exists $i, j \in p^{\mathcal{I}}$ then $i = j$. The signature F for $p^{\mathcal{I}}$ is such that it satisfies $F^{\mathcal{I}} = \bigcap_{1\leq k\leq n} o_k^{\mathcal{I}} \cap \bigcap_{\forall R_1.C\in\mathcal{T}} C^{\mathcal{I}}$.
- Case 5 - Nominals and role fillers partition with qualifications: this case can be reduced to case 4 where additionally nominals satisfy the qualifications.

We do not consider the cases when a partition is for individuals with qualifications without being role fillers since these cases do not occur. A qualification is only applicable on a role filler as defined by the semantics of the language. □

6 The Algebraic Tableau Algorithm for $\mathcal{SHON}_{\mathcal{R}\setminus}$

In this section, we describe an algebraic tableau algorithm which decides the existence of a tableau for a $\mathcal{SHON}_{\mathcal{R}\setminus}$ TBox $\mathcal{T}$. Our algorithm is hybrid because it relies on tableau completion rules working together with an inequation solver to construct a tableau as an abstraction of a model of $\mathcal{T}$. Tableau completion rules work in such a way to (1) decide the satisfiability of concept descriptions that use propositional operators ($\sqcap, \sqcup, \neg$) and $\forall$, $\forall_{\setminus}$ operators, (2) encode numerical restrictions on nominals, role fillers, and their qualifications into a set of inequations processed by an inequation solver, and (3) make sure that a numerical solution satisfies logical restrictions by constructing a pre-model of the solution. The pre-model is represented using a *compressed completion graph* (CCG).

Definition 6.1 [Compressed Completion Graph] The *compressed completion graph* (CCG) is different from the "so-called" completion graphs used in standard tableau algorithms for $\mathcal{SHOQ}$ [16] and is defined as follows.

- A (CCG) is a directed graph $G = (V, E, \mathcal{L}, \mathcal{L}_{\mathrm{E}}, \mathcal{L}_{\mathrm{P}})$. Where nodes represent domain elements and the edges between the nodes represent role relations. Each node $x \in V$ is labeled with three labels: $\mathcal{L}(x)$, $\mathcal{L}_E(x)$ and $\mathcal{L}_P(x)$, and each edge $\langle x, y\rangle \in E$ is labeled with a set, $\mathcal{L}(\langle x, y\rangle) \subseteq N_{\mathrm{R}}$, of role names. $\mathcal{L}(x)$ denotes a set of concept expressions, $\mathcal{L}(x) \subseteq clos(\mathcal{T})$, that the domain element, i_x, represented by x must satisfy. $\mathcal{L}_P(x)$ denotes a partition name and is used as a tag for x based on the partition that i_x belongs to. A partition name includes a role name, a nominal or a qualification name $\mathcal{L}_P(x) \subseteq \mathcal{DS}$.
  - When a role $R \in N_{\mathrm{R}}$ appears in $\mathcal{L}_P(x)$ this means that i_x belongs to the partition for *R-fillers* and can therefore be used as an *R-filler*. When an R-filler is needed for a node y, x is checked as a candidate (see *e*-Rule).
  - When a nominal $o \in N_{\mathrm{o}}$ appears in $\mathcal{L}_P(x)$ this means that $i_x \in o^{\mathcal{I}}$, and o is added to $\mathcal{L}(x)$ when x is created. On the other hand if a nominal $i \in N_{\mathrm{o}}$ does not appear in $\mathcal{L}_P(x)$ this means that i_x satisfies the complement of i, $i_x \in (\neg i)^{\mathcal{I}}$ and $(\neg i)$ is added to $\mathcal{L}(x)$ when x is created (see *fil*-Rule).

· When a qualification name $q \in \mathcal{Q}_{\mathrm{N}}$ appears in $\mathcal{L}_P(x)$ this means that i_x satisfies the qualifying concept mapped to q, $i_x \in \theta(q)^{\mathcal{I}}$ and $\theta(q)$ is added to $\mathcal{L}(x)$ when x is created. As with the nominals case, if a qualification name $p \in \mathcal{Q}_{\mathrm{N}}$ does not appear in $\mathcal{L}_P(x)$ this means that i_x satisfies the complement of the qualifying concept mapped to p, $i_x \in \dot{\neg}_Q(\theta(p))^{\mathcal{I}}$ and $\dot{\neg}_Q\theta(p)$ is added to $\mathcal{L}(x)$ when x is created (see *fil*-Rule).

$\mathcal{L}_{\mathrm{E}}(x)$ denotes a set ξ_x of inequations that must have a non-negative integer solution. The set ξ_x is the encoding of number restrictions, qualifications and nominals (as defined in Section 6.1) that must be satisfied for x. In order to make sure that numerical restrictions local for a node x are satisfied while the global restrictions carried with nominals are not violated, the inequation solver collects all inequations and variable assignment in $\mathcal{L}_{\mathrm{E}}$ before returning a distribution. This makes sure that an initial distribution of nominals and/or role fillers is globally preserved while still satisfying the numerical restrictions (a distribution of role fillers) at each level.

- There is no distinction between nodes having a nominal in their label and other nodes.
- The CCG relies on the use of proxy nodes (see Definition 6.2) as representatives for individuals of same restrictions. Proxy nodes were first introduced in [13].
- Using $\mathcal{L}_P(x)$ as a tagging allows for the re-use of existing nodes instead of creating new ones. For example if the roles R, S appear in $\mathcal{L}_P(x)$ then x can be used as an R-filler and then re-used as an S-filler or vice versa.
- No blocking strategies are implemented and no merging of existing nodes is possible. Termination is a natural consequence of the re-use of nodes.
- An inequation solver collects and checks the satisfiability of numerical restrictions.

Definition 6.2 [Proxy node] A proxy node is a representative for the elements of each partition. Proxy nodes can be used due to Lemma 6.3 since partitions are disjoint and all elements within a partition P satisfy P's signature.

6.1 The Algebraic Reasoning

Given a partitioning $\mathcal{P}$ for the decomposition set $\mathcal{DS}$ for $\mathcal{T}$ (see Def. 5.3 and 5.4), one can reduce a conjunction of $(\geq nR)$ and $(\leq mR)$ in $\mathcal{L}(x)$ to a set of inequations and check their satisfiability using an inequation solver based on the following principles.

P0: Mapping Cardinalities to Variables. We assign a variable name v for each partition name P such that v can be mapped to a non-negative integer value n using $\sigma : \mathcal{V} \to \mathbb{N}$ such that $\sigma(v)$ denotes the cardinality of $P^{\mathcal{I}}$. Let $\mathcal{V}$ be the set of all variable names and α: $\mathcal{V} \to \mathcal{P}$ be a one-to-one mapping between each partition name $P \in \mathcal{P}$ and a variable $v \in \mathcal{V}$ such that $\alpha(v) = P$, and if a non-negative integer n is assigned to v using σ then $\sigma(v) = n = \#P^{\mathcal{I}}$. Given $L \subseteq \mathcal{DS}$, let V_L denote the set of variable names mapped to partitions satisfying $L^{\mathcal{I}}$, V_L is defined as

$$V_L = \begin{pmatrix} \{v \in \mathcal{V} \mid p \in \alpha(v) \textit{ for each } p \in (L \cap N_{\mathrm{R}})\} \cap \\ \{v \in \mathcal{V} \mid oq \in \alpha(v) \textit{ for each } oq \in (L \cap (N_o \cup \mathcal{Q}_{\mathrm{N}}))\} \cap \\ \{v \in \mathcal{V} \mid oq \notin \alpha(v) \textit{ for each } \neg oq \in L,\ oq \in (N_o \cup \mathcal{Q}_{\mathrm{N}}))\} \end{pmatrix}$$

P1: Encoding Number Restrictions, Qualifications and Nominals Into Inequations. Since the partitions in $\mathcal{P}$ are mutually disjoint and the cardinality function is ad-

ditive one can encode a cardinality restriction on a partition's elements using ξ such that $\xi(L, \geq, n) = \sum_{v \in V_L} \sigma(v) \geq n$, and $\xi(L, \leq, m) = \sum_{v \in V_L} \sigma(v) \leq m$ where $L \subseteq \mathcal{DS}$. Hence, a lower (upper) bound on the cardinality of the set of domain elements distributed over the partitions in $\mathcal{P}$ can be encoded into inequations as follows:

- (i) Bounds on role fillers: concepts of the form $(\geq nR)$ and $(\leq mR)$ in the label of a node x express lower and upper bounds n and m, respectively, on the cardinality of the set $FIL(R, i_x)$ for some $R \in N_{\mathrm{R}}$. These bounds can be reduced into inequations using $\xi(L, \geq, n)$ and $\xi(L, \leq, m)$ for $L = \{R\}$ or $L = \{R, q\}$, if additionally, we have $\forall S.C$ such that $(R \sqsubseteq_* S)$ with $C \in \mathcal{DS}$ and $\theta(q) = C$. Assuming v_p is mapped to a partition $p \subseteq \mathcal{P}$ in Example 2.3, the bounds on $FIL(R_1, i_x)$ such that $\mathcal{L}(x) = \{\geq 1R_1, \forall R_1.(o \sqcap \leq 1R_2 \sqcap \forall R_2.o \sqcap \forall R \backslash R_2.\neg o)\}$ are encoded into inequation (8) and those on $FIL(R_2)$ for a node y such that $\mathcal{L}(y) = \{o, \leq 1R_2, \forall R_2.o, \forall R \backslash R_2.\neg o\}$ are encoded into inequation (9).
- (ii) Bounds imposed by nominals: Nominals carry cardinality restrictions; they not only name individuals but also allow for counting individuals. Therefore, the cardinality of a partition with a nominal o can only be equal to 1 based on the nominals semantics; $\#o^{\mathcal{I}} = 1$. This bound on the cardinality of the nominals partitions can be encoded into inequations using $\xi(\{o\}, \geq, 1)$ and $\xi(\{o\}, \leq, 1)$ for each nominal $o \in N_{\mathrm{o}}$. In the case of Example (2.3) then the nominals semantics is encoded into inequations (10) and (11).

$$v_{\{\mathrm{R}_1\}} + v_{\{\mathrm{R}_1,\mathrm{o}\}} + v_{\{\mathrm{R}_1,\mathrm{R}_2\}} + v_{\{\mathrm{R}_1,\mathrm{R}_2,\mathrm{o}\}} \geq 1 \tag{8}$$

$$v_{\{\mathrm{R}_2,\mathrm{o}\}} + v_{\{\mathrm{R}_1,\mathrm{R}_2,\mathrm{o}\}} \leq 1 \tag{9}$$

$$v_{\{\mathrm{o}\}} + v_{\{\mathrm{R}_1,\mathrm{o}\}} + v_{\{\mathrm{R}_2,\mathrm{o}\}} + v_{\{\mathrm{R}_1,\mathrm{R}_2,\mathrm{o}\}} \geq 1 \tag{10}$$

$$v_{\{\mathrm{o}\}} + v_{\{\mathrm{R}_1,\mathrm{o}\}} + v_{\{\mathrm{R}_2,\mathrm{o}\}} + v_{\{\mathrm{R}_1,\mathrm{R}_2,\mathrm{o}\}} \leq 1 \tag{11}$$

When the nominals semantics is encoded into inequations together with the bounds on role fillers, the interaction between nominals and role fillers is handled while preserving that there is one individual for each $o \in N_{\mathrm{o}}$: $\#o^{\mathcal{I}} = 1$.

P2: Getting a Solution. Given a set ξ_x of inequations in $\mathcal{L}_E(x)$, an integer solution defines the mapping σ for each variable v occurring in ξ_x to a non-negative integer n denoting the cardinality of the corresponding partition. For example, assuming $\sigma(v_{\{R_1,R_2\}}) = 1$ and $\alpha(v_{\{R_1,R_2\}}) = \{R_1, R_2\}$, this means that the corresponding partition $(\alpha(v_{\{R_1,R_2\}}))^{\mathcal{I}}$ must have 1 element; $\#(FIL(R_1) \cap FIL(R_2)) = 1$. Additionally, by setting the objective function to minimize the sum of all variables, a minimum number of role fillers is ensured at each level. σ then defines a distribution of individuals that is consistent with the numerical restrictions encoded in ξ_x and the hierarchy expressed in $\mathcal{R}$.

Lemma 6.3 (Using a Proxy Individual) *Given a graph G as a representation of a model $\mathcal{I}$ for a TBox $\mathcal{T}$. Let P be a non-empty partition in $\mathcal{P}^{\mathcal{I}}$ and n a non-negative integer assigned by the inequation solver such that $n = \#P$. It is sufficient to create one proxy node in G as a representative of the n individuals in P.*

Proof. Lemma 6.3 is an easy consequence of Lemma 5.5. Creating a proxy node x for P in G allows to test the satisfiability of P's signature (see Section 5.2). If x satisfies the signature, then m elements can also satisfy it and m is decided by the inequation solver. x cannot violate cardinality bounds on role fillers and nominals since these bounds are numerically satisfied by the inequation solver. However, if x does not satisfy the signature of P due to a clash, this means that P must be empty because its signature is unsatisfiable.□

6.2 *Deciding KB Consistency*

Let $\mathcal{T}$ be a preprocessed TBox rewritten into $\mathcal{T} = \{\top \sqsubseteq C'_{\mathcal{T}}\}$ with $C'_{\mathcal{T}} = rw(C_{\mathcal{T}}, N_{\mathrm{R}}, \mathcal{R}, Q^{\neg}_{\mathrm{C}})$ and let $\mathcal{P}$ be the corresponding global partitioning. To decide the consistency of $\mathcal{T}$ we need to test the consistency of $C'_{\mathcal{T}}$ using $i \in N_{\mathrm{o}}$ new in $\mathcal{T}$ such that $i^{\mathcal{I}} \in {C'_{\mathcal{T}}}^{\mathcal{I}}$ and every new individual satisfies $C'_{\mathcal{T}}$. The algorithm starts with the CCG $G = (\{r_0\}, \emptyset, \emptyset, \mathcal{L}_{\mathrm{E}}, \emptyset)$. With $\mathcal{L}_{\mathrm{E}}(r_o) = \bigcup_{o \in N_{\mathrm{o}}} \{\xi(\{o\}, \leq, 1), \xi(\{o\}, \geq, 1)\}$ which is an encoding of the nominal semantics. The node r_0 is artificial and is not considered as part of the pre-model, it is only used to process the numerical restrictions on nominals using the inequation solver which returns a distribution for them.

The distribution of nominals (solution) is processed by the *fil*-Rule (see Fig. 3) which non-deterministically initializes the individual nodes for nominals. After at least one nominal is created, G is expanded by applying the completion rules given in Figures 2 and 3 until no more rules are applicable or when a clash occurs. No clash triggers or rules other then the *fil*-Rule apply to r_o. When no rules are applicable or there is a clash, a CCG is said to be *complete*.

Definition 6.4 [Clash] A node x in $(V \setminus \{r_0\})$ is said to contain a *clash* if:

- (i) $\{C, \neg C\} \subseteq \mathcal{L}(x)$, or
- (ii) a subset of inequations $\xi_x \subseteq \mathcal{L}_{\mathrm{E}}(x)$ does not admit a non-negative integer solution.

When G is complete and there is no clash, this means that the numerical as well as the logical restrictions are satisfied (${C'_{\mathcal{T}}}^{\mathcal{I}} \neq \emptyset$) and there exists a pre-model for $\mathcal{T}$: the algorithm returns that $\mathcal{T}$ is consistent. Otherwise the algorithm returns that $\mathcal{T}$ is inconsistent.

6.3 *Strategy of Rule Application*

Given a node x in the CCG, the completion rules in Figures 2 and 3 are applicable based on the following priorities:

- **Priority 1:** $\sqcap$-*Rule*, $\sqcup$-*Rule*, $\forall$-*Rule*, $\forall_+$-*Rule*, *ch-Rule*, $\bowtie$-*Rule*, *e-Rule*.
- **Priority 2:** *fil-Rule*.
- **Priority 3:** $\forall_{\setminus}$-*Rule*.

The rules with Priority 1 can be fired in arbitrary order. The *fil-Rule* has Priority 2 to ensure that all at-least and at-most restrictions for a node x are encoded and satisfied by the inequation solver before creating any new nodes. This justifies why role fillers or nominals are never merged nor removed from G; a distribution of role fillers and nominals either survives into a complete model or fails due to a clash. Also, assigning the *fil-Rule* Priority 2 helps in early clash detection in the case when the inequation solver detects a numerical clash even before new nodes are created. The $\forall_{\setminus}$-*Rule* has Priority 3 to ensure that the semantics of the $\forall_{\setminus}$ operator are not violated. We allow the creation of all possible edges between a node and its successors before applying the $\forall_{\setminus}$ operator semantics. This rule priority is needed to ensure the completeness (see Lemma 7.3 for proof) of the algorithm.

6.4 *Explaining the Rules*

The $\sqcap$-Rule, $\sqcup$-Rule, $\forall$-Rule and the $\forall_+$-Rule in Fig. 2 are similar to the ones in [8,16].

$\sqcap$-Rule	**If** $C \sqcap D \in \mathcal{L}(x)$, and $\{C, D\} \nsubseteq \mathcal{L}(x)$
	Then set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{C, D\}$
$\sqcup$-Rule	**If** $C \sqcup D \in \mathcal{L}(x)$, and $\{C, D\} \cap \mathcal{L}(x) = \emptyset$
	Then set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{E\}$ with $E \in \{C, D\}$
$\forall$-Rule	**If** $\forall R.C \in \mathcal{L}(x)$ and there exists y such that $\mathcal{L}(\langle x, y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$, and $C \notin \mathcal{L}(y)$
	Then set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$
$\forall_+$-Rule	**If** $\forall R.C \in \mathcal{L}(x)$ and there exists y such that $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) \neq \emptyset$, $S \in N_{R^+}$ with $S \sqsubseteq_* R$, and $\forall S.C \notin \mathcal{L}(y)$
	Then set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{\forall S.C\}$

Fig. 2. Completion rules for $\mathcal{SHON}_{\mathcal{R}\setminus}$ - Part I

$\forall_\setminus$-Rule. This rule is used to enforce the semantics of the role set difference operator $\forall_\setminus$ introduced at preprocessing by making sure that all *R-fillers* are labelled. Together with the *ch*-Rule (see explanation below), this rule has the same effect as the *choose*-rule in [16] and allows for the unsatisfiability of concepts like $((\geq 3R.C) \sqcap (\leq 1R.D) \sqcap (\leq 1R.\neg D))$ to be detected (See Example 2 in [10] for details).

$\bowtie$-Rule. This rule encodes the numerical restrictions in the label $\mathcal{L}$ of a node x, for some role $R \in N_R$, into a set of inequations maintained in $\mathcal{L}_{\mathrm{E}}(x)$ (P1 in Section 6.1). An inequation solver is always active and is responsible for finding a non-negative integer solution σ (P2 in Section 6.1) or triggering a clash if no solution is possible. If the inequations added by this rule do not trigger a clash, then the encoded at-least/at-most restriction can be satisfied by a possible distribution of role fillers. We distinguish two cases:

- Case (i): R-fillers of x must also satisfy a qualified restriction C due to a $\forall S.C$ restriction on a role S such that $R \sqsubseteq_* S$ and C is either a nominal or a qualification. Then the numerical restriction is encoded on partitions $P \in \mathcal{P}$ with $P^{\mathcal{I}} \subseteq (C^{\mathcal{I}} \cap FIL(R))$ which means $\{R, \theta^{-}(C)\} \subseteq P$ where $\theta^{-}(C)$ returns the qualification name q mapped to C.
- Case (ii): There exist no qualified restrictions on R-fillers of x due to a $\forall$ restriction on a role S such that $R \sqsubseteq_* S$. In this case the numerical restriction is encoded on partitions $P \in \mathcal{P}$ with $P^{\mathcal{I}} \subseteq FIL(R)$ which means $\{R\} \subseteq P$.

Unlike in [11,8,10], a distinction needs to be made between case (i) and case (ii) in order to preserve completeness of the algorithm. Otherwise the encoded inequations for $(\geq 1R \sqcap \forall S.C) \in \mathcal{L}(x)$ and $(\geq 1R \sqcap \forall S.\dot{\neg}C) \in \mathcal{L}(y)$ with $R \sqsubseteq_* S$ will be encoded on partitions $P \in \mathcal{P}$ such that $P^{\mathcal{I}} \subseteq FIL(R)$ and the qualifications imposed by $\forall S.C$ and $\forall S.\dot{\neg}C$ are lost because now we have $FIL(R, x) \equiv FIL(R, y)$ whereas $FIL(R, x) \subseteq C^{\mathcal{I}}$ and $FIL(R, y) \subseteq (\dot{\neg}C)^{\mathcal{I}}$.

***ch*-Rule.** This rule checks for empty partitions while ensuring completeness of the algorithm. Given a set of inequations in the label ($\mathcal{L}_{\mathrm{E}}$) of a node x and a variable v such that $\alpha(v) = P$ and $P \in \mathcal{P}$ we distinguish between two cases:

$\forall_{\setminus}$-Rule **If** $\forall(R\setminus S).C \in \mathcal{L}(x)$, and there exists y such that $\mathcal{L}(\langle x,y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$, $\mathcal{L}(\langle x,y\rangle) \cap (H(S) \cup \{S\}) = \emptyset$, and $C \notin \mathcal{L}(y)$

Then set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$

$\bowtie$-Rule **If** $(\bowtie nR) \in \mathcal{L}(x)$ for $\bowtie \in \{\leq, \geq\}$,

Then If $\forall S.C \in \mathcal{L}(x)$ with $R \sqsubseteq_* S$ and $\xi(\{R, \theta^-(C)\}, \bowtie, n) \notin \mathcal{L}_{\mathrm{E}}(x)$

Then set $\mathcal{L}_{\mathrm{E}}(x) = \mathcal{L}_{\mathrm{E}}(x) \cup \{\xi(\{R, \theta^-(C)\}, \bowtie, n)\}$

Else If $\xi(\{R\}, \bowtie, n) \notin \mathcal{L}_{\mathrm{E}}(x)$ **then** set $\mathcal{L}_{\mathrm{E}}(x) = \mathcal{L}_{\mathrm{E}}(x) \cup \{\xi(\{R\}, \bowtie, n)\}$

ch-Rule **If** there exists v occurring in $\mathcal{L}_{\mathrm{E}}(x)$ such that $\{v \geq 1, v \leq 0\} \cap \mathcal{L}_{\mathrm{E}}(x) = \emptyset$

Then set $\mathcal{L}_{\mathrm{E}}(x) = \mathcal{L}_{\mathrm{E}}(x) \cup \{V\}$, $V \in \{v \geq 1, v \leq 0\}$

fil-Rule **If** there exists v occurring in $\mathcal{L}_{\mathrm{E}}(x)$ with $\sigma(v) = m$ and $m > 0$, and

there exists no y with $\mathcal{L}_P(y) = \alpha(v)$

Then 1. create a new node y, 2. set $\mathcal{L}_P(y) = \alpha(v)$, 3. set $\mathcal{L}_{\mathrm{E}}(y) = \mathcal{L}_{\mathrm{E}}(x)$,

4. set $\mathcal{L}(y) = \begin{pmatrix} \{C'_{\mathcal{T}}\} \cup \bigcup_{o \in (\alpha(v) \cap N_o)} o \cup \bigcup_{i \in (N_o \setminus \alpha(v))} \neg i \cup \\ \bigcup_{q \in (\alpha(v) \cap \mathcal{Q}_{\mathrm{N}})} \theta(q) \cup \bigcup_{p \in (\mathcal{Q}_{\mathrm{N}} \setminus \alpha(v))} \doteq_{\mathcal{Q}} \theta(p) \end{pmatrix}$

e-Rule **If** $(\bowtie nR) \in \mathcal{L}(x)$, and there exists y such that $R \in \mathcal{L}_P(y)$ and $R \notin \mathcal{L}(\langle x,y\rangle)$

Then If $\forall S.C \in \mathcal{L}(x)$ with $R \sqsubseteq_* S$ and $\theta^-(C) \in \mathcal{L}_P(y)$, or

$\forall S.C \notin \mathcal{L}(x)$ with $R \sqsubseteq_* S$

Then set $\mathcal{L}(\langle x,y\rangle) = \mathcal{L}(\langle x,y\rangle) \cup \{R\}$, and

If $\mathcal{L}_{\mathrm{E}}(x) \not\subseteq \mathcal{L}_{\mathrm{E}}(y)$ **Then** set $\mathcal{L}_{\mathrm{E}}(y) = \mathcal{L}_{\mathrm{E}}(y) \cup \mathcal{L}_{\mathrm{E}}(x)$

Fig. 3. Completion rules for $\mathcal{SHON}_{\mathcal{R}\setminus}$ - Part II

- (i) The case when $P^{\mathcal{I}}$ must be empty ($v \leq 0$); this happens when restrictions on elements of this partition trigger a clash because the signature of P cannot be satisfied. For instance, if $\{\forall R_1.A,\ \forall R_2.\neg A\} \subseteq \mathcal{L}(x)$, $v_{R_1R_2} \geq 1 \in \mathcal{L}_{\mathrm{E}}(x)$ and there exists a node y with $\mathcal{L}_P(y) = \{R_1, R_2\}$ and $\{R_1, R_2\} \subseteq \mathcal{L}(\langle x,y\rangle)$ the qualifications on R_1 and R_2-fillers trigger a clash $\{A, \neg A\} \subseteq \mathcal{L}(y)$ and $v_{R_1R_2} \leq 0$ is enforced.
- (ii) The case when $P^{\mathcal{I}}$ must have at least one element ($1 \leq m \leq \sigma(v)$); if $P^{\mathcal{I}}$ can have at least one element without causing any clash, this means that the signature of P is satisfiable and we can have m elements also in $P^{\mathcal{I}}$ without a clash.

Since the inequation solver is unaware of partition signatures imposing restrictions on role fillers we allow an explicit distinction between cases (i) and (ii). We do this by non-deterministically assigning ≤ 0 or ≥ 1 for each variable v occurring in $\mathcal{L}_{\mathrm{E}}(x)$.

***fil*-Rule**. This rule is used to generate individual nodes depending on the distribution (σ) returned by the inequation solver. The rule is fired for every non-empty partition P based on $\sigma(v)$. It generates one proxy node y as the representative for the m elements

assigned to $P^{\mathcal{I}}$ by the inequation solver. The node y is tagged with its partition name using $\alpha(v)$ in $\mathcal{L}_P(y)$. The set of inequations is accumulated in $\mathcal{L}_E(y)$. Nominals and qualifications satisfied by the partition elements are extracted from the partition name and added to $\mathcal{L}(y)$ and for nominals and qualifications not contained in $\alpha(v)$ their negations are added. $C'_{\mathcal{T}}$ is added to $\mathcal{L}(y)$ to ensure that every node created by the *fil*-Rule also satisfies $C'_{\mathcal{T}}$.

***e*-Rule.** This rule creates the edges between the proxy nodes created by the *fil*-Rule. If $\geq nR \in \mathcal{L}(x)$, for some R, this means that x must be connected to a number r of R-fillers such that $n \leq r$. If $\leq mR \in \mathcal{L}(x)$ then x could be connected to a maximum number r' of R-fillers such that $r' \leq m$. If there exists a node y such that $R \in \mathcal{L}_P(y)$, this means that a distribution of R-fillers has been assigned by the inequation solver such that the numbers n and m are satisfied and y is a representative for a number p of R-fillers such that $r \leq p \leq r'$. We distinguish between two cases:

- Case (i): R-fillers of x must also satisfy a qualified restriction C due to a $\forall S.C$ restriction on a role S such that $R \sqsubseteq_* S$. In this case, if $\theta^{-}(C)$ is also in $\mathcal{L}_P(y)$ then the partition represented by y intersects with $C^{\mathcal{I}}$ and y is a member of C.
- Case (ii): There exists no qualified restrictions on R-fillers of x due to a $\forall S.C$ restriction on a role S such that $R \sqsubseteq_* S$. In this case there is no restriction on the partitions intersecting with R-fillers.

In both cases, an edge can safely be created between x and y such that $R \in \mathcal{L}(\langle x, y\rangle)$ and this edge is also a representative for the number p of edges between x and the p elements represented by y. If S is also in $\mathcal{L}_P(y)$ this means that the p R-fillers represented by y are also S-fillers and y is a representative for a partition $p \in \mathcal{P}$ such that $p^{\mathcal{I}} \subseteq FIL(R) \cap FIL(S)$. Therefore y can be re-used to connect x or another node y having $\geq n'S$ or $\leq m'S$, $n' \leq n$ and $m' \geq m$, in their label. In the case where $n = 0$ or $m = 0$ the CCG will not have any nodes representing the corresponding role fillers, because the inequation solver will not assign a distribution of fillers, and the *e*-Rule will not fire. One might argue that the *e*-Rule does not need to fire for $\leq mR \in \mathcal{L}(x)$. However, if we have a node x with $\{\geq 1R_1, \forall R_1.C, \leq 1R_2, \forall R_2.C, \forall R \setminus R_2.\neg C\} \subseteq \mathcal{L}(x)$ with $R_1 \sqsubseteq R$, and $R_2 \sqsubseteq R$ and a node y such that $\mathcal{L}_P(y) = \{R_1, R_2\}$, then if the *e*-Rule only fires for $\geq 1R_1$ then the edge created between x and y will satisfy only $R_1 \in \mathcal{L}(\langle x, y\rangle)$ and the $\forall_{\setminus}$-Rule propagates $\neg C$ to y leading to a clash making the algorithm incomplete because y has also been assigned as an R_2-filler.

6.5 Example

To better illustrate the calculus, we demonstrate it by checking the consistency of the TBox in Example 3.1 which we adapt to include cycles as follows:

$$\mathcal{T} = \left\{ \begin{array}{l} A \sqsubseteq \geq 1R.A,\ B_1 \sqsubseteq \forall R.C,\ B_2 \sqsubseteq \forall R.\neg C, \\ o \sqsubseteq \geq 1S_1.(A \sqcap B_1) \sqcap \geq 1S_2.(A \sqcap B_2) \end{array} \right\}$$

$\mathcal{T}$ contains cyclic descriptions, nominals and numerical restrictions with qualifications, and can be used to highlight some of the strong features (see Section 8) of the algebraic tableau algorithm presented in this article. Initially, $N_{\mathrm{R}} = \{R, S_1, S_2\}$, $N_{\mathrm{o}} = \{o\}$, $Q^{\neg}_{\mathrm{C}} = \emptyset$. After applying Algorithm 1 to $C_{\mathcal{T}}$ as was illustrated for Example 3.1 in Section 6.1 we have:

$$C'_{\mathcal{T}} = \left(\begin{array}{l} (\neg A \sqcup \geq 1 R_1 \sqcap \forall R_1.A) \sqcap (\neg B_1 \sqcup \forall R.C) \sqcap (\neg B_2 \sqcup \forall R.\neg C) \sqcap \\ (\neg o \sqcup (\geq 1 S_{11} \sqcap \forall S_{11}.(A \sqcap B_1) \sqcap \geq 1 S_{21} \sqcap \forall S_{21}.(A \sqcap B_2))) \end{array} \right)$$

To test the consistency of $\mathcal{T}$, we need to check that at least one individual i is a member of $C'_{\mathcal{T}}$ ($i \sqsubseteq C'_{\mathcal{T}}$ with $i \in N_{\mathrm{o}}$ new in $\mathcal{T}$). Now we have:

$N_{\mathrm{R}} = \{R, R_1, S_1, S_{11}, S_2, S_{21}\}$, $\mathcal{R} = \{R_1 \sqsubseteq R,\ S_{11} \sqsubseteq S_1, S_{21} \sqsubseteq S_2\}$, $N_{\mathrm{o}} = \{o, i\}$

$H(R) = \{R_1\}$, $H(S_1) = \{S_{11}\}$, $H(S_2) = \{S_{21}\}$

$Q_{\mathrm{C}}(R) = \{C, \neg C\}$, $Q_{\mathrm{C}}(S_1) = \emptyset$, $Q_{\mathrm{C}}(S_2) = \emptyset$

$Q^{\neg}_{\mathrm{C}}(R) = \{\neg C, C\}$, $Q^{\neg}_{\mathrm{C}}(S_1) = \emptyset$, $Q^{\neg}_{\mathrm{C}}(S_2) = \emptyset$

$\mathcal{DS} = \{R_1, S_{11}, S_{21}, o, i, \theta^{-}(C)\}$ [9]

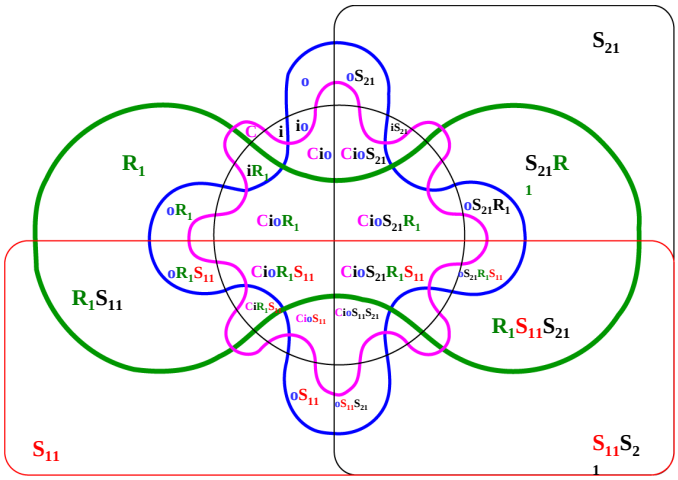


Fig. 4. Atomic Decomposition of $\mathcal{DS} = \{R_1, S_{11}, S_{21}, o, i, \theta^{-}(C)\}$ (assuming $\theta^{-}(C) = C$)

The atomic decomposition of $\mathcal{DS}$ is shown in Fig. 4 [10] which defines the partitioning $\mathcal{P} = \{\{R_1\}, \{S_{11}\}, \{S_{21}\}, \{o\}, \{i\}, \{R_1, S_{11}\}, \{R_1, S_{21}\}, \{R_1, o\}, \{S_{11}, o\}, \ldots, \{R_1, S_{11}, S_{21}, o, i, \theta^{-}(C)\}\}$ of domain elements. We define the set $\mathcal{V}$ of variables associated with each partition in $\mathcal{P}$: $\mathcal{V} = \{v_{\mathrm{R}_1}, v_{\mathrm{S}_{11}}, v_{\mathrm{S}_{21}}, v_{\mathrm{o}}, v_{\mathrm{i}}, \ldots, v_{\mathrm{R}_1\mathrm{S}_{11}\mathrm{S}_{21}\mathrm{oiC}}\}$. The calculus starts with the CCG $G = (\{r_0\}, \emptyset, \mathcal{L}, \mathcal{L}_E, \emptyset)$ and $\mathcal{L}_{\mathrm{E}}(r_0) = \Big\{ \xi(\{o\}, \geq, 1), \xi(\{o\}, \leq, 1), \xi(\{i\}, \geq, 1), \xi(\{i\}, \leq, 1) \Big\}$.

After applying the *ch*-Rule until it is not applicable anymore we might come up with the case where $v_{\mathrm{R}_1\mathrm{S}_{21}\mathrm{i}} \geq 1$ and all other variables are ≤ 0. A clash is detected since no solution is possible for ξ_{r_0} because one variable indexed with o must be ≥ 1 to satisfy the inequations in $\mathcal{L}_{\mathrm{E}}(r_0)$. Let us consider two more cases:

- Case (a): Considering a CCG with choices for the *ch*-Rule rule such as $v_{\mathrm{oi}} \geq 1$ and all other applicable variables are ≤ 0. The CCG for this case is illustrated in Fig. 5(a).
- Case (b): Considering a CCG with choices for the *ch*-Rule rule such as $v_{\mathrm{i}} \geq 1$, $v_{\mathrm{o}} \geq 1$ and all other applicable variables are ≤ 0, The CCG for this case is illustrated in Fig. 5(b).

[9] We only include C to Q_C if C and/or $\neg C$ are used as qualifying concepts. For sake of simplicity we assume in the following that $\theta^{-}(C) = C$.

[10] Some partitions are left unnamed in the figure for better clarity.

We illustrate the application of the completion rules in case (a) where the inequation solver returns a solution σ with $\sigma(v_{\text{oi}}) = 1$ and all other variables are zero. The *fil*-Rule becomes applicable to r_0 and one new node X is created such that:

$$\mathcal{L}_P(X) = \alpha(v_{\text{oi}}) = \{o, i\}, \mathcal{L}_{\text{E}}(X) = \mathcal{L}_{\text{E}}(r_0)$$

$$\mathcal{L}(X) = \left\{ \begin{array}{l} \bigcup_{o \in (\alpha(v) \cap N_o)} o \cup \bigcup_{i \in (N_o \setminus \alpha(v))} \neg i \cup \\ \bigcup_{q \in (Q_{\text{N}} \cap \alpha(v))} \theta(q) \cup \bigcup_{p \in (Q_{\text{N}} \setminus (Q_{\text{N}} \cap \alpha(v)))} \dot{\neg}_Q \theta(p) \cup \{C'_{\mathcal{T}}\} \end{array} \right\}$$

$$\mathcal{L}(X) = \left\{ \begin{array}{l} o, i, \neg C, (\neg A \sqcup (\geq 1R_1 \sqcap \forall R_1.A)) \sqcap (\neg B_1 \sqcup \forall R.C) \sqcap (\neg B_2 \sqcup \forall R.\neg C) \sqcap \\ (\neg o \sqcup (\geq 1S_{11} \sqcap \forall S_{11}.(A \sqcap B_1) \sqcap \geq 1S_{21} \sqcap \forall S_{21}.(A \sqcap B_2))) \end{array} \right\}$$

After applying the $\sqcap$-Rule, $\sqcup$-Rule, and $\bowtie$-Rule to X without having a clash, we get:

$$\mathcal{L}(X) = \left\{ o, i, \neg C, \geq 1R_1, \forall R_1.A, \forall R.C, \neg B_2, \geq 1S_{11}, \forall S_{11}.(A \sqcap B_1), \geq 1S_{21}, \forall S_{21}.(A \sqcap B_2) \right\}$$

$$\mathcal{L}_{\text{E}}(X) = \mathcal{L}_{\text{E}}(X) \cup \left\{ \xi(\{R_1, C\}, \geq, 1), \xi(\{S_{11}\}, \geq, 1), \xi(\{S_{21}\}, \geq, 1) \right\}$$

The *ch*-Rule is applicable several times to X, we consider the cases of adding v_{R_1C}, $v_{S_{11}}$ and $v_{S_{21}}$ all ≥ 1 in $\mathcal{L}_E(X)$ and all other applicable variables set to ≤ 0. The inequation solver now assigns $v_{S_{11}}$, $v_{S_{21}}$, and v_{R_1C} to 1 and the *fil*-Rule becomes applicable and the nodes X_1, X_2, and X_3 are created such that

$$\mathcal{L}_P(X_1) = \alpha(v_{\text{S}_{11}}) = \{S_{11}\}, \mathcal{L}_P(X_2) = \alpha(v_{\text{S}_{21}}) = \{S_{21}\}, \mathcal{L}_P(X_3) = \alpha(v_{\text{R}_1\text{C}}) = \{R_1, C\}$$

$$\mathcal{L}_{\text{E}}(X_1) = \mathcal{L}_{\text{E}}(X_2) = \mathcal{L}_{\text{E}}(X_3) = \mathcal{L}_{\text{E}}(X)$$

$$\mathcal{L}(X_1) = \{\neg o, \neg i, \neg C\} \cup \{C'_{\mathcal{T}}\}, \mathcal{L}(X_2) = \{\neg o, \neg i, \neg C\} \cup \{C'_{\mathcal{T}}\}, \mathcal{L}(X_3) = \{\neg o, \neg i, C\} \cup \{C'_{\mathcal{T}}\}$$

The *e*-Rule is now applicable to X three times and the edges are created between X, X_1, X_2, X_3 as follows: $\mathcal{L}(\langle X, X_1 \rangle) = \{S_{11}\}$, $\mathcal{L}(\langle X, X_2 \rangle) = \{S_{21}\}$, $\mathcal{L}(\langle X, X_3 \rangle) = \{R_1\}$

The $\forall$-Rule becomes applicable three times such that A is added to $\mathcal{L}(X_3)$, $(A \sqcap B_1)$ is added to $\mathcal{L}(X_1)$, and $(A \sqcap B_2)$ is added to $\mathcal{L}(X_2)$. After applying the $\sqcap$-Rule, and $\sqcup$-Rule to X_1, X_2, X_3 without having a clash, we have:

$$\mathcal{L}(X_1) = \{\neg o, \neg i, \neg C, A, B_1, \geq 1R_1, \forall R_1.A, \neg B_2, \forall R.C\}$$

$$\mathcal{L}(X_2) = \{\neg o, \neg i, \neg C, A, B_2, \geq 1R_1, \forall R_1.A, \neg B_1, \forall R.\neg C\}$$

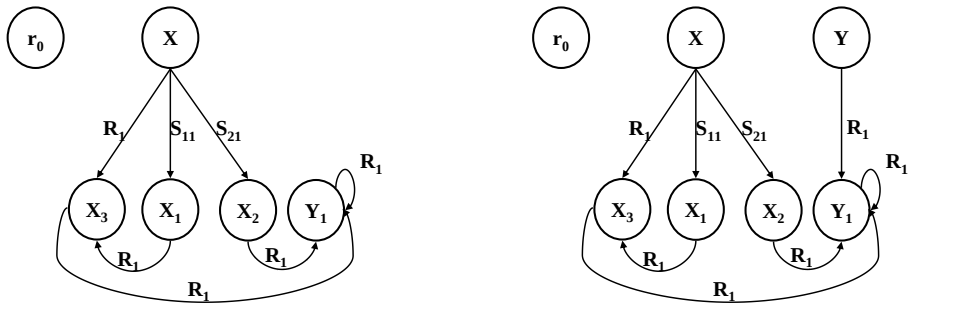
$$\mathcal{L}(X_3) = \{\neg o, \neg i, C, A, \geq 1R_1, \forall R_1.A, \neg B_1, \forall R.\neg C\}$$

The *e*-Rule is applicable to X_1 and an edge is created between X_1 and X_3 with $\mathcal{L}(\langle X_1, X_3 \rangle) = \{R_1\}$. Notice how X_3 has been re-used because it satisfies the conditions for the *e*-Rule and no other node does. The $\bowtie$-Rule is applicable to X_2 and X_3 such that $\xi(\{R_1, \neg C\}, \geq, 1)$ is added to $\mathcal{L}_E(X_2)$ and $\mathcal{L}_E(X_3)$. We consider the case when the *ch*-Rule assigns $v_{R_1} \geq 1$ and all other applicable variables to ≤ 0. The inequation solver collects all inequations and maps v_{R_1} to 1 rendering the *fil*-Rule applicable to X_3 and X_2 and one new node Y_1 is created such that $\mathcal{L}_P(Y_1) = \alpha(v_{R_1}) = \{R_1\}$, $\mathcal{L}_{\text{E}}(Y_1) = \mathcal{L}_{\text{E}}(X_2)$, $\mathcal{L}(Y) = \{\neg o, \neg i, \neg C\} \cup \{C'_{\mathcal{T}}\}$

The *e*-Rule is now applicable to X_2 and an edge is created between X_2 and Y_1 such that $\mathcal{L}(\langle X_2, Y_1 \rangle) = \{R_1\}$. Y_1 is re-used by the *e*-Rule on X_3 to create an edge between X_3 and Y_1 such that $\mathcal{L}(\langle X_3, Y_1 \rangle) = \{R_1\}$. The $\sqcap$-Rule, $\sqcup$-Rule, and $\bowtie$-Rule apply to Y_1 such that

$$\mathcal{L}(Y_1) = \{\neg o, \neg i, \neg C, A, \geq 1R_1, \forall R_1.A, \neg B_1, \forall R.\neg C\}, \mathcal{L}_E(Y_1) = \mathcal{L}_E(Y_1) \cup \xi(\{R_1, \neg C\}, \geq, 1)$$

$\xi(\{R_1, \neg C\}, \geq, 1)$ has already been satisfied by the inequation solver which means that



(a) Case with $\sigma(v_{oi}) = 1$, $\mathcal{L}_P(X) = \{o, i\}$

(b) Case ith $\sigma(v_o) = 1$, $\sigma(v_i) = 1$, $\mathcal{L}_P(X) = \{o, C\}$, $\mathcal{L}_P(Y) = \{i\}$

Fig. 5. CCG for $\mathcal{T}$

the e-Rule is now applicable to Y_1 re-using Y_1 to create an edge such that $\mathcal{L}(\langle Y_1, Y_1\rangle) = \{R_1\}$. No rules are applicable anymore and no clash has been detected: we have a CCG as shown in Fig. 5(a) consisting of the nodes X, X_1, X_2, X_3, Y_1 [11] and the initial TBox is consistent.

7 Proofs

The soundness, completeness and termination of the algorithm presented in this article are consequences of Lemmas 4.2, Lemma 7.1, 7.2, and 7.3.

Lemma 7.1 (Termination) *When started with a $\mathcal{SHOQ}$ TBox $\mathcal{T}$, the proposed algorithm terminates and is worst case double exponential.*

Proof. Let $l = \#clos(\mathcal{T})$, r denote the size of N_{R}, o denote the size of N_{o}, and q denote the size of Q_{N}, termination of the algebraic tableau algorithm is guaranteed due to the following.

- The rewriting in Algorithm 1 can be done in linear time and does not affect termination.
- Computing a partitioning $\mathcal{P}$ for $\mathcal{T}$: in the worst case $\#\mathcal{DS} = \#\{N_{\mathrm{R}} \cup Q_{\mathrm{N}} \cup N_{\mathrm{o}}\}$, and the size of $\mathcal{P}$ is bounded by $2^{r+o+q} - 1$ since we do not consider the empty partition. Although this computation is exponential, it is done only once.
- Getting a distribution of individuals (solution for the inequations) will not affect termination of the completion rules and can be computed in polynomial time since we have a fixed number $(2^{r+o+q} - 1)$ of variables [19].
- The algorithm constructs a graph consisting of a set of arbitrarily interconnected nodes by applying completion rules which do not remove nodes from the graph, nor remove concepts from node labels or edge labels. For each node x:
  - the number of times that the *fil*-Rule can be applied is bounded by the size of $\mathcal{P}$. In the worst case we need to create one individual for each partition. It is not possible to have more nodes in the graph since each node is either a nominal or a role filler and in both cases it must be in some partition in $\mathcal{P}$.
  - the number of times the e-Rule is applied for each $\bowtie nR$ restriction is bounded by n (the largest number used in a QCR restriction). In the worst case individuals satisfying

[11] The node r_0 is and will be ignored since it is not part of the pre-model.

$\bowtie nR$ are distributed into n partitions. The total number that this rule can be applied is bounded by $l * n$.
  - the *ch*-Rule non-deterministically assigns each variable to ≥ 1 or ≤ 0. Each variable is assigned once per completion graph which means that in the worst case when all possible completion graphs are explored, the *ch*-Rule is applied $2^{(2^{r+o+q}-1+1)} - 1$ times.
  - all other rules are applied at most l times.
- Traditional termination problems due to cyclic TBoxes and "yo-yo" effect are not encountered:
  - cyclic definitions do not cause a termination problem since nodes having the same label (case when blocking is needed with other algorithms) will eventually be mapped to the same partition and only one proxy node is created. This justifies why we do not need any blocking strategies, the re-use of individuals acts like a natural block.
  - The "yo-yo" effect of infinitely creating and merging nodes cannot occur since in a given CCG, nodes are neither removed nor merged.

□

Lemma 7.2 (Soundness) *If the completion rules can be applied to $\mathcal{T}$ such that they yield a complete and clash-free CCG, then $\mathcal{T}$ has a tableau.*

Proof. A tableau $\mathrm{T} = (\mathbf{S}, \mathcal{L}', \mathcal{E})$ can be obtained from a clash-free CCG $\mathrm{G} = (V, E, \mathcal{L}, \mathcal{L}_{\mathrm{E}}, \mathcal{L}_P)$ by mapping nodes in G to individuals in T which can be defined from G as T such that: $\mathbf{S} = V \setminus \{r_0\}$, $\mathcal{L}'(x) = \mathcal{L}(x)$, and $\mathcal{E}(R) = \{\langle x, y\rangle \in E \mid (H(R) \cup \{R\}) \cap \mathcal{L}(\langle x, y\rangle) \neq \emptyset\}$. We show that T is either a tableau or can be easily extended to a tableau for $\mathcal{T}$ since properties (i) - (xi) of a tableau (see Def. 4.1) are either satisfied or can be easily satisfied.

- Property (i): Assume there exists an individual x in $\mathbf{S}$ such that $C_{\mathcal{T}} \notin \mathcal{L}'(x)$, this means that the corresponding node x in G also satisfies $C_{\mathcal{T}} \notin \mathcal{L}(x)$. This case is not possible first because x cannot be r_0 and second because $C_{\mathcal{T}}$ is added to $\mathcal{L}(x)$ for every node x created in G by the *fil*-Rule. Hence $C_{\mathcal{T}} \in \mathcal{L}'(x)$ for every $x \in \mathbf{S}$ and Property (i) is satisfied.
- Property (ii): Assume there exists an individual x in $\mathbf{S}$ such that $A \in \mathcal{L}'(x)$ and $\neg A \in \mathcal{L}'(x)$ this means that there exists a corresponding node x in G such that $A \in \mathcal{L}(x)$ and $\neg A \in \mathcal{L}(x)$. This case is not possible since G is clash-free. Hence, Property (ii) is satisfied.
- Property (iii): Assume there exists an individual x in $\mathbf{S}$ such that $C \sqcap D \in \mathcal{L}'(x)$, $C \in \mathcal{L}'(x)$, and $D \notin \mathcal{L}'(x)$ this means that there exists a corresponding node x in G such that $C \sqcap D \in \mathcal{L}(x)$, $C \in \mathcal{L}(x)$, and $D \notin \mathcal{L}(x)$. Having $C \sqcap D \in \mathcal{L}(x)$, $C \in \mathcal{L}(x)$, and $D \notin \mathcal{L}(x)$ makes the $\sqcap$-Rule applicable to x in G however this case is not possible since G is complete. Hence Property (iii) is satisfied and we can similarly prove that Property (iv) is also satisfied.
- Property (v): Assume $\forall S.C \in \mathcal{L}'(x)$ and $\langle x, y\rangle \in \mathcal{E}(S)$ then we must have $C \in \mathcal{L}'(y)$. Having $\langle x, y\rangle \in \mathcal{E}(S)$ means that $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) \neq \emptyset$. Since G is complete and clash free then C must be in $\mathcal{L}(y)$ otherwise the $\forall$-Rule conditions are met and the rule is applicable to G. Since $C \in \mathcal{L}(y)$ this means that $C \in \mathcal{L}'(y)$ and Property (v) is satisfied.
- Property (vi): Assume there exists an individual x in $\mathbf{S}$ such that $\forall S.C \in \mathcal{L}'(x)$ and there exists an individual $y \in \mathbf{S}$ such that $\langle x, y\rangle \in \mathcal{E}(R)$ and R is a transitive role such that $R \sqsubseteq S \in \mathcal{R}$ then we must have $\forall R.C \in \mathcal{L}'(y)$. Since we have $\langle x, y\rangle \in \mathcal{E}(R)$ this means

that $\mathcal{L}(\langle x, y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$, and having $R \in N_{R+}$ with $R \sqsubseteq_* S \in \mathcal{R}$ then we have $\forall R.C \in \mathcal{L}(y)$ otherwise the $\forall_+$ would be applicable. Therefore, since $\forall R.C \in \mathcal{L}(y)$ then $\forall R.C \in \mathcal{L}'(y)$ and Property (vi) is satisfied.

- Property (vii): Assume $\forall R \backslash S.C \in \mathcal{L}'(x)$ and $\langle x, y\rangle \in \mathcal{E}(R)$ but not in $\mathcal{E}(S)$ then we must have $C \in \mathcal{L}'(y)$. Since we have $\langle x, y\rangle \in \mathcal{E}(R)$ and $\langle x, y\rangle \notin \mathcal{E}(S)$, this means that $\mathcal{L}(\langle x, y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$ and $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) = \emptyset$ respectively. C must be in $\mathcal{L}(y)$ otherwise the $\forall_\backslash$-Rule would be applicable to G. Since $C \in \mathcal{L}(y)$ this means that $C \in \mathcal{L}'(y)$ and Property (vii) is satisfied
- Properties (viii): Assume $(\geq nS) \in \mathcal{L}'(x)$ then completeness of G implies that there exist j proxy individuals $y_1 \ldots y_j$ each representing a partition of m_i individuals such that $\sum_{i=1}^{j} m_i = n$ and $S \in \mathcal{L}(\langle x, y_i\rangle)$ $(1 \leq i \leq j)$. Due to Lemma 6.3, we can replicate each y_i, $m_i - 1$ times and set $\mathbf{S} = \mathbf{S} \cup \{y_{i_k}\}$ and $\mathcal{L}(\langle x, y_{i_k}\rangle) = S$ with $1 \leq k \leq m_i - 1$, then we have $\#S^{\mathcal{I}}(x) \geq n$ and property (viii) is satisfied. One might think that replicating individuals could result in violating the nominals semantics (Property xi) for example by replicating a nominal individual. However, this case can never happen since nominals are represented by proxy individuals y_i belonging to a partition with only one individual, $m_i = 1$ always holds for nominals partitions and is encoded by the inequations (see Property (xi) below). Similarly, Property (ix) cannot be violated due to replication of individuals; partition sizes (m_i) are assigned such that all at-least and at-most restrictions are satisfied (see Property (ix) below).
- Property (ix): Assume $(\leq mS) \in \mathcal{L}'(x)$ and $\#S^{\mathcal{I}}(x) \leq m$ is violated. This means that we have j proxy individuals $y_1 \ldots y_j$ each representing a partition of m_i individuals such that $\sum_{i=1}^{j} m_i > m$. This case cannot happen for two reasons: (1) Having the lowest priority for the *fil-Rule*, nodes are created only after making sure that all at-least and at-most restrictions for a node x are satisfied by a distribution of role fillers (a non-negative integer solution for the inequations in $\mathcal{L}_E(x)$). This means that no nodes will be created that violate an at-most restriction. (2) G is clash free which means that for each $(\leq mS) \in \mathcal{L}(x)$ we have $\xi(\{S\}, \leq, m)$ in $\mathcal{L}_E(x)$ and there is no $\xi(\{S\}, \geq, n)$ in $\mathcal{L}_E(x)$ and $n > m$.
- Property (x): If the distribution is not consistent with $\mathcal{R}$, then for some $(R' \sqsubseteq_* R)$, there exists an R'-*filler* y assigned to a partition P with $R' \in P$ and $P^{\mathcal{I}} \subseteq (FIL(R') \setminus FIL(R))$. This case is not possible due to the definition of H(R) which assumes that R is implied in P whenever $R' \in P$ and $R' \in H(R)$. Hence, this property is always satisfied.
- Property (xi): G cannot contain two nodes x and y such that for some nominal $o \in N_{\mathrm{o}}$ we have $o \in \mathcal{L}(x) \cap \mathcal{L}(y)$. Since each node in G is a representative for a partition P then having two nodes x and y with $o \in \mathcal{L}(x) \cap \mathcal{L}(y)$ means that there are two partitions P_1 and P_2 such that $o \in P_1 \cap P_2$. However since partitions are disjoint (Lemma 5.5) and due to the nominals semantics encoded into $\xi(\{o\}, \leq, 1)$ and $\xi(\{o\}, \geq, 1)$ in $\mathcal{L}_E(r_0)$ the inequation solver will assign the nominal o to only one partition P_1 or P_2 and all other partitions will have $\neg o$ in the label of their proxies. In addition, no nodes that are created can be removed or merged, and no nominals individual can be replicated to satisfy Property viii. Therefore, the set of nodes with a nominal o in their label always satisfies property xi.

□

Lemma 7.3 (Completeness) *If $\mathcal{T}$ has a tableau, then the completion rules can be applied to $\mathcal{T}$ such that they yield a complete and clash-free CCG.*

Proof. Let T = $(\mathbf{S}, \mathcal{L}', \mathcal{E})$ be a tableau for $\mathcal{T}$, T can be used to guide the application of the completion rules. We define the mapping function π from nodes in the graph $G = (V, E, \mathcal{L}, \mathcal{L}_{\mathrm{E}}, \mathcal{L}_{\mathrm{P}})$ to individuals in $\mathbf{S}$, inductively with the creation of new nodes, such that for each $x, y \in V$, roles $R, S \in N_{\mathrm{R}}$ and a partition name $p \in \mathcal{P}$ we have:

(i) $\mathcal{L}(x) \subseteq \mathcal{L}'(\pi(x))$

(ii) if $\langle x, y\rangle \in E$ and $S \in \mathcal{L}(\langle x, y\rangle)$, then $\langle \pi(x), \pi(y)\rangle \in \mathcal{E}(S)$

(iii) $\xi(\{R\}, \geq, n) \in \mathcal{L}_{\mathrm{E}}(x)$ implies $\#R^{\mathcal{I}}(\pi(x)) \geq n$

(iv) $\xi(\{R\}, \leq, n) \in \mathcal{L}_{\mathrm{E}}(x)$ implies $\#R^{\mathcal{I}}(\pi(x)) \leq n$

(v) $\xi(\{R, q\}, \geq, n) \in \mathcal{L}_{\mathrm{E}}(x)$ implies $\#(R^{\mathcal{I}}(\pi(x)) \cap \theta(q)^{\mathcal{I}}) \geq n$

(vi) $\xi(\{R, q\}, \leq, n) \in \mathcal{L}_{\mathrm{E}}(x)$ implies $\#(R^{\mathcal{I}}(\pi(x)) \cap \theta(q)^{\mathcal{I}}) \leq n$

The claim is that having a CCG G that satisfies the properties of π we can apply the completion rules defined in Fig. 2 and 3, when applicable, to G without violating the properties of π. Initially G consists of the artificial node r_0 such that $\bigcup_{o \in N_{\mathrm{o}}} \{\xi(\{o\}, \geq, 1), \xi(\{o\}, \leq, 1)\} \subseteq \mathcal{L}_E(r_0)$ and at least one node x_0 with some $o \in \mathcal{L}(x_0)$ is created. Given a tableau T for G, we can set $s_0 = \pi(x_0)$ for some $s_0 \in \mathbf{S}$.

We show that whenever we can apply a completion rule to G, the properties of π are not violated: applying the $\sqcap$-Rule, $\sqcup$-Rule, or the $\forall$-Rule strictly extends the label of a node x and this does not violate properties of π due to properties (i)-(v) of a tableau. Let us consider applying the other rules to a given node x:

- The $\forall_+$**-Rule**: Having a node x in G such that $\forall R.C \in \mathcal{L}(x)$ and there exists a node y with $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) \neq \emptyset$ and S is a transitive role such that $S \sqsubseteq_* R$, this means that there exists $\pi(x) \in \mathbf{S}$ such that $\forall R.C \in \mathcal{L}'(\pi(x))$ and there exists $\pi(y) \in \mathbf{S}$ such that $\langle \pi(x), \pi(y)\rangle \in \mathcal{E}(S)$. Applying the $\forall_+$-Rule adds $\forall S.C$ to $\mathcal{L}(y)$ thus preserving Property (vi) of a tableau ($\forall S.C \in \mathcal{L}'(\pi(y))$) without violating π.
- The $\forall_\backslash$**-Rule**: Having $\forall R\backslash S.C \in \mathcal{L}(x)$ with $\mathcal{L}(\langle x, y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$ and $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) = \emptyset$ this means that $\forall R\backslash S.C \in \mathcal{L}'(\pi(x))$ with $\langle \pi(x), \pi(y)\rangle \in \mathcal{E}(R)$ and $\langle \pi(x), \pi(y)\rangle \notin \mathcal{E}(S)$. Applying the $\forall_\backslash$-Rule adds C to $\mathcal{L}(y)$ which means that C is now in $\mathcal{L}'(\pi(y))$ and Property vii of a tableau is satisfied. This property along with properties of π cannot be violated later for example by having $\langle \pi(x), \pi(y)\rangle \in \mathcal{E}(S)$ due to the strategy of rule application which forces the $\forall_\backslash$-Rule to be applicable to a node only when no other rules are applicable. In particular, the e-Rule cannot be applied to x such that $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) \neq \emptyset$, which would adds $\langle \pi(x), \pi(y)\rangle$ to $\mathcal{E}(S)$, after the $\forall_\backslash$-Rule had been applied. For example, consider the following scenario:
  · Initially let $\{\geq nR, \geq mS, \forall R.A, \forall R\backslash S.\neg A\} \subseteq \mathcal{L}(x)$ and y be a proxy node with $\mathcal{L}_P(y) = \{R, S\}$
  · after applying the *e-Rule* for some $\geq nR \in \mathcal{L}(x)$ and the $\forall$*-Rule* for $(\forall R.A) \in \mathcal{L}(x)$, y is an *R-filler* of x with $\{A\} \subseteq \mathcal{L}(y)$
  · after applying the $\forall_\backslash$*-Rule* for $(\forall R\backslash S.A) \in \mathcal{L}(x)$ we have $\{A, \neg A\} \subseteq \mathcal{L}(y)$ with y an *R-filler* of x.

  This case cannot happen. Due to the strategy of rule applications in Section 6.3, the $\forall_\backslash$*-Rule* cannot be applied if the e-Rule can also be applicable. The rule priorities make sure that the $\forall(R\backslash S)$ semantics are enforced only when no more nodes can be *S-fillers* of x and Properties (v) and (vii) of the definition of a tableau are preserved.

- The $\bowtie$**-Rule**: If $(\geq nR)$ or $(\leq mR) \in \mathcal{L}(x)$, then $(\geq nR)$, $(\leq mR) \in \mathcal{L}'(\pi(x))$, this implies that $\#R^T(\pi(x)) \geq n$, $\#R^T(\pi(x)) \leq m$, (properties viii and ix of a tableau). Applying the $\bowtie$-Rule, extends $\mathcal{L}_E(x)$ wither with $\xi(\{R\}, \geq, n)$ or $\xi(\{R\}, \leq, m)$ if no qualifications on a super-role of R apply or with $\xi(\{R, \theta^-(C)\}, \geq, n)$ or $\xi(\{R, \theta^-(C)\}, \leq, m)$ if a qualification C also applies on R-fillers of x. In both cases the properties of π and those of a tableau are not violated.
- The ***fil*-Rule**: Since the *fil*-Rule has priority 2 then every $(\geq nR)$, $(\leq mR) \in \mathcal{L}(x)$ is already encoded into inequation in $L_E(x)$ and due to the clash freeness of T this means that there exists a distribution of role fillers satisfying every $(\geq nR)$, $(\leq mR) \in \mathcal{L}(x)$. The distribution of fillers is encoded in the solution σ for $\mathcal{L}_E(x)$ and applying the *fil*-Rule creates a proxy individual y as a representative for each corresponding partition based on σ returned by the inequation solver. Every node created is tagged with the proper partition name using $\mathcal{L}_P$ and the set of inequations is propagated using $\mathcal{L}_E(x)$ to y. $\mathcal{L}_P$ is later used by the e-Rule to create the proper edges between the nodes. Since the creating of nodes is guided by the solution, σ, returned by the inequation solver and due to the rule priority, the number of nodes created the *fil*-Rule cannot violate properties of a tableau nor π.
- The ***e*-Rule**: For each $(\geq nR) \in \mathcal{L}(x)$ we have $(\geq nR) \in \mathcal{L}'(\pi(x))$ which means that $\#R^T(\pi(x)) \geq n$ must be satisfied. The e-Rule is applied to connect x to its *R-fillers* such that with each i^{th} $(1 \leq i \leq n)$ application of this rule an edge is created between x and some proxy individual y_i such that $R \in \mathcal{L}_P(y_i)$ and y_i represents m_i (the number of elements assigned to a partition by the inequation solver) individuals of a partition p.

  After all edges are created we have j proxy *R-fillers* each representing m_i individuals such as $\sum_{i=1}^{j} m_i \geq n$. Due to Lemma 6.3 we can replicate each y_i, $m_i - 1$ times and by setting $\mathcal{L}(\langle x, y_{i_k}\rangle) = \{R\}$ with $1 \leq i \leq j$ and $1 \leq k \leq m_i - 1$ and by setting $\pi = \pi[y_{1_1} \to t_{1_1} \ldots y_{i_k} \to t_{i_k}]$ with $t_{1_1} \ldots t_{i_k}$ tableau elements in T satisfying $\#R^T(\pi(x)) \geq n$. We can see that $\#R^T(\pi(x)) \geq n$ is satisfied without violating π. By analogy, we can prove that applying the e-Rule for each $(\leq nR) \in \mathcal{L}(x)$ does not violate π.

The resulting graph G is clash free due to the following:

(i) G cannot contain a node x such that $\{A, \neg A\} \subseteq \mathcal{L}(x)$ since $\mathcal{L}(x) \subseteq \mathcal{L}'(\pi(x))$ and Property ii of the definition of a tableau would be violated.

(ii) G cannot contain a node x such that $\mathcal{L}_{\mathrm{E}}(x)$ is unsolvable. If $\mathcal{L}_{\mathrm{E}}(x)$ is unsolvable, this means that for some role $R \in N_{\mathrm{R}}$ we have:
  - $\{\xi(\{R\}, \geq, n)\} \subseteq \mathcal{L}_{\mathrm{E}}(x)$, and there is no possible distribution of *R-fillers* satisfying $\geq nR \subseteq \mathcal{L}(x)$, hence property viii of a tableau would be violated due to the equivalence properties between $\xi(\{R\}, \geq, n) \in \mathcal{L}_{\mathrm{E}}(x)$ and $\#R^T(\pi(x)) \geq n$ respectively, or
  - $\{\xi(\{R\}, \leq, n)\} \subseteq \mathcal{L}_{\mathrm{E}}(x)$, and there is no possible distribution of *R-fillers* satisfying $\leq mR$ hence property ix of a tableau would be violated due to the equivalence properties between $\xi(\{R\}, \leq, m) \in \mathcal{L}_{\mathrm{E}}(x)$ and $\#R^T(\pi(x)) \leq m$.

8 Discussion

In this section we highlight some of the novel features of our algorithm.

8.1 Completion Graph Characteristics

A compressed completion graph G for a KB $(\mathcal{T}, \mathcal{R})$ consists of the artificial root node r_0, which is not part of the model for KB, and arbitrarily interconnected nodes. We do not enforce a tree-like or forest-like restrictions on the shape of the graph as traditional tableau algorithms for $\mathcal{SHOQ}$ [16] and this is desirable since not all models are necessarily tree-shaped [21]. Such freedom in completion graph construction allows a better handling of KBs with complex structures for example, a KB for the human anatomy does not necessarily have a tree-shaped model (or a tree-shaped completion graph). Restricting a model to a tree-like one would unnecessarily complicate constructing G.

8.2 Using an Inequation Solver

Applying the algebraic algorithm with $\mathcal{T} = \left\{ \begin{array}{l} A \sqsubseteq \geq nR.A,\ B_1 \sqsubseteq \forall R.C,\ B_2 \sqsubseteq \forall R.\neg C, \\ o \sqsubseteq \geq nS_1.(A \sqcap B_1) \sqcap \geq nS_2.(A \sqcap B_2) \end{array} \right\}$ for large values of n ($n = 100$) will not affect the behavior of the algorithm as was reported in [12] for the DL $\mathcal{SHQ}$. This makes its extension to more expressive logics more promising. [12] Additionally, the inequation solver facilitates early clash detection (Definition 6.4 (ii)), and ensures that a minimum number of role fillers is considered by setting the objective function to minimize the sum of variables considered.

8.3 Termination

As illustrated in Section 6.5, termination is naturally inherent. Unlike traditional DL reasoning algorithms for $\mathcal{SHOQ}$, a tree model property accompanied by cycle detection techniques or blocking strategies is not crucial for termination. Nodes created are never merged or pruned which means that we do not need to handle the so-called "yoyo" effect or manage all incoming and outgoing edges of nodes.

8.4 Proxy individuals and their re-use

The completion graph used in this calculus is called "compressed completion graph" and this is due to the use and re-use of proxy nodes as representatives for nodes having common restrictions. Using proxy nodes helps minimize the number of individuals to be created and the number of completion rules to be triggered. When creating a representation for a distribution of domain elements let p_{a} denote the number of partitions used, $P_{\mathrm{a}} = \#\mathcal{P}$, p_o denote the number of nominals, and $p_{\bowtie}$ denote the number of at-least and at-most restrictions, we consider the following cases:

- Case 1: All individuals fall in the same partition and only one proxy is created. The KB is underconstrained and we create an overconstrained representation of it. In this case only one node is created other than r_0.
- Case 2: All individuals satisfying an at-least or an at-most restriction are in the same partition and only one proxy is created for each at-least or at-most restriction. In this case $p_{\mathrm{a}} = \max\{p_{\bowtie}, p_o\}$ if nominals interact with role fillers, or $p_{\mathrm{a}} = (p_{\bowtie} + p_o)$ if nominals do not interact with role fillers. The total number of nodes created equals p_{a}.

[12] Large values of n are known to be problematic for most DL reasoners supporting $\mathcal{SHQ}$.

- Case 3: Individuals satisfying each at-least and at-most restriction of the form $\bowtie nR$ are in n different partitions and n proxy nodes are created for each $\bowtie nR$ restriction. In this case $p_{\mathrm{a}} = (n * p_{\bowtie})$ if nominals interact with role fillers, or $p_{\mathrm{a}} = (n * p_{\bowtie} + p_o)$ if nominals do not interact with role fillers. The total number of nodes created equals p_{a}.

On the other hand, nodes that are created can later be re-used. The re-use of individuals has also been proposed in [21] recently. However, the re-use implemented by our calculus is more informed. Once a node is created, it is tagged based on the partition it belongs to. Which means it is tagged by the signature it can satisfy without violating a number restriction. For example when an individual is assigned to a partition labeled $\{R_1, R_2\}$ this means that this individual is a potential R_1-filler and a potential R_2-filler. The e-Rule uses and re-uses this individual whenever an R_1-filler or an R_2-filler is needed. In a sense, once a distribution of individuals is assigned by the inequation solver, the individual re-use is totally deterministic, there is no guessing of which individuals can be re-used. This form of re-use still ensures termination while preserving soundness and completeness. One could say that the re-use acts like blocking in the case of cyclic descriptions, however, we do not refer to it as a blocking strategy because first, we do not use any cycle detection, and second, the re-use is not intended for termination and it is not only used in the case of cycles. The use of a proxy individual together with the re-use of individuals could work as a double optimization to reduce non-determinism and model sizes especially since KBs are often naturally underconstrained which facilitates individual re-use.

8.5 Caching

The *ch*-Rule in Fig. 3 performs a semantic split for groups of individuals (a single partition) and not necessarily for each individual as is the case with tableau algorithms using a *choose*-rule [16] which chooses a distribution for each role filler. It is interesting to note that the splitting of the *ch*-Rule allows some form of global caching. Partitions represent signatures (Lemma 5.5) and variables are used to represent the cardinalities of these partitions. Then, if a variable must be zero, this means that the signature for the corresponding partitions is unsatisfiable. This result is carried throughout the search by setting the corresponding variable to zero and no individuals are assigned to that partition. However, if a variable v_P is ≥ 1 this means that the signature of $\alpha(v) = P$ is satisfiable and at least one individual x is a member of this signature. Whenever a new individual is needed satisfying the signature of P, x is re-used.

8.6 The EU Example

Consider testing the consistency of the the TBox $\mathcal{T}$ consisting of the axioms (2) and (1) as defined in Example 2.4. We replace the concept names, role and nominal names with a one letter symbol such that $N_{\mathrm{R}} = \{M', M\}$, $\mathcal{R} = \{M' \sqsubseteq M\}$, $N_{\mathrm{o}} = \{i, o_1, \ldots, o_{27}\}$ and

$$C'_{\mathcal{T}} = (\neg E \sqcup o_1 \sqcup \ldots \sqcup o_{27}) \sqcap ((\neg o_1 \sqcap \ldots \sqcap \neg o_{27}) \sqcup E) \sqcap (\neg F \sqcup \geq 30\, M' \sqcap \forall M'.E)$$

The partitioning of $\mathcal{DS} = \{M', i, o_1, \ldots, o_{27}\}$ results in $(2^{29} - 1)$ partitions. However, all instances of $EU\_MemberStates$ are disjoint and we can safely ignore partitions having more then one nominal. Figure 6 shows the corresponding partitioning; in total we only need to consider $(2 * 28 + 1)$ partitions.

Considering an initial distribution of nominals and after applying the completion rules

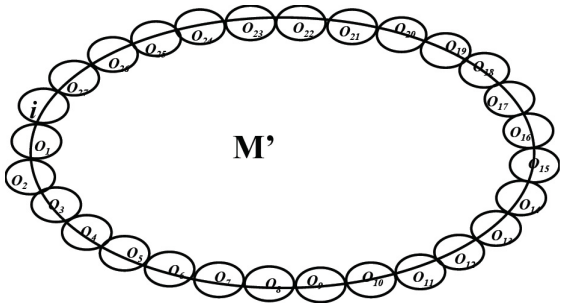


Fig. 6. Partitioning of $\mathcal{DS} = \{M', i, o_1, \ldots, o_{27}\}$

in Fig.2 and Fig.3 then we would have a node (X) such that $\mathcal{L}_P(X) = \{i, \neg o_1, \ldots, \neg o_{27}\}$ and $\mathcal{L}(X) = \{i, \neg E, \neg o_1, \ldots, \neg o_{27}, \geq 30M', \forall M'.E\}$. The $\bowtie$-Rule would add $\xi(\{M'\}, \geq, 30)$ to the set, $\mathcal{L}_E(X)$, of inequations encoding the nominals semantics.

$$\mathcal{L}_E(X) = \left\{ \begin{array}{l} \xi(\{i\}, \geq, 1), \xi(\{i\}, \leq, 1), \xi(\{o_1\}, \geq, 1), \xi(\{o_1\}, \leq, 1), \ldots, \\ \xi(\{o_{27}\}, \geq, 1), \xi(\{o_{27}\}, \leq, 1), \xi(\{M'\}, \geq, 30) \end{array} \right\}$$

The unsatisfiability of $\mathcal{L}_E(X)$ is immediately detected by the inequation solver considering that $v_{M'} \leq 0$ and $v_{M'o_1}, \ldots, v_{M'o_{27}}$ are all ≥ 1 as the initial distribution of nominals.

In comparison with standard tableau algorithms for $\mathcal{SHOQ}$ when checking the satisfiability of $\geq 30M'.EU$, 30 anonymous individuals are created and then non-deterministically identified with the 27 nominals. Considering that we have 30 individuals that need to be distributed over 27 there are $\frac{30!}{3!} = 4.420 * 10^{31}$ cases to be considered. In the case of the algebraic method, one would have to consider, in the worst case, $2^{28*2+1} - 1 = 1.441 * 10^{17} - 1$ cases for the *ch*-Rule until $v_{M'} \leq 0$ and $v_{M'o_1}, \ldots, v_{M'o_{27}}$ are all ≥ 1.

8.7 Possible Optimizations

In the literature there has always been a gap between the design of DL reasoning algorithms and their practical implementation. If not equipped with adequate optimizations, DL reasoning algorithms fail to be practical and we can safely assume that this is also the case with the algebraic tableau algorithm presented in this article.

In particular, the atomic decomposition technique comes with an exponential blow up of variables which, if naïvely treated by the *ch*-Rule gives a double exponential worst case algorithm. On the other hand, the algorithm seems amenable to the optimizations used in [12] such as dependency-directed backtracking, and the use of *"Don't care"* variables and default values. In particular, since many partitions will not be assigned any individuals, one can assume that all partitions are empty by setting all variables to zero and initialize them on demand only. For instance, in most of the cases the partitions including more than one nominal will be empty assuming that nominals are disjoint as with the EU example in Section 8.6. In a sense, the inequation solver only allocates and deals with non-zero variables and the variables not considered are assumed to be zero. Which means that in the average case the *ch*-Rule does not have to deal with an exponential number of variables. Default values for variables worked well with the $\mathcal{SHQ}$ prototype reasoner [13] which used a local atomic decomposition of role fillers and reported dramatic performance improvements in [12].

[13] The algebraic method was combined with dependency-directed backtracking and some other heuristics.

Another possible optimization is discussed in [25] such that if two roles do not share a sub-role or super-role one can omit the partitions where they intersect. This optimization is applicable to $\mathcal{SHOQ}$ when these roles do not interact with a nominal by setting these partitions to empty if a nominal is not included in the partition name. A possible drawback of this optimization is that the re-use of individuals becomes restricted to those intersecting with nominals and role fillers. Also, one could reduce a decomposed qualification over the same role into a single qualification due to the following : $\forall R.C \sqcap \forall R.D \Longleftrightarrow \forall R.(C \sqcap D)$. By doing this the size of Q_{N} can be reduced thus reducing the size of $\mathcal{P}$ and the number of variables as well. Additionally, if C and D are declared as disjoint ($C \sqsubseteq \neg D$), then one can safely ignore the partitions P such that $\{C, \neg D\} \subseteq P$.

Finally, it would be interesting to investigate if the form of caching enabled by the variables could be exploited to yield a single exponential algorithm as in [7,6]. It is part of future work to consider the applicability of these optimizations to the ongoing prototype implementation of the algebraic tableau algorithm presented in this article.

9 Conclusion

This article presents an algebraic tableau reasoning algorithm for $\mathcal{SHOQ}$. Unlike available reasoning algorithms for $\mathcal{SHOQ}$, the algebraic tableau method allows a calculus that is explicitly informed about the numerical restrictions on domain elements. This article not only extends our work in [8] to handle nominals as in [11] and GCIs, but also role hierarchies and role transitivity. The algebraic reasoning is based on the *atomic decomposition technique* which in this article is applied on a global decomposition set allowing the calculus to handle the various interactions between nominals, role fillers and their qualifications.

When creating an abstraction of a model, only one representative element is created for each partition and tagged by the partition signature. Using a representative element not only helps in reducing the size of the pre-model generated but also allows for re-using elements. Due to the re-use, the calculus naturally handles cyclic descriptions without the need for any blocking strategies to ensure termination.

It has been shown in [14,12] that extending a DL reasoning algorithm with an arithmetic component can dramatically improve the average case performance in the case of the DL $\mathcal{SHQ}$. We conjecture that the calculus presented in this article can enable similar performance improvements to the ones reported in [12,14] once equipped with adequate optimizations. In particular, it seems amenable to the optimizations discussed in Section 8.7. A first preliminary empirical evaluation [9] supports our conjecture and demonstrates the suitability of our calculus for significant speed improvements. It is part of ongoing work to report on a more detailed evaluation and extend the calculus to handle $\mathcal{SHOIQ}$ [17], an interesting combination of the DLs $\mathcal{SHIQ}$ and $\mathcal{SHOQ}$ where the interaction between nominals, QCRs and inverse roles ($\mathcal{I}$) becomes problematic.

Acknowledgement

We are grateful for the helpful comments by the anonymous reviewers.

References

[1] Areces, C. and C. Lutz, *Concrete domains and nominals united*, in: C. Areces, P. Blackburn, M. Marx and U. Sattler, editors, *Proceedings of the Fourth Workshop on Hybrid Logics (HyLo'02)* (2002), pp. 145–149.

[2] Baader, F., M. Buchheit and B. Hollunder, *Cardinality restrictions on concepts*, Artificial Intelligence **88** (1996), pp. 195–213.

[3] Baader, F., D. Calvanese, D. McGuinness, D. Nardi and P. Patel-Schneider, "The Description Logic Handbook, 2nd edition," Cambridge University Press, 2007.

[4] Borgida, A. and P. F. Patel-Schneider, *A semantics and complete algorithm for subsumption in the CLASSIC description logic*, Journal of Artificial Intelligence Research **1** (1994), pp. 277–308.

[5] Braüner, T., *Hybrid logic (Stanford Encyclopedia of Philosophy)* (2006), (see http://plato.stanford.edu/entries/logic-hybrid/).

[6] Ding, Y., "Tableau-based Reasoning for Description Logics with Inverse Roles and Number Restrictions," Ph.D. thesis, Concordia University, Montreal, Canada (2008).

[7] Donini, F. M. and F. Massacci, *EXPtime tableaux for $\mathcal{ALC}$*, Artificial Intelligence **124** (2000), pp. 87–138.

[8] Faddoul, J., N. Farsinia, V. Haarslev and R. Möller, *A hybrid tableau algorithm for $\mathcal{ALCQ}$*, in: *Proc. of the 2008 Int. Workshop on Description Logics, also in 18th European Conference on Artificial Intelligence (ECAI 2008)*, 2008, pp. 725–726.

[9] Faddoul, J. and V. Haarslev, *Optimizing algebraic tableau reasoning for $\mathcal{SHOQ}$: First experimental results*, in: *Proc. of the 2010 Int. Workshop on Description Logics, Waterloo, Canada, May 4-7*, 2010.

[10] Faddoul, J., V. Haarslev and R. Möller, *Hybrid reasoning for description logics with nominals and qualified number restrictions*, Technical report, Institute for Software Systems (STS), Hamburg University of Technology (2008), http://www.sts.tu-harburg.de/tech-reports/papers.html.

[11] Faddoul, J., V. Haarslev and R. Möller, *Algebraic tableau algorithm for $\mathcal{ALCOQ}$*, in: *Proceedings of the 2009 International Workshop on Description Logics (DL-2009), Oxford, United Kingdom, July 27–30*, 2009.

[12] Farsiniamarj, N. and V. Haarslev, *Practical reasoning with qualified number restrictions: A hybrid Abox calculus for the description logic $\mathcal{SHQ}$*, AI Communications (Special Issue on Practical Aspects of Automated Reasoning) (2009), 37 pages, in print.

[13] Haarslev, V. and R. Möller, *Optimizing reasoning in description logics with qualified number restrictions*, in: *Description Logics*, 2001, pp. 142–151.

[14] Haarslev, V., M. Timmann and R. Möller, *Combining tableaux and algebraic methods for reasoning with qualified number restrictions*, in: *Description Logics*, 2001, pp. 152–161.

[15] Hollunder, B. and F. Baader, *Qualifying number restrictions in concept languages*, in: *Proc. of the 2nd Int. Conference on Principles of Knowledge Representation and Reasoning,KR-91*, Boston (USA), 1991, pp. 335–346.

[16] Horrocks, I. and U. Sattler, *Ontology reasoning in the $\mathcal{SHOQ}(D)$ description logic*, in: *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)* (2001), pp. 199–204.

[17] Horrocks, I. and U. Sattler, *A tableaux decision procedure for $\mathcal{SHOIQ}$*, in: *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, 2005, pp. 448–453.

[18] Horrocks, I. and U. Sattler, *A tableaux decision procedure for $\mathcal{SHOIQ}$*, Journal of Automated Reasoning (2007), pp. 249–276.

[19] J. Hastad, J. L., B. Helfrich and C. P. Schnorr, "Polynomial time algorithms for finding integer relations among real numbers," Lecture Notes in Computer Science **210**, Publisher Springer Berlin / Heidelberg, 1986 pp. 105–118.

[20] Kazakov, Y. and B. Motik, *A resolution-based decision procedure for $\mathcal{SHOIQ}$*, Journal of Automated Reasoning **40** (2008), pp. 89–116.

[21] Motik, B. and I. Horrocks, *Individual reuse in description logic reasoning*, in: *IJCAR*, 2008, pp. 242–258.

[22] Motik, B., P. F. Patel-Schneider and B. Cuenca Grau, *OWL 2 web ontology language: Direct semantics* (2009), latest version available at http://www.w3.org/TR/owl2-semantics/.

[23] Motik, B., R. Shearer and I. Horrocks, *Optimized reasoning in description logics using hypertableaux*, in: *(CADE-21)*, Lecture Notes in Artificial Intelligence **4603** (2007), pp. 67–83.

[24] Ohlbach, H. J. and J. Koehler, *Reasoning about sets via atomic decomposition*, Technical Report TR-96-031, International Computer Science Institute (ICSI), Berkeley, CA, USA (1996), 41 pages.

[25] Ohlbach, H. J. and J. Koehler, *Modal logics, description logics and arithmetic reasoning*, Artificial Intelligence **109** (1999), pp. 1–31.

[26] Parsia, B., B. Cuenca Grau and E. Sirin, *From wine to water: Optimizing description logic reasoning for nominals*, in: *Proc. of the 10th Int. Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2006, pp. 90–99.

[27] Schaerf, A., *Reasoning with individuals in concept languages*, Data and Knowledge Engineering **13** (1994), pp. 141–176.

[28] Tobies, S., *The complexity of reasoning with cardinality restrictions and nominals in expressive description logics*, Journal of Artificial Intelligence Research **12** (2000), pp. 199–217.