Automated Reasoning in $\mathcal{ALCQ}$ via SMT

Volker Haarslev*, Roberto Sebastiani+, and Michele Vescovi+

* CSE, Concordia University, Montreal, haarslev@cse.concordia.ca
+ DISI, Università di Trento, {rseba, vescovi}@disi.unitn.it

Abstract. Reasoning techniques for qualified number restrictions (QNRs) in Description Logics (DLs) have been investigated in the past but they mostly do not make use of the arithmetic knowledge implied by QNRs. In this paper we propose and investigate a novel approach for concept satisfiability in acyclic $\mathcal{ALCQ}$ ontologies. It is based on the idea of encoding an $\mathcal{ALCQ}$ ontology into a formula in Satisfiability Modulo the Theory of Costs (SMT($\mathcal{C}$)), which is a specific and computationally much cheaper subcase of Linear Arithmetic under the Integers, and to exploit the power of modern SMT solvers to compute every concept-satisfiability query on a given ontology. We implemented and tested our approach, which includes a very effective individuals-partitioning technique, on a wide set of synthesized benchmark formulas, comparing the approach with the main state-of-the-art DL reasoners available. Our empirical evaluation confirms the potential of the approach.

1 Introduction

Description logics (DLs) form one of the major foundations of the semantic web and its web ontology language (OWL). In fact, OWL 2, a recent W3C recommendation, is a syntactic variant of a very expressive DL that supports reasoning with so-called *qualified number restrictions* (QNRs). A sound and complete calculus for reasoning with the DL $\mathcal{ALCQ}$ that adds QNRs to the basic DL $\mathcal{ALC}$ was first proposed in [9]. For example, this calculus decides the satisfiability of an $\mathcal{ALCQ}$ concept $(\geq 5\, s.C \sqcap \geq 5\, s.D \sqcap \leq 2\, s.E)$ by trying to find a model with fillers for the role s such that at least 5 fillers are instances of C, at least 5 fillers are instances of D, and at most 2 fillers are instances of E. It satisfies the at-least restrictions by creating 10 fillers for S, 5 of which are instances of C and 5 are instances of D. A concept choose rule non-deterministically assigns E or $\neg E$ to these fillers. In case the at-most restriction $(\leq 2\, s.E)$ is violated a merge rule non-deterministically merges pairs of fillers for s that are instances of E [9]. Searching for a model in such an arithmetically uninformed way can become very inefficient especially when bigger numbers occur in QNRs or several QNRs interact. To the best of our knowledge this calculus still serves as reference in most tableau-based OWL reasoners (e.g., Pellet [15], FaCT++ [16]) for implementing reasoning about QNRs. The only exception is Racer [7] where conceptual QNR reasoning is based on an algebraic approach [8] that integrates integer linear programming with DL tableau methods.

The work presented in this paper was inspired by two recent novel approaches, combined with the progress in satisfiability modulo theory (SMT) solving techniques. First, [13,14] explored the idea of performing automated reasoning tasks in DLs by

$\bot^{\mathcal{I}} = \emptyset$, $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$, $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$,
$(\exists r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{there exists } y \in \Delta^{\mathcal{I}} \text{ s.t. } (x,y) \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$,
$(\forall r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{for all } y \in \Delta^{\mathcal{I}} \text{ s.t. } (x,y) \in r^{\mathcal{I}} \text{ then } y \in C^{\mathcal{I}}\}$,
$(\geq nr.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid |FIL(r,x) \cap C^{\mathcal{I}}| \geq n\}$,
$(\leq mr.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid |FIL(r,x) \cap C^{\mathcal{I}}| \leq m\}$, $C \sqsubseteq D$ is satisfied iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

Fig. 1: Syntax and semantics of $\mathcal{ALCQ}$ ($n \geq 1$ and $m \geq 0$).

encoding problems into Boolean formulas and by exploiting the power of modern SAT techniques. In particular, the experiments in [13] showed that, in practice and despite the theoretical worst-case complexity limits, this approach could handle most or all the $\mathcal{ALC}$ satisfiablity problems which also the other approaches could handle, with performances which were comparable with, and often better than, those of state-of-the-art tools. Second, a revised and extended algebraic approach was presented for $\mathcal{SHQ}$ [6] and $\mathcal{SHOQ}$ [4]. These approaches represent knowledge about interacting QNRs as systems of linear inequations where numerical variables represent cardinalities of sets of domain elements (e.g., role fillers) divided into mutually disjoint decompositions. On a set of synthetic QNR benchmarks these algebraic approaches demonstrated a superior performance for most test cases [6,5].

The main idea of this paper is thus to encode an $\mathcal{ALCQ}$ ontology into a formula in Satisfiability Modulo the Theory of Costs (SMT($\mathcal{C}$)) [3], which is a specific and computationally much cheaper subcase of Linear Arithmetic under the Integers ($\mathcal{LA}(\mathbb{Z})$), and to exploit the power of modern SMT solvers to compute every concept-satisfiability query on a given ontology. We have implemented and tested our approach (called $\mathcal{ALCQ}2SMT_{\mathcal{C}}$) that includes a very effective individuals-partitioning technique on a wide set of synthesized benchmark formulas and compared it with main state-of-the-art OWL reasoners. Our empirical evaluation demonstrates the potential of our approach and, compared with the tested OWL reasoners, demonstrates a significantly better performance in the case of benchmarks having multiple/balanced sources of complexity.

2 Background

2.1 The Description Logic $\mathcal{ALCQ}$

The logic $\mathcal{ALCQ}$ extends the well-known logic $\mathcal{ALC}$ by adding *qualified number restrictions* (QNRs). In more details, the *concept descriptions* in $\mathcal{ALCQ}$ (namely $\hat{C}, \hat{D}, \ldots$) are inductively defined through the constructors listed in Figure 1, starting from the non-empty and pair-wise disjoint sets of *concept names* N_C (denoted by the letters $A, B, C, \ldots$) and *role names* N_R (denoted by the letters $r, s, \ldots$). It allows for negations, conjunctions/disjunctions, existential/universal restrictions and, indeed, QNRs. An $\mathcal{ALCQ}$ *TBox* (or *ontology*) is a finite set of general concept inclusion (GCI) axioms as defined in Figure 1.

Given a TBox $\mathcal{T}$, we denote with $\mathsf{BC}_{\mathcal{T}}$ the set of the *basic concepts* for $\mathcal{T}$, i.e. the smallest set of concepts containing: (i) the top and the bottom concepts $\top$ and $\bot$; (ii) all the concepts of $\mathcal{T}$ in the form C and $\neg C$ where C is a concept name in N_C. We denote

the basic concepts in $\mathsf{BC}_\mathcal{T}$ with the letters $C, D, \ldots$ (thus, C may represent a concept $\neg C'$ with $C' \in \mathsf{BC}_\mathcal{T}$), whilst we use $\hat{C}, \hat{D}, \ldots$ for complex concepts, i.e. $\hat{C}, \hat{D} \notin \mathsf{BC}_\mathcal{T}$. Our approach is currently restricted to *acyclic* (or *unfoldable*) TBoxes. We call a TBox $\mathcal{T}$ *acyclic* if there exist no cyclic dependencies between its concept names, i.e., named concepts are neither defined directly or indirectly in terms of themselves through the axioms in $\mathcal{T}$.

Semantics. The semantics of $\mathcal{ALCQ}$ is defined in terms of *interpretations*. An interpretation $\mathcal{I}$ is a couple $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is the domain (i.e. a non-empty set of individuals), and $\cdot^\mathcal{I}$ is the interpretation function which maps each concept name (atomic concept) $A \in N_C$ to a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and maps each role name (atomic role) r to a binary relation $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. In Figure 1 the inductive extensions of $\cdot^\mathcal{I}$ to arbitrary concept descriptions are defined, where n and m are positive integer values and $FIL(r, x)$ is the set of the r-fillers of the individual $x \in \Delta^\mathcal{I}$ for the role $r \in N_R$ and is defined as $FIL(r, x) = \{y \in \Delta^\mathcal{I} | (x, y) \in r^\mathcal{I}\}$. An interpretation $\mathcal{I}$ is a *model* of a given TBox $\mathcal{T}$ if and only if the conditions given in Figure 1 are respected for every axiom in $\mathcal{T}$; when this is the case, the TBox $\mathcal{T}$ is said to be *consistent*. A concept $\hat{C}$ is said to be *satisfiable* wrt. $\mathcal{T}$ if and only if there exists a model $\mathcal{I}$ of $\mathcal{T}$ with $\hat{C}^\mathcal{I} \neq \emptyset$, i.e. there exists an individual $x \in \Delta^\mathcal{I}$ as an instance of $\hat{C}$, i.e. such that $x \in \hat{C}^\mathcal{I}$.

Normal Form. We assume wlog. that all $\mathcal{ALCQ}$ concept descriptions are in *negative normal form* (NNF), i.e. negation signs only occurs in front of concept names (see [17] for details). Then, for the sake of an easier exposition, we restrict our attention to those $\mathcal{ALCQ}$ TBoxes in which all axioms are in the following normal form:

$$C \sqsubseteq D \qquad \sqcap_i C_i \sqsubseteq D \qquad C \sqsubseteq \sqcap_i D_i \qquad \Re r.C \sqsubseteq D \qquad C \sqsubseteq \Re r.D \tag{1}$$

with $\Re \in \{\forall, \geq n, \leq m\}$ s.t. $n, m \geq 1$, and $C, C_i, D, D_i \in \mathsf{BC}_\mathcal{T}$. [1] Every given TBox $\mathcal{T}$ can be turned into a normalized TBox $\mathcal{T}'$ (where all concept description in $\mathcal{T}'$ are in NNF) that is a conservative extension of $\mathcal{T}$ by introducing new concept names. The transformation of a TBox $\mathcal{T}$ into $\mathcal{T}'$ can be done in linear time, and the size of $\mathcal{T}'$ is linear wrt. the size of $\mathcal{T}$. We call every non-conjunctive and non-disjunctive concept description occurring in the concept inclusions of $\mathcal{T}'$ a *normal concept* of a normalized TBox $\mathcal{T}'$; we call $\mathsf{NC}_{\mathcal{T}'}$ the set of all the normal concepts of $\mathcal{T}'$. For more details we refer the reader to [17].

2.2 Satisfiablity Modulo Theory with Cost Functions

Satisfiability Modulo (the) Theory $\mathcal{T}$, *SMT*($\mathcal{T}$), is the problem of deciding the satisfiability of a (typically) ground formula under a background theory $\mathcal{T}$. Most state-of-the art SMT solvers are based on the *lazy SMT schema*: in a nutshell, a SAT solver is used to search for a truth assignment μ to the atomic subformulas of the input ground formula φ, s.t. μ tautologically entails φ and μ is found consistent in $\mathcal{T}$ by the $\mathcal{T}$*-solver*. (We refer the reader to, e.g., [12] for details and further references.)

[1] In particular, we avoid redundant existential and at-most restrictions that are replaced by their following equivalents: $\exists r.C \Longrightarrow \geq 1 r.C$ and $\leq 0 r.C \Longrightarrow \forall r.nnf(\neg C)$.

The work in [3] addresses the problem of the satisfiability in some theory $\mathcal{T}$ of a formula φ augmented with a set of *cost functions* $\{cost_1, ..., cost_N\}$ s.t., for every i:

$$cost_i = \sum_{j=1}^{N_i} \textit{if-then-else}(A_{ij}, c_{ij}, 0), \quad lb_i < cost_i \leq ub_i, \tag{2}$$

A_{ij} being Boolean atoms occurring in φ, and N_i, lb_i, ub_i, c_{ij} being integer values ≥ 0. (Intuitively, in (2) $cost_i = \sum_j A_{ij}c_{ij}$ s.t. $A_{ij} \in \{0, 1\}$.) The problem can be encoded into SMT($\mathcal{T} \cup \mathcal{LA}(\mathbb{Z})$). However, [3] remarked the inefficiency of such solution, which does not fully exploit the fact that the values of $cost_i$ derive deterministically from the truth values of all the A_{ij}'s. They proposed instead a specific *theory of costs* $\mathcal{C}$, which is much simpler and computationally much cheaper than $\mathcal{LA}(\mathbb{Z})$, and developed a specific very-fast $\mathcal{T}$-solver for $\mathcal{C}$. In a nutshell, $\mathcal{C}$ consists of: (i) a collection of integer variables $cost_1, \ldots, cost_N$, that we call *cost variables*, denoting the output of the cost functions in (2); (ii) an interpreted predicate BC "*bound cost*" s.t. $\mathsf{BC}(cost_i, c)$ is true iff $cost_i$ is upper-bounded by the integer value c; (i.e., iff $cost_i \leq c$); (iii) an interpreted predicate IC "*incur cost*" s.t. $\mathsf{IC}(cost_i, c_{ij}, j)$ is true if the j-th element of sum (2) is c_{ij}, false if it is 0. Thus, φ is satisfiable in $\mathcal{T}$ under the cost constraints (2) iff the formula

$$\varphi \wedge \textstyle\bigwedge_{i=1}^{N}(\mathsf{BC}(cost_i, ub_i) \wedge \neg\mathsf{BC}(cost_i, lb_i) \wedge \bigwedge_{j=1}^{N_i}(A_{ij} \leftrightarrow \mathsf{IC}(cost_i, c_{ij}, j))) \tag{3}$$

is satisfiable in $\mathcal{T} \cup \mathcal{C}$. A specific $\mathcal{T}$-solver for $\mathcal{C}$ works simply by adding the value c_{ij} [resp. 0] to the current minimum value of $cost_i$ and 0 [resp. c_{ij}] to its current maximum when $\mathsf{IC}(cost_i, c_{ij}, j)$ (i.e. A_{ij}) is assigned to true [resp. false], and by checking if such minimum [resp. maximum] value of $cost_i$ is smaller or equal than ub_i [resp. greater or equal than lb_i]. We refer the reader to [3] for details and further references.

3 Concept Satisfiability via SMT with Costs

3.1 Encoding $\mathcal{ALCQ}$ into SMT($\mathcal{C}$)

The encoding we propose simulates the construction of an interpretation $\mathcal{I}$ by introducing new individuals, assigning individuals to the interpretations of concepts in $\mathcal{T}$, and counting their occurrences in the interpretations. We represent uniquely *individuals* in $\Delta^{\mathcal{I}}$ by means of *labels* σ, represented as non-empty sequences of positive integer values and role names in N_R. A label σ can be either the label 1 or in the form $\sigma'.r.n$, with σ' another label, $r \in N_R$ and $n \geq 1$. With a small abuse of notation, hereafter we may say "the individual σ" meaning "the individual labeled by σ". Moreover, we call *instantiated concept* a pair $\langle \sigma, C \rangle$, s.t. $\sigma \in \Delta^{\mathcal{I}}$ and C is an $\mathcal{ALCQ}$ normal concept of $\mathcal{T}$, representing the fact that the σ is an instance of C in the interpretation $\mathcal{I}$, i.e. $\sigma \in C^{\mathcal{I}}$.

We define $A_{\langle\ ,\ \rangle}$ an injective function which maps one instantiated concept $\langle \sigma, C \rangle$ s.t. C is not in the form $\neg C'$, into a Boolean variable $A_{\langle \sigma,\ C \rangle}$ that we call *concept variable*. The so-called *concept literal* $L_{\langle \sigma,\ C \rangle}$, denotes $\neg A_{\langle \sigma,\ C' \rangle}$ if C is in the form $\neg C'$, $A_{\langle \sigma,\ C \rangle}$ otherwise. The truth value of $L_{\langle \sigma,\ C \rangle}$ states whether the instantiation relation between σ and C [resp. $\neg C$] holds, i.e. if $\langle \sigma, C \rangle$ [resp. $\langle \sigma, \neg C \rangle$] is an existing instantiated concept in $\mathcal{I}$. We conventionally assume that $A_{\langle \sigma,\ \bot \rangle}$ is $\bot$. Notice also that $\langle \sigma, \top \rangle$

means $\sigma \in \Delta^{\mathcal{I}}$, i.e. that if $A_{\langle \sigma, \top \rangle}$ is assigned to true then σ exists in $\Delta^{\mathcal{I}}$. We informally say that σ (meaning $\langle \sigma, \top \rangle$) or $\langle \sigma, C \rangle$ is "enabled" when the respective literal is assigned to true.

We define indiv a function which maps one instantiated concept $\langle \sigma, \Re r.C \rangle$, such that $\Re \in \{\geq n, \leq m\}$ and C is a basic concept (since we are considering concepts in normal form), into a cost variable $\mathsf{indiv}^C_{\sigma.r}$ in the Theory of Costs, that we call *individuals cost variable*. Notice that indiv is not injective since the same cost variable $\mathsf{indiv}^C_{\sigma.r}$ is "shared" among all the instantiated concepts which refer both to the same σ and to QNRs involving the same r and C. However, notice also that $\langle \sigma, \Re r.C \rangle$ and $\langle \sigma, \Re r.\neg C \rangle$ are mapped to different cost variables. The final value of the individuals cost variable $\mathsf{indiv}^C_{\sigma.r}$ represents the number of individuals which are in relation with the individual σ via the role r and are in the interpretation of C, in other words the final value of $\mathsf{indiv}^C_{\sigma.r}$ exactly represents the cardinality of $FIL(r, \sigma) \cap C^{\mathcal{I}}$.

Our encoding works by means of the following principles:

- GCIs are represented via Boolean implications between instantiated concepts.
- Every at-least restriction $\langle \sigma, \geq nr.C \rangle$ is handled by introducing exactly n individuals $\sigma.r.i$ associated to C. The existence of individuals is forced by binding each of them to an incur cost of value 1 for $\mathsf{indiv}^C_{\sigma.r}$, and then fixing a lower-bound for $\mathsf{indiv}^C_{\sigma.r}$.
- When both at-least and at-most restrictions coexist wrt. σ, the encoding allows for sharing individuals separately introduced by distinct at-least restrictions. At-most restrictions are handled by fixing upper-bounds for the respective cost variables.
- It mimics the construction of a labeled tableaux with the difference of the above exposed sharing of individuals which generalizes the merging of pairs of fillers to satisfy at-most QNR.

Definition 1 (**$\mathcal{ALCQ}2SMT_{\mathcal{C}}(\mathcal{T})$ encoding).** *Let $\mathcal{T}$ be an acyclic $\mathcal{ALCQ}$ TBox in normal form. Wlog., we represent every axiom $\hat{C} \sqsubseteq \hat{D}$ of $\mathcal{T}$ as $\sqcap_i \hat{C}_i \sqsubseteq \sqcup_j \hat{D}_j$ where $i, j \geq 1$ and $i = 1$ (resp. $j = 1$), with $\hat{C}_1$ (resp. $\hat{D}_1$) a normal concept, for every normal form* (1) *except for the second (resp. the third) one. The SMT($\mathcal{C}$) encoding $\mathcal{ALCQ}2SMT_{\mathcal{C}}(\mathcal{T})$ for $\mathcal{T}$ is defined as the sextuple $\langle \Sigma^{\mathcal{T}}, \mathcal{I}^{\mathcal{T}}_{-}, \mathcal{I}^{\mathcal{T}}_{+}, A_{\langle\ ,\ \rangle}, \mathsf{indiv}, \varphi^{\mathcal{T}} \rangle$, where:*

- *$\Sigma^{\mathcal{T}}$ is the set of all the possible individuals introduced;*
- *$\mathcal{I}^{\mathcal{T}}_{-}, \mathcal{I}^{\mathcal{T}}_{+}$ represent respectively the set of the implicant (i.e. left-side) and implied (i.e. right-side) instantiated concepts that must be encoded accordingly to their side;*
- *$A_{\langle\ ,\ \rangle}$ and indiv are the functions defined above;*
- *$\varphi^{\mathcal{T}}$ is a CNF formula on propositional- and $\mathcal{C}$-literals encoding $\mathcal{T}$ into SMT($\mathcal{C}$). We represent $\varphi^{\mathcal{T}}$ as the set of its clauses.* [2]

The sets $\Sigma^{\mathcal{T}}, \mathcal{I}^{\mathcal{T}}_{-}, \mathcal{I}^{\mathcal{T}}_{+}$ and $\varphi^{\mathcal{T}}$ are incrementally defined as the minimum sets s.t.:

1. ***Initialization.*** *$1 \in \Sigma^{\mathcal{T}}$, $\langle 1, \top \rangle \in \mathcal{I}^{\mathcal{T}}_{-}$, $\langle 1, \top \rangle \in \mathcal{I}^{\mathcal{T}}_{+}$ and $(A_{\langle 1, \top \rangle}) \in \varphi^{\mathcal{T}}$.*
2. ***Axioms initialization.*** *If $\hat{C} \sqsubseteq \hat{D} \in \mathcal{T}$, then $\{\langle 1, C_i \rangle \mid \hat{C} = \sqcap_i C_i\} \subseteq \mathcal{I}^{\mathcal{T}}_{-}$.*

[2] For better readability we often represent the clauses of $\varphi^{\mathcal{T}}$ as implications.

3. ***Axioms expansion.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\sqcap_i C_i \sqsubseteq \sqcup_j D_j \in \mathcal{T}$, $\{\langle \sigma, C_i\rangle \mid \hat{C} = \sqcap_i C_i\} \subseteq \mathcal{I}^{\mathcal{T}}_- \cup \mathcal{I}^{\mathcal{T}}_+$, *then*

$$\{\langle \sigma, D_j\rangle \mid \hat{D} = \sqcup_j D_j\} \subseteq \mathcal{I}^{\mathcal{T}}_+,$$
$$(\bigwedge_i L_{\langle \sigma,\, C_i\rangle}) \rightarrow (\bigvee_j L_{\langle \sigma,\, D_j\rangle}) \in \varphi^{\mathcal{T}}. \tag{4}$$

4. ***Handle left-side QNRs.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \Re' r.C'\rangle \in \mathcal{I}^{\mathcal{T}}_+$ *with* $\Re' \in \{\geq n', \leq m', \forall\}$, *then*

$$\{\langle \sigma, \Re r.C\rangle \mid \Re r.C \sqsubseteq \hat{D} \in \mathcal{T}\} \subseteq \mathcal{I}^{\mathcal{T}}_-, \Re \in \{\geq n, \leq m, \forall\}.$$

5. ***At-least restrictions: introduce individuals.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \geq nr.C\rangle \in \mathcal{I}^{\mathcal{T}}_+$ *then*

$$\{\sigma.r.k^C_i \mid i = 1, \ldots, n\} \subset \Sigma^{\mathcal{T}},$$
$$\{\langle \sigma.r.k^C_i, C\rangle \mid i = 1, \ldots, n\} \cup \{\langle \sigma.r.k^C_i, \top\rangle \mid i = 1, \ldots, n\} \subset \mathcal{I}^{\mathcal{T}}_-,$$
$$\{\mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, 1, k^C_i) \rightarrow L_{\langle \sigma.r.k^C_i,\, C\rangle} \mid i = 1, \ldots, n\} \subset \varphi^{\mathcal{T}}, \tag{5}$$
$$\{\mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, 1, k^C_i) \rightarrow A_{\langle \sigma.r.k^C_i,\, \top\rangle} \mid i = 1, \ldots, n\} \subset \varphi^{\mathcal{T}}, \tag{6}$$

where $k^C_1 \geq 1$, $k^C_{i+1} = k^C_i + 1$ *and* $k^C_i \neq k^D_j$ *for every* $\langle \sigma, \geq n'r.D\rangle \in \mathcal{I}^{\mathcal{T}}_+$ *with* $C \neq D$ *and* $i = 1, ..., n$, $j = 1, ..., n'$. *We assume consecutive values for all the* $\sigma.r.j$. [3]

6. ***At-least restrictions: fix lower bounds.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \geq nr.C\rangle \in \mathcal{I}^{\mathcal{T}}_+$, *then*

$$((A_{\langle \sigma,\, \geq nr.C\rangle} \wedge A_{\langle \sigma,\, \top\rangle}) \rightarrow \neg\mathsf{BC}(\mathsf{indiv}^C_{\sigma.r}, n-1)) \in \varphi^{\mathcal{T}}, \tag{7}$$

if $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \geq nr.C\rangle \in \mathcal{I}^{\mathcal{T}}_-$, *then*

$$((\neg\mathsf{BC}(\mathsf{indiv}^C_{\sigma.r}, n-1) \wedge A_{\langle \sigma,\, \top\rangle}) \rightarrow A_{\langle \sigma,\, \geq nr.C\rangle}) \in \varphi^{\mathcal{T}}. \tag{8}$$

7. ***Coexisting at-least/at-most: sharing individuals.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \leq mr.E\rangle \in \mathcal{I}^{\mathcal{T}}_+$, $\langle \sigma, \geq nr.C\rangle \in \mathcal{I}^{\mathcal{T}}_+$, $\langle \sigma, \geq n'r.D\rangle \in \mathcal{I}^{\mathcal{T}}_+$, *with* $C \neq D$, *then*

$$\{\langle \sigma.r.k^C_i, D\rangle \mid i = 1, \ldots, n\} \cup \{\langle \sigma.r.k^D_i, C\rangle \mid i = 1, \ldots, n'\} \subset \mathcal{I}^{\mathcal{T}}_-,$$
$$\{\mathsf{IC}(\mathsf{indiv}^D_{\sigma.r}, 1, k^C_i) \rightarrow L_{\langle \sigma.r.k^C_i,\, D\rangle} \mid i = 1, \ldots, n\} \cup$$
$$\{\mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, 1, k^D_i) \rightarrow L_{\langle \sigma.r.k^D_i,\, C\rangle} \mid i = 1, \ldots, n'\} \subset \varphi^{\mathcal{T}}, \tag{9}$$
$$\{\mathsf{IC}(\mathsf{indiv}^D_{\sigma.r}, 1, k^C_i) \rightarrow A_{\langle \sigma.r.k^C_i,\, \top\rangle} \mid i = 1, \ldots, n\} \cup$$
$$\{\mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, 1, k^D_i) \rightarrow A_{\langle \sigma.r.k^D_i,\, \top\rangle} \mid i = 1, \ldots, n'\} \subset \varphi^{\mathcal{T}}. \tag{10}$$

8. ***At-most restrictions: count individuals.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \leq mr.C\rangle \in \mathcal{I}^{\mathcal{T}}_+$ *then*

$$\{\langle \sigma.r.j, C\rangle \mid \sigma.r.j \in \Sigma^{\mathcal{T}}\} \subset \mathcal{I}^{\mathcal{T}}_-,$$
$$\{(L_{\langle \sigma.r.j,\, C\rangle} \wedge A_{\langle \sigma.r.j,\, \top\rangle}) \rightarrow \mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, 1, j) \mid \sigma.r.j \in \Sigma^{\mathcal{T}}\} \subset \varphi^{\mathcal{T}}. \tag{11}$$

[3] Hence, either $k^C_1 = 1$ or $k^C_1 = k^D_{n'} + 1$ for some $\langle \sigma, \geq n'r.D\rangle \in \mathcal{I}^{\mathcal{T}}_+$

9. ***At-most restrictions: fix upper bounds.*** *If* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \leq mr.C \rangle \in \mathcal{I}_{+}^{\mathcal{T}}$, *then*

$$((A_{\langle \sigma,\ \leq mr.C \rangle} \wedge A_{\langle \sigma,\ \top \rangle}) \rightarrow \mathsf{BC}(\mathsf{indiv}_{\sigma.r}^{C}, m)) \ \in \ \varphi^{\mathcal{T}}, \tag{12}$$

if $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \leq mr.C \rangle \in \mathcal{I}_{-}^{\mathcal{T}}$, *then*

$$((\mathsf{BC}(\mathsf{indiv}_{\sigma.r}^{C}, m) \wedge A_{\langle \sigma,\ \top \rangle}) \rightarrow A_{\langle \sigma,\ \leq mr.C \rangle}) \ \in \ \varphi^{\mathcal{T}}. \tag{13}$$

10. ***Universal restrictions.*** *if* $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \forall r.C \rangle \in \mathcal{I}_{+}^{\mathcal{T}}$, *then*

$$\begin{gathered} \{\ \langle \sigma.r.j, C \rangle \mid \sigma.r.j \in \Sigma^{\mathcal{T}}\ \} \ \subset \ \mathcal{I}_{-}^{\mathcal{T}} \\ \{\ ((A_{\langle \sigma,\ \forall r.C \rangle} \wedge A_{\langle \sigma.r.j,\ \top \rangle}) \rightarrow L_{\langle \sigma.r.j,\ C \rangle}) \mid \sigma.r.j \in \Sigma^{\mathcal{T}}\ \} \ \subset \ \varphi^{\mathcal{T}}, \end{gathered} \tag{14}$$

if $\sigma \in \Sigma^{\mathcal{T}}$, $\langle \sigma, \forall r.C \rangle \in \mathcal{I}_{-}^{\mathcal{T}}$, *then*

$$((\mathsf{BC}(\mathsf{indiv}_{\sigma.r}^{\neg C}, 0) \wedge A_{\langle \sigma,\ \top \rangle}) \rightarrow A_{\langle \sigma,\ \forall r.C \rangle}) \ \in \ \varphi^{\mathcal{T}}. \tag{15}$$

Importantly, at the effect of the encoding, left-side at-most (and universal) restrictions behave as right-side at-least restrictions, and vice versa. Thus, for instance, the instantiated concept $\langle \sigma, \leq n-1r.C \rangle \ \in \ \mathcal{I}_{-}^{\mathcal{T}}$ [resp. $\langle \sigma, \forall r.\neg C \rangle \ \in \ \mathcal{I}_{-}^{\mathcal{T}}$, $n \stackrel{def}{=} 1$] must be handled by the encoding as if it were the instantiated concept $\langle \sigma, \geq nr.C \rangle \ \in \ \mathcal{I}_{+}^{\mathcal{T}}$. In order to simplify the exposition, in Definition 1 and afterwards, we generically refer to at-least/at-most restrictions (respectively to the instantiated concepts $\langle \sigma, \geq nr.C \rangle$/ $\langle \sigma, \leq mr.C \rangle$) meaning the right-side ones, but implicitly including left-side at-most (or universal)/at-least restrictions, respectively. The interested reader can find in [17] the complete $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ encoding and some encoding examples.

The following facts concerning $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ hold. (We refer the reader to [17] for the formal proofs.)

Theorem 1. *An $\mathcal{ALCQ}$ acyclic TBox $\mathcal{T}$ in normal form is consistent if and only if the SMT($\mathcal{C}$)-formula $\varphi^{\mathcal{T}}$ of $\mathcal{ALCQ}2SMT_{\mathcal{C}}(\mathcal{T})$ (Definition 1) is satisfiable.*

Theorem 2. *Given an $\mathcal{ALCQ}$ acyclic TBox $\mathcal{T}$ in normal form and the encoding $\mathcal{ALCQ}2SMT_{\mathcal{C}}(\mathcal{T}) = \langle \Sigma^{\mathcal{T}}, \mathcal{I}_{-}^{\mathcal{T}}, \mathcal{I}_{+}^{\mathcal{T}}, A_{\langle\ ,\ \rangle}, \mathsf{indiv}, \varphi^{\mathcal{T}} \rangle$ of Definition 1, every $C \in \mathsf{BC}_{\mathcal{T}}$ is satisfiable wrt. $\mathcal{T}$ iff $\varphi^{\mathcal{T}} \wedge L_{\langle 1,\ C \rangle}$ is satisfiable.*

We remark on some facts about the encoding of Definition 1:

- Point 4. is necessary to force the encoding of axioms having on the left-hand side restrictions wrt. the role r, when other restrictions wrt. r are involved. Such kind of axioms can create cycles in TBoxes (we remark that our encoding ensures termination for acyclic TBoxes).
- In all the clauses of type (5), (6), (9), (10) and (5), (11), every IC-literal has cost value 1 and the same index of the bound individual. This ensures that IC-literals referring to distinct individuals/cost variables are represented by distinct atoms.

– Due to the theory $\mathcal{C}$ clauses (7) and (12), are those concretely ensuring the numerical satisfiability of both at-least and at-most restrictions. In order to be satisfied: (i) a clause of type (7) forces some IC-literals to be assigned to true (thus (5), (9) work in only one direction); (ii) a clause of type (12), instead, bounds the number of IC-literals that can be enabled (motivating the opposite direction of (11)). Clauses (8) and (13) instead enforce the application of an axiom having a left-side QNRs if it is numerically satisfied.

Notice that if, for the same σ, r and C, more than one restriction satisfies the conditions of point 5. with different values of n (being n^* the highest of these values), then only exactly n^* new individuals and n^* clauses (5) and (6) are in $\varphi^{\mathcal{T}}$. In contrast, one distinct clause (7) is in $\varphi^{\mathcal{T}}$ for every different value of n (the same holds for the clauses (12) in case of different values of m wrt. the same σ, r and C).

$\mathcal{ALCQ}$ with general TBoxes has the finite tree model property [11], thus every satisfiable $\mathcal{ALCQ}$ concept is satisfiable in a finite interpretation (in this case of worst-case exponential size) which has the shape of a tree. Intuitively the individuals in $\Sigma^{\mathcal{T}}$ form a super-tree of all such models. Let $\mathcal{N}$ represent the sum of the values occurring in the QNRs of $\mathcal{T}$: a very coarse upper bound to the cardinality of $\Sigma^{\mathcal{T}}$ is $\Theta(|\mathcal{T}|^{\mathcal{N}})$, in fact the number of nested restrictions is bounded by the number of axioms of $\mathcal{T}$ while $\mathcal{N}$ bounds the number of branches in the tree for every nesting level. The size of $\varphi^{\mathcal{T}}$ is, instead, bounded by $\Theta(|\mathcal{T}|^{2\mathcal{N}})$ because for every individual and every concept of $\mathcal{T}$ a fixed number of clauses can be introduced. In [17] we define a terminating queue-based algorithm building $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ by means of expansion rules which mimic Definition 1. Since we are restricted to acyclic TBoxes it is ensured that our encoding algorithm terminates even without introducing blocking techniques [1], in particular, the proposed algorithm is polynomial in the size of the SMT($\mathcal{C}$) formula produced.

4 Partitioning Individuals

One potential drawback of the basic $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ is the high number of individuals introduced, that is linear wrt. the values occurring in the at-least restrictions. This number can increase exponentially when nested restrictions must be encoded, significantly impacting on the size and on the hardness of the resulting SMT($\mathcal{C}$) formula. However, similarly to the hybrid approach of [6,4], we can cope with this problem by encoding groups of individuals having identical properties (instead of using single ones) and by using only one "proxy" individual as representative of the group. We aim at partitioning the individuals introduced in Definition 1 on the basis of the following considerations: [4]

– Individuals are naturally pre-partitioned in groups wrt. r and the predecessor σ.

[4] Notice that here we present a *different* partitioning that avoids the a-priori exponential number of partitions in [6,4] (wrt. the number of coexisting QNRs). In our case, we consider the whole set of individuals necessary to trivially satisfy all the coexisting at-least restrictions, then, only on the basis of the numbers involved in QNRs, we compute a partitioning of such a set, where the target of our approach is to decide which partitions of individuals belong to a concept interpretation.

– If, given σ, r, no at-most restriction exists, all the fillers $\sigma.r.k_i^C$ referring to one at-least restriction can be represented by one single proxy individual.
– Otherwise, the $\sum_j n_j$ distinct individuals introduced by some $\langle \sigma, \geq n_j r.C_j \rangle$ can still be partitioned, but the partitioning must allow for representing possible intersections between the $C_j^{\mathcal{I}}$.

In the latter case not all possible cardinalities of the intersections must be considered. Instead, it is sufficient to distinguish between the empty intersection and some "limit" cases depending on the values occurring in the QNRs. To sum up, given σ, r, we can compute a partitioning of the individuals referring to σ and r by taking into account the values of the restrictions which concern σ and r.

Example 1. Suppose that it is necessary to encode the restrictions: $\langle \sigma, \geq 10 r.C \rangle$ and $\langle \sigma, \geq 1000 r.D \rangle$. The basic $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ encoding would introduce 1010 distinct individuals. Applying the idea explained above, instead, we could divide these 1010 individuals in, e.g., three partitions of respectively 10, 990 and again 10 individuals. If, for example, also $\langle \sigma, \leq 1005 r.\top \rangle$ must be encoded, then the last 10 individuals could be further divided into two distinct partitions. This partitioning allows for representing the cases in which 0, 5, 10, 15, 20, 990, 995, 1000, 1005 or 1010 of these individuals exist in $\Delta^{\mathcal{I}}$ (being part or not of $C^{\mathcal{I}}$ and/or $D^{\mathcal{I}}$). Even if not exhaustive these combinations are enough to represent the significant cases concerning satisfiability.

4.1 Smart Partitioning

In order to handle partitions of individuals we extend $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ with *cumulative labels* and *proxy individuals*. Given a normal/cumulative label σ' and a role r, a *cumulative label* $\sigma'.r.(i \to j)$ represents a group of consecutive individuals by means of the range of integer values $i \to j$, with $i \leq j$, thus it represents a set of individuals whose cardinality is $j - i + 1$. When $i = j$ we can both write $\sigma'.r.(i \to i)$ and $\sigma'.r.i$. With a small abuse of notation, in the following we call *proxy individual* any $\sigma.r.(i \to j)$, meaning both: (i) the cumulative label representing the set of individuals $\sigma.r.i, \sigma.r.i{+}1, \ldots, \sigma.r.j$ and (ii) that $\sigma.r.(i \to j)$ can be one/any of these individuals acting as proxy for all the other individuals of the set.

The idea is to compute a *"smart" partitioning* of the individuals to be encoded into $\mathcal{ALCQ}2SMT_{\mathcal{C}}$. With "*smart*" we mean a "safe but as small as possible" partitioning, i.e. with "a small" number of partitions but "safely" preserving the semantics of the problem, so that the cardinality of the computed partitions allow for representing every relevant case wrt. satisfiability. We formally define our smart partitioning:

Definition 2. *Let $\mathcal{T}$ being an acyclic $\mathcal{ALCQ}$ TBox in normal form. Given $\mathcal{ALCQ}2SMT_{\mathcal{C}}(\mathcal{T})$ (Definition 1), $\sigma \in \Sigma^{\mathcal{T}}$ and $r \in N_R$ we define the* arrays*:* [5]

$$\mathcal{N}^{\geq}_{\sigma.r} \stackrel{def}{=} \{\ n_i \mid \langle \sigma, \geq n_i r.C_i \rangle \in \mathcal{I}^{\mathcal{T}}_{+}\ \}_i \text{ [6] } \textit{and}$$

$$\mathcal{N}^{\leq}_{\sigma.r} \stackrel{def}{=} \{\ m_j \mid \langle \sigma, \leq m_j r.D_j \rangle \in \mathcal{I}^{\mathcal{T}}_{+}\ \}_j.$$

[5] With *array* we mean that equal n_i [resp. m_j] values repeat in $\mathcal{N}^{\geq}_{\sigma.r}$ [resp. $\mathcal{N}^{\leq}_{\sigma.r}$] as many times as they occur in the involved QNRs.

From $\mathcal{N}^{\geq}_{\sigma.r}$ and $\mathcal{N}^{\leq}_{\sigma.r}$, respectively, we define the integer values:

$$N^{\geq}_{\sigma.r} \stackrel{def}{=} \Sigma_{n_i \in \mathcal{N}^{\geq}_{\sigma.r}}\, n_i \quad \textit{and} \quad N^{\leq}_{\sigma.r} \stackrel{def}{=} \Sigma_{m_j \in \mathcal{N}^{\leq}_{\sigma.r}}\, m_j.$$

We define the set $\mathcal{P}_{\sigma.r} \stackrel{def}{=} \mathcal{P}^{\geq}_{\sigma.r} \cup \mathcal{P}^{\leq}_{\sigma.r}$ as the smart partitioning *for the $N^{\geq}_{\sigma.r}$ individuals of $\Sigma^{\mathcal{T}}$ in the form $\sigma.r.k$, where:*

$$\mathcal{P}^{\geq}_{\sigma.r} \stackrel{def}{=} \{\ n_S \mid S \in 2^{\mathcal{N}^{\geq}_{\sigma.r}},\ n_S = \Sigma_{n_k \in S}\, n_k\ \} \quad \textit{and}$$
$$\mathcal{P}^{\leq}_{\sigma.r} \stackrel{def}{=} \{\ m_S \mid S \in 2^{\mathcal{N}^{\leq}_{\sigma.r}},\ m_S = \Sigma_{m_k \in S}\, m_k\ \}.\ ^{7}$$

Finally, we define $p_i \in \mathcal{P}_{\sigma.r}$ the i-th sorted element of $\mathcal{P}_{\sigma.r}$, so that $p_i < p_{i+1}$. We have in particular: $p_1 = 0$ and $p_{|\mathcal{P}_{\sigma.r}|} = \max\{N^{\geq}_{\sigma.r}, N^{\leq}_{\sigma.r}\}$.

As $\mathcal{P}^{\geq}_{\sigma.r}, \mathcal{P}^{\leq}_{\sigma.r}, \mathcal{P}_{\sigma.r}$ are sets, equal values are uniquely represented in them. Given σ, r, and assuming to include in each partition consecutive individuals among $\sigma.r.1, ..., \sigma.r.N^{\geq}_{\sigma.r}$, then $\mathcal{P}_{\sigma.r}$ is the set containing the indexes of the last individual of all the partitions. Hence, partitions can be represented by the proxy individuals $\sigma.r.(p_{j-1}+1 \rightarrow p_j)$, with $j > 1$. For instance, notice that the partitionings shown in Example 1 are computed in accordance with Definition 2. We remark that Definition 2 defines a *safe* [8] partitioning, in fact:

- it takes into account all the values in QNRs for σ, r;
- it considers all the possible sums of the values n_i [resp. m_j] for all the at-least [resp. at-most] restrictions, which allows for representing all the possible lower-bounds [resp. upper-bounds] in case of disjoint concept interpretations;
- the union of $\mathcal{P}^{\geq}_{\sigma.r}, \mathcal{P}^{\leq}_{\sigma.r}$ represents the combination of lower- and upper-bounds;
- by sorting all the possible sums and by using the distance between these values (a partition ranges from $p_{j-1}+1$ to p_j) as the cardinality of the partitions, it allows for representing all the possible intersecting concept interpretations.

We remark that partitioning makes our approach independent from the magnitude/offset of the values occurring in QNRs.

4.2 Exploit Smart Partitioning in $\mathcal{ALCQ}2SMT_{\mathcal{C}}$

Using partitions and proxy individuals does not affect the $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ encoding, because the Theory of Costs allows for arbitrary incur costs. We can enhance Definition 1 by taking advantage of smart partitioning as follows. First we assume that the sets $\Sigma^{\mathcal{T}}, \mathcal{I}^{\mathcal{T}}_{-}, \mathcal{I}^{\mathcal{T}}_{-}$ and the functions $A_{\langle\ ,\ \rangle}$, indiv are defined consistently with the use of proxy individuals. Second, assuming to compute the partitioning $\mathcal{P}_{\sigma.r}$ of Definition 2 for every σ, r, we modify $\mathcal{ALCQ}2SMT_{\mathcal{C}}$ as follows:

[6] $\langle \sigma, \forall r.C_i \rangle \in \mathcal{I}^{\mathcal{T}}_{-}$ must be considered like $\langle \sigma, \leq 0 r.\neg C_i \rangle \in \mathcal{I}^{\mathcal{T}}_{-}$, while $\langle \sigma, \leq n_i - 1 r.C_i \rangle \in \mathcal{I}^{\mathcal{T}}_{-}$ must be considered like $\langle \sigma, \geq n_i r.C_i \rangle \in \mathcal{I}^{\mathcal{T}}_{+}$ and vice versa.

[7] Being 2^X the power set for the set/array X.

[8] I.e. it preserves the semantics of the problem wrt. satisfiability.

– The n clauses of the types (5) and (6) at point 5. are replaced by the following:

$$\{\ \mathsf{IC}(\mathsf{indiv}^C_{\sigma.r},\ cost_j,\ idx_j) \rightarrow L_{\langle \sigma_{proxy_j},\ C\rangle} \mid p_j \in \mathcal{P}_{\sigma.r},\ 0<p_j\leq n\ \} \subset \varphi^{\mathcal{T}},$$
$$\{\ \mathsf{IC}(\mathsf{indiv}^C_{\sigma.r},\ cost_j,\ idx_j) \rightarrow A_{\langle \sigma_{proxy_j},\ \top\rangle} \mid p_j \in \mathcal{P}_{\sigma.r},\ 0<p_j\leq n\ \} \subset \varphi^{\mathcal{T}},$$
$$cost_j = p_j - p_{(j-1)},\ idx_j = k^C_1 + p_{(j-1)},\ \sigma_{proxy_j} = \sigma.r.k^C_1 + p_{(j-1)} \rightarrow k^C_1 + p_j - 1.$$

– Clauses (9), (10) at point 7. are modified accordingly.
– The clauses (11) defined at point 8. must take into account the use of proxy individuals and of incur costs potentially bigger than 1. Hence they are replaced by:

$$\{(L_{\langle \sigma.r.(i\rightarrow j),\ C\rangle} \wedge A_{\langle \sigma.r.(i\rightarrow j),\ \top\rangle}) \rightarrow \mathsf{IC}(\mathsf{indiv}^C_{\sigma.r},\ j{-}i{+}1,\ i) \mid \sigma.r.(i\rightarrow j)\in\Sigma^{\mathcal{T}}\}.$$

– Clauses (14) at point 10. are modified in the same way, handling proxy individuals.

We make the following observations:

– If, for σ, r, the conditions of point 7. of Definition 1 do not hold (e.g. no at-most restriction exists), then an even more efficient partitioning requires only the following two clauses for every $\langle \sigma, \geq nr.C\rangle$:

$$\mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, n, k^C_1) \rightarrow L_{\langle \sigma.r.(k^C_1 \rightarrow k^C_1+n-1),\ C\rangle}, \in \varphi^{\mathcal{T}},$$
$$\mathsf{IC}(\mathsf{indiv}^C_{\sigma.r}, n, k^C_1) \rightarrow A_{\langle \sigma.r.(k^C_1 \rightarrow k^C_1+n-1),\ \top\rangle} \in \varphi^{\mathcal{T}}.$$

– Otherwise, if the conditions of point 7. hold, then $\varphi^{\mathcal{T}}$ contains all the clauses:

$$\{\ \mathsf{IC}(\mathsf{indiv}^C_{\sigma.r},\ p_j - p_{j-1},\ p_{j-1}+1) \rightarrow L_{\langle \sigma.r.(p_{j-1}+1\rightarrow p_j),\ C\rangle} \mid p_j \in \mathcal{P}_{\sigma.r},\ j>1\} \cup$$
$$\{\ \mathsf{IC}(\mathsf{indiv}^C_{\sigma.r},\ p_j - p_{j-1},\ p_{j-1}+1) \rightarrow A_{\langle \sigma.r.(p_{j-1}+1\rightarrow p_j),\ \top\rangle} \mid p_j \in \mathcal{P}_{\sigma.r},\ j>1\}$$

for every $\langle \sigma, \geq nr.C\rangle$, as consequence of point 5. and of the sharing of (proxy) individuals performed at point 7.

An exponential-time algorithm computing the smart partitioning $\mathcal{P}_{\sigma.r}$ (Definition 2) for every given individual σ and the role r is presented in [17]. Taken as input the arrays $\mathcal{N}^{\geq}_{\sigma.r}$ and $\mathcal{N}^{\leq}_{\sigma.r}$, the algorithm is shown to have worst-case complexity $O(2^{\max|\mathcal{N}^{\geq}_{\sigma.r}|,|\mathcal{N}^{\leq}_{\sigma.r}|})$.

5 Empirical Evaluation

We have implemented the encoder $\mathcal{ALCQ}$2SMT in C++; *smart partitioning* (§4) can be enabled optionally (denoted with S.P. hereafter). In combination with $\mathcal{ALCQ}$2SMT, we have used MATHSAT (v. 3.4.1) [2] that is the SMT-solver including the Theory of Costs [3]. We have evaluated the effectiveness of our novel approach by performing an empirical test session on about 700 synthesized[9] and parameterized $\mathcal{ALCQ}$-concept

[9] Due to lack of space we refer the reader to Section 6.1 in [6] for a discussion on why real-world ontologies are not yet available as suitable QNR benchmarks.

satisfiability problems adapted from [6], plus more. In order to compare with the available state-of-the-art reasoners we have executed the following tools on every test case: FACT++ (v. `v1.4.0`) [16], PELLET (v. `2.1.1`) [15], and RACER (RacerPro `1-9-0`) [7].

All the results presented in this section have been obtained on a 64bit biprocessor dual-core `IntelXeon2.66GHz` machine, with `16GB` of RAM. We set a 1000 seconds timeout for every concept satisfiability query. We also fixed a bound of `1GB` of disk space for the SMT($\mathcal{C}$) encoding output from $\mathcal{ALCQ}$2SMT. When reporting the results for one $\mathcal{ALCQ}$2SMT+MATHSAT configuration (either including `S.P.` or not), every CPU time reported is the sum of both the $\mathcal{ALCQ}$2SMT encoding and the MATHSAT solving time (both including the loading and parsing of the respective input problem). [10]

Importantly, with all test problems, all tools under examination (including both the variants of $\mathcal{ALCQ}$2SMT+MATHSAT) agreed on the expected un/satisfiability results when terminating within the timeout, with the exception of PELLET which incorrectly returned "sat" on some `nested_restr_unsat` problems.

Test Description. For our empirical evaluation, we have made use of synthesized test cases adapted from those in [6]. The benchmark problems of [6] focus on concept expressions containing only QNRs and define different sets of indexed problems, increasingly stressing on different sources of complexity at the increase of the index i. Since the values occurring in QNRs are one of the sources of complexity which can strongly affect the performance of reasoning for some tools, wrt. the original test cases of [6] we further parameterized such values making them depend on a parameter n. Below we list the sources of complexity of the reasoning in $\mathcal{ALCQ}$considered in [6] with the relative test set names, the ranges of the indexes i and the values chosen for n in our empirical evaluation: [11]

1. the size of the values occurring in QNRs (test cases: $\texttt{incr\_lin\_sat/unsat}_i$ with i=1–100; $\texttt{incr\_exp\_sat/unsat}_i$ with i=1–6, *satisfiable/unsatisfiable*);
2. the number of QNRs (test cases: $\texttt{restr\_num}_i(n)$ with i =1–100, $n = 1, \underline{5}, 50$, *satisfiable*);
3. effect of backtracking (number of disjoint concepts) (test cases: $\texttt{backtracking}_\texttt{i}(n)$ with i=1–20, $n = 1, 2, \underline{3}, 10$, *unsatisfiable*);
4. the ratio between the number of at-least restrictions and the number of at-most restrictions (test cases: $\texttt{restr\_ratio}_i(n)$ with i=0–14, $n = 1, \underline{5}$, *satisfiable*);
5. the satisfiability versus the unsatisfiability of the input concept expression (test cases: $\texttt{sat\_unsat}_i(n)$ with $i = 1, 2, 4, 6, \ldots, 24$, $n = 1, \underline{10}$, *half-and-half*).

For the sake of fairness of the comparison, we introduced two novel groups of problems which we believe can stress the main limitations of our approach wrt. the competitors. These groups stress two sources of complexity which were not considered in [6]:

6. the variability of the values occurring in QNRs, i.e. in every restriction occurs a unique value (test cases: $\texttt{var\_restr\_num}_i(n)$ with i=1–100, $n = 100$, *satisfiable*);

[10] To make the experiments reproducible, all the plots in full size, the tools, the problems, and the results are available at `http://disi.unitn.it/~rseba/cade11/tests.tar.gz`.

[11] The benchmark of [6] are defined for $\mathcal{SHQ}$ but we have adapted them to $\mathcal{ALCQ}$ by flattening all the role hierarchies to the only role r. The value of n originally used in [6] is underlined.

7. the number of nested QNRs (test cases: nested_restr_sat/unsat$_i(n)$, with i=1–20, $n=5, 50$, *satisfiable/unsatisfiable*).

For a much more detailed description and the exact TBoxes we refer the reader to [6,17].

Results. We compare $\mathcal{ALCQ}$2SMT+MATHSAT against the other state-of-the-art reasoners RACER, FACT++ and PELLET. In Figures 2 and 3 we plot, as representative, the results in the most challenging test cases for every benchmark category. (More plots and all the detailed results can be found in [17].) We notice the following facts about $\mathcal{ALCQ}$2SMT+MATHSAT S.P.:

- in all tests, it performs uniformly much better than plain $\mathcal{ALCQ}$2SMT+MATHSAT;
- with RACER it is the best performer in the incr_lin and incr_exp (sat/unsat) categories (Fig. 2 rows 1,2), solving all problems in negligible time;
- in the nested_restr_sat it is the worst performer, but in nested_restr_unsat it performs better than FACT++ and PELLET (Fig. 2 row 3); [12]
- with FACT++ it is the best performer in the restr_num$_i(5)$ and restr_ratio$_i(5)$ categories (Fig. 3 rows 1,2 left), solving all problems in negligible time;
- it is the absolute best scorer in restr_num$_i(50)$ test set (Fig. 3 row 1 right);
- in the var_restr_num category it performs worse than FACT++ and PELLET, but better than RACER[13] (Fig. 3 row 2 right);
- with RACER it is the best performer in the sat_unsat category (Fig. 3 row 3 left), solving all problems in negligible time;
- it is the worst performer on the backtracking problems (Fig. 3 row 3 right).

Looking into the data we notice a few more facts. First, the size of the encoded problems of $\mathcal{ALCQ}$2SMT –for both the basic and the S.P. variants– never exceed the 1GB-file-size limit, except for the nested_restr test cases with $i \geq 5$ and $i \geq 7$, respectively. (In fact, nested QNRs exponentially affect the size of our encoding.) In general, the encoded problems present a very low number of cost variables, which depends on the number of the QNRs in the input problem, and a possibly huge number of Boolean variables and clauses, which depend on the number of (proxy-) individuals introduced. Second, in the vast majority of the input problems the encoding required by $\mathcal{ALCQ}$2SMT is negligible ($\leq 10^{-2}$s); with S.P. it is significant only with the nested_restr and var_restr_num benchmarks (still ≤ 4s for the hardest problems).

Discussion. The performances of $\mathcal{ALCQ}$2SMT+MATHSAT S.P. wrt. other state-of-the-art reasoners range from some cases where it is much less efficient (backtracking, nested_restr_sat and var_restr_num) up to problems in which it significantly outperforms other tools (incr, sat_unsat and restr_num). Notice that we have specifically designed the nested_restr and var_restr_num problems in order to enforce the exponentiality of the encoding and to maximally inhibit smart partitioning, respectively. The backtracking problems, instead, have been designed in [6] to test the capability of performing dependency-directed backtracking [10]. Since the encoding is decoupled from the search, our approach cannot benefit of this optimization.

[12] Notice that that PELLET gave wrong "sat" results on nested_restr_unsat problems.

[13] RACER's implementation of the algebraic approach [8] is best-case exponential wrt. the number of QNRs.

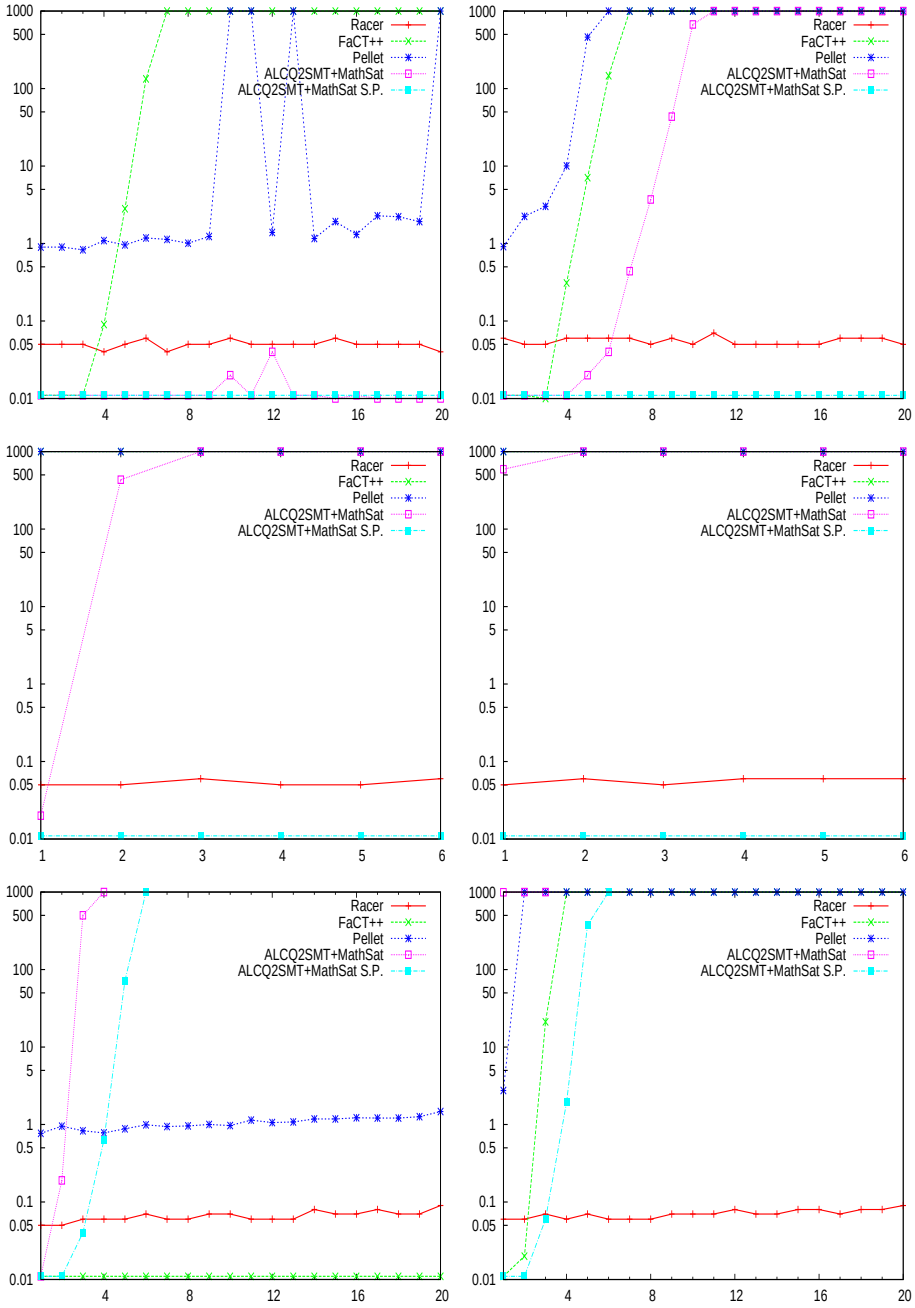

Fig. 2: Tools comparison. From left to right: `sat`, `unsat` problems; 1st row: `incr_lin`; 2nd row: `incr_exp`; 3rd row: `nested_restr`, $n=5$. X axis: test case index; Y axis: CPU time (sec).

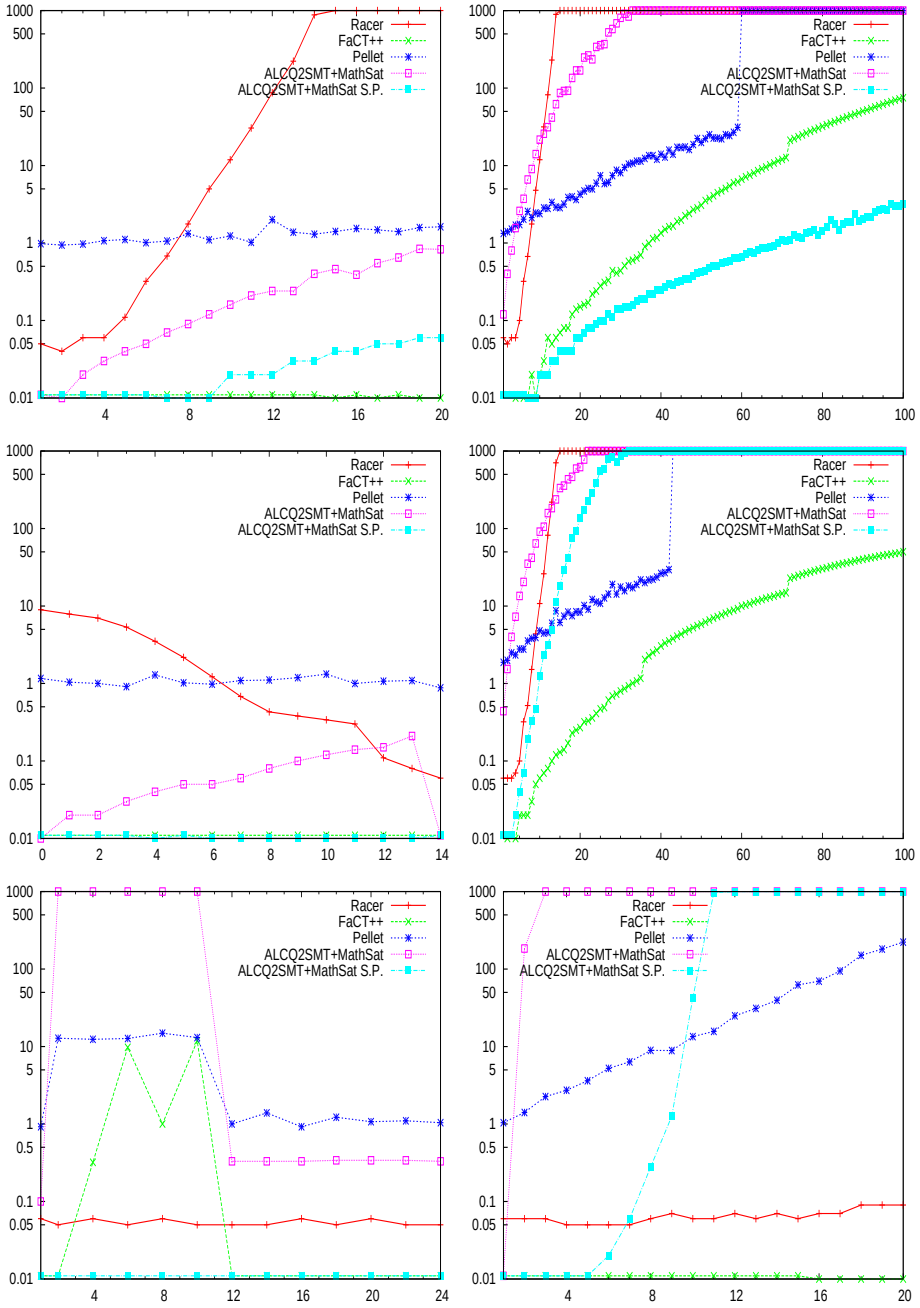

Fig. 3: Tools comparison. From let to right: 1st row: $\texttt{restr\_num}_i(5)$, $\texttt{restr\_num}_i(50)$; 2nd row: $\texttt{restr\_ratio}_i(5)$, $\texttt{var\_restr\_num}_i(100)$. 3rd row: $\texttt{sat\_unsat}_i(10)$, $\texttt{backtracking}_i(10)$. X axis: test case index; Y axis: CPU time (sec).

$\mathcal{ALCQ}$2SMT+MATHSAT `S.P.` instead performs extremely well in those benchmarks presenting multiple/balanced sources of complexity. Moreover, the size of the encoding is not the main complexity issue for our approach, which is very effective also on large or really complex problems (e.g., MATHSAT scales up to problems with more than 10^5 Boolean variables and clauses, and 10^4 cost variables, in the `nested_restr` benchmarks). Finally, smart partitioning is extremely effective, being able to drastically (and exponentially) reduce the size of the output SMT($\mathcal{C}$) problems, up to three orders of magnitude in the extreme `rest_num` and `nested_restr` cases wrt. basic $\mathcal{ALCQ}$2SMT (it exponentially impacts also in the number of cost variables in case of nested QNRs). The empirical evaluation clearly confirms that partitioning makes our approach independent from the magnitude/offset of the values occurring in QNRs.

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. 2003.
2. R. Bruttomesso, A. Cimatti, A. Franzén, A. Griggio, and R. Sebastiani. The MathSAT 4 SMT Solver. In *Proc. of the CAV/08*, LNCS. Springer, 2008.
3. A. Cimatti, A. Franzén, A. Griggio, R. Sebastiani, and C. Stenico. Satisfiability Modulo the Theory of Costs: Foundations and Applications. In *TACAS*, LNCS. Springer, 2010.
4. J. Faddoul and V. Haarslev. Algebraic Tableau Reasoning for the Description Logic $\mathcal{SHOQ}$. *Journal of Applied Logic, Special Issue on Hybrid Logics*, 8(4), 2010.
5. J. Faddoul and V. Haarslev. Optimizing Algebraic Tableau Reasoning for $\mathcal{SHOQ}$: First Experimental Results. In *Proc. of DL, Waterloo, Canada, May 4-7*, 2010.
6. N. Farsiniamarj and V. Haarslev. Practical reasoning with qualified number restrictions: A hybrid abox calculus for the description logic $\mathcal{SHQ}$. *J. AI Communications*, 23(2-3), 2010.
7. V. Haarslev and R. Möller. RACER System Description. In *Proc. of IJCAR'01*, LNAI. Springer, 2001.
8. V. Haarslev, M. Timmann, and R. Möller. Combining Tableaux and Algebraic Methods for Reasoning with Qualified Number Restrictions. In *Proc. of DL'2001*, 2001.
9. B. Hollunder and F. Baader. Qualifying Number Restrictions in Concept Languages. In *Proc. of KR*, Boston (USA), 1991.
10. I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL*, 8(3), 2000.
11. C. Lutz, C. Areces, I. Horrocks, and U. Sattler. Keys, Nominals, and Concrete Domains. *Journal of Artificial Intelligence Research, JAIR*, 23(1), 2005.
12. R. Sebastiani. Lazy Satisfiability Modulo Theories. *Journal on Satisfiability, Boolean Modeling and Computation, JSAT*, 3, p. 141–224, 2007.
13. R. Sebastiani and M. Vescovi. Automated Reasoning in Modal and Description Logics via SAT Encoding: the Case Study of $K(m)/\mathcal{ALC}$-satisfiability. *JAIR*, 35(1), 2009.
14. R. Sebastiani and M. Vescovi. Axiom Pinpointing in Lightweight Description Logics via Horn-SAT Encoding and Conflict Analysis. In *Proc. of CADE-22*, LNCS. Springer, 2009.
15. E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2), 2007.
16. D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proc. of IJCAR'06*, LNAI. Springer, 2006.
17. M. Vescovi, R. Sebastiani, and V. Haarslev. Automated Reasoning on TBoxes with Qualified Number Restrictions via SMT. Tech.Rep. DISI-11-001, Università di Trento, April 2011. Available at `http://disi.unitn.it/~rseba/cade11/techrep.pdf`.