EXPTIME Tableaux Algorithm for Contextualized $\mathcal{ALC}$

Loris Bozzato[1], Martin Homola[2], and Luciano Serafini[1]

[1] Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
[2] FMFI, Comenius University, Mlynská dolina, 84248 Bratislava, Slovakia

{bozzato,serafini}@fbk.eu, homola@fmph.uniba.sk

Abstract. Contextualized Knowledge Repository (CKR) is a DL-based framework for representation and reasoning with context dependent knowledge. It addresses the widely recognized need for contextualization of the Semantic Web data sources. Reasoning with CKR is possible thanks to a reduction to standard DL, and more recently a NEXPTIME tableaux algorithm was introduced for $\mathcal{ALC}$-based CKR. In this paper we present an EXPTIME tableaux algorithm for $\mathcal{ALC}$-based CKR. The algorithm not only formally defines a tableaux decision procedure with optimal complexity, it is also presented in a form that can be effectively applied in practice employing a suitable rule application strategy together with node caching.

1 Introduction

The enormous amount of semantic resources available in the Web in form of data sets and ontologies brings unforeseen opportunities for their reuse in different applications. It is becoming more and more clear that most of the knowledge represented in these resources is not universally valid, but it rather holds under certain circumstances, such as within a given time, location, or specific domain of interest. However, in many cases, information about this validity scope is completely missing from resource meta data.

Thus, the development of formal frameworks enabling specification and principled interpretation of such contextual meta data is an important issue for the Semantic Web and Linked Open Data communities. A number of such formalisms have been proposed, e.g. [2,9]. Contextualized Knowledge Repository (CKR) [11] is one of them. CKR is a logical framework that allows for encapsulation of description logic (DL) knowledge bases (KB) into contexts, and for specification of the contextual structure in a formal meta language. CKR KB can be built on $\mathcal{SROIQ}$ DL (corresponding to OWL 2) or on any of its fragments.

The only decision procedure known for $\mathcal{SROIQ}$-based CKR is based on translation [11] into a single $\mathcal{SROIQ}$ KB. This translation showed decidability of CKR, but for actual reasoning it is not very practical, since it introduces a significant blowup in the KB size. This approach is especially unpractical for any of the more effective fragments of $\mathcal{SROIQ}$: regardless of the initial fragment, the translation in [11] produces a KB that uses constructs strictly in $\mathcal{SROIQ}$ and complexity of $\mathcal{SROIQ}$ is

This paper appeared at the 8th International and Interdisciplinary Conference on Modeling and Using Contex (CONTEXT 2013), Annecy, France, 28th October–1st November, 2013.

very high (2NEXPTIME). For realistic reasoning with DL, tableau algorithms are typically employed. The first known direct tableaux algorithm was devised for $\mathcal{ALC}$-based CKR [4]. It extends the well known tableau algorithm for $\mathcal{ALC}$ [1] and its complexity is NEXPTIME. However, $\mathcal{ALC}$ is EXPTIME-complete [1] and EXPTIME decision procedures for $\mathcal{ALC}$ are known [5,6].

In this paper we describe an EXPTIME-tableau algorithm for $\mathcal{ALC}$-based CKR. We build on the ideas of Goré and Nguyen [6]. In order to eliminate non-determinism, the completion trees are replaced with structures called and-or graphs. In such structure, both branches of a non-deterministic choice introduced by disjunction ($\sqcup$) are explicitly represented. Satisfiability of the branches is propagated bottom-up and if it reaches an initial node, we can be sure that a model exists. To guarantee the exponential bound on the size of the graph, a global caching of nodes is used, enabled by a proper rule-application strategy. In order to reason with contexts, the algorithm of [6] was extended in multiple respects. Multiple node labels were introduced and a number of new expansion rules were added in order to implement the CKR semantics. We have also extended the algorithm with ABoxes, to be able to decide satisfiability of realistic CKR knowledge bases. We have thus reached an optimized EXPTIME decision procedure for $\mathcal{ALC}$-based CKR.

2 Contextualized Knowledge Repositories

Throughout this section we will recall some of the needed definitions for DL and $\mathcal{ALC}$: for precise semantics and other details please refer to [1]. A DL vocabulary $\Sigma = N_{\mathrm{C}} \uplus N_{\mathrm{R}} \uplus N_{\mathrm{I}}$ is composed of the three mutually disjoint subsets N_{C} of atomic concepts, N_{R} of roles, and N_{I} of individuals. In $\mathcal{ALC}$ [10], concepts are inductively defined from atomic concepts and the constructors $\neg, \sqcap$ and $\exists$. A TBox $\mathcal{T}$ is a finite set of general concept inclusions (GCI) of the form $C \sqsubseteq D$. An ABox $\mathcal{A}$ is a finite set of axioms of the form $C(a)$ or $R(a, b)$. A knowledge base is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$.

We briefly introduce the basic definition of CKR, for all details see [11]. A *meta vocabulary* Γ is used to state information about contexts. It contains contextual attributes (called dimensions), their possible values and coverage relations between these values. Formally, it is a DL vocabulary that contains: (a) a finite set of individuals called *context identifiers*; (b) a finite set of roles $\mathbf{A}$ called *dimensions*; (c) for every dimension $A \in \mathbf{A}$, a finite set of individuals D_A, called *dimensional values*, and a role $\prec_A$, called *coverage relation*. The number of dimensions $k = |\mathbf{A}|$ is assumed to be a fixed constant.

Dimensional vectors are used to identify each context with a specific set of dimensional values. Given a meta-vocabulary Γ with dimensions $\mathbf{A} = \{A_1, \ldots, A_k\}$, a *dimensional vector* $\mathbf{d} = \{A_{i_1}{:=}d_1, \ldots, A_{i_m}{:=}d_m\}$ is a (possibly empty) set of assignments such that for every j, h, with $1 \leq j \leq h \leq m$, $d_j \in D_{A_{i_j}}$, and $j \neq h$ implies $i_j \neq i_h$. A dimensional vector $\mathbf{d}$ is *full* if it assigns values to all dimensions (i.e., $|\mathbf{d}| = k$), otherwise it is *partial*. If it is apparent which value belongs to which dimension, we simply write $\{d_1, \ldots, d_m\}$. By d_A (e_A, etc.) we denote the actual value that $\mathbf{d}$ ($\mathbf{e}$, etc.) assigns to the dimension A. The *dimensional space* $\mathfrak{D}_\Gamma$ of Γ is the set of all full dimensional vectors of Γ.

An *object-vocabulary* encodes knowledge inside contexts: it is a standard DL vocabulary Σ closed w.r.t. *concept/role qualification*. That is, for every concept or role symbol X of Σ and every (possibly partial) dimensional vector $\mathbf{d}$, a new symbol $X_{\mathbf{d}}$ is added to Σ, called the *qualification* of X w.r.t. $\mathbf{d}$. If $\mathbf{d}$ is partial then $X_{\mathbf{d}}$ is *partially qualified*, if $\mathbf{d}$ is full, it is *fully qualified*. Qualified symbols are used inside contexts to refer to the meaning of symbols w.r.t. some other context.

Contexts and CKR knowledge bases are formally defined as follows.

Definition 1 (Context). *Given a pair of meta and object vocabularies $\langle \Gamma, \Sigma \rangle$, a* context *is a triple $\langle \mathcal{C}, \dim(\mathcal{C}), \mathrm{K}(\mathcal{C}) \rangle$ where: (a) $\mathcal{C}$ is a context identifier of Γ; (b) $\dim(\mathcal{C})$ is a full dimensional vector of $\mathfrak{D}_\Gamma$; (c) $\mathrm{K}(\mathcal{C})$ is an $\mathcal{ALC}$ knowledge base over Σ.*

Definition 2 (Contextualized Knowledge Repository). *Given a pair of meta and object vocabularies $\langle \Gamma, \Sigma \rangle$, a* CKR knowledge base (CKR) *is a pair $\mathfrak{K} = \langle \mathfrak{M}, \mathfrak{C} \rangle$ where $\mathfrak{C}$ is a set of contexts on $\langle \Gamma, \Sigma \rangle$ and $\mathfrak{M}$, called* meta knowledge*, is a DL knowledge base over Γ such that:*
(a) for $A \in \mathbf{A}$ and $d, d' \in D_A$, if $\mathfrak{M} \models A(\mathcal{C}, d)$ and $\mathfrak{M} \models A(\mathcal{C}, d')$ then $\mathfrak{M} \models d = d'$;
(b) for $\mathcal{C} \in \mathfrak{C}$ with $\dim(\mathcal{C}) = \mathbf{d}$ and for $A \in \mathbf{A}$, we have $\mathfrak{M} \models A(\mathcal{C}, d_A)$;
(c) the relation $\{\langle d, d' \rangle \mid \mathfrak{M} \models d \prec_A d'\}$ is a strict partial order on D_A.

Note that in CKR built over more expressive logics, conditions *(a)–(c)* can be assured directly in the meta knowledge with respective axioms: each $A \in \mathbf{A}$ is declared functional, and each $\prec_A$ is declared irreflexive and transitive. In $\mathcal{ALC}$ we do not have this option, however, since the number of dimensions and contexts is assumed to be finite, the conditions can be verified even without a reasoner (e.g., by some script) once the meta knowledge is modeled. These conditions are needed to assure reasonable properties of contextual space, i.e., acyclicity, dimensional values uniquely determined [11].

In the rest of the paper we assume that CKR knowledge bases are defined over some suitable vocabulary $\langle \Gamma, \Sigma \rangle$, and all concepts are in negation normal form (NNF). We also assume the unique name assumption (UNA) for the meta knowledge (i.e., if $a \neq b$ are two different symbols then $\mathfrak{M} \not\models a = b$). This is just to avoid the confusing possibility of two contexts located as the same place in the dimensional space.

For a CKR $\mathfrak{K}$, we will denote by $\mathcal{C}_{\mathbf{d}}$ a context with $\dim(\mathcal{C}) = \mathbf{d}$. For $\mathbf{d}, \mathbf{e} \in \mathfrak{D}_\Gamma$ and $\mathbf{B}, \mathbf{C} \subseteq \mathbf{A}$, $\mathbf{d}_{\mathbf{B}} := \{(A{:=}d) \in \mathbf{d} \mid A \in \mathbf{B}\}$ is the projection of $\mathbf{d}$ w.r.t. $\mathbf{B}$; and $\mathbf{d}_{\mathbf{B}}{+}\mathbf{e}_{\mathbf{C}} := \mathbf{d}_{\mathbf{B}} \cup \{(A{:=}d) \in \mathbf{e}_{\mathbf{C}} \mid A \notin \mathbf{B}\}$ is the completion of $\mathbf{d}_{\mathbf{B}}$ w.r.t. $\mathbf{e}_{\mathbf{C}}$.

We define strict ($\prec$) and non-strict ($\preceq$) coverage between dimensional values: for $d, d' \in D_A$, $d \prec d'$ if $\mathfrak{M} \models d \prec_A d'$; and $d \preceq d'$ if either $d \prec d'$ or $\mathfrak{M} \models d = d'$. Similarly, for dimensional vectors: $\mathbf{d} \preceq_{\mathbf{B}} \mathbf{e}$ for some $\mathbf{B} \subseteq \mathbf{A}$ if $d_B \preceq e_B$ for each $B \in \mathbf{B}$; and $\mathbf{d} \prec_{\mathbf{B}} \mathbf{e}$ if $\mathbf{d} \preceq_{\mathbf{B}} \mathbf{e}$ and $d_B \prec e_B$ for at least one $B \in \mathbf{B}$. Also, $\mathbf{d} \preceq \mathbf{e}$ if $\mathbf{d} \preceq_{\mathbf{A}} \mathbf{e}$, and $\mathbf{d} \prec \mathbf{e}$ if $\mathbf{d} \prec_{\mathbf{A}} \mathbf{e}$. Finally, for contexts: $\mathcal{C}_{\mathbf{d}} \preceq \mathcal{C}_{\mathbf{e}}$ if $\mathbf{d} \preceq \mathbf{e}$, and $\mathcal{C}_{\mathbf{d}} \prec \mathcal{C}_{\mathbf{e}}$ if $\mathbf{d} \prec \mathbf{e}$. Intuitively, if $\mathcal{C}_{\mathbf{d}} \prec \mathcal{C}_{\mathbf{e}}$, then $\mathcal{C}_{\mathbf{d}}$ is the narrower and $\mathcal{C}_{\mathbf{e}}$ is the broader context.

Example 1. We explain the possibilities in representation and reasoning of the CKR framework with a simple running example. In this example, we want to model the football domain and related competitions: each of these football events can be thus represented as a separate context, defining the situation of each particular competition. An example CKR $\mathfrak{K}_{fb}$ instantiating this scenario is shown in Fig. 1. This CKR uses

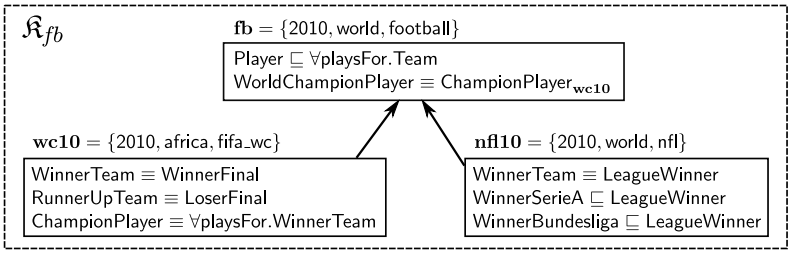

Fig. 1. Example CKR knowledge base $\mathfrak{K}_{fb}$

three dimensions time, location, and topic. It shows three contexts representing the general football context and two particular competitions. The contexts are identified by dimensional vectors $\mathbf{fb}$ (general context of football in 2010), $\mathbf{wc10}$ (FIFA World Cup 2010), and $\mathbf{nfl10}$ (national football leagues in 2010). Axioms are placed inside each context while the associated vector is placed above it. The coverage relation $\prec$ is visualized with arrows. Note that, for instance, it is evident that the interpretation of concept WinnerTeam depends on the particular context of reference: while in the situation of the World Cup the winner team is defined as "the winner of the final match", in the context of national leagues a winner team is "a winner of a national league". Note also that, by using the qualified symbol ChampionPlayer$_{\mathbf{wc10}}$, we can refer to the interpretation of the symbol in the World Cup context from the general football context. $\diamond$

CKR uses DL semantics inside each context combined with some additional semantic restrictions that ensure proper meaning of qualified symbols.

Definition 3. *A* partial DL interpretation *of a vocabulary Σ is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is a possibly empty set, and $\cdot^{\mathcal{I}}$ is a partial function, that is totally defined on N_{C} and N_{R}, with $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for $C \in N_{\mathrm{C}}$ and $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for $R \in N_{\mathrm{R}}$, and it is partially defined on N_{I}, with $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ for $a \in N_{\mathrm{I}}$ on which $\mathcal{I}$ is defined.*

Note that the definition asks for some exceptions to the standard DL semantics [1]. Partial interpretations need not necessarily provide denotations for all individuals of Σ. This is needed for technical reasons: intuitively, all contexts rely on the same object vocabulary Σ, but some element of Σ may not be meaningful in all contexts. Also, interpretations with empty domains are useful to treat inconsistency among contexts [11].

Semantics of non-atomic concepts is defined from the usual interpretation of $\mathcal{ALC}$ constructors [1]. Notice that for each concept C, if $\mathcal{I}$ is the interpretation on the empty set $\Delta_{\mathcal{I}}$, then $C^{\mathcal{I}} = \emptyset$. An interpretation $\mathcal{I}$ *satisfies* an axiom α (denoted $\mathcal{I} \models \alpha$) if it is defined on all the symbols occurring in α and the following holds: $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$; $\mathcal{I} \models C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$; and $\mathcal{I} \models R(a, b)$ iff $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$. $\mathcal{I}$ is a *model* of $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ (denoted $\mathcal{I} \models \mathcal{K}$) iff it satisfies all axioms in $\mathcal{T} \cup \mathcal{A}$; $\mathcal{K}$ is satisfiable if it has a model. Notice that, if $\mathcal{I}$ is the interpretation on the empty domain we have that $\mathcal{I} \models C \sqsubseteq D$ for every pair of $\mathcal{ALC}$ concepts C and D, and $\mathcal{I} \not\models \alpha$ for every assertion α of the form $C(a)$ and $R(a, b)$. A concept C is *satisfiable* w.r.t. an $\mathcal{ALC}$ KB $\mathcal{K}$ iff there exists a model $\mathcal{I}$ of $\mathcal{K}$ s.t. $C^{\mathcal{I}}$ is non-empty. A subsumption formula $C \sqsubseteq D$ is *entailed* by an $\mathcal{ALC}$ KB $\mathcal{K}$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in all models $\mathcal{I}$ of $\mathfrak{K}$. It is well known that these two problems are inter-reducible [1].

Definition 4 (CKR-Model). *A model of a CKR $\mathfrak{K}$ is a collection $\mathfrak{I} = \{\mathcal{I}_\mathbf{d}\}_{\mathbf{d}\in\mathfrak{D}_\Gamma}$ of partial DL interpretations (*local interpretations*) s.t. for all $\mathbf{d},\mathbf{e},\mathbf{f} \in \mathfrak{D}_\Gamma$, $\mathbf{B} \subseteq \mathbf{A}$, $A \in N_\mathrm{C}$, $R \in N_\mathrm{R}$, $X \in N_\mathrm{C} \cup N_\mathrm{R}$, $a \in N_\mathrm{I}$:*

1. $(\top_\mathbf{d})^{\mathcal{I}_\mathbf{f}} \subseteq (\top_\mathbf{e})^{\mathcal{I}_\mathbf{f}}$ *if* $\mathbf{d} \prec \mathbf{e}$
2. $(A_\mathbf{f})^{\mathcal{I}_\mathbf{d}} \subseteq (\top_\mathbf{f})^{\mathcal{I}_\mathbf{d}}$
3. $(R_\mathbf{f})^{\mathcal{I}_\mathbf{d}} \subseteq (\top_\mathbf{f})^{\mathcal{I}_\mathbf{d}} \times (\top_\mathbf{f})^{\mathcal{I}_\mathbf{d}}$
4. $a^{\mathcal{I}_\mathbf{e}} = a^{\mathcal{I}_\mathbf{d}}$ *if* $\mathbf{d} \prec \mathbf{e}$ *and*
   - $a^{\mathcal{I}_\mathbf{d}}$ *is defined or,*
   - $a^{\mathcal{I}_\mathbf{e}}$ *is defined and* $a^{\mathcal{I}_\mathbf{e}} \in \Delta_\mathbf{d}$
5. $(X_{\mathbf{d}_\mathbf{B}})^{\mathcal{I}_\mathbf{e}} = (X_{\mathbf{d}_\mathbf{B}+\mathbf{e}})^{\mathcal{I}_\mathbf{e}}$
6. $(X_\mathbf{d})^{\mathcal{I}_\mathbf{e}} = (X_\mathbf{d})^{\mathcal{I}_\mathbf{d}}$ *if* $\mathbf{d} \prec \mathbf{e}$
7. $(A_\mathbf{f})^{\mathcal{I}_\mathbf{d}} = (A_\mathbf{f})^{\mathcal{I}_\mathbf{e}} \cap \Delta_\mathbf{d}$ *if* $\mathbf{d} \prec \mathbf{e}$
8. $(R_\mathbf{f})^{\mathcal{I}_\mathbf{d}} = (R_\mathbf{f})^{\mathcal{I}_\mathbf{e}} \cap (\Delta_\mathbf{d} \times \Delta_\mathbf{d})$ *if* $\mathbf{d} \prec \mathbf{e}$
9. $\mathcal{I}_\mathbf{d} \models \mathrm{K}(\mathcal{C}_\mathbf{d})$

The semantics takes care that local domains respect the coverage hierarchy (condition 1). Note that $\top_\mathbf{d}$ represents the domain of $\mathcal{I}_\mathbf{d}$ in the context where it appears. Individuals have a rigid meaning: however, the meaning of an individual in a super-context is independent if its meaning in a sub-context is undefined (condition 4). The interpretation of $X_\mathbf{f}$ in any context $\mathcal{C}_\mathbf{d}$ is roofed under $(\top_\mathbf{f})^{\mathcal{I}_\mathbf{d}}$ (conditions 2, 3). The meaning of $X_\mathbf{f}$ in some context $\mathcal{C}_\mathbf{e}$ is based on its context of origin $\mathcal{C}_\mathbf{f}$ if this context is less specific than $\mathcal{C}_\mathbf{e}$ (condition 6); otherwise, at least, any $X_\mathbf{f}$ in $\mathcal{C}_\mathbf{d}$ and $\mathcal{C}_\mathbf{e}$ must be equal on the shared part of their domains (conditions 7,8). Each $\mathcal{I}_\mathbf{d}$ is a DL-model of $\mathcal{C}_\mathbf{d}$ (condition 9). Condition 5 provides the meaning of partially qualified symbols: values for attributes not specified are taken from the dimensions of the context in which the symbol appears. From this reading, we can show that, albeit useful for modeling, partially qualified vectors are a kind of syntactic sugar [11].

Given a CKR $\mathfrak{K}$ and $\mathbf{d} \in \mathfrak{D}_\Gamma$, a concept C is $\mathbf{d}$-*satisfiable* w.r.t. $\mathfrak{K}$ if there exists a CKR model $\mathfrak{I} = \{\mathcal{I}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}$ of $\mathfrak{K}$ such that $C^{\mathcal{I}_\mathbf{d}} \neq \emptyset$; $\mathfrak{K}$ is $\mathbf{d}$-satisfiable if it has a CKR model $\mathfrak{I} = \{\mathcal{I}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}$ such that $\Delta_\mathbf{d} \neq \emptyset$; $\mathfrak{K}$ is *globally satisfiable* if it has a CKR model $\mathfrak{I} = \{\mathcal{I}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}$ such that $\Delta_\mathbf{e} \neq \emptyset$ for every $\mathbf{e} \in \mathfrak{D}_\Gamma$. An axiom α is $\mathbf{d}$-*entailed* by $\mathfrak{K}$ (denoted $\mathfrak{K} \models \mathbf{d} : \alpha$) if for every model $\mathfrak{I} = \{\mathcal{I}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}$ of $\mathfrak{K}$ it holds $\mathcal{I}_\mathbf{d} \models \alpha$. As usual, $\mathbf{d}$-entailment can be reduced to $\mathbf{d}$-satisfiability: in particular $\mathfrak{K} \models \mathbf{d} : C \sqsubseteq D$ iff $C \sqcap \neg D$ is not $\mathbf{d}$-satisfiable w.r.t. $\mathfrak{K}$. Also, C is $\mathbf{d}$-satisfiable w.r.t. $\mathfrak{K}$ iff $\mathfrak{K}'$ is $\mathbf{d}$-satisfiable, where $\mathfrak{K}'$ is obtained from $\mathfrak{K}$ by adding the axiom $C(s_0)$ to $\mathcal{C}_\mathbf{d}$ and s_0 is a new constant not used elsewhere in $\mathfrak{K}$.

In the rest of the paper we will assume that all symbols are fully qualified. Given a $\mathfrak{K}$ with partially qualified symbols, it can be replaced by an equivalent KB $\mathfrak{K}'$ in which any partially qualified symbol $X_{\mathbf{d}_\mathbf{B}}$ appearing in some context $\mathcal{C}_\mathbf{e}$ is replaced by $X_{\mathbf{d}_\mathbf{B}+\mathbf{e}}$ (see [11]), however for sake of legibility we will occasionally use partially qualified symbols as syntactic sugar in the examples.

Given a concept C and a CKR $\mathfrak{K} = \langle\mathfrak{M},\mathfrak{C}\rangle$, we will denote $\mathrm{clos}(C)$ the set of all syntactically correct atomic and complex concepts that occur in C. We will denote by $\mathrm{clos}_\mathfrak{K}(C)$ the set of all syntactically correct atomic and complex concepts that occur in C or in any axiom of the contexts in $\mathfrak{C}$. We denote by $\mathcal{R}_{\mathfrak{K},C}$ the set of roles appearing in C or some or in any axiom of the contexts in $\mathfrak{C}$.

3 EXPTIME Tableaux Algorithm for CKR

The known tableaux algorithm for $\mathcal{ALC}$-based CKR takes at most exponentially many steps before it decides if the input concept/CKR is satisfiable. To deal with disjunction

($\sqcup$) the algorithm makes non-deterministic choices, and therefore it is NEXPTIME. This is similar to the classic tableaux algorithm for $\mathcal{ALC}$. To obtain an EXPTIME algorithm for $\mathcal{ALC}$ the non-determinism has to be eliminated [5,6].

Similarly to Goré and Nguyen [6] we will employ and-or graphs to eliminate non-determinism. This is obtained by explicitly representing both possibilities resulting from a disjunctive choice.

Definition 5 (And-or graph). *Given a CKR $\mathfrak{K}$ and a concept C in NNF and $\mathbf{d} \in \mathfrak{D}_\Gamma$, an* and-or graph *is a quadruple $G = \langle V, E, \mathcal{L}, \{\mathcal{L}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}\rangle$ where:*

(a) V is a set of structured nodes s.t. for every $x \in V$:
- $x.\mathrm{type} \subseteq \{\mathsf{TOP}, \mathsf{AND}, \mathsf{OR}\}$,
- $x.\mathrm{status} \in \{\mathsf{EXP}, \mathsf{UNEXP}, \mathsf{SAT}, \mathsf{UNSAT}\}$,
- *for each $\mathbf{e} \in \mathfrak{D}_\Gamma$, $V_\mathbf{e} \subseteq V$ is a distinguished subset of V;*

(b) $E \subseteq V \times V$ is a set of edges;
(c) $\mathcal{L}$ is a partial labeling function from E to $2^{\mathcal{R}_{\mathfrak{K},C}}$;
(d) for each $\mathbf{e} \in \mathfrak{D}_\Gamma$, $\mathcal{L}_\mathbf{e}$ is a labeling function from $V_\mathbf{e}$ to $2^{\mathrm{clos}_\mathfrak{K}(C)}$.

And-or graphs are similar to completion trees which are often used by tableaux algorithms [4,6] in that they are composed of nodes with labels[3]. There are multiple types of nodes: if AND (OR, TOP) belongs to $x.\mathrm{type}$ then x is called an *and-node (or-node, top-node)*. And-nodes are similar to nodes in a completion tree: their successors represent other nodes related by some role. Or-nodes represent choices: their successors can be understood as alternative versions. And- and or-nodes are mutually exclusive. Top-nodes represent ABox individuals and they can also be and- or or-nodes.

If $\langle x, y\rangle \in E$, then y is a *predecessor* of x and conversely y is a *successor* of x. If in addition $R \in \mathcal{L}(\langle x, y\rangle)$, then y is an *R-predecessor* of x and conversely y is an *R-successor* of x. If y is a successor of an or-node x, then y is an *or-successor* of x and x is an *or-predecessor* of y. *Ancestors*, *R-ancestors*, *or-ancestors* and *descendants*, *R-descendants*, *or-descendants* are defined as transitive closure on the respective type of predecessors and successors as usual.

The algorithm iteratively expands the nodes of the and-or graph using a set of tableaux expansion rules. The status of each node x is tracked in $x.\mathrm{status}$. First the status is UNEXP. If all of the nodes labels are fully expanded, the status is EXP and the node is evaluated. A node x contains a *clash* if for some $\mathbf{d} \in \mathfrak{D}_\Gamma$ and some concept C both $C \in \mathcal{L}_\mathbf{d}(x)$ and $\neg C \in \mathcal{L}_\mathbf{d}(x)$ or if $\bot \in \mathcal{L}_\mathbf{d}(x)$. In such a case the status is set to UNSAT. Otherwise the node is *clash-free* and its status is set to SAT.

In order to limit the size of the generated and-or graph (and in turn the time complexity) node caching [6] will be employed (see also blocking condition in [4]). If two nodes have all labels equal, we only store one of them and use it as representation of both. A node $x \in V$ is a *witness* for another node $y \in V$ if $\mathcal{L}_\mathbf{e}(x) = \mathcal{L}_\mathbf{e}(y)$ for all $\mathbf{e} \in \mathfrak{D}_\Gamma$. If a witness is found, the node y is removed and the witness x is used in its place by redirecting to x all of the incoming edges of y.

[3] Note that, differently from the previous version of the algorithm [4], the labels for edges are not distinguished by dimensional vector, as in the case of node labels. This allow us to simplify the completion rules (namely, by removing the R-rule used by [4]), but special care must be observed when proving the correctness of the algorithm.

In order to deal with the ABox, the ABox individuals are encoded into the and-or graph during the initialization. This technique is well known for logics like $\mathcal{ALC}$ [1]. However in CKR same individuals appearing in different contexts may possibly have different meanings. In the graph, individuals will be represented by nodes with status TOP of the form $a^{\mathbf{g}}$ where $a \in N_{\mathrm{I}}$ and $\mathbf{g} \in \mathfrak{D}_\Gamma$ identifies the context in which the individual was first introduced. To implement condition 4 of CKR-models we will merge nodes when needed using a $\mathrm{merge}(x, y)$ procedure.

Definition 6 (Merging). *Executing* $\mathrm{merge}(x, y)$ *on* $G = \langle V, E, \mathcal{L}, \{\mathcal{L}_{\mathbf{e}}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}\rangle$*, with* $x, y \in V$*, transforms* G *as follows: a) node* x *is added into* $V_{\mathbf{e}}$ *for all* $\mathbf{e} \in \mathfrak{D}_\Gamma$ *s.t.* $y \in V_{\mathbf{e}}$*; b) all concepts from* $\mathcal{L}_{\mathbf{e}}(y)$ *are added into* $\mathcal{L}_{\mathbf{e}}(x)$*, for all* $\mathbf{e} \in \mathfrak{D}_\Gamma$*; c) all edges directed into/from* y *are redirected into/from* x*; d) node* y *is removed from* V*.*

The algorithm uses tableau expansion rules from Table 1. We say that a tableaux rule is *applicable* on some node $x \in V$ if all of its preconditions (the if-part of the rule) are satisfied for x. The and-or graph is *complete* if none of the rules is applicable. The rules can be split into two groups: $\sqcup$- and $\exists$-rules are *generating* because they generate new nodes; the remaining rules are *expanding* because they do not generate new nodes but instead expand the labels within existing nodes. For reasoning inside each context the well known $\mathcal{ALC}$ tableaux rules [1] are used (the left hand side of Table 1) with some notable modifications. The $\sqcup$-rule is adapted for and-or graphs, instead of an non-deterministic choice it generates or-successors. The perspective of the $\forall$-rule is altered in order to unify it with other expanding rules: it is applicable on a node x which has a predecessor y which has a value restriction in its label. The $\exists$-rule implements part of the CKR semantics because it also adds $\top_{\mathbf{f}}$ to the label of the generated $R_{\mathbf{f}}$-successor.

$\sqcap$-rule: **if** $x \in V_{\mathbf{d}}$, $C_1 \sqcap C_2 \in \mathcal{L}_{\mathbf{d}}(x)$, $\{C_1, C_2\} \not\subseteq \mathcal{L}_{\mathbf{d}}(x)$ **then** $\mathcal{L}_{\mathbf{d}}(x) := \mathcal{L}_{\mathbf{d}}(x) \cup \{C_1, C_2\}$

$\sqcup$-rule: **if** $x \in V_{\mathbf{d}}$, $C_1 \sqcup C_2 \in \mathcal{L}_{\mathbf{d}}(x)$, $\{C_1, C_2\} \cap \mathcal{L}_{\mathbf{d}}(x) = \emptyset$ **then** $V_{\mathbf{d}} := V_{\mathbf{d}} \cup \{y, z\}$ with y, z new, $E := E \cup \{\langle x, y\rangle, \langle x, z\rangle\}$, $\mathcal{L}_{\mathbf{d}}(y) := \mathcal{L}_{\mathbf{d}}(x) \cup \{C_1\}$, $\mathcal{L}_{\mathbf{d}}(z) := \mathcal{L}_{\mathbf{d}}(x) \cup \{C_2\}$

$\exists$-rule: **if** $x \in V_{\mathbf{d}}$, $\exists R_{\mathbf{f}}.C \in \mathcal{L}_{\mathbf{d}}(x)$, and there is no $R_{\mathbf{f}}$-successor $y \in V_{\mathbf{d}}$ of x s.t. $C \in \mathcal{L}_{\mathbf{d}}(y)$ **then** $V_{\mathbf{d}} := V_{\mathbf{d}} \cup \{z\}$ with z new, $E := E \cup \{\langle x, z\rangle\}$, $\mathcal{L}(\langle x, z\rangle) := \{R_{\mathbf{f}}\}$, $\mathcal{L}_{\mathbf{d}}(z) := \{C, \top_{\mathbf{f}}\}$

$\forall$-rule: **if** $x \in V_{\mathbf{d}}$, $\forall R_{\mathbf{f}}.C \in \mathcal{L}_{\mathbf{d}}(y), C \notin \mathcal{L}_{\mathbf{d}}(x)$, $y \in V_{\mathbf{d}}$ is an $R_{\mathbf{f}}$-predecessor of x **then** $\mathcal{L}_{\mathbf{d}}(x) := \mathcal{L}_{\mathbf{d}}(x) \cup \{C\}$

$\mathcal{T}$-rule: **if** $x \in V_{\mathbf{d}}$, $C \sqsubseteq D \in \mathrm{K}(\mathcal{C}_{\mathbf{d}})$, $\mathrm{nnf}(\neg C \sqcup D) \notin \mathcal{L}_{\mathbf{d}}(x)$ **then** $\mathcal{L}_{\mathbf{d}}(x) := \mathcal{L}_{\mathbf{d}}(x) \cup \{\mathrm{nnf}(\neg C \sqcup D)\}$

$\Delta\uparrow$-rule: **if** $x \in V_{\mathbf{d}}, \mathbf{d} \prec \mathbf{e}, x \notin V_{\mathbf{e}}$ **then** $V_{\mathbf{e}} := V_{\mathbf{e}} \cup \{x\}$

$\Delta\downarrow$-rule: **if** $x \in V_{\mathbf{e}}, \mathbf{d} \prec \mathbf{e}, x \notin V_{\mathbf{d}}$ $\top_{\mathbf{d}} \in \mathcal{L}_{\mathbf{e}}(x)$ **then** $V_{\mathbf{d}} = V_{\mathbf{d}} \cup \{x\}$

A-rule: **if** $x \in V_{\mathbf{d}} \cap V_{\mathbf{e}}, \mathbf{d} \prec \mathbf{e}$ or $\mathbf{d} \succ \mathbf{e}$, $A_{\mathbf{f}} \in \mathcal{L}_{\mathbf{d}}(x), A_{\mathbf{f}} \notin \mathcal{L}_{\mathbf{e}}(x)$ **then** $\mathcal{L}_{\mathbf{e}}(x) := \mathcal{L}_{\mathbf{e}}(x) \cup \{A_{\mathbf{f}}\}$

$\top_\exists$-rule: **if** $x \in V_{\mathbf{d}}, \exists R_{\mathbf{f}}.C \in \mathcal{L}_{\mathbf{d}}(x)$, $\top_{\mathbf{f}} \notin \mathcal{L}_{\mathbf{d}}(x)$ **then** $\mathcal{L}_{\mathbf{d}}(x) := \mathcal{L}_{\mathbf{d}}(x) \cup \{\top_{\mathbf{f}}\}$

$\top_A$-rule: **if** $x \in V_{\mathbf{e}}, A_{\mathbf{d}} \in \mathcal{L}_{\mathbf{e}}(x)$, $\top_{\mathbf{d}} \notin \mathcal{L}_{\mathbf{e}}(x)$ **then** $\mathcal{L}_{\mathbf{e}}(x) := \mathcal{L}_{\mathbf{e}}(x) \cup \{\top_{\mathbf{d}}\}$

$\top_\sqsubseteq$-rule: **if** $x \in V_{\mathbf{d}}$, $\mathbf{e} \prec \mathbf{f}$, $\neg\top_{\mathbf{e}} \sqcup \top_{\mathbf{f}} \notin \mathcal{L}_{\mathbf{d}}(x)$ **then** $\mathcal{L}_{\mathbf{d}}(x) := \mathcal{L}_{\mathbf{d}}(x) \cup \{\neg\top_{\mathbf{e}} \sqcup \top_{\mathbf{f}}\}$

M-rule: **if** $a^{\mathbf{g}} \in V_{\mathbf{d}}$, $a^{\mathbf{h}} \in V_{\mathbf{e}}$, and $\mathbf{d} \preceq \mathbf{e}$, **then** $\mathrm{merge}(a^{\mathbf{g}}, a^{\mathbf{h}})$

Table 1. CKR tableaux completion rules

Similarly the A-rule propagates concepts between labels. The $\top_A$- and $\top_\exists$-rules ensure conditions 2-3 of the CKR models (and partly the $\exists$-rule is also instrumental in this).

Rules on the right of Table 1 are largely responsible for maintaining the CKR semantics. The $\Delta\downarrow$- and $\Delta\uparrow$-rules propagate nodes across contexts. Finally the $\top_\sqsubseteq$ is responsible to maintain the hierarchy of tops within each contexts, and the M-rule is responsible for note merging. These rules are adapted from the existing NEXPTIME algorithm for $\mathcal{ALC}$-based CKR: refer to [4] for further details. Necessary adaptations include especially implementing the so called *never look behind* strategy [6] under which all nodes are fully expanded before any generating rule is applied. The previously used $\top_R$-rule [4] violated this strategy: its function is now divided between $\exists$- and $\top_\exists$-rules.

The algorithm itself is presented in Fig. 2. It takes as input a CKR $\mathfrak{K}$ and generates an and-or graph. If eventually all top-nodes have status SAT the algorithm concludes that $\mathfrak{K}$ is $\mathbf{d}$-satisfiable. If one of the top-nodes reaches status UNSAT the algorithm concludes the contrary. The and-or graph G is initialized in the first `for each` loop. All of the original ABox individuals are represented as nodes and marked as TOP.

In the main loop that follows G is expanded until all the TOP elements have a defined status SAT or UNSAT. In each iteration, an unexpanded node x is picked and processed. First, all of the expanding rules are applied on x; in this process the node is fully expanded. Consecutively the cache is queried and if the node was generated before, it is discarded and represented by its cached version. If there is a clash, the node is marked as UNSAT. Otherwise the algorithm proceeds with generating the successors of x. Following the never look behind strategy, or-successors are generated first so that we generate all the alternative versions of x which are further expanded in the future iterations of the loop. Only if the $\sqcup$-rule is no longer applicable, the $\exists$-rule is finally applied. Finally the node is pronounced satisfiable if its a leaf node (at this point there is no clash due to the preceding unsatisfiability check). If the rule has successors, its status is set to EXP and its satisfiability will be verified later by propagation from its successors. If the status SAT or UNSAT is already known, it is propagated to the node predecessors by calling the `propagate`(G, x) procedure shown in Fig. 3.

Special care must be observed when top-nodes (or their or-successors) are processed. Occasionally some of these nodes have to be merged using the M-rule: labels of such nodes and also incoming and outgoing edges are combined. Also, if the $\sqcup$-rule is applied on such a node x resulting into successors y and z, all ABox relations need to be propagated to y and z. Note also that after merging and or-branching of top-nodes or their or-successors the neighboring nodes are affected because further expansions may be triggered by the $\forall$-rule; hence these nodes are reset to status UNEXP.

Before discussing correctness of the procedure, we present an example of deduction.

Example 2. We revise the deduction example from [4]. Using the algorithm and our example CKR $\mathfrak{K}_{fb}$, we can show a proof for the following subsumption:

$$\mathfrak{K}_{fb} \models \mathbf{nfl10} : \mathsf{WorldChampionPlayer}_{\mathbf{fb}} \sqsubseteq \forall \mathsf{playsFor}_{\mathbf{wc10}}.\mathsf{WinnerTeam}_{\mathbf{wc10}}$$

Intuitively, we are asking to verify whether, in the context of $\mathbf{nfl10}$, "all of the world champion players play in a winner team in the World Cup". In the initialization step, since $\mathfrak{K}_{fb}$ does not contain ABoxes, we have $V_{\mathbf{nfl10}} = \{s_0^{\mathbf{nfl10}}\}$, $\mathcal{L}_{\mathbf{nfl10}}(s_0^{\mathbf{nfl10}}) = \{\mathsf{WorldChampionPlayer}_{\mathbf{fb}} \sqcap \exists \mathsf{playsFor}_{\mathbf{wc10}}.\neg \mathsf{WinnerTeam}_{\mathbf{wc10}}\}$, and $s_0^{\mathbf{nfl10}}.\mathrm{type} =$

Input: CKR $\mathfrak{K}$, $\mathbf{d} \in \mathfrak{D}_\Gamma$
Output: And-or graph $G = \langle V, E, \mathcal{L}, \{\mathcal{L}_\mathbf{e}\}_{\mathbf{e} \in \mathfrak{D}_\Gamma} \rangle$ s.t. $\mathfrak{K}$ is $\mathbf{d}$-satisfiable iff,
for all $x \in V$ with $\mathsf{TOP} \in x.\mathrm{type}$, $x.\mathrm{status} = \mathsf{SAT}$

```
begin
```
for each $\mathbf{e} \in \mathfrak{D}_\Gamma$ initialize $V_\mathbf{e}$, $\mathcal{L}_\mathbf{e}$, E, and $\mathcal{L}$:
$V_\mathbf{e} := \{a^\mathbf{e} \mid C(a) \in \mathrm{K}(\mathcal{C}_\mathbf{e})\} \cup \{a^\mathbf{e}, b^\mathbf{e} \mid R(a,b) \in \mathrm{K}(\mathcal{C}_\mathbf{e})\}$;
$E := \{\langle a^\mathbf{e}, b^\mathbf{e} \rangle \mid R(a,b) \in \mathrm{K}(\mathcal{C}_\mathbf{e}), \mathbf{e} \in \mathfrak{D}_\Gamma\}$;
$\mathcal{L}_\mathbf{e}(a^\mathbf{e}) := \{C \mid C(a) \in \mathrm{K}(\mathcal{C}_\mathbf{e})\}$, for all $a^\mathbf{e} \in V$;
$\mathcal{L}(\langle a^\mathbf{e}, b^\mathbf{e} \rangle) := \{R \mid R(a,b) \in \mathrm{K}(\mathcal{C}_\mathbf{e})\}$, for all $\langle a^\mathbf{e}, b^\mathbf{e} \rangle \in E$;
$x.\mathrm{type} := \{\mathsf{TOP}\}$, $x.\mathrm{status} := \mathsf{UNEXP}$, for all $x \in V$;

`while` ($y.\mathrm{status} \notin \{\mathsf{SAT}, \mathsf{UNSAT}\}$ for some $y \in V$ s.t. $\mathsf{TOP} \in y.\mathrm{type}$ and
there is $x \in V$ s.t. $x.\mathrm{status} = \mathsf{UNEXP}$)

// Node expansion
`while` (one of $\sqcap, \forall, \mathcal{T}, \Delta_\uparrow, \Delta_\downarrow, A, \top_\exists, \top_A, \top_\sqsubseteq$, M-rules is applicable on x)
`if` (M-rule is applicable on x and some y)
apply M-rule on x and y yielding z; $z.\mathrm{type} := \{\mathsf{TOP}\}$;
$z.\mathrm{status} := \mathsf{UNEXP}$; $w.\mathrm{status} := \mathsf{UNEXP}$ for all successors w of z;
`else` apply one of $\sqcap, \forall, \mathcal{T}, \Delta_\uparrow, \Delta_\downarrow, A, \top_\exists, \top_A, \top_\sqsubseteq$-rules to x;
`end if`;
`end while`;

// Caching (the node is now fully expanded)
`if` (there is a witness $x' \in V$ of x and $\mathsf{TOP} \notin x.\mathrm{type}$ and $\mathsf{TOP} \notin y.\mathrm{type}$ for all or-ancestors y of x)
`for each` ($y \in V$; y predecessor of x)
add $\langle y, x' \rangle$ to E; $\mathcal{L}(\langle y, x' \rangle) := \mathcal{L}(\langle y, x \rangle)$; remove $\langle y, x \rangle$ from E;
`end for each`;
remove x from V;

// Unsatisfiability check
`else if` (x contains a clash) $x.\mathrm{status} := \mathsf{UNSAT}$;

// Or-branching
`else if` ($\sqcup$-rule is applicable on x)
apply $\sqcup$-rule to x yielding new nodes y, z;
`if` ($\mathsf{TOP} \in x.\mathrm{type}$)
double all edges from/to x including labels and redirect to both y and z;
$w.\mathrm{status} := \mathsf{UNEXP}$ for all successors w of y and z;
`end if`;
$y.\mathrm{type} := z.\mathrm{type} := \{\}$; $y.\mathrm{status} := z.\mathrm{status} := \mathsf{UNEXP}$;
$x.\mathrm{type} := x.\mathrm{type} \cup \{\mathsf{OR}\}$; $x.\mathrm{status} := \mathsf{EXP}$;

// And-branching
`else if` ($\exists$-rule is applicable on x)
`while` ($\exists$-rule is applicable on x)
apply $\exists$-rule to x yielding a new node y;
$y.\mathrm{type} := \{\}$; $y.\mathrm{status} := \mathsf{UNEXP}$;
`end while`;
$x.\mathrm{type} := x.\mathrm{type} \cup \{\mathsf{AND}\}$; $x.\mathrm{status} := \mathsf{EXP}$;

// Now x is a leaf node, fully expanded and satisfiable
`else` $x.\mathrm{status} := \mathsf{SAT}$;
`end if`;

// Propagation check
`if` ($x.\mathrm{status} \in \{\mathsf{SAT}, \mathsf{UNSAT}\}$) `propagate`$(G, x)$; `end if`;
`end while`;
`end`.

Fig. 2. EXPTIME procedure for $\mathcal{ALC}$ CKR

Input: and-or graph $G = \langle V, E, \mathcal{L}, \{\mathcal{L}_{\mathbf{e}}\}_{\mathbf{e}\in\mathfrak{D}_\mathcal{I}}\rangle$ and $y \in V$ with $y.\mathrm{status} \in \{\mathsf{SAT}, \mathsf{UNSAT}\}$
Output: modified and-or graph G

```
begin
  queue := {y};
  while (queue not empty)
      extract x from queue;
      for each u ∈ V with (u, x) ∈ E and u.status = EXP
          if (u.type ∈ OR and one of its successors has status SAT)
          or (u.type ∈ AND and all of its successors have status SAT)
            u.status := SAT; queue := queue ∪ {u};
          if (u.type ∈ AND and one of its successors has status UNSAT)
          or (u.type ∈ OR and all of its successors have status UNSAT)
            u.status := UNSAT; queue := queue ∪ {u};
      end for each;
  end while;
end.
```

Fig. 3. `propagate`(G, y) procedure

TOP, $s_0^{\mathbf{nfl10}}.\mathrm{status} = \mathsf{UNEXP}$. Since $\mathcal{C}_{\mathbf{nfl10}}$ is the only initialized context, we write s_0 in place of $s_0^{\mathbf{nfl10}}$ for simplicity of presentation. The expansion starts from s_0 with the following steps:

- $\mathcal{L}_{\mathbf{nfl10}}(s_0) := \mathcal{L}_{\mathbf{nfl10}}(s_0) \cup \{\mathsf{WorldChampionPlayer}_{\mathbf{fb}}, \exists\mathsf{playsFor}_{\mathbf{wc10}}.\neg\mathsf{WinnerTeam}_{\mathbf{wc10}}\}$ by $\sqcap$-rule;
- $V_{\mathbf{fb}} := \{s_0\}$, $\mathcal{L}_{\mathbf{fb}}(s_0) := \{\mathsf{WorldChampionPlayer}\}$ by $\Delta\uparrow$ and A-rule;
- $\mathcal{L}_{\mathbf{nfl10}}(s_0) := \mathcal{L}_{\mathbf{nfl10}}(s_0) \cup \{\top_{\mathbf{wc10}}\}$ by $\top_\exists$-rule;
- $\mathcal{L}_{\mathbf{fb}}(s_0) := \mathcal{L}_{\mathbf{fb}}(s_0) \cup \{\top_{\mathbf{wc10}}\}$, $V_{\mathbf{wc10}} := \{s_0\}$ by A and $\Delta\downarrow$-rule;
- $\mathcal{L}_{\mathbf{fb}}(s_0) := \mathcal{L}_{\mathbf{fb}}(s_0) \cup \{\neg\mathsf{WorldChampionPlayer} \sqcup \mathsf{ChampionPlayer}_{\mathbf{wc10}}\}$, $\mathcal{L}_{\mathbf{wc10}}(s_0) := \mathcal{L}_{\mathbf{fb}}(s_0) \cup \{\neg\mathsf{ChampionPlayer} \sqcup \forall\mathsf{playsFor}.\mathsf{WinnerTeam}\}$ by $\mathcal{T}$-rule;
- The local expansion on s_0 terminates here: a pair of or-successors s_0^0 and s_0^1 are generated by the application of $\sqcup$-rule;
- $\{\neg\mathsf{WorldChampionPlayer}, \mathsf{WorldChampionPlayer}\} \subseteq \mathcal{L}_{\mathbf{fb}}(s_0^0)$, thus this node is recognized as clashing and $s_0^0.\mathrm{status} := \mathsf{UNSAT}$; $\mathcal{L}_{\mathbf{fb}}(s_0^1) := \mathcal{L}_{\mathbf{fb}}(s_0) \cup \{\mathsf{ChampionPlayer}_{\mathbf{wc10}}\}$;
- $\mathcal{L}_{\mathbf{wc10}}(s_0^1) := \mathcal{L}_{\mathbf{wc10}}(s_0^1) \cup \{\mathsf{ChampionPlayer}\}$ by A-rule;
- The local expansion on s_0^1 terminates here: a pair of or-successors s_0^2 and s_0^3 are generated by the application of $\sqcup$-rule;
- $\{\neg\mathsf{ChampionPlayer}, \mathsf{ChampionPlayer}\} \subseteq \mathcal{L}_{\mathbf{wc10}}(s_0^2)$, thus this node is recognized as clashing and $s_0^2.\mathrm{status} := \mathsf{UNSAT}$; $\mathcal{L}_{\mathbf{wc10}}(s_0^3) := \mathcal{L}_{\mathbf{wc10}}(s_0^1) \cup \{\forall\mathsf{playsFor}.\mathsf{WinnerTeam}\}$;
- The local expansion on s_0^3 terminates here: a successor $s_1 \in V_{\mathbf{nfl10}}$ is generated by the application of $\exists$-rule with $\mathcal{L}(\langle s_0^3, s_1\rangle) = \{\mathsf{playsFor}_{\mathbf{wc10}}\}$, $\mathcal{L}_{\mathbf{nfl10}}(s_1) := \{\neg\mathsf{WinnerTeam}, \top_{\mathbf{wc10}}\}$;
- $V_{\mathbf{fb}} := V_{\mathbf{fb}} \cup \{s_1\}$, $\mathcal{L}_{\mathbf{fb}}(s_1) := \{\top_{\mathbf{wc10}}\}$, $V_{\mathbf{wc10}} := V_{\mathbf{wc10}} \cup \{s_1\}$ by $\Delta\uparrow$, A and $\Delta\downarrow$-rules;
- $\mathcal{L}_{\mathbf{wc10}}(s_1) := \mathcal{L}_{\mathbf{wc10}}(s_1) \cup \{\mathsf{WinnerTeam}\}$ by $\forall$-rule;
- $\mathcal{L}_{\mathbf{fb}}(s_1) := \mathcal{L}_{\mathbf{fb}}(s_1) \cup \{\mathsf{WinnerTeam}\}$, $\mathcal{L}_{\mathbf{nfl10}}(s_1) := \mathcal{L}_{\mathbf{nfl10}}(s_1) \cup \{\mathsf{WinnerTeam}\}$ by A-rule;

Last rule application on s_1 leads to a clash, since $\{\neg\mathsf{WinnerTeam}, \mathsf{WinnerTeam}\} \subseteq \mathcal{L}_{\mathbf{nfl10}}(s_1)$. This causes the propagation of UNSAT status to s_0^3 and its predecessors until $s_0.\mathrm{status}$ is assigned UNSAT, closing the deduction. The resulting and-or graph is depicted in Fig. 4: links from OR nodes are represented by dotted arrows, while solid arrows represent links from AND nodes. $\diamond$

3.1 Algorithm correctness

We now prove that the presented algorithm is terminating, sound and complete w.r.t. CKR semantics. This is showed by a correspondence between the generated and-or graph and a CKR model. For space reasons, we provide only a sketch of the main steps.

Theorem 1 (Correctness). *Given a CKR $\mathfrak{K}$ and $\mathbf{d} \in \mathfrak{D}_\Gamma$ on the input, the tableaux algorithm always terminates and $\mathfrak{K}$ is $\mathbf{d}$-satisfiable iff it generates an and-or graph $G = \langle V, E, \mathcal{L}, \{\mathcal{L}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}\rangle$ s.t. for all $x \in V$ with $\mathsf{TOP} \in x.\mathrm{type}$, $x.\mathrm{status} = \mathsf{SAT}$.*

Proof (Sketch.). **Termination.** Follows from Theorem 2 (see below).

Soundness. Assume that the algorithm constructs an and-or graph G such that $x.\mathrm{status} = \mathsf{SAT}$ for all $x \in V$ with $\mathsf{TOP} \in x.\mathrm{type}$. We show that $\mathfrak{K}$ is $\mathbf{d}$-satisfiable by constructing a model $\mathfrak{I} = \{\mathcal{I}_\mathbf{e}\}_{\mathbf{e}\in\mathfrak{D}_\Gamma}$ from G s.t. $\Delta_\mathbf{d} \neq \emptyset$ as follows:

- let $S := \emptyset$; for $x = a^\mathbf{g} \in V$, $\mathsf{TOP} \in x.\mathrm{type}$: if $\mathsf{OR} \notin x.\mathrm{type}$, define $S := S \cup \{x\}$ and $a^{\mathcal{I}_\mathbf{e}} = x$ for $\mathbf{e} \in \mathfrak{D}_\Gamma$ s.t. $a^\mathbf{g} \in V_\mathbf{e}$. Otherwise, if $\mathsf{OR} \in x.\mathrm{type}$, let $S := S \cup \{y\}$ and $a^{\mathcal{I}_\mathbf{e}} = y$ for y an or-successor of x s.t. $y.\mathrm{status} = \mathsf{SAT}$ and $\mathsf{OR} \notin y.\mathrm{type}$.
- for $x \in S$, let $\Delta_\mathbf{e} := \Delta_\mathbf{e} \cup \{x\}$ if $\mathbf{e} \in \mathfrak{D}_\Gamma$ s.t. $x \in V_\mathbf{e}$.
- for $x \in S$, let $A^{\mathcal{I}_\mathbf{e}} := A^{\mathcal{I}_\mathbf{e}} \cup \{x\}$ for all $A \in \mathcal{L}_\mathbf{e}(x)$ for $\mathbf{e} \in \mathfrak{D}_\Gamma$ s.t. $x \in V_\mathbf{e}$.
- for $x \in S$, if y is a R-successor of x: if y is not an OR node, $R^{\mathcal{I}_\mathbf{e}} := R^{\mathcal{I}_\mathbf{e}} \cup \{\langle x, y\rangle\}$, for all $R \in \mathcal{L}(\langle x, y\rangle)$ and $\mathbf{e} \in \mathfrak{D}_\Gamma$ s.t. $x, y \in V_\mathbf{e}$; add y to S if it has not been visited yet. Otherwise, if y is an OR node and z is an or-successor of y s.t. $z.\mathrm{status} = \mathsf{SAT}$ and $\mathsf{OR} \notin z.\mathrm{type}$, $R^{\mathcal{I}_\mathbf{e}} := R^{\mathcal{I}_\mathbf{e}} \cup \{\langle x, z\rangle\}$, for all $R \in \mathcal{L}(\langle x, y\rangle)$ and $\mathbf{e} \in \mathfrak{D}_\Gamma$ s.t. $x, y \in V_\mathbf{e}$; add z to S if it has not been visited yet.

From this model definition, we can show that if $C \in \mathcal{L}_\mathbf{e}(x)$ then $x \in C^{\mathcal{I}_\mathbf{e}}$.

Moreover we can show that $\mathfrak{I}$ satisfies conditions (1–9) of Definition 4. In particular, we note that condition 8 is satisfied: To prove the $\subseteq$ inclusion, assume $\langle x, y\rangle \in R_\mathbf{f}^{\mathcal{I}_\mathbf{d}}$. Then by the construction $\langle x, y\rangle \in (\Delta_\mathbf{d} \times \Delta_\mathbf{d})$. Now we need to distinguish two cases. If $\langle x, y\rangle \in E$, then by application of the $\Delta\uparrow$-rule we obtain $x, y \in V_\mathbf{e}$ thus proving that also $\langle x, y\rangle \in R_\mathbf{f}^{\mathcal{I}_\mathbf{e}}$. The other case, when $\langle x, y\rangle \notin E$ is proven similarly, by showing the property for a $\langle x, y'\rangle \in E$ with y' an or-predecessor of y. On the other hand, to prove the $\supseteq$ inclusion, we assume that $\langle x, y\rangle \in R_\mathbf{f}^{\mathcal{I}_\mathbf{e}} \cap (\Delta_\mathbf{d} \times \Delta_\mathbf{d})$. We can now show that $\langle x, y\rangle \in R_\mathbf{f}^{\mathcal{I}_\mathbf{d}}$ by the application of the $\Delta\downarrow$-rule.

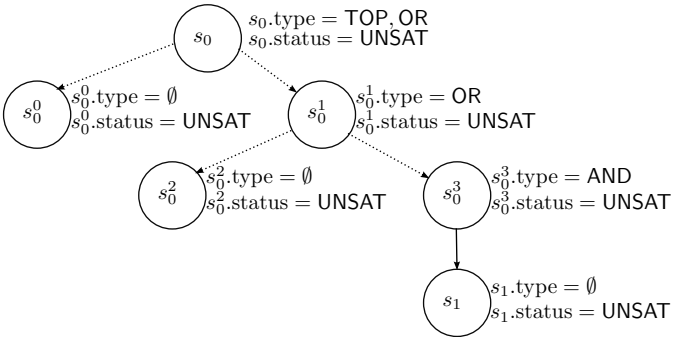

Fig. 4. And-or graph for Example 2

Thus, $\mathfrak{I}$ is indeed a CKR model for $\mathfrak{K}$ and by construction $\Delta_{\mathbf{d}} \neq \emptyset$.

Completeness. Let $\mathfrak{I}$ be a model of $\mathfrak{K}$ s.t. $\mathfrak{K}$ is $\mathbf{d}$-satisfiable w.r.t. $\mathfrak{I}$. The proof is by bi-simulation: we simulate the run of the algorithm on the given input following the branch identified by $\mathfrak{I}$ and inductively construct a partial mapping $\pi : V \to \bigcup_{\mathbf{e} \in \mathfrak{D}_\Gamma} \Delta_{\mathbf{e}}$ for which we will show the following property $(*)$:

> For each node $x \in V$ and for each $\mathbf{e} \in \mathfrak{D}_\Gamma$: (a) if $x \in V_{\mathbf{e}}$ then $\pi(x) \in \Delta_{\mathbf{e}}$; (b) if $C \in \mathcal{L}_{\mathbf{e}}(x)$ then $\pi(x) \in C^{\mathcal{I}_{\mathbf{e}}}$; (c) if $R \in \mathcal{L}(\langle x, y \rangle)$ with $x, y \in V_{\mathbf{e}}$ then $\langle \pi(x), \pi(y) \rangle \in R^{\mathcal{I}_{\mathbf{e}}}$; (d) if $a^{\mathbf{g}} \in V_{\mathbf{e}}$ then $a^{\mathcal{I}_{\mathbf{e}}} = \pi(a^{\mathbf{g}})$.

The proof proceeds by showing that $(*)$ is verified on every node $x \in V$ after each rule application and algorithm step. For example, in the application of the $\sqcup$-rule, with $\mathsf{TOP} \notin x.\mathrm{type}$: if the rule is applicable, $C_1 \sqcup C_2 \in \mathcal{L}_{\mathbf{e}}(x)$. As $(*)$ was satisfied before the rule was applied, we have either $\pi(x) \in C_1^{\mathcal{I}_{\mathbf{e}}}$ or $\pi(x) \in C_2^{\mathcal{I}_{\mathbf{e}}}$. After the rule is applied, let us define $\pi(y) := \pi(x)$ in the case $\pi(x) \in C_1^{\mathcal{I}_{\mathbf{e}}}$ and $\pi(z) := \pi(x)$ otherwise. Hence $(*)$ is still satisfied after the rule was applied: basically, in the definition of π we follow a branch that is consistent with the interpretation $\mathfrak{I}$.

At the end of the run, we can show by contradiction that no node mapped by π contains a clash: suppose that for $x \in V$ (in the domain of π), $\mathbf{e} \in \mathfrak{D}_\Gamma$ and some concept D, we have both $D \in \mathcal{L}_{\mathbf{e}}(x)$ and $\neg D \in \mathcal{L}_{\mathbf{e}}(x)$. However, from $(*)$ this would imply that $\pi(x) \in D^{\mathcal{I}_{\mathbf{e}}} \cap \neg D^{\mathcal{I}_{\mathbf{e}}}$, contradicting the fact that $\mathfrak{I}$ is a CKR interpretation. □

3.2 Computational Complexity

Theorem 2 (Complexity). *The algorithm always terminates and runs in* ExpTime.

Proof (Sketch.). The size of the input KB will be denoted by $|\mathfrak{K}| = |\mathfrak{M}| + \sum_{\mathcal{C} \in \mathfrak{C}} |\mathrm{K}(\mathcal{C})| = m$. Observe that for the number n of contexts in $\mathfrak{K}$ we have $n \leq m$ as adding a context into $\mathfrak{C}$ requires to add a number of axioms into $\mathfrak{M}$. There are at most m^2 possible concepts that can possibly appear in one of the labels. There are at most n labels, therefore the number of concepts in all node's labels is bounded by m^3. Therefore there are at most 2^{m^3} different nodes. For the nodes that are subject to caching this is a firm bound. However, the initial nodes and their or-descendant are not cached. Under each initial node, the binary tree of or-descendants has the depth at most m^3 because with each or-branching at least one label is expanded. Hence the maximum number of these nodes altogether is $m \times 2^{m^3}$. By analysing the main loop and each of the rules we are able to estimate the maximum number of steps required to generate one node by m^6. Hence the total number of steps required to generate the whole and-or graph is bounded by $m^7 \times 2^{m^3}$ and this is bounded by $O(2^{m^5})$ as for $m > 4$ we have $m^7 \leq 2^{m^2}$. Finally, the propagation of the status bottom-up can be alternatively emulated by breadth-first search once the graph is fully generated. Breadth-first search is quadratic in the number of nodes, hence it takes at most $(m \times 2^{m^3})^2$ which is smaller then $O(2^{m^5})$. And hence, as the algorithm is deterministic, it is in ExpTime. □

4 Related Works

The principal references for the definition of our algorithm are the works by Goré and Nguyen proposing deterministic EXPTIME procedures for $\mathcal{ALC}$ [6] and its extensions (e.g. for $\mathcal{SHI}$ in [7]). For our goals, we only adapted the non optimized version of the algorithm from [6]. However, also by the fact that we follow the same *never look behind* approach in our rules and we work on a similar and-or structure, we could easily adopt some of the optimizations from [6]. For example, we can *avoid redundant computations* by noting that, if x is an AND node we can mark it as UNSAT as soon as an UNSAT successor is found (and similarly for OR nodes and SAT successors). We can also investigate *different search strategies* and *cutoffs* for our algorithm, to possibly prune unuseful branches on the base of node statuses.

The main complexity result for CKR has been described in [11]. Through a polynomial reduction to $\mathcal{SROIQ}$, it is proved that reasoning in $\mathcal{SROIQ}$-based CKR is 2NEXPTIME-complete. Thus, the contextualization of knowledge provided by CKR does not imply a complexity jump in reasoning. This "non-jumping property", however, can not be guaranteed for weaker languages by reusing the same translation: given that it introduces role chain axioms in the target, the resulting KB is strictly in $\mathcal{SROIQ}$. This property has been verified for $\mathcal{ALC}$-based CKR by proposing a different reduction in [3]. In order to stay in EXPTIME, the reduction has been adapted to produce an $\mathcal{ALCO}(\sqcup)$ KB. However, using this reduction for reasoning is not practically efficient: the translation adds a large (cubic) number of axioms in order to track complex relations between qualified symbols in a single KB. Thus, while the presented EXPTIME procedure may not add novelty to the complexity result, it clearly allows for more effective reasoning: local reasoning is executed on single node labels and only relevant consequences are propagated to other contexts.

A related proposal for the contextualization of the $\mathcal{ALC}$ is $\mathcal{ALC}_{\mathcal{ALC}}$ [9], a multimodal extension of $\mathcal{ALC}$ which shares with CKR the possibility to formalize the contextual structure in a distinct meta-language. The two frameworks differ from the expressivity of the contextual assertions: for instance, in $\mathcal{ALC}_{\mathcal{ALC}}$ it is also possible to qualify knowledge with respect to classes of contexts. Complexity of reasoning in $\mathcal{ALC}_{\mathcal{ALC}}$ jumps to 2EXPTIME: on the other hand, the result presented for CKR proves that contextualization of $\mathcal{ALC}$ can be obtained without such a complexity jump.

5 Conclusions

Contextualized Knowledge Repository (CKR) is a DL-based representation framework providing a contextual layer for Semantic Web knowledge resources. CKR knowledge bases can be built on top of $\mathcal{SROIQ}$ or any of its fragments: one relevant fragment is the $\mathcal{ALC}$ DL, for which reasoning is known to be EXPTIME-complete. In this paper we presented a sound and complete EXPTIME tableaux algorithm for $\mathcal{ALC}$ based CKR. While it was already known that reasoning in $\mathcal{ALC}$ based CKR is EXPTIME-complete [3], this is the first direct tableaux procedure for this fragment meeting the complexity bound. The algorithm has been defined starting from a NEXPTIME tableaux algorithm [4,8] by adopting the approach introduced for $\mathcal{ALC}$ in [6]. Our procedure is extended to manage CKR contextual semantics and to include ABox reasoning.

As for future directions, we would like to extend the procedure to more expressive DL (e.g. $\mathcal{SROIQ}$) and study its optimizations, possibly by adapting some of the ideas in [6], in view of an implementation. Moreover, the fact that contexts are compatible but, to a certain extent, also independent, may be exploited [4] to study parallelization and distribution of reasoning.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook. Cambridge University Press (2003)
2. Bao, J., Tao, J., McGuinness, D.L.: Context representation for the semantic web. In: WebSci10 (2010)
3. Bozzato, L., Homola, M., Serafini, L.: ExpTime reasoning for contextualized $\mathcal{ALC}$. Tech. Rep. TR-FBK-DKM-2012-1, Fondazione Bruno Kessler, Trento, Italy (2012), `http://dkm.fbk.eu/index.php/Resources`
4. Bozzato, L., Homola, M., Serafini, L.: Towards More Effective Tableaux Reasoning for CKR. In: DL2012. CEUR-WP, vol. 824, pp. 114–124. CEUR-WS.org (2012)
5. Donini, F.M., Massacci, F.: EXPTIME tableaux for $\mathcal{ALC}$. Artif. Intell. 124(1), 87–138 (2000)
6. Goré, R., Nguyen, L.: EXPTIME tableaux for $\mathcal{ALC}$ using sound global caching. In: DL2007. CEUR-WP, vol. 250, pp. 299–306. CEUR-WS.org (2007)
7. Goré, R., Nguyen, L.: EXPTIME Tableaux with Global Caching for Description Logics with Transitive Roles, Inverse Roles and Role Hierarchies. In: TABLEAUX 2007. LNCS, vol. 4548, pp. 133–148. Springer (2007)
8. Homola, M., Bozzato, L., Serafini, L.: Tableaux algorithm for reasoning with contextualized knowledge. Tech. Rep. TR-FBK-DKM-2011-1, Fondazione Bruno Kessler, Trento, Italy (2012), `http://dkm.fbk.eu/index.php/Resources`
9. Klarman, S., Gutiérrez-Basulto, V.: Two-dimensional description logics for context-based semantic interoperability. In: AAAI-11. AAAI Press (2011)
10. Schmidt-Schauß, M., Smolka, G.: Attributive concept descriptions with complements. Artificial Intelligence 48(1), 1–26 (1991)
11. Serafini, L., Homola, M.: Contextualized knowledge repositories for the semantic web. J. of Web Sem., Special Issue: Reasoning with context in the Semantic Web 12 (2012)