Terminological Reasoning in $\mathcal{SHIQ}$ with Ordered Binary Decision Diagrams

Sebastian Rudolph and **Markus Krötzsch** and **Pascal Hitzler**
Institute AIFB, Universität Karlsruhe, Germany
{sru,mak,phi}@aifb.uni-karlsruhe.de

Abstract

We present a new algorithm for reasoning in the description logic $\mathcal{SHIQ}$, which is the most prominent fragment of the Web Ontology Language OWL. The algorithm is based on ordered binary decision diagrams (OBDDs) as a datastructure for storing and operating on large model representations. We thus draw on the success and the proven scalability of OBDD-based systems. To the best of our knowledge, we present the very first agorithm for using OBDDs for reasoning with general Tboxes.

Introduction

In order to leverage intelligent applications for the Semantic Web, scalable reasoning systems for the standardised Web Ontology Language OWL[1] are required. OWL is essentially based on description logics (DLs), with the DL known as $\mathcal{SHIQ}$ currently being among its most prominent fragments. State-of-the art OWL reasoners, such as Pellet, RacerPro or KAON2 already achieve an efficiency which makes them suitable for practical use, however they still fall short of the scalability requirements needed for large-scale applications. New ideas and approaches are therefore needed to further push the performance of OWL reasoning.

In this paper, we present a promising new algorithm for reasoning with $\mathcal{SHIQ}$, which is based on ordered binary decision diagrams (OBDDs) as a datastructure for storing and operating on large model representations (Bryant 1986; Huth & Ryan 2000). The rationale behind the approach is the fact that OBDD-based systems feature impressive efficiency on large amounts of data, e.g. for model checking for hard- and software verification (Burch *et al.* 1990). Our algorithm is indeed based on a reduction of $\mathcal{SHIQ}$ reasoning to standard OBDD-algorithms, and thus allows to draw on available algorithms and standard implementations for OBDDs, such as JavaBDD[2].

The general idea of using OBDDs for reasoning with DLs is not entirely new, and some related results have already been presented in (Pan, Sattler, & Vardi 2006). A closer look also reveals that certain temporal logics to which OBDDs have been applied (e.g. CTL (Huth & Ryan 2000)) are closely related to modal logics which in turn are known to have strong structural similarities to DLs (Schild 1991). Hence it seems almost natural to apply OBDD-based techniques for DL reasoning as well. The results from (Pan, Sattler, & Vardi 2006), however, are still rather restricted since they encompass only terminological reasoning in the basic DL $\mathcal{ALC}$ without general Tboxes.

In essence, OBDDs can be used to represent arbitrary Boolean functions. These Boolean functions are then interpreted as a kind of compressed encoding of – usually very large sets of – process states. Model checking and certain manipulations of the state space can then be done directly on this compressed version without unfolding it. In our approach, we employ OBDDs in a very similar way for encoding DL interpretations. However, as DL reasoning is concerned with all possible models, we will show by model-theoretic arguments that for our purposes it is sufficient to work only with certain representative models.

A birds eyes' perspective on our results is as follows: $\mathcal{SHIQ}$ knowledge bases can be reduced equisatisfiably to $\mathcal{ALCI}b$ knowledge bases. A sound and complete decision procedure based on so-called *domino interpretations* provides the next step. This procedure can in turn be realised by manipulating Boolean functions, which establishes the link with OBDD-algorithms. We will present our results in this sequence, after introducing some notation.

Proofs were omitted due to lack of space, but can be found in (Rudolph, Krötzsch, & Hitzler 2008).

Preliminaries

We first recall some basic definitions of DLs (see (Baader *et al.* 2007) for a comprehensive treatment of DLs) and introduce our notation. Next we define a rather expressive description logic $\mathcal{SHIQ}b$ that extends $\mathcal{SHIQ}$ with restricted Boolean role expressions (Tobies 2001). We will not consider $\mathcal{SHIQ}b$ knowledge bases, but the DL serves as a convenient umbrella logic for the DLs used in this paper. Also, we do not consider assertional knowledge, and hence will only introduce terminological axioms here.

Definition 1 A terminological $\mathcal{SHIQ}b$ knowledge base is based on two disjoint sets of *concept names* N_C and *role names* N_R. A set of *atomic roles* $\mathbf{R}$ is defined as $\mathbf{R} :=$

[1]http://www.w3.org/2004/OWL/

[2]http://javabdd.sourceforge.net

$\mathsf{N}_R \cup \{R^- \mid R \in \mathsf{N}_R\}$. In addition, we set $\mathrm{Inv}(R) := R^-$ and $\mathrm{Inv}(R^-) := R$, and we will extend this notation also to sets of atomic roles. In the sequel, we will use the symbols R, S to denote atomic roles, if not specified otherwise.

The set of *Boolean role expressions* $\mathbf{B}$ is defined as

$$\mathbf{B} ::= \mathbf{R} \mid \neg\mathbf{B} \mid \mathbf{B} \sqcap \mathbf{B} \mid \mathbf{B} \sqcup \mathbf{B}.$$

We use $\vdash$ to denote standard Boolean entailment between sets of atomic roles and role expressions. Given a set $\mathcal{R}$ of atomic roles, we inductively define:

- For atomic roles R, $\mathcal{R} \vdash R$ if $R \in \mathcal{R}$, and $\mathcal{R} \nvdash R$ otherwise,
- $\mathcal{R} \vdash \neg U$ if $\mathcal{R} \nvdash U$, and $\mathcal{R} \nvdash \neg U$ otherwise,
- $\mathcal{R} \vdash U \sqcap V$ if $\mathcal{R} \vdash U$ and $\mathcal{R} \vdash V$, and $\mathcal{R} \nvdash U \sqcap V$ otherwise,
- $\mathcal{R} \vdash U \sqcup V$ if $\mathcal{R} \vdash U$ or $\mathcal{R} \vdash V$, and $\mathcal{R} \nvdash U \sqcup V$ otherwise.

A Boolean role expression U is *restricted* if $\emptyset \nvdash U$. The set of all restricted role expressions is denoted $\mathbf{T}$, and the symbols U and V will be used throughout this paper to denote restricted role expressions. A $\mathcal{SHIQ}b$ *Rbox* is a set of axioms of the form $U \sqsubseteq V$ (role inclusion axiom) or $\mathsf{Tra}(R)$ (transitivity axiom). The set of non-simple roles (for a given Rbox) is inductively defined as follows:

- If there is an axiom $\mathsf{Tra}(R)$, then R is non-simple.
- If there is an axiom $R \sqsubseteq S$ with R non-simple, then S is non-simple.
- If R is non-simple, then $\mathrm{Inv}(R)$ is non-simple.

A role is *simple* if it is atomic (simplicity of Boolean role expressions is not relevant in this paper) and not non-simple. Based on a $\mathcal{SHIQ}b$ Rbox, the set of *concept expressions* $\mathbf{C}$ is the smallest set containing N_C, and all concept expressions given in Table 1, where $C, D \in \mathbf{C}$, $U \in \mathbf{T}$, and $R \in \mathbf{R}$ is a simple role. Throughout this paper, the symbols C, D will be used to denote concept expressions. A $\mathcal{SHIQ}b$ *Tbox* is a set of *general concept inclusion axioms* (GCIs) of the form $C \sqsubseteq D$. A $\mathcal{SHIQ}b$ *knowledge base* KB is the union of a $\mathcal{SHIQ}b$ Rbox and an according $\mathcal{SHIQ}b$ Tbox.

As mentioned above, we will consider only fragments of $\mathcal{SHIQ}b$. In particular, a $\mathcal{SHIQ}$ knowledge base is a $\mathcal{SHIQ}b$ knowledge base without Boolean role expressions, and an $\mathcal{ALCI}b$ knowledge base is a $\mathcal{SHIQ}b$ knowledge base that contains no Rbox axioms and no number restrictions (i.e. axioms $\leq n R.C$ or $\geq n R.C$). The related DL $\mathcal{ALCQI}b$ has been studied in (Tobies 2001).

An interpretation $\mathcal{I}$ consists of a set $\Delta^\mathcal{I}$ called *domain* (the elements of it being called *individuals*) together with a function $\cdot^\mathcal{I}$ mapping individual names to elements of $\Delta^\mathcal{I}$, concept names to subsets of $\Delta^\mathcal{I}$, and role names to subsets of $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The function $\cdot^\mathcal{I}$ is extended to role and concept expressions as shown in Table 1. An interpretation $\mathcal{I}$ *satisfies* an axiom φ if we find that $\mathcal{I} \models \varphi$, where

- $\mathcal{I} \models U \sqsubseteq V$ if $U^\mathcal{I} \subseteq V^\mathcal{I}$,
- $\mathcal{I} \models \mathsf{Tra}(R)$ if $R^\mathcal{I}$ is a transitive relation,
- $\mathcal{I} \models C \sqsubseteq D$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$.

$\mathcal{I}$ *satisfies* a knowledge base KB, $\mathcal{I} \models$ KB, if it satisfies all axioms of KB. *Satisfiability*, *equivalence*, and *equisatisfiability* of knowledge bases are defined as usual.

Name	Syntax	Semantics
inverse role	R^-	$\{\langle x,y\rangle \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid \langle y,x\rangle \in R^\mathcal{I}\}$
role negation	$\neg U$	$\{\langle x,y\rangle \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid \langle x,y\rangle \notin U^\mathcal{I}\}$
role conj.	$U \sqcap V$	$U^\mathcal{I} \cap V^\mathcal{I}$
role disj.	$U \sqcup V$	$U^\mathcal{I} \cup V^\mathcal{I}$
top	$\top$	$\Delta^\mathcal{I}$
bottom	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^\mathcal{I} \setminus C^\mathcal{I}$
conjunction	$C \sqcap D$	$C^\mathcal{I} \cap D^\mathcal{I}$
disjunction	$C \sqcup D$	$C^\mathcal{I} \cup D^\mathcal{I}$
univ. rest.	$\forall U.C$	$\{x \in \Delta^\mathcal{I} \mid \langle x,y\rangle \in U^\mathcal{I}$ implies $y \in C^\mathcal{I}\}$
exist. rest.	$\exists U.C$	$\{x \in \Delta^\mathcal{I} \mid y \in \Delta^\mathcal{I}\colon \langle x,y\rangle \in U^\mathcal{I}, y \in C^\mathcal{I}\}$
qualified	$\leq n\, R.C$	$\{x \in \Delta^\mathcal{I} \mid \#\{y \in \Delta^\mathcal{I} \mid \langle x,y\rangle \in R^\mathcal{I}, y \in C^\mathcal{I}\} \leq n\}$
number rest.	$\geq n\, R.C$	$\{x \in \Delta^\mathcal{I} \mid \#\{y \in \Delta^\mathcal{I} \mid \langle x,y\rangle \in R^\mathcal{I}, y \in C^\mathcal{I}\} \geq n\}$

Table 1: Semantics of role (top) and concept constructors (bottom) in $\mathcal{SHIQ}b$ for an interpretation $\mathcal{I}$ with domain $\Delta^\mathcal{I}$.

For convenience of notation, we abbreviate Tbox axioms of the form $\top \sqsubseteq C$ by writing just C. Statements such as $\mathcal{I} \models C$ and $C \in$ KB are interpreted accordingly. Note that $C \sqsubseteq D$ can thus be written as $\neg C \sqcup D$.

Finally, we will often need to access a particular set of quantified and atomic subformulae of a DL concept. These specific parts are provided by the function $P : \mathbf{C} \to 2^\mathbf{C}$:

$$P(C) := \begin{cases} P(D) & \text{if } C = \neg D \\ P(D) \cup P(E) & \text{if } C = D \sqcap E \text{or } C = D \sqcup E \\ \{C\} \cup P(D) & \text{if } C = \mathsf{Q}U.D \text{ with } \mathsf{Q} \in \{\exists, \forall, \geq n, \leq n\} \\ \{C\} & \text{otherwise} \end{cases}$$

We generalise P to DL knowledge bases KB by defining $P(\mathrm{KB})$ to be the union of the sets $P(C)$ for all Tbox axioms C in KB.

For our further considerations, we will usually express all Tbox axioms as simple concept expressions as explained above. Given a knowledge base KB we obtain its negation normal form $\mathsf{NNF}(\mathrm{KB})$ by converting every Tbox concept into its negation normal form as usual.

It is well-known that KB and $\mathsf{NNF}(\mathrm{KB})$ are equivalent. We will usually require another normalisation step that simplifies the structure of KB by *flattening* it to a knowledge base $\mathsf{FLAT}(\mathrm{KB})$. This is achieved by transforming KB into negation normal form and exhaustively applying the following transformation rules:

- Select an outermost occurrence of $\mathsf{Q}U.D$ in KB, such that $\mathsf{Q} \in \{\exists, \forall, \geq n, \leq n\}$ and D is a non-atomic concept.
- Substitute this occurrence with $\mathsf{Q}U.F$ where F is a fresh concept name (i.e. one not occurring in the knowledge base).
- If $\mathsf{Q} \in \{\exists, \forall, \geq n\}$, add $\neg F \sqcup D$ to the knowledge base.
- If $\mathsf{Q} = \leq n$ add $\mathsf{NNF}(\neg D) \sqcup F$ to the knowledge base.

Obviously, this procedure terminates yielding a flat knowledge base $\mathsf{FLAT}(\mathrm{KB})$ all Tbox axioms of which are Boolean expressions over formulae of the form A, $\neg A$, or $\mathsf{Q}U.A$ with A an atomic concept name.

Proposition 2 Any $\mathcal{SHIQ}b$ knowledge base KB is equisatisfiable to $\mathsf{FLAT}(\mathrm{KB})$.

From $\mathcal{SHIQ}$ to $\mathcal{ALCI}b$

Next, we present a stepwise satisfiability-preserving and polynomial-time transformation from the quite common description logic $\mathcal{SHIQ}$ to the rather "exotic" $\mathcal{ALCI}b$. This will allow to apply the presented reasoning algorithm to terminological $\mathcal{SHIQ}$ knowledge bases.

From $\mathcal{SHIQ}$ to $\mathcal{ALCHIQ}$. As has been shown in (Motik 2006), every $\mathcal{SHIQ}$ knowledge base KB can be converted into an equisatisfiable $\mathcal{ALCHIQ}$ knowledge base, where $\mathcal{ALCHIQ}$ denotes the description logic $\mathcal{SHIQ}$ without transitivity axioms. We let $\Theta_{\mathcal{S}}(\mathrm{KB})$ denote the result of this reduction, which is known to be time polynomial.

From $\mathcal{ALCHIQ}$ to $\mathcal{ALCHI}b^{\leq}$. The DL $\mathcal{ALCHI}b^{\leq}$ is the fragment of $\mathcal{SHIQ}b$ that contains no transitivity axioms and no atleast restrictions ($\geq$). Given an $\mathcal{ALCHIQ}$ knowledge base KB, an $\mathcal{ALCHI}b^{\leq}$ knowledge base $\Theta_{\geq}(\mathrm{KB})$ is obtained by first flattening KB and then applying the following steps until all $\geq n\,R.A$ have been eliminated.

- Choose an occurrence of a subconcept of form $\geq n\,R.A$ in the knowledge base.
- Substitute this occurrence by $\exists R_1.A \sqcap \ldots \sqcap \exists R_n.A$, where $R_1, \ldots, R_n$ are fresh role names.
- For every $i \in \{1, \ldots, n\}$, add $R_i \sqsubseteq R$ to the Rbox.
- For every $1 \leq i < k \leq n$, add $\forall(R_i \sqcap R_k).\bot$ to the Tbox.

Observe that this transformation can be done in polynomial time for unary coding of numbers. Note that the same can be achieved for a binary encoding by using fresh roles as binary digits for complex roles.

Lemma 3 Any $\mathcal{ALCHIQ}$ knowledge base KB is equisatisfiable to the $\mathcal{ALCHI}b^{\leq}$ knowledge base $\Theta_{\geq}(\mathrm{KB})$.

From $\mathcal{ALCHI}b^{\leq}$ to $\mathcal{ALCI}b^{\leq}$. In the presence of restricted role expressions, role subsumption axioms can be easily transformed into Tbox axioms, as the subsequent lemma shows. This allows to dispense with role hierarchies in $\mathcal{ALCHI}b^{\leq}$ thereby restricting it to $\mathcal{ALCI}b^{\leq}$.

Lemma 4 For any role names R, S, the Rbox axiom $R \sqsubseteq S$ and the Tbox axiom $\forall(R \sqcap \neg S).\bot$ are equivalent.

Hence, for any $\mathcal{ALCHI}b^{\leq}$ knowledge base KB, let $\Theta_{\mathcal{H}}(\mathrm{KB})$ denote the $\mathcal{ALCI}b^{\leq}$ knowledge base obtained by substituting every Rbox axiom $R \sqsubseteq S$ by the Tbox axiom $\forall(R \sqcap \neg S).\bot$. The above lemma assures equivalence of KB and $\Theta_{\mathcal{H}}(\mathrm{KB})$ (and hence also their equisatisfiability). Obviously, this reduction can be done in linear time.

From $\mathcal{ALCI}b^{\leq}$ to $\mathcal{ALCIF}b$. $\mathcal{ALCIF}b$ is the fragment of $\mathcal{ALCI}b^{\leq}$ that contains $\leq$ only in functionality restrictions, i.e. axioms of the form $\leq 1\,R.\top$. Given an $\mathcal{ALCI}b^{\leq}$ knowledge base KB, we obtain the $\mathcal{ALCIF}b$ knowledge base $\Theta_{\mathcal{F}}(\mathrm{KB})$ by first flattening KB and then applying the following steps until no more steps are applicable:

- Choose an occurrence of a subconcept of form the shape $\leq n\,R.A$ which is not a functionality axiom $\leq 1\,R.\top$.
- Substitute this occurrence by $\forall(R \sqcap \neg R_1 \sqcap \ldots \sqcap \neg R_n).\neg A$ where $R_1, \ldots, R_n$ are fresh role names.
- For each $i \in \{1, \ldots, n\}$, add $\forall R_i.A$ and $\leq 1\,R_i.\top$ to the Tbox.

Obviously, this transformation can be done in polynomial time (again assuming a unary encoding of the n), and we establish the following equisatisfiability result.

Lemma 5 Any $\mathcal{ALCI}b^{\leq}$ knowledge base KB is equisatisfiable to the $\mathcal{ALCIF}b$ knowledge base $\Theta_{\leq}(\mathrm{KB})$.

From $\mathcal{ALCIF}b$ to $\mathcal{ALCI}b$. We now show how the role functionality axioms of the shape $\leq 1\,R.\top$ can be eliminated from $\mathcal{ALCIF}b$ knowledge base.

Essentially, we do so by adding axioms that enforce that, for every functional role R, any two R-successors coincide with respect to their properties expressible in "relevant" DL terms. While it is rather obvious that those axioms follow from R's functionality, the other direction (a Leibniz-style "identitas indiscernibilium" argument) needs a closer look and some intermediate constructions and results that can be found in the accompanying technical report.

For an $\mathcal{ALCIF}b$ knowledge base KB, let $\Theta_{\mathcal{F}}(\mathrm{KB})$ denote the $\mathcal{ALCI}b$ knowledge base obtained from KB by replacing every role functionality axiom $\leq 1\,R.\top$ by axioms

- $\forall R.\neg D \sqcup \forall R.D$ for every $D \in P(\mathrm{KB} \setminus \{\leq 1\,R.\top \in \mathrm{KB}\})$,
- $\forall(R \sqcap S).\bot \sqcup \forall(R \sqcap \neg S).\bot$ for each atomic role S in KB.

Clearly, also this transformation can be done in polynomial time and space w.r.t. the size of KB, and we establish the missing link for the desired transformation.

Lemma 6 Any $\mathcal{ALCIF}b$ knowledge base KB is equisatisfiable to the $\mathcal{ALCI}b$ knowledge base $\Theta_{\mathcal{F}}(\mathrm{KB})$.

We have thus shown how to transform a $\mathcal{SHIQ}$ knowledge base KB into an equisatisfiable $\mathcal{ALCI}b$ knowledge base $\Theta_{\mathcal{F}}\Theta_{\leq}\Theta_{\mathcal{H}}\Theta_{\geq}\Theta_{\mathcal{S}}(\mathrm{KB})$ in polynomial time. As our next step towards checking satisfiability in $\mathcal{SHIQ}$, we can therefore construct satisfiability checking procedures for $\mathcal{ALCI}b$.

Building Models from Domino Sets

We now introduce the notion of a set of *dominoes* for a given terminological $\mathcal{ALCI}b$ knowledge base. Intuitively, each domino abstractly represents two individuals in an $\mathcal{ALCI}b$ interpretation, based on their concept properties and role relationships. We will see that suitable sets of such two-element pieces suffice to reconstruct models of $\mathcal{ALCI}b$, which also reveals certain model theoretic properties of this not so common DL. In particular, every satisfiable $\mathcal{ALCI}b$ Tbox admits tree-shaped models. This result is rather a by-product of our main goal of decomposing models into unstructured sets of local domino components, but it explains why our below constructions have some similarity with common approaches of showing tree-model properties by "unravelling" models.

After introducing the basics of domino representation, we present a first algorithm for deciding satisfiability of a $\mathcal{ALCI}b$ terminology based on sets of dominoes.

From Interpretations to Dominoes We first introduce the basic notion of a domino set, and its relationship to interpretations. Given a DL language with concepts $\mathbf{C}$ and roles $\mathbf{R}$, a *domino* is an arbitrary triple $\langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle$, where $\mathcal{A}, \mathcal{B} \subseteq \mathbf{C}$ and $\mathcal{R} \subseteq \mathbf{R}$. We will generally assume a fixed language and refer to dominoes over that language only. Interpretations can be deconstructed into sets of dominoes as follows.

Definition 7 Given an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, and a set $C \subseteq \mathbf{C}$ of concept expressions, the *domino projection* of $\mathcal{I}$ w.r.t. C, denoted by $\pi_C(\mathcal{I})$ is the set that contains for all $\delta, \delta' \in \Delta^{\mathcal{I}}$ the triple $\langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle$ with

- $\mathcal{A} = \{C \in C \mid \delta \in C^{\mathcal{I}}\}$,
- $\mathcal{R} = \{R \in \mathbf{R} \mid \langle \delta, \delta' \rangle \in R^{\mathcal{I}}\}$,
- $\mathcal{B} = \{C \in C \mid \delta' \in C^{\mathcal{I}}\}$.

It is easy to see that domino projections do not faithfully represent the structure of the interpretation that they were constructed from. But as we will see below, domino projections capture enough information to reconstruct models of a knowledge base KB, as long as C is chosen to contain at least $P(\mathrm{KB})$. For this purpose, we now introduce the inverse construction of interpretations from arbitrary domino sets.

Definition 8 Given a set $\mathbb{D}$ of dominoes, the induced *domino interpretation* $\mathcal{I}(\mathbb{D}) = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ is defined as follows:

1. $\Delta^{\mathcal{I}}$ consists of all finite nonempty words over $\mathbb{D}$ where, for each pair of subsequent letters $\langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle$ and $\langle \mathcal{A}', \mathcal{R}', \mathcal{B}' \rangle$ in a word, we have $\mathcal{B} = \mathcal{A}'$.
2. For $\delta = \langle \mathcal{A}_1, \mathcal{R}_1, \mathcal{A}_2 \rangle \langle \mathcal{A}_2, \mathcal{R}_2, \mathcal{A}_3 \rangle \ldots \langle \mathcal{A}_{i-1}, \mathcal{R}_{i-1}, \mathcal{A}_i \rangle$ a word and $A \in \mathsf{N}_C$ a concept name, we define $\mathrm{tail}(\delta) := \mathcal{A}_i$, and set $\delta \in A^{\mathcal{I}}$ iff $A \in \mathrm{tail}(\delta)$,
3. For each $R \in \mathsf{N}_R$, we set $\langle \delta_1, \delta_2 \rangle \in R^{\mathcal{I}}$ if either $\delta_2 = \delta_1 \langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle$ with $R \in \mathcal{R}$ or $\delta_1 = \delta_2 \langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle$ with $\mathrm{Inv}(R) \in \mathcal{R}$.

Mark that – following the intuition – the domino interpretation is constructed by conjoining matching dominoes. We find that certain domino projections contain enough information to reconstruct models of a knowledge base.

Proposition 9 Consider a set $C \subseteq \mathbf{C}$ of concept expressions, and an interpretation $\mathcal{J}$, and let $\mathcal{K} := \mathcal{I}(\pi_C(\mathcal{J}))$ denote the interpretation of the domino projection of $\mathcal{J}$ w.r.t. C. Then, for any $\mathcal{ALCI}b$ concept expression $C \in \mathbf{C}$ with $P(C) \subseteq C$, we have that $\mathcal{J} \models C$ iff $\mathcal{K} \models C$. Especially, for any $\mathcal{ALCI}b$ knowledge base KB, $\mathcal{J} \models \mathrm{KB}$ iff $\mathcal{I}(\pi_{P(\mathrm{KB})}(\mathcal{J})) \models \mathrm{KB}$.

Constructing Domino Sets The observation just made can be the basis for designing an algorithm that decides knowledge base satisfiability. Checking satisfiability often amounts to the attempt to construct a (representation of a) model. As we have seen, we may achieve this by trying to construct a model's domino projection. If this can be done, we know that there is a model, if not, there is none.

In what follows, we first describe the iterative construction of such a domino set from a given knowledge base, and then show that it is indeed a decision procedure for knowledge base satisfiability.

Definition 10 Consider an $\mathcal{ALCI}b$ knowledge base KB, and define $C = P(\mathsf{FLAT}(\mathrm{KB}))$. Sets $\mathbb{D}_i$ of dominoes based on concepts from C are constructed as follows:

$\mathbb{D}_0$ consists of all dominoes $\langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle$ which satisfy:

kb: for every concept $C \in \mathsf{FLAT}(\mathrm{KB})$, we have that $\bigsqcap_{D \in \mathcal{A}} D \sqsubseteq C$ is a tautology[3],

ex: for all $\exists U.A \in C$, if $A \in \mathcal{B}$ and $\mathcal{R} \vdash U$ then $\exists U.A \in \mathcal{A}$,

uni: for all $\forall U.A \in C$, if $\forall U.A \in \mathcal{A}$ and $\mathcal{R} \vdash U$ then $A \in \mathcal{B}$.

Given a domino set $\mathbb{D}_i$, the set $\mathbb{D}_{i+1}$ consists of all dominoes $\langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle \in \mathbb{D}_i$ satisfying the following conditions:

delex: for every $\exists U.A \in C$ with $\exists U.A \in \mathcal{A}$, there is some $\langle \mathcal{A}, \mathcal{R}', \mathcal{B}' \rangle \in \mathbb{D}_i$ such that $\mathcal{R}' \vdash U$ and $A \in \mathcal{B}'$,

deluni: for every $\forall U.A \in C$ with $\forall U.A \notin \mathcal{A}$, there is some $\langle \mathcal{A}, \mathcal{R}', \mathcal{B}' \rangle \in \mathbb{D}_i$ such that $\mathcal{R}' \vdash U$ but $A \notin \mathcal{B}'$,

sym: $\langle \mathcal{B}, \mathrm{Inv}(\mathcal{R}), \mathcal{A} \rangle \in \mathbb{D}_i$.

The construction of domino sets $\mathbb{D}_{i+1}$ is continued until $\mathbb{D}_{i+1} = \mathbb{D}_i$. The final result $\mathbb{D}_{\mathrm{KB}} := \mathbb{D}_{i+1}$ defines the *canonical domino set* of KB. The algorithm returns "unsatisfiable" if $\mathbb{D}_{\mathrm{KB}} = \emptyset$, and "satisfiable" otherwise.

Since $\mathbb{D}_0$ is exponential in the size of the knowledge base, the iterative deletion of dominoes must terminate after finitely many steps. Below we will see that this procedure is indeed sound and complete for checking satisfiability.

Note that, in contrast to tableau procedures, the presented algorithm starts with a large set of dominoes and successively deletes undesired dominoes. Indeed, on can show that the constructed domino set is the largest such set from which a domino model can be obtained. The algorithm thus may seem to be of little practical use. In the next section, we will therefore refine the above algorithm to employ Boolean functions as efficient implicit representations of domino sets, such that the efficient computational methods of BDDs can be exploited. Domino sets, however, are well-suited for showing the required correctness properties.

An important property of domino interpretations constructed from canonical domino sets is that the (semantic) concept membership of an individual can typically be (syntactically) read from the domino it has been constructed of.

Lemma 11 Consider an $\mathcal{ALCI}b$ knowledge base KB with non-empty canonical domino set $\mathbb{D}_{\mathrm{KB}}$, and define $C := P(\mathsf{FLAT}(\mathrm{KB}))$ and $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle := \mathcal{I}(\mathbb{D}_{\mathrm{KB}})$. Then for all $C \in C$ and $\delta \in \Delta^{\mathcal{I}}$, we have that $\delta \in C^{\mathcal{I}}$ iff $C \in \mathrm{tail}(\delta)$. Moreover, $\mathcal{I} \models \mathsf{FLAT}(\mathrm{KB})$.

Lemma 11 shows soundness of our decision algorithm. Conversely, completeness can also be proven, which results – together with Proposition 2 – in the following theorem.

Theorem 12 A terminological $\mathcal{ALCI}b$ knowledge base KB is satisfiable iff its canonical domino set $\mathbb{D}_{\mathrm{KB}}$ is non-empty. Definition 10 thus defines a decision procedure for satisfiability of such $\mathcal{ALCI}b$ knowledge bases.

[3]Please note that the formulae in $\mathsf{FLAT}(\mathrm{KB})$ and in $\mathcal{A} \subseteq C$ are such that this can easily be checked by evaluating the Boolean operators in C as if $\mathcal{A}$ was a set of true propositional variables.

Sets as Boolean Functions

We will now introduce how large sets (in our case the canonical domino, respectively the intermediate sets during its construction) can be effectively represented implicitly via Boolean functions. This kind of encoding is rather standard within the field of BDD-based model checking. Due to space reasons, we will only give a very brief overview on OBDDs (for a general reference, see (Huth & Ryan 2000)) and not further elaborate on the technical details of their manipulation in this paper, however, the way of implementing our approach can be directly derived from the algorithm described in this section, as for every operation to be carried out on the Boolean functions (namely combining them, permutation of variables, instantiating variables etc.) there is an algorithmic counterpart for the BDD-based representation.

Boolean Functions and Operations We first explain how sets can be represented by means of Boolean functions. This will enable us, given a fixed finite base set S, to represent every family of sets $\mathbb{S} \subseteq 2^S$ by a single Boolean function.

A *Boolean function* on a set Var of variables is a function $\varphi : 2^{\mathsf{Var}} \to \{\mathit{true}, \mathit{false}\}$. The underlying intuition is that $\varphi(V)$ computes the truth value of a Boolean formula based on the assumption that exactly the variables of V are evaluated to *true*. A simple example are functions of the form $[\![v]\!]$ for some $v \in \mathsf{Var}$, which are defined as $[\![v]\!](V) := \mathit{true}$ iff $v \in V$.

Boolean functions over the same set of variables can be combined and modified in several ways. Firstly, there are the obvious Boolean operators for negation, conjunction, disjunction, and implication. By slight abuse of notation, we will use the common (syntactic) operator symbols $\neg$, $\wedge$, $\vee$, and $\to$ to also represent such (semantic) operators on Boolean functions. Given, e.g., Boolean functions φ and ψ, we find that $(\varphi \wedge \psi)(V) = \mathit{true}$ iff $\varphi(V) = \mathit{true}$ and $\psi(V) = \mathit{true}$. Another operation on Boolean functions is existential quantification over a set of variables $V \subseteq \mathsf{Var}$, written as $\exists V.\varphi$ for some function φ. Given an input set $W \subseteq \mathsf{Var}$ of variables, we define $(\exists V.\varphi)(W) = \mathit{true}$ iff *there is some* $V' \subseteq V$ such that $\varphi(V' \cup (W \setminus V)) = \mathit{true}$. In other words, there must be a way to set truth values of variables in V such that φ evaluates to *true*. Universal quantification is defined analogously, and we thus have $\forall V.\varphi := \neg\exists V.\neg\varphi$ as usual. Mark that our use of $\exists$ and $\forall$ overloads notation, and should not be confused with role restrictions in DL expressions.

Ordered Binary Decision Diagrams Ordered Binary Decision Diagrams are data structures that efficiently encode Boolean functions. Structurally, a *binary decision diagram* (BDD) is a directed acyclic graph whose nodes are labelled by a variable from Var. The only exception are two *terminal* nodes that are labelled by *true* and *false*, respectively. Every non-terminal node has two outgoing edges, again labelled by *true* and by *false*. Every BDD based on a variable set $\mathsf{Var} = \{v_1, \ldots, v_n\}$ represents an n-ary Boolean function $\varphi : 2^{\mathsf{Var}} \to \{\mathit{true}, \mathit{false}\}$. The value $\varphi(V)$ for some $V \subseteq \mathsf{Var}$ is determined by traversing the BDD, beginning from a distinguished root node: at a node labelled with $v \in \mathsf{Var}$, the evaluation proceeds with the node connected by the *true*-edge if $v \in V$, and with the node connected by the *false*-edge otherwise. If a terminal node is reached, its label is returned as a result. An *ordered* BDD (short OBDD) is a BDD for which there is a total order on Var such that any path in the BDD is strictly ascending w.r.t. that order.

For any Boolean function $\varphi : 2^{\mathsf{Var}} \to \{\mathit{true}, \mathit{false}\}$ and any ordering on Var there is (up to isomorphy) exactly one minimal OBDD realising it, called the *reduced* OBDD (ROBDD), and this minimal representative can be efficiently computed from any non-minimal OBDD. This is used to efficiently decide whether two OBDDs encode the same Boolean function. The function that yields *false* for every input is encoded by an ROBDD consisting of just two nodes: the *false*-node, marked as root, and the (unused) *true*-node.

OBDDs for some Boolean formula might be exponentially large in general, but often there is an order which allows for OBDDs of manageable size. Finding the optimal order is NP-complete, but heuristics have shown to yield good approximate solutions. Hence OBDDs are often conceived as efficiently compressed representations of Boolean functions. In addition, many operations on Boolean functions – such as the aforementioned "pointwise" negation, conjunction, disjunction, implication as well as propositional quantification – can be performed directly on the corresponding OBDDs by fast algorithms.

Translating Dominos into Boolean Functions Now, let $\mathrm{KB} = \mathsf{FLAT}(\mathrm{KB})$ be a flattened $\mathcal{ALCI}b$ knowledge base. The variable set Var is defined as $\mathsf{Var} := \mathbf{R} \cup (P(\mathrm{KB}) \times \{1, 2\})$. We thus obtain an obvious bijection between sets $V \subseteq \mathsf{Var}$ and dominoes over the set $P(\mathrm{KB})$ given as $\langle \mathcal{A}, \mathcal{R}, \mathcal{B} \rangle \mapsto (\mathcal{A} \times \{1\}) \cup \mathcal{R} \cup (\mathcal{B} \times \{2\})$. Hence, any Boolean function over Var represents a domino set as the collection of all variable sets for which it evaluates to *true*. We can use this observation to rephrase the construction of $\mathbb{D}_{\mathrm{KB}}$ in Definition 10 into an equivalent construction of a function $[\![\mathrm{KB}]\!]$.

We represent DL concepts C and role expressions U by characteristic Boolean functions over Var as follows. Note that the application of $\wedge$ results in another Boolean function, and is not to be understood as a syntactic formula.

$$[\![C]\!] := \begin{cases} \neg[\![D]\!] & \text{if } C = \neg D \\ [\![D]\!] \wedge [\![E]\!] & \text{if } C = D \sqcap E \\ [\![D]\!] \vee [\![E]\!] & \text{if } C = D \sqcup E \\ [\![\langle C, 1\rangle]\!] & \text{if } C \in P(\mathrm{KB}) \end{cases}$$

$$[\![U]\!] := \begin{cases} \neg[\![V]\!] & \text{if } U = \neg V \\ [\![V]\!] \wedge [\![W]\!] & \text{if } U = V \sqcap W \\ [\![V]\!] \vee [\![W]\!] & \text{if } U = V \sqcup W \\ [\![U]\!] & \text{if } U \in \mathbf{R} \end{cases}$$

We can now define an inferencing algorithm based on Boolean functions.

Definition 13 Given a flattened $\mathcal{ALCI}b$ knowledge base KB and a variable set Var as above, Boolean functions $[\![\mathrm{KB}]\!]_i$ are constructed based on the definitions in Fig. 1:

- $[\![\mathrm{KB}]\!]_0 := \varphi^{\mathbf{kb}} \wedge \varphi^{\mathbf{uni}} \wedge \varphi^{\mathbf{ex}}$,
- $[\![\mathrm{KB}]\!]_{i+1} := [\![\mathrm{KB}]\!]_i \wedge \varphi_i^{\mathbf{delex}} \wedge \varphi_i^{\mathbf{deluni}} \wedge \varphi_i^{\mathbf{sym}}$

The construction terminates as soon as $[\![\mathrm{KB}]\!]_{i+1} = [\![\mathrm{KB}]\!]_i$, and the result of the construction is then defined as $[\![\mathrm{KB}]\!] :=$

$$\varphi^{\mathbf{kb}} := \bigwedge_{C \in \mathrm{KB}} [\![C]\!] \qquad \varphi_i^{\mathbf{delex}} := \bigwedge_{\exists U.C \in P(\mathrm{KB})} [\![\langle \exists U.C, 1\rangle]\!] \to \exists(\mathbf{R} \cup C \times \{2\}).([\![\mathrm{KB}]\!]_i \wedge [\![U]\!] \wedge [\![\langle C, 2\rangle]\!])$$

$$\varphi^{\mathbf{uni}} := \bigwedge_{\forall U.C \in P(\mathrm{KB})} [\![\langle \forall U.C, 1\rangle]\!] \wedge [\![U]\!] \to [\![\langle C, 2\rangle]\!] \qquad \varphi_i^{\mathbf{deluni}} := \bigwedge_{\forall U.C \in P(\mathrm{KB})} [\![\langle \forall U.C, 1\rangle]\!] \to \neg\exists(\mathbf{R} \cup C \times \{2\}).([\![\mathrm{KB}]\!]_i \wedge [\![U]\!] \wedge \neg[\![\langle C, 2\rangle]\!])$$

$$\varphi^{\mathbf{ex}} := \bigwedge_{\exists U.C \in P(\mathrm{KB})} [\![\langle C, 2\rangle]\!] \wedge [\![U]\!] \to [\![\langle \exists U.C, 1\rangle]\!] \qquad \varphi_i^{\mathbf{sym}}(V) := [\![\mathrm{KB}]\!]_i\Big(\{\langle D, 1\rangle \mid \langle D, 2\rangle \in V\} \cup \{\mathrm{Inv}(R) \mid R \in V\} \cup \{\langle D, 2\rangle \mid \langle D, 1\rangle \in V\}\Big)$$

Figure 1: Boolean functions for defining the canonical domino set in Definition 13.

$[\![\mathrm{KB}]\!]_i$. The algorithm returns "unsatisfiable" if $[\![\mathrm{KB}]\!](V) = \mathit{false}$ for all $V \subseteq \mathsf{Var}$, and "satisfiable" otherwise.

We claim that the above algorithm is a correct procedure for checking consistency of terminological $\mathcal{ALCIb}$ knowledge bases. First, note that all necessary computation steps can indeed be implemented algorithmically: Any Boolean function can be evaluated for a fixed variable input V, and equality of two functions can (naively) be checked by comparing the results for all input sets (which are finitely many since Var is). The algorithm terminates since there can be only finitely many Boolean functions over Var.

Concerning soundness and completeness, it is easy to see that the Boolean operations used in constructing $[\![\mathrm{KB}]\!]$ directly correspond to the set operations in Definition 10, such that $[\![\mathrm{KB}]\!](V) = \mathit{true}$ iff V represents a domino in $\mathbb{D}_{\mathrm{KB}}$. Thus soundness and completeness is shown by Theorem 12.

Conclusions and Related Work

The main contribution of this paper is that it provides a new algorithm for terminological reasoning in the description logic $\mathcal{SHIQ}$, based on ordered binary decision diagrams, which is a substantial improvement to (Pan, Sattler, & Vardi 2006). Obviously, experiments will have to be done to investigate whether the conceptual insights really work in practice. A prototype implementation is under way, and will be reported on in the future. OBDDs have shown excellent practical performance in structurally and computationally similar domains, so that some hope for practical applicability of this approach seem to be justified.

Our major contributions in this paper are in fact twofold.

To prove correctness of our algorithm we had to elaborate on the model theoretic properties of $\mathcal{ALCIb}$. The technique was given in terms of Boolean functions being directly transferable into an algorithm based on OBDDs. We thereby provide the theoretical foundations for a novel paradigm for DL reasoning, which can be explored further in terms of implementations and evaluations, and also in other directions.

We also showed how a terminological $\mathcal{SHIQ}$ knowledge base can be converted into an equisatisfiable $\mathcal{ALCIb}$ knowledge base, thereby providing a foundational insight that reasoning in $\mathcal{SHIQ}$ can be done by developing reasoning solutions for $\mathcal{ALCIb}$. In particular, we showed that (qualified) number restrictions can be eliminated if allowing restricted complex role expressions.

The approach of constructing a canonical model (resp. a sufficient representation of it) in a downward manner (i.e. by pruning a larger structure) shows some similarity to Pratt's type elimination technique (Pratt 1979), originally used to decide satisfiability of modal formulae. Canonical models themselves are a widely used notion in modal logic (Blackburn, de Rijke, & Venema 2001), however, due to the additional expressive power of $\mathcal{ALCIb}$ compared to standard modal logics like K (the counterpart of the DL $\mathcal{ALC}$), we had to substantially modify this notion.

Besides implementation and evaluation, in the future we will extend our work towards Abox reasoning and to dealing with more expressive OWL DL constructs such as nominals.

Acknowledgements. Research reported in this paper was supported by the EU in the IST projects ACTIVE (IST-2007-215040) and NeOn (IST-2006-027595) as well as by the German Research Foundation under the ReaSem project.

References

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds. 2007. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

Blackburn, P.; de Rijke, M.; and Venema, Y. 2001. *Modal Logic*. Cambridge University Press.

Bryant, R. E. 1986. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers* 35(8):677–691.

Burch, J.; Clarke, E.; McMillan, K.; Dill, D.; and Hwang, L. 1990. Symbolic model checking: 10^{20} states and beyond. In *Proc. 5th Annual IEEE Symposium on Logic in Computer Science*, 1–33. Washington, D.C.: IEEE Computer Society Press.

Huth, M. R. A., and Ryan, M. D. 2000. *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge University Press.

Motik, B. 2006. *Reasoning in Description Logics using Resolution and Deductive Databases*. Ph.D. Dissertation, Universität Karlsruhe (TH), Germany.

Pan, G.; Sattler, U.; and Vardi, M. Y. 2006. BDD-based decision procedures for the modal logic K. *Journal of Applied Non-Classical Logics* 16(1-2):169–208.

Pratt, V. R. 1979. Models of program logics. In *Proc. 20th Annual Symposium on Foundations of Computer Science*.

Rudolph, S.; Krötzsch, M.; and Hitzler, P. 2008. OBDD-based Tbox reasoning in SHIQ. Technical report, Universität Karlsruhe.

Schild, K. 1991. A correspondence theory for terminological logics: Preliminary report. In *Proc. 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, 466–471.

Tobies, S. 2001. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. Ph.D. Dissertation, RWTH Aachen, Germany.