Solving Advanced Reasoning Tasks using Quantified Boolean Formulas*

Uwe Egly, Thomas Eiter, Hans Tompits, and Stefan Woltran
Technische Universität Wien
Abt. Wissensbasierte Systeme 184/3
Favoritenstraße 9–11, A–1040 Wien, Austria
e-mail: [uwe,eiter,tompits,stefan]@kr.tuwien.ac.at

Abstract

We consider the compilation of different reasoning tasks into the evaluation problem of quantified boolean formulas (QBFs) as an approach to develop prototype reasoning systems useful, e.g., for experimental purposes. Such a method is a natural generalization of a similar technique applied to NP-problems and has been recently proposed by other researchers. More specifically, we present translations of several well-known reasoning tasks from the area of nonmonotonic reasoning (NMR) into QBFs, and compare their implementation in the prototype system QUIP with established NMR-provers. The results show reasonable performance, and document that the QBF approach is an attractive tool for rapid prototyping of experimental knowledge-representation systems.

Introduction

Several important knowledge-representation tasks (KR tasks for short) can be efficiently reduced to SAT, the satisfiability problem of classical propositional logic. Thus, efficient algorithms for SAT can be used to solve such problems. Successful applications of this idea include, e.g., reductions of constrained-based planning problems to SAT (Kautz & Selman 1996).

The feasibility of this approach relies on the proviso that the considered problem is in NP, i.e., that it can be solved by a nondeterministic Turing machine working in polynomial time, and on the fact that SAT is the "prototypical" problem in NP. The latter property refers to the NP-completeness of SAT, stating that any problem in NP is polynomially expressible as SAT instance.

It is natural to apply an analogous method to problems beyond NP—in particular, many interesting KR problems are known to belong to PSPACE, the class of problems solvable in polynomial space. Now, since the prototypical PSPACE-problem is the evaluation of quantified boolean formulas (QBFs), these KR problems can thus be solved by efficient translations to QBFs.

*This work was partially supported by the Austrian Science Fund Project N Z29-INF.

In this paper, we present efficient (polynomial-time) translations of major reasoning problems from several propositional nonmonotonic reasoning (NMR) formalisms into QBFs. To the best of our knowledge, except for an encoding of conditional planning problems into QBFs (Rintanen 1999a), concrete transformations of KR tasks beyond NP into QBFs have not been presented so far. In particular, we provide polynomial-time translations of problems from abduction, default logic, autoepistemic logic, and disjunctive logic programming into QBFs. As well, we recall that propositional circumscription is *ipso facto* a QBF.

In addition, we present a protoype implementation, QUIP, for solving KR problems using the reductions discussed above. QUIP employs as underlying QBF-evaluator the commonly available propositional theorem prover `boole`, which is based on binary decision diagrams (BDDs). Choosing `boole` is motivated by the fact that it can handle arbitrary QBFs, and because it is a highly sophisticated package developed over many years.

In order to evaluate the feasibility of the method in practice, we compare the prototype system QUIP with existing NMR theorem provers. In particular, we compare the different translations of QUIP with Theorist (Poole 1989), DeRes (Cholewinski, Marek, & Truszczynski 1996), `dlv` (Eiter *et al.* 1998), and `smodels` (Niemelä & Simons 1996). As shown by the experimental results, even with no optimization methods applied, our (ad hoc) NMR implementations via QBFs compare reasonably well to these systems, some of which represent the state-of-the-art.

The approach discussed in this paper has been advocated in (Cadoli, Giovanardi, & Schaerf 1998; Rintanen 1999b), where algorithms for evaluating QBFs are presented. Although these Davis-Putnam style algorithms (like the resolution-style algorithm discussed in (Kleine Büning, Karpinski, & Flögel 1995)) could equally be used as underlying QBF-solvers, they suffer from the disadvantage that the input QBF is required to be in *prenex clausal normal form*, i.e., where all quantifiers must be at the front of the formula and its quantifier-free part must be in conjunctive normal form. As a consequence, since the "natural" reductions of KR problems to QBFs (as outlined in the present paper) do in general not yield a QBF in a particular normal form, the adoption of these algorithms would necessitate an additional normal form translation, which may result in an exponential blow-up of the resultant input QBF.

Translations into QBFs

In this section, we discuss how some well-known logical formalisms in AI can be mapped to QBFs in polynomial time. We focus here on major NMR formalisms. For space reasons, the exposition is necessarily succinct.

All formalisms are propositional. We assume a finite set of propositional variables V and constants 1, 0, denoting *truth* and *falsity*, respectively. The set $\mathcal{L}$ of propositional formulas is defined in the usual way, using the sentential connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$. Formulas will be denoted by Greek lower-case letters. A *theory*, T, is a finite set of formulas. In general, a theory T will be identified with the formula $\bigwedge_{\phi\in T}\phi$.

Quantified boolean formulas (QBFs) generalize ordinary propositional formulas by the admission of quantifications over propositional variables (QBFs are denoted by Greek upper-case letters).

The truth value of a QBF Φ without free variables (i.e., where all variables in Φ are within the scope of a quantifier) is recursively defined as follows:

- if $\Phi = 1$, then Φ is true;
- if $\Phi = 0$, then Φ is false;
- if $\Phi = \neg\Psi$, then Φ is true iff Ψ is false;
- if $\Phi = \Phi_1 \vee \Phi_2$, then Φ is true iff either Φ_1 or Φ_2 is true.
- If $\Phi = \exists x\,\Psi$, then Φ is true iff $\Psi[x/0]\vee\Psi[x/1]$ is true.

(Here, $\Psi[v/c]$ denotes the substitution of c for v in Ψ.) The cases of the remaining operators follow from the given ones in the usual way.

Let $S = \{\phi_1,\ldots,\phi_n\}$ and $T = \{\psi_1,\ldots,\psi_n\}$ be sets of formulas. Then, $S \leq T$ abbreviates $\bigwedge_{i=1}^{n}(\phi_i \rightarrow \psi_i)$, and $S < T$ is a shorthand for $(S \leq T) \wedge \neg(T \leq S)$. Furthermore, for a set $P = \{p_1,\ldots,p_n\}$ of propositional variables and a quantifier $\mathsf{Q} \in \{\forall,\exists\}$, we let $\mathsf{Q}P\,\phi$ stand for the formula $\exists p_1 \exists p_2 \cdots \exists p_n\,\phi$.

Abduction. Classical abduction from a theory T on V may be defined as follows (Selman & Levesque 1990; Poole 1989). Let $H \subseteq V$ be a set of *hypotheses*, and let $p \in V$ be a distinguished atom. A subset $E \subseteq H$ is an *abductive explanation* for p from T and H, if

(i) $T \cup E$ is consistent, and

(ii) $T \cup E \models p$, i.e., $T \cup E$ logically implies p.

An explanation E is *minimal*, if no proper subset $E' \subset E$ is an abductive explanation.

Assume that $H = \{h_1,\ldots,h_m\}$, and let $G = \{g_1,\ldots,g_m\}$ be a set of new propositional variables. The following QBF $\mathcal{T}_{abd}(H,p,T)$ expresses whether p has some abductive explanation (by monotonicity of classical logic, equivalently a minimal abductive explanation):

$$\exists G\Big[\exists V\Big(T \wedge (G \leq H)\Big) \wedge \forall V\Big((T \wedge (G \leq H)) \rightarrow p\Big)\Big].$$

Intuitively, G guesses an explanation, and the two conjuncts in the scope of $\exists G$ express conditions (i) and (ii), respectively.

The *relevance problem* is deciding whether a given hypothesis h belongs to some abductive explanation. This is expressed by the following QBF $\mathcal{T}_{abd}^{rel}(H,p,T,h)$:

$$\exists G\Big[\exists V\Big(T \wedge (G \leq H) \wedge h\Big) \wedge \forall V\Big((T \wedge (G \leq H) \wedge h) \rightarrow p\Big)\Big].$$

For minimal abductive explanations, the relevance problem is expressed by a QBF $\mathcal{T}_{abd}^{mrel}(H,p,T,h)$ which results from $\mathcal{T}_{abd}^{rel}(H,p,T,h)$ by adding within the scope of $\exists G$ a conjunct for the minimality check:

$$\bigwedge_{i=1}^{m}\Big[g_i \rightarrow \exists V\Big(T \wedge (G \setminus \{g_i\} \leq H \setminus \{h_i\}) \wedge \neg h_i \wedge \neg p\Big)\Big].$$

It encodes (in terms of the auxiliary variable set G) the well-known property that a set $E \subseteq H$ is minimal iff $E \setminus \{e\}$ is not an explanation, for any $e \in E$, i.e., $T \cup (E \setminus \{e\}) \cup \{\neg p\}$ is satisfiable.

Theorem 1 *The QBFs $\mathcal{T}_{abd}(H,p,T)$, $\mathcal{T}_{abd}^{rel}(H,p,T,h)$, and $\mathcal{T}_{abd}^{mrel}(H,p,T,h)$ evaluate to true iff the answer of the corresponding abduction task is "yes".*

Autoepistemic logic. The language of Moore's autoepistemic logic (Moore 1985) contains the modal operator L, where $L\phi$ intuitively means that ϕ is believed. By $\mathcal{L}_L$ we denote the language $\mathcal{L}$ extended by L. In what follows, formulas $L\phi$ are viewed as propositional variables, which are called *modal atoms*.

A *stable expansion* of an autoepistemic theory $T \subseteq \mathcal{L}_L$ is a set of formulas $E \subseteq \mathcal{L}_L$ such that

$$E = Cn(T \cup \{L\phi \mid \phi \in E\} \cup \{\neg L\phi \mid \phi \notin E\}),$$

where $Cn(\cdot)$ is the classical consequence operator with respect to the extended language $\mathcal{L}_L$.

The existence of a stable expansion can be expressed as follows (Niemelä 1992). Let $T \subseteq \mathcal{L}_L$ be an autoepistemic theory, M be the set of all modal atoms occurring in T, and V be the set of ordinary (non-modal) atoms in T. We say that $\Lambda \subseteq M \cup \{\neg\phi \mid \phi \in M\}$ is *T-full* iff, for all $L\phi \in M$, it holds that (i) $T \cup \Lambda \models \phi$ iff $L\phi \in \Lambda$, and (ii) $T \cup \Lambda \not\models \phi$ iff $\neg L\phi \in \Lambda$.

Proposition 1 (Niemelä 1992) *$T \subseteq \mathcal{L}_L$ has a stable expansion iff there exists a T-full set.*

For T, M, and V as above, this condition is easily translated into the following QBF $\mathcal{T}_{ael}(T)$:

$$\exists M\Big[\forall V\Big(T \rightarrow \bigwedge_{L\phi\in M}(L\phi \rightarrow \phi)\Big) \wedge \bigwedge_{L\phi\in M}\Big(\neg L\phi \rightarrow \exists V(T \wedge \neg\phi)\Big)\Big].$$

Theorem 2 *A finite autoepistemic theory $T \subseteq \mathcal{L}_L$ has a stable expansion iff $\mathcal{T}_{ael}(T)$ evaluates to true.*

Default Logic. A *default theory* is a pair $T = (W, \Delta)$, where $W \subseteq \mathcal{L}$ is a set of formulae and Δ is a set of *defaults* of the form $\frac{\alpha\,:\,\beta}{\gamma}$.[1] Intuitively, the default is applied (γ is concluded) if α is provable and the *justification* β can be consistently assumed. T is said to be *finite* iff W is finite.

The semantics of $T = (W, \Delta)$ is defined in terms of *extensions* (Reiter 1980). Following (Marek & Truszczyński 1993), extensions can be characterised thus. For any $S \subseteq \mathcal{L}$, let $\Delta(S)$ be the monotonic rules $\{\frac{\alpha}{\gamma} \mid \frac{\alpha\,:\,\beta}{\gamma} \in \Delta, \neg\beta \notin S\}$. Then, $E \subseteq \mathcal{L}$ is an extension of T iff $E = Cn^{\Delta(E)}(W)$, where $Cn^{\Delta(E)}(W)$ is the set of all formulae derivable from W using classical logic together with the rules from $\Delta(E)$.

Using this characterisation, we next express the existence of an extension of a finite default theory (W, Δ) in terms of a QBF.

Suppose $\Delta = \{\delta_i = \frac{\alpha_i\,:\,\beta_i}{\gamma_i} \mid 1 \leq i \leq n\}$. Let $D = \{d_1, \ldots, d_n\}$ and $D' = \{d'_1, \ldots, d'_n\}$ be sets of new propositional variables. Intuitively, d_i is true if δ_i is selected into $\Delta(E)$, and d'_i is true if δ_i fires in the construction of E, i.e., if $\gamma_i \in Cn^{D(E)}(T)$. Then, the following QBF expresses existence of an extension:

$$\mathcal{T}_{dl}(T) \;=\; \exists D' \exists D ((D' \leq D) \wedge \Phi_1 \wedge \Phi_2 \wedge \Phi_3 \wedge \Phi_4),$$

where $Cn(W \cup \{\gamma_i \mid d'_i \text{ is true }\})$ is the guess for the extension E and $\Phi_1, \ldots, \Phi_4$ express the following tests (G denotes the set $\{\gamma_1, \ldots, \gamma_n\}$):

- Φ_1 tests whether the justification β_i of each default δ_i in the guessed set $\Delta(E)$ is consistent with the guess for E:
$$\Phi_1 = \textstyle\bigwedge_{i=1}^{n} \Big[d_i \rightarrow \exists V \Big(\beta_i \wedge W \wedge (D' \leq G)\Big)\Big].$$
- Φ_2 tests whether no applicable default in $\Delta(E)$ is missing with respect to the guessed D'; i.e., for every δ_i such that d_i is true but d'_i is false, the set $E \cup \{\neg\alpha_i\}$ is satisfiable:
$$\Phi_2 = \textstyle\bigwedge_{i=1}^{n} \Big[(d_i \wedge \neg d'_i) \rightarrow \exists V \Big(\neg\alpha_i \wedge W \wedge (D' \leq G)\Big)\Big].$$
- Φ_3 tests whether for each default $\delta_i \notin D(E)$, its justification β_i is inconsistent with E, i.e., $\neg\beta_i$ is derivable:
$$\Phi_3 = \textstyle\bigwedge_{i=1}^{n} \Big[\neg d_i \rightarrow \forall V \Big((W \wedge (D' \leq G)) \rightarrow \neg\beta_i\Big)\Big].$$
- Φ_4 tests whether all defaults in $\Delta(E)$ assumed to be applied (d'_i = true) are actually applied (i.e., $Cn^{\Delta(E)}(W) \models \{\gamma_i \mid d'_i = \text{ true }\}$). This amounts to checking whether $\bigwedge_{i=1}^{n}(d'_i \rightarrow \gamma_i) \in Cn^{\Delta(E)}(W)$, i.e., whether $(D' \leq G) \in Cn^{\Delta(E)}(W)$. Applying a result shown in (Gottlob 1995), $\phi \notin Cn^{\Delta(E)}(W)$ iff there exists a subset $C \subseteq G = \{\gamma_1, \ldots, \gamma_n\}$ such that (i) $W \cup C \cup \{\neg\phi\}$ is satisfiable, and (ii) for each $\gamma_i \notin C$, the set $W \cup C \cup \{\neg\alpha_i\}$ is satisfiable. Using the set $C = \{c_1, \ldots, c_n\}$ of new variables, Φ_4 is as follows:
$$\forall C \Big\{ C \leq D \rightarrow \Big[\forall V \neg \Big(W \wedge \neg(D' \leq G) \wedge (C \leq G)\Big) \vee \bigvee_{i=1}^{n} \Big(d_i \wedge \neg c_i \wedge \forall V \neg \Big(W \wedge \neg\alpha_i \wedge (C \leq G)\Big)\Big)\Big]\Big\}.$$

Theorem 3 *A finite default theory $T = (W, \Delta)$ has an extension iff $\mathcal{T}_{dl}(T)$ evaluates to true.*

An alternative (and more succinct) translation of default logic into QBFs is possible using Niemelä's characterisation of extensions in terms of full sets (Niemelä 1995). To this end, for a set Δ of defaults and a set S of formulas, define $j(\Delta) = \{\beta \mid \frac{\alpha\,:\,\beta}{\gamma} \in \Delta\}$ and $\Delta_p(S) = \{\frac{\alpha}{\gamma} \mid \frac{\alpha\,:\,\beta}{\gamma} \in \Delta, \beta \in S\}$.

Rephrasing a definition in (Niemelä 1995), a subset $\Lambda \subseteq j(\Delta)$ is a *full set* for (W, Δ) iff every $\beta \in j(\Delta)$ satisfies the following condition: $\beta \in \Lambda$ iff $\neg\beta \notin Cn^{\Delta_p(\Lambda)}(W)$.

Proposition 2 (Niemelä 1995) *There is a one-to-one correspondence between the extensions and the full sets of (W, Δ). In addition, each extension E is given by $Cn^{\Delta_p(\Lambda)}(W)$, where Λ is the full set corresponding to E.*

We now express the existence of a full set in terms of a QBF. Consider $T = (W, \Delta)$ where $\Delta = \{\frac{\alpha_i\,:\,\beta_i}{\gamma_i} \mid i = 1, \ldots, n\}$.

Let $J = \{j_1, \ldots, j_n\}$ be a set of new variables. Intuitively, j_i is true if β_i can be consistently assumed, i.e., if $\neg\beta_i$ is not provable. Consider the following QBF:

$$\mathcal{T}_{dl}^{fs}(T) \;=\; \exists J \Big[\textstyle\bigwedge_{i=1}^{n} \Big(j_i \leftrightarrow \mathcal{N}(W, \Delta^J, \neg\beta_i)\Big)\Big],$$

where $\Delta^J = \{\frac{\alpha_i}{\gamma_i} \mid \frac{\alpha_i\,:\,\beta_i}{\gamma_i} \in \Delta,\ j_i \text{ is true}\}$ and $\mathcal{N}(T, \Delta^J, \neg\beta_i)$ expresses nonderivability of $\neg\beta_i$ from W using the rules from Δ^J, i.e., $\mathcal{N}(T, \Delta^J, \neg\beta_i)$ states that $\neg\beta_i \notin Cn^{\Delta^J}(W)$. Analogous to the characterisation used for the QBF Φ_4 above, $\mathcal{N}(T, \Delta^J, \neg\beta_i)$ is of the following form ($C = \{c_1, \ldots, c_n\}$ is again a set of new variables):

$$\exists C \Big\{\textstyle\bigwedge_{k=1}^{n} \Big[(j_k \wedge \neg c_k) \rightarrow \Big[\exists V \Big(W \wedge \beta_i \wedge \bigwedge_{l=1}^{n}(\neg j_l \vee \neg c_l \vee \gamma_l)\Big) \wedge \exists V \Big(W \wedge \neg\alpha_k \wedge \bigwedge_{l=1}^{n}(\neg j_l \vee \neg c_l \vee \gamma_l)\Big)\Big]\Big]\Big\}.$$

Theorem 4 *A finite default theory $T = (W, \Delta)$ has an extension iff $\mathcal{T}_{dl}^{fs}(T)$ evaluates to true.*

Brave inference, $(W, \Delta) \models_b \phi$, of a formula ϕ, i.e., membership of ϕ in some extension of (W, Δ), can be easily expressed by adding in $\mathcal{T}_{dl}^{fs}(T)$ the formula $\neg\mathcal{N}(W, \Delta^J, \phi)$. Similarly, cautious inference,

[1] For simplicity, we omit multiple justifications here. Our QBF translations can be easily extended to the more general form of defaults.

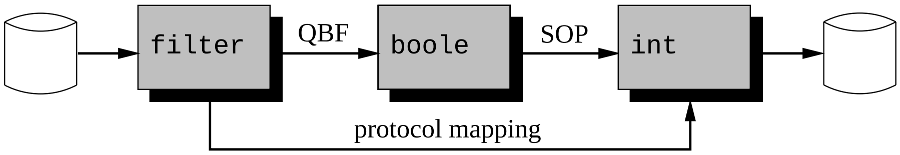

Figure 1: QUIP's system architecture.

$(T, D) \models_c \phi$, i.e., membership of ϕ in all extensions of (W, Δ), can be expressed by adding in $\mathcal{T}_{dl}^{fs}(T)$ the formula $\mathcal{N}(W, \Delta^J, \phi)$ and negating the result.

It is natural to ask how the equivalent translations $\mathcal{T}_{dl}(\cdot)$ and $\mathcal{T}_{dl}^{fs}(\cdot)$ compare with respect to evaluation time. Intuitively, $\mathcal{T}_{dl}^{fs}(\cdot)$ is less involved and should thus be evaluated faster. However, experiments indicate that $\mathcal{T}_{dl}(\cdot)$ has in general better performance results than $\mathcal{T}_{dl}^{fs}(\cdot)$ (cf. comparisons below).

Disjunctive Logic Programming. A *disjunctive logic* program, P, is a set of clauses

$$r: \quad H(r) \leftarrow P(r), N(r),$$

where $H(r)$ is a disjunction of variables, $P(r)$ is a conjunction of variables, and $N(r)$ is a conjunction of negated variables. A Herbrand interpretation I of V is a *stable model* of P (Gelfond & Lifschitz 1988; Przymusinski 1991), if it is a minimal model (with respect to set-inclusion) of the program P^I resulting from P as follows: remove each clause r such that $I \models a$ for some $\neg a$ in $N(r)$, and remove $N(r)$ from all remaining clauses.

We express the existence of a stable model by a QBF. Let P be a program, $V = \{v_1, \ldots, v_n\}$ be the atoms occurring in P, and $V' = \{v'_1, \ldots, v'_n\}$ be a set of new variables. Consider the QBF $\mathcal{T}_{lp}(P)$:

$$\exists V \Big\{ P \wedge \forall V' \Big[\neg(V' < V) \vee \bigvee_{r \in P} \neg\Big((N(r) \wedge B'(r)) \rightarrow H'(r) \Big) \Big] \Big\},$$

where $B'(r)$ and $H'(r)$ result from $B(r)$ and $H(r)$, respectively, by replacing each occurrence of v_i with v'_i $(i = 1, \ldots, n)$.

Theorem 5 *P has a stable model iff $\mathcal{T}_{lp}(P)$ evaluates to true.*

Brave inference, $P \models_b p$, of an atom p is expressed by adding p as a conjunct in the scope of $\exists V$ in $\mathcal{T}_{lp}(P)$, and cautious inference, $P \models_c p$, by adding similarly $\neg p$ there and negating the resulting formula.

Circumscription. In contrast to the formalisms described above, propositional circumscription is already a quantified boolean formula, hence it does not require a separate reduction. We recall the basic concepts of circumscirption.

In the propositional case, the parallel circumscription of a set of atoms $P = \{p_1, \ldots, p_n\}$ in a theory T, where the atoms Q are fixed and the remaining atoms $Z = \{z_1, \ldots, z_m\} = V \setminus (P \cup Q)$ may vary, is given by the following QBF $\mathit{CIRC}(T; P, Z)$, cf. (Lifschitz 1985):

$$T \wedge \forall P' \forall Z' \Big((T[P/P', Z/Z'] \wedge (P' \leq P)) \rightarrow (P \leq P') \Big).$$

Here, $P' = \{p'_1, \ldots, p'_n\}$ and $Z' = \{z'_1, \ldots, z'_m\}$ are sets of new propositional variables corresponding to P and Z, respectively, and $T[P/P', Z/Z']$ results from T by substitution of the variables in $P' \cup Z'$ for those in $P \cup Z$. Circumscriptive inference of a formula ϕ from a theory T is then expressed by the following QBF:

$$\forall V (\mathit{CIRC}(T; P, Z) \rightarrow \phi).$$

QUIP

QUIP implements the transformations described in the previous section. The architecture of QUIP is depicted in Figure 1. The problem description (e.g., a default theory, a disjunctive logic program, an abductive theory, etc.) is read and translated to a QBF by the `filter` program, which is then sent to the QBF-evaluator `boole`. The result of `boole`, usually a formula in disjunctive normal form (often called *sum of products*, SOP), is interpreted by `int`. The latter part associates a meaningful interpretation to the formulas occurring in SOP and provides an explanation in terms of the underlying problem instance (e.g., an extension, a stable model, abductive explanations, etc.). The interpretation relies on a mapping of internal variables of the generated QBF into concepts of the problem description which is provided by `filter`.

The QBF-evaluator `boole` is a commonly available propositional theorem prover based on binary decision diagrams. One of the advantages of `boole` is the fact that it enables a direct processing of the translations discussed in the previous section, without the need of an additional normal-form translation. Moreover, `boole` is in the public domain (it can be downloaded at `http://www.cs.cmu.edu/~modelcheck/bdd.html`) and easily portable to various platforms. Finally, all parts of QUIP are written in C using standard tools like LEX and YACC.

In order to incorporate new formalisms into QUIP, one has to extend the `filter` program responsible for the appropriate reductions, the mapping of the variables, and the interpreter `int`. The deductive engine remains unchanged in this process.

Experimental Results

We compare QUIP with several established tools from the literature on the basis of four benchmark problems. The tools are DeReS (Cholewinski, Marek, & Truszczynski 1996), `dlv` (Eiter *et al.* 1998), `smodels` (Niemelä & Simons 1996), and Theorist (Poole 1989). Three of the four test sets are taken from TheoryBase (Cholewinski *et al.* 1995), a well-known testbed for nonmonotonic formalisms; the other test set consists of abductive diagnosis problems for n-bit full adders. The latter problem is used to compare the abduction part of QUIP with both Theorist and the diagnosis front-end of `dlv`, whilst the problems from TheoryBase are used to compare the two default-logic encodings and the logic-programming encoding of QUIP with DeReS, `dlv` and `smodels`. More exactly, both encodings of the default-logic part of QUIP are compared with DeRes, and the logic-programming encoding is compared with `dlv` and `smodels`. All tests have been performed on a SUN ULTRA 60 with 256MB RAM; the running time is measured in seconds (with an upper time limit set to ten minutes) and comprises the sum of both user time and system time. The following program releases have been used: `dlv` release from November 24, 1999, `smodels` 2.24, and `lderes` 1.1.0.

The results for the abduction problems are shown in Table 1. The full adder is considered as a black box. The observations consists of output values (including carry bits), the input values are given by the theory. Entries with "min" give the running time for the computation of all minimal explanations; the remaining entries reflect the coresponding results for the computation of all non-minimal explanations. Furthermore, the values in the row "Theorist (1)" are determined by using our formalization of the n-bit adder, whereas the row "Theorist (2)" contains the results employing the formalization given in the User's Guide of Theorist. Due to space limitations, the details of the problem descriptions are omitted here. Note, however, that albeit the diagnosis front-end of `dlv` is based on a different semantics than both QUIP and Theorist, the considered formalisations are chosen in such a way that equivalent results are obtained.

Tables 2–4 contain the results for the problems from TheoryBase. This testbed encodes different graph problems either as a default theory or as an equivalent logic program. The measurements represent the running time for the computation of all extensions of the given default theory, or of all stable models of the corresponding logic program (labelled with "LP"). The results for the two default logic reductions of QUIP are indicated by "$\mathcal{T}_{dl}$" and "$\mathcal{T}_{dl}^{fs}$", respectively.

For each graph problem of TheoryBase, different underlying graph-classes can be chosen. The following problems have been chosen (the underlying graph-class is given in parantheses): Δ_{ind} is an encoding for maximal independent sets in a given graph (cycles), Δ_{max} is an encoding for maximal matching (chess graphs), Δ^3_{col} is an encoding for graph coloring (clique graphs).

n	1	2	3	4	5
`dlv`	0	0	1	14	132
`dlv` (min)	0	2	22	231	787
QUIP	0	0	1	12	161
QUIP (min)	0	0	1	3	18
Theorist (1)	> 1500	—	—	—	—
Theorist (2)	54	> 3600	—	—	

Table 1: n-bit full adder

# vertices	15	20	25	30	35	37	40
DeReS	1	26	480	—	—	—	—
QUIP ($\mathcal{T}_{dl}$)	0	0	1	4	19	36	90
QUIP ($\mathcal{T}_{dl}^{fs}$)	1	1	3	8	25	44	113
`dlv` (LP)	0	0	0	1	3	6	15
QUIP (LP)	0	0	1	4	18	40	85
`smodels` (LP)	0	0	0	1	2	4	10

Table 2: DL: test set based on cycle graphs and Δ_{ind}

The parameters of the tables are the corresponding parameters of TheoryBase to determine the size of the underlying graph.

The results of these tests show that QUIP compares sufficiently well. In fact, in some instances of the diagnosis example, QUIP performs actually better than each of `dlv` or Theorist. This can be explained by the fact that both the diagnosis front-end of `dlv` and Theorist are, like QUIP, *ad hoc* implementations developed to demonstrate the feasibility of the corresponding approach. On the other hand, it is clear that QUIP cannot compete with `smodels` or (the logic-programming part of) `dlv` because these tools are highly optimized systems developed with a particular semantics in mind, whereas the purpose of QUIP is to provide a *uniform* method to deal with several knowledge-representation formalisms at the same time. In any case, QUIP demonstrates the practical usefulness of our approach.

Ongoing Work and Conclusion

Our experiments document that moderately sized instances of some NMR problems can be solved reasonably well by using *ad hoc* translations to QBFs. We

# vertices	22	26	30	34	38	42
DeReS	0	2	18	147	1140	—
QUIP ($\mathcal{T}_{dl}$)	0	1	2	6	18	63
QUIP ($\mathcal{T}_{dl}^{fs}$)	1	2	5	10	25	72
`dlv` (LP)	0	0	0	1	3	12
QUIP (LP)	0	0	2	5	18	65
`smodels` (LP)	0	0	0	1	2	7

Table 3: DL: test set based on chess graphs and Δ_{match}

# vertices	2	3	4	5	6
DeRes	0	0	0	0	0
QUIP ($\mathcal{T}_{dl}$)	0	0	1	9	95
QUIP ($\mathcal{T}_{dl}^{fs}$)	0	1	19	237	—
`dlv` (LP)	0	0	0	0	0
QUIP (LP)	0	0	0	4	54
`smodels` (LP)	0	0	0	0	0

Table 4: DL: test set based on clique graphs and Δ^3_{col}

expect a similar behavior for other KR formalisms, and believe that QBF-based problem solvers like QUIP are valuable tools for researchers experimenting with KR formalisms, and in particular with KR logics. Of course, a performance increase will be achieved by designing more sophisticated translations of the problems into QBFs, or by using more advanced BDD packages than `boole`. The power of the curent framework is, however, that it realizes an easy-to-use system handling all problems in PSPACE (providing an appropriate reduction has been implemented).

Our ongoing and future work includes the following issues. We are investigating the implementations of theorem provers for modal logics in QUIP. The validity problem of standard logics like K, T, or S4 is in PSPACE, and can thus be polynomially reduced to QBFs. Notice that currently for many modal logics no theorem provers are available, or only inefficient ones.

Another issue of research concerns alternative translations of the same problem into QBFs. The experimental results of the different QBFs $\mathcal{T}_{dl}(\cdot)$ and $\mathcal{T}_{dl}^{fs}(\cdot)$, which both express existence of a default-logic extension, have shown that the chosen translation crucially affects the performance. Further research is needed to get a clearer picture of more optimized translations.

Different platforms for evaluating QBFs, based on (extensions of) the Davis-Putnam procedure (Cadoli, Giovanardi, & Schaerf 1998; Rintanen 1999b), resolution (Kleine Büning, Karpinski, & Flögel 1995) and binary decision diagrams, will be compared with respect to NMR prototype implementations. We plan to extend QUIP into a system for hybrid parallel QBF evaluation. The idea is to evaluate a particular QBF on different machines using different algorithms in parallel. This approach allows for an easy incorporation of new QBF algorithms, and exploits different strengths of the employed QBF algorithms. As well, for solving subformulas, an intertwined use of these algorithms may be considered.

Results on the above research issues will contribute for a better assessment of the suitability of the QBF approach regarding the computation of KR tasks. For expressing problem descriptions in a function-free language, it would be convenient to have an evaluator for a generalization of QBFs to function-free formulas with variables. Intelligent grounding algorithms, based on ideas in (Eiter *et al.* 1998), might be investigated.

References

Cadoli, M.; Giovanardi, A.; and Schaerf, M. 1998. An Algorithm to Evaluate Quantified Boolean Formulae. In *Proc. AAAI-98*, 262–267.

Cholewinski, P.; Marek, V. W.; Mikitiuk, A.; and Truszcynski, M. 1995. Experimenting with Nonmonotonic Reasoning. In *Proc. ICLP-95*, 267–282.

Cholewinski, P.; Marek, V. W.; and Truszcynski, M. 1996. Default Reasoning System DeReS. In *Proc. KR-96*, 518-528.

Eiter, T.; Leone, N.; Mateis, C.; Pfeifer, G.; and Scarcello, F. 1998. The KR System `dlv`: Progress Report, Comparisons and Benchmarks. In *Proc. KR-98*, 406–417.

Gelfond, M., and Lifschitz, V. 1988. The Stable Model Semantics for Logic Programming. In *Proc. 5th IC-SLP*, 1070–1080.

Gottlob, G. 1995. The Complexity of Propositional Default Reasoning Under the Stationary Fixed Point Semantics. *Information and Computation.* 121:81–92.

Kautz, H., and Selman, B. 1996. Pushing the Envelope: Planning, Propositional Logic and Stochastic Search. In *Proc. AAAI-96*, 1194–1201.

Kleine Büning, H.; Karpinski, M.; and Flögel, A. 1995. Resolution for Quantified Boolean Formulas. *Information and Computation* 117(1):12–18.

Lifschitz, V. 1985. Computing Circumscription. In *Proc. IJCAI-85*, 121–127.

Marek, W., and Truszczyński, M. 1993. *Nonmonotonic Logics.* Springer.

Moore, R. 1985. Semantical Considerations on Nonmonotonic Logics. *Artificial Intelligence* 25:75–94.

Niemelä, I., and Simons, P. 1996. Efficient Implementation of the Well-Founded and Stable Model Semantics. In *Proc. JICSLP-96*, 289–303.

Niemelä, I. 1992. On the Decidability and Complexity of Autoepistemic Reasoning. *Fundamenta Informaticae* 17:117–155.

Niemelä, I. 1995. Towards Efficient Default Reasoning. In *Proc. IJCAI-95*, 312–318.

Poole, D. 1989. Explanation and Prediction: An Architecture for Default and Abductive Reasoning. *Computational Intelligence* 5(1):97–110.

Przymusinski, T. 1991. Stable Semantics for Disjunctive Programs. *New Generation Computing* 9:401–424.

Reiter, R. 1980. A Logic for Default Reasoning. *Artificial Intelligence* 13:81–132.

Rintanen, J. 1999a. Constructing Conditional Plans by a Theorem-Prover. *JAIR* 10:323–352.

Rintanen, J. 1999b. Improvements to the Evaluation of Quantified Boolean Formulae. In *Proc. IJCAI-99*, 1192–1197.

Selman, B., and Levesque, H. J. 1990. Abductive and Default Reasoning: A Computational Core. In *Proc. AAAI-90*, 343–348.