

The Complexity of Futile Questioning

Anja Remshagen

University of West Georgia
Carrollton, Georgia, USA

Klaus Truemper

University of Texas at Dallas
Richardson, Texas, USA

Abstract *Instances of the futile questioning problem abound in applications, especially, in the area of artificial intelligence. The problem can be represented by a particular quantified Boolean formula. A variation of the problem includes also costs for the logic variables. Typically, the problem is simplified instead of solved directly, which may lead to inaccurate results. Alternatively, the problem can be transformed to a standard format. We investigate the complexity of the futile questioning problem and two subclasses. Based on the complexity result, we argue that a transformation to the standard format is not suitable for practical purposes.*

Keywords: Futile questioning, quantified Boolean formula, complexity, polynomial hierarchy, logic computation.

1. Introduction

Many intelligent systems interactively acquire information from the user and make decisions based on the acquired data. Such systems pose questions that are answered by the user. For example, a diagnostic system may query the user to establish a diagnostic goal that the system has selected based on a preliminary analysis. If the goal happens to be unprovable regardless of the answers supplied by the user, then that goal is futile and should be abandoned. If an algorithm can decide whether that situation is at hand, then fruitless lines of questioning can be cut short. In some systems, a cost may be explicitly or implicitly associated with the acquisition of answers.

In artificial intelligence, the futile questioning problem models a variety of applications, like regulatory compliance and planning. An example for regulatory compliance is the system for management of hazardous materials described by Straach and Truemper [17]. The answers to questions determine the setting or situation to be managed. We want to decide if a certain action can be done without violating the regulations. In this

case, futility means that no violation can be proved. In the case of planning problems, a question corresponds to a goal. One wants to determine whether there exists a plan or strategy that—no matter which unknown conditions exist—leads to the goal.

Such problems can be modeled as follows. A knowledge base is given as two theories R and S of propositional logic. Theory R establishes valid relationships among the questions, which are represented by a variable set Q . Theory S models the knowledge domain and includes the negation of the goal or decision to be established. Thus, unsatisfiability of S establishes that goal or decision. The theories R and S share the variable set Q . We want to find answers to the questions, given as a truth assignment for the variables of Q , which are consistent with R but not with S . We refer to this problem as Q-ALL SAT. We also encounter the problem where a cost is associated with the truth assignment to each variable in R , and where answers inducing minimum total cost are searched for. We call this problem Q-MIN UNSAT.

Although instances of the futile questioning problem abound, the problem is typically not tackled directly. Instead, the problem is reduced to a simpler one, and then the simplified problem is solved. Why does the futile questioning problem seem to defy a direct solution? In this paper, we investigate the complexity of the futile questioning problem. Both Q-ALL SAT and Q-MIN UNSAT are at the second level of the polynomial hierarchy, and thus, very hard. Note that Q-ALL SAT is a quantified Boolean formula that can be transformed into a standard format, called *QBF*. QBF is also located at the second and even higher levels of the polynomial hierarchy. Recent interest in QBF has resulted in several solution approaches. However for practical applications, a representation of Q-ALL SAT as a QBF formula is not suitable for the following reason: We have analyzed the complexity of subclasses and show that the complexity of those subclasses is re-

duced by at least one level. A reduction to QBF will disguise the structure of the easier subclasses.

The next section outlines related work. Section 3 defines Q-ALL SAT and Q-MIN UNSAT. Section 4 covers the complexity of the general case. Section 5 determines two subclasses with reduced complexity. Section 6 points out the implication of the presented results.

2. Related Work

The futile question problem has been tackled by different approaches, ranging from rule-based systems to Bayesian networks. See Russell and Norvig [15] for an introduction to those techniques. However, these methods tend to introduce unwanted simplifications.

Mneimneh and Sakallah [11] have developed a direct solution algorithm for the special problem of computing the vertex eccentricity arising in formal hardware verification. The solver takes advantage of the specific structure of the instances. A general solver for QBF has been proposed by Remshagen and Truemper [13]. The solver is based on backtracking search.

Although hardly any work on solution algorithms for the futile questioning problem has been published, there has been great interest in quantified Boolean formulas where all quantifiers precede a formula in conjunctive normal form. This problem, called QBF, is also located at higher levels of the polynomial hierarchy. A variety of approaches to solve QBF have been developed, like search-based methods (Giunchiglia, Narizzano, and Tacchella [6]), methods using binary decision diagrams (Pan and Vardi [12]), and hybrid approaches (Benedetti [2]), to name only a few. QBF does not allow for inclusion of cost.

Rintanen [14] determines the complexity of planning problems. Even though the natural presentation of the considered problems corresponds to the futile questioning problem, Rintanen uses a presentation as QBF.

The complexity of some subclasses of QBF has been determined. Aspvall, Plass, and Tarjan [1] prove that QBF is in \mathcal{P} if the all clauses of the corresponding logic formula have at most two literals. In case of Horn formulas, the complexity of QBF is also reduced to \mathcal{P} , as Kleine Büning, Karpinski, and Flögel [8] show.

The futile questioning problem is related to logic-based abduction, discussed by Eiter and Gottlob [5]. They list various applications and explore the complexity of logic-based abduction.

3. Q-ALL SAT and Q-MIN UNSAT

Define a literal to be a variable or a negated variable. A literal is *positive* if it is a variable and *negative* if it is a negated variable. The variable of a literal l is denoted by $|l|$, that is, $l = |l|$ or $\neg l = |l|$.

A *propositional formula* with the connectives $\neg, \wedge, \vee, \rightarrow$ is defined in the usual way. A formula has *conjunctive normal form* (CNF) if it is a conjunction of *clauses* each of which is a disjunction of literals. We represent a CNF formula as a set of its clauses and a clause as a set of its literals. A *2CNF formula* is a CNF formula where each clause contains at most two literals. A CNF formula is *Horn* if each clause contains at most one positive literal. Let V be a variable set. A *truth assignment* α for V , or short *assignment* for V , is a function on the set of literals $\{v, \neg v \mid v \in V\}$ that assigns a truth value *True/False* to each literal so that $\alpha(v) = \neg\alpha(\neg v)$. A *partial truth assignment* for V assigns truth values to a subset of literal $\{v, \neg v \mid v \in V'\}$, with $V' \subseteq V$.

Given the variable set $V = \{v_1, v_2, \dots, v_n\}$ and an order (v_1, v_2, \dots, v_n) on the variables, an assignment α for V is *lexicographically larger* than an assignment β for V if there is a $k, 1 \leq k < n$, such that for all $i = 1, 2, \dots, k$, $\alpha(v_k) = \beta(v_k)$ holds and $\alpha(v_{k+1}) = \text{True}$ and $\beta(v_{k+1}) = \text{False}$.

A formula is *consistent* or *satisfiable* if there exists an assignment so that the formula evaluates to *True*. Such an assignment is called a *satisfying solution* of T . If no such assignment exists, the formula is called *inconsistent* or *unsatisfiable*. The problem to decide if a CNF formula is unsatisfiable is called *SAT*. If a partial truth assignment of a CNF formula T can be extended to a satisfying solution, we call the partial assignment *T-consistent*. Otherwise, the assignment is *T-inconsistent*. For $X = \{x_1, x_2, \dots, x_n\}$, we write $\forall X$ (respectively $\exists X$) instead of $\forall x_1, x_2, \dots, x_n$ (respectively $\exists x_1, x_2, \dots, x_n$).

An instance of MINSAT is given by a CNF formula T and, for each variable x of T , by a rational nonnegative *True-cost* c_x . The MINSAT problem demands that one either determines T to be unsatisfiable or one produces a satisfying solution α whose total cost—that is $\sum_{x, \alpha(x)=\text{True}} c_x$ —is minimum.

We study two problems called *Q-ALL SAT* and *Q-MIN UNSAT* defined by Truemper [18]. An instance of each problem consists of a quintuple (Q, X, Y, R, S) where Q, X , and Y are variable sets and R and S

are CNF formulas. The formulas R and S have common variable set Q . The remaining variable set of R is X , and the remaining variable set of S is Y . Thus, Q and X and Y are disjoint. In addition, an instance of Q-MIN UNSAT assigns a *True*-cost c_x to each variable x of R , that is $x \in Q \cup X$. Q-ALL SAT is the problem to decide if the quantified formula

$$\forall Q (\exists X R \rightarrow \exists Y S) \quad (1)$$

is *True*. If the formula evaluates to *False*, we call an R -consistent and S -inconsistent truth assignment for Q a counter example of the instance.

In the case of Q-MIN UNSAT, one also has to decide if Formula (1) evaluates to *True*. If the formula is *False*, then Q-MIN UNSAT demands that one finds an R -consistent and S -inconsistent truth assignment for Q with the following property. Define the *induced R -cost* of an R -consistent assignment for Q to be the minimum total cost of a satisfying solution of R that extends the R -consistent assignment for Q . Then the wanted assignment in Q-MIN UNSAT must have minimum induced R -cost. We call such an assignment a solution.

4. Complexity Results

We show that Q-ALL SAT and Q-MIN UNSAT are at the second level of the polynomial hierarchy and thus are considered to be very difficult. To prove the complexity, some of the reductions result in propositional formulas that are not in conjunctive normal form. For example, consider the formula

$$\forall Q (\exists X' R' \rightarrow \exists Y' S') \quad (2)$$

where R' and S' are logic propositional formulas, but not CNF formulas. Thus, determining the truth value of the quantified proposition is not an instance of Q-ALL SAT. We can remedy the problem as follows. A well-known transformation of R' into a CNF formula R with additional variable set X'' is as follows. For each satisfying solution α' of R' , there exists a unique assignment α'' for X'' so that (α', α'') is a satisfying solution of R . For each solution α of R , the restriction of α to the variables of R' is a satisfying assignment of R' . We call R' and R *equivalent*. The transformation from R' to R is linear in the size of R' . Analogously, we obtain an equivalent CNF formula S from S' with variable set $Y = Y' \cup Y''$. If we set $X = X' \cup X''$, we obtain the Q-ALL SAT instance

(Q, X, Y, R, S) . Then Formula 2 evaluates to *True* if and only if (Q, X, Y, R, S) evaluates to *True*.

Remark 1. Let R' and S' be logic propositional formulas with common variable set Q . Then there exists an instance (Q, X, Y, R, S) of Q-ALL SAT with the following properties. An assignment to Q is R' -consistent and S' -inconsistent if and only if it is a counter example of (Q, X, Y, R, S) .

Stockmeyer [16] and Warthall [20] show results for the polynomial hierarchy that prove the following problem to be Π_2^p -complete. An instance (V, W, T) consists of a propositional logic formula T with disjoint sets of variables V and W . The problem demands to decide whether

$$\forall V \exists W T \quad (3)$$

is *True*. We reduce an instance (V, W, T) to an instance of Q-ALL SAT as follows. Introduce a new variable x , and set $X = \{x\}$. Define R to be the empty CNF formula, that is, R has no clauses and thus, is a tautology. Then the quantified formula

$$\exists V (\exists X R \wedge \forall W \neg(\neg T)) \quad (4)$$

evaluates to *True* if and only if the quantified formula

$$\forall V \exists W T \quad (5)$$

evaluates to *True*. Due to Remark 1, we can reduce (V, W, T) in linear time to Q-ALL SAT. Hence, the problem Q-ALL SAT is Π_2^p -hard.

Theorem 1. Q-ALL SAT is Π_2^p -complete.

Proof. It remains to show that Q-ALL SAT is in Π_2^p . Let (Q, X, Y, R, S) be an instance of Q-ALL SAT. We guess an assignment α for Q and use a SAT oracle to verify that α is R -consistent and S -inconsistent. Thus the complement of Q-ALL SAT is Σ_2^p -complete and hence Q-ALL SAT is Π_2^p -complete.

Often, we are interested in a counter example of Q-ALL SAT. By solving Q-ALL SAT, we can determine if a literal l , $|l| \in Q$, is set to *True* by a counter example of a Q-ALL SAT instance: We assign *True* to l and enforce the assignment in R and S . If the resulting instance of Q-ALL SAT evaluates to *False*,

then there exists a counter example of the original instance that sets l to *True*. Hence, deciding if there exists a counter example that sets a literal to *True* is not harder than solving Q-ALL SAT. Analogously, we can determine if all counter examples set a given literal l to *True* by assigning *False* to l . If the resulting instance of Q-ALL SAT evaluates to *True*, all counter examples of the original instance set l to *True*.

We discuss the complexity of Q-MIN UNSAT.

Theorem 2. Q-MIN UNSAT is Δ_3^P -hard.

Proof. Krentel [10] and Wagner [19] present results that directly prove the following problem to be Δ_3^P -complete. An instance of the problem consists of a propositional formula T with disjoint variable sets $V = \{v_1, v_2, \dots, v_n\}$ and $W = \{w_1, w_2, \dots, w_k\}$. Let α be an assignment for V , and let T_α be the formula that results from enforcing α in T . We call an assignment α for V *T-admissible* if $\forall W T_\alpha$ evaluates to *True*. A valid instance of the problem has at least one admissible assignment for V . The problem demands to decide whether the with respect to (v_1, v_2, \dots, v_n) lexicographically smallest admissible assignment α fulfills $\alpha(v_n) = \text{False}$. We reduce an instance of the problem to Q-MIN UNSAT.

Let x and s be new variables. We set $X = \{x\}$, $Y = W \cup \{s\}$, and $S' = (T \rightarrow s) \wedge \neg s$. Let R be the empty CNF formula. Then an assignment to V is R -consistent and S' -inconsistent if and only if it is T -admissible. Observe that S' is not in conjunctive normal form. Due to Remark 1, we can transform S' into an equivalent CNF formula S with additional variable set V' . If we set $Q = V \cup V'$, then we obtain a T -admissible assignment to V from every R -consistent and S -inconsistent assignment to Q . Every T -admissible assignment to V can be extended to an R -consistent and S -inconsistent assignment to Q .

In order to obtain the lexicographically smallest admissible assignment, we assign *True*-costs to the variables of R . For $i = 1, 2, \dots, n$, we set the *True*-cost of v_i to 2^{n-i} . All other variables have *True*-cost 0. Then the unique solution of the resulting instance (Q, X, Y, R, S) of Q-MIN UNSAT corresponds to the lexicographically smallest admissible assignment for V . The value $\alpha(v_n)$ of the minimum cost solution solves the Δ_3^P -complete problem.

5. Easier Subproblems

We have shown that Q-ALL SAT are Q-MIN UNSAT are a very hard problems in general. However, there are several subproblems whose complexity is reduced by one or even two levels in the polynomial hierarchy. We investigate two subproblems of Q-ALL SAT and Q-MIN UNSAT.

5.1 Horn Formulas

It is well known that the complexity of SAT is reduced by one level of the polynomial hierarchy in the case of Horn formulas. For example, Dowling and Gallier [4] describe a polynomial-time solution algorithm. The complexity of Q-ALL SAT is also reduced by one level if all CNF formulas are Horn formulas. We call the subproblem of Q-ALL SAT (respectively Q-MIN UNSAT) where the CNF formulas R and S are Horn formulas *Horn Q-ALL SAT* (respectively *Horn Q-MIN UNSAT*).

Theorem 3. Horn Q-ALL SAT is $\text{co}\mathcal{NP}$ -complete.

Proof. We show that the complement of Horn Q-ALL SAT is \mathcal{NP} -complete. In order to show containment in \mathcal{NP} , we guess an assignment for Q . Since R and S are Horn formulas, we can verify in polynomial time whether the assignment is R -consistent and S -inconsistent.

In order to prove completeness, we reduce an instance of the \mathcal{NP} -complete satisfiability problem to an instance of the complement of Horn Q-ALL SAT. Let T be an instance of the satisfiability problem with clauses T_1, T_2, \dots, T_m and variable set $V = \{v_1, v_2, \dots, v_n\}$. We introduce the new variable sets $V' = \{v'_1, v'_2, \dots, v'_n\}$, $G = \{g_1, g_2, \dots, g_n\}$, and $Z = \{z_1, z_2, \dots, z_m\}$. Set $Q = V \cup V'$, $X = G \cup Z$, $Y = G \cup Z$, and

$$R = \{\neg v_i \vee \neg v'_i \mid i = 1, \dots, n\} \quad (6)$$

$$S = \{v_i \rightarrow g_i, v'_i \rightarrow g_i \mid i = 1, \dots, n\} \cup \quad (7)$$

$$\{v_i \rightarrow z_k \mid v_i \in T_k\} \cup \{v'_i \rightarrow z_k \mid \neg v_i \in T_k\} \quad (8)$$

$$\{\neg g_1 \vee \dots \vee \neg g_n \vee \neg z_1 \vee \dots \vee \neg z_m\} \quad (9)$$

Each solution α of T can be extended to an R -consistent and S -inconsistent assignment of Q by setting v'_i to *True* if and only if $\alpha(v_i) = \text{False}$. Each R -consistent and S -inconsistent assignment α of Q becomes a solution of T if we restrict the assignment α to the variable set V .

In case of nonnegative costs, MINSAT for Horn formulas is in \mathcal{P} like the corresponding SAT problem. But Horn Q-MIN UNSAT is still Δ_2^P -hard.

Theorem 4. Horn Q-MIN UNSAT is Δ_2^P -hard.

Proof. We use the following problem that has been shown to be Δ_2^P -complete by Kadin [7] and Krentel [9]. Let T be a satisfiable CNF formula with the clauses T_1, T_2, \dots, T_m and variable set $V = \{v_1, v_2, \dots, v_n\}$. The problem demands to decide whether the with respect to (v_1, v_2, \dots, v_n) lexicographically smallest satisfying assignment α of T fulfills $\alpha(v_n) = \text{False}$.

We use the same construction as in the proof of Theorem 3 to obtain an instance (Q, X, Y, R, S) of Horn Q-ALL SAT. In order to determine the lexicographically smallest solution of T , we assign a *True*-cost of 2^{n-i} to each variable v_i . All other variables of R receive *True*-cost 0. We obtain the lexicographically smallest solution of the original Δ_2^P -complete problem from the solution of the resulting Q-MIN UNSAT instance.

5.2 2CNF Formulas

Cook [3] has constructed a polynomial-time algorithm to solve SAT for 2CNF formulas. Aspvall, Plass, and Tarjan [1] show that even a quantified Boolean formula where all quantifiers are a prefix of a 2CNF formula can be evaluated in polynomial time. One can use an analogous construction to show that Q-ALL SAT is in \mathcal{P} in case of 2CNF formulas. We call the subclass of Q-ALL SAT (respectively Q-MIN UNSAT) where the formulas R and S of an instance (Q, X, Y, R, S) are 2CNF formulas *2CNF Q-ALL SAT* (respectively *2CNF Q-MIN UNSAT*).

Definition 1. Let T be a 2CNF formula. Then the graph $G_T = (V_T, E_T)$ is the directed graph with vertices $V_T = \{u \mid u \text{ is a literal in } T\}$ and with edges $E_T = \{(u, v) \mid \neg u \vee v \text{ is a clause of } T\} \cup \{(\neg u, u) \mid u \text{ is a clause of } T\}$.

In the following we use the term literal of T and vertex of G_T interchangeably. Aspvall, Plass, and Tarjan [1] prove the following necessary and sufficient criteria of satisfiability of 2CNF formulas.

Remark 2. A 2CNF formula T is unsatisfiable if and only if a literal and its negation are in the same strongly-connected component.

Using the definition of the graph G_T , it is easy to verify the following property for a satisfying solution of T .

Remark 3. Let T be a 2CNF formula. An assignment α of the variables of T is a solution of T if and only if for any two literals u and v with $\alpha(u) = \text{True}$ and $\alpha(v) = \text{False}$, there is no path from u to v in T .

Lemma 1 Let T be a satisfiable CNF formula, and let α be a partial assignment for a subset V of the variables of T . The assignment α is T -inconsistent if and only if there exists a path from a literal u to a literal v in G_T with $|u|, |v| \in V$, $\alpha(u) = \text{True}$ and $\alpha(v) = \text{False}$.

Proof. Let T be a 2CNF formula. Due to Remark 3, a partial assignment α with $\alpha(u) = \text{True}$ and $\alpha(v) = \text{False}$ for two literal u, v that are connected by a path from u to v in G_T cannot be extended to a satisfying solution of T .

Assume that α is an assignment for the subset V of the variable set of T where $\alpha(u) = \text{False}$ or $\alpha(v) = \text{True}$ if there is a path from u to v in G_T . We show that α can be extended to a satisfying assignment α' of T as follows. For all literals u that can be reached by a path from a literal v with $\alpha(v) = \text{True}$, we set $\alpha'(u)$ to *True* and $\alpha'(\neg u)$ to *False*. Assume that α' is not a well-defined assignment. Then α' assigns *True* to a literal u and to its negation. Then there are two not necessarily distinct literals v, v' that are set to *True* by α , and there is a path in G_T from v to u and a path from v' to $\neg u$. By definition of G_T , the path from v to u has a counterpart, that is a path from $\neg u$ to $\neg v$. But then there exists a path from v' to $\neg v$, and α sets v' to *True* and $\neg v$ to *False*. This is a contradiction to the properties of α . Therefore, the extension α' of α is a well-defined assignment.

Observe that for any remaining literal u that has not yet been set to a truth value by α' , the following holds. There is no path from u to a literal set to *False* and no path from a literal that is set to *True* to u . We remove all nodes in G_T that are set to a truth value by α' . Since T is satisfiable, no two vertices u and v are in the same

strongly-connected component. The same holds for the reduced graph. Thus, the reduced graph represents a satisfiable 2CNF formula, and α' can be extended to a satisfying solution of T .

Theorem 5. 2CNF Q-ALL SAT is in \mathcal{P} .

Proof. We describe a polynomial-time solution algorithm for instances (Q, X, Y, R, S) . We first check for satisfiability of R . If R is unsatisfiable, (Q, X, Y, R, S) evaluates to *True*. Otherwise, we check for satisfiability of S . If S is unsatisfiable, (Q, X, Y, R, S) evaluates to *False*. Otherwise, if both R and S are satisfiable, we consider each pair of literals u, v with $|u|, |v| \in Q$. If there is a path from u to v in G_S , but not in G_R , (Q, X, Y, R, S) evaluates to *False*. If no such pair exists, (Q, X, Y, R, S) evaluates to *True*.

Obviously, if R or S is unsatisfiable, the algorithm terminates with the correct result. Assume there is pair of literals u, v with $|u|, |v| \in Q$, and there is a path from u to v in G_S , but not in G_R . Then we can determine an R -consistent and S -inconsistent assignment α as follows. We set u to *True* and v to *False*. By Lemma 1, the assignment can be extended to a solution of R . Let α be the assignment that results by restricting the solution to the variable set Q . By Lemma 1, we also know that α is S -inconsistent. Thus, the algorithm terminates with the correct result in this case as well. We still need to show that the algorithm determines that (Q, X, Y, R, S) evaluates to *False*, if there exists a counter example and if R and S are satisfiable. Thus, assume there exists a counter example α of the Q-ALL SAT instance. Since α is S -inconsistent and due to Lemma 1, there are two literals u, v , with $|u|, |v| \in Q$, such that G_S contains a path from u to v and $\alpha(u) = \text{True}$ and $\alpha(v) = \text{False}$. As α is R -consistent, no such path exists in G_R . Hence, the algorithm determines the correct result.

Although SAT and Q-ALL SAT for 2CNF formulas are in \mathcal{P} , MINSAT restricted to 2CNF formulas is known to be \mathcal{NP} -hard. It is easy to see that MINSAT for 2CNF can be reduced to an instance of 2CNF Q-ALL SAT, Thus, 2CNF Q-MIN UNSAT is at least as hard as MINSAT.

Theorem 6. 2CNF Q-MIN UNSAT is \mathcal{NP} -hard.

6. Conclusion

The futile questioning problem, including Q-ALL SAT and Q-MIN UNSAT, is a natural formulation for a variety of applications in artificial intelligence. We have shown that the problem is at the second level at the polynomial hierarchy. A common solution approach has been to reduce the problem to a simpler one. But a simplification can lead to inaccurate results which are undesirable or even unacceptable, like in medical diagnosis.

In spite of the difficulty of problems at the second or even higher levels, such a problem, called QBF, is currently studied and first solution algorithms are being developed. Special cases of QBF, like 2CNF and Horn formulas, are both reduced to \mathcal{P} . Hence, these subclasses do not seem to be of great practical interest. The question arises whether QBF is a good representation for the futile questioning problem.

We have demonstrated that 2CNF Q-ALL SAT is in \mathcal{P} . However, in the case of Horn formulas the futile questioning problem is reduced by one level at the polynomial hierarchy. That is, it is still at the first level. The structure of the Horn formulas would be lost by a transformation to QBF. We suggest that Q-ALL SAT and Q-MIN UNSAT is a more suitable problem formulation for the development of effective solution algorithm.

References

- [1] Aspvall, B., Plass, M. F., Tarjan, E.: A linear-time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters* 8 (3), (1979) 121–123
- [2] Benedetti, M.: Evaluating QBFs via Symbolic Skolemization. *Proceedings of the Eleventh International Conference on Logic for Programming Artificial Intelligence and Reasoning* (2006).
- [3] Cook, S. A.: The Complexity of Theorem Proving Procedures. *Proceedings Third Annual ACM Symposium on Theory of Computing*, (1971) 151–158.
- [4] Dowling, W. F., Gallier, J. H.: Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *Journal of Logic Programming* 1, (1984) 267–284.
- [5] Eiter, T., Gottlob, G.: The Complexity of Logic-based Abduction. *Journal of the Association for Computing Machinery* 42, (1995) 3–42.
- [6] Giunchiglia, E., Narrizano, M., Tacchella, A.: Backjumping for Quantified Boolean Logic satisfiability. *Artificial Intelligence* 145(1), (2003) 99–120.

- [7] Kadin, J.: $P^{\text{NP}[O(\log n)]}$ and sparse turing-complete sets for NP. *Journal of Computer and Systems Science* 39(3), (1989) 282–298.
- [8] Kleine Büning, H., Karpinski, M., Flögel, A.: Resolution for quantified Boolean formulas. *Information and Computation* 117(1), (1995) 12–18.
- [9] Krentel, M.: The complexity of optimization problems. *Journal of Computer and Systems Science* 36, (1988) 490–509.
- [10] Krentel, M.: Generalizations of OptP to the polynomial hierarchy. *Theoretical Computer Science* 97, (1992) 183–198.
- [11] Mneimneh, M., Sakallah, K.: Computing Vertex Eccentricity in Exponentially Large Graphs: QBF Formulation and Solution. *Proceedings of the Sixth International Conference on Theory and Applications of Satisfiability Testing* (2003).
- [12] Pan, G., Vardi, Y.: Symbolic Decision Procedures for QBF. *Proceedings of the Tenth International Conference on Principles and Practice of Constraint Programming*, (2004).
- [13] Remshagen, A., Truemper, K.: An Effective Algorithm for the Futile Questioning Problem. *Journal of Automated Reasoning* 34(1), (2005) 31–47.
- [14] Rintanen, J.: Constructing Conditional Plans by a Theorem-Prover. *Journal of Artificial Intelligence* 10, (1999) 323–352.
- [15] Russell, S., Norvig P.: *Artificial Intelligence: A Modern Approach*. Prentice Hall (2003).
- [16] Stockmeyer, L. J.: The polynomial hierarchy. *Theoretical Computer Science* 3, (1976) 1–22
- [17] Straach, J., Truemper, K.: Learning to Ask Relevant Questions. *Artificial Intelligence* 111, (1999) 301–327.
- [18] Truemper, K.: *Design of Logic-based Intelligent Systems*. Wiley (2004).
- [19] Wagner, K.: Bounded Query Classes. *SIAM Journal on Computing* 19(5), (1990) 833–846.
- [20] Warthall, C.: Complete sets and the polynomial hierarchy. *Theoretical Computer Science* 3, (1976) 23–33.