On Quantifier Shifting for Quantified Boolean Formulas*

Uwe Egly Hans Tompits Stefan Woltran

Institut für Informationssysteme,
Abteilung Wissensbasierte Systeme 184/3,
Technische Universität Wien,
Favoritenstrasse 9–11, A-1040 Vienna, Austria
[uwe,tompits,stefan]@kr.tuwien.ac.at

Abstract

Since most currently available solvers for quantified Boolean formulas (QBFs) process only input formulas in prenex normal form, suitable translations are required for handling arbitrary formulas. In this paper, we propose a normal form translation incorporating a certain anti-prenexing step in order to obtain QBFs possessing quantifier prefixes such that the number of alternating quantifiers is never greater than the number of alternations obtained by using nondeterministic normal form translations based on usual quantifier shifting rules. Furthermore, our algorithm is deterministic. We show that anti-prenexing is beneficial in some cases for QBF-solvers which are able to process arbitrary QBFs, like BDD-based solvers. We illustrate this point by discussing some experimental results in this direction.

1 Introduction

Solving hard problems like planning or various forms of nonmonotonic reasoning by encoding them into quantified Boolean formulas (QBFs) and computing the truth value of the resultant formulas with a QBF-solver has become an attractive and increasingly important research topic over the last years (cf., e.g., [12, 5, 4, 11]). The QBFs resulting from the encodings are usually *not* in a specific normal form which prevents the application of most of the available QBF-provers [9, 3, 6, 8, 10, 12] without a translation into normal form. The only kind of QBF-solvers which can handle arbitrary formulas is based on binary decision diagrams (BDDs).

In order to make more practicably successful QBF-solvers available for solving the encoded problems, a transformation of an arbitrary QBF into a specific normal form (e.g., prenex CNF) is required. Usually, such a transformation consists of two steps,

*The work was partially supported by the Austrian Science Foundation under grant P15068.

namely (i) the generation of a prenex form with an arbitrary quantifier-free matrix, and (ii) the translation of the matrix into normal form (e.g., CNF). Step (i) is usually based on quantifier-shifting rules derived from well-known equivalences for quantifiers (cf. Proposition 1 below). For Step (ii), different well-known approaches have been proposed.

In this paper, we concentrate on strategies related to Step (i). Usually, (non-deterministic) ad-hoc translations are used which result, in general, in formulas with different quantifier prefixes. Although these resulting formulas are equivalent to each other, the running time of a QBF-solver usually depends on the order of quantifiers. Moreover, it is desirable that the normal form(s) of a formula reflect the "inherent" worst-case complexity of the source formula. This is illustrated in the following example. Consider the QBF

$$\Phi := \exists p \Big[\Big(\forall q \, \exists r \, (p \vee q \vee r) \Big) \wedge \Big(\exists r \, \forall q \, (\neg p \vee q \vee r) \Big) \Big]. \tag{1}$$

Two equivalent QBFs in prenex form resulting from shifting quantifiers to the left are immediately apparent:

$$\begin{aligned} \Phi' &:= \exists p \, \forall q \, \exists r \, \exists r' \, \forall q' \, \Big((p \vee q \vee r) \wedge (\neg p \vee q' \vee r') \Big); \\ \Phi'' &:= \exists p \, \exists r' \, \forall q' \, \forall q \, \exists r \, \Big((p \vee q \vee r) \wedge (\neg p \vee q' \vee r') \Big). \end{aligned}$$

Two observations are central. First, we need some suitable renaming schema for bound variables in order to avoid name conflicts. Second, when shifting quantifiers to the left, formulas may arise with different structures of the quantifier prefix. Here, we claim to prefer Φ'' over Φ', since Φ'' yields a smaller number of quantifier *alternations*. But is Φ'' optimal with respect to the minimal number of such alternations? In fact, it is not, since

$$\exists p \, \exists r \, \exists r' \, \forall q \, \forall q' \, \Big((p \vee q \vee r) \wedge (\neg p \vee q' \vee r') \Big) \tag{2}$$

with one quantifier alternation is equivalent to Φ as well, and, as we will demonstrate, QBF (2) results from Φ by a more sophisticated application of the quantifier shifting rules. Indeed, the crucial point is to shift quantifiers "down" in the formula tree before all quantifiers are shifted "upwards" in order to generate the prenex form. Since down-shifting is opposed to up-shifting (or prenexing), the former is called *anti-prenexing*.

Minimising the number of quantifier alternations can be motivated as follows by taking computational complexity into account. First, recall that QBFs in prenex form are identified as prototypical problems for the classes in the polynomial hierarchy [13]. In particular, the structure of the prenex gives an estimation of the inherent worst-case complexity of a given QBF. Translating arbitrary QBFs into prenex form where the number of alternating quantifiers is minimised thus gives a good characterisation of the original QBF. Related to this issue, we mention that both Φ' and Φ'' fulfill the conditions to be QBFs from the so-called Model A [7]. The methodology of Model A is frequently used to generate (hard) random QBF instances. However, the example above illustrates some weakness of Model A because the quantifier-prefixes in Φ' and Φ'' do not sufficiently characterise the inherent complexity of the generated QBF as reflected by QBF (2).[1] So we shall also apply QBFs in prenex form to our algorithm to derive a possibly more adequate prefix.

A second question arises in the following case. Let

$$\Psi := \Big(\exists p \, \forall q \, \exists r \, ((p \lor q) \to (r \land q))\Big) \land \Big(\exists u \, \exists v \, (u \land v)\Big).$$

The following QBFs are in prenex form satisfying the minimality criterion from above:

$$\Psi' \quad := \quad \exists p \, \forall q \, \exists r \, \exists u \, \exists v \, \psi; \tag{3}$$

$$\Psi'' \quad := \quad \exists u \, \exists v \, \exists p \, \forall q \, \exists r \, \psi, \tag{4}$$

with $\psi = ((p \lor q) \to (r \land q)) \land (u \land v)$. Which one should we prefer? At the moment, we claim that both strategies should be taken into account, and future work on experimental evaluation shall decide the better strategy. We call the strategy from which Ψ' is obtained "shift-to-bottom" and the one yielding Ψ'' "shift-to-top". From an intuitive point of view "shift-to-top" is preferable to "shift-to-bottom" for QBF-solvers based on the procedure of Davis, (Putnam,) Logemann, and Loveland, since the number of dependencies of existential variables on universal variables is reduced. However, it is not so clear whether this observation is true in general (consider, e.g., BDD-based solvers).

Finally, there is a certain class of QBFs which is prototypical for a family of complexity classes, namely D_k^P. Take the QBF

$$\Omega := \exists p \, \forall q \, \Big((\neg p \land q) \lor (p \land \neg q)\Big) \land \forall r \, \exists s \Big((\neg r \lor s) \land (r \lor \neg s)\Big).$$

Constructing a purely prenex QBF of Ω leads to two different minimal quantifier prefixes, viz. $\exists\forall\exists$ as well as $\forall\exists\forall$. This effect hints that the QBF is in fact related to a complexity class D_k^P. From a complexity-theoretical point of view, both prenex forms are not well suited since they characterise a higher complexity class than Ω itself. It seems more appropriate to independently evaluate the first and the second conjunct, respectively.

In this paper, we propose a normal form translation incorporating an anti-prenexing step in order to obtain QBFs possessing quantifier prefixes such that the number of alternating quantifiers is never greater than the number of alternations obtained by using nondeterministic normal form translations based on usual quantifier shifting rules. Furthermore, our algorithm is deterministic. We will show that anti-prenexing is beneficial in some cases for QBF-solvers which are able to process arbitrary QBFs, like BDD-based solvers. We illustrate this point by discussing some experimental results in this direction. Let us remark that refinements of our algorithm are possible, e.g., by including additional optimisations ("pure literal rule").

The remainder of this paper is as follows: The next section introduces the relevant background information. Section 3 sketches an algorithm to prenex formulas. As an intermediate step within this algorithm, we discuss how QBFs can be sufficiently characterised. Section 4 concludes the paper, containing a discussion about possible optimisations within our algorithm, and a brief experimental analysis illustrating the advantage of anti-prenexing for evaluating QBFs.

[1] Of course, QBF (2) is easily identified as true by pure literals, and thus the quantifier prefix still does not reflect the complexity. However, the example is just to illustrate the basic ideas.

2 Preliminaries

Let $\mathcal{P}$ be a set of propositional atoms. Then, the language $\mathcal{L}_{\mathcal{P}}$ of quantified Boolean formulas (QBFs) over $\mathcal{P}$ is obtained by ordinary propositional formulas (including propositional constants $\top$ and $\bot$) over $\mathcal{P}$ plus the additional possibility to quantify over propositional variables. A quantifier is either existential ($\exists$) or universal ($\forall$). QBFs are denoted by Greek upper-case letters.

For an indexed set $P = \{p_1, \ldots, p_n\}$ of propositional variables and a quantifier $\mathsf{Q} \in \{\exists, \forall\}$, we let $\mathsf{Q}P\,\Phi$ stand for the formula $\mathsf{Q}p_1\mathsf{Q}p_2 \ldots \mathsf{Q}p_n\,\Phi$. We say that a QBF $\mathsf{Q}_1P_1 \ldots \mathsf{Q}_nP_n\,\Phi$ is in *prenex* (*normal*) *form* if Φ is a purely propositional formula, i.e., Φ does not contain any quantifiers. For a quantifier $\mathsf{Q} \in \{\exists, \forall\}$, we define $\bar{\mathsf{Q}} = \exists$ if $\mathsf{Q} = \forall$, and $\bar{\mathsf{Q}} = \forall$ if $\mathsf{Q} = \exists$. As usual, for a QBF $\mathsf{Q}p\,\Phi$, Φ is called the *scope* of the quantifier occurrence $\mathsf{Q}p$. An occurrence of a propositional variable p in a QBF Φ is *free* iff it does not appear in the scope of a quantifier $\mathsf{Q}p$ ($\mathsf{Q} \in \{\forall, \exists\}$). If Φ contains no free variable occurrences, then Φ is *closed*, otherwise Φ is *open*. We denote the set of variables occurring free in Φ by *free*(Φ). The set of all quantifiers in Φ is given by $\mathsf{Q}(\Phi) := \{\mathsf{Q}p \mid \mathsf{Q}p\,\Psi \text{ is a subformula of } \Phi\}$; the set of all quantified variables in Φ is given by $quant(\Phi) := \{p \mid \mathsf{Q}p \in \mathsf{Q}(\Phi)\}$.

We also use the concept of a *formula tree*. Informally, the formula tree $\mathcal{T}_\Phi$ of a QBF Φ consists of nodes labelled with quantifiers and connectives as well as propositional variables for leaf nodes, reflecting the formula structure of Φ. The node labelled with the main connective of Φ is called the *root* of $\mathcal{T}_\Phi$ and appears on *top* of $\mathcal{T}_\Phi$. We understand the branching as *downwards*. Thus, we often use the informal notions of "going downwards" (towards to the leaves) and "going upwards" (towards the root) within a formula tree (or simply within a formula).

Concerning the semantics of QBFs, by an *interpretation* we understand a set $M \subseteq \mathcal{P}$ of atoms. Informally, an atom p is true under M iff $p \in M$. In general, the truth value, $\nu_M(\Phi)$, of a QBF Φ under an interpretation M is recursively defined as follows:

1. if $\Phi = \top$, then $\nu_M(\Phi) = 1$;
2. if $\Phi = p$ is an atom, then $\nu_M(\Phi) = 1$ if $p \in M$, and $\nu_M(\Phi) = 0$ otherwise;
3. if $\Phi = \neg\Psi$, then $\nu_M(\Phi) = 1 - \nu_M(\Psi)$;
4. if $\Phi = (\Phi_1 \wedge \Phi_2)$, then $\nu_M(\Phi) = min(\{\nu_M(\Phi_1), \nu_M(\Phi_2)\})$;
5. if $\Phi = \forall p\,\Psi$, then $\nu_M(\Phi) = \nu_M(\Psi[p/\top] \wedge \Psi[p/\bot])$;

where $\Phi[p_1/\Phi_1, \ldots, p_n/\Phi_n]$ denotes the result of uniformly substituting each free occurrence of a variable p_i in Φ by Φ_i, for $1 \leq i \leq n$.

The truth conditions for $\bot$, $\vee$, $\rightarrow$, $\equiv$, and $\exists$ follow from 1.–5. in the usual way. Note that $\exists$ is defined here similarly as in first-order logic, i.e., $\exists p\,\Psi = \neg\forall p\,\neg\Psi$, for each formula Ψ. Hence, the truth value for $\exists$ is given by

$$\nu_M(\exists p\,\Psi) = \nu_M(\Psi[p/\top] \vee \Psi[p/\bot]).$$

Due to the associativity and commutativity of $\wedge$ and $\vee$, we allow n-ary conjunctions and disjunctions (with $n \geq 2$) to appear in arbitrary order.

We say that Φ is *true under* M iff $\nu_M(\Phi) = 1$, otherwise Φ is *false under* M. If $\nu_M(\Phi) = 1$, then M is a *model* of Φ. If Φ is true under every interpretation, then Φ is *valid*. As usual, we write $\models \Phi$ to express that Φ is valid.

It is easily seen that the truth value of a QBF Φ under interpretation M depends only on the free variables in Φ. In particular, closed QBFs are either true under every interpretation or false under every interpretation, i.e., they are either valid or unsatisfiable. Two formulas are *logically equivalent* iff they possess the same models. Thus, formulas Φ and Ψ are logically equivalent iff $\Phi \equiv \Psi$ is valid.

In what follows, we note some useful relations concerning the shifting and renaming of quantifiers, paralleling similar results from standard first-order logic.

Proposition 1 *Let p, q be atoms, $\mathsf{Q} \in \{\forall, \exists\}$, and let Φ, Φ_1, Φ_2, and Ψ be QBFs such that Ψ does not contain free occurrences of p. Then,*

1. $\models (\mathsf{Q}p\,\Psi) \equiv \Psi$;
2. $\models (\mathsf{Q}q\,\Psi) \equiv (\mathsf{Q}p\,\Psi[q/p])$;
3. $\models (\neg \mathsf{Q}p\,\Phi) \equiv \bar{\mathsf{Q}}p(\neg\Phi)$;
4. $\models \exists p\,(\Phi_1 \vee \Phi_2) \equiv (\exists p\,\Phi_1 \vee \exists p\,\Phi_2)$;
5. $\models \forall p\,(\Phi_1 \wedge \Phi_2) \equiv (\forall p\,\Phi_1 \wedge \forall p\,\Phi_2)$;
6. $\models \exists p\,(\Phi_1 \rightarrow \Phi_2) \equiv (\forall p\,\Phi_1 \rightarrow \exists p\,\Phi_2)$;
7. $\models \mathsf{Q}p\,(\Phi \circ \Psi) \equiv (\mathsf{Q}p\,\Phi) \circ \Psi$ *for* $\circ \in \{\wedge, \vee\}$;
8. $\models \mathsf{Q}p\,(\Phi \rightarrow \Psi) \equiv (\bar{\mathsf{Q}}p\,\Phi \rightarrow \Psi)$;
9. $\models \mathsf{Q}p\,(\Psi \rightarrow \Phi) \equiv (\Psi \rightarrow \mathsf{Q}p\,\Phi)$; *and*
10. $\models (\mathsf{Q}p\,\mathsf{Q}q\,\Phi) \equiv (\mathsf{Q}q\,\mathsf{Q}p\,\Phi)$.

Our algorithm basically relies on repetitive application of replacing such equivalent formulas. Therefore, recall that the replacement theorem holds for QBFs.

Proposition 2 *Let Ψ be a subformula of a QBF Φ and assume $\models \Psi \equiv \Psi'$. Then, $\models \Phi \equiv \Phi'$, where Φ' results from Φ by replacing one or more occurrences of Ψ in Φ by Ψ'.*

Straightforward transformation techniques are usually based on a *nondeterministic* application of replacements of equivalent subformulas. Therefore, they result in a number of different prenex forms in general. In other words, considering above replacements as a set of rewriting rules, we get a non-confluent set. In this paper, however, we are concerned with a deterministic algorithm,"hiding" possible nondeterministic choices within a construction of a total order of the elements in $quant(\Phi)$.

Finally, let us briefly recall that QBFs play a central role in complexity theory representing a natural decision problem for the complexity class PSPACE. Moreover, the evaluation problem for a QBF $\mathsf{Q}_1 P_1 \ldots \mathsf{Q}_k P_k \phi$ having prenex normal form with $k \geq 1$ alternating quantifiers is complete for Σ_k^P if the outermost quantifier is existential, and complete for Π_k^P if the outermost quantifier is universal. Recall that $\Sigma_1^P = \mathrm{NP}$, $\Sigma_2^P = \mathrm{NP}^{\mathrm{NP}}$, $\Pi_2^P = \text{co-NP}^{\mathrm{NP}}$, etc. are constituting members of the polynomial hierarchy [13]. We also consider the complexity classes D_k^P, $k \geq 1$, where each D_k^P consists of all problems expressible as the conjunction of a problem in Σ_k^P and a problem in Π_k^P. Hence, the problem of (independently) evaluating two QBFs $\mathsf{Q}_1 P_1 \ldots \mathsf{Q}_k P_k \phi$ and $\bar{\mathsf{Q}}_1 P_1' \ldots \bar{\mathsf{Q}}_k P_k' \phi'$ with k alternating quantifiers is contained in D_k^P.

3 A Prenex Normal-Form Translation

In order to translate arbitrary QBFs into prenex form, our overall strategy is as follows.

1. shift quantifiers down the formula tree;
2. classify QBFs via those paths in the resulting tree which posses a maximal number of quantifier alternations;
3. shift quantifiers to the root of the tree by "collecting" all quantifiers on such a path.

The down shifting of quantifiers (also referred to as *anti-prenexing*) is essential for obtaining prenex QBFs possessing "optimal" quantifier alternations compared to a straightforward approach based on shifting quantifiers outside using the equivalence transformations given in Proposition 1. In fact, it holds that, for any QBF Φ, the number of quantifier alternations in the translated QBF Φ' obtained from our algorithm is never greater than the number of quantifier alternations in a translated QBF Φ'' obtained in a normal-form procedure based on a simple out-shifting of quantifiers. Moreover, reducing the scope of quantifiers is especially beneficial for QBF-solvers allowing input formulas which are not required to enjoy a particular normal form, like, e.g., BDD-based solvers which are able to process arbitrary QBFs. We illustrate this point later on by using particular problem instances in which reducing the quantifier scope yields a significant speed-up of computation time.

In the following, all of the above steps are shown to be polynomial-time computable and equivalence preserving. The algorithm is also applicable to open QBFs, leaving the set of free variables unchanged. With suitable renaming schemes, the original QBF is in principle also reconstructible from the result. For the sake of simplicity, we define our algorithm for QBFs built from connectives, $\wedge$, $\vee$ and $\neg$.

3.1 Down-shifting of Quantifiers

To apply our algorithm, we first use some simple pre-processing deriving so-called *cleansed* QBFs. A cleansed QBF Φ satisfies the following conditions:

1. $\mathit{free}(\Phi) \cap \mathit{quant}(\Phi) = \emptyset$; i.e., no atom occurs both free and quantified in Φ;
2. if $\mathsf{Q}_1 p_1\, \Phi_1$ and $\mathsf{Q}_2 p_2\, \Phi_2$ are different subformula occurrences in Φ, then $p_1 \neq p_2$.

Both properties are easily achieved by renaming bounded variables. The first step of our algorithm takes an arbitrary cleansed QBF Φ and shifts quantifiers as deep into the formula tree $\mathcal{T}_\Phi$ as possible, using the following extended versions of the equivalence retaining rules from Proposition 1 for conjunction and disjunction.

Lemma 1 *Let $\Phi = \Phi_1 \circ \ldots \circ \Phi_n \circ \Psi_1 \circ \ldots \circ \Psi_m$ be a QBF with $\circ \in \{\wedge, \vee\}$, $m, n \geq 0$, and $m + n > 0$, such that an atom p is contained in each $\Phi_1, \ldots, \Phi_n$ but not in $\Psi_1 \circ \ldots \circ \Psi_m$. Moreover, let $p_1, \ldots, p_n$ be globally new atoms. Then,*

1. *for $\circ = \vee$,*
   (a) $\models \exists p\, \Phi \equiv (\exists p_1\, \Phi_1[p/p_1] \vee \ldots \vee \exists p_n\, \Phi_n[p/p_n] \vee \Psi_1 \vee \ldots \vee \Psi_m)$;
   (b) $\models \forall p\, \Phi \equiv \forall p(\Phi_1 \vee \ldots \vee \Phi_n) \vee \Psi_1 \vee \ldots \vee \Psi_m$;
2. *for $\circ = \wedge$,*
   (a) $\models \forall p\, \Phi \equiv (\forall p_1\, \Phi_1[p/p_1] \wedge \ldots \wedge \forall p_n\, \Phi_n[p/p_n] \wedge \Psi_1 \wedge \ldots \wedge \Psi_m)$;
   (b) $\models \exists p\, \Phi \equiv \exists p(\Phi_1 \wedge \ldots \wedge \Phi_n) \wedge \Psi_1 \wedge \ldots \wedge \Psi_m$.

Our algorithm starts with quantifiers located lowest in the formula tree $\mathcal{T}_\Phi$ and then applies to quantifiers iteratively located upwards. In fact, we use the following concepts for the iteration order.

Definition 1 *Let Φ be a cleansed QBF and let $\sigma = \mathsf{Q}_1 p_1, \ldots, \mathsf{Q}_n p_n$ be a sequence of all elements from $\mathsf{Q}(\Phi)$.*

Then, σ is called partial *if, for all i, j with $i > j$ and $\mathsf{Q}_i \neq \mathsf{Q}_j$, it holds that $\mathsf{Q}_i p_i\, \Psi_i$ is not a subformula of $\mathsf{Q}_j p_j\, \Psi_j$. Furthermore, σ is called* strictly partial *if it satisfies the condition for partiality, except that the proviso $\mathsf{Q}_i \neq \mathsf{Q}_j$ is dropped.*

Note that since Φ is assumed to be cleansed, the formulas Ψ_i, Ψ_j are unambiguously identifiable. Strictly partial sequences reflect exactly the dependencies of quantifiers in a given QBF, whilst partial sequences extend the freedom of selecting an order by taking Item (10) from Proposition 1 into account. Obviously, prenex QBFs possess exactly one strictly partial sequence, viz. the quantifier prefix itself in inverse order.

For illustration, recall QBF (1) and transform it into a cleansed form, e.g., into

$$\exists p[\forall q\, \exists r\, (p \vee q \vee r) \wedge \exists r'\, \forall q'\, (\neg p \vee q' \vee r')\,].$$

There are several possible partial sequences for this QBF, e.g.,

$$\exists r, \quad \forall q, \quad \forall q', \quad \exists r', \quad \exists p. \tag{5}$$

(5) is also strictly partial, while

$$\exists r, \quad \forall q, \quad \forall q', \quad \exists p, \quad \exists r'$$

is partial but not strictly partial.

We continue with the description of our anti-prenexing algorithm. First, we define the following recursive operation $S^{\downarrow}(\cdot)$. For any QBF Ψ, each $\mathsf{Q} \in \{\exists, \forall\}$, and any atom p, $S^{\downarrow}(\mathsf{Q}p\, \Psi)$ is given as follows:

1. if $p \notin \mathit{free}(\Psi)$, then $S^{\downarrow}(\mathsf{Q}p\,\Psi) = \Psi$;

2. if $\Psi = p$ then $S^{\downarrow}(\mathsf{Q}p\,\Psi) = \mathsf{Q}p\,\Psi$;

3. if $\Psi = \neg\Psi'$, then $S^{\downarrow}(\mathsf{Q}p\,\Psi) = \neg S^{\downarrow}(\bar{\mathsf{Q}}p\,\Psi')$;

4. if $\Psi = \Psi_1 \circ \ldots \circ \Psi_l \circ \Psi_{l+1} \circ \ldots \circ \Psi_m$ with p occurring in $\Psi_1, \ldots, \Psi_l$ but not in (the possibly empty sequence) $\Psi_{l+1}, \ldots, \Psi_m$ and $m \geq l$, then

$$S^{\downarrow}(\mathsf{Q}p\,\Psi) = S^{\downarrow}(\mathsf{Q}p_1\,\Psi_1[p/p_1]) \circ \ldots \circ S^{\downarrow}(\mathsf{Q}p_l\,\Psi_n[p/p_l]) \circ \Psi_{l+1} \circ \ldots \circ \Psi_m,$$

   where $p_1, \ldots, p_l$ are globally new variables, and $\circ = \vee$ if $\mathsf{Q} = \exists$ and $\circ = \wedge$ if $\mathsf{Q} = \forall$;

5. if $\Psi = \Psi_1 \circ \ldots \circ \Psi_l \circ \Psi_{l+1} \circ \ldots \circ \Psi_m$ with p occurring in $\Psi_1, \ldots, \Psi_l$ but not in $\Psi_{l+1}, \ldots, \Psi_m$ and $m > l$, then

$$S^{\downarrow}(\mathsf{Q}p\,\Psi) = S^{\downarrow}(\mathsf{Q}p\,(\Psi_1 \circ \ldots \circ \Psi_l)) \circ \Psi_{l+1} \circ \ldots \circ \Psi_m$$

   for $\circ = \wedge$ if $\mathsf{Q} = \exists$ and $\circ = \vee$ if $\mathsf{Q} = \forall$.

Observe that the recursion also comes to a halt whenever we have $m = l$ in Step 5, or $\Psi = \mathsf{Q}'q\Psi'$ already has a leading quantifier. This is sufficient for strictly partial sequences. However, if we deal with partial sequences in general we have to allow that equal quantifiers are exchangeable with respect to the given sequence σ. Thus, let σ be a sequence as in Definition 1, then $S^{\downarrow}(\mathsf{Q}p\,\Psi)$ is extended by the following step.

6. if $\Psi = \mathsf{Q}q\Psi'$ and $\mathsf{Q}p$ appears in front of $\mathsf{Q}q$ in σ, then $S^{\downarrow}(\mathsf{Q}p\Psi) = \mathsf{Q}qS^{\downarrow}(\mathsf{Q}p\Psi')$.

Definition 2 *Let Φ be a cleansed QBF and $\sigma = \mathsf{Q}_1\,p_1, \ldots, \mathsf{Q}_n\,p_n$ a partial sequence for Φ. Moreover, let $\Phi_0 = \Phi$ and let Φ_i be the QBF resulting from replacing the subformula*[2] $\mathsf{Q}_i p_i\,\Psi'_i$ *in Φ_{i-1} by $S^{\downarrow}(\mathsf{Q}_i p_i\,\Psi'_i)$.*

Then, Φ_n, which is the final result of applying $S^{\downarrow}(\cdot)$ to all elements in the given sequence σ, is called the scope-cleansed form of Φ (with respect to σ)*, denoted by $C_\sigma(\Phi)$.*

The adequacy of the algorithm, as stated next, follows from Proposition 1 and Lemma 1.

Theorem 1 *Let Φ be an arbitrary cleansed QBF. Then, for each sequence σ of elements from $\mathsf{Q}(\Phi)$,*

1. *$C_\sigma(\Phi)$ is equivalent to Φ;*

2. *the time to construct $C_\sigma(\Phi)$ is at most quadratic in the logical complexity of Φ; and*

3. *$\mathit{free}(C_\sigma(\Phi)) = \mathit{free}(\Phi)$.*

[2]Note that the scope Ψ_i of the quantifier $\mathsf{Q}p_i$ in Φ may has changed during the construction of Φ_i. However, the formula is still identifiable by the unique quantification $\mathsf{Q}_i p_i$.

Theorem 2 *Let Φ be an arbitrary cleansed QBF and σ_1, σ_2 strictly partial sequences of all members from $\mathsf{Q}(\Phi)$. Then, $C_{\sigma_1}(\Phi) = C_{\sigma_2}(\Phi)$ holds.*

For illustration, recall QBF (1) in the cleansed form as above and consider the quantifier sequence σ as in (5).

The first element in σ is $\exists r$ and thus we start with $\exists r\,(p \vee q \vee r)$. Application of Step 4 yields $S^{\downarrow}(\exists r\,(p \vee q \vee r)) = (p \vee q \vee \exists r_1\, r_1)$. Hence, Φ_1 is given by

$$\exists p\,[\forall q\,(p \vee q \vee \exists r_1\, r_1) \wedge \exists r' \forall q' (\neg p \vee q' \vee r')].$$

We proceed by computing $S^{\downarrow}(\forall q\,(p \vee q \vee \exists r_1\, r_1))$. Applying Step 5 yields $(p \vee \forall q_1\, q_1 \vee \exists r_1\, r_1)$. Similar applications in the second conjunct of Φ lead to

$$\Phi_4 = \exists p\,[(p \vee \forall q_1\, q_1 \vee \exists r_1\, r_1) \wedge (\neg p \vee \forall q_2\, q_2 \vee \exists r_2\, r_2)]. \tag{6}$$

Now, since here in the iteration $S^{\downarrow}(\Phi_4)$ no further step is applicable, we end up with the QBF $C_\sigma(\Phi) = \Phi_4$.

Clearly, in (5), subformulas of form $\forall q_1\, q_1$ and $\exists r_1\, r_1$ could straightforwardly be replaced by constants $\bot$ and $\top$, respectively. Note that such simple replacements make a reconstruction of the original formula impossible in general.

3.2 Classification Step

Having constructed $C_\sigma(\Phi)$ and its formula tree, we are now able to give a suitable classification for Φ. We start with the following definitions.

Let $\mathcal{T}_\Phi$ be the formula tree of a QBF Φ. A *q-path*, α, in $\mathcal{T}_\Phi$ is a sequence of quantifiers $\mathsf{Q}_1 p_1 \ldots \mathsf{Q}_n p_n$ resulting from collecting all quantifiers occurring on a path in $\mathcal{T}_\Phi$ starting from the root to its leaf. For a q-path α, define $n(\alpha)$ as the number of quantifier alternations in α plus 1, and let $\mathsf{Q}(\alpha)$ be the leading quantifier, Q_1, in α.

We now define the following classes of QBFs.

Definition 3 *Let Φ be an arbitrary closed QBF in cleansed form, Q some quantifier, σ a sequence of elements from $\mathsf{Q}(\Phi)$, and $n > 0$. Then,*

1. *$\Phi \in \mathcal{C}_n^{\mathsf{Q}}$ iff (i) there exists a q-path α in $\mathcal{T}_{C_\sigma(\Phi)}$ with $n(\alpha) = n$ and $\mathsf{Q}(\alpha) = \mathsf{Q}$, (ii) there is no q-path β of $\mathcal{T}_{C_\sigma(\Phi)}$ such that $n(\beta) > n$, and (iii) each q-path γ of $\mathcal{T}_{C_\sigma(\Phi)}$ with $n(\gamma) = n$ satisfies $\mathsf{Q}(\alpha) = \mathsf{Q}(\gamma)$;*
2. *$\Phi \in \mathcal{C}_n^{D}$ iff (i) there exist q-paths α, β of $\mathcal{T}_{C_\sigma(\Phi)}$ with $n(\alpha) = n(\beta) = n$ and $\mathsf{Q}(\alpha) \neq \mathsf{Q}(\beta)$, and (ii) there is no q-path γ of $\mathcal{T}_{C_\sigma(\Phi)}$ with $n(\gamma) > n$.*

Lemma 2 *The time to classify a QBF Φ with respect to the family of sets in Definition 3 is linear in the logical complexity of $C_\sigma(\Phi)$ (which is at most quadratic in the complexity of Φ).*

Obviously, each closed QBF Φ is contained in exactly one of the classes $\mathcal{C}_n^{\exists}$, $\mathcal{C}_n^{\forall}$, and $\mathcal{C}_n^{D}$ ($n > 0$). In fact, it holds that if Φ is contained $\mathcal{C}_n^{\exists}$ (resp. $\mathcal{C}_n^{\forall}$ or $\mathcal{C}_n^{D}$), then the evaluation problem for Φ is in Σ_n^P (resp. Π_n^P or D_n^P). In general, this gives a better upper bound for classifying the computational complexity of evaluating a given QBF as, e.g., a simple inspection of the quantifier order of the prefix. In Section 4 we briefly mention some optimisations for sharpening these upper bounds.

The inherent complexity of decision problems associated with open QBFs is treatable in a similar manner. If we are interested in *satisfiability* of an open QBF Φ with $\mathit{free}(\Phi) = P$, an upper bound for this problem is derivable via determining the corresponding class for the existential closure $\exists P\, \Phi$ of Φ; to classify the *validity* problem of Φ, we use $\forall P\, \Phi$. With a slight abuse of notation, we say that an open QBF Φ is contained in a class $\mathcal{C}$ if its existential (resp. universal) closure is contained in this class, whenever we are interested in the satisfiability (resp. validity) problem for Φ.

The following theorem expresses a general property for partial sequences, similar to Theorem 2 given for strictly partial sequences.

Theorem 3 *Let Φ be a cleansed QBF. Then, for each partial sequence σ of all elements from $\mathsf{Q}(\Phi)$, $\mathcal{T}_{C_\sigma(\Phi)}$ yields the same classification for Φ with respect to the sets of Definition 3.*

Reconsider our running example (6) from above. Here, we identify two q-paths with a maximal number of quantifier alternations, viz.

$$\alpha_1 = \exists p \forall q_1 \quad \text{and} \quad \alpha_2 = \exists p \forall q_2. \tag{7}$$

For both paths α_i ($i \in \{1, 2\}$), we have $n(\alpha_i) = 2$ and $\mathsf{Q}(\alpha_i) = \exists$. Hence, Φ is classified as $\mathcal{C}_2^{\exists}$, which correctly reflects our analysis in the introduction (cf. QBF (2)).

3.3 Up-shifting of Quantifiers

We now shift quantifiers upwards again, such that the resulting QBF is in the desired prenex form. Similarly to the shift-down procedure, we start with a lemma stating an extended version of quantifier-shifting for conjunction and disjunction.

Lemma 3 *Let $\Phi = \Phi_1 \circ \ldots \circ \Phi_n \circ \Psi_1 \circ \ldots \circ \Psi_m$ be a QBF with $\circ \in \{\wedge, \vee\}$, $n > 0$, and $m \geq 0$, such that each Φ_i is of form $\mathsf{Q}p_i\, \Phi_i'$ and each Ψ_j is either of form $\bar{\mathsf{Q}}q_j\, \Psi_j'$ or quantifier-free. Moreover, assume each p_i occurs only in Φ_i and p is a globally new atom.*

Then, Φ is equivalent to

1. *$\mathsf{Q}p\, (\Phi_1[p_1/p] \circ \ldots \circ \Phi_n[p_n/p]) \circ \Psi_1 \circ \ldots \circ \Psi_m$, for $\circ = \vee$ if $\mathsf{Q} = \exists$ and $\circ = \wedge$ if $\mathsf{Q} = \forall$; and*

2. *$\mathsf{Q}p_1 \ldots \mathsf{Q}p_n(\Phi_1 \circ \ldots \circ \Phi_n) \circ \Psi_1 \circ \ldots \circ \Psi_m$, for $\circ = \wedge$ if $\mathsf{Q} = \exists$ and $\circ = \vee$ if $\mathsf{Q} = \forall$.*

In the subsequent procedure, we use the following notion. For a QBF $\Phi = \Phi_1 \circ \ldots \circ \Phi_n$ ($\circ \in \{\wedge, \vee\}$), we define Φ^{Q} as the result of replacing each Φ_i of form $\mathsf{Q}p_i\, \Phi_i'$ by Φ_i'. Moreover, the set of each such atom p_i is denoted by $P(\Phi, \mathsf{Q})$.

To begin with, we define the following *merging function*, $M(\cdot, \cdot)$. Let Φ be a QBF of form $\Phi_1 \circ \ldots \circ \Phi_n$ ($\circ \in \{\wedge, \vee\}$). Then,

1. if $quant(\Phi) = \emptyset$,
   then $M(\Phi, \mathsf{Q}) = \Phi$; otherwise

2. if, for each $i = 1, \ldots, n$, either $\Phi_i = \bar{\mathsf{Q}} p_i \, \Phi'_i$ or $quant(\Phi'_i) = \emptyset$,
   then $M(\Phi, \mathsf{Q}) = M(\Phi, \bar{\mathsf{Q}})$; otherwise

3. if $\circ = \vee$ for $\mathsf{Q} = \exists$ or $\circ = \wedge$ for $\mathsf{Q} = \forall$, and $P(\Phi, \mathsf{Q}) = \{p_1, \ldots, p_k\}$,
   then $M(\Phi, \mathsf{Q}) = \mathsf{Q}p \, M(\Phi^{\mathsf{Q}}[p_1/p, \ldots, p_k/p], \mathsf{Q})$, where p is a globally new variable;

4. if $\circ = \wedge$ for $\mathsf{Q} = \exists$ or $\circ = \vee$ for $\mathsf{Q} = \forall$, and $P(\Phi, \mathsf{Q}) = \{p_1, \ldots, p_k\}$,
   then $M(\Phi, \mathsf{Q}) = \mathsf{Q}p_1 \ldots \mathsf{Q}p_k \, M(\Phi^{\mathsf{Q}}, \mathsf{Q})$.

The merging function $M(\cdot, \cdot)$ implements the "shift-to-top" strategy as discussed in Section 1. Slight adoptions for the "shift-to-bottom" strategy are rather easy to obtain, although the concrete definition would be a little more cumbersome. Note that Step 3 in the merging function implements the concept of *quantifier fusion*. The correctness of this rule is reflected by Condition 1 of Lemma 3.

We proceed by defining a recursive function, $S^{\uparrow}(\cdot, \cdot)$, as follows. For any QBF Φ and each $\mathsf{Q} \in \{\exists, \forall\}$, $S^{\uparrow}(\Phi, \mathsf{Q})$ is given as follows:

1. if $\Phi = \mathsf{Q}_0 p_0 \, \Phi'$ with $quant(\Phi') = \emptyset$,
   then $S^{\uparrow}(\Phi, \mathsf{Q}) = \Phi$;

2. if $\Phi = \mathsf{Q}_0 p_0 \, \Phi'$ with $quant(\Phi') \neq \emptyset$,
   then $S^{\uparrow}(\Phi, \mathsf{Q}) = \mathsf{Q}_0 p_0 \, S^{\uparrow}(\Phi', \mathsf{Q}_0)$;

3. if $\Phi = \neg\Phi'$ and $S^{\uparrow}(\Phi', \bar{\mathsf{Q}}) = \mathsf{Q}_1 p_1 \ldots \mathsf{Q}_n p_n \phi'$,
   then $S^{\uparrow}(\Phi, \mathsf{Q}) = \bar{\mathsf{Q}}_1 p_1 \ldots \bar{\mathsf{Q}}_n p_n \neg\phi'$;

4. if $\Phi = \Phi_1 \circ \ldots \circ \Phi_n$,
   then $S^{\uparrow}(\Phi, \mathsf{Q}) = M(S^{\uparrow}(\Phi_1, \mathsf{Q}) \circ \ldots \circ S^{\uparrow}(\Phi_n, \mathsf{Q}), \mathsf{Q})$, with $\circ \in \{\wedge, \vee\}$.

To eventually obtain our desired prenex QBF form, we have the following result.

Theorem 4 *For any $\Phi \in \mathcal{C}_n^{\mathsf{Q}}$ and any sequence σ of all elements from $\mathsf{Q}(\Phi)$, let $P(\Phi) = S^{\uparrow}(\Phi_{C_\sigma}, \mathsf{Q})$. Then,*

1. *$P(\Phi)$ is in prenex form;*

2. *$P(\Phi)$ is equivalent to Φ;*

3. *the time to construct $P(\Phi)$ is at most quadratic in the logical complexity of Φ; and*

4. *$free(P(\Phi)) = free(\Phi)$.*

Theorem 5 *Let Φ be an arbitrary cleansed QBF. Then, all strictly partial sequences σ of all elements from $\mathsf{Q}(\Phi)$ yield syntactically identical QBFs $S^{\uparrow}(\Phi_{C_\sigma}, \mathsf{Q})$. Moreover, all partial sequences σ' of all elements from $\mathsf{Q}(\Phi)$ yield equivalent QBFs $S^{\uparrow}(\Phi_{C_{\sigma'}}, \mathsf{Q})$ having the same quantifier structure in their prefix.*

We illustrate the procedure $S^{\uparrow}(\cdot,\cdot)$ on our running example

$$C_\sigma(\Phi) = \exists p\,[(p \vee \forall q_1\, q_1 \vee \exists r_1\, r_1) \wedge (\neg p \vee \forall q_2\, q_2 \vee \exists r_2\, r_2)].$$

We already derived $\Phi \in \mathcal{C}_2^{\exists}$, thus we have "$\exists$" as second argument in $S^{\uparrow}(\cdot,\cdot)$. Proceeding with the recursion yields

$$S^{\uparrow}(C_\sigma(\Phi),\exists) = \exists p\, M\Big(M(p \vee \forall q_1\, q_1 \vee \exists r_1\, r_1, \exists) \wedge M(\neg p \vee \forall q_2\, q_2 \vee \exists r_2\, r_2, \exists), \exists\Big).$$

Consider $M(p \vee \forall q_1\, q_1 \vee \exists r_1\, r_1, \exists)$. Since the second argument is given by $\exists$, we first shift $\exists r_1$ up and afterwards $\forall q_1$, yielding $\exists r_1\, \forall q_1(p \vee q_1 \vee r_1)$; and similarly in the second conjunct. Thus, we arrive at

$$S^{\uparrow}(C_S(\Phi),\exists) = \exists p\, M\Big(\exists r_1 \forall q_1(p \vee q_1 \vee r_1) \wedge \exists r_2 \forall q_2(\neg p \vee q_2 \vee r_2), \exists\Big).$$

Now we compute the remaining merging function. To this end, let $\Psi_1 = (p \vee q_1 \vee r_1)$ and $\Psi_2 = (\neg p \vee q_2 \vee r_2)$. We first consider $M(\exists r_1 \forall q_1 \Psi_1 \wedge \exists r_2 \forall q_2 \Psi_2, \exists)$. Here, no fusion is possible and, due to Condition 4 of the merging function, we shift both quantifiers upwards, resulting in $\exists r_1 \exists r_2 M(\forall q_1 \Psi_1 \wedge \forall q_2 \Psi_2, \exists)$. Since no leading quantifier is an existential one, we apply Condition 2, getting $M(\forall q_1 \Psi_1 \wedge \forall q_2 \Psi_2, \forall)$. Now quantifier fusion is possible, in view of Condition 3, and we get $\forall q(\Psi_1[q_1/q] \wedge \Psi_2[q_2/q])$, and thus as final result

$$S^{\uparrow}(C_S(\Phi),\exists) = \exists p\, \exists r_1\, \exists r_2\, \forall q[(p \vee q \vee r_1) \wedge (\neg p \vee q \vee r_2)].$$

In concluding, we briefly sketch how above procedure is applicable if the given QBF Φ is in $\mathcal{C}_n^D$. Omitting further details, we extend the merging function $M(\cdot,\cdot)$ by

5. $M(\Phi_1 \circ \ldots \circ \Phi_n, D) = M(\Phi_\exists, \exists) \circ M(\Phi_\forall, \forall)$ where Φ_{Q} denotes the conjunction or disjunction of those Φ_i with leading quantifier Q;

and start with $S^{\uparrow}(\Phi, D)$. This yields QBFs of form $\Phi_1 \circ \Phi_2$ where Φ_1, Φ_2 are in prenex form and the leading quantifiers in Φ_1 and Φ_2 are different. We call such QBFs as being in *D-normal form*. Observe that QBFs in D-normal form are straightforwardly evaluated via two independently calls to a QBF-solver, and thus reflect the prototypical problems for the complexity-classes D_k^P.

4 Discussion and Conclusion

In this paper, we presented an algorithm for generating QBFs in prenex form guaranteeing, in some sense, that the resultant QBFs possess an "optimal" number of quantifier alternations compared to procedures based on a straightforward shifting method. One of the distinguishing features of our method is an anti-prenexing step, moving quantifiers temporarily to the inside of a formula.

In the following, we briefly point out possible optimisations and discuss some experimental results concerning the advantage of anti-prenexing with respect to evaluating QBFs using BDD-based QBF-solvers.

To begin with, we note that the scope-cleansed QBF $C_\sigma(\Phi)$ in (6) can easily be simplified by replacing subformulas like $\mathsf{Q}p\,p$ by $\top$ for $\mathsf{Q} = \exists$ and by $\bot$ for $\mathsf{Q} = \forall$. In general, such simplifications may yield that the transformed QBFs possess a smaller number of quantifier alternations, albeit the additional optimisation steps prevent a reconstruction of the original QBF. Another issue is that the possible gain in terms of formula simplification depends on the chosen sequence σ of the elements of $\mathsf{Q}(\Phi)$. Thus, classifications of simplified QBFs may yield different results when $C_\sigma(\Phi)$ is obtained from different sequences σ.

In any case, we suggest the following extension of our basic algorithm:

1. identify possible simplifications;

2. evaluate the resultant QBFs with respect to subformulas containing $\top$ and $\bot$;

3. employ $S^{\downarrow}(\cdot)$;

4. repeat this procedure until no further simplifications are possible.

Currently, the implementation of our algorithm involves the following simplifications: (i) replacing formulas of form $p \circ \neg p \circ \Phi$ by $\top$ for $\circ = \vee$ and by $\bot$ for $\circ = \wedge$, and (ii) the *pure literal rule*. By a pure literal, we understand a literal such that all bound occurrences in some given cleansed QBF have the same polarity. Then, the pure literal rule states that a pure literal p is replaced by $\top$ if $\mathsf{Q} = \exists$ and p occurs only positively or if $\mathsf{Q} = \forall$ and p occurs only negatively, and by $\bot$ for the dual cases.

The second step above means formula simplifications based on the usual valid equivalences associated with $\top$ and $\bot$, like $(\top \wedge \Psi) \equiv \Psi$, etc.

The third step has several consequences. First, it eliminates quantifications which have no effect due to the optimisations. Second, it allows further shiftings of quantifiers deeper into the formula tree, which may become attainable after the applied elimination steps. However, due to possible splittings within $S^{\downarrow}(\cdot)$, new pure literals may be identified.

Finally, we mention some experimental results which show the advantage of anti-prenexing in terms of running time for evaluating QBFs. To wit, we used a class of benchmark examples taken from [1] and compared their running times using several QBF-solvers with variations of these formulas where anti-prenexing is applied. More specifically, we employed the solvers `semprop` [10], `ssolve` [6], and `boole`. The latter is a BDD-based propositional solver publicly available from [2]. As benchmark problems we used the examples `tree-exa2-30` to `tree-exa2-50` from [1]. For each of these examples, applying anti-prenexing yielded a significant reduction of running time for the BDD-solver `boole`. For instance, for `tree-exa2-50`, none of the applied solvers was able to evaluate this formula within 10 minutes, but the anti-prenexed version was computed by `boole` in less than 0.1 second. Although of course further experimental evaluation is needed to obtain significant results, these examples show the potential benefit of applying anti-prenexing steps. Other future work includes exploiting more sophisticated techniques for quantifier shifting which have been proposed for first-order logic as well as a careful evaluation of the potential practical value of such methods when applied to QBF solving.

References

[1] http://wwwjessen.informatik.tu-muenchen.de/∼letz/semprop/.

[2] http://www.cs.cmu.edu/∼modelcheck/bdd.html.

[3] M. Cadoli, A. Giovanardi, and M. Schaerf. An Algorithm to Evaluate Quantified Boolean Formulae. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 262–267, 1998.

[4] U. Egly, T. Eiter, R. Feldmann, V. Klotz, S. Schamberger, H. Tompits, and S. Woltran. On Mechanizing Modal Nonmonotonic Logics. In *Proceedings of the 5th Dutch-German Workshop on Nonmonotonic Reasoning Techniques and their Applications (DGNMR-01)*, pages 44–53, 2001.

[5] U. Egly, T. Eiter, H. Tompits, and S. Woltran. Solving Advanced Reasoning Tasks Using Quantified Boolean Formulas. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00)*, pages 417–422, 2000.

[6] R. Feldmann, B. Monien, and S. Schamberger. A Distributed Algorithm to Evaluate Quantified Boolean Formulas. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00)*, pages 285–290, 2000.

[7] I. P. Gent and T. Walsh. Beyond NP: The QSAT Phase Transition. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*, pages 648–653, 1999.

[8] E. Giunchiglia, M. Narizzano, and A. Tacchella. Backjumping for Quantified Boolean Logic Satisfiability. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*, 2001.

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

[10] R. Letz. Advances in Decision Procedures for Quantified Boolean Formulas. In *Proceedings of the IJCAR 2001 Workshop on Theory and Applications of Quantified Boolean Formulas (QBF-01)*, pages 55–64, 2001.

[11] D. Pearce, H. Tompits, and S. Woltran. Encodings for Equilibrium Logic and Logic Programs with Nested Expressions. In *Proceedings of the 10th Portuguese Conference on Artificial Intelligence (EPIA-01)*, pages 306–320, 2001.

[12] J. Rintanen. Improvements to the Evaluation of Quantified Boolean Formulae. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 1192–1197, 1999.

[13] L. J. Stockmeyer. The Polynomial-Time Hierarchy. *Theoretical Computer Science*, 3:1–22, 1977.