Enumerative Level-2 Solution Counting for Quantified Boolean Formulas

Andreas Plank, Sibylle Möhle, Martina Seidl

Enumerative Level-2 Solution Counting for Quantified Boolean Formulas

Andreas Plank
Institute for Symbolic Artificial Intelligence, JKU Linz, Austria

Sibylle Möhle
Max Planck Institute for Informatics, Saarbrücken, Germany

Martina Seidl
Institute for Symbolic Artificial Intelligence, JKU Linz, Austria

Abstract

We lift the problem of enumerative solution counting to quantified Boolean formulas (QBFs) at the second level. In contrast to the well-explored model counting problem for SAT (#SAT), where models are simply assignments to the Boolean variables of a formula, we are now dealing with tree (counter-)models reflecting the dependencies between the variables of the first and the second quantifier block. It turns out that enumerative counting on the second level does not give the complete model count. We present the – to the best of our knowledge – first approach of counting tree (counter-)models together with a counting tool that exploits state-of-the-art QBF technology. We provide several kinds of benchmarks for testing our implementation and illustrate in several case studies that solution counting provides valuable insights into QBF encodings.

2012 ACM Subject Classification Theory of computation → Automated reasoning

Keywords and phrases QBF, Second-Level Model Counting

Digital Object Identifier 10.4230/LIPIcs.CP.2023.49

Category Short Paper

Supplementary Material *Software*: `https://qcount.pages.sai.jku.at/l2count`

Funding This work has been supported by the LIT AI Lab funded by the State of Upper Austria.

1 Introduction

Over the last decade, solvers for quantified Boolean formulas (QBFs) have become appealing tools to handle PSPACE-hard problems as found in many applications from, for example, artificial intelligence and formal verification (see [21] for a survey). Much progress has been made in the theory and practice of solving [3, 18], partly relying on generalizations of techniques from SAT, but partly being enabled by new genuine QBF techniques. However, some aspects of QBFs are hardly explored yet. One of these is counting the number of solutions of a formula. The problem of counting the number of solutions of a given QBF is also known as #QBF [13]. In contrast to #SAT [9], the counting problem of propositional logic, which is used in many application domains including probabilistic reasoning [7, 19], verification of neural networks [1, 16] and the analysis of software vulnerability [5, 24], #QBF has mainly been studied theoretically [13, 10, 2].

Only recently, some work has been presented dealing with counting the solutions of a QBF at the outer-level [22]. Given a true QBF with first quantifier block $\exists X$, this work is concerned with the problem of counting the number of assignments to the variables in X leading to a QBF that is true. The authors also consider the dual problem, i.e., given a false QBF starting with quantifier block $\forall Y$, how many assignments of the variables Y result in a false QBF? In order to answer these questions, they presented an enumerative approach that

directly resulted from enumerative propositional model counting [6, 8]. Therefore, a solver is used to obtain one solution, which is then excluded from the formula to obtain a different solution. This process is repeated until no further solution is found.

In this work, we go one step further by dealing with solutions of true QBFs that start with quantifier prefix $\forall X \exists Y$ and false QBFs that start with quantifier prefix $\exists X \forall Y$, i.e., in the first case, we count models and in the second case, we count counter-models. For both cases, the solutions are not simply propositional assignments as in outer-level counting, but tree models and counter-models capturing the dependencies between the variables of the first and second quantifier block. While tree (counter-)models are a very convenient way to describe QBF solutions, in practice, QBF solvers represent solutions by Boolean functions. We will use both views on QBF solutions to investigate to what extend enumeration-based counting can be lifted to the second level. Therefore, solutions are excluded by conjunctively/disjunctively adding (negated) functions to the formula until no further solutions are found. It turns out, that enumeration-based solution counting at the second level does not lead to the full model count, but still gives valuable insights about the solutions of a formula. We implemented the approach in a first prototype to evaluate how it works in practice. To this end, we build on state-of-the-art QBF solving technology like incremental solving and function extraction.

This paper is structured as follows. First, we introduce the necessary preliminaries in the next section. Then we present the level-2 counting problem by an example in Section 3 before we discuss enumeration-based solution counting in Section 4. In Section 5, evaluate our implementation on three different types of benchmarks, before we conclude in Section 6.

2 Preliminaries

We consider quantified Boolean formulas of the form $\Pi.\phi$, where $\Pi = Q_1 X_1 \dots Q_n$ is called *quantifier prefix* (with $Q_i \in \{\forall, \exists\}, Q_i \neq Q_{i+1}$ for $i \in \{1, ..., n-1\}$ and $X_1, \dots, X_n$ are pairwise disjoint, non-empty sets of Boolean variables). The *matrix* ϕ is a propositional formula over variables X_i with standard Boolean connectives $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$ and $\oplus$ (XOR). The prefix Π induces an ordering on the variables: $x_i <_\Pi x_j$ if $x_i \in X_i$, $x_j \in X_j$ and $i < j$. If prefix Π is clear from the context, we just write $x_i < x_j$. In this paper, we consider only *closed* formulas, i.e., every variable that occurs in matrix ϕ also occurs in the prefix Π.

A QBF $\Pi.\phi$ is in *prenex conjunctive normal form* (PCNF), if ϕ is a conjunction of clauses. A *clause* is a disjunction of literals and a *literal* is a variable or a negated variable. If l is a literal, then $var(l) = x$ if $l = x$ or $l = \neg x$. As usual, $\bar{l} = x$ if $l = \neg x$ and $\bar{l} = \neg x$ otherwise. For a QBF $\varphi = Q_1 X_1 \dots Q_n X_n.\phi$, $var(\varphi) = X_1 \cup \dots \cup X_n$. An *assignment* σ of φ is a set of literals over (a subset of) $var(\varphi)$ such that there is no $l \in \sigma$ with $\bar{l} \in \sigma$. For an assignment σ, $var(\sigma) = \{var(l) \mid l \in \sigma\}$. If $var(\sigma) = var(\varphi)$, then σ is a *full* assignment, else it is a *partial* assignment. A (partial) X-assignment is an assignment over a (sub-)set of variables X.

Given a propositional formula ϕ and an assignment σ, then ϕ_σ denotes the formula obtained when setting all variables $x \in var(\sigma)$ to true if $x \in \sigma$ and to false if $\neg x \in \sigma$, respectively. Based on this notation, the semantics of a QBF is defined as follows: $\forall x \Pi.\phi$ is true iff $\Pi.\phi_{\{x\}}$ and $\Pi.\phi_{\{\neg x\}}$ are true. A QBF $\exists x \Pi.\phi$ is true iff $\Pi.\phi_{\{x\}}$ or $\Pi.\phi_{\{\neg x\}}$ is true. For example, the QBF $\forall x \exists y.(x \leftrightarrow y)$ is true, while the QBF $\exists y \forall x.(x \leftrightarrow y)$ is false. A *model* for a QBF $\varphi = \Pi.\phi$ with $|var(\varphi)| = m$ is a tree of height $m+1$ such that every node at level $k \in \{1, \dots, m\}$ is labeled with a variable x_k in the order of the prefix, i.e., if variable x_j is at level j and variable x_k is at level k with $j < k$ then $x_j \leq_\Pi x_k$. A node at level k has one child if $x_k \in X_i$ and $Q_i = \exists$, and two children if $Q_i = \forall$. In the graphical representation, a dashed edge indicates that the parent variable is set to false and a solid edge indicates that it is set to true. Examples are shown in Figure 1. The path from the root to the leaves gives a full assignment under which ϕ evaluates to true.

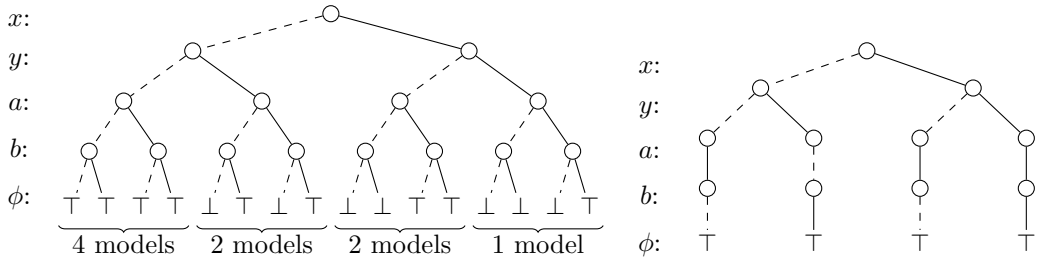

Figure 1 The left tree shows the full assignment tree for the given QBF. Tree models (16 in total) are subtrees of the assignment tree such that universal nodes have two children, existential nodes have one child and all leaves are $\top$. An example of a tree model is shown on the right.

A counter-model is defined dually, with the difference that universal nodes have one child node, existential nodes have two child nodes, and the matrix evaluates to false under the respective assignments. An alternative representation of tree models that is practically more relevant are sets of Skolem functions, so-called Skolem sets. A *Skolem set* F of a true QBF Φ is a set of Boolean functions such that for each each existential variable y of Φ, there is a function $f_y(x_1, \ldots, x_n) \in F$ where $x_1, \ldots, x_n$ are the universal variables of Φ with $x_i < y$. The elements of a Skolem set are called Skolem functions. If we take the matrix of Φ and replace all the existential variables by their Skolem functions, we obtain a valid propositional formula over the universal variables of Φ. A Skolem set for the true QBF $\forall x \exists y.(x \leftrightarrow y)$ is $\{f_y = x\}$. If we replace y by f_y, we obtain the valid formula $(x \leftrightarrow x)$. If σ_X is an assignment to a set of universal variables X, then $f_y(\sigma_X)$ denotes the value of y w.r.t. the assignment of the variables in X. For false QBFs and their tree counter-models, functions for the universal variables are defined dually. Such functions are called *Herbrand functions* and are collected in Herbrand sets. If we replace all universal variables by their Herbrand functions of a Herbrand set, we obtain an unsatisfiable formula.

3 Counting at Level Two

In the following, we introduce the QBF level-2 counting problem by an example. Consider the true QBF

$$\Phi = \forall x, y \exists a, b.(\neg x \vee a) \wedge (\neg y \vee b).$$

On the left of Figure 1 we see the full assignment tree of $\varphi = \Pi.\phi$. Each complete path from the root to a leaf node represents a full assignment to the variables in φ. A tree model of φ is a subtree of this assignment tree such that nodes x and y have two children, nodes a and b have one child each, and the leaves are labeled by $\top$. An example of a tree model is shown on the right of Figure 1. We are concerned with the question of counting the number of tree models of a given true QBF. Alternatively, we could also ask the question how many different Skolem sets a true QBF has. Here it is important that syntactically different, but semantically equivalent Skolem sets are not counted multiple times. For example the Skolem functions $f_a = \top$ and $f_a = (x \vee \neg x)$ of our example are semantically equivalent even though they are syntactically different.

A direct approach to count tree models is to iterate over all assignments σ of the universal variables in the first quantifier block of a true QBF $\varphi = \Pi.\phi$. With a propositional model counter we can determine the number of satisfying assignments of ϕ_σ. If S is the set of all assignments of the universal variables in the first quantifier block, then the total number

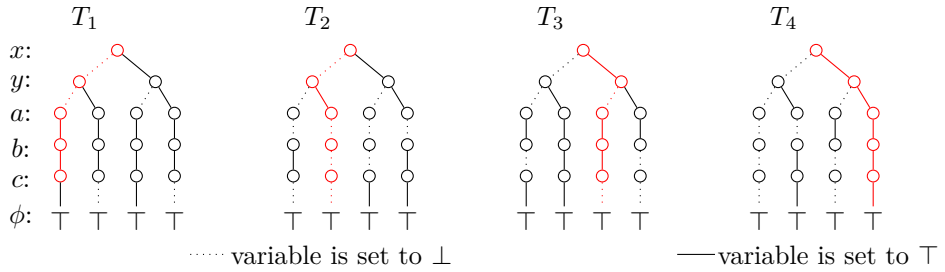

Figure 2 Tree models of $\forall x \forall y \exists a \exists b \exists c.(\neg x \vee a \vee c) \wedge (\neg y \vee b \vee \neg c)$.

of tree models is $\Pi_{\sigma \in S} \# SAT(\phi_\sigma)$. Counting counter-models is defined dually, with the difference that the role of existential and universal quantifiers is exchanged, i.e., the formulas need to have a prefix starting with $\exists X \forall Y$. While this direct approach results in a complete solution counter, it is very inefficient to iterate over all assignments of the variables in the outermost quantifier block and solve a propositional counting problem for each assignment. In the following section, we therefore investigate if an enumeration-based approach relying on recent QBF solving technology is possible.

4 Enumerative Model Counting for 2QBF

In this section, we formalize enumeration-based solution counting for the second level. For technical simplicity, we focus on true 2QBFs (formulas with quantifier prefix $\forall X \exists Y$). To this end, we lift the idea of blocking clauses to blocking functions. Recall that a blocking clause in SAT is the negation of a model σ of a formula ϕ. If ϕ is enriched with $\neg\sigma$, then σ is excluded from the solution space. For QBFs, we introduce the notion of blocking Skolem set as follows.

▶ **Definition 1.** *Let $\Phi = \forall X \exists Y.\phi$ be a true QBF and let F be a Skolem set of Φ. Then $\neg\phi_F$ is a* blocking Skolem set *of Φ where $\phi_F = \bigwedge_{f_y \in F}(y \leftrightarrow f_y)$.*

In the following example, we now use blocking Skolem sets to exclude solutions.

▶ **Example 2.** Consider the true QBF $\Phi = \forall x \forall y \exists a \exists b \exists c.(\neg x \vee a \vee c) \wedge (\neg y \vee b \vee \neg c)$. We now incrementally add blocking Skolem sets $\neg\psi_{F_i}$ until the formula becomes false. The resulting models are shown in Figure 2.

1. One solution of this formula is tree T_1, which is represented by Skolem set $F_1 = \{f_a(x,y) = \top, f_b(x,y) = \top, f_c(x,y) = \neg y\}$. To exclude F_1, we add $\neg\psi_{F_1}$ to Φ and obtain $\Phi_1 = \forall x \forall y \exists a \exists b \exists c.(\neg x \vee a \vee c) \wedge (\neg y \vee b \vee \neg c) \wedge \neg\psi_{F_1}$. Now T_1 is no solution of Φ_1.
2. A solution of Φ_1 is T_2 with Skolem set $F_2 = \{f_a(x,y) = \bot, f_b(x,y) = (x \leftrightarrow y), f_c(x,y) = x\}$. To exclude F_2 as well, we get $\Phi_2 = \forall x \forall y \exists a \exists b \exists c.(\neg x \vee a \vee c) \wedge (\neg y \vee b \vee \neg c) \wedge \neg\psi_{F_1} \wedge \neg\psi_{F_2}$.
3. Next, we get tree model T_3 with Skolem set $F_3 = \{f_a(x,y) = \top, f_b(x,y) = (x \oplus y), f_c(x,y) = \neg x\}$. We exclude F_3 from Φ_2 as before and obtain Φ_3.
4. The next tree model we find is T_4 with Skolem set $F_4 = \{f_a(x,y) = x, f_b(x,y) = y, f_c(x,y) = y\}$.
5. Finally, the QBF $\Phi_4 = \forall x \forall y \exists a \exists b \exists c.\phi \wedge \neg\psi_{F_1} \wedge \neg\psi_{F_2} \wedge \neg\psi_{F_3} \wedge \neg\psi_{F_4}$ is false.

We found four different tree models, each with four branches. Hereby Tseitin transformation is used to add the blocking function to the PCNF formula before each solver call. From these four models, we can assemble more models by combining the red branches of

Figure 2. To calculate the full model count, all possible combinations of the paths need to be considered, resulting in 4^4 models in total (there are four choices for the branch $\{\bar{x}, \bar{y}\}$, four choices for the branch $\{x, \bar{y}\}$ and so on). When we take a closer look, however, the count is not correct, as also $\{\bar{x}, \bar{y}, a, b, \bar{c}\}$ is a model of the matrix of Φ, but it has not been considered.

As the example above shows, blocking Skolem sets can indeed be used to exclude solutions, but they are sometimes too restrictive. To describe what is happening when adding blocking Skolem sets to a formula, we need to introduce the following definitions.

▶ **Definition 3.** *Let $\Phi = \forall X \exists Y.\phi$ be a true QBF and let T_1 and T_2 be tree models of Φ. Then T_1 and T_2 are* disjoint *if there are no complete paths σ_1 of T_1 and σ_2 of T_2 with $\sigma_1 = \sigma_2$.*

A complete path describes a full assignment (i.e., an assignment of all variables), such that the according branch of the tree model evaluates to true. The trees of Figure 2 are disjoint, because all their paths are different. If we want to characterize the models of a QBF in terms of Skolem sets, we get the following definition.

▶ **Definition 4.** *Let $\Phi = \forall X \exists Y.\phi$ be a true QBF and let F and G be Skolem sets of Φ. Then F and G are called* disjoint *if for each full assignment σ_X of the universal variables X there exists an existential variable $y \in Y$ such that $f_y(\sigma_X) \neq g_y(\sigma_X)$ with $f_y \in F$ and $g_y \in G$.*

The notion of disjoint models is rather strong as it forbids to have to have common paths in the tree model representation. We therefore also introduce the notion of *different* models requiring that at least one path of two tree models is different. This notion is transferred to Skolem sets as follows.

▶ **Definition 5.** *Let $\Phi = \forall X \exists Y.\phi$ be a true QBF. Furthermore, let F and G be Skolem sets of Φ. Then F and G are called* different *if there exists a full assignment σ_X of the universal variables X such that there is an existential variable $y \in Y$ with $f_y(\sigma_X) \neq g_y(\sigma_X)$, $f_y \in F$, and $g_y \in G$.*

Obviously, any two disjoint models are different. The other direction does not hold. To count all models of a QBF, we need to count the different models. As the following lemma shows, using blocking Skolem sets excludes disjoint models only.

▶ **Lemma 6.** *Let $\Phi = \forall X \exists Y.\phi$ be a QBF and F be a Skolem set of Φ. If G is a Skolem set of $\Phi' = \forall X \exists Y.\phi \wedge \neg\phi_F$, where $\neg\phi_F$ is a blocking Skolem set as defined above, then F and G are disjoint.*

Proof. Assume that F and G are not disjoint. Then there is an assignment σ_X such that for all $y \in Y$ it holds that $f_y(\sigma_X) = g_y(\sigma_X)$ for $f_y \in F, g_y \in G$. Now we extend the assignment σ_X to an assignment over $X \cup Y$ as follows: $\sigma = \sigma_X \cup \{y \mid f_y(\sigma_X) = \top, f_y \in F\} \cup \{\neg y \mid f_y(\sigma_X) = \bot, f_y \in F\} = \sigma_X \cup \{y \mid g_y(\sigma_X) = \top, g_y \in G\} \cup \{\neg y \mid g_y(\sigma_X) = \bot, g_y \in G\}$. As G is a Skolem set of Φ', σ has to satisfy $\neg\phi_F$. But by its construction, σ also satisfies ϕ_F, leading to a contradiction. Hence, F and G have to be disjoint. ◀

As the following lemma shows, we can use the enumerative approach to calculate the maximum number of pairwise disjoint Skolem sets.

▶ **Proposition 7.** *Let $\Phi = \forall X \exists Y.\phi$ be a true QBF and let $F_1, \ldots, F_m$ be pairwise disjoint Skolem sets of Φ such that*

$$\Phi' = \forall X \exists Y.\phi' = \forall X \exists Y.(\phi \wedge \neg\phi_{F_1} \wedge \ldots \wedge \neg\phi_{F_m})$$

is false. Then m is the maximum number of pairwise disjoint Skolem sets.

Proof. Since Φ' is false, there is at least one assignment σ_X such that ϕ' is unsatisfiable under σ_X. Assume there is a Skolem set F_{m+1} that is pairwise disjoint with $F_1, \ldots, F_m$. Let $\sigma = \sigma_X \cup \{x \mid f_x(\sigma_X) = \top, f_x \in F\} \cup \{\neg x \mid f_x(\sigma_X) = \bot, f_x \in F\}$. Since F_{m+1} is a Skolem set of Φ, ϕ is satisfied by σ. Further, σ satisfies $\neg\phi_{F_i}$ with $1 \leq i \leq m$, because F_i and F_{m+1} are disjoint. Hence, ϕ' is satisfied by σ. This contradicts the assumption that ϕ' is unsatisfiable under σ_X. ◀

The maximum number of pairwise disjoint models is determined by the assignments of the variables in the outermost universal quantifier block that, when the universal variables are assigned accordingly, lead to the fewest propositional models.

▶ **Lemma 8.** *Let $\Phi = \forall X \exists Y.\phi$ be a true QBF and let S be the set of all full assignments of universal variables X. Then the maximum number of pairwise disjoint models of Φ is $min(\{\#SAT(\phi_\sigma) \mid \sigma \in S\})$.*

The proof of the lemma above follows directly from the construction of disjoint models. For example, the QBF $\forall x, y \exists a, b.(\neg x \vee a) \wedge (\neg y \vee b)$ with the assignment tree shown in Figure 1 has only one disjoint model, because for the assignment $\sigma = \{x, y\}$, there is only one assignment to $\{a, b\}$ that satisfies the matrix under σ. In contrast, the QBF $\forall x \forall y \exists a \exists b \exists c.(\neg x \vee a \vee c) \wedge (\neg y \vee b \vee \neg c)$ from Example 2 has four disjoint models, because of the assignment $\{x, y\}$. In practical encodings, those assignments of the universal variables determine the number of disjoint models for which the assignment of the outermost variables not immediately falsifies the formula. The number of disjoint models gives us a lower bound for the full model count.

▶ **Proposition 9.** *Given a true QBF $\Phi = \forall X \exists Y.\phi$ with $|X| = n$ and m pairwise disjoint tree models. Then Φ has at least m^{2^n}* different *tree models.*

If we know that a true QBF $\Phi = \forall X \exists Y.\phi$ with $|X| = n$ has m pairwise disjoint models, we can calculate the full model count as follows. Let $\Phi' = \forall X \exists Y.\phi'$ with $\phi' = \phi \wedge \neg\phi_{F_1} \wedge \ldots \wedge \neg\phi_{F_m}$ be the false QBF obtained by enriching Φ with the m blocking Skolem sets. Furthermore, let S be the set of assignments of variables X such that ϕ'_σ is true $\forall \sigma \in S$. Then the full model count is

$$\prod_{\sigma \in S} \#SAT(\phi'_\sigma) * m^{2^n - |S|}$$

While the enumerative approach also works for true QBFs with prefix $\forall X \exists Y \forall Z \Pi.\phi$ for counting disjoint partial Skolem sets that consider only functions of the set Y, it is not possible to obtain a full model count of partial Skolem sets in the Y variables by using a propositional model counter. Here, the approach needs to be applied recursively.

The enumerative approach for counting disjoint models also directly transfers to an enumerative approach for counting disjoint counter-models of false QBFs starting with the prefix $\exists X \forall Y \exists Z$.[1]

▶ **Proposition 10.** *Let $\Phi = \exists X \forall Y \exists Z.\phi$ be a false QBF and let $H_1, \ldots, H_m$ be disjoint Herbrand sets such that*

$$\Phi' = \forall X \exists Y.(\phi \vee \phi_{H_1} \vee \ldots \vee \phi_{H_m})$$

is true with $\phi_H = \bigwedge_{h_y \in H}(y \leftrightarrow h_y)$. Then m is the maximum number of pairwise disjoint Herbrand sets over the variables from Y.

[1] Note that we include the last quantifier block to deal with formulas in PCNF. Otherwise, the universal variables could be immediately removed.

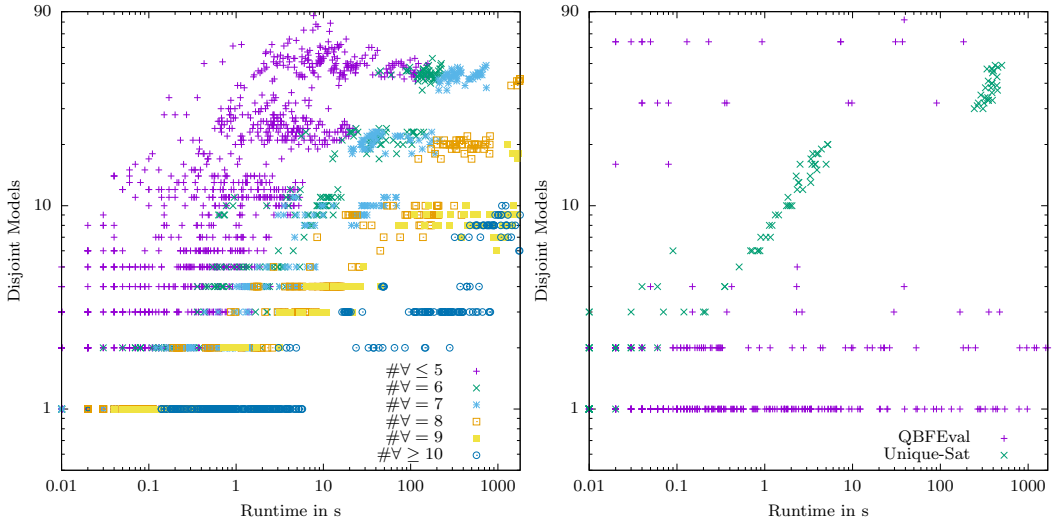

Figure 3 Runtime related to number of disjoint models for randomly generated formulas (left) and Unique-SAT/QBFEval benchmarks (right).

5 Evaluation

We implemented the previously described approach in the tool qCounter.[2] As backend solver our tool relies on the QBF solver DepQBF 6.03 [15] which is able to produce Q-resolution proofs for true and false formulas from which Skolem and Herbrand functions can be extracted. We used the QBF certification framework QBFCert [17] to obtain the functions in the Aiger format.[3] We implemented a small tool to convert the blocking functions of the variables from the second quantifier block. Therefore, the Skolem functions needed to be negated and appended conjunctively to the current QBF by using the incremental interface of DepQBF. To disjunctively add the Herbrand functions, an extra Tseitin transformation step is necessary to obtain a formula in PCNF. While this transformation introduces auxiliary variables we can avoid an exponential increase of the formula using this technique[23, 12]. For this we also used the incremental interface of DepQBF with its ability to add temporary clauses. To calculate the full model count as described above, we employed the propositional model counter Ganak [20] together with the SAT solver Lingeling [4].

Currently, there exist no standard benchmark sets which we could use to evaluate the correctness and performance of our implementation. To this end, we propose three different benchmark sets: (1) randomly generated formulas, (2) QBF encodings of the unique-SAT problem (does a given propositional formula have exactly one solution?) and (3) benchmarks from past QBF Evaluations with a suitable quantifier structure. Whereas the benchmarks from (1) and (2) are constructed in such a way that the number of models is known, this is not the case for the benchmarks from (3). Details follow below. All experiments were performed on a cluster of dual-socket AMD EPYC 7313 @ 16 × 3.7GHz machines with 4GB memory limit and 1800 seconds as timeout.

[2] The tool, the benchmarks and the log-files are publicly available at `https://qcount.pages.sai.jku.at/l2count`

[3] `http://fmv.jku.at/aiger/`

5.1 Randomly Generated Formulas

We provide a generator that returns true 2-QBFs in PCNF with quantifier structure $\forall X \exists Y$ such that the number of (disjoint) models is known. The parameters n (size of X), m (size of Y) have to be provided. For producing the clauses of the matrix, the generator iterates over all possible assignments of the universal variables X and excludes propositional models by randomly assigning the existential variables Y and building blocking clauses. For example, for a prefix $\forall x_1, x_2 \exists y_1, y_1$ the clause $(x_1 \vee \bar{x}_2 \vee y_1 \vee y_2)$ prohibits in any QBF model that in the branch where x_1 is set to false and x_2 is set to true, both y_1 and y_2 are both set to false. The number of disjoint models is determined by the branch with the smallest number of propositional models (see also Lemma 8).

We generated 3950 formulas with $2 \leq |X|, |Y| \leq 11$ having up to 100 disjoint models each. For 2608 of these formulas, the number of disjoint models could be determined. The results are shown in the left plot of Figure 3. We cluster the results according to the size of X, indicating that not only the number of disjoint models impacts the runtime, but also the size of the first quantifier block.

5.2 Unique-SAT Encodings

The question whether a given propositional formula ϕ has exactly one model can be encoded by a QBF $\exists X \forall . \psi$ as shown in [11]. In this QBF, X contains the variables of ϕ, Y is a copy of the variables of ϕ, and ψ is constructed in such a way that the QBF is true iff ϕ has exactly one model. The prefix fits for counting the counter-models at level-2 in case the QBF is false. The number of QBF counter-models can be determined based on the number of the models of ϕ. If ϕ is unsatisfiable and has n variables, then the number of disjoint counter-models equals the number of different counter-models and is 2^{2^n}. If ϕ has m models, then it has $(m-1)$ disjoint models and $(2^n)^{2^n - m}(m-1)^m$ different models.

We generated 497 Unique-SAT formulas based on true propositional formulas with 26 to 240 variables and 61 and 643 clauses by using the random generator from [14]. We considered only propositional formulas with more than one model in order to count disjoint counter-models of the resulting false QBFs. The performance of our tool is shown in the right plot of Figure 3. The number of disjoint counter-models enumerated by our tool could be quickly validated with a propositional model counter. In this way, we could check the results of our tool for false formulas.

5.3 Benchmarks from QBF Evaluations

As a final set of benchmarks, we considered formulas from the main tracks of QBFEval 2022 and QBFEval 2008.[4] From the QBFEval 2022 set, we identified 18 true formulas and 102 false formulas with a suitable prefix structure that could be solved by plain DepQBF within a time limit of 1800 seconds. In a similar way, we selected two true formulas and 683 false formulas from the QBFEval 2008 set. For the 2022 formulas, we could determine the number of disjoint models of 3 formulas and the number of disjoint counter-models of 53 formulas. For the 2008 formulas, we could determine the number of disjoint models for both formulas and the number of disjoint counter-models of 285 formulas. Interestingly, many of the formulas have only one disjoint model indicating that the search space is strongly

[4] `http://www.qbflib.org`

restricted for certain assignments of the variables in the outermost quantifier block. On the other hand, for some of the formulas we could find up to 82 disjoint (counter-)models. Details are shown in the right plot of Figure 3.

6 Conclusion

We considered the problem of counting level-2 (counter-)models for QBFs. It turned out that enumerating Skolem/Herbrand functions does not provide the full count of solutions as it is the case for propositional model counting and counting QBF models at the outer level. We characterized the subset of solutions which can be counted in an enumerative manner. This subset often connects to interesting solutions of a QBF encoding. We also provided the first practical implementation of an enumerative method for level-2 solution counting. Our approach cannot only be used for counting, but also for explicitly enumerating solutions which might also have some applications in better understanding and debugging QBF encodings. We provided several sets of benchmarks of true and false instances to evaluate our implementation.

We consider this work as an important first step to full solution counting, i.e., for QBFs with an arbitrary number of quantifier blocks. While a recursive application of our approach seems possible, it has to be expected that this approach will be inefficient. Hence, alternative ways need to be explored like a tight integration of solving and counting or exploiting approaches that process the prefix in a reverse order as it is for example done in certain preprocessing techniques.

References

1 Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S. Meel, and Prateek Saxena. Quantitative verification of neural networks and its security applications. In *Proc. of the 2019 ACM SIGSAC Conf. on Computer and Communications Security*, pages 1249–1264. ACM, 2019.

2 Michael Bauland, Elmar Böhler, Nadia Creignou, Steffen Reith, Henning Schnoor, and Heribert Vollmer. Quantified constraints: The complexity of decision and counting for bounded alternation. *Electron. Colloquium Comput. Complex.*, TR05-024, 2005.

3 Olaf Beyersdorff, Janota Mikolás, Florian Lonsing, and Martina Seidl. Quantified Boolean Formulas. In *Handbook of Satisfiability*, volume 336, pages 1177–1221. IOS Press, 2021.

4 Armin Biere. CaDiCaL, Lingeling, Plingeling, Treengeling and YalSAT Entering the SAT Competition 2018. In *Proc. of SAT Competition 2018 – Solver and Benchmark Descriptions*, volume B-2018-1 of *Department of Computer Science Series of Publications B*, pages 13–14. University of Helsinki, 2018.

5 Fabrizio Biondi, Michael A. Enescu, Annelie Heuser, Axel Legay, Kuldeep S. Meel, and Jean Quilbeuf. Scalable approximation of quantitative information flow in programs. In *Proc. of Int. Conf. on Verification, Model Checking, and Abstract Interpretation*, volume 10747 of *LNCS*, pages 71–93. Springer, 2018.

6 Elazar Birnbaum and Eliezer L. Lozinskii. The good old Davis-Putnam procedure helps counting models. *J. Artif. Intell. Res.*, 10:457–477, 1999.

7 Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic SAT calls. In *Proc. of Int. Joint Conf. on Artificial Intelligence*, pages 3569–3576. IJCAI/AAAI Press, 2016.

8 Olivier Dubois. Counting the number of solutions for instances of satisfiability. *Theor. Comput. Sci.*, 81(1):49–64, 1991.

9 Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Model counting. In *Handbook of Satisfiability*, pages 993–1014. IOS Press, 2021.

10 Lane A. Hemaspaandra and Heribert Vollmer. The satanic notations: counting classes beyond #P and other definitional adventures. *SIGACT News*, 26(1):2–13, 1995.

11 Hans Kleine Büning and Theodor Lettmann. *Propositional logic - deduction and algorithms*. Cambridge University Press, 1999.

12 Elias Kuiter, Sebastian Krieter, Chico Sundermann, Thomas Thüm, and Gunter Saake. Tseitin or Not Tseitin? The Impact of CNF Transformations on Feature-Model Analyses. In *Proc. of the 37th IEEE/ACM Int. Conf. on Automated Software Engineering*. ACM, 2023.

13 Richard E. Ladner. Polynomial space counting problems. *SIAM J. Comput.*, 18(6):1087–1097, 1989.

14 Massimo Lauria, Jan Elffers, Jakob Nordström, and Marc Vinyals. CNFgen: A Generator of Crafted Benchmarks. In *Proc. of the 20th Int. Conf. on Theory and Applications of Satisfiability Testing*, volume 10491 of *LNCS*, pages 464–473. Springer, 2017.

15 Florian Lonsing and Uwe Egly. DepQBF 6.0: A search-based QBF solver beyond traditional QCDCL. In *Proc. of the 26th Conf. on Automated Deduction*, volume 10395 of *LNCS*, pages 371–384. Springer, 2017.

16 Nina Narodytska, Aditya A. Shrotri, Kuldeep S. Meel, Alexey Ignatiev, and João Marques-Silva. Assessing heuristic machine learning explanations with model counting. In *Proc. of the Int. Conf. on Theory and Applications of Satisfiability Testing*, volume 11628 of *LNCS*, pages 267–278. Springer, 2019.

17 Aina Niemetz, Mathias Preiner, Martina Seidl, and Armin Biere. Resolution-based certificate extraction for QBF - (tool presentation). In *Proc. of the 15th Int. Conference on Theory and Applications of Satisfiability Testing*, volume 7317 of *LNCS*, pages 430–435. Springer, 2012.

18 Luca Pulina and Martina Seidl. The 2016 and 2017 QBF solvers evaluations (QBFEVAL'16 and QBFEVAL'17). *Artif. Intell.*, 274:224–248, 2019.

19 Tian Sang, Paul Beame, and Henry A. Kautz. Performing Bayesian inference by weighted model counting. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence*, pages 475–482. AAAI Press / The MIT Press, 2005.

20 Shubham Sharma, Subhajit Roy, Mate Soos, and Kuldeep S. Meel. Ganak: A scalable probabilistic exact model counter. In *Proc.of Int. Joint Conf. on Artificial Intelligence*, pages 1169–1176. Int. Joint Conf. on Artificial Intelligence Organization, 2019.

21 Ankit Shukla, Armin Biere, Luca Pulina, and Martina Seidl. A survey on applications of quantified Boolean formulas. In *Proc. of the Int. Conf. on Tools with Artificial Intelligence*, pages 78–84. IEEE, 2019.

22 Ankit Shukla, Sibylle Möhle, Manuel Kauers, and Martina Seidl. Outercount: A first-level solution-counter for quantified boolean formulas. In *Proc. of the 15th Int. Conf on Intelligent Computer Mathematics*, volume 13467 of *LNCS*, pages 272–284. Springer, 2022.

23 G. S. Tseitin. *On the Complexity of Derivation in Propositional Calculus*, pages 466–483. Springer, 1983.

24 Ziqiao Zhou, Zhiyun Qian, Michael K. Reiter, and Yinqian Zhang. Static evaluation of noninterference using approximate model counting. In *Proc. of IEEE Symposium on Security and Privacy*, pages 514–528. IEEE Computer Society, 2018.