Model Counting Using the Inclusion-Exclusion Principle

Huxley Bennett and Sriram Sankaranarayanan ⋆

University of Colorado, Boulder, CO. first.lastname@colorado.edu

The inclusion-exclusion principle is a well-known mathematical principle used to count the number of elements in the union of a collection of sets in terms of intersections of sub-collections. We present an algorithm for counting the number of solutions of a given k-SAT formula using the inclusion-exclusion principle. The key contribution of our work consists of a novel subsumption pruning technique. Subsumption pruning exploits the alternating structure of the terms involved in the inclusion-exclusion principle to discover term cancellations that can account for the individual contributions of a large number of terms in a single step.

The Inclusion-Exclusion Principle and #SAT

Given sets $A_1, \ldots, A_m, m > 0$, the inclusion-exclusion principle states that $|\bigcup_{i=1}^{m} A_i| = \sum_{i=1}^{m} |A_i| - \sum_{1 \leq i < j \leq m} |A_i \cap A_j| + \cdots (-1)^{m+1} |A_1 \cap A_2 \cap \cdots \cap A_m|$. It is well-known that this principle can be applied to count the number of solutions of a given k-CNF formula [4, 3].

Let φ be a k-CNF formula consisting of variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$. Each clause $\mathsf{lits}(C_i) :\ \{\ell_{(1,k)}, \ldots, \ell_{(i,k)}\}$ is a set of k literals, each literal of the form $\ell_i :\ x_j$ or $\ell_i : \neg x_j$. We will count the number $N_U :\ \#\text{UNSAT}(\varphi)$ of solutions that *do not satisfy* φ using the inclusion-exclusion principle. Let $A_1, \ldots, A_m$ denote the sets of variable assignments which *dis-satisfy* the clauses $C_1, \ldots, C_m$, respectively, in φ. Therefore, $N_U = |\bigcup_{i=1}^{m} A_i|$ can be calculated using the inclusion-exclusion principle, as a summation ranging over all subsets of clauses $S \subseteq \{C_1, \ldots, C_m\}$:

$$N_U = \sum_{S \subseteq \{C_1, \ldots, C_m\}} t(S) \quad \text{where} \quad t(S) = \begin{cases} 0 & \text{if } \exists\, j,\ \{x_j, \neg x_j\} \subseteq \mathsf{lits}(S) \\ \left((-1)^{|S|+1} \cdot 2^{n - |\mathsf{lits}(S)|}\right) & \text{otherwise} \end{cases},$$

where $\mathsf{lits}(S)$ represents all the literals appearing in the clauses of S. Given N_U, we may obtain the number of satisfying solutions as $2^n - N_U$. Note that the number of terms involved in the summation is exponential in the formula size.

One solution to improving the complexity of this procedure is to prune away terms involving subsets S where $N(S) = 0$ in the summation above. This is achieved by avoiding subsets S which include *interfering clauses* C_j, C_k that contain a variable x_i and its negation $\neg x_i$. Such an optimization has been proposed elsewhere [3, 4]. In this work, we present yet another optimization through subsumption pruning.

⋆ This work was partially supported by the National Science Foundation (NSF) award CNS-1016994.

Tree Exploration and Subsumption Pruning

We now present a brief sketch of our subsumption pruning technique. More details are available from an extended version of this paper [1].

Our technique arranges the terms in the inclusion-exclusion formula as a tree and performs a recursive depth-first tree exploration to consider *non-interfering* clause sequences of the form $[C_{i1}; \ldots; C_{id}]$. Each node v in the search tree is defined by its current clause sequence $S : [C_{i1}; C_{i2}; \ldots; C_{id}]$, where $d \geq 1$ is the depth of the node. The node is associated with the term $t(S) = (-1)^{d+1} 2^{n - |\mathsf{lits}(S)|}$. Through the search, we maintain the invariant that S is interference free and that $1 \leq i_1 < \cdots < i_d \leq m$.

Consider a node $S : [C_{i1}; \ldots; C_{ij}]$. Let $T : [S; C_l]$ be a child of S extended by adding the clause C_l. We say that S *subsumes* T iff $\mathsf{lits}(S) = \mathsf{lits}(T)$. In other words, every literal in the clause C_l is already contained in some clause in S. The main theorem in this paper takes advantage of subsumptions to make a drastic improvement on the basic scheme given previously:

Theorem 1. *Let T_j be a subsumed child of S in the search tree. Considering any child $T_l : [S; C_l]$ of S, where $l > j$ and the corresponding child $T_l' : [T_j : C_l]$, then $t(T_l') = -t(T_l)$ and $t(\mathsf{subtree}(T_l')) = -t(\mathsf{subtree}(T_l))$. We conclude that $t(\mathsf{subtree}(S)) = \sum_{i=1}^{j-1} t(\mathsf{subtree}(T_i))$.*

This theorem, whose proof is in the extended version of the paper, concludes that if S subsumes one of its children T_j, then due to the alternating sum involved in the inclusion-exclusion principle that the children of S and T_j cancel each other out. In practice this means that we need only explore the children $T_1, \ldots, T_{j-1}$ of S, a significant improvement over evaluating all of the children of S, especially when j is small.

Preliminary experimental evaluations of our technique is reported in our extended report [1]. We present a summary of these results obtained over randomly generated k-SAT instances. (A) The application of our subsumption pruning technique provides a significant speedup (2-3x) on most of the larger instances. Nevertheless, the technique itself is limited in the size of formulae that can be handled, especially when compared to other approaches to counting using DPLL [2]. (B) An integration of our technique inside DPLL-based model counters compares favorably to existing DPLL-based model counters CDP and Relsat.

Currently, we are in the process of evaluating our approach over structured benchmarks and analyzing the expected running times for our technique over randomly generated formulae.

References

1. H. Bennett and S. Sankaranarayanan. Model counting using the inclusion-exclusion principle, 2011. Draft (available upon request).
2. C. P. Gomes, A. Sabharwal, and B. Selman. Model counting. In *Handbook of Satisfiability*, chapter 20. IOS Press, 2008.
3. K. Iwama. CNF-satisfiability test by counting and polynomial average time. *SIAM Journal on Computing*, 18(2):385–391, 1989.
4. E. L. Lozinskii. Counting propositional models. *Information Processing Letters*, 41:327–332, 1992.