Short XORs for Model Counting: From Theory to Practice

Carla P. Gomes[1,⋆], Joerg Hoffmann[2], Ashish Sabharwal[1,*], and Bart Selman[1,*]

[1] Dept. of Computer Science, Cornell University, Ithaca NY 14853-7501, U.S.A
{gomes,sabhar,selman}@cs.cornell.edu
[2] University of Innsbruck, Technikerstraße 21a, 6020 Innsbruck, Austria
joerg.hoffmann@deri.org

Abstract. A promising approach for model counting was recently introduced, which in theory requires the use of large random XOR or parity constraints to obtain near-exact counts of solutions to Boolean formulas. In practice, however, short XOR constraints are preferred as they allow better constraint propagation in SAT solvers. We narrow this gap between theory and practice by presenting experimental evidence that for structured problem domains, very short XOR constraints can lead to probabilistic variance as low as large XOR constraints, and thus provide the same correctness guarantees. We initiate an understanding of this phenomenon by relating it to structural properties of synthetic instances.

1 Introduction

The dramatic advances in Boolean satisfiability or SAT technology have led to the exploration of new possible applications of SAT solvers. Some of the most exciting such applications go beyond pure satisfiability testing. For example, they involve random sampling from the set of satisfying truth assignments and counting the total number of satisfying assignments. These techniques are particularly promising in the context of applications to probabilistic reasoning. Computationally speaking, counting and sampling are considerably harder than satisfiability testing per se. We recently introduced an approach to counting [1] and sampling [2] that relied on adding XOR or parity constraints [3] (converted to the usual CNF form) to the original problem instance. We showed how one can then use standard state-of-the-art SAT solvers running on the augmented problem instance to compute bounds on the model count of the original problem instance, and to sample near-uniformly from the solution space.

This XOR framework provides probabilistic correctness guarantees for XOR constraints of any length. However, the best results are obtained by using "full-length" XORs, i.e., XORs containing half of the variables of the formula. In our experiments with available SAT solvers, we consistently observed that reasoning with long XORs is computationally much more expensive than reasoning

⋆ Supported by IISI, Cornell University, AFOSR grant F49620-01-1-0076.

J. Marques-Silva and K.A. Sakallah (Eds.): SAT 2007, LNCS 4501, pp. 100–106, 2007.

with short XORs. This appears to be because long XOR constraints hamper unit propagation on which SAT solvers rely heavily. Therefore, for the practical applicability of our XOR techniques, a key question is whether relatively short XOR constraints can already provide much of the power of full-length XORs, when considering purely the *quality* of model count bounds and solution samples.

Fortunately, this is the case, as we demonstrate in this paper. In particular, we show that while random 1-XORs (single literals) or random 2-XORs (2 variable XORs) may lead to rather weak bounds on solution counts or sample quality due to large variance, the situation improves dramatically when one considers only slightly longer XORs. In fact, good quality bounds and samples can often be obtained with XORs of 5 to 10 variables, even when the original formula has several hundred variables. To demonstrate this, we systematically consider the *variance* — which directly determines the bound quality — of solution counts obtained in repeated runs with XORs of different lengths. We show that the variance decreases drastically beyond length 1 or 2 XORs. We first demonstrate this phenomenon on a range of practical problem instances, and then provide further insights into the trade-offs between solution space structure and the required length of XORs by considering a class of synthetic problem instances.

2 Background

An XOR *constraint* D over a set of Boolean variables V is the logical "xor" or parity of a subset of $V \cup \{1\}$; a truth assignment for V satisfies D iff it sets an *odd number* of elements in D to TRUE. The value 1 allows us to express even parity. For instance, $D = \{a, b, c, 1\}$ is TRUE as an XOR constraint when an even number of a, b, c are TRUE. Our focus will be on formulas which are a logical conjunction of a formula in Conjunctive Normal Form (CNF) and some XOR constraints. The latter are translated into CNF using additional variables.

XOR-based model counting and sampling methods [1,2] work as follows. Given a formula F, one adds (i.e., conjoins) an appropriate number s of randomly chosen XOR constraints to F to create a new formula F'. F', which in expectation can be shown to have a factor 2^s fewer satisfying assignments than F, is then fed to either an off-the-shelf complete SAT solver or to an exact model counter. When large XORs are used, one can use certain probabilistic independence conditions and transform the result into (a bound on) the model count of F or a random solution sample for F, with guarantees. The length of XORs influences the independence assumption and thus affects the quality of the process; formally, with "short" XORs, only a lower bound model count can be guaranteed.

For one of our analytic computations for the "ideal" case of large XORs, we will use random variables which are the sum of indicator random variables: $Y = \sum_\sigma Y_\sigma$, $Y_\sigma \in \{0, 1\}$. Linearity of expectation says that $\mathbb{E}[Y] = \sum_\sigma \mathbb{E}[Y_\sigma]$. When various Y_σ are *pairwise independent*, i.e., knowing Y_{σ_2} tells us nothing about Y_{σ_1}, even variance behaves linearly: $\mathrm{Var}[Y] = \sum_\sigma \mathrm{Var}[Y_\sigma]$.

3 The Setup for Empirical Evaluation

In all our experiments, we vary a parameter $k, 1 \leq k \leq n/2$, and study the *standard deviation* of the quantity $X = 2^s \times residualSolutionCount$ after a certain number s of random XORs of length k are added to a formula F. The expected value of X is known to be the true model count of F. When the probabilistic variance of X is small (specifically, when $\mathrm{Var}[X] \leq \mathbb{E}[X]$), algorithm `MBound` [1] is able to provide much more than lower bounds: it can compute *near-exact* model counts by computing both a lower bound and an upper bound. This condition was formally proved to hold when $k = n/2$. In this empirical study, we define $k \in \{1, 2, \ldots, n/2\}$ to be *(empirically) good* or *sufficient* for F when $\mathrm{Var}_s[X] \leq \mathbb{E}[X]$ even for XORs of length k, where $\mathrm{Var}_s[X]$ denotes the sample standard deviation (**s.s.d.** for short) of X obtained experimentally.

Since formulas vary in the number of variables and the number of solutions, one must normalize for this when comparing the behavior of XORs on different formulas in the same plot. We do so by two means: (1) we plot the s.s.d. of the normalized solution counts, $X' = X/trueCount$, and (2) the number of XORs we use is fewer than $\log_2 trueCount$ by a constant amount. The first condition guarantees consistent expected values, namely, $\mathbb{E}[X'] = 1$ for all formulas. The second condition, as we will see shortly, ensures that as k approaches $n/2$, s.s.d.$[X]$ approaches the same ideal value for all formulas under consideration.

The ideal curve in all our plots corresponds to the standard deviation of the normalized obtained solution count, X', when full-length XORs are used. Note that this is really a single ideal value to which all s.s.d. plots should converge as the length k increases. We plot it as a horizontal line to easily visually infer for what k are XORs of length k already good for the formula under consideration.

We compute the ideal curve analytically as follows. For concreteness, let Y denote the residual model count obtained after adding s random XORs of length $n/2$ to a formula F with 2^{s^*} solutions. In the notation above, $X = 2^s \times Y$, so that $\mathrm{Var}[X] = 2^{2s} \times \mathrm{Var}[Y]$. Following our earlier analysis [1], we can write $Y = \sum_\sigma Y_\sigma$, where the summation is over all solutions σ of F and Y_σ is a 0-1 random variable indicating whether σ is present in the residual solutions. As argued in that analysis, random variables Y_σ are pairwise independent. Also, $\mathbb{E}[Y_\sigma] = 2^{-s}$ and $\mathrm{Var}[Y_\sigma] = 2^{-s}(1-2^{-s}) \approx 2^{-s}$. Because of pairwise-independence, $\mathrm{Var}[Y] = \sum_\sigma \mathrm{Var}[Y_\sigma] = 2^{s^*-s}(1-2^{-s}) \approx 2^{s^*-s}$. It follows that $\mathrm{Var}[X] \approx 2^{s^*+s}$. For the variance of the normalized model count, we get $\mathrm{Var}[X'] = \mathrm{Var}[X]/2^{2s^*} \approx 2^{-(s^*-s)} = 2^{-remainingXors}$, where *remainingXors* is defined as $(s^* - s)$, i.e., the amount by which the number of XORs added was fewer than the number needed to get down to a single solution. The corresponding s.s.d. is s.s.d.$[X'] = \sqrt{2^{-(s^*-s)}}$, ignoring the relatively tiny $(1 - 2^{-s})$ term.

The ideal curve depicting the behavior of XORs of length $n/2$ is therefore shown as the horizontal line s.s.d.$[X'] = \sqrt{1/2^{s^*-s}}$. We note that when $s = s^*$, so that a single solution is expected to survive, s.s.d.$[X']$ becomes 1. The quantity $s^* - s$, which plays an important role in our experiments, will be referred to as the number of *remaining* XORs. As mentioned above, all formulas plotted in a

single figure for comparison will have the same number of remaining XORs, and will therefore converge to the same ideal value as the length of XORs increases.[1]

4 Experimental Results and Discussion

For each formula F on n variables that we consider, we will vary a parameter k within a sub-range of $\{1, 2, \ldots, n/2\}$ on the horizontal axis, and plot on the vertical axis the sample standard deviation of the normalized quantity $X' = (2^s \times \mathit{residualSolutionCount})/\mathit{totalSolns}$ after a certain number s of random XORs of length k are added to F. For each s.s.d. computation (i.e., for each data point in the plots to follow), we used 1,000 samples in most cases to get a reasonable estimate of the true standard deviation for that length, and up to 50,000 samples in some cases for very short XORs. The number of residual solutions was computed using the exact model counter `Relsat` [4].

We present results on formulas from four domains: Latin squares, logistics planning, circuit synthesis, and random formulas (Figs. 1-4). In each case, there is a dramatic drop in the s.s.d. as the length of XORs is increased even slightly.

The **Latin square formulas** considered have 100 to 150 variables each. The most constrained formula, `ls7R30`, has less than 2^5 solutions, while the least constrained one, `ls7R36`, has 2^{14} solutions. These formulas theoretically require XORs of length 50-75 for near-exact model counting with `MBound`. We performed experiments with 3 remaining XORs. Interestingly, we see from Fig. 1 that at lengths 6 to 8, the s.s.d. already drops to the ideal value. The **logistics planning problem** here has 352 variables and roughly 2^{19} solutions. We again see from Fig. 2 that the variance drops sharply till XOR length 25. At lengths 40 to 50, we are already very close to the ideal behavior. The **circuit synthesis formulas** are for finding minimal size circuits for a given Boolean function. We consider the instance `2bitmax_6` with 252 variables and ideal XOR length 126. This formulas has roughly 2^{97} solutions and we used 87 XORs. Fig. 3 shows that while the s.s.d. is fairly high for very short XORs, it drops dramatically as the length increases to 7, and gets very close to the ideal value at around length 10.

Our **random 3-CNF formulas** are selected from the under-constrained region where model counting is known to be computationally hard [4], i.e., with clause-to-variable ratios significantly below the satisfiability threshold of ≈ 4.26. We consider four 100 variable formulas at ratios 3.3, 3.8, 3.96, and 4.2, respectively. The number of solutions ranges from 2^{32} to 2^{14}. The plots in Fig. 4 indicate that random formulas in general show much higher variance than more structured, real-world formulas considered earlier. In particular, the ratio 4.2 formula achieves ideal behavior at XOR length more than 40. On the positive side, as these formulas become less and less constrained, shorter and shorter XORs surprisingly begin to be sufficient. E.g., the ratio 3.3 formula works well

[1] One could alternatively consider plotting various formulas while keeping the number of XORs fixed, rather than keeping the number of remaining XORs fixed. As the above calculation shows, their s.s.d. plots will then eventually converge to different values, making formula-to-formula comparison not very meaningful.

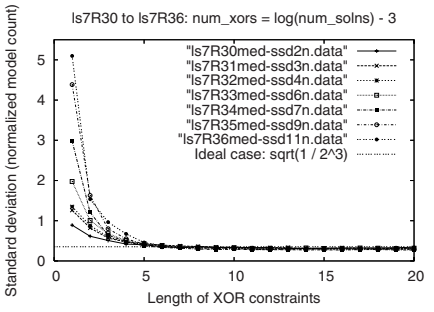

Fig. 1. Latin square formulas of order 7 (100-150 variables)

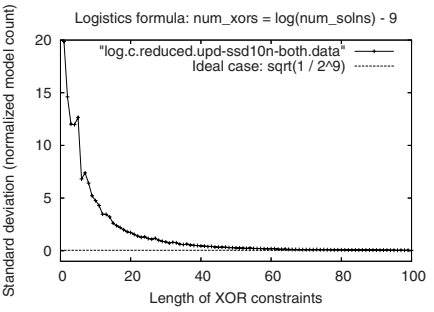

Fig. 2. A logistics planning problem (352 variables after simplification)

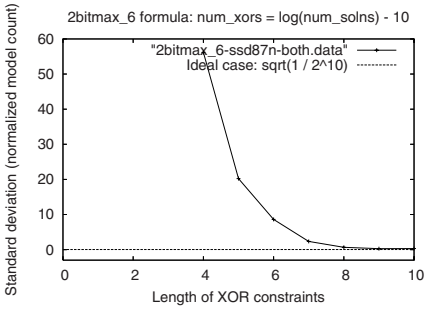

Fig. 3. A circuit synthesis problem (252 variables)

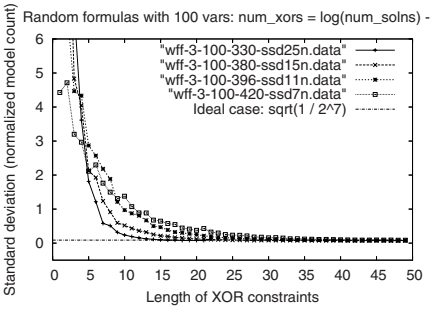

Fig. 4. Random formulas at ratios 3.3, 3.8, 3.96, and 4.2 (100 variables)

even at length 15. This trend suggests that random formulas interestingly become more suitable for shorter XORs as we go into the highly under-constrained region, which is traditionally seen as the harder region for model counting.

In order to better understand the behavior of XOR constraints of various lengths, we explore hand-crafted families of formulas which will help us relate XORs to an intrinsic structural feature of formulas, namely their *backbone*: the set of variables each of which takes the same value in every solution to the formula. We first consider very simple **fixed-backbone formulas** `string-n-t`, with n variables and backbone size $n - t$. The first t variables of the formula are completely unconstrained, while the last $n - t$ variables are fixed to 1. This formula has exactly 2^t solutions, which we will denote by: $1^{n-t} *^t$.

Fig. 5 plots the s.s.d. for these formulas with $n = 50$ variables, $t = 20, 30, 40$, or 49 unconstrained variables, and the corresponding backbone size $30, 20, 10$, or 1. We see that formulas with larger backbone size clearly require larger XORs. This is explained qualitatively by the fact that randomly chosen XORs become more likely to only involve backbone variables as the backbone size increases. When an XOR constraint only involves backbone variables, we encounter unwanted

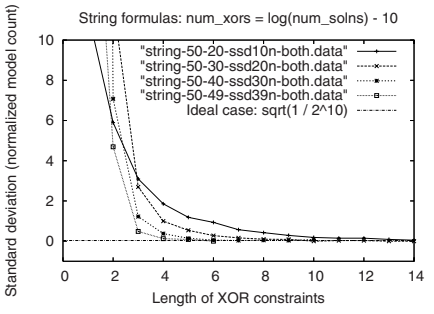

Fig. 5. Fixed-backbones formulas (50 variables)

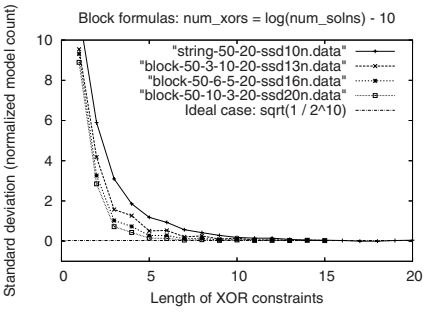

Fig. 6. Interleaved-backbones formulas (50 variables)

behavior: the constraint is either satisfied by all solutions or falsified by all solutions. While this still cuts down the solution space in half on average, there is high variance in the residual count. On the other hand, with small backbones, randomly chosen XORs are very likely to involve at least one non-backbone variable (in this case, one unconstrained variable). When this happens, some of the solutions satisfy the constraint and others don't. This still cuts down the solution space in half on average, but now with lower variance.

The **interleaved-backbones formulas** we consider next attempt to replace a large *global* backbone for all solutions into many overlapping (and conflicting) *local* backbones for solution clusters. These local backbones are interleaved together, allowing all possible combinations of their constituent "blocks," thereby giving the XORs more freedom. These formulas are `block-n-m-k-t`, constructed as follows. There are n variables divided up into m blocks of size k each, and t unconstrained variables ($n = mk + t$). Each block is constrained to have all its variables take the same value. The blocks themselves are, however, independent of each other. We can represent this formula by its solution space: $a_1^k\ a_2^k \ldots a_m^k\ *^t$, where each $a_i \in \{0, 1\}$. Recall that the formula, `string-n-t`, has exactly 2^t solutions. The number of solutions of `block-n-m-k-t` is 2^{t+m}, which increases as the number of blocks in the backbone split is increased, allowing more freedom.

Fig. 6 gives the results for 50 variable block formulas with 30 unconstrained variables and the rest split into 3, 6, and 10 blocks. We see that as the backbone is split into more and more blocks, the variance decreases. In particular, the variance is the highest when there are no blocks (the string formula at the top) and the lowest when the backbone is split into 10 blocks.

5 Concluding Remarks

XOR-streamlining is a promising approach for model counting and sampling. We provided evidence that relatively short XORs can be surprisingly powerful on practical problem instances. While large global backbones are bad for XORs, our synthetic formulas based on interleaved backbones provide intuitive explanation

that solution spaces consisting of many clusters with large local backbones are still fine. We believe that this latter structure is more likely to be present in real-world formulas than large global backbones or uniformly distributed solutions.

References

1. Gomes, C.P., Sabharwal, A., Selman, B.: Model counting: A new strategy for obtaining good bounds. In: 21th AAAI, Boston, MA (2006) 54–61
2. Gomes, C.P., Sabharwal, A., Selman, B.: Near-uniform sampling of combinatorial spaces using XOR constraints. In: 20th NIPS, Vancouver, B.C. (2006)
3. Valiant, L.G., Vazirani, V.V.: NP is as easy as detecting unique solutions. Theoretical Comput. Sci. **47**(3) (1986) 85–93
4. Bayardo Jr., R.J., Pehoushek, J.D.: Counting models using connected components. In: 17th AAAI, Austin, TX (2000) 157–162