Solving Satisfiability Modulo Counting for Symbolic and Statistical AI Integration with Provable Guarantees

Jinzhao Li, Nan Jiang, Yexiang Xue*
{li4255, jiang631, yexiang}@purdue.edu
Department of Computer Science, Purdue University

Association for the Advancement of Artificial Intelligence

Satisfiability Modulo Counting (SMC)

The SMC problem is to determine if there exists $\mathbf{x} = (x_1, x_2, \dots, x_n) \in \mathcal{X} = \{0,1\}^n$ and $\mathbf{b} = (b_1, b_2, \dots, b_k) \in \{0,1\}^k$ that satisfies the formula:

$$\phi(\mathbf{x}, \mathbf{b}) \wedge \left[b_i \Rightarrow \left(\sum_{\mathbf{y}_i \in \mathcal{Y}_i} f_i(\mathbf{x}, \mathbf{y}_i) \geq 2^{q_i} \right) \right], \forall i \in \{1, \dots, k\},$$

where each b_i is a Boolean predicate that is true if and only if the corresponding model count exceeds a threshold. Bold symbols (i.e., $\mathbf{x}, \mathbf{y}_i$ and $\mathbf{b}$) are vectors of Boolean variables. ϕ, $f_1, \dots, f_k$ are Boolean functions. $\sum f_i$ computes the number of satisfying assignments (model counts) of f_i.

Challenges:

- It is challenging to solve SMC because of their highly intractable nature (NP^{PP} -complete)– still intractable even with good satisfiability solvers and model counters
- Current exact solvers struggle with generalizing to **large-scale problems** due to their intractable nature.
- Randomized methods either cannot quantify the quality of their solutions, or they provide one-sided guarantees, or **their guarantees can be arbitrarily loose**.

Contribution:

- **We propose XOR-SMC, a polynomial algorithm with accesses to NP-oracles, to solve highly intractable SMC problems with constant approximation guarantees.**

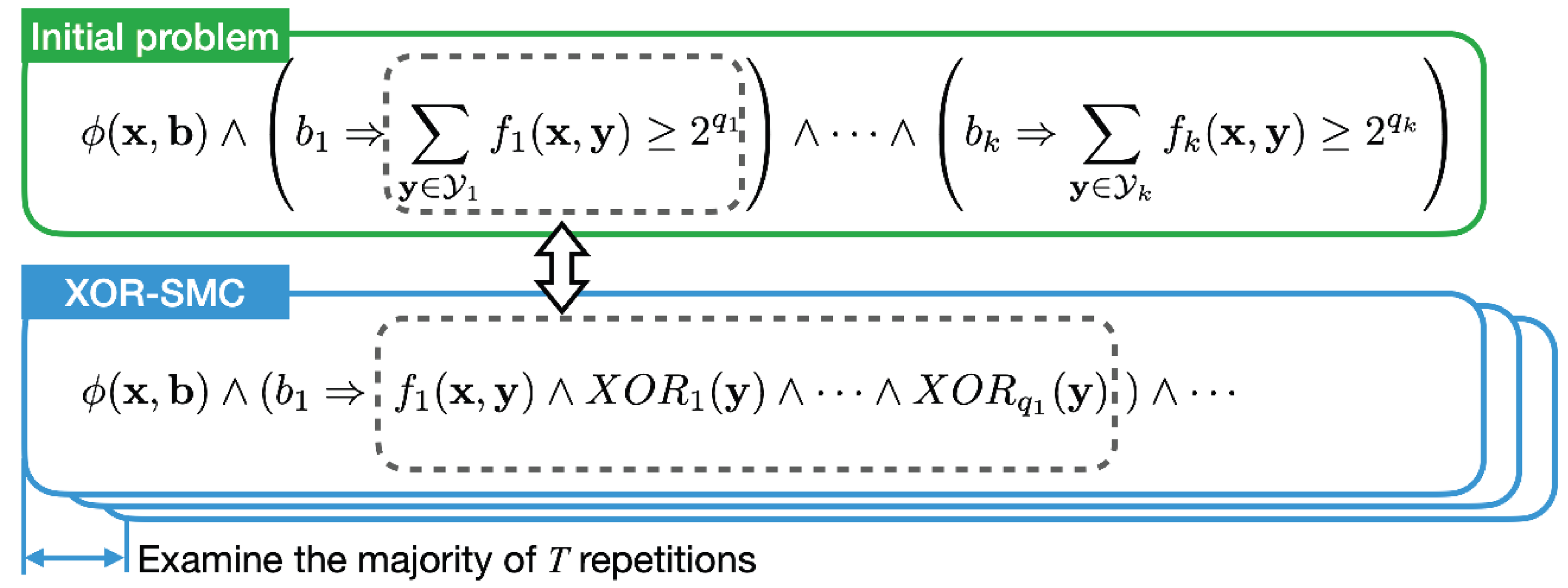
Preliminaries: XOR Counting

For a single predicate in the SMC problem: $\sum_{\mathbf{y} \in \mathcal{Y}} f(\mathbf{x}, \mathbf{y})$, suppose we would like to know if it exceeds 2^q. Consider the satisfiability (SAT) formula:

$$f(\mathbf{x}, \mathbf{y}) \wedge XOR_1(\mathbf{y}) \wedge \dots \wedge XOR_q(\mathbf{y})$$

Here, $XOR_1, \dots, XOR_q$ are randomly sampled XOR constraints. The SAT formula above is likely to be satisfiable if more than 2^q different $\mathbf{y}$ vectors render $f(\mathbf{x}, \mathbf{y})$ true. Conversely, it is likely to be unsatisfiable if $f(\mathbf{x}, \mathbf{y})$ has less than 2^q satisfying assignments.

The XOR-SMC Algorithm



- As illustrated by Figure, the key motivation behind our proposed XOR-SMC algorithm is to notice that XOR-Counting described in preliminaries section can be written as a Boolean formula.
- When we ***embed*** this Boolean formula into a SMC problem, the problem translates into a Satisfiability-Modulo-SAT problem, or equivalently, an ***SAT problem***.
- Examining the satisfiability status of the majority of the embeddings reduces error rates and gets ***a constant approximation guarantee.***

Algorithm 1: XOR-SMC (ϕ, $\{f_i\}_{i=1}^k$, $\{q_i\}_{i=1}^k$, η, c)

```
T ← ⌈((n+k) ln 2 − ln η) / α(c,k)⌉;
for t = 1 to T do
    for i = 1 to k do
        ψ_i^(t) ← f_i(x, y_i^(t));
        for j = 1, ..., q_i do
            ψ_i^(t) ← ψ_i^(t) ∧ XOR_j(y_i^(t));
        end
        ψ_i^(t) ← ψ_i^(t) ∨ ¬b_i;
    end
    ψ_t ← ψ_1^(t) ∧ ··· ∧ ψ_k^(t);
end
φ* ← φ ∧ Majority(ψ_1, ..., ψ_T);
if there exists (x, b, {y_i^(1)}_{i=1}^k, ..., {y_i^(T)}_{i=1}^k) that
  satisfies φ* then
    return True;
else
    return False;
end
```

Convert model counting to SAT formula

SMC translates to SAT

Constant Approximation Guarantee

Main Theorem (see details in the paper):

Let $0 < \eta < 1$ and $c \geq log(k+1) + 1$. Select $T = \lceil ((n+k) \ln 2 - \ln \eta)/\alpha(c,k) \rceil$, we have

- *Suppose there exists $\mathbf{x}_0 \in \{0,1\}^n$ and $\mathbf{b}_0 \in \{0,1\}^k$, such that $SMC(\phi, f_1, \dots, f_k, q_1 + c, \dots, q_k + c)$ is true,*

$$\phi(\mathbf{x}_0, \mathbf{b}_0) \wedge \left(\bigwedge \left(b_i \Rightarrow \sum_{\mathbf{y}_i \in \mathcal{Y}_i} f_i(\mathbf{x}_0, \mathbf{y}_i) \geq 2^{q_i + c} \right) \right),$$

Then algorithm XOR-SMC($\phi, \{f_i\}_{i=1}^k, \{q_i\}_{i=1}^k, \eta, c$) returns true with probability greater than $1 - \eta$.

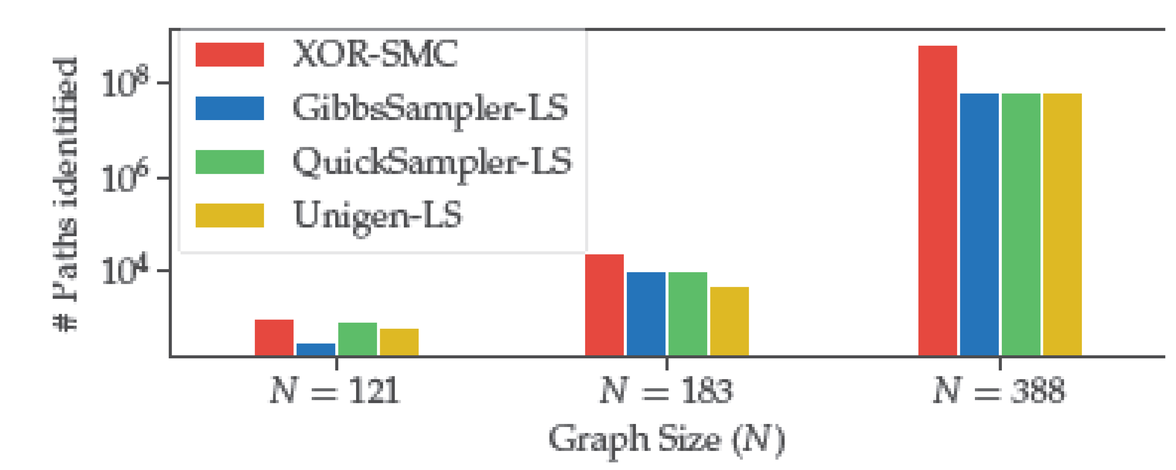
- *Contrarily, suppose $SMC(\phi, f_1, \dots, f_k, q_1 + c, \dots, q_k + c)$ is not satisfiable, i.e., $\forall \mathbf{x}, \mathbf{b}$,*

$$\neg \left(\phi(\mathbf{x}, \mathbf{b}) \wedge \left(\bigwedge \left(b_i \Rightarrow \sum_{\mathbf{y}_i \in \mathcal{Y}_i} f_i(\mathbf{x}, \mathbf{y}_i) \geq 2^{q_i - c} \right) \right) \right),$$

Then algorithm XOR-SMC($\phi, \{f_i\}_{i=1}^k, \{q_i\}_{i=1}^k, \eta, c$) returns false with probability greater than $1 - \eta$.

Experiments: Shelter Allocation

We evaluate XOR-SMC on emergency shelter allocation problems, which aim to optimize accessibility (measured by the number of paths) from residential areas to shelters.



	Graph Size		
	$N=121$	$N=183$	$N=388$
XOR-SMC (ours)	**0.04*h***	**0.11*h***	**0.16*h***
Gibbs-LS	0.56*h*	0.66*h*	6.97*h*
QuickSampler-LS	0.31*h*	0.29*h*	0.62*h*
Unigen-LS	0.08*h*	0.07*h*	0.42*h*

Our XOR-SMC finds the best shelter allocation plan in different sized maps in the shortest time.

Acknowledgement

This research was supported by NSF grants CCF-1918327.