Sampling Techniques for Constraint Satisfaction and Beyond

Kuldeep S. Meel

Rice University

Joint work with Supratik Chakraborty (IITB), Daniel J. Fremont (UCB), Dror Fried (Rice), Sanjit A. Seshia (UCB), Moshe Y. Vardi (Rice)

@Princeton       June-23-2014
Life in The 21st Century!

How do we guarantee that the systems work correctly?
Motivating Example

How do we verify that this circuit works?

- Try for all values of a and b
  - $2^{128}$ possibilities (10^{22} years)
  - Not scalable
Simulation-Based Verification

- Dominant paradigm in recent years

- Hardware design is simulated with test vectors

- Test vectors represent different verification scenarios
Constrained-Random Simulation

Sources for Constraints

- Designers:
  1. $100 < b < 200$
  2. $300 < a < 451$
  3. $40 < a < 50$ and $30 < b < 40$

- Past Experience:
  1. $400 < a < 2000$
  2. $120 < b < 230$

- Users:
  1. $1000 < a < 1100$
  2. $20000 < b < a < 22000$

Problem: How can we uniformly sample the values of $a$ and $b$ satisfying the above constraints?
Problem Formulation

Set of Constraints

SAT Formula

Given a SAT formula, can one uniformly sample solutions without enumerating all solutions while scaling to real world problems?

Scalable Uniform Generation of SAT-Witnesses
Uniform Generation of SAT-Witnesses

- Sketch based Synthesis
- Scalable Uniform Generation
- Constrained Random Simulation
- Automatic Problem Generation
Sketch-Based Synthesis

- Given: Sketch and correctness condition
- Large space of programs that satisfy the correctness conditions
- Goal: Get the optimal program (running time, memory)
- Uniformly sample from the space of programs
Outline

▪ Sampling Techniques via Uniform Generation

▪ Extension to model counting and biased sampling

▪ Discussion on hashing

▪ Future Directions
Uniform Generation

Ref: “A Scalable Near-Uniform Generator” (CAV 2013)
“Balancing Scalability and Uniformity in SAT-Witness Generator” (DAC 2014)
Prior Work

<table>
<thead>
<tr>
<th>BDD-based</th>
<th>SAT-based heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guarantees: strong</td>
<td>Guarantees: weak</td>
</tr>
<tr>
<td>Performance: weak</td>
<td>Performance: strong</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theoretical Work</th>
<th>Heuristic Work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guarantees: strong</td>
<td>Guarantees: weak</td>
</tr>
<tr>
<td>Performance: weak</td>
<td>Performance: strong</td>
</tr>
</tbody>
</table>

BGP Algorithm

XORSample’
Our Contribution

- **BDD-based**
  - Guaranettes: strong
  - Performance: weak

- **UniGen**
  - Guarantees: strong
  - Performance: strong

- **SAT-based heuristics**
  - Guarantees: weak
  - Performance: strong

- **Theoretical Work**
  - Guarantees: strong
  - Performance: weak

- **Heuristic Work**
  - Guarantees: weak
  - Performance: strong

- **BGP Algorithm**

- **UniGen**

- **XORSample’**

- **INDUSTRY**

- **ACADEMIA**
Partitioning into equal “small” cells
Partitioning into equal “small” cells

Pick a random cell

Pick a random solution from this cell
How to Partition?

How to partition into roughly equal small cells of solutions without knowing the distribution of solutions?

Universal Hashing
[Carter-Wegman 1979, Sipser 1983]
Universal Hashing

- Hash functions from mapping \( \{0,1\}^n \) to \( \{0,1\}^m \)
  - \( 2^n \) elements to \( 2^m \) cells

- Random inputs => All cells are *roughly* equal

- Universal hash functions:
  - Adversarial (any distribution) inputs => All cells are *roughly* equal

- Need stronger bounds on range of the size of cells
Lower Universality $\Rightarrow$ Lower Complexity

- $H(n,m,r)$: Family of $r$-universal hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$ ($2^n$ elements to $2^m$ cells)

- Higher the $r$ => Stronger guarantees on range of size of cells

- $r$-wise universality $\Rightarrow$ Polynomials of degree $r-1$

- Lower universality $\Rightarrow$ lower complexity
Hashing-based Approaches

Solution space

n-universal hashing

3GP Algorithm

All cells should be small

Uniform Generation
Scaling to Thousands of Variables

n-universal hashing

2-universal hashing

Solution space

Random

BGP Algorithm

All cells should be small

Uniform Generation

UniGen

Only a randomly chosen cells needs to be “small”

Near-Uniform Generation
Scaling to Thousands of Variables

From tens of variables to thousands of variables!

BGP Algorithm

All cells should be small

UniGen

Only a randomly chosen cells needs to be “small”

Uniform Generation

Near-Uniform Generation
UniGen

$R_F$
UniGen

$R_F$

IsSmall?

NO
UniGen
UniGen

IsSmall?

YES
UniGen

IsSmall？

YES

Select a solution randomly from the partition.
Strong Theoretical Guarantees

- **Near-Uniformity**

  For every solution $y$ of $R_F$

  \[
  \frac{1}{(6.84 + \varepsilon)} \times \frac{1}{|R_F|} \leq \Pr[y \text{ is output}] \leq \frac{(6.84 + \varepsilon)}{|R_F|}
  \]

- **Success Probability**

  UniGen succeeds with probability at least 0.52

  - In practice, succ. probability $> 0.9$

- **Polynomial calls to SAT Solver**
Results: Uniformity

- Benchmark: case110.cnf; \#var: 287; \#clauses: 1263
- Total Runs: $4 \times 10^6$; Total Solutions: 16384
Results: Uniformity

- Benchmark: case110.cnf;  #var: 287;  #clauses: 1263
- Total Runs: $4 \times 10^6$;  Total Solutions : 16384
2-3 Orders of Magnitude Faster

Time(s)

Benchmarks

case47

case_3_b14_3

case105

case8

case203

case145

case61

case9

case15

case140

case_2_b14_1

case_3_b14_1

UniGen

XORSample'

squaring14

case_2_ptb_1

case_1_ptb_1

case_2_b14_2

case_3_b14_2
Outline

- Sampling Techniques via Uniform Generation

- Extension to model counting and biased sampling

- Discussion on hashing

- Future Directions
Approximate Model Counting

Ref: “A Scalable Approximate Model Counter” (CP 2013)
What is Model Counting?

- Given a SAT formula $F$
- $R_F$: Set of all solutions of $F$
- Problem ($\#SAT$): Estimate the number of solutions of $F$ ($\#F$) i.e., what is the cardinality of $R_F$?
- E.g., $F = (a \lor b)$
- $R_F = \{(0,1), (1,0), (1,1)\}$
- The number of solutions ($\#F$) = 3

$\#P$: The class of counting problems for decision problems in NP!
Practical Applications

Exciting range of applications!

- Probabilistic reasoning/Bayesian inference
- Planning with uncertainty
- Multi-agent/ adversarial reasoning

[Roth 96, Sang 04, Bacchus 04, Domshlak 07]
Counting through Partitioning
Counting through Partitioning

Pick a random cell

Total # of solutions = #solutions in the cell * total # of cells
ApproxMC in Action
ApproxMC in Action

Algorithm

Median

690  710  730  730  731  831  ..........  834

\( t \)
Strong Theoretical Results

ApproxMC (CNF: F, tolerance: $\varepsilon$, confidence: $\delta$)

Suppose ApproxMC($F,\varepsilon,\delta$) returns $C$. Then,

$$\Pr \left[ \frac{\#F}{1+\varepsilon} \leq C \leq (1+\varepsilon) \#F \right] \geq \delta$$

ApproxMC runs in time polynomial in $\log (1-\delta)^{-1}$, $|F|$, $\varepsilon^{-1}$ relative to SAT oracle
Can Solve a Large Class of Problems

Large class of problems that lie beyond the exact counters but can be computed by ApproxMC.
Mean Error: Only 4% (allowed: 75%)

Mean error: 4% – much smaller than the theoretical guarantee of 75%
Weighted/Biased Sampling

Ref: “Distribution-Aware Sampling and Weighted Model Counting for SAT” (To Appear in AAAI 204)
Partition into (weighted) equal “small” cells
Partition into (weighted) equal “small” cells

Pick a random cell

Pick (by weight) a random solution from this cell
Projection Counting/Sampling

- What if I care about only few variables?

- \((a=0, b = 0, c = 1), (a = 0, b = 0, c=0), (a = 0, b = 1, c=0)\)

- Partition only on the projected subspace
Outline

- Sampling Techniques via Uniform Generation
- Extension to model counting and biased sampling
- Discussion on hashing
- Future Directions
XOR-Based Hashing

- 3-universal hashing
- Partition $2^n$ space into $2^m$ cells
- Variables: $X_1$, $X_2$, $X_3$, ..., $X_n$
- Pick every variable with prob. $\frac{1}{2}$, XOR them and equate to 0/1 with prob. $\frac{1}{2}$
- $X_1 + X_3 + X_6 + \ldots + X_{n-1} = 0$ (Cell ID: 0/1)
- $m$ XOR equations -> $2^m$ cells
- The cell: $F \&\&$ XOR (CNF+XOR)
XOR-Based Hashing

- CryptoMiniSAT: Efficient for CNF+XOR
- Avg Length : n/2
- Smaller the XORs, better the performance

How to shorten XOR clauses?
Independent Variables

- Set of variables such that assignments to these uniquely determine assignments to rest of variables for formula to be true

- \((a \lor b = c) \Rightarrow \) Independent Support: \(\{a, b\}\)

- \# of auxiliary variables introduced: 2-3 orders of magnitude

- Hash only on the independent variables (huge speedup)
Future Directions
Extension to More Expressive Domains (SMT, CSP)

- Efficient 3-independent hashing schemes
  - Extending bit-wise XOR to SMT provides guarantees but no advantage of SMT progress

- Solvers to handle F + Hash efficiently
  - CryptoMiniSAT has fueled progress for SAT domain
  - Similar solvers for other domains?
Exploring CNF+XOR

- Very little understanding as of now
- Can we observe phase transition?
- Eager/Lazy approach for XORs?
- How to reduce size of XORs further?
Potentially New Connections

- **Near-Uniformity**

For every solution \( y \) of \( R_F \)

\[
\frac{1}{(6.84+\varepsilon)} \times \frac{1}{|R_F|} \leq \Pr [y \text{ is output}] \leq \frac{6.84+\varepsilon}{|R_F|}
\]

- **Almost-Uniformity**

For every solution \( y \) of \( R_F \)

\[
\frac{1}{(1+\varepsilon)} \times \frac{1}{|R_F|} \leq \Pr [y \text{ is output}] \leq \frac{1+\varepsilon}{|R_F|}
\]
Potentially New Connections

- Polynomial inter-reducibility of near-uniform generation and approximate model counting [Jerrum-Valiant-Vazirani, 1986]
Potentially New Connections

- Is there a similar relation between near-uniform generation (much weaker than almost uniform generation) and approximate model counting?
Some Questions?

- Approximate Model Counting
- Near-Uniform Generation
- Almost Uniform Generation
Publications


Collaborators

- Prof. Supratik Chakraborty (IITB)
- Daniel J. Fremont (UCB)
- Dr. Dror Fried (Rice)
- Prof. Sanjit A. Seshia (UCB)
- Prof. Moshe Vardi (Rice)
Impact of Independent Variables

Time(s)

Benchmarks

Independent
All

59