Balancing Scalability and Uniformity in SAT-Witness Generator

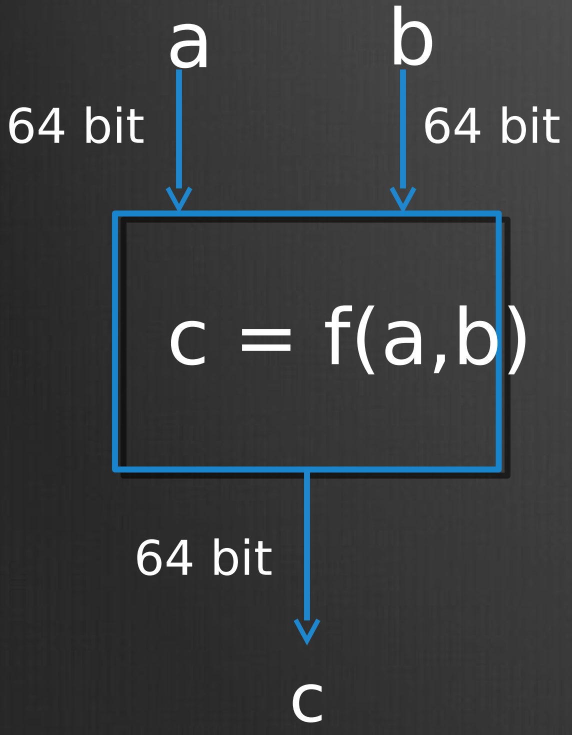
Supratik Chakraborty[1], **Kuldeep S Meel[2]**, Moshe Y Vardi[2]

[1]Indian Institute of Technology Bombay, India
[2]Department of Computer Science, Rice University

Simulation-Based Verification

- Dominant paradigm in recent years
- Hardware design is simulated with test vectors
- Test vectors represent different verification scenarios

Constrained-Random Simulation

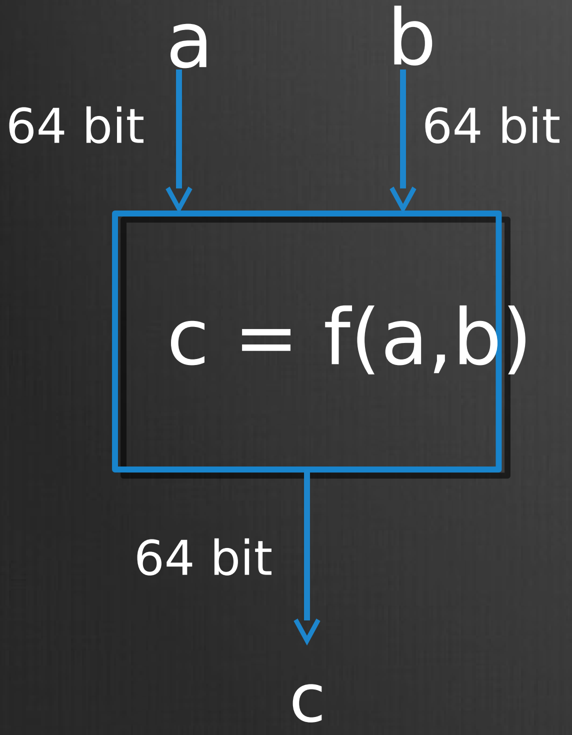


Sources for Constraints

- 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



a

b

64 bit

64 bit

c = f(a,b)

64 bit

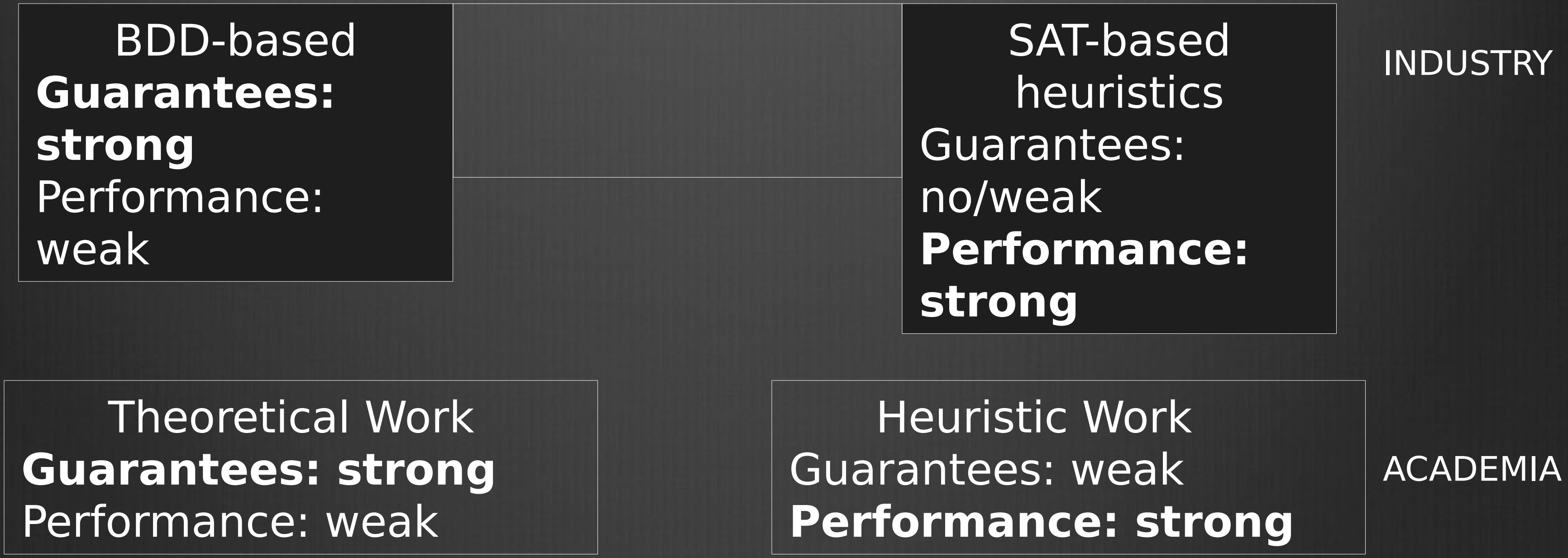
c

Set of Constraints

SAT Formula

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

Prior Work



BDD-based
Guarantees: strong
Performance: weak

SAT-based heuristics
Guarantees: no/weak
Performance: strong

INDUSTRY

Theoretical Work
Guarantees: strong
Performance: weak

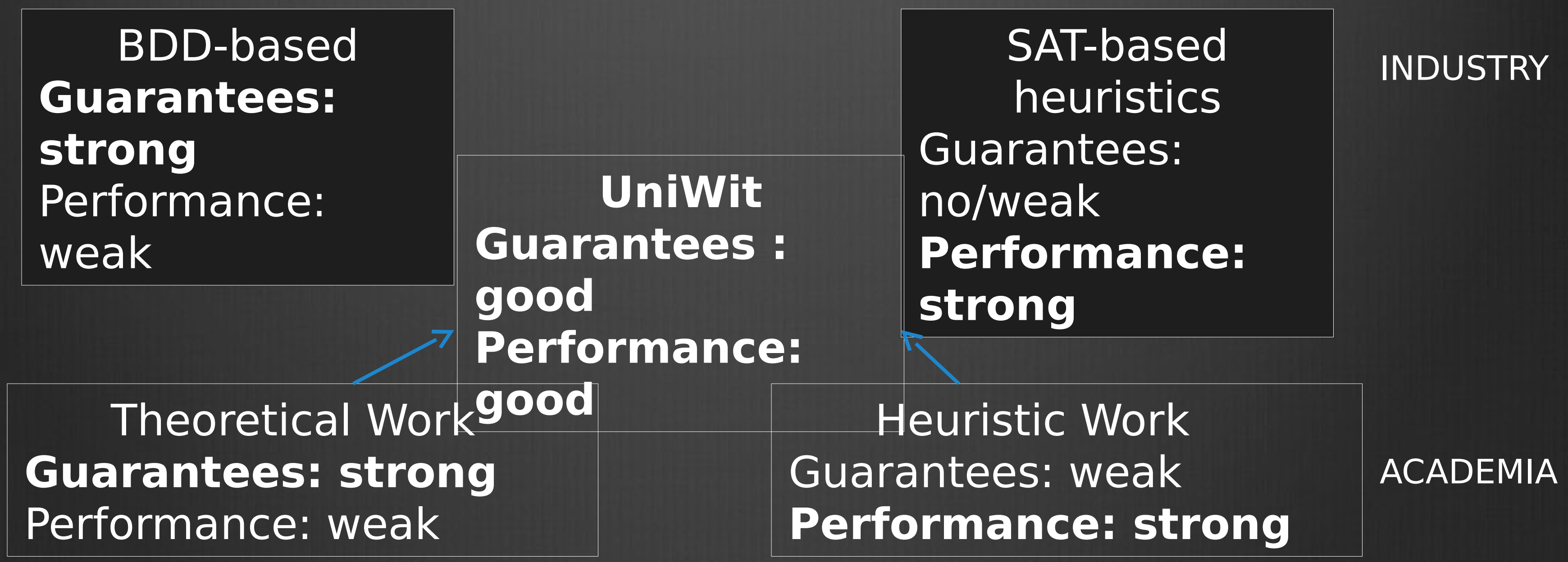
Heuristic Work
Guarantees: weak
Performance: strong

ACADEMIA

BGP Algorithm

XORSample’

Our CAV’13 Work



BDD-based
Guarantees: strong
Performance: weak

UniWit
Guarantees : good
Performance: good

SAT-based heuristics
Guarantees: no/weak
Performance: strong

INDUSTRY

Theoretical Work
Guarantees: strong
Performance: weak

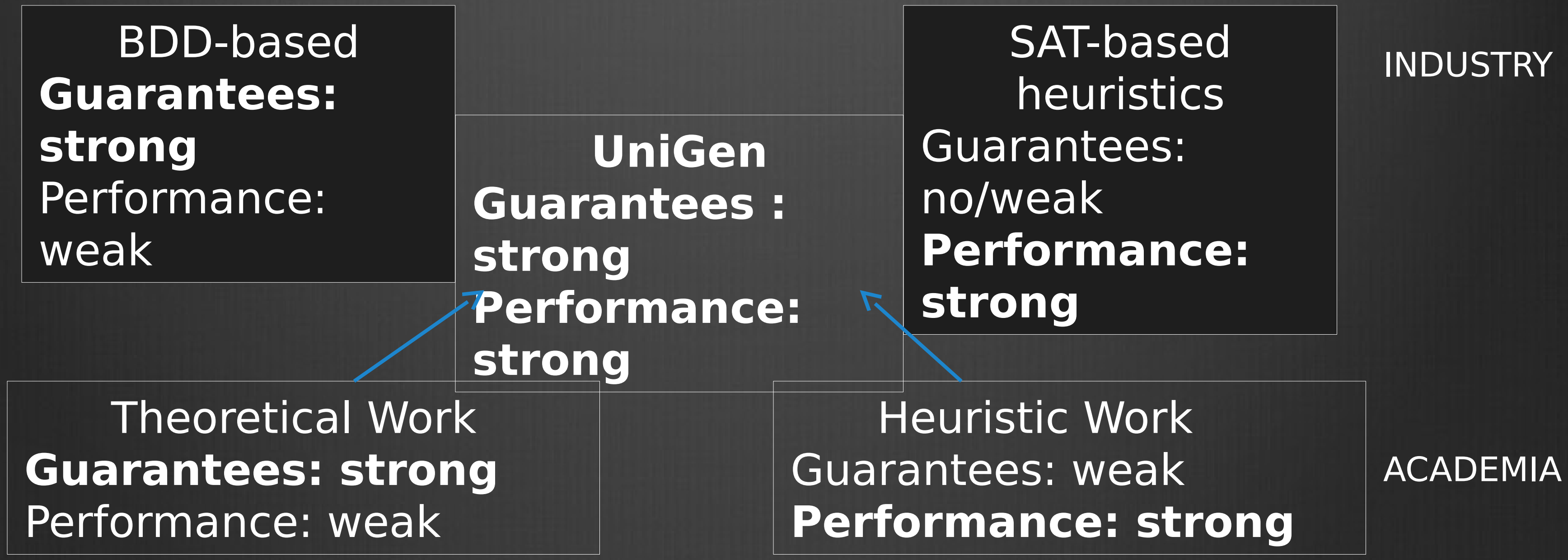
Heuristic Work
Guarantees: weak
Performance: strong

ACADEMIA

BGP Algorithm

XORSample’

Our Contribution (DAC’14)



BDD-based
Guarantees: strong
Performance: weak

UniGen
Guarantees : strong
Performance: strong

SAT-based heuristics
Guarantees: no/weak
Performance: strong

INDUSTRY

Theoretical Work
Guarantees: strong
Performance: weak

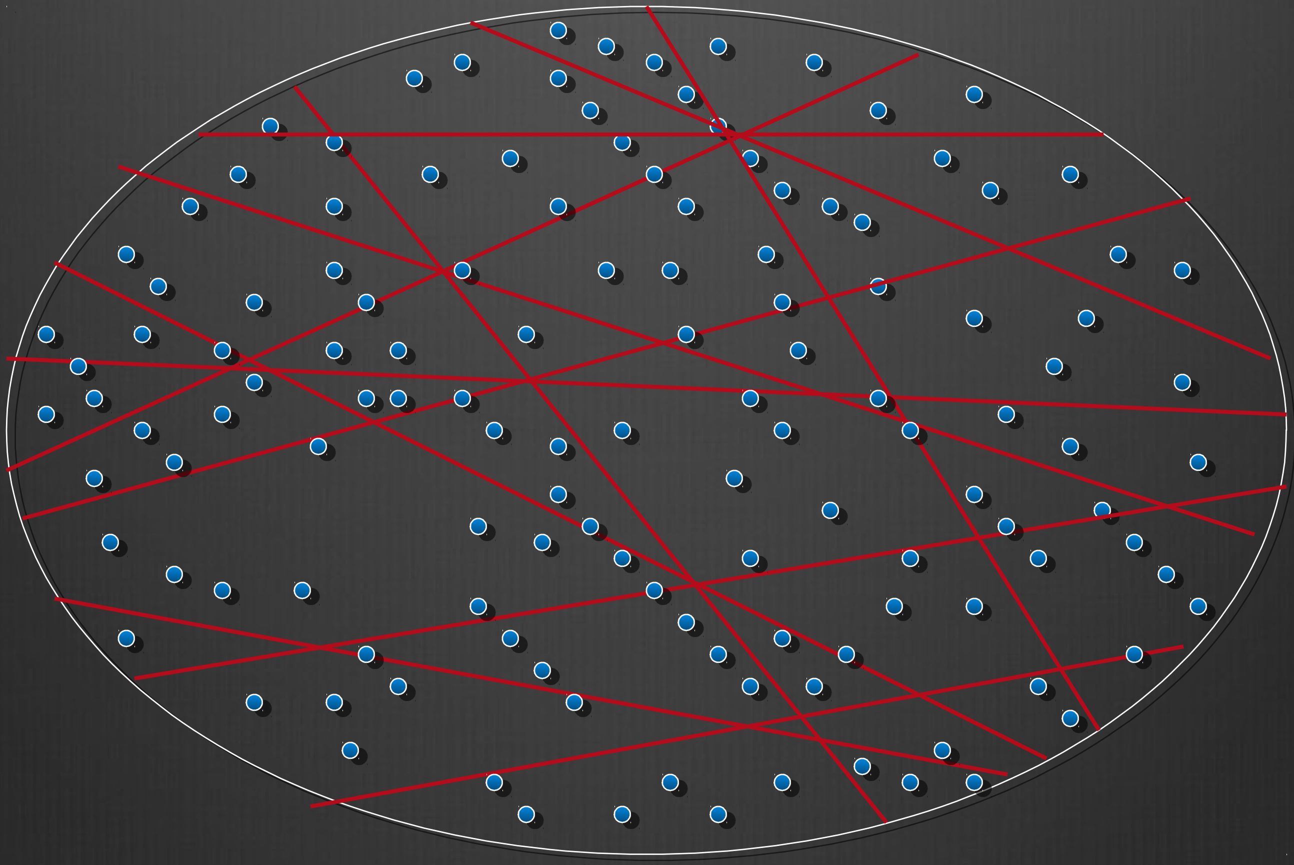
Heuristic Work
Guarantees: weak
Performance: strong

ACADEMIA

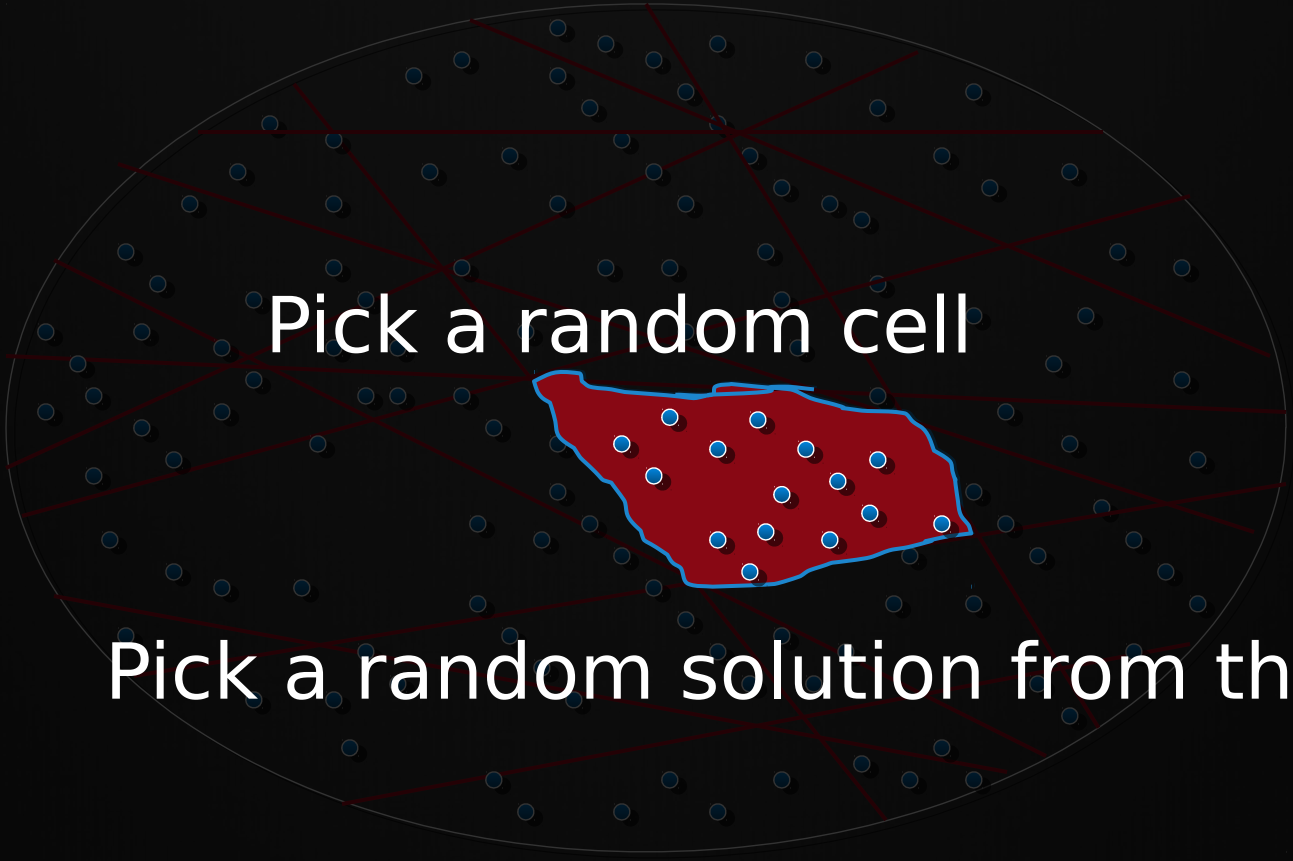
BGP Algorithm

XORSample’

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?

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

Strong Theoretical Guarantees

- Near-Uniformity

For every solution y of R_F

$1/(6.84+\varepsilon) \times 1/|R_F| <= \Pr$ [y is output] $<= (6.84+\varepsilon) /|R_F|$

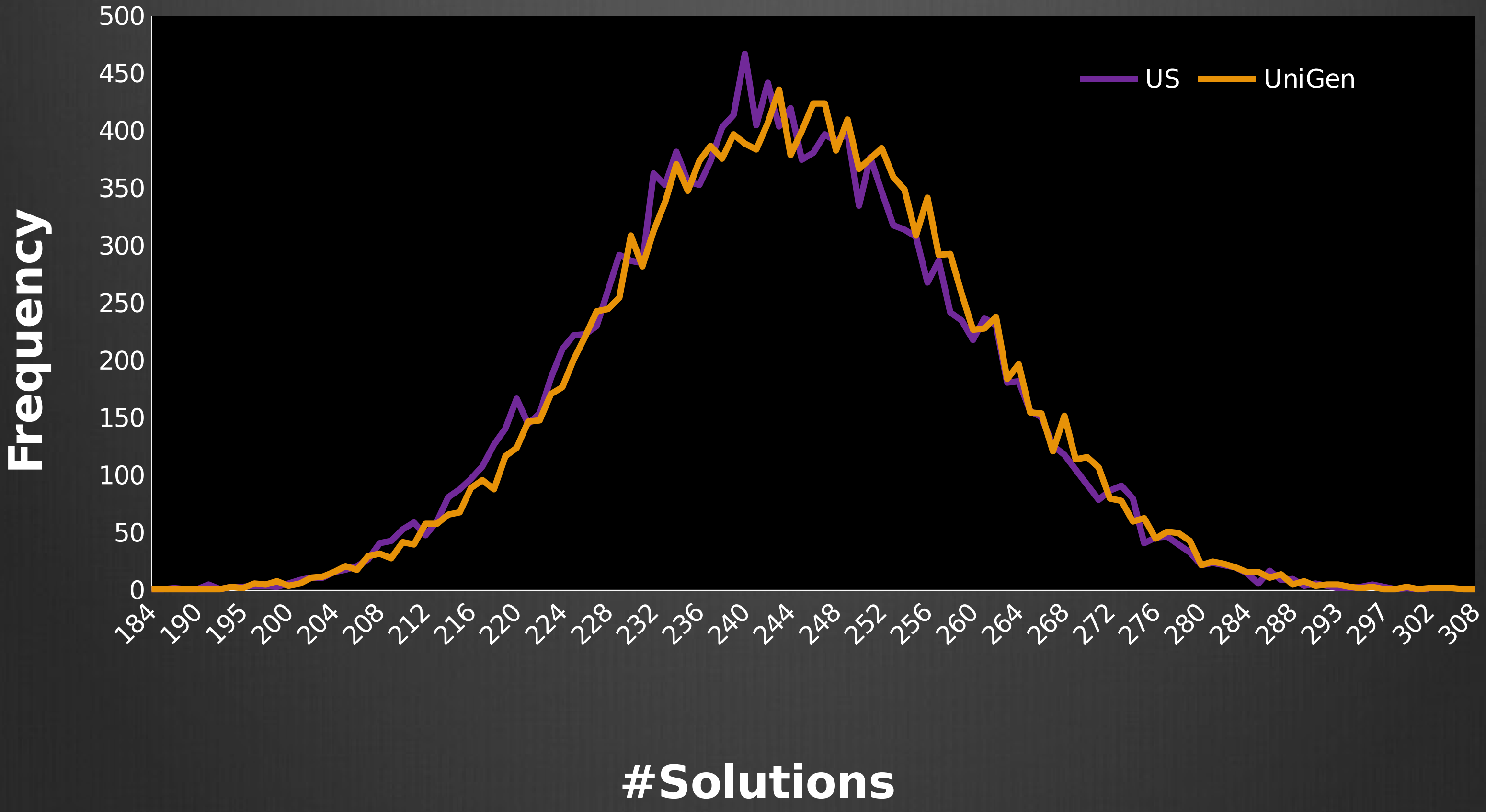
- Success Probability

UniGen succeeds with probability at least 0.52

  - In practice, succ. probability > 0.9

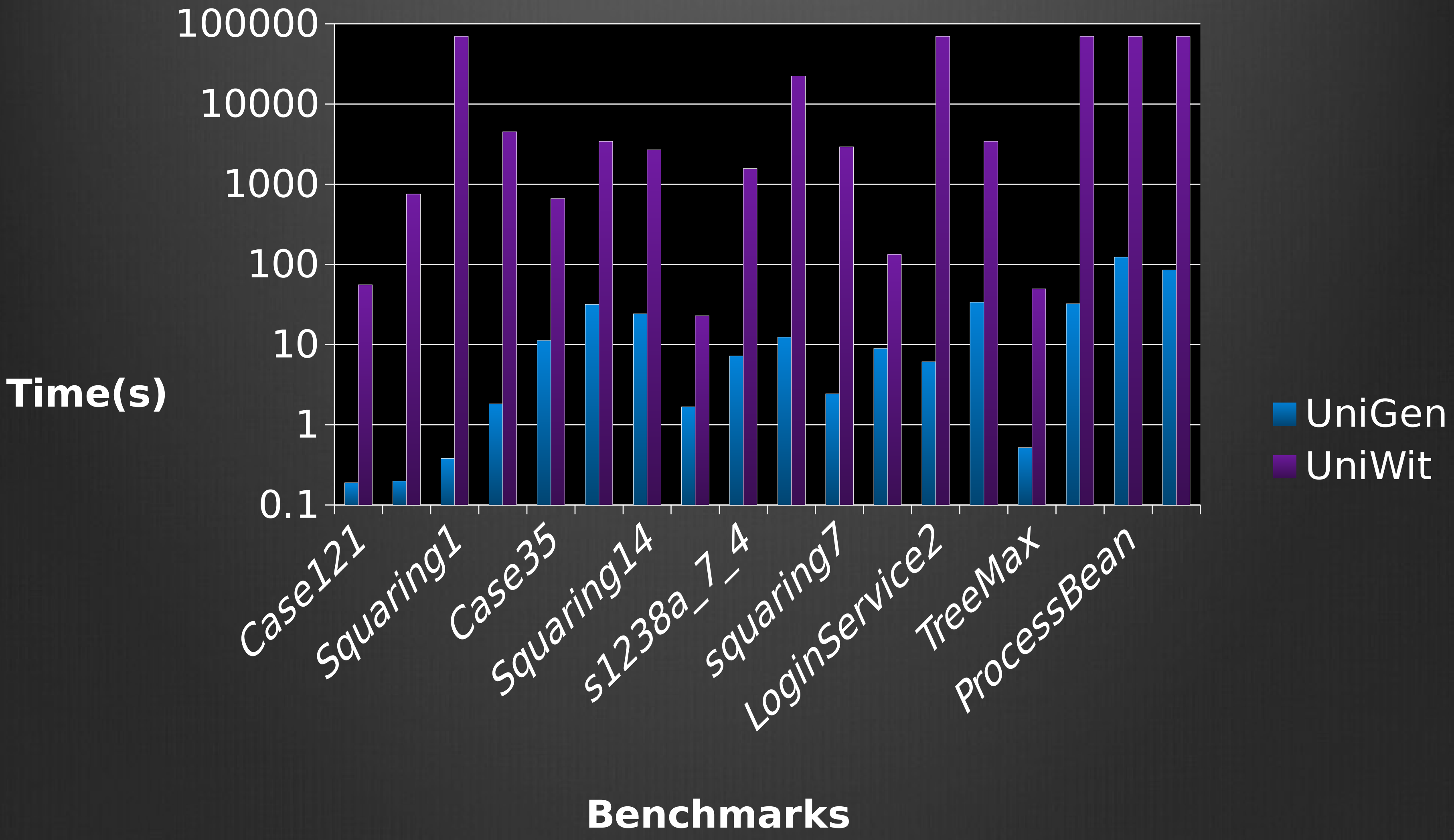
- Polynomial number of calls to SAT Solver

Results: Uniformity



- Benchmark: case110.cnf; #var: 287; #clauses: 1263

2-3 Orders of Magnitude Faster



Takeaways

- Uniform Generation had diverse applications
- Prior work either did not provide guarantees or did not scale.
- Proposed a new scalable approach based on hashing that provides strong guarantees
- Runs 2-3 orders of magnitude faster than prior state-of-art tools