INC: A Scalable Incremental Weighted Sampler

Suwei Yang[1,2], Victor Liang[2], Kuldeep S. Meel[1]

FMCAD22

[1]National University of Singapore, [2]Grabtaxi Holdings

Weighted Sampling

- Given logical formula F and literal weight function W, sample satisfying assignments according to the weights

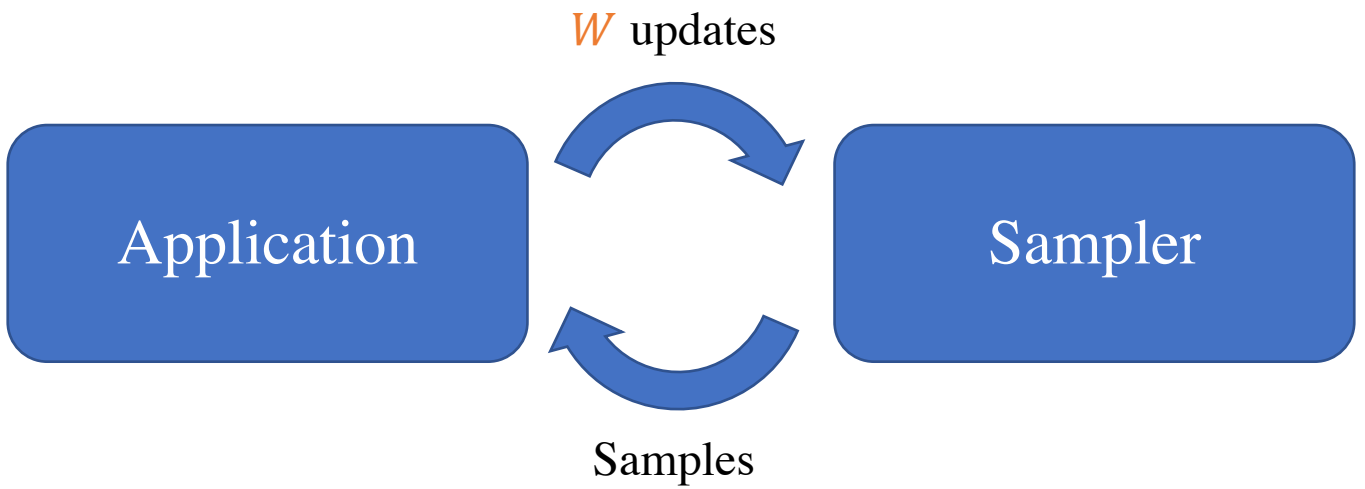
- Logical formula F defines sampling set

- Weight function W defines sampling distribution

Incremental Weighted Sampling

- Weighted sampling over multiple rounds

- Incremental weight function W updates at each round



Incremental Weighted Sampling

- Software testing [PAPDC19, BLM20]
  - Valid configurations expressed as logical formula F
  - Test cases sampled over multiple rounds
  - Updated weight function at each round, bias towards uncovered cases

Existing State Of The Art

- Incremental sampling with WAPS [GSRM19]:
  - Compiles formula F into a DAG (d-DNNF)
  - Annotate edges of DAG with joint probabilities and sample
  - Given updated weights, re-annotate DAG edges and sample

Existing State Of The Art

Performance of WAPS under incremental setting not ideal

Benchmark	Sampler	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Total
h8max	WAPS	90.3	104.2	92.4	116.0	94.3	94.1	112.9	92.9	94.4	120.4	1011.9

Incremental rounds (R2, …, R10) not much faster than R1, sometimes slower

Existing State Of The Art

Performance of WAPS under incremental setting not ideal

Benchmark	Sampler	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Total
h8max	WAPS	90.3	104.2	92.4	116.0	94.3	94.1	112.9	92.9	94.4	120.4	1011.9

Incremental rounds (R2, …, R10) not much faster than R1, sometimes slower

Insight: Annotate and Sampling takes up most of the time, not Compilation!

Existing State Of The Art

- WAPS major design choices:
  - Deterministic Decomposable Negation Normal Form (d-DNNF) [Darwiche02]
  - Annotate using arbitrary precision math
  - 2 passes of d-DNNF to annotate and sample

INC

INC – weighted sampler optimized for incremental settings

Design choices	WAPS	INC
Knowledge compilation form	d-DNNF	Probabilistic OBDD[∧] (PROB)
Annotate	Arbitrary precision math	Log-space math
Algorithm	2 Passes	1 combined bottom-up pass

PROB is based on OBDD[∧] [LLY17]

Outline

1. Probabilistic OBDD[∧] (PROB)
2. Annotate weight function W on PROB
3. Weighted sampling with INC
4. Experiment results

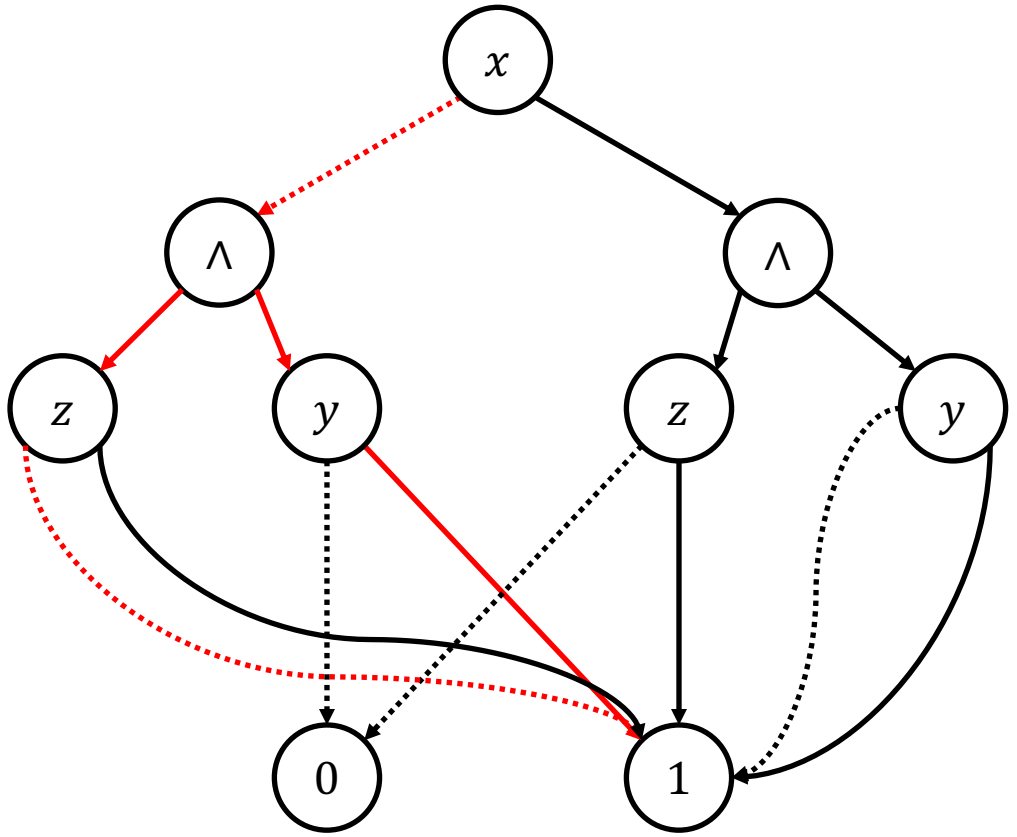
PROB Structure

- Probabilistic OBDD[∧] (PROB):
  - ∧-nodes – conjunctions in assignment space
  - Decision nodes – logical disjunctions in assignment space
  - 1, 0 – nodes – represents True and False
  - Weights applied on edges of decision nodes
  - Satisfying assignment – tree from root node to 1-node

PROB Structure

x	y	z	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$F = (x \vee y) \wedge (\neg x \vee z)$$



Weight Function

- Weight function W defines probability distribution
- The weight function gives each literal a weight, $W(l)$
- The weight of an assignment τ is given by $\prod_{l \in \tau} W(l)$

Annotate

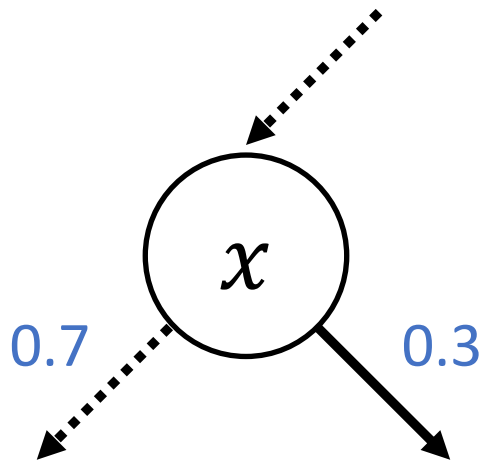
- Decision node representing variable x

Literal	Weight
$W(\neg x)$	0.7
$W(x)$	0.3

  - x is set to either true or false

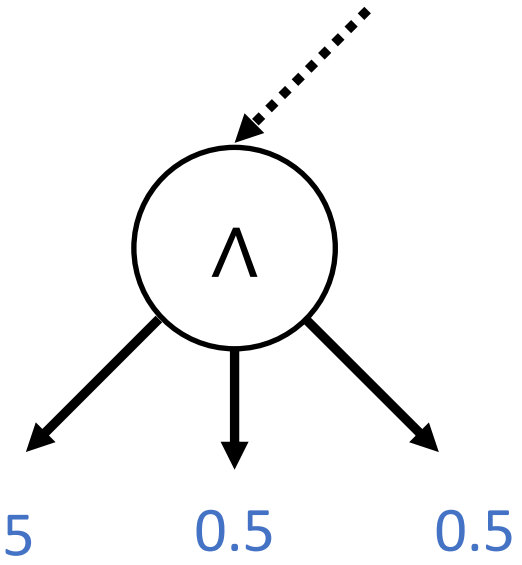
  - Joint probability is

$$W(\neg x) \times P\ (lo\ branch) + W(x) \times P\ (hi\ branch)$$



Annotate

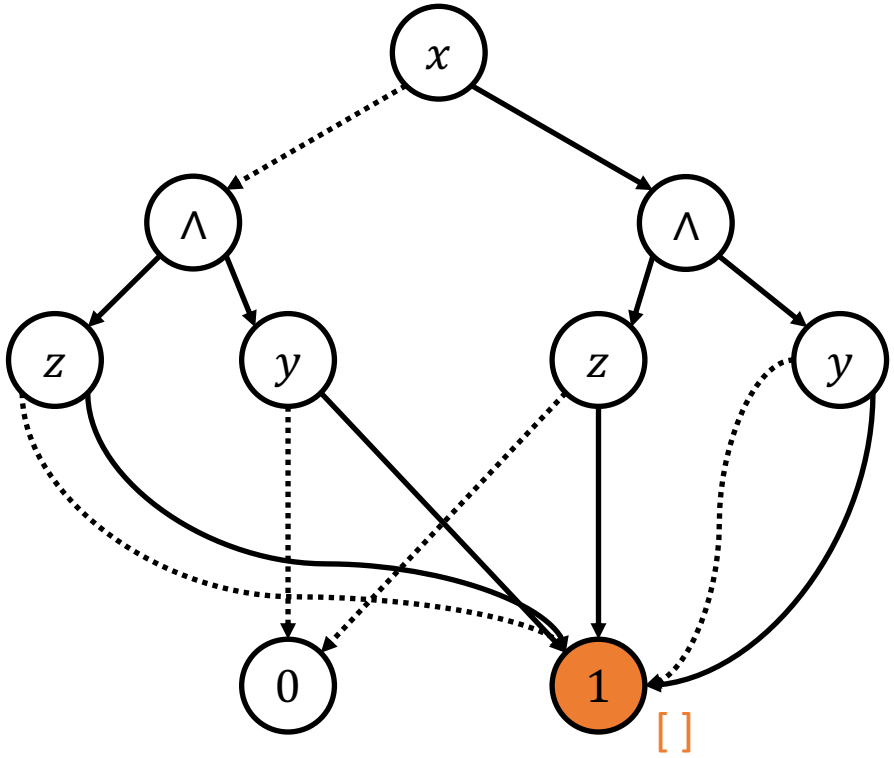
- Conjunction node
  - Child branches have mutually disjoint variable sets
  - Joint probability is product of joint probability of each child branch



$0.5 \times 0.5 \times 0.5 = 0.125$

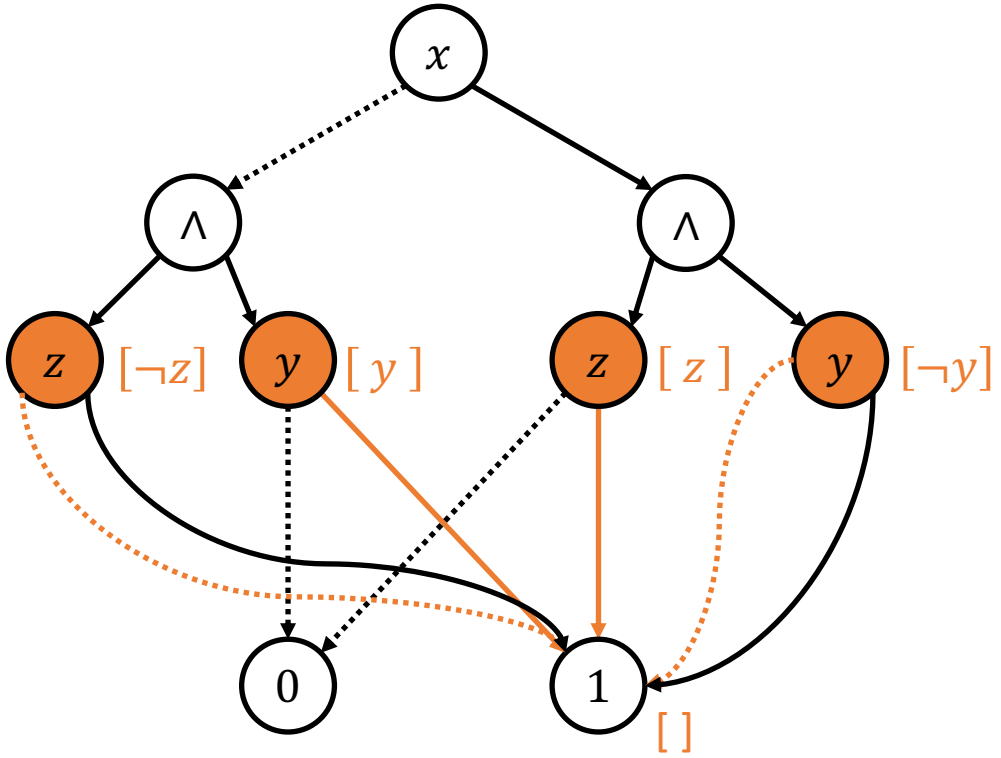
Weighted Sampling With INC

- Bottom-up pass, starting from 1-node
- At decision node, sample variable assignment according to joint probabilities of child branches
- At ∧-node, concatenate partial assignments from child nodes
- Output sampled assignment at root node



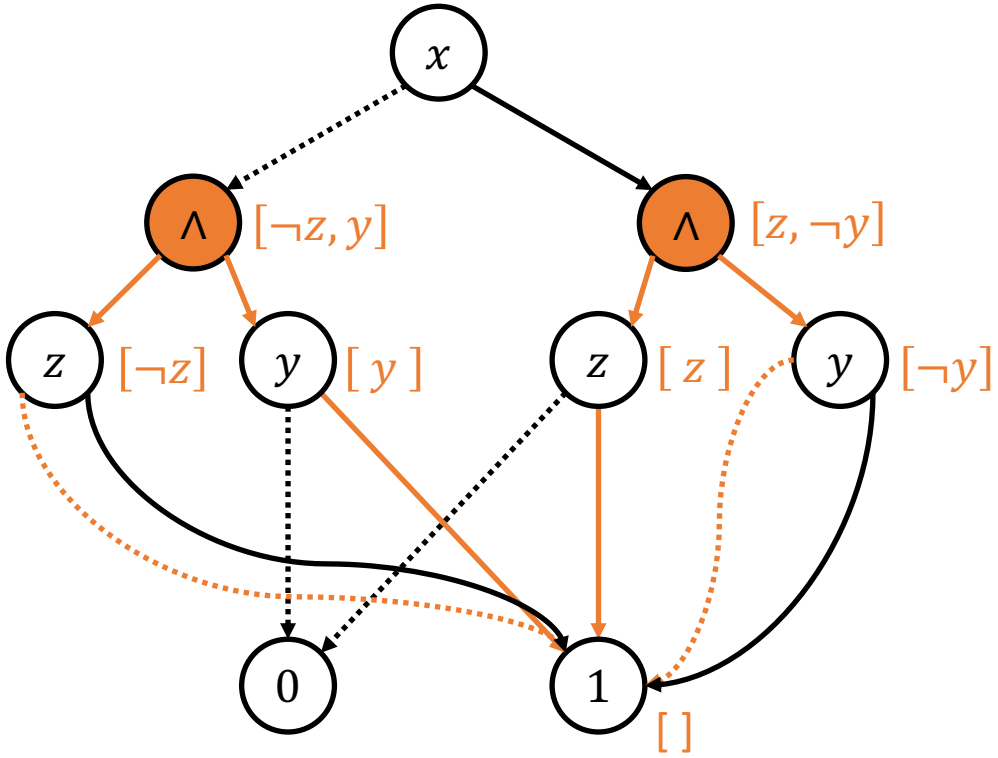
Weighted Sampling With INC

- Bottom-up pass, starting from 1-node
- At decision node, sample variable assignment according to joint probabilities of child branches
- At ∧-node, concatenate partial assignments from child nodes
- Output sampled assignment at root node



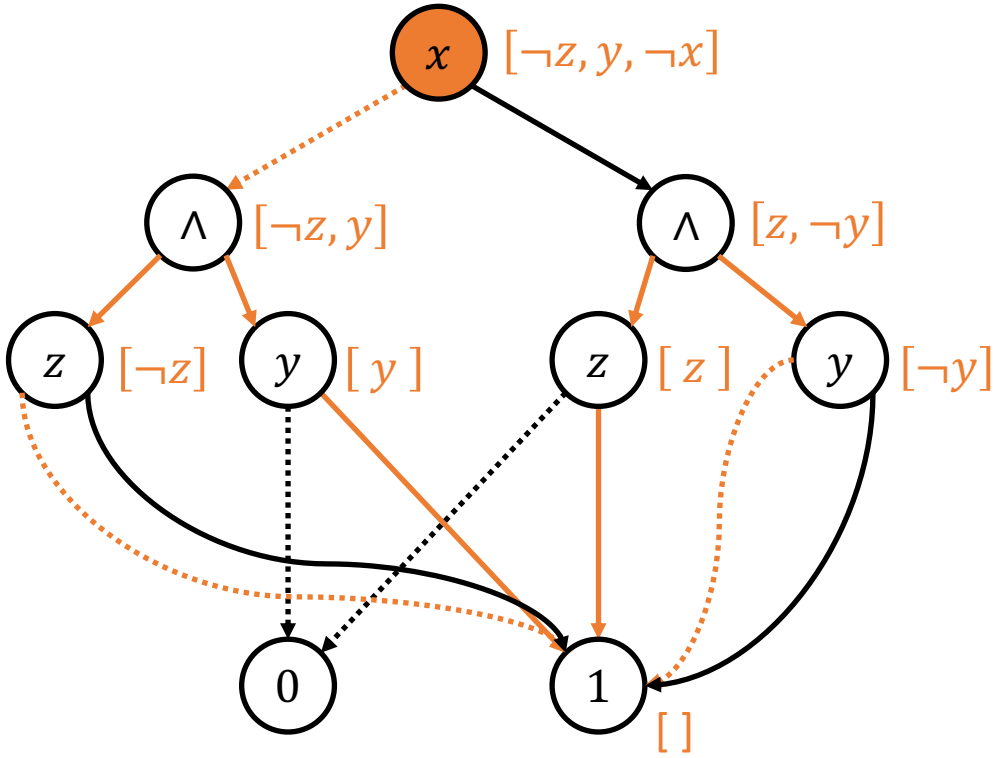
Weighted Sampling With INC

- Bottom-up pass, starting from 1-node
- At decision node, sample variable assignment according to joint probabilities of child branches
- At ∧-node, concatenate partial assignments from child nodes
- Output sampled assignment at root node



Weighted Sampling With INC

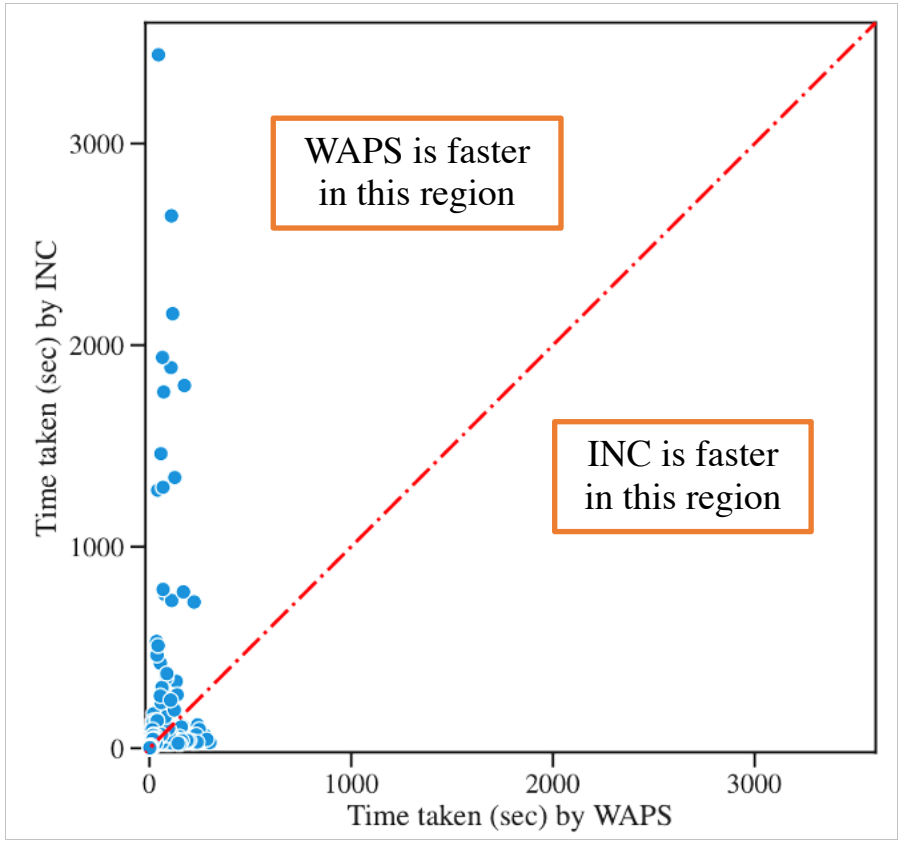
- Bottom-up pass, starting from 1-node
- At decision node, sample variable assignment according to joint probabilities of child branches
- At ∧-node, concatenate partial assignments from child nodes
- Output sampled assignment at root node



Experiments

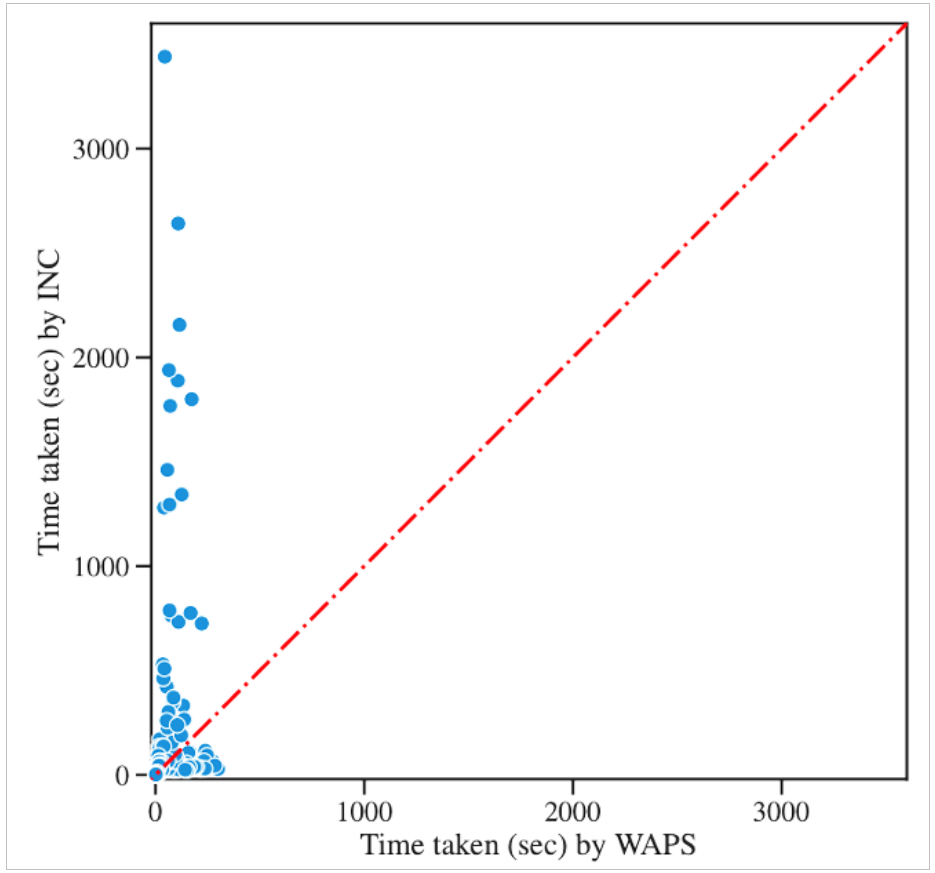
- 896 Benchmarks – 17 to 25615 variables, 31 to 82417 clauses
- Incremental weighted sampling application [BLM20]
  - Comparison to WAPS as sampler component
  - 10 rounds (R1 to R10), 100 samples each
  - Timeout – 3600s
  - R1: Compile + Annotate + Sample
  - R2 to R10: Annotate + Sample

Experiments: Runtime Comparison

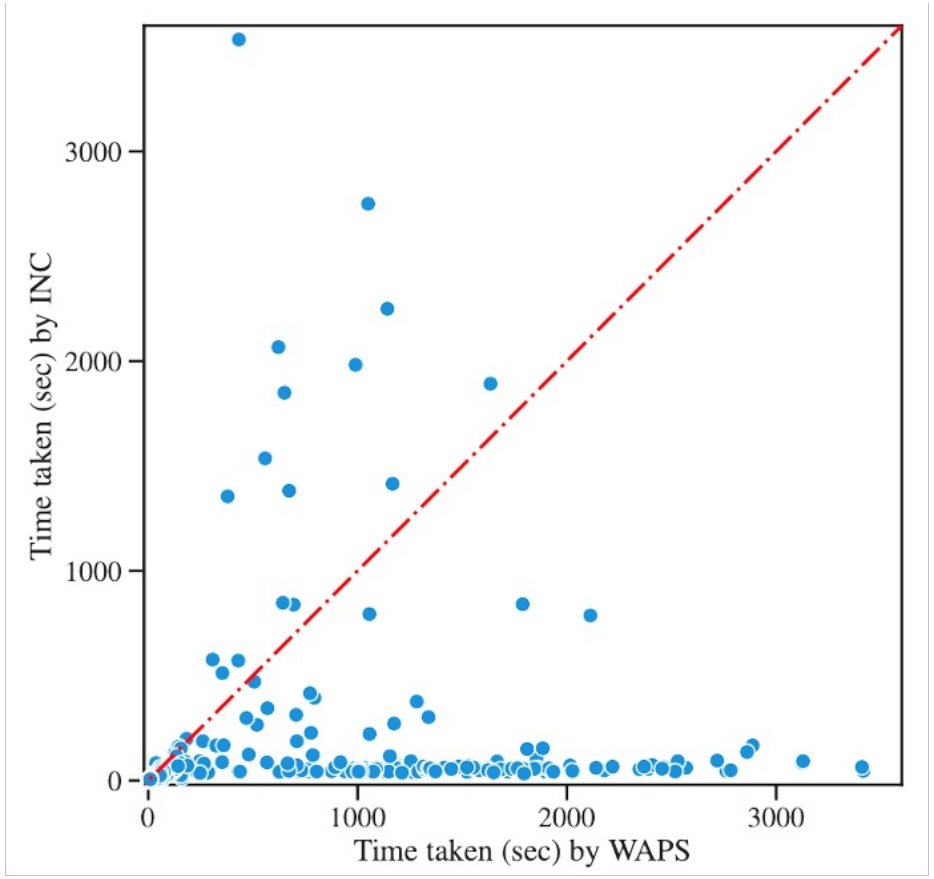


Scatter plot of runtime of initial round (R1) – Compilation + Annotate + Sample

Experiments: Runtime Comparison



Initial Round



Over 10 Rounds

Scatter plot of runtime comparison between INC and WAPS

Experiments: Runtime Statistics

Result statistics (median)	WAPS	INC
Initial round component (R1)	0.44×	1.0×
Incremental round component SUM(R2-R10)	4.48×	1.0×
Total runtime	1.69×	1.0×

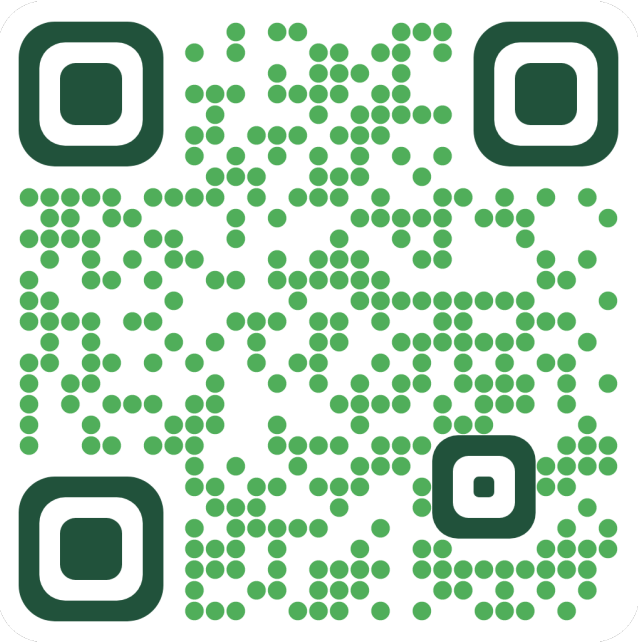
Experiments: Design Choice Analysis

- DAG size (d-DNNF vs PROB):
  - d-DNNF has more nodes in all cases, 4.6× PROB size
- Annotate (arbitrary precision vs log-space):
  - Arbitrary precision slower in >= 75% of cases, 1.12× runtime of log-space

INC's design trades compilation time with incremental performance!

Summary

- INC – incremental weighted sampler
  - Uses PROB for sampling
  - Log-space annotate
  - Fast and scalable incremental sampling routine



https://github.com/grab/inc-weighted-sampler

suwei.yang@comp.nus.edu.sg

End of slides

Experiments: Runtime Examples

Benchmark	Tool	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Total	Speed
or-50-5-5-UC-10	WAPS	**56.6**	56.3	52.5	59.4	52.5	53.6	59.4	53.2	53.4	61.7	**558.6**	1.0×
(100, 253)	INC	1461.3	**7.6**	**8.4**	**8.4**	**8.4**	**8.4**	**8.5**	**8.5**	**8.4**	**8.5**	1536.3	0.4×
or-100-20-9-UC-30	WAPS	**73.0**	69.1	66.7	76.0	66.5	66.9	76.6	66.0	66.9	78.6	706.1	1.0×
(200, 528)	INC	269.5	**4.7**	**4.8**	**4.8**	**4.9**	**5.1**	**4.8**	**4.8**	**4.8**	**5.1**	**313.4**	2.3×
s953a_15_7	WAPS	**1.7**	1.1	1.1	1.2	1.0	1.1	1.2	1.1	1.1	1.3	11.9	1.0×
(602, 1657)	INC	4.9	**0.7**	**0.7**	**0.7**	**0.7**	**0.7**	**0.7**	**0.7**	**0.7**	**0.7**	**11.5**	1.0×
h8max	WAPS	90.3	104.2	92.4	116.0	94.3	94.1	112.9	92.9	94.4	120.4	1011.9	1.0×
(1202, 3072)	INC	**34.1**	**2.1**	**2.2**	**2.4**	**2.3**	**2.4**	**2.2**	**2.4**	**2.4**	**2.3**	**55.7**	18.2×
innovator	WAPS	195.5	221.9	201.3	244.4	200.1	206.7	247.2	202.0	202.9	257.4	2179.3	1.0×
(1256, 50452)	INC	**32.8**	**1.6**	**1.8**	**1.9**	**1.9**	**1.9**	**1.8**	**1.9**	**1.9**	**1.9**	**49.4**	44.1×

Experiments: Runtime Statistics

Statistic	WAPS MEAN(R2 to R10) / WAPS R1	INC MEAN(R2 to R10) / INC R1	WAPS R1 / INC R1	WAPS SUM(R2 to R10) / INC SUM(R2 to R10)	WAPS Total / INC Total
Mean	0.74	0.064	1.03	15.66	6.12
Std	0.24	0.040	1.47	26.42	10.73
Median	0.67	0.059	0.44	4.48	1.69
Max	1.25	0.188	10.65	172.66	73.96

Experiments: Distribution Comparison

- Benchmark Case110
  - 0.75 positive literal weights
  - 0.25 negative literal weights
- 1 million samples