On the Sparsity of XORs in Approximate Model Counting

Durgesh Agrawal[1] Bhavishya[1] Kuldeep S. Meel[2]

[1]Indian Institute of Technology, Kanpur
[2]School of Computing, National University of Singapore

SAT 2020

Model Counting

- Given
  - Boolean variables $X_1, X_2, \cdots X_n$
  - Formula F over $X_1, X_2, \cdots X_n$
- $\mathsf{Sol}(F) = \{$ solutions of F $\}$

Model Counting

- Given
  - Boolean variables $X_1, X_2, \cdots X_n$
  - Formula F over $X_1, X_2, \cdots X_n$
- $\mathsf{Sol}(F) = \{$ solutions of F $\}$
- Model Counting: Determine $|\mathsf{Sol}(F)|$

Model Counting

- Given
  - Boolean variables $X_1, X_2, \cdots X_n$
  - Formula F over $X_1, X_2, \cdots X_n$
- $\mathrm{Sol}(F) = \{$ solutions of F $\}$
- Model Counting: Determine $|\mathrm{Sol}(F)|$
- Given $F := (X_1 \vee X_2)$

Model Counting

- Given
  - Boolean variables $X_1, X_2, \cdots X_n$
  - Formula F over $X_1, X_2, \cdots X_n$
- $\mathsf{Sol}(F) = \{$ solutions of F $\}$
- Model Counting: Determine $|\mathsf{Sol}(F)|$
- Given $F := (X_1 \vee X_2)$
- $\mathsf{Sol}(F) = \{(0,1),(1,0),(1,1)\}$

Model Counting

- Given
  - Boolean variables $X_1, X_2, \cdots X_n$
  - Formula F over $X_1, X_2, \cdots X_n$
- $\mathrm{Sol}(F) = \{$ solutions of F $\}$
- Model Counting: Determine $|\mathrm{Sol}(F)|$
- Given $F := (X_1 \vee X_2)$
- $\mathrm{Sol}(F) = \{(0,1),(1,0),(1,1)\}$
- $|\mathrm{Sol}(F)| = 3$

Model Counting

- Given
  - Boolean variables $X_1, X_2, \cdots X_n$
  - Formula F over $X_1, X_2, \cdots X_n$
- $\mathsf{Sol}(F) = \{$ solutions of F $\}$
- Model Counting: Determine $|\mathsf{Sol}(F)|$
- Given $F := (X_1 \vee X_2)$
- $\mathsf{Sol}(F) = \{(0,1),(1,0),(1,1)\}$
- $|\mathsf{Sol}(F)| = 3$
- Model counting is #P-complete (Valiant 1979)

Different Shades of Approximation

- Probabilistic $(1 + \varepsilon)$-Approximation

$$\Pr\left[\frac{|\mathsf{Sol}(F)|}{1+\varepsilon} \leq \mathit{ApproxCount}(F, \varepsilon, \delta) \leq |\mathsf{Sol}(F)|(1+\varepsilon)\right] \geq 1 - \delta$$

Different Shades of Approximation

- Probabilistic $(1+\varepsilon)$-Approximation

$$\Pr\left[\frac{|\mathsf{Sol}(F)|}{1+\varepsilon} \leq \mathit{ApproxCount}(F,\varepsilon,\delta) \leq |\mathsf{Sol}(F)|(1+\varepsilon)\right] \geq 1-\delta$$

- Constant Factor Approximation: $(4,\delta)$

$$\Pr\left[\frac{|\mathsf{Sol}(F)|}{4} \leq \mathit{ConstantCount}(F,\delta) \leq 4\cdot|\mathsf{Sol}(F)|\right] \geq 1-\delta$$

Different Shades of Approximation

- Probabilistic $(1+\varepsilon)$-Approximation

$$\Pr\left[\frac{|\mathsf{Sol}(F)|}{1+\varepsilon} \leq ApproxCount(F,\varepsilon,\delta) \leq |\mathsf{Sol}(F)|(1+\varepsilon)\right] \geq 1-\delta$$

- Constant Factor Approximation: $(4,\delta)$

$$\Pr\left[\frac{|\mathsf{Sol}(F)|}{4} \leq ConstantCount(F,\delta) \leq 4\cdot|\mathsf{Sol}(F)|\right] \geq 1-\delta$$

- From 4 to 2-factor

  Let $G = F_1 \wedge F_2$ (i.e., two identical copies of F)

$$\frac{|\mathsf{Sol}(G)|}{4} \leq C \leq 4\cdot|\mathsf{Sol}(G)| \implies \frac{|\mathsf{Sol}(F)|}{2} \leq \sqrt{C} \leq 2\cdot|\mathsf{Sol}(F)|$$

Different Shades of Approximation

- Probabilistic $(1+\varepsilon)$-Approximation

$$\Pr\left[\frac{|\mathrm{Sol}(F)|}{1+\varepsilon} \leq \mathit{ApproxCount}(F,\varepsilon,\delta) \leq |\mathrm{Sol}(F)|(1+\varepsilon)\right] \geq 1-\delta$$

- Constant Factor Approximation: $(4,\delta)$

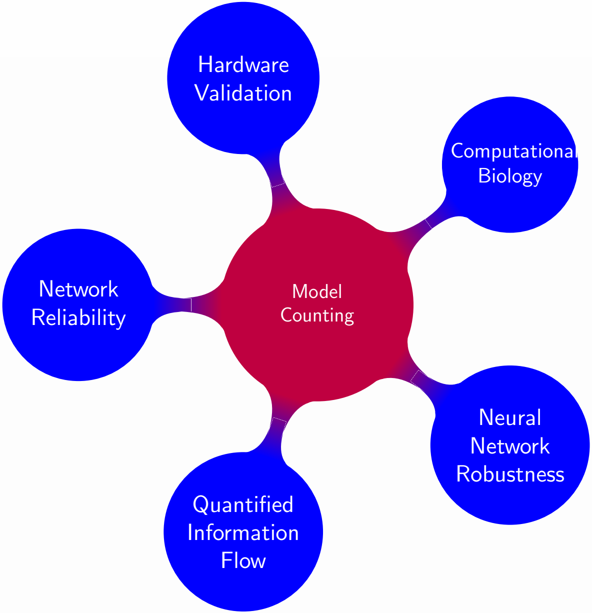
$$\Pr\left[\frac{|\mathrm{Sol}(F)|}{4} \leq \mathit{ConstantCount}(F,\delta) \leq 4\cdot|\mathrm{Sol}(F)|\right] \geq 1-\delta$$

- From 4 to 2-factor
  Let $G = F_1 \wedge F_2$ (i.e., two identical copies of F)

$$\frac{|\mathrm{Sol}(G)|}{4} \leq C \leq 4\cdot|\mathrm{Sol}(G)| \implies \frac{|\mathrm{Sol}(F)|}{2} \leq \sqrt{C} \leq 2\cdot|\mathrm{Sol}(F)|$$

- From 4 to $(1+\varepsilon)$-factor
  Construct $G = F_1 \wedge F_2 \ldots F_{\frac{1}{\varepsilon}}$ And then we can take $\frac{1}{\varepsilon}$-root

Applications across Computer Science

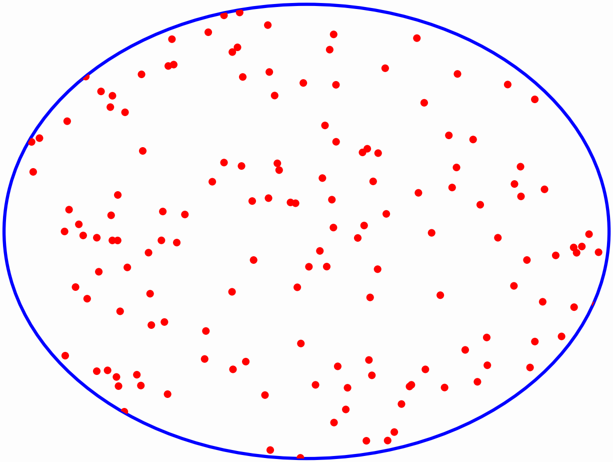


As Simple as Counting Dots

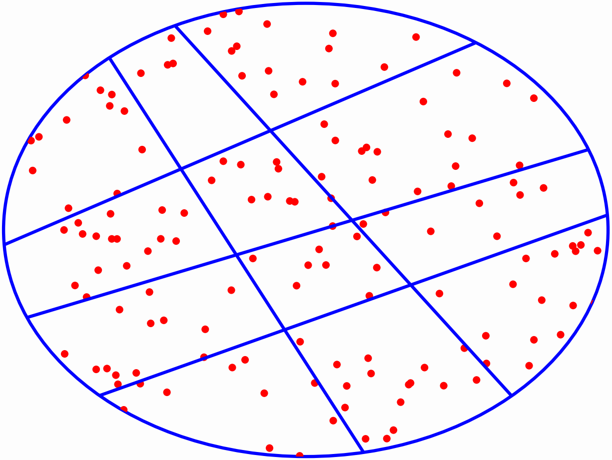
The Rise of Hashing-based Approach: Promise of Scalability and Guarantees

(S83,GSS06,GHSS07,CMV13b,EGSS13b,CMV14,CDR15,CMV16,ZCSE16,AD16 KM18,ATD18,SM19,ABM20,SGM20)

As Simple as Counting Dots

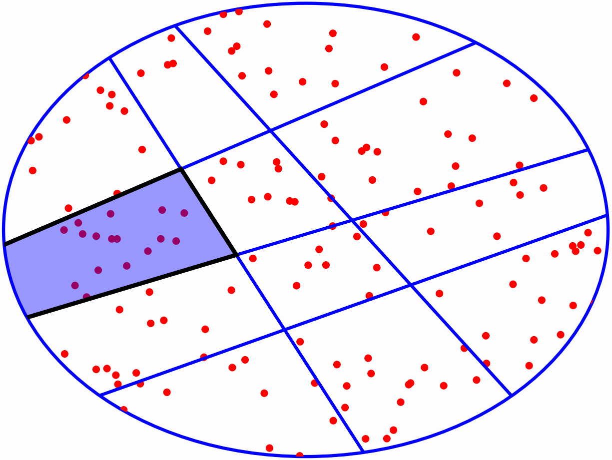


As Simple as Counting Dots



As Simple as Counting Dots

Pick a random cell



Estimate $=$ Number of solutions in a cell $\times$ Number of cells

Challenges

Challenge 1 What is exactly a *small cell* ?

Challenges

Challenge 1 What is exactly a *small cell* ?

Challenge 2 How to partition into roughly equal small cells of solutions without knowing the distribution of solutions?

Challenges

Challenge 1 What is exactly a *small cell* ?

- A cell is small cell if it has $\approx$ thresh solutions.
- Two choices for thresh.
  - thresh $= \mathrm{constant} \rightarrow$ 4-factor approximation
  - thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$ gives $(1+\varepsilon)$-approximation directly

Challenges

Challenge 1 What is exactly a *small cell* ?

- A cell is small cell if it has $\approx$ thresh solutions.
- Two choices for thresh.
  - thresh $= \text{constant} \rightarrow$ 4-factor approximation
  - thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$ gives $(1+\varepsilon)$-approximation directly
- Z_m be the number of solutions in a randomly chosen cell ; we are interested in cases $\mathsf{E}[Z_m] \geq 1$

Challenges

Challenge 1 What is exactly a *small cell* ?

- A cell is small cell if it has $\approx$ thresh solutions.
- Two choices for thresh.
  - thresh $= \mathrm{constant} \rightarrow$ 4-factor approximation
  - thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$ gives $(1+\varepsilon)$-approximation directly
- Z_m be the number of solutions in a randomly chosen cell ; we are interested in cases $\mathsf{E}[Z_m] \geq 1$
- For thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$, we need
  dispersion index: $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])} \leq \mathrm{some\ constant}$
- For thresh $= \mathrm{constant}$, sufficient to have
  coefficient of variation: $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq \mathrm{some\ constant}$

Challenges

Challenge 1 What is exactly a *small cell* ?

- A cell is small cell if it has $\approx$ thresh solutions.
- Two choices for thresh.
  - thresh $= \mathrm{constant} \rightarrow$ 4-factor approximation
  - thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$ gives $(1+\varepsilon)$-approximation directly
- Z_m be the number of solutions in a randomly chosen cell ; we are interested in cases $\mathsf{E}[Z_m] \geq 1$
- For thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$, we need dispersion index: $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])} \leq \mathrm{some\ constant}$
- For thresh $= \mathrm{constant}$, sufficient to have coefficient of variation: $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq \mathrm{some\ constant}$

Techniques based on thresh $= \mathcal{O}(\frac{1}{\varepsilon^2})$ such as ApproxMC scale significantly better than those based on thresh $= \mathrm{constant}$.

Challenges

Challenge 1 What is exactly a *small cell* ?

Challenge 2 How to partition into roughly equal small cells of solutions without knowing the distribution of solutions?

Challenges

Challenge 1 What is exactly a *small cell* ?

Challenge 2 How to partition into roughly equal small cells of solutions without knowing the distribution of solutions?

- Designing function h : assignments $\rightarrow$ cells (hashing)
- Solutions in a cell α: $\mathsf{Sol}(F) \cap \{y \mid h(y) = \alpha\}$

Challenges

Challenge 1 What is exactly a *small cell* ?

Challenge 2 How to partition into roughly equal small cells of solutions without knowing the distribution of solutions?

- Designing function h : assignments $\rightarrow$ cells (hashing)
- Solutions in a cell α: $\mathsf{Sol}(F) \cap \{y \mid h(y) = \alpha\}$
- Choose h randomly from a specially constructed large family H of hash functions

  Carter and Wegman 1977

2-wise Independent Hashing

- Let H be family of 2-wise independent hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$

$$\forall y_1, y_2 \in \{0,1\}^n, \alpha_1, \alpha_2 \in \{0,1\}^m, h \xleftarrow{R} H$$

$$\Pr[h(y_1) = \alpha_1] = \Pr[h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)$$

$$\Pr[h(y_1) = \alpha_1 \wedge h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)^2$$

2-wise Independent Hashing

- Let H be family of 2-wise independent hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$

$$\forall y_1, y_2 \in \{0,1\}^n, \alpha_1, \alpha_2 \in \{0,1\}^m, h \xleftarrow{R} H$$

$$\Pr[h(y_1) = \alpha_1] = \Pr[h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)$$

$$\Pr[h(y_1) = \alpha_1 \wedge h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)^2$$

- The power of 2-wise independence
  - Z_m be the number of solutions in a randomly chosen cell
  - $\mathsf{E}[Z_m] = \frac{|\mathsf{Sol}(F)|}{2^m}$
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m]$

2-wise Independent Hashing

- Let H be family of 2-wise independent hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$

$$\forall y_1, y_2 \in \{0,1\}^n, \alpha_1, \alpha_2 \in \{0,1\}^m, h \xleftarrow{R} H$$

$$\Pr[h(y_1) = \alpha_1] = \Pr[h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)$$

$$\Pr[h(y_1) = \alpha_1 \wedge h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)^2$$

- The power of 2-wise independence
  - Z_m be the number of solutions in a randomly chosen cell
  - $\mathsf{E}[Z_m] = \frac{|\mathsf{Sol}(F)|}{2^m}$
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m]$ (Interested only in the cases when $\mathsf{E}[Z_m] \geq 1$)

2-wise Independent Hashing

- Let H be family of 2-wise independent hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$

$$\forall y_1, y_2 \in \{0,1\}^n, \alpha_1, \alpha_2 \in \{0,1\}^m, h \xleftarrow{R} H$$

$$\Pr[h(y_1) = \alpha_1] = \Pr[h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)$$

$$\Pr[h(y_1) = \alpha_1 \wedge h(y_2) = \alpha_2] = \left(\frac{1}{2^m}\right)^2$$

- The power of 2-wise independence
  - Z_m be the number of solutions in a randomly chosen cell
  - $\mathsf{E}[Z_m] = \frac{|\mathsf{Sol}(F)|}{2^m}$
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m]$ (Interested only in the cases when $\mathsf{E}[Z_m] \geq 1$)
  - Z_m be the number of solutions in a randomly chosen cell
  - dispersion index: $\frac{\sigma^2[Z_m]}{\mathsf{E}[Z_m]} \leq 1$
  - coefficient of variation: $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq 1$

2-wise Independent Hashing

- Variables: $X_1, X_2, \cdots X_n$
- To construct $h : \{0,1\}^n \rightarrow \{0,1\}^m$, choose m random XORs
- Pick every X_i with prob. $\frac{1}{2}$ and XOR them
  - $X_1 \oplus X_3 \oplus X_6 \cdots \oplus X_{n-2}$
  - Expected size of each XOR: $\frac{n}{2}$

2-wise Independent Hashing

- Variables: $X_1, X_2, \cdots X_n$
- To construct $h : \{0,1\}^n \to \{0,1\}^m$, choose m random XORs
- Pick every X_i with prob. $\frac{1}{2}$ and XOR them
  - $X_1 \oplus X_3 \oplus X_6 \cdots \oplus X_{n-2}$
  - Expected size of each XOR: $\frac{n}{2}$
- To choose $\alpha \in \{0,1\}^m$, set every XOR equation to 0 or 1 randomly

$$X_1 \oplus X_3 \oplus X_6 \cdots \oplus X_{n-2} = 0 \tag{Q_1}$$
$$X_2 \oplus X_5 \oplus X_6 \cdots \oplus X_{n-1} = 1 \tag{Q_2}$$
$$\cdots \tag{$\cdots$}$$
$$X_1 \oplus X_2 \oplus X_5 \cdots \oplus X_{n-2} = 1 \tag{Q_m}$$

- Solutions in a cell: $F \wedge Q_1 \cdots \wedge Q_m$

2-wise Independent Hashing

- Variables: $X_1, X_2, \cdots X_n$
- To construct $h : \{0,1\}^n \rightarrow \{0,1\}^m$, choose m random XORs
- Pick every X_i with prob. $\frac{1}{2}$ and XOR them
  - $X_1 \oplus X_3 \oplus X_6 \cdots \oplus X_{n-2}$
  - Expected size of each XOR: $\frac{n}{2}$
- To choose $\alpha \in \{0,1\}^m$, set every XOR equation to 0 or 1 randomly

$$
\begin{aligned}
X_1 \oplus X_3 \oplus X_6 \cdots \oplus X_{n-2} &= 0 && (Q_1) \\
X_2 \oplus X_5 \oplus X_6 \cdots \oplus X_{n-1} &= 1 && (Q_2) \\
\cdots & && (\cdots) \\
X_1 \oplus X_2 \oplus X_5 \cdots \oplus X_{n-2} &= 1 && (Q_m)
\end{aligned}
$$

- Solutions in a cell: $F \wedge Q_1 \cdots \wedge Q_m$
- Performance of state of the art SAT solvers degrade with increase in the size of XORs (SAT Solvers != SAT oracles)

The Hope of Short XORs

- If we pick every variable X_i with probability p (Sparse: $p \ll \frac{1}{2}$; Dense: $p = \frac{1}{2}$)
  - Expected Size of each XOR: np
  - $\mathsf{E}[Z_m] = \frac{|\mathsf{Sol}(F)|}{2^m}$ (Interested only in the cases when $\mathsf{E}[Z_m] \geq 1$)

The Hope of Short XORs

- If we pick every variable X_i with probability p (Sparse: $p \ll \frac{1}{2}$; Dense: $p = \frac{1}{2}$)
  - Expected Size of each XOR: np
  - $\mathsf{E}[Z_m] = \frac{|\mathsf{Sol}(F)|}{2^m}$ (Interested only in the cases when $\mathsf{E}[Z_m] \geq 1$)
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \sum\limits_{\sigma_1 \in \mathsf{Sol}(F)} \sum\limits_{\substack{\sigma_2 \in \mathsf{Sol}(F) \\ w = d(\sigma_1, \sigma_2)}} r(w, p, m)$
    - where, $r(w, p, m) = \left(\left(\frac{1}{2} + \frac{(1-2p)^w}{2} \right)^m - \frac{1}{2^m} \right)$
  - For $p = \frac{1}{2}$, we have $\frac{\sigma^2[Z_m]}{\mathsf{E}[Z_m]} \leq 1$

The Core Technical Challenge

- Bounding the variance (EGSS14,ZCSE16)
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$

The Core Technical Challenge

- Bounding the variance (EGSS14,ZCSE16)
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$
  - $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq 1$ for $p = \mathcal{O}(\frac{\log m}{m})$

The Core Technical Challenge

- Bounding the variance (EGSS14,ZCSE16)
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$
  - $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq 1$ for $p = \mathcal{O}(\frac{\log m}{m})$
  - Sparse XORs can be used in techniques that first compute constant factor approximation
  - From linear to logarithmic size XORs!

The Core Technical Challenge

- Bounding the variance (EGSS14,ZCSE16)
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$
  - $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq 1$ for $p = \mathcal{O}(\frac{\log m}{m})$
  - Sparse XORs can be used in techniques that first compute constant factor approximation
  - From linear to logarithmic size XORs!

BUT Constant-factor approximation techniques don't scale

The Core Technical Challenge

- Bounding the variance (EGSS14,ZCSE16)
  - $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$
  - $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2} \leq 1$ for $p = \mathcal{O}(\frac{\log m}{m})$
  - Sparse XORs can be used in techniques that first compute constant factor approximation
  - From linear to logarithmic size XORs!

  BUT Constant-factor approximation techniques don't scale
- Can we use these bounds for techniques such as ApproxMC that compute $(1 + \varepsilon)$-approximation directly?

Our Contributions (1/2)

Explicit identification of the need for stronger bounds

- $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$ (EGSS14,ZCSE16)

Our Contributions (1/2)

Explicit identification of the need for stronger bounds

- $\sigma^2[Z_m] \leq \mathsf{E}[Z_m] + \eta$ (EGSS14,ZCSE16)
- We show $\eta \in \Omega((\mathsf{E}[Z_m])^2)$

Theorem (Informal)

The currently best known bounds on $\sigma^2[Z_m]$ are insufficient for techniques such as ApproxMC that have thresh $= \mathcal{O}(1/\varepsilon^2)$

- Open Problem: Derive better bounds

Our Contributions (2/2)

Do sparse XORs + constant factor techniques fare better than dense XOR-based techniques?

Our Contributions (2/2)

Do sparse XORs + constant factor techniques fare better than dense XOR-based techniques?

SparseCount2 SparseCount + Search Technique of CMV16

Our Contributions (2/2)

Do sparse XORs + constant factor techniques fare better than dense XOR-based techniques?

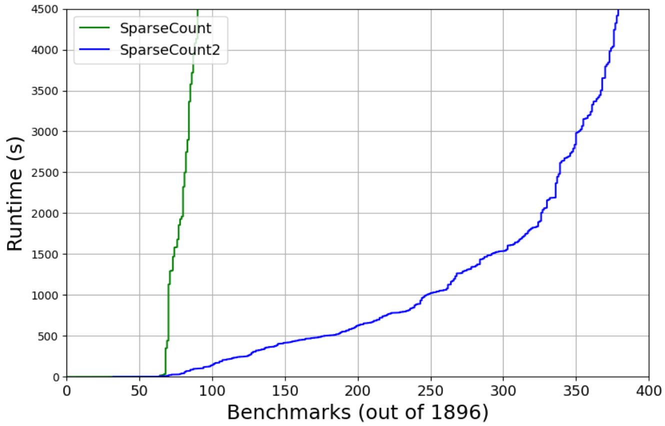
SparseCount2 SparseCount + Search Technique of CMV16

Fair comparison Same code base in C++ and identical underlying SAT solver: CryptoMiniSat

Parameters $\varepsilon = 0.8$, $\delta = 0.2$

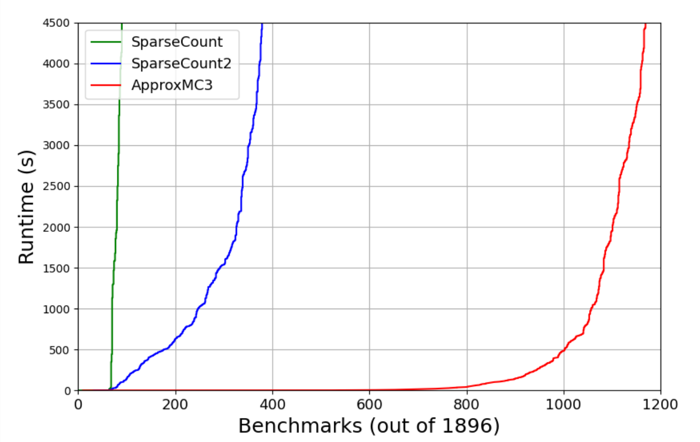
Our Contributions (2/2)

Do sparse XORs + constant factor techniques fare better than dense XOR-based techniques?



Our Contributions(2/2)

Do sparse XORs + constant factor techniques fare better than dense XOR-based techniques?



- The sparse XORs do make the SAT calls of SparseCount2 easier

Decoding the Surprise

- The sparse XORs do make the SAT calls of SparseCount2 easier
  BUT

Decoding the Surprise

- The sparse XORs do make the SAT calls of SparseCount2 easier
  BUT
- Not All SAT calls are Equal

Decoding the Surprise

- The sparse XORs do make the SAT calls of SparseCount2 easier BUT
- Not All SAT calls are Equal
  - 4-factor to $(1+\varepsilon)$ requires multiple copies; so SparseCount2 has to work with large formula
  - Weak bounds on $\frac{\sigma^2[Z_m]}{(\mathsf{E}[Z_m])^2}$ leads to larger number of SAT calls

Epilogue on Sparse XORs

- Low Density Parity Code-based Sparse XORs for approximate model counting (AT, SAT-16; ADT, SAT-18)
- Loss of theoretical guarantees

> *... the number of repetitions t explodes to over 1 million, making the derivation of rigorous* (ε, δ)*-approximations via Theorem 4 unrealistic.*

Conclusion

- The rise of hashing-based techniques for approximate counting
- The runtime of SAT solvers depend on the size of XORs

Conclusion

- The rise of hashing-based techniques for approximate counting
- The runtime of SAT solvers depend on the size of XORs
- Short XORs suffice

Conclusion

- The rise of hashing-based techniques for approximate counting
- The runtime of SAT solvers depend on the size of XORs
- Short XORs suffice in theory
  - In particular, there exist hashing-based techniques that can use short XORs

Conclusion

- The rise of hashing-based techniques for approximate counting
- The runtime of SAT solvers depend on the size of XORs
- Short XORs suffice in theory
  - In particular, there exist hashing-based techniques that can use short XORs
  - Such techniques are significantly inefficient (379 vs 1169 benchmarks)

Conclusion

- The rise of hashing-based techniques for approximate counting
- The runtime of SAT solvers depend on the size of XORs
- Short XORs suffice in theory
  - In particular, there exist hashing-based techniques that can use short XORs
  - Such techniques are significantly inefficient (379 vs 1169 benchmarks)

One last thing . . . some news to cheer you up

Conclusion

- The rise of hashing-based techniques for approximate counting
- The runtime of SAT solvers depend on the size of XORs
- Short XORs suffice in theory
  - In particular, there exist hashing-based techniques that can use short XORs
  - Such techniques are significantly inefficient (379 vs 1169 benchmarks)

One last thing . . . some news to cheer you up

- New bounds on $\sigma^2[Z_m]$ via Isoperimetric Inequalities

(Meel ⓡ Akshay, LICS 2020)

- Achieves $\frac{\sigma^2[Z_m]}{\mathsf{E}[Z_m]} \leq 1.1$ (=constant)
- Speedup over ApproxMC3 – The first instance of sparse XORs leading to runtime speedup over state of the art.