Discovering Frequent Patterns in Sensitive Data

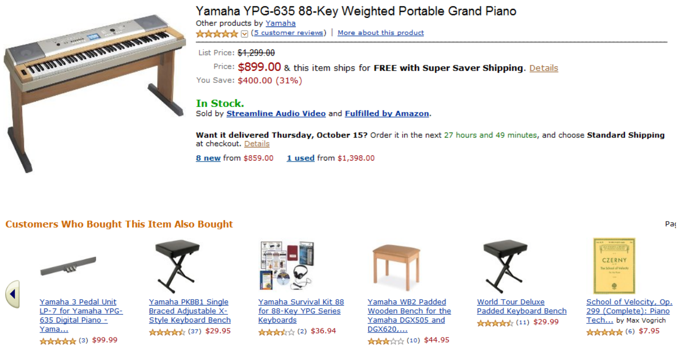
Abhradeep Guha Thakurta
azg161@cse.psu.edu

Department of Computer Science and Engineering
Pennsylvania State University

Joint work with Raghav Bhaskar and Srivatsan Laxman, Microsoft Research India and Adam Smith, Pennsylvannia State University

Frequent Pattern Mining (FPM)

- Widely used tool for exploratory data analysis
- Application: Recommendation systems (e.g. Amazon, Wal-Mart)



- Two variants of FPM:
  - Threshold: return all patterns with frequency above θ
  - Top-k: return k most frequent patterns

Top-*k* Frequent Pattern Mining (FPM)

- **Notation.**
  - U: Universe of patterns
  - T: Data set of n records
  - Frequency of a pattern

$$= \frac{\text{\# of records in which it appears}}{n}$$

  - **Output:** The k most frequent patterns in the data set T and their frequencies

EMR Data
427.9DX, 44140PX, 44120PX, 93503PX, 276.3DX, 518.5DX
373.2DX, 92002PX, 427.9DX, 410.91DX, 44120PX
573.9DX, 155.2DX, 276.3DX, 44120PX, 570DX
:
92002PX, 573.9DX, 427.9DX

FPM Output

PATTERN	FREQUENCY
427.9DX, 518.5DX	12345
427.9DX, 44120PX	12333
573.9DX, 276.3DX	12222
92002PX, 155.2DX	9876
373.2DX, 410.91DX	9777
:	:
155.2DX, 570DX	7654

Need for privacy

- The data set T may contain potentially sensitive information about an individual

Need for privacy

- The data set T may contain potentially sensitive information about an individual
- Want to protect the privacy of individual records in T
  - *e.g.*, Medical records
- Caution: Releasing exact results does not preserve privacy

Need for privacy

- The data set T may contain potentially sensitive information about an individual
- Want to protect the privacy of individual records in T
  - *e.g.*, Medical records
- Caution: Releasing exact results does not preserve privacy
  - *e.g.*, it is known that inverse FPM is NP-hard [Mie03]
    - Thus, it is hard to recover the entire data set
  - But it might be easy to recover specific pieces of information

Need for privacy

- The data set T may contain potentially sensitive information about an individual
- Want to protect the privacy of individual records in T
  - *e.g.*, Medical records
- Caution: Releasing exact results does not preserve privacy
  - *e.g.*, it is known that inverse FPM is NP-hard [Mie03]
    - Thus, it is hard to recover the entire data set
  - But it might be easy to recover specific pieces of information

Example of privacy breach for FPM

T_1

Data Set
427.9DX, 44140PX, 44120PX, 93503PX, 276.3DX, 518.5DX
373.2DX, 92002PX, 427.9DX, 410.91DX, 44120PX
573.9DX, 155.2DX, 276.3DX, 44120PX, 570DX
:
92002PX, 573.9DX, 427.9DX

PATTERN	FREQUENCY
427.9DX, 518.5DX	12345
427.9DX, 44120PX	12333
573.9DX, 276.3DX	12222
92002PX, 155.2DX	9876
373.2DX, 410.91DX	9777
:	:
155.2DX, 570DX	7654

T_2

Data Set
373.2DX, 92002PX, 427.9DX, 410.91DX, 44120PX
573.9DX, 155.2DX, 276.3DX, 44120PX, 570DX
:
92002PX, 573.9DX, 427.9DX

PATTERN	FREQUENCY
427.9DX, 518.5DX	12344
427.9DX, 44120PX	12332
573.9DX, 276.3DX	12222
92002PX, 155.2DX	9876
373.2DX, 410.91DX	9777
:	:
155.2DX, 570DX	7654

First row of T_1 must contain 427.9DX,518.5DX,44120PX

This work

- Provides algorithms for releasing high-frequency patterns while providing a rigorous privacy guarantee
  - We use differential privacy [DMNS06]

This work

- Provides algorithms for releasing high-frequency patterns while providing a rigorous privacy guarantee
  - We use differential privacy [DMNS06]
- Two algorithms:
  - Score perturbation-based algorithm (adapting [DMNS06])
  - Exponential sampling-based algorithm (adapting [MT07])

This work

- Provides algorithms for releasing high-frequency patterns while providing a rigorous privacy guarantee
  - We use differential privacy [DMNS06]
- Two algorithms:
  - Score perturbation-based algorithm (adapting [DMNS06])
  - Exponential sampling-based algorithm (adapting [MT07])
- Rigorous privacy and utility guarantees
- The experimental results support theoretical predictions

Differential Privacy

- Output should not reveal information about any individual record
- Informally, the output of FPM should not change by much by changing one record of T

Differential Privacy

- Output should not reveal information about any individual record
- Informally, the output of FPM should not change by much by changing one record of T

[DMNS06] A randomized algorithm $\mathcal{A}$ is ϵ-differentially private if for all data sets $T, T' \in \mathcal{D}^n$ differing in at most one record and for all events $\mathcal{O} \subseteq Range(\mathcal{A})$:

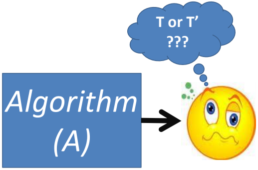
$$\Pr[\mathcal{A}(T) \in \mathcal{O}] \leq e^{\epsilon} \Pr[\mathcal{A}(T') \in \mathcal{O}]$$

T

Data Set
427.9DX, 44140PX, 44120PX, 93503PX, 276.3DX, 518.5DX
373.2DX, 92002PX, 427.9DX, 410.91DX, 44120PX
573.9DX, 155.2DX, 276.3DX, 44120PX, 570DX
:
92002PX, 573.9DX, 427.9DX

T'

Data Set
92002PX, 570DX, 427.9DX
373.2DX, 92002PX, 427.9DX, 410.91DX, 44120PX
573.9DX, 155.2DX, 276.3DX, 44120PX, 570DX
:
92002PX, 573.9DX, 427.9DX



Why differential privacy?

- Protects against arbitrary side information
  - Adversary learns the same thing whether or not Alice's record was there in the data set

Why differential privacy?

- Protects against arbitrary side information
  - Adversary learns the same thing whether or not Alice's record was there in the data set
- Protects against attacks like re-identification, attribute linkage etc

Why differential privacy?

- Protects against arbitrary side information
  - Adversary learns the same thing whether or not Alice's record was there in the data set
- Protects against attacks like re-identification, attribute linkage etc
- Widely studied since 2006

Why differential privacy?

- Protects against arbitrary side information
  - Adversary learns the same thing whether or not Alice's record was there in the data set
- Protects against attacks like re-identification, attribute linkage etc
- Widely studied since 2006
- Differentially private algorithms exist for
  - learning [BDMN05,KLNRS08], statistical inference [DL09], recommendation systems [MM09]

Related work [AH05],[EGS03]

- Randomized response: Each entry in the data set T is independently randomized before allowing data mining algorithm to access it

Related work [AH05],[EGS03]

- Randomized response: Each entry in the data set T is independently randomized before allowing data mining algorithm to access it
- [AH05],[EGS03] considered randomized response in the context of FPM

Related work [AH05],[EGS03]

- Randomized response: Each entry in the data set T is independently randomized before allowing data mining algorithm to access it
- [AH05],[EGS03] considered randomized response in the context of FPM
  - Work of [AH05] is a generalization of [EGS03]
  - Privacy guarantees are equivalent to differential privacy

Related work [AH05],[EGS03]

- Randomized response: Each entry in the data set T is independently randomized before allowing data mining algorithm to access it
- [AH05],[EGS03] considered randomized response in the context of FPM
  - Work of [AH05] is a generalization of [EGS03]
  - Privacy guarantees are equivalent to differential privacy
  - No formal utility guarantees

Related work [AH05],[EGS03]

- Randomized response: Each entry in the data set T is independently randomized before allowing data mining algorithm to access it
- [AH05],[EGS03] considered randomized response in the context of FPM
  - Work of [AH05] is a generalization of [EGS03]
  - Privacy guarantees are equivalent to differential privacy
  - No formal utility guarantees
  - Our algorithms perform consisently better (in experiments)

Need for approximate utility

- By definition, any non-trivial differentially private algorithm has to introduce error in the output

Need for approximate utility

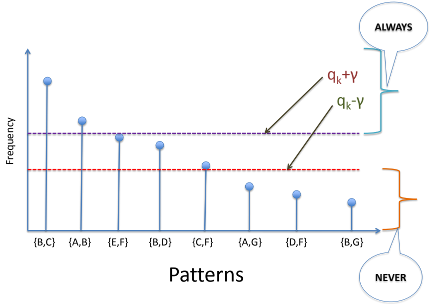
- By definition, any non-trivial differentially private algorithm has to introduce error in the output
- Differentially private FPM will
  - insert low frequency patterns in the output
  - remove high frequency patterns from the output
  - perturb the frequencies of the patterns being output
- An "useful" FPM output should have small error

Need for approximate utility

- By definition, any non-trivial differentially private algorithm has to introduce error in the output
- Differentially private FPM will
  - insert low frequency patterns in the output
  - remove high frequency patterns from the output
  - perturb the frequencies of the patterns being output
- An “useful” FPM output should have small error
- To quantify utility, we
  - introduce a notion of “approximate” top frequent patterns
  - evaluate our algorithms both theoretically and empirically with respect to this notion

Approximate utility for FPM

Let q_k be the k^{th} highest frequency based on data set T

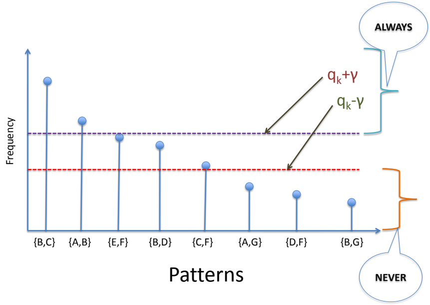


An FPM output is (γ, η)-useful if:

- *(Soundness)* No pattern in the output has frequency less than $(q_k - \gamma)$

Approximate utility for FPM

Let q_k be the k^{th} highest frequency based on data set T

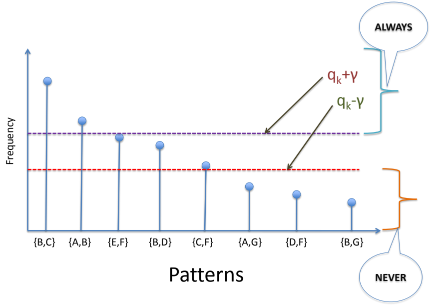


An FPM output is (γ, η)-useful if:

- *(Soundness)* No pattern in the output has frequency less than $(q_k - \gamma)$
- *(Completeness)* Every pattern with frequency greater than $(q_k + \gamma)$ is in the output

Approximate utility for FPM

Let q_k be the k^{th} highest frequency based on data set T

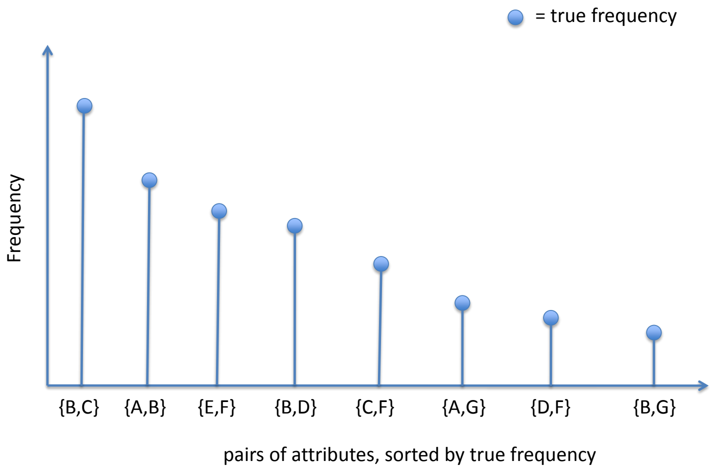


An FPM output is (γ, η)-useful if:

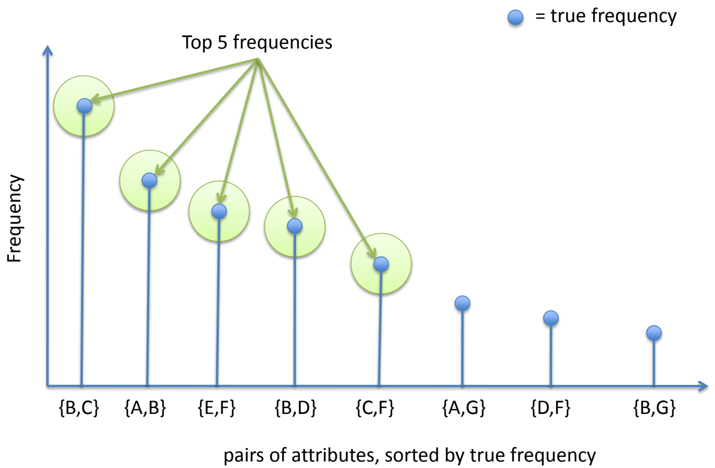
- *(Soundness)* No pattern in the output has frequency less than $(q_k - \gamma)$
- *(Completeness)* Every pattern with frequency greater than $(q_k + \gamma)$ is in the output
- *(Precision)* The reported frequency for every pattern in the output is within η of its true frequency

Score perturbation-based algorithm

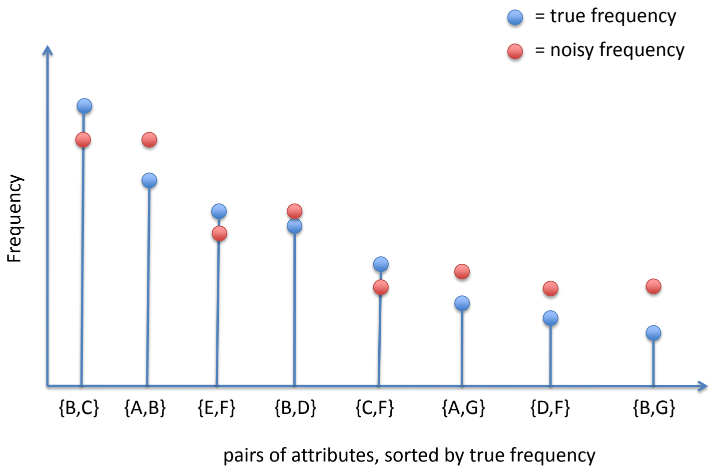
Score perturbation-based algorithm



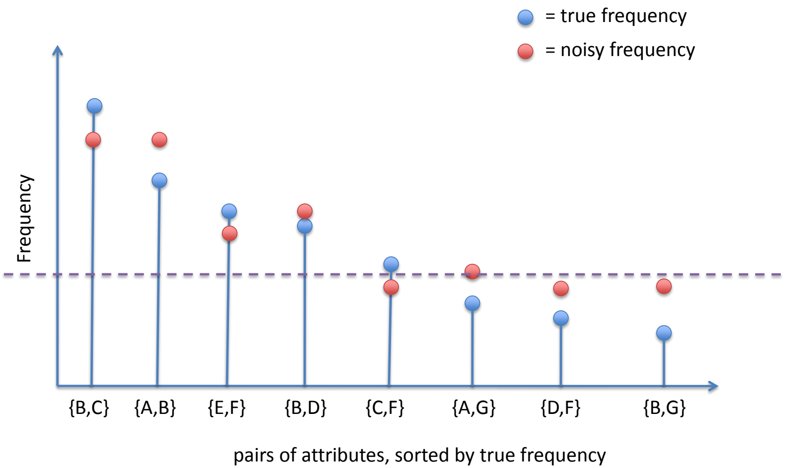
Score perturbation-based algorithm



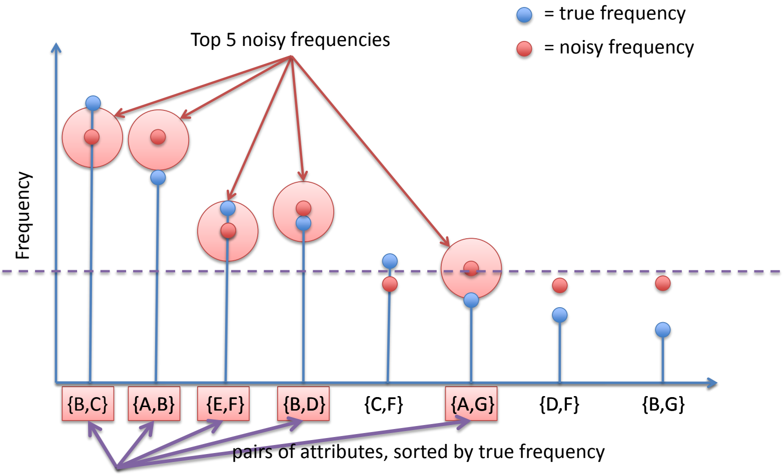
Score perturbation-based algorithm



Score perturbation-based algorithm



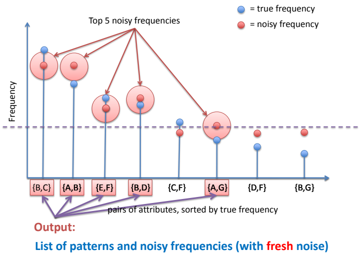
Score perturbation-based algorithm



Output:

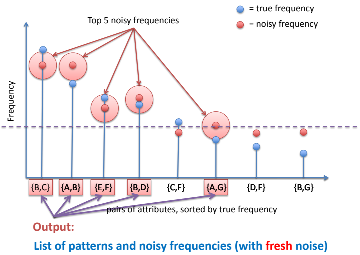
List of patterns and noisy frequencies (with fresh noise)

Details of the algorithm



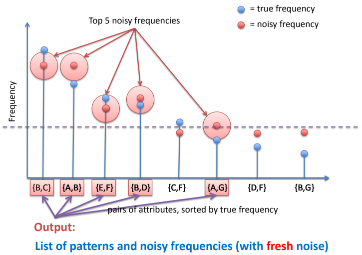
- How much noise?
  - Laplace noise with $\lambda = \Theta\left(\frac{k}{\epsilon n}\right)$
  - $Lap(\lambda) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}}$

Details of the algorithm



- How much noise?
  - Laplace noise with $\lambda = \Theta\left(\frac{k}{\epsilon n}\right)$
  - $Lap(\lambda) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}}$
- Straightforward implementation needs time $O(|U|)$
  - Might be exponentially large
  - *e.g.*, Frequent Itemset Mining: m items $\rightarrow 2^m$ itemsets

Details of the algorithm



- How much noise?
  - Laplace noise with $\lambda = \Theta\left(\frac{k}{\epsilon n}\right)$
  - $Lap(\lambda) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}}$
- Straightforward implementation needs time $O(|U|)$
  - Might be exponentially large
  - *e.g.*, Frequent Itemset Mining: m items $\rightarrow 2^m$ itemsets
- Our implementation takes time "roughly" $\propto k$

Analysis (Privacy)

Theorem: *The algorithm is ϵ-differentially private*

Analysis (Privacy)

Theorem: *The algorithm is ϵ-differentially private*

- **Naive analysis:**
  - Consider the frequencies of $|U|$ patterns as a vector of length $|U|$
  - Assure privacy for each element of the vector individually using [DMNS06] style analysis
  - Requires $\Theta\left(\frac{|U|}{\epsilon n}\right)$ noise for ϵ-differential privacy
- **Our analysis:** $\Theta\left(\frac{k}{\epsilon n}\right)$ noise suffices

Analysis (Performance)

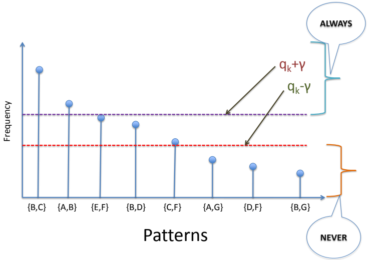
Theorem (Utility): *For all $\rho > 0$: with probability at least $1 - \rho$, the output is (γ, η)-useful, where*

$$\gamma = \frac{8k}{\epsilon n}\left(\log \frac{|U|}{\rho}\right)$$

and

$$\eta = \frac{2k}{n\epsilon} \ln\left(\frac{k}{\rho}\right)$$

Take away: Privacy does not degrade the utility by too much

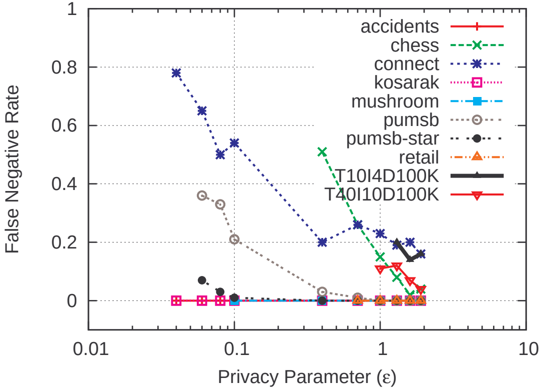


Experimental results (Frequent Itemset Mining)

- All the data sets from the FIMI repository (`http://fimi.cs.helsinki.fi/`)
- Accurate results for a wide range of parameters ($k, \epsilon, \gamma, \rho$)
- Error rates match theoretical predictions
- This talk: variation of FNR (False Negative Rate) with ϵ
  - Note that False Positive Rate is not an effective measure of utility because the # of true negatives is inherently high

Score perturbation-based algorithm: Variation of FNR vs ϵ

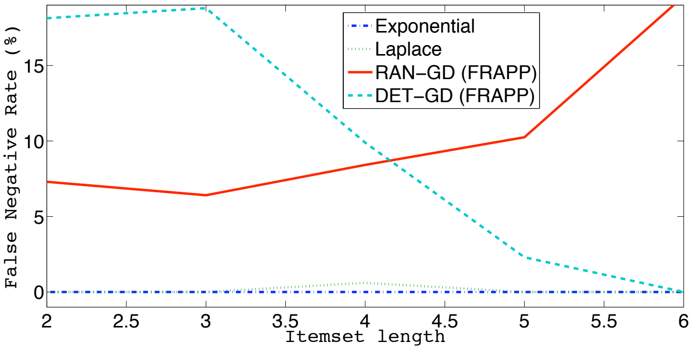
Parameters: $\rho = 0.1, k = 10$ and the size of the itemsets mined$= 3$



(N.B. Av. transaction length: Connect:44,Kosarak: 8.09)

Randomized response [AH05]

- [AH05] introduces the FRAPP framework
  - DET-GD and RAN-GD are two algorithms under the FRAPP framework
- Use the CENSUS data set used by [AH05]



Conclusion

- This work:
  - First work towards providing both formal privacy and utility guarantees for FPM

Conclusion

- This work:
  - First work towards providing both formal privacy and utility guarantees for FPM
  - Two algorithms which provide a strong notion privacy and are accurate on a wide range of data sets

Conclusion

- This work:
  - First work towards providing both formal privacy and utility guarantees for FPM
  - Two algorithms which provide a strong notion privacy and are accurate on a wide range of data sets
  - Far more accurate than previous, randomized-response algorithms

Conclusion

- This work:
  - First work towards providing both formal privacy and utility guarantees for FPM
  - Two algorithms which provide a strong notion privacy and are accurate on a wide range of data sets
  - Far more accurate than previous, randomized-response algorithms
  - Our algorithms are also useful for the more general problem of private ranking [KKMN09, GMW+09]

Conclusion

- This work:
  - First work towards providing both formal privacy and utility guarantees for FPM
  - Two algorithms which provide a strong notion privacy and are accurate on a wide range of data sets
  - Far more accurate than previous, randomized-response algorithms
  - Our algorithms are also useful for the more general problem of private ranking [KKMN09, GMW+09]
- In the paper:
  - Another algorithm: Exponential sampling-based
  - Implementation details for both the algorithms
  - Comprehensive experimental results

Conclusion

- This work:
  - First work towards providing both formal privacy and utility guarantees for FPM
  - Two algorithms which provide a strong notion privacy and are accurate on a wide range of data sets
  - Far more accurate than previous, randomized-response algorithms
  - Our algorithms are also useful for the more general problem of private ranking [KKMN09, GMW+09]
- In the paper:
  - Another algorithm: Exponential sampling-based
  - Implementation details for both the algorithms
  - Comprehensive experimental results
- Open Problem:Can we have differentially private algorithms for other high dimensional problems?

References

Shipra Agrawal and Jayant R. Haritsa.
A framework for high-accuracy privacy-preserving mining.
In *ICDE*, pages 193–204, 2005.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith.
Calibrating noise to sensitivity in private data analysis.
In *TCC*, pages 265–284, 2006.

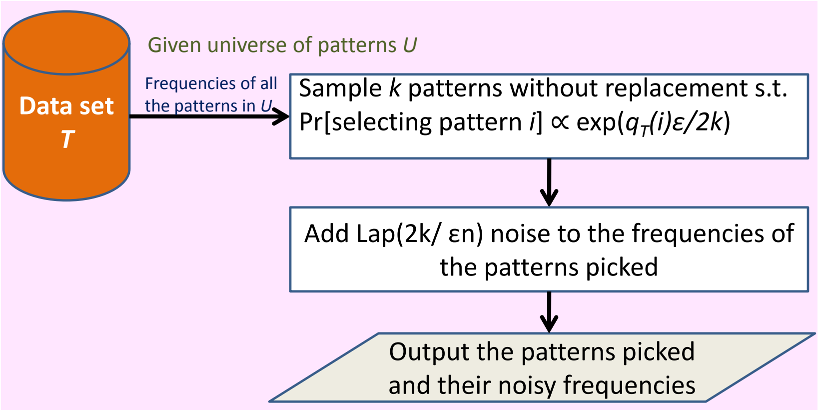
Michaela Götz, Ashwin Machanavajjhala, Guozhang Wang, Xiaokui Xiao, and Johannes Gehrke.
Privacy in search logs.
CoRR, abs/0904.0682, 2009.

Aleksandra Korolova, Krishnaram Kenthapadi, Nina Mishra, and Alexandros Ntoulas.
Releasing search queries and clicks privately.
In *WWW*, pages 171–180, 2009.

Taneli Mielikäinen.
On inverse frequent set mining.
In *2nd Workshop on Privacy Preserving Data Mining (PPDM 2003)*, pages 18–23. IEEE Computer Society, 2003.

Frank McSherry and Kunal Talwar.
Mechanism design via differential privacy.
In *FOCS*, pages 94–103, 2007.

Exponential sampling-based algorithm



Analysis

- The privacy guarantee is same as score perturbation-based algorithm
- The utility guarantee is better by a small constant factor
- The algorithm runs in $O(|U| \log^* |U|)$

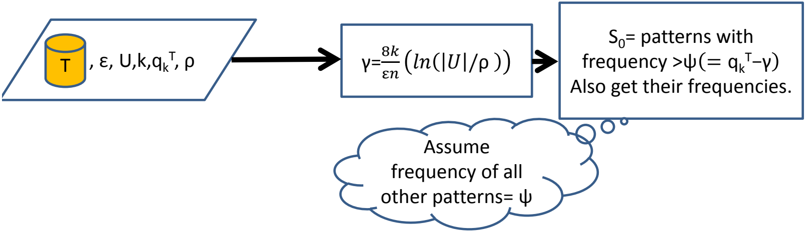
Exponential sampling-based algorithm: Running time on various data sets

Data sets	FIM (ms)	*Exp Mech* (ms)		
		$\frac{\varepsilon}{2} = 0.06$	$\frac{\varepsilon}{2} = 0.7$	$\frac{\varepsilon}{2} = 1.3$
accidents	897	878(1.0)	875(1.0)	895(1.0)
chess	61	-	77(1.3)	89(1.4)
connect	273	364(1.3)	284(1.0)	300(1.1)
kosarak	1077	1073(1.0)	1084(1.0)	1058(0.98)
mush	105	10542(100.1)	78(0.8)	125(1.2)
pumsb	386	834(2.2)	393(1.0)	389(1.0)
pumsb*	288	317(1.1)	288(1.0)	289(1.0)
retail	150	-	183(1.2)	172(1.2)
T10	530	-	6912(13.1)	1339(2.5)
T40	6191	-	33006(5.3)	14190(2.3)

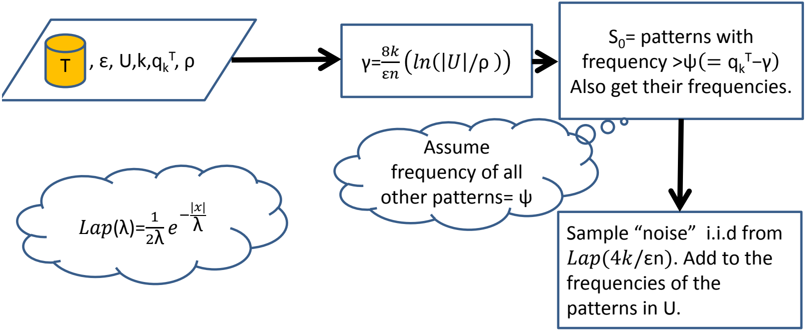
mush=mushroom, pumsb*=pumsb-star, T10=T10I4D100K, T40=T40I10D100K

Table: Run-time overhead due to privacy step

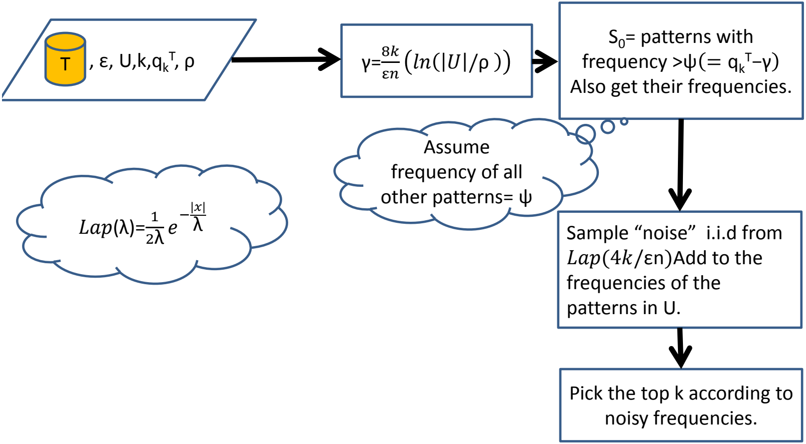
Score perturbation-based algorithm



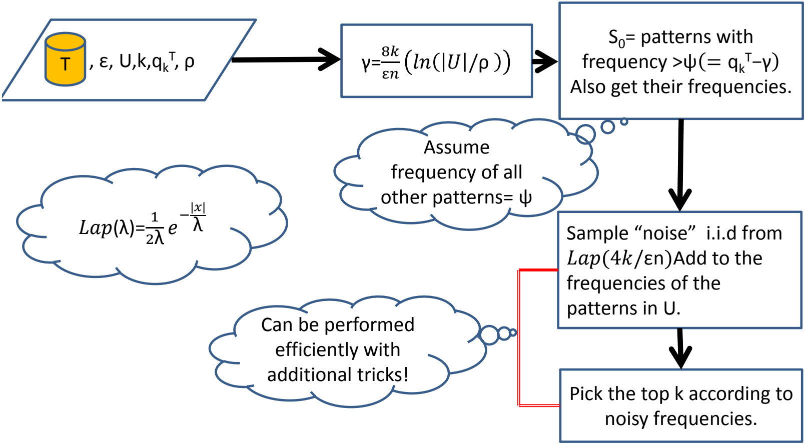
Score perturbation-based algorithm



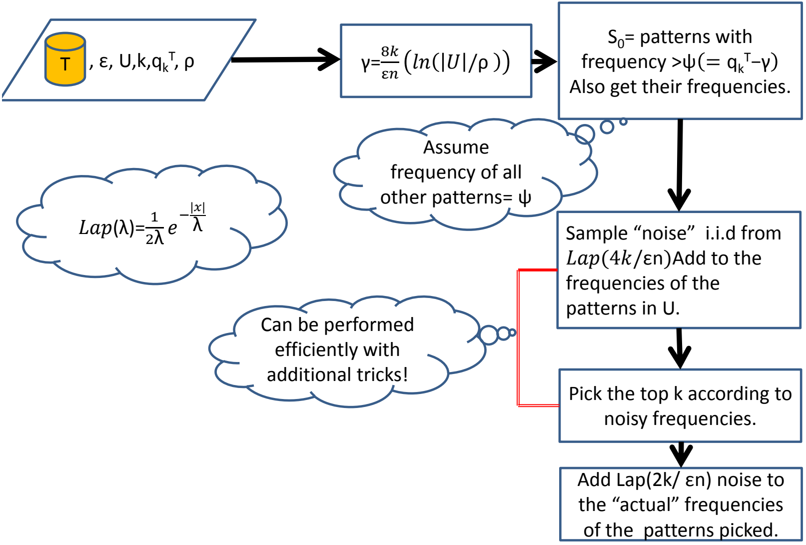
Score perturbation-based algorithm



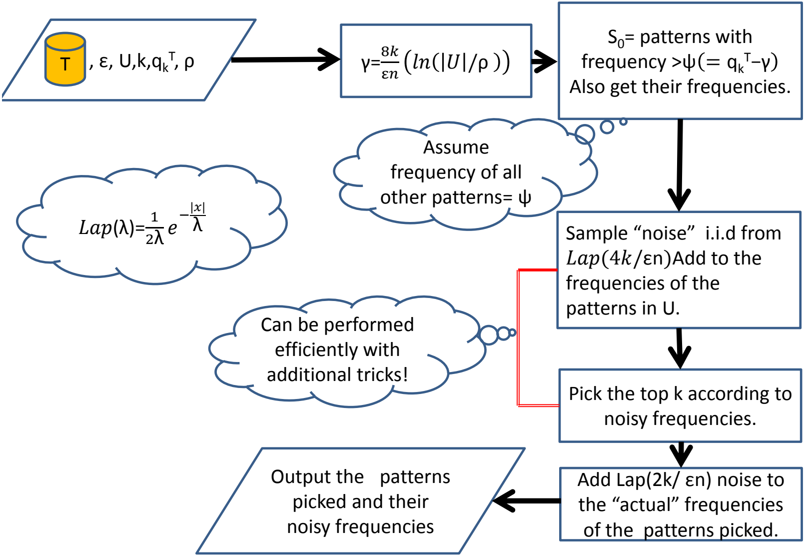
Score perturbation-based algorithm



Score perturbation-based algorithm

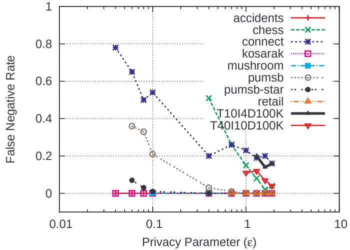


Score perturbation-based algorithm

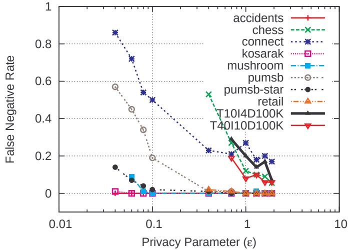


Two algorithms: Variation of FNR vs ϵ

Parameters: $\rho = 0.1$, $k = 10$ and the size of the itemsets mined$= 3$



(g) Score perturbation-based



(h) Exponential sampling-based