Secure Sublinear Time
Differentially Private Median Computation

Jonas Böhler (SAP Security Research)
Florian Kerschbaum (University of Waterloo)
NDSS 2020

Motivation & Preliminaries

Distributed Private Learning

Parties with **confidential data** want to learn **statistics over joint data** while **preserving privacy**

- Real-world examples
  - Ad conversions: link online ads with offline purchases
    - Google & Mastercard [B18]

  - Tax fraud detection
    - Estonian Tax and Customs Board (MTA) & Companies [BJSV15]

  - Studies
    - MTA & Estonian Ministry of Education and Research [BKKRST16]
    - Boston Women's Workforce Council & Employers [LVBJV16]

Our focus: **2 semi-honest parties** computing **rank-based statistics**, especially the **median**

Why rank-based statistics & median?

Rank of a value w.r.t. a data set D: *first* position in sorted data (zero-indexed)

Data set D	0	1	1	2	3	5	8	13	21	34
Rank	0	1		3	4	5	6	7	8	9

Rank-based statistics: versatile & robust

- min
- max
- In general, k^{th}-ranked element (p^{th}-percentile)
  - **median**
    - “typical value” in data
    - more **robust** to outliers than mean

Example: income in Medina, Washington

- Population ≈3,000
- **Median Income** ≈ $186,000
- **Average Income** ≫$1,000,000,000
  - „outliers“ Jeff Bezos and Bill Gates

Why Differential Privacy (DP) and Secure Compuation?

Median is one value from the data

- no protection (not even aggregation), we want **privacy guarantee**

ϵ**-DP** is a strong **privacy guarantee** used by Google, Microsoft, Apple

- restricts output differences when input changes in one element
- implementation models
  – with trusted third party (better accuracy)
  – without trusted party (better privacy)
- we want **high accuracy without trusted third party**

Secure Computation

- „simulate“ trusted party with cryptography

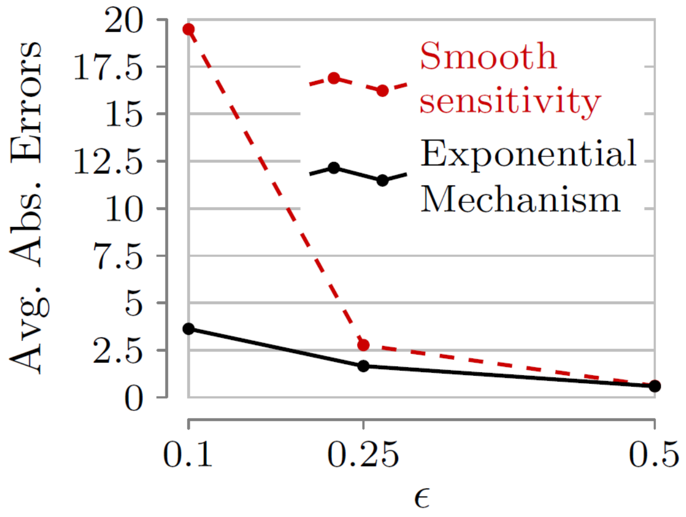
Why use the Exponential Mechanism?

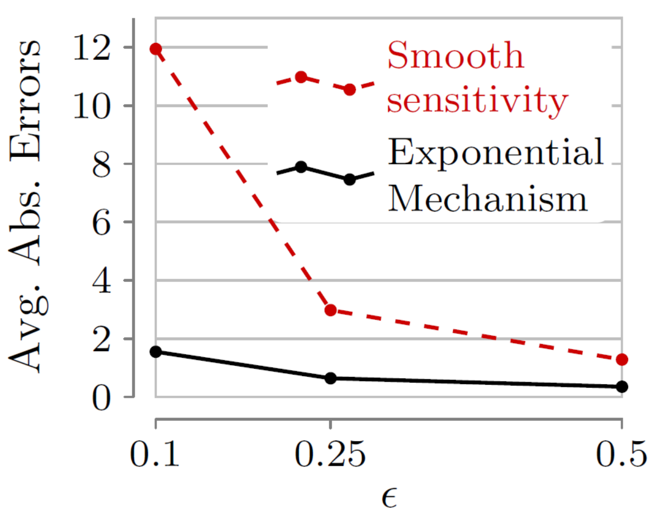
Differential Privacy techniques

Additive Noise	*Probabilistic Selection*
▪ E.g., *Laplace mechanism* [DR14] – outputs $\mathrm{median}(D) + Laplace\left(\frac{\Delta_s}{\epsilon}\right)$ ▫ *Smooth sensitivity* Δ_s [NRS07]: analyzes data D to reduce noise – Δ_s computation time linear in data size	▪ *Exponential mechanism* [MT07] – outputs m with probability $\propto \exp\left(\epsilon \frac{u(D,m)}{2\Delta u}\right)$ ▫ *Utility function* u: „scores" closeness to median – Linear in size of the data domain

Exponential mechanism for the median

- achieves **high accuracy** for low ϵ
  - low avg. absolute error
    - Credit card transactions (left)
    - Walmart shipment weights (right)
- we provide **sublinear-time** computation



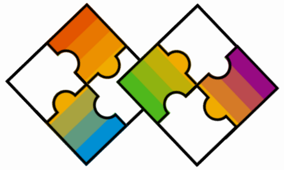


Secure Computation Basics

Secure Computation techniques

Secret Sharing [Shamir79]

- Efficient for arithmetic computations, e.g.,
  - addition / subtraction
  - scalar multiplication



Garbled Circuits [Yao86]

- Efficient for Boolean computations, e.g.,
  - comparison

We use both techniques and optimize our protocol for their respective advantages

- Implemented with **ABY** [DSZ15]
  - 2PC with *secret sharing* and *garbled circuits*
  - provides conversions between techniques

Secure Exponential Mechanism for Median

Overview

Exponential mechanism outputs domain element m with probability $\propto \exp\left(\epsilon \cdot u(D,m)\right)$

- running time linear in size of data domain
- costly secure exponentiation

Large data domain? We use sorted data.

- *sorted unique* data has *data-independent* utility scores
  – extendable to *non-unique* data and entire data domain
- Large data size? We prune data.
  – iterative pruning, preserves the median [AMP10]
    - **DP-relaxation**: prune-neighbors [HMFS17]

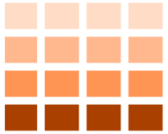
Costly secure exponentiation? We don‘t need it.

- *data-independent* utility function → *data-independent* exponentiation

Step by step

Party A

Party B



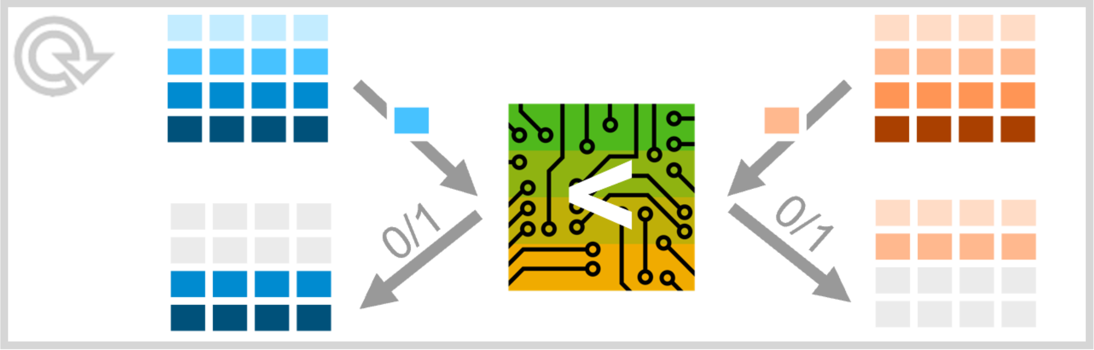
Parties A, B with sorted data

Step by step

Parties A, B with sorted data

- (I) *Prune large data* [AMP10]
  - parties compare *their* medians, keep half of data with *mutual* median

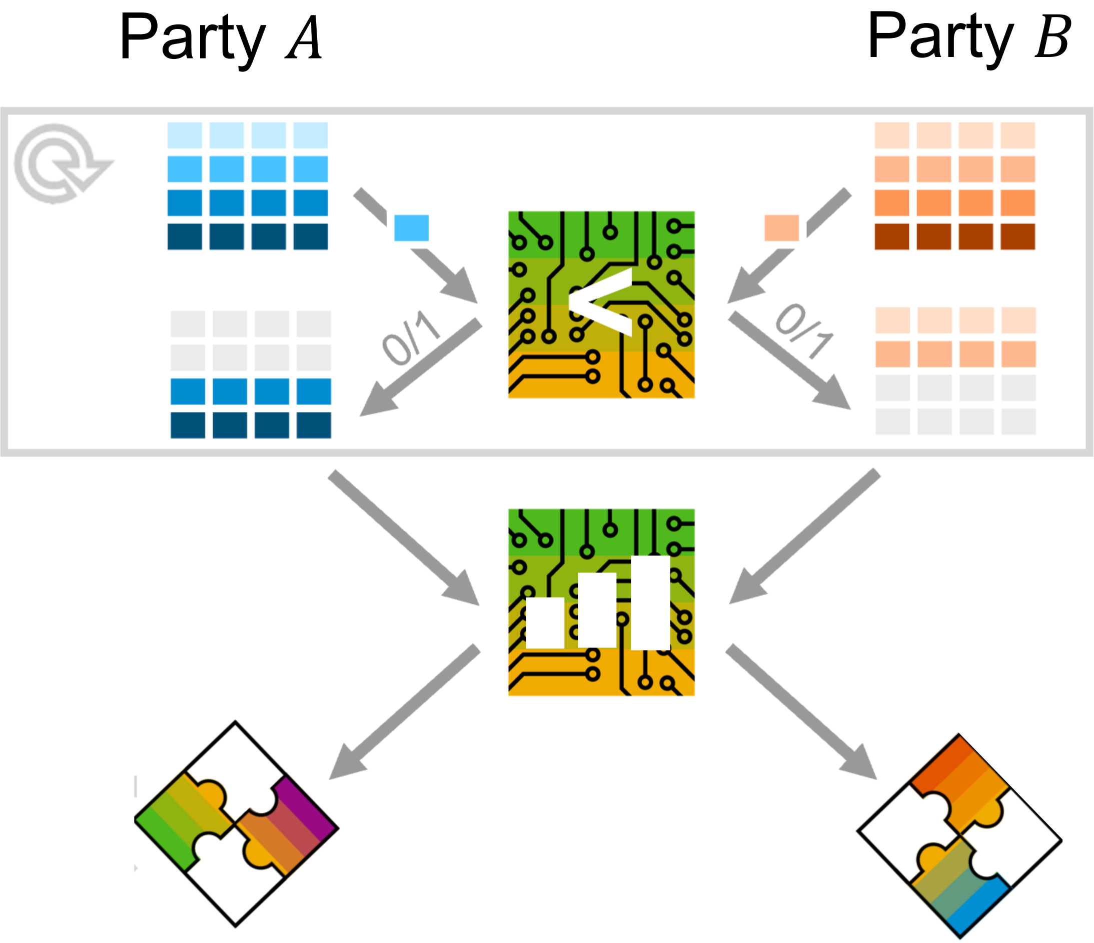
Party A Party B



Step by step

Parties A, B with sorted data

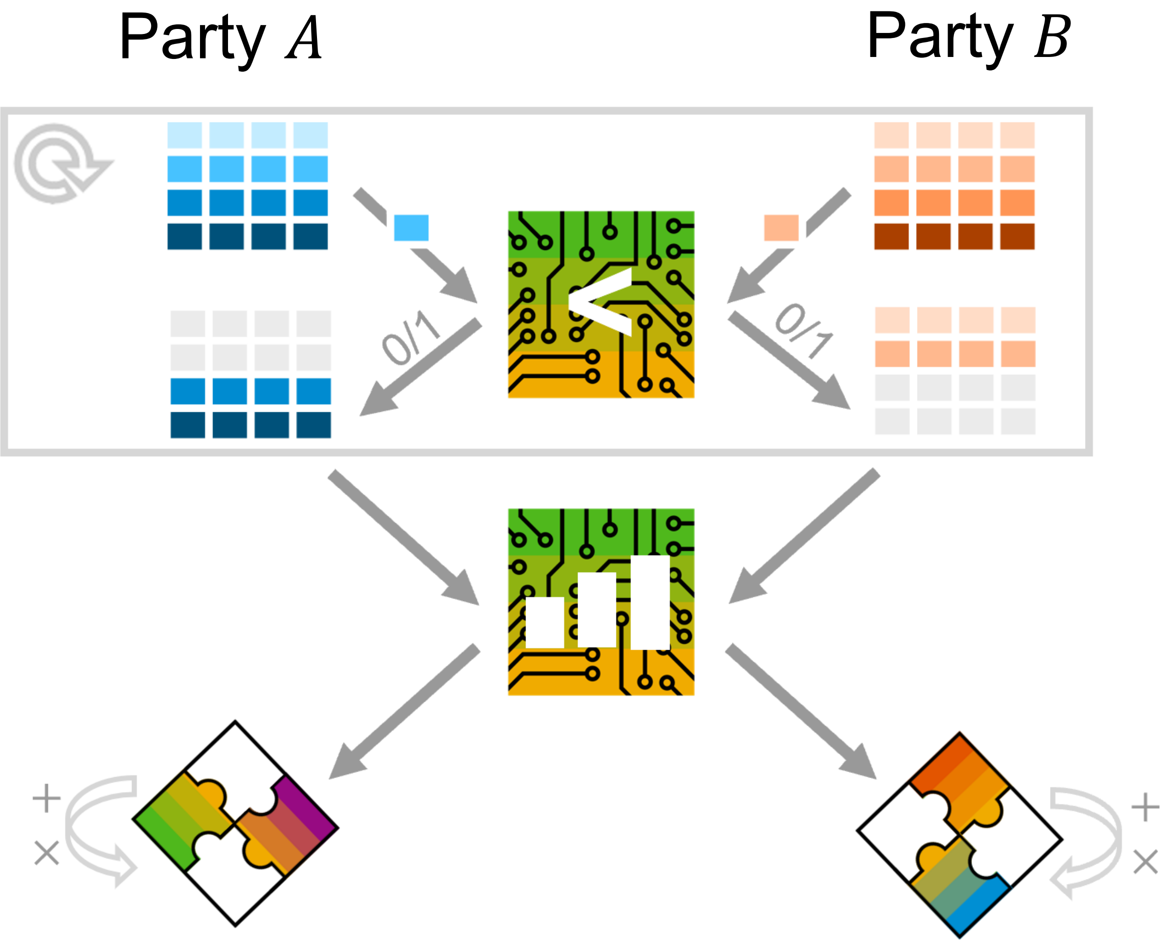
- (I) *Prune large data* [AMP10]
  - parties compare *their* medians, keep half of data with *mutual* median
- (II) *Oblivious Merge* [HEK12] & *Secret Share*
  - merge faster than sort
  - secret sharing for efficient arithmetics



Step by step

Parties A, B with sorted data

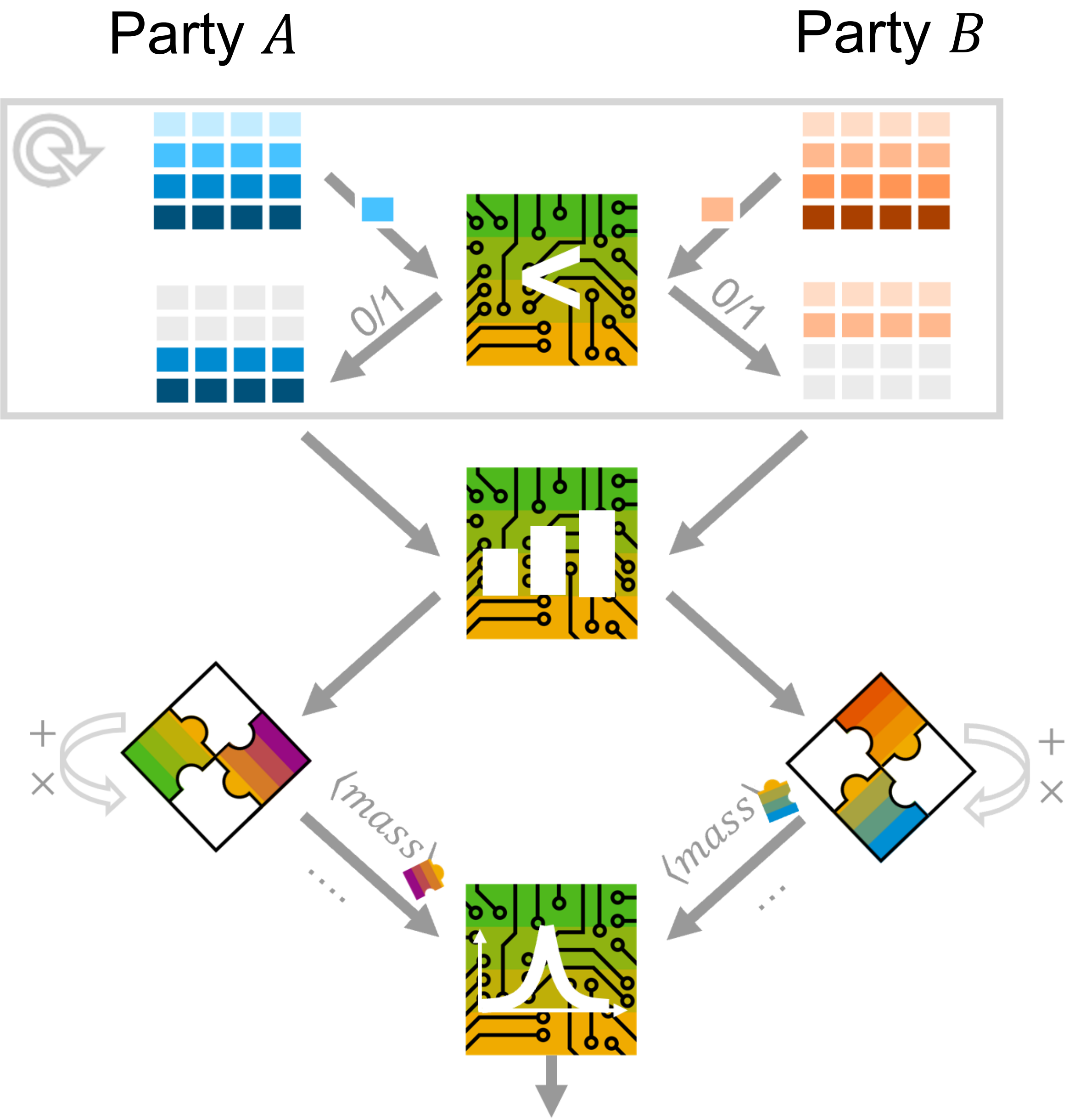
- (I) *Prune large data* [AMP10]
  - parties compare *their* medians, keep half of data with *mutual* median
- (II) *Oblivious Merge* [HEK12] & *Secret Share*
  - merge faster than sort
  - secret sharing for efficient arithmetics
- (III) *Selection Probability*
  - $\langle mass(j) \rangle = \sum_{i<j} \exp(u'(i)\epsilon) \cdot \langle gap(i) \rangle$
    - u': simplified utility function
    - gap: count of consecutive elements with same utility



Step by step

Parties A, B with sorted data

- (I) *Prune large data* [AMP10]
  - parties compare *their* medians, keep half of data with *mutual* median
- (II) *Oblivious Merge* [HEK12] & *Secret Share*
  - merge faster than sort
  - secret sharing for efficient arithmetics
- (III) *Selection Probability*
  - $\langle mass(j)\rangle = \sum_{i<j} \exp(u'(i)\epsilon) \cdot \langle gap(i)\rangle$
    - u': simplified utility function
    - gap: count of consecutive elements with same utility
- (IV) *Median Selection*
  - Select output via CDF

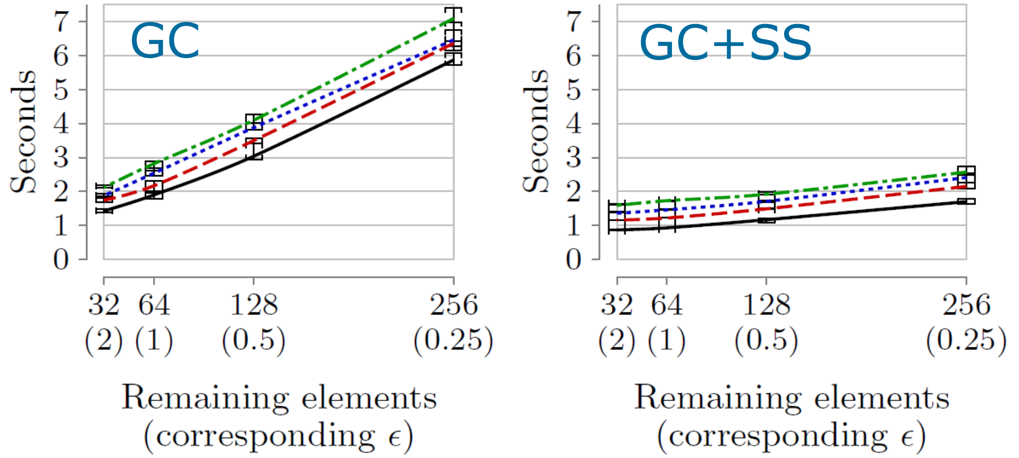


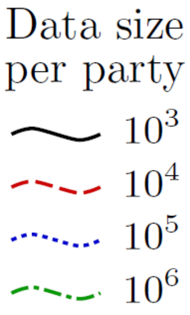
Evaluation

Running time in WANs

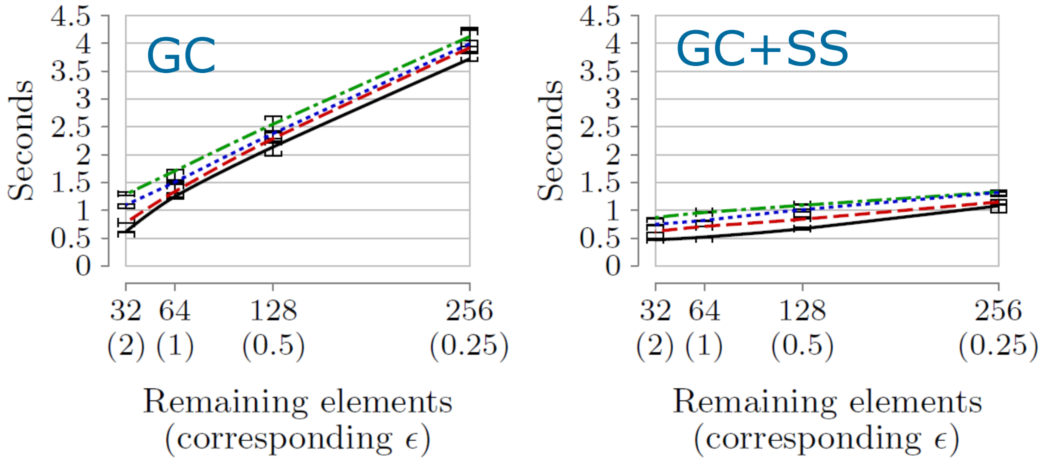
- Evaluated in different AWS regions
  - runs in seconds for millions of records
  - with real-world latency, bandwidth
  - on t2.medium instances
- Two versions:
  - GC: only garbled circuits
  - GC+SS: garbled circuits & secret sharing

~25 ms RTT, ~160 MBits/s *(Ohio and Canada)*

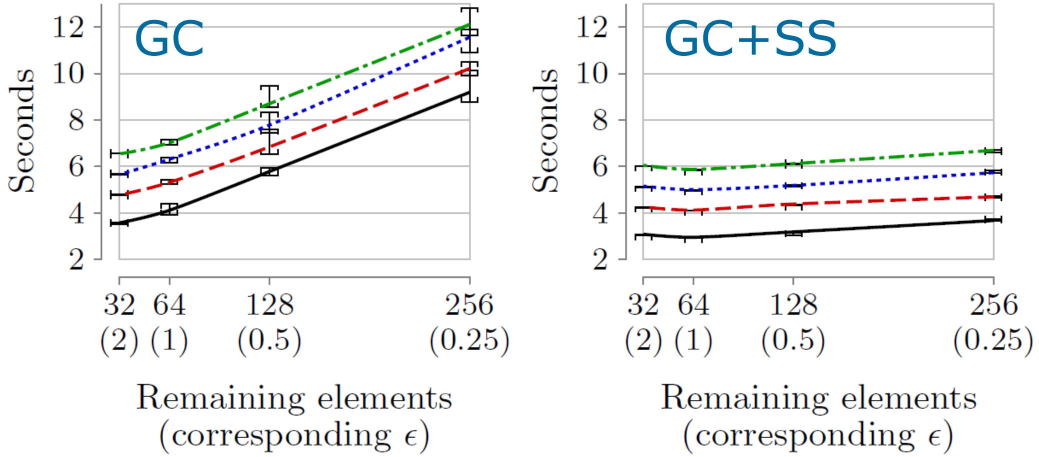




~12 ms RTT, ~430 MBits/s *(Ohio and N. Virginia)*



~100 ms RTT, ~100 MBits/s *(Ohio and Frankfurt)*



Conclusion

- *sublinear time* median selection
  – normally, exponential mechanism is linear in domain size

Secure Sublinear Time Differentially Private Median Computation

- secure computation
  – efficient implementation **without trusted third party**

- strong privacy guarantee
  – adopted by industry
- via exponential mechanism
  – **high accuracy, low ϵ**

- extensible to rank-based statistics
  – versatile & robust

Thank you.

Contact information:

jonas.boehler@sap.com

THE BEST RUN

References

[AMP10] G. Aggarwal, N. Mishra, and B. Pinkas, “Secure computation of the median (and other elements of specified ranks),” Journal of cryptology, 2010.

[DR14] C. Dwork and A. Roth, “The algorithmic foundations of differential privacy,” Foundations and Trends in Theoretical Computer Science, 2014

[DSZ15] D. Demmler, T. Schneider, and M. Zohner, “Aby-a framework for efficient mixed-protocol secure two-party computation.” NDSS, 2015.

[B18] https://www.bloomberg.com/news/articles/2018-08-30/google-and-mastercard-cut-a-secret-ad-deal-to-trackretail-sales

[BJSV15] D. Bogdanov, M. Jõemets, S. Siim, and M. Vaht, “How the estonian tax and customs board evaluated a tax fraud detection system based on secure multi-party computation,” FC 2015

[BKKRST16] D. Bogdanov, L. Kamm, B. Kubo, R. Rebane, V. Sokk, and R. Talviste, “Students and taxes: a privacy-preserving study using secure computation,” PETS 2016.

[HEK12] Y. Huang, D. Evans, and J. Katz, “Private set intersection: Are garbled circuits better than custom protocols?”, NDSS 2012.

[HMFS17] X. He, A. Machanavajjhala, C. Flynn, and D. Srivastava, “Composing differential privacy and secure computation: A case study on scaling private record linkage,” CCS, 2017.

[LVBJV16] A. Lapets, N. Volgushev, A. Bestavros, F. Jansen, & M. Varia. "Secure MPC for analytics as a web application." *IEEE Cybersecurity Development (SecDev)*. 2016.

[LLSY16] N. Li, M. Lyu, D. Su, and W. Yang. Differential privacy: From theory to practice. Synthesis Lectures on Information Security, Privacy, & Trust, 2016.

[MT07] F. McSherry and K. Talwar, “Mechanism design via differential privacy,” FOCS 2007.

[NRS07] K. Nissim, S. Raskhodnikova, and A. Smith, “Smooth sensitivity and sampling in private data analysis,” STOC, 2007.

[Shamir79] A. Shamir. How to share a secret. Communications of the ACM, 1979.

[Yao86] A. C.-C. Yao, “How to generate and exchange secrets,” FOCS, 1986.