A MASSIVELY DISTRIBUTED ALGORITHM FOR PRIVATE AVERAGING WITH MALICIOUS ADVERSARIES

Aurélien Bellet (INRIA, Magnet team)

Joint work with Pierre Dellenbach and Jan Ramon

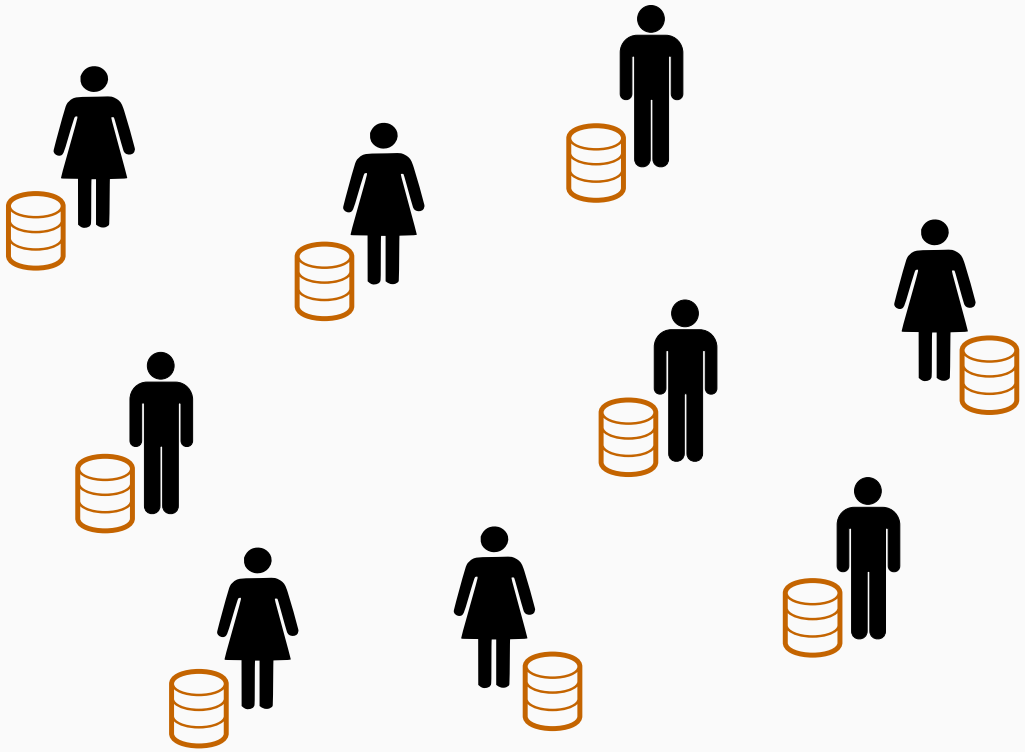
Séminaire SSA, Télécom ParisTech
October 18, 2018

OUTLINE

1. Context
2. Problem setting and related work
3. A protocol for private gossip averaging
4. A procedure to detect cheaters
5. Conclusion & Perspectives

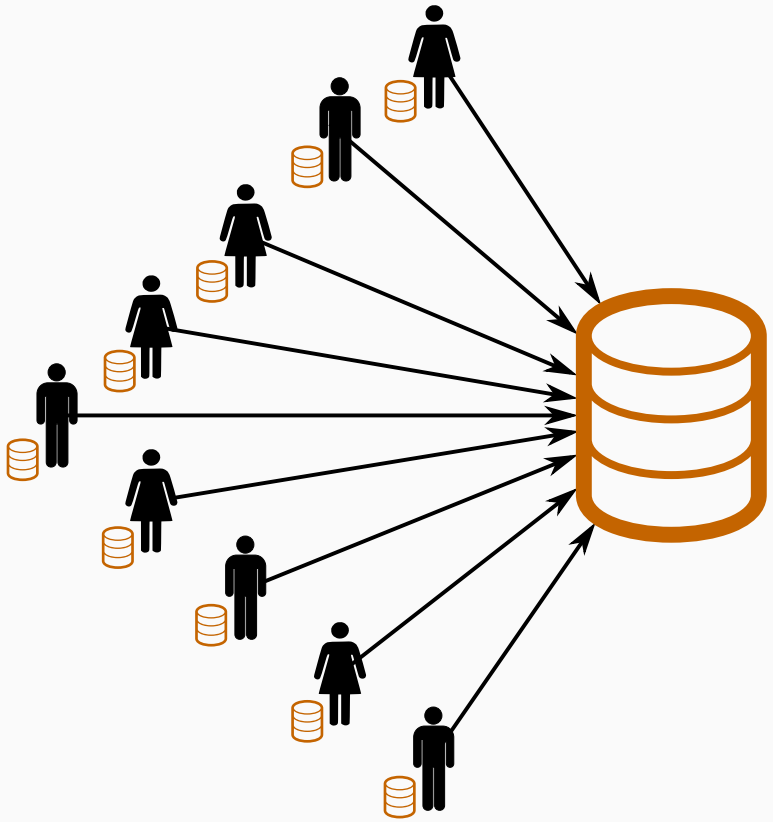
CONTEXT

MOTIVATION: ESTIMATE STATISTICS ABOUT USER BEHAVIOR

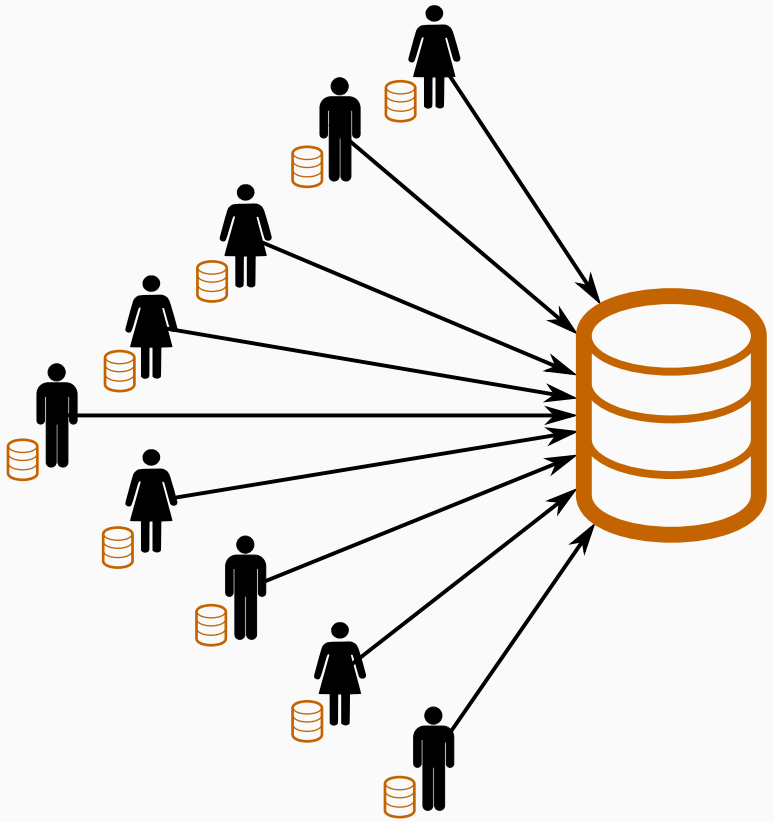


- Thousands to millions of users (devices)
- Applications: surveys, telemetry (phones, “smart” meters), and more generally any kind of data analysis and machine learning

CLASSIC APPROACH: CENTRALIZE DATA

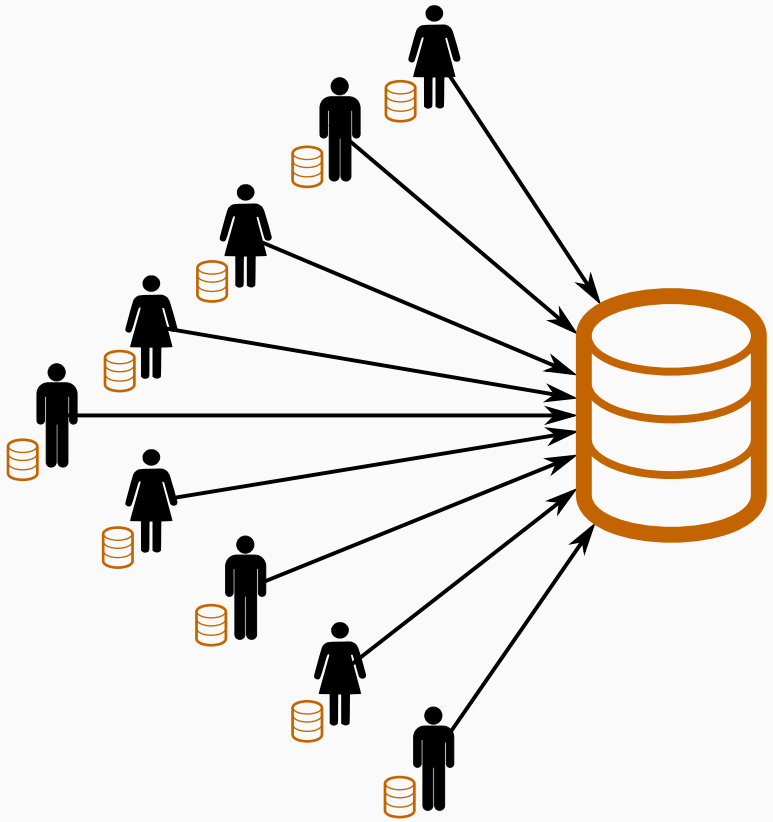


CLASSIC APPROACH: CENTRALIZE DATA



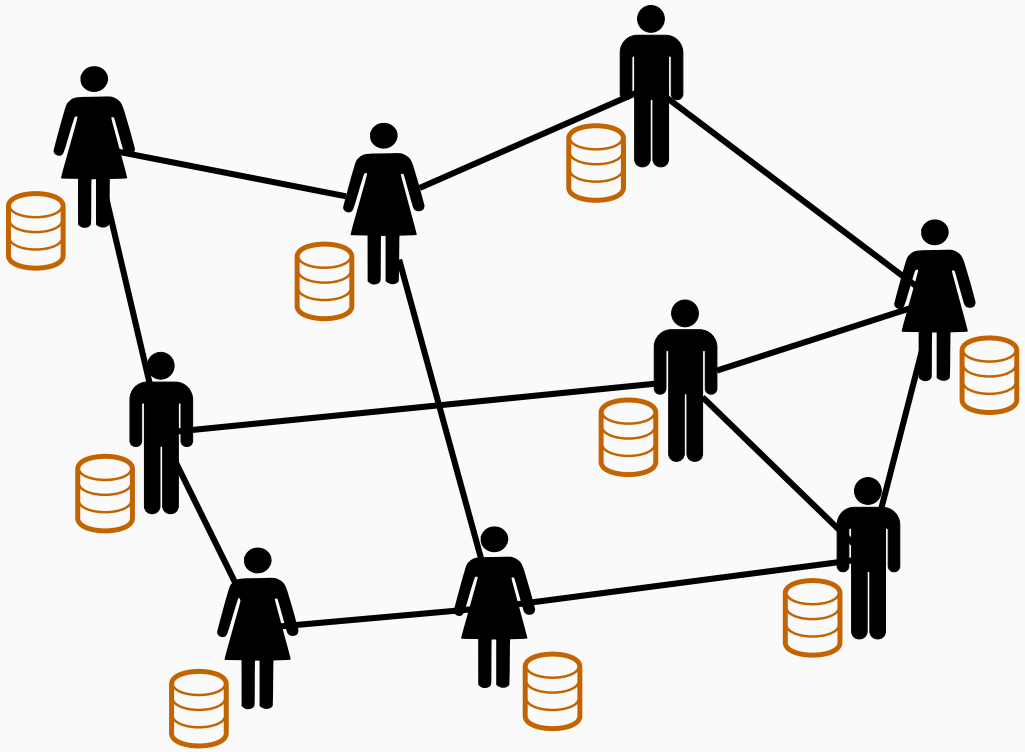
- Easy to estimate the statistics of interest

CLASSIC APPROACH: CENTRALIZE DATA



- Easy to estimate the statistics of interest
- Privacy risks
  - Central server may not be trustworthy: individual data can be misused, sold to third parties...
  - Data may leak inadvertently or due to security breach

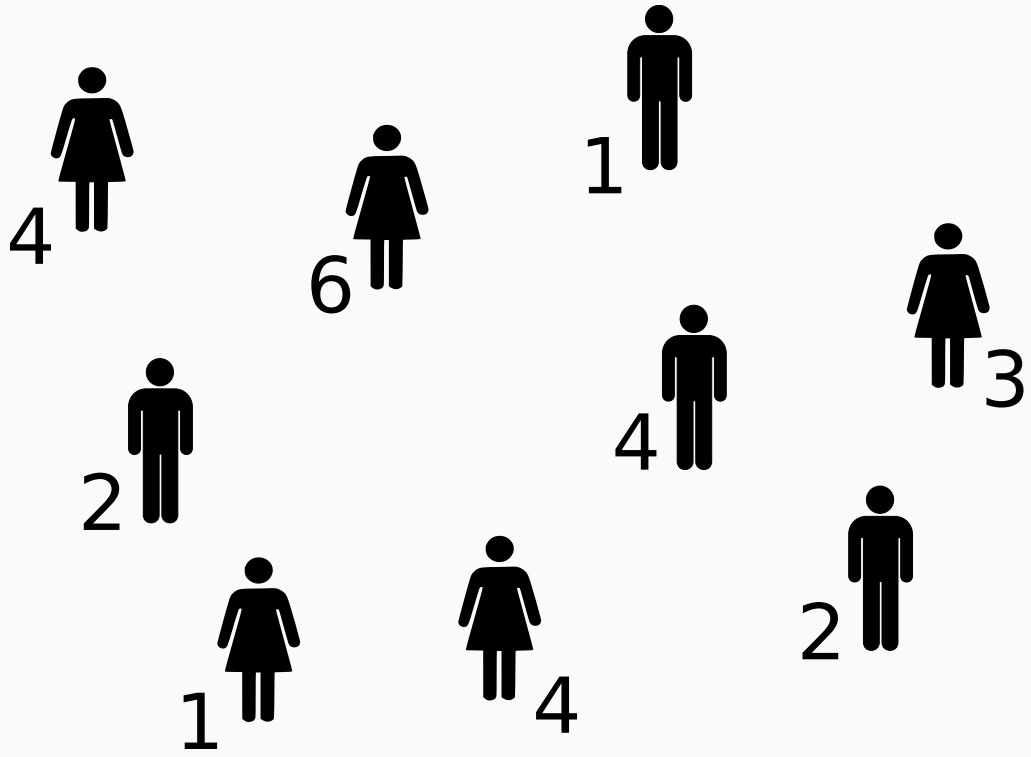
OUR APPROACH



- Formal privacy guarantees
- Scalability through P2P exchanges over random graph
- Computes exact quantity of interest
- Achieves robustness to malicious users

PROBLEM SETTING AND RELATED WORK

PRIVATE AVERAGING



- A set $U = \{1, \ldots, n\}$ of users
- Each user $u \in U$ holds a private value $X_u \in \mathbb{R}$
- **Goal of the protocol:** compute the average $X^{avg} = \frac{1}{n} \sum_{u=1}^{n} X_u$ while protecting the privacy of individual values
- Need to define formal adversary and privacy models

ADVERSARY MODEL

- We use two commonly adopted adversary models [Goldreich, 1998]
- Each user is either honest-but-curious or malicious
- Honest-but-curious (honest for short) users ($U^H \subseteq U$)
  - use all information received to infer knowledge about other users
  - follow protocol and communicate over secure channels
- Malicious users (proportion $f = 1 - |U^H|/n$)
  - send incorrect values to influence output or infer information
  - can collude with other malicious users
- Malicious users are seen as a single malicious party (adversary)

PRIVACY MODEL

- We take a Bayesian view of privacy promoted by recent work [Kasiviswanathan and Smith, 2014, Kifer and Machanavajjhala, 2014]
- $P(X_u)$: prior belief of adversary about private value X_u of $u \in U^H$
- We will consider Gaussian priors $X_u \sim \mathcal{N}(0, \sigma_X^2)$ with $\sigma_X^2 > 0$
- $\mathcal{I}$: information gathered by adversary during the protocol
- $P(X_u \mid \mathcal{I})$: posterior belief about private value X_u of $u \in U^H$

PRIVACY MODEL

- "No free lunch" theorem: cannot guarantee that nothing is learned when prior belief is bad [Kifer and Machanavajjhala, 2011]
- **Our privacy notion:** for some $\epsilon \in [0, 1)$ we want to ensure

$$\frac{var(X_u \mid \mathcal{I})}{var(X_u)} \geq 1 - \epsilon$$

- Interpretation: observing $\mathcal{I}$ did not reduce much the uncertainty of the adversary prediction

RELATED WORK: LOCAL DIFFERENTIAL PRIVACY

- A randomized function F is ϵ-locally differentially private [Duchi et al., 2012] if $\forall X, X'$ and all output O in the range of F:

$$Pr[F(X) = O] \leq e^{\epsilon} Pr[F(X') = O]$$

- Assume bounded domain: $|X_u| \leq B_X$ for all u
- Laplace mechanism: each user u independently shares $F(X_u) = X_u + \eta$, with $\eta \sim Lap(0, 2B_X/\epsilon)$
  - $F(X_u)$ is ϵ-LDP
  - $\widehat{X^{avg}} = \frac{1}{n} \sum_{u=1}^{n} F(X_u)$ is an unbiased estimate of X^{avg}
  - $\widehat{X^{avg}}$ has variance $8B_X^2/n\epsilon^2$ (**this is optimal for LDP**)
- **Error can be large + lack of robustness to malicious users**

RELATED WORK: CRYPTO

- **Privacy notion:** parties learn nothing more than the output X^{avg}
- Assumes computationally bounded adversary
- Additively homomorphic encryption (e.g., [Paillier, 1999])
  - Public key to encrypt, private key to decrypt
  - Product of two cypher texts gives encryption of the sum:

$$\varepsilon(m_1 + m_2) = \varepsilon(m_1) \cdot \varepsilon(m_2)$$

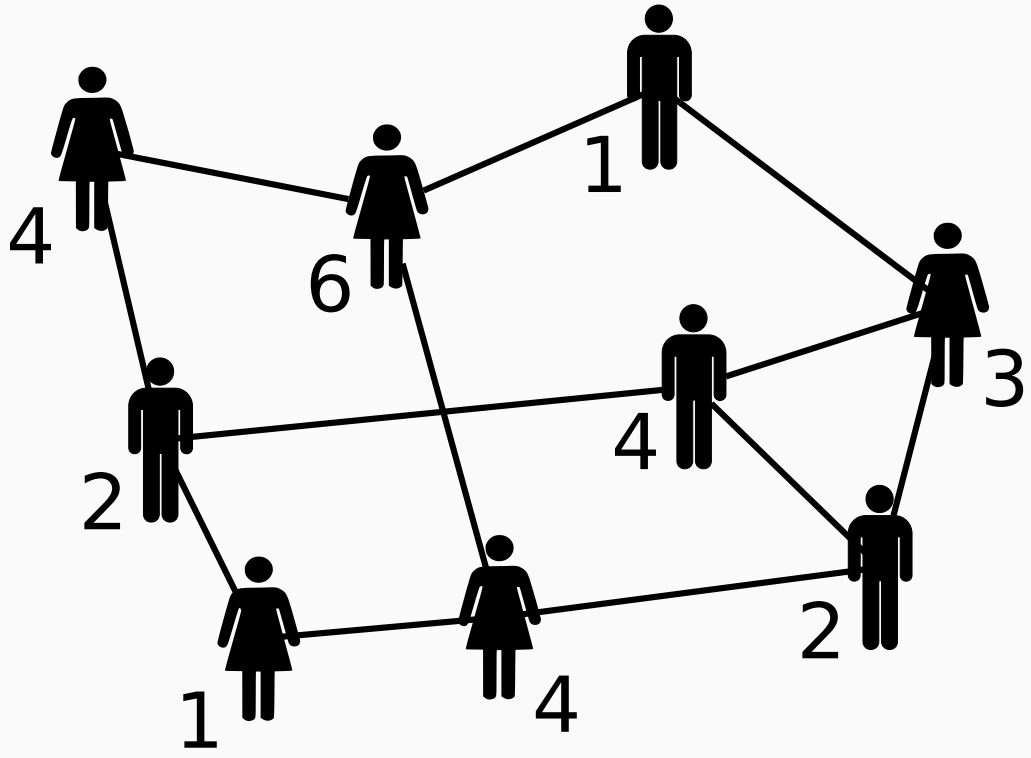
  - **Need a trusted third party** to keep the private key
- Secure Multi-Party Computation (see [Lindell and Pinkas, 2009])
  - "Simulates" a trusted third party
  - Some robustness to malicious users
  - **Typically intractable for large number of parties**

A PROTOCOL FOR PRIVATE GOSSIP AVERAGING

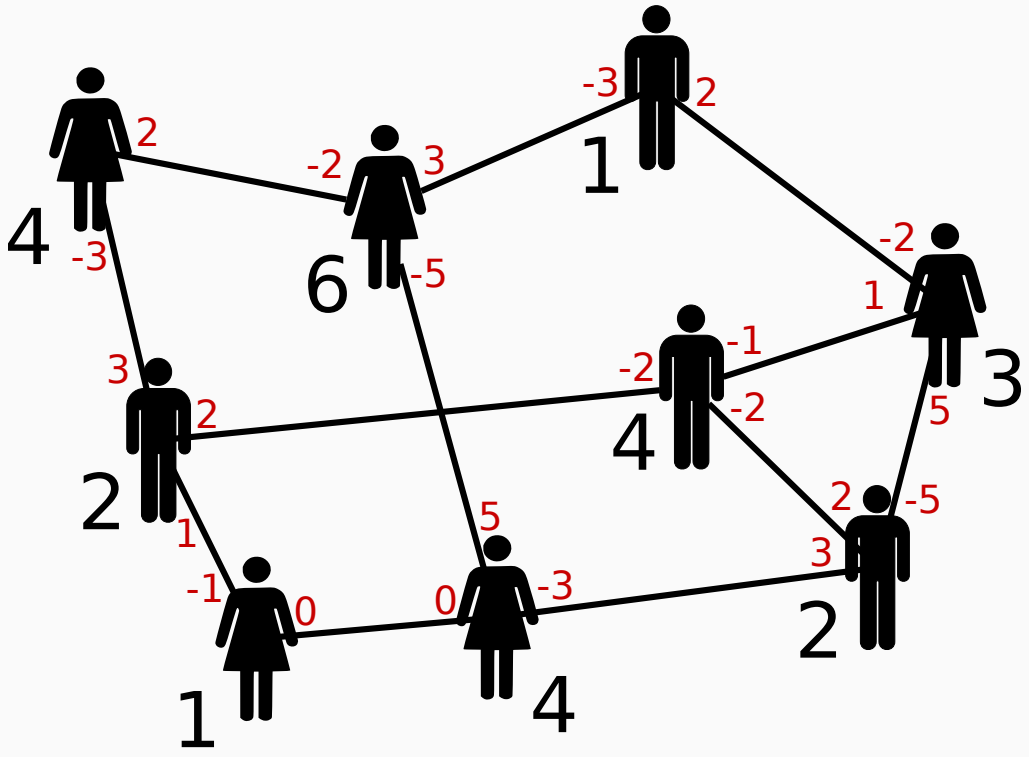
NETWORK GRAPH CONSTRUCTION

- We rely on communication over a sparse random network graph
- This is key to ensure privacy, scalability and robustness
- Random k-out graph $G = (U, E)$ [Bollobás, 2001]:
  - Each (honest) user selects k other users uniformly at random
  - Edge $(u, v) \in E$ created if u selected v or v selected u
- Many good properties:
  - Degrees roughly equal to k for n large enough
  - Tight high probability bounds for connectivity of subgraph $G^H = (U^H, E^H)$ of honest users [Yağan and Makowski, 2013]
  - Number of honest neighbors of a honest node tightly concentrated around $k(1 - f)$

RANDOMIZATION PROTOCOL: OVERVIEW

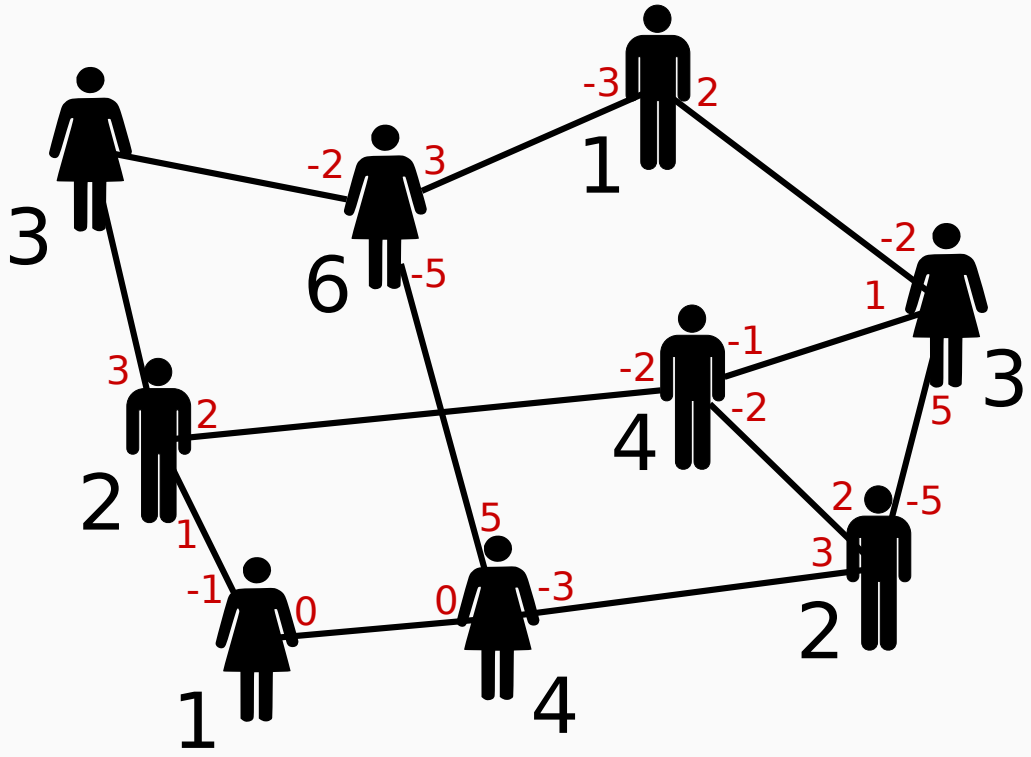


RANDOMIZATION PROTOCOL: OVERVIEW



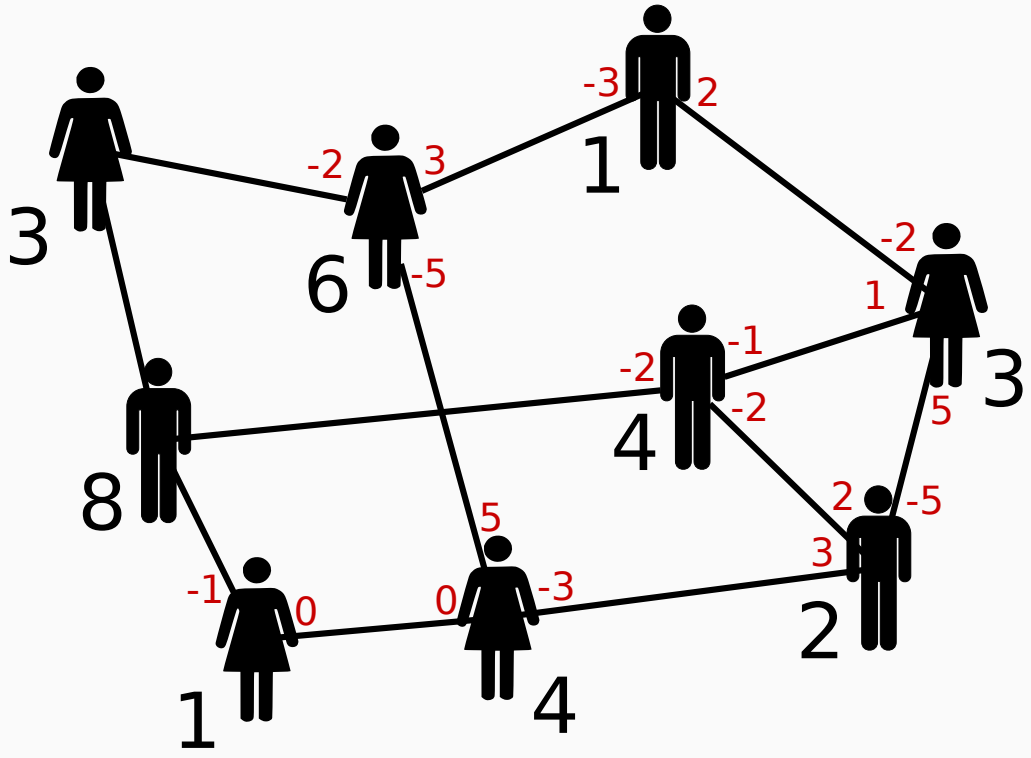
- For each edge $(u, v) \in E$, u and v add/subtract a random value

RANDOMIZATION PROTOCOL: OVERVIEW



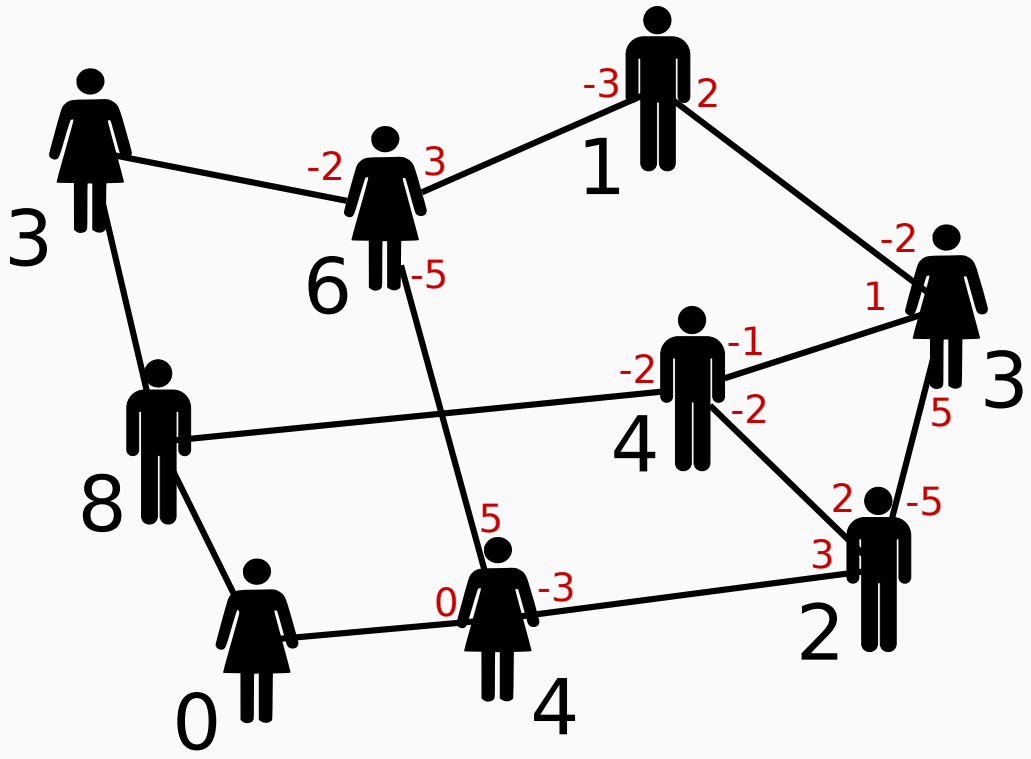
- For each edge $(u, v) \in E$, u and v add/subtract a random value

RANDOMIZATION PROTOCOL: OVERVIEW



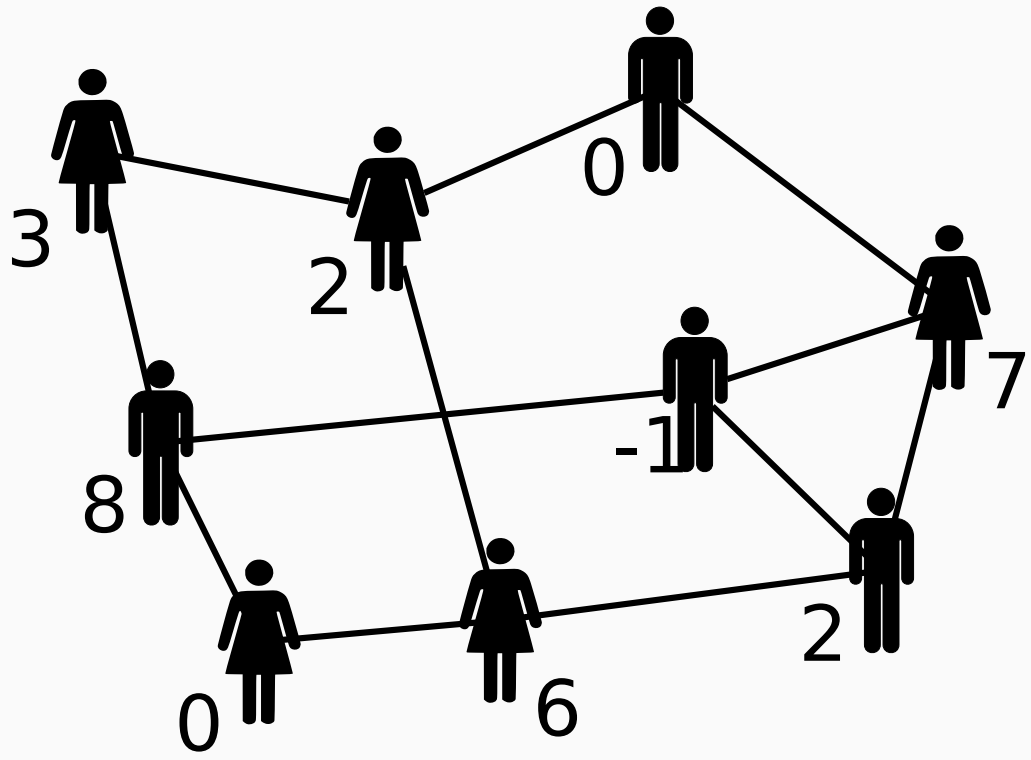
- For each edge $(u, v) \in E$, u and v add/subtract a random value

RANDOMIZATION PROTOCOL: OVERVIEW



- For each edge $(u, v) \in E$, u and v add/subtract a random value

RANDOMIZATION PROTOCOL: OVERVIEW



- For each edge $(u, v) \in E$, u and v add/subtract a random value
- Average is unchanged but private values are obfuscated

GOPA RANDOMIZATION PROTOCOL: ALGORITHM

Algorithm 1: GOPA Randomization protocol

Input: graph $G = (U, E)$, private values $X = (X_u)_{u \in U}$, distribution $\mu : \mathbb{R} \to [0, 1]$
for all neighbor pairs $(u, v) \in E$ s.t. $u < v$ **do**
    u and v jointly draw a random number $\delta \in \mathbb{R}$ from μ
    $\delta_{u,v} \leftarrow \delta$, $\delta_{v,u} \leftarrow -\delta$
end for
for all users $u \in U$ **do**
    $\Delta_u \leftarrow \sum_{v \in N(u)} \delta_{u,v}$
    $\widetilde{X}_u \leftarrow X_u + \Delta_u$
end for
Output: noisy values $\widetilde{X} = (\widetilde{X}_u)_{u \in U}$

PRIVACY ANALYSIS: INFORMATION OBSERVED

- The information $\mathcal{I}$ gathered by the adversary contains:
  (i) Noisy values $\widetilde{X}$ of all users
  (ii) The network graph
  (iii) Noise values for exchanges involving a malicious user
- Note: this implies that the adversary learns the average over honest users
- The only unknowns are the private values of honest users, and noise values exchanged between them

Theorem ([Dellenbach et al., 2018])

Assume the following:

- *Prior belief* $X_u \sim \mathcal{N}(0, \sigma_X^2)$ *for all honest users* $u \in U^H$,
- *Noise distribution* $\mu = \mathcal{N}(0, \sigma_\delta^2)$.

Denote by $e_u \in \mathbb{R}^n$ *the indicator vector of the* u*-th coordinate and let* $M = (I + \frac{\sigma_\delta^2}{\sigma_X^2} L^H)^{-1} \in \mathbb{R}^{n \times n}$*, where* L^H *is the Laplacian matrix of the graph* G^H*. Then we have for all honest user* $u \in U^H$*:*

$$\frac{var(X_u \mid \mathcal{I})}{var(X_u)} = 1 - e_u^\top M e_u.$$

Proof technique: Formalize $\mathcal{I}$ by a linear system, and work out the variance expression for X_u conditionally upon this linear system.

PRIVACY ANALYSIS: INTERPRETATION

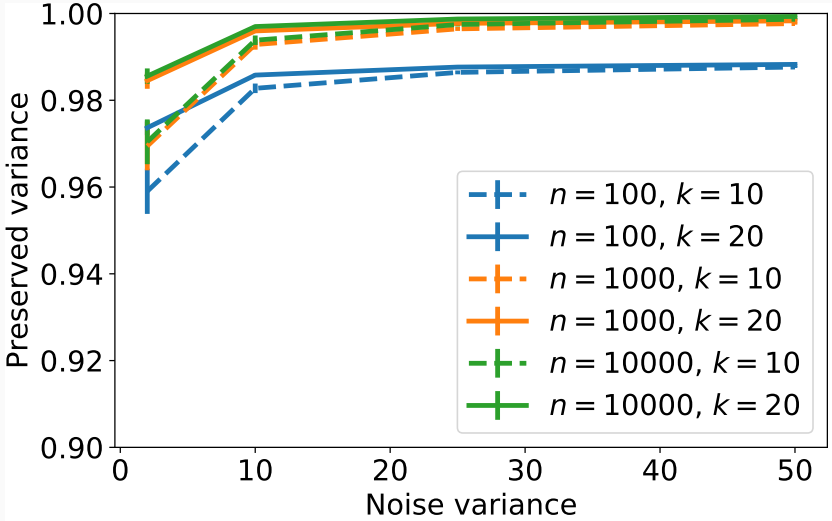
- The privacy guarantee only depends on the interactions between honest users $\rightarrow$ **robust to arbitrary behavior of malicious users**
- Given any $y \in \mathbb{R}^n$, $My = (I + \frac{\sigma_\delta^2}{\sigma_X^2} L^H)^{-1} y$ is the solution to

$$\min_{s \in \mathbb{R}^n} \|s - y\|_2^2 + \alpha s^\top L^H s, \quad \text{with } \alpha = \sigma_\delta^2 / \sigma_X^2 \geq 0$$

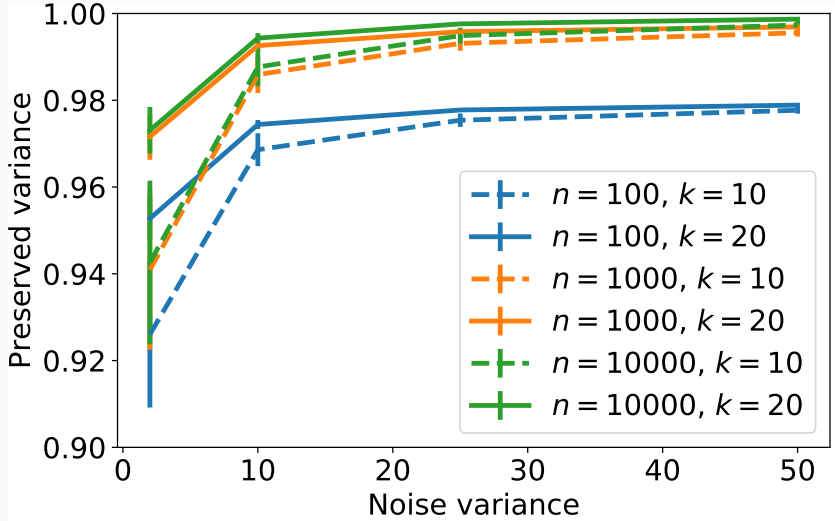
- This problem is known as graph smoothing
  - Popular in SSL [Zhou et al., 2003], MTL [Evgeniou and Pontil, 2004], signal processing on graphs [Shuman et al., 2013]...
- M can be shown to be doubly stochastic, and we smooth $y = e_u$
- Hence the larger σ_δ^2 and the denser G^H, the closer $e_u^\top M e_u$ is to 0
  - When $\sigma_\delta^2 = 0$, we have $\epsilon = 1$ (no privacy)
  - As $\sigma_\delta^2 \to \infty$, we have $\epsilon \to 1/|U_H|$ (if G^H is connected)

PRIVACY ANALYSIS: INTERPRETATION

- When G^H is connected, all honest users contribute to keeping each private value safe (not only direct neighbors)
- When G^H is not connected, smoothing occurs independently in each connected component
- Using a denser graph can compensate for smaller noise variance $\rightarrow$ floating point representation remains compact and we avoid catastrophic failures in case of user drop-out



(a) 10% of malicious nodes



(b) 50% of malicious nodes

COMPUTING THE AVERAGE

- Our privacy guarantees hold under the knowledge of the noisy values $\widetilde{X} = (\widetilde{X}_u)_{u \in U}$
- Each user u may thus safely share $\widetilde{X}_u$ so that one can compute $X^{avg} = \frac{1}{n} \sum_{u=1}^{N} \widetilde{X}_u$
- For instance, users can naturally rely on gossip averaging [Boyd et al., 2006] over the network graph
- **But how to ensure that malicious users do not manipulate the outcome?**

A PROCEDURE TO DETECT CHEATERS

REQUIREMENTS FOR CHEATER DETECTION

- It is impossible to force a user to give the “correct” initial input
- But malicious users may cheat during the randomization protocol by adding/subtracting wrong noise values
- **Goal:** ensure that given input X, the protocol will either output X^{avg} or detect cheaters (with high probability)
- We want to avoid relying on a trusted verification entity

INGREDIENTS OF THE PROCEDURE

- We ask users to publish some (potentially encrypted) values during execution
- Published values are publicly available and cannot be edited
  - Can be achieved without central authority via decentralized commitment schemes like blockchain
- For encryption we use the Paillier scheme [Paillier, 1999]
  - Public (encryption) key K^{pub}, secret (decryption) key K^{priv}
  - A message $m \in \mathbb{Z}_N^*$ is encrypted into a cypher text $\varepsilon(m)$ using K^{pub}
  - A cypher text $\varepsilon(m)$ is decrypted into plain text m using K^{priv}
  - Additive homomorphic property: $\varepsilon(m_1 + m_2) = \varepsilon(m_1) \cdot \varepsilon(m_2)$

CHEATER DETECTION PROCEDURE

Algorithm 2: Verification procedure for the GOPA randomization protocol

1: **Input:** each user u has generated its own Paillier encryption scheme and has published
- Public key K_u^{pub}, $\varepsilon_u^P(X_u)$ (before the execution of Algorithm 1)
- Noise values $\varepsilon_u^P(\delta_{u,v})$ (when exchanging with $v \in N(u)$ during Algorithm 1)
- $\varepsilon_u^P(\widetilde{X}_u)$, $\varepsilon_u^P(\Delta_u)$ (at the end of Algorithm 1)

CHEATER DETECTION PROCEDURE

Algorithm 2: Verification procedure for the GOPA randomization protocol

1: **Input:** each user u has generated its own Paillier encryption scheme and has published
   - Public key K_u^{pub}, $\varepsilon_u^P(X_u)$ (before the execution of Algorithm 1)
   - Noise values $\varepsilon_u^P(\delta_{u,v})$ (when exchanging with $v \in N(u)$ during Algorithm 1)
   - $\varepsilon_u^P(\widetilde{X}_u)$, $\varepsilon_u^P(\Delta_u)$ (at the end of Algorithm 1)

2: **for all** user $u \in U$ **do**
3: $\quad \varepsilon^{\Delta} \leftarrow \prod_{v \in N(u)} \varepsilon_u^P(\delta_{u,v})$, $\varepsilon^{\widetilde{X}} \leftarrow \varepsilon_u^P(X_u) \cdot \varepsilon_u^P(\Delta_u)$
4: $\quad$ Verify that $\varepsilon_u^P(\Delta_u) = \varepsilon^{\Delta}$ and $\varepsilon_u^P(\widetilde{X}_u) = \varepsilon^{\widetilde{X}}$ (if not, add u to cheater list)
5: **end for**

CHEATER DETECTION PROCEDURE

Algorithm 2: Verification procedure for the GOPA randomization protocol

1: **Input:** each user u has generated its own Paillier encryption scheme and has published
   - Public key K_u^{pub}, $\varepsilon_u^P(X_u)$ (before the execution of Algorithm 1)
   - Noise values $\varepsilon_u^P(\delta_{u,v})$ (when exchanging with $v \in N(u)$ during Algorithm 1)
   - $\varepsilon_u^P(\widetilde{X}_u)$, $\varepsilon_u^P(\Delta_u)$ (at the end of Algorithm 1)

2: **for all** user $u \in U$ **do**
3: $\quad \varepsilon^{\Delta} \leftarrow \prod_{v \in N(u)} \varepsilon_u^P(\delta_{u,v})$, $\varepsilon^{\widetilde{X}} \leftarrow \varepsilon_u^P(X_u) \cdot \varepsilon_u^P(\Delta_u)$
4: $\quad$ Verify that $\varepsilon_u^P(\Delta_u) = \varepsilon^{\Delta}$ and $\varepsilon_u^P(\widetilde{X}_u) = \varepsilon^{\widetilde{X}}$ (if not, add u to cheater list)
5: **end for**

6: $\quad$ **for all** user $u \in U$ **do**
7: $\quad\quad$ Draw random subset V_u of $N(u)$ with $|V_u| = \lceil (1-\beta) d_u \rceil$
8: $\quad\quad$ Ask u to publish $\delta_{u,v}$ for $v \in V_u$
9: $\quad\quad$ **for all** $v \in V_u$ **do**
10: $\quad\quad\quad$ Verify that $\varepsilon_v^P(\delta_{v,u}) = \varepsilon_v(-\delta_{u,v})$, $\varepsilon_u^P(\delta_{u,v}) = \varepsilon_u(\delta_{u,v})$ (if not, add u, v to cheater list)
11: $\quad\quad$ **end for**
12: $\quad$ **end for**

DETECTION GUARANTEES

Proposition ([Dellenbach et al., 2018])

Let C be the number of times a user cheated during the randomization protocol. If we apply the verification procedure (Algorithm 2), then the probability that a cheater is detected is at least $1 - \beta^{2C}$.

- Guards against large-scale cheating: the more cheating, the more likely one gets caught
- Publishing a fraction of noise values in plain text decreases the privacy guarantees (i.e., ignored edges in G^H)
- If β is agreed in advance, this can be compensated by constructing a network graph of larger degree

CONCLUSION & PERSPECTIVES

Wrapping up

- A scalable and robust protocol for private averaging
- A verification procedure to prevent cheating

Ongoing work

- Generalization of privacy analysis to non-Gaussian priors
- Combination with differential privacy
- Implementation, integration within more complex algorithms
- Extension to higher-order U-statistics

Privacy-Preserving Machine Learning workshop at NIPS 2018

`https://ppml-workshop.github.io/ppml/`

Thank you for your attention!

Questions?

REFERENCES I

[Bollobás, 2001] Bollobás, B. (2001).
Random Graphs.
Cambridge University Press.

[Boyd et al., 2006] Boyd, S., Ghosh, A., Prabhakar, B., and Shah, D. (2006).
Randomized Gossip Algorithms.
IEEE/ACM Transactions on Networking, 14(SI):2508–2530.

[Dellenbach et al., 2018] Dellenbach, P., Bellet, A., and Ramon, J. (2018).
Hiding in the Crowd: A Massively Distributed Algorithm for Private Averaging with Malicious Adversaries.
Technical report, arXiv:1803.09984.

[Duchi et al., 2012] Duchi, J. C., Jordan, M. I., and Wainwright, M. J. (2012).
Privacy Aware Learning.
In *NIPS*.

[Evgeniou and Pontil, 2004] Evgeniou, T. and Pontil, M. (2004).
Regularized multi-task learning.
In *SIGKDD*.

[Goldreich, 1998] Goldreich, O. (1998).
Secure multi-party computation.
Manuscript. Preliminary version.

References II

[Kasiviswanathan and Smith, 2014] Kasiviswanathan, S. P. and Smith, A. (2014).
On the 'Semantics' of Differential Privacy: A Bayesian Formulation.
Journal of Privacy and Confidentiality, 6(1):1–16.

[Kifer and Machanavajjhala, 2011] Kifer, D. and Machanavajjhala, A. (2011).
No free lunch in data privacy.
In *SIGKDD*.

[Kifer and Machanavajjhala, 2014] Kifer, D. and Machanavajjhala, A. (2014).
Pufferfish: A framework for mathematical privacy definitions.
ACM Transactions on Database Systems, 39(1):3:1–3:36.

[Lindell and Pinkas, 2009] Lindell, Y. and Pinkas, B. (2009).
Secure Multiparty Computation for Privacy-Preserving Data Mining.
Journal of Privacy and Confidentiality, 1(1):59–98.

[Paillier, 1999] Paillier, P. (1999).
Public-key cryptosystems based on composite degree residuosity classes.
In *EUROCRYPT*.

[Shuman et al., 2013] Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2013).
The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains.
IEEE Signal Processing Magazine, 30(3):83–98.

References III

[Yağan and Makowski, 2013] Yağan, O. and Makowski, A. M. (2013).
On the Connectivity of Sensor Networks Under Random Pairwise Key Predistribution.
IEEE Transactions on Information Theory, 59(9):5754–5762.

[Zhou et al., 2003] Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2003).
Learning with Local and Global Consistency.
In *NIPS*.