Near-Universally-Optimal Differentially Private Minimum Spanning Trees

Richard Hladík[1,2], Jakub Tětek[2]

[1] ETH Zürich
[2] INSAIT, Sofia University "St. Kliment Ohridski"

FORC 2025

Near-Universally-Optimal Differentially Private Minimum Spanning Trees

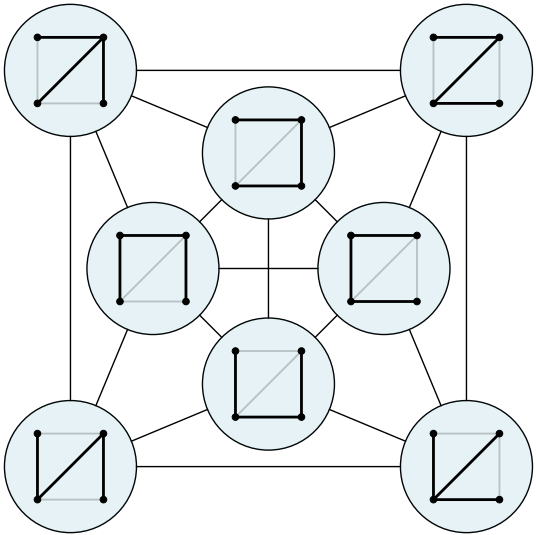
Richard Hladík[1,2], Jakub Tětek[2]

[1] ETH Zürich
[2] INSAIT, Sofia University "St. Kliment Ohridski"

FORC 2025

Intro

This talk: differential privacy bounds via combinatorial properties of spanning trees



Setup [Sea16]

- public knowledge: city transport map

Setup [Sea16]

- public knowledge: city transport map
- private dataset: utilization statistics of each line

Setup [Sea16]

- public knowledge: city transport map
- private dataset: utilization statistics of each line
- **goal:** privately release the (approximate) MST of G

Setup [Sea16]

- public knowledge: city transport map
- private dataset: utilization statistics of each line
- **goal:** privately release the (approximate) MST of G
  - not just its weight (that's easier), release the spanning tree itself

Setup [Sea16]

- public knowledge: city transport map
- private dataset: utilization statistics of each line
- **goal:** privately release the (approximate) MST of G
  - not just its weight (that's easier), release the spanning tree itself
  - can't just release a MST (not private), instead release a random spanning tree of G that's "close to minimal" in expectation

Setup (more formally)

- **Input:**
  - public unweighted connected graph G
  - private real edge weights $\mathbf{w}$

Setup (more formally)

- **Input:**
  - public unweighted connected graph G
  - private real edge weights $\mathbf{w}$
- **Goal:** mechanism A that returns a probability distribution over all spanning trees of G such that:

Setup (more formally)

- **Input:**
  - public unweighted connected graph G
  - private real edge weights $\mathbf{w}$
- **Goal:** mechanism A that returns a probability distribution over all spanning trees of G such that:
  - ε-**differential privacy**: *robustness to small change in weights*

$$\forall \underbrace{\mathbf{w} \sim \mathbf{w}'}_{\|\mathbf{w}-\mathbf{w}'\|_1 \leq 1} \ \forall T \in \underbrace{\mathcal{T}(G)}_{\text{spanning trees of } G} \ P[A(G,\mathbf{w}) = T] \leq e^{\varepsilon} P[A(G,\mathbf{w}') = T]$$

Setup (more formally)

- **Input:**
  - public unweighted connected graph G
  - private real edge weights $\mathbf{w}$
- **Goal:** mechanism A that returns a probability distribution over all spanning trees of G such that:
  - ε-**differential privacy**: *robustness to small change in weights*

$$\forall \underbrace{\mathbf{w} \sim \mathbf{w}'}_{\|\mathbf{w}-\mathbf{w}'\|_1 \le 1} \ \forall T \in \underbrace{\mathcal{T}(G)}_{\text{spanning trees of } G} \ P[A(G,\mathbf{w}) = T] \le e^{\varepsilon} P[A(G,\mathbf{w}') = T]$$

  - **approximate MST**: *the expected weight is small*

$$\text{additive error} = \mathbb{E}_{T \sim A(G,\mathbf{w})}[\mathbf{w}(T)] - \mathbf{w}(\underbrace{T^*}_{\text{MST of } G}) \quad \text{is small}$$

Previous work & our contribution

Previous work & our contribution

reference	$\mathbb{E}$[error]	notes
[Sea16]	$\forall G \quad \mathcal{O}(n \log n/\varepsilon)$	add Lap$(1/\varepsilon)$ to every edge

Previous work & our contribution

reference	$\mathbb{E}$[error]	notes
[Sea16]	$\forall G \quad \mathcal{O}(n \log n/\varepsilon)$	add Lap$(1/\varepsilon)$ to every edge
[Sea16]	$\exists G \quad \Omega(n/\varepsilon)$	worst-case construction

Previous work & our contribution

reference	$\mathbb{E}$[error]	notes
[Sea16]	$\forall G \quad \mathcal{O}(n \log n/\varepsilon)$	add Lap$(1/\varepsilon)$ to every edge
[Sea16]	$\exists G \quad \Omega(n/\varepsilon)$	worst-case construction
Our result 1	$\forall G \quad \mathcal{O}(D_G \log n/\varepsilon)$	$D_G \in [1, n)$; tighter analysis of [Sea16]

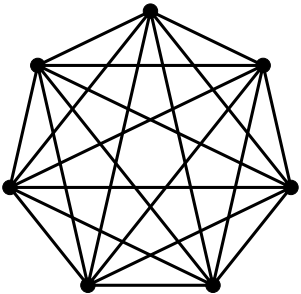
Previous work & our contribution

reference	$\mathbb{E}$[error]	notes
[Sea16]	$\forall G \quad \mathcal{O}(n \log n/\varepsilon)$	add Lap$(1/\varepsilon)$ to every edge
[Sea16]	$\exists G \quad \Omega(n/\varepsilon)$	worst-case construction
Our result 1	$\forall G \quad \mathcal{O}(D_G \log n/\varepsilon)$	$D_G \in [1, n)$; tighter analysis of [Sea16]
Our result 2	$\forall G \quad \Omega(D_G/\varepsilon)$	graph-specific lower bound

Previous work & our contribution

reference	$\mathbb{E}$[error]	notes
[Sea16]	$\forall G \quad \mathcal{O}(n \log n/\varepsilon)$	add Lap$(1/\varepsilon)$ to every edge
[Sea16]	$\exists G \quad \Omega(n/\varepsilon)$	worst-case construction
Our result 1	$\forall G \quad \mathcal{O}(D_G \log n/\varepsilon)$	$D_G \in [1, n)$; tighter analysis of [Sea16]
Our result 2	$\forall G \quad \Omega(D_G/\varepsilon)$	graph-specific lower bound
Our result 3	$\exists G \quad \Omega(n \log n/\varepsilon)$	special case; implies [Sea16] is worst-case tight

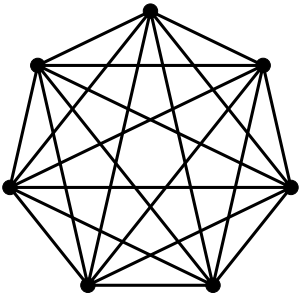
Why care about D_G?



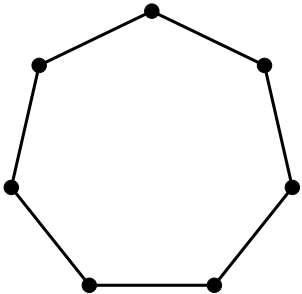
$D_G = n - 1$

$\Theta(n \log n/\varepsilon)$ expected error

Why care about D_G?



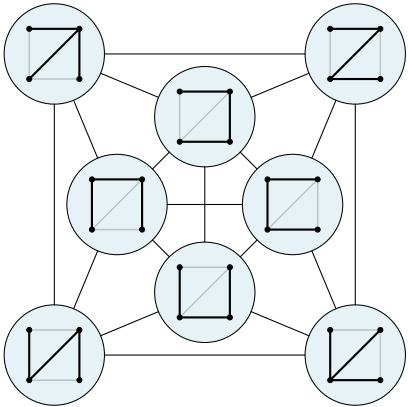
$D_G = n - 1$
$\Theta(n \log n/\varepsilon)$ expected error



$D_G = 1$
$\Theta(\log n/\varepsilon)$ expected error

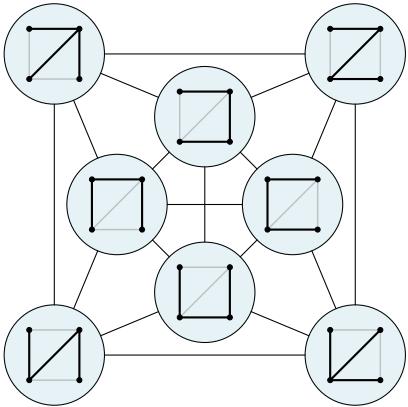
Defining D_G

- $\mathcal{T}$ = space of spanning trees of G



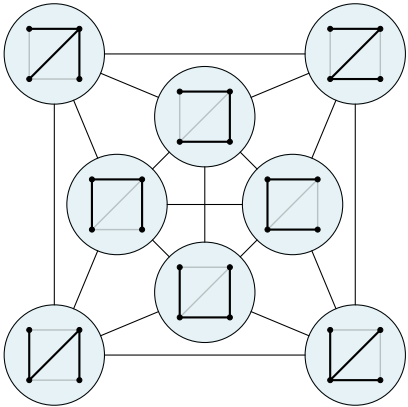
Defining D_G

- $\mathcal{T}$ = space of spanning trees of G
- $d_{\mathcal{T}}(T_1, T_2) \coloneqq |T_1 \setminus T_2| = |T_2 \setminus T_1| = \frac{1}{2}|T_1 \Delta T_2|$
  - = number of edge swaps to go from T_1 to T_2



Defining D_G

- $\mathcal{T}$ = space of spanning trees of G
- $d_{\mathcal{T}}(T_1, T_2) \coloneqq |T_1 \setminus T_2| = |T_2 \setminus T_1| = \frac{1}{2}|T_1 \Delta T_2|$
  - = number of edge swaps to go from T_1 to T_2
- $D_G \coloneqq \mathrm{diam}(\mathcal{T}) = \max_{T_1, T_2} d_{\mathcal{T}}(T_1, T_2)$



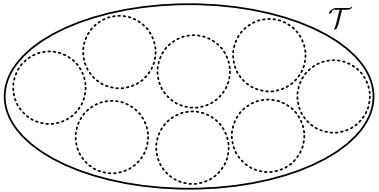
Lower bound

Claim: On any G, $\Omega(D_G/\varepsilon)$ expected error is needed

Lower bound

Claim: On any G, $\Omega(D_G/\varepsilon)$ expected error is needed
High-level proof summary:

- use well-known packing lower bounds [Vad17]
- lower bound reduces to fitting many balls into the space of spanning trees



- do that via error-correcting codes

Lower bound 1/3: reduction to the private spanning tree problem

- **Problem:** privately release a given spanning tree

Lower bound 1/3: reduction to the private spanning tree problem

- **Problem:** privately release a given spanning tree
- **Input:** public graph G, private $T_x \in \mathcal{T}$

Lower bound 1/3: reduction to the private spanning tree problem

- **Problem:** privately release a given spanning tree
- **Input:** public graph G, private $T_x \in \mathcal{T}$
- **Output:** random $T \in \mathcal{T}$ such that:

Lower bound 1/3: reduction to the private spanning tree problem

- **Problem:** privately release a given spanning tree
- **Input:** public graph G, private $T_x \in \mathcal{T}$
- **Output:** random $T \in \mathcal{T}$ such that:
  - closeness to the original tree: $d_{\mathcal{T}}(T, T_x)$ is small in expectation

Lower bound 1/3: reduction to the private spanning tree problem

- **Problem:** privately release a given spanning tree
- **Input:** public graph G, private $T_x \in \mathcal{T}$
- **Output:** random $T \in \mathcal{T}$ such that:
  - closeness to the original tree: $d_{\mathcal{T}}(T, T_x)$ is small in expectation
  - ε-DP: $\forall \underbrace{T_x \sim T_y}_{d(T_x, T_y)=1}, T \in \mathcal{T} \quad \Pr[A(G, T_x) = T] \leq e^{\varepsilon} \Pr[A(G, T_y) = T]$

Lower bound 1/3: reduction to the private spanning tree problem

- **Problem:** privately release a given spanning tree
- **Input:** public graph G, private $T_x \in \mathcal{T}$
- **Output:** random $T \in \mathcal{T}$ such that:
  - closeness to the original tree: $d_{\mathcal{T}}(T, T_x)$ is small in expectation
  - ε-DP: $\forall \underbrace{T_x \sim T_y}_{d(T_x, T_y)=1}, T \in \mathcal{T} \quad \Pr[A(G, T_x) = T] \leq e^{\varepsilon} \Pr[A(G, T_y) = T]$

Observation: This problem is reducible to private MST: (up to the error scaling by a $\frac{1}{2}$ factor)

- given $T_x \in \mathcal{T}$, define $\mathbf{w}_x(e) = 0$ if $e \in T_x$, $\frac{1}{2}$ otherwise.
- return the private MST of $(G, \mathbf{w}_x)$

Lower bound 1/3: reduction to the private spanning tree problem

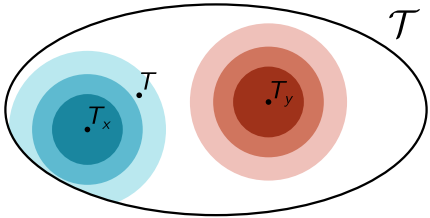
- **Problem:** privately release a given spanning tree
- **Input:** public graph G, private $T_x \in \mathcal{T}$
- **Output:** random $T \in \mathcal{T}$ such that:
  - closeness to the original tree: $d_{\mathcal{T}}(T, T_x)$ is small in expectation
  - ε-DP: $\forall \underbrace{T_x \sim T_y}_{d(T_x, T_y)=1}, T \in \mathcal{T} \quad \Pr[A(G, T_x) = T] \leq e^{\varepsilon} \Pr[A(G, T_y) = T]$

Observation: This problem is reducible to private MST: (up to the error scaling by a $\frac{1}{2}$ factor)

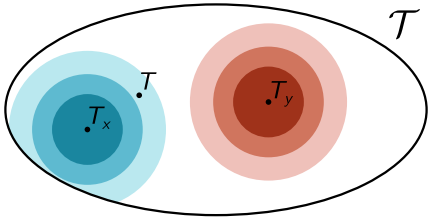
- given $T_x \in \mathcal{T}$, define $\mathbf{w}_x(e) = 0$ if $e \in T_x$, $\frac{1}{2}$ otherwise.
- return the private MST of $(G, \mathbf{w}_x)$

All we need is a lower bound for the private spanning tree problem.

Lower bound 2/3: reduction to packing (intuition)

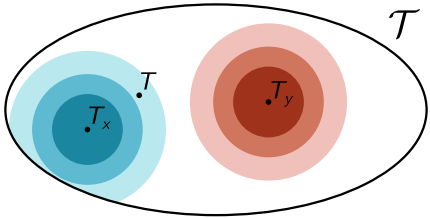


Lower bound 2/3: reduction to packing (intuition)



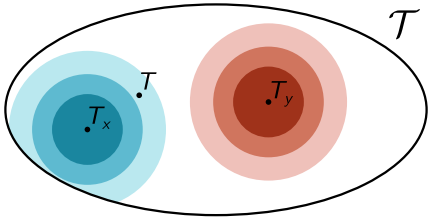
- take $T_x, T_y \in \mathcal{T}$ far apart

Lower bound 2/3: reduction to packing (intuition)



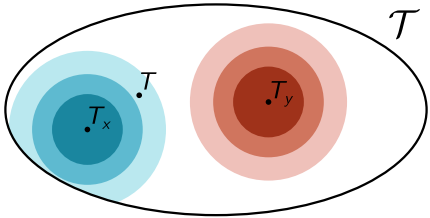
- take $T_x, T_y \in \mathcal{T}$ far apart
- for any T with $d(T_x, T)$ small: $d(T_y, T)$ is large
- intuitively, we want $\Pr[A(G, T_x) = T]$ large and $\Pr[A(G, T_y) = T]$ small

Lower bound 2/3: reduction to packing (intuition)



- take $T_x, T_y \in \mathcal{T}$ far apart
- for any T with $d(T_x, T)$ small: $d(T_y, T)$ is large
- intuitively, we want $\Pr[A(G, T_x) = T]$ large and $\Pr[A(G, T_y) = T]$ small
- but at the same time, ε-DP says they can't be too different

Lower bound 2/3: reduction to packing (intuition)



- take $T_x, T_y \in \mathcal{T}$ far apart
- for any T with $d(T_x, T)$ small: $d(T_y, T)$ is large
- intuitively, we want $\Pr[A(G, T_x) = T]$ large and $\Pr[A(G, T_y) = T]$ small
- but at the same time, ε-DP says they can't be too different
- if we find enough pairwise distant $T_1, \ldots, T_k$, we get a lower bound

Lower bound 3/3: finding a good packing

Theorem (simplified): Let $\mathcal{C} \subseteq \mathcal{T}$ satisfy $d(T_x, T_y) \geq D_G/10$ for all distinct $T_x, T_y \in \mathcal{C}$. Let A be a ε-DP mechanism for MST. Then $\exists \mathbf{w}$ such that $A(G, \mathbf{w})$ has $\Omega(\log |\mathcal{C}|/\varepsilon) - \mathcal{O}(1)$ expected error.

Lower bound 3/3: finding a good packing

Theorem (simplified): Let $\mathcal{C} \subseteq \mathcal{T}$ satisfy $d(T_x, T_y) \geq D_G/10$ for all distinct $T_x, T_y \in \mathcal{C}$. Let A be a ε-DP mechanism for MST. Then $\exists \mathbf{w}$ such that $A(G, \mathbf{w})$ has $\Omega(\log |\mathcal{C}|/\varepsilon) - \mathcal{O}(1)$ expected error.

To finish the proof, we need to find a large $\mathcal{C}$:

Lower bound 3/3: finding a good packing

Theorem (simplified): Let $\mathcal{C} \subseteq \mathcal{T}$ satisfy $d(T_x, T_y) \geq D_G/10$ for all distinct $T_x, T_y \in \mathcal{C}$. Let A be a ε-DP mechanism for MST. Then $\exists \mathbf{w}$ such that $A(G, \mathbf{w})$ has $\Omega(\log |\mathcal{C}|/\varepsilon) - \mathcal{O}(1)$ expected error.

To finish the proof, we need to find a large $\mathcal{C}$:

- when G is a clique, we can find $\mathcal{C}$ greedily, $\log |\mathcal{C}| = \Theta(n \log n)$

Lower bound 3/3: finding a good packing

Theorem (simplified): Let $\mathcal{C} \subseteq \mathcal{T}$ satisfy $d(T_x, T_y) \geq D_G/10$ for all distinct $T_x, T_y \in \mathcal{C}$. Let A be a ε-DP mechanism for MST. Then $\exists \mathbf{w}$ such that $A(G, \mathbf{w})$ has $\Omega(\log |\mathcal{C}|/\varepsilon) - \mathcal{O}(1)$ expected error.

To finish the proof, we need to find a large $\mathcal{C}$:

- when G is a clique, we can find $\mathcal{C}$ greedily, $\log |\mathcal{C}| = \Theta(n \log n)$
- for general G, it turns out that $\mathcal{T}$ embeds $\mathbb{Z}_2^D$, and we can find $\mathcal{C}$ with $\log |\mathcal{C}| = \Theta(D_G)$ using error-correcting codes

Lower bound 3/3: finding a good packing

Theorem (simplified): Let $\mathcal{C} \subseteq \mathcal{T}$ satisfy $d(T_x, T_y) \geq D_G/10$ for all distinct $T_x, T_y \in \mathcal{C}$. Let A be a ε-DP mechanism for MST. Then $\exists \mathbf{w}$ such that $A(G, \mathbf{w})$ has $\Omega(\log |\mathcal{C}|/\varepsilon) - \mathcal{O}(1)$ expected error.

To finish the proof, we need to find a large $\mathcal{C}$:

- when G is a clique, we can find $\mathcal{C}$ greedily, $\log |\mathcal{C}| = \Theta(n \log n)$
- for general G, it turns out that $\mathcal{T}$ embeds $\mathbb{Z}_2^D$, and we can find $\mathcal{C}$ with $\log |\mathcal{C}| = \Theta(D_G)$ using error-correcting codes

$$\Rightarrow \Omega(n \log n/\varepsilon) - \mathcal{O}(1) \text{ worst-case l.b. } (\exists G)$$
$$\& \quad \Omega(D_G/\varepsilon) - \mathcal{O}(1) \text{ universal l.b. } (\forall G)$$

Conclusion

Further results:

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)
- exponential mechanism can be implemented in poly time, gives error $\mathcal{O}(\log |\mathcal{T}|/\varepsilon) = \mathcal{O}(D_G \log n/\varepsilon)$

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)
- exponential mechanism can be implemented in poly time, gives error $\mathcal{O}(\log |\mathcal{T}|/\varepsilon) = \mathcal{O}(D_G \log n/\varepsilon)$

Open questions:

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)
- exponential mechanism can be implemented in poly time, gives error $\mathcal{O}(\log |\mathcal{T}|/\varepsilon) = \mathcal{O}(D_G \log n/\varepsilon)$

Open questions:

- gap between $\mathcal{O}(D \log n/\varepsilon)$ u.b. and $\Omega(D/\varepsilon)$ l.b.
  - can we make the bounds tight?

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)
- exponential mechanism can be implemented in poly time, gives error $\mathcal{O}(\log |\mathcal{T}|/\varepsilon) = \mathcal{O}(D_G \log n/\varepsilon)$

Open questions:

- gap between $\mathcal{O}(D \log n/\varepsilon)$ u.b. and $\Omega(D/\varepsilon)$ l.b.
  - can we make the bounds tight?
  - $D \leq \log |\mathcal{T}| \leq D \log n$
  - ... so maybe $\log |\mathcal{T}|$ is the right graph parameter?

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)
- exponential mechanism can be implemented in poly time, gives error $\mathcal{O}(\log |\mathcal{T}|/\varepsilon) = \mathcal{O}(D_G \log n/\varepsilon)$

Open questions:

- gap between $\mathcal{O}(D \log n/\varepsilon)$ u.b. and $\Omega(D/\varepsilon)$ l.b.
  - can we make the bounds tight?
  - $D \leq \log |\mathcal{T}| \leq D \log n$
  - ... so maybe $\log |\mathcal{T}|$ is the right graph parameter?
- can similar approach work for other problems (SSSP)?

Conclusion

Further results:

- Sealfon's algorithm has $\mathcal{O}(D_G \log n/\varepsilon)$ error
- results for ℓ_∞-neighborhood ($\|\mathbf{w} - \mathbf{w}'|_\infty \leq 1$)
- exponential mechanism can be implemented in poly time, gives error $\mathcal{O}(\log |\mathcal{T}|/\varepsilon) = \mathcal{O}(D_G \log n/\varepsilon)$

Open questions:

- gap between $\mathcal{O}(D \log n/\varepsilon)$ u.b. and $\Omega(D/\varepsilon)$ l.b.
  - can we make the bounds tight?
  - $D \leq \log |\mathcal{T}| \leq D \log n$
  - ... so maybe $\log |\mathcal{T}|$ is the right graph parameter?
- can similar approach work for other problems (SSSP)?

Thank you!

Bibliography I

[Sea16] Adam Sealfon. "Shortest Paths and Distances with Differential Privacy". In: *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 – July 01, 2016*. ACM, 2016, pp. 29–41. DOI: 10.1145/2902251.2902291. URL: https://doi.org/10.1145/2902251.2902291.

[Vad17] Salil Vadhan. "The complexity of differential privacy". In: *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich* (2017), pp. 347–450. DOI: 10.1007/978-3-319-57048-8_7.

Backup Slides