CPSC 768:
Scalable and Private Graph Algorithms

Lecture 25: Learning-Augmented Algorithms

Quanquan C. Liu
quanquan.liu@yale.edu

Announcements

- **Final project report and presentation: April 24th (last day of class)**
  - Final project presentation is a 30 min presentation
- **Last day of Open Problem Sessions: April 26th** (last week of classes)
  - Will be turned into a reading group/continue with OPS, stay tuned!

What are learning-augmented algorithms?

- **Classical Algorithms (intro to algorithms courses)**
  - Worst-case guarantees
  - Limited adaptivity to input (special classes, closely related inputs)

What are learning-augmented algorithms?

- **Classical Algorithms (intro to algorithms courses)**
  - Worst-case guarantees
  - Limited adaptivity to input (special classes, closely related inputs)

- **Machine Learning Based Approaches**
  - Stronger performance due to adaptivity (learning) inputs
  - No worst-case guarantees

What are learning-augmented algorithms?

- **Classical Algorithms (intro to algorithms courses)**
  - Worst-case guarantees
  - Limited adaptivity to input (special classes, closely related inputs)
- **Machine Learning Based Approaches**
  - Stronger performance due to adaptivity (learning) inputs
  - No worst-case guarantees

- **Learning-Augmented Algorithms**
  - Adaptive
  - Often has worst-case guarantees

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**
  - What part of the **input can we reasonably learn**?

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**
  - What part of the **input can we reasonably learn**?
  - How do we use the **learned advice** to **obtain better theoretically proven bounds**?

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**
  - What part of the **input can we reasonably learn**?
  - How do we use the **learned advice** to **obtain better theoretically proven bounds**?
  - **Example:**

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**
  - What part of the **input can we reasonably learn**?
  - How do we use the **learned advice** to **obtain better theoretically proven bounds**?
  - **Example:**
    - Frequency estimation in a stream

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**
  - What part of the **input can we reasonably learn**?
  - How do we use the **learned advice** to **obtain better theoretically proven bounds**?
  - **Example:**
    - Frequency estimation in a stream—learn infrequent elements

What are learning-augmented algorithms?

- **Use machine learning advice to inform algorithmic procedure**
  - What part of the **input can we reasonably learn**?
  - How do we use the **learned advice** to **obtain better theoretically proven bounds**?
  - **Example:**
    - Frequency estimation in a stream
      - Learn infrequent elements

What Guarantees Do We Want?

- **Consistency**
  - If the predictions are **high quality**, then algorithms performs **much better than worst-case** algorithm

What Guarantees Do We Want?

- **Consistency**
  - If the predictions are **high quality**, then algorithm performs **much better than worst-case** algorithm
- **Competitiveness**
  - If the predictions are **low quality**, then **algorithm does not perform any worse than worst-case** algorithm

What Guarantees Do We Want?

- **Consistency**
  - If the predictions are **high quality**, then algorithm performs **much better than worst-case** algorithm
- **Competitiveness**
  - If the predictions are **low quality**, then **algorithm does not perform any worse than worst-case** algorithm
- **Robustness**
  - Performance of algorithm **degrades gracefully** as function of prediction error

What Guarantees Do We Want?

Often cannot satisfy all three!

- **Consistency**
  - If the predictions are **high quality**, then algorithm performs **much better than worst-case** algorithm
- **Competitiveness**
  - If the predictions are **low quality**, then **algorithm does not perform any worse than worst-case** algorithm
- **Robustness**
  - Performance of algorithm **degrades gracefully** as function of prediction error

Learned Binary Search

- Suppose we have a **sorted** list of integers

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; how can we use it?

1	3	6	7	8	9	11	15	16	18	19	25	36	78	98	99

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	3	6	7	8	9	11	15	16	18	19	25	36	78	98	99

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

First decide go left or right

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** predicted index of 15; **doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Start doubling search

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction: predicted index** of 15; **doubling search from predicted index!**

1	3	6	7	8	9	11	15	16	18	19	25	36	78	98	99

Start doubling search

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction: predicted index** of 15; **doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

→

Start doubling search

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	3	6	7	8	9	11	15	16	18	19	25	36	78	98	99

→

Start doubling search

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Start doubling search

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Start doubling search

Learned Binary Search

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Start doubling search

What is the runtime?

Learned Binary Search

Why is the prediction reasonable?

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** **predicted index** of 15; **doubling search from predicted index!**

1	3	6	7	8	9	11	15	16	18	19	25	36	78	98	99

Start doubling search

What is the runtime?

$$\boldsymbol{O}(\log(\boldsymbol{L_1} - \boldsymbol{error}))$$

Learned Binary S

Why is the prediction reasonable?

Many real-world instances give you a prediction—e.g. library books

- Suppose we have a **sor**
- **Problem:** where is 15?
- **Prediction:** **predicted index of 15, doubling search from predicted index!**

1	**3**	**6**	**7**	**8**	**9**	**11**	**15**	**16**	**18**	**19**	**25**	**36**	**78**	**98**	**99**

Start doubling search

What is the runtime?

$$\boldsymbol{O}(\log(\boldsymbol{L_1} - \boldsymbol{error}))$$

Learned Binary

Satisfies all three desired properties!

- Suppose we have a **sorted** list of integers
- **Problem:** where is 15?
- **Prediction:** predicted index of 15; **doubling search from predicted index!**

1	3	6	7	8	9	11	15	16	18	19	25	36	78	98	99

Start doubling search

What is the runtime?

$$\boldsymbol{O}(\log(\boldsymbol{L_1} - \boldsymbol{error}))$$

Learned Bloom Filter

- A bloom filter is a data structure that provides **approximate membership queries**
  - Given: a set of elements $S = \{x_1, \dots, x_n\}$

Learned Bloom Filter

- A bloom filter is a data structure that provides **approximate membership queries**
  - Given: a set of elements $S = \{x_1, \ldots, x_n\}$
  - Goal: given x, determine if x is in S

Learned Bloom Filter

- A bloom filter is a data structure that provides **approximate membership queries**
  - Given: a set of elements $S = \{x_1, \ldots, x_n\}$
  - Goal: given x, determine if x is in S
- Bloom filter gives **low space**, **randomized solution**

Learned Bloom Filter

- A bloom filter is a data structure that provides **approximate membership queries**
  - Given: a set of elements $S = \{x_1, \dots, x_n\}$
  - Goal: given x, determine if x is in S
- Bloom filter gives **low space**, **randomized solution**
- **Many applications!**

Learned Bloom Filter

- **Bloom filter contains:**

Learned Bloom Filter

- **Bloom filter contains:**
  - Array of m bits, which represents S

Learned Bloom Filter

- **Bloom filter contains:**
  - Array of m bits, which represents S
  - k independent hash functions $h_1, \dots, h_k$, each hashing elements in S to $\{1, \dots, m\}$

Learned Bloom Filter

- **Bloom filter contains:**
  - Array of m bits, which represents S
  - k independent hash functions $h_1, \ldots, h_k$, each hashing elements in S to $\{1, \ldots, m\}$
- **Initialization:** for each $x \in S$ and $i \in \{1, \ldots, k\}$, set bit $h_i(x)$ in array to 1

Learned Bloom Filter

- **Bloom filter contains:**
  - Array of m bits, which represents S
  - k independent hash functions $h_1, \ldots, h_k$, each hashing elements in S to $\{1, \ldots, m\}$
- **Initialization:** for each $x \in S$ and $i \in \{1, \ldots, k\}$, set bit $h_i(x)$ in array to 1
- **On query of x':**
  - **Report in S** if and only if $\boldsymbol{h_i(x') = 1}$ **for all** $\boldsymbol{i \in \{1, \ldots, k\}}$

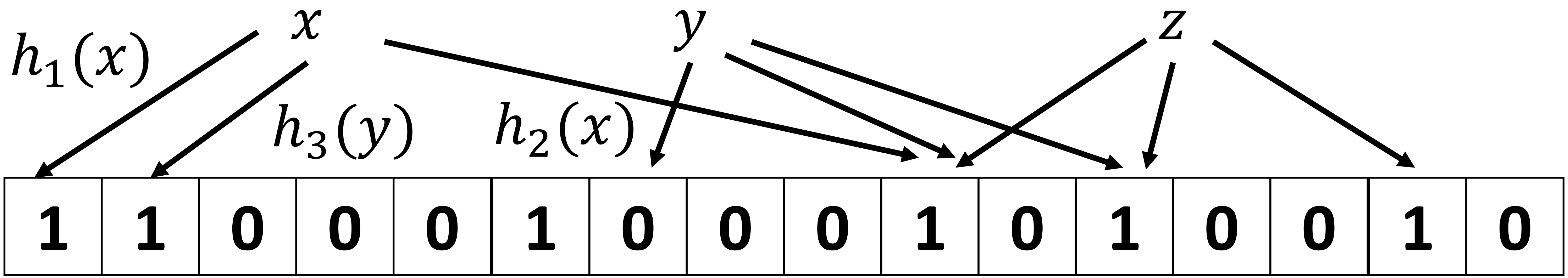
Learned Bloom Filter

- **Initialization:** for each $x \in S$ and $i \in \{1, \dots, k\}$, set bit $h_i(x)$ in array to 1
- **On query of x':**
  - **Report in S** if and only if $\boldsymbol{h_i(x') = 1}$ **for all** $\boldsymbol{i \in \{1, \dots, k\}}$

1	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**

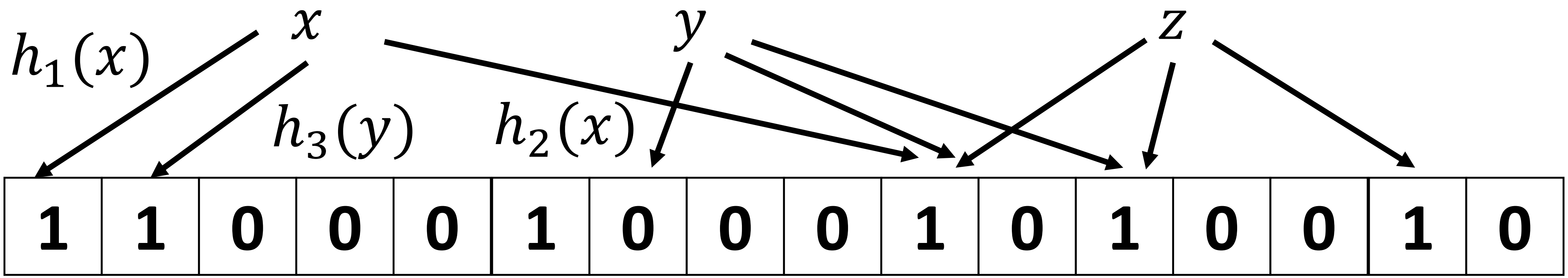
Learned Bloom Filter

- **Initialization:** for each $x \in S$ and $i \in \{1, \ldots, k\}$, set bit $h_i(x)$ in array to 1
- **On query of x':**
  - **Report in S** if and only if $\boldsymbol{h_i(x') = 1}$ **for all** $\boldsymbol{i \in \{1, \ldots, k\}}$



Learned Bloom Filter

- **Guarantees:**
  - **No false negatives:** if return NO, not in S
  - **False positives:** returns YES but not in S



Learned Bloom Filter

- **Guarantees:**
  - **No false negatives:** if return NO, not in S
  - **False positives:** returns YES but not in S
    - **Probability:** For query x and for each $i \in [k]$, $P[h_i(x) = 1] \leq \frac{kn}{m}$

Learned Bloom Filter

- **Guarantees:**
  - **No false negatives:** if return NO, not in S
  - **False positives:** returns YES but not in S
    - **Probability:** For query x and for each $i \in [k]$, $P[h_i(x) = 1] \leq \frac{kn}{m}$
    - $P[all\ 1s] \leq \left(\frac{kn}{m}\right)^k$

Learned Bloom Filter

- **Guarantees:**
  - **No false negatives:** if return NO, not in S
  - **False positives:** returns YES but not in S
    - **Probability:** For query x and for each $i \in [k]$, $P[h_i(x) = 1] \leq \frac{kn}{m}$
    - $P[all\ 1s] \leq \left(\frac{kn}{m}\right)^k$
    - Suppose pick $k = \log_2 n$ and $m = 3n \log_2 n$, then probability of failure is $\left(\frac{n \log_2 n}{3n \log_2 n}\right)^{\log_2 n} = \left(\frac{1}{3}\right)^{\log_2 n} < \frac{1}{n}$

Learned Bloom Filter

- **ML Oracle:**

Learned Bloom Filter

- **ML Oracle:**
  - Some **trained function $f(x)$ provides useful information**

Learned Bloom Filter

- **ML Oracle:**
  - Some **trained function $f(x)$ provides useful information**
- What function would be useful for us?

Learned Bloom Filter

- **ML Oracle:**
  - Some **trained function $f(x)$ provides useful information**
- What function would be useful for us?
  - Function that tells us whether $x \in S$!

Learned Bloom Filter

- **ML Oracle:**
  - Some **trained function** $f(x)$ **provides useful information**
- What function would be useful for us?
  - Function that tells us whether $x \in S$!
- **Caveat:**
  - We can’t learn this exactly!

Learned Bloom Filter

- **ML Oracle:**
  - Some **trained function $f(x)$ provides useful information**
- What function would be useful for us?
  - Function that tells us whether $x \in S$!
- **Caveat:**
  - We can't learn this exactly!
  - Will be non-zero **false positive and negative rate**

Learned Bloom Filter

- [Kraska, Beutal, Chi, Dean, Polyzotis, SIGMOD ‘18]

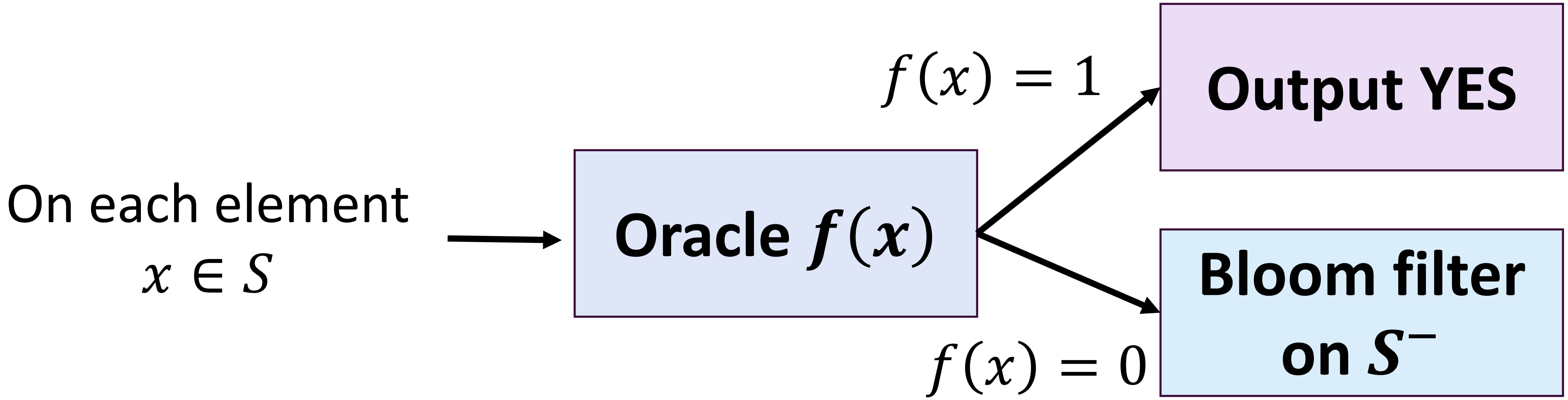
Learned Bloom Filter

- [Kraska, Beutal, Chi, Dean, Polyzotis, SIGMOD '18]

On each element
$x \in S$ → **Oracle** $\boldsymbol{f(x)}$

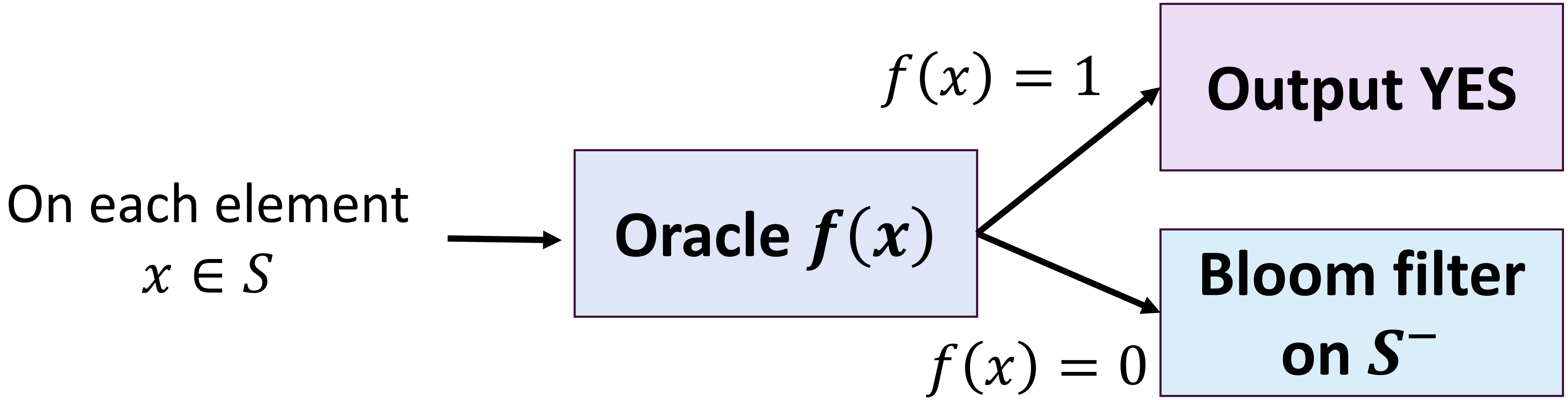
Learned Bloom Filter

- [Kraska, Beutal, Chi, Dean, Polyzotis, SIGMOD ‘18]



Learned Bloom Filter

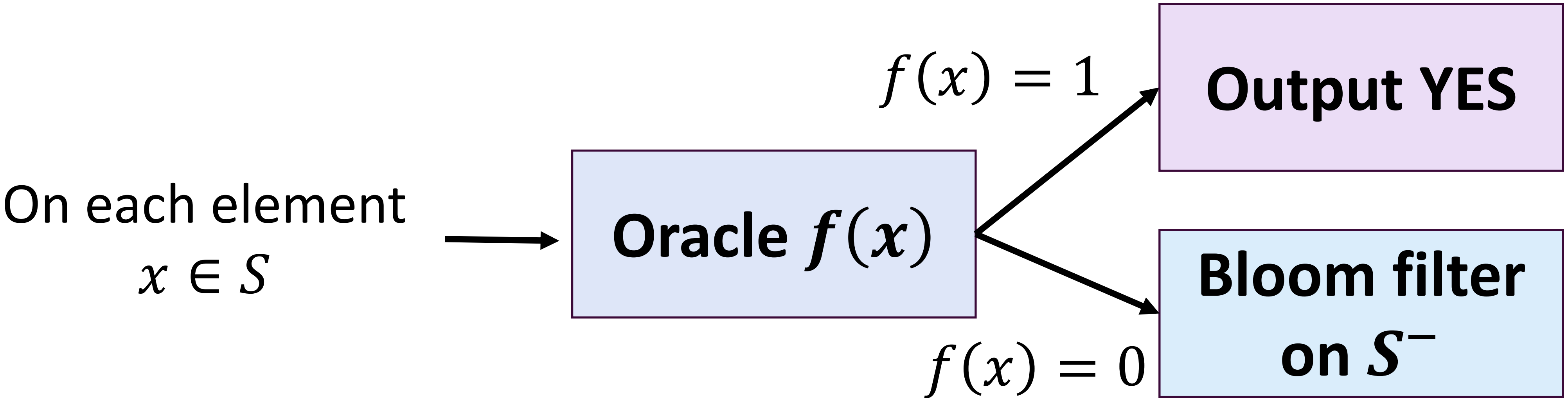
- [Kraska, Beutal, Chi, Dean, Polyzotis, SIGMOD '18]



No false negatives!

Learned Bloom Filter

- [Kraska, Beutal, Chi, Dean, Polyzotis, SIGMOD ‘18]

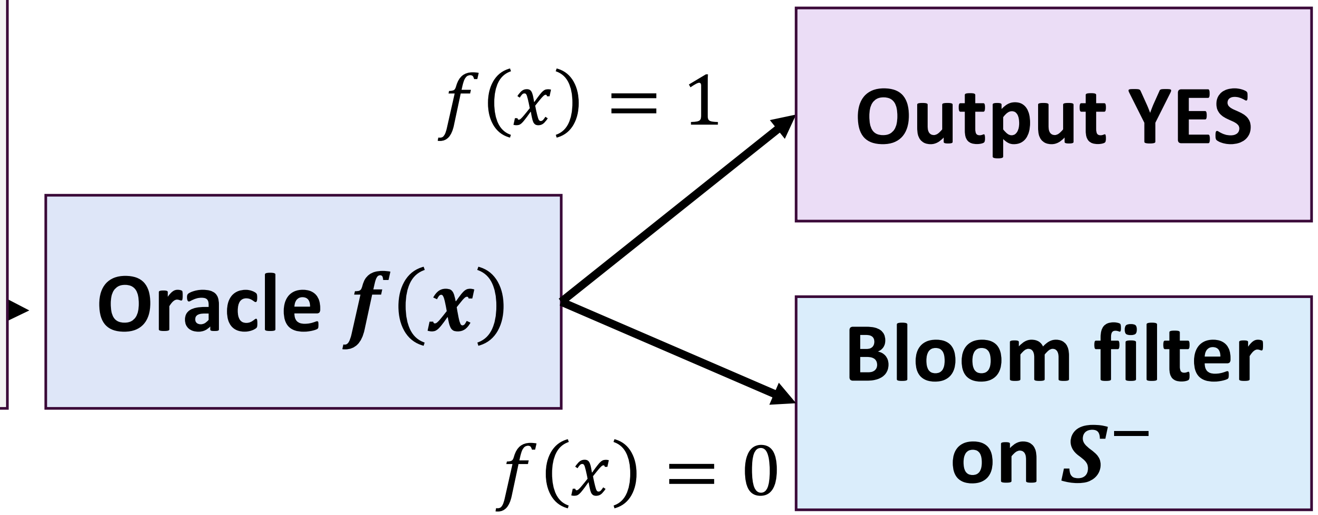


No false negatives!

Learned Bloom Filter

Chi, Dean, Polyzotis, SIGMOD ‘18]

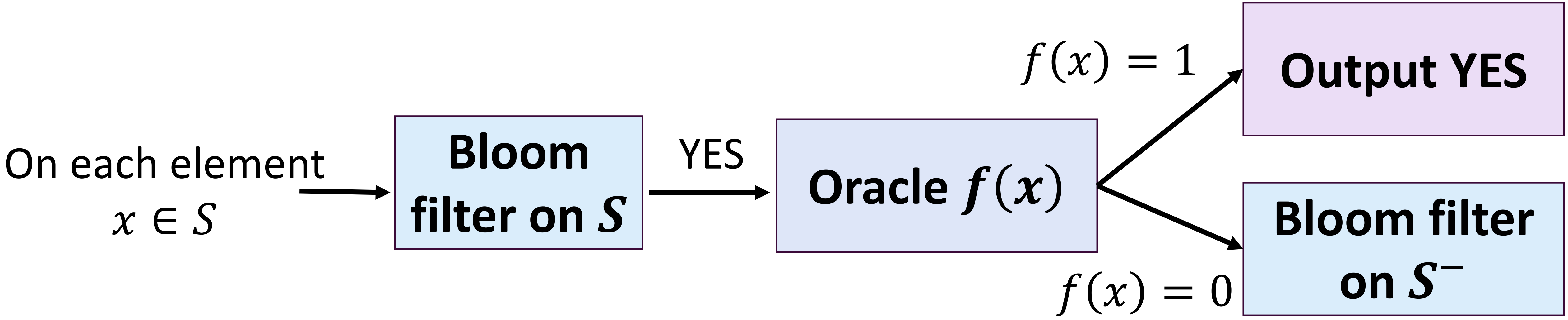
Caveat: false positive rate depends on the oracle



No false negatives!

Sandwiched Learned Bloom Filter

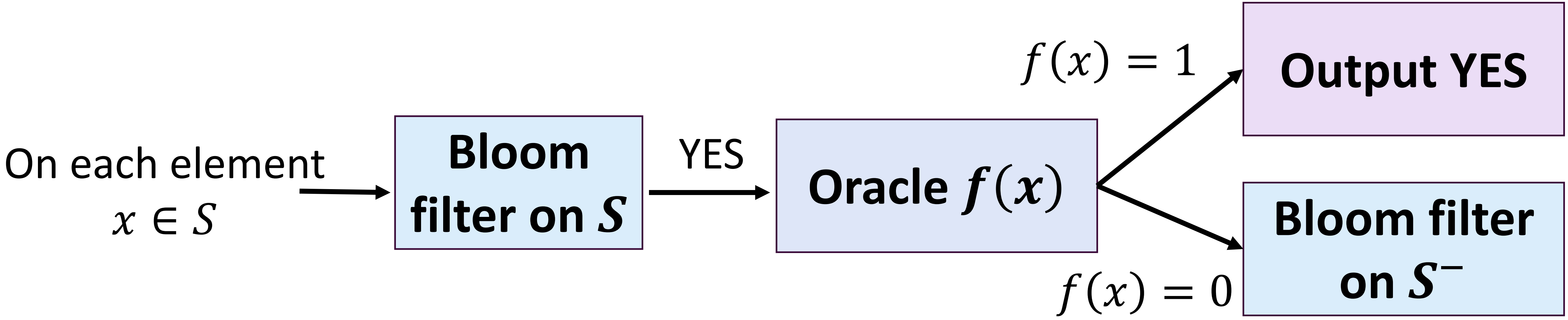
- [Mitzenmacher, NeurIPS ‘18]



No false negatives!

Sandwiched Learned Bloom Filter

- [Mitzenmacher, NeurIPS ‘18]

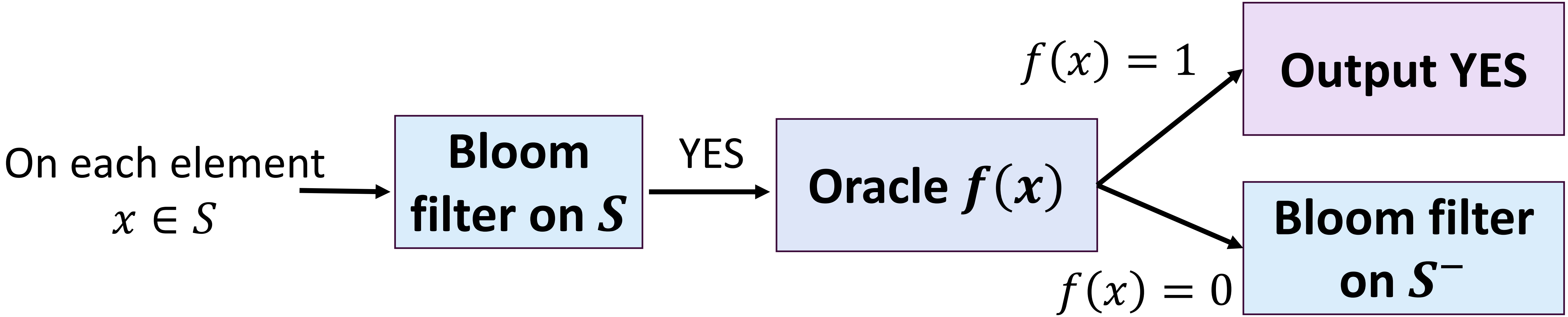


False positive rate same or less than Bloom filter!

Sandwiched Learned Bloom Filter

- [Mitzenmacher, NeurIPS '18]

Better analysis and performance in practice!



False positive rate same or less than Bloom filter!

Learning-Augmented Dynamic Graph Algorithms

- Why do we need them?

Types of Dynamic Algorithms

- **Incremental/Decremental** vs. **Fully Dynamic**
  - Incremental/decremental algorithms:
    - Only edge insertions/deletions, respectively
- Sometimes **large gap in runtimes**
  - **Polynomial** or **exponential gaps** in runtimes

Types of Dynamic Algorithms

	Best Fully Dynamic		Best Partially Dynamic	
Planar Digraph APSP	$\widetilde{O}\left(n^{2/3}\right)$	[FR06, Kle05]	$\widetilde{O}(\sqrt{n})$	[DGWN22]
Triconnectivity	$O(n^{2/3})$	[GIS99]	$\widetilde{O}(1)$	[HR20, PSS17]
k-Edge Connectivity	$n^{o(1)}$	[JS22]	$\widetilde{O}(1)$	[CDK+21]
Dynamic DFS Tree	$\widetilde{O}\left(\sqrt{mn}\right)$	[BCCK19]	$\widetilde{O}(n)$	[BCCK19, CDW+18]
Minimum Spanning Forest	$\widetilde{O}(1)$	[HDLT01]	$\widetilde{O}(1)$	[Epp94]
APSP	$\left(\frac{256}{k^2}\right)^{4/k}$-Approx $\widetilde{O}\left(n^k\right)$ update $\widetilde{O}(n^{k/8})$ query	[FGNS23]	$(2r-1)^k$-Approx $\widetilde{O}\left(m^{1/(k+1)}n^{k/r}\right)$	[CGH+20]
AP Maxflow/Mincut	$O(\log(n)\log\log n)$-Approx $\widetilde{O}\left(n^{2/3+o(1)}\right)$	[CGH+20]	$O\left(\log^{8k}(n)\right)$-Approx. $\widetilde{O}\left(n^{2/(k+1)}\right)$	[Gor19, GHS19]
MCF	$(1+\varepsilon)$-Approx $\widetilde{O}(1)$ update $\widetilde{O}(n)$ query	[CGH+20]	$O(\log^{8k}(n))$-Approx. $\widetilde{O}\left(n^{2/(k+1)}\right)$ update $\widetilde{O}(P^2)$ query	[Gor19, GHS19]
Strongly Connected Components	$\Omega(m^{1-\varepsilon})$ query or update	[AW14]	$\widetilde{O}(m)$	[Rod13]
Uniform Sparsest Cut	$2^{O(\log^{5/6}(n))}$-Approx $2^{O(\log^{5/6}(n))}$ update $O(\log^{1/6}(n))$ query	[GRST21]	$O\left(\log^{8k}(n)\right)$-Approx $\widetilde{O}\left(n^{2/(k+1)}\right)$ $O(1)$ query	[Gor19, GHS19]
Submodular Max	$1/4$-Approx $\widetilde{O}(k^2)$	[DFL+23]	0.3178-Approx $\widetilde{O}(\text{poly}(k))$	[FLN+22]

[LS ’23]

Learning-Augmented Dynamic Graph Algorithms

- Why do we need them?
- Can we use predictions to bridge the gap?

Learning-Augmented Dynamic Graph Algorithms

- Why do we need them?
- Can we use predictions to bridge the gap?
  - Yes!

Learning-Augmented Dynamic Graph Algorithms

- Why do we need them?
- Can we use predictions to bridge the gap?
  - Yes!
- **Predictions on the edge updates [L-Srinivas ‘23]**

Learning-Augmented Dynamic Graph Algorithms

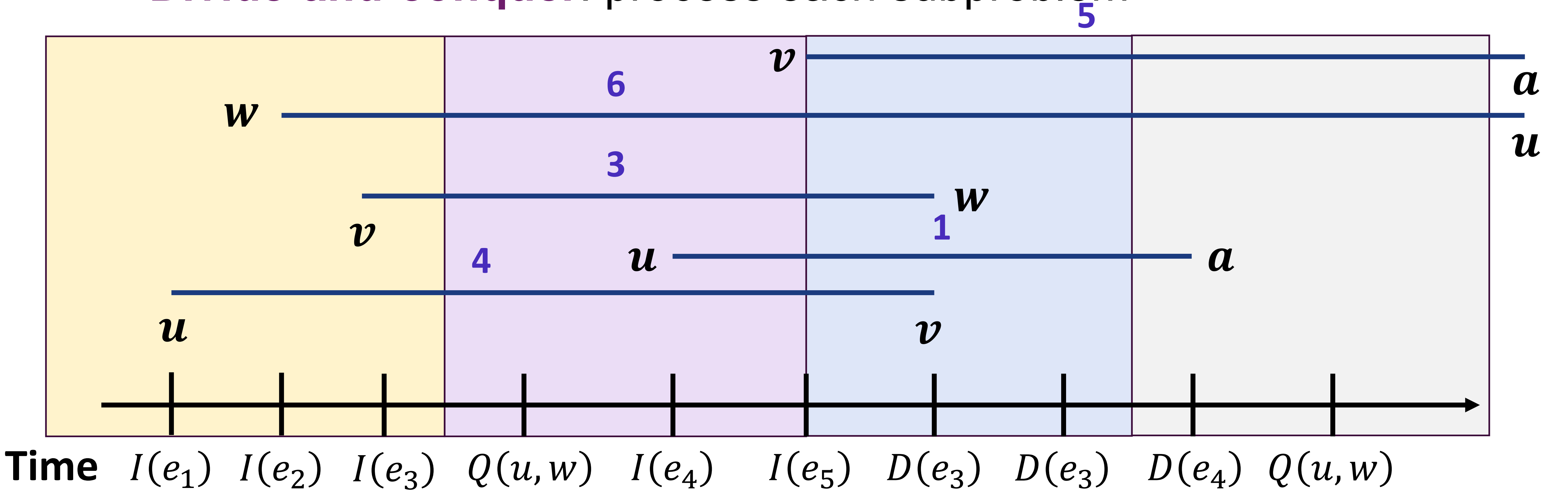
- Why do we need them?
- Can we use predictions to bridge the gap?
  - Yes!
- **Predictions on the edge updates [L-Srinivas '23]**
  - For each edge update, give **prediction on when update occurs**

Learning-Augmented Dynamic Graph Algorithms

- Why do we need them?
- Can we use predictions to bridge the gap?
  - Yes!
- **Predictions on the edge updates [L-Srinivas '23]**
  - For each edge update, give **prediction on when update occurs**
  - Assume one edge insertion/deletion occurs on a day, give prediction on the day of the edge insertion/deletion

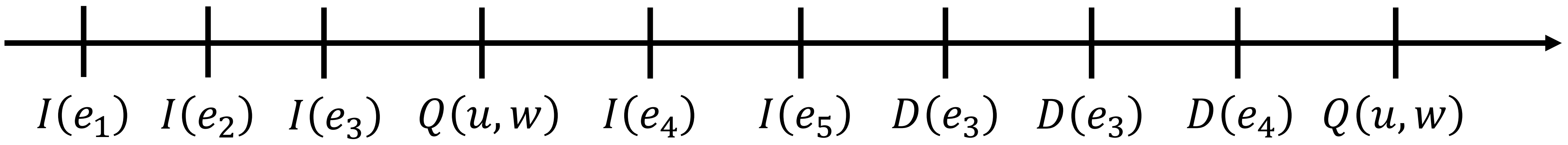
Offline-Dynamic Connectivity

- Geometric representation of the problem
- **Divide-and-conquer**: process each subproblem

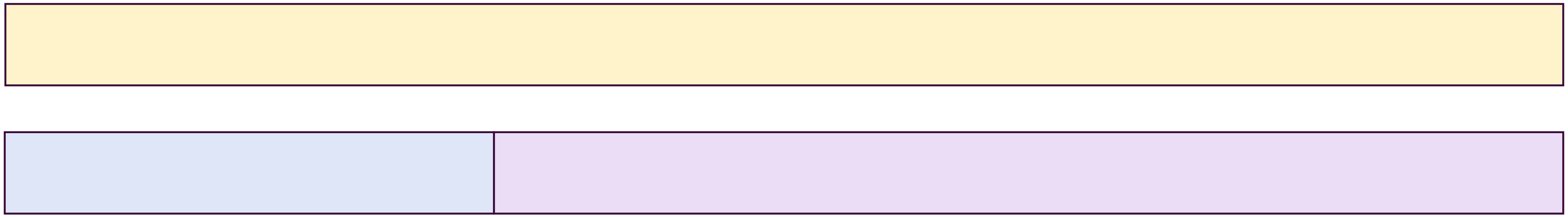


Random Partition Tree Data Structure

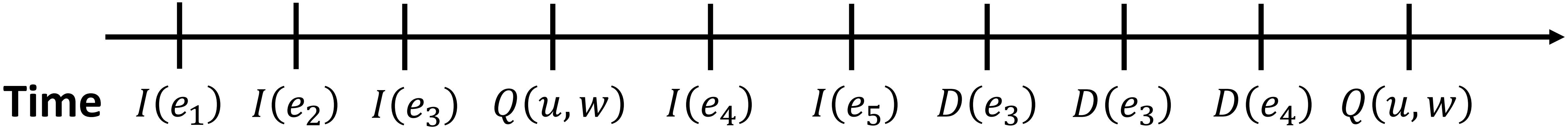
Pick a uniformly at random divider for subproblems



Random Partition Tree Data Structure

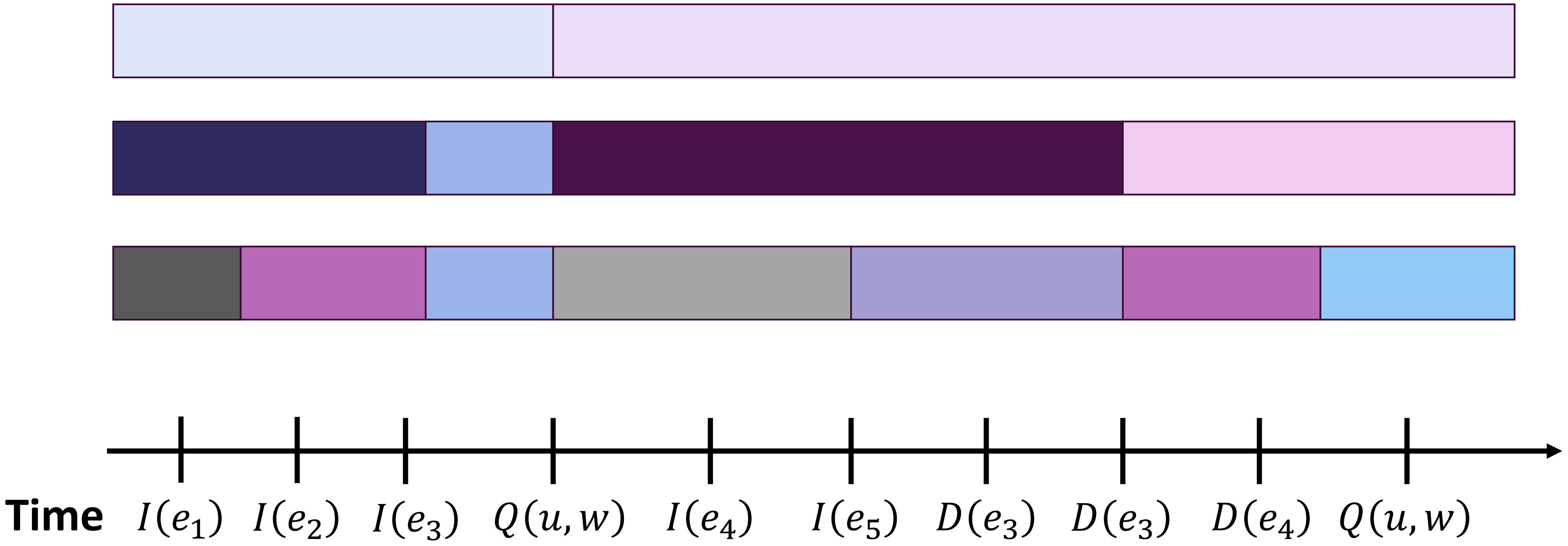


Pick a uniformly at random divider for subproblems



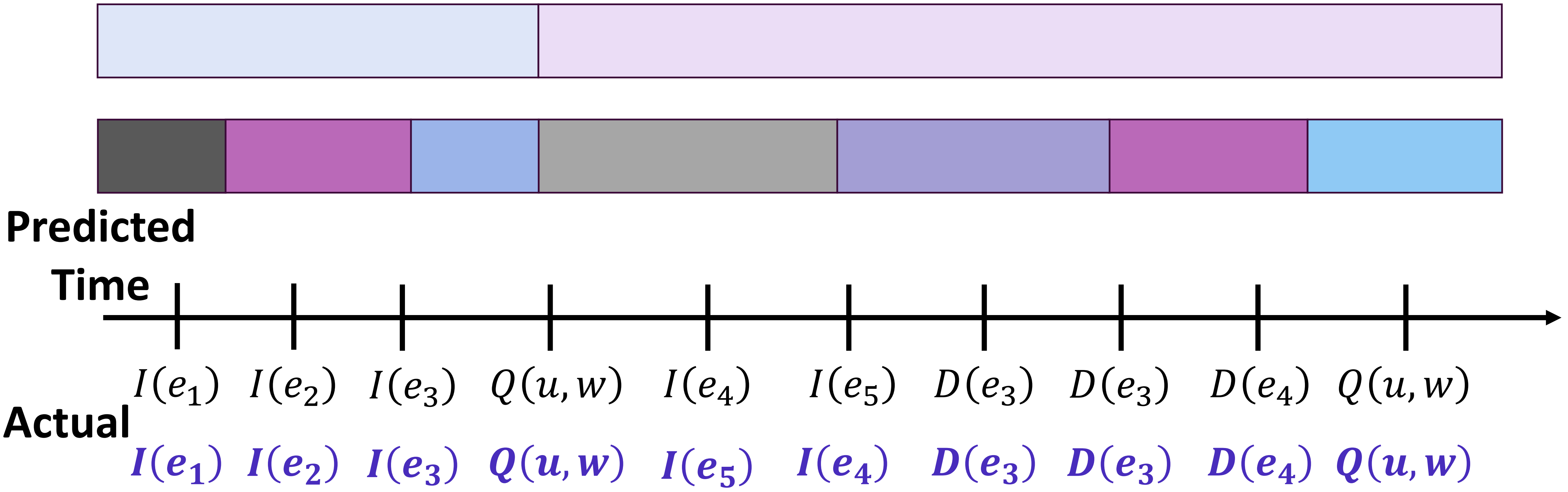
Random Partition Tree Data Structure

Run offline divide-and-conquer algorithm on subproblems



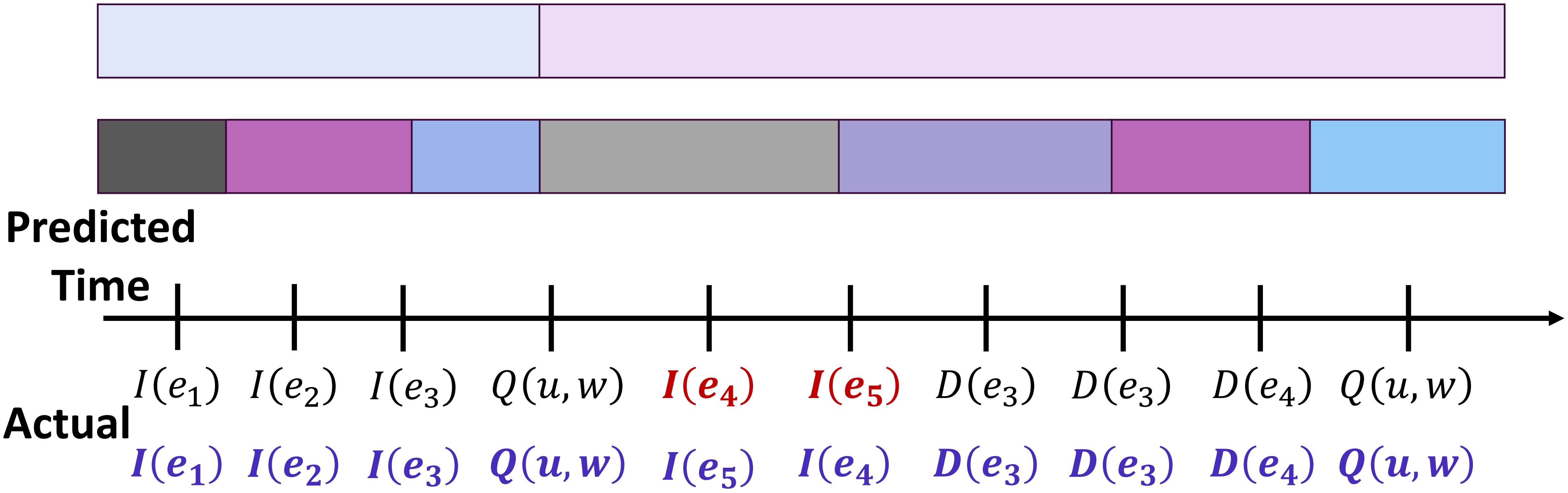
Random Partition Tree Data Structure

Run offline divide-and-conquer algorithm on subproblems



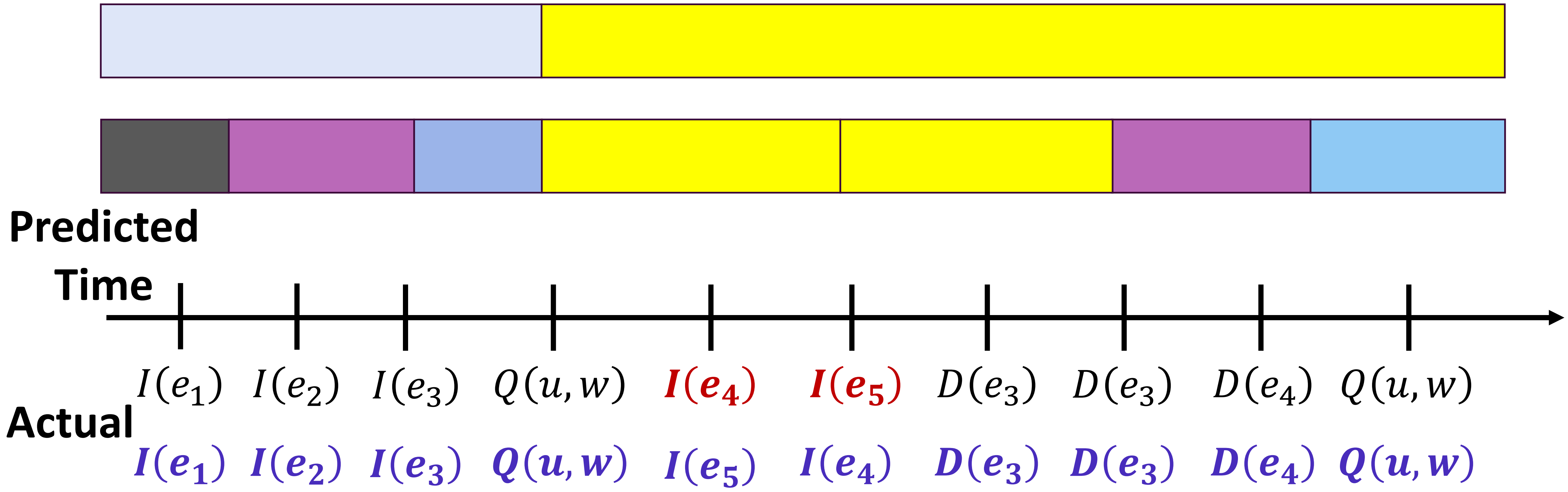
Random Partition Tree Data Structure

Run offline divide-and-conquer algorithm on subproblems



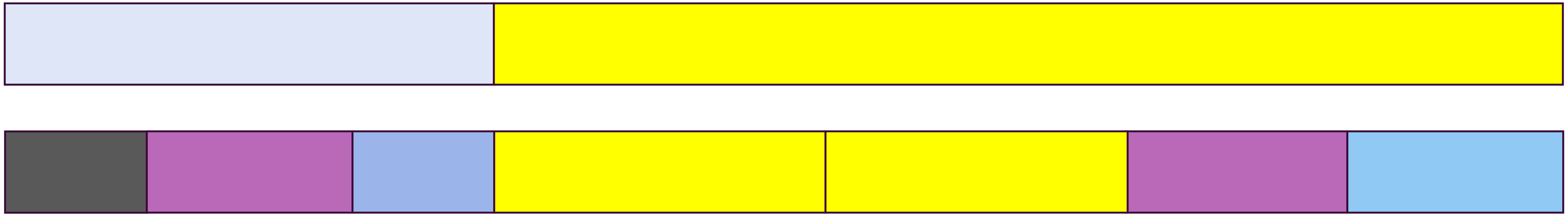
Random Partition Tree Data Structure

Recompute computation of largest subtree containing errors and children



Random Partition Tree Data Structure

Purpose of the random partition tree: in expectation size of subproblem (largest subtree) equal to error



Predicted

Time

$I(e_1)$ $I(e_2)$ $I(e_3)$ $Q(u,w)$ $\boldsymbol{I(e_4)}$ $\boldsymbol{I(e_5)}$ $D(e_3)$ $D(e_3)$ $D(e_4)$ $Q(u,w)$

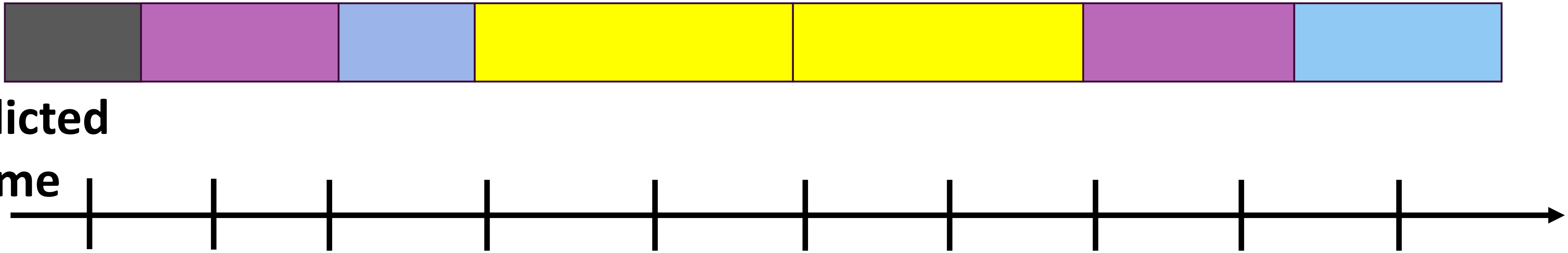
Actual

$\boldsymbol{I(e_1)}$ $\boldsymbol{I(e_2)}$ $\boldsymbol{I(e_3)}$ $\boldsymbol{Q(u,w)}$ $\boldsymbol{I(e_5)}$ $\boldsymbol{I(e_4)}$ $\boldsymbol{D(e_3)}$ $\boldsymbol{D(e_3)}$ $\boldsymbol{D(e_4)}$ $\boldsymbol{Q(u,w)}$

Random Partition Tree Data Structure

Purpose of the random partition tree: in expectation size of subproblem (largest subtree) equal to L_1 error

Bad event: very large subtree for small error



Predicted Time

$I(e_1)$ $I(e_2)$ $I(e_3)$ $Q(u,w)$ $\boldsymbol{I(e_4)}$ $\boldsymbol{I(e_5)}$ $D(e_3)$ $D(e_3)$ $D(e_4)$ $Q(u,w)$

Actual

$\boldsymbol{I(e_1)}$ $\boldsymbol{I(e_2)}$ $\boldsymbol{I(e_3)}$ $\boldsymbol{Q(u,w)}$ $\boldsymbol{I(e_5)}$ $\boldsymbol{I(e_4)}$ $\boldsymbol{D(e_3)}$ $\boldsymbol{D(e_3)}$ $\boldsymbol{D(e_4)}$ $\boldsymbol{Q(u,w)}$

Random Partition Tree Data Structure

Runtime same as partially dynamic with an additional $\widetilde{O}(\log(L_1 - error))$

[L-Srinivas '23]

Predicte

Time

$I(e_1)$ $I(e_2)$ $I(e_3)$ $Q(u,w)$ $\boldsymbol{I(e_4)}$ $\boldsymbol{I(e_5)}$ $D(e_3)$ $D(e_3)$ $D(e_4)$ $Q(u,w)$

Actual

$\boldsymbol{I(e_1)}$ $\boldsymbol{I(e_2)}$ $\boldsymbol{I(e_3)}$ $\boldsymbol{Q(u,w)}$ $\boldsymbol{I(e_5)}$ $\boldsymbol{I(e_4)}$ $\boldsymbol{D(e_3)}$ $\boldsymbol{D(e_3)}$ $\boldsymbol{D(e_4)}$ $\boldsymbol{Q(u,w)}$

Conclusion

- **Streaming, distributed, parallel and dynamic techniques** can be used in a variety of problems across models

Conclusion

- **Streaming, distributed, parallel and dynamic techniques** can be used in a variety of problems across models
- **Modern algorithms must take into account stronger adversaries**
  - **Privacy-violating**
  - **Decentralized adversaries**

Conclusion

- **Streaming, distributed, parallel and dynamic techniques** can be used in a variety of problems across models
- **Modern algorithms must take into account stronger adversaries**
  - **Privacy-violating**
  - **Decentralized adversaries**
- **New tools to help us make algorithms practical!**

Conclusion

- **Streaming, distributed, parallel and dynamic techniques** can be used in a variety of problems across models
- **Modern algorithms must take into account stronger adversaries**
  - **Privacy-violating**
  - **Decentralized adversaries**
- **New tools to help us make algorithms practical!**
  - **Models that model modern architecture**
  - **ML-based methods**

Conclusion

- **Streaming, distributed, parallel and dynamic techniques** can be used in a variety of problems across models
- **Modern algorithms must take into account stronger adversaries**
  - **Privacy-violating**
  - **Decentralized adversaries**
- **New tools to help us make algorithms practical!**
  - **Models that model modern architecture**
  - **ML-based methods**
- **Graphs at the forefront of these developments!**

Conclusion

Exciting new fields with lots of potential for development and broad impact

- **Streaming, distributed, parallel and dynamic techniques** can be used in a variety of problems across models
- **Modern algorithms must take into account stronger adversaries**
  - **Privacy-violating**
  - **Decentralized adversaries**
- **New tools to help us make algorithms practical!**
  - **Models that model modern architecture**
  - **ML-based methods**
- **Graphs at the forefront of these developments!**