On Dynamic Graph Algorithms with Predictions

Jan van den Brand
Georgia Tech

Sebastian Forster (né Krinninger)
University of Salzburg

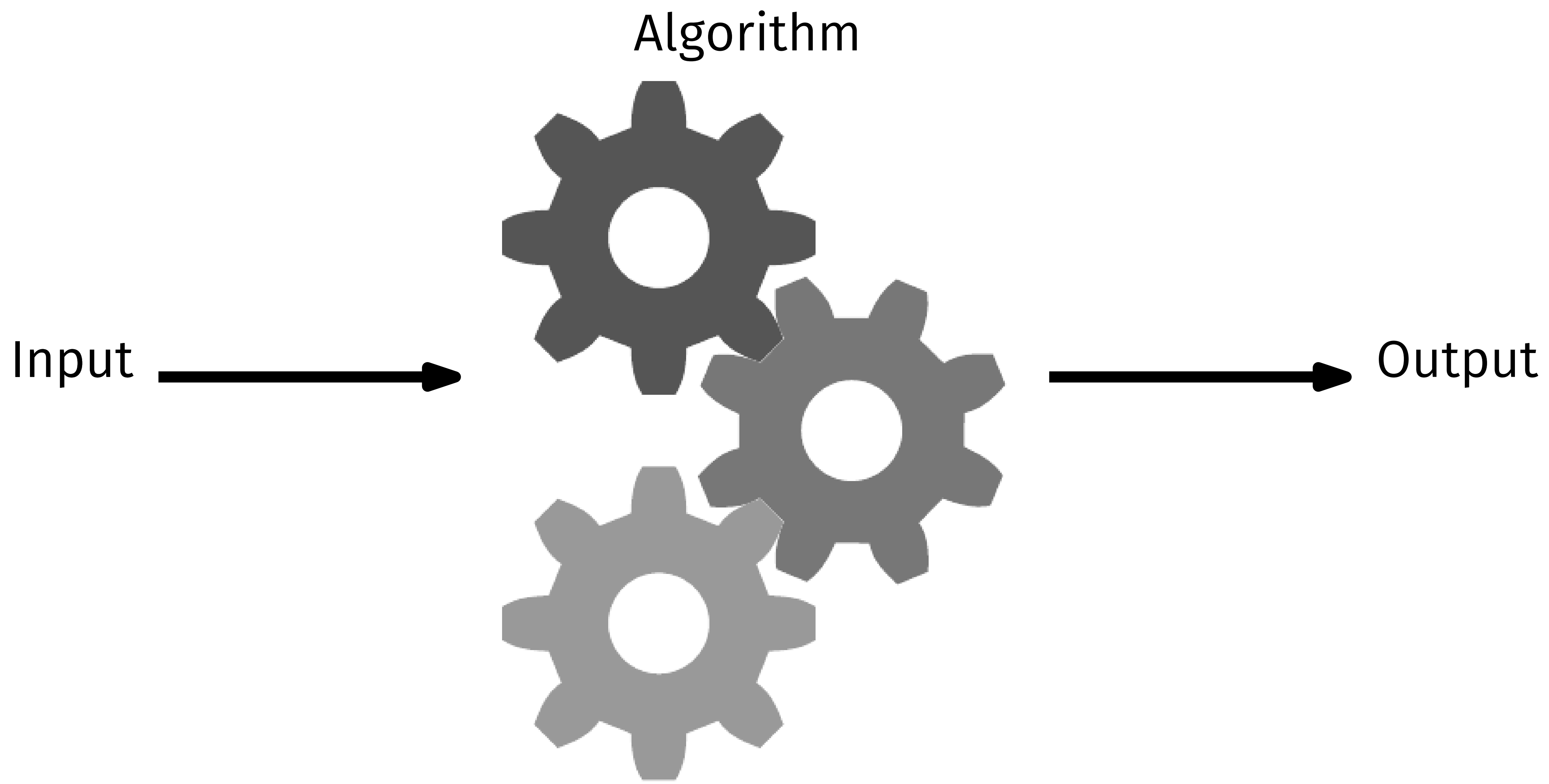
Yasamin Nazari
VU Amsterdam and CWI

Adam Polak
Bocconi University

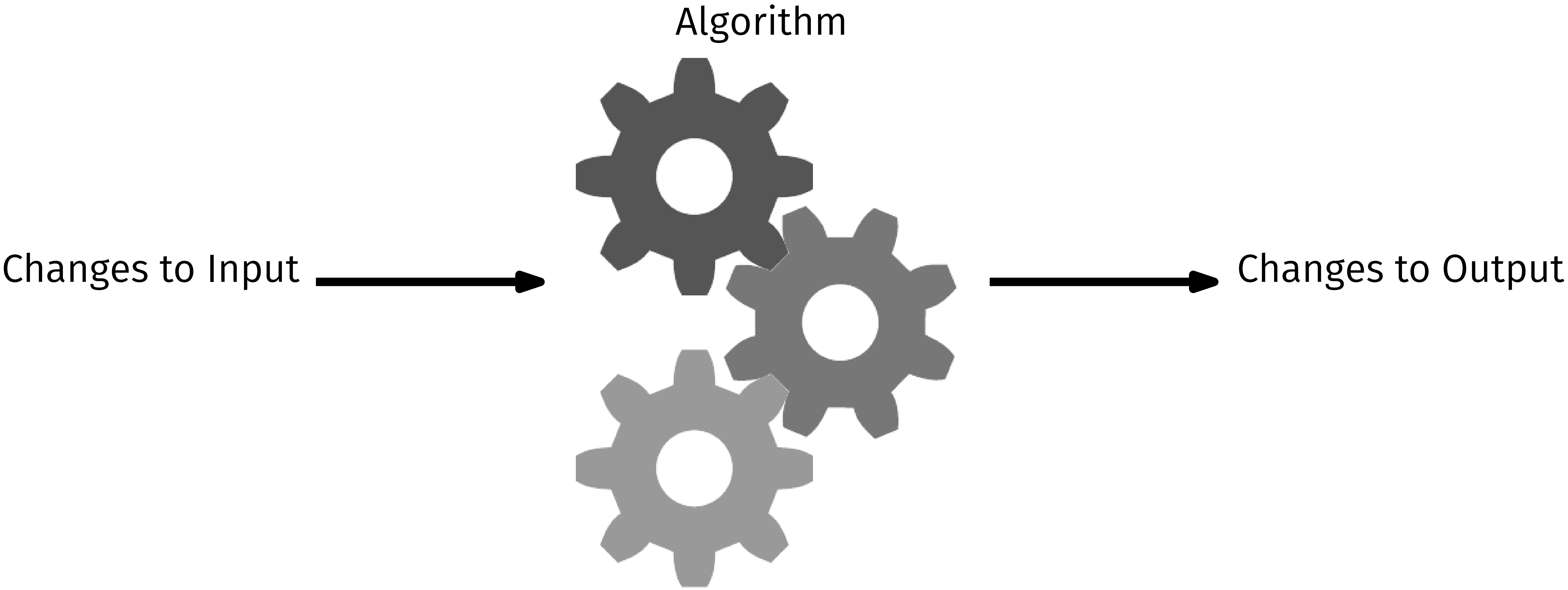
This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 947702)

Dynamic Environments

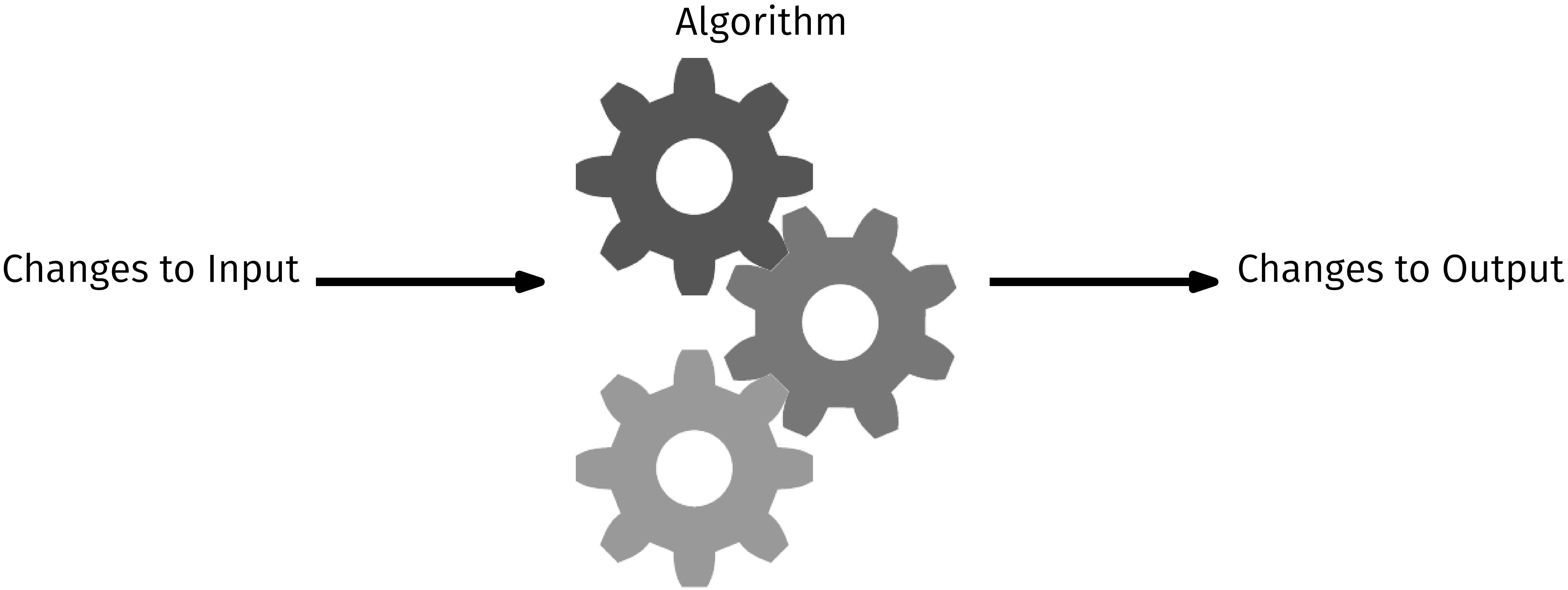
Dynamic Algorithms



Dynamic Algorithms



Dynamic Algorithms



Updates

- Insertions (*incremental*) or deletions (*decremental*) of edges/vertices, or both (*fully dynamic*)
- Fast worst-case or amortized update time

Other Models of Evolving Data

Streaming Algorithms

- Focus on space usage
- Standard model: insertions
- Impossibility results for single-pass streaming

Other Models of Evolving Data

Streaming Algorithms

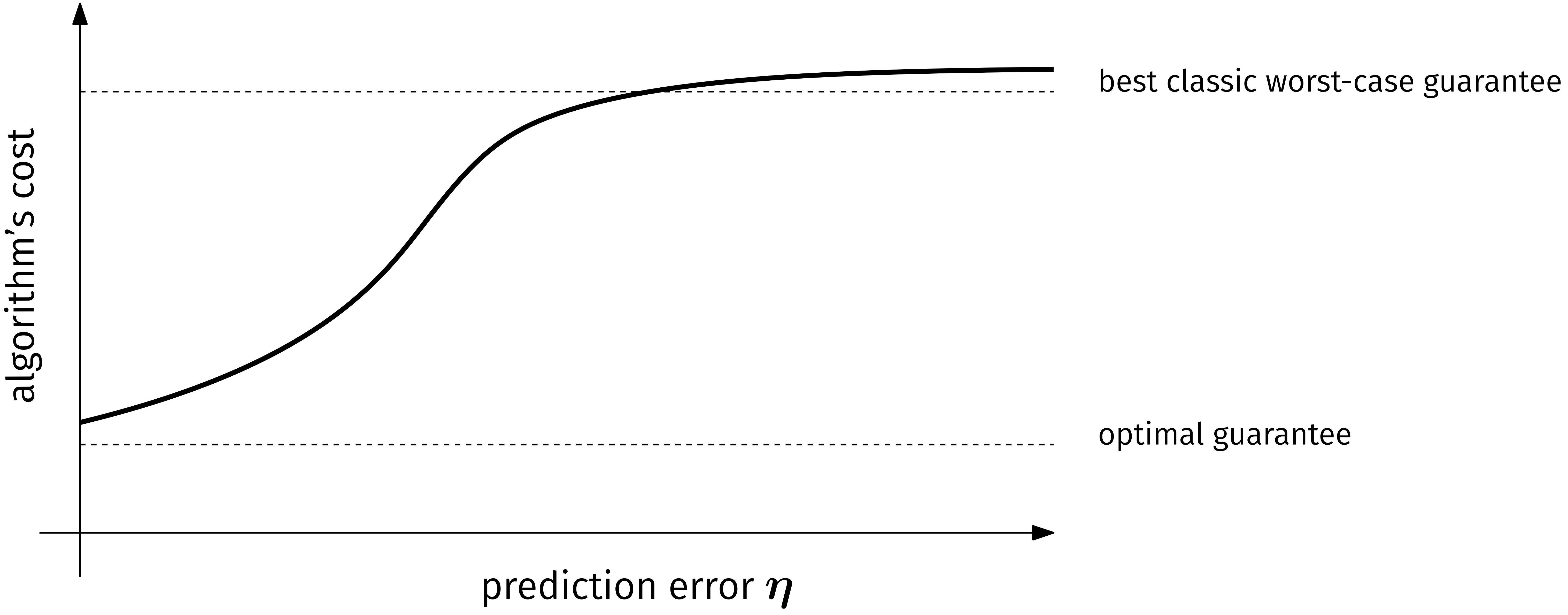
- Focus on space usage
- Standard model: insertions
- Impossibility results for single-pass streaming

Online Algorithms

- Focus on competitive ratio of output
- Standard model: irrevocable decisions
- Online algorithms with recourse

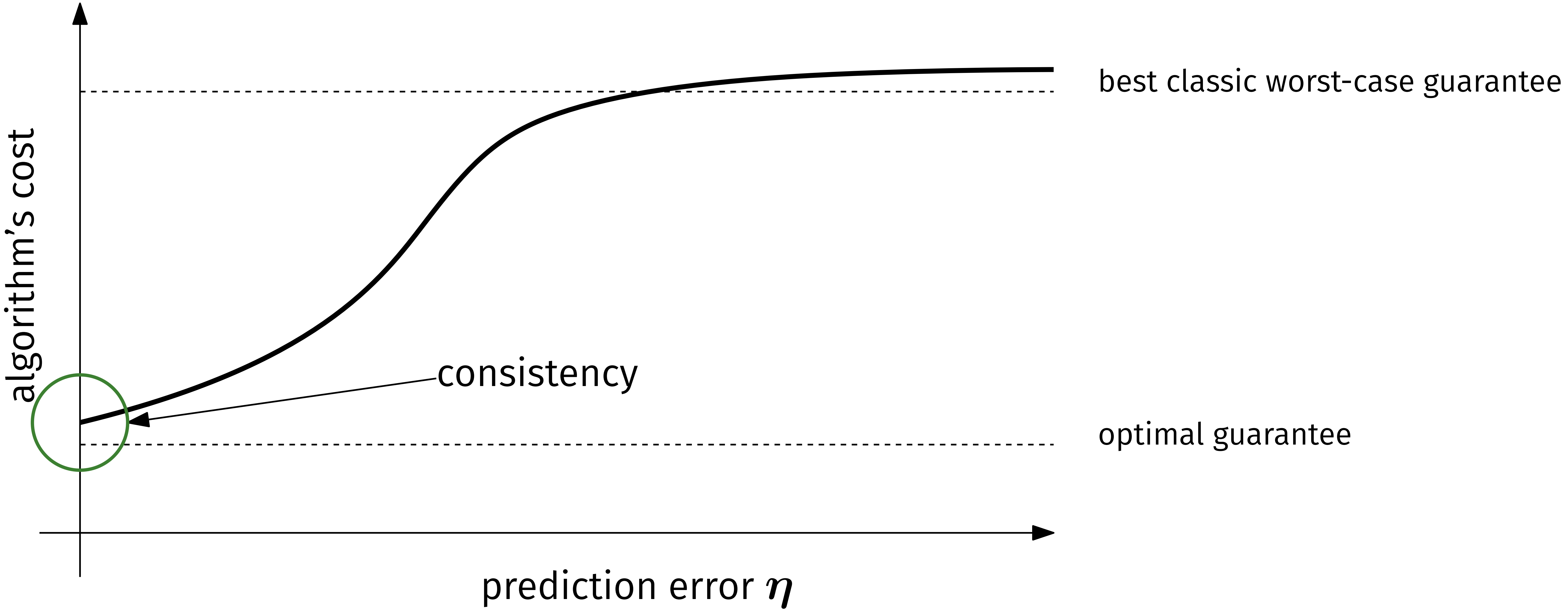
Learning-Augmented Algorithms

Input + black-box **predictions** (possibly inaccurate, with some **error** η)



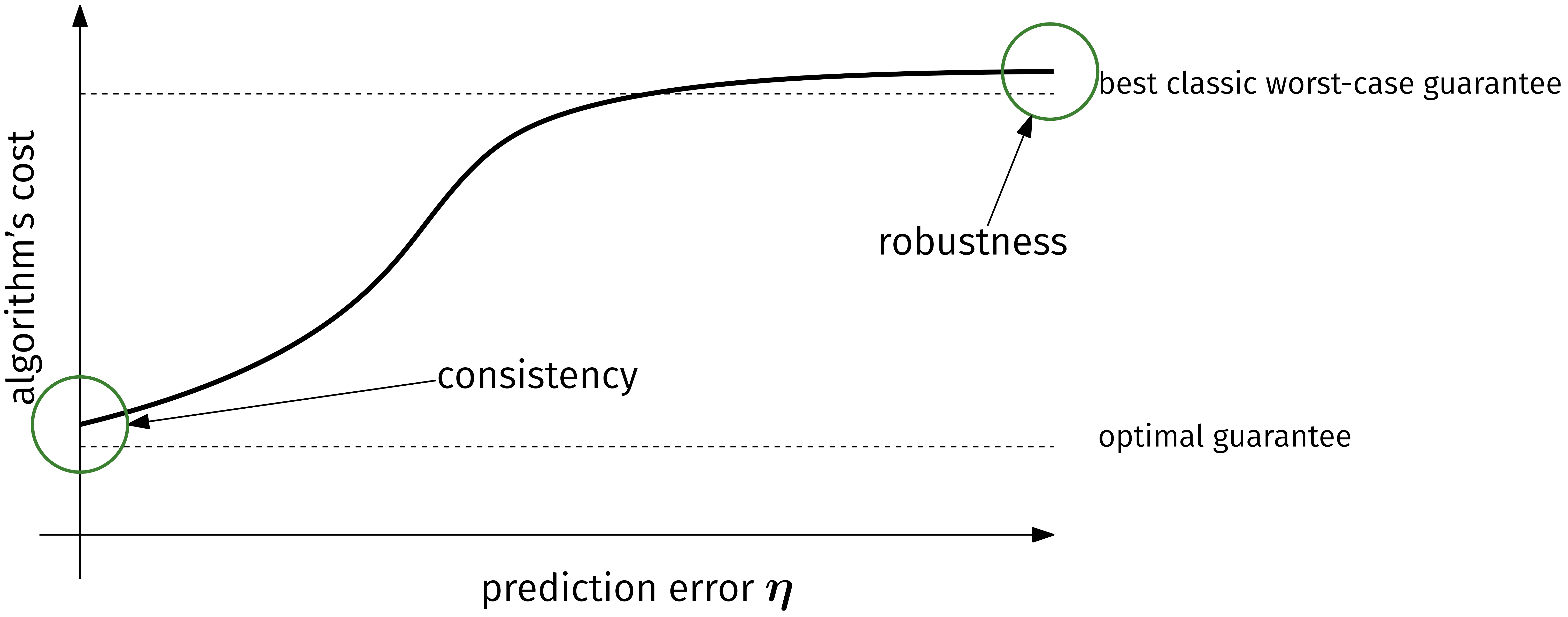
Learning-Augmented Algorithms

Input + black-box **predictions** (possibly inaccurate, with some **error** $\boldsymbol{\eta}$)



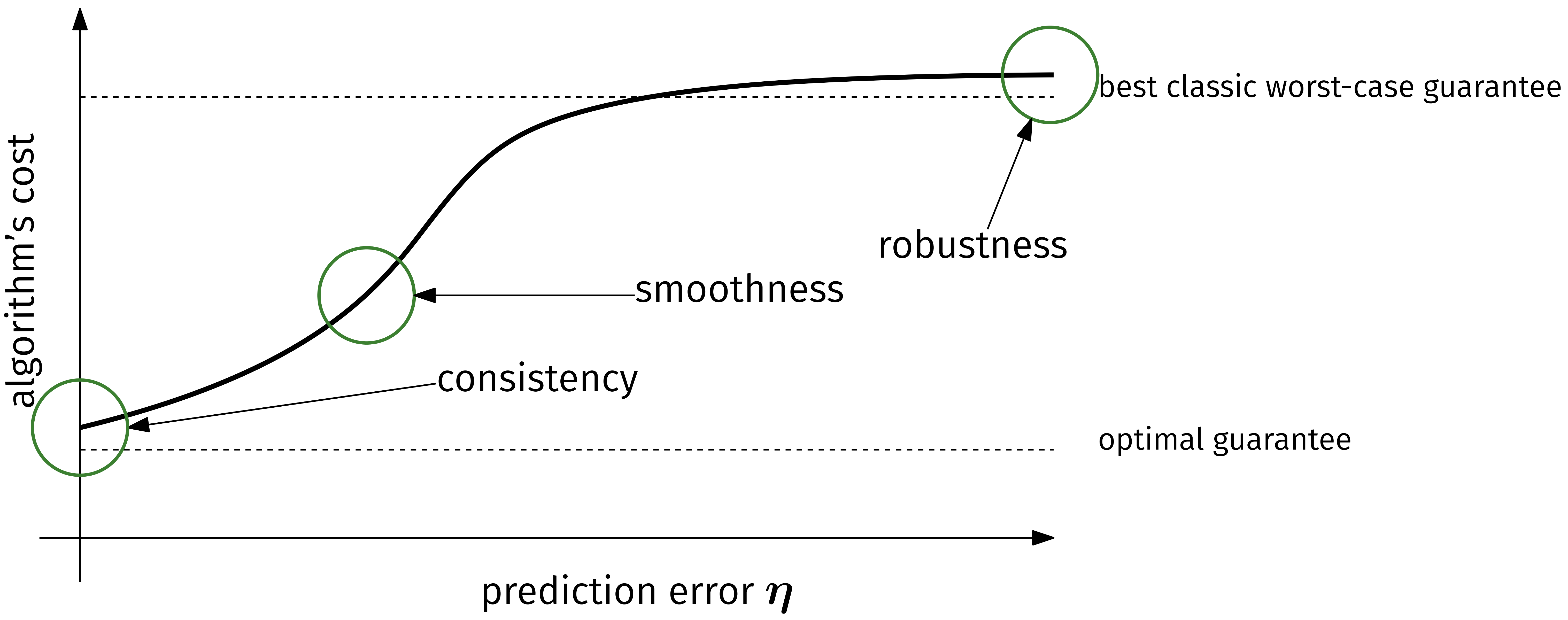
Learning-Augmented Algorithms

Input + black-box **predictions** (possibly inaccurate, with some **error** η)



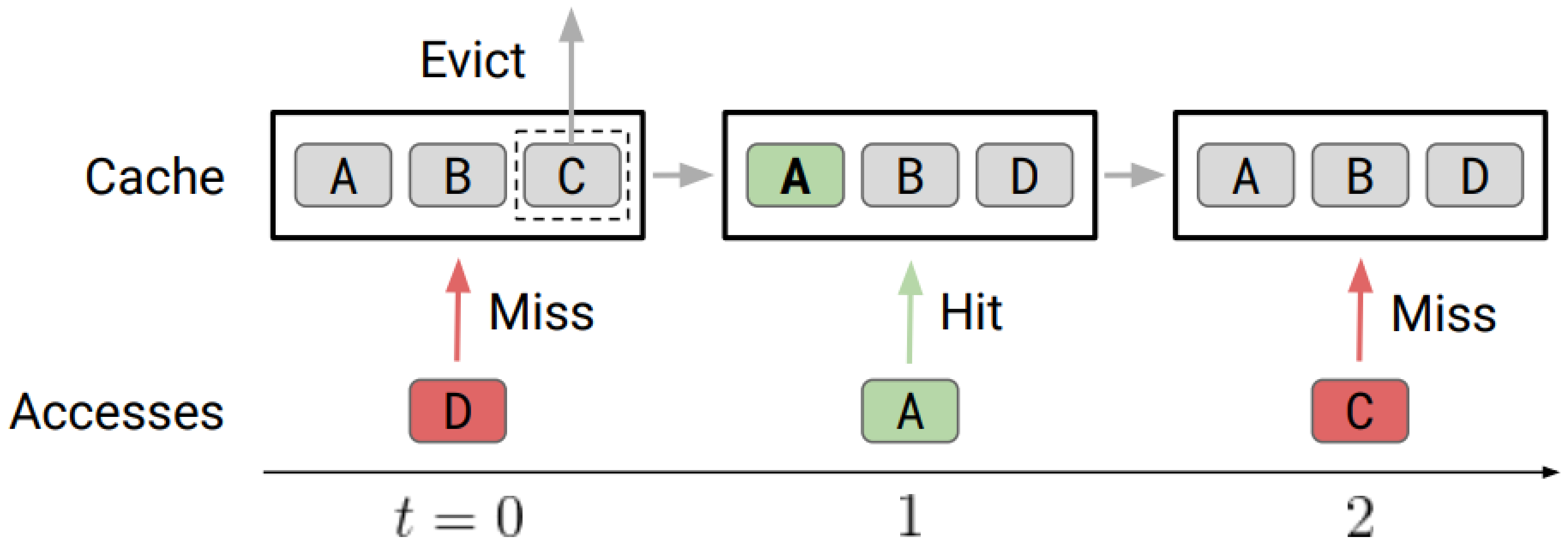
Learning-Augmented Algorithms

Input + black-box **predictions** (possibly inaccurate, with some **error** η)



Predictions can improve **competitive ratio** of **online** algorithms

First applications:
Ski-rental and ad allocation, [Mahdian, Nazerzadeh, Saberi ’07]
Caching [Lykouris, Vassilvitskii, ICML’18]



source: arxiv.org/abs/2006.16239

Possible Predictions:
Number of days skiings,
Frequencies of keywords used,
When will the currently requested item be requested again?

How can predictions improve **dynamic** algorithms?

- Predict **future** updates
- Can we improve the **update time**?

How can predictions improve **dynamic** algorithms?

- Predict **future** updates
- Can we improve the **update time**?

Perfect and **full** knowledge about future = **offline** dynamic algorithms

How can predictions improve **dynamic** algorithms?

- Predict **future** updates
- Can we improve the **update time**?

Perfect and **full** knowledge about future = **offline** dynamic algorithms

Ideal Result:
- With predictions we can do what is (provably) impossible without them
- (Provable) Gap between offline and online problem

How can predictions improve **dynamic** algorithms?

- Predict **future** updates
- Can we improve the **update time**?

Perfect and **full** knowledge about future = **offline** dynamic algorithms

Ideal Result:

- With predictions we can do what is (provably) impossible without them
- (Provable) Gap between offline and online problem

What is provably impossible? $\rightarrow$ Conditional lower bounds

Discussion of Conditional Lower Bounds

[Abboud, Vassilevska Williams ’14]
Influential paper presenting conditional lower bounds for dynamic problems based on static hardness assumptions:

- Strong Exponential Time Hypothesis
- No truly subquadratic 3SUM
- No truly subcubic all-pairs shortest paths
- No almost linear time triangle detection
- No truly subcubic combinatorial Boolean matrix multiplication

Discussion of Conditional Lower Bounds

[Abboud, Vassilevska Williams ’14]
Influential paper presenting conditional lower bounds for dynamic problems based on static hardness assumptions:

- Strong Exponential Time Hypothesis
- No truly subquadratic 3SUM
- No truly subcubic all-pairs shortest paths
- No almost linear time triangle detection
- No truly subcubic combinatorial Boolean matrix multiplication

“Problem”: These reductions also apply to the *offline* dynamic model

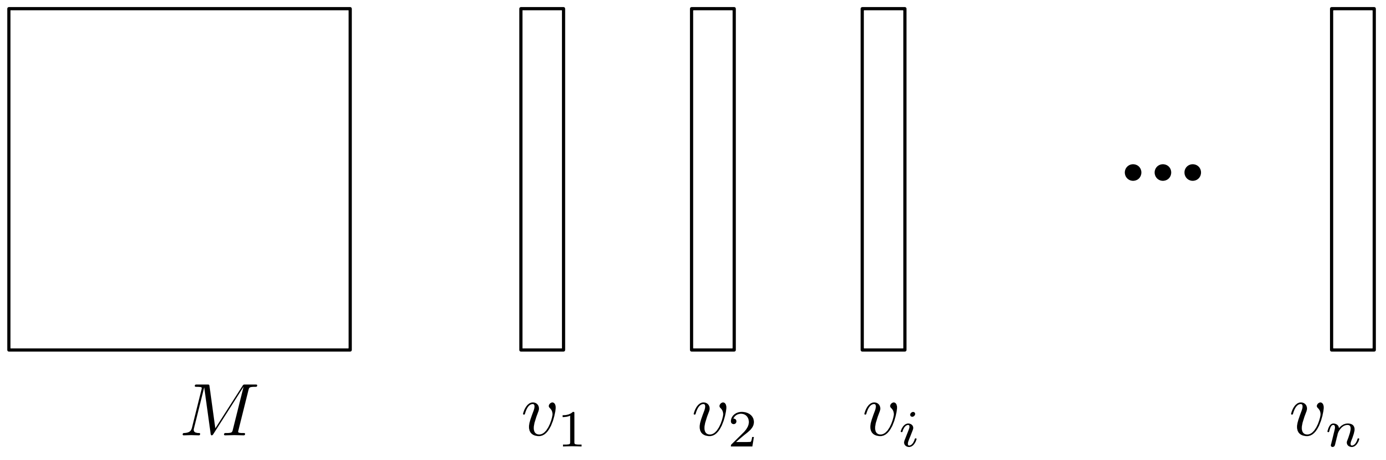
Online Matrix-Vector Multiplication Hypothesis [Henzinger, K, Nanongkai, Saranurak '15]

Input:

- M — Boolean $n \times n$ matrix , given **offline**
- $v_1, \ldots, v_n$ — Boolean $n \times 1$ vectors, given one by one **online**

Output: $Mv_1, Mv_2, \ldots, Mv_n$

Hypothesis: requires $\mathbf{n^{3-o(1)}}$ time



Does **not hold offline** — compute $M \cdot [v_1, \ldots, v_n]$ in $O(n^\omega)$ time

Online Matrix-Vector Multiplication **with Predictions** [this work]

Input:

- M — Boolean $n \times n$ matrix , given **offline**
- $\hat{v}_1, \ldots, \hat{v}_n$ — **predicted** vectors, given **offline**
- $v_1, \ldots, v_n$ — Boolean $n \times 1$ vectors, given one by one **online**

Output: $Mv_1, Mv_2, \ldots, Mv_n$

Online Matrix-Vector Multiplication **with Predictions** [this work]

Input:
- M — Boolean $n \times n$ matrix , given **offline**
- $\hat{v}_1, \ldots, \hat{v}_n$ — **predicted** vectors, given **offline**
- $v_1, \ldots, v_n$ — Boolean $n \times 1$ vectors, given one by one **online**

Output: $Mv_1, Mv_2, \ldots, Mv_n$

Algorithm:
- Preprocessing in $O(n^\omega)$ time

$$M \cdot [\hat{v}_1, \ldots, \hat{v}_n]$$

- Each request in $O(n\eta_i)$ time, $\eta_i = ||v_i - \hat{v}_i||_1 = ||v_i - \hat{v}_i||_0$

$$Mv_i = M(v_i - \hat{v}_i + \hat{v}_i) = M(v_i - \hat{v}_i) + M\hat{v}_i$$

$O(n\eta_i)$ $O(n)$

Total time: $O(n^\omega + n\eta)$, $\eta = \sum_{i=1}^{n} \eta_i$

Incremental All-Pairs Reachability

Update: Insert directed edge (u, v)
Query: Is there a path from a to b?

Upper bound: $O(nm)$ total update time (for n nodes and m edges) [Italiano, TCS 86]
Lower bound: n^2 updates and n^2 queries require $n^{3-o(1)}$ time, under OMv

Incremental All-Pairs Reachability

Update: Insert directed edge (u, v)
Query: Is there a path from a to b?

Upper bound: $O(nm)$ total update time (for n nodes and m edges) [Italiano, TCS 86]
Lower bound: n^2 updates and n^2 queries require $n^{3-o(1)}$ time, under OMv

Offline = all-pairs **bottleneck** paths [this work]
($O(n^{(3+\omega)/2}) \leqslant O(n^{2.687})$ **pre-processing** time, via min-max matrix product [DP09])

$$D[\boldsymbol{a}, \boldsymbol{b}] = \min_{\boldsymbol{P} \in \{\text{paths from } \boldsymbol{a} \text{ to } \boldsymbol{b}\}} \max_{\boldsymbol{e} \in \boldsymbol{P}} weight(\boldsymbol{e})$$

Edge weights = arrival time
If $D[\boldsymbol{a}, \boldsymbol{b}] \leq j \implies$ after j insertions: path from $\boldsymbol{a}$ to $\boldsymbol{b}$

Incremental All-Pairs Reachability

Update: Insert directed edge (u, v)
Query: Is there a path from a to b?

Upper bound: $O(nm)$ total update time (for n nodes and m edges) [Italiano, TCS 86]
Lower bound: n^2 updates and n^2 queries require $n^{3-o(1)}$ time, under OMv

Offline = all-pairs **bottleneck** paths [this work]
($O(n^{(3+\omega)/2}) \leqslant O(n^{2.687})$ **pre-processing** time, via min-max matrix product [DP09])

Prediction: *sequence of insertions.* How do we handle prediction errors?

Incremental All-Pairs Reachability

Update: Insert directed edge (u, v)
Query: Is there a path from a to b?

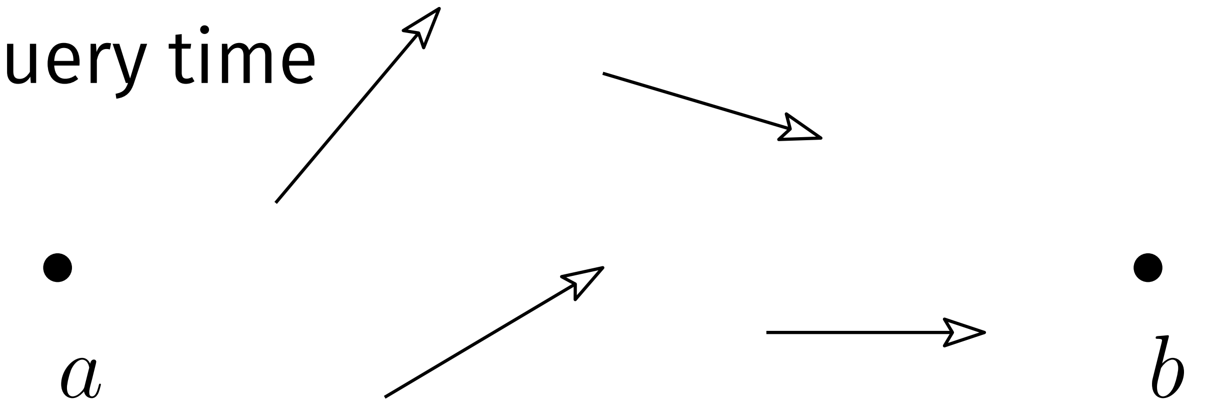
Upper bound: $O(nm)$ total update time (for n nodes and m edges) [Italiano, TCS 86]
Lower bound: n^2 updates and n^2 queries require $n^{3-o(1)}$ time, under OMv

Offline = all-pairs **bottleneck** paths [this work]
($O(n^{(3+\omega)/2}) \leqslant O(n^{2.687})$ **pre-processing** time, via min-max matrix product [DP09])

Prediction: *sequence of insertions.* How do we handle prediction errors?

k non-predicted edges "on the side" $\implies$ $O(k^2)$ query time

$k \leqslant \ell_\infty$ error of predictions



Incremental All-Pairs Reachability

Update: Insert directed edge (u, v)
Query: Is there a path from a to b?

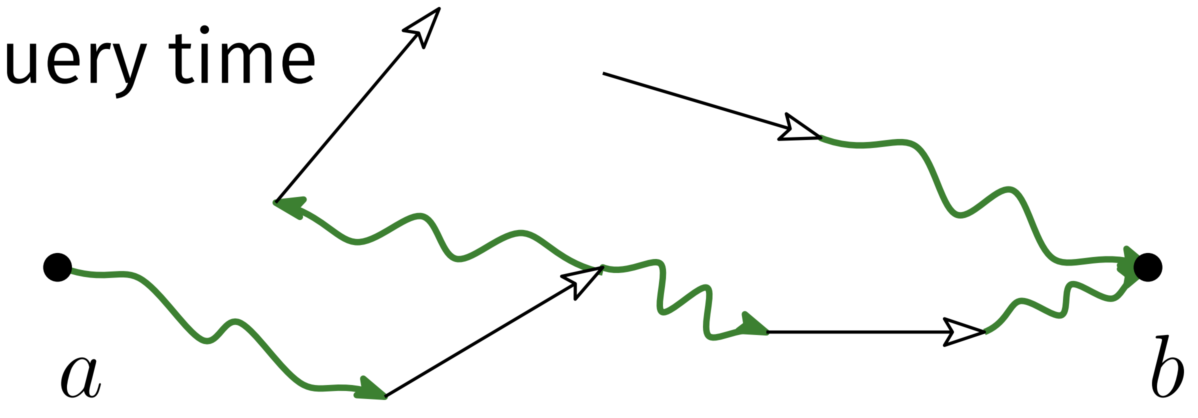
Upper bound: $O(nm)$ total update time (for n nodes and m edges) [Italiano, TCS 86]
Lower bound: n^2 updates and n^2 queries require $n^{3-o(1)}$ time, under OMv

Offline = all-pairs **bottleneck** paths [this work]
($O(n^{(3+\omega)/2}) \leqslant O(n^{2.687})$ **pre-processing** time, via min-max matrix product [DP09])

Prediction: *sequence of insertions.* How do we handle prediction errors?

k non-predicted edges "on the side" $\implies O(k^2)$ query time

$k \leqslant \ell_\infty$ error of predictions



Our results for graph problems

Partially dynamic
(edge insertions **or** deletions)

Prediction: **update sequence**

- Transitive closure
- Aproximate APSP

Preprocessing	$O(n^{2.687})$
Update	$O(1)$
Query	$O(\eta^2)$

ℓ_∞ prediction error

Our results for graph problems

Partially dynamic
(edge insertions **or** deletions)

Prediction: **update sequence**

- Transitive closure
- Aproximate APSP

Preprocessing	$O(n^{2.687})$
Update	$O(1)$
Query	$O(\eta^2)$

ℓ_∞ prediction error

Fully dynamic
(edge insertions **and** deletions)

Prediction: **update sequence**

- Triangle detection
- Exact matching
- Single-source reachability
- many more...

Preprocessing	$O(n^{2.373})$
Update	$O(n^{1.373} + n\eta_i)$
Query	$O(n^{1.373} + n\eta_i)$

ℓ_1 error per operation

Our results for graph problems

Partially dynamic
(edge insertions **or** deletions)

Prediction: **update sequence**

- Transitive closure
- Aproximate APSP

Preprocessing	$O(n^{2.687})$
Update	$O(1)$
Query	$O(\eta^2)$

ℓ_∞ prediction error

Fully dynamic
(edge insertions **and** deletions)

Prediction: **update sequence**

- Triangle detection
- Exact matching
- Single-source reachability
- many more…

Preprocessing	$O(n^{2.373})$
Update	$O(n^{1.373} + n\eta_i)$
Query	$O(n^{1.373} + n\eta_i)$

ℓ_1 error per operation

Fully dynamic
(edge insertions **and** deletions)

Prediction: **deletion** times
(given during insertions)

- All-pairs shortest paths (Exact APSP)
- Uses a general reduction

Preprocessing	—
Update	$\tilde{O}(n^2)$
Query	$\tilde{O}(n^2\eta_i)$

ℓ_1 error per operation

Our results for graph problems

see also [Liu, Srinivas '24]

Partially dynamic
(edge insertions **or** deletions)

Prediction: **update sequence**

- Transitive closure
- Aproximate APSP

Preprocessing	$O(n^{2.687})$
Update	$O(1)$
Query	$O(\eta^2)$

ℓ_∞ prediction error

see also [Henzinger, Saha, Seybold, Ye, '24]

Fully dynamic
(edge insertions **and** deletions)

Prediction: **update sequence**

- Triangle detection
- Exact matching
- Single-source reachability
- many more...

Preprocessing	$O(n^{2.373})$
Update	$O(n^{1.373} + n\eta_i)$
Query	$O(n^{1.373} + n\eta_i)$

ℓ_1 error per operation

Fully dynamic
(edge insertions **and** deletions)

Prediction: **deletion** times
(given during insertions)

- All-pairs shortest paths (Exact APSP)
- Uses a general reduction

Preprocessing	—
Update	$\tilde{O}(n^2)$
Query	$\tilde{O}(n^2\eta_i)$

ℓ_1 error per operation

Discussion and Future Directions

- More problems with faster offline algorithms?
  (see [McCauley et al. ICALP '25], [Górkiewicz, Karczmarz ICALP '25])

- Insisting on offline-online gap might be too restrictive
  More fine-grained or practical models for algorithms with predictions?

Discussion and Future Directions

- More problems with faster offline algorithms?
  (see [McCauley et al. ICALP ’25], [Górkiewicz, Karczmarz ICALP ’25])

- Insisting on offline-online gap might be too restrictive
  More fine-grained or practical models for algorithms with predictions?

Thank you!