max planck institut informatik

Complexity Theory of Polynomial-Time Problems

Lecture 10: Dynamic Algorithms II

Sebastian Krinninger
(with slides by Thatchaphol Saranurak)

Exam

- Oral exam
- Tentative date:
  - September 5-9
  - (In lecture, students preferred this over end-of-July date)

Limits of Dynamic Algorithms?

Even-Shiloach: Incremental/decremental SSSP with total time $O(mn)$ and constant query time

Amortized $O(n)$ per update

The success story of dynamic algorithms:

Connectivity: In an undirected graph, maintain fully dynamic data structure that answers *connectivity* queries (is u connected to v) for any pair of nodes.

After a long line of research:

Theorem: There is a randomized fully dynamic connectivity algorithm with worst-case update time $O(\log^5 n)$ and query time $O(\log n)$

[Kapron et al. '13]

A Simple Problem?

What about directed graphs?

SSR: Maintain which nodes can be reached from a source node s in a directed graph

#SSR: Maintain number of reachable nodes from a source node s in a directed graph

Upper bounds with constant query time and total update time

- $O(m)$ incremental
- $O(m)$ decremental in directed acyclic graphs
- $O(m\sqrt{n \log n})$ decremental in genral graphs [Chechik et al. ‘16]

What about fully dynamic algorithms?

What about worst-case bounds?

Today's Theorems

Theorem: There is no incremental algorithm for #SSR with **worst-case** update and query time $O(n^{1-\epsilon})$ unless OVH fails.

[Abboud/V. Williams '14]

Theorem: There is no fully dynamic algorithm for #SSR with **amortized** update and query time $O(n^{1-\epsilon})$ unless OVH fails.

[Abboud/V. Williams '14]

Theorem: There is no incremental algorithm for SSR with **worst-case** update time $O(n^{1-\epsilon})$ and query time $O(n^{2-\epsilon})$ unless the OMv Conjecture fails.

[Henzinger et al. '15]

Theorem: There is no fully dynamic algorithm for SSR with **amortized** update $O(n^{1-\epsilon})$ and query time $O(n^{2-\epsilon})$ unless the OMv Conjecture fails.

[Henzinger et al. '15]

Today’s Plan: Conditional Lower Bounds

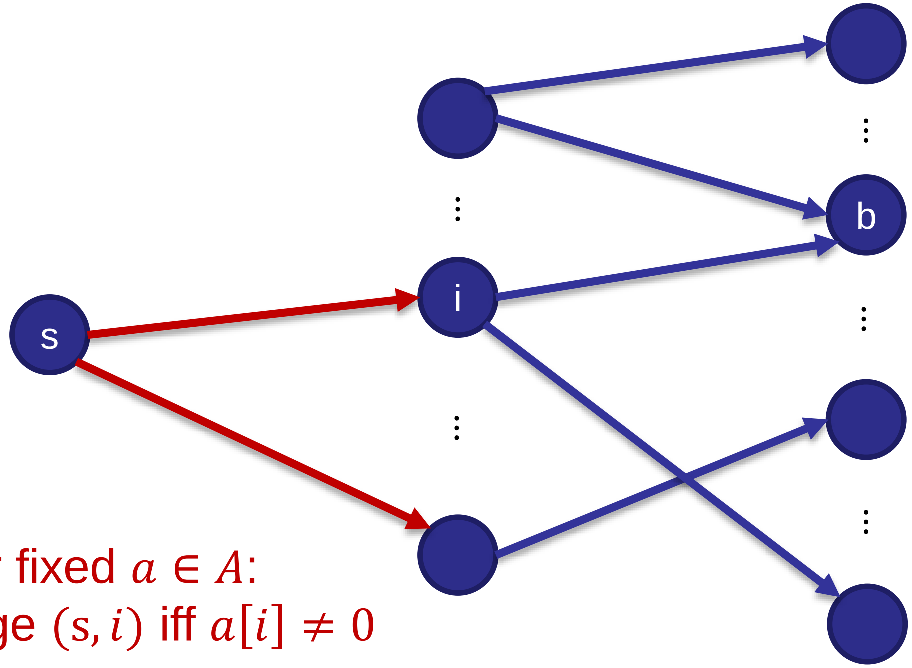
1. Lower bound for #SSR based on OV
2. OMv Conjecture and equivalence to OuMv
3. Lower bounds for SSR based on OuMv

1. Lower bound for #SSR based on OV

Reduction from OV

Given: Sets of d-dimensional vectors A and B of size $|A| = |B| = n$
Question: Are there $a \in A$ and $b \in B$ such that a and b are orthogonal?

Initialization: d dimensions n vectors of B



At initialization, for every i and $b \in B$, insert:
edge (i, b) iff $b[i] \neq 0$

For fixed $a \in A$:
edge (s, i) iff $a[i] \neq 0$

Correctness

Claim: Let k be the number of 1-entries of $a \in A$.
After inserting all edges (a, i) such that $a[i] \neq 0$:
no $b \in B$ orthogonal to a *if and only if*
s can reach $n + k + 1$ nodes

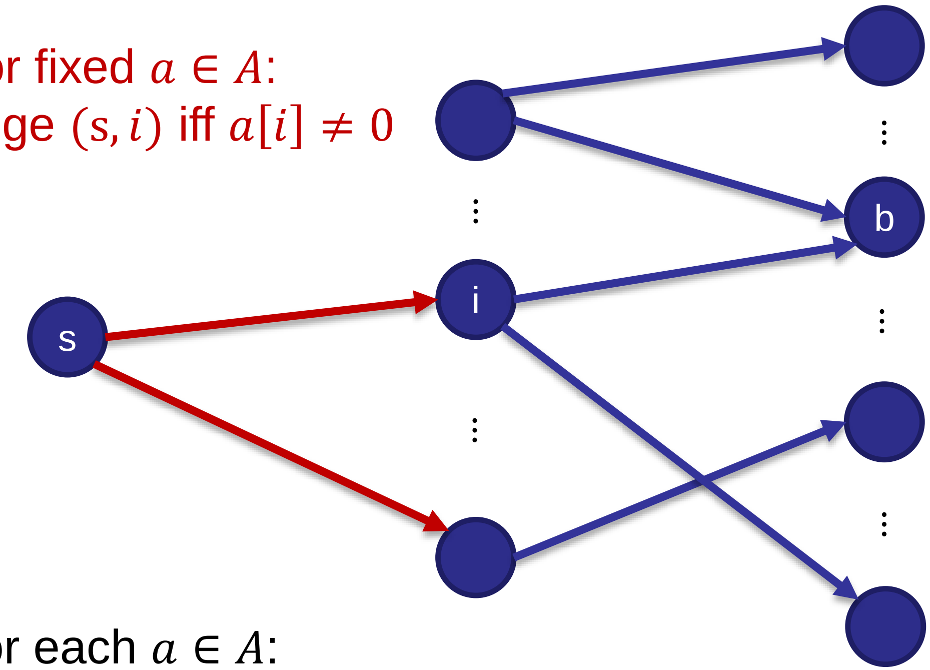
"⇒" Assume no $b \in B$ orthogonal to a
Then for every $b \in B$ there is an i such that $a[i] \neq 0$ and $b[i] \neq 0$
Thus, for every node b on right side there is some path $s \rightarrow i \rightarrow b$
⇒ s can reach $n + k + 1$ nodes (including itself)

"⇐" Assume s can reach $n + k + 1$ nodes
Then s reaches *all* nodes b on the right side
(because middle nodes only reachable if $a[i] \neq 0$)
Thus, for every b on right side there is a path from s to b
Must have the form $s \rightarrow i \rightarrow b$ for some middle node i
Then: $a[i] \neq 0$ and $b[i] \neq 0$ and a and b are *not orthogonal*
⇒ No $b \in B$ orthogonal to a

Dynamic Algorithm

Given: Sets of d-dimensional vectors A and B of size $|A| = |B| = n$
Question: Are there $a \in \mathrm{A}$ and $b \in B$ such that a and b are orthogonal?

For fixed $a \in A$:
edge (s, i) iff $a[i] \neq 0$



edge (i, b) iff $b[i] \neq 0$

For each $a \in A$:
- For every $a[i] \neq 0$: insert edge (s, i)
- Query number of nodes reachable from s
- If #nodes reachable from $s < n + k + 1$: output ‘yes’
- Delete all edges leaving s

Output ‘no’

Running Time

Assumption: There is a fully dynamic algorithm for #SSR with amortized update time $n^{1-\epsilon}$ and query time $n^{1-\epsilon}$

For each $a \in A$:

For every $a_i \neq 0$: insert edge (s, i)
Query number of nodes reachable from s
If #nodes reachable from $s < n + k + 1$: output 'yes'
Delete all edges leaving s

Output 'no'

#nodes: $n + d + 1 = O(n + d)$

#insertions: $\leq nd$

#deletions: $\leq nd$

#queries: $\leq n$

Total time: $O(nd \cdot (n + d)^{1-\epsilon} + n \cdot (n + d)^{1-\epsilon}) = n^{2-\epsilon} \cdot \text{poly}(d)$

Contradicts OV Hypothesis!

Worst-Case Lower Bound

Assumption: There is an incremental algorithm for #SSR with worst-case update time $n^{1-\epsilon}$ and query time $n^{1-\epsilon}$

Fully dynamic

For every i and $b \in B$, insert:

For each $a \in A$:

For every $a_i \neq 0$: insert edge (s, i)

Query number of nodes reachable from s

If #nodes reachable from $s < n + k + 1$: output 'yes'

Delete all edges leaving s

Output 'no'

Incremental

Instead of deleting edges leaving s:

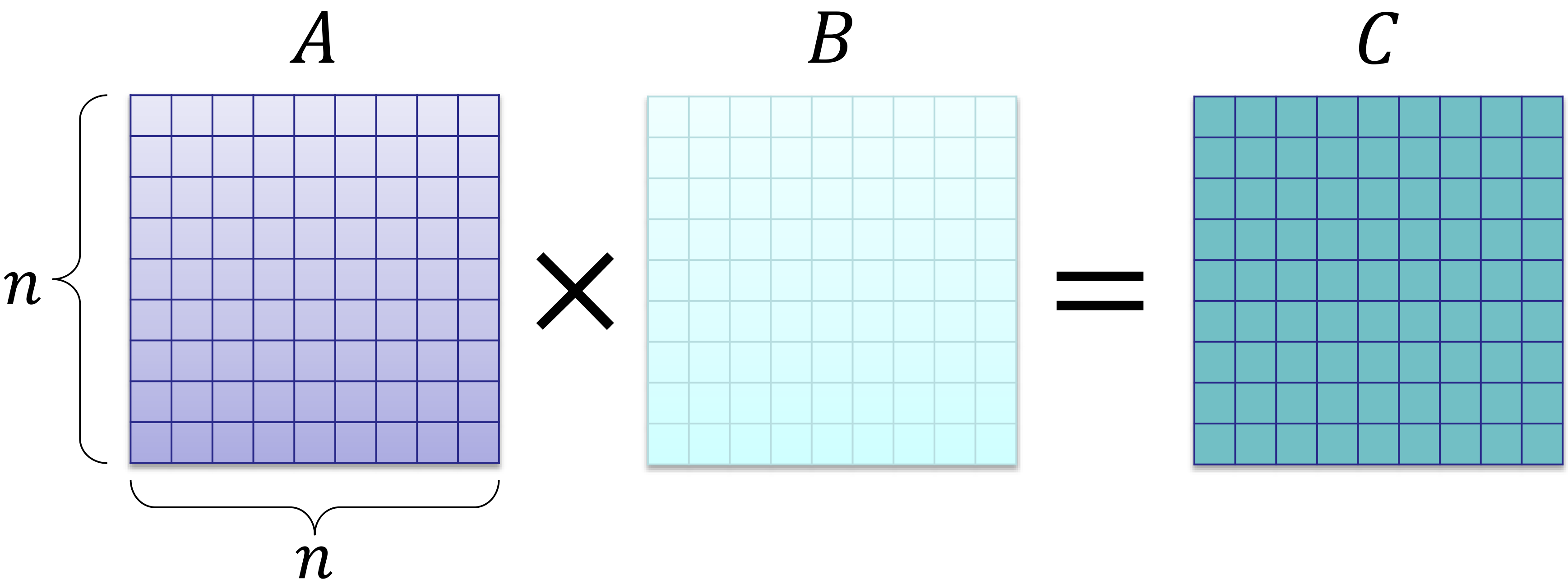
1. Observe complete state of the machine after initialization and before inserting first edges (s, i)
2. Record changes to the state while processing insertions: $O(d(n+d)^{1-\epsilon})$
   (changes to memory cells, etc.)
3. Undo changes and roll back to state before insertions
   Takes same amount of time as processing insertions: $O(d(n+d)^{1-\epsilon})$

2. OMv Conjecture and equivalence to OuMv

Boolean Matrix Multiplication

Input: Boolean (0/1) matrices A and B

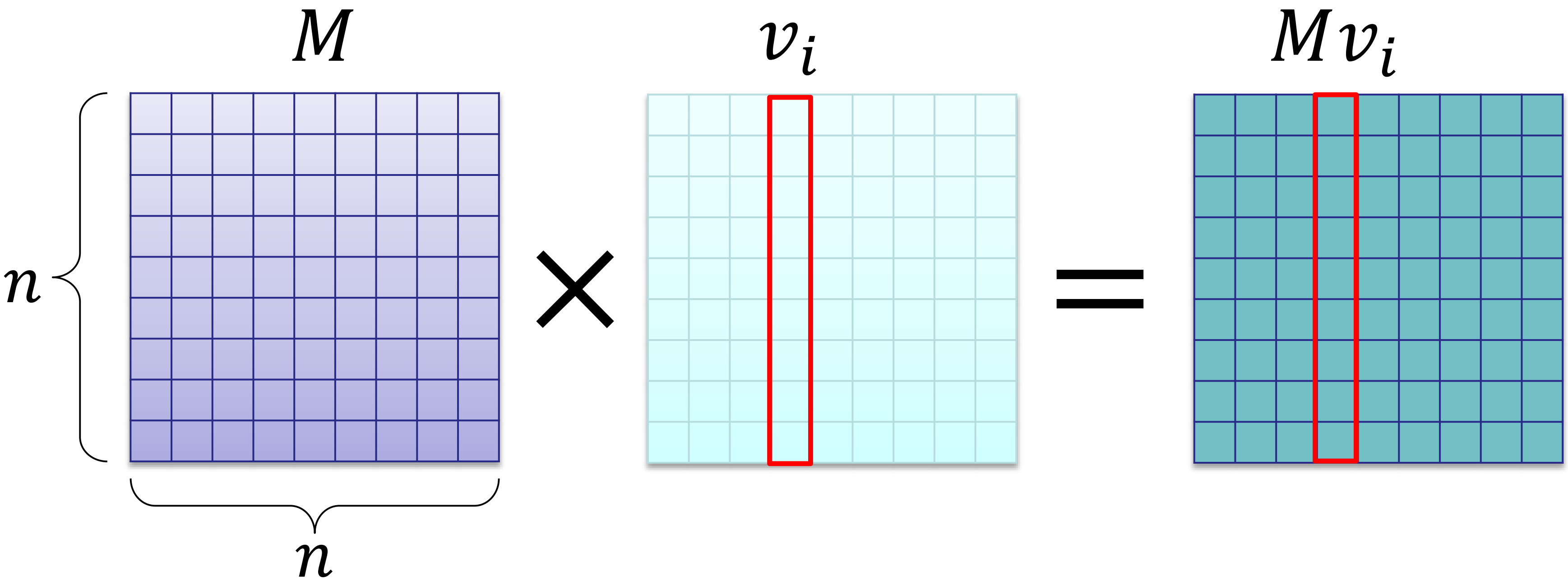
Output: $A \times B$ where + is OR and $*$ is AND



Online Boolean Matrix Multiplication

Input: Boolean $n \times n$ matrix M
Online sequence of vectors $v_1, \ldots, v_n \in \{0,1\}^n$

Output: Mv_i **before** v_{i+1} arrives *("query")*



OMv Conjecture: No algorithm with total time $O(n^{3-\epsilon})$ (for some $\epsilon > 0$).
(not even with polynomial-time preprocessing)

[Henzinger et al.'15]

Motivation

- Column-wise BMM: second matrix is given as sequence of vectors
- If OMv is refuted: radical new approach for fast BMM
  (Conceptually very different from Strassen-like approaches)
- Provides tight lower bounds for a dozen of dynamic graph problems
- Most of the reductions are almost trivial

Before obtaining useful lower bounds: hardness of intermediate problems

Hardness for Sequence of Length $\sqrt{n}$

Lemma: Cannot solve OMv for sequences of length $\sqrt{n}$ in total time $n^{2.5-\epsilon}$ (for some $\epsilon > 0$), unless original OMv Conjecture fails.

To handle OMv for sequence of length n:

Restart algorithm after each subsequence of length $\sqrt{n}$

Time: $\sqrt{n} \cdot n^{2.5-\epsilon} = n^{3-\epsilon}$ (contradicting OMv)

OuMv Problem

Input: Boolean $n \times n$ matrix M
Online sequence of pairs of vectors $(u_1, v_1), \dots, (u_n, v_n)$

Output: $u_i^T M v_i$ before (u_{i+1}, v_{i+1}) arrives

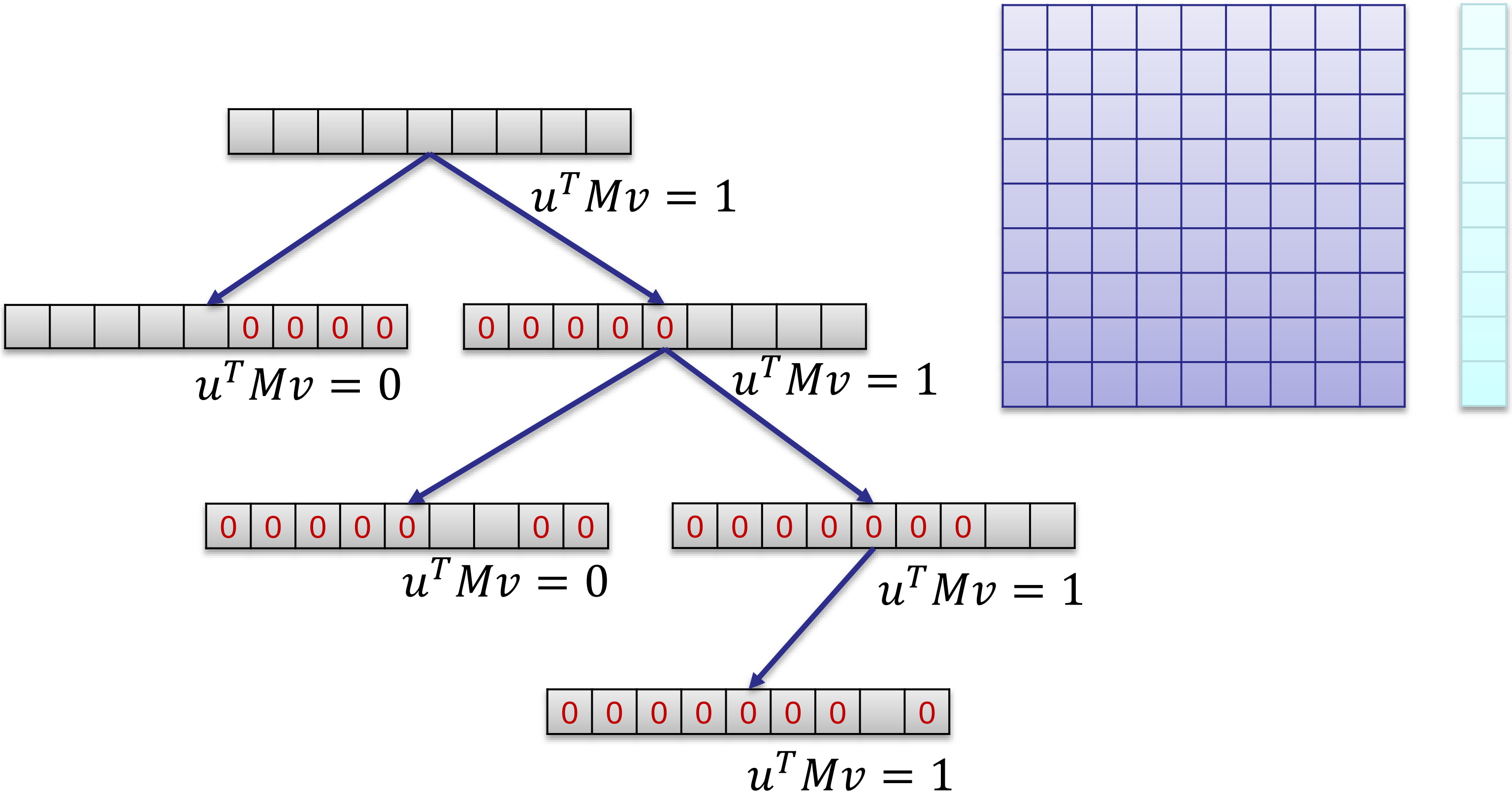
Main difference to OMv: n output bits instead of n^2

Theorem: If there is an algorithm for OuMv with running time $T(n) = n^{3-\epsilon}$, then

- there is an algorithm with running time $O(n^{2.5-\epsilon})$ for OMv of length $\sqrt{n}$ and thus
- there is an algorithm with running time $O(n^{3-\epsilon})$ for OMv

Using OuMv to Find a Single Witness

Can we find a **witness** index t s.t. $u[t] = 1$ and $(Mv)[t] = 1$?
Idea: Bisection of index set
(At least) one half must contain witness



Can isolate a single witness with $O(\log n)$ queries

Finding all Witnesses

Repeat:
- Set $u[t] = 0$ for every witness t found so far
- Find new witness

Until no witness found anymore

We spend $O(\log n)$ queries per witness

w_i: #witnesses of (u_i, v_i)

Find all witnesses of (u_i, v_i) with $O(1 + w_i \log n)$ queries

Find all witnesses of $(u_1, v_1), \dots, (u_n, v_n)$ with $O(n + \sum_{i=1}^{n} w_i \log n)$ queries

But: Need to restart after n queries

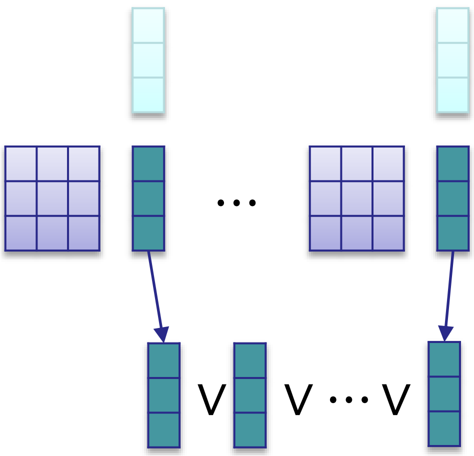
Total time: $O\left(\frac{n+\sum_{i=1}^{n} w_i \log n}{n} \cdot T(n)\right) = O\left(\left(1 + \frac{\sum_{i=1}^{n} w_i \log n}{n}\right) \cdot T(n)\right)$

$T(n)$: running time of OuMv algorithm on $n \times n$ matrix and sequence of length n

Witnesses for OR-OMv Subproblem

OR-OMv Subproblem:

- Given $n \times n$ Boolean matrices $M_1, \dots, M_k$
- Online sequence k-tuples of vectors $\in \{0,1\}^n$:
  $$(v_{1,1}, \dots, v_{1,k}), \dots, (v_{n,1}, \dots, v_{n,k})$$
- Compute $M_1 v_{i,1} \vee \cdots \vee M_k v_{i,k}$ online



Algorithm: k "witness-finding" instances of OuMv

Set $u_{i,1} = (1, \dots, 1)^T$
For $j = 1$ to k:
  $b_{i,j}$ = vector of witnesses of $u_{i,j} M_j v_{i,j}$
  $u_{i,j+1} = u_{i,j} - b_{i,j}$

Idea: No 1-entry at any position t is "lost" from result because position t set to 0 in $u_{i,j}$ only if result already contains 1 at position t

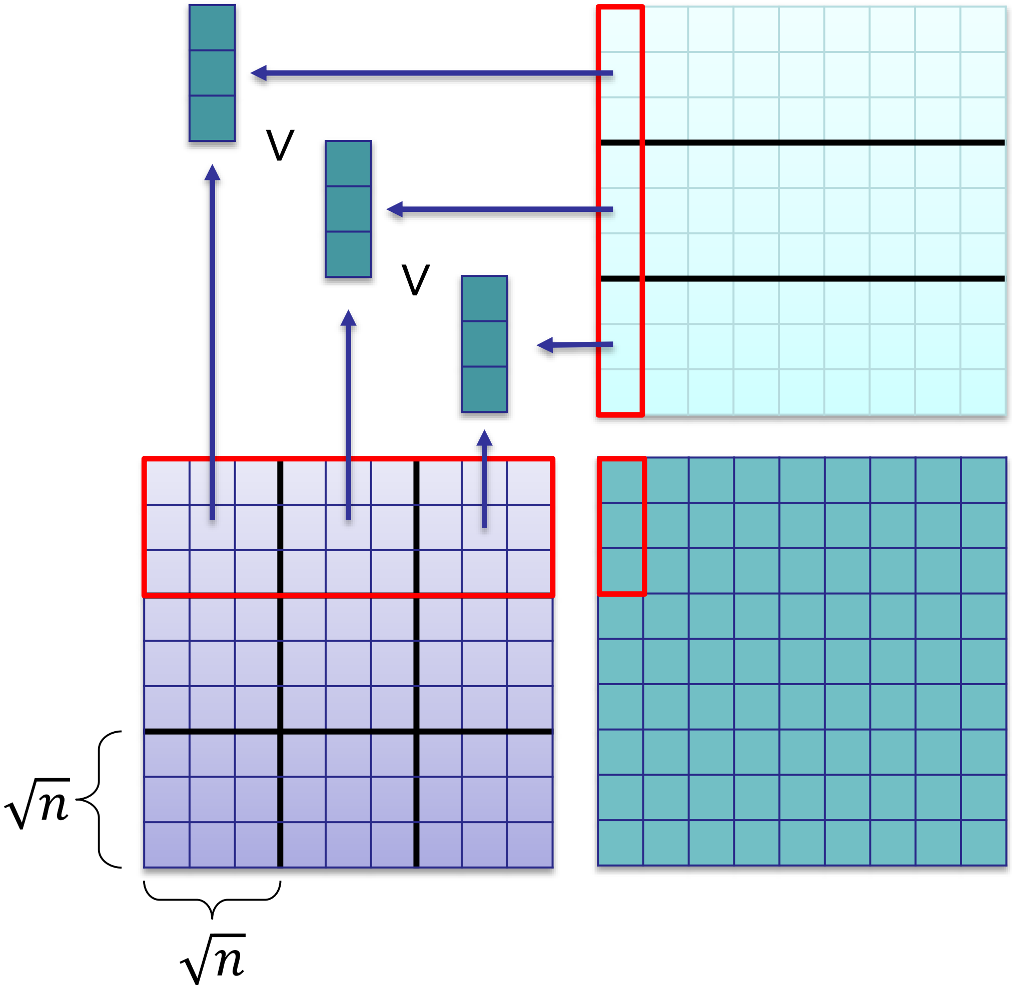
Observation: $b_{i,1} \vee \cdots \vee b_{i,k} = M_1 v_{i,1} \vee \cdots \vee M_k v_{i,k}$

W_j: #witnesses found by instance j

Running time: $O\left(\sum_{j=1}^{k}\left(1 + \frac{W_j \log n}{n}\right) \cdot T(n)\right) = O\left(\left(k + \frac{\sum_{j=1}^{k} W_j}{n}\right) \cdot T(n) \log n\right)$

$$= O\left(\left(k + \frac{n^2}{n}\right) \cdot T(n) \log n\right) = O\big((k+n) \cdot T(n) \log n\big)$$

Multiplication via Smaller Blocks



$\sqrt{n}$ instances of OR-OMv subproblem with parameters $n' = \sqrt{n}$ and $k = \sqrt{n}$

Total running time:

$$O\left(\sqrt{n} \cdot (k + n') \log n' \cdot T(n')\right)$$
$$= O\left(\sqrt{n} \cdot (\sqrt{n} + \sqrt{n}) \log n' \cdot T(n')\right)$$
$$= O\left(n \log n \cdot T(\sqrt{n})\right)$$
$$= O(n \log n \cdot n^{1.5-\epsilon})$$
$$= O\left(n^{2.5-\delta}\right)$$

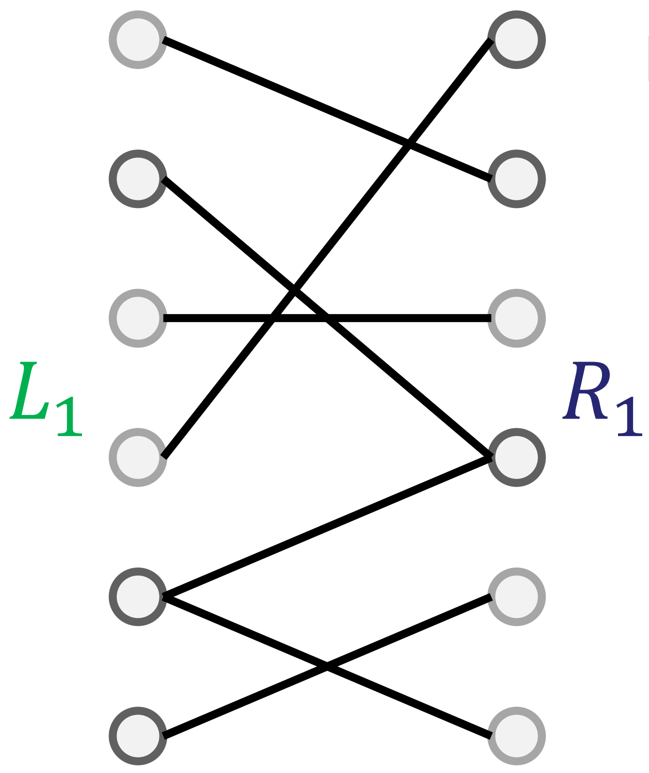
Contradicts OMv conjecture

3. OMv-Hardness of Graph Problems

Edge Query Problem

1. Preprocess

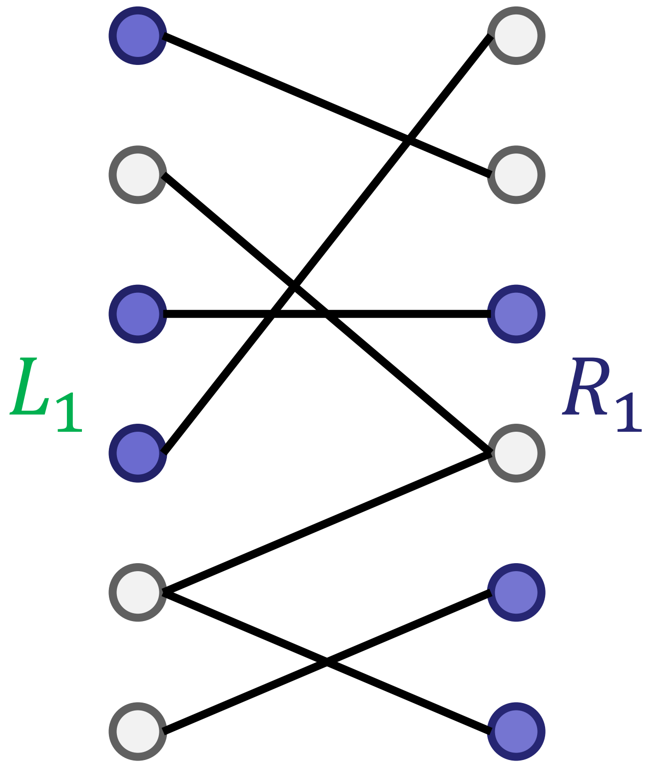
2. n Queries



Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

1. Preprocess

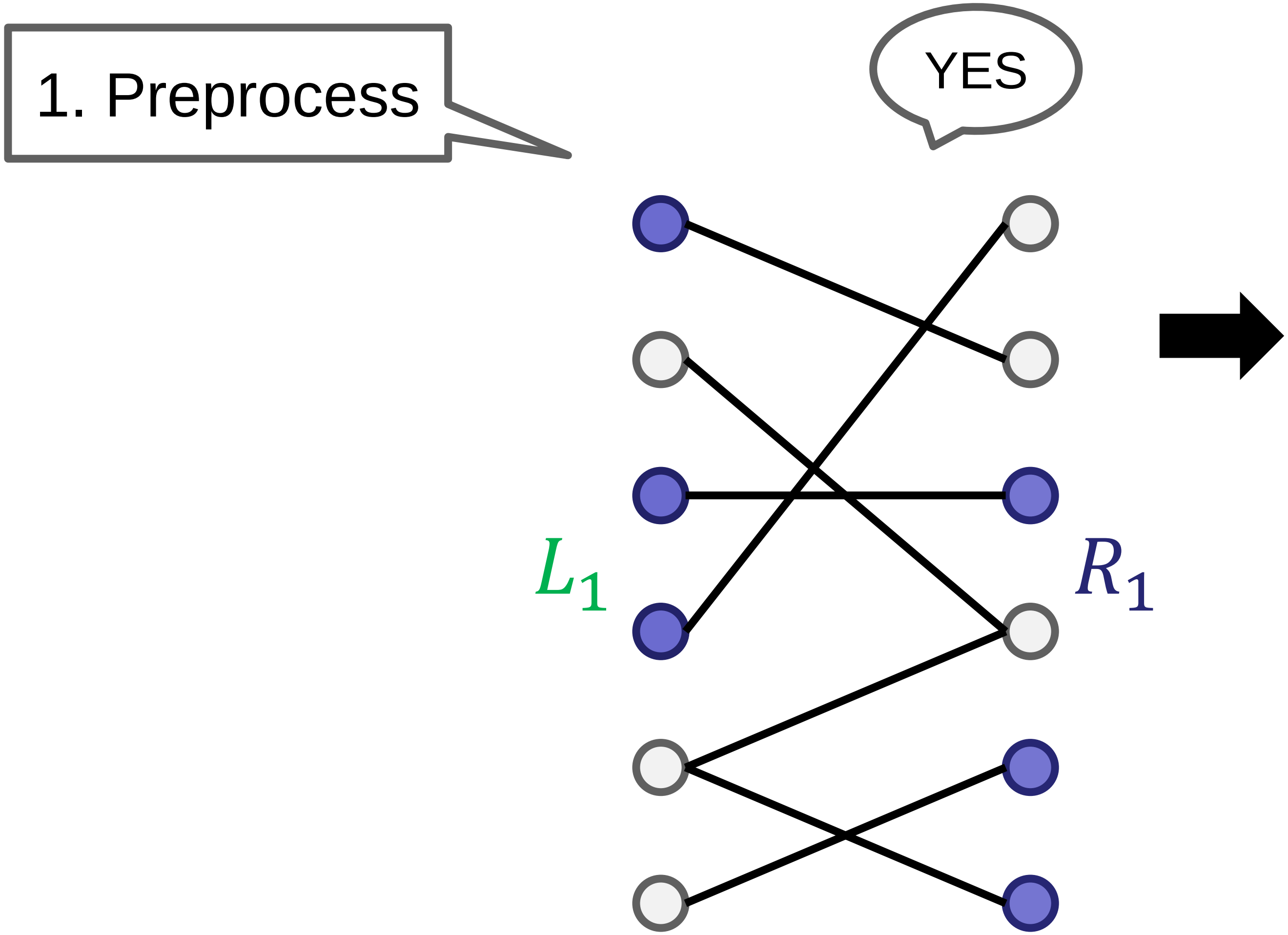


2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

max planck institut
informatik

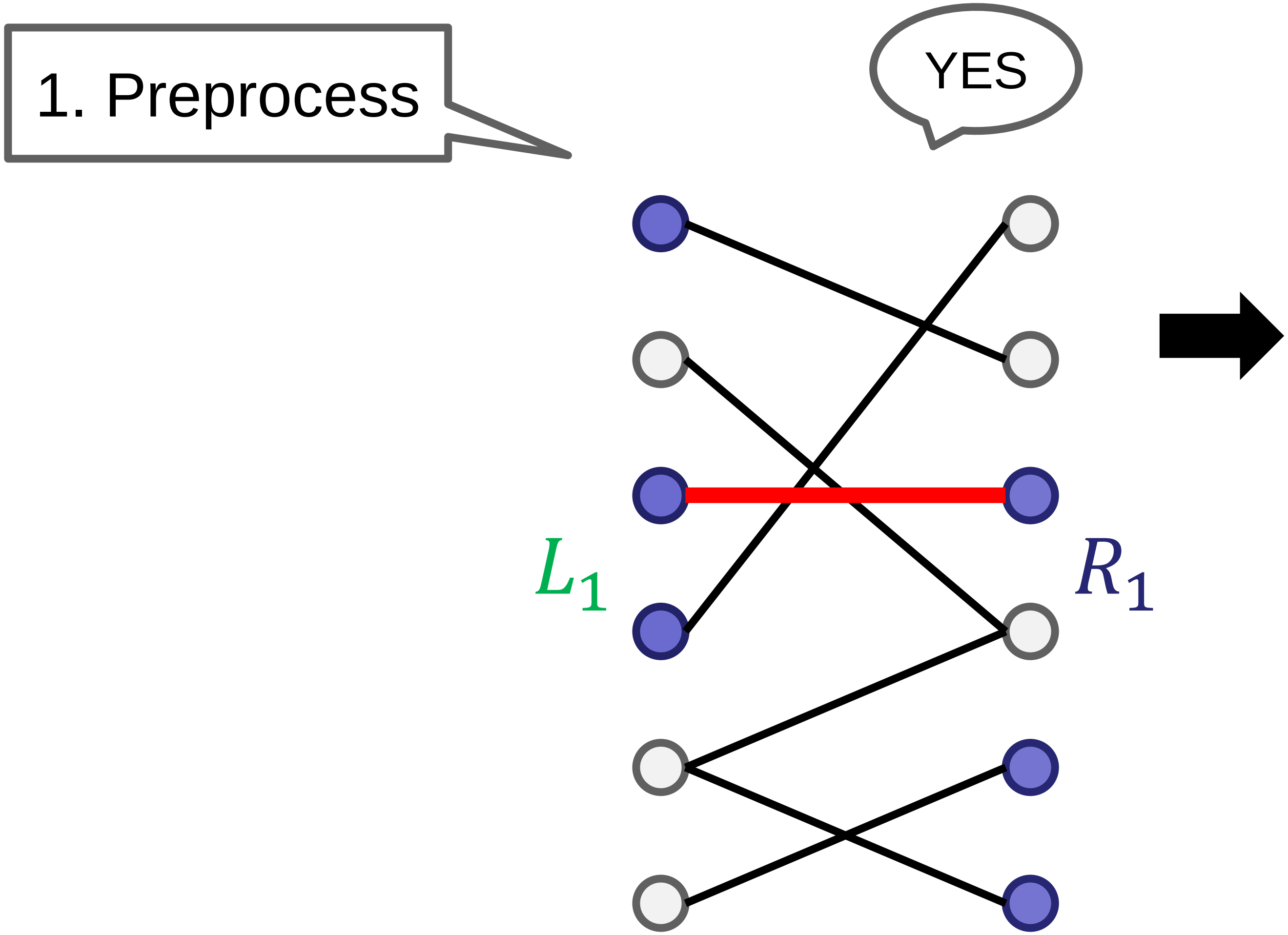
Edge Query Problem



2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

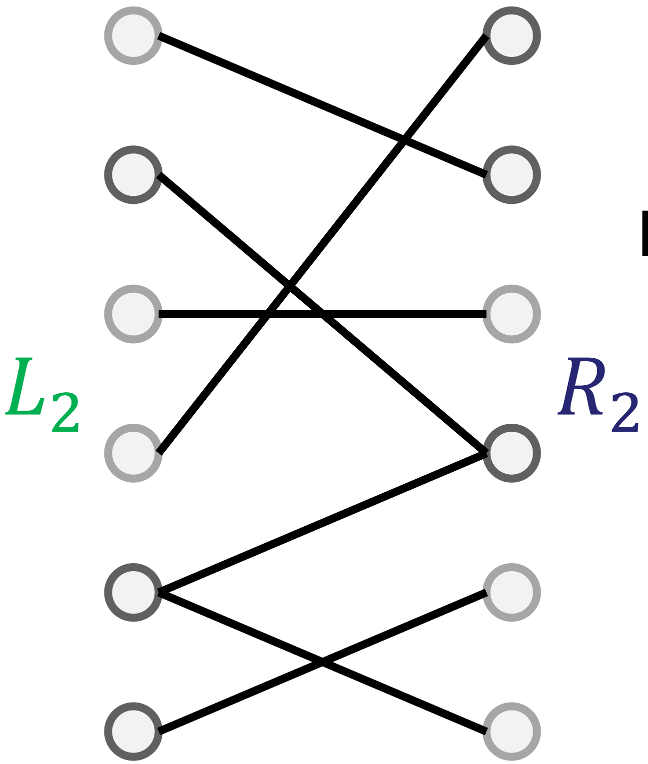


2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

1. Preprocess

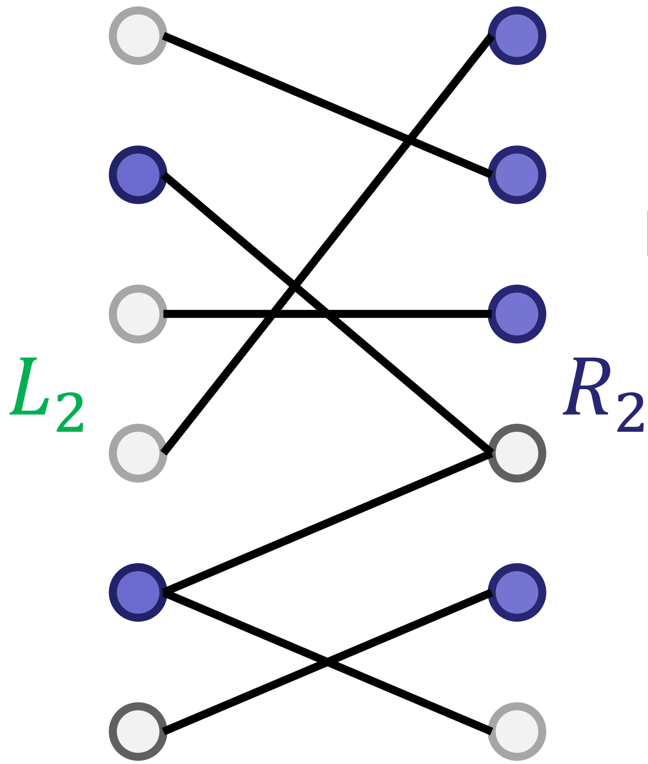


2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

1. Preprocess

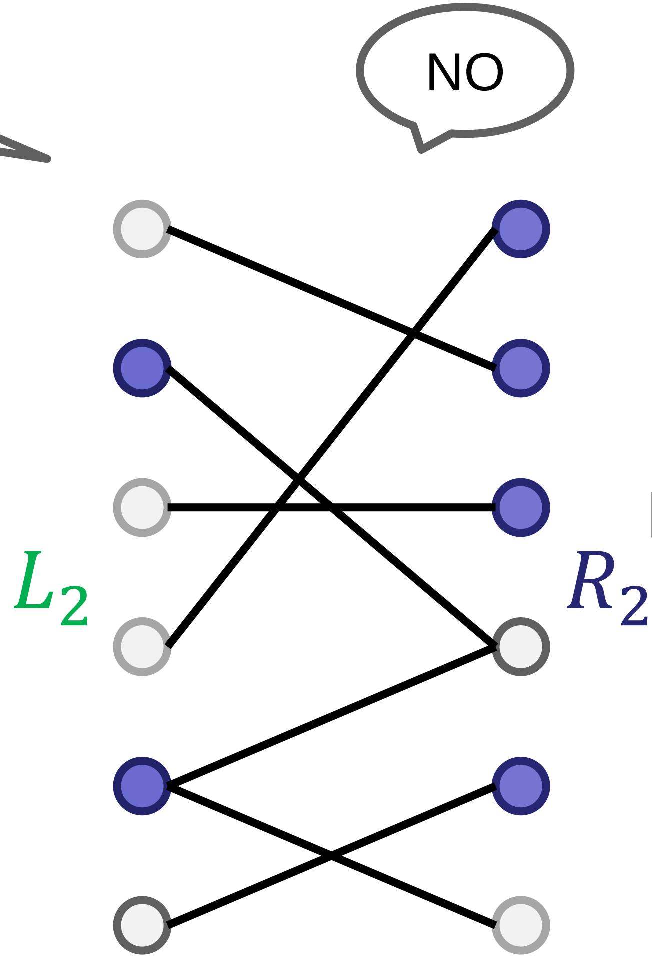


2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

1. Preprocess

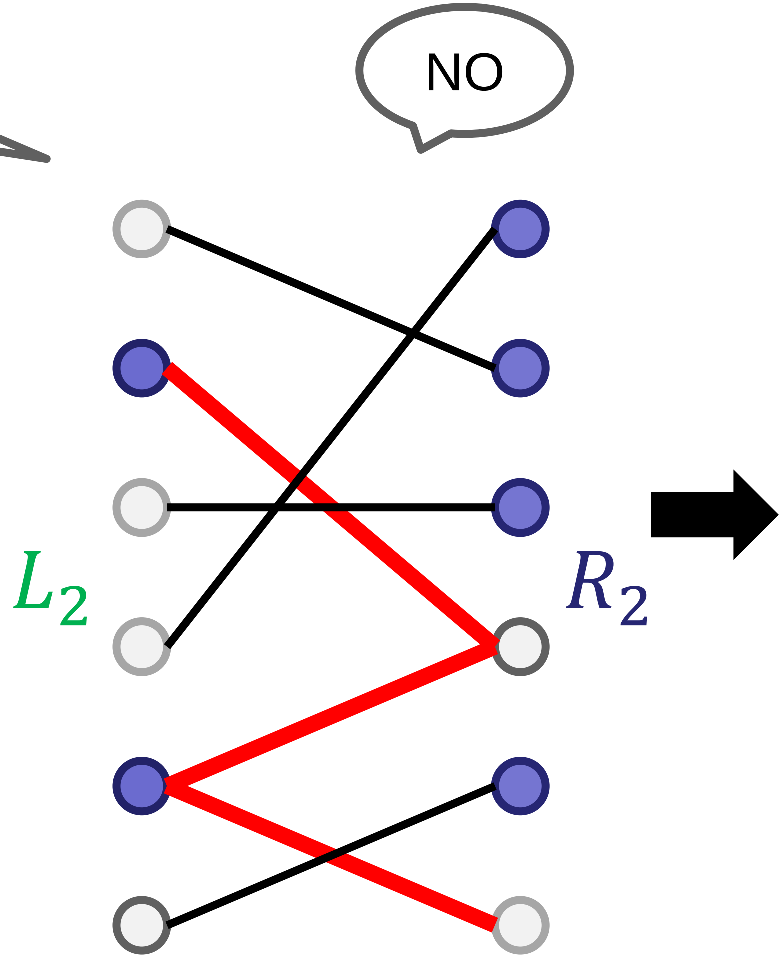


2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

1. Preprocess

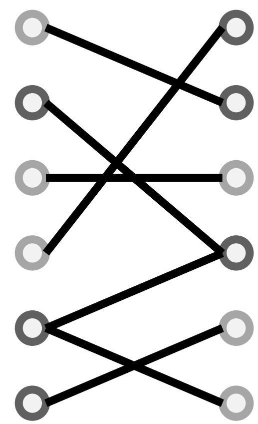


2. n Queries

Set(L_1, R_1)
Edge(L_1, R_1)?
Set(L_2, R_2)
Edge(L_2, R_2)?
…
Set(L_n, R_n)
Edge(L_n, R_n)?

Edge Query Problem

Preprocess:



$poly(n)$ time
e.g. n^{100}

Input: (L_1, R_1)
...
(L_n, R_n)

Output: Any edge linking L_1 and R_1?
…
Any edge linking L_n and R_n?

Theorem

Edge Query

- Preprocess: $poly(n)$
- Time (for n queries): $n^{3-\epsilon}$

↓

OMv conjecture fails

Reduction from OuMv to Edge-Query

Input Problem: OuMv on $n \times n$ matrix M
with online sequence $(u_1, v_1), \ldots, (u_n, v_n)$

G: graph defined by adjacency matrix $M' = \begin{bmatrix} 0 & M \\ M^T & 0 \end{bmatrix}$

Symmetry

When vector pair (u_i, v_i) arrives perform edge query with

S_i: set indicated by $x_i = \begin{bmatrix} u_i \\ 0 \end{bmatrix}$

T_i: set indicated by $y_i = \begin{bmatrix} 0 \\ v_i \end{bmatrix}$

Observation 1: $x_i^T M' y_i = u_i^T M v_i$

Observation 2: There is an edge in $E(S_i, T_i)$ iff $x_i^T M' y_i = 1$

max planck institut
informatik

We will show…

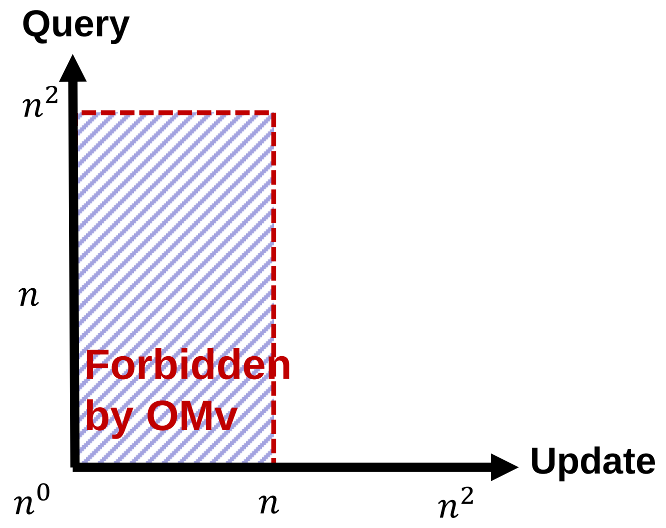
Fully dynamic SSR
- Preprocess: $poly(n)$
- Update: $n^{1-\epsilon}$ (amortized)
- Query: $n^{2-\epsilon}$

↓

Edge Query
- Preprocess: $poly(n)$
- Time (for n queries): $n^{3-\epsilon}$

↓

OMv conjecture fails

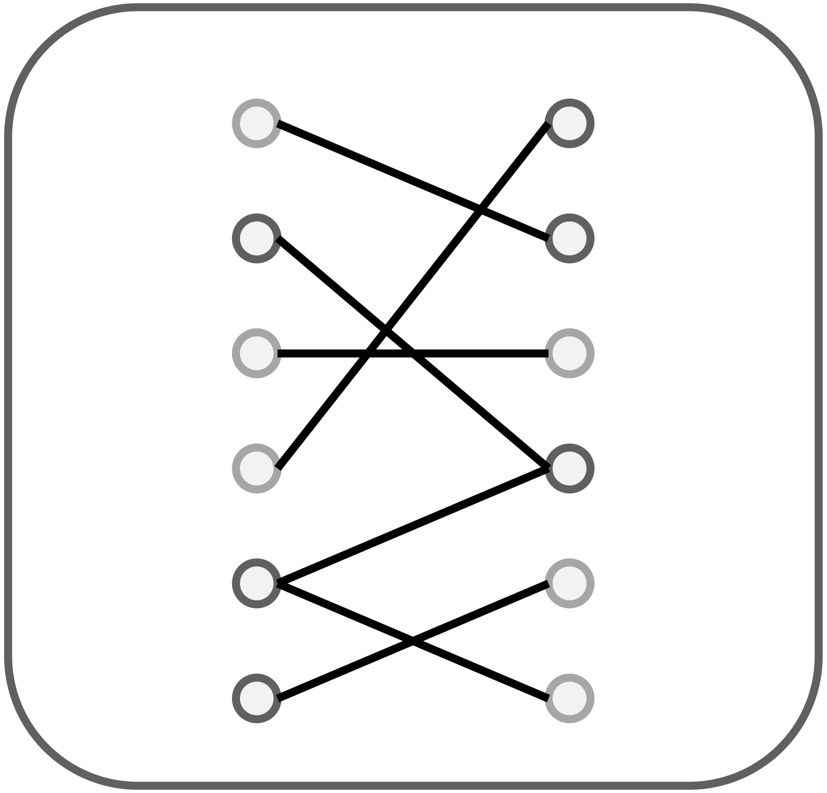


Recomputation from scratch upon query is optimal point on trade-off curve!

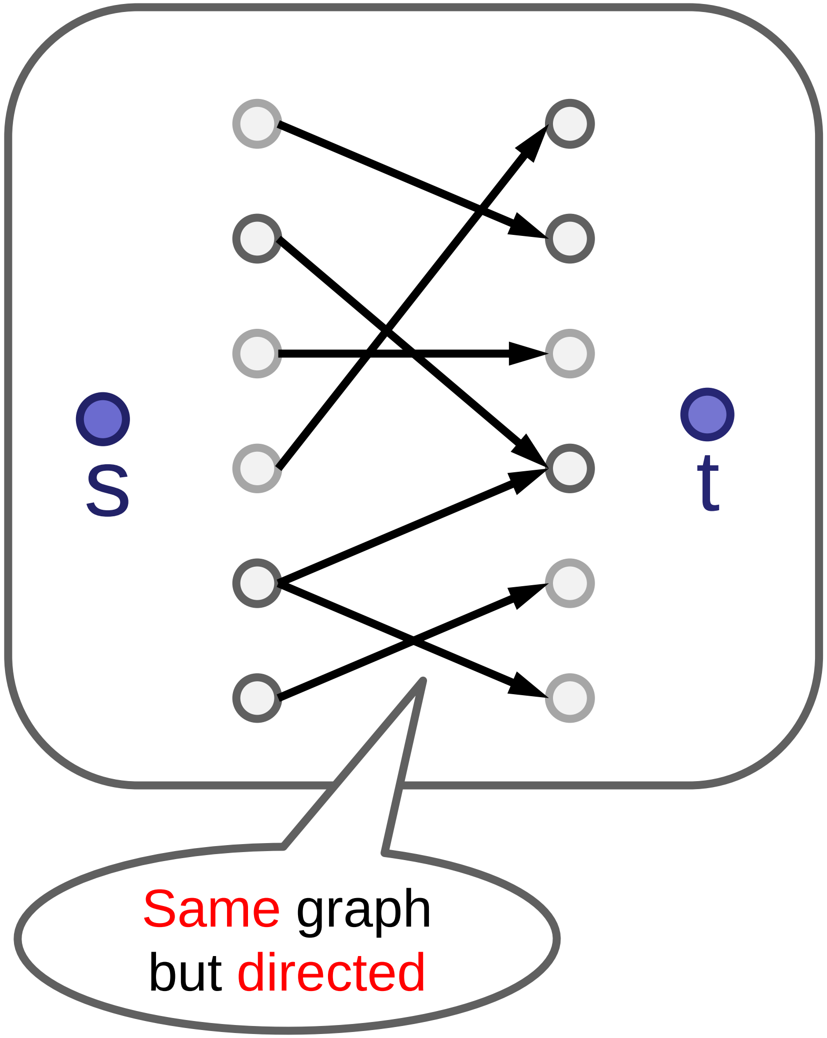
Lower bound on worst-case incremental/decremental: **rollback technique!**

Preprocess

Edge Query

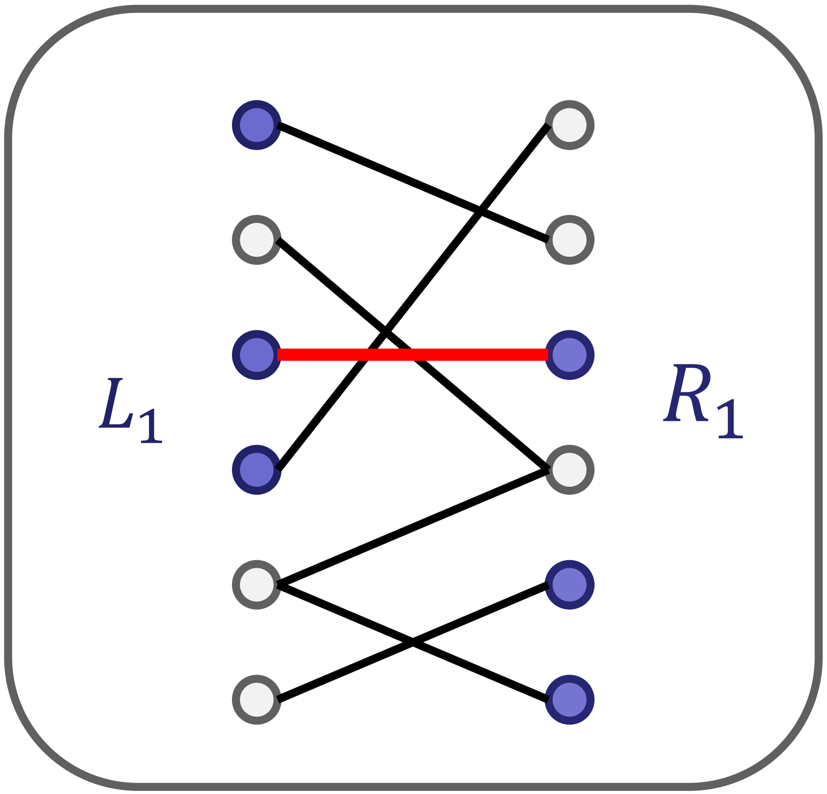


SSR

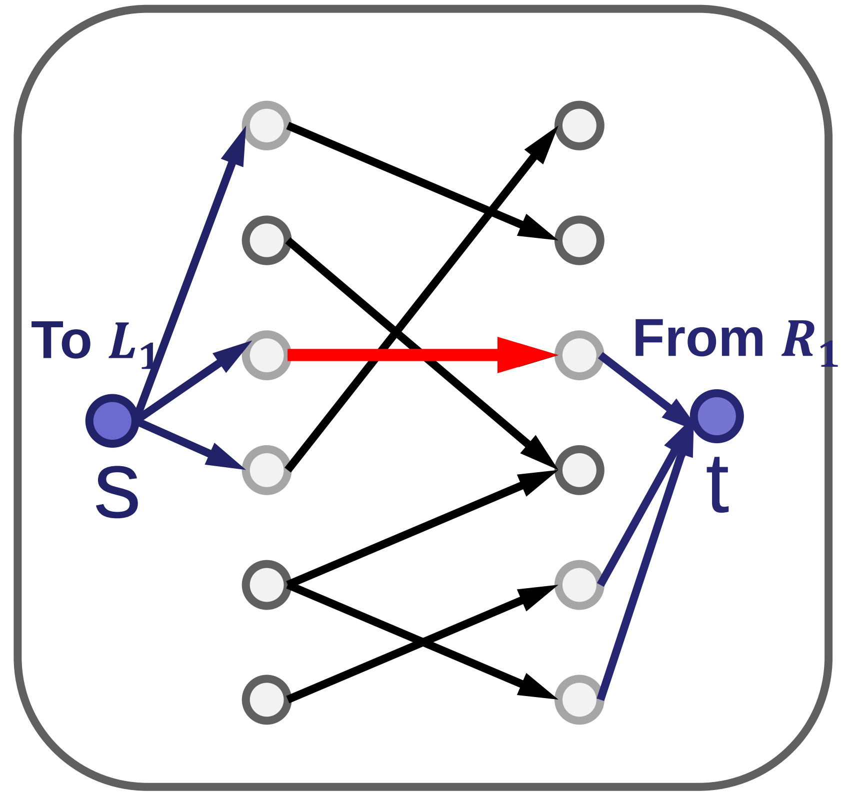


Edge(L_1, R_1)?

Edge Query



SSR

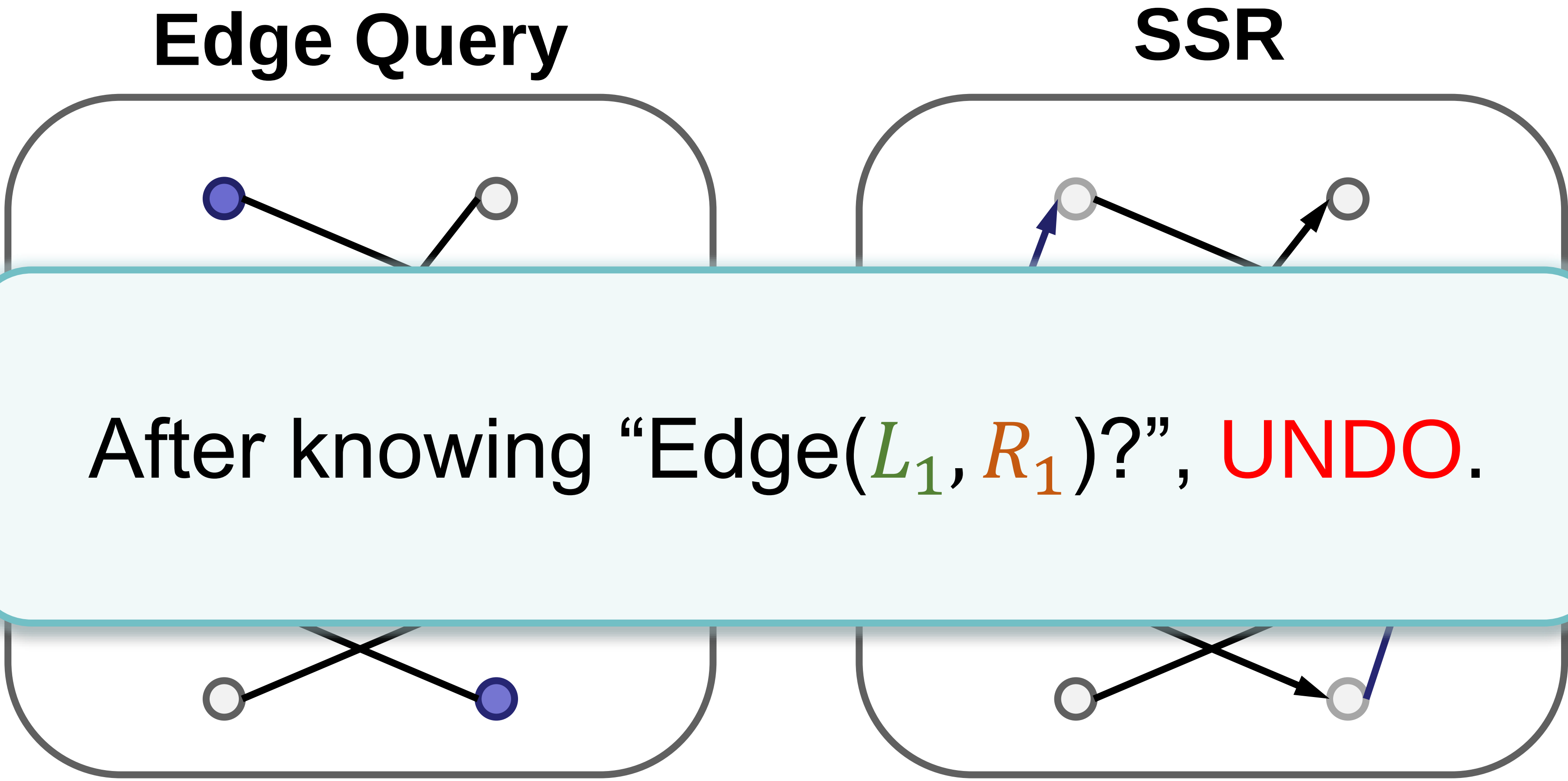


∃ an edge linking L_1 and R_1

After $O(n)$ updates…

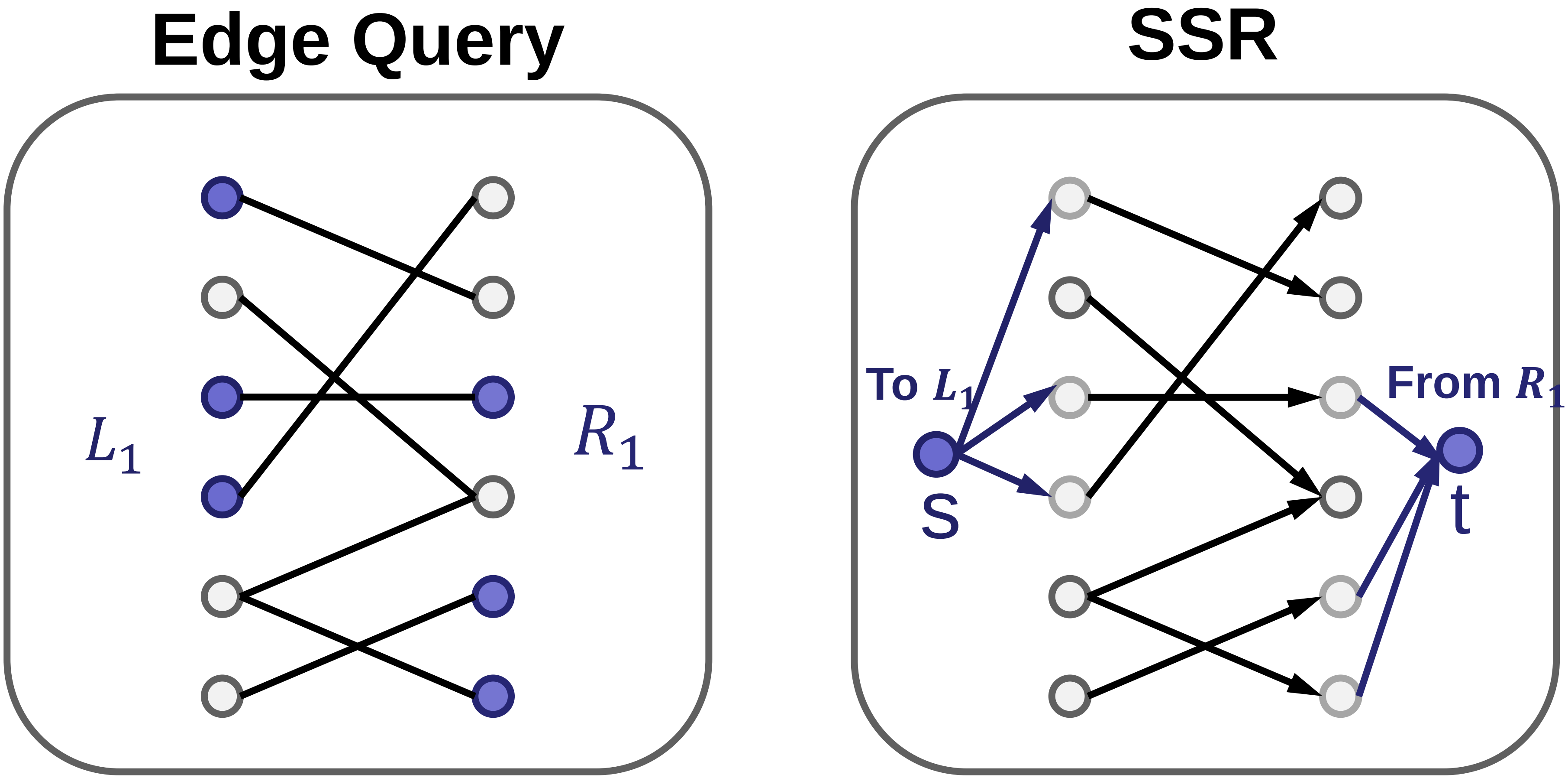
s can reach t

Edge(L_1, R_1)?



After knowing "Edge(L_1, R_1)?", UNDO.

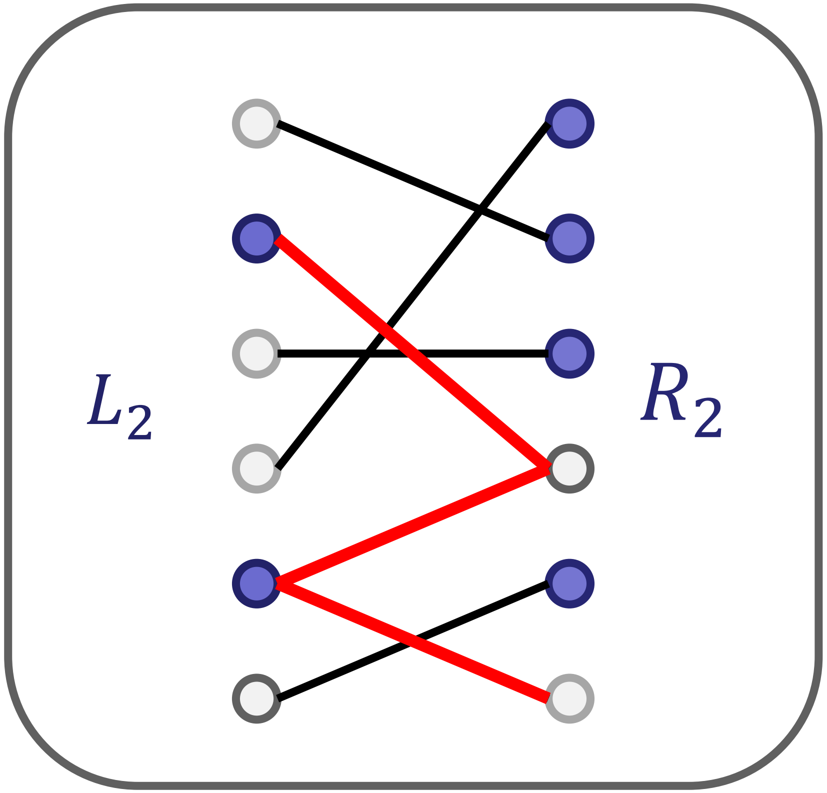
Edge(L_1, R_1)?



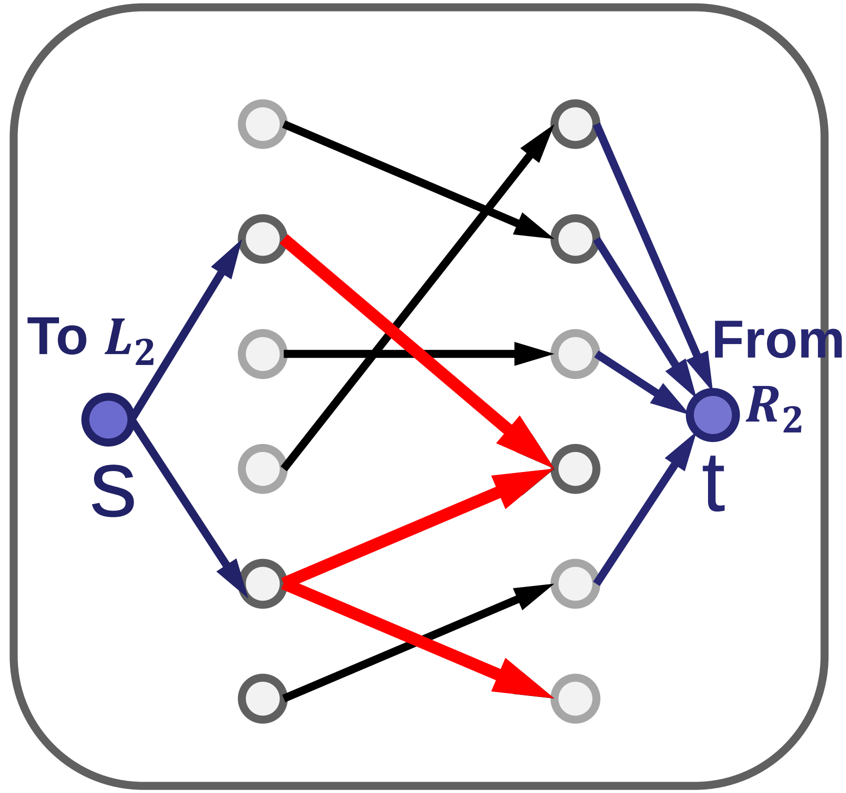
Use $O(n)$ updates.

Edge(L_2, R_2)? (another example)

Edge Query



SSR



Not ∃ an edge linking L_2 and R_2

↔

After $O(n)$ updates…

s can not reach t

Edge Query

Edge(L_i, R_i)?

SSR

Can answer using

#updates: $O(n)$

#query: 1

Edge Query

For $i = 1, \ldots, n$
Edge(L_i, R_i)?

SSR

Can answer using
#updates: $O(n^2)$
#queries: n

Edge Query

For $i = 1, \ldots, n$

Edge(L_i, R_i)?

SSR

Can answer using

#updates: $O(n^2)$

#queries: n

Edge Query

- Preprocess: $poly(n)$
- Time (for n queries):

ss-Reach

- Preprocess: $poly(n)$
- Update: $n^{1-\epsilon}$ (amortized)
- Query: $n^{2-\epsilon}$

Edge Query

For $i = 1, \ldots, n$
Edge(L_i, R_i)?

SSR

Can answer using
#updates: $O(n^2)$
#queries: n

Edge Query
- Preprocess: $poly(n)$
- Time (for n queries):
  $O(n^2) \times n^{1-\epsilon}$

ss-Reach
- Preprocess: $poly(n)$
- Update: $n^{1-\epsilon}$ (amortized)
- Query: $n^{2-\epsilon}$

Edge Query

For $i = 1, \dots, n$
Edge(L_i, R_i)?

SSR

Can answer using
#updates: $O(n^2)$
#queries: n

Edge Query
- Preprocess: $poly(n)$
- Time (for n queries):

$$O(n^2) \times n^{1-\epsilon} + n \times n^{2-\epsilon}$$

ss-Reach
- Preprocess: $poly(n)$
- Update: $n^{1-\epsilon}$ (amortized)
- Query: $n^{2-\epsilon}$

Edge Query

For $i = 1, \ldots, n$
Edge(L_i, R_i)?

SSR

Can answer using
#updates: $O(n^2)$
#queries: n

Edge Query
- Preprocess: $poly(n)$
- Time (for n queries):

$$O(n^2) \times n^{1-\epsilon}$$
$$+ n \times n^{2-\epsilon}$$
$$\leq O(n^{3-\epsilon})$$

ss-Reach
- Preprocess: $poly(n)$
- Update: $n^{1-\epsilon}$ (amortized)
- Query: $n^{2-\epsilon}$

OMv conjecture fails

max planck institut informatik

Q.E.D.

Many popular conjectures…

Conjectures	References
BMM (Boolean Matrix Multiplication)	STOC'02 Chan, ESA'04 Rodiity, Zwick, FOCS'14 Abboud, V-Williams
3SUM	FOCS'14 Abboud, V-Williams, Arxiv'14 Kopelowitz, Pettie, Porat
Multiphase (Based on 3SUM)	STOC'10 Patrascu
Triangle	FOCS'14 Abboud, V-Williams
APSP (All Pair Shortest Path)	FOCS'14 Abboud, V-Williams
SETH (Strong Exponential Time Hypothesis)	FOCS'14 Abboud, V-Williams
Matching Triangle (Based on 3SUM, APSP and SETH)	STOC'15 Abboud, V-Williams, Yu