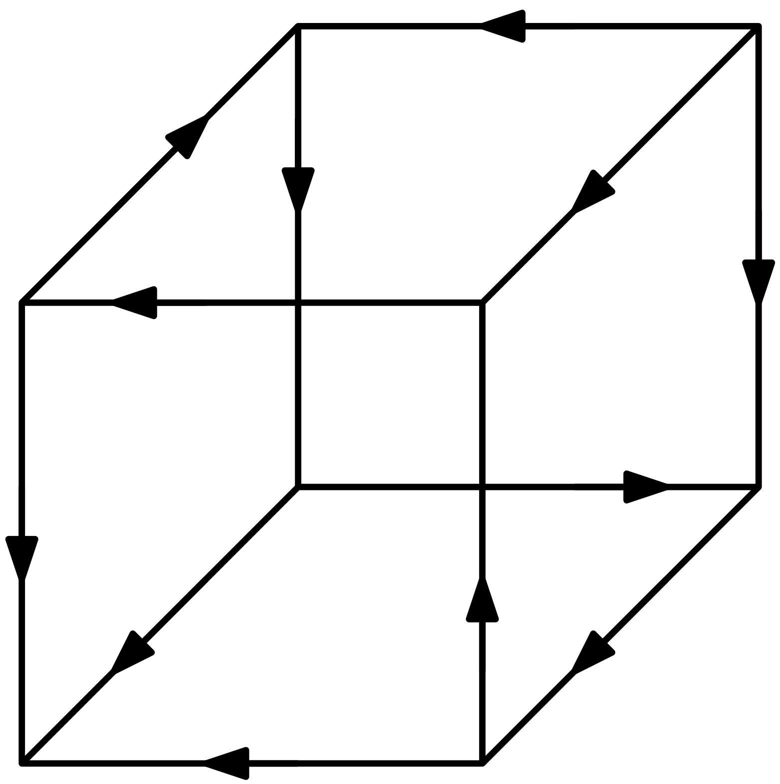
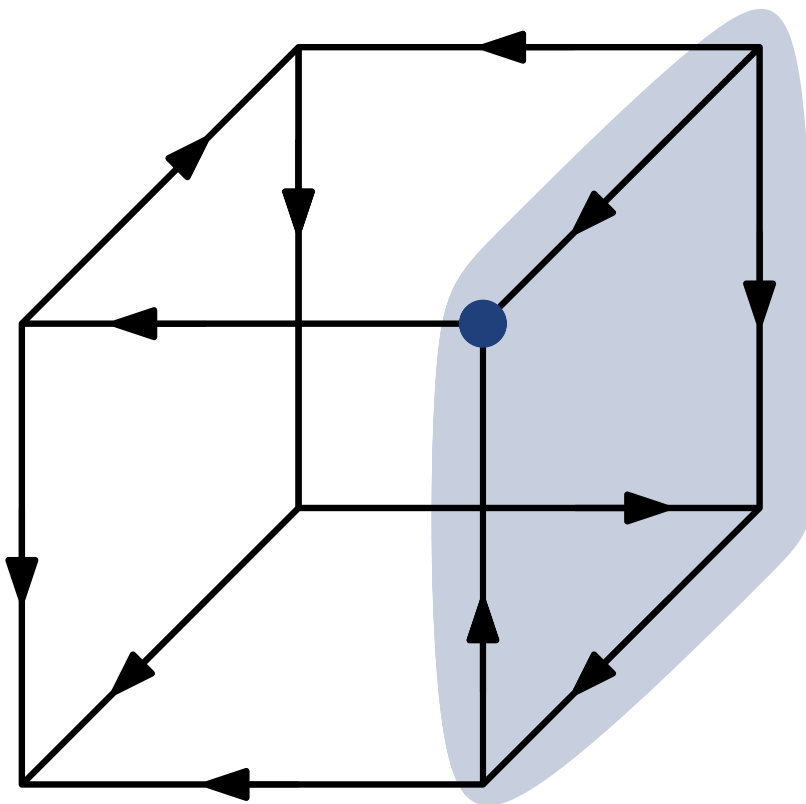
Realizability Makes a Difference: A Complexity Gap for Sink-Finding in USOs

Simon Weber and Joel Widmer

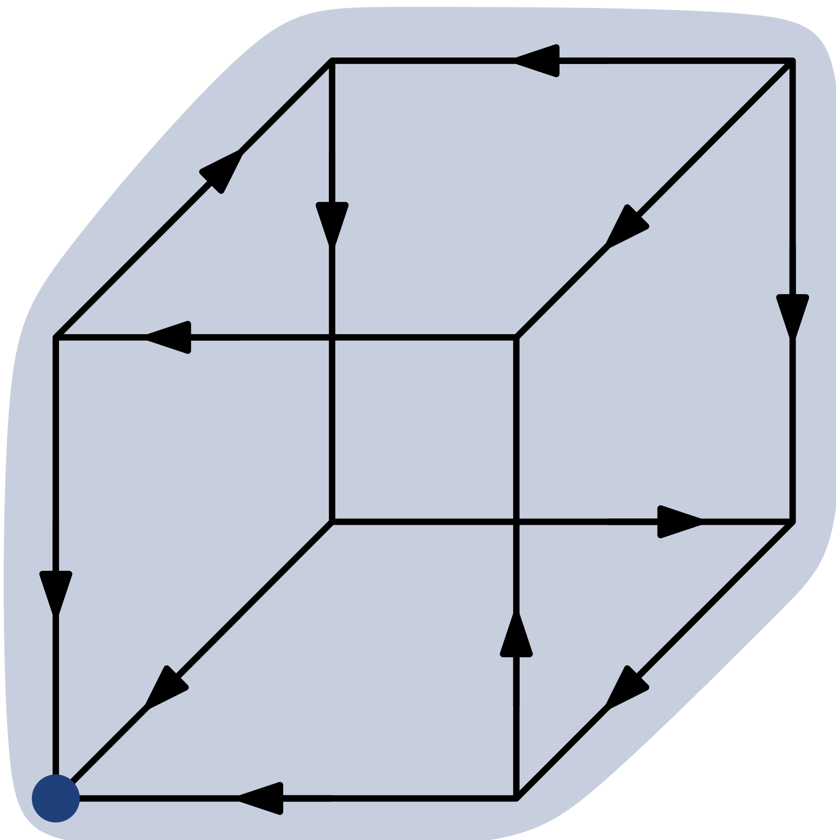
Unique Sink Orientations



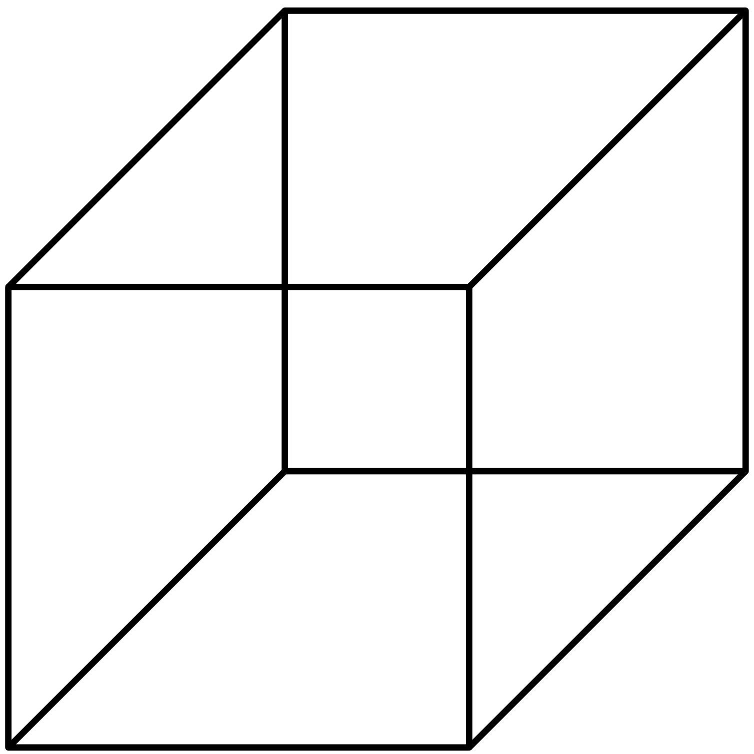
Unique Sink Orientations



Unique Sink Orientations



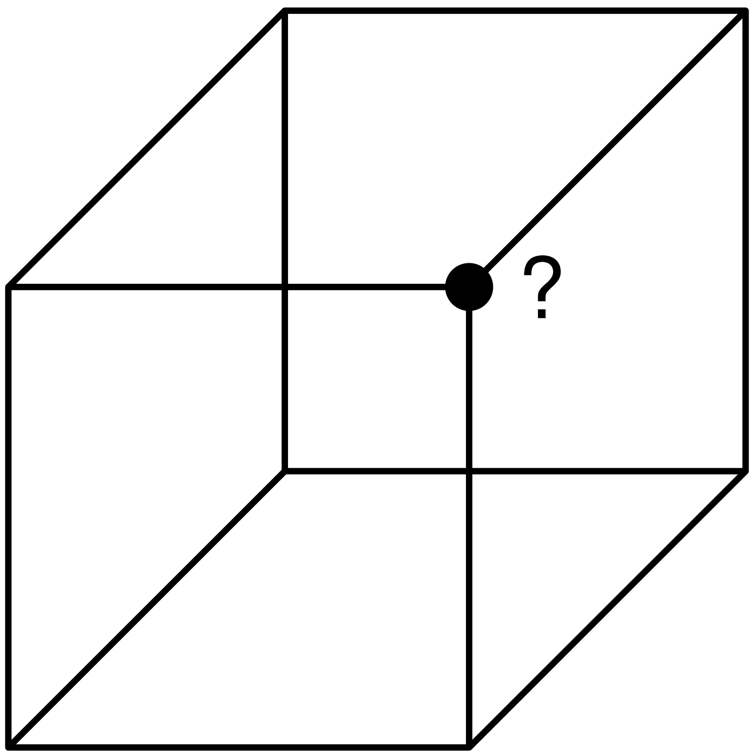
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

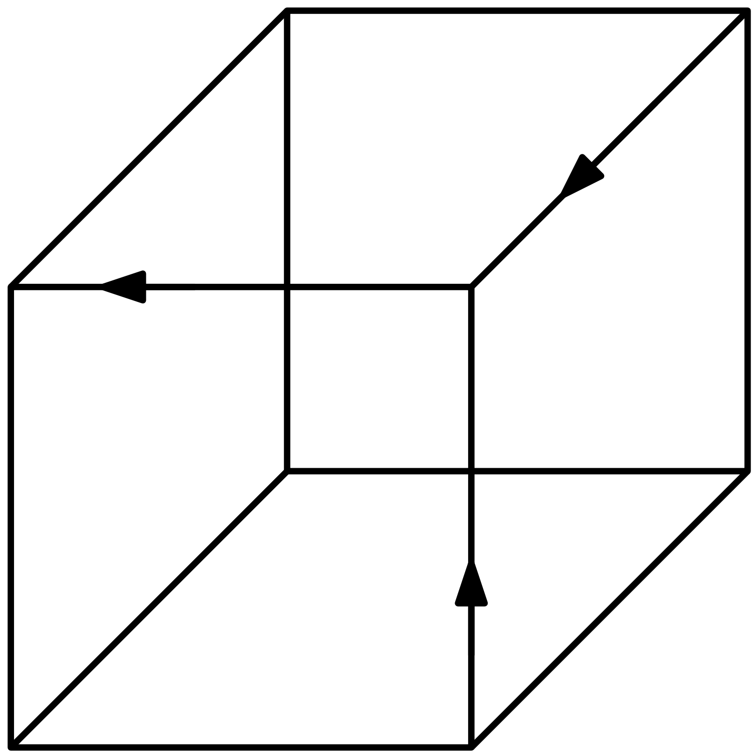
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

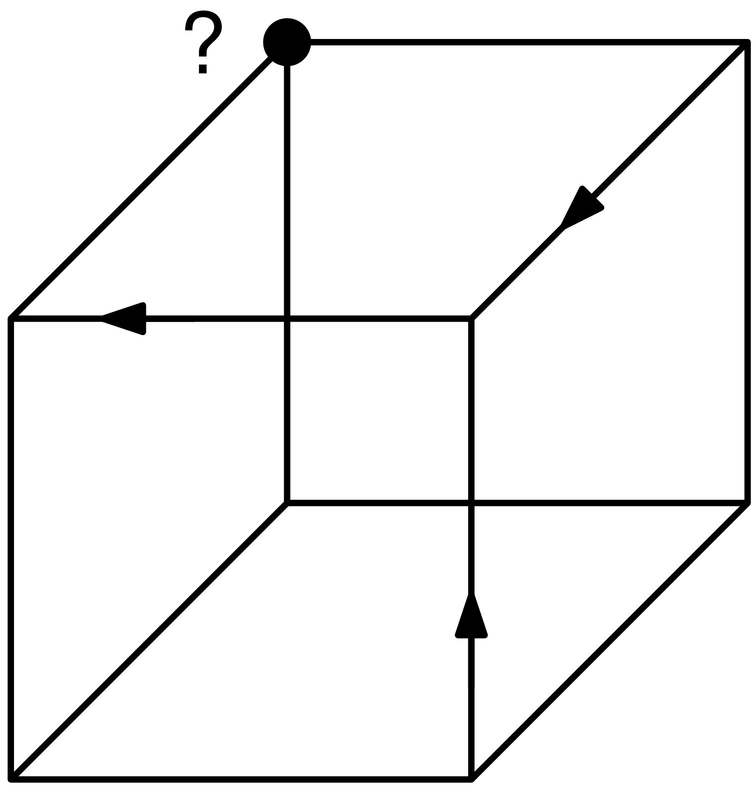
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

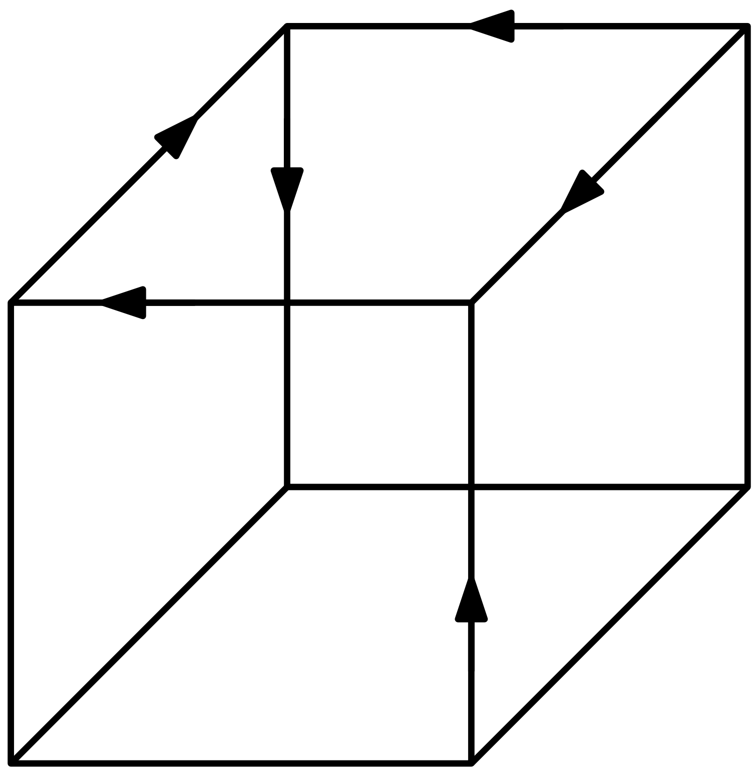
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

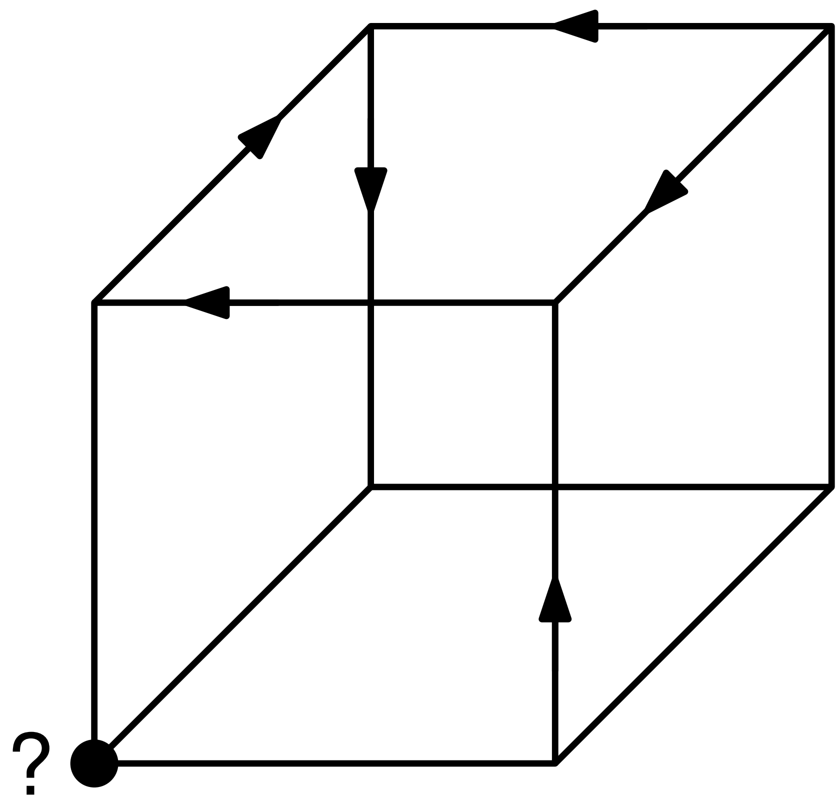
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

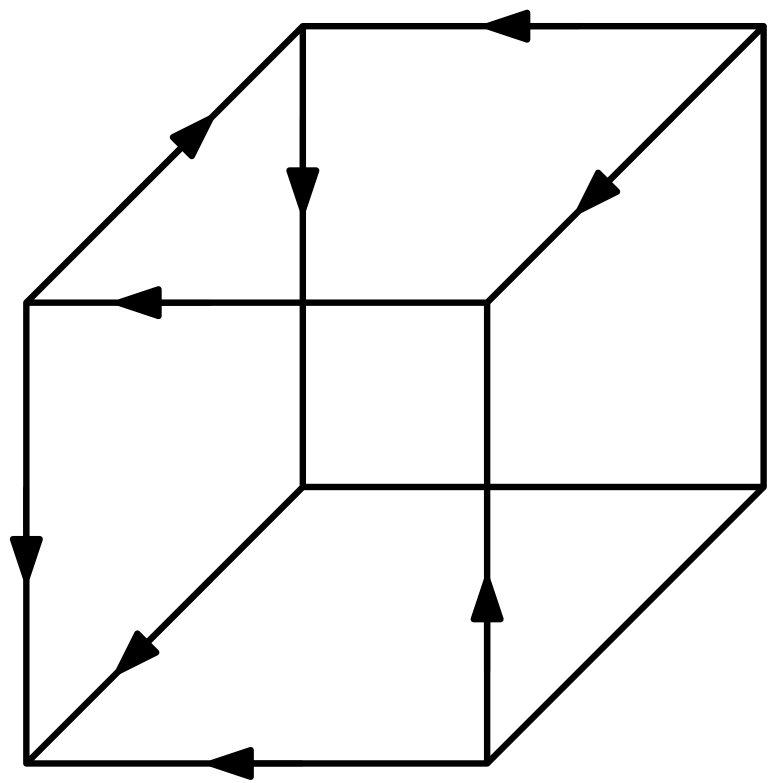
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

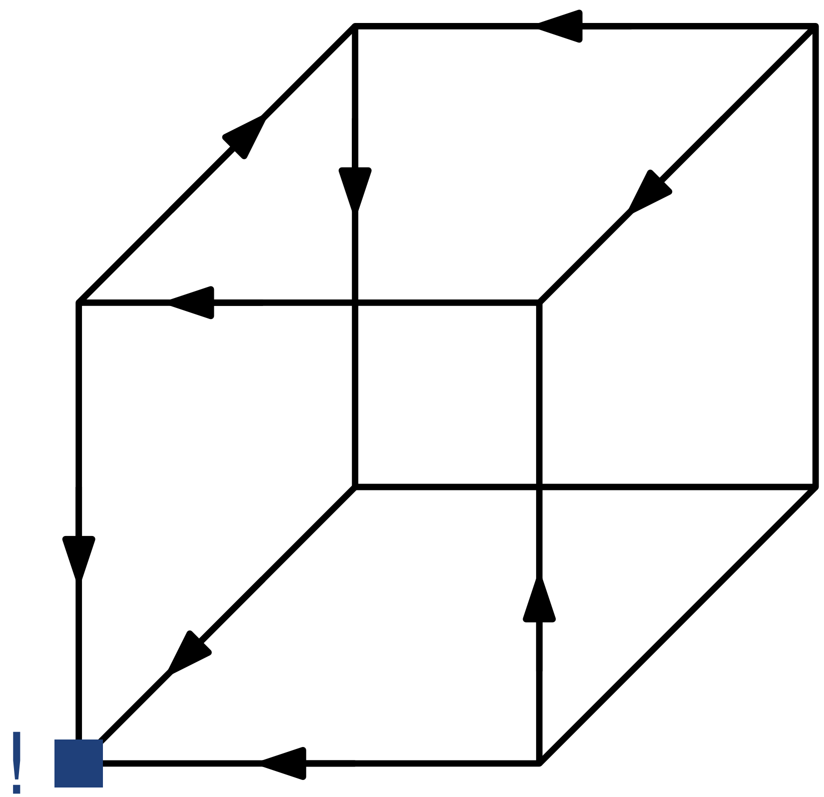
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

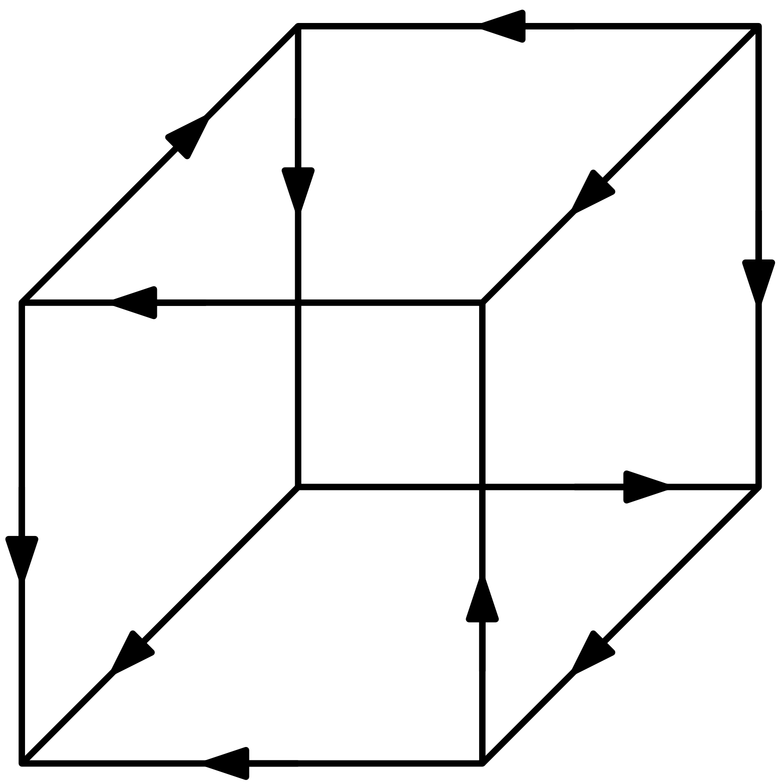
Unique Sink Orientations



Sink-finding:

Find the global sink using *vertex evaluations*

Unique Sink Orientations

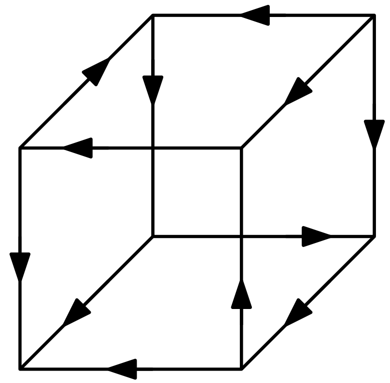


Sink-finding:

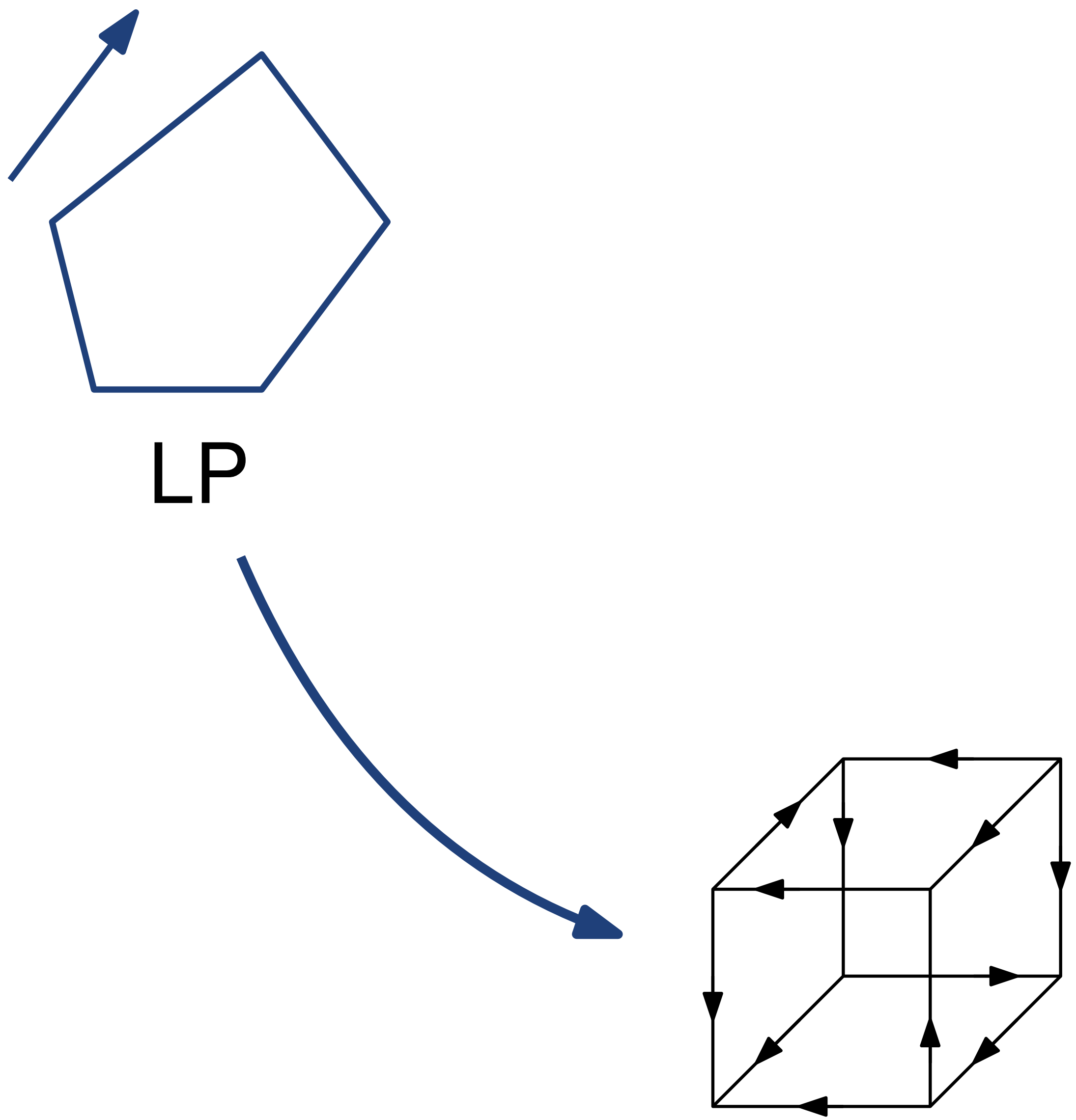
Find the global sink using *vertex evaluations*

Runtime = Number of vertex evaluations

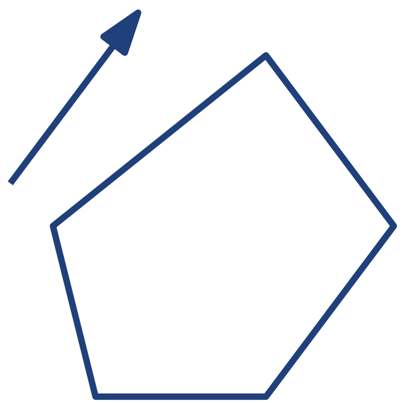
Realizability

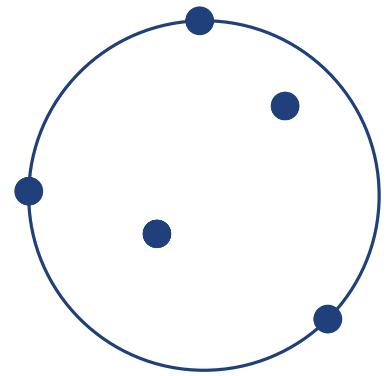


Realizability



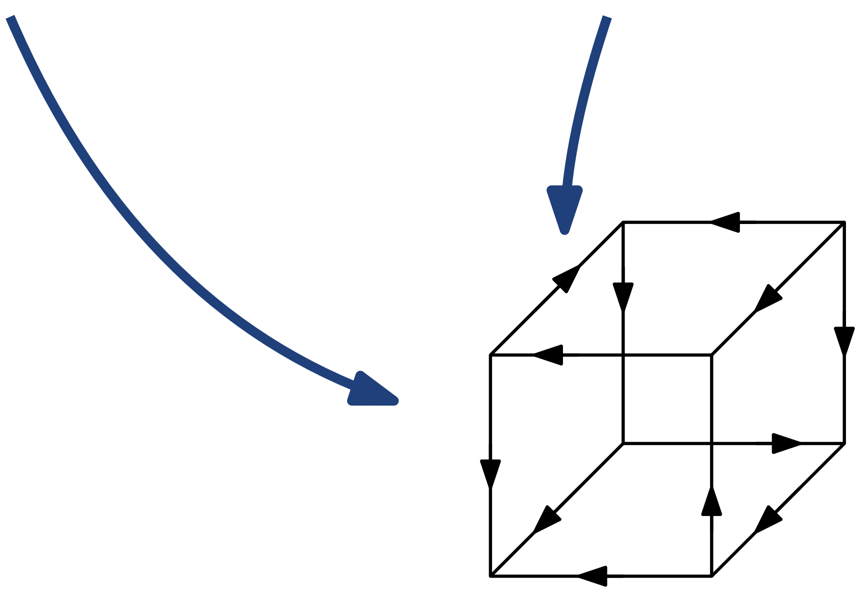
Realizability



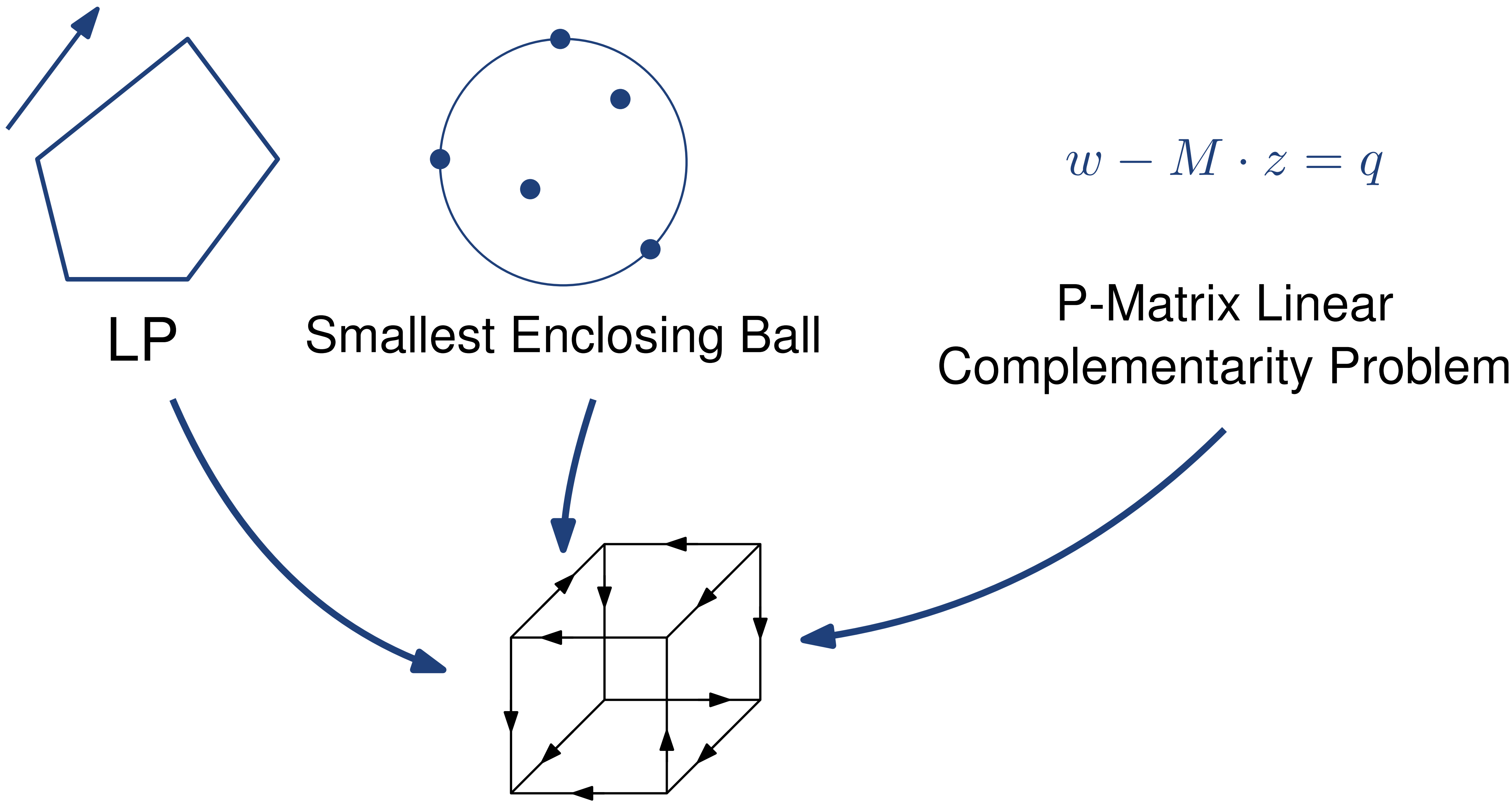


LP

Smallest Enclosing Ball



Realizability



Realizability

Definition: A USO is called *realizable* if it can be obtained through the reduction from P-LCP.

Realizability

Definition: A USO is called *realizable* if it can be obtained through the reduction from P-LCP.

This is a small subset!

Realizability

Definition: A USO is called *realizable* if it can be obtained through the reduction from P-LCP.

This is a small subset!

Theorem: There are $2^{\Theta(2^n \log n)}$ USOs of the n-cube. Out of those, only $2^{\Theta(n^3)}$ are realizable.

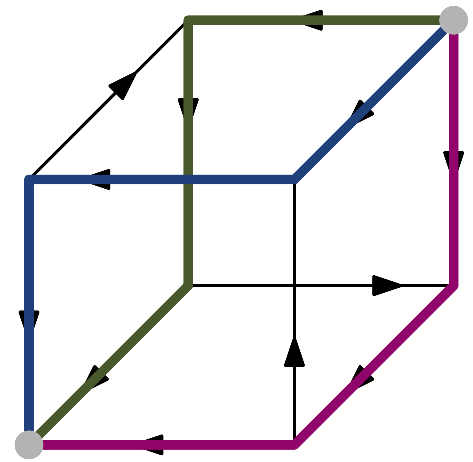
Realizability

Definition: A USO is called *realizable* if it can be obtained through the reduction from P-LCP.

This is a small subset!

Theorem: There are $2^{\Theta(2^n \log n)}$ USOs of the n-cube. Out of those, only $2^{\Theta(n^3)}$ are realizable.

Holt-Klee property



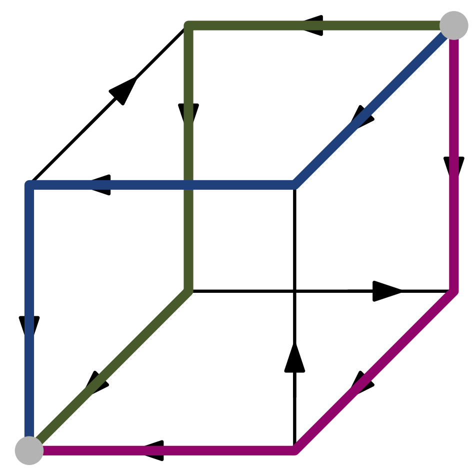
Realizability

Definition: A USO is called *realizable* if it can be obtained through the reduction from P-LCP.

This is a small subset!

Theorem: There are $2^{\Theta(2^n \log n)}$ USOs of the n-cube. Out of those, only $2^{\Theta(n^3)}$ are realizable.

Holt-Klee property



Necessary, but not sufficient!
Fulfilled by $2^{\Omega(2^n/\sqrt{n})}$ USOs

Query Complexity Lower Bounds

Theorem [Schurr, Szábo, 2004]: A deterministic sink-finding algorithm needs $\Omega(n^2/\log n)$ vertex evaluations in the worst case.

Query Complexity Lower Bounds

Theorem [Schurr, Szábo, 2004]: A deterministic sink-finding algorithm needs $\Omega(n^2/\log n)$ vertex evaluations in the worst case.

Theorem [Borzechowski, W., 2023]: *On realizable USOs*, a deterministic sink-finding algorithm needs at least n vertex evaluations in the worst case.

Query Complexity Lower Bounds

Theorem [Schurr, Szábo, 2004]: A deterministic sink-finding algorithm needs $\Omega(n^2/\log n)$ vertex evaluations in the worst case.

Theorem [Borzechowski, W., 2023]: *On realizable USOs*, a deterministic sink-finding algorithm needs at least n vertex evaluations in the worst case.

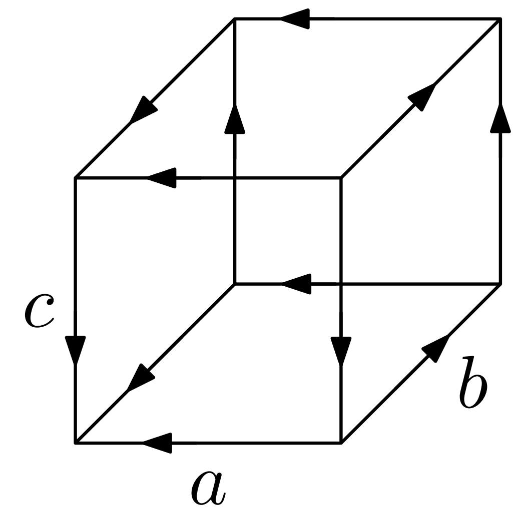
The best algorithms are exponential!

Query Complexity of Random Facet

Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.

Query Complexity of Random Facet

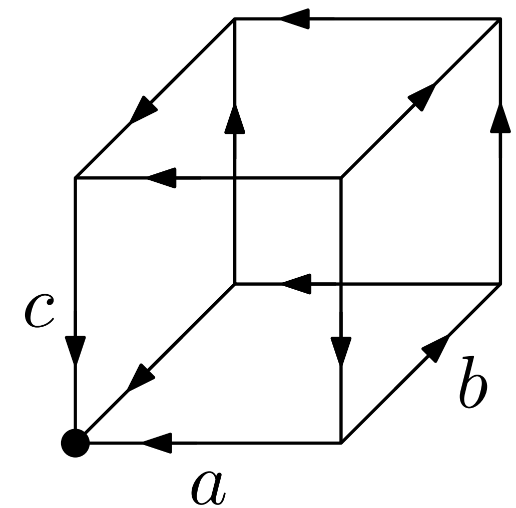
Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.



Matoušek USO

Query Complexity of Random Facet

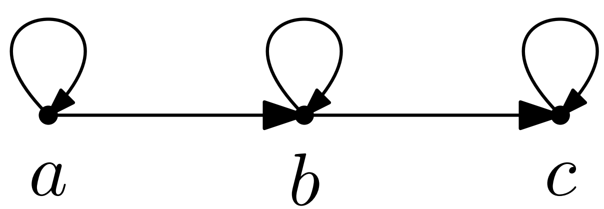
Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.



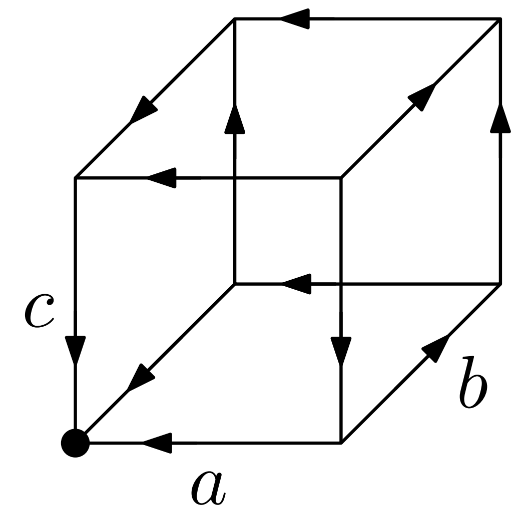
Matoušek USO

Query Complexity of Random Facet

Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.



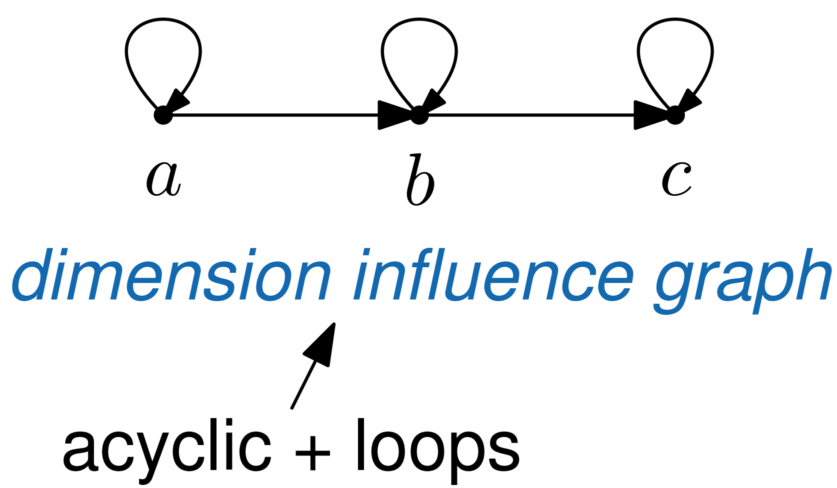
dimension influence graph

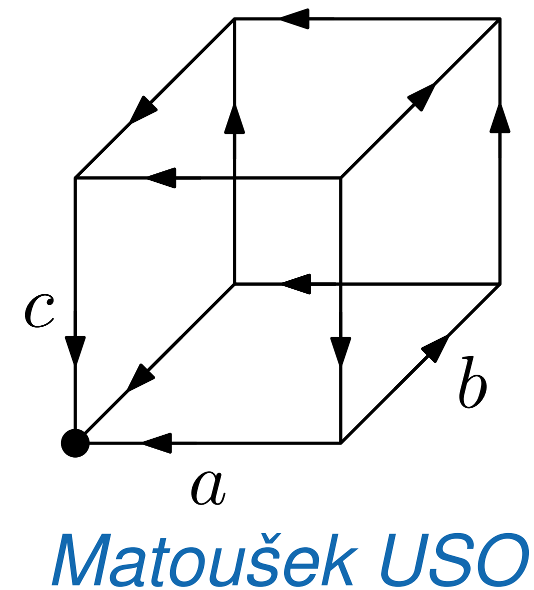


Matoušek USO

Query Complexity of Random Facet

Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.





Query Complexity of Random Facet

Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.

Theorem [Gärtner, 1994]: The expected number of queries needed by Random Facet on *realizable* Matoušek USOs is $O(n^2)$.

Query Complexity of Random Facet

Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.

Theorem [Gärtner, 1994]: The expected number of queries needed by Random Facet on *realizable* Matoušek USOs is $O(n^2)$.

For Random Facet, there is a clear *complexity gap* in realizable vs. non-realizable Matoušek USOs.

Query Complexity of Random Facet

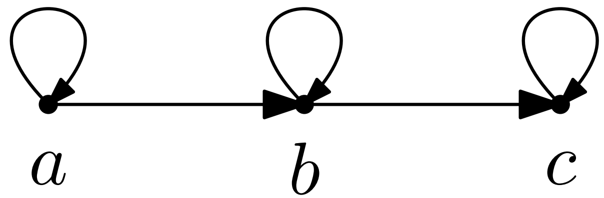
Theorem [Matoušek, 1994]: The expected number of queries needed by the *Random Facet* algorithm is $2^{\Omega(\sqrt{n})}$ in the worst case.

Theorem [Gärtner, 1994]: The expected number of queries needed by Random Facet on *realizable* Matoušek USOs is $O(n^2)$.

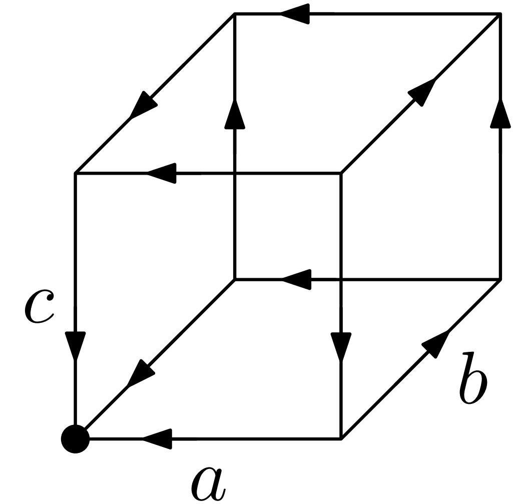
For Random Facet, there is a clear *complexity gap* in realizable vs. non-realizable Matoušek USOs.

Is this specific to Random Facet or *inherent to the problem*?

Matoušek-type USOs

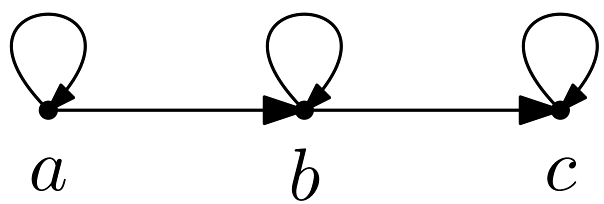


dimension influence graph

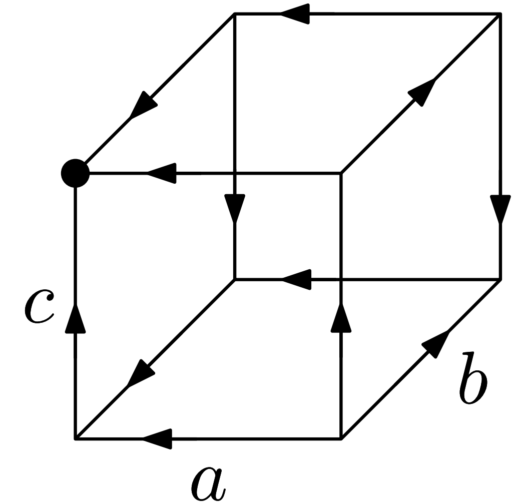


Matoušek USO

Matoušek-type USOs



dimension influence graph



Matoušek-type USO

The Complexity of Sink-Finding on Matoušek-type USOs

Theorem [W., Widmer, 2023]: An optimal deterministic algorithm needs exactly n queries to find the sink of a Matoušek-type USO in the worst case.

The Complexity of Sink-Finding on Matoušek-type USOs

Theorem [W., Widmer, 2023]: An optimal deterministic algorithm needs exactly n queries to find the sink of a Matoušek-type USO in the worst case.

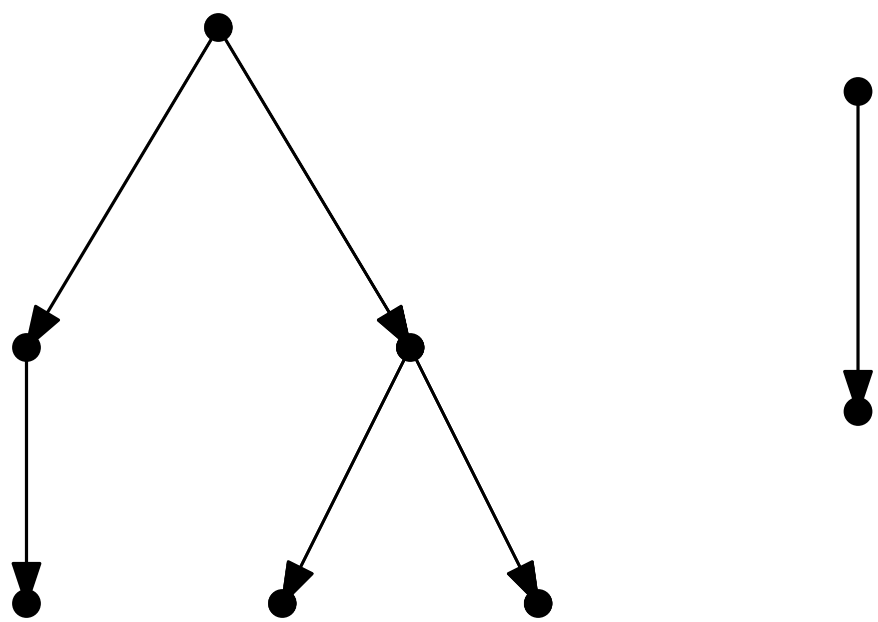
Theorem [W., Widmer, 2023]: An optimal deterministic algorithm needs between $O(\log^2 n)$ and $\Omega(\log n)$ queries to find the sink of a *realizable* Matoušek-type USO in the worst case.

The Realizable Matoušek-type USOs

Theorem [W., Gärtner, 2021]: A Matoušek-type USO is realizable if and only if its dimension influence graph is the *reflexive transitive closure of an arborescence*.

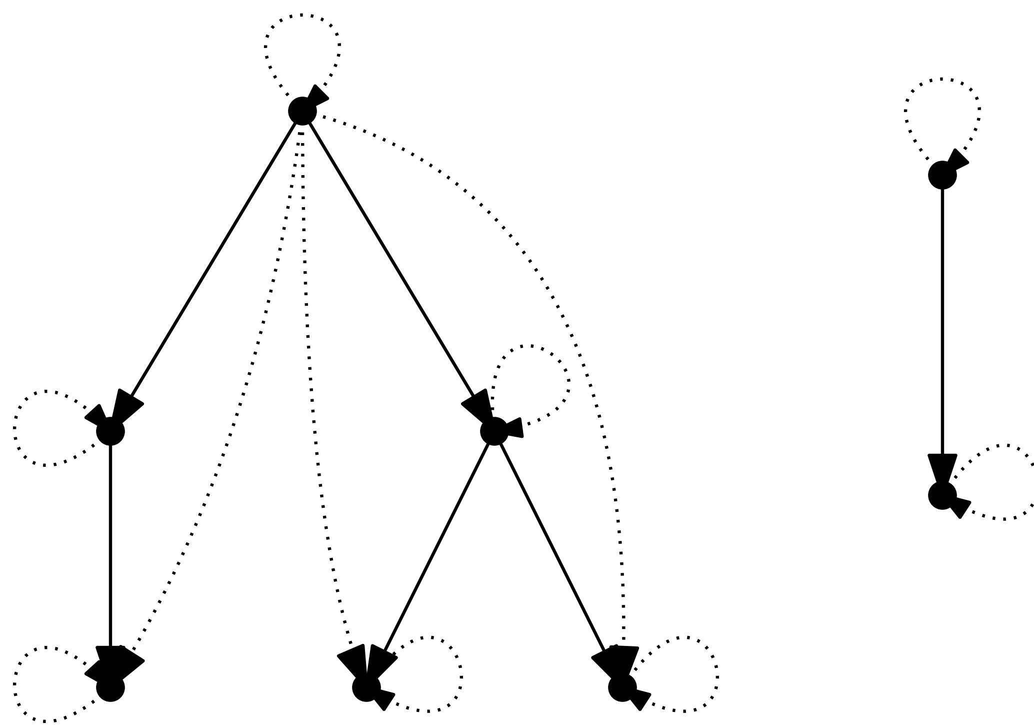
The Realizable Matoušek-type USOs

Theorem [W., Gärtner, 2021]: A Matoušek-type USO is realizable if and only if its dimension influence graph is the *reflexive transitive closure of an arborescence*.

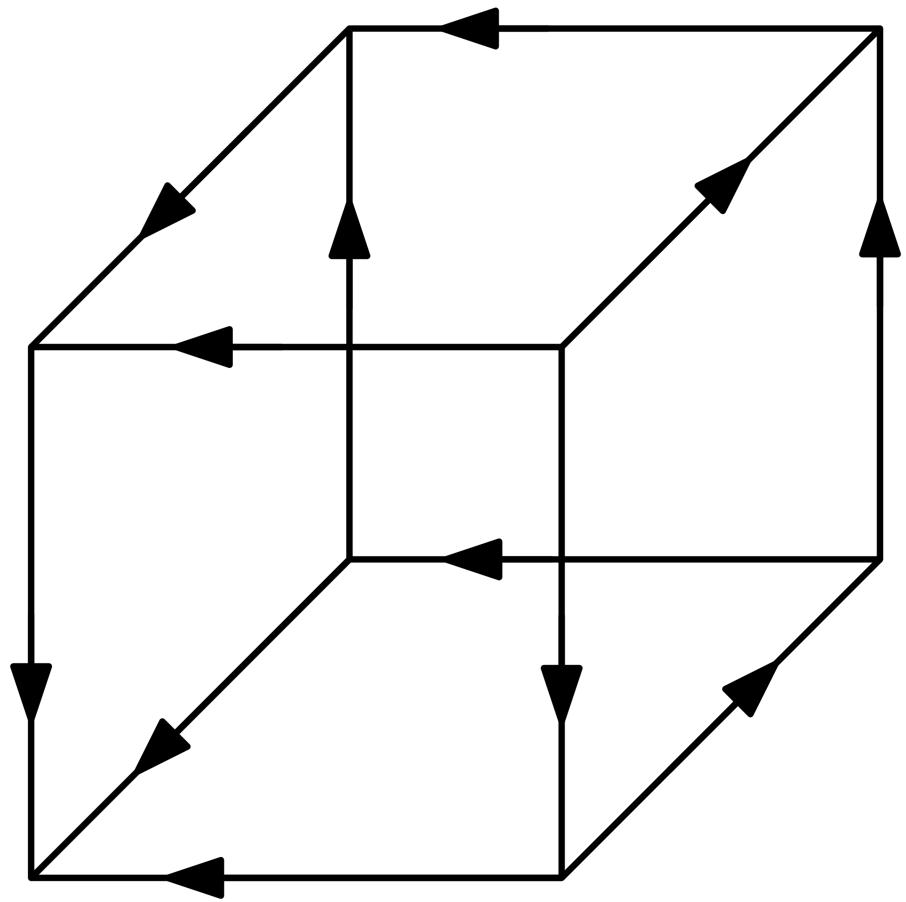


The Realizable Matoušek-type USOs

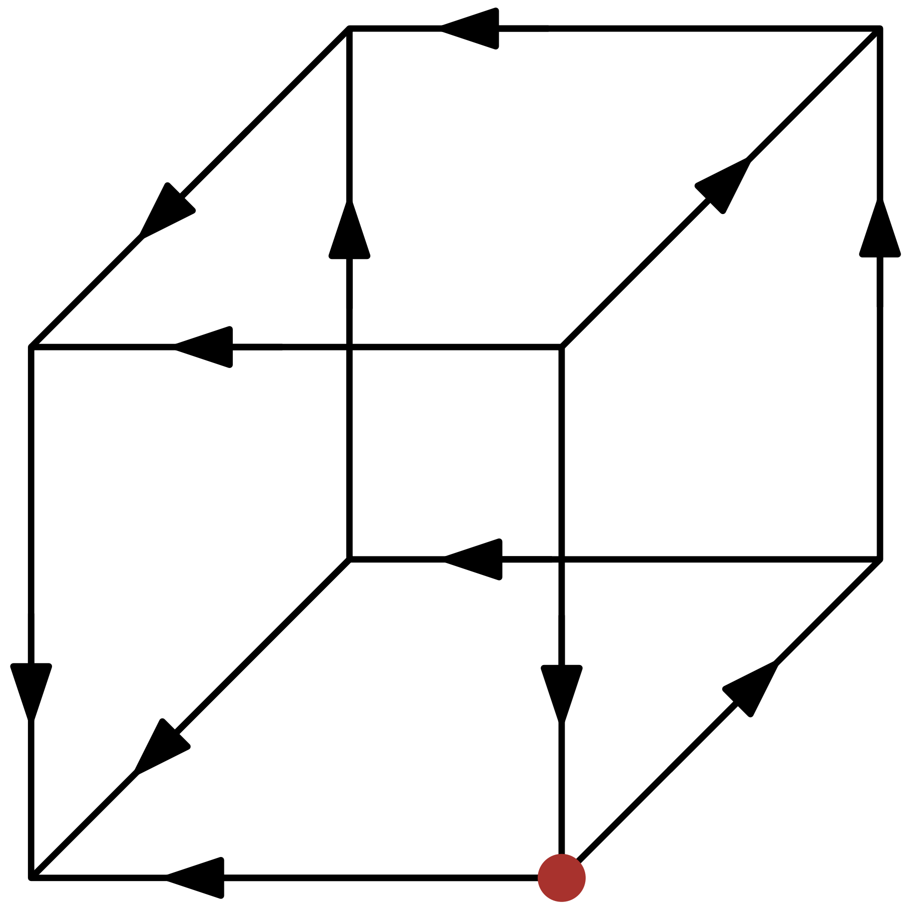
Theorem [W., Gärtner, 2021]: A Matoušek-type USO is realizable if and only if its dimension influence graph is the *reflexive transitive closure of an arborescence*.



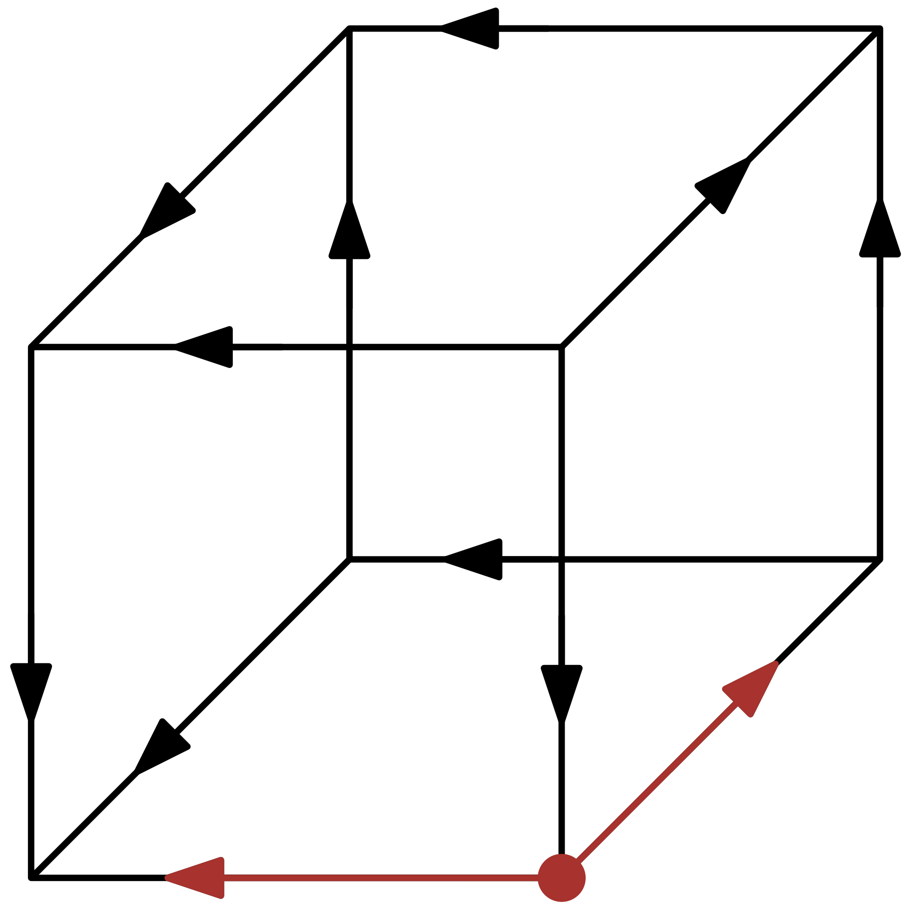
General Matoušek-type USOs: Upper Bound



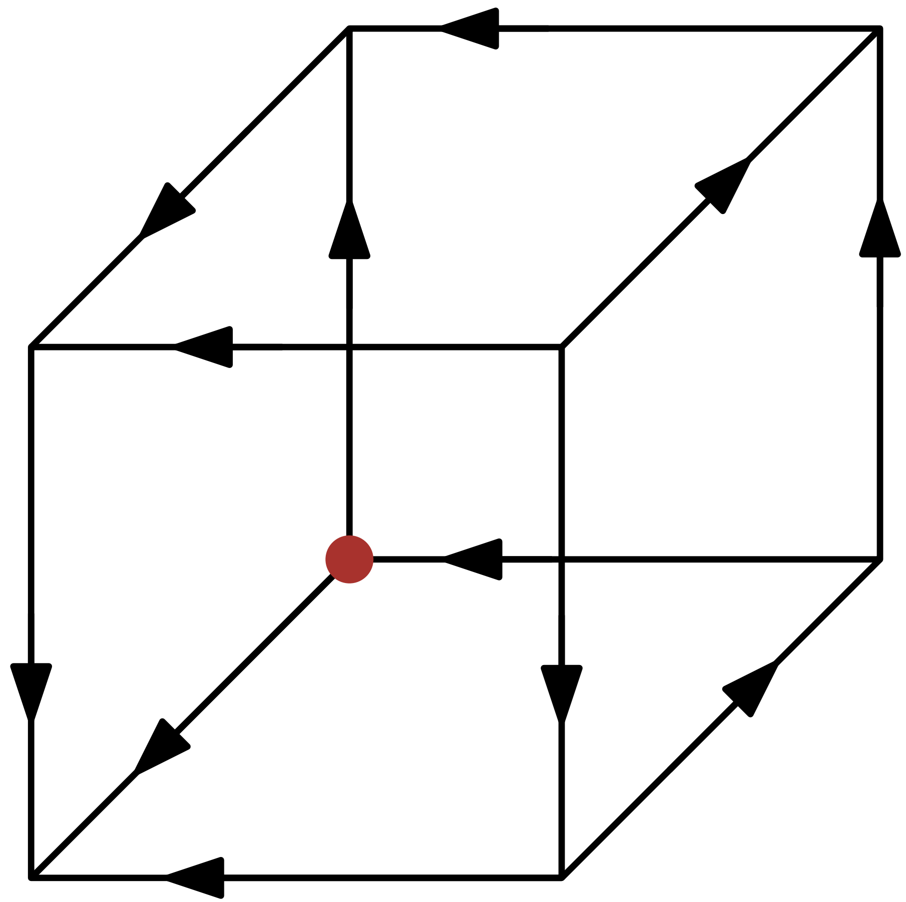
General Matoušek-type USOs: Upper Bound



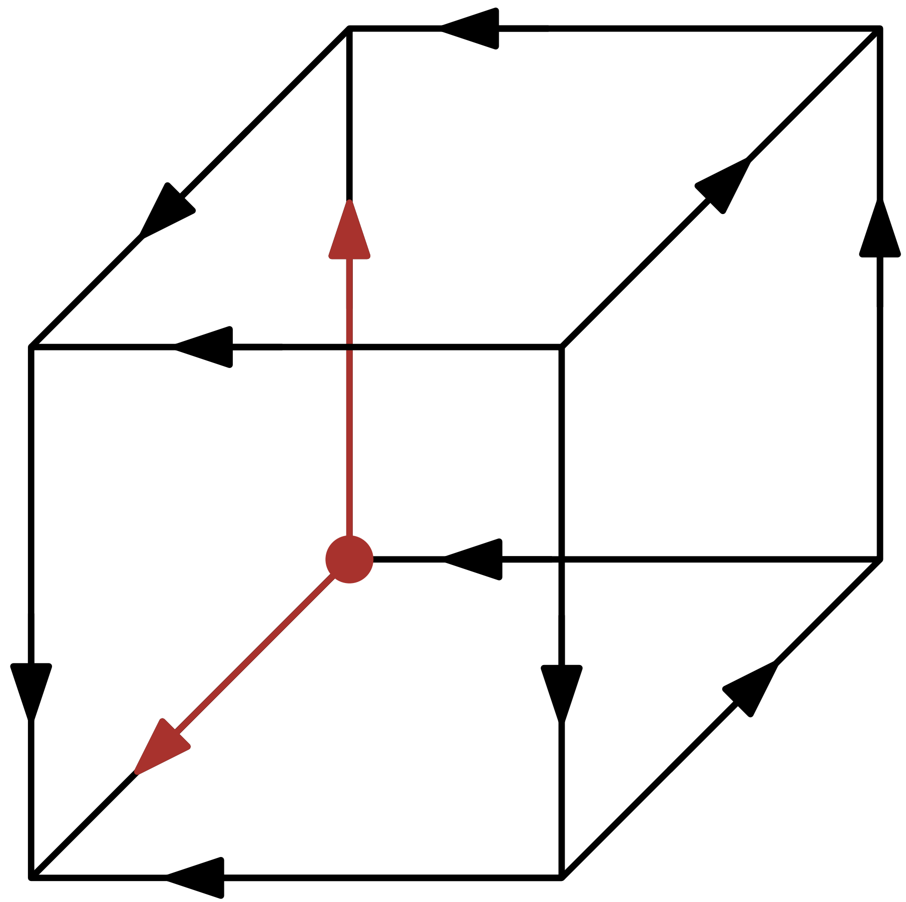
General Matoušek-type USOs: Upper Bound



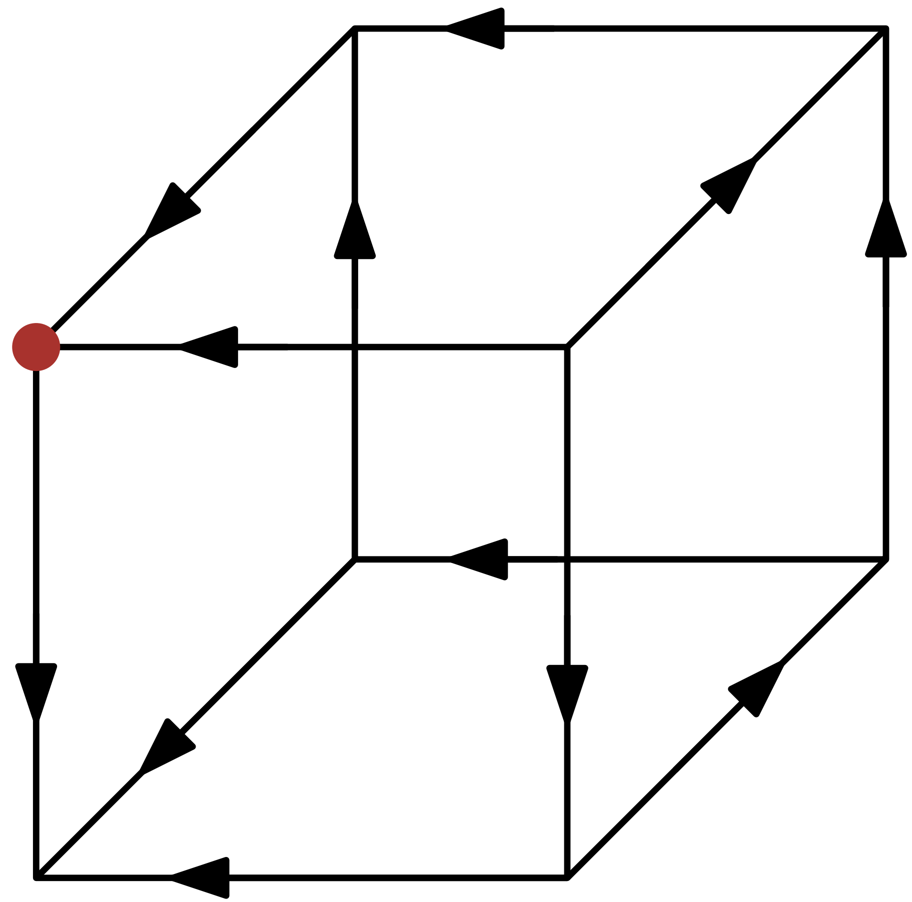
General Matoušek-type USOs: Upper Bound



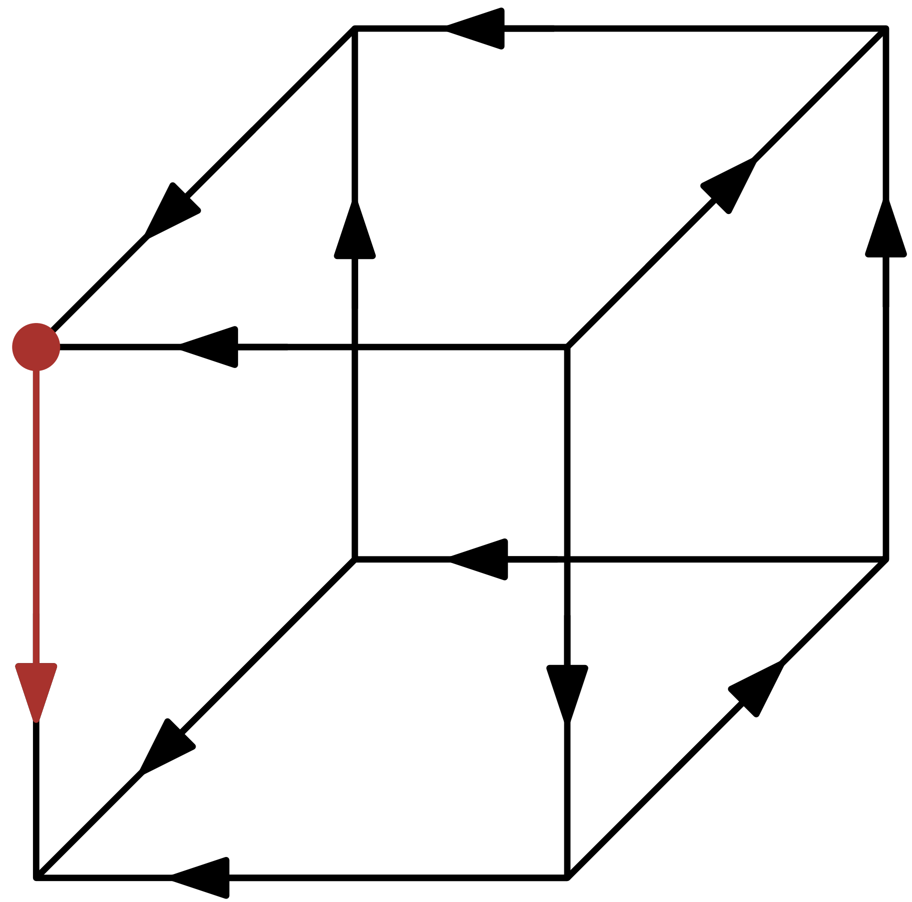
General Matoušek-type USOs: Upper Bound



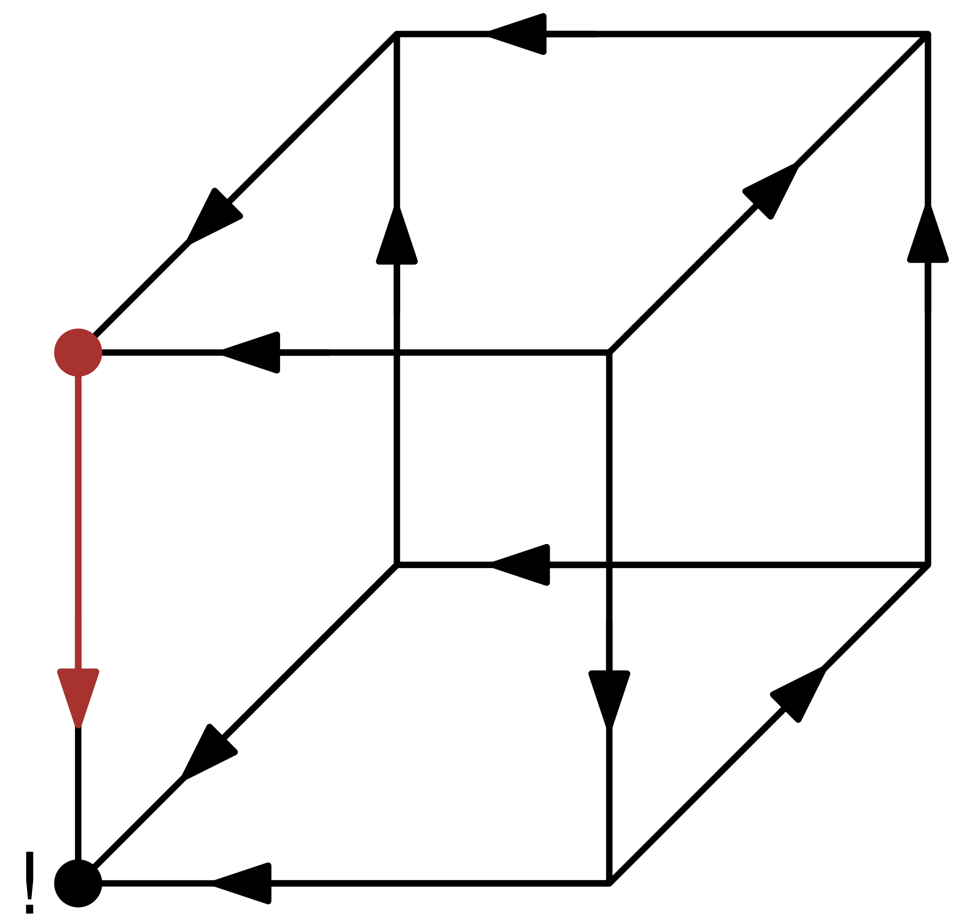
General Matoušek-type USOs: Upper Bound



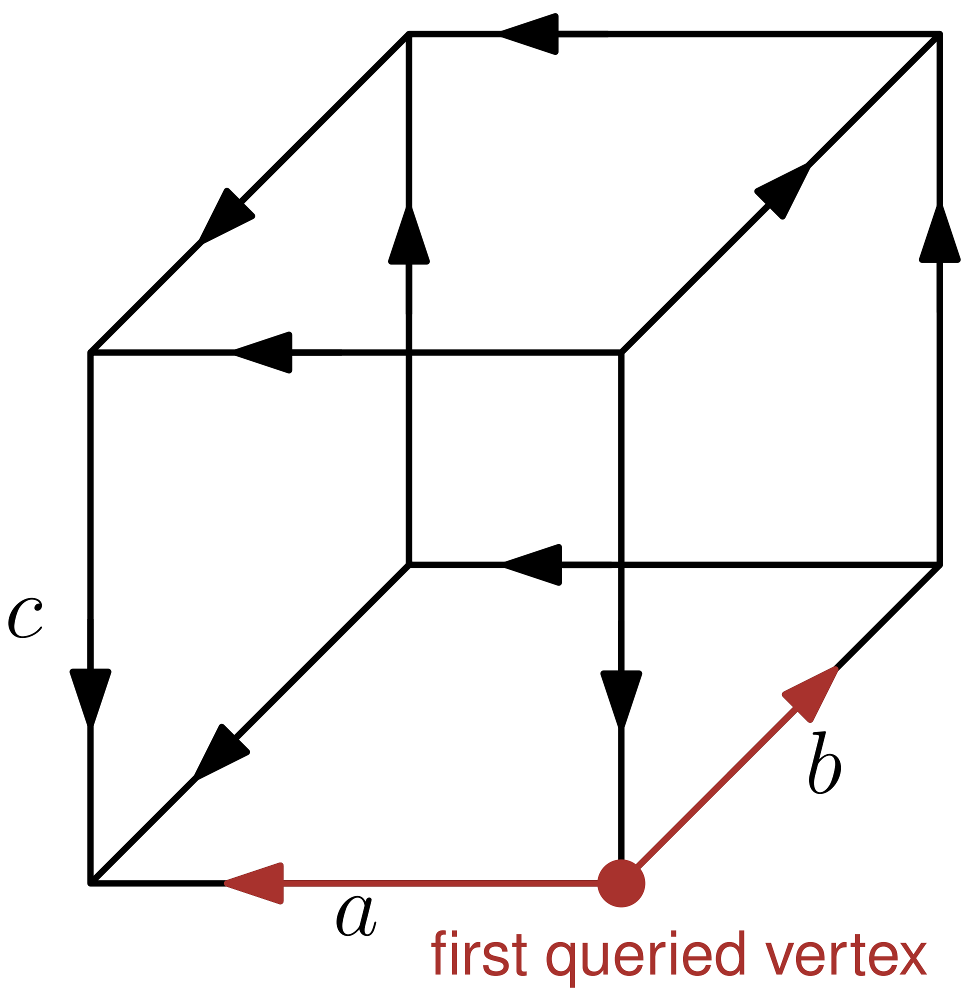
General Matoušek-type USOs: Upper Bound



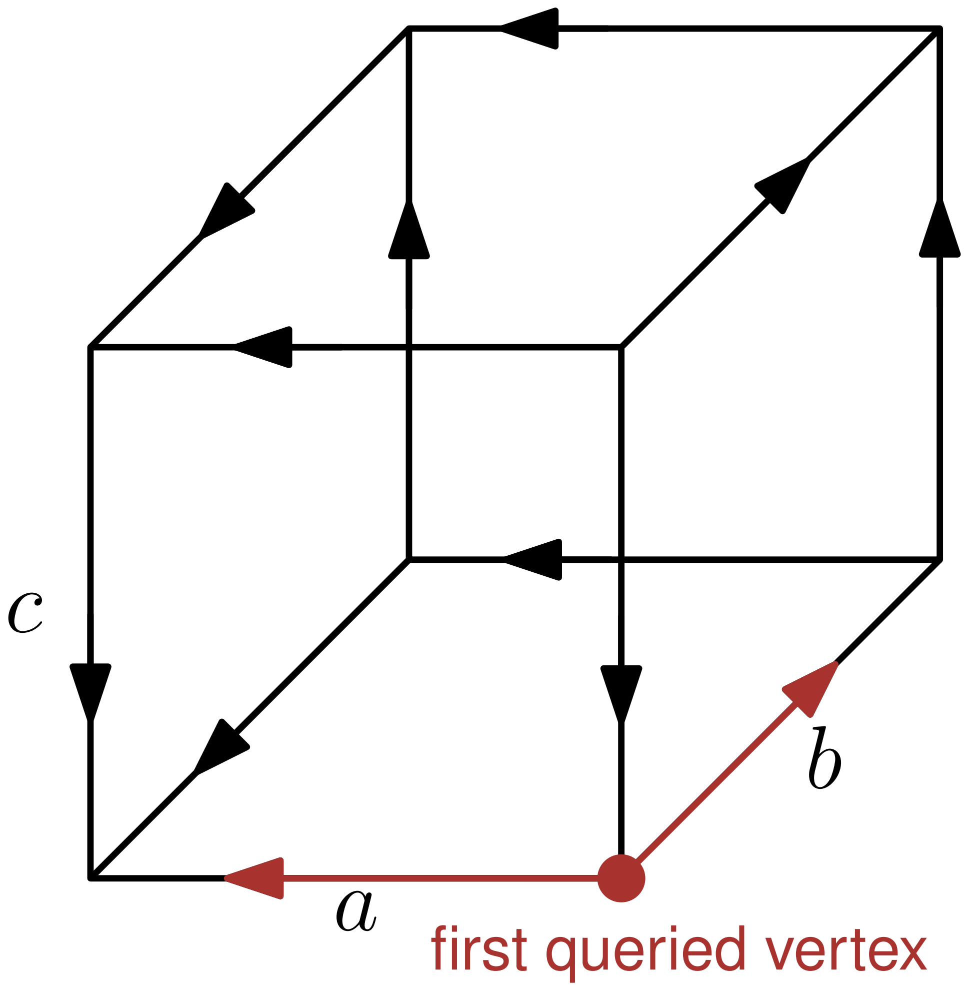
General Matoušek-type USOs: Upper Bound



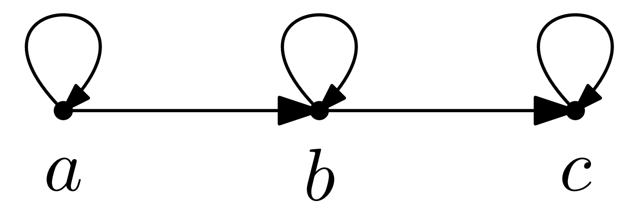
Alternative Views



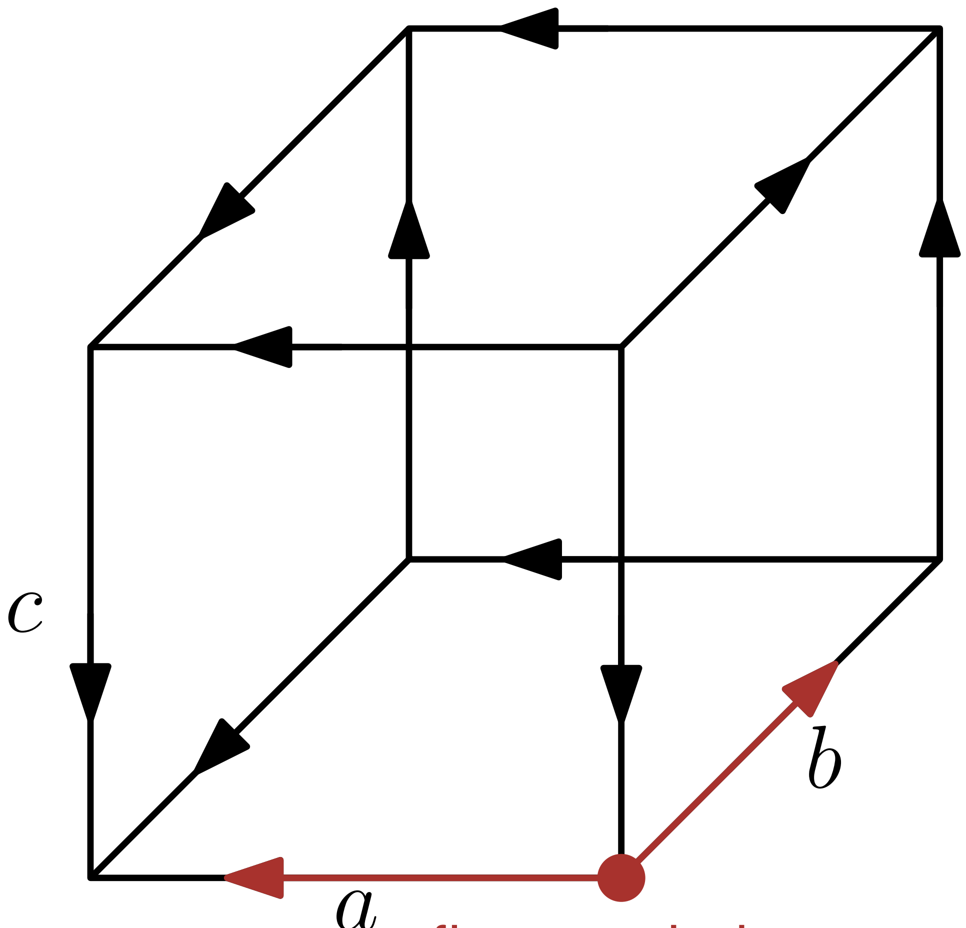
Alternative Views



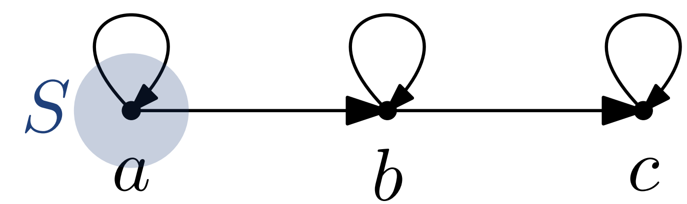
Find set of dimensions S such that a and b have odd numbers of in-neighbors in S, and c has an even number



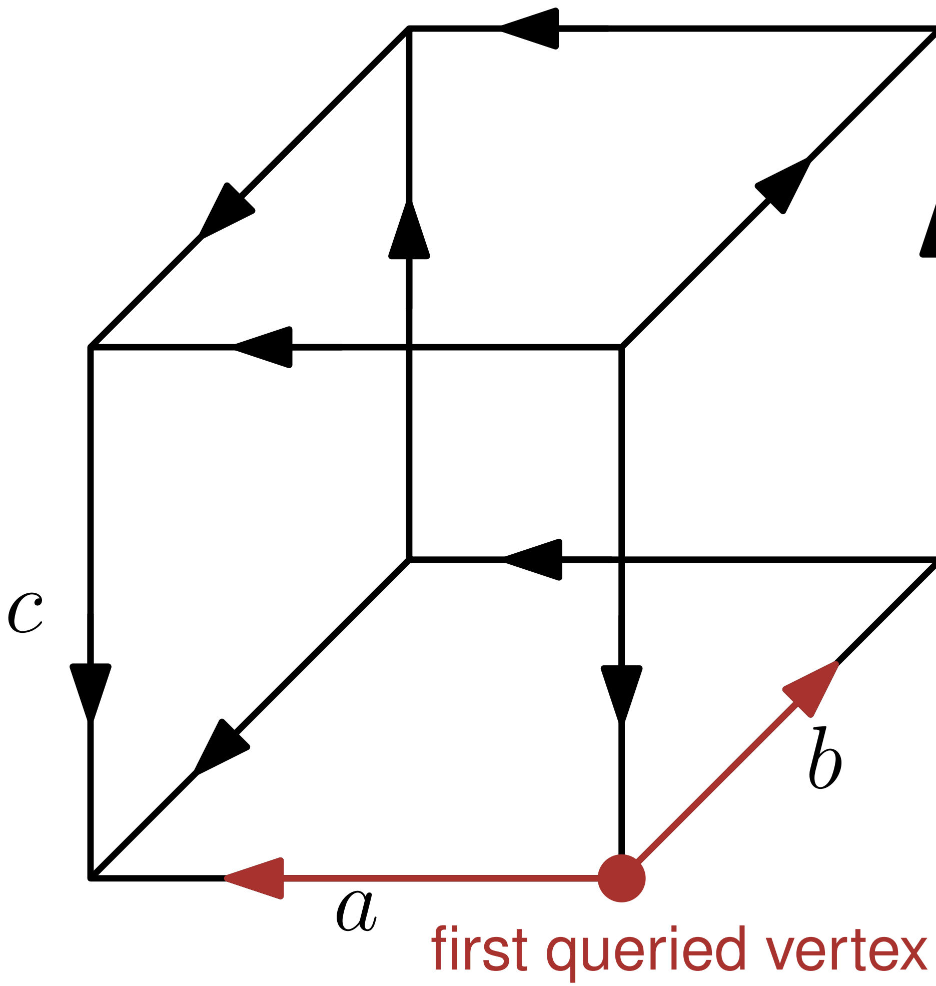
Alternative Views



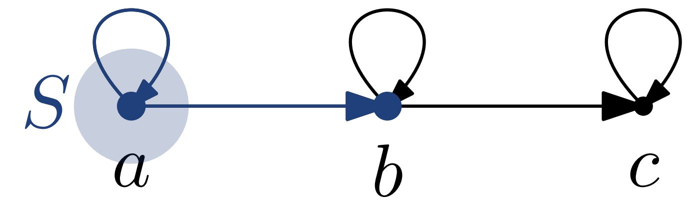
Find set of dimensions S such that a and b have odd numbers of in-neighbors in S, and c has an even number



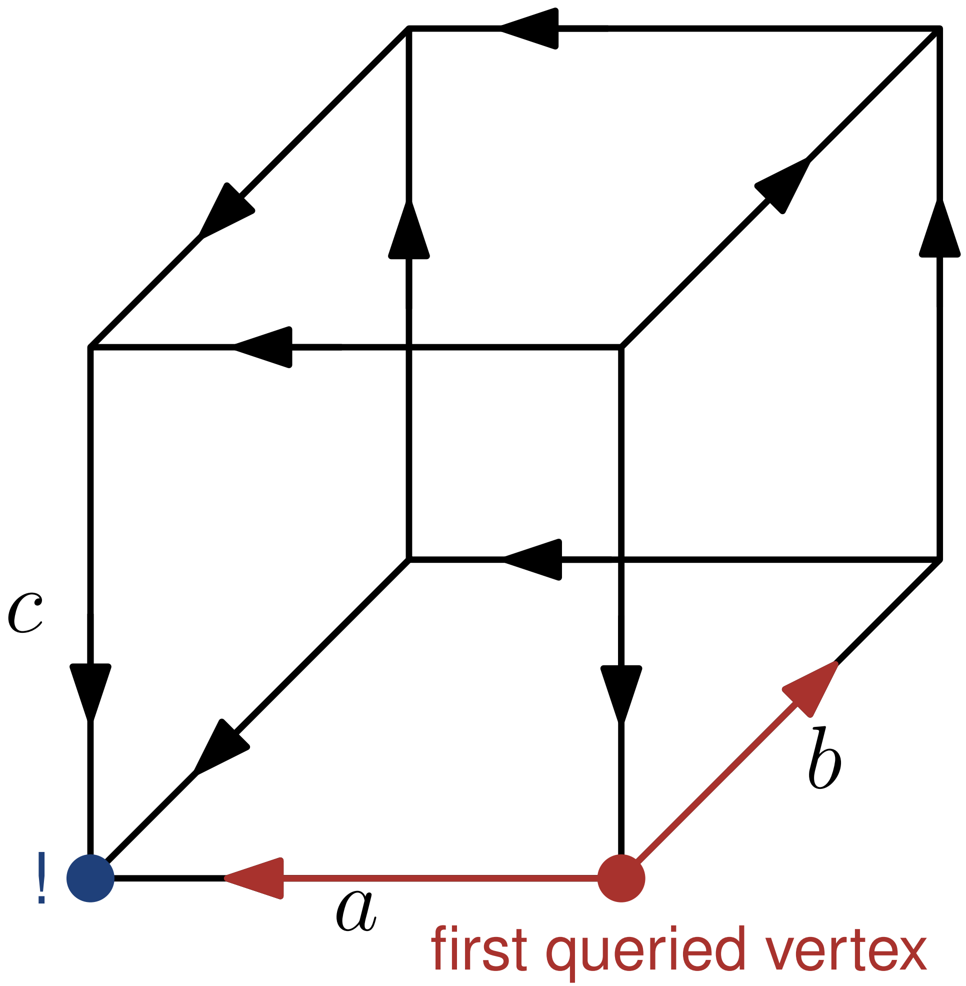
Alternative Views



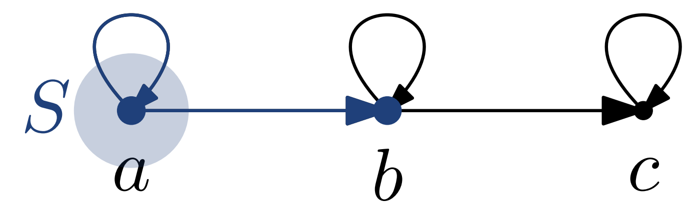
Find set of dimensions S such that a and b have odd numbers of in-neighbors in S, and c has an even number



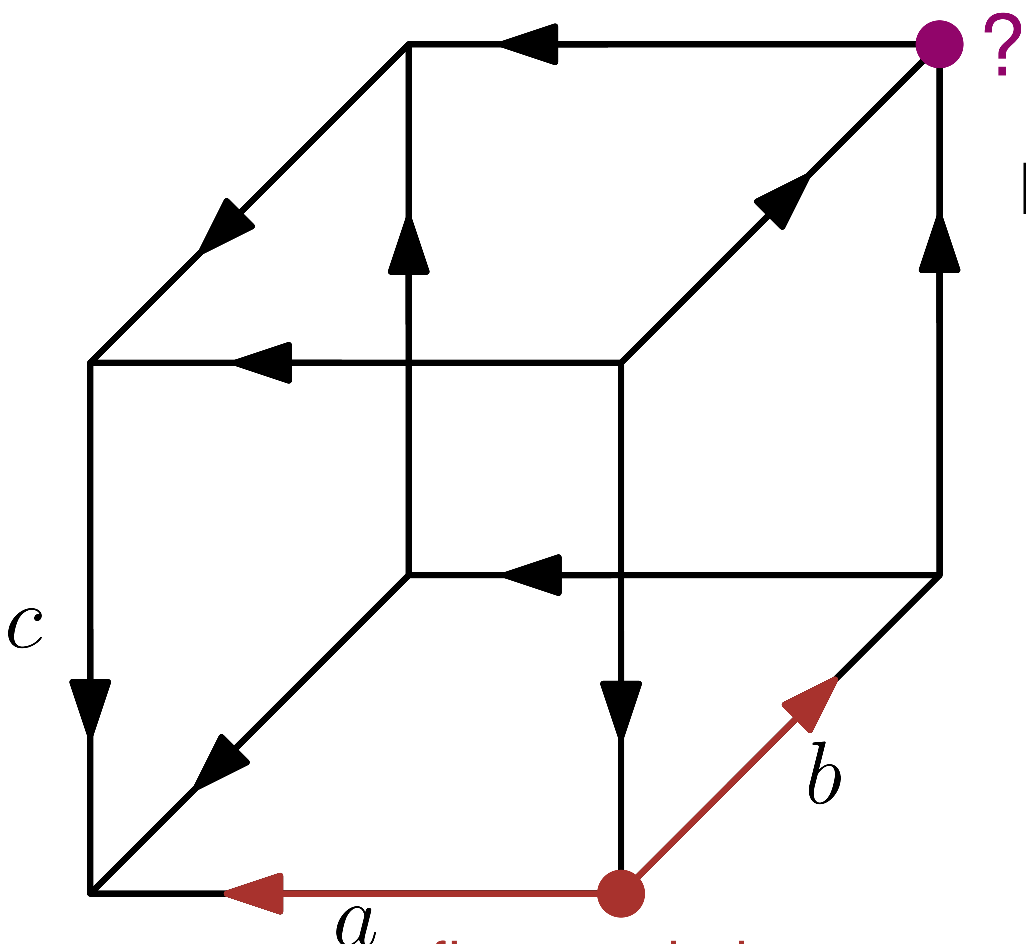
Alternative Views



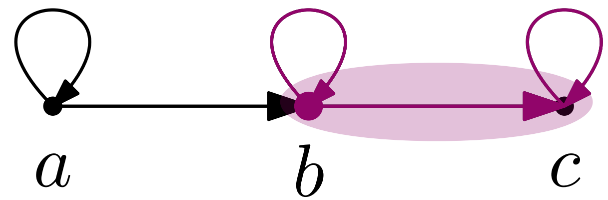
Find set of dimensions S such that a and b have odd numbers of in-neighbors in S, and c has an even number



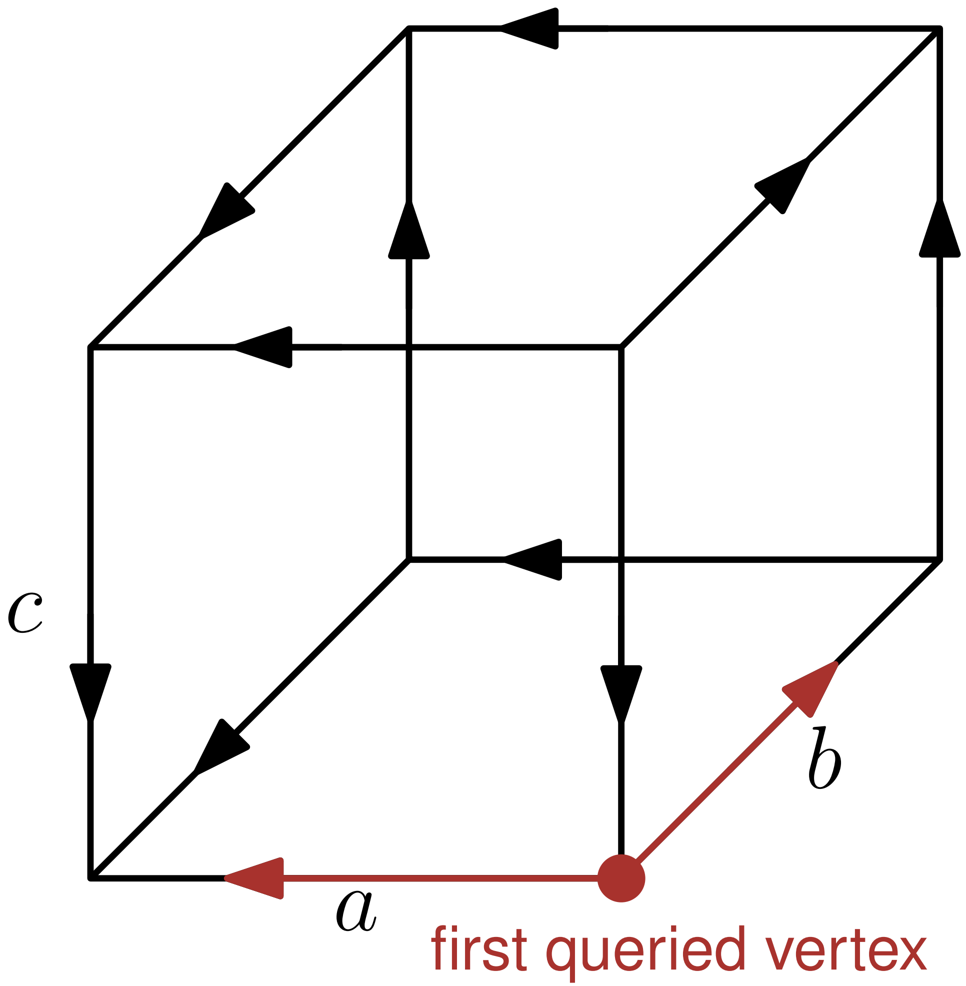
Alternative Views



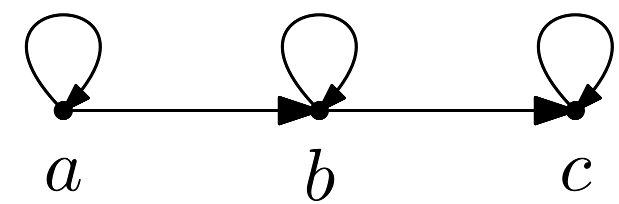
Find set of dimensions S such that a and b have odd numbers of in-neighbors in S, and c has an even number



Alternative Views



Find set of dimensions S such that a and b have odd numbers of in-neighbors in S, and c has an even number



$$Mx = y$$

General Matoušek-type USOs: Lower Bound

Adversarial, adaptive oracle:

General Matoušek-type USOs: Lower Bound

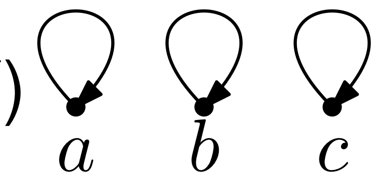
Adversarial, adaptive oracle:

- The first queried vertex is never a sink ($y \neq 0$)

General Matoušek-type USOs: Lower Bound

Adversarial, adaptive oracle:

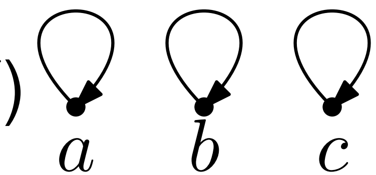
- The first queried vertex is never a sink ($y \neq 0$)
- Set dimension influence graph G to be "empty" ($M = I$)



General Matoušek-type USOs: Lower Bound

Adversarial, adaptive oracle:

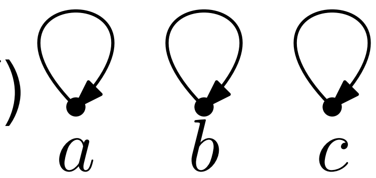
- The first queried vertex is never a sink ($y \neq 0$)
- Set dimension influence graph G to be "empty" ($M = I$)
- On every query:
  Would replying according to the current dimension influence graph let the algorithm figure out the sink?



General Matoušek-type USOs: Lower Bound

Adversarial, adaptive oracle:

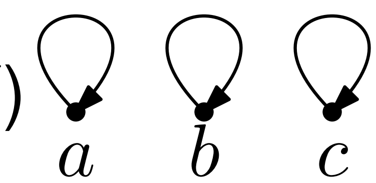
- The first queried vertex is never a sink ($y \neq 0$)
- Set dimension influence graph G to be "empty" ($M = I$)
- On every query:
  Would replying according to the current dimension influence graph let the algorithm figure out the sink?
  - No? Answer according to G.



General Matoušek-type USOs: Lower Bound

Adversarial, adaptive oracle:

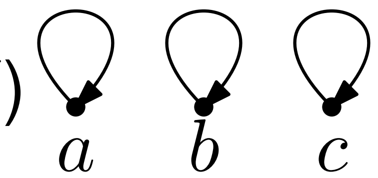
- The first queried vertex is never a sink ($y \neq 0$)
- Set dimension influence graph G to be “empty” ($M = I$)
- On every query:
  Would replying according to the current dimension influence graph let the algorithm figure out the sink?
  - No? Answer according to G.
  - Yes? Change G.



General Matoušek-type USOs: Lower Bound

Adversarial, adaptive oracle:

- The first queried vertex is never a sink ($y \neq 0$)
- Set dimension influence graph G to be "empty" ($M = I$)
- On every query:
  Would replying according to the current dimension influence graph let the algorithm figure out the sink?
  - No? Answer according to G.
  - Yes? Change G.

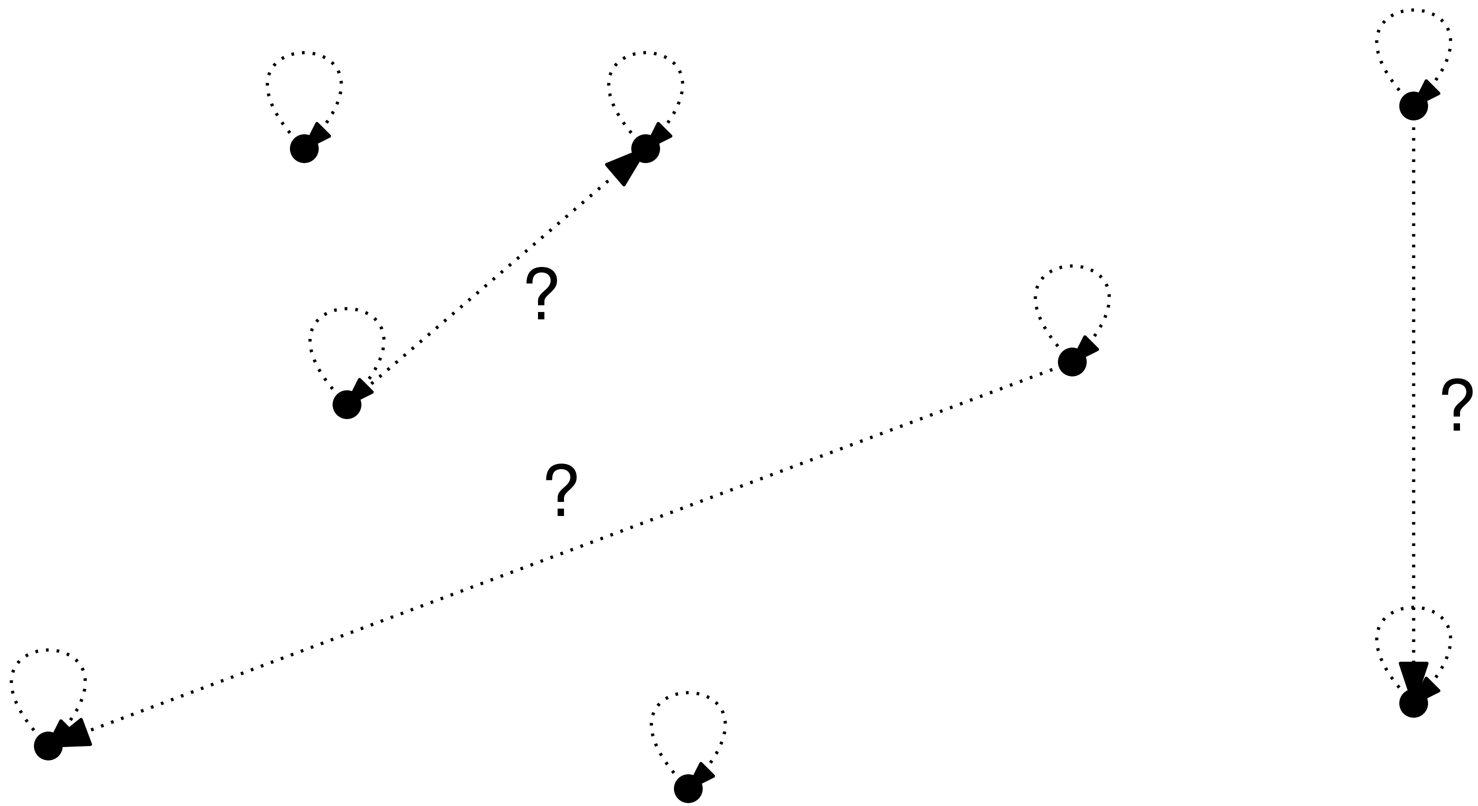


Need *consistency*, *legality*, and *uncertainty*.

Realizable Matoušek-type USOs: Upper Bound

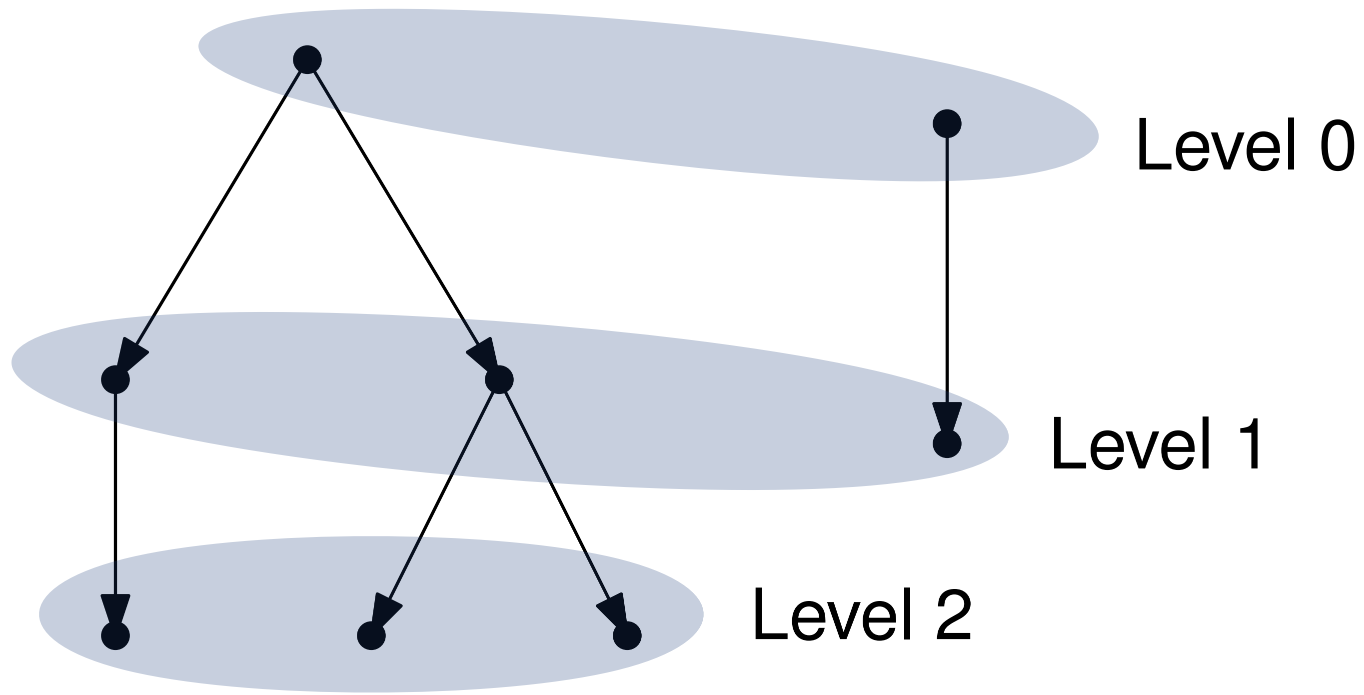
We recover the whole dimension influence graph!

Realizable Matoušek-type USOs: Upper Bound

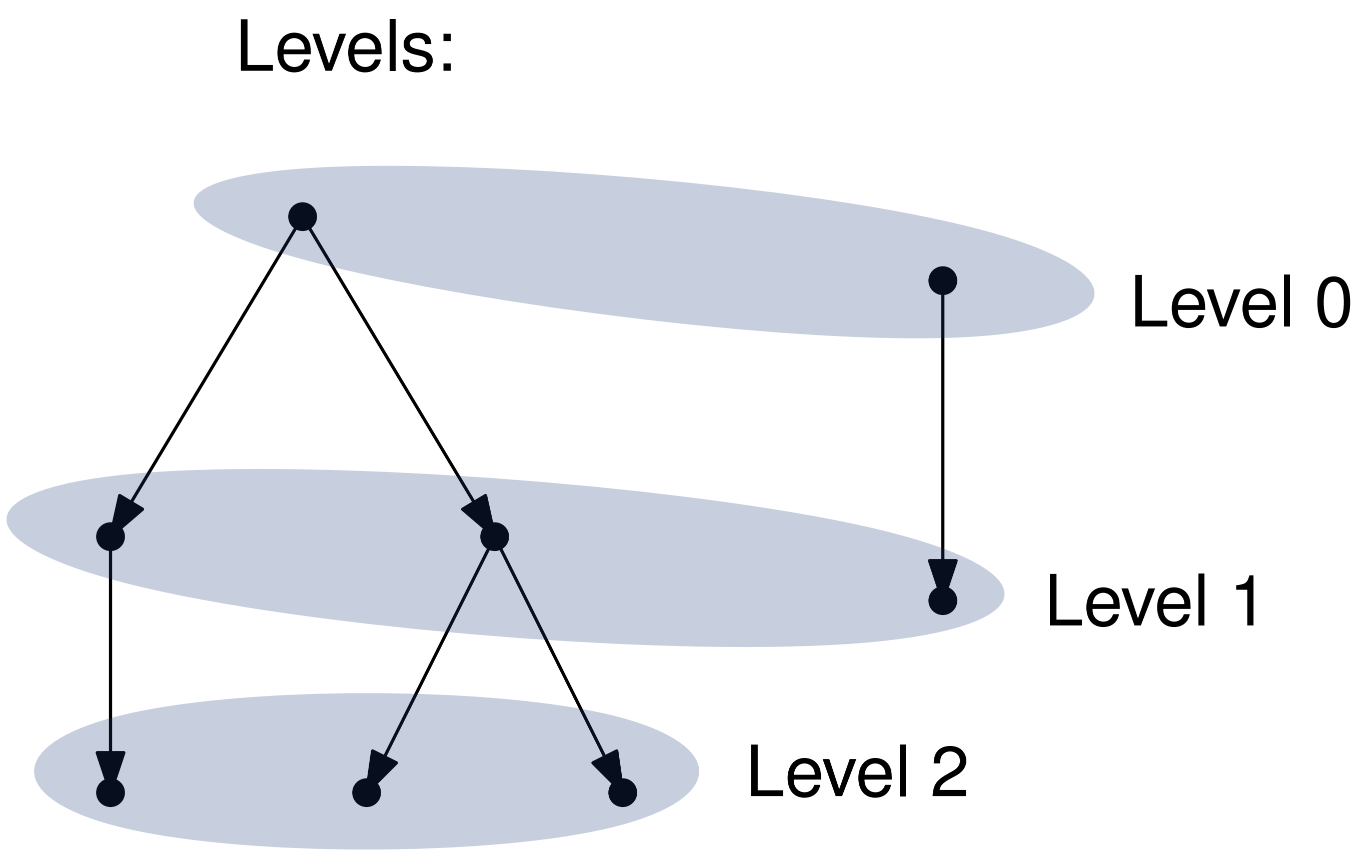


Realizable Matoušek-type USOs: Upper Bound

Levels:



Realizable Matoušek-type USOs: Upper Bound



Level $=$ in-degree -1

Realizable Matoušek-type USOs: Upper Bound

Levelling: $O(\log n)$

- First query $\{1, \ldots, n\}$: $\Rightarrow$ set of dimensions with odd in-degree

Realizable Matoušek-type USOs: Upper Bound

Levelling: $O(\log n)$

- First query $\{1, \ldots, n\}$: $\Rightarrow$ set of dimensions with odd in-degree
- Query set of dimensions with odd in-degree: $\Rightarrow$ set of dimensions with *odd in-degree among those with odd in-degree*

Realizable Matoušek-type USOs: Upper Bound

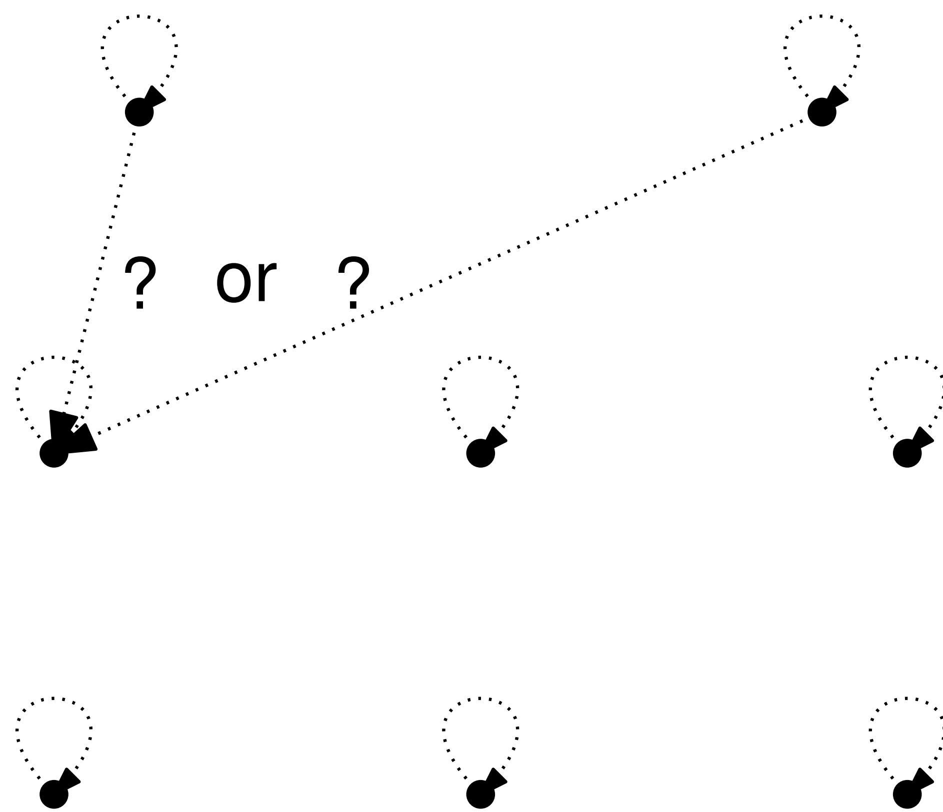
Levelling: $O(\log n)$

- First query $\{1, \ldots, n\}$: $\Rightarrow$ set of dimensions with odd in-degree
- Query set of dimensions with odd in-degree: $\Rightarrow$ set of dimensions with *odd in-degree among those with odd in-degree*

This reveals something about the second-LSB of their in-degree

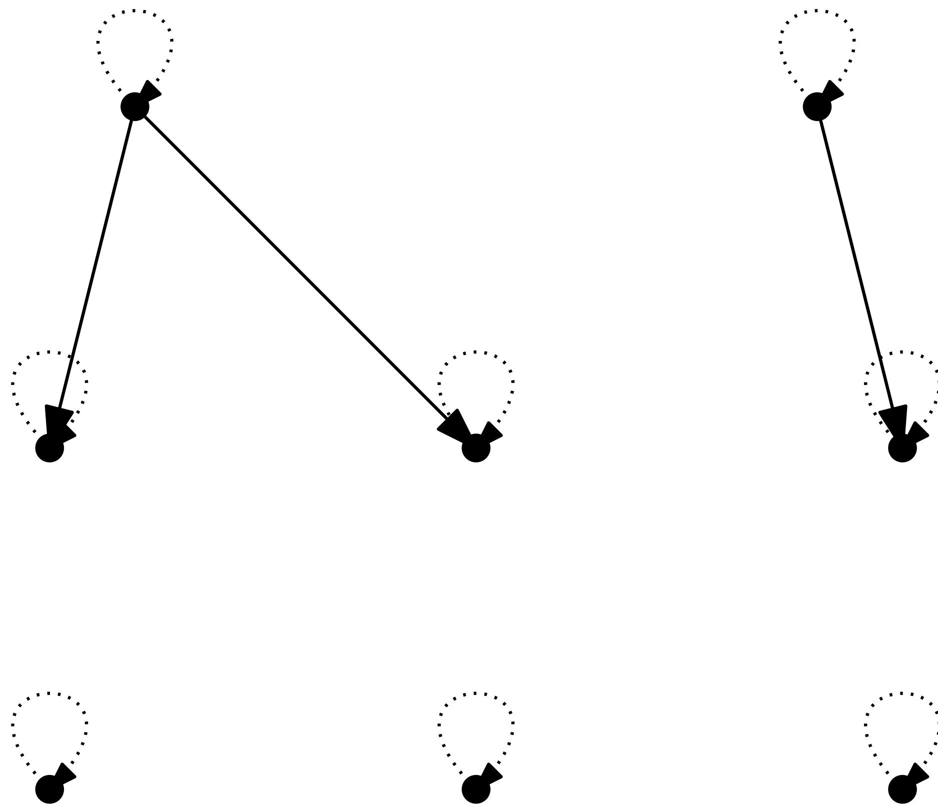
Realizable Matoušek-type USOs: Upper Bound

Levelling: $O(\log n)$



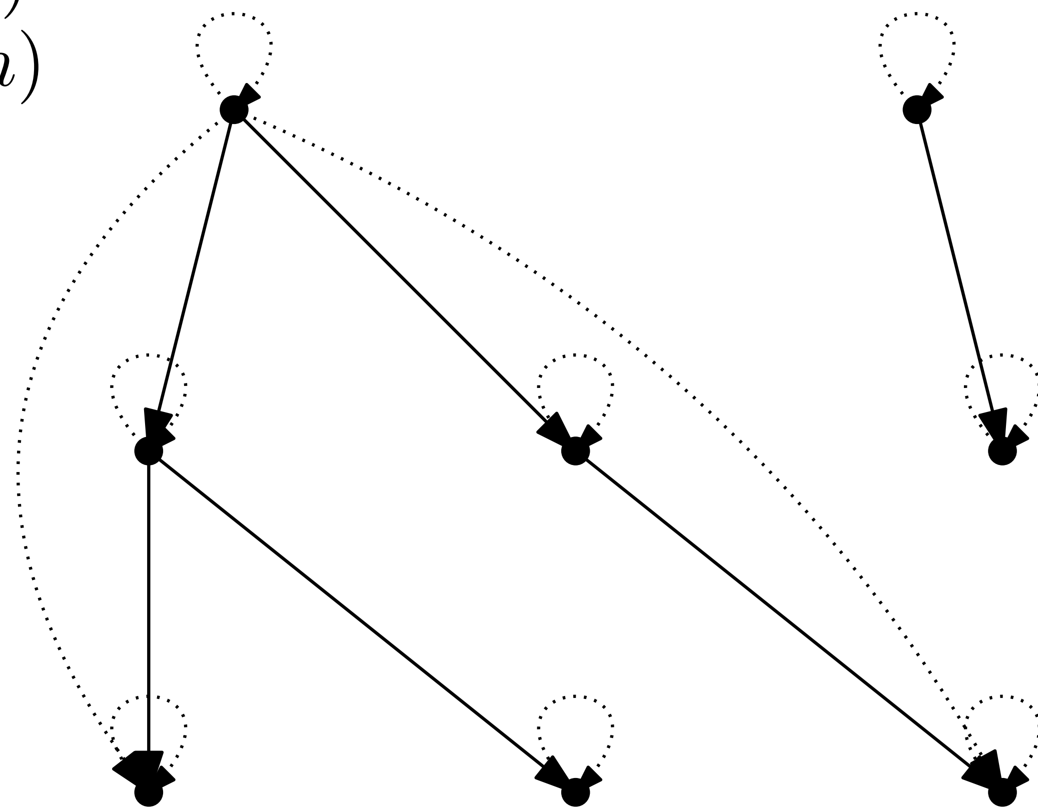
Realizable Matoušek-type USOs: Upper Bound

Levelling: $O(\log n)$
One level: $O(\log n)$



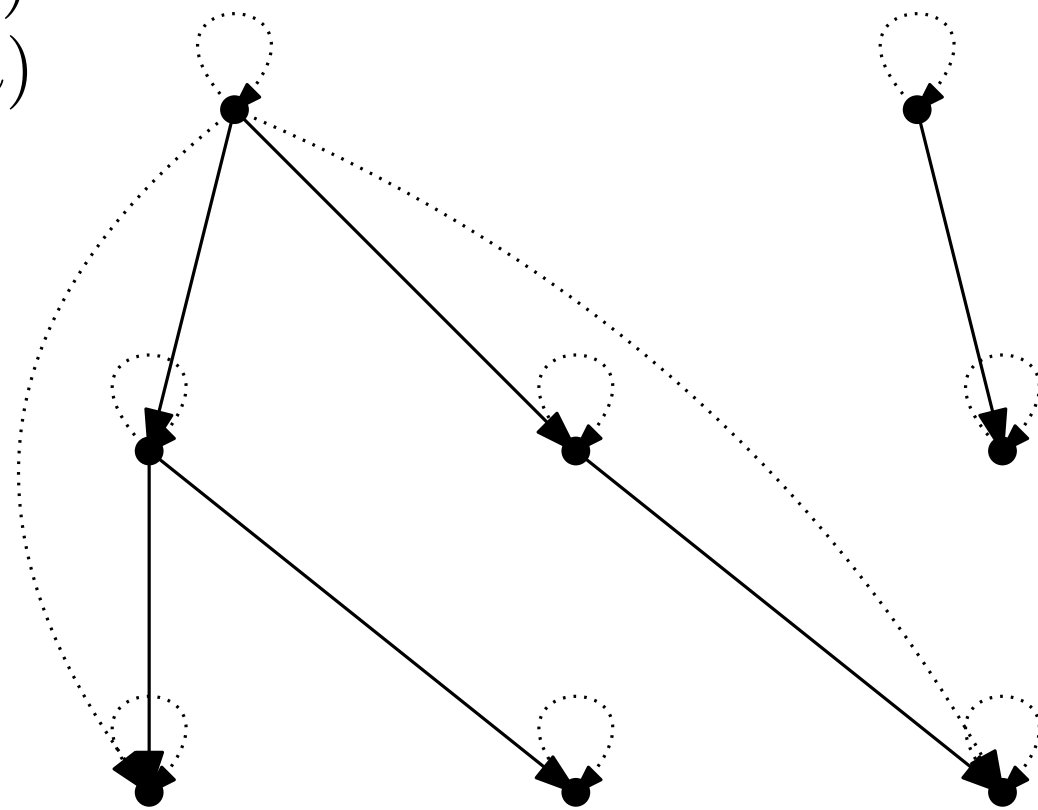
Realizable Matoušek-type USOs: Upper Bound

Levelling: $O(\log n)$
One level: $O(\log n)$
All levels: $O(n \log n)$



Realizable Matoušek-type USOs: Upper Bound

Levelling: $O(\log n)$
One level: $O(\log n)$
All levels: $O(\log^2 n)$



Open Questions

Can we find a complexity gap or algorithms making use of realizability for larger (more relevant) classes of USOs?

Open Questions

Can we find a complexity gap or algorithms making use of realizability for larger (more relevant) classes of USOs?

What is the true query complexity of sink-finding on realizable Matoušek-type USOs?

Open Questions

Can we find a complexity gap or algorithms making use of realizability for larger (more relevant) classes of USOs?

What is the true query complexity of sink-finding on realizable Matoušek-type USOs?

Does the complexity gap also hold for randomized algorithms?