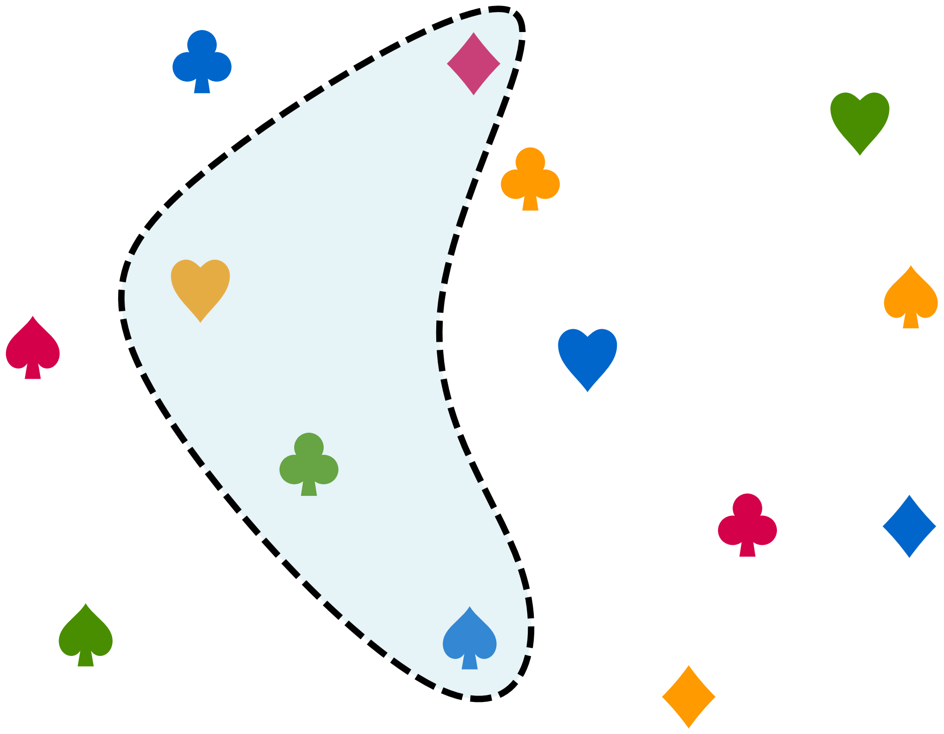
Sublinear-Round Parallel Matroid Intersection

Joakim Blikstad

KTH Royal Institute of Technology

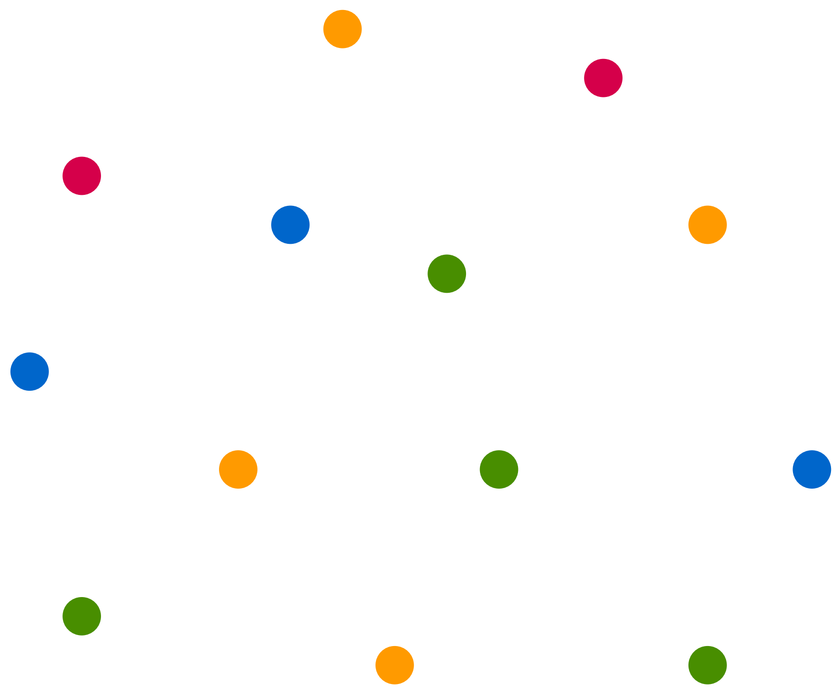
ICALP 2022



Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

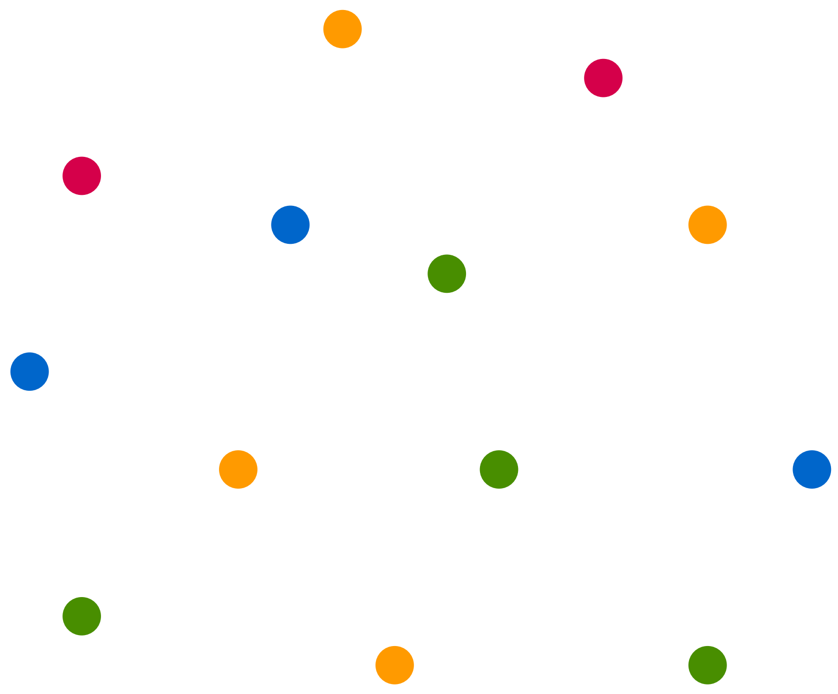
1. Ground set V of n elements



Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

1. Ground set V of n elements
2. Notion of independence $\mathcal{I}$

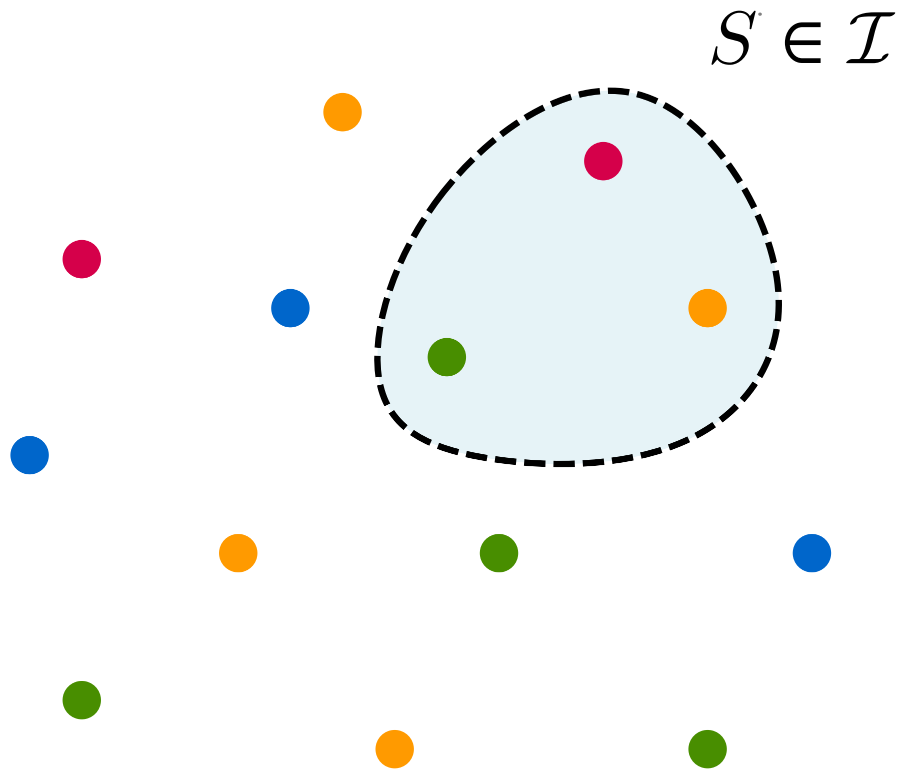


Eg. Colourful Matroid
"no duplicate colours"

Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

1. Ground set V of n elements
2. Notion of independence $\mathcal{I}$

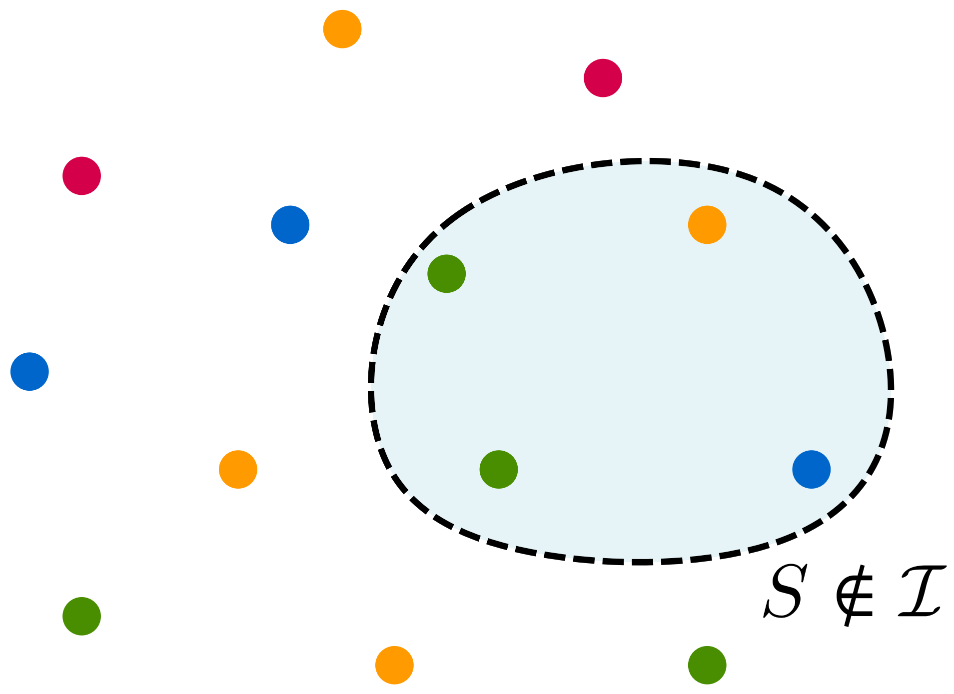


Eg. Colourful Matroid
"no duplicate colours"

Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

1. Ground set V of n elements
2. Notion of independence $\mathcal{I}$

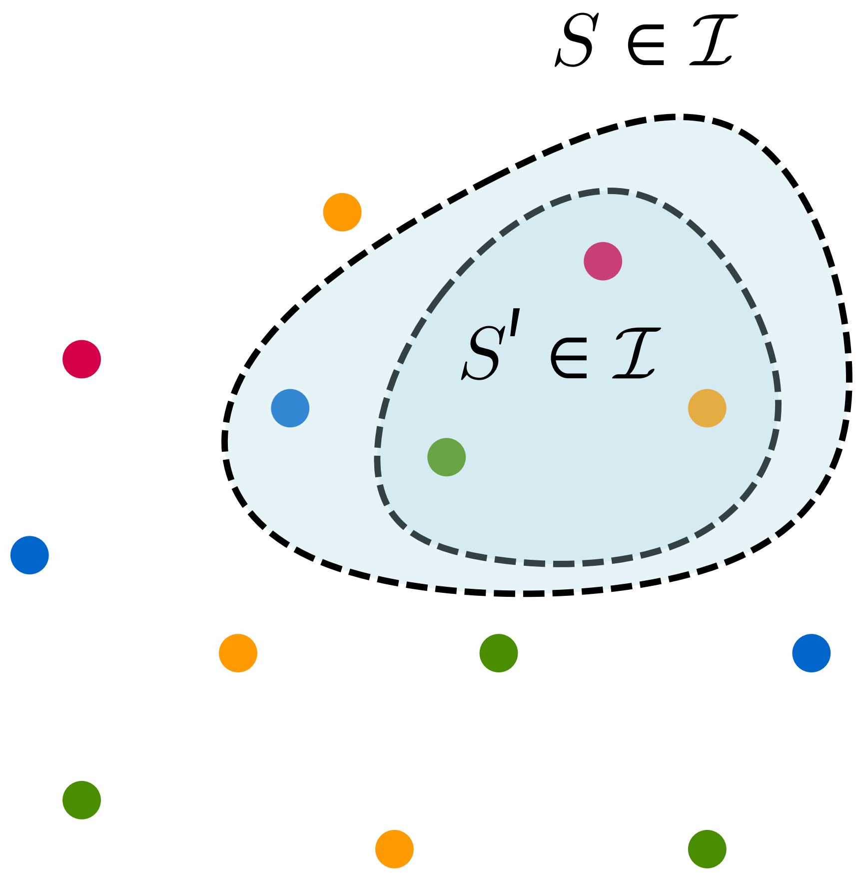


Eg. Colourful Matroid
“no duplicate colours”

Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

1. Ground set V of n elements
2. Notion of independence $\mathcal{I}$
   - Downward closure



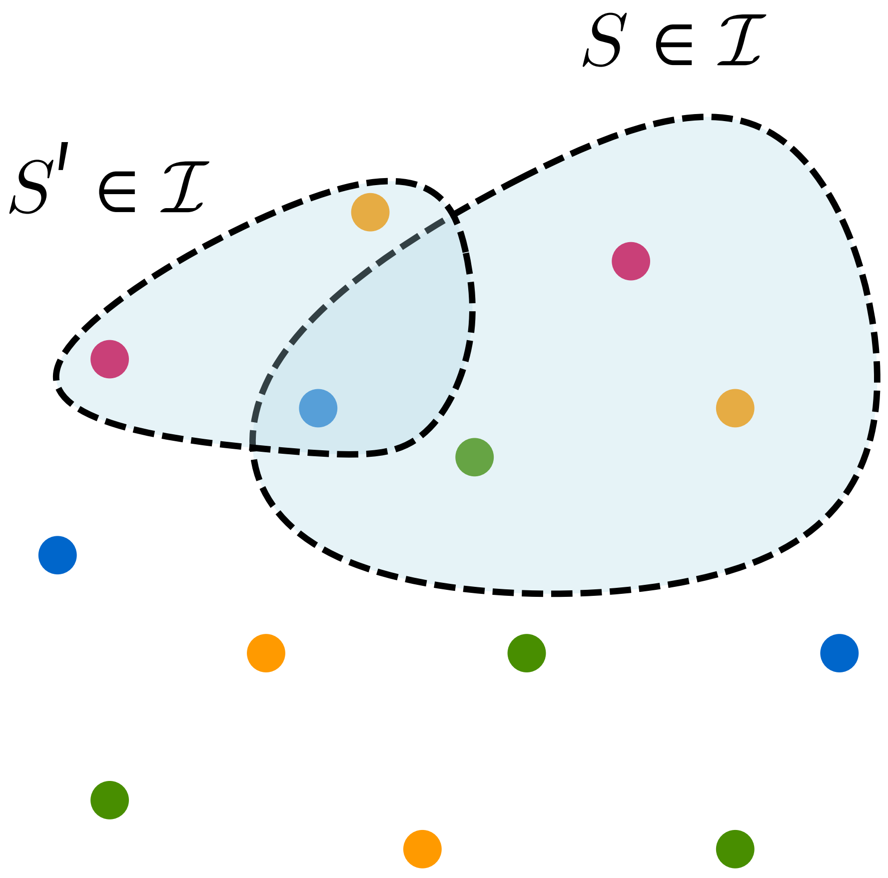
Eg. Colourful Matroid
"no duplicate colours"

Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

1. Ground set V of n elements
2. Notion of independence $\mathcal{I}$
   - Downward closure
   - Exchange property

   "All maximal independent sets have the same size"



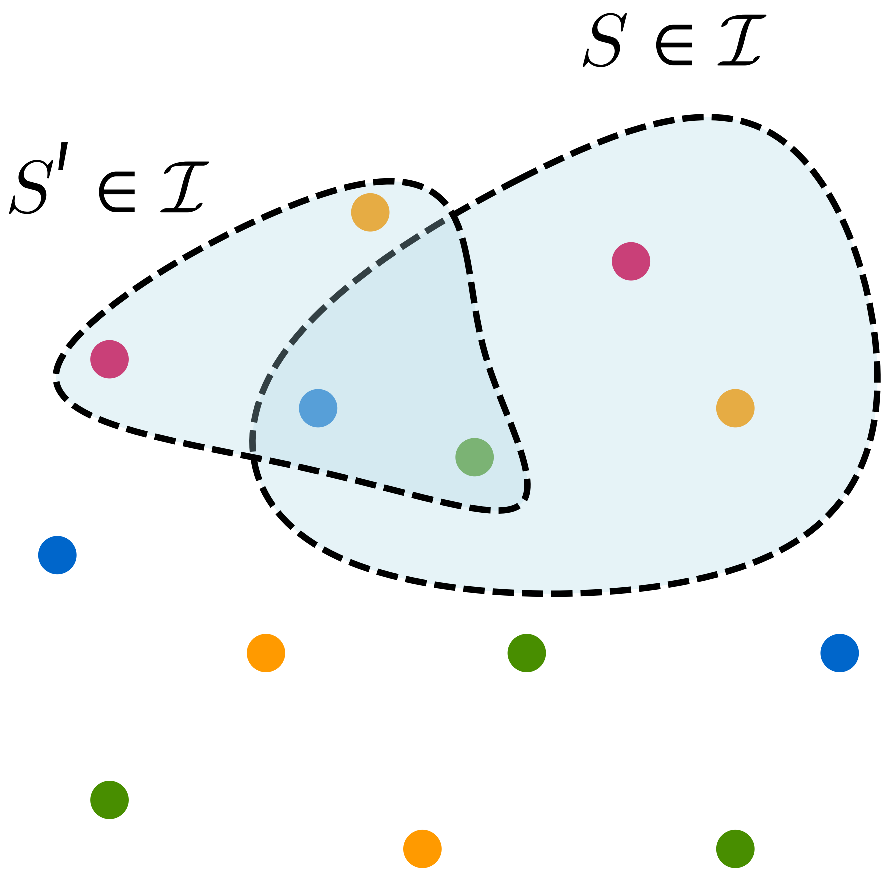
Eg. Colourful Matroid
"no duplicate colours"

Matroids

Matroid $\mathcal{M} = (V, \mathcal{I})$

1. Ground set V of n elements
2. Notion of independence $\mathcal{I}$
   - Downward closure
   - Exchange property

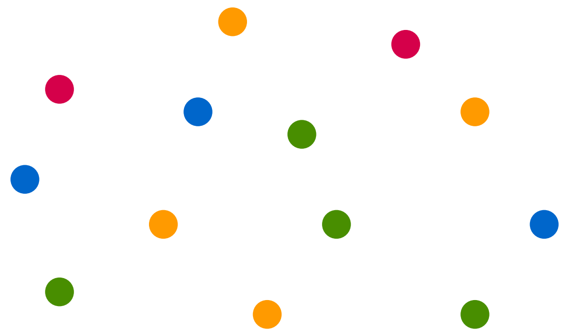
   "All maximal independent sets have the same size"



Eg. Colourful Matroid
"no duplicate colours"

Matroids: Examples

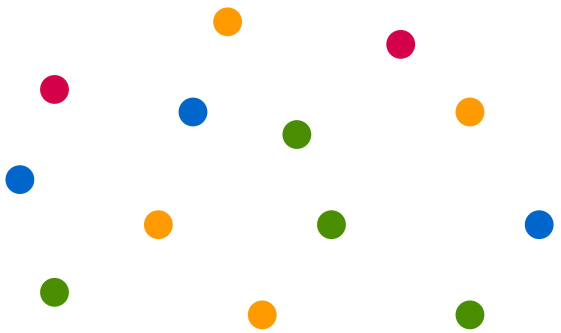
Colourful Matroid



$\mathcal{I}$ =“no duplicate colours”

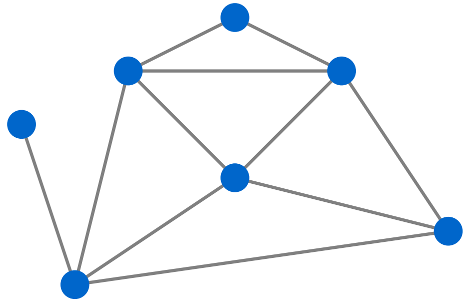
Matroids: Examples

Colourful Matroid



$\mathcal{I}$ =“no duplicate colours”

Graphic Matroid

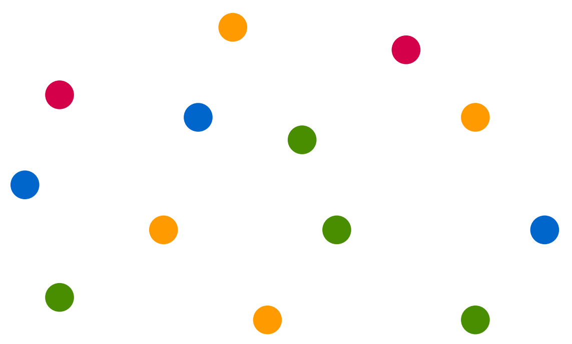


V = edges

$\mathcal{I}$ =“no cycles”

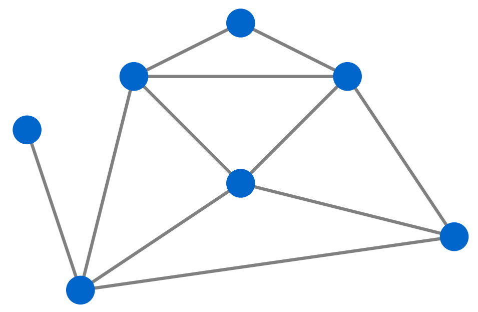
Matroids: Examples

Colourful Matroid



$\mathcal{I}$ ="no duplicate colours"

Graphic Matroid



V = edges

$\mathcal{I}$ ="no cycles"

Linear Matroid

$(2, 1, 4, 2, 3, 3)$

$(1, 0, 1, 0, 1, 0)$

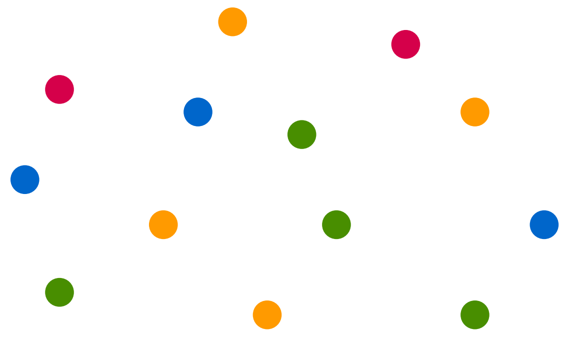
$(3, 1, 5, 2, 4, 3)$

V = vectors

$\mathcal{I}$ ="linear independence"

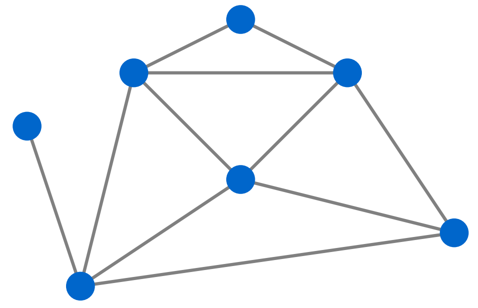
Matroids: Examples

Colourful Matroid



$\mathcal{I}$ ="no duplicate colours"

Graphic Matroid



V = edges

$\mathcal{I}$ ="no cycles"

Linear Matroid

$(2, 1, 4, 2, 3, 3)$

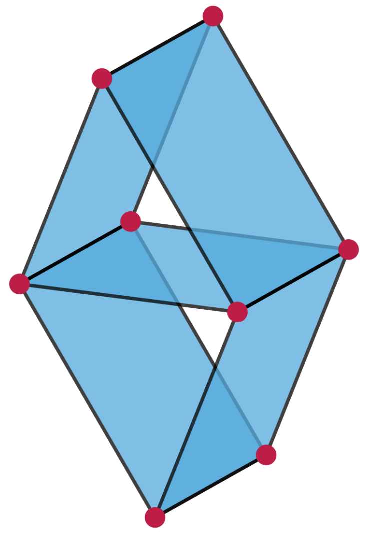
$(1, 0, 1, 0, 1, 0)$

$(3, 1, 5, 2, 4, 3)$

V = vectors

$\mathcal{I}$ ="linear independence"

Vámos Matroid



Matroid Intersection

Given two matroids:

- $\mathcal{M}_1 = (V, \mathcal{I}_1)$
- $\mathcal{M}_2 = (V, \mathcal{I}_2)$

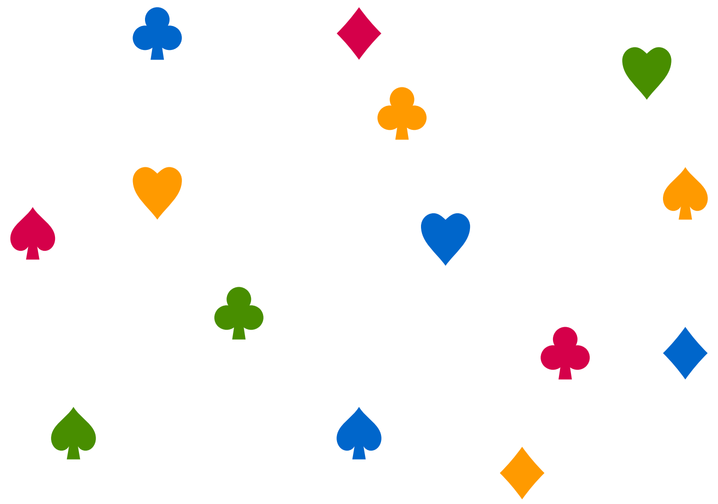
Find a *common independent set* $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ of maximum size.

Matroid Intersection

Given two matroids:

- $\mathcal{M}_1 = (V, \mathcal{I}_1)$
- $\mathcal{M}_2 = (V, \mathcal{I}_2)$

Find a *common independent set* $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ of maximum size.



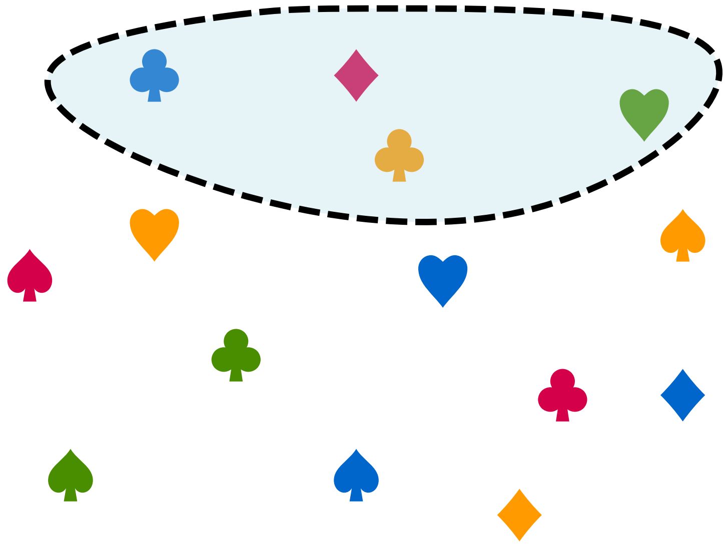
$\mathcal{M}_1 =$ "distinct suits"
$\mathcal{M}_2 =$ "distinct colours"

Matroid Intersection

Given two matroids:

- $\mathcal{M}_1 = (V, \mathcal{I}_1)$
- $\mathcal{M}_2 = (V, \mathcal{I}_2)$

Find a *common independent set* $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ of maximum size.



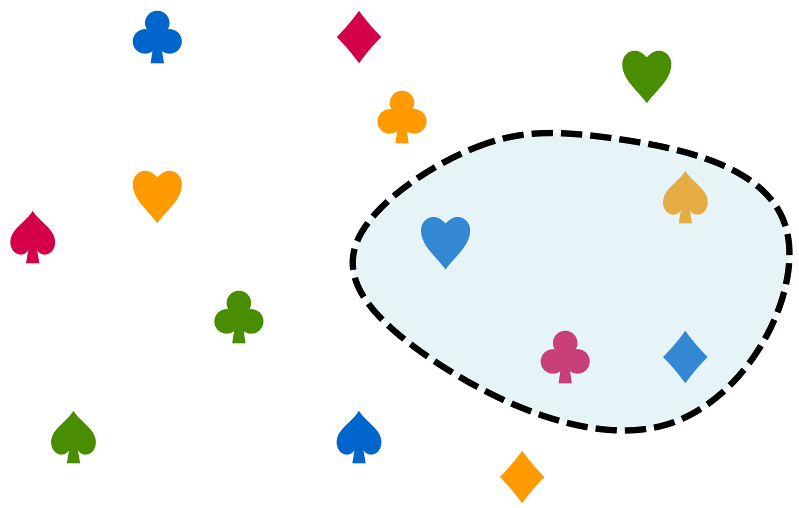
$\mathcal{M}_1 =$ "distinct suits"
$\mathcal{M}_2 =$ "distinct colours"

Matroid Intersection

Given two matroids:

- $\mathcal{M}_1 = (V, \mathcal{I}_1)$
- $\mathcal{M}_2 = (V, \mathcal{I}_2)$

Find a *common independent set* $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ of maximum size.



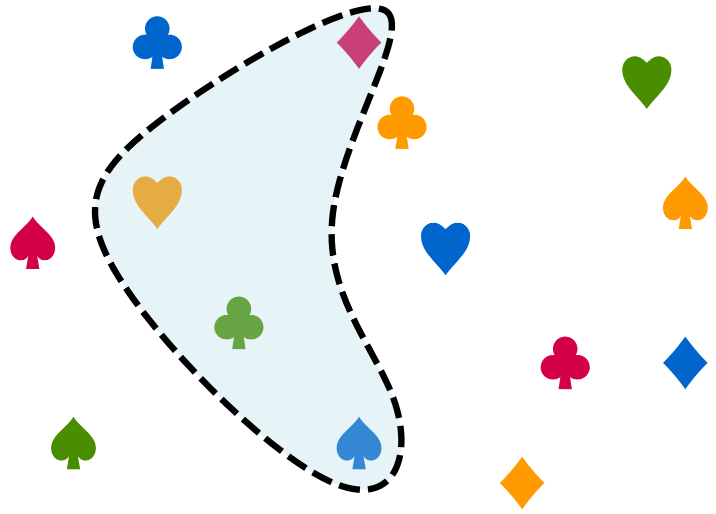
$\mathcal{M}_1 =$ "distinct suits"
$\mathcal{M}_2 =$ "distinct colours"

Matroid Intersection

Given two matroids:

- $\mathcal{M}_1 = (V, \mathcal{I}_1)$
- $\mathcal{M}_2 = (V, \mathcal{I}_2)$

Find a *common independent set* $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ of maximum size.



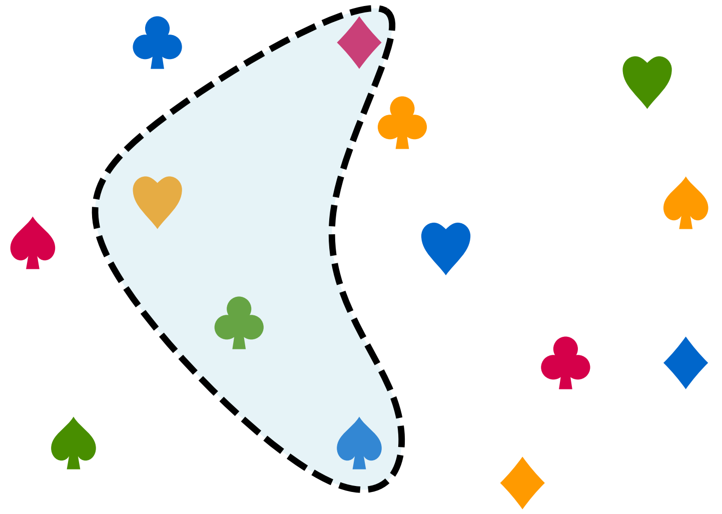
$\mathcal{M}_1$ = "distinct suits"
$\mathcal{M}_2$ = "distinct colours"

Matroid Intersection

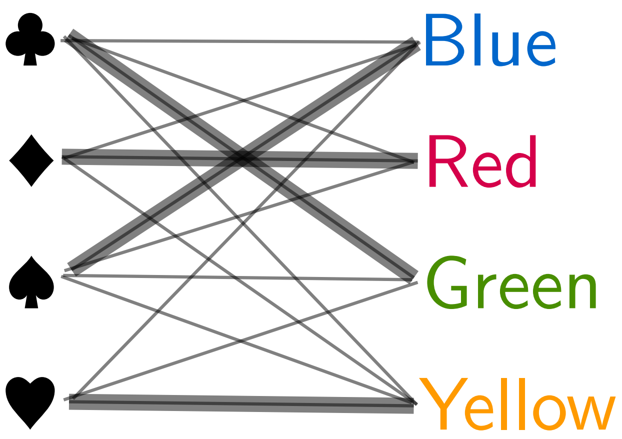
Given two matroids:

- $\mathcal{M}_1 = (V, \mathcal{I}_1)$
- $\mathcal{M}_2 = (V, \mathcal{I}_2)$

Find a *common independent set* $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ of maximum size.



$\mathcal{M}_1 =$ "distinct suits"
$\mathcal{M}_2 =$ "distinct colours"



Matroid Intersection: Examples

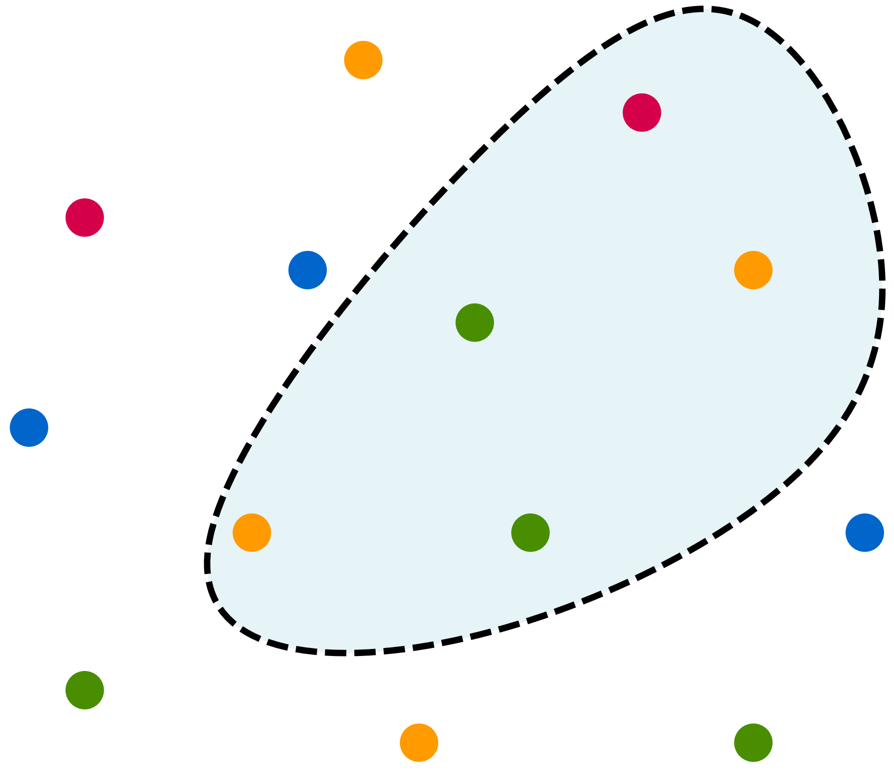
- Bipartite matching
- Arborescence (directed spanning tree)
- Rainbow spanning trees
- Tree/Arborescence packing
- Directed min-cut
- Graph orientation problems
- Matroid partitioning & union
- ...

Also connections to *Submodular Function Minimization*

Matroid Rank

$$\begin{aligned}\mathrm{rk}(S) &= \max\{|A| : A \subseteq S, A \in \mathcal{I}\}\\ &= \text{size of a maximum independent set in } S\\ &= \text{size of a \textit{maximal} independent set in } S\end{aligned}$$

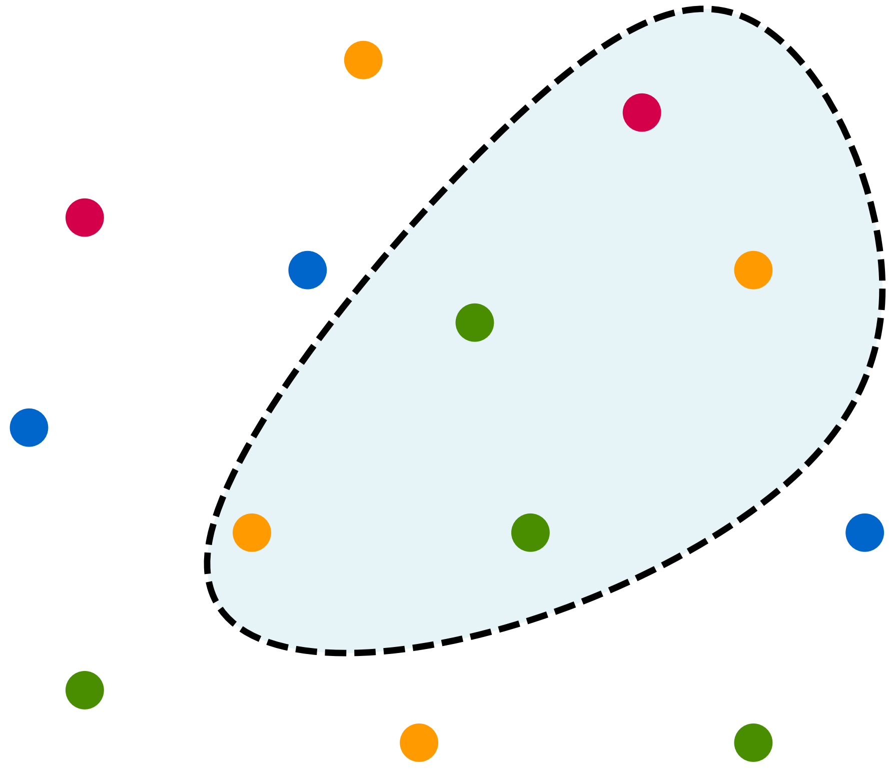
$\mathrm{rk}(S) = 3 =$ #distinct colours



Matroid Rank

$$\begin{aligned}\mathrm{rk}(S) &= \max\{|A| : A \subseteq S, A \in \mathcal{I}\}\\ &= \text{size of a maximum independent set in } S\\ &= \text{size of a \textit{maximal} independent set in } S\end{aligned}$$

$\mathrm{rk}(S) = 3 = \#$distinct colours



Properties:

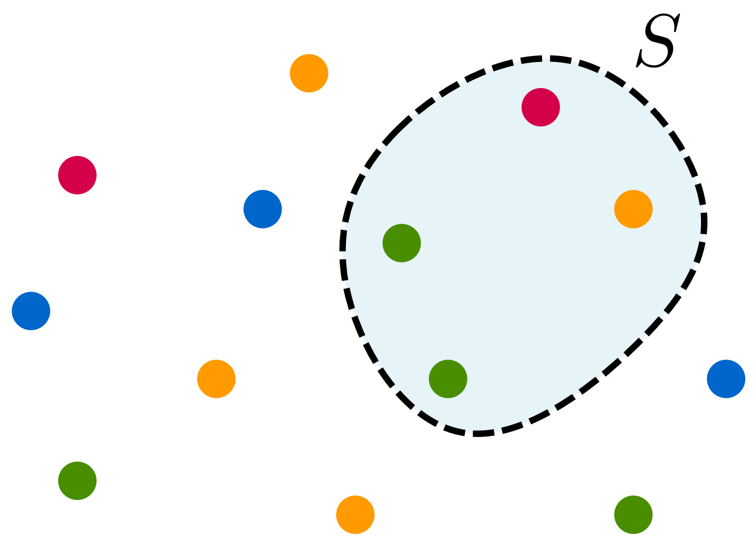
- $S \in \mathcal{I} \iff \mathrm{rk}(S) = |S|$
- Submodular (Diminishing returns)
  If $A \subseteq B$, and $x \notin B$ then:
  $\mathrm{rk}(A+x) - \mathrm{rk}(A) \geq \mathrm{rk}(B+x) - \mathrm{rk}(B)$

Query Access

How to access a matroid?

Oracle Access

- Independence query: "Is $S \in \mathcal{I}$?"
- Rank query: "What is $\mathrm{rk}(S)$?"

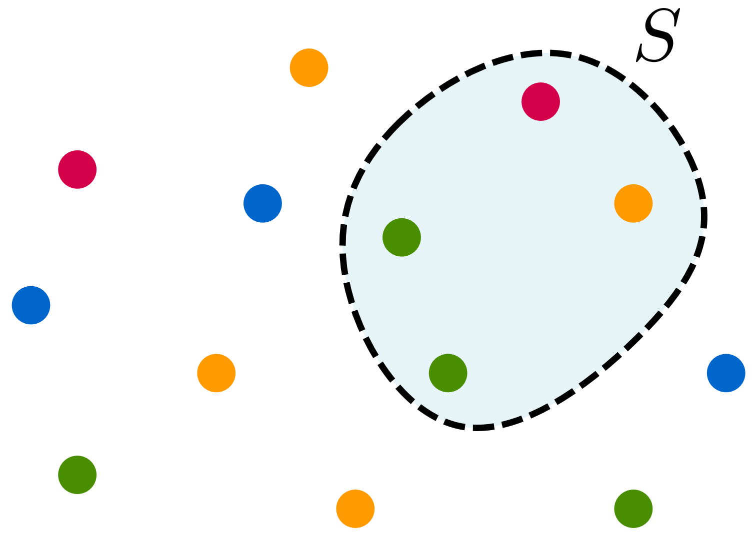


Query Access

How to access a matroid?

Oracle Access

- Independence query: "Is $S \in \mathcal{I}$?" **"NO"**
- Rank query: "What is $\mathrm{rk}(S)$?" **"3"**

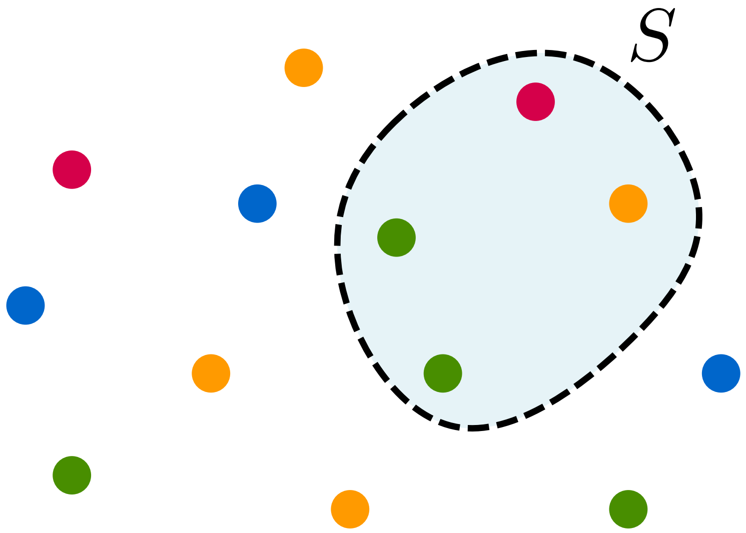


Query Access

How to access a matroid?

Oracle Access

- Independence query: "Is $S \in \mathcal{I}$?" **"NO"**
- Rank query: "What is $\mathrm{rk}(S)$?" **"3"**



Important:
We do not know the underlying structure of the matroids!

Parallel Query Algorithms

Runs in rounds

Round: Issue a set of k queries simultaneously:

$$\text{“What is } \mathrm{rk}(S_1)?\text{”}, \ \text{“What is } \mathrm{rk}(S_2)?\text{”}, \ldots, \text{“What is } \mathrm{rk}(S_k)?\text{”}$$

Can only depend on answers to queries in previous rounds!

Parallel Query Algorithms

Runs in rounds

Round: Issue a set of k queries simultaneously:

$$\text{“What is } \mathrm{rk}(S_1)\text{?”}, \ \text{“What is } \mathrm{rk}(S_2)\text{?”}, \ \ldots, \ \text{“What is } \mathrm{rk}(S_k)\text{?”}$$

Can only depend on answers to queries in previous rounds!

Tradeoff:

- Total number of queries used
- Number of rounds (*adaptivity*)

Parallel Query Algorithms

Runs in rounds

Round: Issue a set of k queries simultaneously:

“What is $\mathrm{rk}(S_1)$?”, “What is $\mathrm{rk}(S_2)$?”, ..., “What is $\mathrm{rk}(S_k)$?”

Can only depend on answers to queries in previous rounds!

Tradeoff:

- Total number of queries used $O(n^{3/2})$
- Number of rounds (*adaptivity*) $O(n^{3/2})$

Parallel Query Algorithms

Runs in rounds

Round: Issue a set of k queries simultaneously:

$$\text{“What is } \mathrm{rk}(S_1)?\text{”},\ \ \text{“What is } \mathrm{rk}(S_2)?\text{”},\ \dots,\ \text{“What is } \mathrm{rk}(S_k)?\text{”}$$

Can only depend on answers to queries in previous rounds!

Tradeoff:

Total number of queries used	$O(n^{3/2})$	$O(2^n)$
Number of rounds (*adaptivity*)	$O(n^{3/2})$	1

Parallel Query Algorithms

Runs in rounds

Round: Issue a set of k queries simultaneously:

$$\text{“What is } \mathrm{rk}(S_1)\text{?”, “What is } \mathrm{rk}(S_2)\text{?”, ..., “What is } \mathrm{rk}(S_k)\text{?”}$$

Can only depend on answers to queries in previous rounds!

Tradeoff:

Total number of queries used	$O(n^{3/2})$	$O(\mathrm{poly}(n))$	$O(2^n)$
Number of rounds (*adaptivity*)	$O(n^{3/2})$	**?**	1

Main Question:
How many rounds do we need if we can only use $O(\mathrm{poly}(n))$ queries in total?

Can we do. . .

$O(n)$-rounds?

Can we do. . .

$O(n)$-rounds? YES Straightforward (Edmonds 60s)

Can we do. . .

$O(n)$-rounds? YES Straightforward (Edmonds 60s)

$O(\text{polylog}(n))$-rounds?

Can we do...

$O(n)$-rounds? YES Straightforward (Edmonds 60s)

$O(\text{polylog}(n))$-rounds? YES For bipartite matching and linear matroid intersection (Lovász'79, KUW'86, FGT'19, GT'20)

Can we do...

$O(n)$-rounds?	YES	Straightforward (Edmonds 60s)
$O(\mathrm{polylog}(n))$-rounds?	YES	For bipartite matching and linear matroid intersection (Lovász'79, KUW'86, FGT'19, GT'20)
	NO	For general matroids: $\tilde{\Omega}(n^{1/3})$ (indep: KUW'86, rank: CCK'21)

Can we do...

$O(n)$-rounds? YES Straightforward (Edmonds 60s)

$O(\text{polylog}(n))$-rounds? YES For bipartite matching and linear matroid intersection (Lovász'79, KUW'86, FGT'19, GT'20)

NO For general matroids: $\tilde{\Omega}(n^{1/3})$ (indep: KUW'86, rank: CCK'21)

$o(n)$-rounds?

Can we do. . .

$O(n)$-rounds? YES Straightforward (Edmonds 60s)

$O(\mathrm{polylog}(n))$-rounds? YES For bipartite matching and linear matroid intersection (Lovász'79, KUW'86, FGT'19, GT'20)

NO For general matroids: $\tilde{\Omega}(n^{1/3})$ (indep: KUW'86, rank: CCK'21)

$o(n)$-rounds? YES!

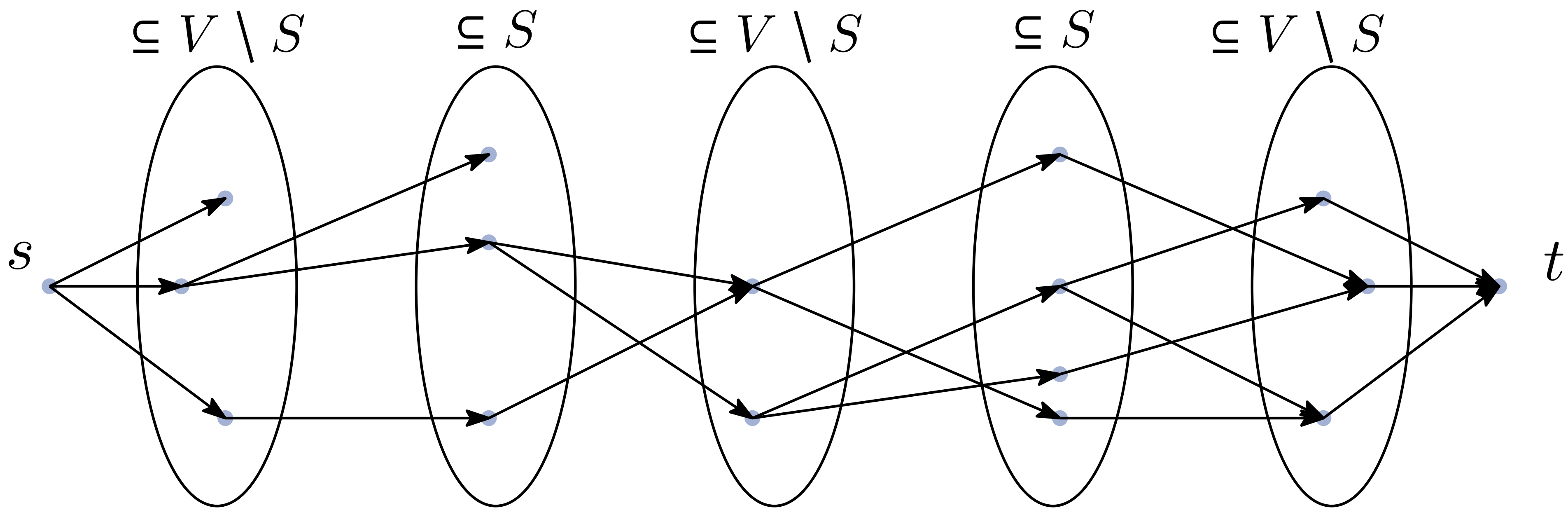
Main Theorem:
Matroid Intersection can be solved using $\mathrm{poly}(n)$ total queries and:

- $O(n^{3/4}) = O(n^{0.75})$ rounds (rank-oracle)
- $O(n^{7/8}) = O(n^{0.875})$ rounds (independence-oracle)

Exchange Graph & Augmenting Paths [Edmonds'60s]

Given $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ construct the *exchange graph* $G(S)$.

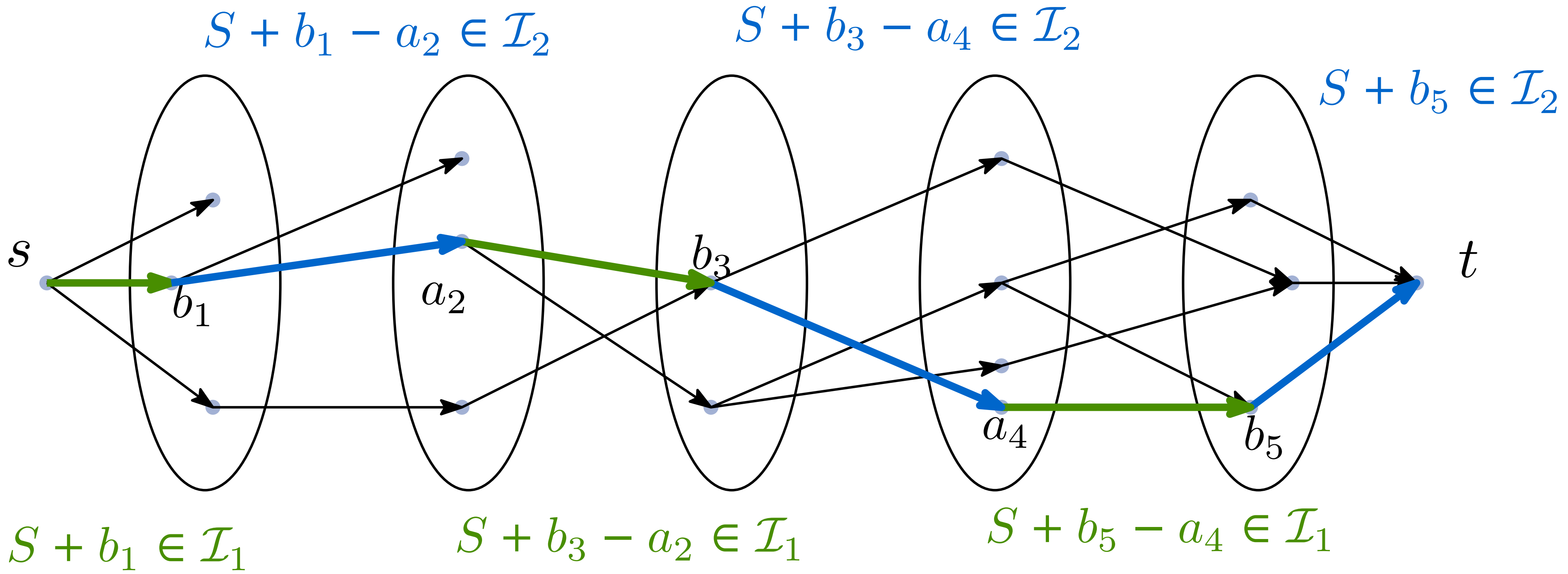
s, t-path $\iff$ can increase size of S!



Exchange Graph & Augmenting Paths [Edmonds'60s]

Given $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ construct the *exchange graph* $G(S)$.

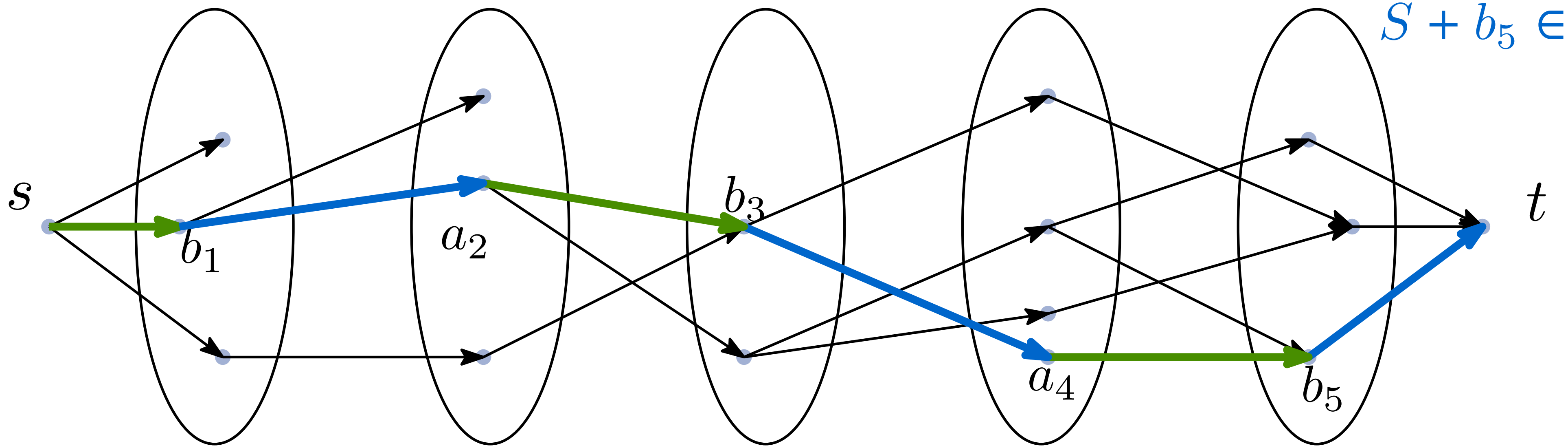
s, t-path $\iff$ can increase size of S!



Exchange Graph & Augmenting Paths [Edmonds'60s]

$$\implies S + b_1 - a_2 + b_3 - a_4 + b_5 \in \mathcal{I}_2$$

$S + b_1 - a_2 \in \mathcal{I}_2$ $\quad$ $S + b_3 - a_4 \in \mathcal{I}_2$ $\quad$ $S + b_5 \in \mathcal{I}_2$



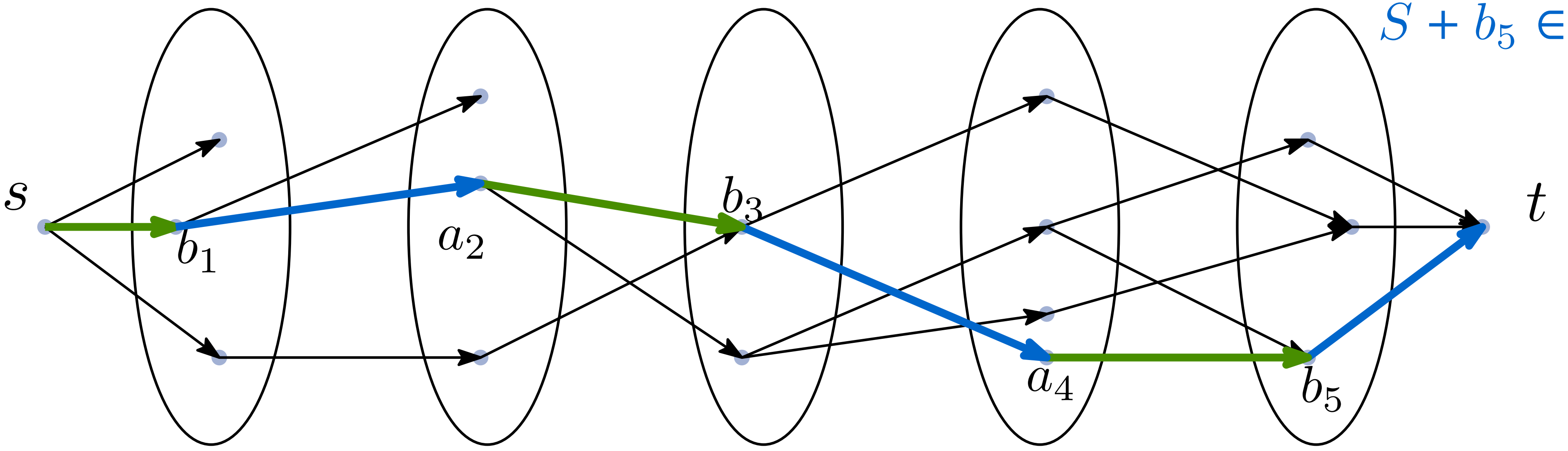
$S + b_1 \in \mathcal{I}_1$ $\quad$ $S + b_3 - a_2 \in \mathcal{I}_1$ $\quad$ $S + b_5 - a_4 \in \mathcal{I}_1$

$$\implies S + b_1 - a_2 + b_3 - a_4 + b_5 \in \mathcal{I}_1$$

Exchange Graph & Augmenting Paths [Edmonds'60s]

$$\implies S + b_1 - a_2 + b_3 - a_4 + b_5 \in \mathcal{I}_2$$

$S + b_1 - a_2 \in \mathcal{I}_2$ $\quad$ $S + b_3 - a_4 \in \mathcal{I}_2$ $\quad$ $S + b_5 \in \mathcal{I}_2$



$S + b_1 \in \mathcal{I}_1$ $\quad$ $S + b_3 - a_2 \in \mathcal{I}_1$ $\quad$ $S + b_5 - a_4 \in \mathcal{I}_1$

$$\implies S + b_1 - a_2 + b_3 - a_4 + b_5 \in \mathcal{I}_1$$

Common independent set $S' := S + b_1 - a_2 + b_3 - a_4 + b_5$ of size $|S'| = |S| + 1$

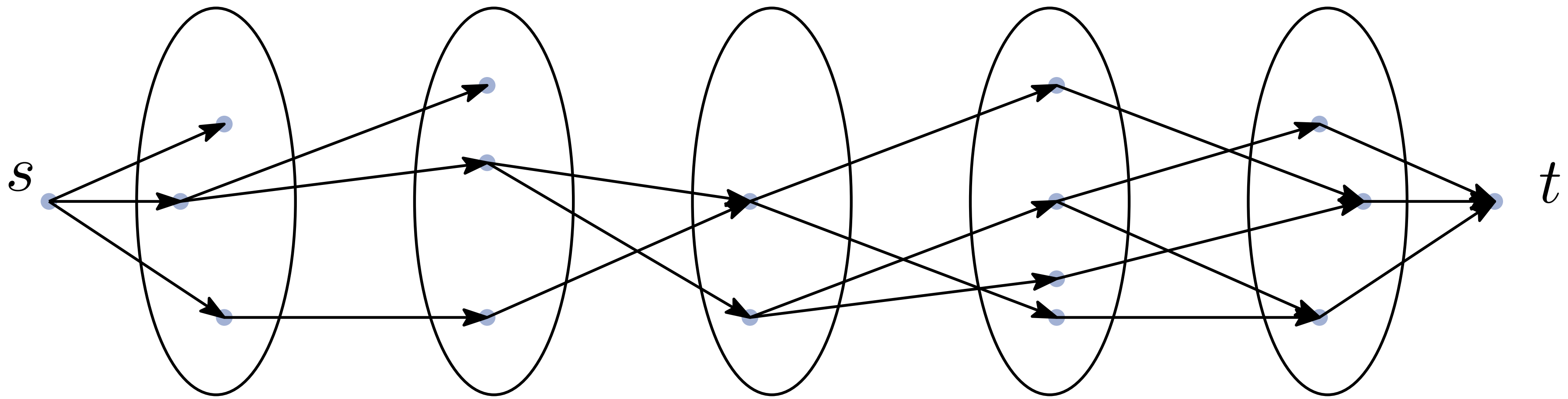
Linear-round Algorithm [Edmonds'60s]

Algorithm $O(n)$ rounds

1. $S = \varnothing$
2. In parallel find all the edges of the exchange graph $G(S)$

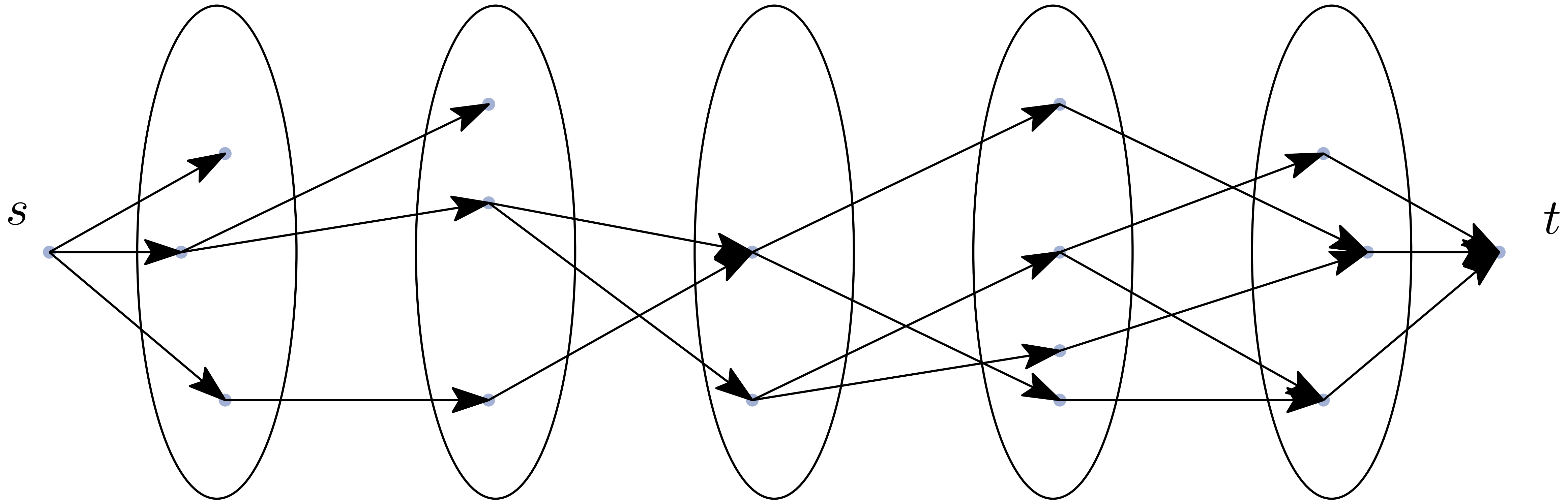
   ▷ 1 round of $O(n^2)$ queries
3. If there is an augmenting path, augment along it and repeat

   ▷ only repeats $O(n)$ times



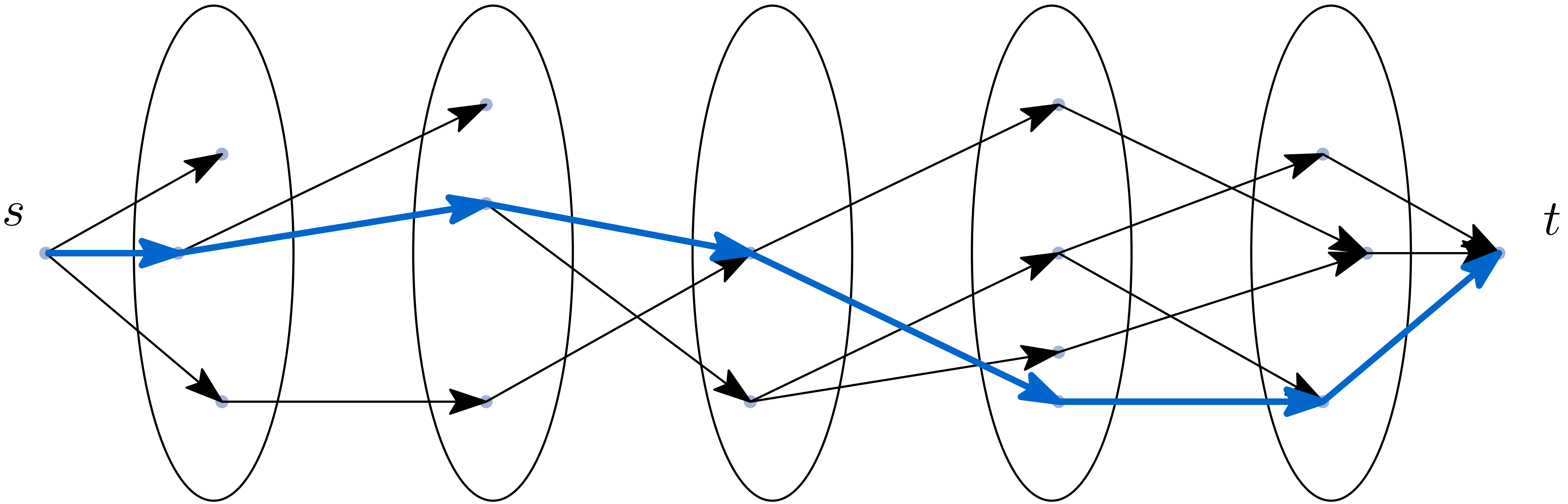
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly. . .



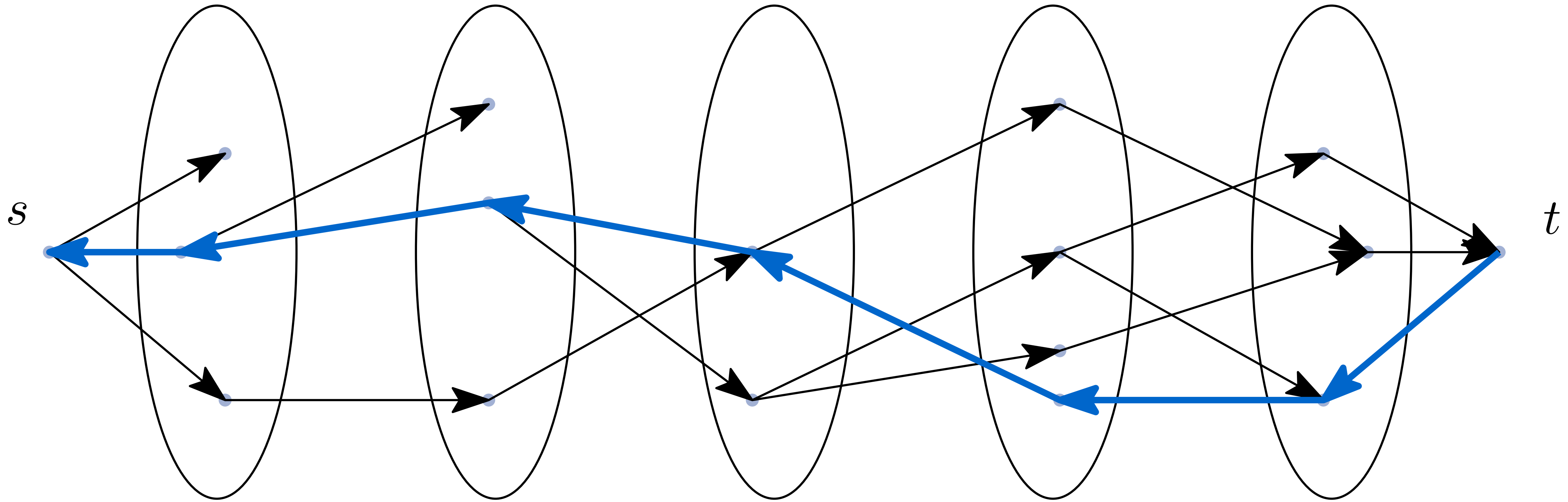
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly...



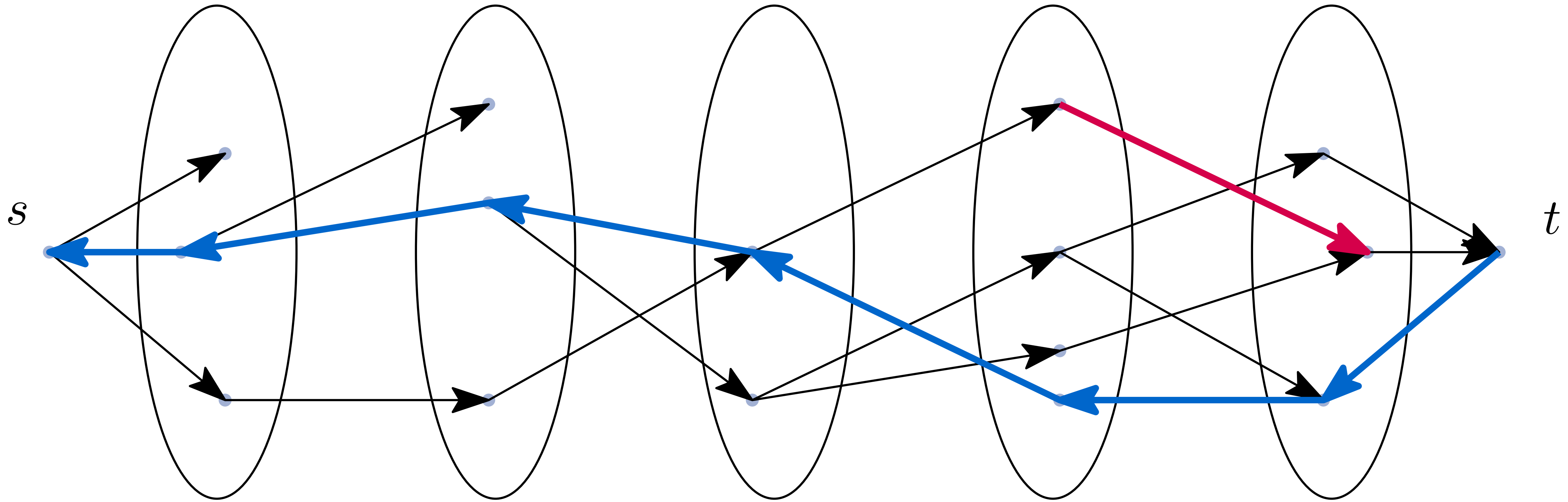
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly...



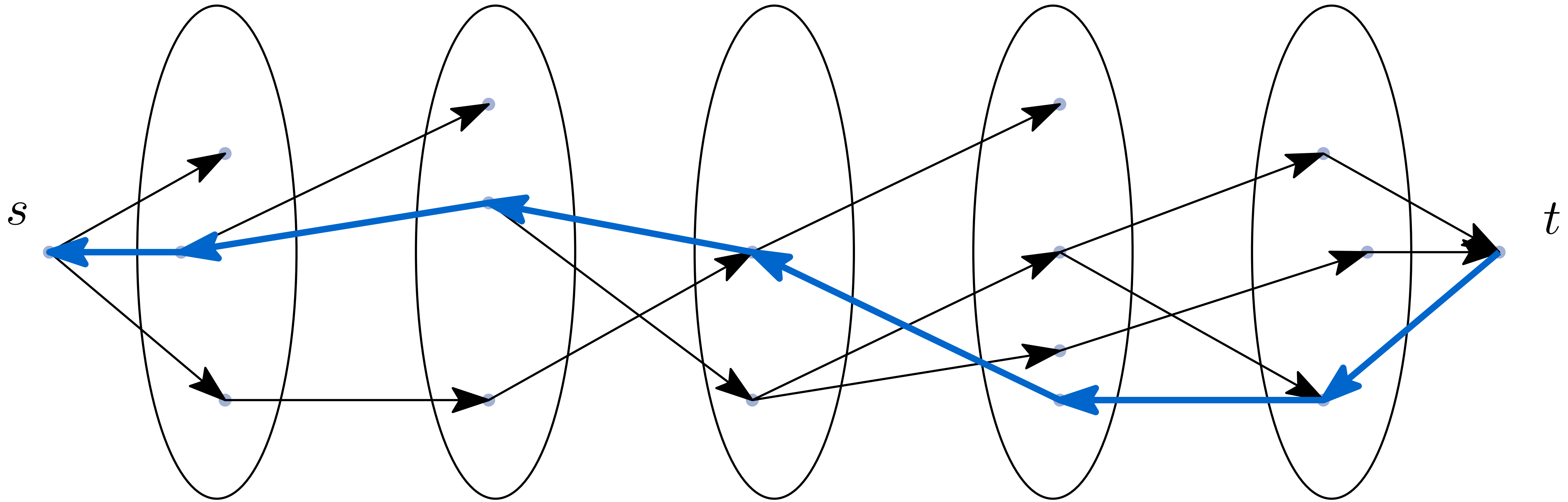
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly. . .



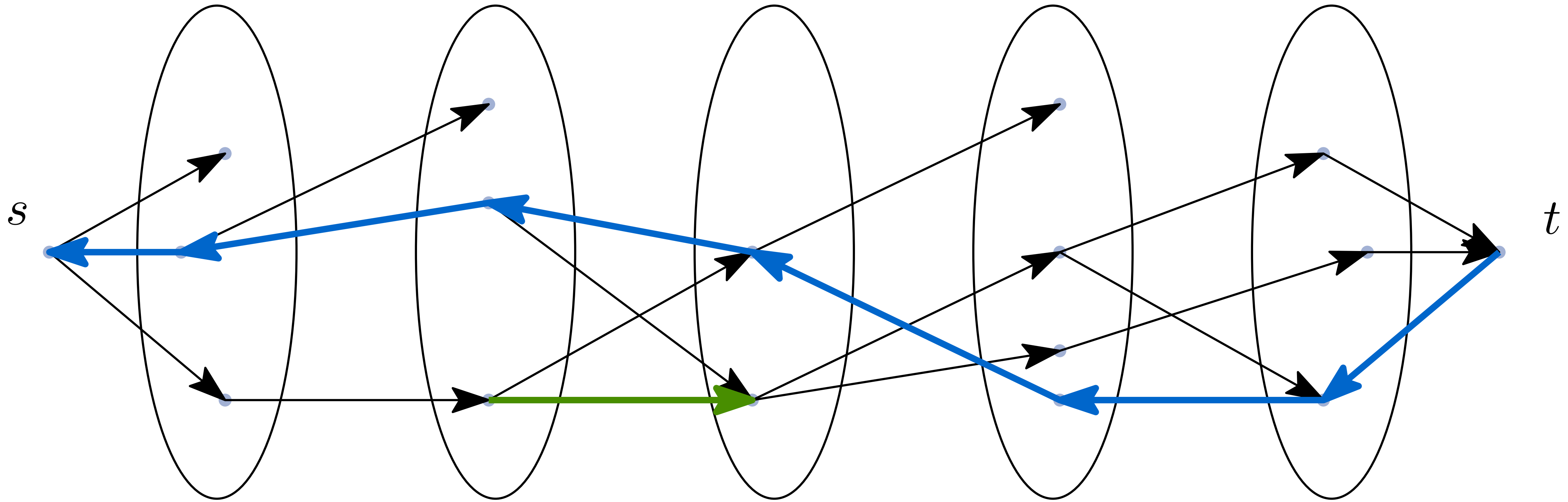
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly...



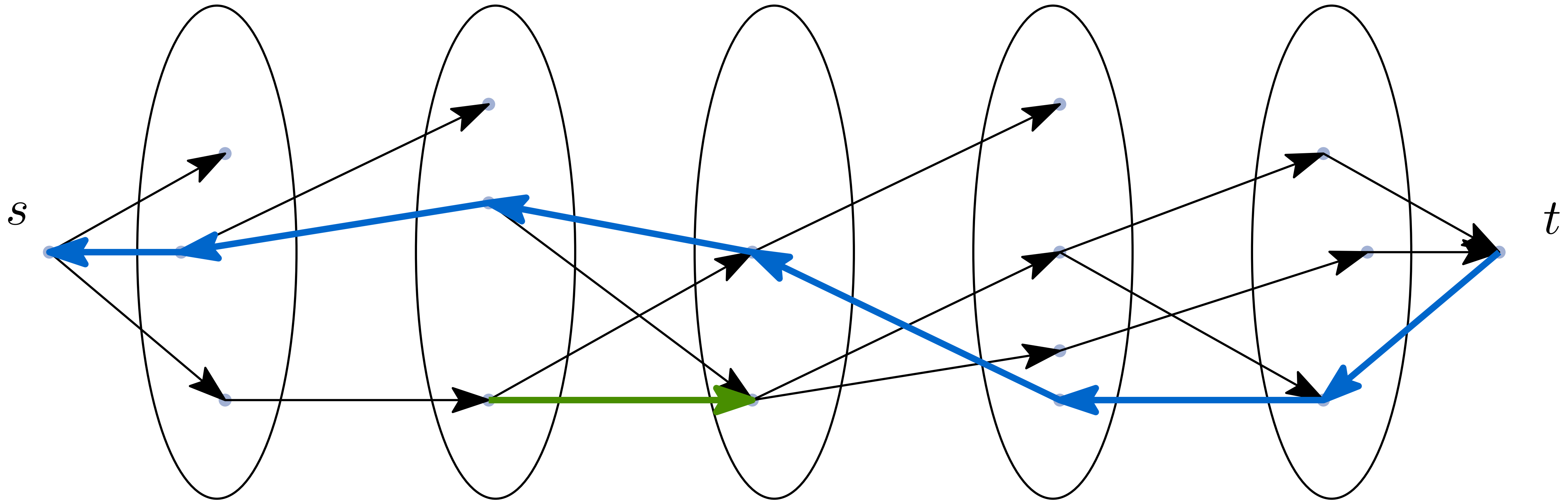
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly...



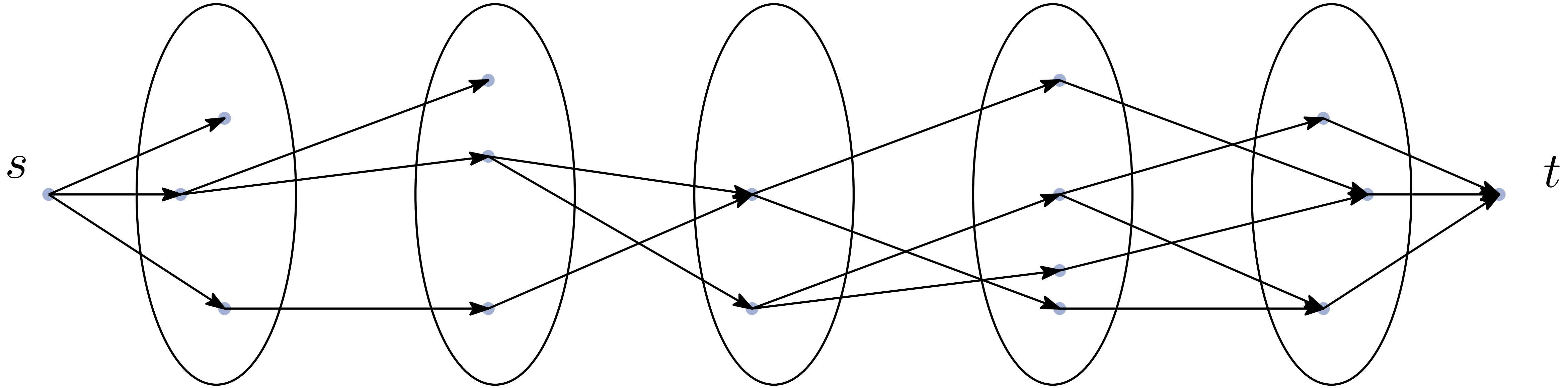
Augmenting in Parallel?

Exchange graph $G(S)$ behaves weirdly...

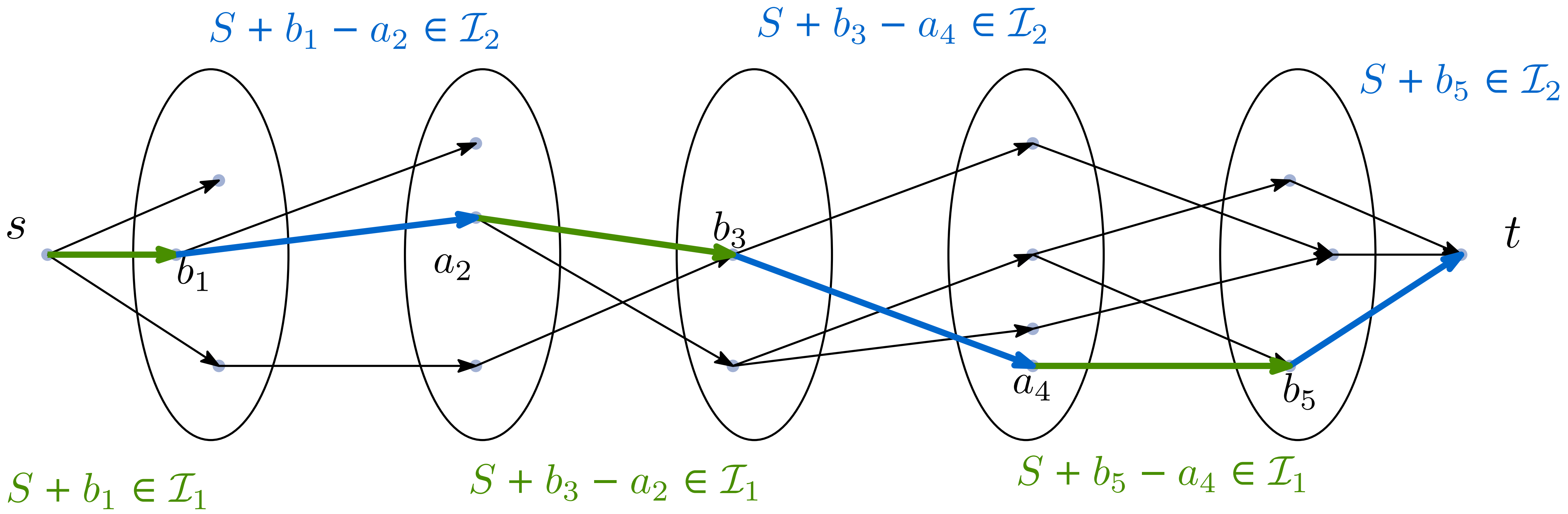


- Disjoint paths not necessarily "compatible"
- Need to handle the inserted edges

Augmenting Sets [Chakrabarty-Lee-Sidford-Singla-Wong'19]



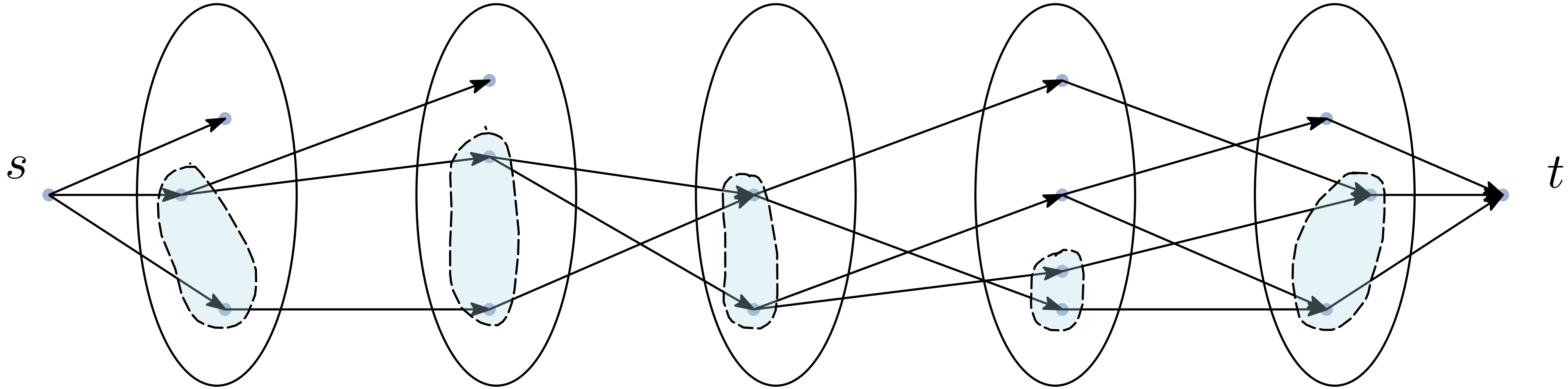
Augmenting Sets [Chakrabarty-Lee-Sidford-Singla-Wong'19]



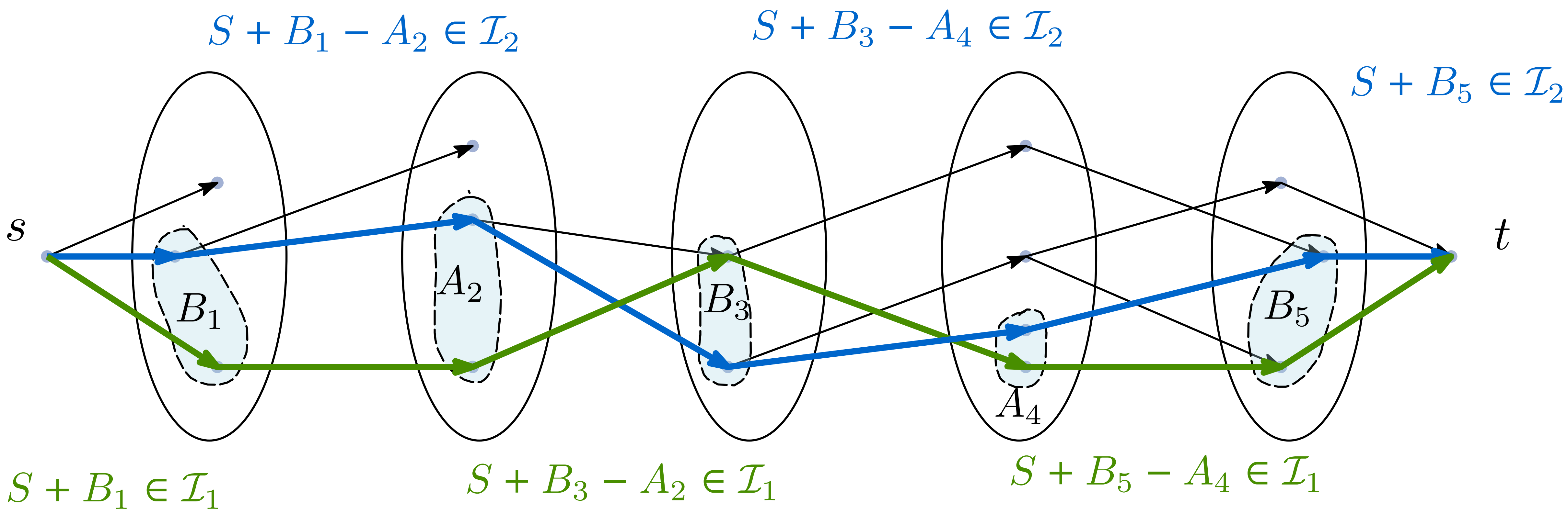
$$\implies S + b_1 - a_2 + b_3 - a_4 + b_5 \in \mathcal{I}_1$$

$$\implies S + b_1 - a_2 + b_3 - a_4 + b_5 \in \mathcal{I}_2$$

Augmenting Sets [Chakrabarty-Lee-Sidford-Singla-Wong'19]



Augmenting Sets [Chakrabarty-Lee-Sidford-Singla-Wong'19]



$$\implies S + B_1 - A_2 + B_3 - A_4 + B_5 \in \mathcal{I}_1 \cap \mathcal{I}_2$$

Sublinear-round Algorithm

Key Lemma ("Blocking-Flow" Approximation Algorithm)
We can get a $(1-\varepsilon)$-approximation of the matroid intersection problem in $O(\sqrt{n}/\varepsilon)$ rounds of $poly(n)$ many rank-queries.

Sublinear-round Algorithm

Key Lemma ("Blocking-Flow" Approximation Algorithm)
We can get a $(1-\varepsilon)$-approximation of the matroid intersection problem in $O(\sqrt{n}/\varepsilon)$ rounds of $poly(n)$ many rank-queries.

Exact Algorithm

1. Run $O(\sqrt{n}/\varepsilon)$-round $(1-\varepsilon)$-approximation algorithm
2. Now we have $|S| \geq \mathrm{OPT} - O(n\varepsilon)$
3. Do these augmentations one-by-one, in a single round each

Sublinear-round Algorithm

Key Lemma (“Blocking-Flow” Approximation Algorithm)
We can get a $(1-\varepsilon)$-approximation of the matroid intersection problem in $O(\sqrt{n}/\varepsilon)$ rounds of $poly(n)$ many rank-queries.

Exact Algorithm

1. Run $O(\sqrt{n}/\varepsilon)$-round $(1-\varepsilon)$-approximation algorithm with $\varepsilon = n^{-1/4}$
2. Now we have $|S| \geq \mathrm{OPT} - O(n\varepsilon) = \mathrm{OPT} - n^{3/4}$
3. Do these augmentations one-by-one, in a single round each

$$O(n^{3/4})\text{-round algorithm!}$$

Sublinear-round Algorithm

Key Lemma (“Blocking-Flow” Approximation Algorithm)
We can get a $(1-\varepsilon)$-approximation of the matroid intersection problem in $O(\sqrt{n}/\varepsilon)$ rounds of $poly(n)$ many rank-queries.

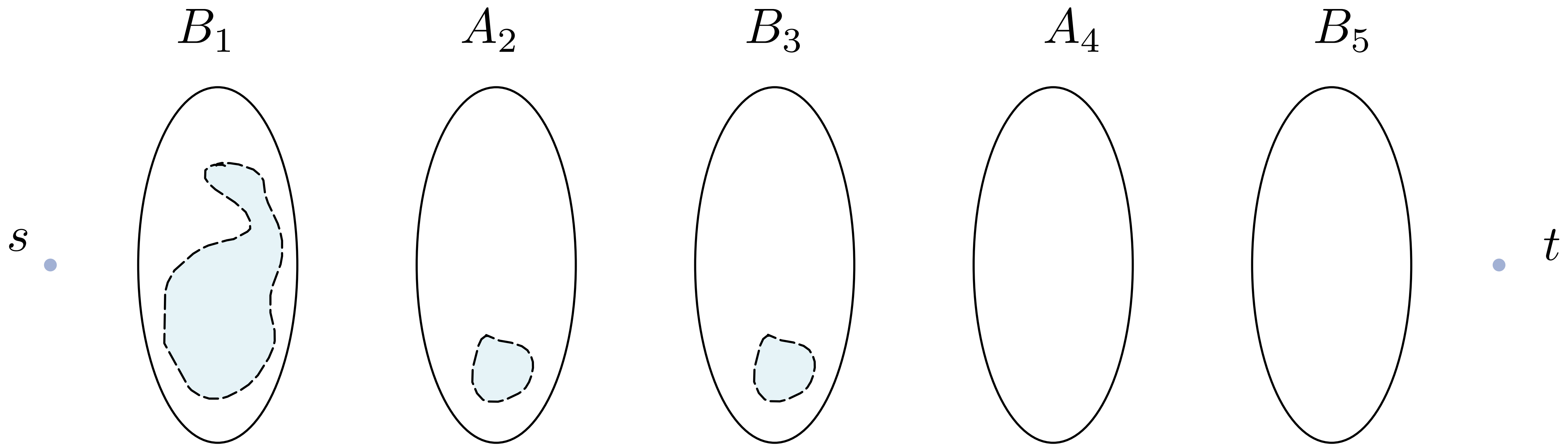
Exact Algorithm

1. Run $O(\sqrt{n}/\varepsilon)$-round $(1-\varepsilon)$-approximation algorithm with $\varepsilon = n^{-1/4}$
2. Now we have $|S| \geq \mathrm{OPT} - O(n\varepsilon) = \mathrm{OPT} - n^{3/4}$
3. Do these augmentations one-by-one, in a single round each

$O(n^{3/4})$-round algorithm!

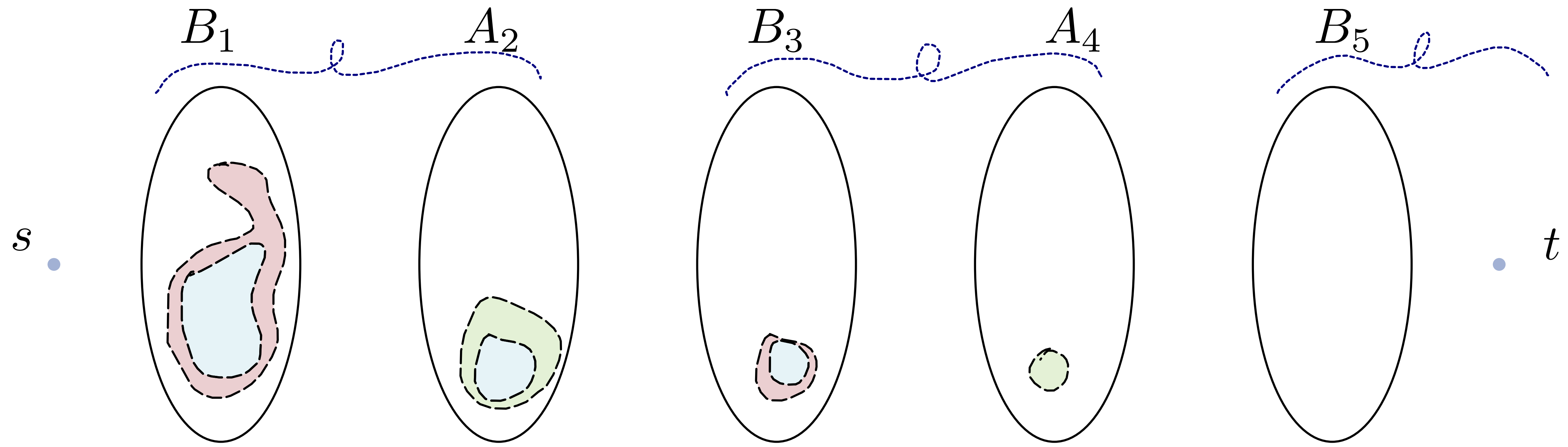
Refining: Combining algorithms of [CLSSW'19] and [KUW'86]

Partial Augmenting Set, "*Staircase*"



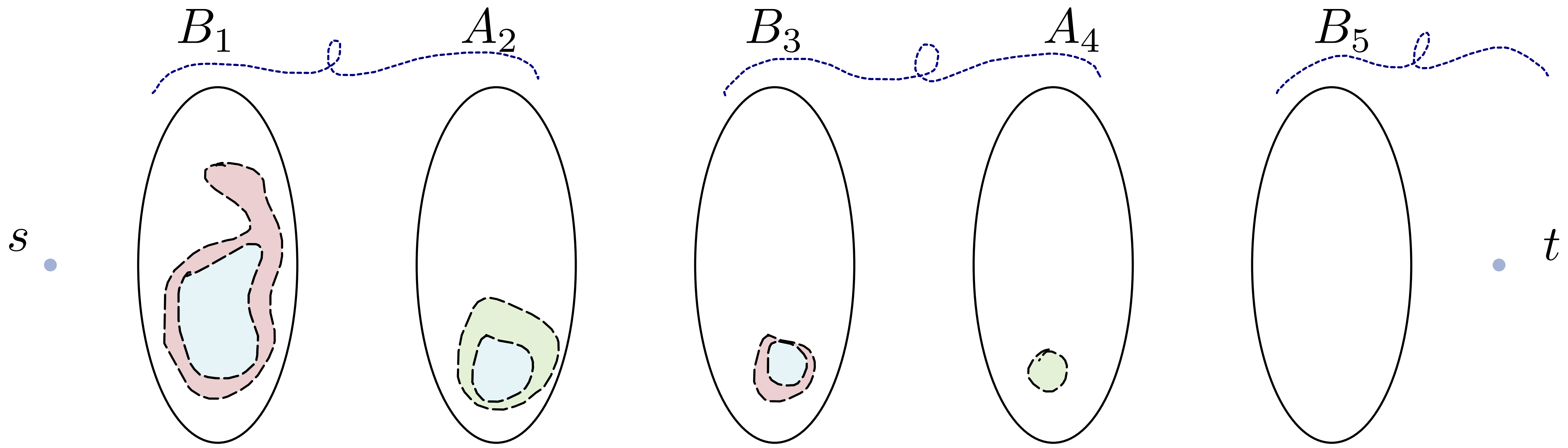
Refining: Combining algorithms of [CLSSW'19] and [KUW'86]

In $O(1)$ rounds accessing $\mathcal{M}_2$:



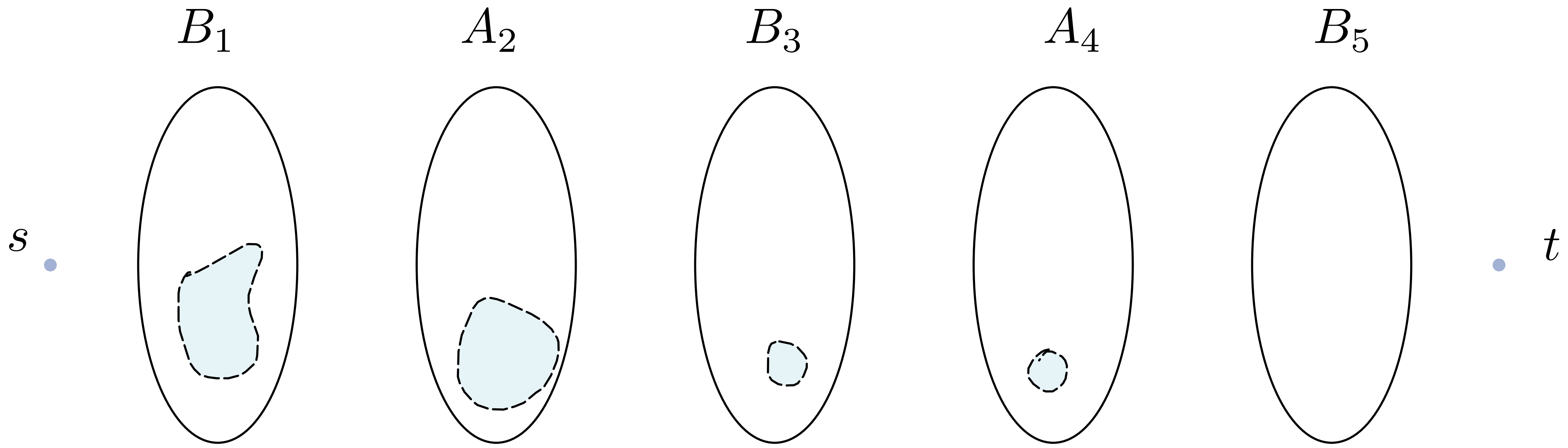
Refining: Combining algorithms of [CLSSW'19] and [KUW'86]

In $O(1)$ rounds accessing $\mathcal{M}_2$:



Roughly $|B_1| - |B_\ell|$ elements are newly *discarded* or *selected*

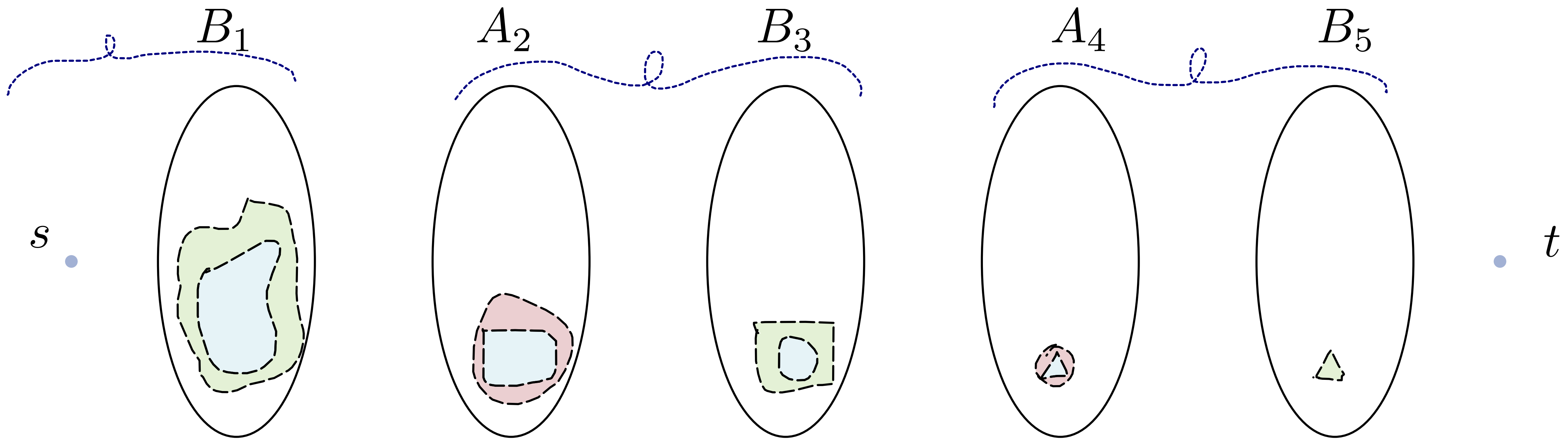
Refining: Combining algorithms of [CLSSW'19] and [KUW'86]



Roughly $|B_1| - |B_\ell|$ elements are newly *discarded* or *selected*

Refining: Combining algorithms of [CLSSW'19] and [KUW'86]

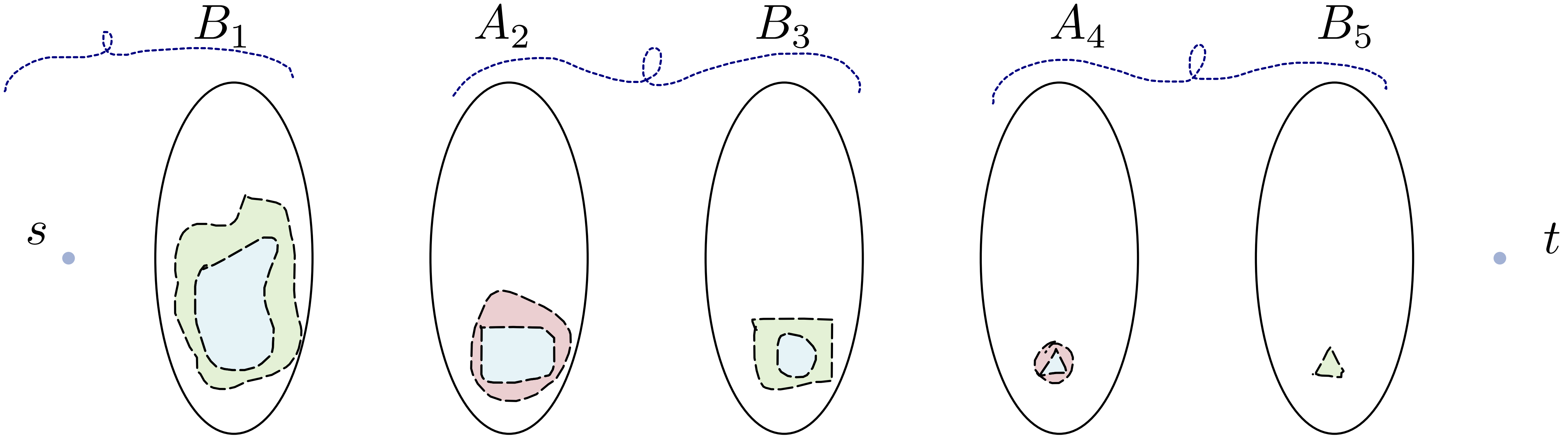
In $O(1)$ rounds accessing $\mathcal{M}_1$:



Roughly $|B_1| - |B_\ell|$ elements are newly *discarded* or *selected*

Refining: Combining algorithms of [CLSSW'19] and [KUW'86]

In $O(1)$ rounds accessing $\mathcal{M}_1$:

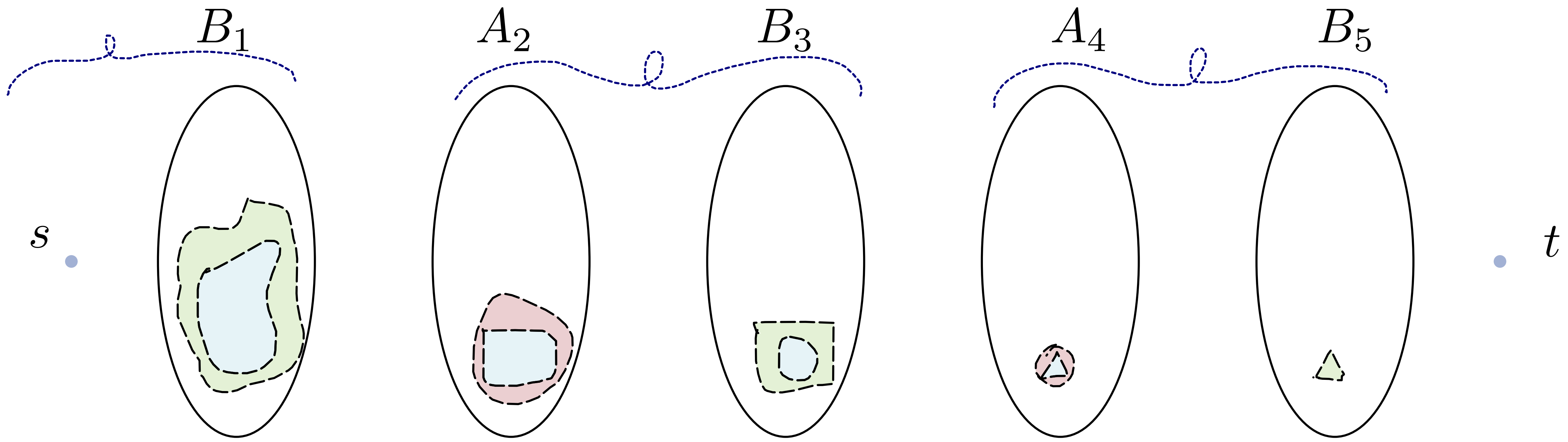


Roughly $|B_1| - |B_\ell|$ elements are newly *discarded* or *selected*

Each element can go *free* → *selected* → *discarded*

Refining: Combining algorithms of [CLSSW'19] and [KUW'86]

In $O(1)$ rounds accessing $\mathcal{M}_1$:

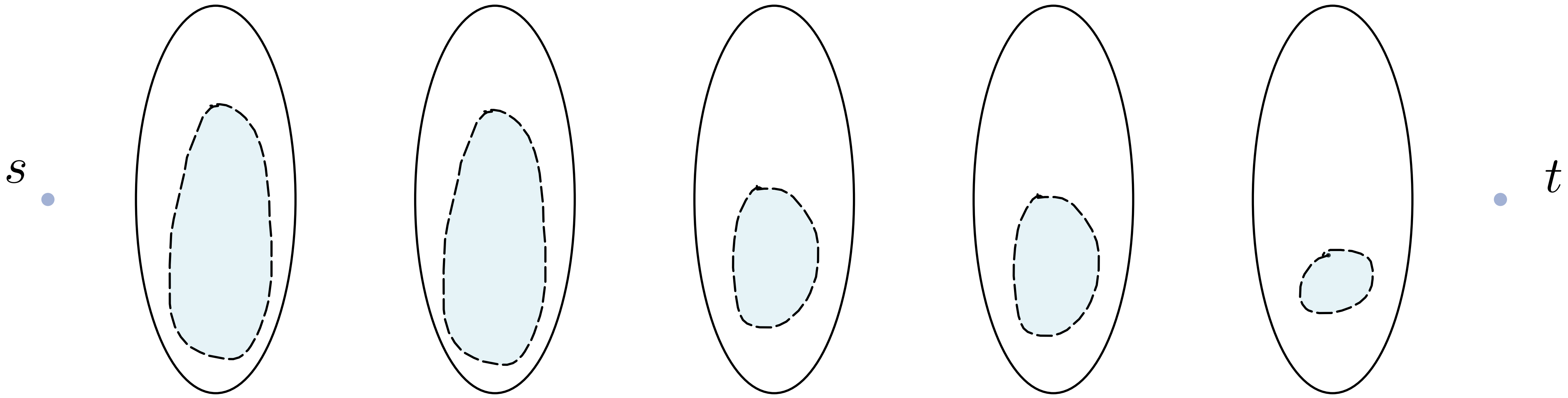


Roughly $|B_1| - |B_\ell|$ elements are newly *discarded* or *selected*

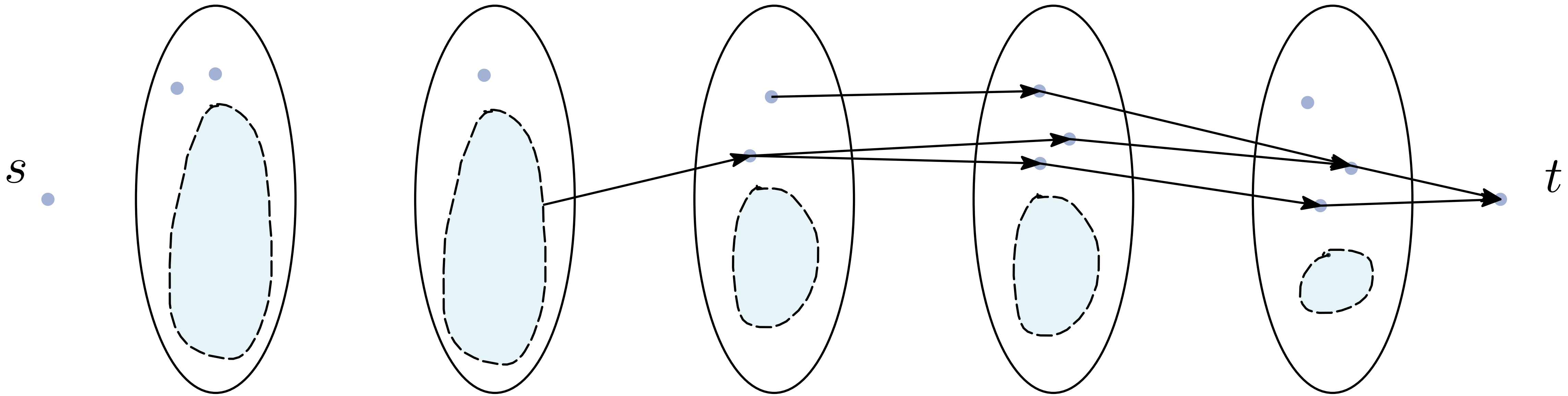
Each element can go *free* → *selected* → *discarded*

Only $\sqrt{n}$ rounds until $|B_1| - |B_\ell| \leq \sqrt{n}$

Falling back to paths

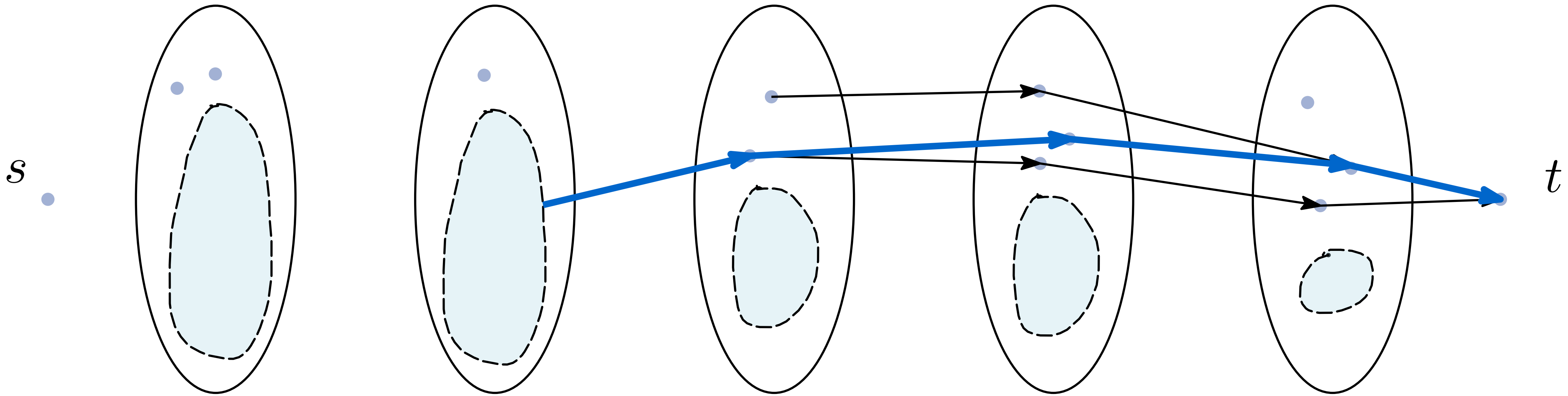


Falling back to paths



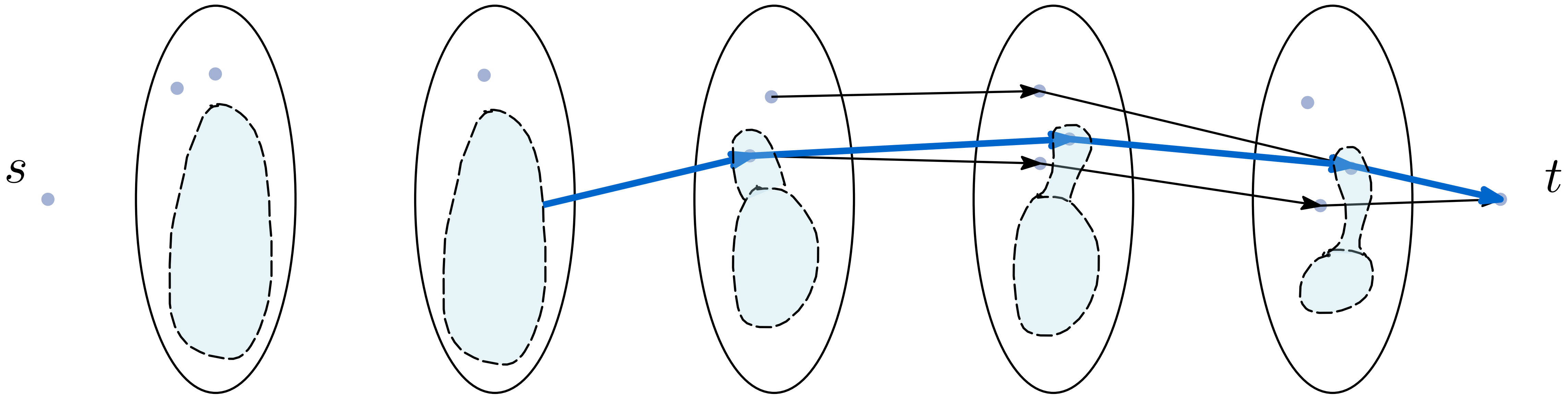
- Can define a graph *with respect to* our “staircase”

Falling back to paths



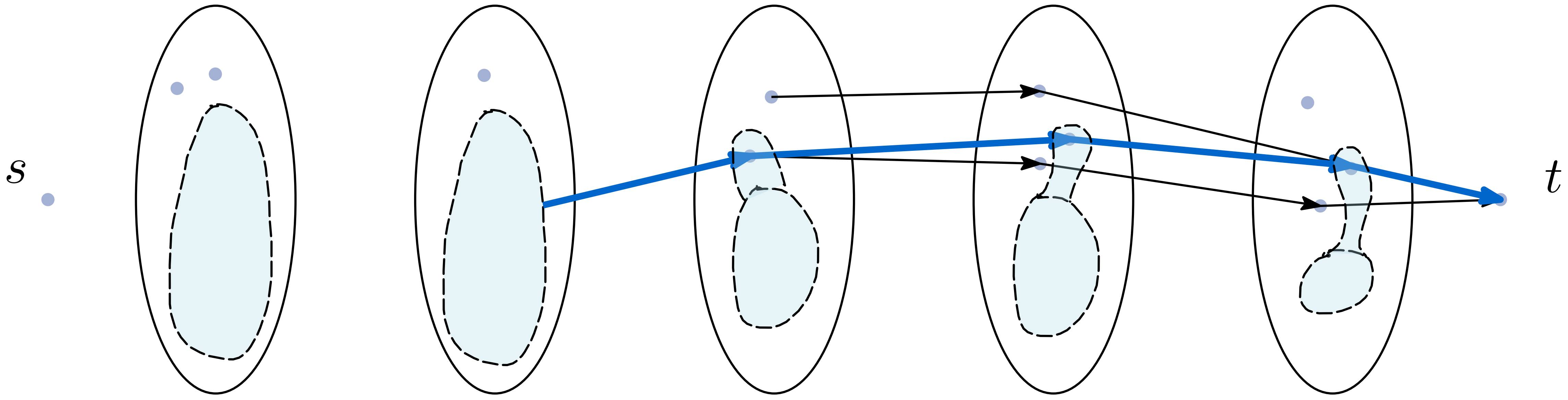
- Can define a graph *with respect to* our “staircase”
- In a single round we can find an “augmenting path”

Falling back to paths



- Can define a graph *with respect to* our “staircase”
- In a single round we can find an “augmenting path”

Falling back to paths



- Can define a graph *with respect to* our “staircase”
- In a single round we can find an “augmenting path”
- Only need to repeat $O(\sqrt{n})$ times

Open Problems

- What is the actual number of rounds required?
  - Somewhere between $\tilde{\Omega}\left(n^{1/3}\right)$ and $O\left(n^{3/4}\right)$. $\sqrt{n}$?
- What about *submodular function minimization* (SFM)?
- What about *weighted* matroid intersection?
  - Similar $O(n)$ one-by-one algorithm works.
  - Can we also acheive sublinear number of rounds?

Thanks!