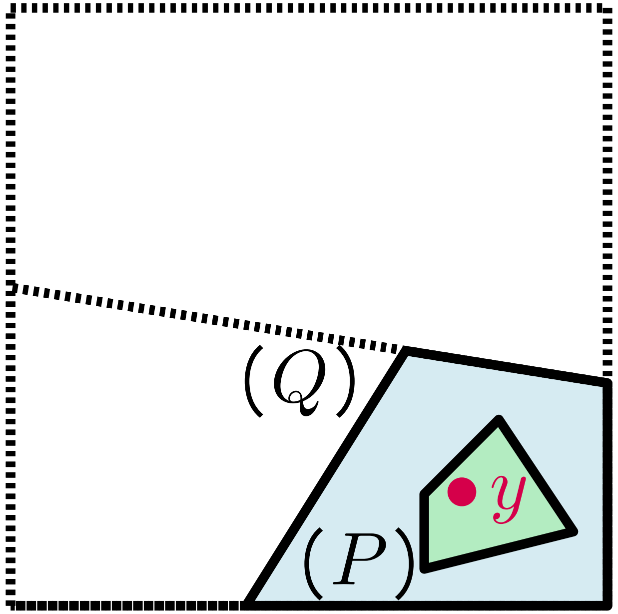
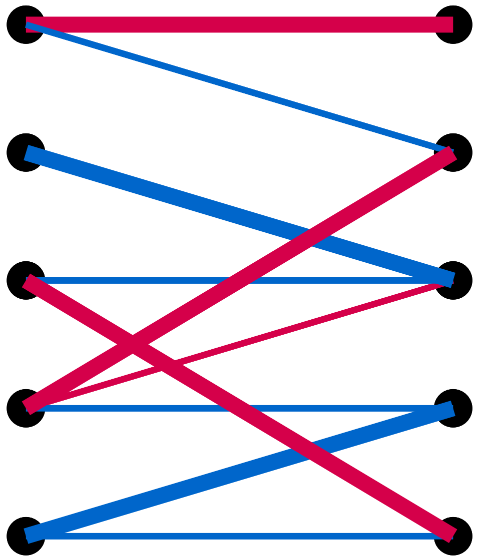
Nearly Optimal Communication and Query Complexity of Bipartite Matching

Joakim Blikstad

KTH Royal Institute of Technology





To appear in FOCS'22

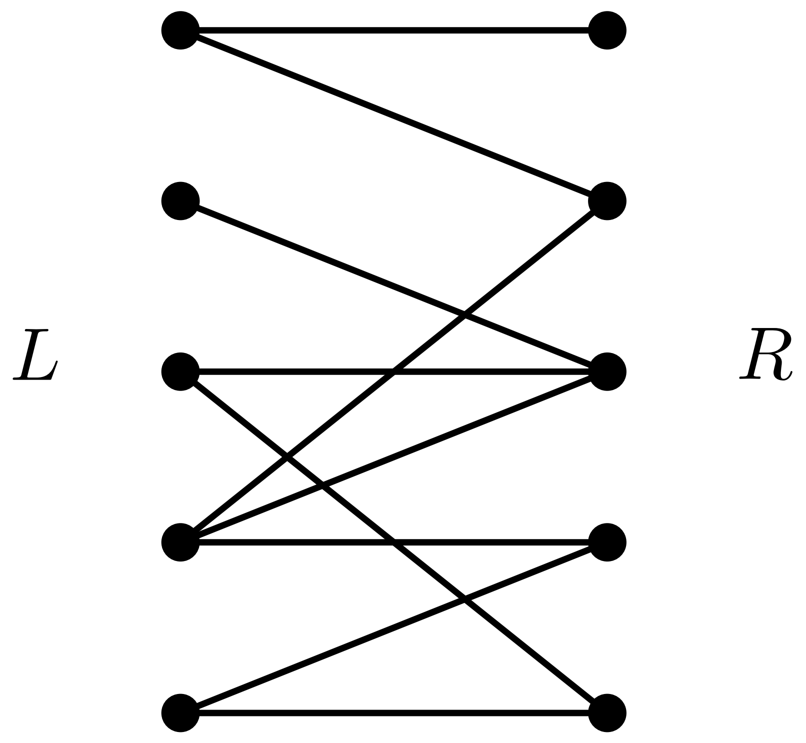
Joint work with: Jan van den Brand, Yuval Efron, Danupon Nanongkai, and Sagnik Mukhopadhyay.

TCS+ talk, fall 2022

Biparite Matching

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

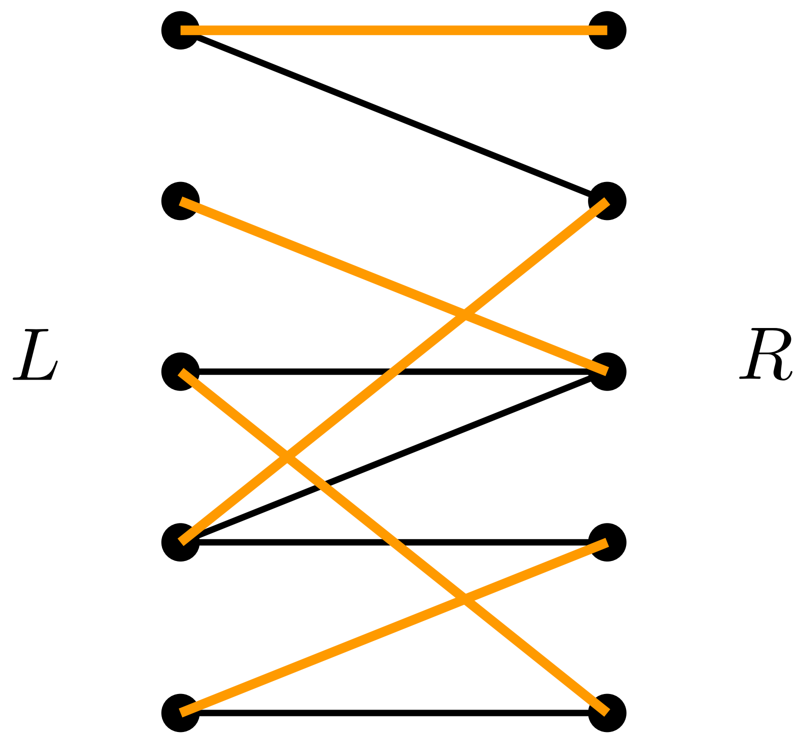
Goal: Find a maximum matching $M \subseteq E$ of G



Biparite Matching

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

Goal: Find a maximum matching $M \subseteq E$ of G



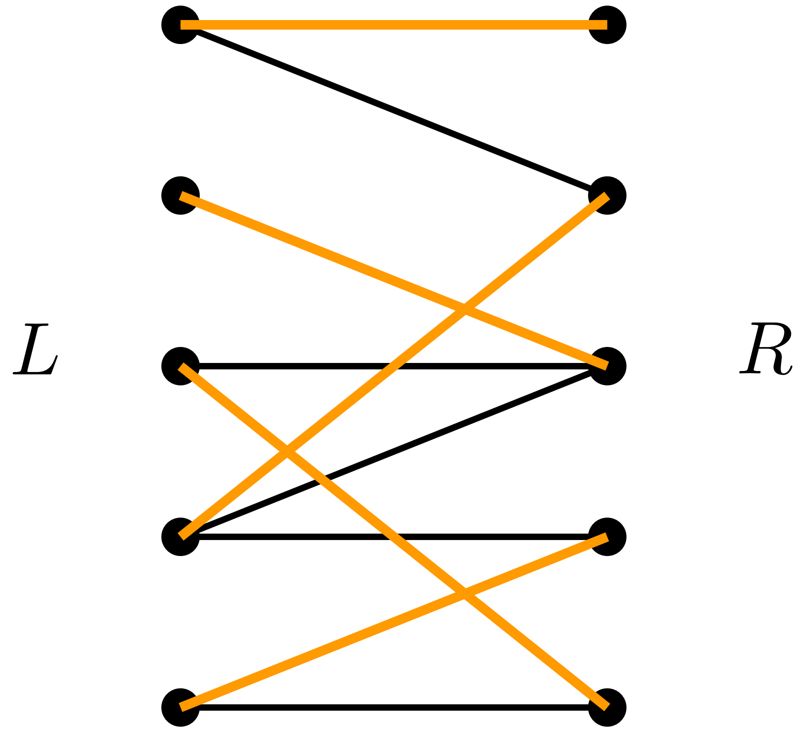
Biparite Matching

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

Goal: Find a maximum matching $M \subseteq E$ of G

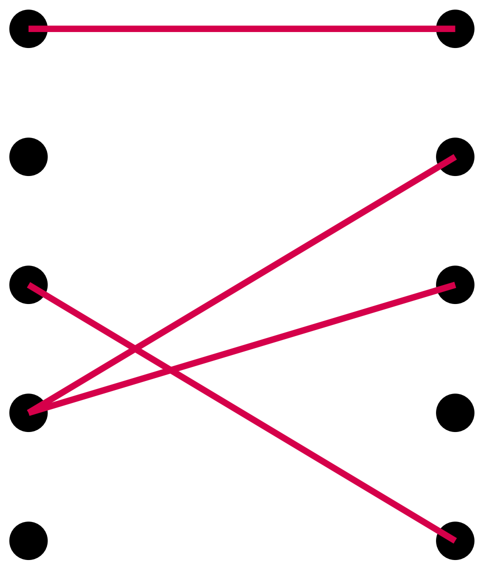
Solve (sequentially) in:

- $\tilde{O}(m + n\sqrt{n})$ [vdBLNPSSSW'20]
- $O(m^{1+o(1)})$ [CKLPGS'22]



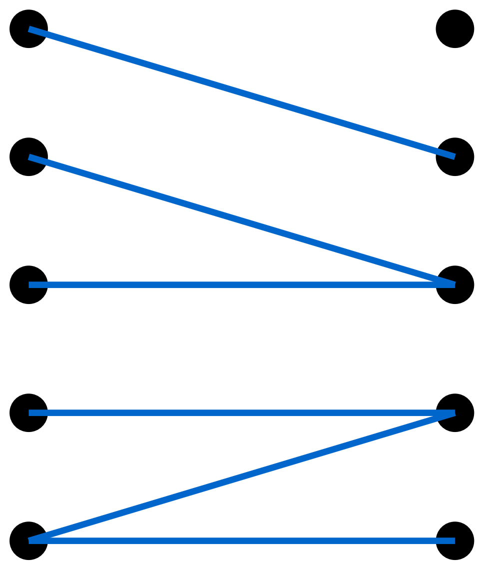
Two-Party Communication Model

Alice



E_A

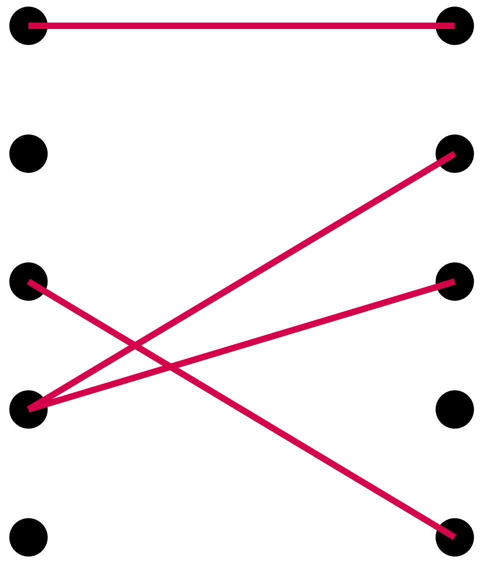
Bob



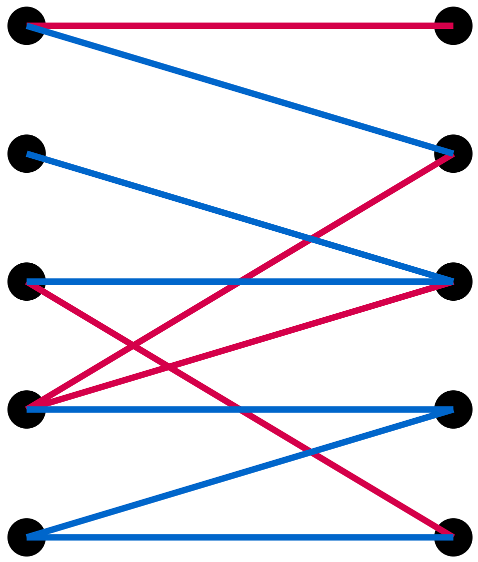
E_B

Two-Party Communication Model

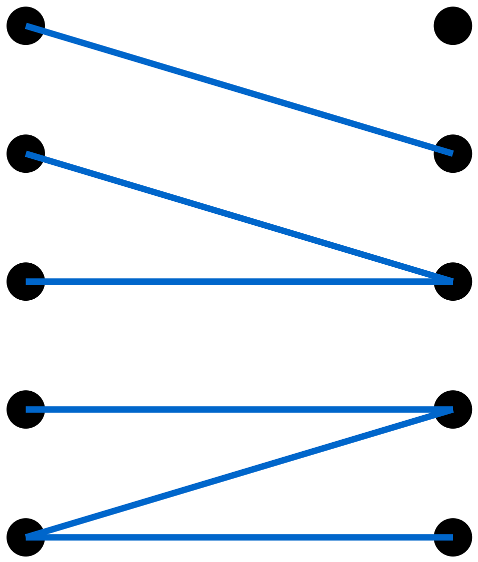
Alice **Bob**



E_A



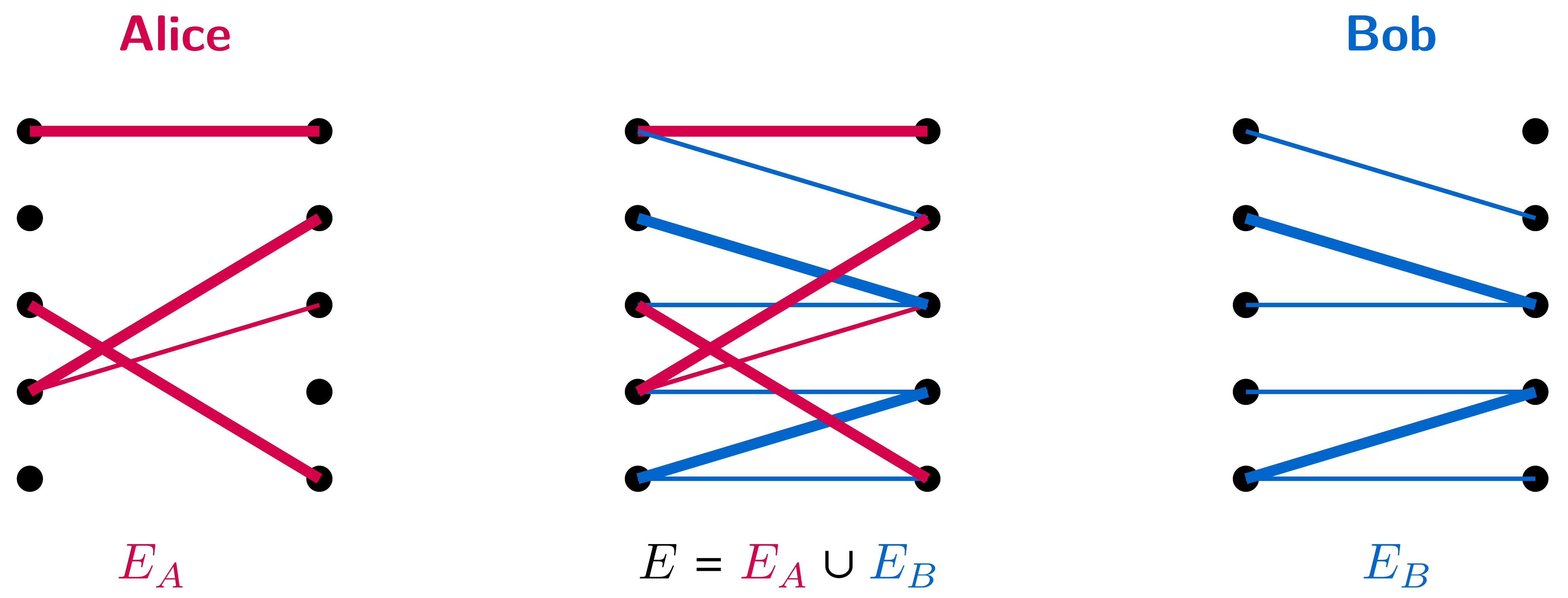
$E = E_A \cup E_B$



E_B

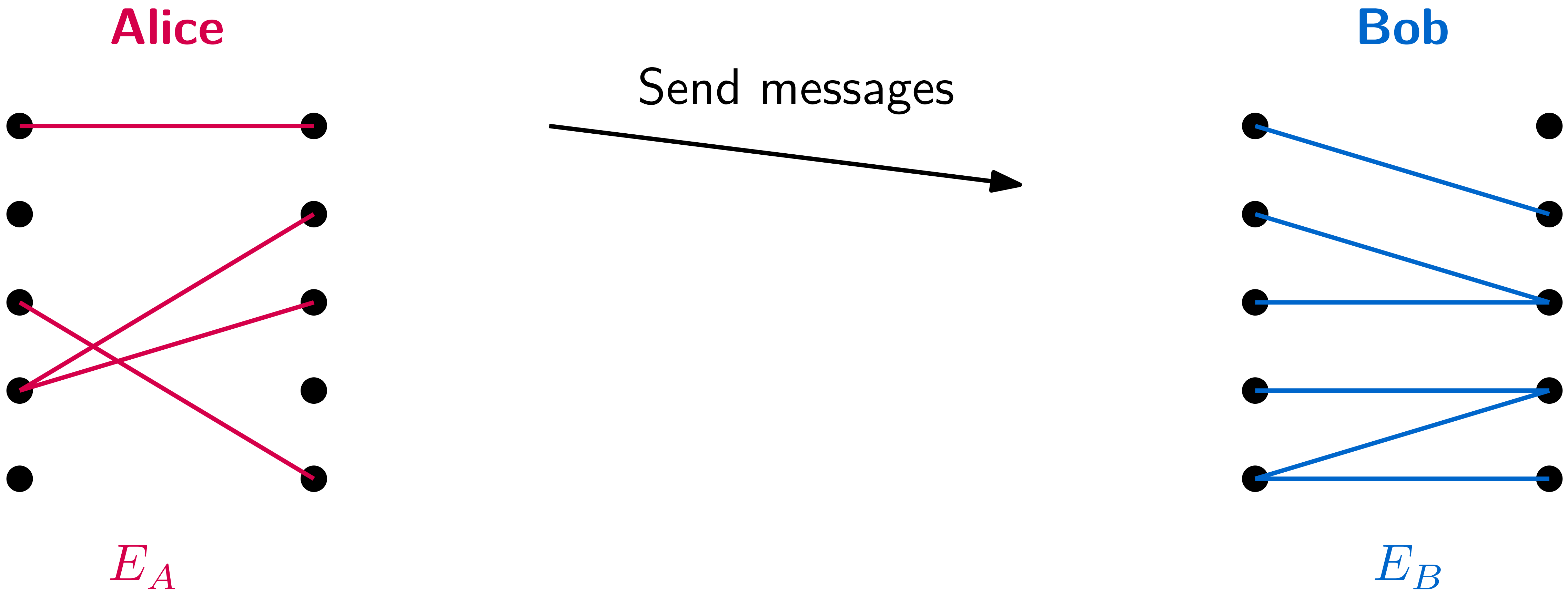
Goal: Solve matching on the union of their graphs

Two-Party Communication Model



Goal: Solve matching on the union of their graphs

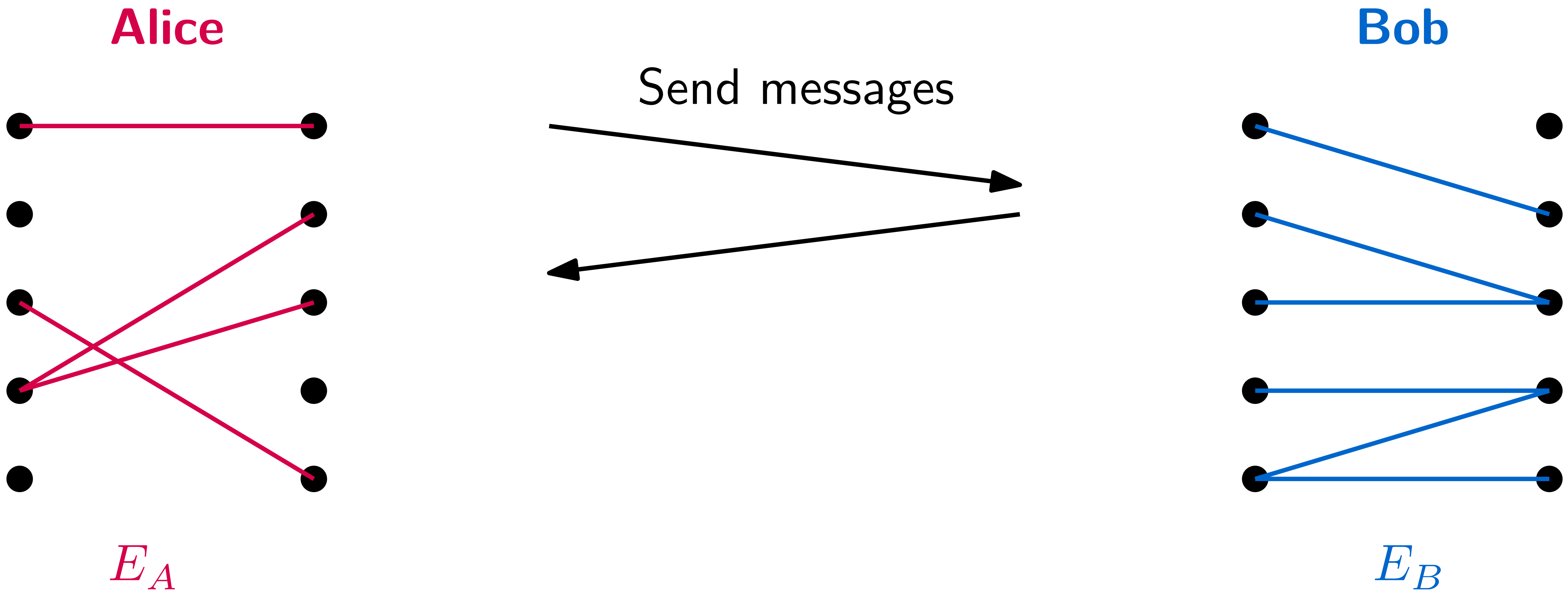
Two-Party Communication Model



Goal: Solve matching on the union of their graphs

With as few bits of communication!

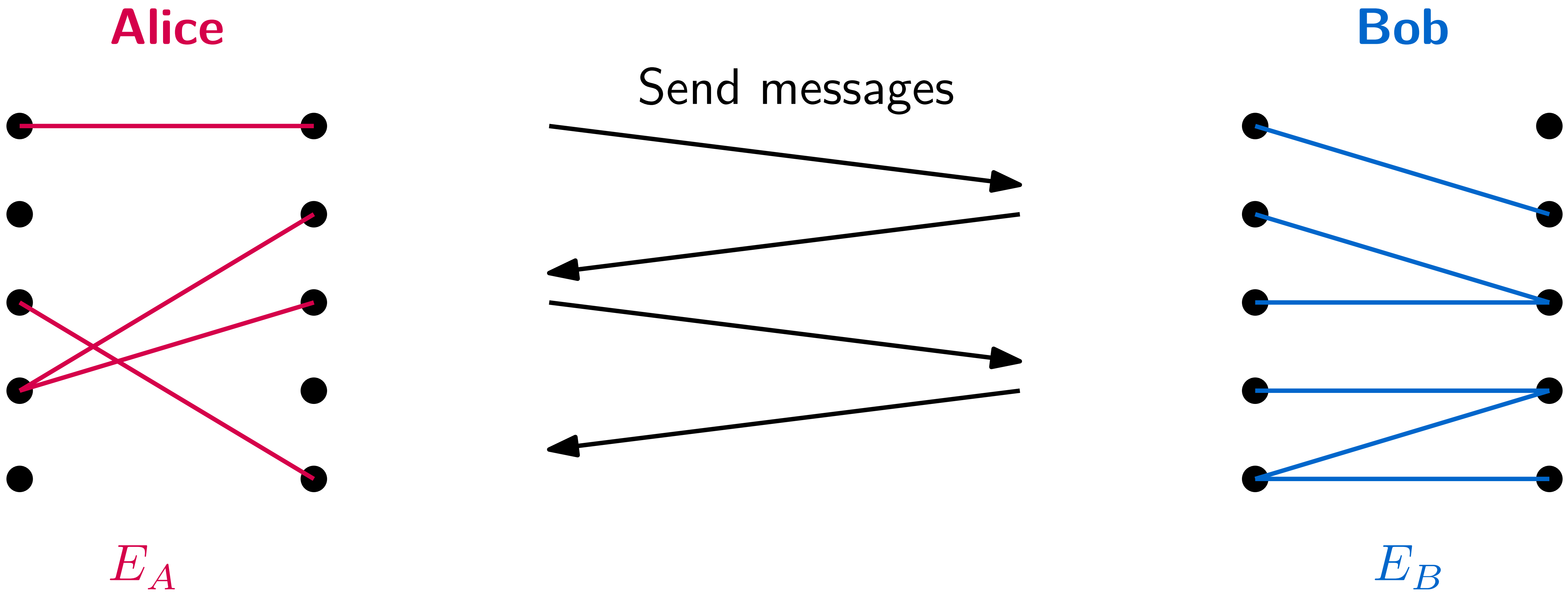
Two-Party Communication Model



Goal: Solve matching on the union of their graphs

With as few bits of communication!

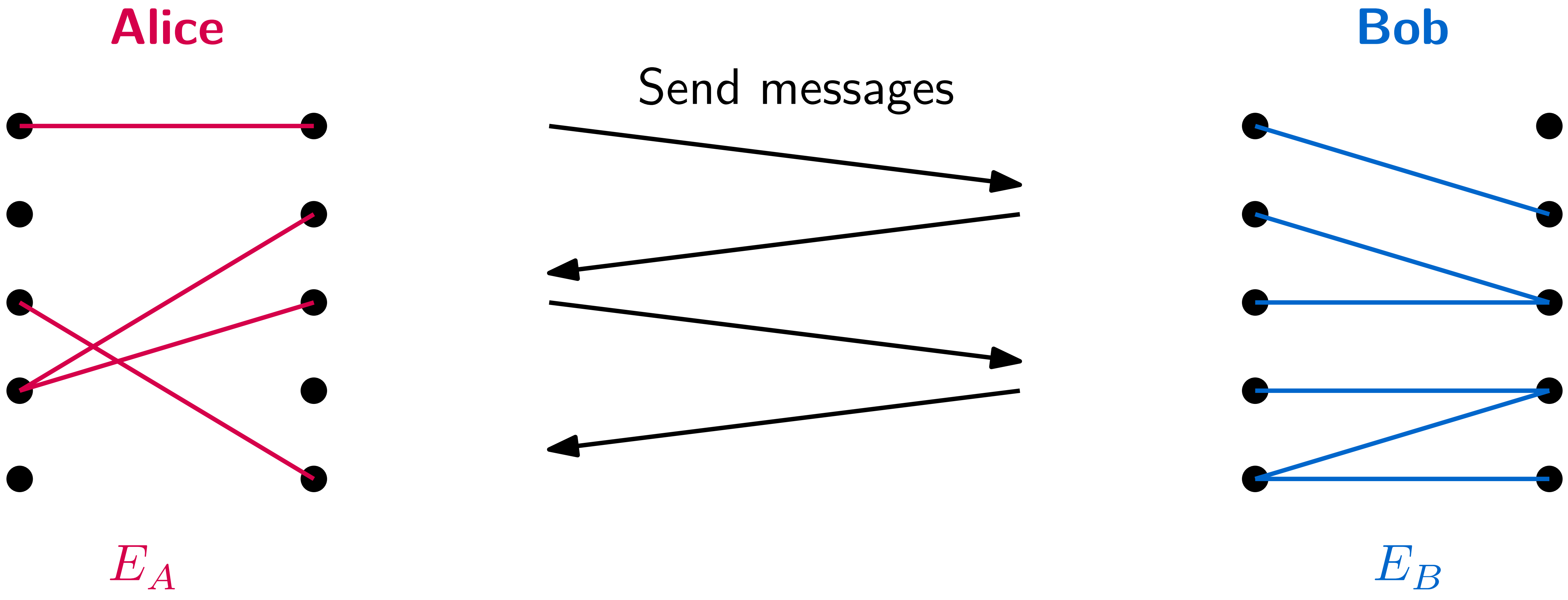
Two-Party Communication Model



Goal: Solve matching on the union of their graphs

With as few bits of communication!

Two-Party Communication Model

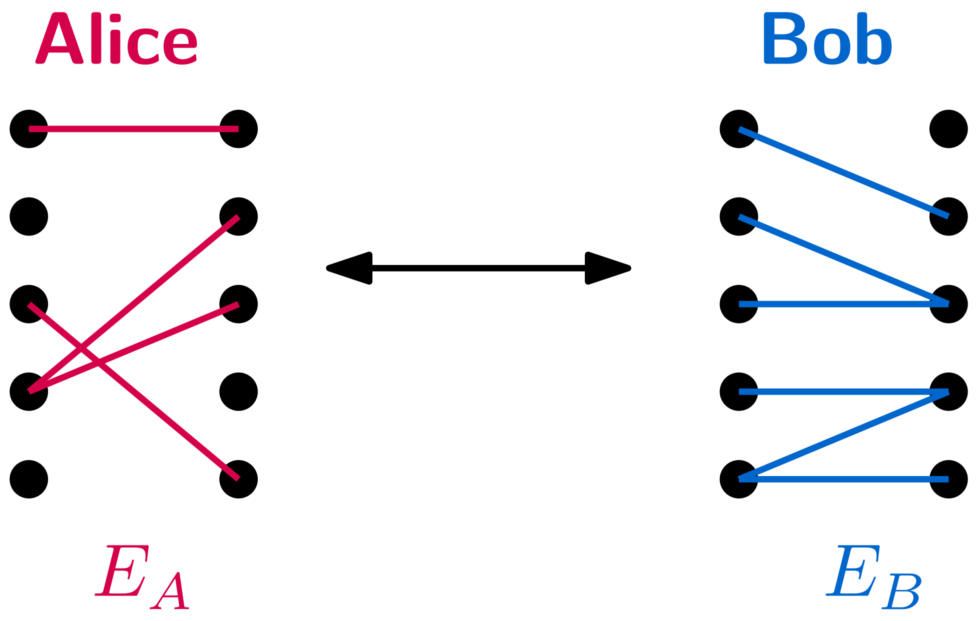


Goal: Solve matching on the union of their graphs

With as few bits of communication!

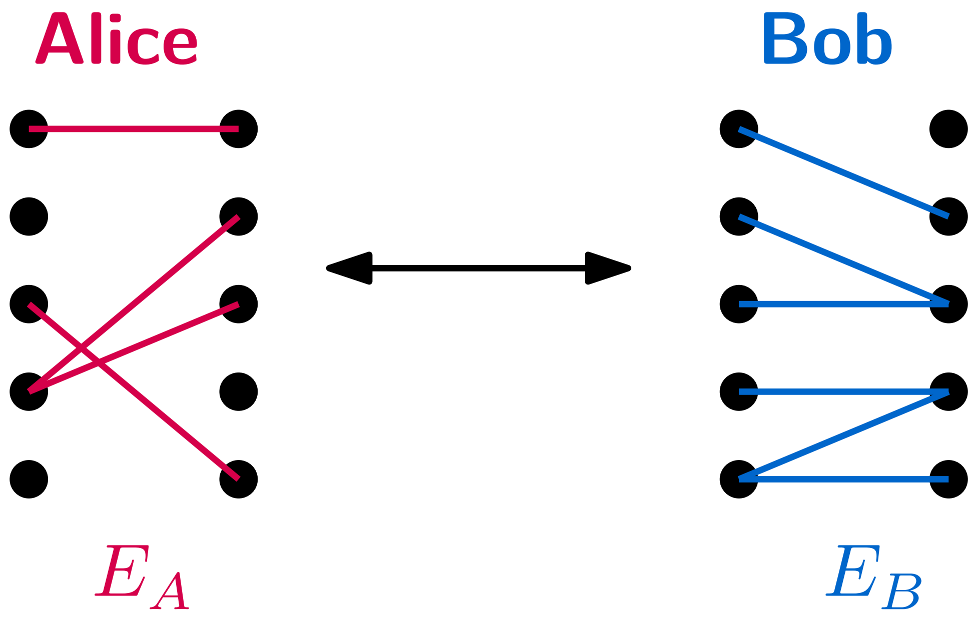
Note: Do not care about internal running time

First tries — Upper Bounds



First tries — Upper Bounds

Sending an edge: $O(\log n)$ bits



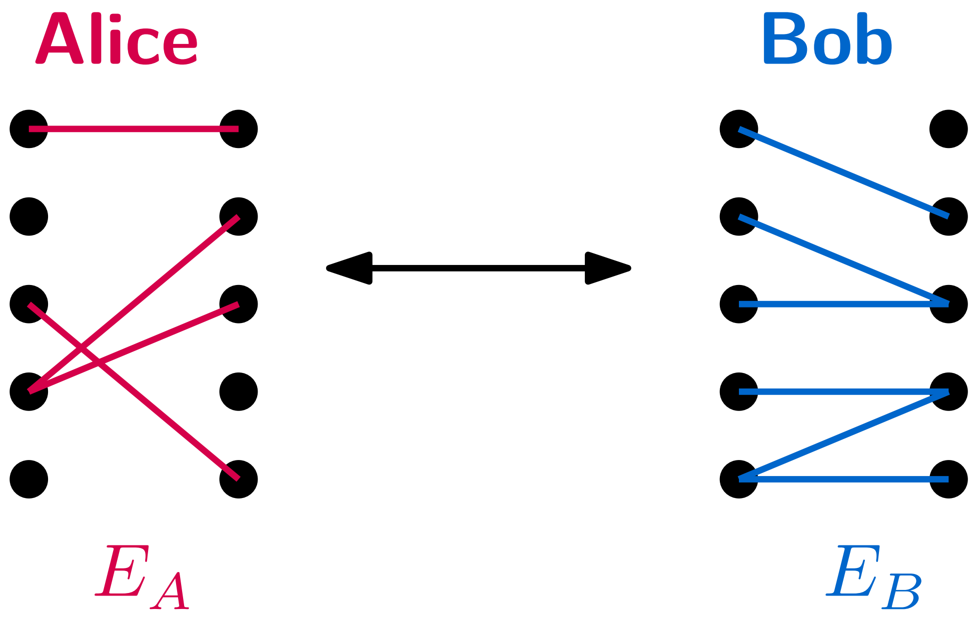
First tries — Upper Bounds

Sending an edge: $O(\log n)$ bits

Trivial Protocol:

Alice sends all her edges to Bob:

- $O(m \log n)$
- $O(n^2)$



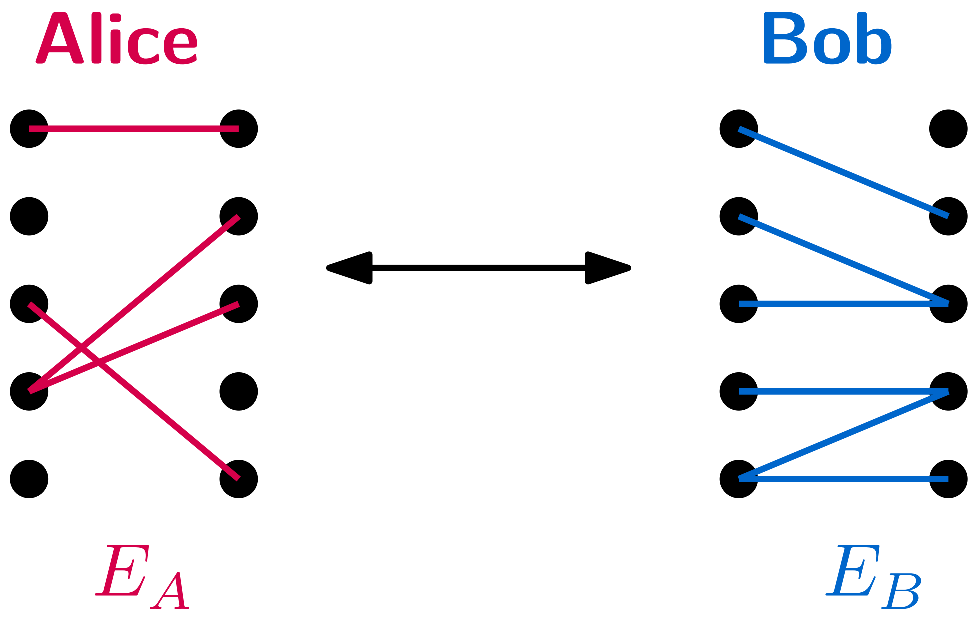
First tries — Upper Bounds

Sending an edge: $O(\log n)$ bits

Trivial Protocol:

Alice sends all her edges to Bob:

- $O(m \log n)$
- $O(n^2)$



Hopcroft-Karp: (Blocking-Flow)

- Sequential: $O(m\sqrt{n})$ running time

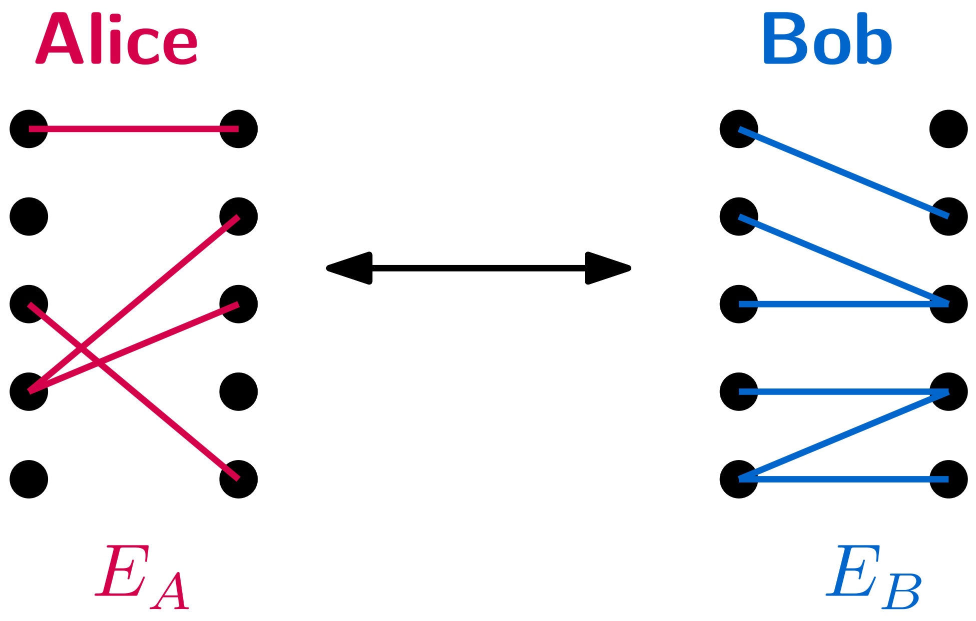
First tries — Upper Bounds

Sending an edge: $O(\log n)$ bits

Trivial Protocol:

Alice sends all her edges to Bob:

- $O(m \log n)$
- $O(n^2)$



Hopcroft-Karp: (Blocking-Flow)

- Sequential: $O(m\sqrt{n})$ running time
- Communication: $O(n\sqrt{n} \log n)$ bits

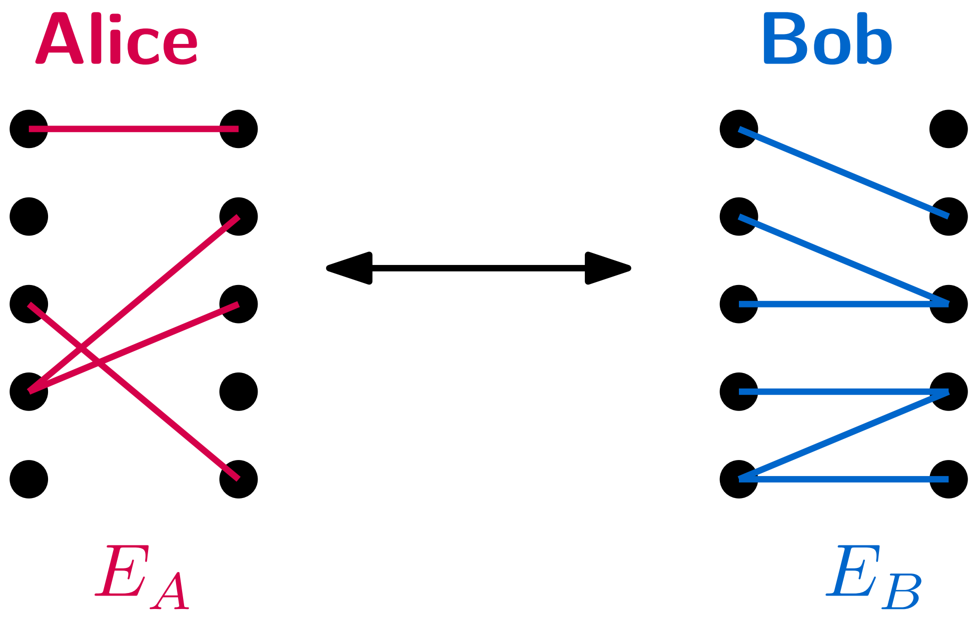
First tries — Upper Bounds

Sending an edge: $O(\log n)$ bits

Trivial Protocol:

Alice sends all her edges to Bob:

- $O(m \log n)$
- $O(n^2)$



Hopcroft-Karp: (Blocking-Flow)

- Sequential: $O(m\sqrt{n})$ running time
- Communication: $O(n\sqrt{n} \log n)$ bits

Idea: BFS / DFS need only $O(n \log n)$ bits of communication

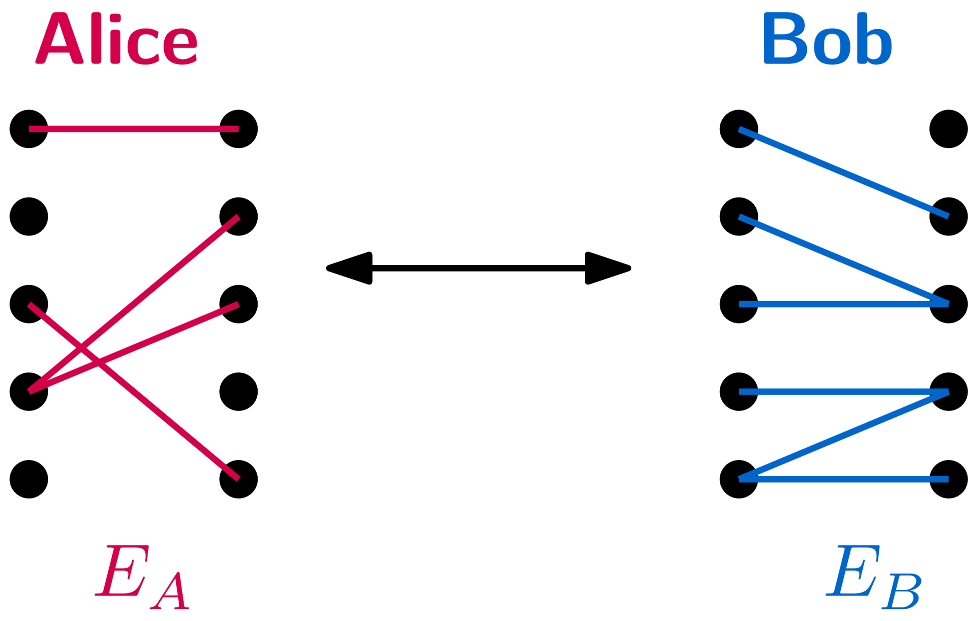
First tries — Upper Bounds

Sending an edge: $O(\log n)$ bits

Trivial Protocol:

Alice sends all her edges to Bob:

- $O(m \log n)$
- $O(n^2)$



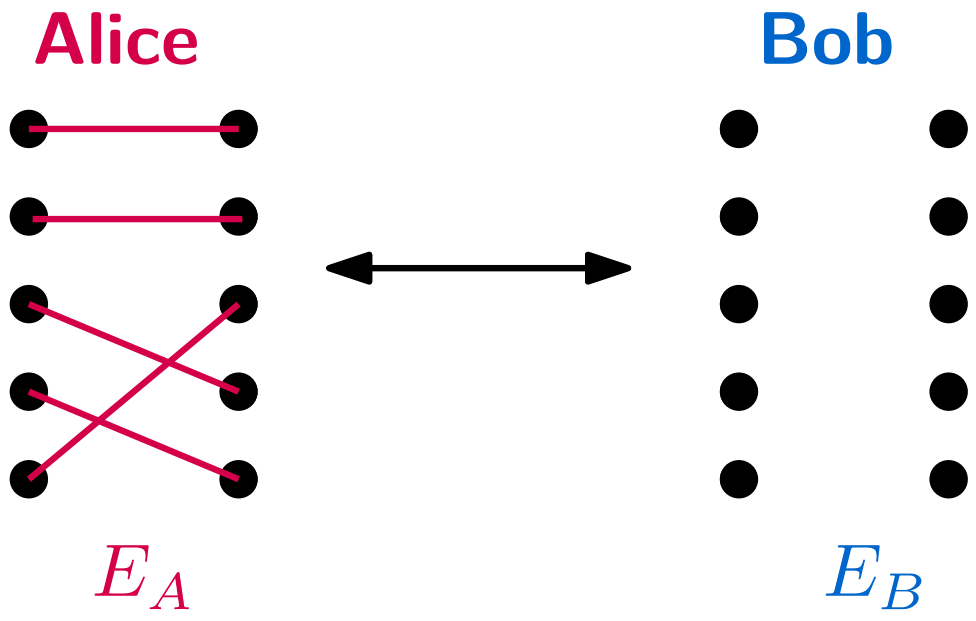
Hopcroft-Karp: (Blocking-Flow)

- Sequential: $O(m\sqrt{n})$ running time
- Communication: $O(n\sqrt{n}\log n)$ bits

Idea: BFS / DFS need only $O(n \log n)$ bits of communication

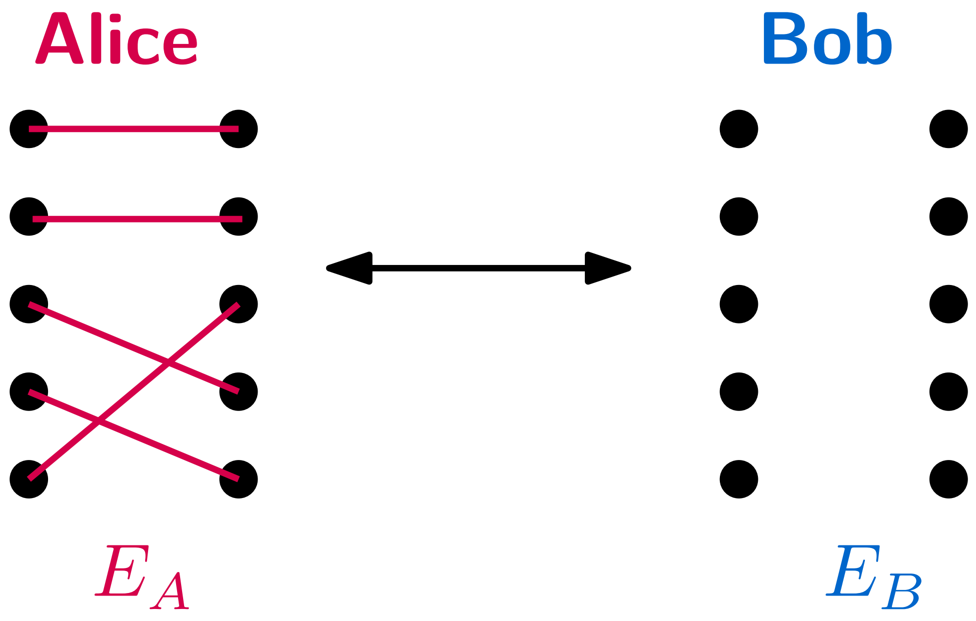
Converting $O(m^{1+o(1)})$ sequential $\longrightarrow$ $O(n^{1+o(1)})$ communication seems difficult

Lower Bounds



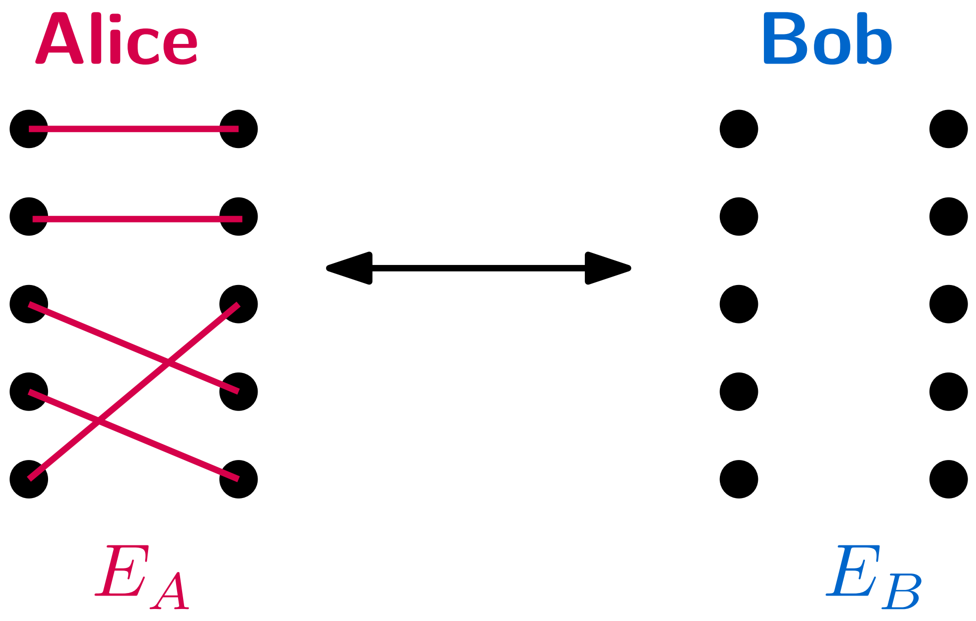
Lower Bounds

If Bob needs to output the matching:
$\Omega(n \log n)$ bits lower bound



Lower Bounds

If Bob needs to output the matching:
$\Omega(n \log n)$ bits lower bound



Theorem: [HMT'88]
$\Omega(n \log n)$ bits are needed to output the **size** of the maximum matching

$\uparrow$ only deterministic

$\Omega(n)$ randomized

$\Omega(n \log n)$ $O(n\sqrt{n} \log n)$

Major Question†**:** What is the Communication Complexity of Bipartite Matching?

† [Hajnal, Maass, Turan STOC'88];[Ivanyos, Klauck, Lee, Santha, de Wolf FSTTCS'12]; [Dobzinski, Nisan, Oren STOC'14]; [Nisan SODA'21]; [Beniamini, Nisan STOC'21]; [Zhang ICALP'04]

$\Omega(n \log n)$ $O(n\sqrt{n} \log n)$

Major Question†**:** What is the Communication Complexity of Bipartite Matching?

Main Result:
One can solve bipartite matching in $O(n \log^2 n)$ bits of communication.

† [Hajnal, Maass, Turan STOC'88];[Ivanyos, Klauck, Lee, Santha, de Wolf FSTTCS'12]; [Dobzinski, Nisan, Oren STOC'14]; [Nisan SODA'21]; [Beniamini, Nisan STOC'21]; [Zhang ICALP'04]

$\Omega(n \log n)$ $O(n\sqrt{n} \log n)$

Major Question†**:** What is the Communication Complexity of Bipartite Matching?

Main Result:
One can solve bipartite matching in $O(n \log^2 n)$ bits of communication.

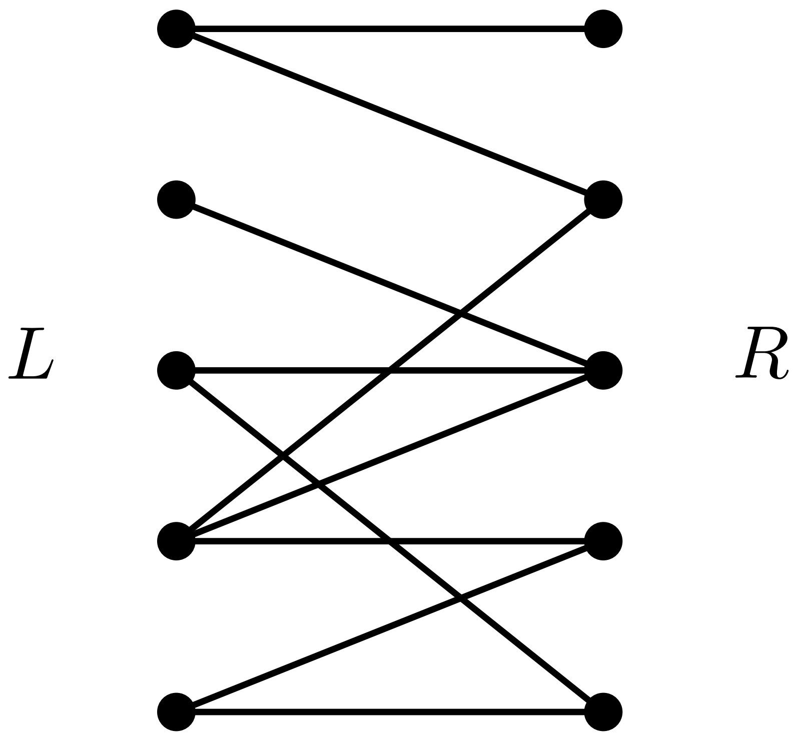
Highlights:

- Follow from **simple** applications of known techniques (cutting planes method)
- Very slow runtime, but efficient communication
- Only "finds" $O(n \log n)$ edges

† [Hajnal, Maass, Turan STOC'88];[Ivanyos, Klauck, Lee, Santha, de Wolf FSTTCS'12]; [Dobzinski, Nisan, Oren STOC'14]; [Nisan SODA'21]; [Beniamini, Nisan STOC'21]; [Zhang ICALP'04]

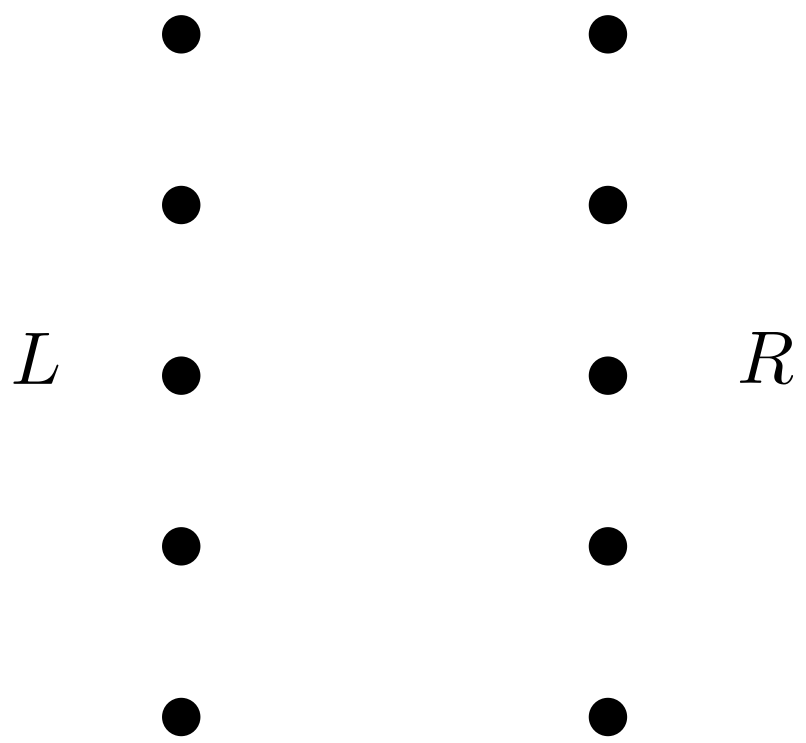
Query Models

- Hidden biparite graph $G = (L \cup R, E)$



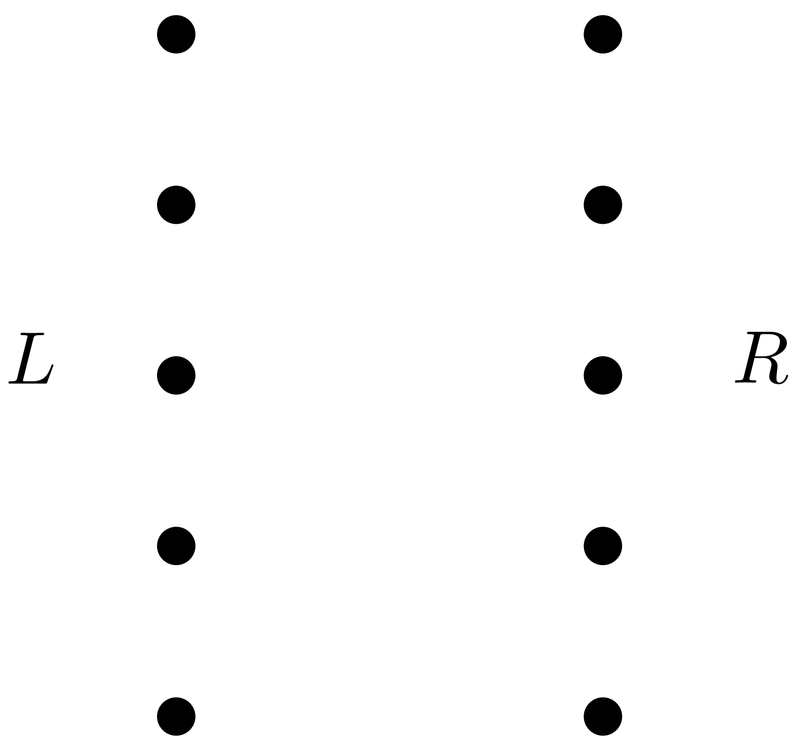
Query Models

- Hidden biparite graph $G = (L \cup R, E)$



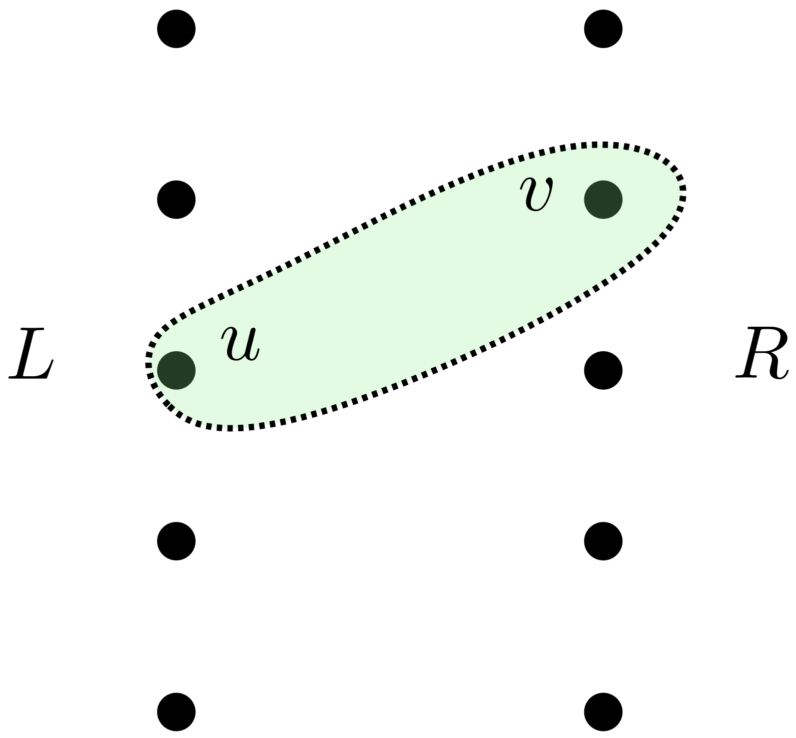
Query Models

- Hidden biparite graph $G = (L \cup R, E)$
- Query access:



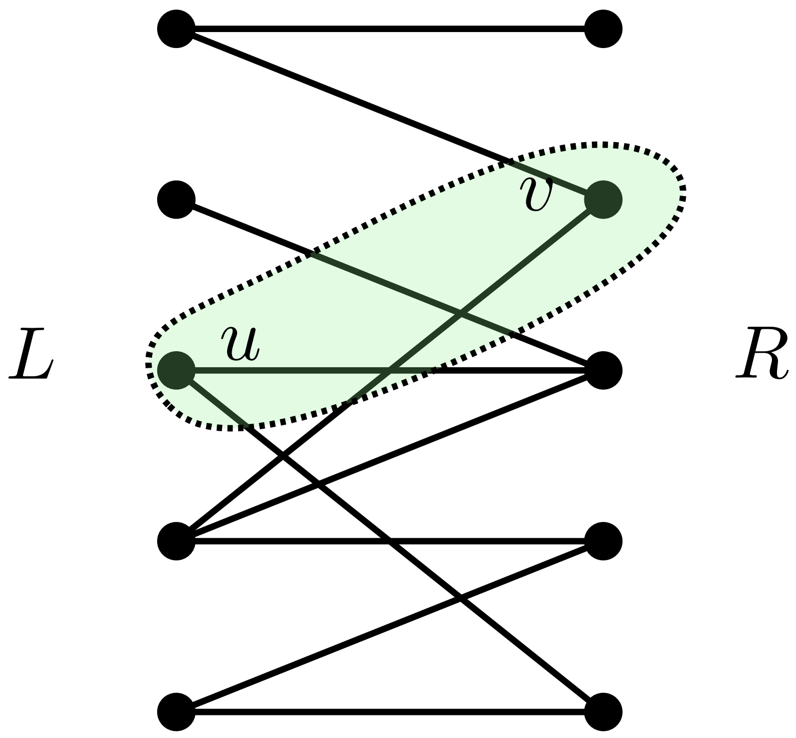
Query Models

- Hidden biparite graph $G = (L \cup R, E)$
- Query access:
  - Edge-Query: "Is $(u, v) \in E$?" "NO"



Query Models

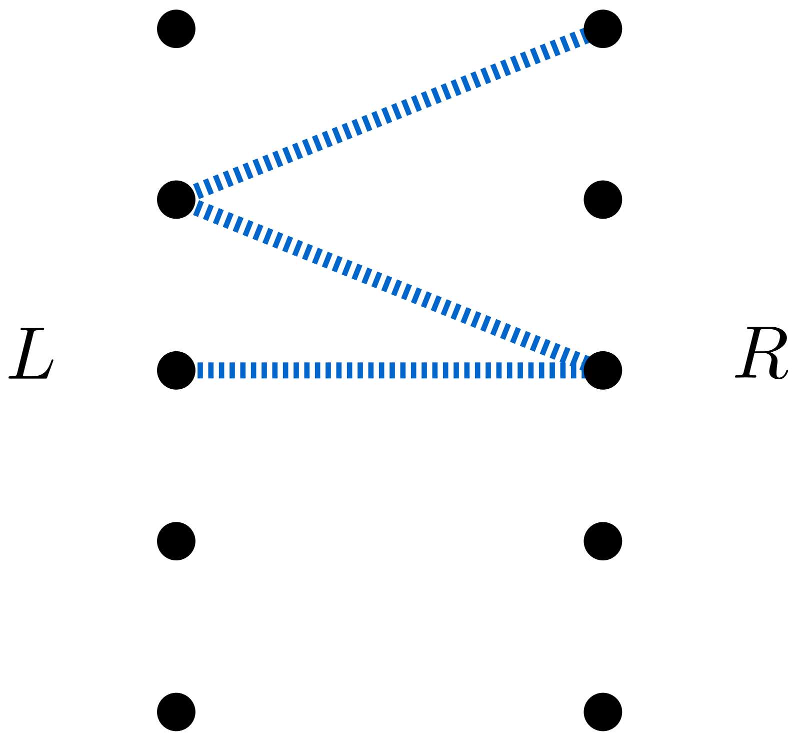
- Hidden biparite graph $G = (L \cup R, E)$
- Query access:
  - Edge-Query: "Is $(u, v) \in E$?" "NO"



Query Models

$(S \subseteq L \times R)$

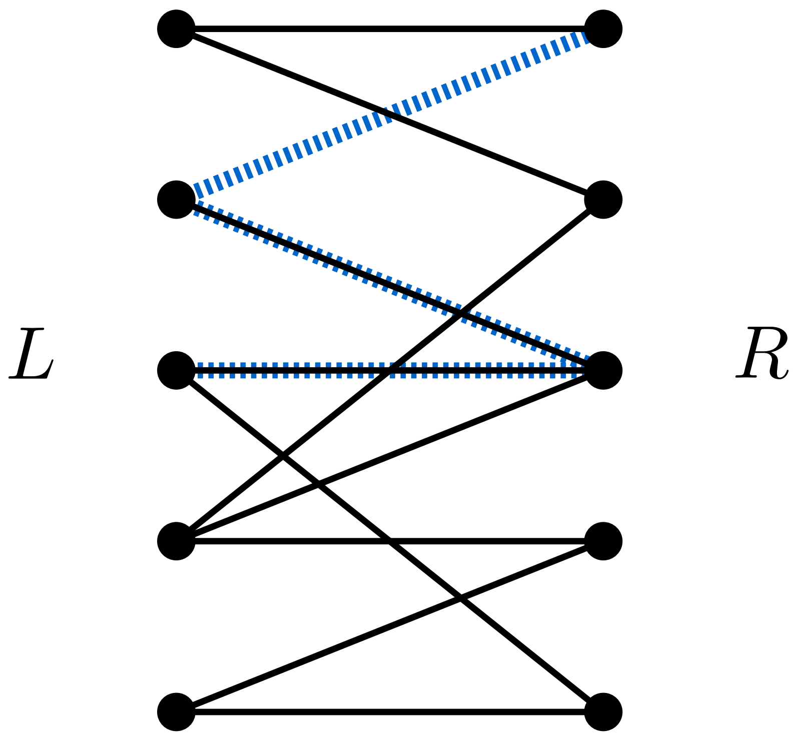
- Hidden biparite graph $G = (L \cup R, E)$
- Query access:
  - Edge-Query: "Is $(u, v) \in E$?"
  - OR-Query: "Is $|S \cap E| \geq 1$?" "YES"
  - XOR-Query: "Is $|S \cap E|$ odd?" "NO"
  - AND-Query: "Is $|S \cap E| = |\mathrm{S}|$" "NO"



Query Models

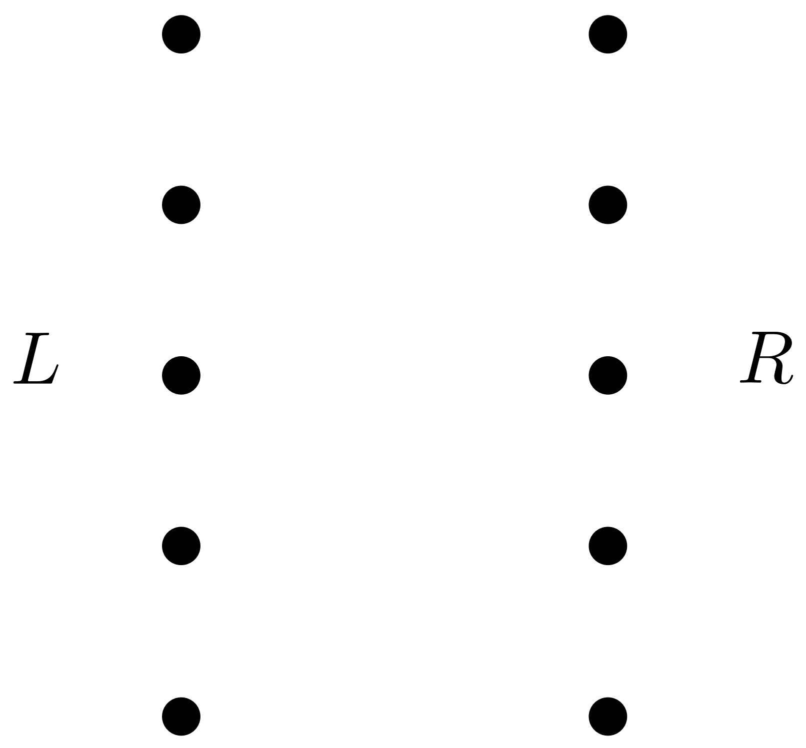
- Hidden biparite graph $G = (L \cup R, E)$
- Query access:
  - Edge-Query: "Is $(u, v) \in E$?"
  - OR-Query: "Is $|S \cap E| \geq 1$?" "YES"
  - XOR-Query: "Is $|S \cap E|$ odd?" "NO"
  - AND-Query: "Is $|S \cap E| = |\mathrm{S}|$" "NO"

$(S \subseteq L \times R)$



Query Models

- Hidden biparite graph $G = (L \cup R, E)$
- Query access:
  - Edge-Query: "Is $(u, v) \in E$?"
  - OR-Query: "Is $|S \cap E| \geq 1$?"
  - XOR-Query: "Is $|S \cap E|$ odd?"
  - AND-Query: "Is $|S \cap E| = |\mathrm{S}|$"
  - Quantum-Edge-Query



Query Models

- Hidden biparite graph $G = (L \cup R, E)$
- Query access:

	Deterministic:	Randomized:
Edge-Query: "Is $(u,v) \in E$?"	$\Theta(n^2)$	$\Theta(n^2)$
OR-Query: "Is $\lvert S \cap E\rvert \geq 1$?"	$\tilde{\Omega}(n), \tilde{O}(n\sqrt{n})$	$\Omega(n), \tilde{O}(n\sqrt{n})$
XOR-Query: "Is $\lvert S \cap E\rvert$ odd?"	$\Theta(n^2)$	$\Omega(n), \tilde{O}(n\sqrt{n})$
AND-Query: "Is $\lvert S \cap E\rvert = \lvert S\rvert$"	$\Theta(n^2)$	$\Omega(n), O(n^2)$
Quantum-Edge-Query	—	$\tilde{\Omega}(n\sqrt{n}), \tilde{O}(n^{7/4})$

[Yao'88], [Zha'04], [DHHM'06], [IKLSdW'12], [LL'15], [BN'15], [Nis'15], [DNO'19], [Ben'22]

Query Models

- Hidden biparite graph $G = (L \cup R, E)$
- Query access:

	Deterministic:	Randomized:
Edge-Query: "Is $(u, v) \in E$?"	$\Theta(n^2)$	$\Theta(n^2)$
OR-Query: "Is $\lvert S \cap E\rvert \geq 1$?"	$\tilde{\Theta}(n)$	$\tilde{\Theta}(n)$
XOR-Query: "Is $\lvert S \cap E\rvert$ odd?"	$\Theta(n^2)$	$\tilde{\Theta}(n)$
AND-Query: "Is $\lvert S \cap E\rvert = \lvert S\rvert$"	$\Theta(n^2)$	$\Theta(n^2)$
Quantum-Edge-Query	—	$\tilde{\Theta}(n\sqrt{n})$

Green: new tight upper-bound!

Red: new tight lower-bound!

[Yao'88], [Zha'04], [DHHM'06], [IKLSdW'12], [LL'15], [BN'15], [Nis'15], [DNO'19], [Ben'22]

The Algorithms

Our Algorithms

Key Idea: Apply Cutting Planes Method to the Dual Vertex Cover LP.

Think "Ellipsoid Method"

Our Algorithms

Key Idea: Apply Cutting Planes Method to the Dual Vertex Cover LP.

Think "Ellipsoid Method"

- [Vempala, Wang, Woodruff SODA'20]:
  Solving general LPs in Communication Model with Cutting Planes:
  $\tilde{O}(\mathrm{dimension}^3 \cdot \#\text{bits per constraint})$ communication

Our Algorithms

Key Idea: Apply Cutting Planes Method to the Dual Vertex Cover LP.

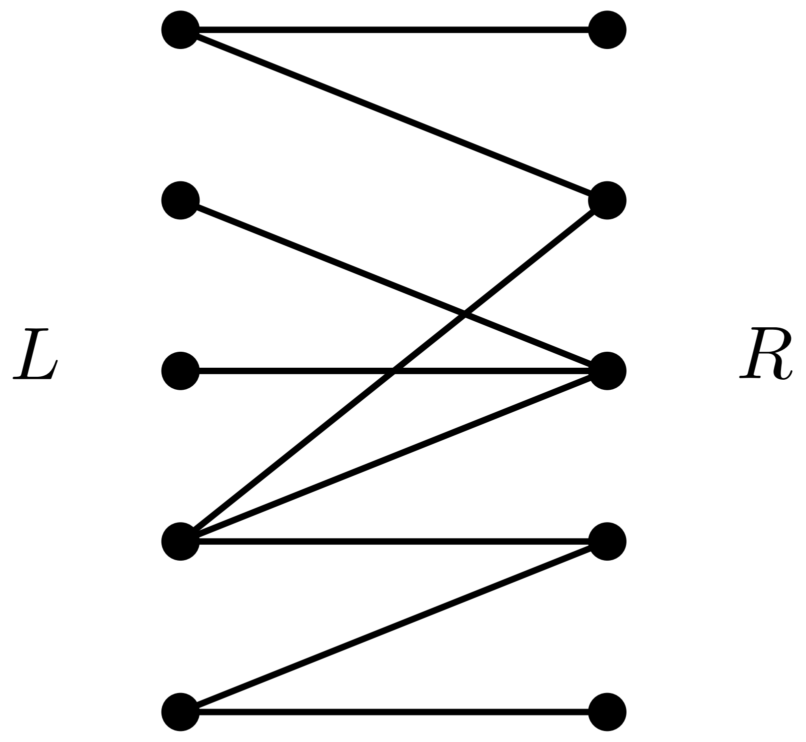
Think "Ellipsoid Method"

- [Vempala, Wang, Woodruff SODA'20]:
  Solving general LPs in Communication Model with Cutting Planes:
  $\tilde{O}(\mathrm{dimension}^3 \cdot \#\text{bits per constraint})$ communication
- Crucial properties of Dual Vertex Cover LP:
  - Low dimension (n instead of m)
  - Constraints are "short" (low support = cheap to send)
  - Volume is small
  - … but not too small

Dual: Minimum Vertex Cover

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

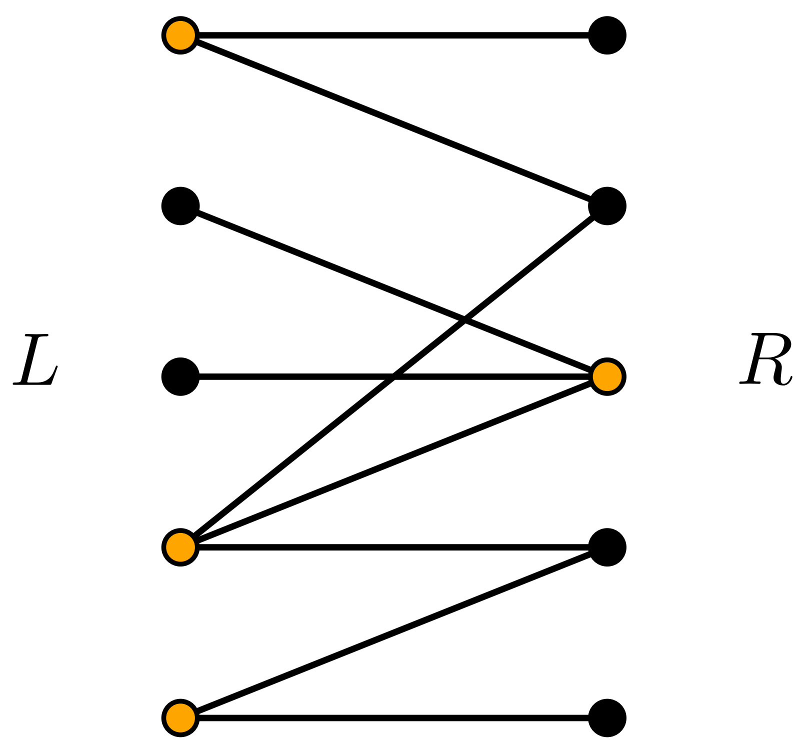
Goal: Find smallest set C of vertices covering all edges



Dual: Minimum Vertex Cover

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

Goal: Find smallest set C of vertices covering all edges

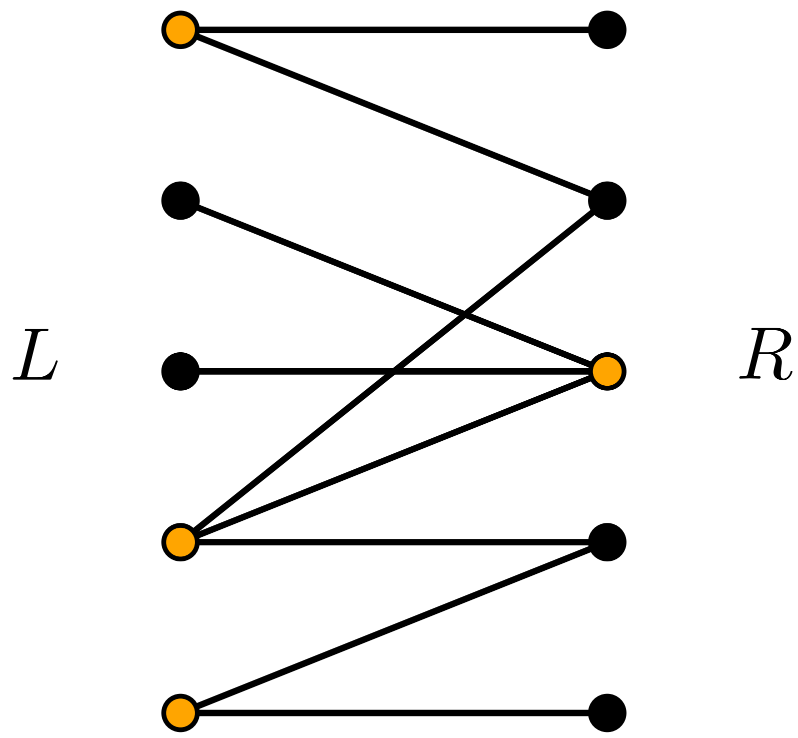


Dual: Minimum Vertex Cover

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

Goal: Find smallest set C of vertices covering all edges

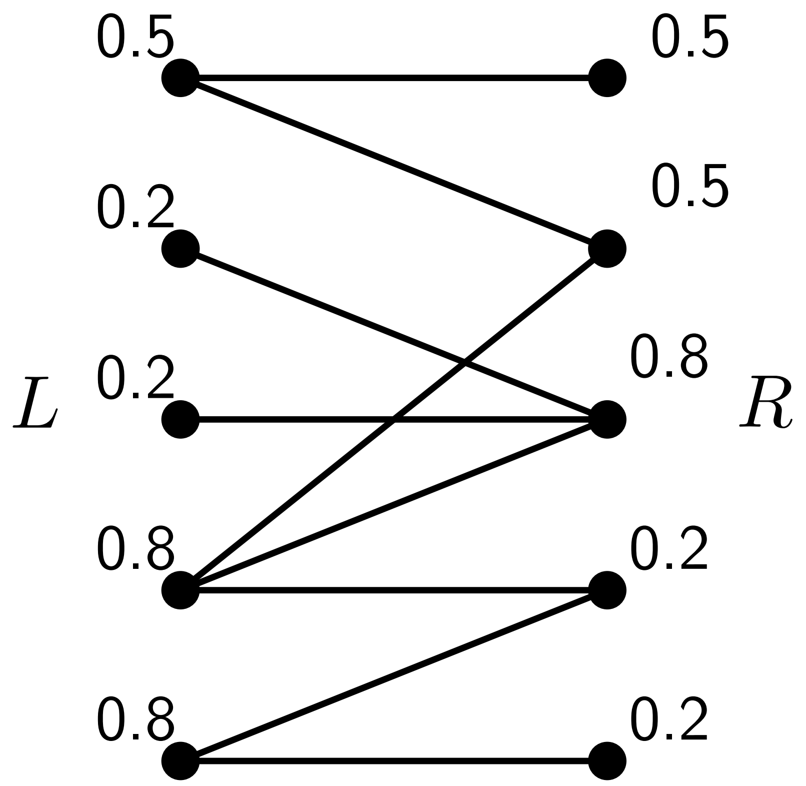
Kőnig's Theorem:
|max-matching| = |min-vertex cover|
(in bipartite graphs!)



Dual: Minimum Vertex Cover

Given: Graph $G = (L \cup R, E)$ with $|L| = |R| = n$, $|E| = m$

Goal: Find smallest set C of vertices covering all edges



Kőnig's Theorem:
|max-matching| = |min-vertex cover|
(in bipartite graphs!)

Def: Fractional vertex cover x:
$x_u + x_v \geq 1$ for all edges (u, v)

Dual Linear Program: Minimum Vertex Cover

$$
\begin{array}{lll}
\min & \sum_{v \in V} x_v & \\
\text{s.t.} & x_v + x_u \geq 1 & \forall (u,v) \in E \\
 & 0 \leq x \leq 1 &
\end{array}
\qquad (P)
$$

Dual Linear Program: Minimum Vertex Cover

$$
\begin{array}{lll}
\min & \sum_{v \in V} x_v & \\
\text{s.t.} & x_v + x_u \geq 1 \quad \forall (u,v) \in E_A & \\
 & x_v + x_u \geq 1 \quad \forall (u,v) \in E_B & (P) \\
 & 0 \leq x \leq 1 &
\end{array}
$$

Dual Linear Program: Minimum Vertex Cover

$$\begin{aligned} &\textstyle\sum_{v \in V} x_v \leq n - 1 \\ &x_v + x_u \geq 1 \quad \forall (u,v) \in E_A \\ &x_v + x_u \geq 1 \quad \forall (u,v) \in E_B \\ &0 \leq x \leq 1 \end{aligned} \qquad (P)$$

- (P) feasible $\iff$ No perfect matching exists

Dual Linear Program: Minimum Vertex Cover

$$\begin{aligned}
&\textstyle\sum_{v\in V} x_v \le n - \frac{1}{2} \\
&x_v + x_u \ge 1 \quad \forall (u,v) \in E_A \\
&x_v + x_u \ge 1 \quad \forall (u,v) \in E_B \\
&0 \le x \le 1
\end{aligned} \qquad (P)$$

- (P) feasible $\iff$ No perfect matching exists
- (P) feasible $\implies \mathrm{Vol}(P) \ge \left(\frac{1}{20n}\right)^{5n}$

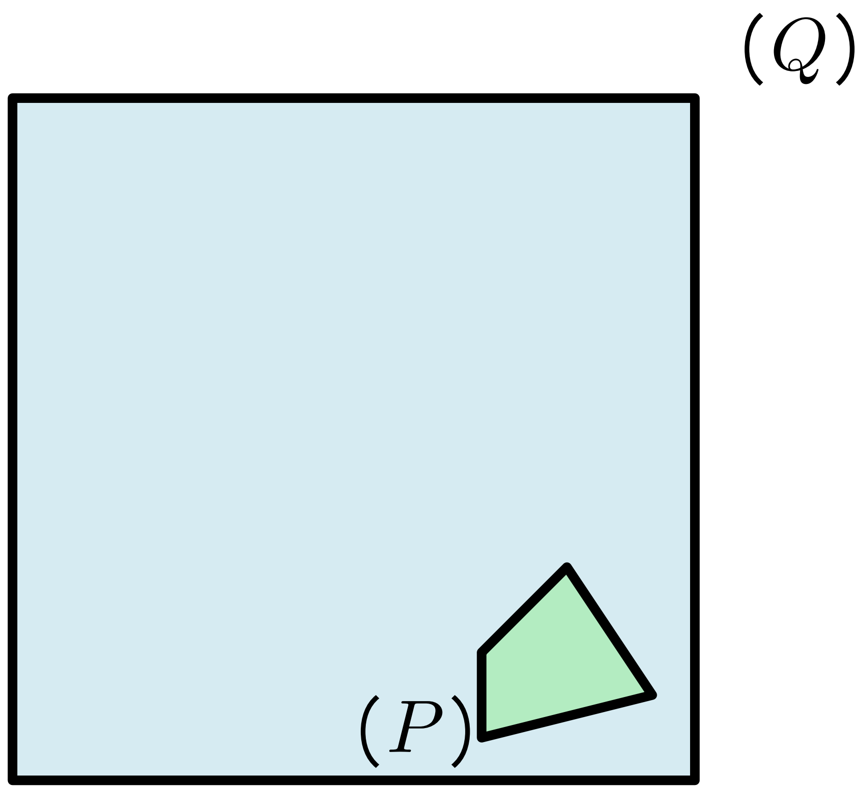
Cutting Planes Method

Separation Oracle:

Given $y \in \mathbb{R}^n$, return either:

- "y is in (P)"
- Violated hyperplane: "$c^\top x \leq d$"
  - valid for all $x \in (P)$
  - not valid for y

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



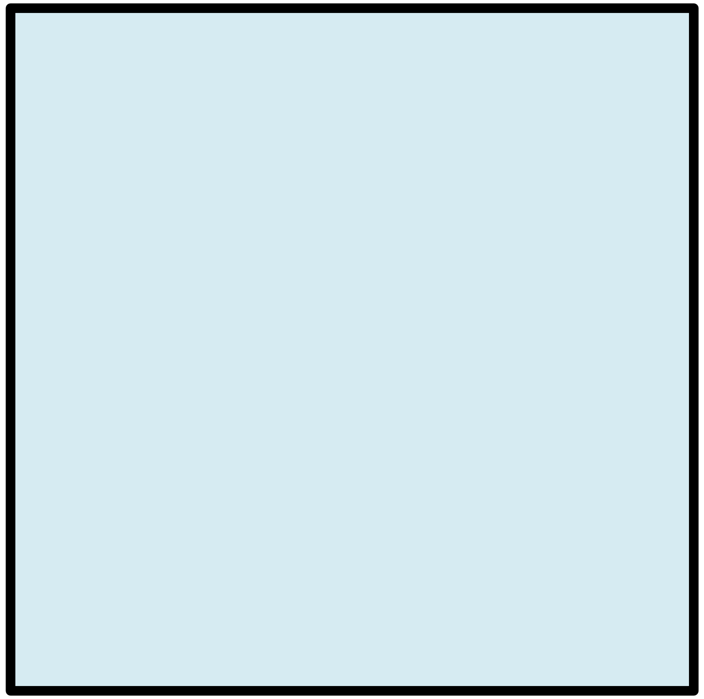
Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]

(Q)

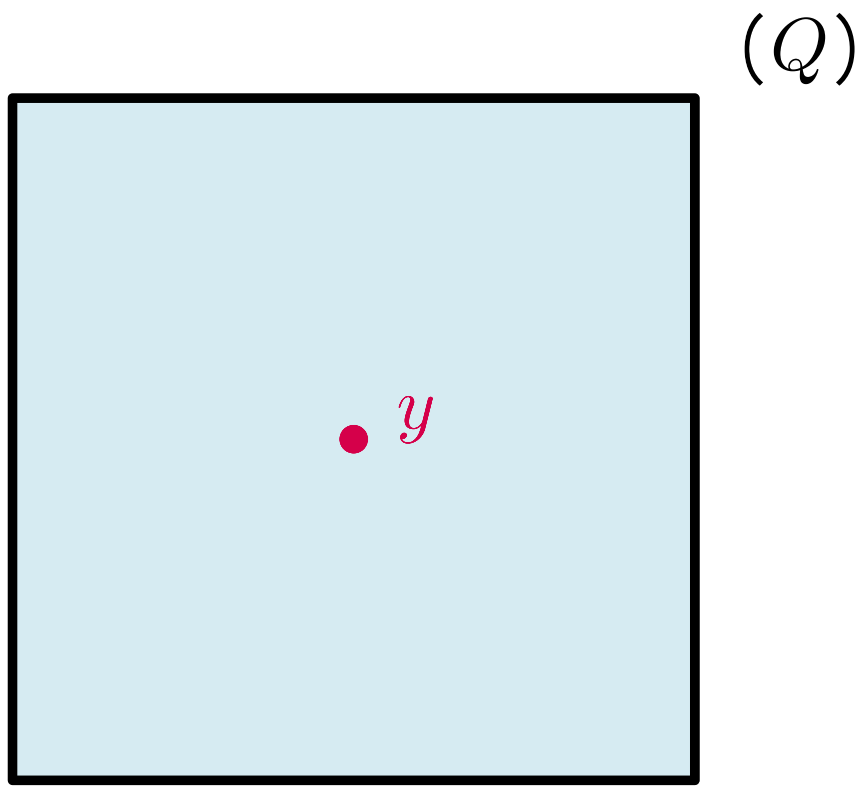


Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



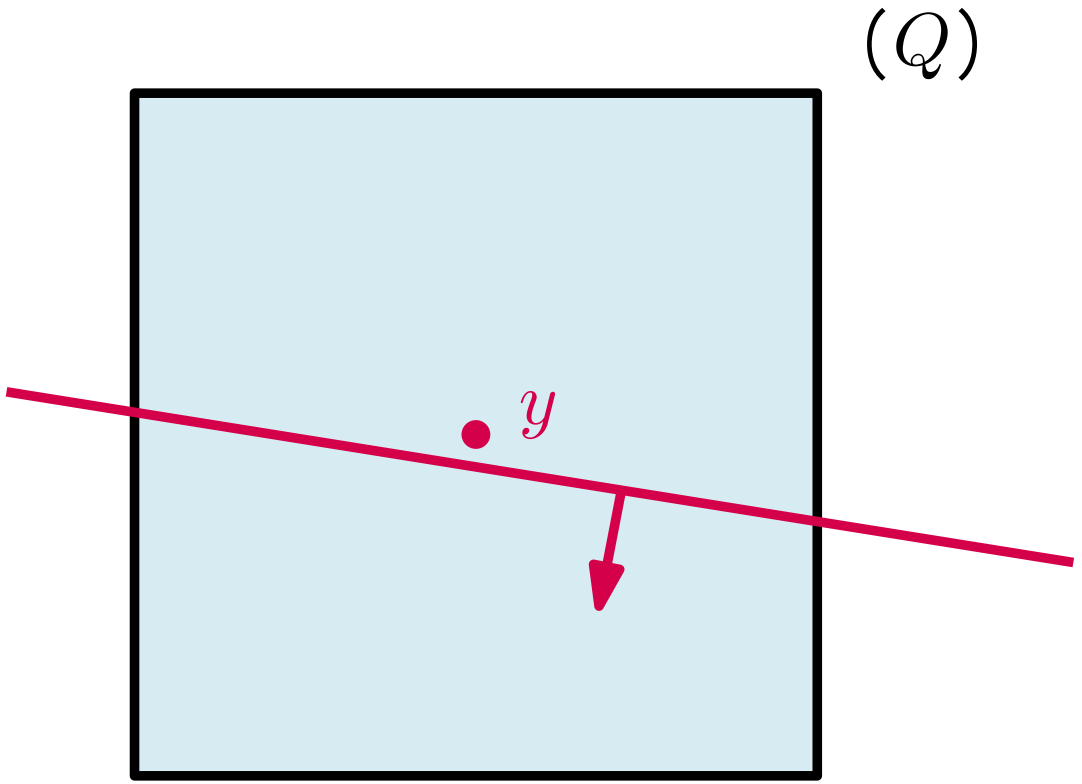
Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

1. Pick $y = \frac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



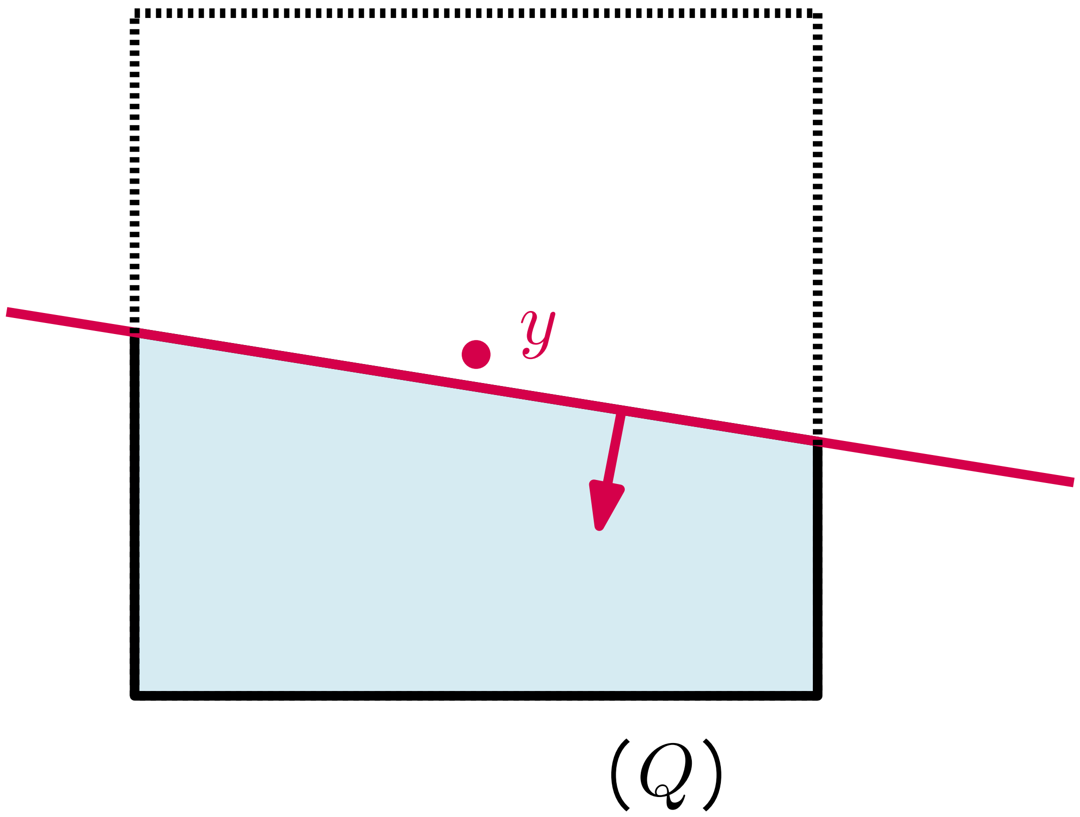
Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

1. Pick $y = \frac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)
2. Call separation oracle and update (Q)

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



Goal: Find point in unknown polytope (P)

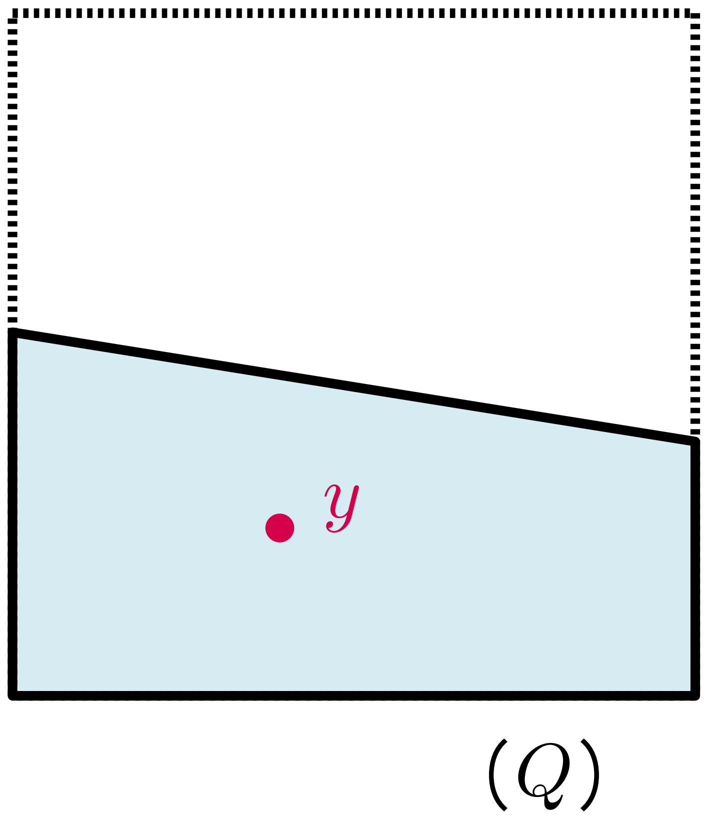
Given: (Q) containing (P)

Separation Oracle

1. Pick $y = \frac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)
2. Call separation oracle and update (Q)

Lemma: [Grünbaum]
$\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

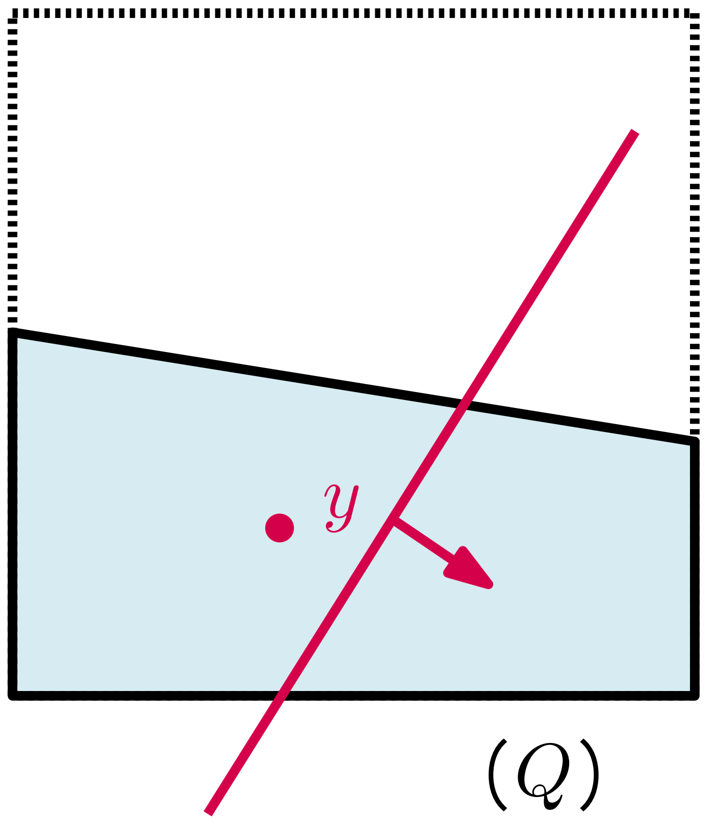
1. Pick $y = \frac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)

2. Call separation oracle and update (Q)

   Lemma: [Grünbaum]
   $\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction

3. Repeat!

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

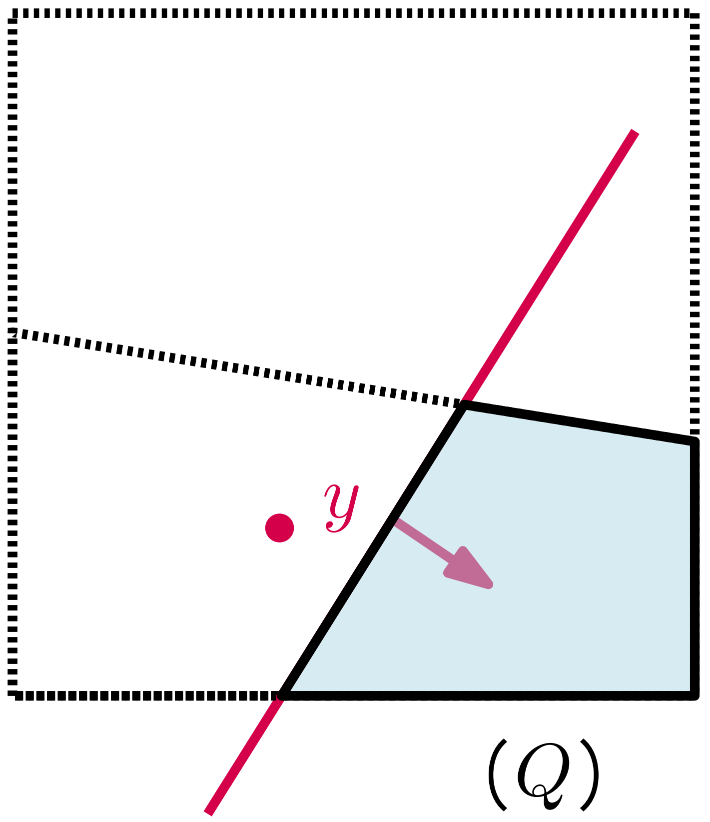
1. Pick $y = \dfrac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)

2. Call separation oracle and update (Q)

   Lemma: [Grünbaum]
   $\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction

3. Repeat!

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



Goal: Find point in unknown polytope (P)

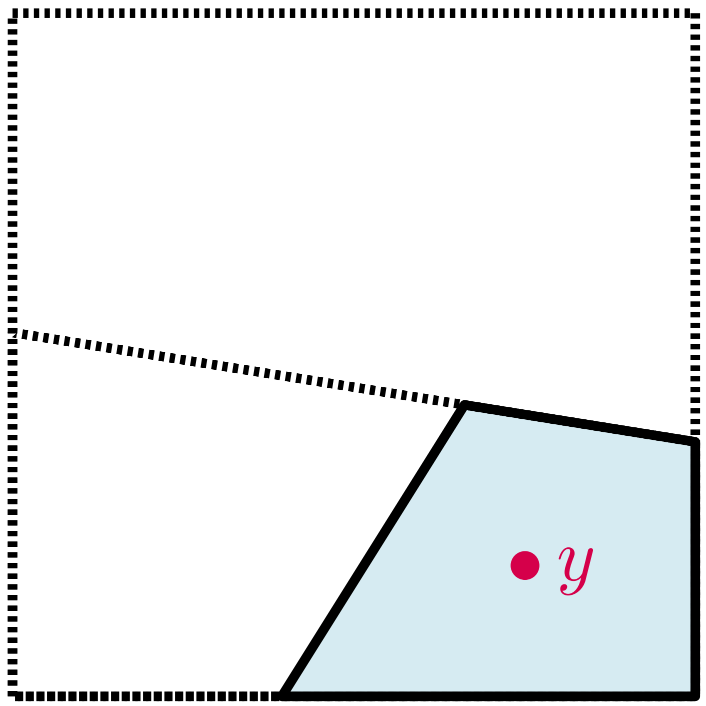
Given: (Q) containing (P)

Separation Oracle

1. Pick $y = \frac{\int_Q z\mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)
2. Call separation oracle and update (Q)

   Lemma: [Grünbaum]
   $\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction
3. Repeat!

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

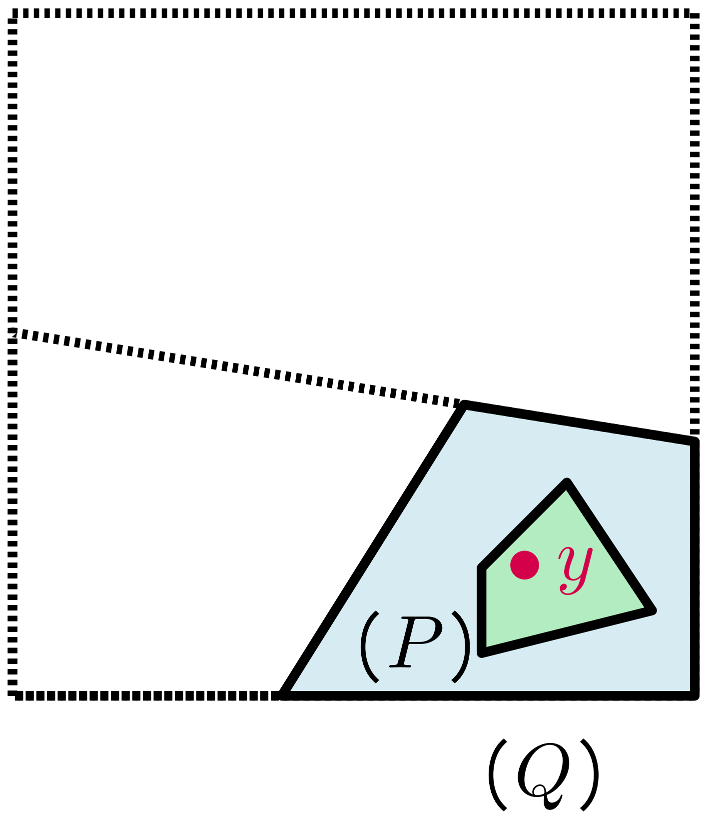
1. Pick $y = \frac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)

2. Call separation oracle and update (Q)

   Lemma: [Grünbaum]
   $\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction

3. Repeat!

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]



Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

1. Pick $y = \dfrac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)

2. Call separation oracle and update (Q)

   Lemma: [Grünbaum]
   $\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction

3. Repeat!

Center-of-gravity Cutting Planes [Levin'65] [Newman'65]

Goal: Find point in unknown polytope (P)

Given: (Q) containing (P)

Separation Oracle

$\#P$-hard to compute
We don't care! $\longrightarrow$

1. Pick $y = \frac{\int_Q z \mathrm{d}z}{\int_Q \mathrm{d}z}$ = center-of-gravity of (Q)

2. Call separation oracle and update (Q)

   Lemma: [Grünbaum]
   $\mathrm{Vol}(Q)$ decreases by a $(1 - \frac{1}{e})$-fraction

3. Repeat!

Cutting Planes for Biparite Matching

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (Q)$$

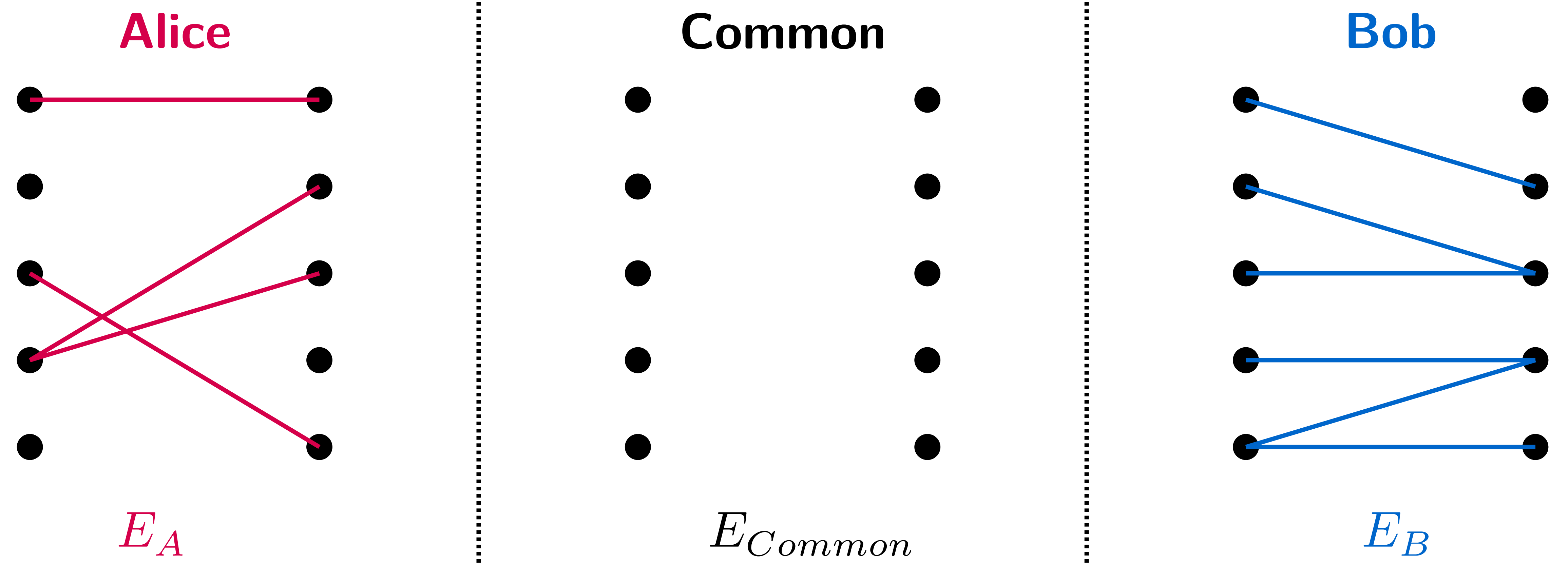
$$0 \leq x \leq 1$$

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (P)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_B$$

$$0 \leq x \leq 1$$



Cutting Planes for Biparite Matching

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (Q)$$

$$0 \leq x \leq 1$$

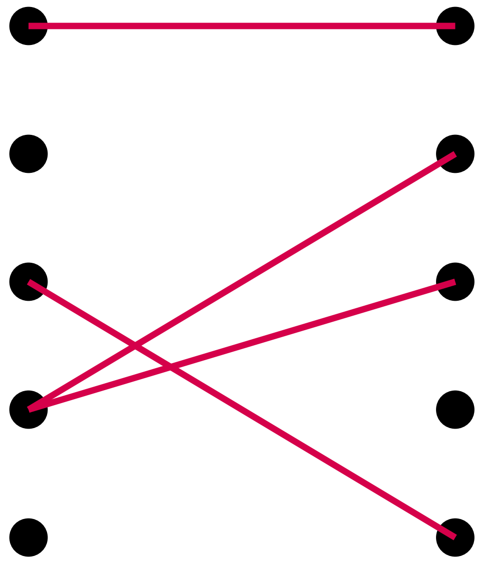
$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (P)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_B$$

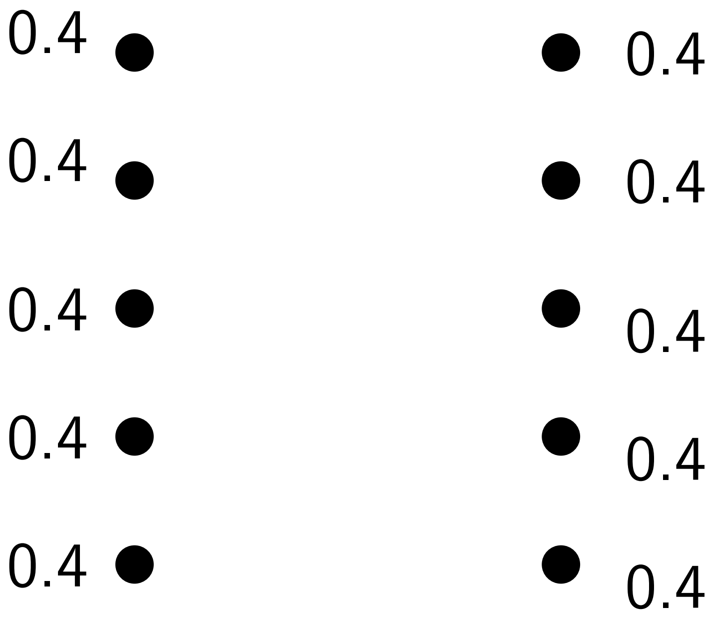
$$0 \leq x \leq 1$$

Alice



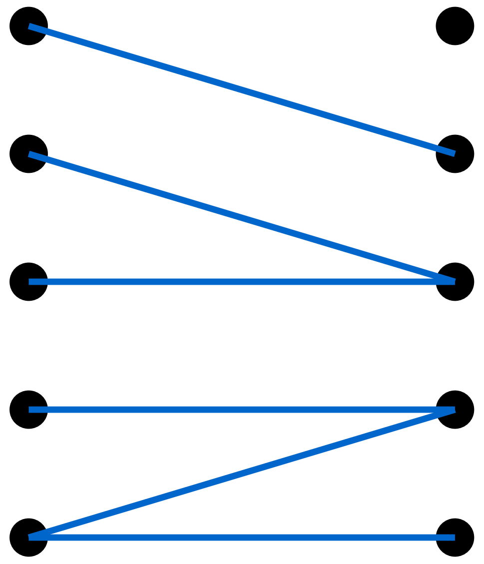
E_A

Common



E_{Common}

Bob



E_B

Cutting Planes for Biparite Matching

$$\sum_{v\in V} x_v \le n - \tfrac{1}{2} \qquad (Q)$$

$$x_v + x_u \ge 1 \quad \forall (u,v) \in E_{Common}$$

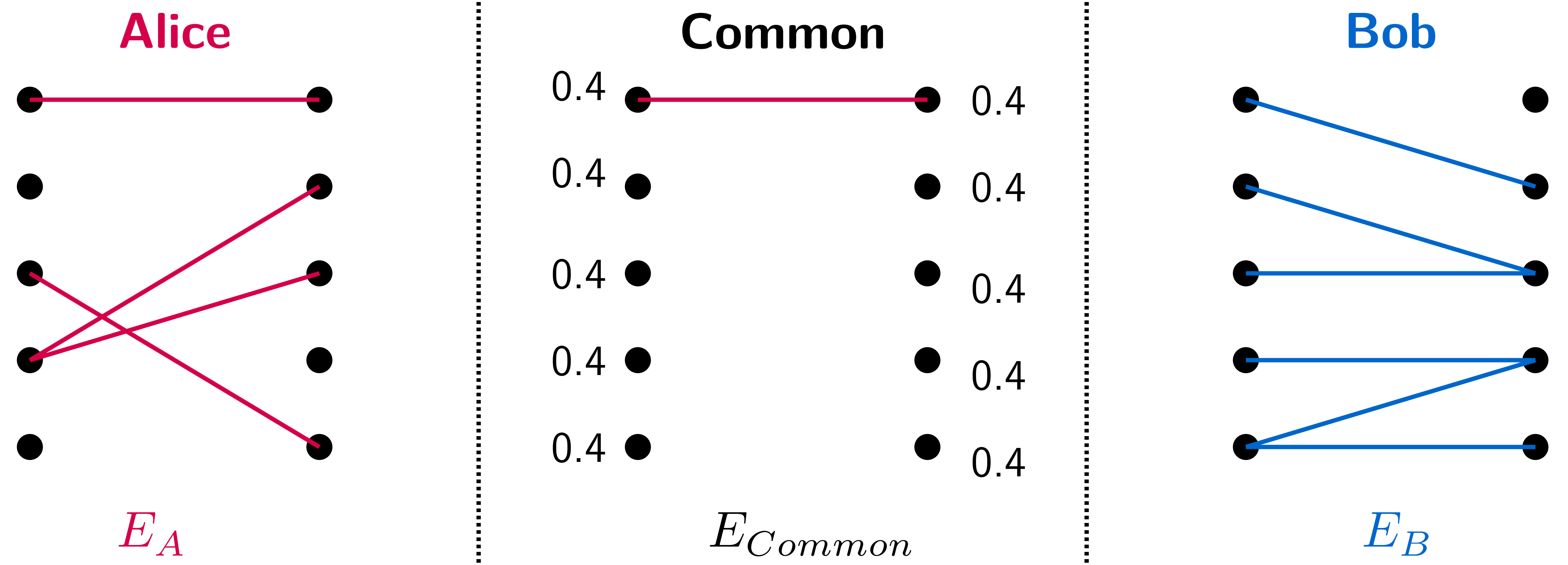
$$0 \le x \le 1$$

$$\sum_{v\in V} x_v \le n - \tfrac{1}{2} \qquad (P)$$

$$x_v + x_u \ge 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \ge 1 \quad \forall (u,v) \in E_B$$

$$0 \le x \le 1$$



Cutting Planes for Biparite Matching

$$\sum_{v \in V} x_v \le n - \frac{1}{2} \qquad (Q)$$

$$x_v + x_u \ge 1 \quad \forall (u,v) \in E_{Common}$$

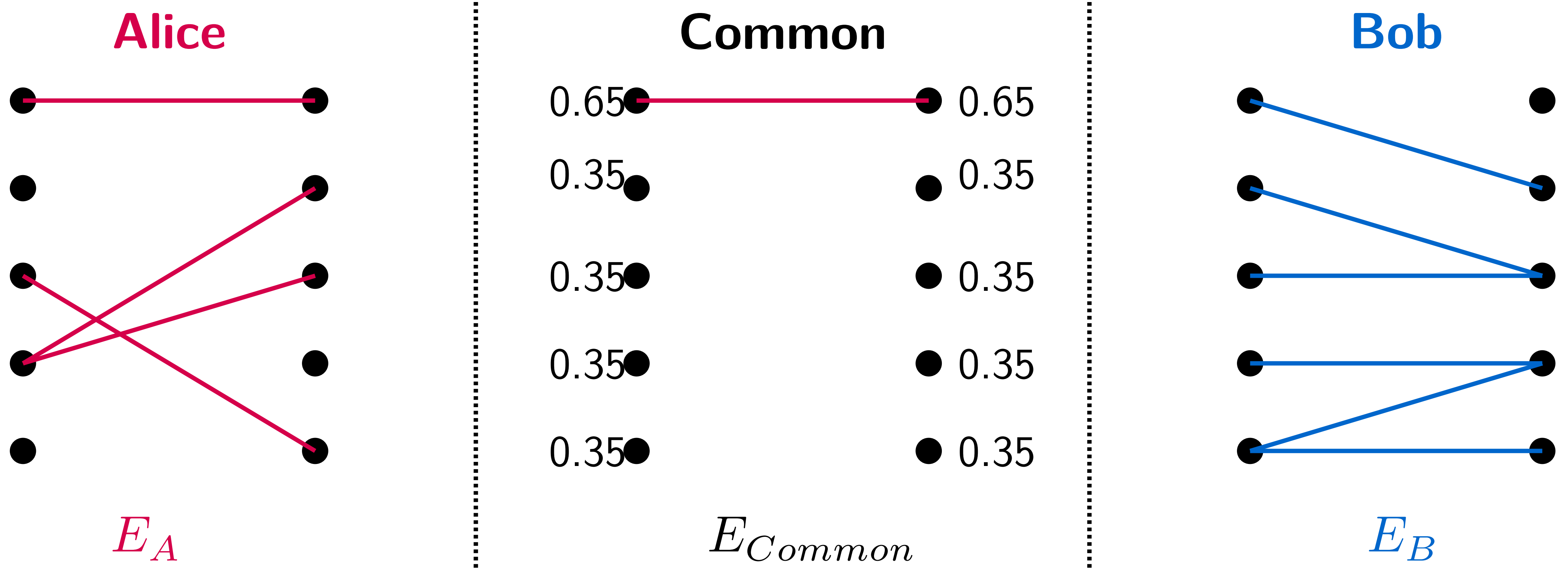
$$0 \le x \le 1$$

$$\sum_{v \in V} x_v \le n - \frac{1}{2} \qquad (P)$$

$$x_v + x_u \ge 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \ge 1 \quad \forall (u,v) \in E_B$$

$$0 \le x \le 1$$



Cutting Planes for Biparite Matching

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (Q)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_{Common}$$

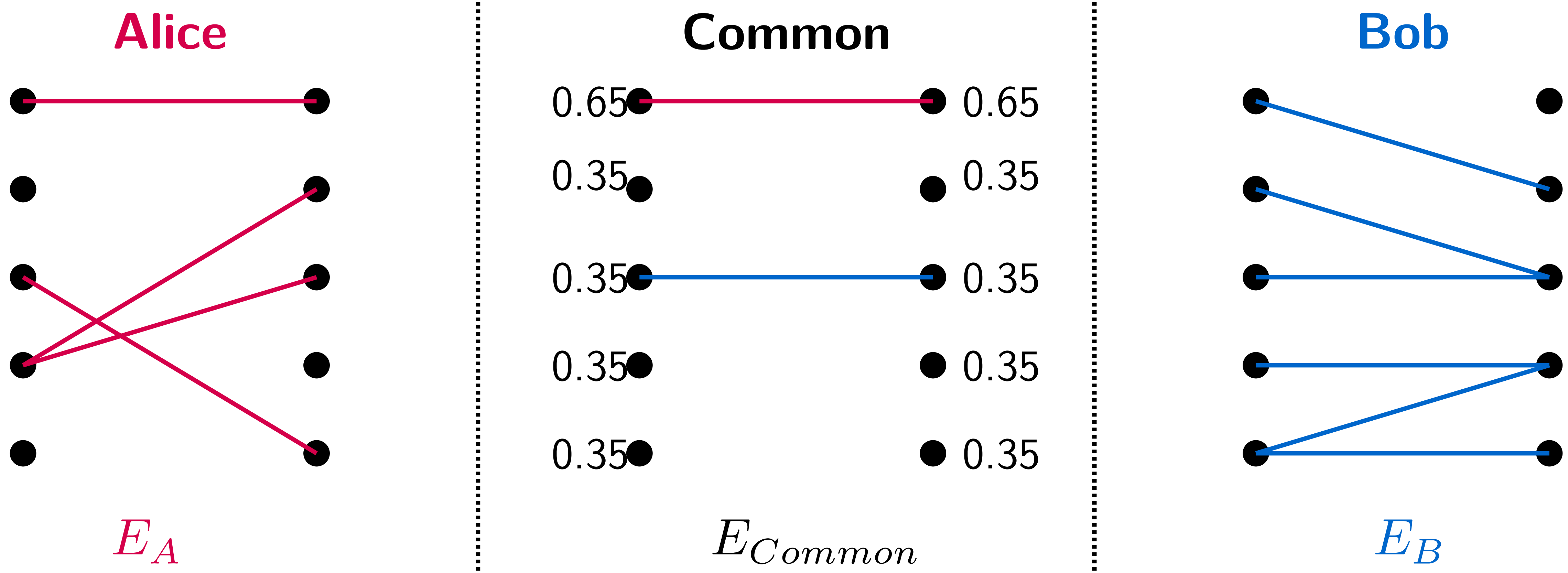
$$0 \leq x \leq 1$$

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (P)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_B$$

$$0 \leq x \leq 1$$



Cutting Planes for Biparite Matching

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (Q)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_{Common}$$

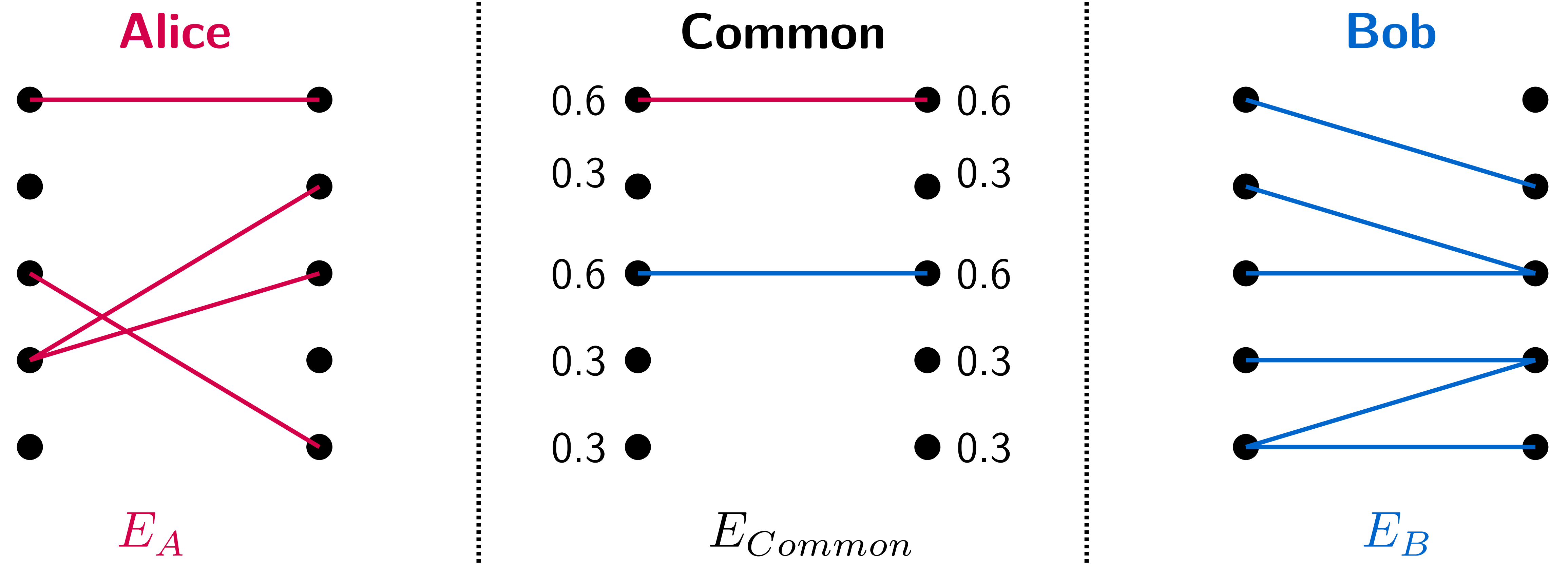
$$0 \leq x \leq 1$$

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (P)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_B$$

$$0 \leq x \leq 1$$



Cutting Planes for Biparite Matching

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (Q)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_{Common}$$

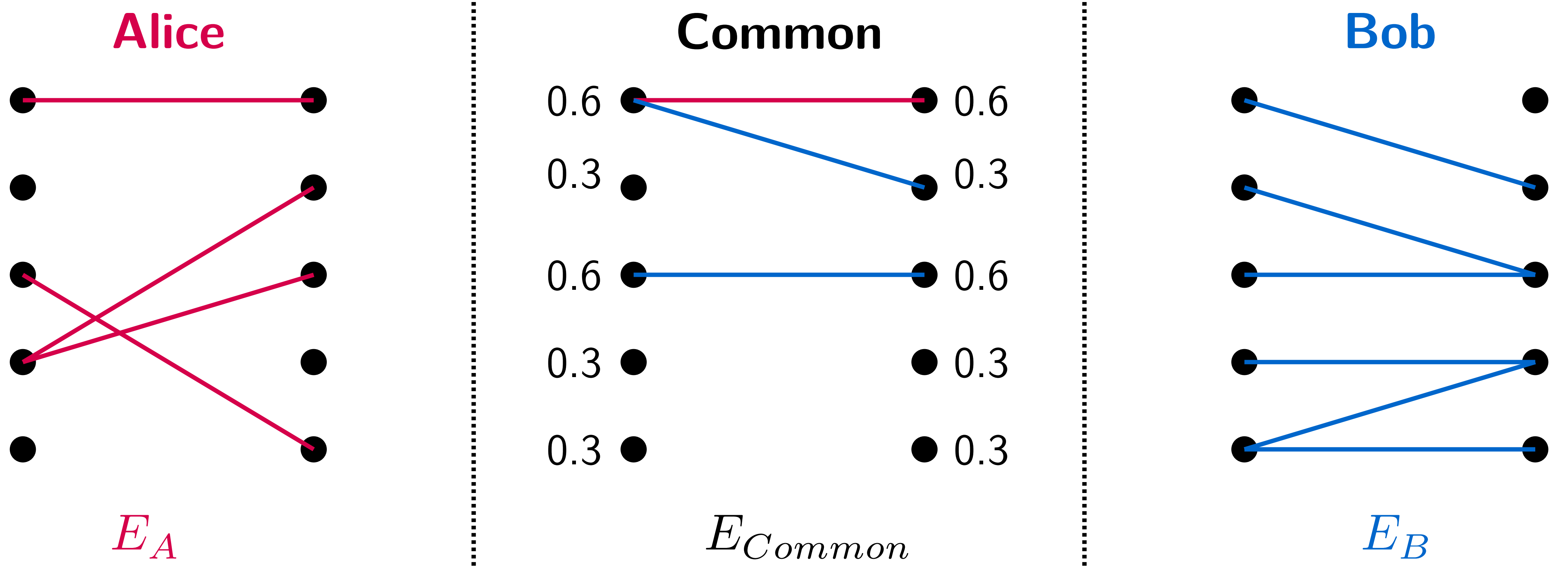
$$0 \leq x \leq 1$$

$$\sum_{v \in V} x_v \leq n - \frac{1}{2} \qquad (P)$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_A$$

$$x_v + x_u \geq 1 \quad \forall (u,v) \in E_B$$

$$0 \leq x \leq 1$$



Algorithm

- $E_{common} = \varnothing, \quad Q = \left\{x \in [0,1]^V : \sum x_v \leq n - \frac{1}{2}\right\}$
- While $vol(Q) > 0$:
  - Let c = center-of-gravity(Q) "fractional vertex cover"
  - If either Alice or Bob have an edge (u, v) violating c: add it to E_{common} and add "$x_v + x_u \geq 1$" to (Q)
  - If not, **return** c as a fractional vertex cover
- E_{common} must now contain a perfect matching.

OR-Query Algorithm

- $E_{common} = \varnothing, \quad Q = \left\{x \in [0,1]^V : \sum x_v \leq n - \frac{1}{2}\right\}$
- While $vol(Q) > 0$:
  - Let c = center-of-gravity(Q) "fractional vertex cover"
  - If either Alice or Bob have an edge (u, v) violating c: add it to E_{common} and add "$x_v + x_u \geq 1$" to (Q)
  - If not, **return** c as a fractional vertex cover
- E_{common} must now contain a perfect matching.

OR-Query Algorithm

- $E_{common} = \varnothing, \quad Q = \left\{x \in [0,1]^V : \sum x_v \le n - \frac{1}{2}\right\}$
- While $vol(Q) > 0$:
  - Let c = center-of-gravity(Q) "fractional vertex cover"
  - If either Alice or Bob have an edge (u, v) violating c: add it to E_{common} and add "$x_v + x_u \ge 1$" to (Q)
  - If not, **return** c as a fractional vertex cover
- E_{common} must now contain a perfect matching.

OR-Query Algorithm

- $E_{common} = \varnothing, \quad Q = \left\{x \in [0,1]^V : \sum x_v \leq n - \frac{1}{2}\right\}$
- While $vol(Q) > 0$:
  - Let c = center-of-gravity(Q) "fractional vertex cover"
  - Binary search with OR-queries to find violated edge in $S = \{(u,v) \in L \times R : c_u + c_v < 1\}$
  - If not, **return** c as a fractional vertex cover
- E_{common} must now contain a perfect matching.

Analysis

- Violated constraint "$x_v + x_u \geq 1$" corresponds to edges. $\implies$ $O(\log n)$ bits

Analysis

- Violated constraint "$x_v + x_u \geq 1$" corresponds to edges. $\Longrightarrow$ $O(\log n)$ bits
- Terminates when either:
  - Fractional vertex cover of size $< n$ is found. $\Longrightarrow$ no perfect matching!
  - (Q) becomes empty. $\Longrightarrow$ perfect matching!

Analysis

- Violated constraint "$x_v + x_u \geq 1$" corresponds to edges. $\Longrightarrow$ $O(\log n)$ bits
- Terminates when either:
  - Fractional vertex cover of size $< n$ is found. $\Longrightarrow$ no perfect matching!
  - (Q) becomes empty. $\Longrightarrow$ perfect matching!
- Volume:
  - Initially ≤ 1 (contained in $[0,1]^{2n}$).
  - Always $\geq \left(\frac{1}{20n}\right)^{5n}$ whenever (Q) is non-empty.

$\Longrightarrow$ $O(n \log n)$ iterations

Analysis

- Violated constraint "$x_v + x_u \geq 1$" corresponds to edges. $\Longrightarrow$ $O(\log n)$ bits
- Terminates when either:
  - Fractional vertex cover of size $< n$ is found. $\Longrightarrow$ no perfect matching!
  - (Q) becomes empty. $\Longrightarrow$ perfect matching!
- Volume:
  - Initially ≤ 1 (contained in $[0, 1]^{2n}$).
  - Always $\geq \left(\frac{1}{20n}\right)^{5n}$ whenever (Q) is non-empty.

$\Longrightarrow$ $O(n \log n)$ iterations

Main Result:
One can solve bipartite matching in $O(n \log^2 n)$ bits of communication.

Extensions

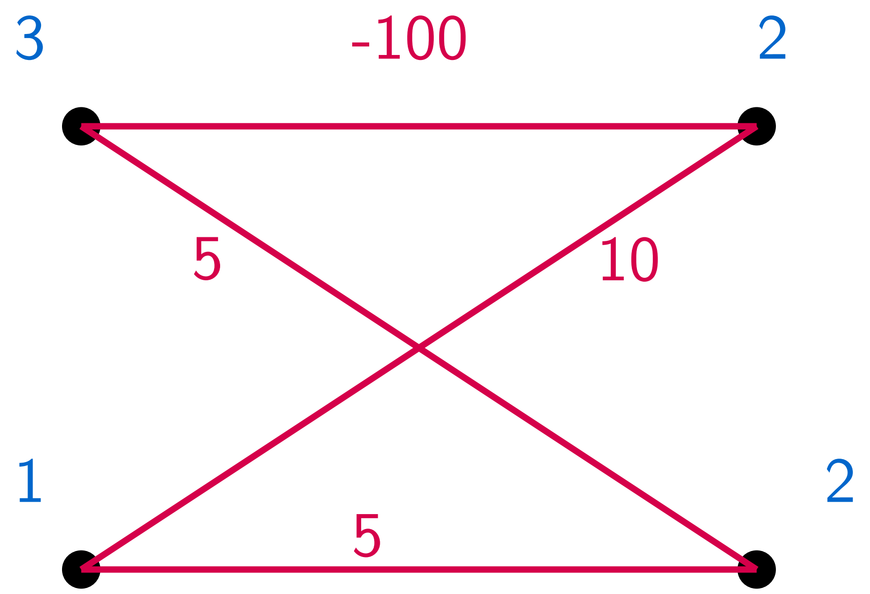
Weights and Demands!

$$
\begin{array}{lll}
\min & \sum_{v \in V} x_v & \\
\text{s.t.} & x_v + x_u \geq 1 & \forall (u,v) \in E_A \\
& x_v + x_u \geq 1 & \forall (u,v) \in E_B \\
& 0 \leq x \leq 1 &
\end{array}
$$

Weights and Demands!

$$\begin{array}{llll} \min & \sum_{v \in V} b_v x_v & & \\ \text{s.t.} & x_v + x_u \geq c_{uv} & \forall (u,v) \in E_A \\ & x_v + x_u \geq c_{uv} & \forall (u,v) \in E_B \\ & 0 \leq x \leq W & \end{array}$$

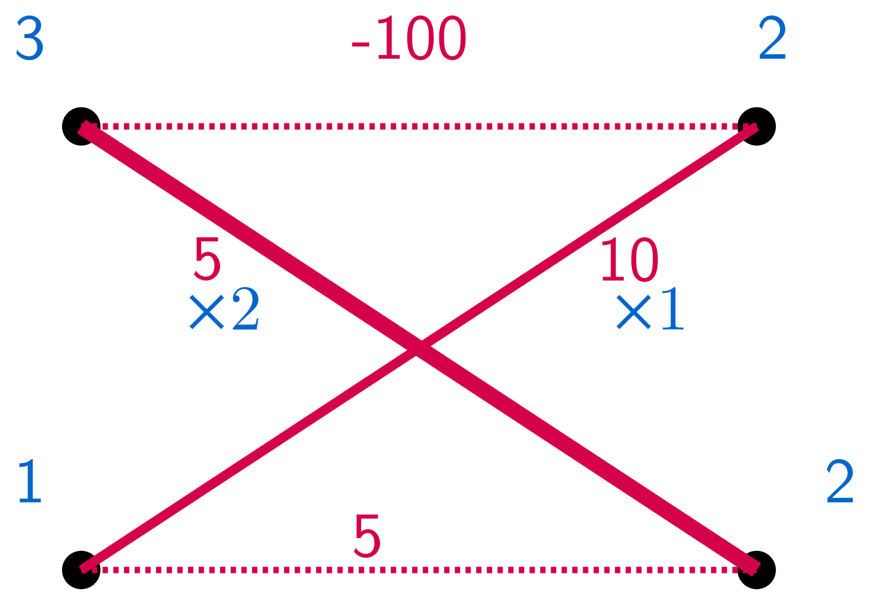
- $W := \max\{|c_{uv}|, |b_v|, 1\}$
- Maximum-cost b-matching



Weights and Demands!

$$\begin{aligned} \min \quad & \sum_{v \in V} b_v x_v \\ \text{s.t.} \quad & x_v + x_u \geq c_{uv} \quad \forall (u,v) \in E_A \\ & x_v + x_u \geq c_{uv} \quad \forall (u,v) \in E_B \\ & 0 \leq x \leq W \end{aligned}$$

- $W := \max\{|c_{uv}|, |b_v|, 1\}$
- Maximum-cost b-matching



Other (Equivalent & Weaker) Problems

Theorem:
If weights/costs/capacities/demands are poly(n), then we can solve the following using $O(n \log^2 n)$ communication:

- Maximum-cost bipartite perfect b-matching
- Maximum-cost bipartite b-matching
- Vertex-capacitated minimum-cost (s, t)-flow
- Transshipment
- Negative-weight single source shortest path
- Minimum mean cycle

Other (Equivalent & Weaker) Problems

Theorem:
If weights/costs/capacities/demands are poly(n), then we can solve the following using $O(n \log^2 n)$ communication:

- Maximum-cost bipartite perfect b-matching
- Maximum-cost bipartite b-matching
- Vertex-capacitated minimum-cost (s, t)-flow
- Transshipment
- Negative-weight single source shortest path
- Minimum mean cycle

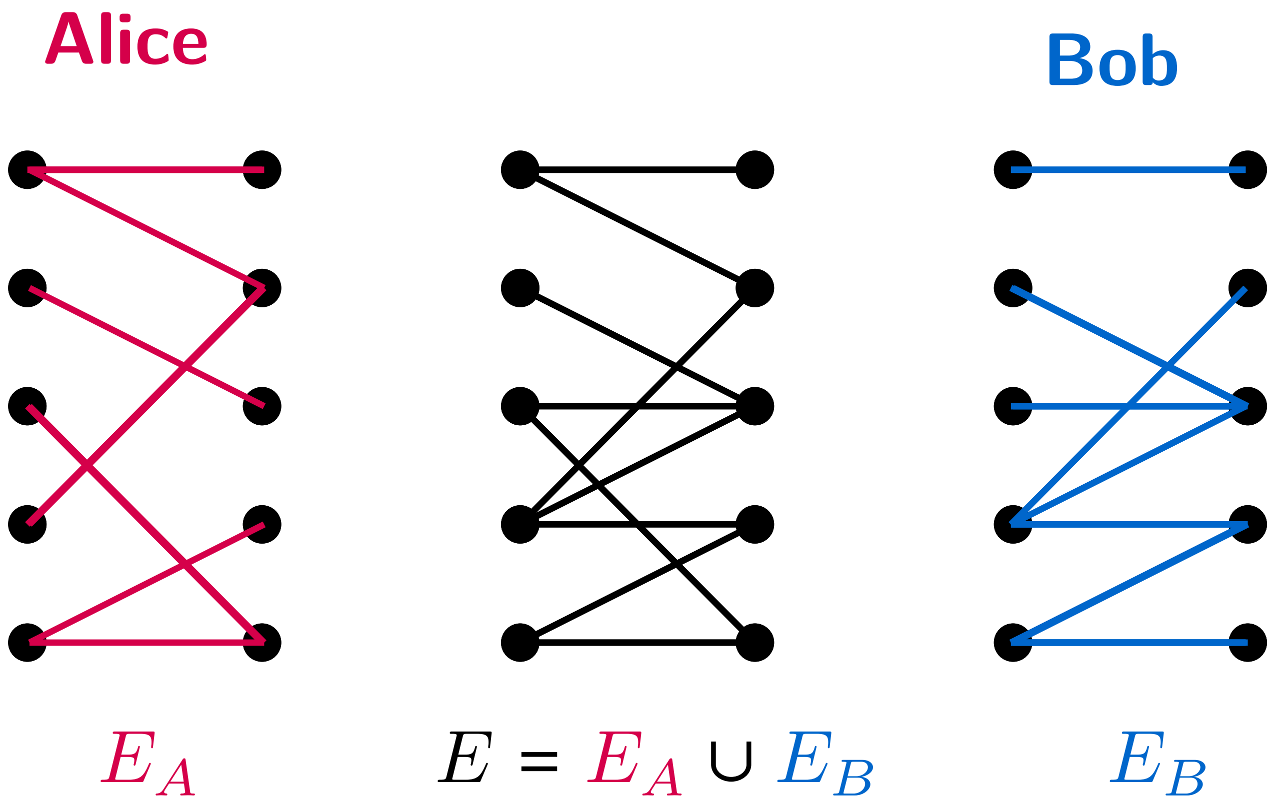
Note: All these have $O(n)$ edges in their answer!

Query Lower-Bounds

- AND-query $S = \{(u, v) \in L \times R\}$:
  "Is $|S \cap E| = |S|$?"

Query Lower-Bounds

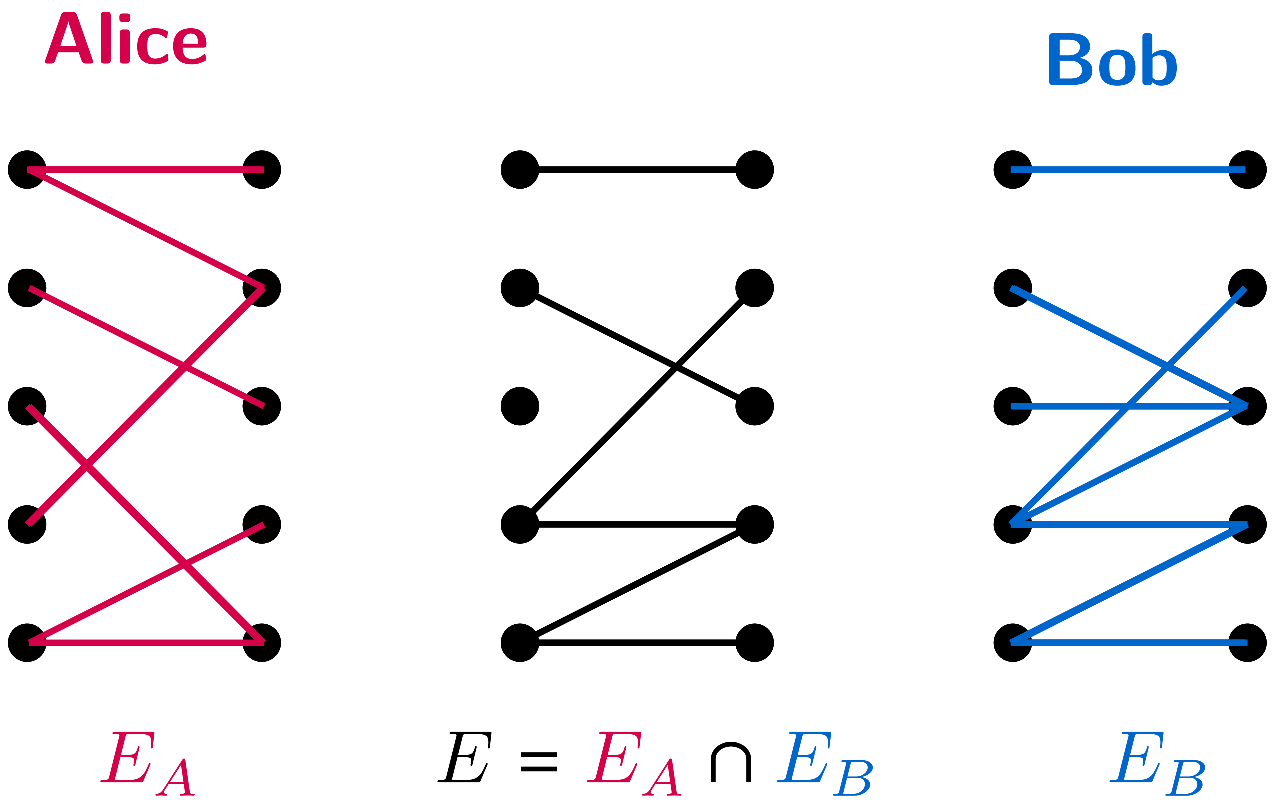
- AND-query S =$\{(u,v) \in L \times R\}$:
  "Is $|S \cap E|$ = $|S|$?"



Query Lower-Bounds

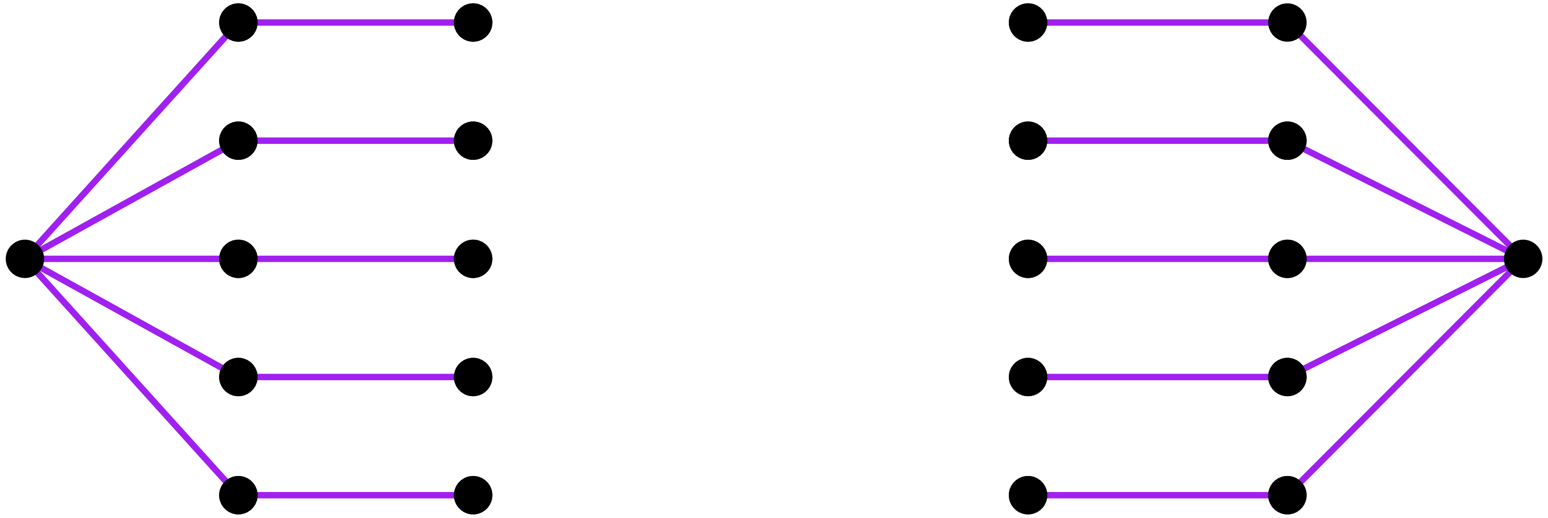
- AND-query $S = \{(u,v) \in L \times R\}$:
  "Is $|S \cap E| = |S|$?"

- AND-query algorithm $\implies$ communication protocol on intersection graph



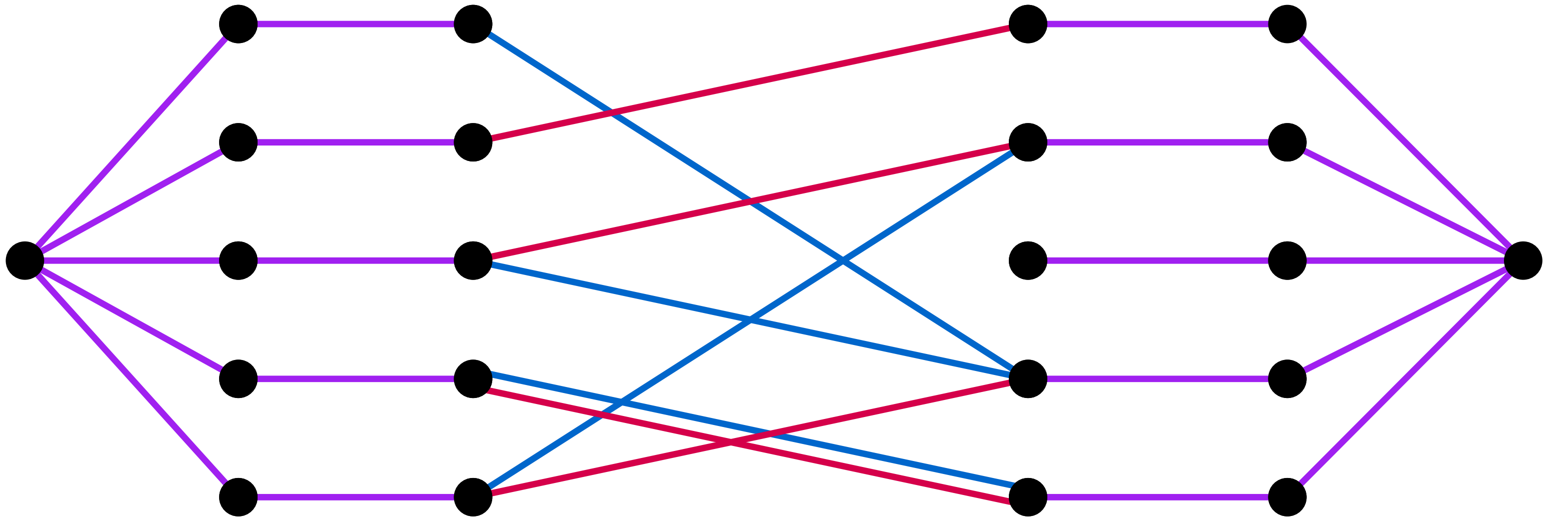
Query Lower-Bounds

- AND-query S =$\{(u,v) \in L \times R\}$:
  "Is $|S \cap E|$ = $|S|$?"



Query Lower-Bounds

- AND-query S $=\{(u,v) \in L \times R\}$:
  "Is $|S \cap E| = |S|$?"



- Perfect matching $\iff$ edges intersect
- Set-Intersection on $\approx n^2$ bits. Needs $\Omega(n^2)$ communication!

Summary — Results

Models	**Previous papers**		**This paper**
	Lower bounds	Upper bounds	
Two-party communication	$\Omega(n)$ Rand, $\Omega(n \log n)$ Det, Footnote 1 and 2	$\tilde{O}(n^{1.5})$ [DNO19, IKL+12]	$O(n \log^2 n)$, Det Theorem 1.1
Quantum edge query	$\Omega(n^{1.5})$ [Zha04, Ben22b]	$O(n^{1.75})$ [LL15]	$\tilde{O}(n^{1.5})$ Theorem 1.3
OR-query	$\Omega(n)$ Rand, $\Omega(n \log n)$ Det, [BN21]	$\tilde{O}(n^{1.5})$ Det, [Nis21]	$O(n \log^2 n)$, Det Theorem 1.3
XOR-query	$\Omega(n)$ Rand $\Omega(n^2)$ Det [BN21]	$\tilde{O}(n^{1.5})$ Rand Lemma 2.14 and [Nis21]	$O(n \log^2 n)$, Rand Theorem 1.3
AND-query	$\Omega(n)$ Rand, $\Omega(n^2)$ Det [BN21]	$O(n^2)$ Trivial	$\Omega(n^2)$, Rand Theorem 1.3

Open Problems :)

Open Problem — Round vs Communication Tradeoff

- Restricting the #rounds:
  - Streaming
  - Distributed
  - MPC
  - ...

	Rounds	**Communication**
trivial:	1	$\Theta(n^2)$
	?	?
cutting-planes:	$O(n \log n)$	$O(n \log^2 n)$

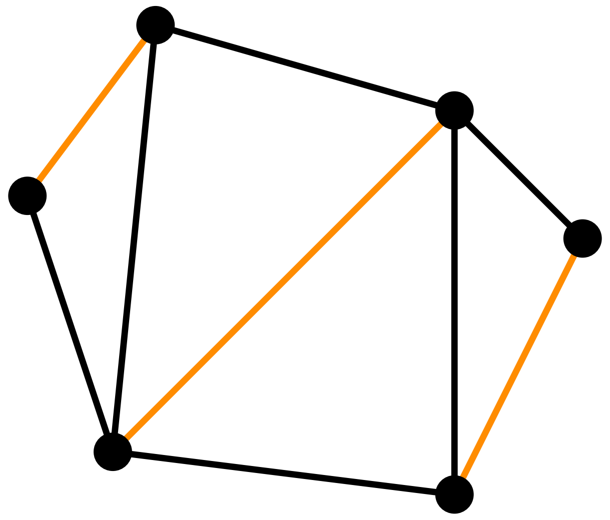
Open Problem — Approximation

- Finding an α-approximation instead? (size version)

Approximation	**Communication**
1	$O(n \log^2 n)$
?	?
2	$O(\log n)$

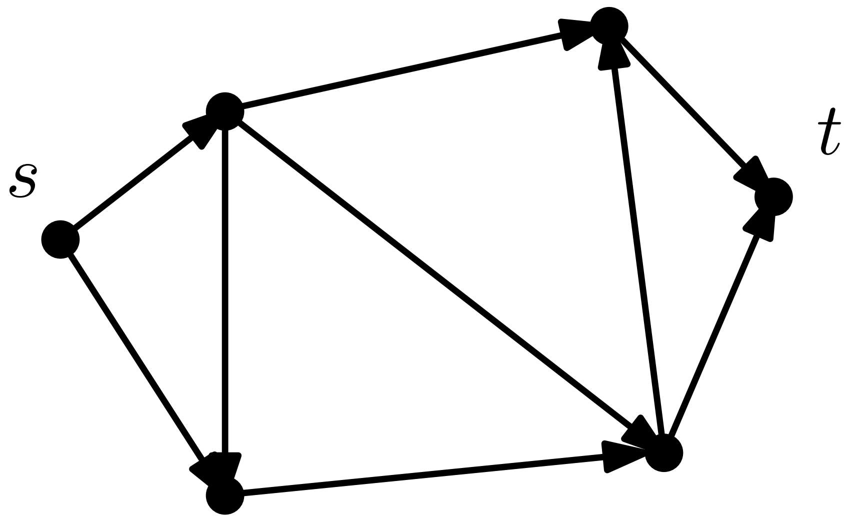
Open Problems — General Matching

- Communication and Query complexity of General Matching?
  - Interplay between *general* and *bipartite* matching unclear. . .
  - Optimal fractional matching by same approach.
  - Answer also has only $O(n)$ edges.
  - Unwieldy Linear Program. . .



Open Problems — Max Flow

- Communication and Query complexity of s,t-(min-cost)-max-flow?
  - Both the dual & primal have $\approx n^2$ variables
  - Answer may include all $\approx n^2$ edges
  - Nondeterministic (certificate) complexity are still low: $\tilde{O}(n)$



Open Problems

- Rounds vs Communication tradeoff
- Approximate bipartite matching
- Communication complexity of other problems?
  - General Matching
  - Max flow
  - Matroid intersection
  - . . .
- Other query models, e.g. *demand queries* (one-sided OR)
- Multiparty communication
- . . .

Thanks!