Auction-Based Algorithms for Weighted Matroid Intersection with Nearly-Linear Query Complexity

Chien-Chung Huang & Yusuke Kobayashi & Tatsuya Terao

ENS Ulm

Kyoto University

Kyoto University

What is Matroid? - Generalization of Linear Independence

Def: A finite set E and non-empty collection $\mathcal{I}$ of subset of E

s.t.

① $S' \subseteq S \in \mathcal{I} \Rightarrow S' \in \mathcal{I}$

② $S, T \in \mathcal{I}, |S| > |T| \Rightarrow \exists e \in S \setminus T$ s.t. $T \cup \{e\} \in \mathcal{I}$

$S \in \mathcal{I}$: S is independent

What is Matroid? - Generalization of Linear Independence

Def: A finite set E and non-empty collection $\mathcal{I}$ of subset of E

s.t. ① $S' \subseteq S \in \mathcal{I} \Rightarrow S' \in \mathcal{I}$

② $S, T \in \mathcal{I}, |S| > |T| \Rightarrow \exists e \in S \setminus T$ s.t. $T \cup \{e\} \in \mathcal{I}$

e.g.

- Linear matroid

$$\begin{pmatrix} \boxed{1} & \boxed{1} & \boxed{2} \\ 2 & 1 & 2 \end{pmatrix}$$

$E = \{①, ②, ③\}$

$\mathcal{I} = \{\phi, \{①\}, \{②\}, \{③\}, \{①,②\}, \{①,③\}\}$

- partition matroid
- laminar matroid
- graphic matroid

etc...

Weighted Matroid Intersection

input : two matroids $M_1 = (E, I_1)$, $M_2 = (E, I_2)$

weights $w \in \mathbb{R}_+^E$

output : common independent set $X \in I_1 \cap I_2$

maximizing $w(X) := \sum_{e \in X} w(e)$

Weighted Matroid Intersection

input : two matroids $M_1 = (E, I_1)$, $M_2 = (E, I_2)$

weights $w \in \mathbb{R}_+^E$

output : common independent set $X \in I_1 \cap I_2$

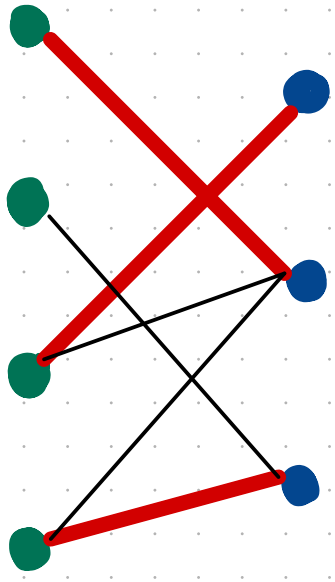
maximizing $w(X) := \sum_{e \in X} w(e)$

e.g. Bipartite Matching

E = edges

I_1 = each left vertex has at most 1 edge

I_2 = each right vertex has at most 1 edge



Weighted Matroid Intersection

input : two matroids $M_1=(E,I_1)$, $M_2=(E,I_2)$

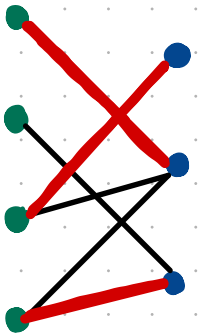
weights $w \in \mathbb{R}_+^E$

output : common independent set $X \in I_1 \cap I_2$

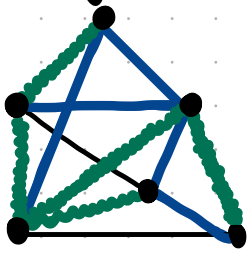
maximizing $w(X) := \sum_{e \in X} w(e)$

Generalizes many important combinatorial optimization problems!

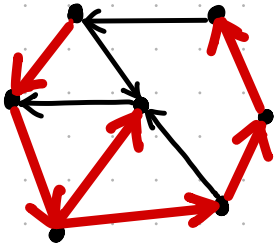
Bipartite Matching



Packing Spanning Tree



Arborescences



Weighted Matroid Intersection

input : two matroids $M_1 = (E, I_1)$, $M_2 = (E, I_2)$
weights $w \in \mathbb{R}_+^E$

output : common independent set $X \in I_1 \cap I_2$
maximizing $w(X) := \sum_{e \in X} w(e)$

Goal : Efficient $(1-\varepsilon)$-approx. algo !

Find $X \in I_1 \cap I_2$ with $w(X) \geq (1-\varepsilon) \cdot w(OPT_w)$

Weighted Matroid Intersection

input : two matroids $M_1 = (E, I_1)$, $M_2 = (E, I_2)$
weights $w \in \mathbb{R}_+^E$

output : common independent set $X \in I_1 \cap I_2$
maximizing $w(X) := \sum_{e \in X} w(e)$

Goal : Efficient $(1-\varepsilon)$-approx. algo !

Extensive prior work on matroid intersection and other polynomial-time solvable problems

Weighted Matroid Intersection

input : two matroids $M_1 = (E, I_1)$, $M_2 = (E, I_2)$
weights $w \in \mathbb{R}_+^E$

output : common independent set $X \in I_1 \cap I_2$
maximizing $w(X) := \sum_{e \in X} w(e)$

Goal : Efficient $(1-\varepsilon)$-approx. algo !

Extensive prior work on matroid intersection and other polynomial-time solvable problems

e.g. linear-time $(1-\varepsilon)$-approx. weighted matching [Duan-Pettie, FOCS'10]

How do we access the matroids?

① independence oracle

"Is $X \in \mathcal{I}$?"

How do we access the matroids?

① independence oracle

"Is $X \in \mathcal{I}$?"

② rank oracle

$$\text{rank}_{\mathcal{M}}(X) := \max\{|Y| \mid Y \subseteq X, Y \in \mathcal{I}\}$$

* rank oracle is more powerful than independence oracle

Prior Work on Exact Algorithms

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

unweighted

of independence-queries

Edmonds, Lawler, Aigner-Dowling ['70s]	$O(nr^2)$
Cunningham [SICOMP'86]	$O(nr^{3/2})$
Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2)$
Nguyễn ['19]	$\tilde{O}(nr)$
Chakrabarty-Lee-Sidford-Singla-Wong [FOCS'19]	
Blikstad-v.d.Brand-Mukhopadhyay-Nanongkai [STOC'21]	$\tilde{O}(n^{9/5})$
Blikstad [ICALP'21]	$\tilde{O}(nr^{3/4})$

Prior Work on Exact Algorithms

$n := |E|$, $r := \max_{X \in I_1 \cup I_2} |X|$

unweighted	# of independence-queries	weighted	# of independence-queries
Edmonds, Lawler, Aigner-Dowling ['70s]	$O(nr^2)$	Edmonds, Lawler, Frank Brezovec-Cornuéjols-Glover ['70s, 80s]	$O(nr^2)$
Cunningham [SICOMP'86]	$O(nr^{3/2})$		
Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2)$		
Nguyễn ['19] Chakrabarty-Lee-Sidford-Singla-Wong [FOCS'19]	$\tilde{O}(nr)$	Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2 \log W)$
Blikstad-v.d.Brand-Mukhopadhyay-Nanongkai [STOC'21]	$\tilde{O}(n^{9/5})$	Ene-Nguyễn ['20]	$\tilde{O}(nr \log W)$
Blikstad [ICALP'21]	$\tilde{O}(nr^{3/4})$		

Prior Work on Exact Algorithms

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

unweighted	# of independence-queries	weighted	# of independence-queries
Edmonds, Lawler, Aigner-Dowling ['70s]	$O(nr^2)$	Edmonds, Lawler, Frank Brezovec-Cornuéjols-Glover ['70s, 80s]	$O(nr^2)$
Cunningham [SICOMP'86]	$O(nr^{3/2})$		
Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2)$		
Nguyễn ['19] Chakrabarty-Lee-Sidford-Singla-Wong [FOCS'19]	$\tilde{O}(nr)$	Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2 \log W)$
Blikstad-v.d.Brand-Mukhopadhyay-Nanongkai [STOC'21]	$\tilde{O}(n^{9/5})$	Ene-Nguyễn ['20]	$\tilde{O}(nr \log W)$
Blikstad [ICALP'21]	$\tilde{O}(nr^{3/4})$		

lower bound

Blikstad-Mukhopadhyay-Nanongkai-Tu [STOC'23] Harvey [SODA'08]	$\Omega(n \log n)$ $\geq (\log_2 3)n - o(n)$

$(1-\varepsilon)$-approx. unweighted matroid intersection

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

	# of independence-query	
Cunningham [STOC'86] (observed in Chekuri-Quanrud [SODA'16])	$\tilde{O}(nr/\varepsilon)$	Det.
Chakrabarty-Lee-Sidford-Singla-Wong [FOCS'19]	$\tilde{O}(n^{1.5}/\varepsilon^{1.5})$	Det.
Blikstad [ICALP'21]	$\tilde{O}(n\sqrt{r}/\varepsilon)$	Det.
Quanrud [ICALP'24]	$\tilde{O}(n/\varepsilon + r^{3/2}/\varepsilon^3)$	Ran.
Blikstad-Tu [SOSA'25]	$\tilde{O}(n/\varepsilon + r/\varepsilon^5)$	Ran.
this work	$\tilde{O}(n/\varepsilon^2)$	Det.

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

$(1-\varepsilon)$-approx. weighted matroid intersection

	# of independence-query	
Huang-Kakimura-Kamiyama [SODA'16]	$\tilde{O}(nr^{3/2}/\varepsilon)$	Det.
Chekuri-Quanrud [SODA'16]	$\tilde{O}(nr/\varepsilon^2)$	Det.
Chekuri-Quanrud [SODA'16] + Blikstad [ICALP'21]	$\tilde{O}(n\sqrt{r}/\varepsilon^2)$	Det.

Reduction to the unweighted problems
inspired by Duan-Pettie'10

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

$(1-\varepsilon)$-approx. weighted matroid intersection

	# of independence-query	
Huang-Kakimura-Kamiyama [SODA'16]	$\tilde{O}(nr^{3/2}/\varepsilon)$	Det.
Chekuri-Quanrud [SODA'16]	$\tilde{O}(nr/\varepsilon^2)$	Det.
Chekuri-Quanrud [SODA'16] + Blikstad [ICALP'21]	$\tilde{O}(n\sqrt{r}/\varepsilon^2)$	Det.
Quanrud [ICALP'24]	$\tilde{O}(n/\varepsilon + r^{3/2}/\varepsilon^4)$	Ran.

inspired by Assadi [SOSA'24]

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

$(1-\varepsilon)$-approx. weighted matroid intersection

	# of independence-query	
Huang-Kakimura-Kamiyama [SODA'16]	$\tilde{O}(nr^{3/2}/\varepsilon)$	Det.
Chekuri-Quanrud [SODA'16]	$\tilde{O}(nr/\varepsilon^2)$	Det.
Chekuri-Quanrud [SODA'16] + Blikstad [ICALP'21]	$\tilde{O}(n\sqrt{r}/\varepsilon^2)$	Det.
Quanrud [ICALP'24]	$\tilde{O}(n/\varepsilon + r^{3/2}/\varepsilon^4)$	Ran.
Blikstad-Tu [SOSA'25] + Dudeja-Grinberger [IPCO'26]	$\tilde{O}(n\,\varepsilon^{-o(1/\varepsilon)})$	Ran.

Reduction to the unweighted problems inspired by Bernstein-Dudeja-Langley [STOC'21]

$n := |E|$, $r := \max_{X \in I_1 \cap I_2} |X|$

$(1-\varepsilon)$-approx. weighted matroid intersection

	# of independence-query	
Huang-Kakimura-Kamiyama [SODA'16]	$\tilde{O}(nr^{3/2}/\varepsilon)$	Det.
Chekuri-Quanrud [SODA'16]	$\tilde{O}(nr/\varepsilon^2)$	Det.
Chekuri-Quanrud [SODA'16] + Blikstad [ICALP'21]	$\tilde{O}(n\sqrt{r}/\varepsilon^2)$	Det.
Quanrud [ICALP'24]	$\tilde{O}(n/\varepsilon + r^{3/2}/\varepsilon^4)$	Ran.
Blikstad-Tu [SOSA'25] + Dudeja-Grinberger [IPCO'26]	$\tilde{O}(n\,\varepsilon^{-O(1/\varepsilon)})$	Ran.
this work	$\tilde{O}(n/\varepsilon^2)$	Det.

Main Thm.

$O(n \varepsilon^{-2} \log^2(n))$ independence-query deterministic $(1-\varepsilon)$-approx. algo. for weighted matroid intersection

Main Thm.

$O(n\varepsilon^{-2}\log^2(n))$ independence-query deterministic $(1-\varepsilon)$-approx. algo. for weighted matroid intersection

New auction algorithm

inspired by
- Connection between auction and stable matching
- Fleiner's matroidal generalization of stable matching

Overall advantages
- No reduction to the unweighted problems
- Simple algorithm

Warm Up: Rank-Query Algo.

more powerful than independence-query!

exact weighted		(1-ε)-approx. unweighted	
Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2 \log W)$	Chakrabarty-Lee-Sidford-Singh-Wong [FOCS'19]	$O(n \log(n)/\varepsilon)$
Tu [ISAAC'22]	$\tilde{O}(nr^{3/4} \log W)$	this work	$O(n \log(n)/\varepsilon)$

Warm Up: Rank-Query Algo.

more powerful than independence-query!

exact weighted		(1-ε)-approx. unweighted	
Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2 \log W)$	Chakrabarty-Lee-Sidford-Singla-Wong [FOCS'19]	$O(n \log(n)/\varepsilon)$
Tu [ISAAC'22]	$\tilde{O}(nr^{3/4} \log W)$	this work	$O(n \log(n)/\varepsilon)$

augmenting-path

auction algo.

Warm Up: Rank-Query Algo.

more powerful than independence-query!

exact weighted		(1-ε)-approx. unweighted	
Lee-Sidford-Wong [FOCS'15]	$\tilde{O}(r^2 \log W)$	Chakrabarty-Lee-Sidford-Singh-Wong [FOCS'19]	$O(n \log(n)/\varepsilon)$
Tu [ISAAC'22]	$\tilde{O}(nr^{3/4} \log W)$	this work	$O(n \log(n)/\varepsilon)$

(1-ε)-approx. weighted	
CLSSW19 + Dudeja-Grinberg [IPCO'26]	$\tilde{O}(n \varepsilon^{-O(1/\varepsilon)})$
this work	$O(n \log(n)/\varepsilon)$

Existing auction algo. for matroid intersection

Shigeno-Iwata [ORL'95]

Fujishige-Zhang [JORSJ'95]

Blikstad-Tu [SOSA'25]

"Batch-Update"

Assign $w_1(e)$, $w_2(e)$ for each $e \in E$

maintain (w_1-max. base S_1 of M_1
w_2-max. base S_2 of M_2

Existing auction algo. for matroid intersection

Shigeno-Iwata [ORL'95]

Fujishige-Zhang [JORSJ'95]

Blikstad-Tu [SOSA'25]

"Batch-Update"

assign $w_1(e)$, $w_2(e)$ for each $e \in E$

maintain (w_1-max. base S_1 of M_1
w_2-max. base S_2 of M_2

unweighted

$O(n/\varepsilon^2)$-independence-query

εn-additive approx.

+ randomized sparsification technique [Quanrud'24]

Existing auction algo. for matroid intersection

Shigeno-Iwata [ORL'95]

Fujishige-Zhang [JORSJ'95]

Blikstad-Tu [SOSA'25]

"Batch-Update"

assign $w_1(e)$, $w_2(e)$ for each $e \in E$

maintain (w_1-max. base S_1 of M_1
w_2-max. base S_2 of M_2

unweighted

$O(n/\varepsilon^2)$-independence-query εn-additive appx.

+ randomized sparsification technique [Quanrud'24]

incur a factor of $\varepsilon^{-O(1/\varepsilon)}$

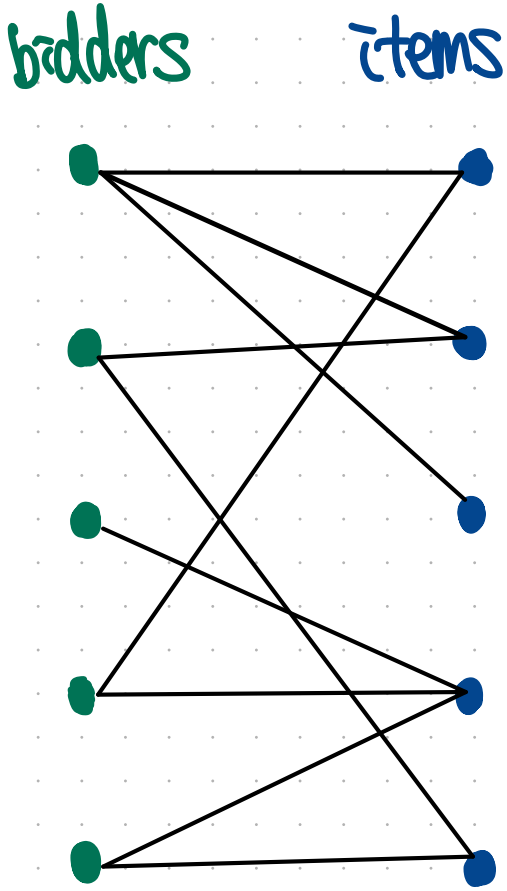
weighted

+ reduction to the unweighted problems [Dudeja-Grinberger'26]

Auction Algorithm for Bipartite Matching

Bertsekas '88, Dobzinski-Nisan-Oren [STOC'14], Assadi-Liu-Tarjan [SOSA'21]

$(1-\varepsilon)$-approx. algo.



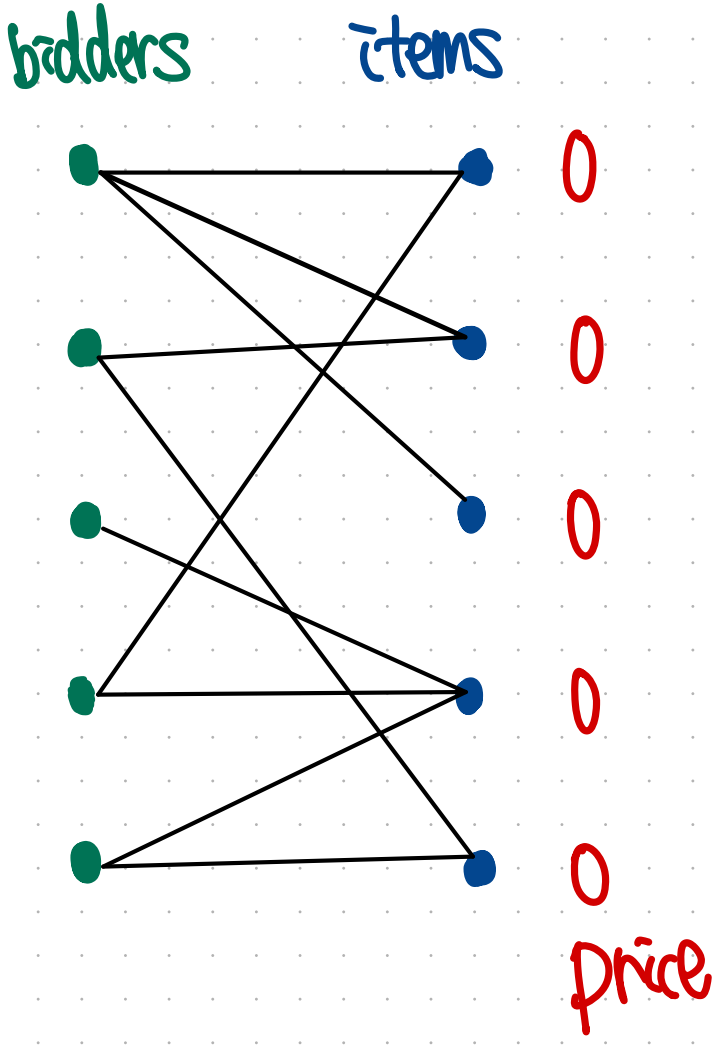
Auction Algorithm for Bipartite Matching

Bertsekas '88, Dobzinski-Nisan-Oren [STOC'14], Assadi-Liu-Tarjan [SOSA'21]

$(1-\varepsilon)$-approx. algo.

Initially, no item is allocated.

every item has price 0.



Auction Algorithm for Bipartite Matching

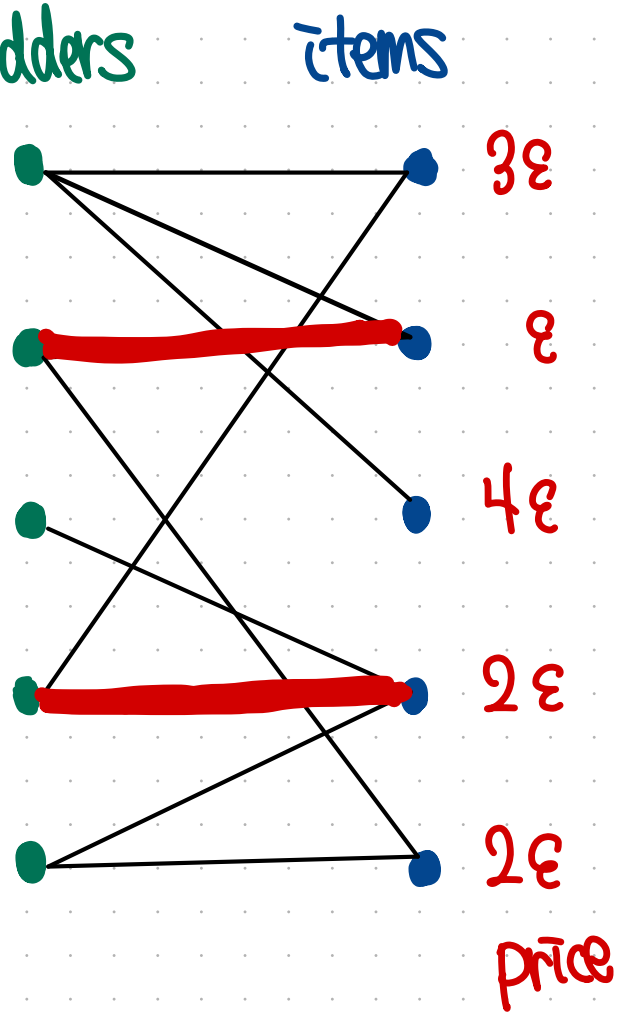
Bertsekas'88, Dobzinski-Nisan-Oren [STOC'14], Assadi-Liu-Tarjan [SOSA'21]

$(1-\varepsilon)$-approx. algo.

Initially, no item is allocated.

every item has price 0.

repeat



Auction Algorithm for Bipartite Matching

Bertsekas'88, Dobzinski-Nisan-Oren [STOC'14], Assadi-Liu-Tarjan [SOSA'21]

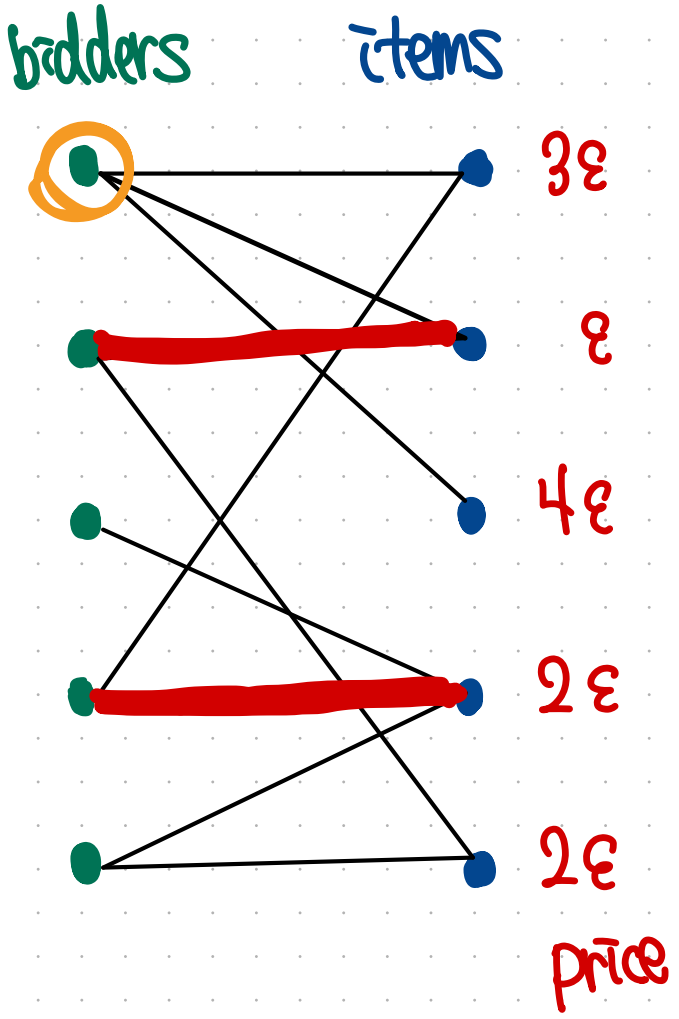
$(1-\varepsilon)$-approx. algo.

Initially, no item is allocated.

every item has price 0.

repeat

- choose an unallocated bidder that has a neighbor item with price < 1



Auction Algorithm for Bipartite Matching

Bertsekas '88, Dobzinski-Nisan-Oren [STOC'14], Assadi-Liu-Tarjan [SOSA'21]

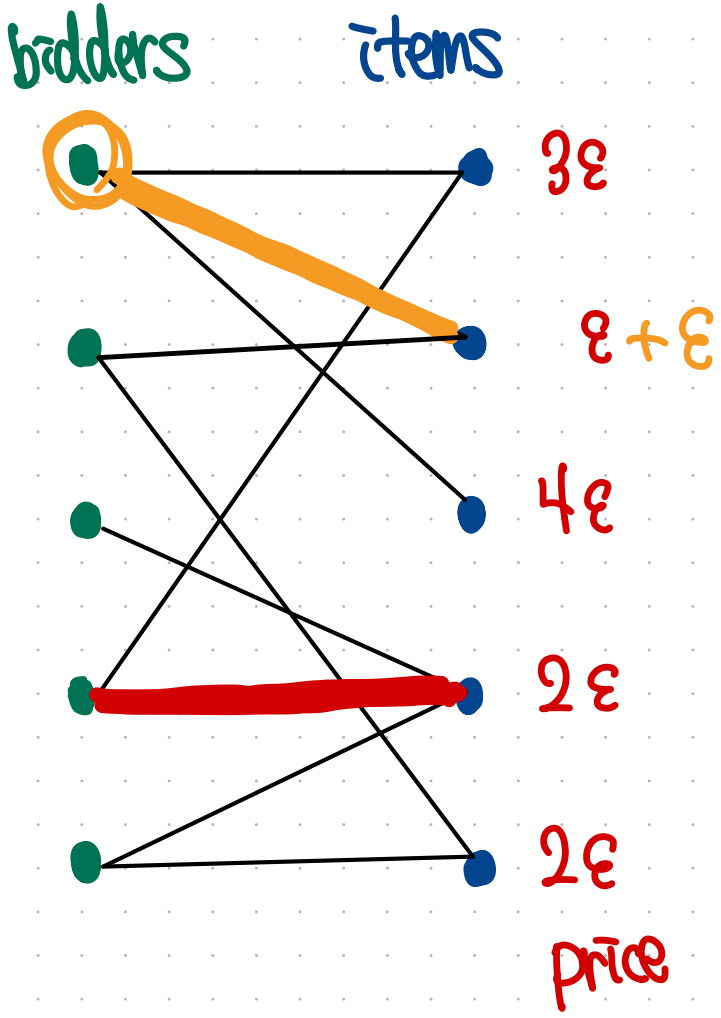
$(1-\varepsilon)$-approx. algo.

Initially, no item is allocated.

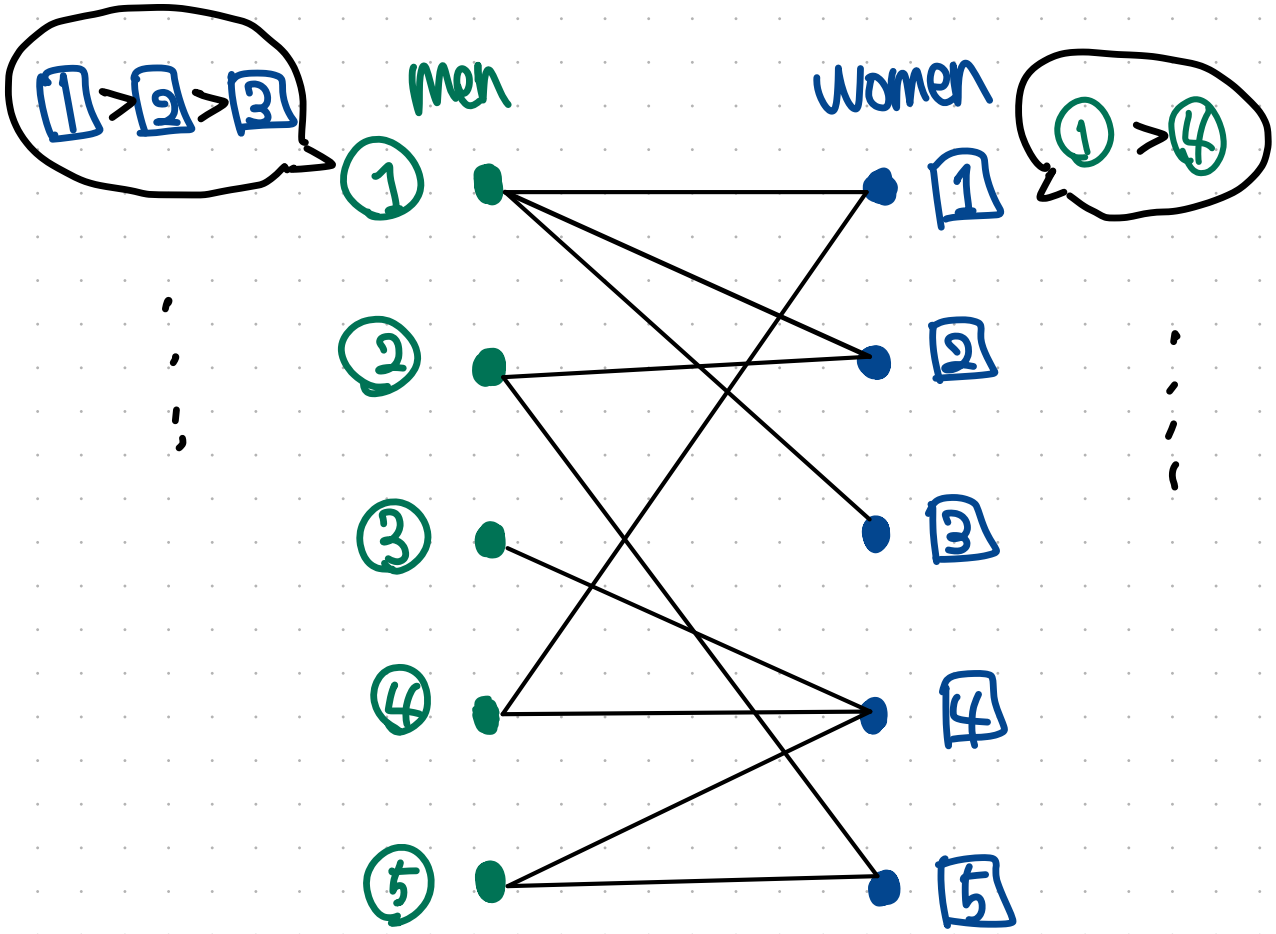
every item has price 0.

repeat

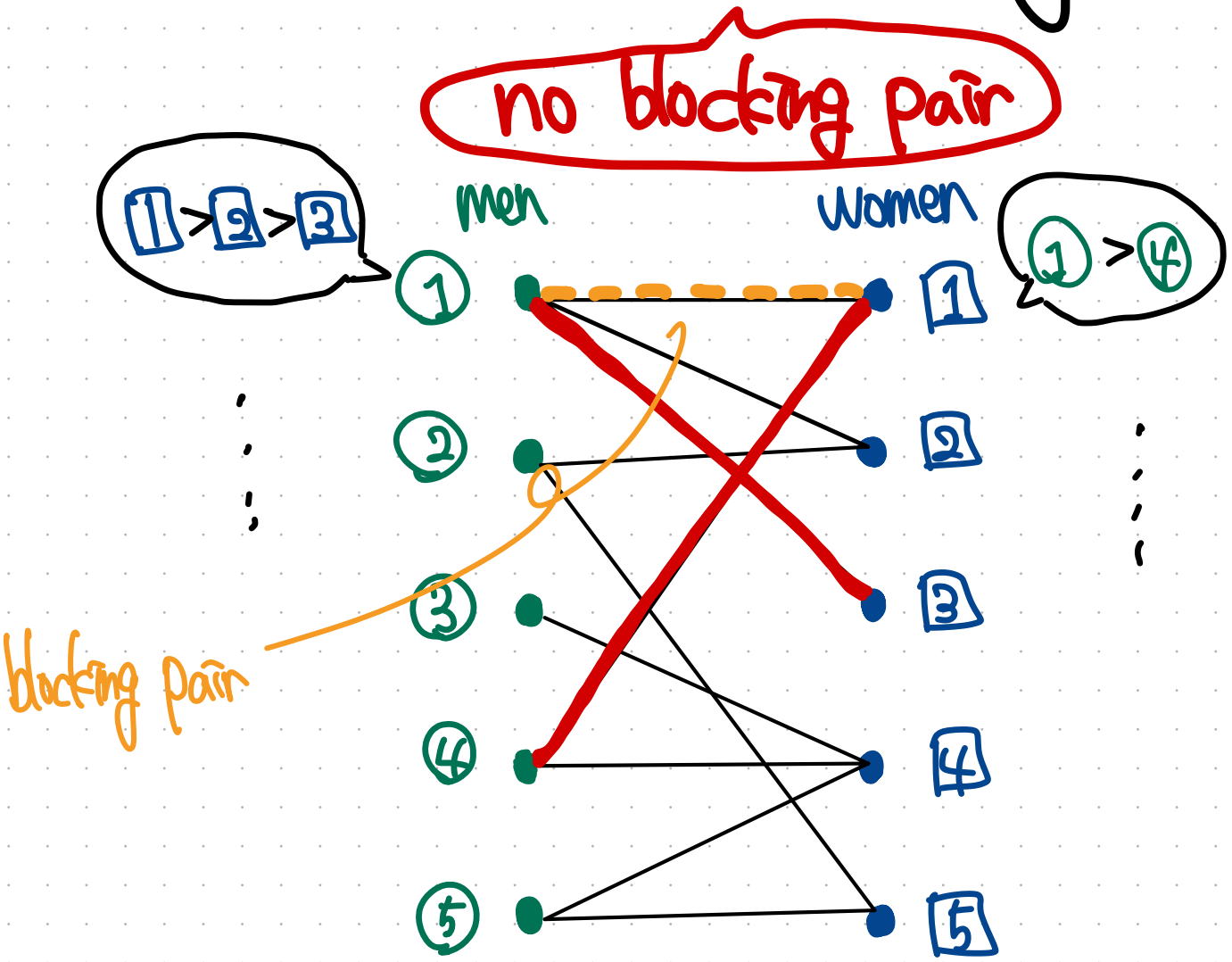
- choose an unallocated bidder that has a neighbor item with price < 1
- the bidder report an item of min. price among its neighboring items
- the reported item is allocated to the bidder, and its price is increased by ε.



Connection between Auction Algo. and Stable Matching



Connection between Auction Algo. and Stable Matching

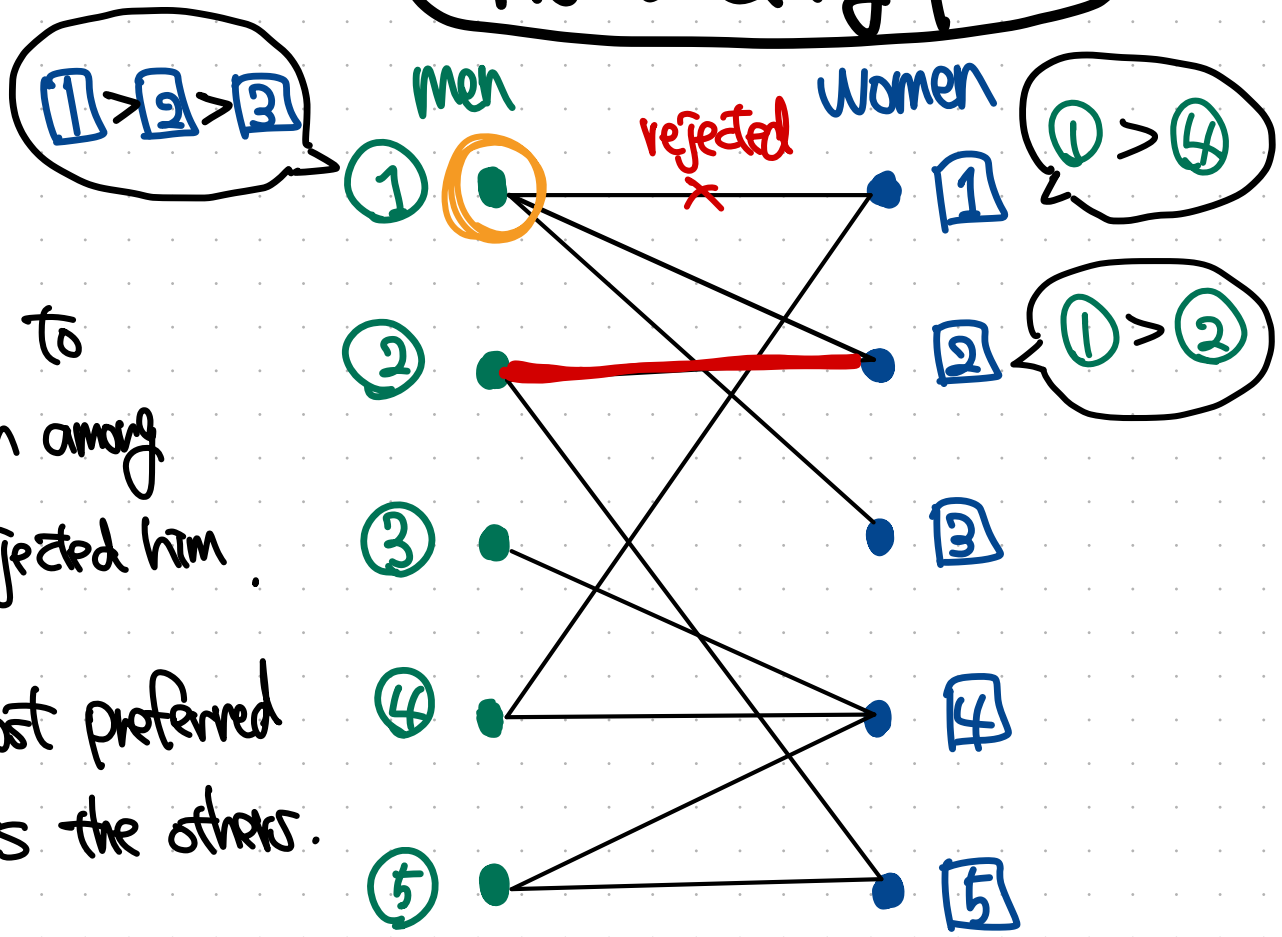


Connection between Auction Algo. and Stable Matching

Gale - Shapley algo.

repeat

- unmatched man proposes to his most preferred woman among those who have not rejected him.
- woman keeps her most preferred proposal and rejects the others.



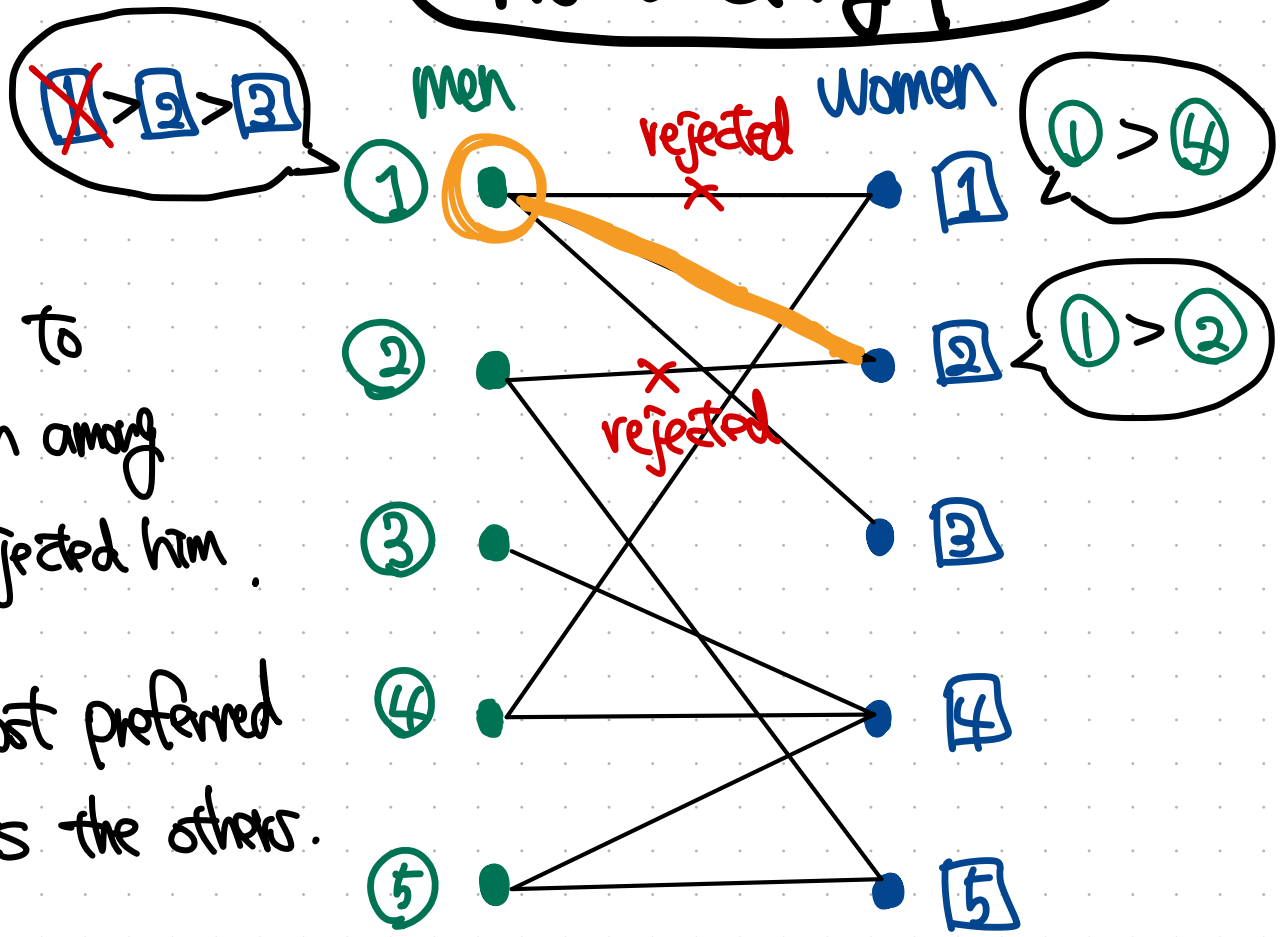
Connection between Auction Algo. and Stable Matching

Gale - Shapley algo.

repeat

- unmatched man proposes to his most preferred woman among those who have not rejected him.
- woman keeps her most preferred proposal and rejects the others.

no blocking pair

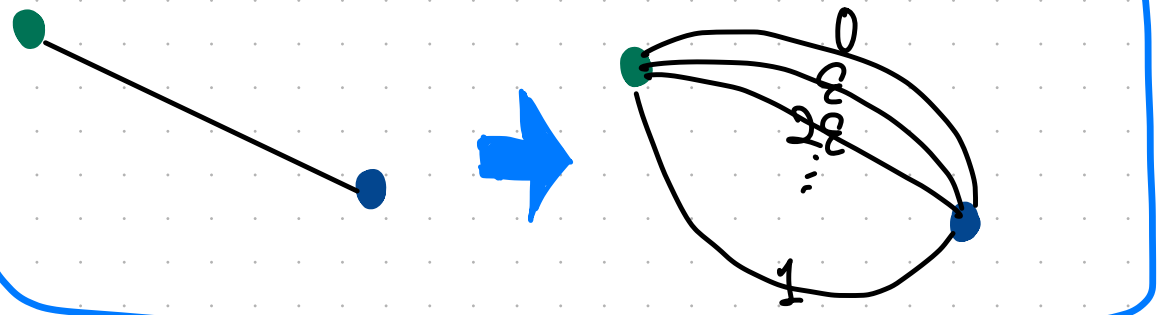


Connection between Auction Algo. and Stable Matching

- auction algo. can be viewed as

Gale-Shapley algo on a graph with copied edges

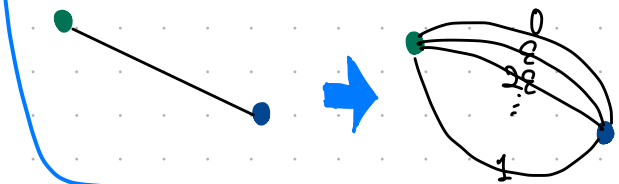
$$\tilde{E} = \{(e, \alpha) \mid e \in E, \alpha \in \{0, \varepsilon, 2\varepsilon, \dots, 1\}\}$$

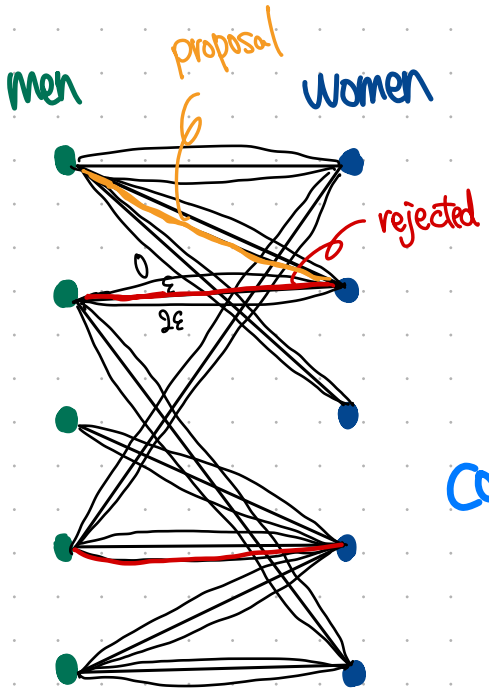


Connection between Auction Algo. and Stable Matching

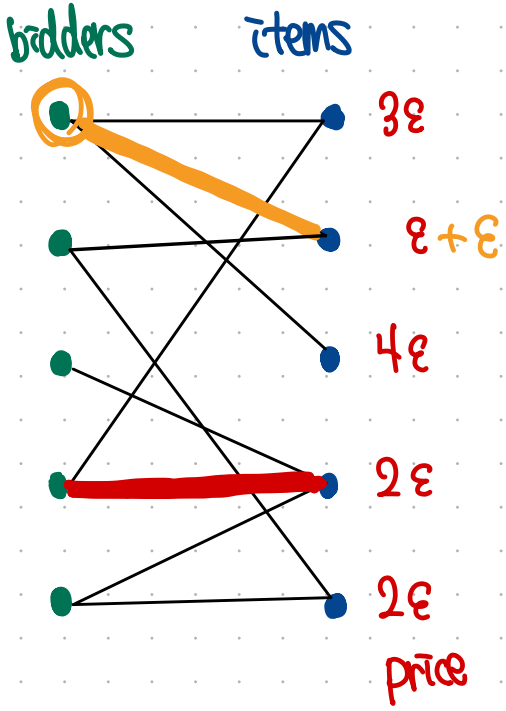
- auction algo. can be viewed as Gale-Shapley algo on a graph with copied edges

$\tilde{E} = \{(e,\alpha) \mid e \in E, \alpha \in \{0, \varepsilon, 2\varepsilon, \cdots, 1\}\}$





correspondence



Matroidal Generalization of Stable Matching

[Fleiner, IPCO'01]

$\mathcal{M}_1 = (E, \mathcal{I}_1)$ Alice

$\mathcal{M}_2 = (E, \mathcal{I}_2)$ Bob

$X \in E$: stable $\overset{\text{def}}{\Leftrightarrow}$ no blocking element $e \in E \setminus X$

$\forall i \in \{1, 2\}$,

① $X + e \in \mathcal{I}_i$

or

② $\exists f \in X$ s.t. $X + e - f \in \mathcal{I}_i$

the player associated with $\mathcal{M}_i$ prefers e to f

Matroidal Generalization of Stable Matching

[Fleiner, IPCO'01]

$M_1 = (E, \mathcal{I}_1)$ — Alice

$M_2 = (E, \mathcal{I}_2)$ — Bob

Generalization of Gale-Shapley algo. (modified version)

- $R \leftarrow \emptyset$: rejected elements

repeat

- Alice proposes her most preferred element $e \in E \setminus (X \cup R)$ s.t. $X + e \in \mathcal{I}_1$
- If $X + e \in \mathcal{I}_2$ then Bob accepts e, $X \leftarrow X + e$
- Else
  - $f \leftarrow$ Bob's least preferred element in $C_2(X|e)$
  - Bob rejects f, $R \leftarrow R + f$, $X \leftarrow X + e - f$

unique circuit in $X+e$ in M_2

no blocking element $e \in E \setminus X$

$\forall i \in \{1, 2\}$,

① $X + e \in \mathcal{I}_i$

or

② $\exists f \in X$ s.t. $X + e - f \in \mathcal{I}_i$

the player associated with M_i prefers e to f

Apply the generalized Gale-Shapley algo. to $\tilde{E} = \{(e,\alpha) \mid e \in E, \alpha \in \{0, \varepsilon, 2\varepsilon, \dots, 1\}\}$

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

- Find $e \in E \setminus X$ minimizing $p(e)$ s.t. $X + e \in I_1$
- if "no such element exists" or $p(e) > 1$
  - return X
- else
  - if $X + e \in I_2$ then $X \leftarrow X + e$
  - else find $f \in C_2(X, e)$ minimizing $p(f)$
    - $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon$

Apply the Gale-Shapley algo.
to $\tilde{E} = \{(e,x) \mid e \in E, x \in \{0, \varepsilon, \dots, 1\}\}$

➡ $(1-\varepsilon)$-approx. auction algo.

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

  Find $e \in E \setminus X$ minimizing $p(e)$ s.t. $X + e \in I_1$

  if "no such element exists" or $p(e) > 1$

    return X

  else

    if $X + e \in I_2$ then $X \leftarrow X + e$

    else find $f \in C_2(X, e)$ minimizing $p(f)$

      $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon$

Apply the Gale-Shapley algo. to $\tilde{E} = \{(e, x) \mid e \in E, x \in \{0, \varepsilon, \dots, 1\}\}$

→ $(1-\varepsilon)$-approx. auction algo.

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

  Find $e \in E \setminus X$ minimizing $p(e)$ s.t. $X + e \in I_1$

  if "no such element exists" or $p(e) > 1$

    return X

  else

    if $X + e \in I_2$ then $X \leftarrow X + e$

    else find $f \in C_2(X, e)$ minimizing $p(f)$

      $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon$

$O(\log n)$ queries by binary search

Apply the Gale-Shapley algo. to $\tilde{E} = \{(e, x) \mid e \in E, x \in \{0, \varepsilon, \dots, 1\}\}$ ➡ **$(1-\varepsilon)$-approx. auction algo.**

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

Find $e \in E \setminus X$ minimizing $p(e)$ s.t. $X + e \in I_1$

if "no such element exists" or $p(e) > 1$

  return X

else

  if $X + e \in I_2$ then $X \leftarrow X + e$

  else find $f \in C_2(X, e)$ minimizing $p(f)$

    $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon$

$O(n/\varepsilon)$ iterations

$O(\log n)$ queries by binary search

Extension to Weighted Matroid Intersection

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

- Find $e \in E \setminus X$ maximizing $w(e) - p(e)$ s.t. $X + e \in I_1$
- if "no such element exists" or $p(e) > w(e)$
  - return X
- else
  - if $X + e \in I_2$ then $X \leftarrow X + e$
  - else
    - find $f \in C_2(X, e)$ minimizing $p(f)$
    - $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon \cdot w(f)$

Extension to Weighted Matroid Intersection

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

- Find $e \in E \setminus X$ maximizing $w(e) - p(e)$ s.t. $X + e \in I_1$
- if "no such element exists" or $p(e) > w(e)$
  - return X
- else
  - if $X + e \in I_2$ then $X \leftarrow X + e$
  - else find $f \in C_2(X, e)$ minimizing $p(f)$
    - $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon \cdot w(f)$

inspired by auction algo. for weighted bipartite matching

[Zheng-Henzinger IPCO'23], [Liu-Ke-Khuller APPROX'23]

Extension to Weighted Matroid Intersection

"$(1-\varepsilon)$-approx. $O(n \log(n)/\varepsilon)$ rank-queries"

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

- Find $e \in E \setminus X$ maximizing $w(e) - p(e)$ s.t. $X + e \in I_1$
- if "no such element exists" or $p(e) > w(e)$
  - return X
- else
  - if $X + e \in I_2$ then $X \leftarrow X + e$
  - else find $f \in C_2(X, e)$ minimizing $p(f)$
    - $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon \cdot w(f)$

inspired by auction algo. for weighted bipartite matching

[Zheng-Henzinger IPCO'23], [Liu-Ke-Khuller APPROX'23]

Extension to Weighted Matroid Intersection

"$(1-\varepsilon)$-approx. $O(n \log(n)/\varepsilon)$ rank-queries"

$X \leftarrow \emptyset$, $p(e) \leftarrow 0$ (for each $e \in E$)

repeat

  Find $e \in E \setminus X$ maximizing $w(e) - p(e)$ s.t. $X + e \in I_1$

  if "no such element exists" or $p(e) > w(e)$

    return X

  else

    if $X + e \in I_2$ then $X \leftarrow X + e$

    else find $f \in C_2(X, e)$ minimizing $p(f)$

      $X \leftarrow X + e - f$, $p(f) \leftarrow p(f) + \varepsilon \cdot w(f)$

the binary search does not work in the independance oracle model!

Independence-Query Algo.

inspired by

Fleiner's implementation of generalized Gale-Shapley

+ Auction algo. for unweighted bipartite matching by Assadi-Liu-Tarjan [SOSA'21]

Computes a sequence of $O(1/\epsilon^2)$ maximal matching of the input graph

↓

"Organize the algorithm into rounds"

Independence-Query Algo.

"Organize the algorithm into rounds"

$X \leftarrow \emptyset$

repeat T rounds

- $X^*, Y \leftarrow X$, $\pi^*(e) \leftarrow \pi(e)$ (for each $e \in E$)
- for each $e \in E \setminus X^*$
  - if $g(e) - g^*(e) > 0$ and $Y+e$ is contained in a g^*-max. base of M_1
    - $Y \leftarrow Y+e$, $\pi(e) \leftarrow p(e)$
    - if $X+e \in I_2$ then $X \leftarrow X+e$
    - else Find $f \in C_2(X(e)-e$ minimizing $\pi(f)$
      - $X \leftarrow X+e-f$
- $X^t \leftarrow X$

return X^t maximizing $w(X^t)$ over $t \in \{0,1,\cdots,T\}$

Independence-Query Algo.

"Organize the algorithm into rounds"

$X \leftarrow \emptyset$

repeat T rounds

  $X^*, Y \leftarrow X$, $\pi^*(e) \leq \pi(e)$ (for each $e \in E$)

  for each $e \in E \setminus X^*$

    if $f(e) = f^*(e) > 0$ and $Y+e$ is contained in a f^*-max. base of M_1

      $Y \leftarrow Y+e$, $\pi(e) \leq p(e)$

      if $X+e \in I_2$ then $X \leftarrow X+e$

      else [Find $f \in C_2(X(e)-e$ minimizing $\pi(f)$
             $X \leftarrow X+e-f$

  $X^t \leftarrow X$

return X^t maximizing $w(X^t)$ over $t \in \{0,1,\cdots,T\}$

"Represent the price implicitly"

maintain $\pi(e)$ only for $e \in X$

Compute $p(e)$ from π for $e \in E \setminus X$

➡ Reduce the cost of price updates

Independence-Query Algo.

"Organize the algorithm into rounds"

$T = O(\log(n)/\epsilon^2)$

"Represent the price implicitly"

$X \leftarrow \emptyset$

repeat T rounds

- $X^*, Y \leftarrow X$, $\pi^*(e) \leftarrow \pi(e)$ (for each $e \in E$)
- for each $e \in E \setminus X^*$
  - if $g(e) = g^*(e) > 0$ and $Y+e$ is contained in a g^*-max. base of M_1
    - $Y \leftarrow Y+e$, $\pi(e) \leftarrow p(e)$
    - if $X+e \in I_2$ then $X \leftarrow X+e$
    - else
      - Find $f \in C_2(X(e)-e$ minimizing $\pi(f)$
      - $X \leftarrow X+e-f$
- $X^t \leftarrow X$

return X^t maximizing $w(X^t)$ over $t \in \{0,1,\cdots,T\}$

Each rounds uses $O(n \log(n))$ independence-query

Conclusion

- $O(n\varepsilon^{-2}\log^2(n))$ independence-query deterministic $(1-\varepsilon)$-approx. algo. for weighted matroid intersection

Conclusion

- $O(n\varepsilon^{-2}\log^2(n))$ independence-query deterministic $(1-\varepsilon)$-approx. algo. for weighted matroid intersection

New auction algorithm

inspired by
- Connection between auction and stable matching
- Fleiner's matroidal generalization of stable matching

Overall advantages

- No reduction to the unweighted problems
- Simple algorithm
- Semi-Streaming implementation with $O(1/\varepsilon^2)$ passes