Random Order Vertex Arrival Contention Resolution Schemes For Matching, With Applications

Hu Fu[1], Zhihao Gavin Tang[1], **Hongxun Wu**[2], Jinzhao Wu[3], and Qianfan Zhang[2]

[1]ITCS, Shanghai University of Finance and Economics

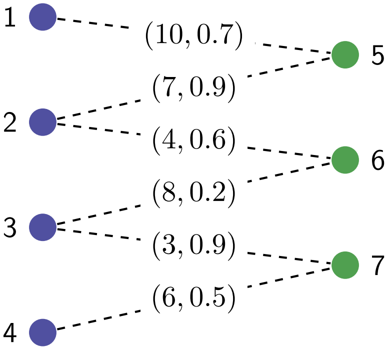
[2]IIIS, Tsinghua University

[3]Peking University

Query Commit

Query-Commit MWBM

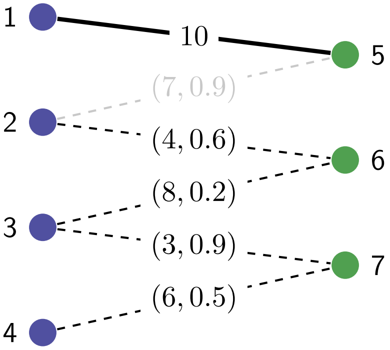
Chen, Immorlica, Karlin, Mahdian, and Rudra [CIKMR09] first considered the matching problem in query-commit model.



Query Commit

Query-Commit MWBM

For each edge e there is a tuple (w_e, p_e). Each edge e independently exists with probability p_e and has weight w_e.

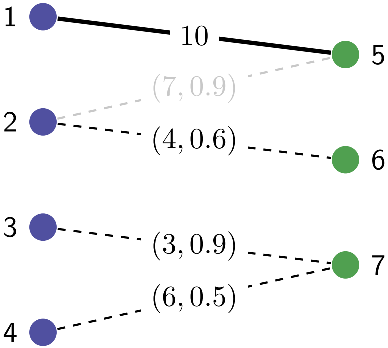


Utility $= 10$

Query Commit

Query-Commit MWBM

For each edge e there is a tuple (w_e, p_e). Each edge e independently exists with probability p_e and has weight w_e.

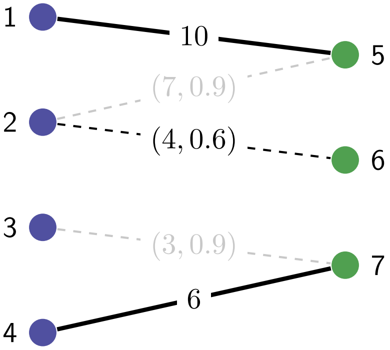


Utility $= 10$

Query Commit

Query-Commit MWBM

For each edge e there is a tuple (w_e, p_e). Each edge e independently exists with probability p_e and has weight w_e.

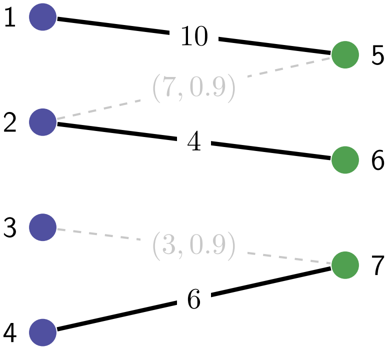


Utility $= 10 + 6$

Query Commit

Query-Commit MWBM

For each edge e there is a tuple (w_e, p_e). Each edge e independently exists with probability p_e and has weight w_e.

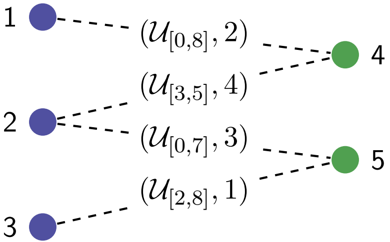


Utility $= 10 + 6 + 4 = 20$

Price of Information

PoI MWBM

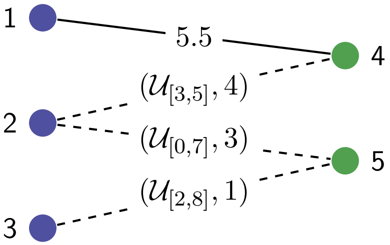
Singla [Singla17] introduced the price-of-information model.



Price of Information

PoI MWBM

For each edge e there is a tuple $(\mathcal{D}_e, \pi_e)$. The weight of edge e follows from distribution $\mathcal{D}_e$ and we can query it by paying cost π_e.

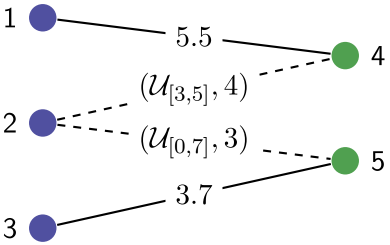


Total cost $= 2$

Price of Information

PoI MWBM

For each edge e there is a tuple $(\mathcal{D}_e, \pi_e)$. The weight of edge e follows from distribution $\mathcal{D}_e$ and we can query it by paying cost π_e.

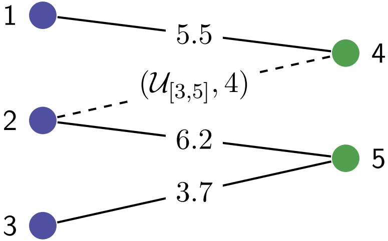


Total cost $= 2 + 1$

Price of Information

PoI MWBM

For each edge e there is a tuple $(\mathcal{D}_e, \pi_e)$. The weight of edge e follows from distribution $\mathcal{D}_e$ and we can query it by paying cost π_e.

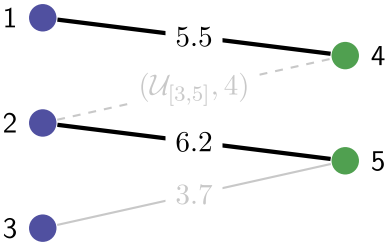


Total cost $= 2 + 1 + 3 = 6$

Price of Information

PoI MWBM

For each edge e there is a tuple $(\mathcal{D}_e, \pi_e)$. The weight of edge e follows from distribution $\mathcal{D}_e$ and we can query it by paying cost π_e.



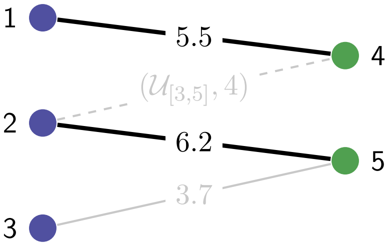
Total cost $= 2 + 1 + 3 = 6$

Max weight $= 5.5 + 6.2 = 11.7$

Price of Information

PoI MWBM

For each edge e there is a tuple $(\mathcal{D}_e, \pi_e)$. The weight of edge e follows from distribution $\mathcal{D}_e$ and we can query it by paying cost π_e.



$$\text{Total cost} = 2 + 1 + 3 = 6$$

$$\text{Max weight} = 5.5 + 6.2 = 11.7$$

$$\text{Utility} = \text{Max weight} - \text{Total cost} = 5.7$$

Stochastic Matching

The paper by Gamlath, Kale, and Svensson proved the following result.

Theorem ([GKS19])

For bipartite graphs, there is a $1 - \frac{1}{e} \approx 0.632$*-approximation algorithm for maximum weight matching in query-commit (and price of information) model.*

Stochastic Matching

The paper by Gamlath, Kale, and Svensson proved the following result.

Theorem ([GKS19])

For bipartite graphs, there is a $1 - \frac{1}{e} \approx 0.632$-approximation algorithm for maximum weight matching in query-commit (and price of information) model.

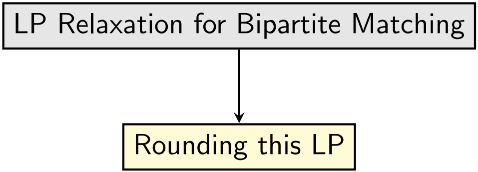
Theorem (Our result)

For general graphs, there is a $\frac{8}{15} \approx 0.533$-approximation algorithm for maximum weight matching in query-commit (and price of information) model.

Result in [GKS19]

$$\boxed{\text{LP Relaxation for Bipartite Matching}}$$

Result in [GKS19]



Result in [GKS19]

LP Relaxation for Bipartite Matching

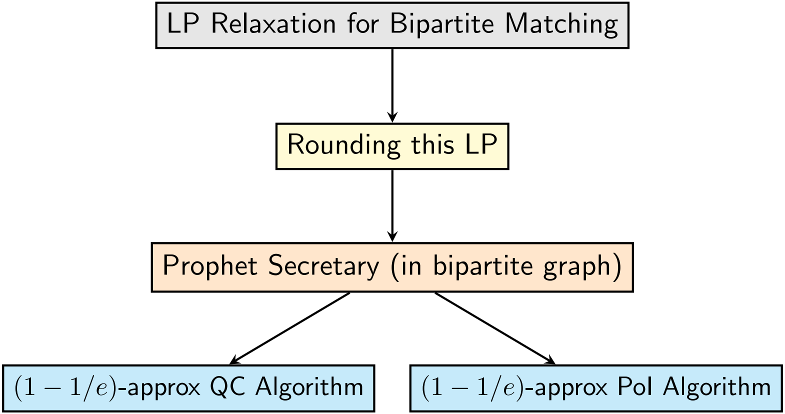
↓

Rounding this LP

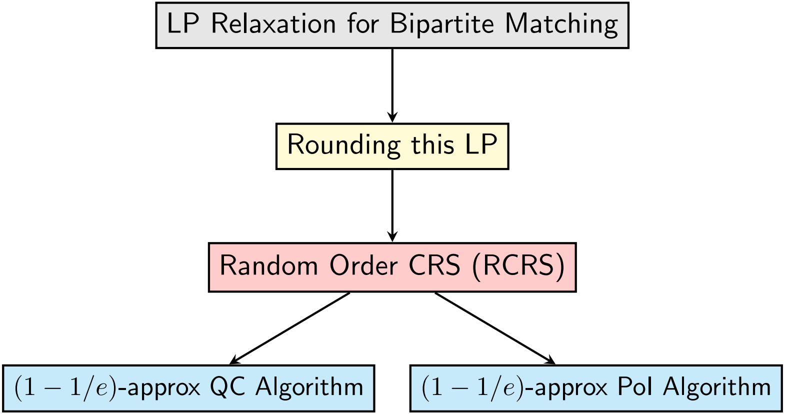
↓

Prophet Secretary (in bipartite graph)

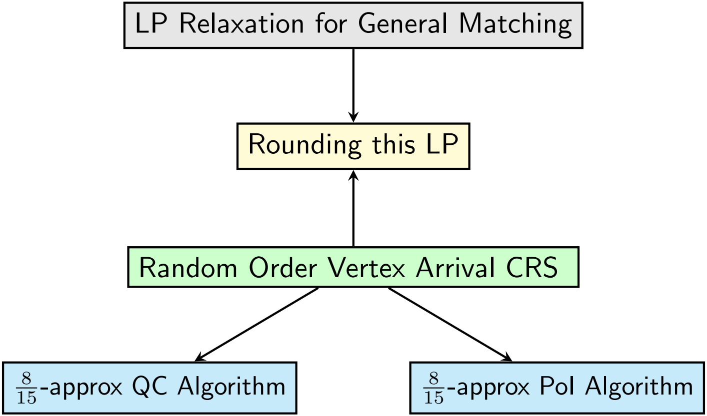
Result in [GKS19]



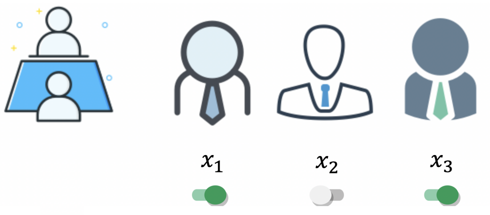
Our Observation



Our Observation



(Random) Contention Resolution Schemes

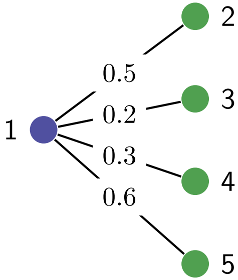


Given a set $[n]$ of elements:

- i is active independently with probability x_i where $\sum_{i \in [n]} x_i \leq 1$.
- (Elements arrive one by one in a random order.)
- We can select at most one element (which must be active).

A CRS is called α-selectable if $\Pr[i \text{ selected} \mid i \text{ active}] \geq \alpha$ for all i.

Techniques in [GKS19]



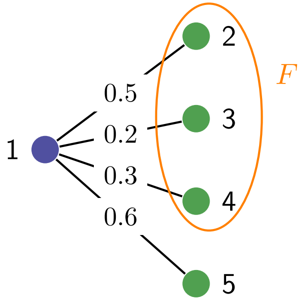
LP Relaxation

For bipartite graph $G = (A \cup B, E)$,

$$
\begin{aligned}
\max \sum_{e \in E} & x_e \cdot w(e) & \\
s.t. \sum_{e \in F} & x_e \leq f(F) & \forall v \in A, F \subseteq \delta(v) \\
\sum_{e \in \delta(u)} & x_e \leq 1 & \forall u \in B \\
& x_e \geq 0 & \forall e \in E
\end{aligned}
$$

where $f(F) = 1 - \prod_{e \in F}(1 - p_e)$.

Techniques in [GKS19]



$f(F) = 1 - 0.5 \times 0.8 \times 0.7$

$x_{(1,2)} + x_{(1,3)} + x_{(1,4)} \leq f(F)$

LP Relaxation

For bipartite graph $G = (A \cup B, E)$,

$$\begin{aligned} \max \sum_{e \in E} & x_e \cdot w(e) & \\ s.t. \sum_{e \in F} & x_e \leq f(F) & \forall v \in A, F \subseteq \delta(v) \\ \sum_{e \in \delta(u)} & x_e \leq 1 & \forall u \in B \\ & x_e \geq 0 & \forall e \in E \end{aligned}$$

where $f(F) = 1 - \prod_{e \in F}(1 - p_e)$.

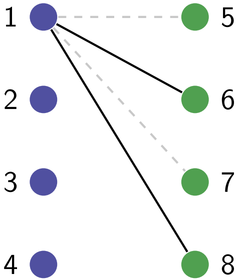
Techniques in [GKS19]

Lemma ([GSK19])

For any vertex $v \in A$, there exists a distribution D^v over permutations of $\delta(v)$ such that:

- *Sample $\sigma \sim D_v$. Each edge e is the first edge that exists in σ with probability exactly x_e^*.*

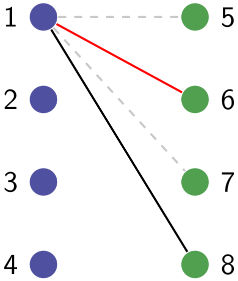
Techniques in [GKS19]



RCRS

- For each vertex $v \in A$, it activates the first e_i that exists.
  - Each edge is activated exactly with probability x_e^*.

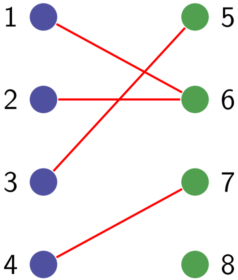
Techniques in [GKS19]



RCRS

- For each vertex $v \in A$, it activates the first e_i that exists.
  - Each edge is activated exactly with probability x_e^*.

Techniques in [GKS19]

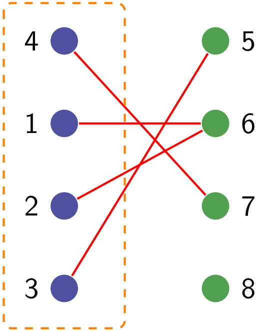


RCRS

- For each vertex $v \in A$, it activates the first e_i that exists.
  - Each edge is activated exactly with probability x_e^*.
- For each vertex $v' \in B$, multiple edges may be active.
  - We sample a random order of all vertices in A and pick one by RCRS.

$$\sum_{e \in \delta(u)} x_e^* \leq 1 \quad \forall u \in B$$

Techniques in [GKS19]

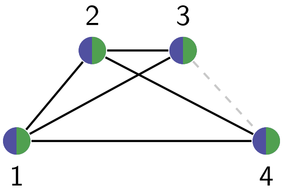


RCRS

- For each vertex $v \in A$, it activates the first e_i that exists.
  - Each edge is activated exactly with probability x_e^*.
- For each vertex $v' \in B$, multiple edges may be active.
  - We sample a random order of all vertices in A and pick one by RCRS.

$$\sum_{e \in \delta(u)} x_e^* \leq 1 \quad \forall u \in B$$

Generalize [GKS19] to General Graphs



LP Relaxation

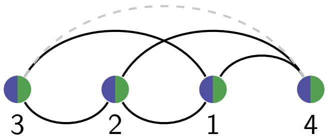
$$\max \sum_{e \in E} x_e \cdot w(e)$$

$$s.t. \sum_{e \in F} x_e \leq f(F) \quad \forall v \in V, F \subseteq \delta(v)$$

$$x_e \geq 0 \quad \forall e \in E$$

where $f(F) = 1 - \prod_{e \in F}(1 - p_e)$.

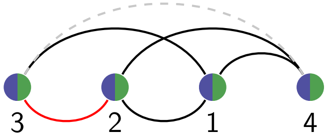
Generalize [GKS19] to General Graphs



Generalize RCRS

We sample a random arrival order of all vertices. Let $\delta'(v)$ be the edges to vertices that arrives before v.

Generalize [GKS19] to General Graphs



Generalize RCRS

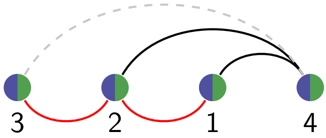
We sample a random arrival order of all vertices. Let $\delta'(v)$ be the edges to vertices that arrives before v.

Lemma

For any vertex $v \in V$, there exists a distribution D^v over permutations of $\delta'(v)$ such that

- *Sample $\sigma \sim D_v$. Each edge e is the first edge that exists in σ with probability exactly x^*_e.*

Generalize [GKS19] to General Graphs



Generalize RCRS

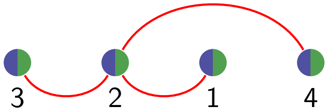
We sample a random arrival order of all vertices. Let $\delta'(v)$ be the edges to vertices that arrives before v.

Lemma

For any vertex $v \in V$, there exists a distribution D^v over permutations of $\delta'(v)$ such that

- *Sample $\sigma \sim D_v$. Each edge e is the first edge that exists in σ with probability exactly x^*_e.*

Generalize [GKS19] to General Graphs



Generalize RCRS

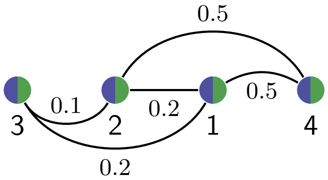
We sample a random arrival order of all vertices. Let $\delta'(v)$ be the edges to vertices that arrives before v.

Lemma

For any vertex $v \in V$, there exists a distribution D^v over permutations of $\delta'(v)$ such that

- *Sample $\sigma \sim D_v$. Each edge e is the first edge that exists in σ with probability exactly x_e^*.*

Random Order Vertex Arrival CRS



Given a graph $G = (V, E, \boldsymbol{x})$ satisfying $\sum_{e \in \delta(u)} x_e \leq 1$ for each $u \in V$, all vertices of G arrive online in a uniformly random order.

Upon the arrival of a vertex v, at most one edge e connecting v and an arrived vertex is *active* w.p. x_e.

Random Order Vertex Arrival CRS

Given a graph $G = (V, E, \boldsymbol{x})$ satisfying $\sum_{e \in \delta(u)} x_e \leq 1$ for each $u \in V$, all vertices of G arrive online in a uniformly random order.

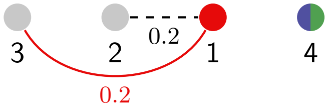
Upon the arrival of a vertex v, at most one edge e connecting v and an arrived vertex is *active* w.p. x_e.

Random Order Vertex Arrival CRS

Given a graph $G = (V, E, \boldsymbol{x})$ satisfying $\sum_{e \in \delta(u)} x_e \leq 1$ for each $u \in V$, all vertices of G arrive online in a uniformly random order.

Upon the arrival of a vertex v, at most one edge e connecting v and an arrived vertex is *active* w.p. x_e.

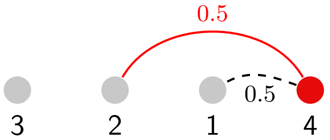
Random Order Vertex Arrival CRS



Given a graph $G = (V, E, \boldsymbol{x})$ satisfying $\sum_{e \in \delta(u)} x_e \leq 1$ for each $u \in V$, all vertices of G arrive online in a uniformly random order.

Upon the arrival of a vertex v, at most one edge e connecting v and an arrived vertex is *active* w.p. x_e.

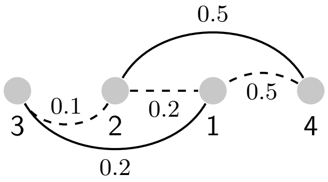
Random Order Vertex Arrival CRS



Given a graph $G = (V, E, \boldsymbol{x})$ satisfying $\sum_{e \in \delta(u)} x_e \leq 1$ for each $u \in V$, all vertices of G arrive online in a uniformly random order.

Upon the arrival of a vertex v, at most one edge e connecting v and an arrived vertex is *active* w.p. x_e.

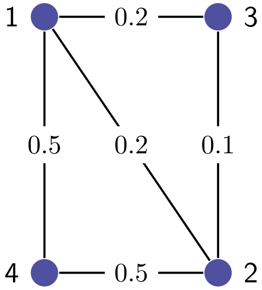
Random Order Vertex Arrival CRS



The scheme must irrevocably decide whether to select the active edge (if any exists), upon the arrival of each vertex.

A vertex arrival RCRS is c-*selectable* if $\Pr[e \text{ is selected} \mid e \text{ is active}] \geq c$ for every $e \in E$.

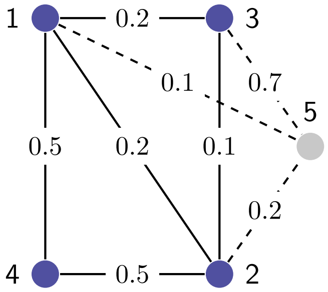
Our Algorithm



Our algorithm briefly consists of three steps:

- Add dummy vertices and edges to the graph so that the resulting graph is 1-regular, i.e. $\sum_{(u,v)\in E} x_{u,v} = 1$ for every $u \in V$.
- Prune each edge from x_e to $w_e = f(x_e) \stackrel{\text{def}}{=} \frac{3x_e}{3+2x_e}$.
- Run greedy on the pruned instance.

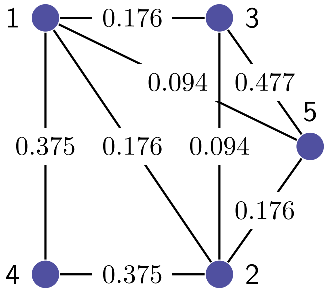
Our Algorithm



Add dummy vertices

We can make the graph 1-regular, i.e. $\sum_{(u,v)\in E'} x_{u,v} = 1$ for every $u \in V'$, by adding at most $|V|$ dummy vertices.

Our Algorithm

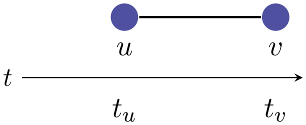


Prune and Greedy

We then prune each edge from x_e to $w_e = f(x_e) \overset{\text{def}}{=} \frac{3x_e}{3+2x_e}$ and run greedy on the pruned instance.

That is to say, for any active edge $e = (u, v)$, if both u and v is not matched, it will select it with probability $\frac{f(x_e)}{x_e}$.

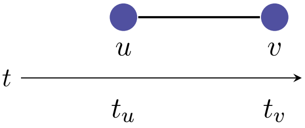
Analysis



We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
= {} & f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\
= {} & f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

Analysis



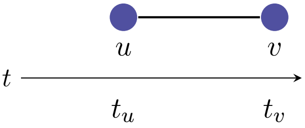
We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
&= f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\
&= f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

where

$$
\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
&= \sum_{z \neq u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}
$$

Analysis



We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
= {} & f(x_{uv}) \cdot \Pr[t_u \leq t, u \text{ unmatched } @t \mid t_v = t] \\
= {} & f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

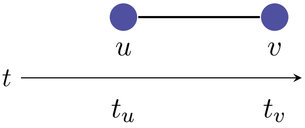
where

$$
\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
= {} & \sum_{z \neq u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}
$$

A direct analysis

$$
\sum_{z \neq u,v} \Pr[(u,z) \text{ matched before } t \mid t_v = t] \leq \sum_{z \neq u,v} t^2 f(x_{uz}) \leq t^2
$$

Analysis



We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
= {} & f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\
= {} & f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

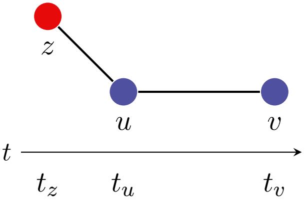
where

$$
\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
= {} & \sum_{z \neq u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}
$$

Recursive analysis

$$
\begin{aligned}
&\Pr[(u,z) \text{ matched before } t \mid t_v = t] \\
= {} & \int_0^t \Big(\Pr[u \to z \mid t_u = s, t_v = t] + \Pr[z \to u \mid t_z = s, t_v = t] \Big) \mathrm{d}s
\end{aligned}
$$

Analysis



We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
&= f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\
&= f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

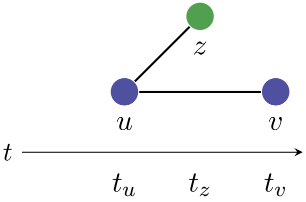
where

$$
\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
&= \sum_{z \neq u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}
$$

Recursive analysis

$$
\begin{aligned}
&\Pr[(u,z) \text{ matched before } t \mid t_v = t] \\
&= \int_0^t \Big(\Pr[u \to z \mid t_u = s, t_v = t] + \Pr[z \to u \mid t_z = s, t_v = t] \Big) \mathrm{d}s
\end{aligned}
$$

Analysis



We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
&= f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\
&= f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

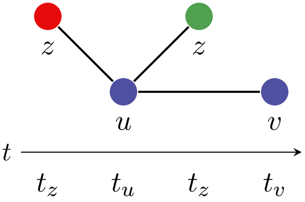
where

$$
\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
&= \sum_{z \neq u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}
$$

Recursive analysis

$$
\begin{aligned}
&\Pr[(u,z) \text{ matched before } t \mid t_v = t] \\
&= \int_0^t \Big(\Pr[u \to z \mid t_u = s, t_v = t] + \Pr[z \to u \mid t_z = s, t_v = t]\Big) \mathrm{d}s
\end{aligned}
$$

Analysis



We want to lower bound

$$\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
&= f(x_{uv}) \cdot \Pr[t_u \leq t, u \text{ unmatched } @t \mid t_v = t] \\
&= f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}$$

where

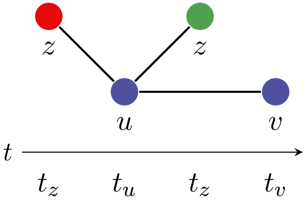
$$\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
&= \sum_{z \neq u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}$$

Recursive analysis

$$\begin{aligned}
&\Pr[(u,z) \text{ matched before } t \mid t_v = t] \\
&= \int_0^t \Big(\Pr[u \to z \mid t_u = s, t_v = t] + \Pr[z \to u \mid t_z = s, t_v = t] \Big) \mathrm{d}s
\end{aligned}$$

Need an upper bound rather than a lower bound.

Analysis



We want to lower bound

$$
\begin{aligned}
&\Pr[v \to u \mid t_v = t] \\
&= f(x_{uv}) \cdot \Pr[t_u \le t, u \text{ unmatched } @t \mid t_v = t] \\
&= f(x_{uv}) \cdot (t - \Pr[u \text{ matched } @t \mid t_v = t])
\end{aligned}
$$

where

$$
\begin{aligned}
&\Pr[u \text{ matched } @t \mid t_v = t] \\
&= \sum_{z \ne u,v} \underbrace{\Pr[(u,z) \text{ matched before } t \mid t_v = t]}_{\text{upper bound}}
\end{aligned}
$$

Recursive analysis

$$
\begin{aligned}
&\Pr[(u,z) \text{ matched before } t \mid t_v = t] \\
&= \int_0^t \Big(\Pr[u \to z \mid t_u = s, t_v = t] + \Pr[z \to u \mid t_z = s, t_v = t] \Big) \mathrm{d}s
\end{aligned}
$$

Need an upper bound rather than a lower bound. **1-regularity!**

Conclusion

Theorem (Our result)

For general graphs, there is a $\frac{8}{15} \approx 0.533$*-approximation algorithm for maximum weight matching in query-commit (and price of information) model.*

Conclusion

Theorem (Our result)

For general graphs, there is a $\frac{8}{15} \approx 0.533$*-approximation algorithm for maximum weight matching in query-commit (and price of information) model.*

- There is no vertex arrival RCRS for matching better than $\frac{1}{2} + \frac{1}{2e^2} \approx 0.567$-selectable.

Conclusion

Theorem (Our result)

For general graphs, there is a $\frac{8}{15} \approx 0.533$*-approximation algorithm for maximum weight matching in query-commit (and price of information) model.*

- There is no vertex arrival RCRS for matching better than $\frac{1}{2} + \frac{1}{2e^2} \approx 0.567$-selectable.
- One interesting open problem is to close the gap here.

Thank you!