When LP is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings

Jian Li, University of Maryland, College Park

Joint work with Nikhil Bansal (IBM), Anupam Gupta (CMU), Julian Mestre (MPI), Viswanath Nagarajan (IBM), Atri Rudra (SUNY Buffalo)

Problem Definition

Stochastic Matching [Chen, Immorlica, Karlin, Mahdian, and Rudra'09]

- Given:
  - A probabilistic graph $G(V,E)$.
  - Existential prob. p_e for each edge e.
  - Patience level t_v for each vertex v.
- Probing $e=(u,v)$: The only way to know the existence of e.
  - We can probe (u,v) only if $t_u>0, t_v>0$.
  - If e indeed exists, we should add it to our matching.
  - If not, $t_u = t_u - 1$, $t_v = t_v - 1$.

Problem Definition

- Output: A strategy to probe the edges
  - Edge probing: an (adaptive or non-adaptive) ordering of edges
  - Matching-probing: *k* rounds; In each round, probe a matching
- Objectives:
  - Unweighted Max. *E[cardinality of the matching]*
  - Weighted: Max. *E[weight of the matching]*

Motivations

- Online dating
  - Existential prob p_e : estimation of the success prob. based on dz ă Ŝ NJ ă Ω LJNJ ₴ ₸ ƛ f Ŝ ă Φ

Motivations

- Online dating
  - Existential prob. p_e : estimation of the success prob. based on dz ă Ŝ NJ ă Ω LJNJ ƨ Ŧ ƛ f Ŝ ă Φ
  - Probing edge $e=(u,v)$: u and v are sent to a date.

A LP Upper Bound

- Variable y_e : Prob. that any algorithm probes e

$$\text{maximize} \quad \sum_{e \in E} w_e \cdot x_e$$

$$\text{subject to} \quad \sum_{e \in \partial(v)} x_e \cdot \ 1 \quad \forall v \in V$$

At most 1 edge in $\partial(v)$ is matched

$$\sum_{e \in \partial(v)} y_e \cdot \ t_v \quad \forall v \in V$$

At most t_v edges in $\partial(v)$ are probed

$$x_e = p_e \cdot y_e \quad \forall e \in E$$

$$0 \cdot \ y_e \cdot \ 1 \quad \forall e \in E$$

x_e: Prob. e is matched

A Simple 8-Approximation

An edge (u,v) is *safe* if $t_u > 0, t_v > 0$ and neither u nor v is matched

Algorithm:

- Pick a permutation π on edges uniformly at random
- For each edge e in the ordering π, do:
  - If e is not safe then do not probe it.
  - If e is safe then probe it w.p. y_e/α.

A Simple 8-Approximation

Analysis:

Lemma: For any edge (u,v), at the point when (u,v) is considered under π, *Pr(u loses its patience) ≤ 1/2α* .

Proof: Let U be #probes incident to u and before e

$$\mathbb{E}[U] = \sum_{e \in \partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi \text{ AND } e \text{ is probed}],$$

$$\cdot \sum_{e \in \partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi] \cdot \frac{y_e}{\alpha},$$

$$= \sum_{e \in \partial(u)} \frac{y_e}{2\alpha} \cdot \frac{t_u}{2\alpha}.$$

By the Markov inequality: $\Pr[U \geq t_u] \cdot \frac{\mathbb{E}[U]}{t_u} \cdot \frac{1}{2\alpha}.$

A Simple 8-Approximation

Analysis:

Lemma: For any edge $e=(u,v)$, at the point when (u,v) is considered under π, $Pr(u \text{ is matched}) \leq 1/2\alpha$.

Theorem: The algorithm is a-approximation.

Proof: When e is considered,

$Pr(e \text{ is not safe}) \leq Pr(u \text{ is matched}) + Pr(u \text{ loses its patience}) +$

$Pr(v \text{ is matched}) + Pr(v \text{ loses its patience}) \leq 2/\alpha$

Therefore,

$E[\text{our solution}] = \Sigma_e w_e Pr(e \text{ is safe}) (y_e/\alpha) p_e$

$\geq (1-2/\alpha)(1/\alpha) \Sigma_e w_e y_e p_e \geq 1/8 \, OPT \quad (\alpha=4)$

Recall $\Sigma_e w_e y_e p_e$ is an upper bound of OPT

An Improved Approx. for Bipartite Graphs

Algorithm:

- (x,y) ...
- y' ... integral solution using dependent rounding [Gandhi et al. 06] and Let $E' = \{e \mid \hat{e}_e = 1\}$.
  - (Marginal distribution) $\Pr(\hat{e}_e = 1) = y_e$;
  - (Degree preservation) $Deg_{E'}(v) \le t_v$; (Recall $\sum_{e \in \partial(\varpi)} y_e \le t_v$)
  - (Negative Correlation) $\Pr(\bigwedge_{e \in \mathcal{S}} (\hat{e}_e = 1)) \le \prod_{e \in \mathcal{S}} y_e$.
- Probe the edges in E' in random order.

For matching probe model:

- $M_1, \ldots, M_h$ ...
- Probe $\{M_1, \ldots, M_h\}$ in random order.

Final Remarks and Open Questions

- Quite recently, Adamczyk has proved that the greedy algorithm is a 2-approximation for the unweighted version.
- Better approximations? (Unweighted: 2; Weighted bipartite: 3; Weighted: 4).
- o(k)-approximation for stochastic k-set packing? Or $\Omega(k)$ is the best possible?
- Any lower bound?

Thanks

An Improved Approx. for Bipartite Graphs

Analysis (sketch):

! ă ă dz Y Ŝ ó Ŝ Ƙ I· Ø Ŝ Ŏ Ƙ ƨ ă Ŝ y 9 Ω Φ

Consider a particular edge $e = (u, v)$.

Let $B(e, \pi)$ be the set of incident edges that appear before e in the random order π.

$$\Pr\left[\, e \text{ is safe} \mid E' \,\right] \geq \mathbb{E}_\pi\left[\prod_{f \in B(e,\pi)} (1 - p_f) \mid E' \right];$$

We claim that

$$\mathbb{E}_\sigma\left[\prod_{f \in B(e,\sigma)} (1 - p_f) \mid E' \right] = \int_0^1 \prod_{f \in \partial_{E'}(e)} (1 - x p_f)\, \mathrm{d}x.$$

An Improved Approx. for Bipartite Graphs

Analysis cont: To see $\mathbb{E}_\sigma\left[\prod_{f\in B(e,\sigma)}(1-p_f)\mid E'\right]=\int_0^1\prod_{f\in\partial_{E'}(e)}(1-xp_f)\,\mathrm{d}x.$

Consider this random experiment: For each edge in $\partial_{E'}(e)$, we pick a random real in [0,1]. This produces a uniformly random ordering. Let r.v. $A_f = (1-p_f)$ if f goes before e, and $A_f = 1$ o.w.

Then, we consider $\mathbb{E}[\prod_{f\in\partial_{E'}(e)}A_f]=\int_0^1\mathbb{E}\left[\prod_{f\in\partial_{E'}(e)}A_f\mid a_e=x\right]\mathrm{d}x$

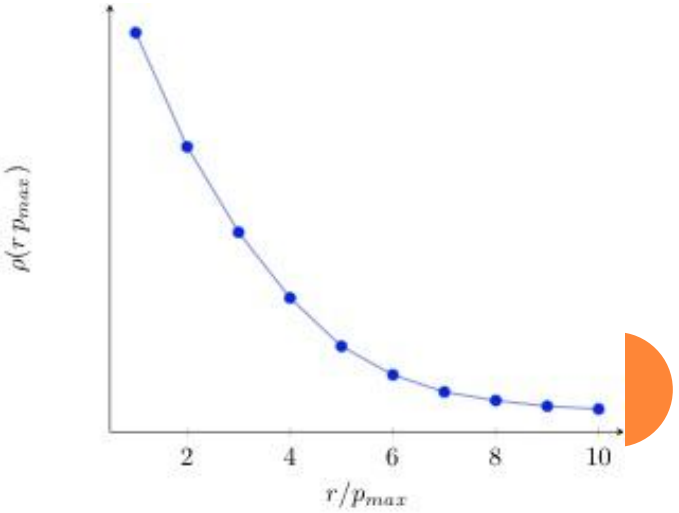
An Improved Approx. for Bipartite Graphs

Analysis cont:

Define $\rho(r, p_{\max})$ to be the optimal value of

$$\begin{aligned} \text{maximize} \quad &= \int_0^1 \prod_{f \in \partial_{E'}(e)} (1 - x p_f)\, \mathrm{d}x \\ \text{subject to} \quad &\sum_{f \in \partial_{E'}(e)} p_f \cdot r \\ &0 \cdot p_f \cdot p_{\max}. \end{aligned}$$

We can show $\rho(r, p_{\max})$ is convex and decreasing on r



An Improved Approx. for Bipartite Graphs

Analysis cont:

$$\mathbb{E}[\,\mathrm{ALG}\,] = \sum w_e\, p_e \boxed{\Pr[e \in E']} \cdot \Pr[\, e \text{ was safe} \mid e \in \widehat{E}\,]$$

Marginal Prob.

exity.

ion.

An Improved Approx. ς General Graphs

Algorithm:

- *(x,y)* н h LJ ű ƛ Y l· ƭ ă ƻ ƭ dz ű ƛ ƻ y ƻ Ŧ ű Ƙ Ŝ
- Randomly partition vertices into *A* and *B.*
- Run the previous algorithm on the bipartite graph G(A,B).

Thm: It is a 2/ ´(1, p_{max})-approximation.

If p_{max} Ŋ м Σ ű Ƙ Ŝ NJ l· ű ƛ Ŋ p_{max} Ŋ ű Ŝ y Ř ƙ Ŝ ű NJ l· ű Φ to 2/(1- м к Ŝ ύ Ŧ о Φ м р