Fair Matchings

Shruti Patil

Chennai Mathematical Institute

April 2024

Overview

- What are fair matchings?
- An efficient combinatorial algorithm based on LP duality to compute a fair matching in G.
- In designing our combinatorial algorithm, we show how to solve a generalized version of the minimum weighted vertex cover problem in bipartite graphs, using a single-source shortest paths computation
- Optimality

Definition

- Let $G = (A \cup B, E)$ be a bipartite graph, where every vertex ranks its neighbors in an order of preference (with ties allowed) and let r be the largest rank used.
- A matching M is fair in G if it has maximum cardinality, subject to this, M matches the minimum number of vertices to rank r neighbors, subject to that, M matches the minimum number of vertices to rank $(r-1)$ neighbors, and so on.

- The problem can be solved in polynomial time by defining a weight function as follows:

$$\text{for any edge } e \text{ with incident ranks } i \text{ and } j,$$
$$w(e) = n^{(i-1)} + n^{(j-1)}.$$

- A maximum cardinality matching of minimum weight is a fair matching in G. Such a matching can be found using the maximum weight matching algorithm by setting $w(e) = 4^{nr} - n^{i-1} - n^{j-1}$ (r is the largest rank used in any preference list).
- This is expensive even if we use the fastest maximum-weight bipartite matching algorithms. The running time will be O(rmn)
- One technique is an iterative combinatorial algorithm for the fair matching problem with running time $\tilde{O}\left(r^* m\sqrt{n}\right)$.

- This algorithm is based on LP Duality and in ith iteration, we solve dual to a variant of max-weight matching problem.

- *Generalized minimum weighted vertex cover problem*:
  Let $G_i = (A \cup B, E_i)$ be a bipartite graph with edge weights given by $w_i : E_i \to \{0, 1, \ldots, c\}$. $K_{i-1} \subseteq A \cup B$ satisfy the property that there is a matching in G that matches all $v \in K_{i-1}$. Find a cover $\{y_u^i\}_{u \in A \cup B}$ so that $\sum_{u \in A \cup B} y_u^i$ is minimized subject to
  (1) for each $e = (a, b)$ in E_i, we have $y_a^i + y_b^i \geq w_i(e)$
  (2) $y_u^i \geq 0$ if $u \notin K_{i-1}$.

A Combinatorial technique

Input : $G = (A \cup B, E)$ and r is the worst (largest) rank used in any preference list.
For $i \in \{1, \ldots, (r-1)\}$,

$$w_i(e) =$$

$$\begin{cases} 2 & \text{if both } a \text{ and } b \text{ rank each other as rank } \leq r-i \text{ neighbors} \\ 1 & \text{if exactly one of } \{a, b\} \text{ ranks the other as a rank } \leq r-i \text{ neighbor} \\ 0 & \text{otherwise} \end{cases}$$

$$e = (a, b).$$

Signature

For a matching M in G,

$$\textbf{signature}\ (M) = (|M|, w_1(M), \ldots\ w_{r-1}(M)),$$

where $w_i(M) = \sum_{e \in M} w_i(e)$, for $1 \le i \le r-1$.

- Let $w_0 : E \to \{1\}$ so that $|M| = w_0(M) = \sum_{e \in M} w_0(e)$.

For any matching M and $0 \le j \le r-1$,
$\textbf{signature}_j(M) = (w_0(M), \ldots, w_j(M))$ be the $(j+1)$ -tuple

- Let OPT be a fair matching. Then signature(OPT) $\succeq$ signature (M) for any matching M in G, where $\succeq$ is the lexicographic order on signatures.

A matching M is $j+1$-optimal if signature$_j(M) =$ signature $_j(\mathrm{OPT})$.

Algorithm

- Our algorithm runs for r^* iterations, where $r^* \leq r$ is the largest index i such that $w_{i-1}(OPT) > 0$.
- For any $j \geq 0$, the $j+1$-th iteration of this algorithm will our algorithm solves the minimum weighted vertex cover problem in a subgraph G_j, By computing a maximum w_j -weight matching M_j with the constraint, all vertices of a critical subset $K_{j-1} \subseteq A \cup B$ have to be matched.
- The problem of computing the matching M_j will be our primal of the $j+1$-th iteration and the minimum weighted vertex cover problem is the dual.
- $j = 0$ corresponds to $G_0 = G$ and $K_{-1} = \emptyset$.
- Since the constraint matrix is totally unimodular so the corresponding polytope is integral, the problem is solved as a linear program.

Primal and Dual

Primal :

$$\max \sum_{e \in E} w_j(e) x_e^j$$

$$\sum_{e \in \delta(v)} x_e^j \leq 1 \qquad \forall v \in A \cup B$$

$$\sum_{e \in \delta(v)} x_e^j = 1 \qquad \forall v \in K_j$$

$$x_e^j \geq 0 \qquad \forall e \text{ in } G_j$$

Dual :

$$\min \sum_{v \in V} y_v^j$$

$$y_a^j + y_b^j \geq w_j(e) \quad \forall e = (a, b) \text{ in } G_j$$

$$y_v^j \geq 0 \quad \forall v \in (A \cup B) \backslash K_j$$

$\delta(v)$ is the set of edges incident on vertex v.

Proposition

- M_j and y^j are the optimal solutions to the primal and dual programs of the j-th iteration, iff the following holds:
  1. if u is unmatched in M_j (thus u has to be outside K_j), then $y^j_u = 0$;
  2. if $e = (u, v) \in M_j$, then $y^j_u + y^j_v = w_j(e)$.

  Follows from the complementary slackness conditions in the LP duality theorem. This gives us :
- prune edges : if $e = (u, v)$ and $y^j_u + y^j_v > w_j(e)$, then no optimal solution to the primal program of the $j + 1$-th iteration can contain e. So we prune such edges from G_j and let G_{j+1} denote the resulting graph. The graph G_{j+1} will be used in the next iteration.

Solving the Dual Problem

- expand the set K_{j-1} : if $y_u^j > 0$ and $u \notin K_{j-1}$, then u has to be matched in every optimal solution of the primal program of the $j+1$-th iteration. So u should be added to the critical set. Adding vertices like u to K_{j-1} gives the critical set K_j for the next iteration.

Solving the dual

- For $1 \leq j \leq r-1$,
  let $G_j = (A \cup B, E_j)$ be the subgraph in the j-th iteration and
  let $K_{j-1} \subseteq A \cup B$ be the critical set of vertices.
- For every $e \in E_j$, $w_j(e) \in \{0, 1, 2\}$.
  We show how to solve the dual problem efficiently for
  $w_j(e) \in \{0, 1, \ldots, c\}$ for each $e \in E_j$.

Transformation

Let M_j be the optimal solution of the primal program. We know that M_j matches all vertices in K_{j-1}.

- Add a new vertex z to A and let $A' = A \cup \{z\}$.
  Add an edge of weight 0 from z to each vertex in $B \setminus K_{j-1}$.
- Direct all edges $e \in E_j \setminus M_j$ from A' to B,
  set the edge weight $d(e) = -w_j(e)$;
- Also direct all edges in M_j from B to A',
  let the edge weight $d(e) = w_j(e)$.
- Create a source vertex s and add a directed edge of weight 0 from s to each unmatched vertex in A'. (Fig. 1)

Let $\mathcal{R}$ denote the set of all vertices in $A' \cup B$ that are reachable from s.

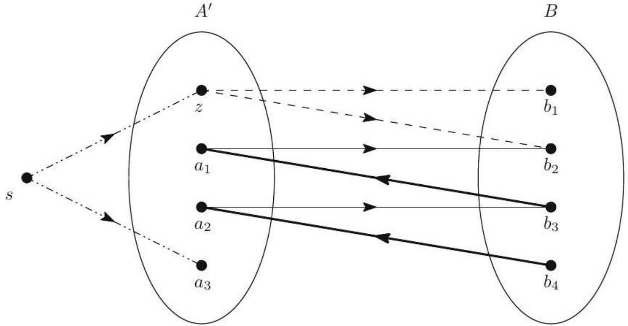

Fig. 1 The bold edges are edges of M_j and are directed from B to A' while the edges of $E_j \backslash M_j$ are directed from A' to B

Lemma

Lemma 1 : By the above transformation,

1. $B \backslash K_{j-1} \subseteq \mathcal{R}$.
2. There is no edge between $A' \cap \mathcal{R}$ and $B \backslash \mathcal{R}$.
3. M_j projects on to a perfect matching between $A' \backslash \mathcal{R}$ and $B \backslash \mathcal{R}$.

- There may be some edges in $E_j \setminus M_j$ directed from $A' \setminus R$ to $B \cap \mathcal{R}$.
- Some vertices of $A \setminus K_j$ can be in $A \setminus \mathcal{R}$.
- Delete all edges from $A' \setminus \mathcal{R}$ to $B \cap \mathcal{R}$ from G_j to get H_j
- Add a directed edge from the source vertex s to each vertex in $B \setminus \mathcal{R}$.
- Each of these edges e has weight $d(e) = 0$.
- Now all vertices can be reached from s.
- There cannot be negative-weight cycle, otherwise, we can augment M_j along this cycle and get matching of larger weight, this gives contradiction to the optimality of M_j.

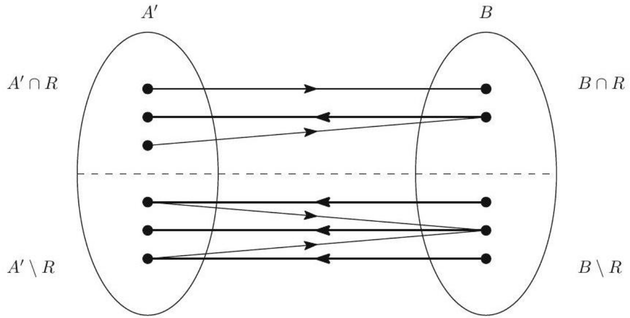

Fig. 2 The set $A' \cup B$ in the graph H_j is split into two parts:

$$(A' \cup B) \cap \mathcal{R} \text{ and } (A' \cup B) \setminus \mathcal{R}$$

- Applying the single-source shortest paths algorithm from the source vertex s in H_j where edge weights are given by $d(\cdot)$ take $O(m\sqrt{n})$ time.
- Let d_v be the distance label of vertex $v \in A' \cup B$.
- We define an initial vertex cover as :
  If $a \in A'$, let $\tilde{y}_a := d_a$; if $b \in B$, let $\tilde{y}_b := -d_b$. (will adjust this later.)
- **Lemma 2** : The constructed initial vertex cover $\{\tilde{y}_v\}_{v \in A' \cup B}$ for the graph H_j satisfies the following properties:

1. For each vertex $v \in ((A \cup B) \cap \mathcal{R}) \backslash K_{j-1}, \tilde{y}_v \geq 0$.
2. If $v \in (A \cup B) \backslash K_{j-1}$ is unmatched in M_j, then $\tilde{y}_v = 0$.
3. For each edge $e = (a, b) \in M_j$, we have $\tilde{y}_a + \tilde{y}_b = w_e^j$.
4. For each edge $e = (a, b) \in H_j$, we have $\tilde{y}_a + \tilde{y}_b \geq w_e^j$.

Proof of Lemma 2

- 1. Suppose $a \in (A \cap \mathcal{R}) \backslash K_{j-1}$ and $\tilde{y}_a < 0$.
  There is an alternating path P starting from some unmatched vertex $a' \in (A' \cap \mathcal{R}) \backslash K_{j-1}$ and ending at a.
  Since $d_a = \tilde{y}_a < 0$, the distance from a' to a along path P must be negative, Therefore,

  $$\sum_{e \in M_j \cap P} w_e < \sum_{e \in P \backslash M_j} w_e$$

  we can replace the matching M_j by $M_j \oplus P$, which gives a feasible matching with weight larger than M_j, a contradiction.
  2. $b \in (B \cap \mathcal{R}) \backslash K_{j-1}$. $\tilde{y}_b = -d_b$.
  Claim : $d_b \leq 0$. If not, then the shortest distance from s to b >0 . But this cannot be, since there is a path with edges (s, z) and (z, b), and this path has weight $= 0$.

Proof

- 1. By Lemma 1.3, an unmatched vertex must be in $\mathcal{R}$.
     If this unmatched vertex is $a \in (A \cap \mathcal{R}) \setminus K_{j-1}$, By construction, there is only one path from s to a,and its weight $= 0$.
  2. If this unmatched vertex is $b \in (B \cap \mathcal{R}) \setminus K_{j-1}$.
     By (1), $\tilde{y}_b \geq 0$. Let $\tilde{y}_b > 0$. Then $d_b = -\tilde{y}_b < 0$.
     The shortest path from s to b cannot go through $(A \cup B) \setminus \mathcal{R}$.
     The shortest path from s to b must have the edge from s to some unmatched $a \in (A' \cap \mathcal{R}) \setminus K_{j-1}$, followed by an augmenting path P (of odd length) ending at b.
     We can replace M_j by $M_j \oplus P$ to get a matching of larger weight, contradiction.

Proof

- Suppose $e = (a, b) \in M_j$. It is directed from b to a.
  e is the only incoming edge of a, So e is part of the shortest path tree rooted at s.
  As a result, $-\tilde{y}_b + w_e^j = d_b + d(e) = d_a = \tilde{y}_a$.
- Suppose $e = (a, b)$ outside M_j in H_j.
  It is directed from a to b.
  So $\tilde{y}_a - w_e^j = d_a + d(e) \geq d_b = -\tilde{y}_b$.

We possibly still do not have a valid cover for the dual program due to the following two reasons.

- Some vertex $a \in A \setminus K_{j-1}$ has $\tilde{y}_a < 0$.
- The edges deleted from G_j to form H_j are not properly covered by the initial vertex cover $\{\tilde{y}_v\}_{v \in A \cup B}$.
  We can fix this by defining $\delta = \max\{\delta_1, \delta_2, 0\}$,

$$\text{where } \delta_1 = \max_{e=(a,b)\in E} \left\{ w_e^j - \tilde{y}_a - \tilde{y}_b \right\} \quad \text{and} \quad \delta_2 = \max_{a \in A \setminus K_j} \{-\tilde{y}_a\}$$

- We can compute δ in $O(n+m)$ time.
- If $\delta = 0$, the initial cover is already a valid solution to the dual program then it is also optimal, by next theorem.

Optimality

The final vertex cover is then built as follows (assume that $\delta > 0$ exists)

1. For each vertex $u \in (A \cup B) \cap \mathcal{R}$, let $y_u = \tilde{y}_u$;
2. For each vertex $a \in A \backslash \mathcal{R}$, let $y_a = \tilde{y}_a + \delta$;
3. For each vertex $b \in B \backslash \mathcal{R}$, let $y_b = \tilde{y}_b - \delta$.

Theorem

The final vertex cover $\{y_v\}_{v \in A \cup B}$ is an optimal solution for the dual program.

Proof

- Feasibility
  By Lemma 2.1 and the choice of δ, all vertices $a \in A \backslash K_{j-1}$ have $y_a \geq 0$. By Lemma 1.1 and Lemma 2.1, all vertices $b \in B \backslash K_{j-1}$ have $y_b \geq 0$. Also by Lemma 1.2 and Lemma 2.3, and the choice of δ, all edges in E_j are properly covered. So $\{y_v\}_{v \in A \cup B}$ is feasible.

Proof

- Optimality

  We have :

$$w_j\left(M_j\right) = \sum_{e \in M_j} w_e^j$$

$$= \sum_{e=(a,b) \in M_j, b \in \mathcal{R}} (\tilde{y}_a + \tilde{y}_b) + \sum_{e=(a,b) \in M_j, b \notin \mathcal{R}} (\tilde{y}_a + \delta) + (\tilde{y}_b - \delta)$$

$$= \sum_{e=(a,b) \in M_j} y_a + y_b = \sum_{u \in A \cup B} y_u$$

  the equality holds because if a vertex u is unmatched, $\tilde{y}_u = 0$ (Lemma 2) and since u must be in $\mathcal{R}$, $y_u = \tilde{y}_u = 0$.

  By the LP duality theorem, we conclude that the cover $\{y_v\}_{v \in A \cup B}$ is optimal.