Maximum Weighted Bipartite Matching

Pramook Khungurn

December 11, 2019

1 Maximum Weight Bipartite Matching Problem

- We consider the problem of **maximum weight bipartite matching** (MWBM).
- As input, we are given a weighted bipartite graph $G = (V, E)$ where
  - $V = X \cup Y$,
  - $X \cap Y = \emptyset$, and
  - $E \subseteq X \times Y$.

  Also, there's a function $w : E \to \mathbb{R}^+ \cup \{0\}$.
- A **matching** is a subset $M \subseteq E$ such that, for every vertex $v \in V$, at most one edge in M is incident upon v.
- The **size of matching** M, denoted by $|M|$, is the number of edges in M.
- The **weight of matching** M, denoted by $w(M)$, is the sum of the weights of the edges in M. That is,

$$w(M) = \sum_{e \in M} w(e).$$

- The MWBM problem wants to find a matching M whose weight is the maximum among all possible matchings.

2 The Assignment Problem

- In the **assignement problem**, we are given a complete weighted bipartite graph, and we want to find the maximum weight matching.
- The MWBM problem can be reduced to the assignment problem.

  This can be done by:
  - introducing dummy nodes so that $|X| = |Y|$, and
  - for every pair of vertices (x, y) such that $(x, y) \notin E$, creating a new edge (x, y) with weight 0.
- A maximum weight matching in a complete bipartite graph can be made *perfect*. (That is, every vertex is incident to an edge.)
- So, the assignment problem is to find a perfect matching with maximum weight.

3 Feasible Labeling

- A **vertex labeling** is a function $\ell : V \to \mathbb{R}$.
- A **feasible labeling** is one such that

$$\ell(x) + \ell(y) \geq w(x, y)$$

  for all $x \in X$ and $y \in Y$.
- An edge (x, y) is called **tight** if $\ell(x) = \ell(y) = w(x, y)$.
- The **equality graph** with respect to a labeling ℓ is $G_\ell = (V, E_\ell)$ where E_ℓ is the set of tight edges.
- **Theorem 3.1.** *If ℓ is feasible and M is a perfect matching in G_ℓ, then M is a maximum weight matching.*

  Proof. Denote edge $e \in E$ by $e = (e_x, e_y)$.

  Let M' be any perfect matching in G (not necessarily in E_ℓ). Since every vertex $v \in V$ is incident to exactly one edge in M', we have that

$$w(M') = \sum_{e \in M'} w(e) \leq \sum_{e \in M'} (\ell(e_x) + \ell(e_y)) = \sum_{v \in V} \ell(v).$$

  Hence, $\sum_{v \in V} \ell(v)$ is an upper bound on the cost of any perfect matching.

  Now, let M be a perfect matching in E_ℓ. Then, $w(M) = \sum_{e \in M} w(e) = \sum_{v \in V} \ell(v)$. So, $w(M') \leq w(M)$ and M is optimal. □

- If you wonder where the heck the above theorem comes from, it comes from writing the matching problem as a linear programming and take the dual.

4 The Hungarian Algorithm

- With respect to a graph G (not necessarily complete) a matching M in G,
  - a vertex is **free** is it is incident to no edges in M,
  - a vertex is **matched** if it is not free,
  - a path in G is **alternating** if its edges alternate between M and $E - M$,
  - a path is **augmenting** if both end points are free, and
  - the **residual graph** of G with respect to M is a directed graph $G' = (V', E')$ where
    - $V' = V$, and
    - for each edge $(x, y) \in E$,
      - if $(x, y) \notin M$, then $(x, y) \in E'$, and
      - if $(x, y) \in M$, then $(y, x) \in E'$
- The sketch of the algorithm is as follows:
  1. Start with a feasible labeling ℓ, and a maximum size matching M in G_ℓ.
  2. If M is perfect, we are done.
  3. If not, then then we find another feasible labeling ℓ' such that $E_\ell \subset E_{\ell'}$.
     Then, we set ℓ to ℓ', recompute M, and go back to Step 2.

- After Step 3, either M or E_ℓ increases in size. Hence, the algorithm must terminate.
- An initial feasible labeling is given by:
  - $\ell(y) = 0$ for all $y \in Y$, and
  - $\ell(x) = \max y \in Y\{w(x,y)\}$ for all $x \in X$.
- Before we go on to find how to find labeling ℓ' such that $E_\ell \subset E_{\ell'}$, we need to define one more set of terminology.
- Let ℓ be a feasible labeling.

  The **neighbor** of a vertex $u \in V$ is the set $N_\ell(u) = \{v : (u,v) \in E_\ell\}$.

  The **neighbof** of the set $S \subseteq V$ is the set $N_\ell(V) = \bigcup_{u \in S} N_\ell(u)$.
- The process of finding ℓ' where $E_\ell \subset E_{\ell'}$ uses the following lemma:

  Lemma 4.1. *Let $S \subseteq X$ and $T = N_\ell(S) \neq Y$. Let*

  $$\alpha_\ell = \min_{x \in S, y \neq T} \{\ell(x) + \ell(y) - w(x,y)\}.$$

  Define ℓ' as follows:

  $$\ell'(v) = \begin{cases} \ell(v) - \alpha_\ell, & \text{if } v \in S, \\ \ell(v) + \alpha_\ell, & \text{if } v \in T, \\ \ell(v), & \text{otherwise.} \end{cases}$$

  Then, ℓ' is a feasible labeling, and
  - *if $(x,y) \in E_\ell$ for $x \in S$ and $y \in T$, then $(x,y) \in E_{\ell'}$,*
  - *if $(x,y) \in E_\ell$ for $x \notin S$ and $y \notin T$, then $(x,y) \in E_{\ell'}$, and*
  - *there exists some edge $(x,y) \in E_{\ell'}$ for $x \in S$ and $y \notin T$.*

  Proof. We first show that $E_{\ell'}$ is a feasible labeling. Let $x \in X$ and $y \in Y$. There four cases three cases:

  1. $x \in S$ and $y \in T$. In this case, $\ell'(x) + \ell'(y) = \ell(x) - \alpha_\ell + \ell(y) + \alpha_\ell = \ell(x) + \ell(y) \geq w(x,y)$.
  2. $x \notin S$ and $y \in T$. In this case, $\ell'(x) + \ell'(y) = \ell(x) + \ell(y) + \alpha_\ell \geq \ell(x) + \ell(y) \geq w(x,y)$. This is simply because $\alpha_\ell \geq 0$.
  3. $x \in S$ and $y \notin T$. In this case,
  $$\begin{aligned} \ell'(x) + \ell'(y) &= \ell(x) - \alpha_\ell + \ell(y) \\ &= w(x,y) + (\ell(x) + \ell(y) - w(x,y)) - \min_{x \in S, y \notin T} \{\ell(x) + \ell(y) - w(x,y)\} \\ &\geq w(x,y). \end{aligned}$$
  4. $x \notin S$ and $y \notin T$. In this case, $\ell'(x) + \ell'(y) = \ell(x) + \ell(y) \geq w(x,y)$.

  So, ℓ' is a feasible labeling.

  From the above analysis, in Case 1 and Case 4, we have that $\ell'(x) + \ell'(y) = \ell(x) + \ell(y)$. Thus, an edge (x,y) remains in $E_{\ell'}$ if it is (1) already in E_ℓ, and (2) either $x \in S$ and $y \in T$ or $x \neq S$ and $y \neq T$.

  Also, there exists an edge (x,y) with $x \in S$ and $y \notin T$ where $\ell(x) + \ell(y) - w(x,y)$ achieve its minimum. This edge cannot already be in E_ℓ; otherwise, it would already be included in the neighborhood $N_\ell(S)$. After the update, we see that $\ell(x) + \ell(y) - w(x,y)$ goes to 0. Hence, this is a new edge in $E_{\ell'}$ which is not in E_ℓ before. □

- It remains to find a set $S \subseteq X$ such that $N_S(X) \neq Y$.

  Lemma 4.2. *Let M be a maximum size matching in a bipartite graph G. Suppose there exists some vertices in X that are free. Let L be the vertices reachable from any free vertex in X in the residual graph of G with respect to M. Then, $C = (X - L) \cup (B \cap L)$ is a vertex cover and $|C| = |M|$.*

  Proof. If C is not a vertex cover, then there exists an edge $(x, y) \in E$ such that $x \in X \cap L$ and $y \in Y - L$.

  First, we claim that $(x, y) \notin E - M$. Otherwise, we have that $x \in X \cap L$, which means it is reachable from a free vertex in the residual graph. Moreover, $(x, y) \in E'$, so this we can follow a path from a free vertex to x and then to y.

  Next, we claim that $(x, y) \notin M$. Otherwise, the fact that $(x, y) \in M$ means that x is matched. Now, if a matched vertex x is reachable from a free vertex, it means that it must be reached to the directed edge (y, x). This implies that y is reachable from a free vertex, but this contradicts the fact that $y \in B - L$.

  So, such an edge (x, y) does not exist, and C is a vertex cover.

  We now show that $|C| \leq |M|$.

  First, we have that no vertices in $X - L$ are free because free vertices in X are included in L by definition.

  Also, no vertices in $Y \cap L$ are free. Otherwise, a path from a free node to a free vertex in Y exists and is an augmenting path. This contradicts the fact that M is not a maximum matching.

  Moreover, there cannot be any edge $(x, y) \in M$ where $x \in X - L$ and $y \in Y \cap L$. Othewise, x would be included in L. So, every vertex in $X - L$ and $Y \cap L$ is incident to an edge in M, but no two vertices in $(X - L) \cup (Y \cap L)$ can share an edge. This means that $|M| \geq |(X - L) \cup (Y \cap L)| = |C|$.

  Now, the size of a maximum matching is a lower bound on the size of a vertex cover. Hence, $|M| \leq |C|$. It follows that $|C| = |M|$. □

- Now, let M be a maximum matching in G_ℓ. If M is not perfect, then there exists some vertices that are free. We let L be the set of vertices reachable from these free vertices in the residual graph of G_ℓ with respect to M. We can then set $S = X \cap L$.

  Observe that $N_\ell(S) = Y \cap L$.

  It follows that $|N_\ell(S)| = |Y \cap L| \leq |(X - L) \cup (Y \cap L)| = |C| = |M| < |Y|$. So, there exists some vertex $y \in Y$ such that $y \notin N_\ell(S)$.

- The Hungarian algorithm:

  1. Generate initial labeling ℓ and maximum cardinality matching M in E_ℓ.
  2. If M is perfect, stop.
  3. Let v be a free vertex with respect to M.
     Construct an alternating tree in G_ℓ with respect to M, eninating from v.
     Let L be the set of vertices reachable from any free vertex in X, including the free vertices themselves.
  4. Set $S = X \cap L$ and $T = Y \cap L$.
     Compute the new weight ℓ' according to the process in Lemma 4.1.
  5. Add the new edge created by this process to G_ℓ and recompute M.
  6. Go to Step 2.

- We will find at most $|V|/2 = O(|V|)$ augmenting paths. Finding an augmenting path requires a breadth first search, which takes $O(|V|^2)$ because we have a complete graph. So, augmenting the paths take $O(|V|^3)$ time.

  Updating the weight takes $O(|V|^2)$ time. However, we only need up update the weight $O(|V|)$ time because, each time we update, there will always be a new augmenting path. So, updating the weights take $O(|V|^3)$ time as well.

  All in all, the algorithm takes $O(|V|^3)$ time.