1 Matching in Bipartite Graphs

- We saw earlier how a maximum cardinality matching in a bipartite graph can be computed by reduction to network flow.

- In this lecture, we revisit the matching problem, both to better understand its structural properties, and to develop an algorithm for the weighted matching. While the weighted matching can be computed using the min-cost flows, our specialized algorithm will be more efficient.

- We start with the problem of finding a perfect matching.

- Given a bipartite graph $G = (U, V, E)$, a matching is a subset of vertex disjoint edges $M \subset E$. That is, each edge of the matching pairs a vertex $u \in U$ with a unique partner $v \in V$.

- The figure shows a bipartite graph and a matching $M = \{(u_2, v_1), (u_3, v_2), (u_4, v_4)\}$ (shown as thick edges).
• When the bipartite graph has $|U| = |V|$, a matching $M$ is called perfect if $|M| = |U| = |V|$. That is, $M$ pairs every vertex with a partner.

• A fundamental question, which was answered by Hall, is the following: is there a characterization of graphs $G$ that contain a perfect matching?

• Given a subset of nodes $S \subset U$, we define its neighborhood $N(S) \subset V$ as the set of vertices that are neighbors of some vertex in $S$.

• In the figure, for instance, the neighborhood of $\{u_1\}$ is $\{v_2, v_4\}$, while the neighborhood of $\{u_1, u_2, u_3\}$ is the entire set $V$.

• **Hall’s Theorem:** $G$ has a perfect matching if and only if $|N(S)| \geq |S|$, for all $S \subset U$.

• In order to prove Hall’s theorem, we need the concept of an alternating paths.

• Suppose $G$ is a bipartite graph and $M$ is a matching in $G$. An alternating path $P$ in $G$ is one whose edges alternate between edges of $M$ and edges not in $M$.

• We also called a vertex of $G$ matched if it is incident to an edge of $M$, and free otherwise.

• An alternating path is called an augmenting path if both of its endpoints are free.

• **Fact.** If a graph with matching $M$ contains an augmenting path, then we can construct a new matching $M'$ of larger size with $|M'| = |M| + 1$. 

---

In the figure, we see an example of a matching $M$ and an augmenting path. The matching $M$ pairs the vertices $u_1, u_2, u_3, u_4$ with $v_2, v_3, v_4$ respectively. The augmenting path is highlighted, showing how it can be used to construct a new matching $M'$ with one more edge.
• The figure shows an example of augmenting path, and updating a matching of size 2 to a matching of size 3.

• **Proof of Hall’s Theorem.**

  • The direction PM ⇒ |N(S)| ≥ |S| is easy, and we reason as follows.

    1. Consider any set S ⊆ U, and let mate(u) ∈ N(S) be the vertex matched with u.
    2. mate(u_i) ≠ mate(u_j), for any i ≠ j, and so we must have |∪_{u∈S} mate(u)| ≥ |S|.
    3. Finally, since mate(u) ∈ N(u) ⊆ V, we must have |N(S)| ≥ |S|.

  • For the converse direction, we assume |N(S)| ≥ |S| holds, for all S ⊆ U, and show a perfect matching.

    1. Suppose M is a max-cardinality matching in G, and it is not perfect.
    2. Let u ∈ U be a free vertex in it, and let Z be the set of all vertices reachable from u by an alternating path.
    3. There is no free vertex in Z (except u), because otherwise we will have an augmenting path, contradicting M’s maximum size.
    4. Define L = Z ∩ U, and R = Z ∩ V.
    5. We note that N(L) = R, because otherwise we could further grow the set Z.
    6. Since each vertex in L (except u) is matched with someone in R, and the mates are distinct, we must have |R| = |L| − 1.
    7. But then |N(L)| < |L|, which is a contradiction to our hypothesis that N(S) ≥ |S|, for all S.

• This completes the proof of Hall’s theorem.

2 **Weighted Bipartite Matching: Assignment Problem**

• We now consider the problem of computing an optimal matching in a bipartite graph where edges have weights.

• This is also called the assignment problem, where the goal is to assign tasks to machines to maximize the benefit. The input is a compatibility graph, in which each edge (x, y) is associated with a non-negative weight w(x, y) ≥ 0 representing the benefit of assigning x to y.
Without loss of generality, we can assume that $G = (X, Y, E)$ is a complete bipartite graph, since any missing edge can be given weight $w(e) = 0$.

Since a complete bipartite graph with $|X| = |Y|$ always has a perfect matching, we want to compute a maximum weight perfect matching in $G$.

**Pricing Method and Feasible Vertex Labeling.** The method we describe is called the Hungarian Method, and it introduces preliminary ideas of LP duality.

At an intuitive level, we associated labels $\ell(x), \ell(y)$ with vertices, which can be interpreted as prices. An edge is only feasible if its endpoints’ total price is at least as large as the weight of the edge.

Throughout the algorithm, we will maintain real-valued labels $\ell()$ such that

$$\ell(x) + \ell(y) \geq w(x, y), \quad \forall x \in X, y \in Y.$$

We can obtain an initial feasible labeling as follows:

$$\ell(x) = \max_{y \in Y} \{w(x, y)\} \quad \text{for } x \in X$$

$$\ell(y) = 0 \quad \text{for } y \in Y$$
• [Equality Graph.] This is the subgraph $G_\ell = (X,Y,E_\ell)$ that only includes edges with equality of prices, namely,

$$E_\ell = \{(x,y) \mid \ell(x) + \ell(y) = w(x,y)\}.$$

• The main structural result on weighted matching is the following theorem.

For any feasible labeling $\ell$, if $G_\ell$ contains a perfect matching $M$, then $M$ is an optimal (maximum weight) assignment.

For instance, the following figure shows an equality graph with a perfect matching.
The proof of this lemma is straightforward:

1. In a PM, each vertex is covered exactly once, so

\[ w(M) = \sum_{e \in M} w(e) = \sum_{v \in X \cup Y} \ell(v) \]

2. Any other assignment \( M' \) in \( G \) satisfies

\[ w(M') = \sum_{e \in M'} w(e) \leq \sum_{v \in X \cup Y} \ell(v) \]

3. Thus, \( w(M') \leq w(M) \), and \( M \) must be optimal.

The Hungarian method consists of finding a labeling for which \( G_\ell \) contains a perfect matching. The algorithm works as follows.

1. Initialize vertex labeling \( \ell \). Determine \( G_\ell \).
2. Pick any matching \( M \) in \( G_\ell \).
3. If $M$ perfect, stop. Otherwise, pick a free vertex $u \in X$. Set $S = \{u\}$, and $T = \emptyset$.

4. If $N(S) = T$, update labels: $\alpha_\ell = \min_{x \in S, y \not\in T} \{\ell(x) + \ell(y) - w(x, y)\}$

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

(Now $N(S) \neq T$.)

5. If $N(S) \neq T$, pick $y \in N(S) - T$.

- If $y$ is free, then $u-y$ is an bf augmenting path; augment $M$ and go to 3.
- If $y$ is matched, say, to $z$, then extend the alternating tree as follows: $S = S \cup \{z\}$, $T = T \cup \{y\}$. Go to 4.

- Illustration of the Hungarian Method.

1. A $3 \times 3$ assignment problem.
2. Initial labels and equality graph.
3. Initial matching $(x_1, y_1), (x_2, y_2)$.
4. $S = \{x_3\}$, $T = \emptyset$.
5. Since $N(S) \neq T$, we do step 5. Choose $y_2 \in N(S) - T$.
6. $y_2$ matched, add $y_2x_2$ (grow tree).
7. Since $N(S) = T$, do step 4.

8. $S = \{x_2, x_3\}$, and $T = \{y_2\}$.

9. Calculate $\alpha$:

$$
\alpha = \min_{x \in S, y \notin T} \begin{cases} 
6 + 0 & x_2 y_1 \\
6 + 0 & x_2 y_3 \\
5 + 0 & x_3 y_1 \\
5 + 0 & x_3 y_3 
\end{cases} 
= 4
$$


11. New equality graph has a perfect matching.

### 2.1 Analysis

- Relabeling ensures that at least one new edge is added to $G_\ell$, and no edge of $G_\ell$ is removed.

- In a worst-case, all edges of $G$ would eventually appear in $G_\ell$, and so a perfect matching is guaranteed to be found.

- **Time Complexity.**
1. Algorithm has \( n \) phases, in each phase matching size grows by 1.
2. Keep track of edge with smallest slack: \( \ell(x) + \ell(y) - w(x, y) \), where \( x \in S, y \notin T \).

3. Initial slack calculation takes \( O(n^2) \) time.
4. When a vertex moves from \( \bar{S} \) to \( S \), we compute slacks for all \( y \notin T \).
5. In each phase, at most \( n \) vertices go from \( \bar{S} \) to \( S \), so \( n \) slack re-calculations, each at \( O(n) \) time, for a total of \( O(n^2) \).
6. Total algorithm takes \( O(n^3) \).

### 2.2 Other Results on Matching

- **[Bipartite Cardinality Matching]**
  \( O(\sqrt{nm}) \) time. Maxflow on unit capacity networks. [Even-Tarjan]

- When \( G \) is non-bipartite, we cannot use network flow. In fact, the matching problem in general (non-bipartite) graphs is technically much more challenging.

- Even Hall’s Theorem is no longer valid: consider a triangle graph.

- For non-bipartite graphs, the analog of Hall’s Theorem is provided by Tutte’s Theorem: \( G \) has a perfect matching iff for all \( S \subseteq V \), \( oc(G - s) \leq |S| \), where \( oc \) is number of odd-cardinality components.
• **[Non-Bipartite Cardinality Matching:]** First polynomial time, $O(n^4)$, algorithm in 1957 by Edmonds. Current best $O(\sqrt{nm})$ [Micali-Vazirani].

• **[Bipartite Weighted Matching:]** $O(nm + n^2 \log n)$ strongly poly. $O(\sqrt{nm} \log(nC))$ scaling algorithm.

• **[Non-Bipartite Weighted Matching:]** $O(n^3)$ by Edmonds+Gabow ’75. Current best $O(nm + n^2 \log n)$.

3 **Stable Matching**

• Stable Matching problem originated in 1962 when David Gale and Lloyd Shapley, two mathematical economists, asked the following question:

    Could one design a college admissions, medical internship, or job recruiting process that was self-enforcing?

• What does that mean? Think of college admissions. In senior year of high school, students apply to many colleges.

    1. Each student has a clear preference order among colleges: he/she prefers some college more than others.
    2. Similarly, colleges rank the applicants, preferring some to others.
    3. However, neither side really knows who else is offering who admission.
    4. A student $S$ may prefer college $C$ but if $C$ has not made him an offer yet, he may end up accepting college $D$, only to learn later than $C$ offers him admission, and at which point $C$ finds out that its top student chose some other college.

• This can, and does, become easily quite chaotic. In fact, what happened in medical school internship was the straw that broke the camel’s back. College were offering earlier and earlier admissions, and asking for pre-commitments. At some point, they were reaching back to sophomores! This lead to the creation of National Resident Matching Program.

• What Gale-Shapley asked was: **is there a matching in which no “pair has a regret”**. That is, no student or college wishes to retract its offer.

• Lloyd Shapley and Alvin Roth received 2012 Nobel Prize in Economics, for “theory of stable allocations and practice of market design.” (David Gale had died in 2008.)
4 Stable Matching Problem

- The problem abstracted as one of arranging “marriage” in a society of $n$ men ($A, B, \ldots, Z$) and $n$ women ($a, b, \ldots, z$).
- Each man (woman) ranks all women (man), in descending order of preference.
- An example for $n = 3$, with the preference lists:
  - $A : c > a > b$
  - $B : c > b > a$
  - $C : a > b > c$
  - $a : C > B > A$
  - $b : B > A > C$
  - $c : B > C > A$

- A matching is a 1-to-1 correspondence (monogamous, heterosexual marriage).
- A pair $(m, w)$ is unstable if $m$ and $w$ like each other more than their assigned partners.
- A matching is called unstable if it has a unstable pair (e.g. risk of elopement).
- Does a stable matching exist? How does one find it?
- In our example, the matching \{$(A, c), (B, b), (C, a)$\} has an unstable pair $(B, c)$.
- On the other hand, the matching \{$(A, b), (B, c), (C, a)$\} is stable.

4.1 Stable Matching Algorithm

- Gale-Shapley showed that a stable matching always exists for any preferences. Constructive proof: they gave an algorithm to find a stable matching.
- Basic principle: Man proposes, woman disposes.
- Each unattached man proposes to the highest-ranked woman in his list, who has not already rejected him.
- If the man proposing to her is better than her current mate, the woman dumps her current partner, and becomes engaged to the new proposer.
• Since no man proposes to the same woman twice, the algorithm terminates, and the proof will show that the result is a stable matching.

• In particular, the algorithm has the following high-level description.

    1. \textit{LIST}: list of unattached men.
    2. \textit{cur}(m): highest ranked woman in \( m \)'s list, who has not rejected him.
    3. Initialize \( \text{LIST} = \{1, 2, \ldots, n\} \) and \( \text{cur}(i) = M(i, 1) \).
    4. Choose a man, say, Bob, from \( \text{LIST} \). Bob proposes to Alice, where Alice = \text{cur}(Bob).
    5. If Alice unattached, Bob and Alice are engaged.
    6. If Alice is engaged to, say, John, but prefers Bob, she dumps John, and Bob and Alice are engaged. Otherwise, she rejects Bob.
    7. The rejected man rejoins \( \text{LIST} \), and updates his \textit{cur}.
    8. Output the engaged pairs when \( \text{LIST} = \emptyset \).

• Illustrate on the example.

• The algorithm terminates in \( O(n^2) \) steps, since each step moves one \textit{cur} pointer, and there are at most \( n^2 \) preferences.

• Remains to prove the matching is always stable.

4.2 Correctness of GS

• Proof by contradiction: suppose the resulting matching has a unstable pair (Dick, Laura). That is, Dick and Laura both prefer each other to their assigned mates.

• Dick must have proposed to Laura at some point.

• During the algorithm, Laura also rejected Dick in favor of some she prefers more.

• Since no woman ever switches to a man less desirable than her current partner, Laura’s current partner must be more desirable than Dick.

• Thus, the pair (Dick, Laura) is not unstable.
4.3 Properties of Gale-Shapley

- In fact, the GS matching has many interesting properties, in addition to just being stable.

- Fact 1. Any women $w$ remains engaged from the point at which she receives her first proposal. The sequence of partners she is engaged to improves with time.

- Fact 2. The sequence of women to whom a man $m$ proposes gets worse and worse.

- If a man $m$ is free at some point then there must be a woman he has not proposed yet. For $n$ women to be engaged, there must be $n$ men engaged.

- The matching at the end is a perfect matching. In fact, all executions of the algorithm yield the same stable matching.

- The following characterization of the matching found may seem surprising:

  Let $best(m)$ denote the best possible woman $m$ can be married to in any stable matching. Then, the matching found is $(m,best(m))$.

- This may be surprising at many levels.

  1. First, there is no reason to believe that $(m,best(m))$ is a matching at all, much less stable.

  2. After all, why can’t it be the case that two men have the same best valid partner.

  3. Second, it shows that the simple GS algorithm finds such a matching, meaning there is no matching in which men could hope to do better.

  4. Finally, it shows that the order of proposals has not effect on the final matching.

- Proof.

  1. We prove the man-optimality of GS by contradiction.

  2. Consider the first time when some man $m$ is rejected by his best valid partner $w = best(m)$ in the GS matching. (Valid pairings only consider partners in stable matchings.)

  3. The woman $w$ rejects $m$ in favor of, say, $m'$ who she likes more than $m$.

  4. Let us mark this instant as $R$ (rejection), and return to it later to derive a contradiction.

  5. By the hypothesis that $w = best(m)$, there exists a stable matching $S'$ in which $m$ and $w$ are paired.
6. In this matching $S'$, the man $m'$ is paired with someone else, say, $w' \neq w$. Thus, $S'$ contains stable pairs $(m, w)$ and $(m', w')$.

7. We now consider what the execution of GS tells us about the preferences of $m'$ between $w$ and $w'$.

8. Since $R$ was the first event in GS where any man was rejected by his best valid partner, at the instant $R$ the following must be true:

   (a) $m'$ has not been rejected by his $\text{best}(m')$, and therefore has not been rejected by any of his valid partners. (Recall that valid partners are those pairs that are stable, with best being the highest ranked such partner.) The rank of $w'$ in preference list of $m'$ cannot be higher than that of $\text{best}(m')$, it also means that $m'$ has not been rejected by $w'$.

   (b) because $m'$ is paired with $w$, he must have been rejected by every woman that comes before $w$ in his list.

9. Therefore, $w'$ must be after $w$ in the preference list of $m'$. That is, $m'$ prefers $w$ to $w'$.

10. But this contradicts the stability of $S'$ which has $(m, w)$ and $(m', w')$ pairings: both $m'$ and $w$ prefer each other to their assigned partners in $S'$.

11. Thus, our initial assumption that $m$ was rejected by $\text{best}(m)$ is false.

• One can also show that in stable matching GS, each woman is paired with the worst possible man.