Matchings in Graphs

Lecturer: *Rajesh Chitnis* Meeting: 10
Scribe: *Meena Mahajan* 25 March

Today we consider a variant of the classical stable matching problem. This variant is called the popular matching problem (see [**?**, **?**]), which is a bit ironic since it is relatively less well-known! It was first formulated in 1975, and after a long period of no progress, the first polynomial time algorithm was given in 2005 in [**?**].

The setting is as follows: There is a set $A = \{a_1, \dots, a_n\}$ of applicants, and a set $P = \{p_1, p_2, \dots, p_t\}$ of posts. For each $a \in A$, there is a non-empty subset $\mathsf{List}(a) \subseteq P$ of posts a is willing to take up, with a preference order on $\mathsf{List}(a)$ allowing ties. We construct the set E of edges (a, p) where $p \in \mathsf{List}(a)$. Rank-1 edges are edges from some a to a top-choice of a; rank-2 edges are edges from some a to second-choice posts of a and so on. Let r be the maximum depth of the preference order. Then E can be partitioned into $E_1, E_2, \dots, E_r$ where E_i consists of rank-i edges.

Let M be a matching. For $u \in A \cup P$, we say that $u \in M$ if u is matched in M. If $u \in M$, we denote by $M(u)$ the vertex matched to u in M.

Let M, M' be matchings in G. Applicant a prefers M' over M if

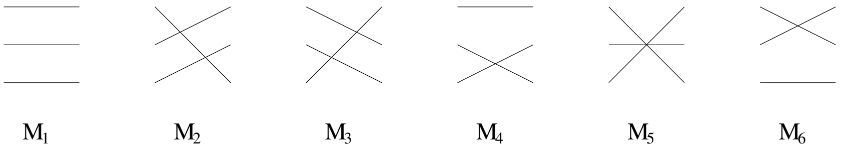
- $a \notin M$ and $a \in M'$, or
- $a \in M$, $a \in M'$, and a prefers $M'(a)$ to $M(a)$.

We say that M' is more popular than M, denoted $M' >_{\text{pop}} M$, if the number of applicants preferring M' to M exceeds the number of applicants preferring M to M'.

A matching M^* is said to be **popular** if there is no M such that $M >_{\text{pop}} M^*$.

Note that it is not at all clear why a popular matching must exist. And with good reason – there are indeed graphs where there is no popular matching. This happens because the $>_{\text{pop}}$ relation is not acyclic.

Example 1 *Suppose there are three applicants and three posts, and all three have preference $p_1 > p_2 > p_3$. Consider the matchings of size 3:*



Then $M_3 >_{pop} M_2 >_{pop} M_1 >_{pop} M_3$, and $M_6 >_{pop} M_5 >_{pop} M_4 >_{pop} M_6$. Thus there is no popular matching of size 3.

We want to address the following questions:

1. Decide if there is a popular matching,

2. If yes, find one.

3. If yes, find one of maximum size.

The first question is simpler to tackle if we consider only matchings where all applicants are matched, that is, A-perfect matchings. But there may not be such a matching that is also popular. The following result however shows that we can reduce the general problem to the special case.

Lemma 2 *Popular matchings reduces to instances where if there is a popular matching, there is an A-perfect matching.*

Proof: Let $G = (A \cup P, E)$ be an instance of popular matchings, with lists $L(a)$ for $a \in A$. Introduce n new last-resort posts, one per applicant; call these posts $l(a)$ for $a \in A$. Augment the list $\mathsf{List}(a)$ of each applicant a with $l(a)$, placed at the end of the list. Note that $l(a)$ appears only in $\mathsf{List}(a)$ and in no other list. Call this instance H.

The following map from matchings in G to A-perfect matchings in H is a bijection: For matching M in G, define $M' = M \cup \{(a, l(a) \mid a \notin M\}$. It is easy to see that M is popular in G if and only if M' is popular in H.

Thus, G has a popular matching if and only if H has an A-perfect popular matching. This is the desired reduction. ■

Henceforth, without loss of generality, we assume that we are looking for A-perfect popular matchings. We also assume that for each a, there is a reserved last-resort post $l(a)$.

Addressing question 1 above, we first consider the easy case when there are no ties; each applicant has a total order on the elements of $\mathsf{List}(a)$. This means that in each of the graphs $G_i = (A \cup P, E_i)$, each a is either isolated or pendant.

Definition 3 1. *For each $a \in A$, let $f(a)$ denote the unique first post in* $\mathsf{List}(a)$.

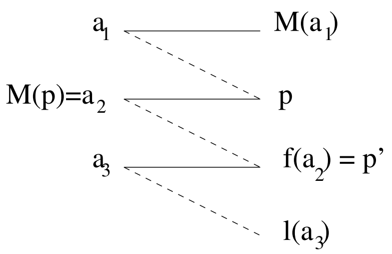
2. *A post $p \in P$ is called an f-post if it is some applicant's first choice; $p = f(a)$ for some $a \in A$.*

3. *If p is an f-post, let $f(p)$ denote the set of applicants for whom it is the first choice; $f(p) = \{a \in A \mid f(a) = p\}$.*

Lemma 4 *In every popular matching, no f-post is unmatched, and all f-posts are matched to applicants for whom they are first choice. That is, if M is popular and p is an f-post, then $p \in M$ and $M(p) \in f(p)$.*

Proof: Let M be a popular matching and let p be an f-post.

If $p \notin M$, then pick any a such that $f(a) = p$. If $a \notin M$, then let $M' = M \cup \{(a, p)\}$. Otherwise, let $M' = M \setminus \{(a, M(a))\} \cup \{(a, p)\}$. Then a prefers M' to M, and all other applicants are indifferent between M' and M. So $M' >_{\text{pop}} M$, contradicting the claim that M is popular. So every f-post p must be matched in M.



If $M(p) \notin f(p)$, pick any $a_1 \in f(p)$. Let $a_2 = M(p)$. By assumption, $p \neq f(a_2) = p'$, say. Since p' is an f-post, as shown above, it is matched in M. Let $a_3 = M(p')$. Now construct M' by promoting a_1 to her first choice p, promoting a_2 to her first choice p', and demoting a_3 to $l(a_3)$. See the figure alongside; solid lines represent edges in M, dashed lines represent edges in M'. (Note, $M(a_1)$ may not even exist.) Clearly, a_1 and a_2 prefer M' to M since they now get their first choices, while only a_3 prefers M to M'. So $M' >_{\text{pop}} M$, contradicting the claim that M is popular.

■

Definition 5 *For each $a \in A$, let $s(a)$ denote the first post in $\mathsf{List}(a)$ that is not an f-post.*

Note that $s(a)$ need not be a's second choice; it can be far down the list in $\mathsf{List}(a)$. However, it is always defined, because $l(a)$ is not an f-post.

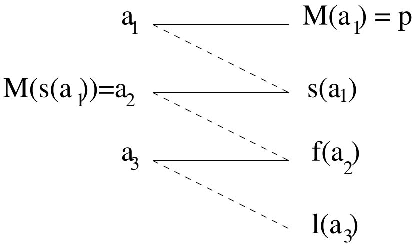
The following lemma says that each applicant either gets her first-choice post, or the first post in her list that is not anyone else's first-choice post.

Lemma 6 *In any popular matching M, for all $a \in A$, $M(a) \in \{f(a), s(a)\}$.*

Proof: Let M be a popular matching, and let a be an arbitrary applicant with $M(a) = p$.

Case 1: p lies above $s(a)$ in the preference order of a. Since p is above $s(a)$, p is the first-choice post for at least one applicant. By Lemma 4, $M(p) \in f(p)$. Since $a = M(p)$, we have $p = f(a)$.

Case 2: p lies below $s(a)$ in the preference order of a.



If $s(a) \notin M$, then promoting a to $s(a)$ gives a matching more popular than M. Otherwise, let $a_1 = a$ and $a_2 = M(s(a))$. By definition of $s(a)$, it is not the first-choice post of any applicant; in particular, $s(a) \neq f(a_2)$. If $f(a_2) = p$, then by Lemma 4, $a = M(p)$ has p as first choice; that is, $f(a) = p$. Since this is not the case, we conclude that $f(a_2) \neq p$. Thus p, $s(a)$, and $f(a_2)$ are all distinct. Again by Lemma 4, we know that $f(a_2) \in M$; let $a_3 = M(f(a_2))$. Promoting a_1 to $s(a)$ and a_2 to $f(a_2)$, and demoting a_3 to $l(a_3)$, gives a matching more popular than M. See figure alongside. Either way, we contradict the popularity of M.

Since a was chosen arbitrarily, the result holds for all applicants. ■

We now obtain a nice characterization of popular matchings, that will be used in the decision algorithm.

Theorem 7 *A matching M is popular if and only if it satisfies the following conditions:*

1. *every f-post is matched in M, and*
2. *for every $a \in A$, $M(a) \in \{f(a), s(a)\}$.*

Proof: $\Rightarrow$: This follows directly from Lemma 4 and Lemma 6.

$\Leftarrow$: Let M be a matching satisfying the given conditions, and assume to the contrary that M is not a popular matching. So there is a matching M' such that $M' >_{\text{pop}} M$. Let A_1 be the set of applicants who prefer M' to M, let A_2 the set of applicants who prefer M over M', and let A_3 be the rest. By definition of $>_{\text{pop}}$, $|A_1| > |A_2|$.

We will construct an injective map from A_1 to A_2, hence concluding that $|A_1| \leq |A_2|$ and obtaining a contradiction. Pick any $a \in A_1$. Clearly, $M(a) \neq f(a)$, because if it were, then a would not prefer any matching over M. So by the given conditions, $M(a) = s(a)$. Since a prefers M' over M, clearly $p = M'(a)$ is higher than $s(a)$ in $\mathsf{List}(a)$. By definition of $s(a)$, p is some applicant's first-choice and so an f-post. So by the given conditions, $p \in M$. Let $M(p) = a'$. By the given conditions, either $p = f(a')$ or $p = s(a')$. But p is an f-post, and so it cannot be $s(a'')$ for any a''. So $p = f(a')$. We map a to a'. Clearly, a' prefers M, where she gets her first choice, to M', where she gets something less preferred, and so $a' \in A_2$.

To see why this map is injective, note that there is an $M' - M$ alternating path of length 2 from a to a': $a \rightarrow_{M'} p \rightarrow_M a'$. If some other a'' were also mapped to a', there would be a similar alternating path $a'' \rightarrow_{M'} p \rightarrow_M a'$, implying that M' matches p to both a and a'', a contradiction. ■

Decision Algorithms

This characterization is crucial in obtaining an efficient algorithm. Using this, we can throw away from G all edges except those connecting an applicant a to $f(a)$ or $s(a)$, getting a very sparse graph. This gives an extremely simple algorithm for the decision question:

```
Input: An instance $G' = (A \cup P', E)$, $\{L(a)\}_{a \in A}$ of popular matchings.
Add last-resort posts to get instance $G$.
Find $f(a)$ for all $a \in A$.
Find $s(a)$ for all $a \in A$.
Construct reduced instance $H = (A \cup P_H, E_H)$ where $E_H = \{(a, f(a)), (a, s(a)) \mid a \in A\}$.
if H has an $A$-perfect matching then
    Return Yes
else
    Return No
end if
```

The correctness of this algorithm follows from Theorem 7: If H has an A-perfect matching M, then it has an A-perfect matching M' where every f-post is matched: simply pick, for each un-matched post p, any $a \in f(p)$, and promote a to p.

Steps 2-5 require time $O(n+t+m)$ where m is the number of edges in G' (the total size of all the lists). The time requirement is dominated by step 6: checking whether H has an A-perfect matching. Using the best bipartite algorithm for maximum matching, we get a time bound of $O(\sqrt{(n+|P_H|)}|E_H|)$. Since there are $2n$ edges in H, this is $O(n\sqrt{n})$.

This time can be improved by a simple trick. In H, perform the following as long as possible:

1. If there is a degree-0 post, delete it.

2. If there is a degree-1 post p adjacent to a, then add (a,p) to the matching and delete both a and p.

Note: as a result of step 2 above, another degree-2 post could now become degree-1, or another degree-1 post could now become degree-0. So we ahve to repeat as long as possible. However, throughout this, each applicant a continues to have degree exactly 2; we never remove a neighbouring post p unless we match up a and p and remove a as well. Thus, when this ends, let the resulting graph be $H' = (A' \cup P'_H, E'_H)$. It is easy to see that H has an A-perfect matching if and only H' has an A'-perfect matching. In H', each post p has degree at least 2, while each applicant a has degree exactly 2. By Hall's Theorem, if $|A'| > |P'_H|$, then H' has no A'-perfect matching. Otherwise, $|P'_H| \leq |A'| = \frac{1}{2}\sum_p \deg(p) \geq |P'_H|$, and so $|P'_H| = |A'|$. So every vertex has degree exactly two, and the graph decomposes into disjoint even cycles; hence it has a perfect matching.

To summarize, H' has an A'-perfect matching if and only if $|A'| \leq |P'_H|$.

Obtaining H' from H requires time $O(n+|P_H|+|E_H|) \in O(n)$. Thus the overall time for the decision algorithm is $O(n+m)$.

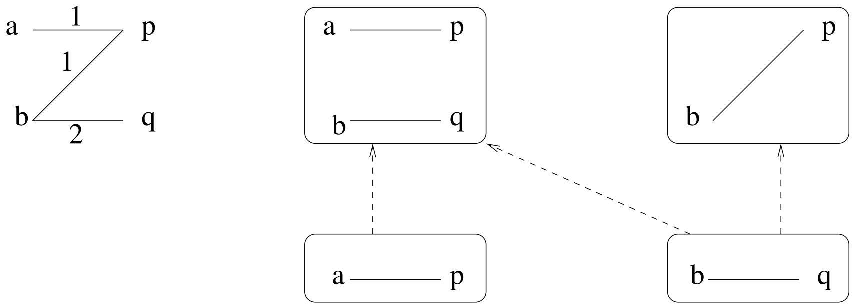
Search Algorithm

To find a popular matching, we can modify the decision algorithms above. Construct H and find a maximum matching M in H. If M is A-perfect, then let M' be the matching obtained by doing necessary promotions to match all f-posts. Now, delete from M' all last-resort matching edges; the resulting matching is the popular algorithm.

Time requirement: M can be obtained in $O(n\sqrt{n})$ time by an implementation of Edmonds's algorithm. Using the cleverer trick above, we can prune H to H'. If $|P'_H| = |A'|$, then finding the perfect matching in H' is trivial in time $O(n)$ because H' has disjoint even cycles. So the overall time requirement, even to find a popular matching if one exists, is just $O(n+m)$.

Maximum size popular matchings

The next question we want to address is to find a popular matching of maximum

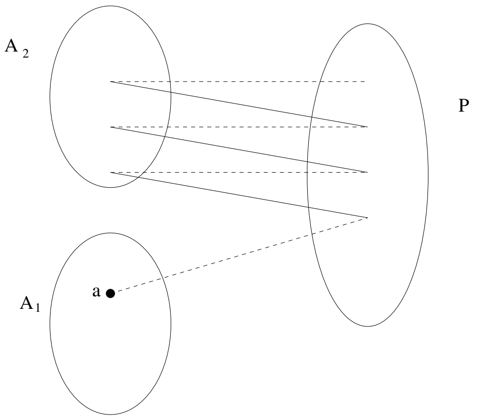


size. There can be popular matchings of different sizes; in the figure alongside, there are four matchings, and the dashed lines indicate $>_{\text{pop}}$. The matchings $\{(a,p),(b,q)\}$ and $\{(b,p)\}$ are both popular. And the algorithms we have described above may well find the smaller one. So we need a new trick. Here is the strategy.
Let A_1 be the set of applicants for whome $s(a)$ is the last-resort post. These applicants will either get their first choice or will be left unmatched. Let A_2 be the rest.

$$A_1 = \{a \in A \mid s(a) = l(a)\}; \quad A_2 = A \setminus A_1.$$

We want a popular matching in G', that is, an A-perfect matching in G or in H that minimizes the number of a in A_1 matched to $s(a)$. We do this as follows.

```
 1: Input: An instance $G' = (A \cup P', E)$, $\{L(a)\}_{a \in A}$ of popular matchings.
 2: Add last-resort posts to get instance $G$.
 3: Find $f(a)$ for all $a \in A$.
 4: Find $s(a)$ for all $a \in A$.
 5: Construct reduced instance $H = (A \cup P_H, E_H)$ where $E_H = \{(a, f(a)), (a, s(a)) \mid a \in A\}$.
 6: **if** H has no matching saturating $A_2$ **then**
 7:    Return No
 8: **else**
 9:    Find such a matching $M$.
10:    Delete all $(a, l(a))$ edges from $H$. (This leaves $M$ untouched.)
11:    **while** $M$ is not maximum **do**
12:       Augment $M$; this will match one more $A_1$ applicant.
13:    **end while**
14:    Match all $f$-posts by appropriate promotions.
15: **end if**
16: Return $M$.
```



To implement this algorithm efficiently, let us consider the most costly step: augment (step 12). The graph has a special structure: from an $a \in A_1$, there is a unique alternating path Q_a since nodes in A_2 have degree exactly 2. This alternating path is augmenting if and only if it ends in P. The crucial insight we uses is the following claim; proof is left as an easy exercise.

Claim 8 *No future augmenting path will contain an edge from Q_a (whether or not Q_a is augmenting).*

Hence it suffices to consider the vertices of A_1 in any fixed order; replace steps 11-13 above by the steps below.

```
for $a \in A_1$ do
  if $Q_a$ is augmenting then
    Augment $M$ along $Q_a$.
  end if
end for
```

Allowing ties

So far, we have only considered the case when there are no ties. If applicants can have ties in their lists, then more work is required, which we will not discuss here. We can still obtain polynimial-time ($O(m\sqrt{n})$ time) algorithms, using the Gallai-Edmonds decomposition.

Optimal popular matchings

Finally, we can also consider the question of finding a popular matching that is optimal by some criterion other than size. Possible criteria are: maximise the number of rank-1 edges, or, minimise the number of low-ranked edges, or, produce a most fair matching etc etc. Various such versions are known to have polynomial time algorithms.

References

[AIKM07] David J. Abraham, Robert W. Irving, Telikepalli Kavitha, and Kurt Mehlhorn. Popular matchings. *SIAM Journal on Computing*, 37(4):1030–1045, 2007. preliminary version in SODA 2005.

[KN09] Telikepalli Kavitha and Meghana Nasre. Optimal popular matchings. *Discrete Applied Mathematics*, 157(14):3181 – 3186, 2009.