APPLICATIONS OF POPULAR MATCHINGS ON CAMPUS

Yutaro Ikeda, Ayumi Igarashi and Maiko Shigeno
Graduate School of Systems and Information Engineering,
University of Tsukuba,
Tsukuba, Ibaraki, 305-8573, Japan
ikeda90(at)sk.tsukuba.ac.jp

Abstract

Min-cost matchings have been widely used to solve resource assignment models. To consider the popularity of assignments, Abraham et al. proposed “Popular matching.” This paper investigates the availability of popular matchings in some real-world problems raised in campus. In real-world settings, a matching needs to be obtained even if popular matchings do not exist. Hence, we show an algorithm which finds a nearly popular matching. By comparing the results obtained by this algorithm and by a min-cost matching algorithm, we discuss some conditions such that a popular matching provides a better result than a min-cost matching. Keywords: popular matching, resource assignment, real-world problem.

1. INTRODUCTION

There arise many matching problems in several areas of scheduling. Indeed, some resource assignments, which assign available resources to task activities efficiently, have basically matching structure in bipartite graphs. Many resource assignments in real-world have some side-constraints which complicate the problems. In this paper, however, we deal with a simple matching problem on bipartite graphs, which is a basic frame for some scheduling problems.

We consider the problem that assigns applicants to posts. The problem has one-sided preference lists. That is to say, every applicants has an ambition to be assigned to some posts although posts do not have any wish. When each applicant represents his/her ambition by a numerical score, it is an effective and widely used method to solve an assignment by the minimum-cost matching algorithm on a bipartite graph. However, in many cases, applicant’s ambition is submitted by ranking such as its first, second, and third choices. For such one-sided preference lists, researchers discussed some optimality criteria, Pareto optimality (Abraham et al. 2004), rank-maximality (Irving et al. 2006), fairness (Kavitha and Nasre 2009), and so on.

For the problem setup considered in this paper, an alternative way to assign is a “popular matching” which have been introduced by Gardenfors (Gardenfors 1975) in the context of the stable marriage problem. A matching $M$ is referred to as popular, if no other matching beats it by majority vote, where each applicant votes for the matching in which he/she is assigned preferred post or abstains if both assigned posts are indifferent. The popular matching problem, considered by Abraham et al. in (Abraham et al. 2007), is to determine whether a given instance admits a popular matching, and to find such a matching if one exists. This popular matching is one of reasonable definitions to persuade the applicants not assigned to the desired posts.

In this paper, we apply a popular matching problem to real-world problems raised in our campus. We consider the following three assignment problems: how to allocate classrooms to each applicant, how to assign students to each advisory professor, and how to assign students to each laboratory. By comparing the results obtained by popular matchings and by min-cost matchings, we verify the availability of popular matchings.

2. POPULAR MATCHING

Let $G = (A \cup P; E)$ be a bipartite graph. We call the nodes in $A$ applicants and the nodes in $P$ posts. Each applicant has his/her own partial ranking of the posts, i.e., a preference list of the posts. According to the collection of preference lists of all applicants, the edge set $E$ is partitioned into $k$ disjoint subsets $E = E_1 \cup E_2 \cup \ldots \cup E_k$. If $(a, p) \in E_i$ and $(a, p') \in E_j$ for $i < j$, then the applicant $a$ prefers $p$ to $p'$. An element in $E_i$ ($1 \leq i \leq k$) is called an edge of rank $i$. A matching $M$ is a set of edges such that no two edges share an endpoint. The post to which an applicant $a$ is matched in $M$ is denoted by $M(a)$. We say that an applicant $a$ prefers matching $M'$ to $M$ if (i) $a$ is matched in $M'$ and unmatched in $M$, or (ii) $a$ is matched in both $M'$ and $M$, and $a$ prefers $M'(a)$ to $M(a)$. We say $M'$ is more popular than $M$ if the number of applicants that prefer $M'$ to $M$ exceeds the number of applicants that prefer $M$ to $M'$. A matching $M$ is popular if there is no matching that is more popular than $M$.

The popular matching problem is to determine whether a given instance admits a popular matching, and to find such a matching if one exists. In (Mahdian 2006; Abraham et al. 2007), popular matchings do exist with good probability when preference lists are randomly constructed. For instances with no popular matching, McCutchen (McCutchen 2008) generalized popularity criterion and introduced least-
unpopular matchings. Unfortunately, it is NP-hard to find a least-unpopular matching.

Even if any popular matching does not exist, real-world problems require that we find an appropriate nearly popular matching, especially an applicant-complete matching in which every applicant is matched to a post. Note that every popular matching is applicant-complete, by introducing a unique last resort post for each applicant. In order to find an applicant-complete nearly popular matching, we employ a multi-phase algorithm. The algorithm is based on the property indicated in Theorem 1.) For this reason, some notions are defined. According to our applications described in the next section, we assume that the given preference lists are strict ordered, i.e., no applicant is indifferent between any two posts on his/her preference list. A post p is called f-post if it is a first-ranked post on some applicant’s preference list, i.e., if there exists an edge of rank one incident to p. Let s(a) be the non-f-post on a’s preference list. A reduced graph G’ is the induced subgraph of G by E1 ∪ {⟨a, s(a)⟩ ∈ E | a ∈ A}, where G is a graph by adding the last resort posts to G.

Theorem 1 (Abraham et al. 2007) A matching M is popular if and only if (i) every f-post is matched in M, and (ii) M is an applicant-complete matching on the reduced graph G’.

Our multi-phase algorithm is as follows.

initial phase: Find a maximum matching M under the condition (i) in Theorem 1 in the reduced graph G’ (Such a matching is referred to as maximum f-post-complete.)

If M is applicant-complete, then stop. Otherwise, go to the recurrence phase.

recurrence phase: Let M(A) be the set of applicants matched by M. Delete M(A) and {M(a) | a ∈ M(A)} from G and reconstruct the reduced graph G’. Find a maximum f-post-complete matching M’ in G’, and replace M by M ∪ M’. If M is applicant-complete, then stop. Otherwise, repeat the recurrence phase.

Each phase can find an f-post-complete matching by maximum matching algorithm (see (Kavitha and Nasre 2009)). The algorithm repeats recurrence phases at most \(\lceil |A|/2 \rceil \) times, since in each iterations, at least one applicant is assigned to f-post and at least one other applicant a is assigned to s(a) if \(\{a, s(a)\} \in E \mid a \in A\) ≠ Ø. This multi-phase algorithm finds an applicant-complete matching. Note that we can obtain a popular matching if the algorithm stops at the initial phase. If an instance does not contain any popular matching, the algorithm return a matching we expect to be nearly popular. However, it is not estimated theoretically how the obtained matching is wrong in popularity. So, in the next section, we evaluate the obtained matching by computational experiments.

### Table 1 An example of preference list

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th></th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>p1</td>
<td>p2</td>
<td>p3</td>
</tr>
<tr>
<td>a1</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>a2</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>a3</td>
<td>1</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>a4</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

3. APPLICATIONS

We verify the availability of popular matchings in real-world problems. Three assignment problems which arise in our campus are treated. We investigate the existence of popular matchings for the preference lists obtained from these problems, and compare popularity between a matching obtained by our multi-phase algorithm and a min-cost matching where the cost of each edge is given by its rank number. The popularity is evaluated by the number of applicants who prefer the one matching than the other.

3.1 Implementation

In our implementation, we used Hungarian algorithm to find min-cost matchings (see, for example, (Lawler 1976)). Matchings obtained by our multi-phase algorithm and Hungarian algorithm, respectively, depend on the stored order of elements of A and P in data structure. In order to avoid this influence, each instance is solved ten times by rearranging elements in A and P, respectively, at random, and return a matching chosen by certain criteria. For instance, for a preference list shown in Table 1 (a), which does not contain any popular matching, our multi-phase algorithm may return both of Ml = \{⟨a1, p1⟩, ⟨a2, p2⟩, ⟨a3, p3⟩, ⟨a4, p4⟩\} and M2 = \{⟨a1, p1⟩, ⟨a2, p2⟩, ⟨a3, p3⟩, ⟨a4, p4⟩\}. Although there is no difference in popularity between Ml and M2, it is obvious that M1 is more desirable. Corresponding to this example, we choose a min-cost matching among near popular matchings obtained by the multi-phase algorithm. We denote this obtained matching by MP.

On the other hand, for a preference list shown in Table 1 (b), both of Ml = \{⟨a1, p1⟩, ⟨a2, p2⟩, ⟨a3, p3⟩\} and M2 = \{⟨a1, p2⟩, ⟨a2, p3⟩, ⟨a3, p1⟩\} are min-cost matchings, although M2 is more popular than Ml. Thus, we need to compare popularity among min-cost matchings in order to find a desired matching. We call a matching M most popular among a collection of matchings, if the number of times that M becomes more popular when comparing it with other matching is largest. By comparing all pairs of candidate ten min-cost matchings, we choose the most popular matching. We denote this obtained matching by M_{MC}.

Although a matching represents a one-to-one assignment, our applications require one-to-some assignments. We transform such one-to-some assignment problems to matching problems by copying multiple nodes.
3.2 Assignment of classrooms

In our campus, classrooms are assigned to some clubs for activities out side of class. Under existing circumstances, some clubs are often dissatisfied, because all bookings of classrooms are on a first-come-first-served basis. Usually, classroom resources can be assigned more efficiently by coordination.

We employ popular matchings to this classroom assignment. In this case, each club corresponds to an applicant and each classroom to a post in our model. We made inquiry about the preferences of classrooms for 20 clubs, and compute matchings $M_P$ and $M_{MC}$. In this case, the obtained matching $M_P$ is almost popular, since there are only two clubs which is not assigned any classroom at the initial phase. The popularity between $M_P$ and $M_{MC}$ does not have any difference. In both matchings, 18 clubs are assigned the same classrooms. One club prefers $M_P$ to $M_{MC}$ and one club prefers $M_{MC}$ to $M_P$.

In the classroom assignment, each club has classrooms regularly using, which implies that a few clubs scramble for one classroom. Thus, we obtain good results by both of our multi-phase algorithm and a min-cost matching algorithm.

3.3 Assignment of advisers

In our department, every student is assigned to a professor to take advices on his/her graduation thesis. For this assignment, students submit their first, second and third choice professors. According to this preference list, a coordinator assigns each student to a professor such that each professor becomes adviser for about three students.

In our instance, there are 56 students and 22 professors. This experimental investigation changes upper bounds of the number of students whom one professor is assigned. The larger the upper bound, the more students are satisfied by the assignment. If the upper bound is 9, then all students can be assigned to their first choice professors. Table 2 compares the numbers of students who prefer $M_{MC}$ to $M_P$ ($M_{MC} > M_P$) and of students who prefer $M_P$ to $M_{MC}$ ($M_P > M_{MC}$) for each upper bound.

Table 2 Comparing popularity for the adviser assignment

<table>
<thead>
<tr>
<th>upper bounds</th>
<th>$M_{MC} &gt; M_P$</th>
<th>$M_P &gt; M_{MC}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

It is surprising that $M_{MC}$ is more popular than $M_P$ for every upper bound. Indeed, preference lists of students tend to be hard to admit popular matchings, because the similar choices arise from several students in the same research fields. Fig. 1 illustrates the dispersion of preference lists.

This graph indicates the number of students who choose each professor. The short length of preference list, each of

![Fig. 1 Bias of preference lists for the adviser assignment](image)

which at most three professors are ranked, also causes to admit no popular matchings.

3.4 Laboratory assignment

In any campus, the laboratory assignment is important for students. Actually, each student behaves strategically to get an desired assignment. Separately from the real assignment, we made inquiry about preference lists for students and make a laboratory assignment. The preference list consists of student's from first to fifth choices. We examine the influence of upper bounds of the number of students assigned to a laboratory, such as the adviser assignment. In our data, there exist 54 students and 23 laboratories. If the upper bound is 10, then all students can belong to their first choice laboratories. Table 3 shows the results.

Table 3 Comparing popularity for the laboratory assignment

<table>
<thead>
<tr>
<th>upper bounds</th>
<th>$M_{MC} &gt; M_P$</th>
<th>$M_P &gt; M_{MC}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

In this case, it depends on the upper bound whether $M_P$ is more popular than $M_{MC}$. As in the case of the adviser assignment, a few laboratories are ranked in many preference lists, which causes to admit no popular matching (See Fig. 2) . However, the number of choices ranked in a preference list is larger than one for the adviser assignment. This fact may derive the tendency that $M_P$ is more popular than $M_{MC}$.
3.5 Additional experiments

Although Mahdian (Mahdian 2006) and Abraham et al. (Abraham et al. 2007) said that popular matchings do exist with good probability when preference lists are randomly constructed, our three applications do not admit popular matchings. Moreover, our multi-phase algorithm based on popular matchings sometimes gives a result not more popular than one by a min-cost matching algorithm. One of the reasons seems to be that posts ranked in preference lists are biased. To examine the influence of such biased preference lists, we create artificially data with 50 applicants and 50 posts. Each applicant ranks all the posts by strict order. Two types preference lists are generated as follows. For each applicant, generate number $p_i$ for post $i$ ($i = 1, \ldots, 50$) by uniform random numbers.

(1) uniform preference lists: The preference list of the applicant is given by the decreasing order of the value of $p_i$.

(2) biased preference lists: The preference list of the applicant is given by the decreasing order of the value of $p_i \times i$.

Figs. 3 and 4 show biases of the generated preference lists. These graphs indicate the number of applicants ranking for each post in the uniform preference list and the biased preference list, respectively.

As described in (Mahdian 2006; Abraham et al. 2007), the uniform preference lists admit popular matchings many times. Indeed, when the upper bound is more than 3, there exists a popular matching always. Thus, obtained matching $M_F$ is preferred to $M_{MC}$. On the other hand, the biased preference lists make hard to admit popular matchings. We can obtain popular matchings when the upper bound is more than 25. Moreover, when the upper bound is less than 10, $M_{MC}$ tends to be preferred to $M_F$.

In additional simulations which are not reported detail here, we observe the tendency that $M_F$ are to be liked than $M_{MC}$ for preference lists with strong bias.

4. CONCLUSION

In this paper, we applied popular matchings to three real world assignment problems with one-side preference lists. We compared the results obtained by popular matchings and by min-cost matchings. Real-world problems require some matching nearly popular when the instance has no popular matching. Thus, we showed a multi-phase algorithm which
gives a suitable matching even if any popular matching does not exist.

In some real-world problems, an assignment obtained by our multi-phase algorithm is not more popular than one by min-cost matching algorithm. This is because a few posts are ranked in high priority of many applicants.

As we observed at the experiments for artificial data, we conjecture that a matching obtained by our algorithm is better than min-cost matchings when applicants have uniform preference lists. Proving this hypotheses is a future work.

References


