A METHOD FOR THE LABORATORY ASSIGNMENT PROBLEM WITH LOWER AND UPPER BOUNDS FOR CAPACITY

JiHong Park and Hiroshige Dan
Kansai University
3-3-35, Yamate-cho, Suita-shi, Osaka 564-8680, Japan
dan@kansai-u.ac.jp

Abstract

In this paper, we consider a problem to find an assignment of students to laboratories for their graduation study in universities. The previous research by Kataoka and Ibaraki (2008) proposed a method for this problem. In their method, a capacity of each laboratory is determined a priori, and students are assigned to laboratories as much as a determined capacity with a certain desirable stability. In this paper, based on their research, we propose a method for the laboratory assignment problem with lower and upper bounds of capacity. In our method, we do not have to determine a capacity of each laboratory in advance. Moreover, our method can compute an assignment which has the same stability as the previous method.

Keywords: Laboratory assignment problem, stable marriage problem, combinatorial optimization, algorithm.

1. Introduction

In Japanese universities, students are generally assigned to laboratories (or seminars) to do their graduation study. In this paper, we call a problem to determine this assignment the laboratory assignment problem (LAP).

One of the most important points of LAP is to find an assignment to satisfy students’ and laboratories’ request as much as possible. This situation is similar to the stable marriage problem (SMP) proposed by Gale and Shapley (1962). SMP is a problem to determine a one-to-one mapping between the same number of men and women, and this mapping needs stability in a certain criterion. This setting of SMP is very simple, and many problems which generalize SMP have proposed. For example, the college admissions problem (Roth, 1985 and Balinski and Sönmez, 1999) is to determine successful applicants. Also, a problem to find an assignment of medical school graduates to hospitals as interns has been researched by Roth (1984, 1991). Moreover, LAP has been researched by Kataoka and Ibaraki (2008). They have found a similar structure between SMP and LAP and proposed an assignment algorithm to find a stable matching in a certain criterion. This algorithm is based on the algorithm by Gale and Shapley (1962).

One of the most different points between SMP and LAP is a mapping property. SMP is to find a one-to-one mapping, on the other hand, LAP is a many-to-one mapping. Generally speaking, the number of students is greater than that of laboratories, then more than one student would be assigned to each laboratory. Therefore we have to determine a capacity of each laboratory as fair as possible. This problem is similar to an apportionment problem (Ibaraki and Katoh, 1988), that is, a problem to find a fair allocation of legislative seats. Actually, Kataoka and Ibaraki (2008) have used a view of an apportionment problem to determine a capacity of each laboratory. However, it is very difficult to define the “fairness” in an apportionment problem, also, in LAP.

To overcome this difficulty, in this paper, we propose a method for finding a stable matching of LAP with lower and upper bounds of capacity. Our proposed method is based on the method by Kataoka and Ibaraki (2008) and makes it possible to avoid fixing a capacity of each laboratory and find a matching with a certain stability.

2. Preliminaries

In this section, we give some definitions and preliminaries which will be necessary in this paper.

2.1 Stable Marriage Problem

SMP has been proposed by Gale and Shapley (1962). This problem is a basis of matching between two groups. First, we define SMP, and explain a method for this problem by Gale and Shapley.

2.1.1 Definition of Stable Marriage Problem

We suppose that there exist two sets, $H$ and $D$. $H$ is a set of $n$ men and $D$ is a set of $n$ women. If a man $h \in H$ prefers $d \in D$ to $d' \in D$, then this preference is denoted by $d >_h d'$. In the same manner, we can denote woman $d$'s preference by $>_d$. Now we assume that there are no ties in a preference list of each person.

We call $(h,d)$ ($h \in H, d \in D$) a pair. In the case that a set $M$ has $n$ pairs and these $n$ pairs are one-to-one correspondences between $n$ men and $n$ women, we call $M$ a matching. If $(h,d) \in M$, a man $h$ is a partner of a woman $d$ in $M$, and vice versa. These relations are denoted by $h = p_M(d)$ and $d = p_M(h)$. In addition, if a set $M'$ satisfies $|M'| \leq n$ and
all men and women do not belong to more than one pair in 
\( M' \), then we call \( M' \) a partial matching. Moreover, if a pair 
\((h, d) \notin M' \) which satisfies \( d >_h p_M(h) \) and \( h >_d p_M(d) \)
exists, then we call \((h, d) \) an unstable pair for \( M \). A stable 
matching is a matching which does not have an unstable pair, and an unstable matching is a matching which has an unstable pair.

SMP is to find a stable matching for given \( n \) men and \( n \) women.

2.1.2 Gale-Shapley algorithm

Gale and Shapley (1962) proposed the basic algorithm to 
solve SMP. It is usually called the Gale-Shapley (GS) algo-

The GS algorithm

Input A set of \( n \) men \( H \), a set of \( n \) women \( D \) and preference 
lists of every person.

Output A (stable) matching \( M \)

Step 1 Every man and woman is set free.

Step 2 Repeat the following while there exists a man who is 
free:

Step 2-1 A man \( h \) proposes to the first woman \( d \) on \( h \)'s 
preference list to whom \( h \) has not proposed yet.

Step 2-2 If \( d \) is free, then \( h \) and \( d \) get engaged to each other. Otherwise, if \( d \) has been engaged to a man
\( h' \) and \( h' >_d h \) holds, then \( d \) rejects \( h' \)'s proposal 
\((h \) remains free). On the other hand, if \( h' >_d h \)
holds, then \( h' \) and \( d \) break off their engagement 
and \( h \) and \( d \) get engaged to each other \( (h' \) becomes 
free).

Step 3 Output a current set of engaged pairs as a matching 
\( M \).

In the GS algorithm, a man makes a proposal to a woman.
It is a man-oriented GS (MGS) algorithm. Moreover, if we 
exchange a role of man and woman, then we obtain a 
woman-oriented GS (WGS) algorithm. It has been well
known that MGS and WGS algorithms usually give different 
m matchings, even if preference lists of every man and woman 
are the same (Gale and Shapley, 1962).

The following theorems about the GS algorithm hold:

**Theorem 1** (Gale and Shapley, 1962) The GS algorithm 
always terminates and generates a stable matching. Accord-
ingly, there exists a stable matching for any given instance 
of the stable marriage problem.

**Theorem 2** (Gale and Shapley, 1962) MGS algorithm 
yields the same stable matching for any order of propos-
als. Moreover, in this stable matching, each man has the 
best partner in any stable matching, that is, there is no sta-
ble matching in which a man has a better partner than a 
matching which MGS yields.

**Theorem 3** (McVitie and Wilson, 1971) A stable match-
ing which is obtained by MGS algorithm is worst for ev-
every woman, that is, there is no stable matching in which a
woman has a worse partner than a matching which MGS 
yields.

Of course, under WGS algorithm, theorems 2 and 3 hold 
with exchanging a role of man and woman.

2.2 Stability of Matching Problems with Ties of Prefer-
eence

SMP in Section 2.1.1 does not allow ties in a preference 
list of each person, however, we can also consider SMP 
which allows ties. In the following, \( d =_h d' \) means that 
\( h \in H \) prefer \( d \in D \) and \( d' \in D \) equally. For SMP with ties, 
three types of stability have been proposed by Gusfield 
and Irving (1989) and Irving (1994). Table 1 shows these 
three types of stability. Each stability type requires that 
there is no unstable pair under its definition. From Table 1, 
we can understand that if a matching is super-stable, then it 
is strongly stable. Also, if a matching is strongly stable, then 
it is weakly stable. It has been known that there may not

<table>
<thead>
<tr>
<th>Table 1 Stability with Ties of Preference</th>
<th>( d &gt;_h p_M(h) )</th>
<th>( d =_h p_M(h) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1 )</td>
<td>( 1 ), ( 2 ), ( 3 )</td>
<td>( 1 ), ( 2 )</td>
</tr>
<tr>
<td>( 2 )</td>
<td>( 1 ), ( 2 )</td>
<td>( 1 ), ( 2 )</td>
</tr>
<tr>
<td>( 3 )</td>
<td>( 1 ), ( 2 )</td>
<td>( 1 ), ( 2 )</td>
</tr>
</tbody>
</table>

\*1: \((h, d)\) is unstable for a super-stable matching
\*2: \((h, d)\) is unstable for a strongly stable matching
\*3: \((h, d)\) is unstable for a weakly stable matching

be a super- or strongly stable matching for SMP with ties 
(Gusfield and Irving, 1989 and Irving, 1994).

2.3 Laboratory Assignment Problem

In this section, we explain LAP and a method for solving 
it (Kataoka and Ibaraki, 2008) \(^1\).

2.3.1 Definition of LAP

Now we define LAP by using the concept of SMP in Sec-
ction 2.1.1.

First, we summarize symbols which are used in the fol-

- \( S \): set of students \(|S| = n \)
- \( s_i \in S \) \((i = 1, 2, \ldots, n)\): students
  (We also use \( s \) for simplicity)
- \( L \): set of laboratories \(|L| = m \)
- \( l_i \in L \) \((i = 1, 2, \ldots, m)\): laboratories
  (We also use \( l \) for simplicity)
- \( c_i \) \((i = 1, 2, \ldots, m)\): capacities of \( l_i \)

To define LAP, we replace sets of men \( H \) and women \( D \) in 
SMP with sets of students \( S \) and laboratories \( L \), respectively.
We assume that \(|H| = |D| \) in SMP, however, the condition

\(^1\)We have slightly modified the method for LAP which has been pro-
posed by Kataoka and Ibaraki (2008). However, their expression and ours 
are the same essentially.
\(|S| = |L|\) does not hold necessarily in LAP. The capacity of laboratories satisfies the condition \(\sum_{i=1}^{m} c_i = n\), and each student will be assigned to one laboratory.

In LAP, ties in the preference lists of students and laboratories are allowed. Then, we modify the definition of matching and 'free/engaged' concept as below:

- A matching \(\mathcal{M}\) is a set of \(n\) student-laboratory pairs. In any matching \(\mathcal{M}\), each student \(s_j (i = 1, 2, \ldots, n)\) is emerged one time, and each laboratory \(l_i (i = 1, 2, \ldots, m)\) is emerged \(c_i\) times.
- A student is either 'unassigned' or 'temporary assigned'. A temporary assignment \(\mathcal{M}'\) is a subset of any matching. When \(s\)'s partner is \(l\) in \(\mathcal{M}'\), it is denoted by \(l = p_{\mathcal{M}}(s)\). Similarly, \(s \in p_{\mathcal{M}'}(l)\) means that one of \(l\)'s partner is \(s\) in \(\mathcal{M}'\). Note that \(p_{\mathcal{M}'}(l)\) is a set because plural students could be assigned in one laboratory.

Kataoka and Ibaraki (2008) classified unstable pairs of LAP as Table 2. Based on this classification, Kataoka and Ibaraki (2008) proposed five types of stability in LAP as follows:

- super-stable: there is no tie-unstable pair
- strongly stable: there is no laboratory-unstable and student-unstable pair
- student-stable: there is no student-unstable pair
- laboratory-stable: there is no laboratory-unstable pair
- weakly stable: there is no strictly unstable pair

Note that super-, strongly and weakly stability for LAP are the same as we have defined in Section 2.2. Moreover, from the definition above, a super-stable matching is strongly-stable, a strongly-stable matching is student-stable and laboratory-stable, and a student-stable or laboratory-stable matching is weakly stable.

LAP is to find a stable matching for preference lists of students and laboratories. A stable matching has to satisfy one of the stabilities above.

### 2.3.2 Algorithm for LAP

In this section, we explain the algorithm for LAP which has been proposed by Kataoka and Ibaraki (2008).

As seen previously, LAP allows that preference lists of students and laboratories have ties, then each student's and laboratory's preference becomes a partial order. However, since the algorithm needs a total order, we have to convert a partial order into a total order. Any total order which does not contradict an original partial order is acceptable. \(\mathcal{T}_j\) and \(\mathcal{T}_l\) denote a student \(s_j\)'s and laboratory \(l_i\)'s total order, respectively.

Moreover, Kataoka and Ibaraki (2008) have assumed that the whole of total orders of students and laboratories is not unveiled, and the algorithm would unveil the part of lists which is needed in it. In this meaning, \(\mathcal{T}_j\) is an implicit total order of \(s_j\), and \(\mathcal{T}_l\) is an implicit total order of \(l_i\). Moreover, the veiled part of total orders is considered as ties by concerned students and laboratories. At this point, a part of students' and laboratories' partial order which are unveiled for every student and laboratory is called an explicit partial order.

The following is the algorithm for LAP which has been proposed by Kataoka and Ibaraki (2008):

**Algorithm I**

**Input** A set of students \(S\), a set of laboratories \(L\), capacities of laboratories \(c_i (i = 1, 2, \ldots, m)\), implicit total orders of students \(\mathcal{T}_j (j = 1, 2, \ldots, n)\), and implicit total orders of each laboratory \(\mathcal{T}_l (i = 1, 2, \ldots, m)\).

**Output** A matching \(\mathcal{M}\)

**Step 1** Every student is set unassigned.

**Step 2** If there exist unassigned students, such students applies to the first laboratory which they have not applied yet. If there exist laboratories whose capacity is bigger than or equal to the sum of the number of temporarily assigned students and applying students, then applying students are temporarily assigned. Otherwise, a laboratory \(l_i\) chooses \(c_i\) students by \(\mathcal{T}_l\) and they are assigned temporarily to that laboratory. The rest of students are set unassigned.

**Step 3** If the number of temporarily assigned students of each laboratory \(i\) is equal to \(c_i (i = 1, 2, \ldots, m)\), then output a current assignment as a matching \(\mathcal{M}\). Otherwise, go to Step 2.

We can consider that Algorithm I is a natural extension of the GS algorithm for the case that we replace "man", "woman", "proposes" and "engaged" with "student", "laboratory", "apply" and "temporarily assigned", respectively.

Kataoka and Ibaraki (2008) have shown the following theorems:

**Theorem 4** (Kataoka and Ibaraki, 2008, Theorem 4.3) Algorithm I generates the unique matching which depends on implicit total orders of students and laboratories.

**Theorem 5** (Kataoka and Ibaraki, 2008, Theorem 4.4) Algorithm I generates a strongly stable matching which depends on explicit partial orders of students and laboratories.

One of important points about Algorithm I is to unveil only a part of laboratories' preference for students which is needed in it. In detail, rejected students in Step 2 would know a part of laboratory's preference. Kataoka and Ibaraki (2008) used this fact to show Theorem 5. Besides, when preference lists with ties are open completely, there exist...
instances which have no strongly stable matching (Gusfield and Irving, 1989, Irving, 1994).

3. Laboratory Assignment Problem with Lower and Upper Bounds of Capacity

3.1 Definition of LAP with Bounds

In this section, we define LAP with lower and upper bounds of capacity and propose a method for finding a stable matching.

In LAP proposed by Kataoka and Ibaraki (2008), each laboratory has a fixed capacity\(^2\). However, in our problem, it has lower and upper bounds of capacity. This is a natural extension of the original LAP.

In the following, the symbols defined in Section 2.3.1 are still valid, and we add the symbols below:

- \(c_i, \bar{c}_i\) (\(i = 1, 2, \ldots, m\)): the upper and lower bounds of capacity of laboratories \(l_i\) (\(i = 1, 2, \ldots, m\)) which satisfy

\[
\sum_{i=1}^{m} c_i \leq n \leq \sum_{i=1}^{m} \bar{c}_i
\]

- \(b_i\) (\(i = 1, 2, \ldots, m\)): the numbers of temporarily assigned students of laboratories \(l_i\) (\(i = 1, 2, \ldots, m\))
- \(L_1 := \{l_i \mid b_i < \bar{c}_i\}\)
- \(L_2 := \{l_i \mid b_i = \bar{c}_i\}\)
- \(L_3 := \{l_i \mid \bar{c}_i < b_i \leq c_i\}\)

3.2 Proposed Algorithm

In this section, we propose an algorithm for LAP with bounds. Also, we assume that the whole of preference lists is not unveiled and this algorithm would show a part of lists which is needed in it. This is the same way of Algorithm I.

Algorithm II

**Input** A set of students \(S\), a set of laboratories \(L\), lower and upper bounds of capacity of each laboratory \(c_i, \bar{c}_i\) (\(i = 1, 2, \ldots, m\)), implicit total orders of student \(T_{fj}\) (\(j = 1, 2, \ldots, n\)), and implicit total orders of each laboratory \(T_{lj}\) (\(i = 1, 2, \ldots, m\)).

**Output** A matching \(\mathcal{M}\)

**Step 1**

- **Step 1-1** Every student is set unassigned.
- **Step 1-2** If there exist unassigned students, such students applies to the first laboratory which they have not applied yet. If there exist laboratories whose upper bounds for their capacity is bigger than or equal to the sum of the number of temporarily assigned students and applying students, then applying students are temporarily assigned. Otherwise, a laboratory \(l_i\) chooses \(c_i\) students by \(T_{lj}\) and they are assigned temporarily to that laboratory. The rest of students are set unassigned.

- **Step 1-3** If \(\sum_{i=1}^{m} b_i = n\) holds, then go to Step 2. Otherwise, go to Step 1-2.

**Step 2**

- **Step 2-1** If \(L_1 = \emptyset\) holds, then output a current assignment as a matching \(\mathcal{M}\) and terminate.
- **Step 2-2** Every laboratory \(l_i \in L_3\) picks up the student \(s_{\hat{i}}\) who is the last in \(T_{lj}\) from the temporary assigned students of \(l_i\). Moreover, choose one student \(s\) from \(s_{\hat{i}}\) to \(l_i \in L_3\) arbitrarily. \(s\) is set unassigned.
- **Step 2-3** \(s\) applies to the first laboratory \(l_{\hat{w}}\) which he/she has not applied yet. \(s\) is temporarily assigned to \(l_{\hat{w}}\).
- **Step 2-4** If \(l_w \in L_2 \cup L_3\) holds, then pick up the student \(s_{\hat{w}}\) who is the last in \(T_{lj}\) from the temporary assigned student of \(l_w\). Set \(s_{\hat{w}}\) unassigned and \(s := s_{\hat{w}}\), and go to Step 2-3. Otherwise, \(s\) is temporarily assigned to \(l_w\) and go to Step 2-1.

Note that Step 1 in Algorithm II is almost the same as Algorithm I in Section 2.3.2. Moreover, if \(L_1 \neq \emptyset\) holds in Step 2-1, then \(L_3 \neq \emptyset\) holds in Step 2-2 from (1).

3.3 Property of the Proposed Algorithm

The following lemma claims that Algorithm II generates a matching which satisfies lower and upper bounds of capacity.

**Lemma 1** Algorithm II always terminates, and \(c_i \leq b_i \leq \bar{c}_i\) (\(i = 1, 2, \ldots, m\)) hold at the end of it.

The following theorem shows that a matching generated by Algorithm II is strongly stable.

**Theorem 6** Algorithm II generates a strongly stable matching which depends on explicit partial orders of students and laboratories.

The proof of this theorem is similar to that of Theorem 5 by Kataoka and Ibaraki (2008).

Next, we consider the uniqueness of a solution obtained by Algorithm II.

\(b_i\) (\(i = 1, 2, \ldots, m\)) denote the numbers of temporarily assigned students at the end of Step 1 in Algorithm II. When we set that \(b_i\) in Algorithm II is equal to \(c_i\) in Algorithm I, Step 1 in Algorithm II would behave in the same way as Algorithm I. Then, from Theorem 4, we know that a temporary assignment at the end of Step 1 of Algorithm II is unique.

In Step 2-2 of Algorithm II, every laboratory \(l_i \in L_3\) picks up the last student \(s_{\hat{i}}\) in \(l_i\)'s list, and one student \(s\) is chosen from \(s_{\hat{i}}\) to \(l_i \in L_3\). We can choose \(s\) randomly, then a matching obtain by Algorithm II is not necessarily unique. However, if we prepare a particular list of all students a priori and choose \(s\) by it, then Step 2 of Algorithm II behave uniquely and we obtain a unique matching by Algorithm II.

Finally, we estimate the computational complexity of Algorithm II. First, we consider the computational complexity

\(^2\)In detail, Kataoka and Ibaraki (2008) proposed a method for determining a capacity of each laboratory.
of students' application for laboratories. The number of students is $n$ and the number of laboratories is $m$. Then, the number of application by students is $mn$ at most. Moreover, the determination of laboratory to apply for can be done by $O(1)$ running time. Consequently, the computational complexity of students' application is $O(mn)$.

Next we consider the computational complexity of laboratories' process. As seen above, students would apply for laboratories $mn$ times at most. And each laboratory has a sorted preference list and check it when a laboratory determines student(s) to refuse. Each laboratory compares $m$ students at most simultaneously, however, we assume that the preference of students is not biased too much and the number of students to be compared at the same time is $O(1)$. Then, the computational complexity to sort them and find the last student in a laboratory's list is $O(1)$. Hence, the computational complexity of laboratories' process is $O(mn)$.

Consequently, the computational complexity of Algorithm II is $O(mn)$ if the number of students to be compared at the same time is $O(1)$.

4. Numerical Experiments

In this section, we introduce some numerical experiments by our proposed algorithm, Algorithm II.

In these experiments, we set $n = 8$ and $m = 3$. Moreover, $T_s$ and $T_l$ are given a priori as Tables 3 and 4. In addition, we set $(\bar{c}_1, \bar{c}_1) = (2, 4), (\bar{c}_2, \bar{c}_2) = (2, 4)$ and $(\bar{c}_3, \bar{c}_3) = (3, 4)$. 

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Students' Preference $T_s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>student</td>
<td>$s_1$</td>
</tr>
<tr>
<td>$l_1$</td>
<td>$s_2$</td>
</tr>
<tr>
<td>$l_2$</td>
<td>$s_3$</td>
</tr>
<tr>
<td>$l_3$</td>
<td>$s_4$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4</th>
<th>Laboratories' Preference $T_l$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laboratory</td>
<td>$l_1$</td>
</tr>
<tr>
<td>$l_1$</td>
<td>$s_1$</td>
</tr>
<tr>
<td>$l_2$</td>
<td>$s_3$</td>
</tr>
<tr>
<td>$l_3$</td>
<td>$s_5$</td>
</tr>
</tbody>
</table>

Under this setting, we have conducted numerical experiments, then we have obtained two patterns, Tables 5 and 6. We now explain how to see Table 5. A row "Step 1" shows a matching obtained by Algorithm II (at the end of Step 2). Moreover, rows between "Step 1" and "Step 2" show the change of temporary assignment of each laboratory. For example, the next row of "Step 1" in Table 5 means that $s_5$ is chosen as $\hat{s}$ in Step 2-2 and $s_5$ applies for $l_1$ which is the first laboratory among unapplied ones. And the next row means that $s_2$ is chosen as $\hat{s}$ in Step 2-4 and $s_2$ applies for $l_3$. Table 6 can be interpreted in the same way.

Under this setting, $l_1$ is the only laboratory which does not satisfy the lower bound at the end of Step 1, and the number of shortage is 2. This is equal to the number of times to choose $\hat{s}$ in Step 2-2. By the way, when Algorithm II executes Step 2-2 first, there are two candidate for $\hat{s}$, $s_5$ and $s_2$. Tables 5 and 6 show the cases in which $s_5$ or $s_2$ is chosen as $\hat{s}$ first, respectively. We can see that both cases behave to increase the number of temporarily assigned students of $l_3$. We will show another results during the presentation.

5. Conclusion

In this research, we have proposed a method for LAP with lower and upper bounds on capacity. Kataoka and Ibaraki (2008) proposed a method for LAP with fixed capacity, and we have extended their results. Our proposed method can avoid fixing a capacity of each laboratory a priori, and find a strongly stable matching.

Finally, we mention future research topics. Our proposed algorithm regards the number of temporarily assigned students at the end of Step 2 as a capacity of each laboratory and guarantees the strong stability of a matching. However, there are some instances which has a better matching, in a sense, than a result of our proposed method. For example, Tables 5 and 6 are results for the same
instance, and the difference is just a \textit{s}_3's assigned laboratory. Table 5 is better for \textit{s}_3 than Table 6, and Tables 5 and 6 are equal to all students but \textit{s}_3. Moreover, Tables 5 and 6 satisfy lower and upper bounds of laboratories. Then, for a person who know lower and upper bounds, Table 6 is superior to Table 5. To overcome this difficulty, we should consider a new concept of stability and find an algorithm to obtain a stable matching under it.

References


