Stable Roommates

Michael St. Jules

November 16, 2013

Numberings of definitions and results are the same as in my project for consistency.

Definitions.
An **instance** of the stable roommates problem is a set of $X = \{1, 2, \ldots, n\}$ of individuals and for each $x \in X$, a preference list $L(x)$, that is a permutation of the remaining individuals in X.

x **prefers** y **to** z, if y appears before z in $L(x)$.

Preference lists (not necessarily the original ones) together are called a (preference) **table**, and a table T' is a **subtable** of a table T, if $x \in X$ is in $y \in X$'s list in T' implies the same in T and x comes before z on y's list in T' implies the same in T.
Tables are with respect to the instance, i.e. subtables of the original table.

A **matching** for an instance of the stable roommates problem is a bijection $\mathcal{M} : X \to X$ such that $\mathcal{M}(x) \neq x$ for all $x \in X$ and $\mathcal{M}(x) = y$ iff $\mathcal{M}(y) = x$.
$\mathcal{M} = \{(x, \mathcal{M}(x)) \in X \times X | x \in X\}$.
Note: n must be even for a matching to exist.

A pair $(x, y) \in X \times X$ is a **blocking pair** for the matching $\mathcal{M}$ if x and y both prefer each other to their own partners in $\mathcal{M}$.
A matching $\mathcal{M}$ is **stable** or a **solution**, if there is no blocking pair in $X \times X$ for $\mathcal{M}$.

Not every instance of the stable roommates problem has a solution, e.g.

1 [2 3 4]
2 [3 1 4]
3 [1 2 4]
4 [1 2 3]

Three matchings, but each has a blocking pair:
{(1,2), (3,4)} is blocked by (2,3); {(1,3), (2,4)}, by (1,2); and {(1,4), (2,3)}, by (1,3).

Irving's algorithm: 2 phases, Phase 1 is like the G-S stable marriage algorithm:

Phase 1

while $\exists x \in X$ who has no proposal held and has not proposed to everyone, do

- *next* := the next on x's list to whom x has not yet proposed
- x proposes to *next*
- if *next* holds no proposal, then
  - *next* holds x's proposal
- else y := the individual from whom *next* holds a proposal
  - if *next* prefers y to x, then *next* rejects *proposer*
  - else
    - *next* rejects y
    - *next* holds x's proposal

Lemma 3. After the moment $x \in X$ is first proposed to, x always holds a proposal.

Lemma 4. If $y \in X$ rejects $x \in X$ in Phase 1, then x and y cannot be partners in any stable matching.

Corollary 4.1. If during Phase 1, $x \in X$ proposed to $y \in X$, then, in any stable matching

(i) x cannot have a better partner than y;

(ii) y cannot have a worse partner than x.

Corollary 4.2. If after Phase 1, $x \in X$ has no proposal held and has proposed to everyone on his list, then no stable matching exists.

Corollary 4.3. If, after Phase 1, $y \in X$ holds a proposal from $x \in X$, then y's preference list can be reduced (without eliminating possible partners) by deleting from it

(i) all those to whom y prefers x (i.e. all those after x);

(ii) all those who hold a proposal from a person whom they prefer to y (including those who have rejected y, i.e. all those before x);

In the resulting table,

(iii) y is first on x's list; and x, last on y's;

(iv) in general, b appears on a's list if and only if a appears on b's.

We will refer to this table as the Phase 1 table.
For a particular instance, the Phase 1 table is always the same.

Definition 5. A table T for an instance is **stable** if, in T:

(i) x is first in y's list, abbreviated $f_T(x) = y$, iff y is last in x's list, abbreviated $l_T(y) = x$;

(ii) for $x, y \in X$, x is not in y's list iff y is not in x's iff x prefers $l_T(x)$ to y or y prefers $l_T(y)$ to x (in the original table);

(iii) no list is empty.

f_T and l_T are inverse functions $X \to X$.
Note that once all of the $f_T(x)$ are determined (or all of the $l_T(x)$ are), then the whole table is determined by (ii).

Lemma 6. Definition 5.(i) and (ii) hold for the Phase 1 table, and so if no list is empty, the table is stable.

Lemma 7. If each list in a stable table T contains exactly one person, then they specify a stable matching.

Proof. By Definition 5.(i), the lists specify a matching. Suppose that $(x, y) \in X \times X$ but x and y are not paired in the matching. Then x and y are not in each others' lists in T. Then, by Definition 5.(ii), x prefers the last (i.e. sole) person on his list to y, or y, the last (i.e. sole) person on his list, to x, so that (x, y) cannot be a blocking pair. The matching is therefore stable.

□

Corollary 7.1. If each list in Phase 1 table contains exactly one person, then they specify a stable matching.

Phase 2 overview:
We prove that at each step until termination, Definition 5.(i) and (ii) hold:

Base case: Lemma 6 (exit reporting that no stable matching exists if any list is empty).

Inductive step:

1. reduced lists contain exactly one person each: done by Lemma 7.
2. at least one list is empty: end the procedure since no stable matching.
3. no empty list, but one has at least 2 in it: search for a *rotation* and *eliminate* it, i.e. reduce the preference lists again with respect to this rotation in such a way that Definition 5.(i) and (ii) continue to hold. If any list becomes empty, we exit reporting that no stable matching exists. Otherwise, the inductive step holds.

Need to justify 2 and 3.

Definition 8. A **rotation** in a stable table is a cyclic sequence $(a_1, b_1), (a_2.b_2), \ldots, (a_r, b_r) \in X \times X$ of pairs of individuals such that, where $a_{r+1} = a_1$ and $b_{r+1} = b_1$:

(i) the a_i are distinct;

(ii) b_i is first on a_i's list ($f_T(a_i) = b_i$) for each i; equivalently, a_i is last on b_i's list for each i ($l_T(b_i) = a_i$);

(iii) b_{i+1} is the second on a_i's list for each i.

$$
\begin{array}{rlll}
a_1 & [\, b_1 & b_2 \ldots] & b_1[\ldots \quad a_r \ldots \quad a_1] \\
a_2 & [\, b_2 & b_3 \ldots] & b_2[\ldots \quad a_1 \ldots \quad a_2] \\
a_3 & [\, b_3 & b_4 \ldots] & b_3[\ldots \quad a_2 \ldots \quad a_3] \\
 & \vdots & & \vdots \\
a_i & [\, b_i & b_{i+1} \ldots] & b_i[\ldots a_{i-1} \ldots \quad a_i] \\
 & \vdots & & \vdots \\
a_{r-1} & [\, b_{r-1} & b_r \ldots] & b_{r-1}[\ldots a_{r-2} \ldots a_{r-1}] \\
a_r & [\, b_r & b_1 \ldots] & b_i[\ldots a_{r-1} \ldots \quad a_r]
\end{array}
$$

Lemma 9. A stable table in which a list has length at least 2 contains a rotation.

Proof. To find a rotation, we let $p_1 \in X$ be an individual whose list has length at least 2, and define, inductively:
q_{i+1} = the second in p_i's reduced list
p_{i+1} = the last in q_{i+1}'s reduced list (so q_{i+1} is first in p_{i+1}'s)
until this sequence repeats some p_s, so that $p_{s+r} = p_s$.

Definition 5.(i) ensures that p_i will have at lest 2 individuals in his reduced list so that we can choose q_{i+1} at each step.

Let $a_i = p_{s+i-1}$ for $i = 1, \ldots, r$ and $b_i = q_{s+i-1}$, for $i = 1, \ldots, r$ to give us a rotation.

We call $p_1, \ldots, p_{s-1}$ the **tail** for the rotation.

□

Having found a rotation,

• *eliminate* the rotation: force each b_i to reject a_i and have each a_i propose to b_{i+1}

• $x \in X$ and $y \in X$ are removed from each other's lists in the rotation elimination if (and only if) one of them is b_i for some i and the other succeeds a_{i-1} in b_i's list.

a_i [b_i $b_{i+1} \ldots$] becomes a_i [$b_{i+1} \ldots$], and
b_i [$\ldots a_{i-1} \ldots a_i$] becomes b_i [$\ldots a_{i-1}$].
remove b_i from the lists of the successors of a_{i-1} in b_i's list.

Lemma 11. Let $(a_1, b_1), \ldots, (a_r, b_r)$ be a rotation in a stable table T. Then,

(i) in any stable matching contained in T, either a_i and b_i are partners for all values of i or for no value of i;

(ii) if there is such a stable matching in T in which a_j and b_j are partners, then modifying it so that a_i and b_{i+1} are instead partners for each i also gives a stable matching.

Corollary 11.1. If the original problem instance admits a stable matching, then there is a stable matching contained in any of the tables in the sequence of reductions.

Corollary 11.2. If one or more among the lists in one of the tables in the sequence is empty, then the original problem instance admits no stable matching.

Correctness of the algorithm follows.

Phase 2

reduce the preference lists as described in Corollary 4.3
if any list becomes empty during the reductions, then
    exit the procedure reporting that there is no stable matching
if, after the reductions, each list has a unique individual in it, then
    return $\mathcal{M} := \{(x, y) \in X \times X \mid y \text{ is in } x\text{'s reduced list}\}$
while there remains an individual in X with at least 2 in his reduced list, do
    find a rotation using the tail of the previous one (if any)
    eliminate the rotation
    if any list becomes empty during the reductions, then
        exit the procedure reporting that there is no stable matching
return $\mathcal{M} := \{(x, y) \in X \times X \mid y \text{ is in } x\text{'s reduced list}\}$

Theorem 12. The running time of Irving's algorithm is $O(n^2)$.

Proof. We set up a ranking matrix, such that $rank[x, y] = i \Leftrightarrow y$ is ith on x's original list, to check preferences in constant time.
- Setting up the ranking matrix : $O(n^2)$.
- Phase 1 : $O(n^2)$.
- All removals (Phase 1 and 2): $O(n^2)$.
- Checking and constructing a stable matching: $O(n^2)$.
- Time spent finding rotations + updating maintained values (in a "careful implementation"): $O(n^2)$.

"Careful implementation":
- store the first unmatched individual in X
- store the first, second and last in the list of each $x \in X$ for implicit removals
- start search the for a rotation at the end of tail of the previous rotation (if any) and otherwise at the first unmatched.

□