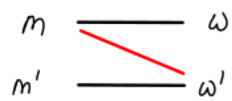
1.1 Stable Matching Problem

Goal: Given M men and W women. Each man ranks all women and every woman ranks all men. We want to order the men and women into pairs so that

a. each man and each woman appears in exactly one pair

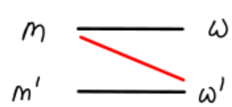
b. there is no (m, w) and (m', w') pairs such that man m prefers w' to w and w' prefers m to m', as a picture:



If this was the case man m and woman w' would both be better off forming a (m, w') pair instead.

Definitions

- $M := \{m_1, m_2, ..., m_M\}$ is the set of men
- $W := \{w_1, w_2, ..., w_W\}$ is the set of women
- $W \times M := \{(w, m), w \in W, m \in M\}$ is the set of all ordered pairs of men and women from set M and W, respecrively
- A *matching* J is a subset of $W \times M$, i.e. $J \subseteq W \times M$, such that every $m \in M$ and every $w \in W$ appears in at most one pair.
- A $perfectmatching\ J'$ is a subset of $W \times M$, $J' \subseteq W \times M$, such that every $m \in M$ and every $w \in W$ appears in exactly one pair.
- A pair $(w', m) \in W \times M$ is called an instability with respect to a matching J if $(w', m) \notin J$, but $(w, m) \in J$ and $(w', m') \in J$ and m and w' prefer the other to their partners in J, i.e. m ranks w' higher than w and w' ranks m higher than m'.



- A matching J is stable if it is perfect and there is no instability with respect to J

Conclusion: Our goal above can be rephrased as follows: given a set M of M men and a set W of W women, where every man ranks all women and every woman ranks all men, is there a stable matching $J \subseteq W \times M$?

Questions:

- How to find an answer to this question?

- $\longrightarrow$ come up with a corresponding so-called algorithm

Definition: An algorithm is a finite list of instructions such that

1. each instruction can be precisely described (in words, pictures etc.)
2. starting from an initial state and with initial input, the instructions describe a computation that proceeds through a finite number of successive states, eventually producing an output and terminating at a final state
3. each state is well-defined and depends only on the input and results of previous states
4. the final state is reached after a finite number of states per every valid initial state and initial input (no infinite loops)

Note: the transition from one state to the next state may not necessarily be deterministic (e.g. randomized algorithms).

Group work: Find an algorithm that finds a stable matching for M=N (15 minutes)

$\longrightarrow$ show slide with my algorithm and go through one example (see next page)

Strategic considerations:

- How do we convince ourselves that a given algorithm is correct?

  $\longrightarrow$ prove the correctness of the algorithm

- How lond does the algorithm take to return an answer and how does this change as we vary the input to the algorithm (here M and N)? In particular, does the algorithm always terminates after a finite number of steps, provided the input is valid (here, M and N are natural numbers > 0)?

  $\longrightarrow$ determine the time and space complexity of the algorithm

1.1.1 Time complexity of the stable matching algorithm

Motivation: We would like to know how the number of step that the algorithm takes to derive an answer varies as function of $M = N$.

For example, if we double M i.e. $M \rightarrow 2M$, does this double the number of steps that the algorithm takes to return a solution?

Group work:

1. What happens to a man m during the execution of the algorithm?

   Observation 1: m remains paired starting when he gets first proposal and his parner w get better (as ranked by m) as the algorithm proceeds.

2. What happens to a woman w during the execution of the algorithm?

<u>Observation 2:</u> The sequence of men to which w proposed get worse as the algorithm proceeds.

The Stable Matching Algorithm:

set all $m \in M$ and $w \in W$ to free
While some free woman w hasn't proposed to every man {
  $m \leftarrow$ highest ranking man that w hasn't yet proposed to
  if m is free {pair w and m}
  else {
    $w' \leftarrow$ current fiance of m
    if m prefers w to w' {
      set w' free
      pair w and m
    }
  }
}

Theorem: The stable Matching algorithm (the Gale-Shepley algorithm) terminates after $\leq N^2$ iterations of the while-loop if $N = M = W$ where W denotes the number of women and M denotes the number of men.

Proof: At every iteration of the while-loop a woman poposes to a man she hasn't yet proposed to. Given that there are N men, each woman makes at most N proposals. As there are N women, there are therefore at most N^2 iterations of the while-loop. $\square$

Question: Does the theorem imply that the total execution time of the algorithm is at most $O(N^2)$?

Answer: No. The total execution time also depends on

(a) How many steps it takes to identify and unpaired woman.
(b) How many steps it takes a womand to decide whom to propose to.
(c) How many steps it takes for a man m to decide if he prefers woman w to w'.

Group work: Propose data structures so that tasks (a) to (c) can be answered in $O(1)$ time.

Answers:

For (a), keep a list/stack/queue of unpaired women.

For (b), store, for each woman separately

- array of men ranked order of decreasing preference and
- the rank of the last man she proposed to

For (c), store, for each man, separately,

- array indexed by women (use natural numbers) with their rank in the man's preference list and
- the number of his current partner (or null, if unpaired).

Note: The data structure details are sometimes left unspecified in the algorithm but obviously need to be determined for any implementation of the algorithm

1.1.2 Correctness of the stable matching algorithm

Lemma 1: If a woman w is unpaired, then there is at least two man that she hasn't proposed to yet.

Proof: (use proof by contrapositive) According to observation 1, if woman w has proposed to every man, then every man is engaged. Since the number of woman (W) is equal to the number of men (M), this implies that every woman including w is engaged. □

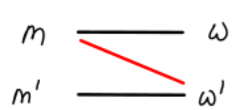
Reminder: A proof by contrapositive explore the fact that "$P \Rightarrow Q$" is equivalent to "$notQ \Rightarrow notP$". For example, "'If it is my car, then it is red"' is equivalent to "If the car is not red, then it is not mine".

Lemma 2: The set of pairs $J \subseteq W \times M$ returned by the algorithm is a perfect matching, i.e. each $w \in W$ and each $m \in M$ appears in exactly one pair.

Proof: The algorithm terminates when every woman is paired. We know that every woman is paired to at most one man at any time. So, at the end of the algorithm, every woman is paired to exactly one man. As the number of women (w) is equal to the number of men (M), this implies that every man is paired to exactly one woman. □

Theorem: The matching returned by the algorithm is stable.

Proof: (We use a proof by contradiction.) Suppose there was an instability, i.e. a pair (w, m) and (w', m') where m prefers w' to w and also w' prefers m to m'.



From the algorithm we know that w's last proposal was to m', as they would otherwise not form a pair. Did w' propose to m?

(a) Suppose the answer is "no". As woman w' ranks m higher than m' and as women propose to men in order of decreasing preference, this would be a contradiction.

(b) Suppose the answer is "yes". If w' proposed to m and as she is no longer paired to m, she must have been at some point in the algorithm rejected in favour of another woman w''. Either $w'' = w$ or w'' was also rejected in favour of some other woman etc. until the last woman was eventually rejected in favour of w, i.e. the final partner of m. Either way, m prefers w to w' (see also observation 1), which is a contradiction to our starting point. □

Reminder: A proof by contradiction establishes the validity of a proposition by showing that the proposition being false would imply a contradiction.

Note: It can be shown that using this algorithm, women are paired to the best possible partner out of all stable matchings, whereas men end up with the worst possible partner.