Module 1 - Algorithm Development

Describing the Problem

Session 1 - 5th Jan, 2022

Stages of Algorithm development

Step by Step

- Describe the Problem
- Identifying a suitable technique
- Design of an algorithm
- Proof of Correctness of the algorithm

Stable Matching Problem

Outline

- Students apply for Internship to Companies
- Companies hire Students as Interns
- In both the cases, they have their own preferences
- If a student receives more than one offer, he has a choice to select
- Similarly, a company will have more choice in the selection of Intern
- If any student, denies an offer, it has to be given to the student in next waitlist of the company
- Similarly, If a students receives an offer from the company that he is being waiting for, then he will be forced to say “NO” to the order, currently he is holding.
- So in this scenario, Given a set of preferences among employers and applicants, can we assign applicants to employers so that for every employer E, and every applicant A who is not scheduled to work for E, at least one of the following two things is the case?
  - E prefers every one of its accepted applicants to A; or
  - A prefers her current situation over working for employer E.
- If this holds, the outcome is stable: individual self-interest will prevent any applicant/employer deal from being made behind the scenes.

Formulating the Problem

Step 1

- Each applicant is looking for a single company, but each company is looking for many applicants;
- Initially, to understand, lets eliminate the company from the scene. Each of n applicants applies to each of n companies, and each company wants to accept a single applicant.
- We observe that this special case can be viewed as the problem of devising a system by which each of n men and n women can end up getting married: our problem naturally has the analogue of two “genders”—the applicants and the companies—and in the case we are considering, everyone is seeking to be paired with exactly one individual of the opposite gender.

Matching the Suitable Preference

Not possible all the Time

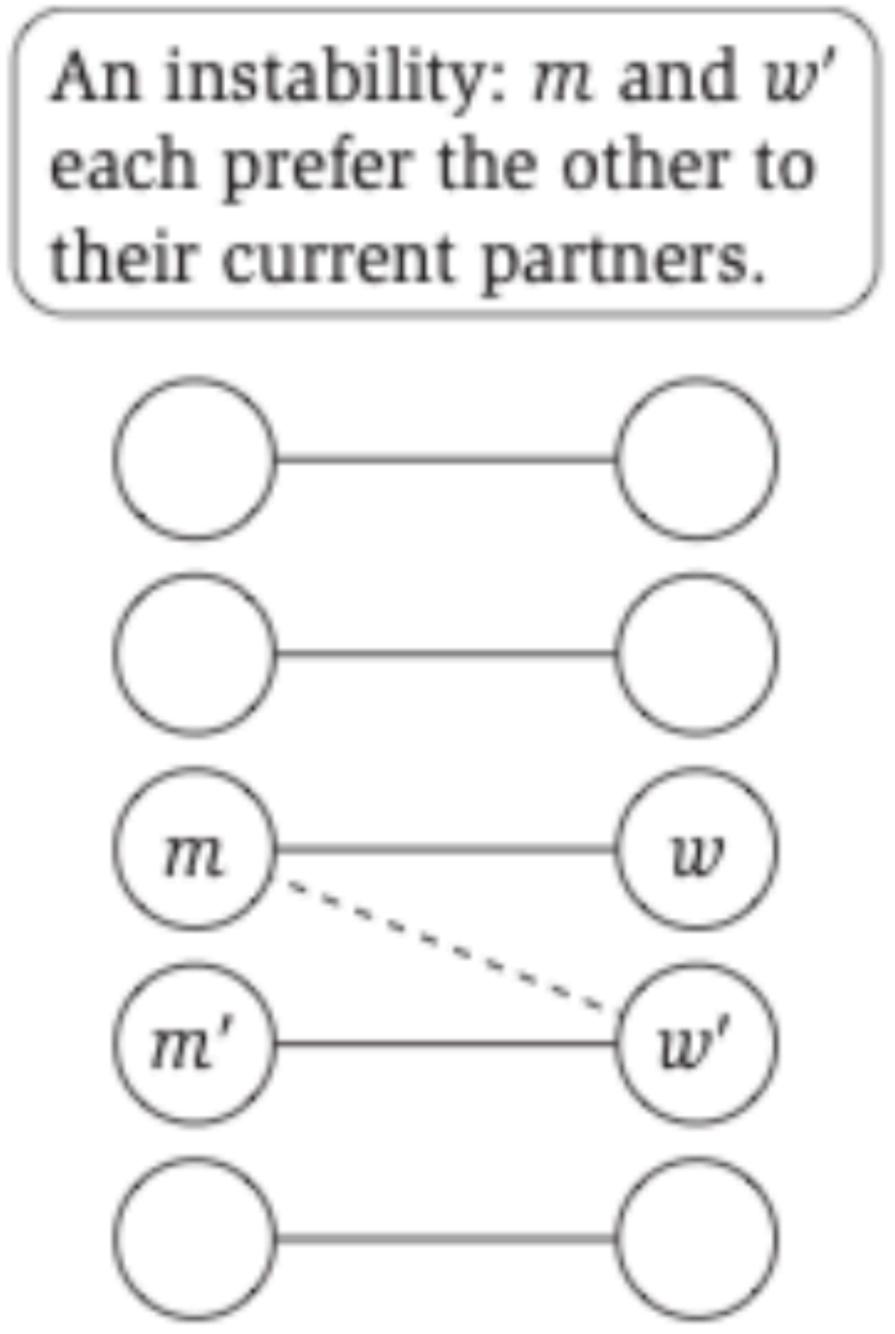
An instability: m and w' each prefer the other to their current partners.
m
w
m'
w'

Figure 1.1 Perfect matching S with instability (m, w').

So consider a set $M = \{m_1, \ldots, m_n\}$ of n men, and a set $W = \{w_1, \ldots, w_n\}$ of n women. Let $M \times W$ denote the set of all possible ordered pairs of the form (m, w), where $m \in M$ and $w \in W$. A *matching* S is a *set* of ordered pairs, each from $M \times W$, with the property that each member of M and each member of W appears in at most one pair in S. A *perfect matching* S' is a matching with the property that each member of M and each member of W appears in *exactly* one pair in S'.

- In the present situation, a perfect matching corresponds simply to a way of pairing off the men with the women, in such a way that everyone ends up married to somebody, and nobody is married to more than one person—there is neither singlehood nor polygamy.

Instability

A painful process

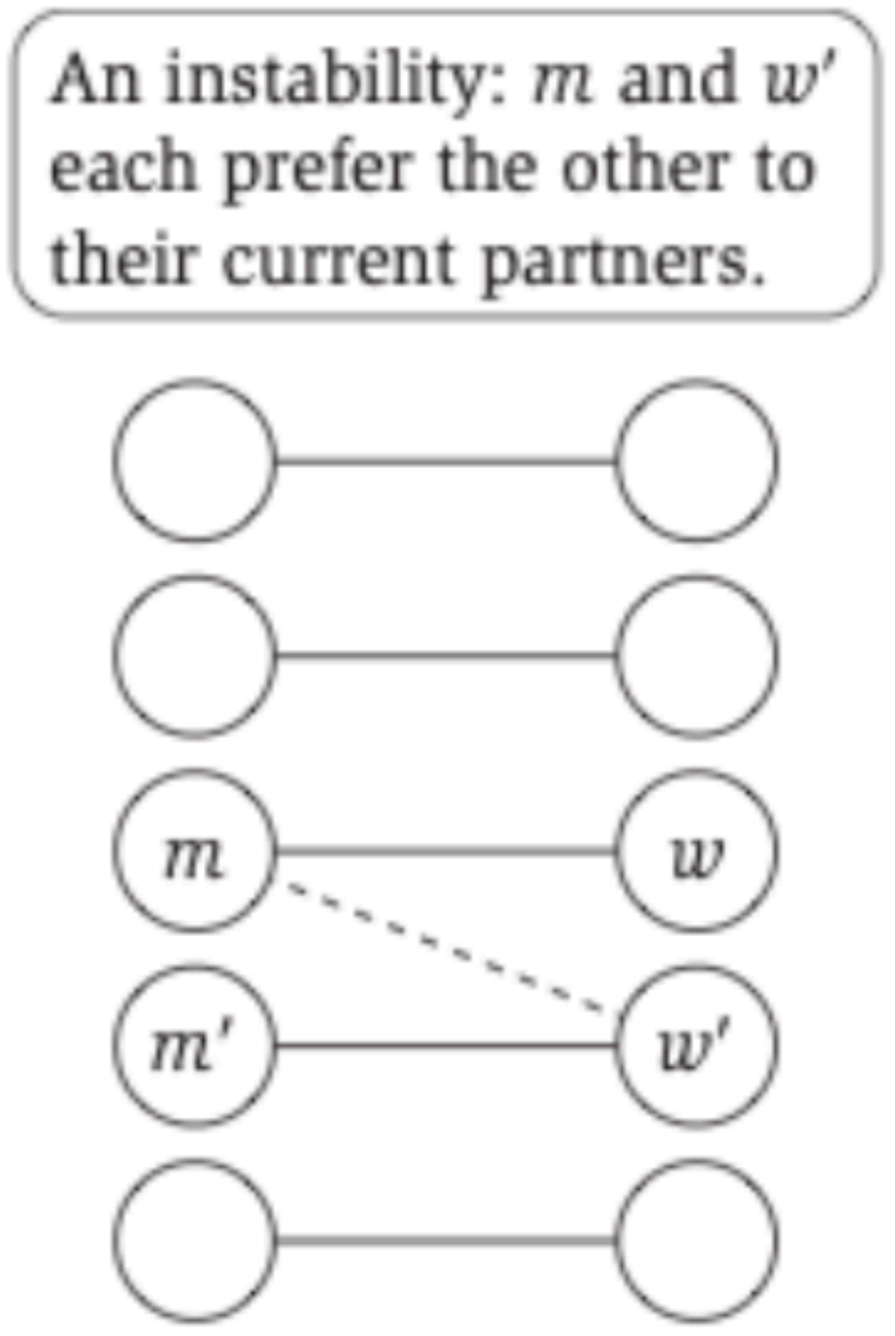
An instability: m and w' each prefer the other to their current partners.
m
w
m'
w'

Figure 1.1 Perfect matching S with instability (m, w').

- Each man m ∊ M ranks all the women; we will say that m prefers w to w′ if m ranks w higher than w′. We will refer to the ordered ranking of m as his preference list. We will not allow ties in the ranking. Each woman, analogously, ranks all the men.
- For a perfect matching, the following situation may arise.
  - There are two pairs (m,w) and (m′,w′) in S with the property that m prefers w′ to w, and w′ prefers m to m′.
- The set of marriages is not self- enforcing. We’ll say that such a pair (m, w′) is an instability with respect to S: (m, w′) does not belong to S, but each of m and w′ prefers the other to their partner in S.

Our Goal

Common perspective

- Set of marriages with no instabilities. We'll say that a matching S is stable if (i) it is perfect, and (ii) there is no instability with respect to S.
    - Does there exist a stable matching for every set of preference lists?
    - Given a set of preference lists, can we efficiently construct a stable matching if there is one?
- For Instance,
    - m prefers w to w′.
    - m′ prefers w′ to w.
    - w prefers m′ to m.
    - w′ prefers m to m′.

Module 1 - Algorithm Development

Designing an algorithm

Session 2 - 7th Jan, 2022

Designing the Algorithm

Motivation

- Initially, everyone is unmarried
    - ***Situation:*** Suppose an unmarried man *m* chooses the woman *w* who ranks highest on his preference list and *proposes* to her. Can we declare immediately that (*m*, *w*) will be one of the pairs in our final stable matching? Not necessarily: at some point in the future, a man *m*′ whom *w* prefers may propose to her.
    - On the other hand, it would be dangerous for *w* to reject *m* right away; she may never receive a proposal from someone she ranks as highly as *m*. So a natural idea would be to have the pair (*m*, *w*) enter an intermediate state—*engagement*.

Designing the Algorithm

Motivation (Continued..)

- Suppose we are now at a state in which some men and women are free— not engaged—and some are engaged.
  - ***Situation:*** The next step could look like this. An arbitrary free man *m* chooses the highest-ranked woman *w* to whom he has not yet proposed, and he proposes to her. If *w* is also free, then *m* and *w* become engaged.
  - Otherwise, w is already engaged to some other man m′. In this case, she determines which of m or m′ ranks higher on her preference list; this man becomes engaged to w and the other becomes free.

Designing the Algorithm

Motivation (Continued..)

- Finally, the algorithm will terminate when no one is free;
  - ***Situation:*** At this moment, all engagements are declared final, and the resulting perfect matching is returned.

Gale-Shapely Algorithm

A Straight forward Approach

```
Initially all $m \in M$ and $w \in W$ are free
While there is a man $m$ who is free and hasn't proposed to
every woman
   Choose such a man $m$
   Let $w$ be the highest-ranked woman in $m$'s preference list
      to whom $m$ has not yet proposed
   If $w$ is free then
      $(m, w)$ become engaged
   Else $w$ is currently engaged to $m'$
      If $w$ prefers $m'$ to $m$ then
         $m$ remains free
      Else $w$ prefers $m$ to $m'$
         $(m, w)$ become engaged
         $m'$ becomes free
      Endif
   Endif
Endwhile
Return the set $S$ of engaged pairs
```

- This algorithm doesn't produces the set of Stable states immediately, instead it enters into many intermediate steps.

Analysing the Algorithm

Several cases to be considered

- First consider the view of a woman w during the execution of the algorithm. For a while, no one has proposed to her, and she is free. Then a man m may propose to her, and she becomes engaged. As time goes on, she may receive additional proposals, accepting those that increase the rank of her partner. So we discover the following.
  - ***Case 1:*** w remains engaged from the point at which she receives her first proposal; and the sequence of partners to which she is engaged gets better and better (in terms of her preference list).
    - The view of a man m during the execution of the algorithm is rather different. He is free until he proposes to the highest-ranked woman on his list; at this point he may or may not become engaged. As time goes on, he may alternate between being free and being engaged; however, the following property does hold.

Analysing the Algorithm

Case by Case

- ***Case 2:*** The sequence of women to whom m proposes gets worse and worse (in terms of his preference list).
- ***Case 3:*** The G-S algorithm terminates after at most n^2 iterations of the While loop.
    - In the case of the present algorithm, each iteration consists of some man proposing (for the only time) to a woman he has never proposed to before.
    - So if we let P(t) denote the set of pairs (m, w) such that m has proposed to w by the end of iteration t, we see that for all t, the size of P(t + 1) is strictly greater than the size of P(t).
    - But there are only n^2 possible pairs of men and women in total, so the value of P(·) can increase at most n^2 times over the course of the algorithm.
    - It follows that there can be at most n^2 iterations.

Analysing the Algorithm

Case by Case

- Two things to infer at this stage
  - First, there are executions of the algorithm (with certain preference lists) that can involve close to n2 iterations, so this analysis is not far from the best possible.
  - Second, there are many quantities that would not have worked well as a progress measure for the algorithm, since they need not strictly increase in each iteration
    - For example, the number of free individuals could remain constant from one iteration to the next, as could the number of engaged pairs.

Analysing the Algorithm

Case by Case

- ***Case 4:*** If m is free at some point in the execution of the algorithm, then there is a woman to whom he has not yet proposed.
- ***Case 5:*** The set S returned at termination is a perfect matching.
    - The set of engaged pairs always forms a matching. Let us suppose that the algorithm terminates with a free man m. At termination, it must be the case that m had already proposed to every woman, for otherwise the While loop would not have exited.
    - But this contradicts (Case 4), which says that there cannot be a free man who has proposed to every woman
    - Finally, we prove the main property of the algorithm—namely, that it results in a stable matching.

Analysing the Algorithm

Case by Case

- ***Case 6:*** Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.
  - We have already seen, in (1.5), that S is a perfect matching. Thus, to prove S is a stable matching, we will assume that there is an instability with respect to S and obtain a contradiction. As defined earlier, such an instability would involve two pairs, (m, w) and (m′, w′), in S with the properties that
  - m prefers w′ to w, and
  - w′ prefers m to m′.
  - *Special Case:* All executions yields the same matching.

Analysing the Algorithm

Case by Case

- ***Case 7:*** Every execution of the G-S algorithm results in the set S*
    - The contradiction reveals that, when ever a man receives a rejection, he is likely to get a partner who is low in priority.
- ***Case 8:*** In the stable matching S*, each woman is paired with her worst valid partner.
    - Suppose there were a pair (m, w) in S_* such that m is not the worst valid partner of w. Then there is a stable matching S′ in which w is paired with a man m′ whom she likes less than m. In S′, m is paired with a woman w′/= w; since w is the best valid partner of m, and w′ is a valid partner of m, we see that m prefers w to w′.
    - But from this it follows that (m, w) is an instability in S′, contradicting the claim that S′ is stable and hence contradicting our initial assumption.
- ***Case 9:*** There is no instability with respect to returned matching S.

Extra Materials to learn

Some Samples

- **Solved Exercise 1:**
  Consider a town with n men and n women seeking to get married to one another. Each man has a preference list that ranks all the women, and each woman has a preference list that ranks all the men.

- The set of all 2n people is divided into two categories: good people and bad people. Suppose that for some number k, $1 \leq k \leq n - 1$, there are k good men and k good women; thus there are $n - k$ bad men and $n - k$ bad women.

- Everyone would rather marry any good person than any bad person. Formally, each preference list has the property that it ranks each good person of the opposite gender higher than each bad person of the opposite gender: its first k entries are the good people (of the opposite gender) in some order, and its next $n - k$ are the bad people (of the opposite gender) in some order.

- Show that in every stable matching, every good man is married to a good woman.

Extra Materials to learn

Some Samples

- **Solution Exercise 1:**
  A natural way to get started thinking about this problem is to assume the claim is false and try to work toward obtaining a contradiction. What would it mean for the claim to be false? There would exist some stable matching M in which a good man m was married to a bad woman w.

- Now, let’s consider what the other pairs in M look like. There are k good men and k good women. Could it be the case that every good woman is married to a good man in this matching M? No: one of the good men (namely, m) is already married to a bad woman, and that leaves only k – 1 other good men. So even if all of them were married to good women, that would still leave some good woman who is married to a bad man.

- Let w′ be such a good woman, who is married to a bad man. It is now easy to identify an instability in M: consider the pair (m, w′). Each is good, but is married to a bad partner. Thus, each of m and w′ prefers the other to their current partner, and hence (m, w′) is an instability. This contradicts our assumption that M is stable, and hence concludes the proof.

Extra Reading

Additional Exposure

- The five representative problems
- Full context of G-S Algorithm and sample use case, Click Here
- Example - College Admissions
- Implementation Challenge at SPOJ or Codechef
- Sample Solution at Geek for Geeks