Introduction and Matching

Julius January 10, 2024

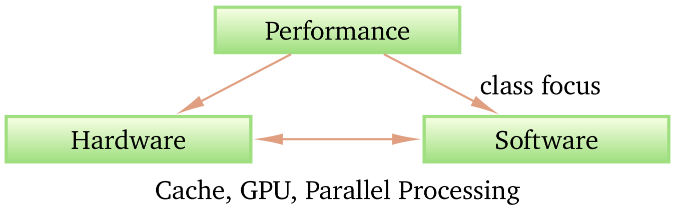
1 High Level Syllabus

- Introduction
- Reviews of some prerequisite topics
- Major algorithmic techinques: Greedy, Divide and Conquer (Exam1), Dynamic Programming
- Network Flow: One tipical probelm can be used by all others indirectly (Exam2)
- Computational Complexity Theory: A couple of computational classes necessary for designing algorithms.
- Approximation Algorithms: Numerical Analysis
- Linear Programming: to solve whole bunch of problems (Exam3)

2 Introduction

Key attributres of an Algorithm: **Correctiness** and **Performance**. Video Algorithmsa are Taking over the world: Christopher Steiner at TEDxOrgangeCoast.

1. ~~"An Algorithm is a set of instructions in machine language"~~
2. ~~"... As algorithmic science advanced on Wall Street ..."~~
3. ~~"... invite 6 million algorithms for a listen ... "~~



When studying a problem, we go throught the following **four steps**:

1. Come up with a concise problem statement
2. Present a solution
3. Prove the correctness of your solution
4. Analyse its performance (complexity analysis)

3 Stable Matching

Problem: How do we match n mem with n women so they stay together ever after?

3.1 Come up with a concise problem statement

we have a set of n mmen $M = \{m_1, m_2, \dots, m_n\}$, and a set of n women $W = \{w_1, w_2, \dots, w_n\}$

Matching: $\mathcal{S}$ a set of ordered pairs, no pairs share the same man or woman.

Perfect Matching: $\mathcal{S}'$ is a matching with the propety that each member of M and each member of W appear in exactly one pair in $\mathcal{S}'$.

Notion of perferences: Each man $m \in M$ ranks all women: m prefers w to w' if m ranks w higher than w'. Then Ordered ranking of m in his preference list: $P_{m_i} = \{w_{i_1}, w_{i_2}, \dots, w_{i_n}\}$. Similary, each woman w ranks all man: $P_{w_j} = \{m_{j_1}, m_{j_2}, \dots, m_{j_n}\}$

Define instability: Set there are two pairs (m,w) and (m', w') in a matching $\mathcal{S}$, while m prefer w' over w and w' prefer m over m'. Such a pair (m,w') is an instability with respect to the matching $\mathcal{S}$.

Define stable matching: A matching $\mathcal{S}$ is stable if it is a perfect matching and has no instability.

Define input and expected output: Input: Perference lists for a set of n men and n women. Output: A set a n marriages (ordered pairs) with no instabilities.

3.2 Present a Solution

Gale-Shapley Algorithm O(nm): Initially all $m \in M$ and $w \in W$ are free.

```
    while there is a free man:
        choose a free man
        let w be the highest-ranked woman of m1 to whom he has not yet proposed
        if(w is free):
            (m1, w) become engaged
        else if(w is engaged to m2 but prefers m1 to her fiance m2):
            (m1, w) become engaged
            m2 becomes free
        else:
            w rejects m1, and m1 remains free
    return the set of engaged pairs
```

3.3 Prove the correctness of this solution

we will prove that the while loop terminates in at most n^2 iterations with the following **Observations:**

a. Each iteration of the while loop consists of a free man proposing to a woman, while the number of free individuals could remain constant from one iteration to the next, as could the number of engaged pairs.

b. After her first engagement, a woman will always stay engaged.

Theorem 3.1. *While Loop in Gale-Shapley algorithm terminates*

Proof 1. It's impossible for a man to propose more than n women, and we n man, so the while loop terminates in at most n^2 iterations. □

Proof 2. Let $P(t)$ denote the set of propose pairs (m, w) that m has proposed to w, by the end of iteration t. $\forall t \in \mathbb{N}^+, P(t+1) > P(t)$. But there are only n^2 possible propose pairs of men and women in total, so the value of $P(\cdot)$ can increase at most n^2 times over the course of while loop of algorithm. □

Lemma 3.2. *For the free man in the execution of algorithm, there is a woman to whom he has not yet proposed.*

Proof. Suppose there comes a point when m is free but has already proposed to every woman. Then each of the n women is engaged at this point in time. Since the set of engaged pairs forms a matching, there must also be n engaged men at this point. But there are only n men total, and m is not engaged, so this is a contradiction. □

Theorem 3.3. *At termination we have a perfect matching*

Proof. Let us suppose that the algorithm terminates with a free man m. After the While loop, man m must had already proposed to every woman. But this contradicts Lemma 3.2, which says that there cannot be a free man who has proposed to every woman. □

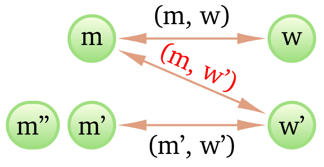
Theorem 3.4. *There are no instability in our perfect matching*

Proof. First, some observations:

1. From the woman's perspective, she starts signle and once she gets engaged, she can only gets into better engagement with respect to her preference.

2. From the man's perspective, he starts single and once he gets engaged, he might get dropped repeatedly only to settle for a lower ranking woman with respect to his preference.

We will assume that an instability exists in our perfect matching produced by Gale-Shapley algorithm, involving two pairs (m, w) and (m', w'). Did m propose to w' at some point in the execution of the algorithm?



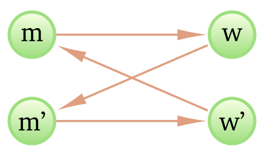
If no, then w must be higher than w' on his preference list. -> Contradiction.

If yes, he must have been rejected in favor of m'' and due to observation 1, either m'' is m' or m' is better than m''. -> Contradiction.

□

Lemma 3.5. *The Stable Matching is not unique.*

Proof. Consider the following example, where arrows indicate preference:
Man propose: $(m, w), (m', w')$; Women propose: $(w, m'), (w', m)$



□

3.4 Complexity Analysis

Operations involved in each iteration of the while loop:
Step 1&5: Identify a free man. Place a man back in the free list of man.

	Get	Put
Stack	O(1)	O(1)
Queue	O(1)	O(1)
Linked List	O(1)	O(1)
Array	O(1)	O(1)

Step 2: For a man m, identify the highest ranked woman to whom he has not yet proposed.
Keep an array $Next[1 \dots n]$ where $Next[m]$ points to the position of the next woman on m's preference list that m will propose to. And given the men's preference lists: $ManPref[1 \dots n][1 \dots n]$, where $ManPref[m][i]$ denote the i^{th} woman on man m's preference list. we can get $w = ManPref[m][Next[m]]$ in **O(1)**.
Step 3: For a woman w, decide if w is engaged, and if so to whom.
Keep an current array $current[1 \dots n]$ where $current[w]$ is Null if w is single and set to m if w is engaged to m in **O(1)**.
Step 4: For a woman w and two man m and m', decide if w prefers m to m'.

$WomanPref_i$	3	8	4	31	1	$\cdots$
	1	2	3	4	5	6

$RankingList_i$	5		1	3		$\cdots$
	1	2	3	4	5	6

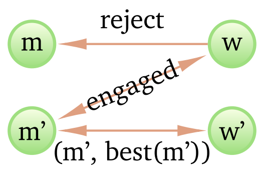
We can change the $WomanPref_i$ list of man m_i to $RankingList_i$, so to get the rank of m and m' in **O(1)**.
Preparation before entering Gale-Shapley iteration:
Create a $WomanRanking$ array where $WomanRanking[w][m]$ contains the rank of man m based on w's preference, which takes $O(n^2)$ time.
The Gale-Shapley iteration takes $O(n^2)$ time, so the total time complexity is $O(n^2)$.

3.5 Extensions

Def: Woman m is in a **valid partner** of man m if there exists a stable matching that contains the pair (m, w). Consider the preference list for man m_i, $P_{m_i} = \{w_{i_1}, w_{i_2}, \cdots, w_{i_k}, \cdots, w_{i_m}, \cdots, w_{i_l}, \cdots, w_{i_n}\}$, where $w_{i_k}, w_{i_m}, w_{i_l}$ is the only three valid partners of m_i, and w_{i_k} is the highest ranked valid and **best partner** of m_i.

Lemma 3.6. *Every execution of the Gale-Shapley algorithm (when men are proposing) results in the same stable matching regardless of the order in which men propose, denoted as* $S^* : \{(m, best(m)) : m \in M\}$

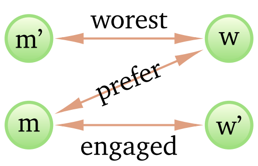
Proof. Suppose m is the **first** man rejected by its valid partner w in favor of m' in a stable matching S' where (m, w) is a valid pair and w = best(m). The rejection happens because this moment w has formed an engagement with m'. Since w is already a valid partner of m, let's say in S', m' paired with w'.



In the rejection execution, m' had not been rejected by any valid partner when he engaged to w. Since man proposed in decreading order of preference, m' must prefer w to w'. At the same time, w prefer m' to m for rejection. Since $(m', w) \notin S'$, it follows that (m', w) is an instability of S'. □

Lemma 3.7. *In the stable matching S^*, women end up with their worest valid partners.*

Proof. Suppose there is a pair (m, w) in S^* where m is not the worst valid partner of w. So there must be another stable matching S', where w is paired with a man m' whom she likes even less than m. Let's say in S', m is paired with a valid partner $w' \neq w$.



Since (m, w) is a pair in S^*, according to Lemma 3.6, w must be the best valid partner of m so m prefer w to w'. From that (m, w) is an instability in S', so S' is not stable. Contradiction. □

Lemma 3.8. *When run Gale-Shapley algorithm with men propose, there is stable matching that no man is matched to his highest-ranked woman.*

Proof. Consider the following example with man proposing,

$$P_m = \begin{bmatrix} P_{m_1} \\ P_{m_2} \\ P_{m_3} \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & w_3 \\ w_1 & w_2 & w_3 \\ w_2 & w_1 & w_3 \end{bmatrix} \quad P_w = \begin{bmatrix} P_{w_1} \\ P_{w_2} \\ P_{w_3} \end{bmatrix} = \begin{bmatrix} m_3 & m_2 & m_1 \\ m_2 & m_1 & m_3 \\ m_1 & m_2 & m_3 \end{bmatrix}$$

we end up with $(m_1, w_3), (m_2, w_2), (m_3, w_1)$, all man are rejected by their highest-ranked woman. □

It's remarkable that in the above example, the women end up with their highest-ranked partners, which means that for Lemma 3.7, the worest valid partners are not necessarily at the bottom of the preference list. They might be the same as the best valid partners, when the Stable Matching is **unique**.

4 Bipartite Matching

Independent Set: Given a graph G = (V , E), we say a set of nodes $S \subseteq V$ is independent if no two nodes in S are joined by an edge.
Bipartite: Graph $G = (V, E)$ is bipartite if its node set V can be partitioned into independent sets X and Y in such a way and every edge in E has one end in X and the other end in Y.
Connected Bipartite Graph: A bipartite graph G is connected if there is a path between every pair of vertices, regardless of the set that they are in.
Matching: A matching in a graph $G = (V, E)$ is a set of edges $M \subseteq E$ with the property that each node appears in at most one edge of M.
Perfect Matching: A matching M is perfect if every node of G appears in exactly one edge of M.
Bipartite Matching Problem: Given an arbitrary bipartite graph G, find a matching of maximum size. If |X| = |Y| = n, then there is a perfect matching if and only if the maximum matching has size n.

Theorem 4.1. *In a connected bipartite graph, the bipartite is unique.*

Proof. Suppose we have already found a bipartite X, Y where all edges in the graph go between X and Y. Now let's assume that we have another bipartition X', Y'. Then there must be at least one node i that used to be in X and now is not in X' but in Y'. We can run BFS starting from node i as the root node. In that case, all nodes at level 2 that have belonged to Y should now in X', all node that used to be in X are now in Y' and so on. In other words, all node that used to be in X are now in Y' and all nodes that used to be in Y are now in X'. So the bipartition is unique. □