Comp 260: Advanced Algorithms — Lecturer: Lenore Cowen
Tufts University, Spring 2004 — Scribe: Patrika Agarwal

Lecture 2: Marriage and Perfect Matching[1]

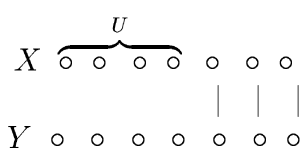
1 Perfect Matching

Theorem 1.1 *There exists a polynomial time algorithm to find a perfect matching.*

Proof: (By Construction) We will use the following algorithm to build a polynomial time algorithm for perfect matching:

The Augmenting Path Algorithm

Input: A bipartite graph G with a partition of X (*knights*) and Y (*ladies*), a matching M with k edges in G, and a set U of all unmatched vertices in X (see figure below).
Output: A matching M^* with at least one more edge than M or a proof that G violates Hall's condition [3].



The Algorithm:
Note: The method duplicates the proof of Hall's theorem.
Let $u \in U$ be any unmatched vertex in X. Grow a BFS tree from $u \in U$. Consider only unmatched edges from X to Y and only matched edges from Y to X. To reconstruct the path back, we need to remember for each $y \in Y$ (each *lady*) who is its predecessor in X. We don't need to maintain this information for X (the *knights*) because there is only one matched edge. Note that all paths will be created from alternating unmatched and matched edges.

[1]These lecture notes are adapted in part from a set of lecture notes scribed by Eynat Rafalin in 2002.

We have 2 cases:
Case 1: We find a path that starts and ends with an unmatched edge (*an augmenting path*). We can create a new matching that has one more edge i.e., output M^*, the matching that is identical to matching M except that the matched and unmatched edges are swapped along the augmenting path.
Case 2: We cannot find an augmenting path. Call S the set of vertices we reach in X and T the set of vertices we reach in Y. If we cannot find an augmenting path then $|T| = |S - u|$. We know that T is the neighborhood of S ($T = N(S)$) and because the size of u is one (one node) we get $|T| = |N(S)| = |S| - 1$. This violates Hall's condition.

Time complexity: In each step we add one more edge. We need n edges for the final matching and therefore there are n steps. In each step a BFS tree is created. Each vertex is visited once but there are at most n^2 vertices. Therefore the total time complexity is $nO(n^2) = O(n^3)$.
Bounds: In the case of perfect matching the bound of $O(n^3)$ from the BFS algorithm above was improved to $O(n^{2.5})$ (see Hopcroft and Karp [4]). But even this bound is not known to tight: there is no reason to think that by extreme cleverness we might be able to improve this further, but it would need some new ideas.

In the problem above we are looking only for a perfect matching. If we were looking for *maximum matching*, the algorithm would have to grow edges from each vertex u in U, and not only from u. The proof that this works follows from the Konig-Egerváry [1] theorem, and is omitted.

2 Stable Marriage

Definition 2.1 *A marriage (matching) is stable if and only if there is no man x and woman a such that x prefers a to his current partner and a prefers x to her current partner.*

Theorem 2.2 *For every set of preference rankings among all vertices in a complete bipartite graph, there is always a stable matching.*

Motivation: This algorithm is used to match medical school students to internships. Each student ranks the medical schools and each school ranks the students. The goal is to perform a match without creating a situation where a student and a school both prefer to match to each other rather than their assigned partners, so there is no motivation for individual pairs to thwart the system and create chaos.

Example: A set of 4 men $\{x, y, z, w\}$ and 4 women $\{a, b, c, d\}$. Each ranked the other group according to "liking".

Men:	Women:
$x : a > b > c > d$	$a : z > x > y > w$
$y : a > c > b > d$	$b : y > w > x > z$
$z : c > d > a > b$	$c : w > x > y > z$
$w : c > b > a > d$	$d : x > y > z > w$

Any matching where $x - b$ and $w - a$ is unstable because x prefers a over b and a prefers x over w:

An example of a stable matching:

$$x - a$$
$$y - b$$
$$z - d$$
$$w - c$$

Proof: The proof is by construction:

The Gale-Shapley Proposal Algorithm (1962) [2]

Input: Preference ranking by each of the n women and n men.
Iteration: In round i, each man proposes to the highest woman on his list who has not previously rejected him. If every woman receives exactly one proposal - stop and output the matching. Otherwise, at least one woman receives at least two proposals. Every woman receiving more than one proposal rejects all but the highest one on her list. Every woman receiving a proposal says *maybe* to the most attractive proposal.

Correctness of the algorithm
1. Termination: In a pass, if the algorithm does not terminate then some man strikes off a new women from his list. Hence, the algorithm must terminate in n^2 steps.
2. The marriage is stable: We need to show that the final output cannot be such that x is matched to b and y is matched to a while x prefers a to

b and a prefers x to y. If the algorithm terminated with these marriages, x never proposed to a (otherwise a would have never accepted y). But, x did propose to b, therefore x must prefer b to a. Contradiction.

Are the marriage always unique? NO
In the above example, if the men propose the resulting stable marriage (matching) is as shown above. However, if the women propose, the stable marriage is as shown below:

$$x - d$$
$$y - b$$
$$z - a$$
$$w - c$$

It can be proved that the Gale-Shapley Proposal Algorithm favors the proposers: The proposers are at least as happy as in any other stable marriage.
Bounds: It is not known how to improve the bound of $O(n^2)$ rounds in the case of a stable solution.

The Stable Roommate Problem In this problem we do not insist that the graph be bipartite. For this problem there are sets of preferences where there is no stable solution.

References

[1] E. Egerváry, *On combinatorial properties of matrices* Mat. Lapok, Vol 38(16-28), 1931.

[2] D. Gale and L. Shapley *College admissions and the stability of marriage* American Mathematical Monthly, Vol 69(9-15), 1962.

[3] P. Hall, *On representations of distinct subsets* Journal of the London Math. Society, Vol 10(26-30), 1935.

[4] J. E. Hopcroft and R. M. Karp *An* $n^{5/2}$ *algorithm for maximum matchings in bipartite graphs* SIAM Journal on Computing, Springer Verlag (Heidelberg, FRG and NewYork NY, USA)-Verlag, Vol. 2(225-231), 1973.

[5] D. König, *Graphen und Matrizen* Math. Lapok, Vol. 38(116-119), 1931.