TD: Matchings with Preferences

Algorithmique et Programmation

Pierre Senellart
pierre.senellart@ens.fr

November 17, 2016

The goal of this exercise session is to study the computation of *optimal matchings* in weighted graphs, where weights express *preferences*. We will consider several different settings and definitions of optimality, but in all settings we fix a weighted graph $G = (V, E, w)$ with V a finite set of n vertices, $E \subseteq V^2$ a set of (possibly directed) edges, and $w : E \to \mathbb{R}^+$ a weight function.

1 Stable Heterosexual Marriage

We first consider the setting where V is partitioned into two subsets of equal size, $V = X \cup Y$, such that G is bipartite complete, i.e., $(u, v) \in E \Leftrightarrow (u \in X \wedge v \in Y) \vee (u \in Y \wedge v \in X)$.

A *stable* matching M in G is a perfect matching, i.e., a one-to-one mapping from X to Y, such that:

$$\forall x \in X, \forall y \in Y,\ y = M(x) \vee w(x, M(x)) \geqslant w(x, y) \vee w(y, M^{-1}(y)) \geqslant w(y, x).$$

1a. Explain the title of this section.

1b. Propose a naïve algorithm for finding all stable matchings in a graph. What is its asymptotic complexity?

1c. First consider the case where w is a one-to-one function. The Gale–Shapley algorithm proceeds as follows:

Input: Bipartite complete weighted graph $G = (X \cup Y, E, w)$ with w one-to-one
Output: Stable matching M
begin
  $M \leftarrow \emptyset$;
  $Free_X \leftarrow X$;
  $Free_Y \leftarrow Y$;
  while $Free_X \neq \emptyset$ **do**
    $x \leftarrow \text{pop}(Free_X)$;
    for $y \in Y$ *ordered by descending* $w(x, y)$ **do**
      if $y \in Free_Y$ **then**
        $M \leftarrow M \cup \{(x, y)\}$;
        $Free_Y \leftarrow Free_Y \backslash \{y\}$;
        exit for;
      else
        if $w(y, x) > w(y, M^{-1}(y))$ **then**
          $M \leftarrow M \backslash \{(M^{-1}(y), y)\} \cup \{(x, y)\}$;
          $Free_X \leftarrow Free_X \cup \{M^{-1}(y)\}$;
          exit for;
  return M;

1cα) Show this algorithm is correct, i.e., it provides a stable matching in G.

1cβ) What is the asymptotic complexity of this algorithm? You can modify the algorithm to add extra bookkeeping if needed to lower the complexity, as long as the semantics is the same.

1cγ) Is there always a unique stable matching? Prove your affirmation.

1cδ) Show that the Gale–Shapley algorithm is sexist, and illustrate on an example why this can be an issue for computing stable marriages. (More precisely, show that the stable matching computed by the algorithm can be the best possible matching for one of the two subsets, and the worst for the other.)

1d. We no longer assume w to be one-to-one, i.e., the weights associated to two different edges can in this question be identical.

1dα) Give an algorithm that computes a stable matching and prove this algorithm is correct. What is its asymptotic complexity?

1dβ) Take X={1,3}, Y={2,4} and the following weights:

u	v	$w(u, v)$
1	2	1
1	4	2
3	2	1
3	4	1
2	1	1
4	1	1
2	3	1
4	3	1

What are all stable matchings in this graph? Is this expected?

1dγ) Propose an alternative (but simple!) definition of stable matching for which there is a single stable matching in the previous example.

1dδ) Show that, for this alternative definition, there are graphs with no stable matchings.

2 Stable Roommates (or Stable Homosexual Marriage)

In this part, n is even and G is a complete directed graph without self-edges, i.e., $E = \{(u,v) \in V^2 \mid u \neq v\}$. We assume for simplicity again that w is one-to-one.

A *stable* matching is defined again as a perfect matching (that we this time see as a partition of V into a collection M of $\frac{n}{2}$ pairs of vertices), with a similar condition as before:

$$\forall x \in V, \forall y \in V,\ \{x,y\} \in M \vee \\ (\exists x' \exists y' \{x,y'\} \in M \wedge \{x',y\} \in M \wedge (w(x,y') \geqslant w(x,y) \vee w(y,x') \geqslant w(y,x))).$$

Show that there is a graph of 4 vertices without any stable matching.

3 Maximum-Weight Heterosexual Marriage

We consider again the setting where V is partitioned into two subsets of equal size, $V = X \cup Y$. This time, G is bipartite, but contains only edges from X to Y (and all such edges): $(u,v) \in E \Leftrightarrow u \in X \wedge v \in Y$. We are not interested in stable matchings, but in *maximum-weight* (respectively, minimum-weight) perfect matchings, i.e., perfect matchings such that $\sum_{x \in X} w(x, M(x))$ is maximum (respectively, minimum).

3a. Show that maximum-weight matching reduces to minimum-weight matching in time $O(n^2)$ (i.e., a solution to the maximum-weight matching problem can be obtained by solving a minimum-weight matching problem with $O(n^2)$ time overhead). Note that weights must always be positive.

3b. Show Kőnig's theorem: in a bipartite graph, the number of edges in a maximum matching is the same as the number of vertices in a minimum vertex cover.

3c. We now present a simplified form of the Hungarian algorithm to find a minimum-weight matching.

We first do the following initial processing:

- Write $X = \{x_1 \dots x_{n/2}\}$ and $Y = \{y_1 \dots y_{n/2}\}$.
- Construct an $(n/2) \times (n/2)$ matrix A such that $A(i,j) = w(x_i, y_j)$
- For every row i, we set for every column j: $A(i,j) \leftarrow A(i,j) - \min_j A(i,j)$
- For every column j, we set for every row i: $A(i,j) \leftarrow A(i,j) - \min_i A(i,j)$

3cα) What is the asymptotic complexity of this initial processing?

3cβ) Show that the resulting matrix has at least one zero on each line and one zero on each column.

3cγ) Show that if the resulting matrix G' has $n/2$ zeroes that have distinct columns and distinct rows from one another, then one can immediately obtain from these zeroes a minimum-weight matching in G.

3d. If this property is not verified, we continue the algorithm by running the following succession of steps until the property is verified:

- Find the minimum number of rows and columns to mark so that every zero in the matrix is marked.
- Let μ be the minimum value in the matrix that is neither in a marked row nor in a marked column.
- Subtract μ from every unmarked row, and add μ to every marked column.

3dα) Show that finding the minimum number of rows and columns to mark is equivalent to finding the maximum number of zeros that have distinct columns and rows from one another.

3dβ) Show that the procedure converges, and that the algorithm is correct.

3e. What is the overall asymptotic complexity of this algorithm? A refinement of the algorithm presented here can actually be used to lower the complexity to $O(n^3)$.