Maximum Matchings

Given a graph $G = (V, E)$ a subset of the edges $M \subseteq E$ is a *matching*, if no two edges in M have a common endpoint. The *maximum matching problem* aims at finding a matching $M \subseteq E$ of maximum cardinality in a given graph $G = (V, E)$. In the *maximum weight matching problem* we have a graph $G = (V, E)$ and a weight function $w : E \longrightarrow \mathbb{R}_+$ as input, and want to find a matching $M \subseteq E$ for which $w(M) = \sum_{e \in M} w(e)$ is as large as possible.

For a standard IP formulation, consider binary variables x_e associated to the edges $e \in E$ of the graph, indicating membership in a matching

$$x_e = \begin{cases} 1, & \text{if } e \in M \\ 0, & \text{otherwise.} \end{cases}$$

Then we can formulate the maximum (cardinality) matching problem as

$$\begin{aligned} \nu(G) = \quad \max \quad & \sum_{e \in E} x_e \\ s.t. \quad & \\ & \sum_{e \ni v} x_e \leq 1 \qquad \forall v \in V \\ & x_e \in \{0, 1\} \quad \forall e \in E. \end{aligned} \tag{IP-M}$$

The linear programming relaxation of this problem then looks like

$$\begin{aligned} \nu^*(G) = \quad \max \quad & \sum_{e \in E} x_e \\ s.t. \quad & \\ & \sum_{e \ni v} x_e \leq 1 \quad \forall v \in V \\ {\color{red}1 \geq} \; & x_e \geq 0 \quad \forall e \in E. \end{aligned} \tag{LP-M}$$

We denoted by $\nu(G)$ the size of a maximum matching of G, and by $\nu^*(G)$ the optimum of the LP-relaxation of the matching problem, also called the *fractional matching* number of G. Note that in the LP-relaxation the red inequalities are not necessary to claim – they follow form the previous block of inequalities.

Let us also consider for a moment a "dual" problem: A subset of the vertices $C \subseteq V$ is called a *vertex cover* if $C \cap e \neq \emptyset$ for all edges $e \in E$ (or in other words, if $V \setminus C$ is a stable set). Let us denote by $\tau(G)$ the size of a smallest vertex cover of G.

For a standard IP formulation of this problem, consider binary variables y_v associated to the vertices $v \in V$:

$$y_v = \begin{cases} 1, & \text{if } v \in C \\ 0, & \text{otherwise.} \end{cases}$$

Then we can formulate the minimum (cardinality) vertex cover problem as

$$\begin{aligned} \tau(G) = \quad \min \quad & \sum_{v \in V} y_v \\ s.t. \quad & \\ y_u + y_v \quad & \geq 1 \qquad \forall e = (u,v) \in E \\ y_v \quad & \in \{0,1\} \quad \forall v \in V. \end{aligned} \tag{IP-VC}$$

The linear programming relaxation of this problem then looks like

$$\begin{aligned} \tau^*(G) = \quad \min \quad & \sum_{v \in V} y_v \\ s.t. \quad & \\ y_u + y_v \quad & \geq 1 \quad \forall e = (u,v) \in E \\ 1 \geq \quad y_v \quad & \geq 0 \quad \forall v \in V. \end{aligned} \tag{LP-VC}$$

We denote by $\tau(G)$ the *vertex cover number* of a graph G, and by $\tau^*(G)$ the optimum of the LP-relaxation, also called the *fractional vertex cover number* of G. Note again that the red inequalities are redundant, and not necessary to include.

Let us observe that (LP-M) and (LP-VC) are dual (if we delete the red inequalities). Thus, we obtain the following relations:

$$\nu(G) \leq \nu^*(G) = \tau^*(G) \leq \tau(G).$$

Given a graph $G = (V, E)$ and a subset $S \subseteq V$ of its vertices, we denote by $N(S)$ the *neighborhood* of S, i.e. $N(S) = \{v \in V \mid \exists u \in S,\ (u,v) \in E\}$. We shall simply write $N(v)$ if $S = \{v\}$.

Given a matching $M \subseteq E$, let us denote by $V(M)$ the set of vertices incident to (covered by) some edge of M. The matching M is called *maximal* if $V \setminus V(M)$ is a stable set; it is called *perfect* if $V = V(M)$ (clearly, $|V|$ must be even in this case).

The above definitions readily imply that

$$\nu(G) = \max_{\substack{M \text{ is a} \\ \text{maximal matching}}} |M| \leq \tau(G) \leq 2 \min_{\substack{M \text{ is a} \\ \text{maximal matching}}} |M| \leq 2\nu(G).$$

We shall see that computing $\nu(G)$ is a polynomially solvable task, while computing $\tau(G)$ is NP-hard, in general. The above inequalities imply a linear time 2-approximation for $\tau(G)$. [HOW??]

Given a graph $G = (V, E)$ and a matching $M \subseteq E$, a simple path $P \subseteq E$ (as a set of edges) is called *M-alternating* if $P \setminus M$ is also a matching. An M-alternating path is called *M-augmenting* if the first and last vertices of P are not incident to edges of M. The most basic and still simple observation, which is behind most cited results, is the following:

Theorem 1 (Petersen (1891), Berge (1957), Norman & Rabin (1959)) *Given a graph $G = (V, E)$, a matching $M \subseteq E$ is maximum if and only if there are no M-augmenting paths in G.*

Proof. If M is a matching and P is an M-augmenting path, then $M' = M \triangle P = (M \setminus P) \cup (P \setminus M)$ is also a matching and $|M'| > |M|$. Conversely, if M' and M are matchings, $|M'| > |M|$, then $M \cup M'$ must contain an M-augmenting path. $\square$

Maximum Sized Matchings in Bipartite Graphs

The following simple lemma will be very useful in many of the following proofs.

Lemma 1 *Let $G = (A \cup B, E)$ be a bipartite graph, with vertices $A \cup B$, $A \cap B = \emptyset$, and edges $E \subseteq A \times B$. Let us further consider a maximum matching M of G, and let us consider all alternating paths starting at some vertex in $W = A \setminus V(M)$. Denote by $V \subseteq A \cap V(M)$ and $Y \subseteq B \cap V(M)$ be the set of vertices that can be reached by such an M-alternating path starting* *in W. Let us further denote by $U \subseteq A \cap V(M)$ and $X \subseteq B \cap V(M)$ the sets of vertices of $V(M)$ that cannot be reached by such M-alternating paths, and finally set $Z = B \setminus V(M)$. Then we have no edges in G between the sets $V \cup W$ and $X \cup Z$. In other words $V \cup W \cup X \cup Z$ is a stable set, and $U \cup Y$ is a vertex cover of G.*

Proof. WHY? See in class. $\square$

Maximum matchings in bipartite graphs

One of the most well-known, fundamental result establishes that in bipartite graphs both matching and covering are tractable problems.

Theorem 2 (König (1931), Egerváry (1931)) *If $G = (V(G), E(G))$ is bipartite, then $\nu(G) = \tau(G)$.*

Proof. Clearly, $\nu(G) \leq \tau(G)$ holds in an arbitrary graph G. To see the equality, let us denote by A and B the two classes of vertices of G, i.e., $V(G) = A \cup B$ and $E(G) \subseteq A \times B$. Let further M denote a maximum matching, i.e., $\nu(G) = |M|$. Then, by Lemma 1 we have that $C = U \cup Y$ is a vertex cover, and $|U| = |X|$ and $|X| + |Y| = |M|$ from which the claim follows. (Note that $\nu(G) \leq \tau(G)$ for every graph, thus if we have equality for a matching and vertex cover pair, then that matching must be of maximum size and that vertex cover must be of minimum size. $\square$

The next statement is also known as the *Marriage theorem*:

Theorem 3 (Frobenius (1917), Hall (1935)) *Given a bipartite graph $G = (V(G), E(G))$, $V(G) = A \cup B$, $A \cap B = \emptyset$, $E(G) \subseteq A \times B$. Then, we have $\nu(G) = |A|$ if and only if $|A| \leq |B|$ and*

$$|S| \leq |N(S)| \quad \textit{for all} \quad S \subseteq A.$$

In particular, G has a perfect matching if and only if $|A| = |B|$ and the above inequalities hold.

Proof. Clearly, if $\nu(G) = |A|$, then the above conditions must hold. To see the reverse, it is clear that $|A| \leq |B|$ is necessary, so let us assume this, and let us consider a maximum matching M, and assume indirectly that $|M| < |A| \leq |B|$. Let us then again consider the sets as in Lemma 1, and consider the set $S = V \cup W \subseteq A$. Then, by Lemma 1 we have $N(S) = Y$, with $|V| = |Y|$, and with $W \neq \emptyset$ by our indirect assumption, proving that $|N(S)| = |Y| < |V \cup W| = |S|$. $\square$

There are several interesting consequences of the above (equivalent) theorems, some of which are also equivalent with the above statements.

Given a real matrix $A \in \mathbb{R}^{m \times n}$, a set of nonzeros in it is called *independent* if no two share a row or column (called *lines*).

Corollary 1 (Egerváry (1931)) *The maximum number of independent nonzeros in a given real matrix A is equal to the minimum number of lines (rows and columns) containing all nonzero elements of A.*

Proof. This is just a direct reformulation of Theorem 2. $\square$

A finite *poset* (more precisely, a ***p**artially **o**rdered **set***) is an irreflexive, transitive relation $<$ on a finite set P. Two distinct elements $a, b \in P$, $a \neq b$ are called *comparable* if either $a < b$ or $b < a$. A subset $C \subseteq P$ in which every pair of elements are comparable is called a *chain* of P. A subset $A \subseteq P$ in which no two elements are comparable is called an *anti chain* of P.

Corollary 2 (Dilworth (1950)) *The minimum number of chains needed to cover all elements of P is the same as the size of a maximum cardinality anti chain in P.*

Proof. (By **Fulkerson (1956)**.) Take two copies of each element of P, say e.g., $A = \{s' \mid s \in P\}$ and $B = \{s'' \mid s \in P\}$, and consider the bipartite graph $G = (V, E)$ on vertices $V = A \cup B$, in which $(a', b'') \in E$ whenever $a < b$ in P. Let $M \subseteq E$ be a maximum matching and $C \subseteq V$ be a minimum vertex cover in G. By Theorem 2 we know that $|M| = |C|$ and exactly one endpoint of each edge in M belongs to C.

Consider the pairs $(a, b) \in P \times P$ corresponding to edges $(a', b'') \in M$. Since at most one matching edge is incident with $a' \in A$ and $a'' \in B$ for every $a \in P$, these (a,b) pairs naturally form disjoint chains in P. The remaining singletons, not covered by these chains form an anti chain. Let us consider each of these as chains of length 1, obtaining in this way a chain decomposition of P.

If $L = \{a_1, a_2, ..., a_k\}$ is a chain in this decomposition then, we claim that, there exists a unique index $1 \leq i \leq k$ for which $\{a'_i, a''_i\} \cap C = \emptyset$. (WHY??!!) Denoting this unique element by $a(L) = a_i$, let us consider the set $T = \{a(L) \mid L \text{ is a chain of the chain decomposition of } P \text{ obtained above}\}$. Then T is an anti chain. (WHY??!!) Thus, we have an anti chain and a chain decomposition which are of the same size. $\square$

A real matrix $B \in \mathbb{R}_+^{m \times n}$ is called *bi-stochastic* if the sum of its elements in every row and column equals to 1. (Clearly, $m = n$ must hold, right?) A binary matrix $P \in \{0,1\}^{n \times n}$ is called a *permutation matrix* if it has in each row and in each column exactly one nonzero elements. Clearly, permutation matrices are bi-stochastic.

Corollary 3 (Birkhoff (1946), von Neumann (1953)) *Every bi-stochastic matrix is the convex combination of permutation matrices. Or, in other words, the set of $n \times n$ bi-stochastic matrices (as subset of $\mathbb{R}_+^{n^2}$) forms a bounded, convex polyhedron, the vertices of which correspond, in a one-to-one way, to the permutation matrices.*

Proof. Let us prove the statement by induction on the number of nonzeros. Clearly, any $n \times n$ bi-stochastic matrix must have at least n nonzeros, and if it has exactly n, then it must be a permutation matrix.

Let us assume that we already proved the statement for bi-stochastic matrices having at most $k \geq n$ nonzeros, and let $B = (b_{ij}) \in \mathbb{R}_+^{n \times n}$ be a bi-stochastic matrix with $k+1$ nonzero elements. We claim that the minimum number of lines (rows and columns) needed to cover all nonzeros of B is n. This is because denoting by I and J the sets of rows and columns in such a minimum cover, we can write

$$n = \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} \leq \sum_{i \in I} \sum_{j=1}^{n} b_{ij} + \sum_{j \in J} \sum_{i=1}^{n} b_{ij} = |I| + |J|,$$

and clearly, we do not need more than n lines in a minimum cover. Thus, by Corollary 1 we can conclude that B has n independent nonzeros $b_{i,\pi(i)} > 0$,

$i = 1, ..., n$ for some permutation π of the indices $\{1, 2, ..., n\}$. Let P be the permutation matrix corresponding to this permutation, and let $\alpha = \min_{1 \le i \le n} b_{i,\pi(i)} > 0$. Then, we can write

$$B = \alpha P + (1 - \alpha) B'$$

for some real matrix B'. It is easy to see that B' is again bi-stochastic, and has one less nonzeros than B does. $\square$

Corollary 4 (König (1935)) *Any k-regular bipartite graph has a perfect matching, and in fact, its edge set can be decomposed into k pairwise disjoint perfect matchings.*

Proof. Direct consequence of Corollary 3. $\square$

Homework Exercise: Let $G = (A \cup B, E)$ be a bipartite graph, $A \cap B = \emptyset$, $E \subseteq A \times B$, and let $d_G(v)$ denote the degree of vertex v in G. Let us further assume that the following conditions hold:

$$\sum_{v \in N(u)} \frac{1}{d_G(v)} = 1 \quad \text{for all} \quad u \in B.$$

Prove that G has a perfect matching.
Hint: Do not try to show that only the regular bipartite graphs satisfy the above conditions. This is not a consequence of Corollary 4!

Algorithms to find a maximum matching in bipartite graphs

Theorem 1 implies the very simple idea, of using augmenting paths to increase the size of a matching, successively, until no augmenting paths are found. This idea can easily be realized in case of bipartite graphs, as shown by the following algorithm, proposed by H. Kuhn in 1955 (he called it the *Hungarian Method* in honor of the contributions to this field by D. König and J. Egerváry, whose results and proofs are behind the correctness of this approach).

We consider a bipartite graph $G = (A \cup B, E)$, $A \cap B = \emptyset$, $E \subseteq A \times B$, and a matching M found in some way (e.g., in the previous steps; initially we can construct a maximal (not necessarily of maximum size) matching, by scanning through the edges once (in an arbitrary order). We shall describe the *augmentation* step of the Hungarian method, which consists in repeating this step until no augmenting path is found (at most $|V|/2$-times).

For a vertex $v \in V(M)$ let us denote by $u = \mu_M(v)$ the unique vertex for which $(u, v) \in M$, and let $\mu_M(S) = \{\mu_M(v) \mid v \in S\}$ for a subset $S \subseteq V(M)$.

The Augmentation Step of the Hungarian Method (unweighted case):

Initialization: Let $X_0 = A \setminus V(M)$, $Y_0 = \emptyset$, and $T = B \setminus V(M)$. If $X_0 = \emptyset$, then STOP (M is maximum), otherwise set $k = 0$, and continue.

Main Loop: Set $k = k + 1$, $Y_k = N(X_{k-1}) \setminus (Y_1 \cup \cdots \cup Y_{k-1})$.

- If $Y_k = \emptyset$ then STOP: "NO AUGMENTING PATH IS FOUND".
- If $Y_k \cap T \neq \emptyset$ then STOP: "AUGMENTING PATH IS FOUND"; construct the augmenting path P and output $M = P \Delta M$.
- Otherwise set $X_k = \mu_M(Y_k)$, and repeat the **Main Loop**.

Analysis of complexity: The **Initialization** is executed only once, and clearly can be implemented to run in $O(|V|)$ time.

For being able to retrace the augmenting path, when found, we need to use a labelling procedure, similar to that in shortest path algorithms. In the kth step, Y_k can be computed in $O(\sum_{u \in X_{k-1}} d_G(u))$ time, and all other substeps can be computed in $O(|Y_k|)$ time. Since the sets X_k and Y_k, $k = 0, 1, ...$ are pairwise disjoint, nonempty before termination, and $\bigcup_k Y_k \subseteq V(M)$, the **Main Loop** is not executed more than $|M|$ times. Thus, in total we need $O(|V|) + \sum_{k=1}^{|M|}[O(\sum_{u \in X_{k-1}} d_G(u)) + O(|Y_k|)] = O(|V| + |E|)$ time for completion. $\square$

Proof of correctness: If we stop in the kth step ($k \leq |M|$ as we argued above) with "AUGMENTING PATH IS FOUND", then we have a vertex $v_k \in Y_k \cap T$ (i.e., $v_k \in B \setminus V(M)$), who is a neighbor of a vertex $u_{k-1} \in X_{k-1}$. By the definition, $u_{k-1} = \mu_G(v_{k-1})$ for a vertex $v_{k-1} \in Y_{k-1}$. Repeating this argument, we get a sequence of consecutively neighboring vertices, until we arrive to $u_0 \in X_0$. The path P formed by the edges $(u_0, v_1), (v_1, u_1), ..., (v_{k-1}, u_{k-1}), (u_{k-1}, v_k)$ forms an M-alternating path, and both of its endpoints are outside of $V(M)$. Thus, it is an augmenting path.

If we stop in the kth step with "NO AUGMENTING PATH IS FOUND", then let us consider the sets $D_A = X_0 \cup X_1 \cup \cdots \cup X_k$, $C_A = A \setminus D_A$, and $C_B = Y_1 \cup \cdots \cup Y_k$. Since we did not stop earlier, we have $C_A \subseteq V(M)$, $C_B \subseteq V(M)$, and $\mu_M(C_B) \subseteq D_A$. Thus, we have that the set $C = C_A \cup C_B$ contains exactly one point from each edge of M, i.e. $|C| = |M|$. Furthermore, we have $N(D_A) \subseteq C_B$ (here comes Merlin's luck in the King Arthur story!!), since otherwise the procedure would continue. Hence, C is a vertex cover, and since $|M| = |C|$, it is a minimum size vertex cover, and thus M is a maximum matching. □

Clearly, starting with an arbitrary matching (e.g., containing only one edge) and repeating the above augmentation at most $|V|/2$ times, we obtain a maximum matching in $O(|V||E|)$ total time.

Perfect matching in k-regular bipartite graphs

Let us consider a k-regular bipartite graph $G = (A \cup B, E)$, and apply the following algorithm:

Algorithm 1 Perfect Matching in Regular Bipartite Graphs

```
procedure SCHRIJVER(G = (A ∪ B, E))
    Set w(e) = 1 for all e ∈ E.
    while There exists a cycle C ⊆ E with w(e) > 0 for all e ∈ C do
        Decompose C = M ∪ N into two matchings (w(M) ≥ w(N).)
        Set w := w + χ(M) − χ(N).
    end while
    return M = {e ∈ E | w(e) > 0}.
end procedure
```

Theorem 4 (Schrijver (1988)) *For a k-regular bipartite graph the above algorithm terminates with a perfect matching in $O(km)$ time.*

Proof. Note first that $w(\delta(u)) = \sum_{(u,v)\in E} w(u,v) = k$ at all time during this algorithm (WHY??). Thus, as long as we have an edge with $0 < w(e) < k$, there is a cycle C in the subgraph $\{e \in E \mid w(e) > 0\}$ (WHY??). This proves that the algorithm outputs a perfect matching, if it stops.

Note next that $Q = \sum_{e\in E} w^2(e)$ changes in each main iteration by

$$\sum_{e\in M} \big((w(e)+1)^2 - w^2(e)\big) \;+\; \sum_{e\in N} \big((w(e)-1)^2 - w^2(e)\big)$$
$$= \; 2w(M) + |M| - 2w(N) + |N| \;\geq\; |M| + |N| = |C|.$$

Since at termination we have $Q = \frac{1}{2}nk^2$, and since finding a cycle C takes $O(|C|)$ time on average (WHY??) the algorithm terminates in $O(km) = O(k^2 n)$ time. $\square$

Algorithm 2 Perfect Matching in Regular Bipartite Graphs

procedure ALON(k-regular bipartite graph $G = (A \cup B, E)$)
  Set $2^t \geq kn$, $\alpha = \left\lfloor \frac{2^t}{k} \right\rfloor$, and $\beta = 2^t - \alpha k$.
  Replace each edge of G by α parallel copies.
  Add a perfect matching F to G in which each edge is replaced by β parallel copies.
  Call the resulting 2^t-regular bipartite graph H.
  while $\deg_H(v) > 1$ for some/all vertices of H **do**
    Find an Eulerian orientation of the edges of H.
    If $Q = A \to B$ edges have more F-edges, then delete Q, otherwise delete all $B \to A$ edges from H.
  end while
  return $M = E(H)$.
end procedure

Theorem 5 (Alon (2000)) *The above algorithm terminates in t iterations with a perfect matching M. The total computing time is $O(mt) = O(m \log m)$.*

Proof. See in class. $\square$

Weighted bipartite matching and assignment

Integrality of the bipartite matching polytope

Given a bipartite graph $G = (V, E)$ let

$$P_G = \left\{ x \in \mathbb{R}_+^E \;\middle|\; \sum_{v \in N(u)} x(u,v) \leq 1 \text{ for all } u \in V \right\}.$$

P_G is called the matching polytope of G.

Theorem 6 (Hoffman and Kruskal (1956), Heller and Tompkins (1956)) *P_G has integral vertices.*

Proof. Fact 1: $x \in P_G$ is a vertex of P_G iff there are no $y, z \in P_G$ such that $x = \frac{1}{2}y + \frac{1}{2}z$.

Fact 2: Assume $x \in P_G$ is a vertex, and define $F = \{(u,v) \in E \mid 0 < x(u,v) < 1\}$. Then, either $F = \emptyset$ or it contains a cycle $C \subseteq F \subseteq E$.

If $F = \emptyset$ then x is integral, and we are done. Otherwise $F \neq \emptyset$, and we have a cycle $C \subseteq F$. Then, we must have $|C|$ even (since G is bipartite), and thus changing the x-values alternating by $\pm\epsilon$ along this cycle we can derive a contradiction with Fact 1. More precisely, let M_1 and M_2 be the two matchings for which $C = M_1 \cup M_2$, and define for all edges $(u,v) \in E$

$$y(u,v) = \begin{cases} x(u,v) + \epsilon, & \text{if } (u,v) \in M_1 \\ x(u,v) - \epsilon, & \text{if } (u,v) \in M_2 \\ x(u,v), & \text{otherwise,} \end{cases} \quad \text{and} \quad z(u,v) = \begin{cases} x(u,v) - \epsilon, & \text{if } (u,v) \in M_1 \\ x(u,v) + \epsilon, & \text{if } (u,v) \in M_2 \\ x(u,v), & \text{otherwise.} \end{cases}$$

Now, if $\epsilon > 0$ is small enough, then $y, z \in P_G$, and this is a contradiction by Fact 1, since we have $x = \frac{1}{2}y + \frac{1}{2}z$ by our construction. □

Theorem 7 (Egerváry (1931)) *Let $G = (V, E)$ be a bipartite graph and $w : E \to \mathbb{R}_+$ be nonnegative edge weights. Then the maximum weight of a matching in G is equal to the minimum value of $y(V) = \sum_{v \in V} y(v)$, where $y : V \to \mathbb{R}_+$ satisfies*

$$y(u) + y(v) \;\geq\; w(e) \quad \textit{for each edge} \quad (u,v) \in E.$$

Proof. Linear programming duality in the special case of matchings and vertex covers in bipartite graphs – two decades earlier than general LP-duality was discovered! □

The Hungarian Method

Algorithm 3 Weighted Bipartite Matching (following Kuhn (1955))

---

procedure HUNGARIAN METHOD(Bipartite graph $G = (A \cup B, E)$ and edge weights $w : E \to \mathbb{R}_+$.)
  Initialize $M = \emptyset$.
  Set $P = \{e\}$, where $e \in E$ is the heaviest edge (by w.)
  repeat
    Update $M = M \Delta P$.
    Orient edges of M from B to A and assign length $\ell(e) = w(e)$.
    Orient other edges from A to B and assign length $\ell(e) = -w(e)$.
    Let $X \subseteq A$ and $Y \subseteq B$ be the vertices not covered by M.
    Find shortest $X \to Y$ path P.
  until $\ell(P) < \infty$
  return M.
end procedure

Lemma 2 *In iteration k the matching M is a w-heaviest matching of k edges in G.*

Proof. Induction by $|M|$. See in class. $\square$

Theorem 8 *The Hungarian Method returns a matching of maximum weight, and can be implemented to run on $O(n(m + n \log n))$ time.*

Proof. Note that the edge weights are redefined in every iteration. A key observation is that if M is a w-heaviest matching of $k = |M|$ edges, then the weighted graph in iteration k doe snot have negative cycles. Thus, Dijkstra's algorithm will correctly return a shortest path. See/hear the rest in class. $\square$