Introduction to Modern Combinatorial Optimization **POSTECH, Winter 2025**
Lecture #3: linear programming for bipartite matching January 14, 2025
Lecturer: Dabeen Lee

Outline

In this lecture, we consider the weighted version of the bipartite matching problem, for which we explain the linear programming-based approach.

1 Linear programming formulation for bipartite matching

In Lecture 2, we learned the augmenting path algorithm for computing a maximum matching in a bipartite graph. Here, the objective is to maximize the number of edges in a matching, for which we treat all edges equally. In some applications, however, there can be disparities between the importance of edges. We can model such scenarios by setting some arbitrary weights on the edges. To elaborate, for a graph $G = (V, E)$, let us assume that each edge $e \in E$ has a weight $w_e \in \mathbb{R}$. With the edge weights, an alternate objective is to find a matching M that maximizes the total weight sum of its edges, given by

$$w(M) = \sum_{e \in M} w_e.$$

This problem is referred to as the **maximum weight bipartite matching** problem. Although one may extend the augmenting path algorithm to the weighted case, let us discuss another method that has a slightly different flavor.

The method we cover in this section is based on **linear programming**. The idea is as follows. For each edge $e \in E$, let us introduce a **variable** x_e to indicate whether e is picked for our matching M or not, i.e.,

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

In this case, $x \in \{0,1\}^{|E|}$ is the **incidence vector**, or the **characteristic vector**, of matching M given by

$$M = \{e \in E : x_e = 1\}.$$

Then we have

$$\sum_{e \in M} w_e = \sum_{e \in E} w_e x_e.$$

where $w \in \mathbb{R}^{|E|}$ consists of the edge weights. Note, however, that any subset $S \subseteq E$, not necessarily a matching, can have its incidence vector in $\{0,1\}^{|E|}$. Hence, the next question is about how to guarantee that x corresponds to some matching. What we know is that a vertex u is incident to at most one edge of a matching. We may impose the condition by setting

$$\sum_{v \in V : uv \in E} x_{uv} \le 1 \tag{3.1}$$

where the sum is taken over the neighbors of u.

Lemma 3.1. *Let $G = (V, E)$ be a graph, not necessarily bipartite, and let $x \in \{0,1\}^{|E|}$. Then x satisfies (3.1) for all $u \in V$ if and only if x is the incidence vector of some matching M of G.*

Proof. Let M be given by $M = \{e \in E : x_e = 1\}$. Note that M is a matching if and only if each vertex u is incident to at most one edge in M, which is equivalent to (3.1). $\square$

We consider the following **optimization problem** to compute a maximum weight matching:

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E} w_e x_e \\ \text{subject to} \quad & \sum_{v \in V : uv \in E} x_{uv} \le 1 \quad \text{for all } u \in V, \\ & x_e \in \{0,1\} \quad \text{for all } e \in E. \end{aligned} \tag{3.2}$$

The meaning of this formulation is to select a vector x that achieves the maximum value of $\sum_{e \in E} w_e x_e$ among the vectors satisfying (3.1) for all $u \in V$ and $x \in \{0,1\}^{|E|}$. Here, $\sum_{e \in E} w_e x_e$ is an **objective**, and the condition (3.1) and $x_e \in \{0,1\}$ are called **constraints**. In general, an optimization problem has three components: variables, an objective, and constraints. The objective and constraints are written with the variables, and the goal is to determine values for the variables that satisfy the constraints and maximize (or minimize) the objective. For (3.2), constraint (3.1) is referred to as the **degree constraint** and constraint $x \in \{0,1\}^{|E|}$ is called the **binary constraint**.

Proposition 3.2. *Let $G = (V, E)$ be a graph, not necessarily bipartite, and let $w \in \mathbb{R}^{|E|}$. Then solving the optimization problem (3.2) computes a maximum weight matching in G.*

Proof. By Lemma 3.1, we can encode any matching by a vector x satisfying the degree and binary constraints of (3.2). For such a vector x, the objective represents the weight of the corresponding matching. $\square$

In (3.2), both the **objective** $\sum_{e \in E} w_e x_e$ and the constraint $\sum_{v \in V : uv \in E} x_{uv}$ are **linear functions** in x. Here, a linear function in x is a function of the form

$$c^\top x = \sum_{e \in E} c_e x_e$$

for some $c \in \mathbb{R}^{|E|}$. An optimization problem whose objective and constraints are given by linear functions is called a **linear program**. However, the binary constraint in (3.2) generates discontinuity and thus cannot be represented by a linear function[1]. Nevertheless, an optimization problem whose objective and constraints except for the binary constraint on its variables are given by linear functions is called a **binary linear program**. In general, a binary linear program is an **integer linear program** which in general can take any integer-valued variables. It is known that integer linear programming and binary linear programming are NP-hard, while linear programming admits a **polynomial time** algorithm such as the **ellipsoid method** and the **interior-point algorithm**. Hence, a common approach to tackle an integer linear program is to obtain its **linear programming (LP) relaxation**. The LP relaxation relaxes and removes the constraints to impose that the variables have integer values, and as a result, it becomes a linear program.

The LP relaxation of (3.2) is given by

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E} w_e x_e \\ \text{subject to} \quad & \sum_{v \in V : uv \in E} x_{uv} \le 1 \quad \text{for all } u \in V, \\ & x_e \ge 0 \quad \text{for all } e \in E \end{aligned} \tag{3.3}$$

[1] To be more precise, the set of vectors satisfying the binary constraint is not **linearly representable**.

where $x_e \geq 0$ replaces the binary constraint $x_e \in \{0,1\}$. (3.3) is indeed a linear program as its constraints are represented by linear functions. Note that any x satisfying the constraints of (3.3) would have $x_e \leq 1$ for all $e \in E$, because x_e appears in the degree constraint of an endpoint of e.

Lemma 3.3. *Let $G = (V, E)$ be a graph, not necessarily bipartite, and let $w \in \mathbb{R}^{|E|}$. Then the optimal value of the LP relaxation (3.3) is greater than or equal to the optimal value of (3.2).*

Proof. As (3.3) is a relaxation of (3.2), any x satisfying the constraints of (3.2) also satisfies the constraints of (3.3). Hence, (3.3) can achieve whatever value a solution to (3.2) can attain. □

By Lemma 3.3, the linear program (3.3) provides an upper bound on the maximum size of a matching in any graph that is not necessarily bipartite. However, the issue is that an optimal solution x^* to (3.3) can have fractional parts in $(0,1)$ in which case x^* does not correspond to a matching. Nevertheless, we can prove that bipartite graphs do not have such an issue.

Theorem 3.4. *Let $G = (V, E)$ be a bipartite graph, and let $w \in \mathbb{R}^{|E|}$. Then the LP relaxation (3.3) has an optimal solution x^* that satisfies $x^*_e \in \{0,1\}$ for all $e \in E$. Moreover, one can find a maximum matching in G by solving the linear program (3.3).*

Proof. Let $\bar{x} \in [0,1]^{|E|}$ be an optimal solution to (3.3). Consider $S = \{e \in E : \bar{x}_e > 0\}$ and the subgraph H of G obtained by deleting the edges that are not in S.

Breaking cycles Suppose that H contains a cycle C. Since H is bipartite, C is an even cycle. Let $e_1, \ldots, e_{2k}$ for some $k \geq 1$ be the edges of C where e_{i-1} and e_i are consecutive for $i = 1, \ldots, 2k$ and $e_0 = e_{2k}$. We take $\delta > 0$ as

$$\delta = \min\{\bar{x}_e : \ e \in \{e_1, \ldots, e_{2k}\}\}.$$

Take two consecutive edges e_{i-1} and e_i for some $i \in \{1, \ldots, 2k\}$. Then we have $\bar{x}_{e_{i-1}} + \bar{x}_{e_i} \leq 1$ as e_{i-1} and e_i share a common vertex. This means that $\bar{x}_{e_i} \leq 1 - \bar{x}_{e_{i-1}}$. Since $\bar{x}_{e_{i-1}} \geq \delta$, we have $\bar{x}_{e_i} \leq 1 - \delta$ for any i, and therefore,

$$\max\{\bar{x}_e : \ e \in \{e_1, \ldots, e_{2k}\}\} \leq 1 - \delta.$$

Based on this, we construct two feasible solutions to (3.3) from $\bar{x}$. First, $\tilde{x}$ is obtained by updating the coordinates of $\bar{x}$ corresponding to the edges of C as

$$\tilde{x}_{e_1} = \bar{x}_{e_1} + \delta, \quad \tilde{x}_{e_2} = \bar{x}_{e_2} - \delta, \quad \ldots, \quad \tilde{x}_{e_{2k-1}} = \bar{x}_{e_{2k-1}} + \delta, \quad \tilde{x}_{e_{2k}} = \bar{x}_{e_{2k}} - \delta.$$

The other coordinates of $\tilde{x}$ remain the same as $\bar{x}$. Similarly, we obtain $\hat{x}$ by setting

$$\hat{x}_{e_1} = \bar{x}_{e_1} - \delta, \quad \hat{x}_{e_2} = \bar{x}_{e_2} + \delta, \quad \ldots, \quad \hat{x}_{e_{2k-1}} = \bar{x}_{e_{2k-1}} - \delta, \quad \hat{x}_{e_{2k}} = \bar{x}_{e_{2k}} + \delta.$$

Let us argue that both $\tilde{x}$ and $\hat{x}$ satisfy the constraints of (3.3). Since $\bar{x}_{e_i} \in [\delta, 1-\delta]$ for $i = 1, \ldots, 2k$, we have $\tilde{x}_{e_i}, \hat{x}_{e_i} \in [0,1]$ for $i = 1, \ldots, 2k$. For any edge e not on the cycle C, we have $\tilde{x}_e = \hat{x}_e = \bar{x}_e \in [0,1]$. Moreover, by our construction of alternatively adding and subtracting the same value of δ on the cycle, we have

$$\sum_{v:uv\in E} \bar{x}_{uv} = \sum_{v:uv\in E} \tilde{x}_{uv} = \sum_{v:uv\in E} \hat{x}_{uv}$$

for any $u \in V$. As $\bar{x}$ satisfies the degree constraint (3.1), so do $\tilde{x}$ and $\hat{x}$. Then, since $\bar{x}$ is an optimal solution to (3.3), we have $w^\top \tilde{x} \leq w^\top \bar{x}$ and $w^\top \hat{x} \leq w^\top \bar{x}$. On the other hand, by our choice of $\tilde{x}$ and $\hat{x}$, we have $\tilde{x} + \hat{x} = 2\bar{x}$, in which case $w^\top \tilde{x} + w^\top \hat{x} = 2w^\top \bar{x}$. This in turn implies that

$$w^\top \tilde{x} = w^\top \bar{x} \quad \text{and} \quad w^\top \hat{x} = w^\top \bar{x}.$$

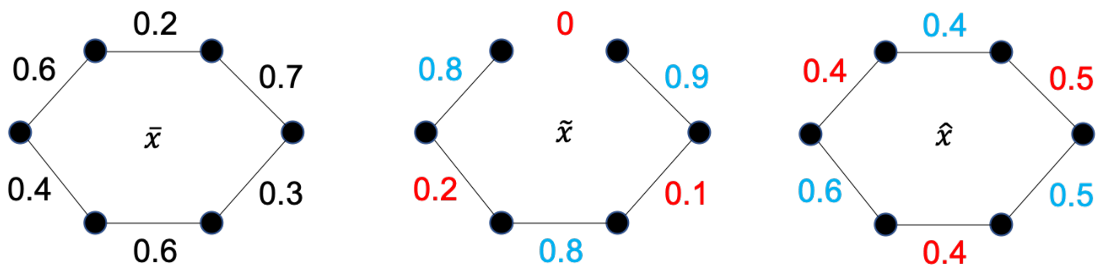

Figure 3.1: breaking a cycle

Therefore, both $\tilde{x}$ and $\hat{x}$ are optimal solutions to (3.3). Moreover, due to our construction, either $\tilde{x}$ or $\hat{x}$ has a value 0 for the position of an edge on C. Without loss of generality, we may assume that $\tilde{x}$ has the property. Then we may replace $\bar{x}$ by $\tilde{x}$ as $\tilde{S} = \{e \in E : \tilde{x}_e > 0\} \subseteq S$ and the cycle C no longer exists in $\tilde{S}$.

Breaking trees Breaking all cycles in the subgraph H, we may assume that H is a forest. Suppose that a connected component of H is a tree T on at least three vertices. Then T has a pair of two vertices u, v of degree one such that the uv-path on T has length at least two. Let the edges of the uv-path be enumerated by $e_1, \ldots, e_k$ for some $k \geq 2$. Take δ as

$$\delta = \min\{\bar{x}_e : \ e \in \{e_1, \ldots, e_k\}\}.$$

As before, we can argue that

$$\max\{\bar{x}_e : \ e \in \{e_1, \ldots, e_k\}\} \leq 1 - \delta.$$

Based on this, we take two feasible solutions $\tilde{x}$ and $\hat{x}$ by setting

$$\tilde{x}_{e_1} = \bar{x}_{e_1} + \delta, \quad \tilde{x}_{e_2} = \bar{x}_{e_2} - \delta, \quad \ldots,$$

$$\hat{x}_{e_1} = \bar{x}_{e_1} - \delta, \quad \hat{x}_{e_2} = \bar{x}_{e_2} + \delta, \quad \ldots,$$

while the other coordinates remain the same as $\bar{x}$. As before, we can argue that $\tilde{x}_e, \hat{x}_e \in [0, 1]$ for

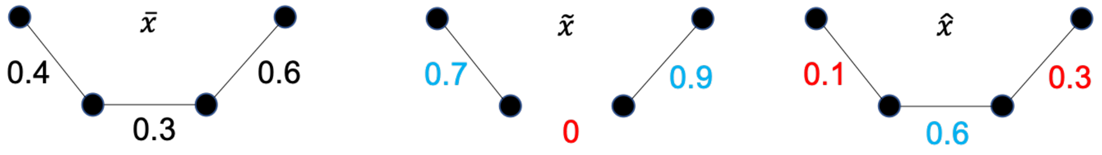

Figure 3.2: breaking a tree

all $e \in E$. Moreover, for an internal vertex w of the uv-path, we have

$$\sum_{w:ww' \in E} \bar{x}_{ww'} = \sum_{w:ww' \in E} \tilde{x}_{ww'} = \sum_{w:ww' \in E} \hat{x}_{ww'}.$$

For u, we have

$$\sum_{u:uw \in E} \tilde{x}_{uw} = \bar{x}_{e_1} + \delta \leq 1, \qquad \sum_{u:uw \in E} \hat{x}_{uw} = \bar{x}_{e_1} - \delta \leq 1.$$

Similarly, the degree constraint for v is also satisfied under $\tilde{x}$ and $\hat{x}$. Then $\tilde{x}$ and $\hat{x}$ satisfy the constraints of (3.3). At the same time, we have $\tilde{x} + \hat{x} = 2\bar{x}$. As $\bar{x}$ is an optimal solution, it follows that $\tilde{x}$ and $\hat{x}$ are also optimal to (3.3). Moreover, by our choice of δ, either $\tilde{x}$ or $\hat{x}$ has a value 0 for the coordinate of an edge on the uv-path. Without loss of generality, we assume that $\tilde{x}$ has the property. In this case, we can replace $\bar{x}$ by $\tilde{x}$ as $\tilde{S} = \{e \in E : \tilde{x}_e > 0\} \subseteq S$ and the tree T becomes splitted into two smaller trees.

After breaking all cycles and all trees with at least three vertices, each connected component of H has at most two vertices, which means that S is a matching. Suppose for a contradiction that $\bar{x}_e < 1$ for some $e \in S$ with $w_e \neq 0$. Then, for a sufficiently small $\delta > 0$, we have $\bar{x}_e - \delta, \bar{x}_e + \delta \leq 1$. However, as $w_e \neq 0$, it follows that either $w_e(\bar{x}_e - \delta)$ or $w_e(\bar{x}_e + \delta)$ is strictly greater than $w_e\bar{x}_e$, contradicting the optimality of $\bar{x}$. Hence, for any $e \in S$ with $w_e \neq 0$, we have $\bar{x}_e = 1$. For $e \in S$ with $w_e = 0$, we may update $\bar{x}$ by setting $\bar{x}_e = 1$ without changing the objective value. After all, we get that $\bar{x} \in \{0,1\}^{|E|}$ and S is a maximum weight matching. □

Consequently, the optimal value of (3.3) equals that of (3.2) when the graph G is bipartite. Moreover, the proof of Theorem 3.4 provides the following algorithm for computing a maximum weight matching in a bipartite graph. Theorem 3.4 guarantees that Algorithm 1 returns a maximum

Algorithm 1 LP-based algorithm for maximum weight bipartite matching

```
Solve the linear program (3.3) and get an optimal solution $\bar{x}$
Take $S = \{e \in E : \bar{x}_e > 0\}$ and the corresponding subgraph $H$
while $H$ contains a cycle do
    Find a cycle $C$ in $H$
    Break $C$ by updating $\bar{x}$ and $S$
end while
while $H$ contains a tree with at least three vertices do
    Take a connected component $T$ of $H$
    Break $T$ by updating $\bar{x}$ and $S$
end while
Return $S$
```

weight matching in a bipartite graph.

In practice, we may use the **simplex method** for solving the LP relaxation (3.3). For (3.3), we can in fact argue that the simplex method directly finds an optimal solution $\bar{x}$ with $\bar{x}_e \in \{0,1\}$ for all $e \in E$.