Combinatorial Optimization Jan – May, 2025

Lecture 2: Bipartite Min Cost Perfect Matching

Lecturer: T. Kavitha *Scribe: Soham Chatterjee*

1 Maximum Matching

We have the following theorem relating the sizes of minimum vertex cover and maximum matching in a bipartite graph from last lecture:

Theorem 1.1 König-Egerváry, 1931

In a bipartite graph, the size of a maximum matching is equal to the size of minimum vertex cover.

We will use König-Egerváry Theorem to prove Hall's Marriage Theorem, which establishes a necessary and sufficient condition for the existence of a perfect matching in a bipartite graph.

Theorem 1.2 Hall's Theorem, 1935

A bipartite graph $G = (L \cup R, E)$ has an L-perfect matching if and only if for all $S \subseteq L$: $|N(S)| \geq |S|$

Proof: Suppose G has a L-perfect matching. Let M be the L-perfect matching. Then let $S \subseteq L$. Then $N(S) \supseteq T$ where $T := \{v \in R : (u, v) \in M, u \in S\}$. Then $|T| = |S|$. Therefore we have $|N(S)| \geq |T| = |S|$.

Now suppose for all $S \subseteq L$, $|S| \leq |N(S)|$. Suppose G doesn't have a perfect matching. Let M be a maximum L-matching in G. So $|M| < |L|$. Let C be the minimum vertex cover. By König-Egerváry Theorem $|C| = |M|$. Therefore $|C| < |L|$. Hence take $S = \{u \in L : u \notin C\}$ and take $T = R \cap C$. Therefore we have $N(S) \subseteq T$. Hence we have

$$|C| = |L \cap C| + |T| = |L| - |S| + |T| \implies |S| = |L| - |C| + |T| > |T| \geq |N(S)|$$

Hence we got a set $S \subseteq L$ for which $|S| > |N(S)|$. Hence contradiction ↯. Therefore G has a L-perfect matching. ∎

2 Minimum Cost Perfect Matching

Bipartite Min Cost Perfect Matching
Input: Graph $G = (L \sqcup R, E)$ with $|L| = |R|$ and cost function $c : E \to \mathbb{R}$.
Question: Find a perfect matching M with minimum cost $c(M) = \sum_{e \in M} c(e)$

WLOG we can always assume G is the complete bipartite graph. Since if its not complete then we can add those edges with their cost being ∞. So from now on we will assume G is a complete bipartite graph.

2.1 Constructing an LP

We will first write a integer program for this problem. Since the bipartite graph is complete we will take a $n \times n$ symbolic matrix X and cost function is also a $n \times n$ matrix C where $c_{i,j} = c(i,j)$ for the edge $(i,j) \in E$.

Integer Program:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i,j} c_{i,j} x_{i,j} \\
\text{subject to} \quad & \sum_{j=1}^{n} x_{i,j} = 1 \quad \forall\, i \in [n], \\
& \sum_{i=1}^{n} x_{i,j} = 1 \quad \forall\, j \in [n], \\
& x_{i,j} \in \{0,1\} \quad \forall\, i,j \in [n]
\end{aligned}$$

We will see the LP-relaxation of this by replacing the constraint $x_{i,j} \in \{0,1\}$ by $0 \leq x_{i,j} \leq 1$.

$$\begin{aligned}
\text{minimize} \quad & \sum_{i,j} c_{i,j} x_{i,j} \\
\text{subject to} \quad & \sum_{j=1}^{n} x_{i,j} = 1 \quad \forall\, i \in [n], \\
& \sum_{i=1}^{n} x_{i,j} = 1 \quad \forall\, j \in [n], \\
& x_{i,j} \geq 0 \quad \forall\, i,j \in [n]
\end{aligned}$$

Observation 1. *The first two constraints of the LP suggests that the matrix X is doubly stochastic.*

The feasible region of the LP-relaxation contains all possible doubly stochastic matrix with each entry being non-negative. The feasible region is a bounded polyhedron or polytope. We will call this polytope P. We are optimizing a linear constraint over a polytope the optimum will be attained at one of the "corners" or *extreme points*.

Definition 2.1.1: Extreme Point

$u \in Q$ is an extreme point of a set Q if u cannot be written as $\lambda x + (1-\lambda)y$ where $x, y \in Q$, $x \neq y$ and $\lambda \in (0,1)$.

2.2 Finding Extreme Point of the LP

We aim to show in the following theorem that the extreme points of the polytope P correspond to perfect matchings in a bipartite graph. Every perfect matching corresponds to a permutation matrix. Therefore specifically, we will prove that any doubly stochastic matrix is a convex combination of permutation matrices. This implies that every extreme point of P is a $0-1$ vector, corresponding to a permutation matrix.

Theorem 2.2.1 Birkhoff-Von Neumann Theorem, 1946

Every doubly stochastic matrix can be written as a convex combination of permutation matrices.

Proof: Suppose there exists values $u_i \; \forall\, i \in L$ and $v_j \; \forall\, j \in R$ such that

$$u_i + v_j \leq c_{i,j} \; \forall\, i \in L, j \in R$$

Then for any perfect matching M we have

$$\sum_{(i,j) \in M} c_{i,j} \geq \sum_{i \in L} u_i + \sum_{j \in R} v_j$$

Therefore $\sum_{i\in L} u_i + \sum_{j\in R} v_j$ gives a lower bound on the cost of minimum cost perfect matching for the bipartite graph. So to get the best lower bound we would like to maximize this quantity. So we have the following LP:

$$\begin{aligned} \text{maximize} \quad & \sum_{i\in L} u_i + \sum_{j\in R} v_j \\ \text{subject to} \quad & u_i + v_j \le c_{i,j} \quad \forall\, i \in L, j \in R \end{aligned}$$

Let D be the polyhedron generated by the constraint $u_i + v_j \le c_{i,j}\ \forall\ i \in L, j \in R$. Now consider any $X \in P$. Then we have

$$\sum_{i\in L}\sum_{j\in R} c_{i,j}x_{i,j} \ge \sum_{i\in L}\sum_{j\in R}(u_i+v_j)x_{i,j} = \left(\sum_{i\in L}\sum_{j\in R} x_{i,j}\right) + \left(\sum_{j\in R} v_j \sum_{i\in L} x_{i,j}\right) = \sum_{i\in L} u_i + \sum_{j\in R} v_j$$

Therefore we have

$$\min_{\text{perfect matching } M} \sum_{(i,j)\in M} c_{i,j} \ge \min_{x\in P} \sum_{i\in L}\sum_{j\in R} c_{i,j}x_{i,j} \ge \max_{(u,v)\in D} \sum_{i\in L} u_i + \sum_{j\in R} v_j$$

Thus we get a primal-dual relation between the two LP problems.

Now our goal is to come up with a perfect matching M and $(u, v) \in D$ such that $\sum_{(i,j)\in M} c_{i,j} = \sum_{i\in L} u_i + \sum_{j\in R} v_j$. Then M will be optimal solution to the LP-relaxation. Given solution u^*, v^* of the dual LP a perfect matching M would satisfy equality if it contains only the edges (i, j) such that $c_{i,j} = u_i + v_j$ by *complementary slackness*. But for a given (u, v) we may not be able to find a perfect matching among the edges with $c_{i,j} = u_i + v_j$.

We will describe an algorithm now which performs a series of iterations to obtain an appropriate u and v. It maintains a feasible solution for the dual problem and finds an "almost" primal feasible solution x satisfying complementary slackness. Satisfying complementary slackness we are working in the subgraph $G' = (V, E')$ where $E' = \{(i, j) : c_{i,j} = u_i + v_j\}$.

Algorithm 1: Min-Cost-BPM

Input: Complete bipartite graph $G = (L \sqcup R, E)$ with $|L| = n$ and cost function $c : E \to \mathbb{R}$
Output: Find minimum cost perfect matching

begin
  Initialize $u_i \longleftarrow 0\ \forall\ i \in L$
  $v_j \longleftarrow \min_{i\in L} c_{i,j}\ \forall\ j \in R$
  willMatchingInc $\longleftarrow$ *True*
  while *True* **do**
    $E' \longleftarrow \{(i, j) : c_{i,j} = u_i + v_j\}$
    if *willMatchingInc* == *True* **then**
      $M \longleftarrow$ Find-Maximum-Matching$(G' = (V, E'))$
      if *M is perfect matching* **then**
        return M
    $(\hat{i}, \hat{j}) \longleftarrow \arg\min_{i\in\mathcal{E}_L} \min_{j\in\mathcal{E}_R\cup\mathcal{U}_R} (c_{i,j} - u_i - v_j)$
    if $(\hat{i}, \hat{j}) \in \mathcal{E}_L \times \mathcal{U}_R$ **then**
      willMatchingInc $\longleftarrow$ *False*
      Relabel vertices to obtain $O_L, O_R, \mathcal{E}_L, \mathcal{E}_R, \mathcal{U}_L, \mathcal{U}_R$
    $\delta \longleftarrow c_{(\hat{i},\hat{j})} - u_{\hat{i}} - v_{\hat{j}}$
    $u_i \longleftarrow u_i + \delta, \forall\ i \in \mathcal{E}_L$
    $v_j \longleftarrow v_j - \delta, \forall\ i \in O_R$

In the algorithm when M is not a perfect matching then it updates u, v to get a new feasible solution which has value $\sum_{i\in L} u_i + \sum_{j\in R} v_j$ greater than before which we will prove now and then runs the while loop again.

In the Find-Maximum-Matching algorithm if M is not a perfect matching then $C = O_L \cup O_R \cup \mathcal{U}_L$ is the minimum vertex cover of $G = (V, E')$. In order to update u, v to get u', v' we will use the information of the minimum vertex cover.

By the algorithm we obtain new u', v' where

$$u'_i = \begin{cases} u_i + \delta & \forall\ i \in \mathcal{E}_L \\ u_i & \forall\ i \in O_L \cup \mathcal{U}_L \end{cases} \qquad v'_j = \begin{cases} v_i - \delta & \forall\ i \in O_R \\ v_i & \forall\ i \in \mathcal{E}_R \cup \mathcal{U}_R \end{cases}$$

Claim 2.2.2

(u', v') is also a feasible solution of the LP

$$\begin{aligned} &\text{maximize} && \sum_{i \in L} u_i + \sum_{j \in R} v_j \\ &\text{subject to} && u_i + v_j \leq c_{i,j} \quad \forall\ i \in L, j \in R \end{aligned}$$

with

$$\sum_{i \in L} u'_i + \sum_{j \in R} v'_j > \sum_{i \in L} u_i + \sum_{j \in R} v_j$$

Proof: The edges in $G' = (V, E')$ are in $O_L \times \mathcal{E}_R$, $\mathcal{E}_L \times O_R$ and $\mathcal{U}_L \times \mathcal{U}_R$. Therefore $\mathcal{E}_L \cup \mathcal{E}_R \cup \mathcal{U}_R$ is an independent set. Let after updating the vectors u, v we denote it by $\hat{u}, \hat{v}$. For edges in $(i, j) \in E \cap (O_L \times \mathcal{E}_R \cup \mathcal{U}_L \times \mathcal{U}_R)$ we have $\hat{u}_i = u_i$ and $\hat{v}_j = v_j$. Therefore

$$\hat{u}_i + \hat{v}_j = u_i + v_j \leq c_{i,j}$$

For edges in $(i, j) \in \mathcal{E}_L \times O_R$ we have $\hat{u}_i = u_i + \delta$ and $\hat{v}_j = v_j - \delta$. Therefore

$$\hat{u}_i + \hat{v}_j = u_i + \delta + v_j - \delta = u_i + v_j \leq c_{i,j}$$

Therefore $(\hat{u}, \hat{v})$ is a feasible solution of the LP.
Now we have

$$\sum_{i \in L} \hat{u}_i + \sum_{j \in R} \hat{v}_j - \sum_{i \in L} u_i - \sum_{j \in R} v_j = \delta(|\mathcal{E}_L| - |O_R|) = \delta(|L| - |C|) = \delta(n - |C|)$$

Since every vertex of O_R is matched and matched with vertices of $\mathcal{E}_L$ and $\mathcal{E}_L$ also contains the starting unmatched vertices of L we have $|\mathcal{E}_L| > |O_R|$. Hence the value strictly increases. ■

Claim 2.2.3

In each iteration of while loop size of the edge set E' increases by 1.

Proof: In any iteration of the while loop in the graph $G' = (V, E')$ none of the the edges in $\mathcal{E}_L \times (\mathcal{E}_R \cup \mathcal{U}_R)$ are present. Now in line 9 we are taking δ to be the minimum of $c_{i,j} - u_i - v_j$ over all edges of $\mathcal{E}_L \times (\mathcal{E}_R \cup \mathcal{U}_R)$. Let that edge is (i', j'). Then $c_{i',j'} - u_i - v_j = \delta$. Then in line 10 the algorithm updates $\hat{u}_{i'} = u_{i'} + \delta$ but $\hat{v}_{j'}$ is unchanged since $j' \in \mathcal{E}_R \cup \mathcal{U}_R$. Then in the next iteration of the while loop we have for the edge (i', j')

$$c_{i',j'} - \hat{u}_{i'} - \hat{v}_{j'} = c_{i',j'} - u_{i'} - \delta - v_{j'} = 0$$

Therefore the edge (i', j') is in the edge set constructed in line 5 in the next iteration of the while loop. The edges in $(\mathcal{E}_L \times O_R) \cup (O_L \times \mathcal{E}_R) \cup (\mathcal{U}_L \times \mathcal{U}_R)$ continue to remain tight after our update of u_i, v_j values. Since each time one edge is becoming tight the edge set is increasing by 1 in each iteration of the while loop. ■

Since each time one edge is becoming tight the edge set is increasing by one and henceforth the algorithm terminates. Therefore in after $O(n^2)$ iterations all the edges has been added and therefore the matching found is perfect.

Now if an edge of $\mathcal{E}_L \times \mathcal{U}_R$ becomes tight then the maximum matching continues to remain maximum matching. So only the labels of vertices are changed and new odd, even and unreachable sets are obtained. Therefore we don't need to run the algorithm for maximum matching again. So it takes $O(n^2)$ time for such iterations.

When an edge of $\mathcal{E}_L \times \mathcal{E}_R$ becomes tight then there exists an augmenting path and the size of the maximum matching increases. In this case only we run the Find-Maximum-Matching algorithm on the tight edges. In each such

iterations the size of the maximum matching increases. Therefore there can be $O(n)$ iterations of this kind. Therefore in these iterations it takes $O(n^3)$.

Therefore the total time taken by the algorithm is $O(n^2) \times O(n^2) + O(n) \times O(n^3) = O(n^4)$. Now since carefully choosing the cost function we can make any extreme point of the polytope of LP be the optimum solution of the linear program this proves that all the extreme points are integral. ■

Therefore we have proved both that we can find the minimum cost perfect matching in a bipartite graph in $O(n^4)$ times and every doubly stochastic matrix can be written as a convex combination of permutation matrix.