Lecture 24

Consider the constraint matrix of the bipartite matching problem.

- let $u_1, \ldots, u_n$ be the vertices
- let $e_1, \ldots, e_m$ be the edges.

The constraint matrix looks as follows:

$$M = \begin{array}{c} \\ u_1 \\ \vdots \\ u_n \end{array} \begin{array}{c} e_1 \; \cdots \; e_m \\ \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} \end{array}$$

→ this is a 0/1 matrix with $M[u_i, e_j] = 1$ if u_i is one of the endpoints of e_j and $M(u_i, e_j) = 0$ otherwise

Consider any $k \times k$ submatrix of M.

- if there is an all-0's row or column then the determinant of this submatrix is 0.
- if there is a row or column with exactly one 1 then expand using that row/column. Continue doing this step till we are either left with a 1×1 matrix (so determinant $= \pm$ value of this elmt) or there is an all-0's row/column (in this case determinant is 0) or we have a matrix where every row has ≥ 2 1's and every column has exactly 2 1's.

Claim. In the last case, the determinant of this matrix is 0.

Proof. Observe that every row also has exactly two 1's. This is because there are exactly $2s$ 1's in this matrix if its dimension is $s \times s$ — since every column has exactly two 1's. Since every row has at least two 1's and there are s rows, every row has exactly two 1's. So this matrix is a disjoint collection of cycles.

Since the graph is bipartite, each cycle is of even length.

We will come up with a linear combination of the rows of this $s \times s$ matrix so that it becomes the all-0's vector. This means the rows of this matrix are linearly dependent - hence its determinant is 0.

$$\begin{array}{c|c} & e_{j_1} \ \ldots \ e_{j_s} \\ \begin{matrix} u_{i_1} \\ \\ \\ u_{i_s} \end{matrix} & \left[\begin{matrix} \\ \\ \\ \\ \end{matrix} \qquad\qquad \right] \end{array} \equiv \text{a disjoint collection of even length cycles.}$$

Multiply rows indexed by vertices in A with -1 and rows indexed by vertices in B with $+1$ and add all the rows. Observe that we get the all-0's vector. ▨

Let us now prove König-Egerváry theorem. Consider the LP dual to the bipartite matching LP. The variables here are y_u $\forall$ vertices u.

$$\min \sum_{u \in A \cup B} y_u$$

subject to

$$y_u + y_v \geq 1 \quad \forall (u,v) \in E$$
$$y_u \geq 0 \quad \forall u \in A \cup B.$$

This is the minimum fractional vertex cover LP. However since the constraint matrix (this is the transpose of the constraint matrix of the primal LP) is totally unimodular, this is the minimum integral vertex cover LP - note that this is so only for bipartite graphs. due to total unimodularity

So we have size of maximum matching = size of maximum fractional matching = size of minimum fractional vertex cover = size of vertex cover.

due to LP-duality

due to total unimodularity

Exercise. Show that every doubly stochastic matrix is a convex combination of permutation matrices.

- a doubly stochastic matrix is an $n \times n$ matrix with non-negative entries such that every row sum is 1 and every column sum is 1.
- a permutation matrix is an $n \times n$ matrix with 0/1 entries such that every row and every column has exactly one 1 in it.
- a convex combination is a linear combination where all coefficients are non-negative and sum to 1.

The above statement is called Birkhoff-von Neumann theorem.

Complementary Slackness

We have our primal LP on k variables and dual LP on n variables.

$(x_1^*, \dots x_k^*)$ is a primal optimal solution and $(y_1^*, \dots, y_n^*)$ is a dual optimal solution if and only if:

- for every $1 \leq j \leq k$: either $x_j^* = 0$ or $\sum_i a_{ij} y_i^* = c_j$
- for every $1 \leq i \leq n$: either $y_i^* = 0$ or $\sum_j a_{ji} x_j^* = b_i$

The above conditions are called complementary slackness conditions. Why do they hold?

We have $\sum_i \sum_j a_{ij} x_j^* y_i^* = \sum_i y_i^* \left(\sum_j a_{ij} x_j^*\right)$

$$\geq \sum_i y_i^* b_i$$

$\sum_i \sum_j a_{ij} x_j^* y_i^* = \sum_j x_j^* \left(\sum_i a_{ij} y_i^*\right)$

$\leq \sum_j x_j^* c_j$

Since $(x_1^*, \dots, x_k^*)$ is primal optimal & $(y_1^*, \dots, y_n^*)$ is dual optimal, we have $\sum_j c_j x_j^* = \sum_i b_i y_i^*$.

So the above complementary slackness conditions hold.

Minimum cost or weight perfect matching in bipartite graphs

Our input is a complete bipartite graph $G = (A \cup B, E)$ where every edge (i,j) has a cost c_{ij} associated with it.

The goal is to find a perfect matching M minimizing $\sum_{(i,j) \in M} c_{ij}$

We can solve this problem by solving the following LP:

$$\min \sum_{i,j} c_{ij} x_{ij}$$

subject to

$$\sum_j x_{ij} = 1 \quad \forall\, i \in A$$

$$\sum_i x_{ij} = 1 \quad \forall\, j \in B$$

$$x_{ij} \geq 0 \quad \forall\, i \in A,\, j \in B.$$

Since the above constraint matrix is totally unimodular (why?), there is an optimal solution to the above LP that is integral and we can assume that the LP-solver returns this solution. Any integral feasible solution is the incidence vector of a perfect matching. Thus we have found a min-cost perfect matching.

Though we can solve an LP in polynomial time, the running time is quite large. We prefer combinatorial algorithms. So we will now see a combinatorial algorithm which is based on linear programming. This algorithm is "primal-dual". Let us write down the dual LP.

$$\max \sum_i u_i + \sum_j v_j$$

subject to

$$u_i + v_j \leq c_{ij} \quad \forall\, i \in A,\, j \in B$$

Note: We have no non-negativity constraints on u_i, v_j since the primal LP constraints are equalities.

Our goal is to find a dual feasible solution $(u_i)_{i \in A}$, $(v_j)_{j \in B}$ and a perfect matching M such that we have $\sum_{(i,j) \in M} c_{ij} = \sum_{i \in A} u_i + \sum_{j \in B} v_j$.

Then weak LP-duality implies that M is a primal optimal solution.

For the above equality to hold, the matching M has to contain only those edges (i,j) for which $c_{ij} = u_i + v_j$ (by complementary slackness).

The algorithm works as follows.

- Start with any dual feasible solution, say $u_i = 0$ for all i and $v_j = \min_i c_{ij}$ for all j.

In a given iteration, the algorithm has a dual feasible solution (u, v). We check if the subgraph G_0 whose edge set consists of edges (i,j) such that $c_{ij} = u_i + v_j$ admits a perfect matching or not.

- Here we use the max-size matching algorithm in bipartite graphs seen earlier in the course.

If G_0 has a perfect matching then the edge incidence vector of this matching is a primal feasible solution and satisfies complementary slackness with the current dual solution. Hence this is a min-cost perfect matching in G.

Suppose G_0 does not have a perfect matching. Then the algorithm will update the dual solution such that the value of the dual solution increases (recall that we are maximizing the dual).

Let $\delta = \min_{i \in A', j \in B'} (c_{ij} - u_i - v_j)$

where A' = set of vertices in A reachable by an $\underbrace{\text{alternating path}}_{\text{a path whose edges alternate between being in } M \text{ and not in } M}$ from an unmatched vertex in A

a path whose edges alternate between being in M and not in M.

and B' = set of vertices in B <u>not</u> reachable by an alternating path from an unmatched vertex in A.

By definition of A' and B', note that $c_{ij} > u_i + v_j \ \forall\, i \in A'$ and $j \in B'$. Thus $\underline{\delta > 0}$.

Let us update the dual solution as follows:

$$u_i = \begin{cases} u_i & \text{for } i \in A - A' \\ u_i + \delta & \text{for } i \in A' \end{cases} \qquad v_j = \begin{cases} v_j & \text{for } j \in B' \\ v_j - \delta & \text{for } j \in B - B'. \end{cases}$$

Check that the above (u, v) is dual feasible. Also check that the difference between the values of the new dual solution and the old dual solution is equal to $\delta\left(\frac{n}{2} - k\right)$, where k is the size of maximum matching or the size of minimum vertex cover in G_0. Since $k < n/2$, the value of the dual solution strictly increases.

We repeat the above procedure until the algorithm terminates. At that point, we have an incidence vector of a perfect matching and a dual feasible solution that satisfy complementary slackness. Thus we find a min-cost perfect matching.

We still have to show that the algorithm terminates. Observe that whenever we update the dual solution, at least one more vertex in B becomes reachable by an alternating path from an unmatched vertex in A and no vertex in B becomes unreachable. Thus this algorithm runs for $\leq n/2$ iterations.