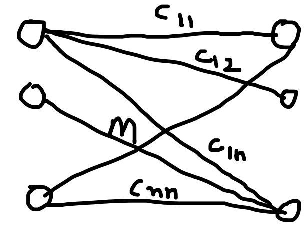
Persons  jobs

$C = \begin{bmatrix} & \end{bmatrix}$

22 Lecture 22

Assignment problem

[22.1] What is assignment problem? Suppose that there are n machines and n jobs. Let c_{ij} be the cost of making the ith machine do the jth job. The task of assigning one job to each machine so that the total cost is minimized is an ASSIGNMENT PROBLEM (AP). We use $c_{ij} = \infty$ or a large number M, to mean that the machine i cannot do job j.

0	0
0	0
0	0

σ

$\begin{pmatrix}1&2&3\\1&2&3\end{pmatrix}$, $\begin{pmatrix}1&2&3\\2&1&3\end{pmatrix}$, $\begin{pmatrix}1&2&3\\1&3&2\end{pmatrix}$, $\begin{pmatrix}1&2&3\\2&3&1\end{pmatrix}$,

$\begin{pmatrix}1&2&3\\3&1&2\end{pmatrix}$, $\begin{pmatrix}1&2&3\\3&2&1\end{pmatrix}$ $\rightarrow$ permutations

$$c(\sigma) = c_{1\sigma(1)} + c_{2\sigma(2)} + c_{3\sigma(3)}$$
$$= c_{11} + c_{23} + c_{32}$$

[22.2] What are the feasible solutions? Note that a permutation σ of $[n] := \{1, 2, \ldots, n\}$ is a solution of the ap with the understanding that machine i gets the job $\sigma(i)$. So the cost of the solution σ is $c(\sigma) = c_{1\sigma(1)} + \cdots + c_n\sigma(n)$. So the feasible set is a finite set with $n!$ elements.

$S_n \rightarrow$ set of all permutations of $[n]$ $:= \{1, 2, \cdots, n\}$

A 0-1-permutation matrix is also viewed as a solution.

$\begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix} \longrightarrow$ permutation matrices

[22.3] First impression about ap Our aim is to minimize $c(\sigma)$ over S_n, the set of all permutations of $[n]$.

If we try to do it by checking each permutations, then for $n = 50$, it will take the fastest computers billions of years. So this is not practical.

[22.4] Viewing ap as btp or lpp A solution to the ap may be viewed as a solution to the corresponding btp with $a_i = b_j = 1$ by Corollary [18.14], where each bfs has n basic variables 1 and $n - 1$ basic variables 0 (each bfs is degenerate).

Hence an ap may be viewed as a special case of an slpp. We may use simplex method with Bland's rule to obtain a solution. However, due to degeneracy, it may take a lot of time.

[22.5] A special algorithm Because of the very special nature of the problem, there is a special method to obtain a minimum solution. It is called the HUNGARIAN METHOD.

The name of this method was given by professor H. Kuhn, for *Eugene Egervary*, who published his ideas of extensions of Koñig's works, in a paper, in Hungarian language in 1931. This paper contained ideas for weighted bipartite matching problem, which, of course is the assignment problem.

$$\begin{array}{c|c} & +1 \quad -3 \quad +2 \quad -3 \\ \begin{array}{c}-2\\-1\\-1\\+2\end{array} & \begin{bmatrix}2&3&4&1\\2&4&2&3\\3&1&5&2\\7&2&7&1\end{bmatrix}\end{array}$$

-2

$\rightarrow -3$

value 11

$-2 \quad -3$

value $-2+2+6+0$
$= 6$

$$\begin{array}{c|c} & +1 \quad -3 \quad +2 \quad -3 \\ \begin{array}{c}-2\\-1\\-1\\+2\end{array} & \begin{bmatrix}2&3&4&1\\2&4&2&3\\3&1&5&2\\7&2&7&1\end{bmatrix}\end{array}$$

value 12

value 7

$4+0+3+0=7$

[22.6] **Lemma: reduction method** Suppose that an ap with cost matrix $C_{n\times n}$ is given. Let $d_1, \dots, d_n, e_1, \dots, e_n$ be some real numbers. Define a new cost matrix B by $b_{ij} = c_{ij} - d_i - e_j$, $i, j = 1, \dots, n$. Let $\sigma \in S_n$ be any permutation. Show that $c(\sigma)$ (cost of σ wrt C) and the $b(\sigma)$ always differ by the same constant. Thus, σ is a minimum solution for the cost matrix B iff σ is a minimum solution for the cost matrix C.

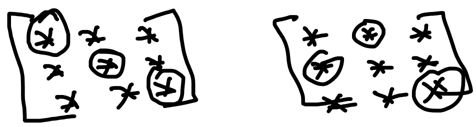
Proof. We show that for each solution σ, the difference $c(\sigma) - b(\sigma)$ is the same constant.

$\|$

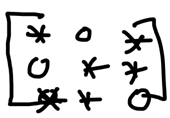
$d_1 + \cdots + d_n + e_1 + \cdots + e_n$

For any σ, we have $\sum_i [c_{i\sigma(i)} - d_{i\sigma(i)}] = \sum_i [d_i + e_{\sigma(i)}] = \sum_i d_i + \sum_j e_j$. The next assertion follows easily. ■

[22.7] **Terminology : diagonal** Suppose that we have a square matrix A and let σ be a permutation. Circle the positions $(1, \sigma(1)), \dots, (n, \sigma(n))$. For some time, let us call the collection of these positions a DIAGONAL. So, there are 3! diagonals of a 3×3 matrix. You can draw them.



$\longrightarrow$ 3! diagonals.



[22.8] **A crucial lemma** Suppose that we have an ap with a nonnegative cost matrix C, and that C has a diagonal of 0's. Then this diagonal gives us a minimum solution for this ap.

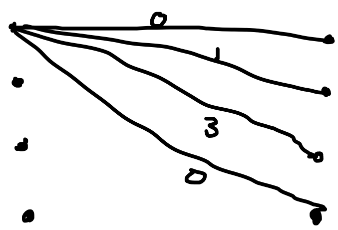
Proof. As the cost of this diagonal is 0 and the cost of any diagonal is nonnegative, the conclusion follows. ■

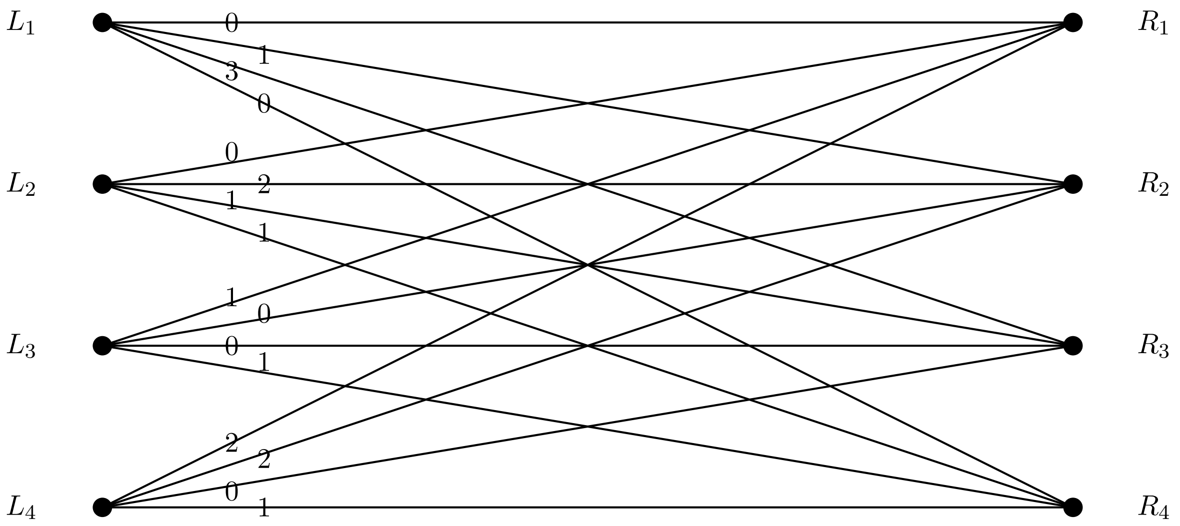
[22.9] **Idea** So, it is a good idea to create more and more zeros in our cost matrix by using [22.6] while keeping the matrix nonnegative. This is precisely the idea of Hungarian algorithm. Of course, one has to show that this can be done till we get a minimum solution.

Graph theoretic meaning of ap

Let $C \in M_n$ be a nonnegative matrix. With C, associate a complete weighted bipartite graph G_C in the following way. The parts are $\{L_1, \dots, L_n\}$ (put these vertices on the left one above another) and $\{R_1, \dots, R_n\}$. Take the weight of the edge $[L_i, R_j]$ as c_{ij}.

[22.10] **Example** Graph G_C for $C = \begin{bmatrix} 0 & 1 & 3 & 0 \\ 0 & 2 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 2 & 2 & 0 & 1 \end{bmatrix}$ is





[22.11] **Diversion** Are you thinking about how many pairs of intersecting lines are there? You may, but that discussion is not in our to do list.

[22.12] **About matchings**

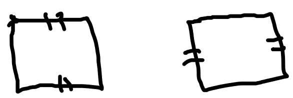
a) A MATCHING in a graph G is a collection of edges which do not share any endvertices.

$\{[L_1, R_1], [L_2, R_3]\} \rightarrow$ a matching has size 2.

b) For example, in the above graph G_C, the set $\{[L_1, R_2], [L_2, R_3]\}$ is a matching. So is the set $\{[L_1, R_1], [L_2, R_2], [L_3, R_3]\}$.

c) A MAXIMUM MATCHING is a matching of the maximum possible size. ✓

d) A PERFECT MATCHING is a matching that involves all the vertices of the graph.



e) In the previous graph G_C, the set $M = \{[L_1, R_2], [L_2, R_3], [L_3, R_4], [L_4, R_1]\}$ is a perfect matching (also a maximum matching). It corresponds to the diagonal $\{c_{12}, c_{23}, c_{34}, c_{41}\}$ in the matrix C.

f) The cost (weight) of a matching is the sum of the weights of the edges involved.

g) Thus, for the above matching M in e), we have $c(M) = c_{1,2} + c_{2,3} + c_{3,4} + c_{4,1} = 5$.

[22.13] Discussion

a) So the assignment problem asks us to find a minimum cost perfect matching from the graph G_C.

b) First, do we even know how to find a maximum matching in a given bipartite graph which may not be complete?

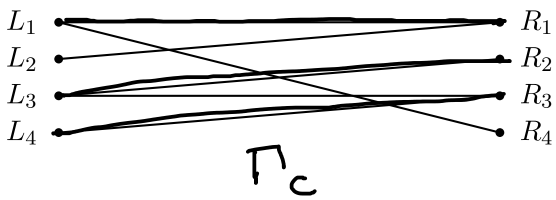
c) If we know the answer then we can use that to tell whether the 0-weighted set of edges contain a perfect matching. If the answer turns out to be yes, then we are done.

d) If those edges do not contain a perfect matching, may be we can use the reduction method to create some more zeros and proceed.

[22.14] Definition : defining Γ_C differently

a) Normally, given a 0-1 matrix $C \in M_n$, one assigns a bipartite graph Γ_C taking the 1's for edges. However, since the zeros are important to us here, we shall consider that the graph $\boldsymbol{\Gamma_C}$ is made by taking the 0's for the edges, that is, L_i is adjacent to R_j if $c_{ij} = 0$. And we shall keep the matrix C nonnegative.

b) For example, the graph Γ_C for $C = \begin{bmatrix} 0 & 1 & 3 & 0 \\ 0 & 2 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 2 & 2 & 0 & 1 \end{bmatrix}$ is



[22.15] Points and lines

a)In this context, an element 0 of C will be called a POINT. In Γ_C it means an edge.

$(1,\phi)$

$(1,4)$

b) A row or a column of C will be called a LINE. In Γ_C it stands for a vertex.

c) A set of points in C are called INDEPENDENT if no two points are contained in a line. In Γ_C it means a matching.

d) In the previous example, $\{c_{14}, c_{21}, c_{32}\}$ is an independent set of points. It stands for the matching $\{[L_1, R_4], [L_2, R_1], [L_3, R_2]\}$ in Γ_C. This is not a maximum matching.

[22.16] Current questions and their difficulties

a) So, our main question is that given a nonnegative $C \in M_n$, how do we know whether C has a diagonal of zeros? Equivalently, whether Γ_C has a perfect matching?

b) If not, what is the maximum size of an independent set of points (zeros) in C? Equivalently, what is the maximum size of matching in Γ_C?

c) It is not an easy task. For example, you may try to find a maximum independent set of points (zeros) in the given matrix and imagine the difficulty with a matrix of size 50.

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

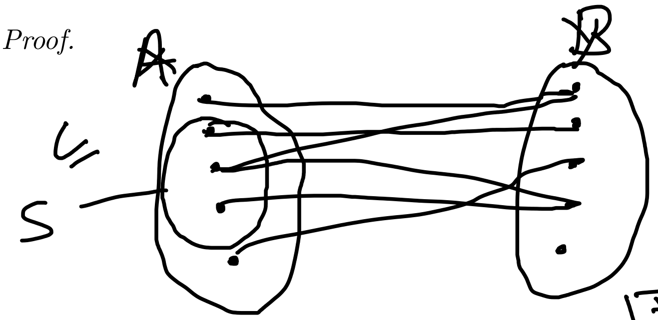
~~matrix~~ max set of ind zeros.

Hall's theorem and bipartite matching algorithm

Hall's theorem gives a necessary and sufficient condition for the existence of a perfect matching in a bipartite graph. It's proof gives an idea to write an algorithm to find a maximum matching. For a set of vertex S, let $N(S)$ mean the set of neighbors of S (set of vertices that are adjacent to some vertex of S).

[22.17] **<u>Hall's Theorem</u>** Let G be a bipartite graph with parts A and B with $|A| = |B| = n$. Then G has a perfect matching iff G satisfies the condition '$|N(S)| \geq |S|$ for each $S \subseteq A$'.

Proof.

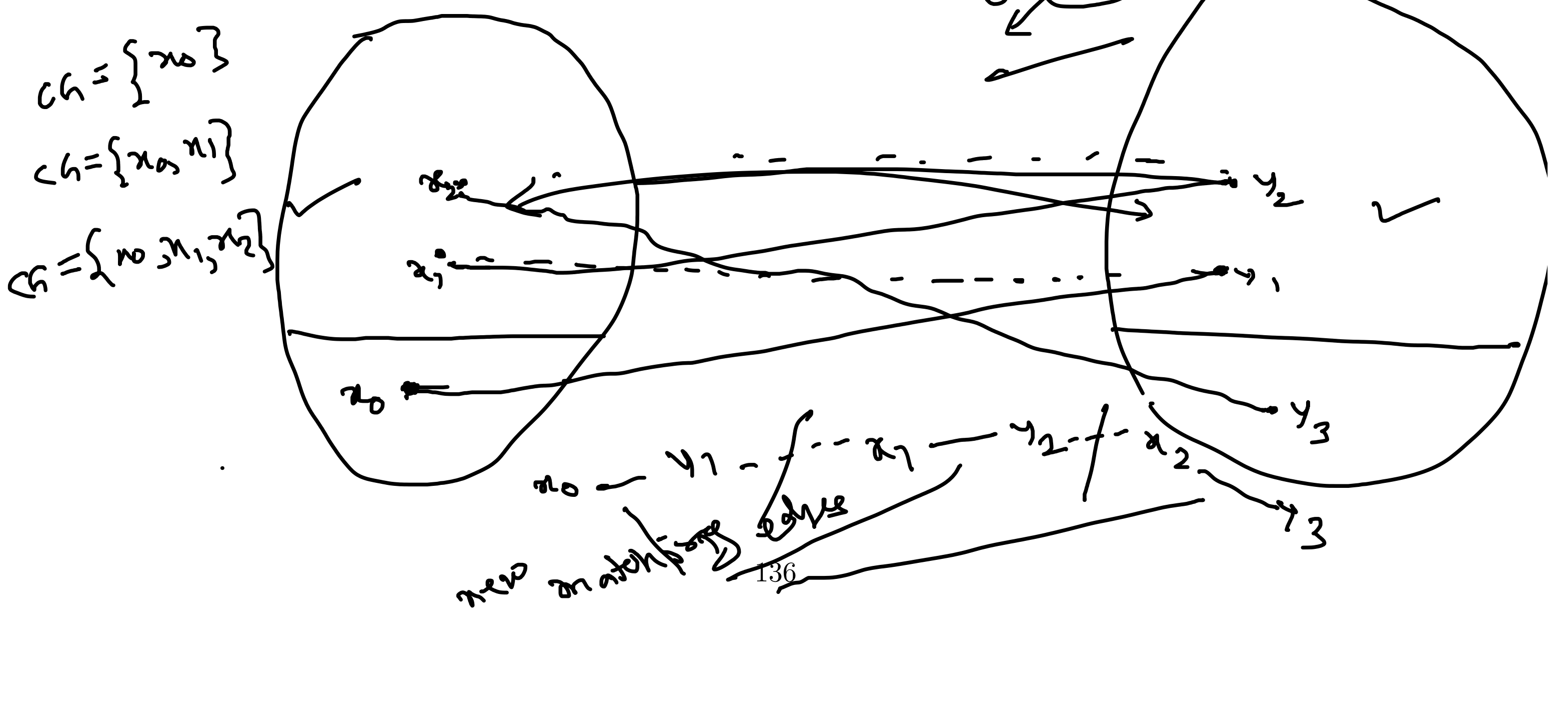


$|S| = 3$

$|N(S)| \geq 3$

$\boxed{\forall\ S \subseteq A \text{ we } \cancel{\text{must}} \text{ have } |N(S)| \geq |S|}$

Take a Max matching M



The necessity of the condition is easy it see. To show the sufficiency, suppose that the condition holds and still G does not have a perfect matching. Take a maximum matching M and an unmatched vertex $x_0 \in A$.

Imagine that elements of A are persons and elements of B are jobs and an edge between $a \in A$ and $b \in B$ is there if a has given the job b as a choice. In this setting, M stands for a maximum possible assignment of jobs and in that x_0 does not get a job. If $S \subseteq A$, then $N(S)$ would mean the total job choices of S.

Put current group $CG = \{x_0\}$. By the hypothesis, the total job choices given by CG is at least $|CG|$. If these job choices has an unassigned job, we are done.

So assume that all the job choices by CG are already taken. Then take the persons who have taken these jobs and include them in CG. Suppose, $x_{11}, \ldots, x_{1k_1}$ are those persons.

So right now $CG = \{x_0, x_{11}, \ldots, x_{1k_1}\}$. By the hypothesis, the total job choices given by CG is at least $|CG|$. If these job choices has an unassigned job, we will show that we are done. Otherwise, these jobs are already taken. Then take the persons who have taken these jobs and include them in CG.

This cannot continue indefinitely, as each time the size of CG is increasing[14] and it cannot exceed $|A|$. Thus, there will come a stage, when the job choices of CG will include an unassigned job j_0.

There is a path from j_0 to x_0 of the form

$$j_0,\ x_{k,*},\ j(x_{k,*}),\ x_{k-1,*},\ j(x_{k-1,*}),\ \cdots,\ x_{1,*},\ j(x_{1,*}),\ x_0.$$

Here we use $x_{i,*}$ to mean some person who was added in the ith stage and $j(x_{i,*})$ means his job.

In this case, if we do the reassignment as

$$j_0 \to x_{k,*},\ j(x_{k,*}) \to x_{k-1,*},\ \cdots,\ j(x_{2,*}) \to x_{1,*},\ j(x_{1,*}) \to x_0$$

we get an assignment of one more job. This is a contradiction. ∎

[14] This is because, at each stage CG consists of x_0 and the persons who have taken the jobs identified till that stage and hence in the next stage the number of job choices will be larger than the number of current job choices.