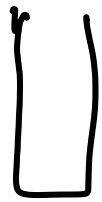
COL 702 Lecture 13 Sept 11

Some applications of the Matroid theorem

Knapsack :



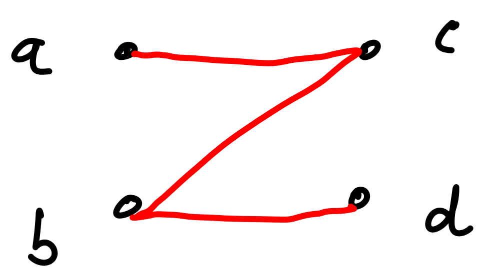
$B = 10$

	x_1	x_2	x_3
Weights	6	3	9

Maximal feasible subsets have different cardinalities

$\{x_1, x_2\}$ $\{x_3\}$

Matching



Maximal subsets

$\{(a,c)\ (b,d)\}$

$\{(b,c)\}$

Greedy doesn't work for Knapsack/Matching

Job scheduling problem

Set of jobs $J_1, J_2, \ldots J_n$

Time spans $\Delta_1 \; \Delta_2 \; \ldots \; \Delta_n$

Deadlines $d_1 \; d_2 \; \ldots \; d_n$

Penalty (of not completing within deadlines) $p_1 \; p_2 \; \ldots \; p_n$

Goal: Schedule all jobs so as to minimise penalty incurred.

Special case $\Delta_i = 1$

Example

	J_1	J_2	J_3
deadlines	1	2	1
penalty	5	2	8

A schedule is a mapping of Jobs to time slots, such that no more than one job can be done in the same time unit

Possible schedules $\{J_1, J_2\}$ (reverse cannot be scheduled)

Penalty 8 $\quad$ 1 $\quad$ 2

✓ 5 $\quad \{J_3, J_2\}$

Any subset of a feasible schedule is also feasible

<u>Subset system framework</u>:

S : set of jobs

Independent subsets $\mathcal{I}$: subset of jobs that can be scheduled without missing deadlines

Objective : Minimizing the penalty of the jobs not scheduled is equivalent to maximizing penalty of the scheduled

Note the defn of independent subsets do not have the scheduling information

<u>Problem</u> : Given a set of feasible jobs, how do actually schedule

schedule	J_{i_1}	J_{i_2}	J_{i_3}	J_{i_4}	$\cdots$	J_{i_k}
Time slots	t_1	t_2	t_3	$\cdot$		t_k
deadlines	d_1	d_2	d_3			d_k

If it is feasible then $t_i \le d_i$

Suppose J_a is scheduled before J_b

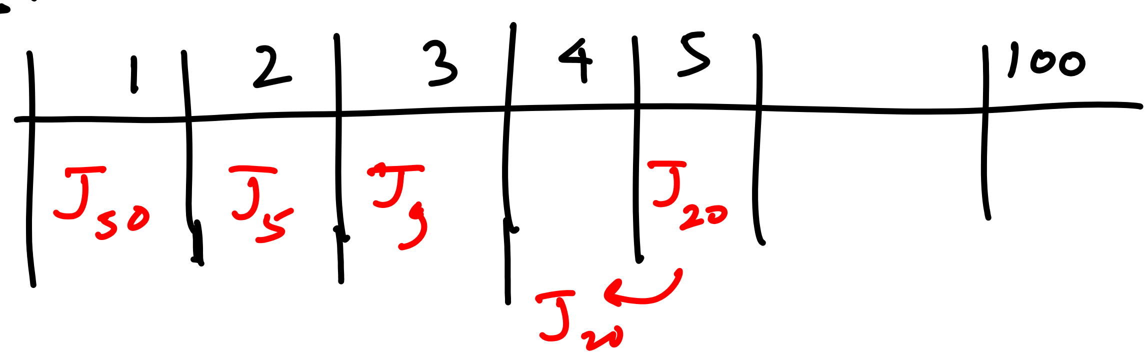
$$t_a < t_b$$

but $d_a > d_b$

<u>Claim</u> Swapping the time slots of J_a J_b will preserve feasibility

<u>Problem</u>: Given a set of feasible jobs, design an efficient algorithm for scheduling

<u>Observation</u>: time slots



In a feasible schedule with gaps, we can maintain feasibility by removing the gaps

Is it a Matroid

We will try to prove the exchange property

We have two sets of feasible jobs say S_1 and S_2 s.t. $|S_2| = |S_1| + 1$

Can we find a job $J' \in S_2 - S_1$ s.t. $S_1 \cup \{J'\}$ is feasible

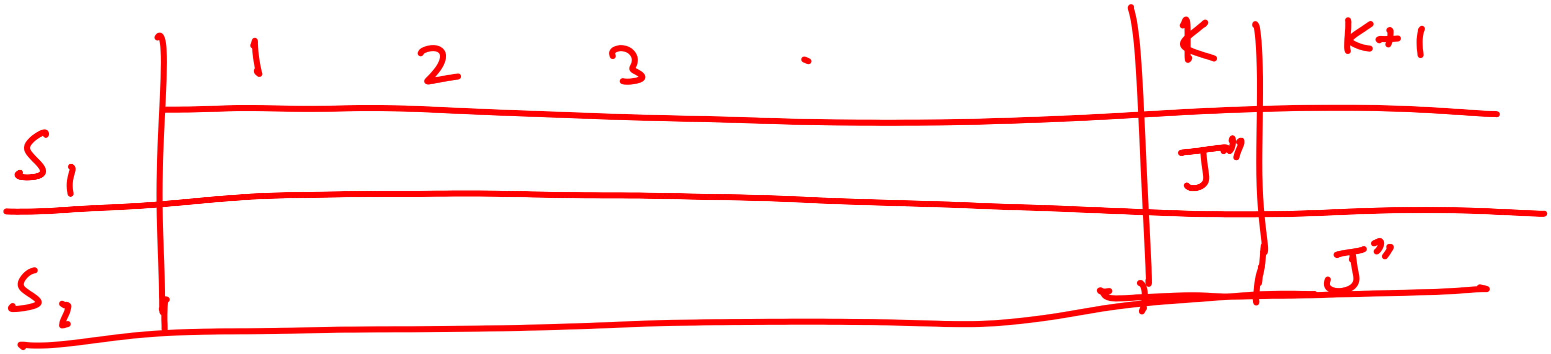
	1	2	3	4	. . i		k	$K+1$
S_1	J_1	J_2	J_3		J_i		J_k	
S_2	J'_1	J'_2	J'_3		J'_i		J'_k	J'_{K+1}

We have feasible schedule of S_1 and S_2

<u>Case 1</u> $J'_{K+1} \notin S_1$ Easy: Include J'_{K+1} in S_1

Case 2 $J'_{k+1} \in S_1$. Suppose $J_i = J'_{k+1}$

Then we can transform the schedules



Apply the same argument to $S_1 - J''$ and $S_2 - J''$ until S_1 has no jobs and S_2 has one job. Clearly we can include that job to S_1

Exchange property holds and therefore greedy works

Some points of Generic greedy

1. Even if generic greedy doesn't work (i.e. not matroid) some other version of greedy may work (eg. Prims)
2. Even if greedy doesn't work to give the maximum profit, it may still be effective

Effective: May still give some guarantees like 50% of the maximum etc.

Knapsack problem

$B = 13$

	x_1	x_2	x_3
wt	5	9	8
profit	20	40	30
ratio	4	> 4	< 3

Suppose we look at profit/wt ratio

Choose the object with the best ratio until we cannot add any more

Suppose the decreasing order of ratios is

$$y_1 \quad y_2 \quad y_3 \cdots \boxed{y_k} \mid y_{k+1}$$

Full

Choose $\max [\{y_1, y_2 \cdots y_k\}, y_{k+1}\}$

This guarantees a soln that is
at least $\frac{1}{2}$ as good as optimum