

# NOTE BOOK

Rokshana Ahmed

## KNAPSACK

- we have  $n$  jobs, the  $i$ th of which takes  $w_i$  seconds to finish, and has value  $v_i$ .

Our CPU is available for  $W$  seconds.

A set  $S$  of jobs is feasible if:

$$\sum_{i \in S} w_i \leq W$$

What is a set  $S$  of jobs satisfying  $\sum_{i \in S} w_i \leq W$  and having maximum total value?

We assume that  $W, w_1, \dots, w_n$  are integers.

$$OPT(i, w) = \max_{S \subseteq \{1, 2, \dots, i\}} \sum_{j \in S} v_j$$

$$\sum_{j \in S} w_j \leq W$$

$w_j$  = "length" of job;  
 $v_j$  = "payment" you get for running job  $j$

T: If  $O_{i,w}$  is an optimal solution (a set of jobs) for the jobs  $\{1, \dots, i\}$  and with  $w$  available seconds. Then,

- If  $i \notin O_{i,w}$ , then  $OPT(i, w) = OPT(i-1, w)$ , and
- If  $i \in O_{i,w}$ , then  $OPT(i, w) = v_i + OPT(i-1, w-w_i)$

Moreover, if  $w_i > w$ , then  $i \notin O_{i,w}$ , thus  $OPT(i, w) = OPT(i-1, w)$

KNAPSACK ( $n, w$ )

INITIALIZE THE ARRAY  $M[0 \dots n][0 \dots w]$   $O(nw)$

LET  $M[0][w] = 0 \quad \forall 0 \leq w \leq W$   $O(w)$

{ FOR  $i = 1 \dots n$   
 } FOR  $w = 0 \dots W$

$O(n \cdot w)$

```

    IF ( $w_i > w$ ) or ( $M[i-1][w] > v_i + M[i-1][w - w_i]$ )
         $M[i][w] = M[i-1][w]$ 
    ELSE:
         $M[i][w] = v_i + M[i-1][w - w_i]$ 
    RETURN M
  
```

We could run  $M = \text{KNAPSACK}(n, w)$ , and then  $M[n][w]$  is the value of the best solution

**FIND SOL (n, w)**

$M = \text{KNAPSACK}(n, w) \quad O(n \cdot w)$

$S = []$

$w = w$

$i = n$

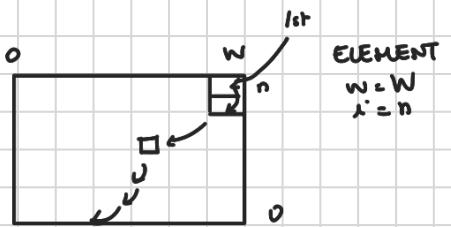
{ WHILE  $i > 0$ :

```

      IF  $M[i][w] == M[i-1][w]$ 
           $i -= 1$  // throw away i
      ELSE:
          S.append(i) // schedule i
           $w -= w_i$ 
           $i -= 1$ 
    
```

RETURN S

$\quad // \sum_{j \in S} v_j = M[n][w] \text{ and } \sum_{j \in S} w_j \leq w$



IF  $M[i][w] \neq M[i-1][w]$ :

S.append(i)  
 $w -= w_i$

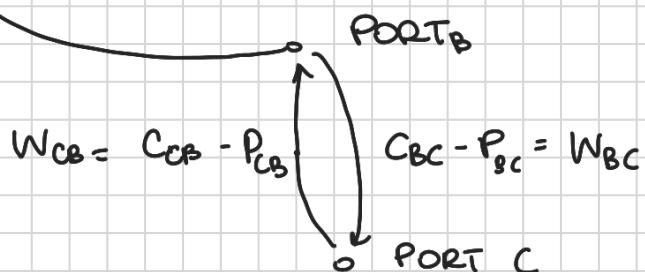
$i -= 1$

**SHORTEST PATHS ON GRAPHS WITH "NEGATIVE AND/OR POSITIVE" WEIGHTS.**

# cruise ship / taxi drive

PORT A

$$C_{AB} - P_{AB} = W_{AB}$$

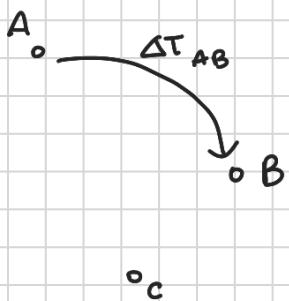


$$W_{AB} > 0 \quad C_{AB} > P_{AB}$$

$$W_{AB} = 0 \quad C_{AB} = P_{AB}$$

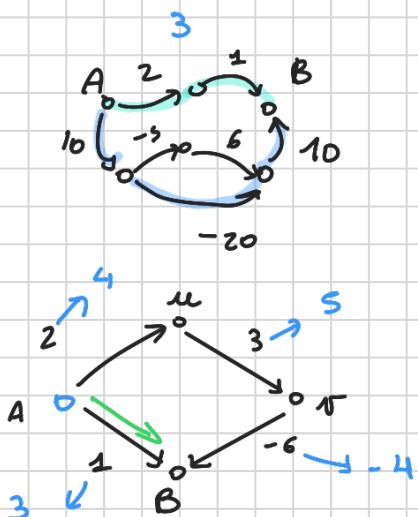
$$W_{AB} < 0 \quad C_{AB} < P_{AB}$$

## CHEMICAL REACTIONS



$V = \{A, B, C\}$  chemical compounds

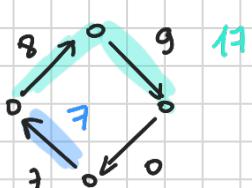
How to go from  $s$  to  $t$  with the smallest increase in temperature



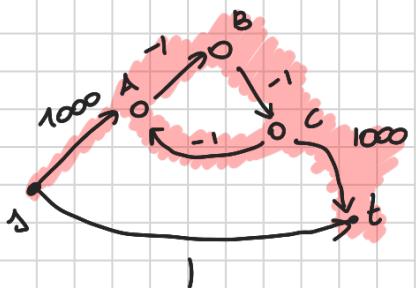
A TO B

DJIKSTRA DOESN'T WORK HERE

|   |   |
|---|---|
| A | 0 |
| B | 1 |
| u | 2 |



ADD  $\frac{-\min W_e}{e}$  to each edge  
(so that edge weights become non-neg.)



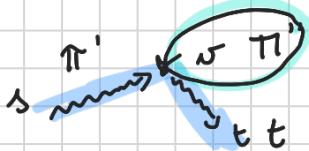
We assume that the graph has no negative cycles

L1 : IF  $G$  is a graph with no negative cycles, then  $\exists$  exists a shortest path from  $s$  to  $t$  that is simple (it has no node repetition), and that contains  $\leq n-1$  edges.

P: Let  $\pi$  be a  $s-t$  shortest path with minimal number of edges.

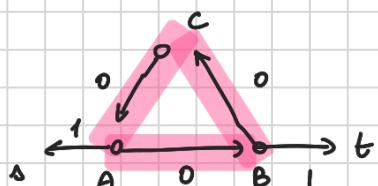
Suppose that  $\exists \pi'$  s.t.  $\pi = \pi' \cup \pi'' \cup \pi'''$

Since  $G$  contains no negative cycles, it must be that  $\cup \pi'''$  is a cycle having a positive total edge weight.



Consider now, the path  $\tilde{\pi} = \pi' \cup \pi''$ . This path has fewer edges than  $\pi$ . Moreover the total weight of  $\tilde{\pi}$  is no larger than the total weight of  $\pi$ . Since  $\pi$  has more edges than  $\tilde{\pi}$ , and since the total weight of  $\pi$  is not larger than that of  $\tilde{\pi}$ ,  $\pi$  cannot be a  $s-t$  shortest path with a minimal numbers of edges. CONTRADICTION.

Then no node is repeated in a shortest path with a minimal number of edges — since we have  $n$  nodes, such a path has at most  $n-1$  edges ■



$$w(s \xrightarrow{1} AB \xrightarrow{0} t) = 2$$

$$w(s \xrightarrow{2} ABC \xrightarrow{0} AB \xrightarrow{0} t) = 2$$

Let  $\text{OPT}(i, v)$  be the minimum cost/weight of a path from  $v$  to  $t$ , having at most  $i$  edges.

Let  $\pi$  be a path of  $i$  edges from  $v$  to  $t$ , having minimum cost  $\text{OPT}(i, v)$



$\pi$  has at most  $i$  edges

- If  $\pi$  has at most  $i-1$  edges, then  $\text{OPT}(i, v) = \text{OPT}(i-1, v)$
- If  $\pi$  has exactly  $i$  edges, then  $\exists w$  such that  $\pi$  starts from  $v$ , moves to  $w$ , and then progresses optimally to  $t$  using at most  $i-1$  edges. That is,

$$\text{OPT}(i, v) = c_{vw} + \text{OPT}(i-1, w)$$

This DP is called "the BELLMAN-FORD" algorithm.

$$\text{OPT}(0, t) = 0$$

$$\text{OPT}(0, v) = \infty \quad \forall v \neq t$$

L2 : If  $i > 0$ ,  $\text{OPT}(i, v) = \min_w (\text{OPT}(i-1, v), \min_w (c_{vw} + \text{OPT}(i-1, w)) \quad (v, w) \in E$

BELLMAN-FORD ( $G, s, t$ ) :

INITIALIZE  $M[0 \dots n-1][v \in V(G)]$

$M[0][t] = 0$

$M[0][v] = +\infty \quad \forall v \in V - \{t\}$

FOR  $i = 1 \dots n-1$ :

FOR  $v \in V(G)$ :

$M[i][v] = M[i-1][v]$

FOR  $w$  ST.  $(v, w) \in E(G)$ :

$$\delta = c_{vw} + M[i-1][w]$$

IF  $\delta < M[i][v]$ :

$$M[i][v] = \delta$$

RETURN  $M$  (or  $M[n-1][s]$ )

} RECURRENCE OF L2

THM: Bellman-Ford returns the shortest distance from  $s$  to  $t$  if the graph contains no negative cycles.

P: Apply L1 and L2.