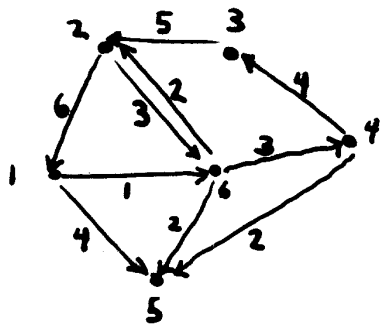
ALL PAIRS SHORTEST PATH PROBLEM

INPUT: A weighted directed graph $G = (V, E)$, $c(i,j)$ for all edges $(i,j) \in E$

OUTPUT FOR EVERY PAIR OF VERTICES (u,v) THE COST OF THE CHEAPEST PATH FROM u TO v.

EXAMPLE



* ASSUME ALL EDGES NOT PRESENT HAVE COST ∞

	1	2	3	4	5	6
1	0	3	8	4	3	1
2	6	0	10	6	5	3
3	11	5	0	11	10	8
4	15	9	4	0	14	11
5	∞	∞	∞	∞	0	∞
6	8	2	7	3	2	0

ALL PAIRS SHORTEST PATHS, CONT'd

STEP 1 For $0 \le k \le n$, $1 \le i, j \le n$

$A(k,i,j)$ = the cost of a cheapest path from i to j, from those paths involving only intermediate vertices in $\{1, 2, .., k\}$

$A(n,i,j)$ $1 \le i, j \le n$ are the values we are ultimately interested in.

STEP 2

$A(0, i, i) = 0$

$A(k=0, i, j) = c(i,j)$ $\quad i \neq j$

For $k > 0$:

$$A(k,i,j) = \min \begin{Bmatrix} A(k-1, i, j) \\ A(k-1, i, k) + A(k-1, k, j) \end{Bmatrix}$$

If cheapest path from i to j involving only $\{1..k\}$ does not involve vertex k, then cost is $A(k-1,i,j)$

If cheapest path does involve k, then path p looks like:

So cost is $A(k-1,i,k) + A(k-1,k,j)$

Step 4 computing an optimal solution

```
PRINTOPT (i j)
    K ← B'[i,j]
    IF K = 0 then
        put "edge from" i "to" j
    ELSE
        PRINTOPT(i,k)
        PRINTOPT (K,j)
    END IF
END PRINTOPT
```

IF i≠j THEN PRINTOPT(i,j) END IF

RUNTIME

1. step 3 $O(n^3)$
2. (Exercise) Step 2 runs in time $O(n)$

∴ RUNTIME $O(n^3)$

Step 3 Algorithm

All_Pairs_CP

```
For i=1..n do
    B[i,i] ← 0
    B'[i,i] ← 0
    For j=1..n such that j≠i do
        B[i,j] ← C(i,j)
        B'[i,j] ← 0
    END FOR
END FOR

FOR k=1..n do
    For i=1..n do
        For j=1..n do
            IF B[i,k] + B[k,j] < B[i,j] then
                B[i,j] ← B[i,k] + B[k,j]
                B'[i,j] ← k
            END IF
        END FOR
    END FOR
END FOR
```

In k^{th} iteration of outer loop,

$B[i,j] = A[k,i,j]$ and

$B'[i,j]$ is smallest numbered vertex k' s.t. there is a cheapest path from i to j involving only $\{1 \ldots k\}$ and NO vertex larger than k' on path.

CORRECTNESS OF ALL_PAIRS_CP:

We want to prove the following Lemma

Lemma $\forall k, i, j \quad 0 \le k \le n \quad 1 \le i, j \le n$ after the k^{th} execution of the body of the outer for-loop, the following holds:

$B[i,j] = A(k,i,j)$

$B'[i,j]$ is the smallest number such that $\exists$ a path p from i to j with only intermediate vertices in $\{1 \ldots B'[i,j]\}$ and such that p has cost $A(k,i,j)$

Proof by induction on k

ACTIVITY SELECTION WITH PROFITS

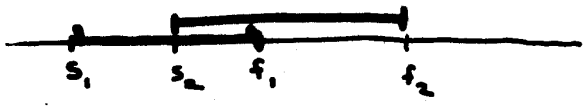
Input $(s_1, f_1, g_1) \ldots (s_n, f_n, g_n)$

start time, finish time, profit

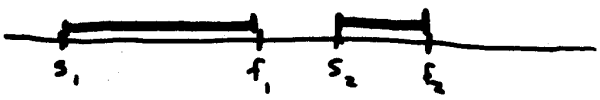
Two activities $i, j \in [1, \ldots n]$, $i \neq j$ are **compatible** if

$f_i \leq s_j$ or $f_j \leq s_i$

ie:



← not compatible



← compatible

A feasible schedule $S \subseteq \{1, 2, \ldots n\}$ is a set such that every two distinct $i, j \in S$ are compatible.

The profit of S, $P(S) = \sum_{i \in S} g_i$

Output A feasible schedule with maximal profit.

- Assume without loss of generality $s_i < f_i$ $\forall i$
- Assume activities are sorted so that $f_1 \leq f_2 \leq \ldots \leq f_n$
- Let distinct finish times be $u_1 < u_2 < \ldots < u_k$, $k \leq n$
  Let $u_0 = 0$
- For each i, Let $H(i)$ be largest $\ell \in \{0, 1, \ldots, k\}$ such that $u_\ell \leq s_i$

} Precomputation

Runtime $\sim O(n \log n)$

DYNAMIC PROGRAMMING ALGORITHM FOR ACTIVITY SELECTION WITH PROFITS

STEP 1 FOR $0 \le j \le k$, let

$A^*(j)$ = largest profit attainable by feasibly scheduling activities that all finish by time u_j

We want to compute $A(k)$

STEP 2 Recurrence for A

$A(0) = 0$

For $1 \le j \le K$

$$A(j) = \max \left\{ A(j-1), \max_{1 \le i \le n} \{ g_i + A(H(i)) \mid f_i = u_j \} \right\}$$

$A = A^*$:

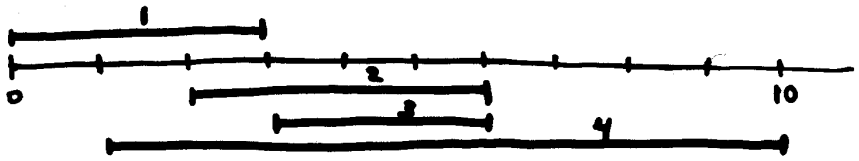
- Clearly $A^*(0) = A(0) = 0$
- For $1 \le j \le k$, consider a best possible S where all activities finish by u_j.

  Case 1 No activity in S finishes at exactly time u_j

  Then $P(S) = A^*(j-1) = A(j-1)$

  Case 2 otherwise exactly one activity in S finishes at time u_j. If this is activity i then all other activities in S finish by s_i and hence by time $u_{H(i)}$, so S has profit $g_i + A^*(H(i))$.

Example



Activity i	1	2	3	4
Start s_i	0	2	3	2
Finish f_i	3	6	6	10
Profit g_i	20	30	20	30
$H(i)$	0	0	1	0

j	0	1	2	3
u_j	0	3	6	10
$A(j)$	0	20	40	40

$A(0) = 0$

$A(1) = \max\{0,\ 20 + A(H(1))\} = 20$

$A(2) = \max\{20,\ 30 + A(H(2)),\ 20 + A(H(3))\} = 40$

$A(3) = \max\{40,\ 30 + A(H(4))\} = 40$