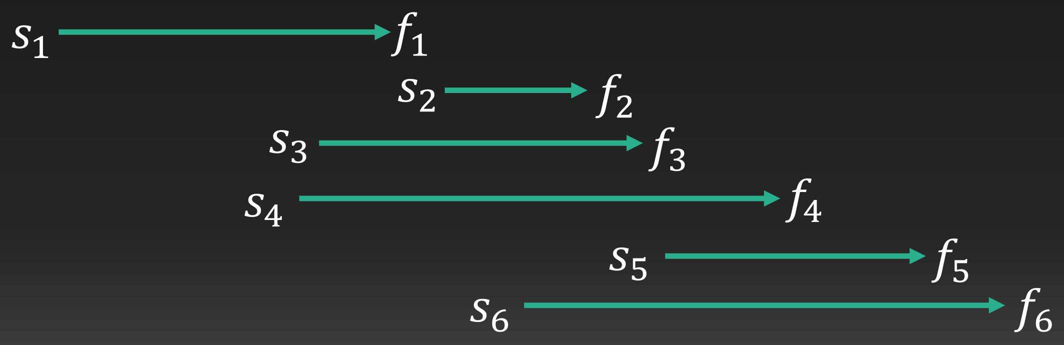
CS 4800: Algorithms & Data

Lecture 12

February 21, 2017

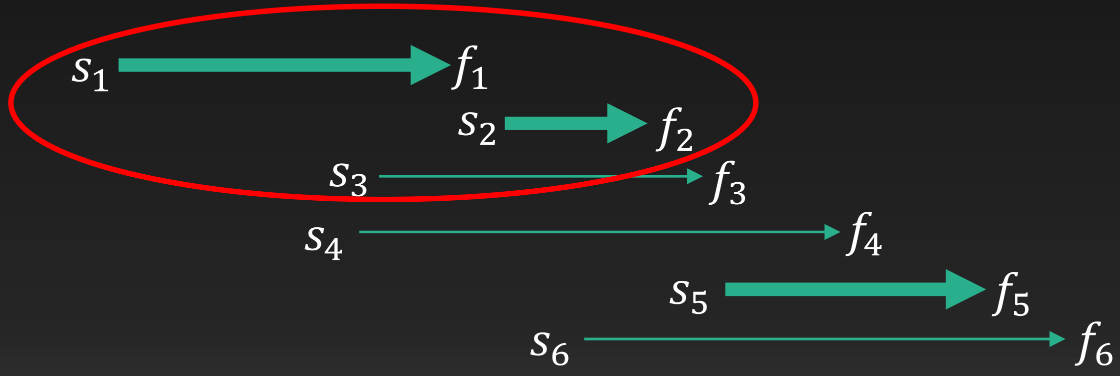
Problem statement

- n activities
- Start times : $s_1, s_2, \ldots, s_n$
- Finish times: $f_1 \leq f_2 \leq \cdots \leq f_n$ (sorted)
- Find largest subset of activities that are compatible



Dynamic Programming

- Best(i): Maximum # compatible activities finishing by f_i
- Optimal substructure: consider activities comprising Best(i) and its prefixes.



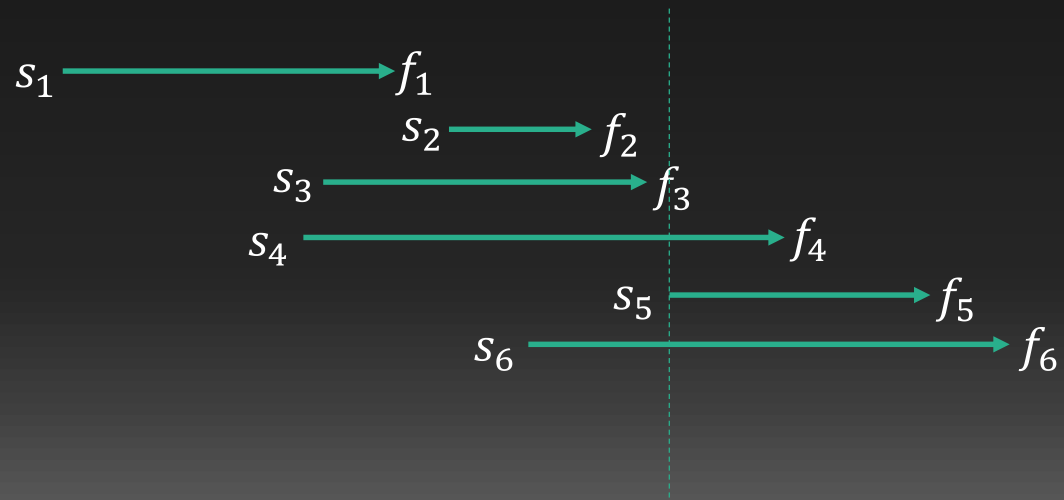
- Claim. The prefix is optimal.

Recursive relation

Not pick i

- Either pick activity i or not
- $$Best(i) = \max\begin{cases} Best(i-1) \\ 1 + Best(j)\ where\ j = \max k\ s.t\ f_k \le s_i \end{cases}$$

Pick i

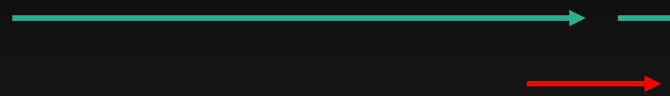


Dynamic Programming

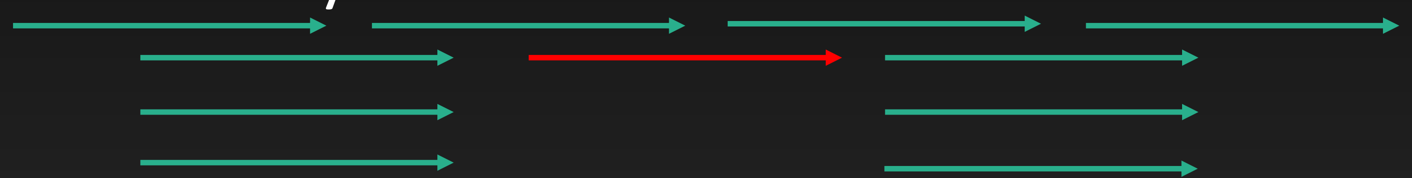
- $Best(0) \leftarrow 0$
- $f_0 \leftarrow -\infty$
- For i from 1 to n
  - Use binary search to find max j s.t. $f_j \leq s_i$
  - Best(i) = max(Best(i-1), 1+Best(j))

Various greedy rules

- Pick shortest activity



- Pick activity with fewest conflicts



- Pick activity first to start

- Pick activity first to finish

Exchange argument

Claim: First activity to finish is part of some optimal solution.

Proof.

Consider an optimal solution X that does not include activity 1.

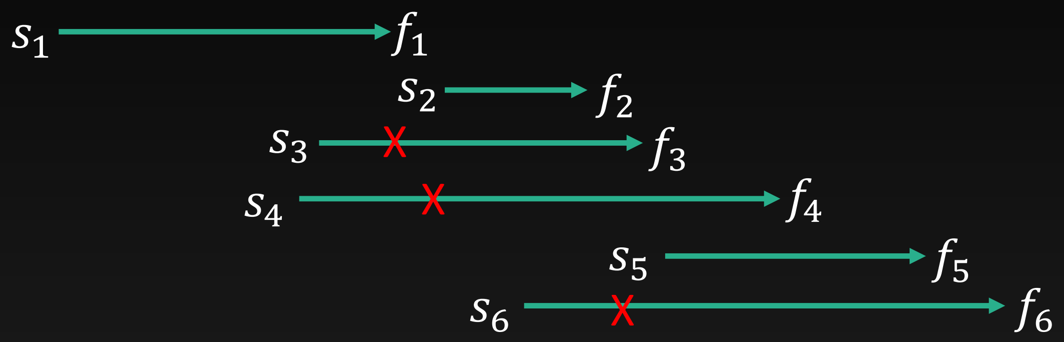
Let i be the first activity to finish in X.

Because act. 1 finishes before i, act. 1 does not conflict with any activity in $X \setminus \{i\}$

Therefore, $\mathrm{X}' = X \setminus \{i\} \cup \{1\}$ is also conflict-free.

X’ has the same size of X and thus, it is also optimal.

Greedy algorithm



Find first activity to finish. Add to solution.
Remove conflicting activities.

Repeat.

Greedy algorithm

- $count \leftarrow 1$ // number of activities we pick
- $X[count] \leftarrow 1$ // X[.]: IDs of activities we pick
- For i from 2 to n
  - If $S[i] \geq F\big[X[count]\big]$
    - $count \leftarrow count + 1$
    - $X[count] \leftarrow i$
- Return $X[1 \dots count]$

Greedy is optimal

Induction hypothesis: Greedy is optimal for any instance of size n.
Base case: Greedy is optimal for n=1
Inductive case: Assume Greedy is optimal for $n < k$. Will prove for n=k.
By Claim, activity 1 belongs to some optimal solution. Thus, the best solution that includes 1 is also optimal.

Greedy picks 1 and then perform greedy on the set of activities not conflicting with 1 (a sub-instance of size < k).

By induction, greedy picks an optimal solution for the sub-instance i.e. it finds the best solution containing 1.

Therefore, greedy also finds an optimal solution for n=k.