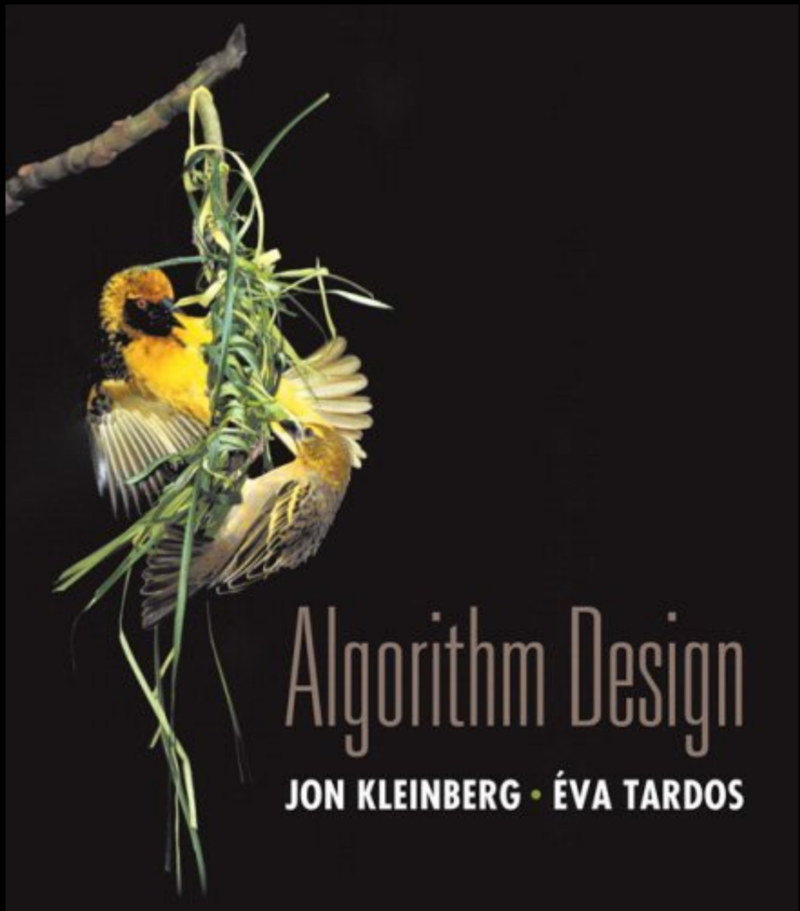


Chapter 4

Greedy Algorithms

Slides by Kevin Wayne.

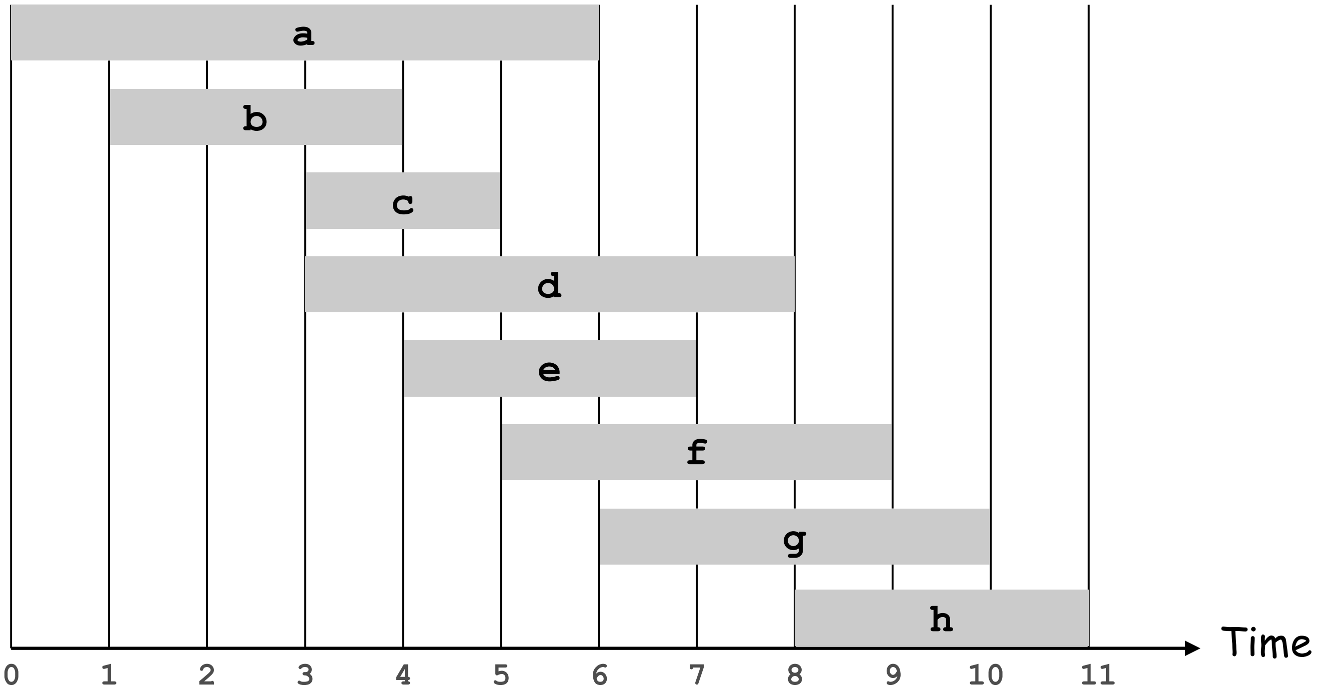
4.1 Interval Scheduling

Section 4.1 del libro di testo KT

Interval Scheduling

Interval scheduling. Instance $\mathcal{I} = \{I_1, \ldots, I_n\}$

- Job j starts at s_j and finishes at f_j → $I_j = (s_j, f_j)$
- Two jobs are compatible if they don't **overlap**.
- **Goal**: find maximum subset A of mutually compatible jobs.



Interval Scheduling: Greedy Algorithms

Greedy template:

1. Consider all jobs in some fixed **order**. Set **A**:= *Empty-Set*
2. For each **j=1,…,n**, IF I_J is **compatible** with all jobs in **A** **THEN include** I_J in **A**

Possible Ordering Criteria:

- [Earliest start time] Consider jobs in ascending order of s_j.
- [Earliest finish time] Consider jobs in ascending order of f_j.
- [Shortest interval] Consider jobs in ascending order of $f_j - s_j$.
- [Fewest conflicts] For each job **j**, count the *number* of **conflicting jobs** c_j. Schedule in **ascending** order of c_j.

Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.

Greedy 1: **Earliest Start Time**

Does it work ?

1st Step of **Problem Solvers**: Find bad situations for the Algorithm.

Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.

Greedy 1: **Earliest** Start Time

counterexample for earliest start time

Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.

Greedy 2: **Shortest Interval**

Bad Situations ???

Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.

Greedy 2: **Shortest Interval**

counterexample for shortest interval

Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.

Greedy 3: **FEWEST** CONFLICTS

IDEA: Use GRAPH MODELLING;

Which is the problem in terms of GRAPHS?

Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.

Greedy 3: **FEWEST** CONFLICTS

counterexample for fewest conflicts

Interval Scheduling: Greedy Algorithm

Greedy 4. Consider jobs in increasing order of finish time. Take each job provided it's compatible with the ones already taken.

set of jobs selected

```
Sort jobs by finish times so that f1 ≤ f2 ≤ ... ≤ fn.

A ← φ
for j = 1 to n {
   if (job j compatible with A)
      A ← A ∪ {j}
}
return A
```

Implementation. TIME= O(n log n).

- Remember job j^* that was added last to A.
- Job j is compatible with A iff $s_j \geq f_{j^*}$.

- Let Let $i_1, i_2, \ldots i_k$ denote set *A* of jobs selected by ***Greedy***.
- Let $j_1, j_2, \ldots j_m$ denote set of jobs in any solution (ordered w.r.t. finish time).

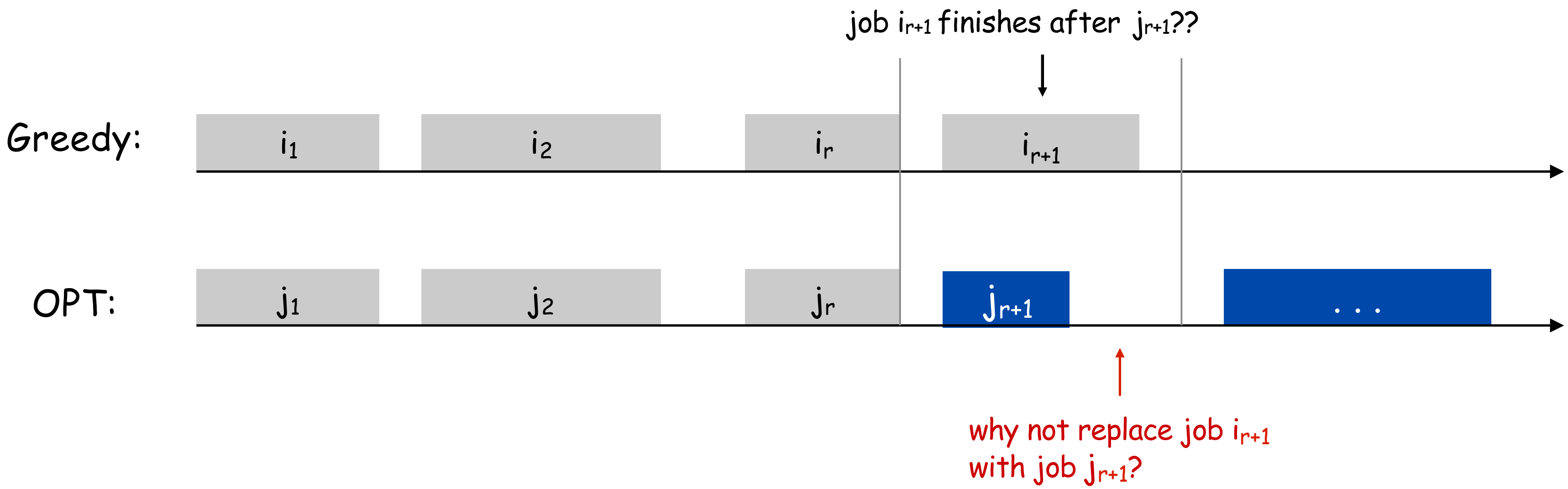
Lemma 1 (*Greedy Stays Ahead*). For any $r = 1, \ldots, k$ it holds

$$f(i_r) \le f(j_r)$$

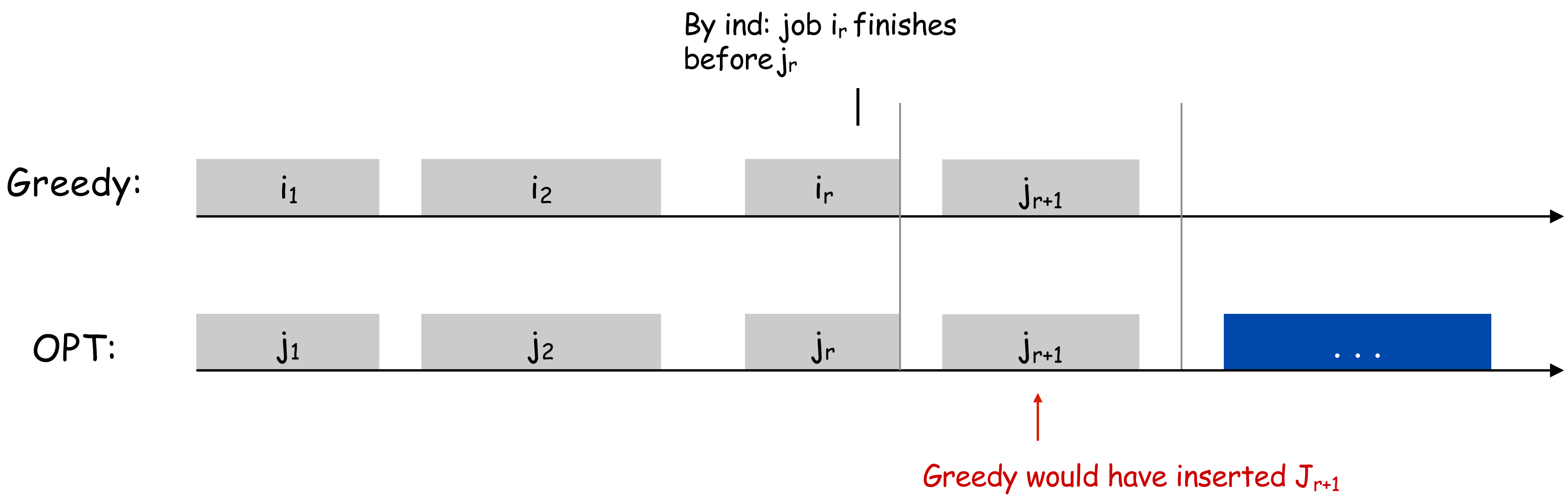
Proof. By induction on r**.** $r = 1$ **is trivial.**

....

Interval Scheduling: Analysis



Interval Scheduling: Analysis



Interval Scheduling

EXERCISE I:

Prove that the Greedy Algorithm based on the **earliest finish time** is **optimal**

Hint: Use Lemma 1 (Greedy stays ahead)

So, we have an **optimal** greedy algorithm for **Interval Scheduling** that run in time **O(n log n)**

Domande di Autovalutazione

- Che size $|I|$ ha la generica istanza $I=\{I_1 \ldots, I_n\}$ del problema **Interval Scheduling**?
- La size $|I|$ come dipende dai valori di starting time s_i e finish time f_i degli intervalli?
- Il tempo dell'algoritmo **Greedy** ottimale basato sul finish time dipende dai valori s_i ed f_i ? In che modo? In quale punto del codice?
- Nel confronto con una generica soluzione ottima $J = \{j_1, j_2, \ldots J_m\}$, il **Lemma 1** garantisce che la size r della soluzione greedy A è tale che $m <= r$? Perché? (Esercizio I)