Algorithm Theory

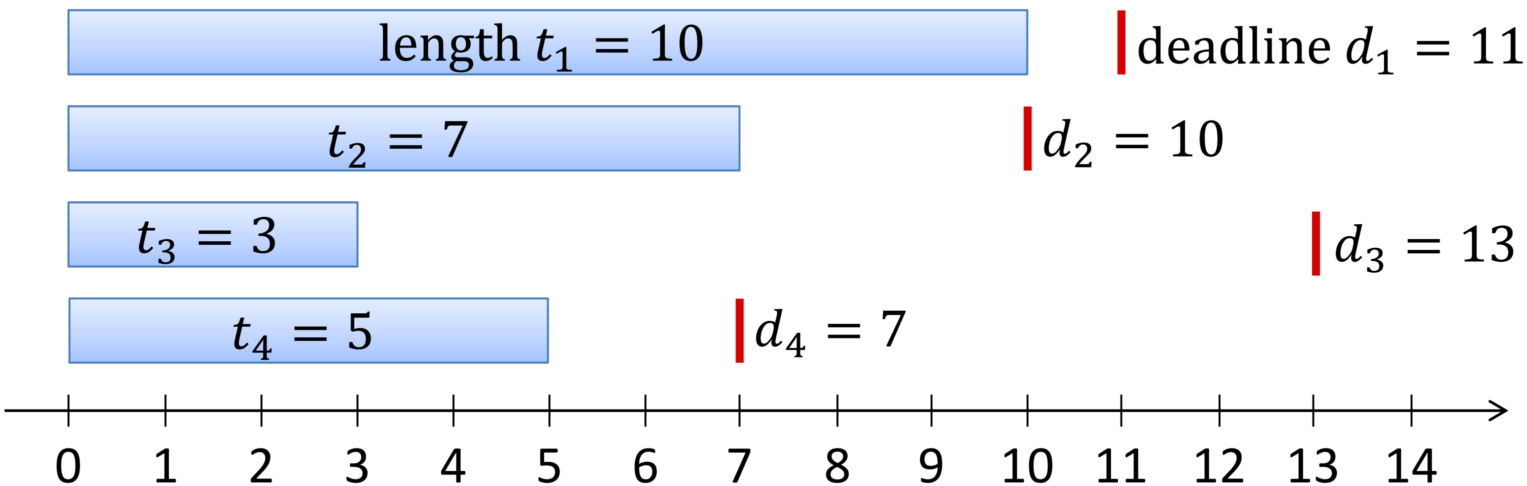
Chapter 2
Greedy Algorithms

Part III:
Exchange Arguments

Fabian Kuhn

Back to Scheduling

- Given: n requests / jobs with deadlines:



- Goal: schedule all jobs with minimum lateness L
  - Schedule: $s(i)$, $f(i)$: start and finishing times of request i
    Note: $f(i) = s(i) + t_i$
  - Lateness L_i of request i : $L_i := \max\{0, f(i) - d_i\}$
- Lateness $L := \max\left\{0, \max_i\{f(i) - d_i\}\right\} = \max_i L_i$
  - largest amount of time by which some job finishes late
- Many other natural objective functions possible…

Greedy Algorithm?

Schedule jobs in order of increasing length?

- Ignores deadlines: seems too simplistic…
- E.g.:

$t_1 = 10$ deadline $d_1 = 10$

$t_2 = 2$ … $d_2 = 100$

Schedule: $t_2 = 2$ $t_1 = 10$

Schedule by increasing slack time?

- Should be concerned about slack time: $d_i - t_i$

$t_1 = 10$ deadline $d_1 = 10$

$t_2 = 2$ $d_2 = 3$

Schedule: $t_1 = 10$ $t_2 = 2$

Greedy Algorithm

Schedule by earliest deadline?

- Schedule in increasing order of d_i
- Ignores lengths of jobs: too simplistic?
- Earliest deadline is optimal!

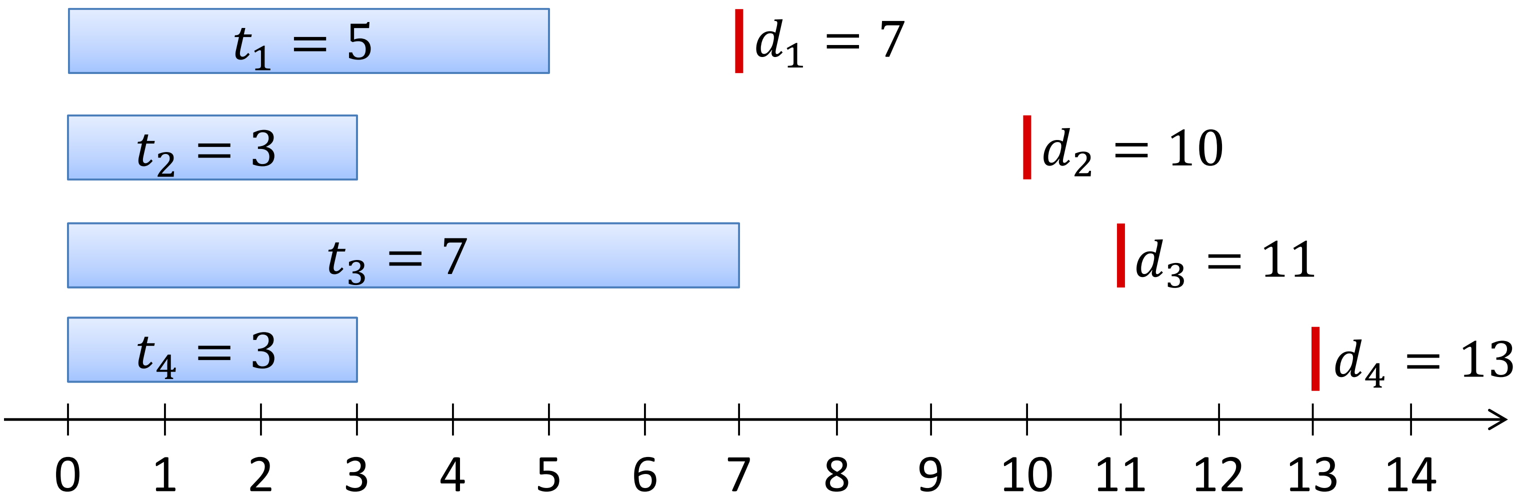
Algorithm:

- Assume jobs are reordered such that $d_1 \leq d_2 \leq \cdots \leq d_n$
- Start/finishing times:
  - First job starts at time $s(1) = 0$
  - Duration of job i is t_i: $f(i) = s(i) + t_i$
  - No gaps between jobs: $s(i+1) = f(i)$

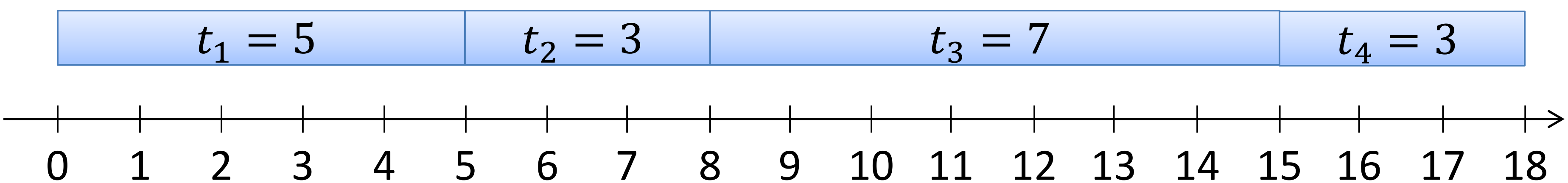
  (idle time: gaps in a schedule → alg. gives schedule with no idle time)

Example

Jobs ordered by deadline:



Schedule:

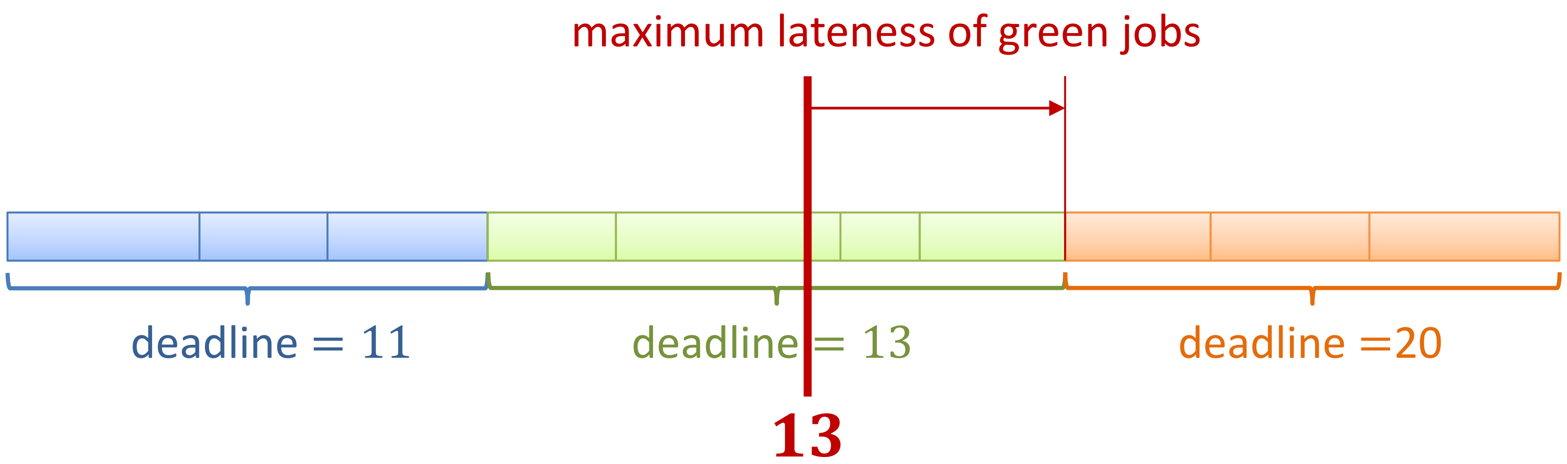


Lateness: job 1: 0, job 2: 0, job 3: 4, job 4: 5

Basic Facts

1. There is an optimal schedule with no idle time
   – Can just schedule jobs earlier…

2. Inversion: Job i scheduled before job j if $d_i > d_j$
   Schedules with no inversions have the same maximum lateness

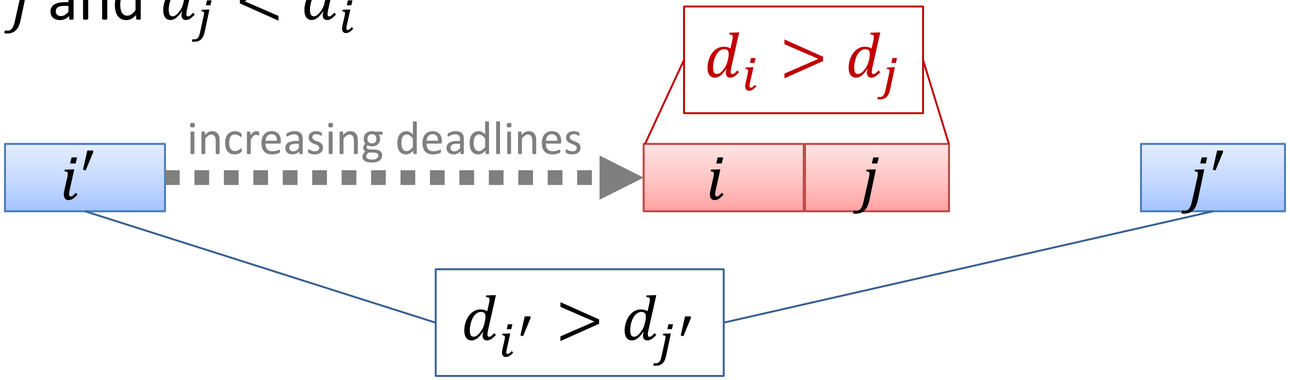


Earliest Deadline is Optimal

Theorem:
There is an optimal schedule $\mathcal{O}$ with no inversions and no idle time.

Proof:

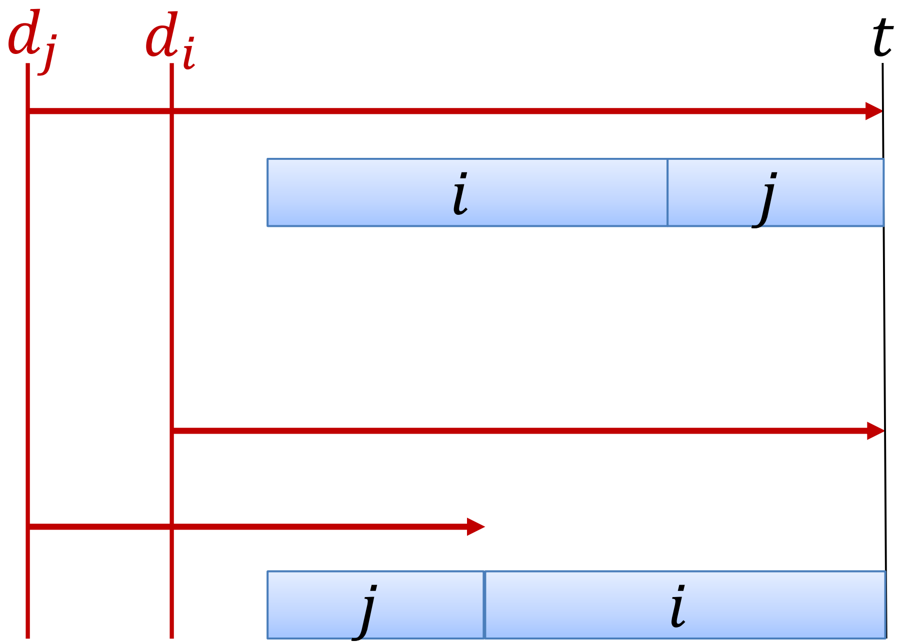
- Consider some schedule $\mathcal{O}'$ with no idle time
- If $\mathcal{O}'$ has inversions, $\exists$ pair (i, j), s.t. i is scheduled immediately before j and $d_j < d_i$



- Claim: Swapping i and j gives a schedule with
  1. Fewer inversions
  2. Maximum lateness no larger than in $\mathcal{O}'$

Earliest Deadline is Optimal

Claim: Swapping i and j: maximum lateness no larger than in $\mathcal{O}'$



Lateness $L_j = \max\{0, t - d_j\}$

Max. lateness after swap:

$$L_i' = \max\{0, t - d_i\} \leq L_j$$

$$L_j' = \max\{0, L_j - t_i\} \leq L_j$$

Exchange Argument

- General approach that often works to analyze greedy algorithms

- Start with any solution
- Define basic exchange step that allows to transform solution into a new solution that is not worse
- Show that exchange step move solution closer to the solution produced by the greedy algorithm
- Number of exchange steps to reach greedy solution should be finite…

Another Exchange Argument Example

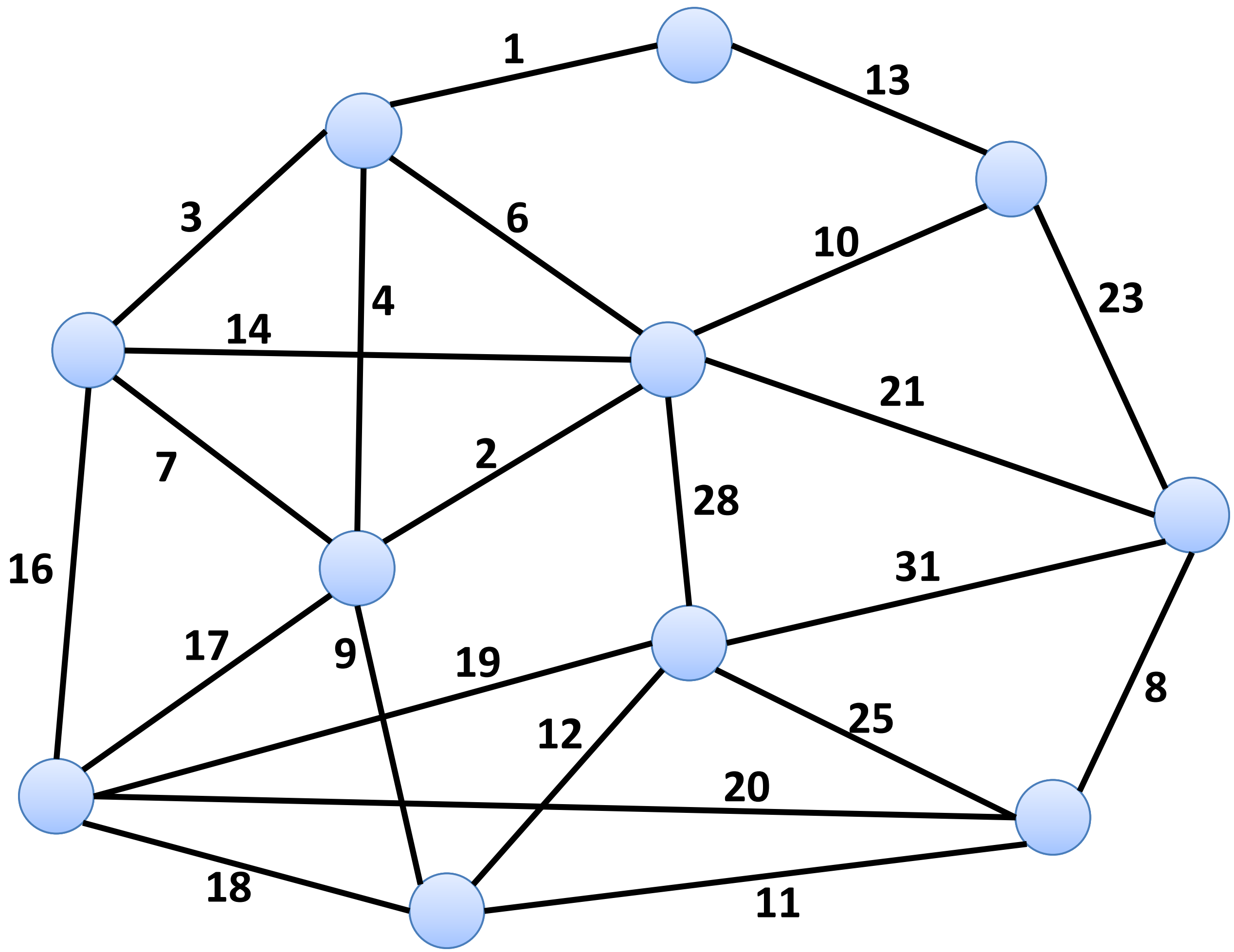
- **Minimum spanning tree (MST)** problem
  - Classic graph-theoretic optimization problem

- **Given**: weighted graph
- **Goal**: spanning tree with min. total weight

- Several greedy algorithms work

- Kruskal's algorithm:
  - Start with empty edge set
  - As long as we do not have a spanning tree:
    add minimum weight edge that doesn't close a cycle

Kruskal Algorithm: Example



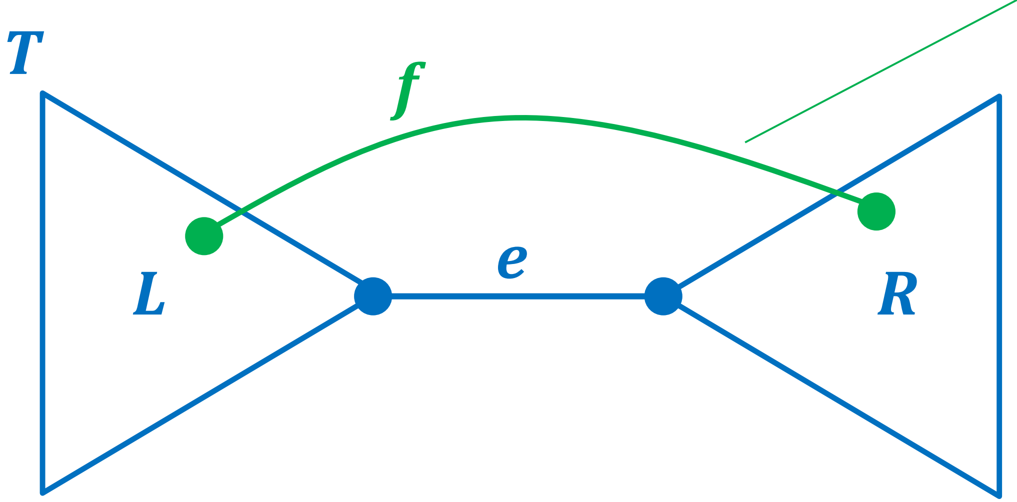
Kruskal is Optimal

- Basic exchange step: swap to edges to get from tree T to tree T_K
  - Swap out edge not in Kruskal tree T_K, swap in edge in Kruskal tree T_K
  - Swapping does not increase total weight
- For simplicity, assume, weights are unique

T : any spanning tree

T_K: Kruskal tree

Assume that $T \neq T_K \implies \exists e \in T \setminus T_K$



f is the lightest edge connecting L and R

$\implies f \in T_K \setminus T$
$\implies w(f) < w(e)$

$T' := T \setminus \{e\} \cup \{f\}$ is a spanning tree of smaller total weight than T.