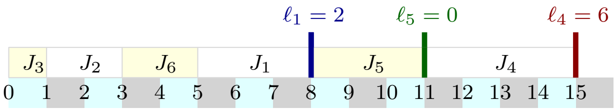
19.4
Scheduling to Minimize Lateness

Scheduling to Minimize Lateness

1. Given jobs $J_1, J_2, \ldots, J_n$ with deadlines and processing times to be scheduled on a single resource.
2. If a job i starts at time s_i then it will finish at time $f_i = s_i + t_i$, where t_i is its processing time. d_i: deadline.
3. The lateness of a job is $\ell_i = \max(0, f_i - d_i)$.
4. Schedule all jobs such that $L = \max \ell_i$ is minimized.

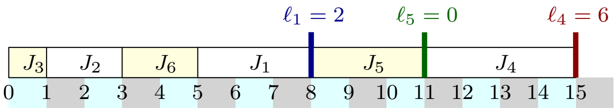
	J_1	J_2	J_3	J_4	J_5	J_6
t_i	3	2	1	4	3	2
d_i	6	8	9	9	14	15



Scheduling to Minimize Lateness

1. Given jobs $J_1, J_2, \ldots, J_n$ with deadlines and processing times to be scheduled on a single resource.
2. If a job i starts at time s_i then it will finish at time $f_i = s_i + t_i$, where t_i is its processing time. d_i: deadline.
3. The lateness of a job is $\ell_i = \max(0, f_i - d_i)$.
4. Schedule all jobs such that $L = \max \ell_i$ is minimized.

	J_1	J_2	J_3	J_4	J_5	J_6
t_i	3	2	1	4	3	2
d_i	6	8	9	9	14	15



Greedy Template

```
Initially R is the set of all requests
curr_time = 0
max_lateness = 0
while R is not empty do
    choose i ∈ R
    curr_time = curr_time + t_i
    if (curr_time > d_i) then
        max_lateness = max(curr_time − d_i, max_lateness)

return max_lateness
```

Main task: Decide the order in which to process jobs in R

Greedy Template

```
Initially R is the set of all requests
curr_time = 0
max_lateness = 0
while R is not empty do
    choose i ∈ R
    curr_time = curr_time + t_i
    if (curr_time > d_i) then
        max_lateness = max(curr_time − d_i, max_lateness)

return max_lateness
```

Main task: Decide the order in which to process jobs in R

Three Algorithms

1. Shortest job first — sort according to t_i.
2. Shortest slack first — sort according to $d_i - t_i$.
3. EDF = Earliest deadline first — sort according to d_i.

Counter examples for first two: exercise

Three Algorithms

1. Shortest job first — sort according to t_i.
2. Shortest slack first — sort according to $d_i - t_i$.
3. EDF = Earliest deadline first — sort according to d_i.

Counter examples for first two: exercise

Earliest Deadline First

Theorem 19.1.

Greedy with EDF *rule minimizes maximum lateness.*

Proof via an exchange argument.

Idle time: time during which machine is not working.

Lemma 19.2.

If there is a feasible schedule then there is one with no idle time before all jobs are finished.

Earliest Deadline First

Theorem 19.1.

Greedy with EDF *rule minimizes maximum lateness.*

Proof via an exchange argument.

Idle time: time during which machine is not working.

Lemma 19.2.

If there is a feasible schedule then there is one with no idle time before all jobs are finished.

Earliest Deadline First

Theorem 19.1.

Greedy with EDF *rule minimizes maximum lateness.*

Proof via an exchange argument.

Idle time: time during which machine is not working.

Lemma 19.2.

If there is a feasible schedule then there is one with no idle time before all jobs are finished.

Earliest Deadline First

Theorem 19.1.

Greedy with EDF *rule minimizes maximum lateness.*

Proof via an exchange argument.

Idle time: time during which machine is not working.

Lemma 19.2.

If there is a feasible schedule then there is one with no idle time before all jobs are finished.

Inversions

EDF = Earliest Deadline First

Assume jobs are sorted such that $d_1 \leq d_2 \leq \ldots \leq d_n$. Hence EDF schedules them in this order.

Definition 19.3.

A schedule S is said to have an inversion if there are jobs i and j such that S schedules i before j, but $d_i > d_j$.

Claim 19.4.

If a schedule S has an inversion then there is an inversion between two adjacent scheduled jobs.

Proof: exercise.

Inversions

EDF = Earliest Deadline First

Assume jobs are sorted such that $d_1 \leq d_2 \leq \ldots \leq d_n$. Hence EDF schedules them in this order.

Definition 19.3.

A schedule S is said to have an inversion if there are jobs i and j such that S schedules i before j, but $d_i > d_j$.

Claim 19.4.

If a schedule S has an inversion then there is an inversion between two adjacent scheduled jobs.

Proof: exercise.

Proof sketch of Optimality of EDF

- Let S be an optimum schedule with smallest number of inversions.
- If S has no inversions then this is same as EDF and we are done.
- Else S has two adjacent jobs i and j with $d_i > d_j$.
- Swap positions of i and j to obtain a new schedule S'

Claim 19.5.

Maximum lateness of S' is no more than that of S. And S' has strictly fewer inversions than S.

THE END

...

(for now)