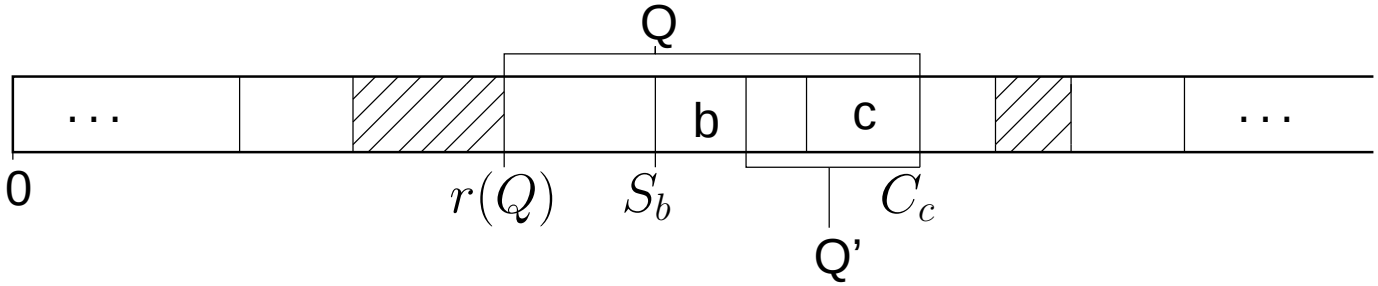
Single machine models: Maximum Lateness

Approximation ratio for EDD for problem $1|r_j, d_j < 0|L_{max}$

- structure of a schedule



- interference job b: last scheduled job from Q with $q_b < q_c$
- Lemma: For the objective value $L_{max}(EDD)$ of an EDD schedule we have: (Proofs were asked as exercise)
  1. $L_{max}(EDD) - L^*_{max} < q_c$
  2. $L_{max}(EDD) - L^*_{max} < p_b$
- Theorem: EDD is 2-approximation algorithm for $1|r_j, d_j < 0|L_{max}$

Single machine models: Maximum Lateness

Approximation ratio for EDD for problem $1|r_j, d_j < 0|L_{max}$

- Remarks:
  - EDD is also a 2-approximation for $1|prec, r_j, d_j < 0|L_{max}$ (uses modified release and due dates)
  - by an iteration technique the approximation factor can be reduced to $3/2$

Single machine models: Maximum Lateness

Enumerative methods for problem $1|r_j|L_{max}$

- we again will use head-body-tail notation
- Simple branch and bound method:
  - branch on level i of the search tree by selecting a job to be scheduled on position i
  - if in a node of the search tree on level i the set of already scheduled jobs is denoted by S and the finishing time of the jobs from S by t, for position i we only have to consider jobs k with

$$r_k < \min_{j \notin S}(\max\{t, r_j\} + p_j)$$

  - lower bound: solve for remaining jobs $1|r_j, pmtn|L_{max}$
  - search strategy: depth first search + selecting next job via lower bound

Single machine models: Maximum Lateness

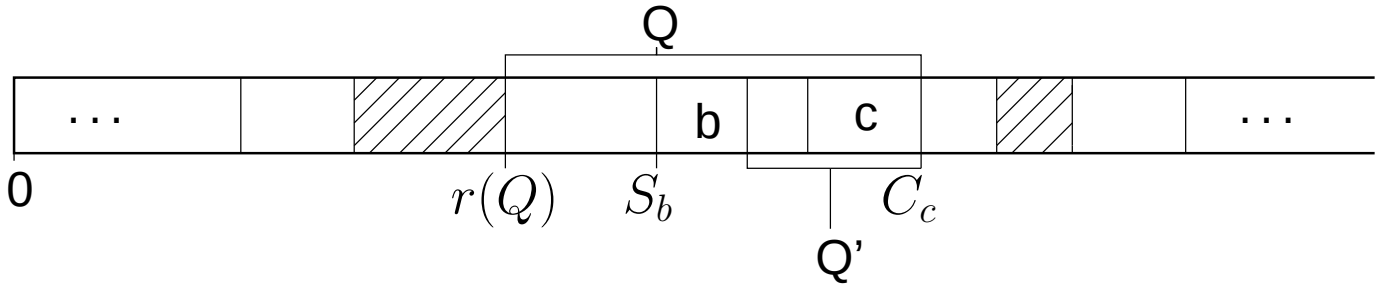
Advanced b&b-methods for problem $1|r_j|L_{max}$

- node of search tree = restricted instance
- restrictions = set of precedence constraints
- branching = adding precedence constraints between certain pairs of jobs
- after adding precedence constraints, modify release and due dates
- apply EDD to instance given in a node
  - critical sequence has no interference job: EDD solves instance optimal
    $\longrightarrow$ backtrack
  - critical sequence has an interference job: branch

Single machine models: Maximum Lateness

Advanced b&b-methods for problem $1|r_j|L_{max}$ (cont.)

branching, given set Q, critical job c, interference job b, and set Q' of jobs from Q following b

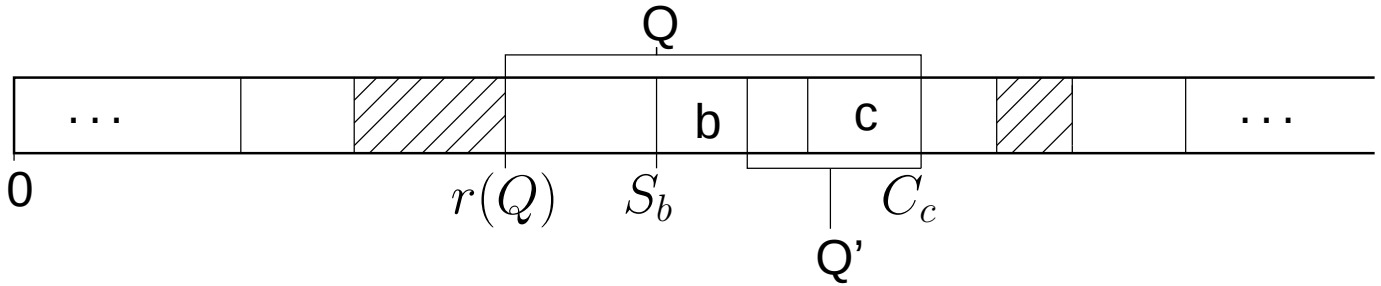


- $L_{max} = S_b + p_b + p(Q') + q(Q') < r(Q') + p_b + p(Q') + q(Q')$
- if b is scheduled between jobs of Q' the value is at least $r(Q') + p_b + p(Q') + q(Q')$; i.e. worse than the current schedule

Single machine models: Maximum Lateness

Advanced b&b-methods for problem $1|r_j|L_{max}$ (cont.)

branching, given set Q, critical job c, interference job b, and set Q' of jobs from Q following b



- $L_{max} = S_b + p_b + p(Q') + q(Q') < r(Q') + p_b + p(Q') + q(Q')$
- if b is scheduled between jobs of Q' the value is at least $r(Q') + p_b + p(Q') + q(Q')$; i.e. worse than the current schedule
- branch by adding either $b \rightarrow Q'$ or $Q' \rightarrow b$

Single machine models: Maximum Lateness

Advanced b&b-methods for problem $1|r_j|L_{max}$ (cont.)

- lower bounds in a node: maximum of
  - lower bound of parent node
  - $r(Q') + p(Q') + q(Q')$
  - $r(Q' \cup \{b\}) + p(Q' \cup \{b\}) + q(Q' \cup \{b\})$

  using the modified release and due dates
- upper bound UB: best value of the EDD schedules
- discard a node if lower bound $\geq UB$
- search strategy: select node with minimum lower bound

Single machine models: Maximum Lateness

Advanced b&b-methods for problem $1|r_j|L_{max}$ (cont.)

- speed up possibility:
  - let $k \notin Q' \cup \{b\}$ with $r(Q') + p_k + p(Q') + q(Q') \geq UB$
  - if $r(Q') + p(Q') + p_k + q_k \geq UB$ then add $k \rightarrow Q'$
  - if $r_k + p_k + p(Q') + q(Q') \geq UB$ then add $Q' \rightarrow k$

Single machine models: Number of Tardy Jobs

Problem $1||\sum U_j$:

- Structure of an optimal schedule:
  - set S_1 of jobs meeting their due dates
  - set S_2 of jobs being late
  - jobs of S_1 are scheduled before jobs from S_2
  - jobs from S_1 are scheduled in EDD order
  - jobs from S_2 are scheduled in an arbitrary order
- Result: a partition of the set of jobs into sets S_1 and S_2 is sufficient to describe a solution

Single machine models: Number of Tardy Jobs

Algorithm for $1||\sum U_j$ (Moore-Hodgson)

1. enumerate jobs such that $d_1 \leq \ldots \leq d_n$;
2. $S_1 := \emptyset;\ t := 0$;
3. FOR j:=1 TO n DO
4. $\quad S_1 := S_1 \cup \{j\}; t := t + p_j$;
5. $\quad$ IF $t > d_j$ THEN
6. $\quad\quad$ Find job k with largest p_k value in S_1;
7. $\quad\quad S_1 := S_1 \setminus \{k\}; t := t - p_k$;
8. $\quad$ END
9. END

Single machine models: Number of Tardy Jobs

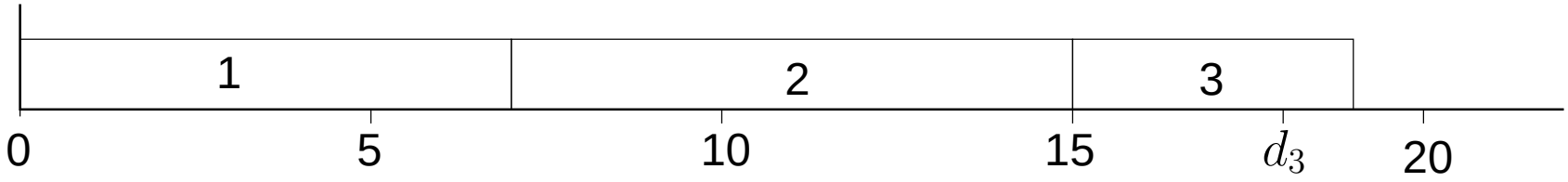
Remarks on Moore-Hodgson

- Principle: schedule jobs in order of increasing due dates and always when a job gets late, remove the job with largest processing time; all removed jobs are late
- complexity: $O(n \log(n))$
- Example: $n = 5;\ p = (7, 8, 4, 6, 6);\ d = (9, 17, 18, 19, 21)$

Single machine models: Number of Tardy Jobs

Remarks on Moore-Hodgson

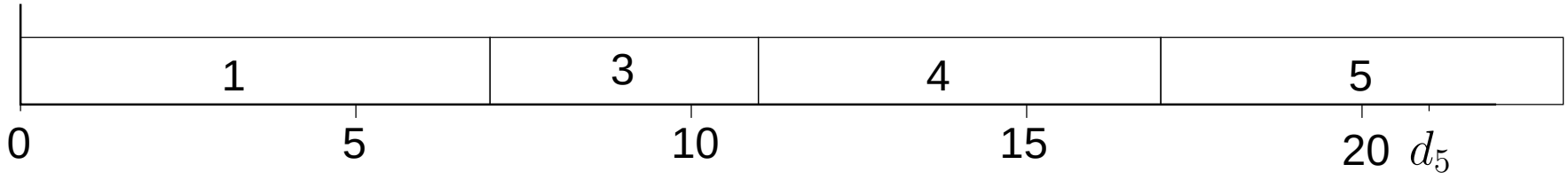
- Principle: schedule jobs in order of increasing due dates and always when a job gets late, remove the job with largest processing time; all removed jobs are late
- complexity: $O(n \log(n))$
- Example: $n = 5;\ p = (7, 8, 4, 6, 6);\ d = (9, 17, 18, 19, 21)$



Single machine models: Number of Tardy Jobs

Remarks on Moore-Hodgson

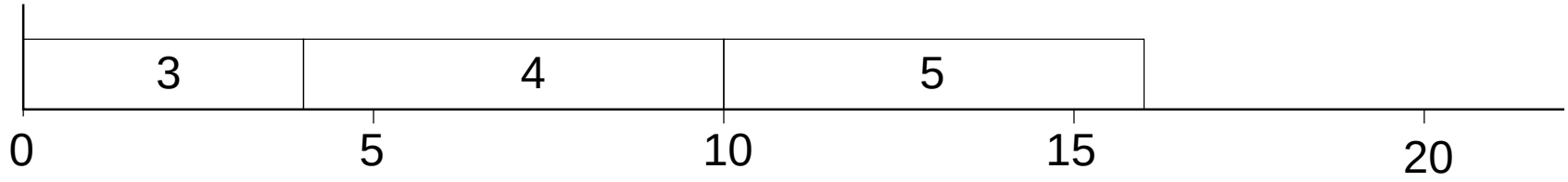
- Principle: schedule jobs in order of increasing due dates and always when a job gets late, remove the job with largest processing time; all removed jobs are late
- complexity: $O(n \log(n))$
- Example: $n = 5;\ p = (7, 8, 4, 6, 6);\ d = (9, 17, 18, 19, 21)$



Single machine models: Number of Tardy Jobs

Remarks on Moore-Hodgson

- Principle: schedule jobs in order of increasing due dates and always when a job gets late, remove the job with largest processing time; all removed jobs are late
- complexity: $O(n \log(n))$
- Example: $n = 5;\ p = (7, 8, 4, 6, 6);\ d = (9, 17, 18, 19, 21)$



- Moore-Hodgson computes an optimal solution
  Proof on the board

Single machine models: Weighted Number of Tardy Jobs

Problem $1||\sum w_jU_j$

- problem $1||\sum w_jU_j$ is NP-hard even if all due dates are the same; i.e. $1|d_j = d|\sum w_jU_j$ is NP-hard
  Proof on the board (reduction from PARTITION)
- priority based heuristic (WSPT-rule):
  schedule jobs in increasing p_j/w_j order

Single machine models: Weighted Number of Tardy Jobs

Problem $1||\sum w_jU_j$

- problem $1||\sum w_jU_j$ is NP-hard even if all due dates are the same; i.e. $1|d_j = d|\sum w_jU_j$ is NP-hard
  Proof on the board (reduction from PARTITION)
- priority based heuristic (WSPT-rule):
  schedule jobs in increasing p_j/w_j order
- WSPT may perform arbitrary bad for $1||\sum w_jU_j$:

Single machine models: Weighted Number of Tardy Jobs

Problem $1||\sum w_jU_j$

- problem $1||\sum w_jU_j$ is NP-hard even if all due dates are the same; i.e. $1|d_j = d|\sum w_jU_j$ is NP-hard
  Proof on the board (reduction from PARTITION)
- priority based heuristic (WSPT-rule):
  schedule jobs in increasing p_j/w_j order
- WSPT may perform arbitrary bad for $1||\sum w_jU_j$:

  $n = 3; p = (1, 1, M);\ w = (1+\epsilon, 1, M-\epsilon); d = (1+M, 1+M, 1+M)$

$$\sum w_jU_j(WSPT)/\sum w_jU_j(opt) = (M-\epsilon)/(1+\epsilon)$$

Dynamic Programming for $1||\sum w_jU_j$

- assume $d_1 \leq \ldots \leq d_n$
- as for $1||\sum U_j$ a solution is given by a partition of the set of jobs into sets S_1 and S_2 and jobs in S_1 are in EDD order
- Definition:
  - $F_j(t) :=$ minimum criterion value for scheduling the first j jobs such that the processing time of the on-time jobs is at most t
- $F_n(T)$ with $T = \sum_{j=1}^{n} p_j$ is optimal value for problem $1||\sum w_jU_j$
- Initial conditions:

$$F_j(t) = \begin{cases} \infty & \text{for } t < 0;\ j = 1, \ldots, n \\ 0 & \text{for } t \geq 0;\ j = 0 \end{cases} \tag{1}$$

Dynamic Programming for $1||\sum w_j U_j$ (cont.)

- if $0 \leq t \leq d_j$ and j is late in the schedule corresponding to $F_j(t)$, we have $F_j(t) = F_{j-1}(t) + w_j$
- if $0 \leq t \leq d_j$ and j is on time in the schedule corresponding to $F_j(t)$, we have $F_j(t) = F_{j-1}(t - p_j)$

Single machine models: Weighted Number of Tardy Jobs

Dynamic Programming for $1||\sum w_j U_j$ (cont.)

- if $0 \leq t \leq d_j$ and j is late in the schedule corresponding to $F_j(t)$, we have $F_j(t) = F_{j-1}(t) + w_j$
- if $0 \leq t \leq d_j$ and j is on time in the schedule corresponding to $F_j(t)$, we have $F_j(t) = F_{j-1}(t - p_j)$
- summarizing, we get for $j = 1, \ldots, n$:

$$F_j(t) = \begin{cases} \min\{F_{j-1}(t - p_j), F_{j-1}(t) + w_j\} & \text{for } 0 \leq t \leq d_j \\ F_j(d_j) & \text{for } d_j < t \leq T \end{cases} \quad (2)$$

Single machine models: Weighted Number of Tardy Jobs

DP-algorithm for $1||\sum w_j U_j$

1. initialize $F_j(t)$ according to (1)
2. FOR $j := 1$ TO n DO
3. FOR $t := 0$ TO T DO
4. update $F_j(t)$ according to (2)
5. $\sum w_j U_j(OPT) = F_n(d_n)$

Single machine models: Weighted Number of Tardy Jobs

DP-algorithm for $1||\sum w_jU_j$

1. initialize $F_j(t)$ according to (1)
2. FOR $j := 1$ TO n DO
3. FOR $t := 0$ TO T DO
4. update $F_j(t)$ according to (2)
5. $\sum w_jU_j(OPT) = F_n(d_n)$

- complexity is $O(n\sum_{j=1}^{n} p_j)$
- thus, algorithm is pseudopolynomial