Embedded Software Engineering

3 Unit Course, Winter 2010

CS Department, Univ. of Salzburg

RT Scheduling

Christoph Kirsch

www.cs.uni-salzburg.at/~ck/teaching/ESE-Winter-2010

Earliest Due Date (EDD)

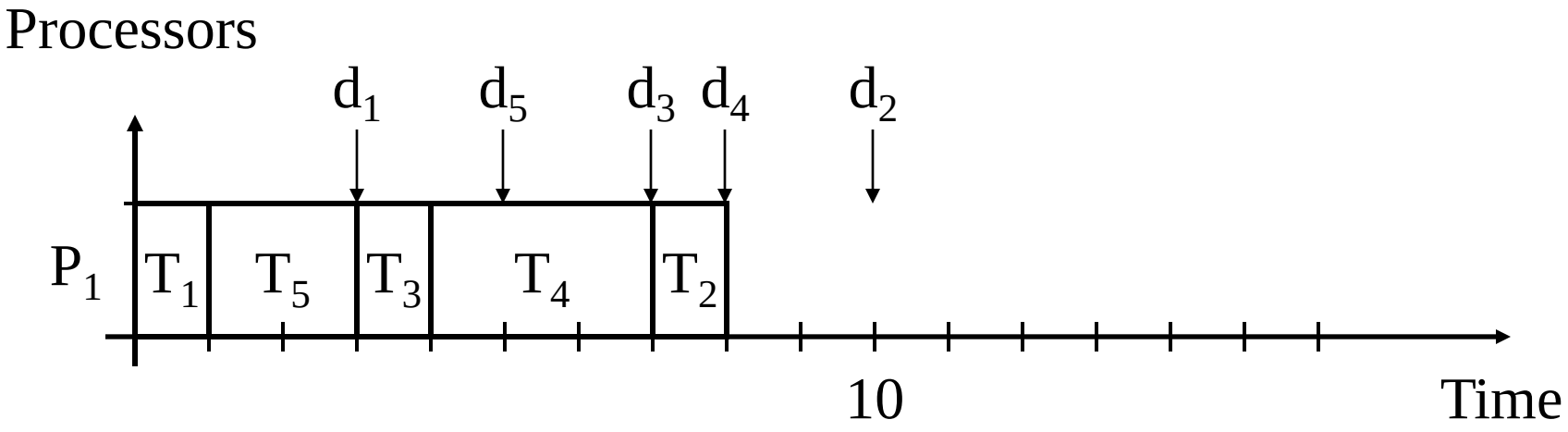
- The schedulability test for the *earliest due date* algorithm holds for a given set of n tasks, if:
  - $\forall i \in \{1,\ldots,n\}.\ f_i \leq d_i$ where $f_i = \sum_{k=1}^{i} C_k$
- The test is *exact*

- The *earliest due date* algorithm executes all tasks in a given set of n tasks in the order of non-decreasing deadlines

EDD Example

	T_1	T_2	T_3	T_4	T_5
C_i	1	1	1	3	2
d_i	3	10	7	8	5

Buttazzo97

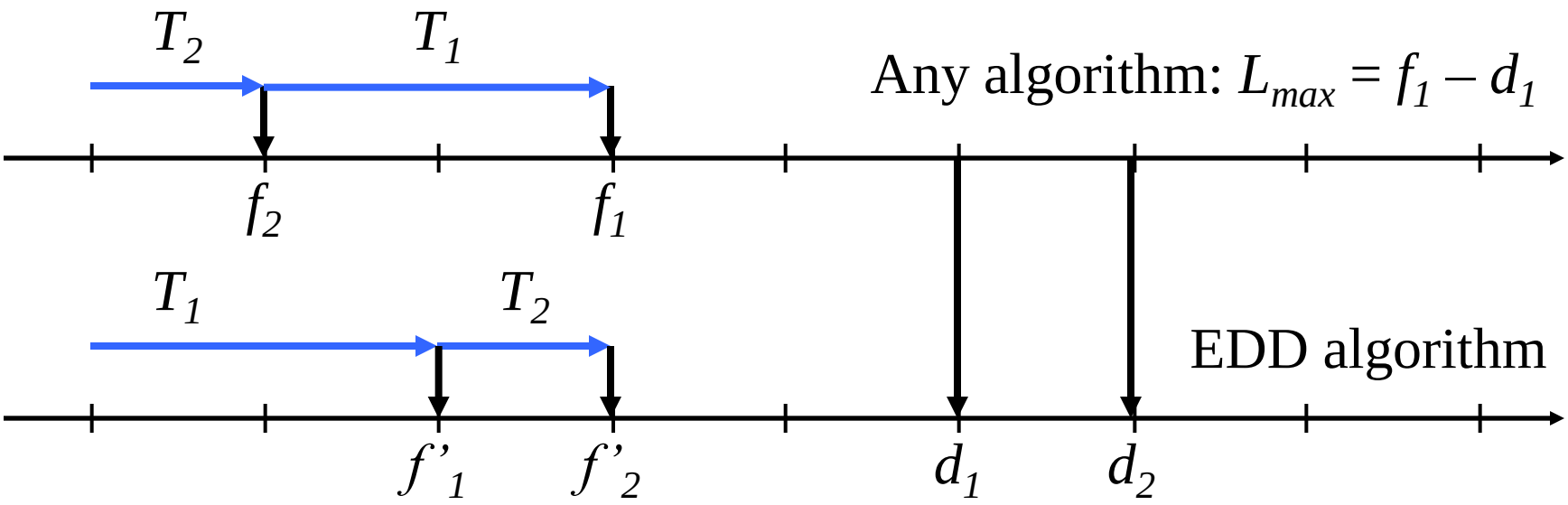


Assume, then Guarantee

- *Resource* assumptions:
  - single processor
  - no administrative overhead
- *Task* assumptions:
  - *independent*, i.e., no *precedence* constraints
  - release times are equal for all tasks
  - WCET(T_i) = C_i given
  - absolute deadlines given
- *Optimality* guarantee:
  - EDD is optimal wrt. feasibility
  - EDD is optimal wrt. maximum lateness

Proof

- *Interchange argument*:
  In a non-EDD schedule $\exists T_1, T_2$ with $d_1 \leq d_2$
  but T_2 executes before T_1



- Exchanging does not increase maximum lateness
- There are only finitely many transpositions

Earliest Deadline First (EDF)

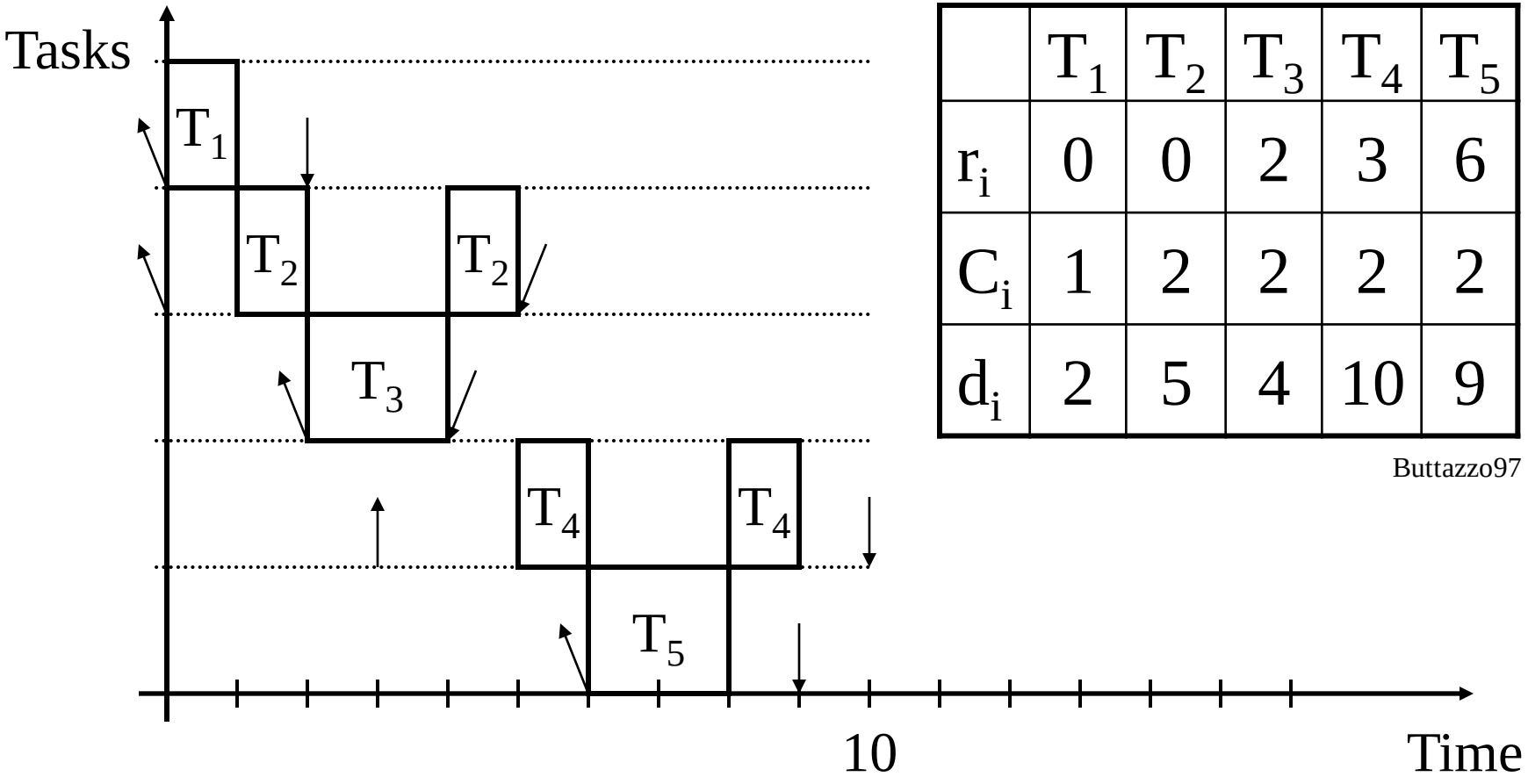
- The schedulability test for the *earliest deadline first* algorithm holds for a given set of *n* tasks, if:
  - At any instant *t* where a task is released

    $\forall i \in \{1,\ldots,n\}.\ f_i \leq d_i$ where $f_i = \sum_{k=1}^{i} c_k(t)$ and
    $c_k(t)$ is the remaining WCET of T_i at t

- The test is *exact*

- The *earliest deadline first* algorithm executes at any instant, given a set of *n* tasks, the task with the earliest deadline: dynamic priority assignment algorithm

EDF Example



	T_1	T_2	T_3	T_4	T_5
r_i	0	0	2	3	6
C_i	1	2	2	2	2
d_i	2	5	4	10	9

Buttazzo97

Assume, then Guarantee for EDF

- *Resource* assumptions:
  - single processor
  - no administrative overhead
- *Task* assumptions:
  - *preemptive*
  - *independent*, i.e., no *precedence* constraints
  - release times given
  - WCET(T_i) = C_i given
  - relative deadlines given
- *Optimality* guarantee:
  - EDF is optimal wrt. feasibility
  - EDF is optimal wrt. maximum lateness

Proof for EDF

- Based on the interchange argument for EDD:
  - Exchange time slices instead of tasks because of possible preemptions