ARTIST Summer School in Europe 2009

Autrans (near Grenoble), France

September 7-11, 2009

Techniques for multiprocessor real-time scheduling

Invited Speaker: Sanjoy Baruah

The University of North Carolina

SEVENTH FRAMEWORK PROGRAMME

Techniques for multiprocessor real-time scheduling

Sanjoy Baruah

The University of North Carolina at Chapel Hill

Supported by the US National Science Foundation, the US Army Research Office, the US Air Force Office of Scientific Research, Northrop Grumman Corp., and Intel Corp.

Techniques for multiprocessor real-time scheduling

Why multiprocessors?

- provide greater computing capacity, at lower cost
- many real-time applications are inherently parallelizable
- uniprocessor systems are becoming obsolete

-(multicore CPU's)

Goal: A theory of multiprocessor real-time scheduling

Techniques for multiprocessor Earliest Deadline First scheduling

Why multiprocessors?

- provide greater computing capacity, at lower cost
- many real-time applications are inherently parallelizable
- uniprocessor systems are becoming obsolete

-(multicore CPU's)

Goal: A theory of multiprocessor real-time scheduling

Why Earliest Deadline First (EDF)?

- more widely studied on multiprocessors
- analysis techniques (appear to) generalize

Techniques for multiprocessor Earliest Deadline First scheduling

Overview of presentation

* Task and machine model
  * Sporadic tasks; partitioned and global scheduling
* The demand bound function (DBF)
* Overview of theoretical results
  * Algorithms and lower bounds
* Pragmatic considerations

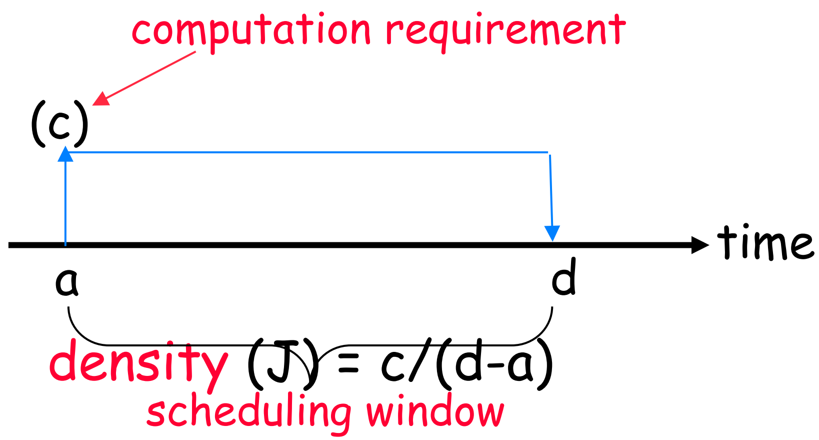
Task model

Jobs executing on $m > 1$ identical processors

Job J = (a, c, d)

- Preemptable

- Not parallelizable



Recurring tasks or processes

- finite (a priori known) number of them
- generate the jobs
- represent code within an infinite loop
- different tasks are assumed independent

The sporadic task model

Task $\tau_i = (C_i, D_i, T_i)$

- worst-case execution requirement
- relative deadline
- minimum inter-arrival separation ("period")

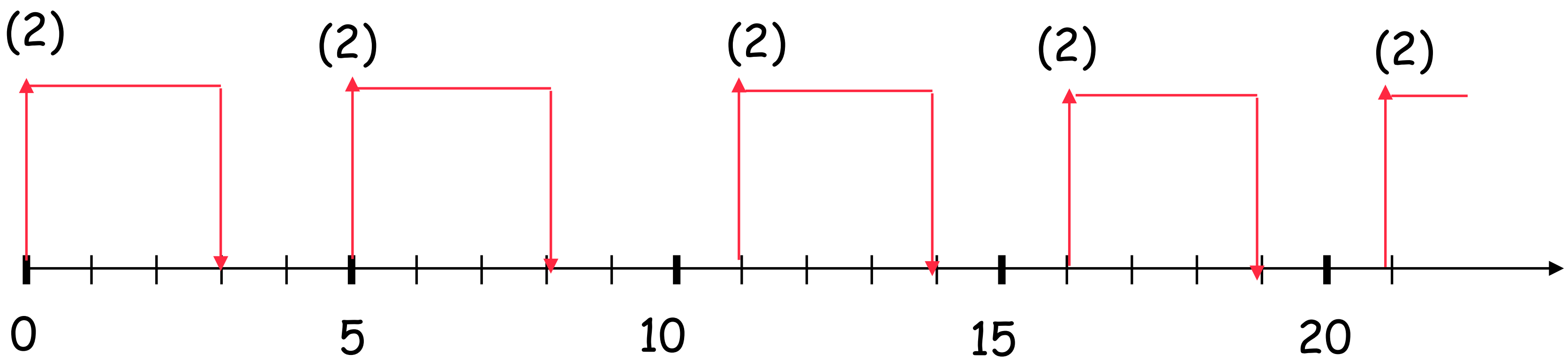
Jobs

- first job arrives at any t
- consecutive arrivals ≥ T_i
- each job has execution r
- each job has its deadline

NOTATION:

- $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$; ($D_i \leq T_i$ for all i)
- $dens_{max}(\tau) = \max_{\text{all } \tau_i \text{ in } \tau} \left(C_i / D_i \right)$

Example: τ_i =(2, 3, 5)



Global and partitioned scheduling

1.PARTITIONED

- Each task <u>assigned</u> to a processor

2. GLOBAL

- A job may execute on any processor
- A preempted job may resume on any processor

Global schedulability dominates partitioned schedulability

Global scheduling may have higher run-time overhead

The Demand Bound Function

$DBF(\tau_i, t) \equiv$ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t

$$load(\tau) \equiv \max_{\text{all } t} \left(\sum_{\tau_i \in \tau} DBF(\tau_i, t) \Big/ t \right)$$

$$DBF(\tau_i, t) = C_i \times \max\left(0, \left\lfloor \frac{t - D_i}{T_i} \right\rfloor + 1\right)$$

RESULT: Any sporadic task system τ is EDF-schedulable on a preemptive un

Maximum total execution requirement by jobs of sporadic task system τ over any time-interval of length t

RESULT: Any L&L task system τ is RM-schedulable on a preemptive uniprocessor if $load(\tau) \le \ln_e 2$

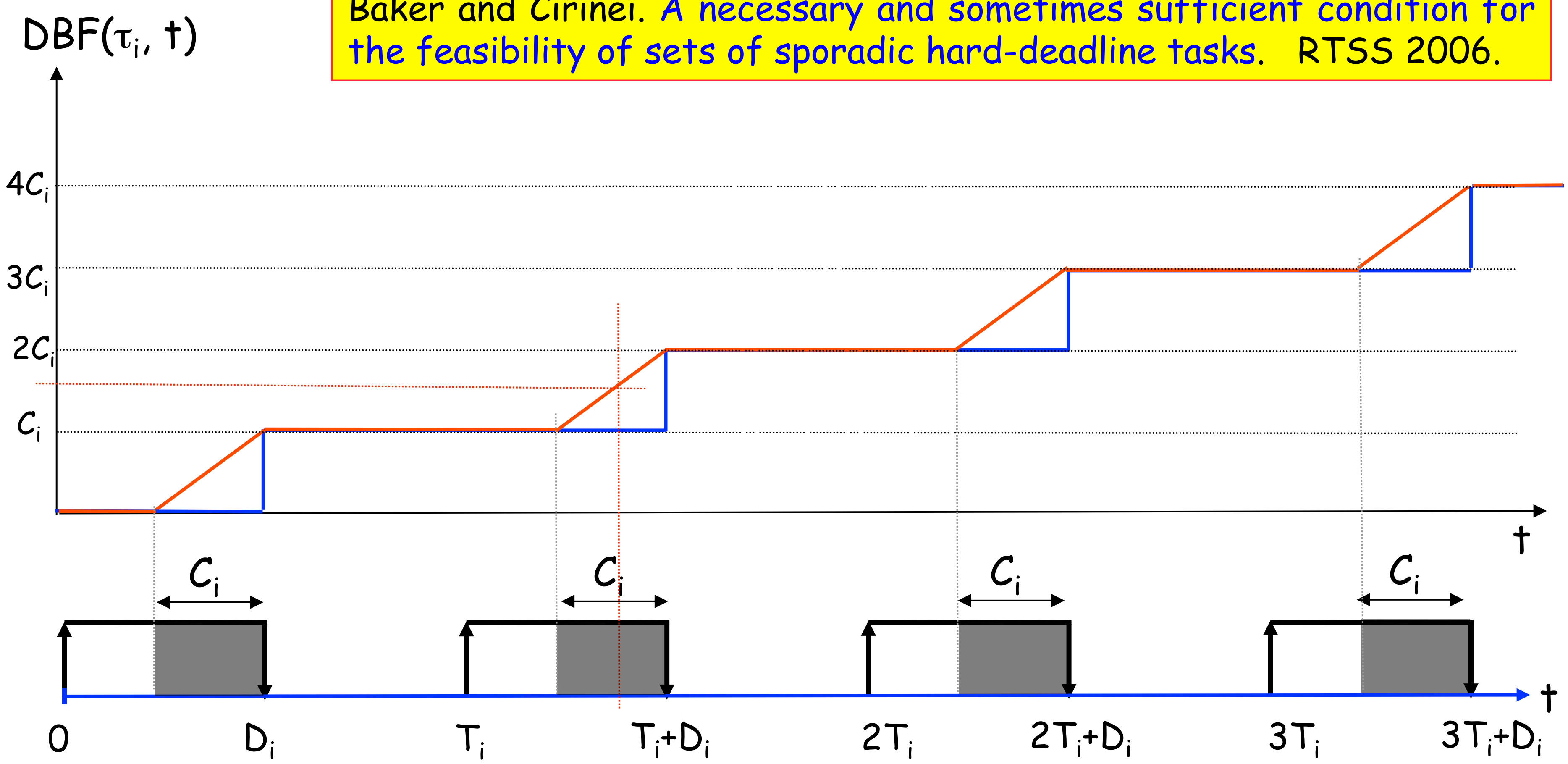
RESULT: Any sporadic task system τ is DM-schedulable on a preemptive uniprocessor if $load(\tau) \le 0.567\ldots$

Ω, the solution to the equation $x = \ln_e(1/x)$

The Demand Bound Function

DBF(τ_i, t) ≡ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t

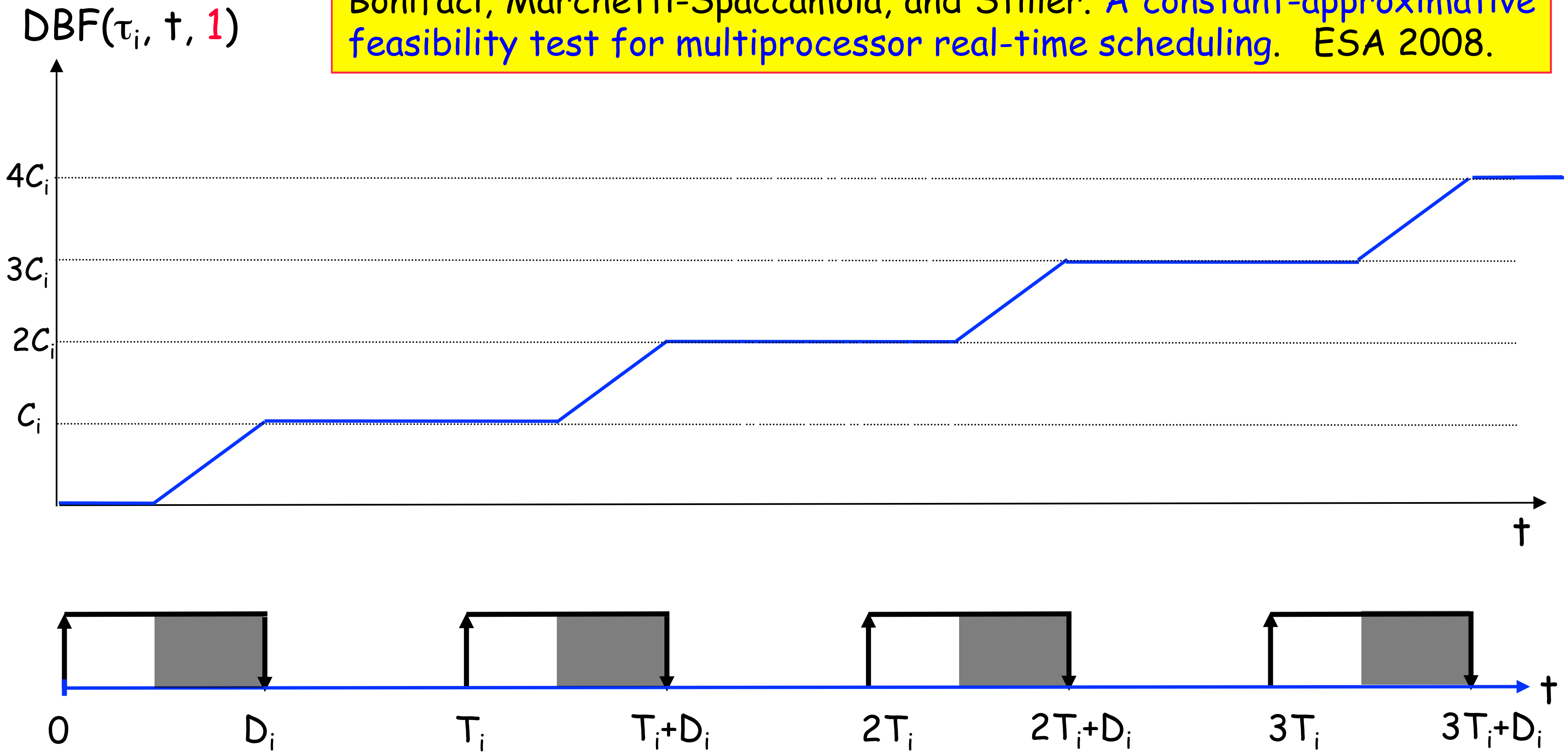
Baker and Cirinei. *A necessary and sometimes sufficient condition for the feasibility of sets of sporadic hard-deadline tasks.* RTSS 2006.



The Demand Bound Function

DBF(τ_i, t, s) ≡ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t, when executing on a speed-s processor ($s \leq 1$)

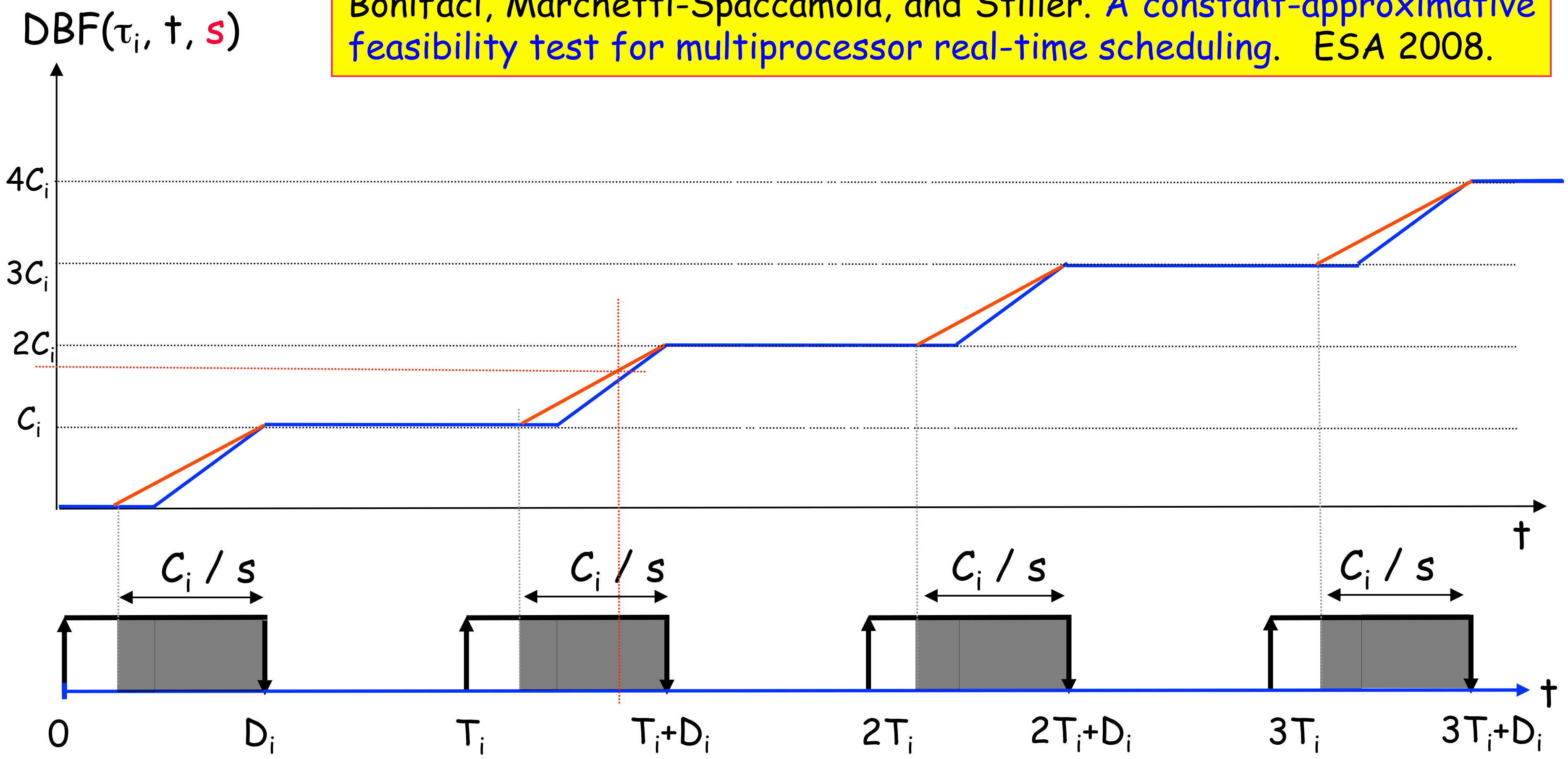
Bonifaci, Marchetti-Spaccamola, and Stiller. *A constant-approximative feasibility test for multiprocessor real-time scheduling*. ESA 2008.



The Demand Bound Function

$DBF(\tau_i, t, s) \equiv$ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t, when executing on a speed-s processor ($s \leq 1$)

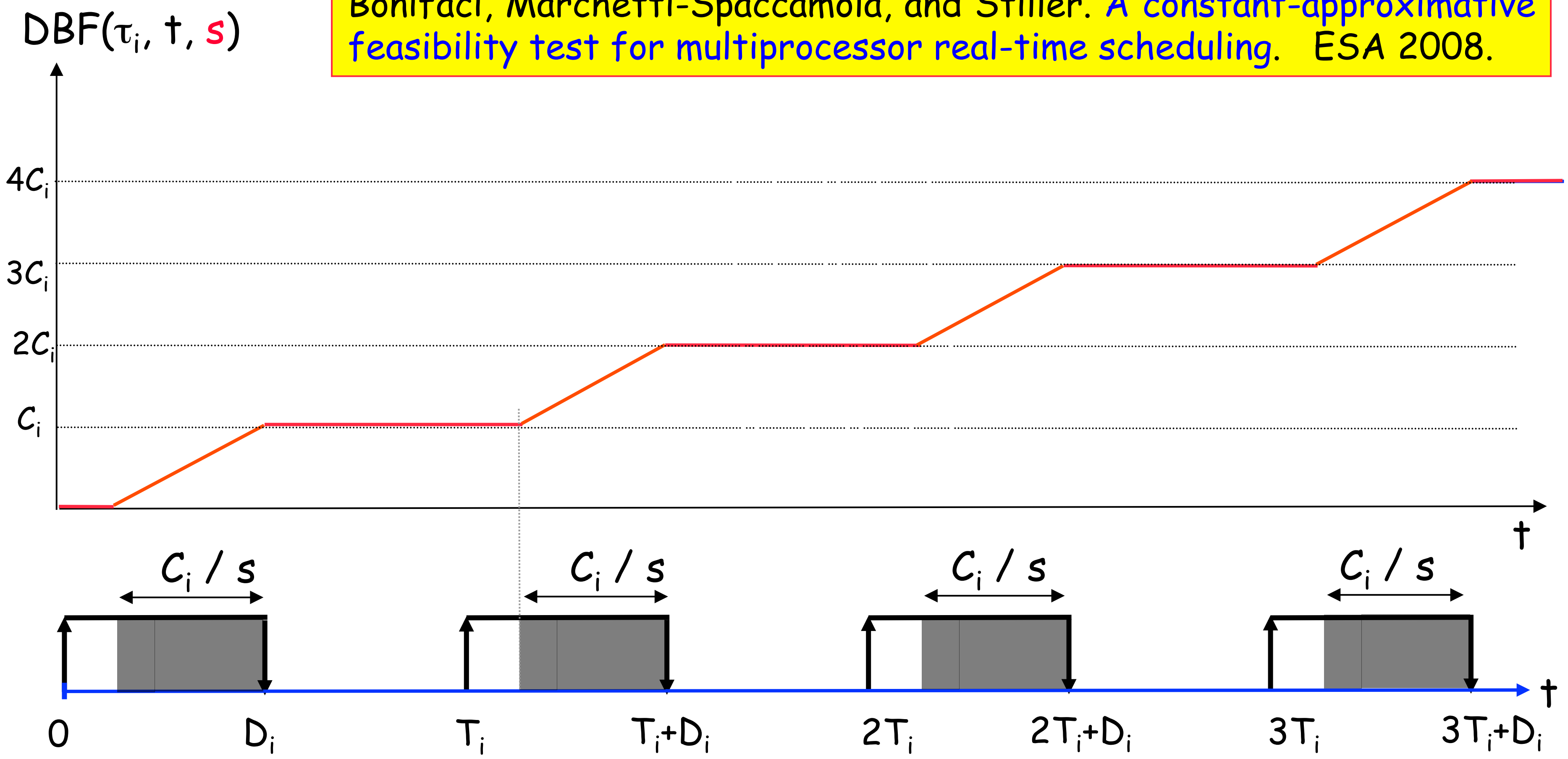
Bonifaci, Marchetti-Spaccamola, and Stiller. *A constant-approximative feasibility test for multiprocessor real-time scheduling*. ESA 2008.



The Demand Bound Function

$\mathrm{DBF}(\tau_i, t, s) \equiv$ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t, when executing on a speed-s processor ($s \le 1$)

Bonifaci, Marchetti-Spaccamola, and Stiller. *A constant-approximative feasibility test for multiprocessor real-time scheduling*. ESA 2008.



The Demand Bound Function

$DBF(\tau_i, t, s) \equiv$ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t, when executing on a speed-s processor ($s \leq 1$)

$$\boxed{load(\tau,s) \equiv \max_{\text{all } t}\left(\sum_{\tau_i \in \tau} DBF(\tau_i, t, s) \Big/ t\right)}$$

RESULT: A <u>necessary</u> condition for τ to be [EDF-]schedulable on m speed-s processors: $load(\tau,s) \leq m\, s$

GLOBAL EDF SCHEDULING

On $m > 1$ processors

At each instant, schedule m active jobs with the earliest deadlines

- (fewer active jobs than processors: idle remaining processors)

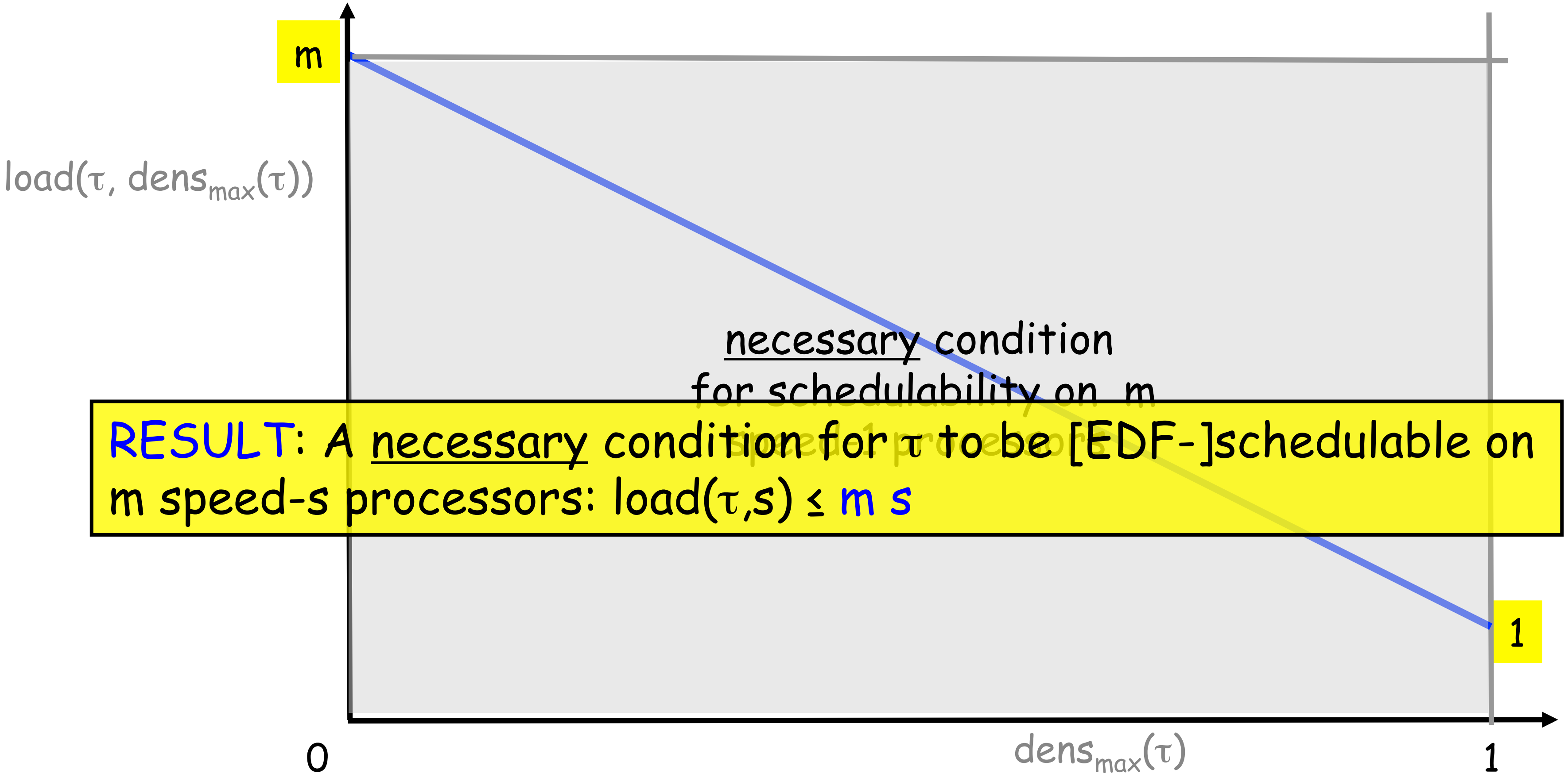
A global EDF schedulability test

RESULT: A <u>sufficient</u> condition for τ to be EDF-schedulable on m speed-1 processors: $\text{load}(\tau, \text{dens}_{max}(\tau)) \le [m - (m-1) \times \underline{\text{dens}_{max}}(\tau)]$

$$\boxed{\max_{\text{all } \tau_i \text{ in } \tau} \left(C_i / D_i \right)}$$

A global EDF schedulability test

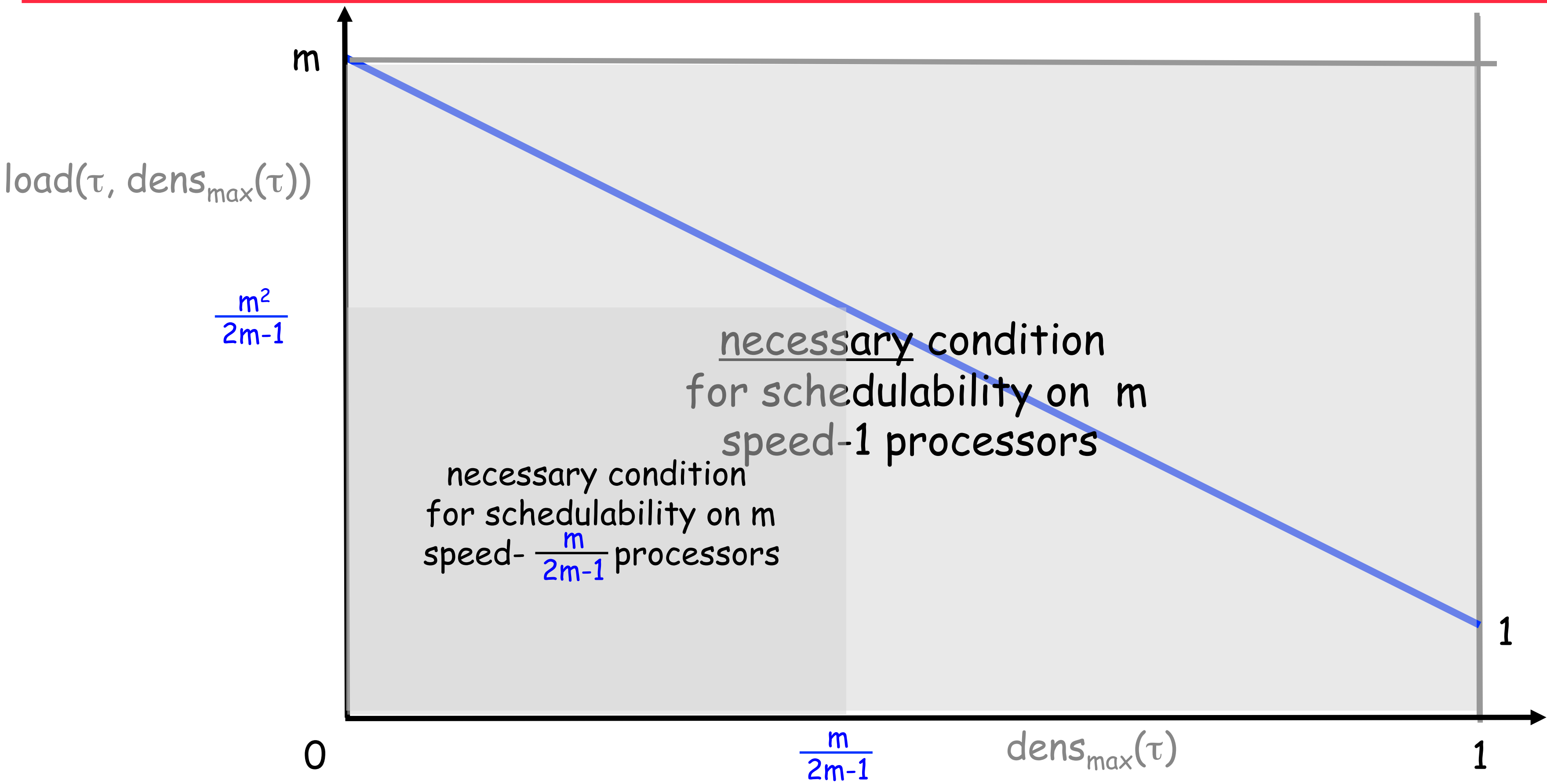
RESULT: A <u>sufficient</u> condition for τ to be EDF-schedulable on m speed-1 processors: $\boxed{load(\tau, dens_{max}(\tau)) \leq [m - (m-1) \times dens_{max}(\tau)]}$



RESULT: A <u>necessary</u> condition for τ to be [EDF-]schedulable on m speed-s processors: $load(\tau,s) \leq m\, s$

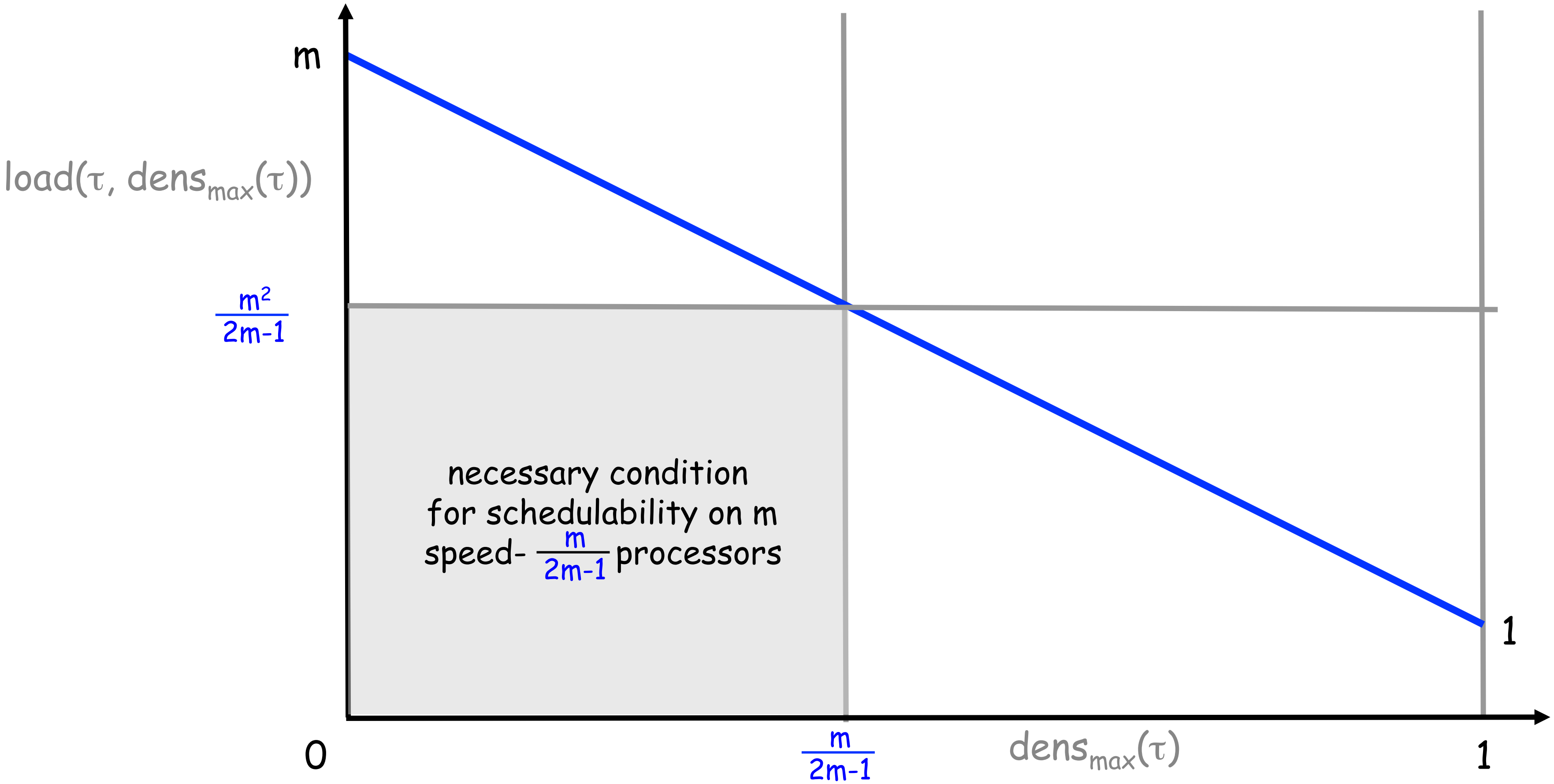
A global EDF schedulability test

Any sporadic task system schedulable upon m processors is global EDF schedulable on m processors that are each (2 - 1/m) times as fast



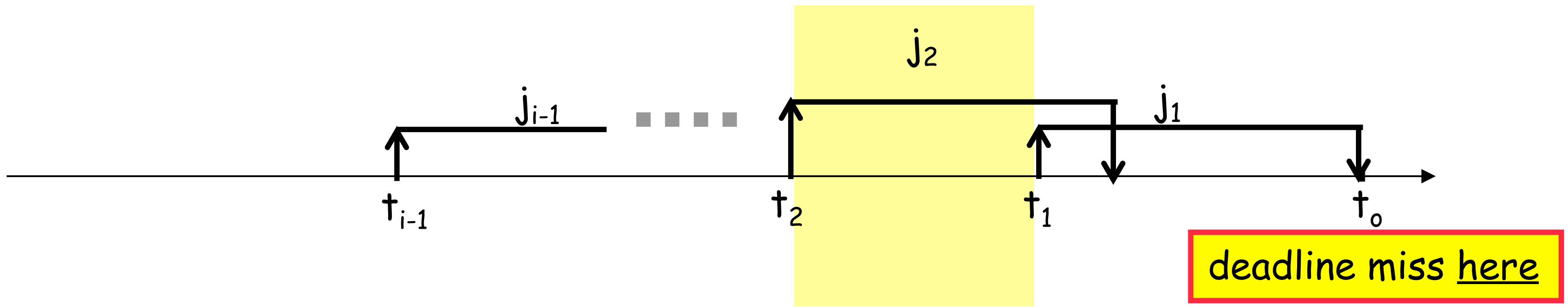
A global EDF schedulability test - Proof

RESULT: A sufficient condition for τ to be EDF-schedulable on m speed-1 processors: $load(\tau, dens_{max}(\tau)) \leq [m - (m-1) \times dens_{max}(\tau)]$



A global EDF schedulability test - Proof

RESULT: A sufficient condition for τ to be EDF-schedulable on m speed-1 processors: $load(\tau, dens_{max}(\tau)) \leq [m - (m-1) \times dens_{max}(\tau)]$



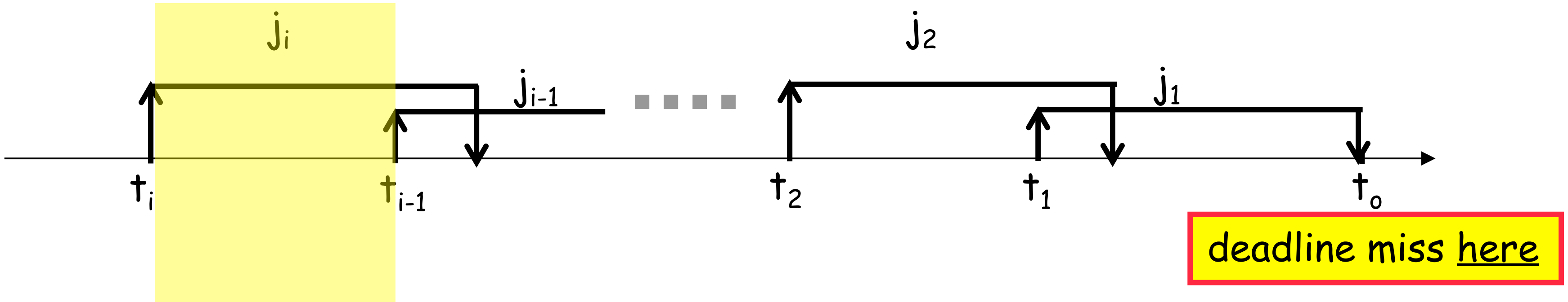
Job j_2:

- arrives at $t_2 < t_1$
- is active at t_1
- has executed for $< (t_1 - t_2) \times dens_{max}(\tau)$ by t_1

$\Rightarrow$ All m procs are busy for $> (t_1 - t_2) \times (1 - dens_{max}(\tau))$ over $[t_2, t_1)$
$\equiv$ Total idled capacity over $[t_2, t_1) < (m-1) \times (t_1 - t_2) \times dens_{max}(\tau)$

A global EDF schedulability test - Proof

RESULT: A sufficient condition for τ to be EDF-schedulable on m speed-1 processors: $load(\tau, dens_{max}(\tau)) \le [m - (m-1) \times dens_{max}(\tau)]$



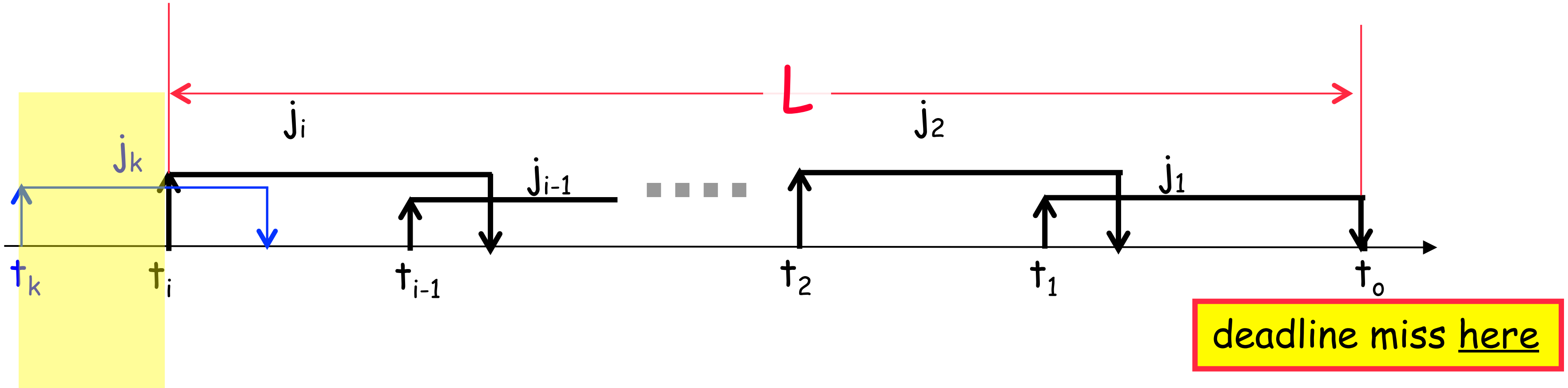
Job j_i:

- arrives at $t_i < t_{i-1}$
- is active at t_{i-1}
- has executed for $< (t_{i-1} - t_i) \times dens_{max}(\tau)$ by t_{i-1}

$\Rightarrow$ Total idled capacity over $[t_i, t_{i-1})$ $<$ $(m-1) \times (t_{i-1} - t_i) \times dens_{max}(\tau)$

A global EDF schedulability test - Proof

RESULT: A sufficient condition for τ to be EDF-schedulable on m speed-1 processors: $load(\tau, dens_{max}(\tau)) \leq [m - (m-1) \times dens_{max}(\tau)]$

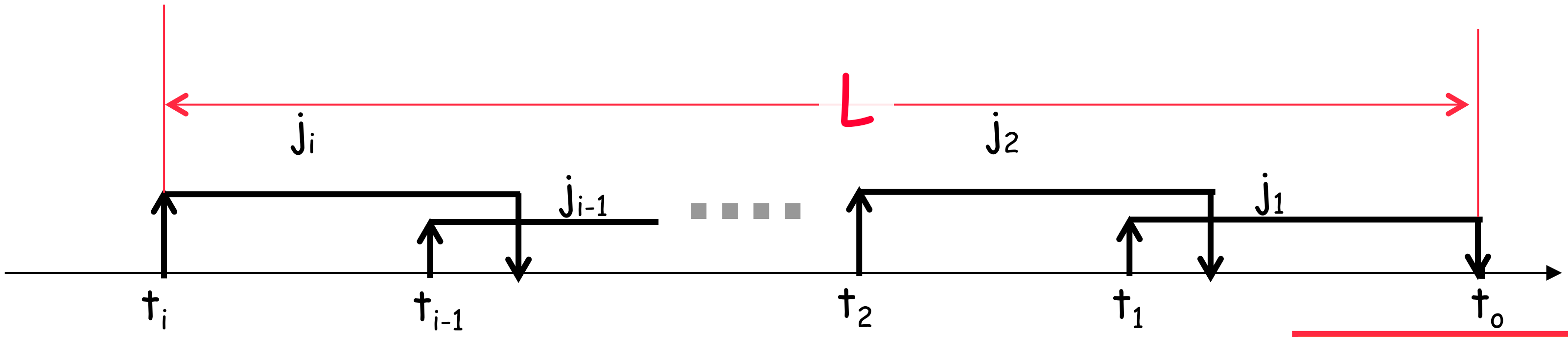


Repeat until no further such jobs

- Any job arriving at $t_k < t_i$ has executed for at least $(t_i - t_k) \times dens_{max}(\tau)$ over $[t_k, t_i)$

A global EDF schedulability test - Proof

RESULT: A sufficient condition for τ to be EDF-schedulable on m speed-1 processors: $load(\tau, dens_{max}(\tau)) \leq [m - (m-1) \times dens_{max}(\tau)]$



deadline miss here

Repeat until no further such jobs

1. total exec. requirement over $[t_i,t_o)$ is $\leq dbf(\tau,L,dens_{max}(\tau))$
2. total work done over $[t_i,t_o)$ is $= mL$ – total idle time over $[t_i,t_o)$

$\geq mL - (m-1) \times L \times dens_{max}(\tau)$

$dbf(\tau, L, dens_{max}(\tau)) > [m - (m-1) \times dens_{max}(\tau)] \times L$

$\Rightarrow$ $\boxed{load(\tau, dens_{max}(\tau)) > [m - (m-1) \times dens_{max}(\tau)]}$

PARTITIONED EDF SCHEDULING

1. partitioning algorithm Must be defined

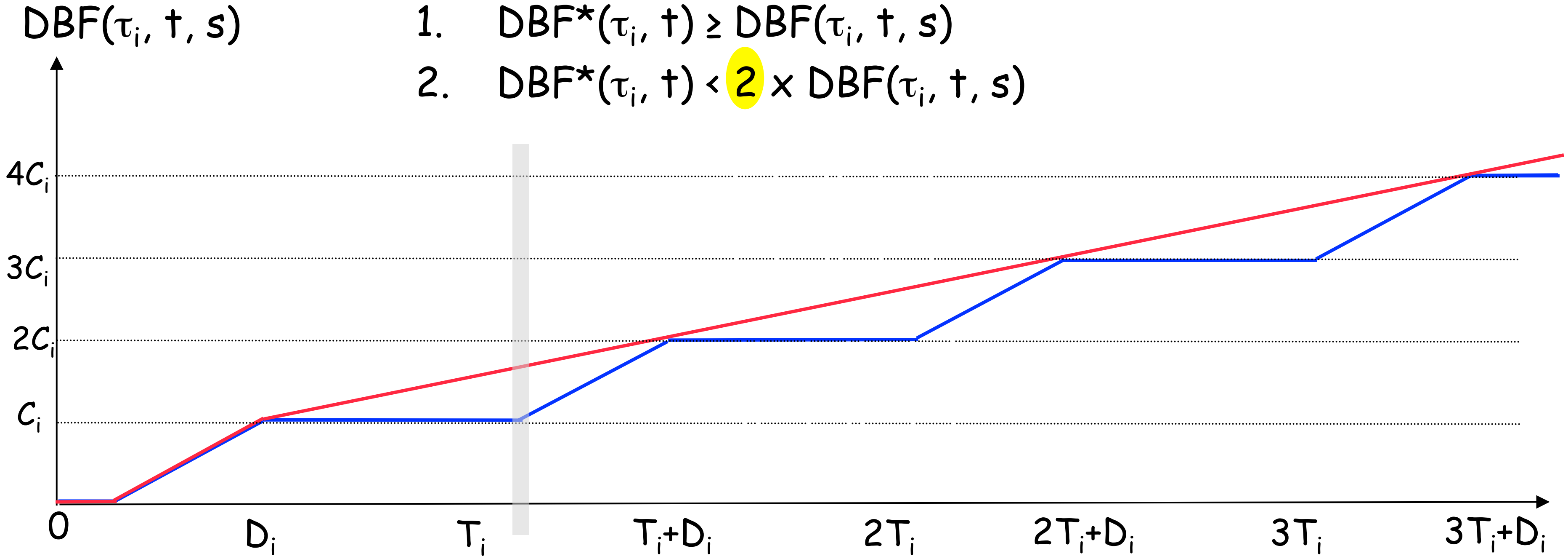
2. uniproc. EDF scheduling

Can reuse results for uniproc EDF

$DBF^*(\tau_i, t)$: an approximation to $DBF(\tau_i, t, s)$

$DBF(\tau_i, t, s) \equiv$ maximum cumulative execution requirement of jobs of sporadic task τ_i in any interval of length t, when executing on a speed-s processor ($s \leq 1$)

1. $DBF^*(\tau_i, t) \geq DBF(\tau_i, t, s)$
2. $DBF^*(\tau_i, t) < 2 \times DBF(\tau_i, t, s)$



$$2 - \left(\frac{C_i}{(T_i * s)}\right)$$

A partitioning algorithm

Sporadic task system $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$, on m speed-1 processors

```
assume D_i ≤ D_{i+1} for all i
for i := 1 to n
      - assign τ_i to any processor j such that
```

τ_i "fits" on processor j

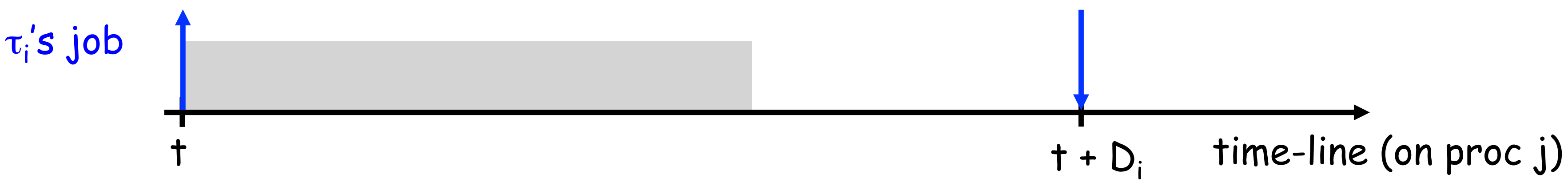
A partitioning algorithm

Sporadic task system $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$, on m speed-1 processors

assume $D_i \leq D_{i+1}$ **for all** i
for i := 1 **to** n
    - assign τ_i **to any processor** j **such that**

$$D_i - \left\{ \sum_{\tau_k \text{ assigned to proc } j} DBF^*(\tau_k, D_i) \right\} \geq C_i$$

(Upper bound on) execution <u>already</u> allocated over $[t, t + D_i)$

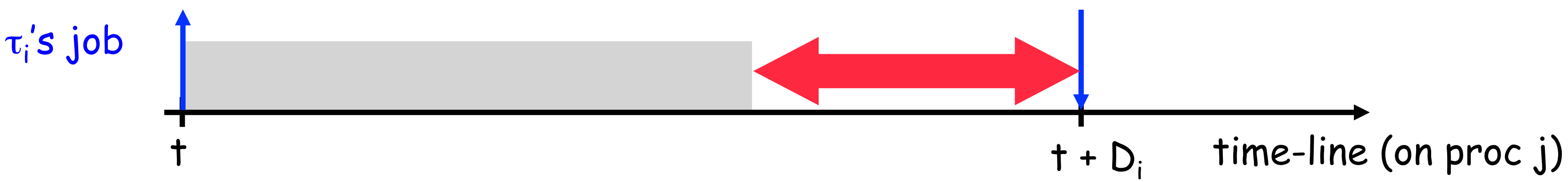


A partitioning algorithm

Sporadic task system $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$, on m speed-1 processors

assume $D_i \leq D_{i+1}$ for all i
for i := 1 to n
- assign τ_i to any processor j such that

$$D_i - \left\{ \sum_{\tau_k \text{ assigned to proc } j} \mathrm{DBF}^*(\tau_k, D_i) \right\} \geq C_i$$



A partitioning algorithm

Sporadic task system $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$, on m speed-1 processors

```
assume D_i ≤ D_i+1 for all i
for i := 1 to n
      - assign τ_i to any processor j such that
```

$$D_i - \left\{ \sum_{\tau_k \text{ assigned to proc } j} \mathrm{DBF}^*(\tau_k, D_i) \right\} \geq C_i$$

```
      - if no such processor j, then return failure
```

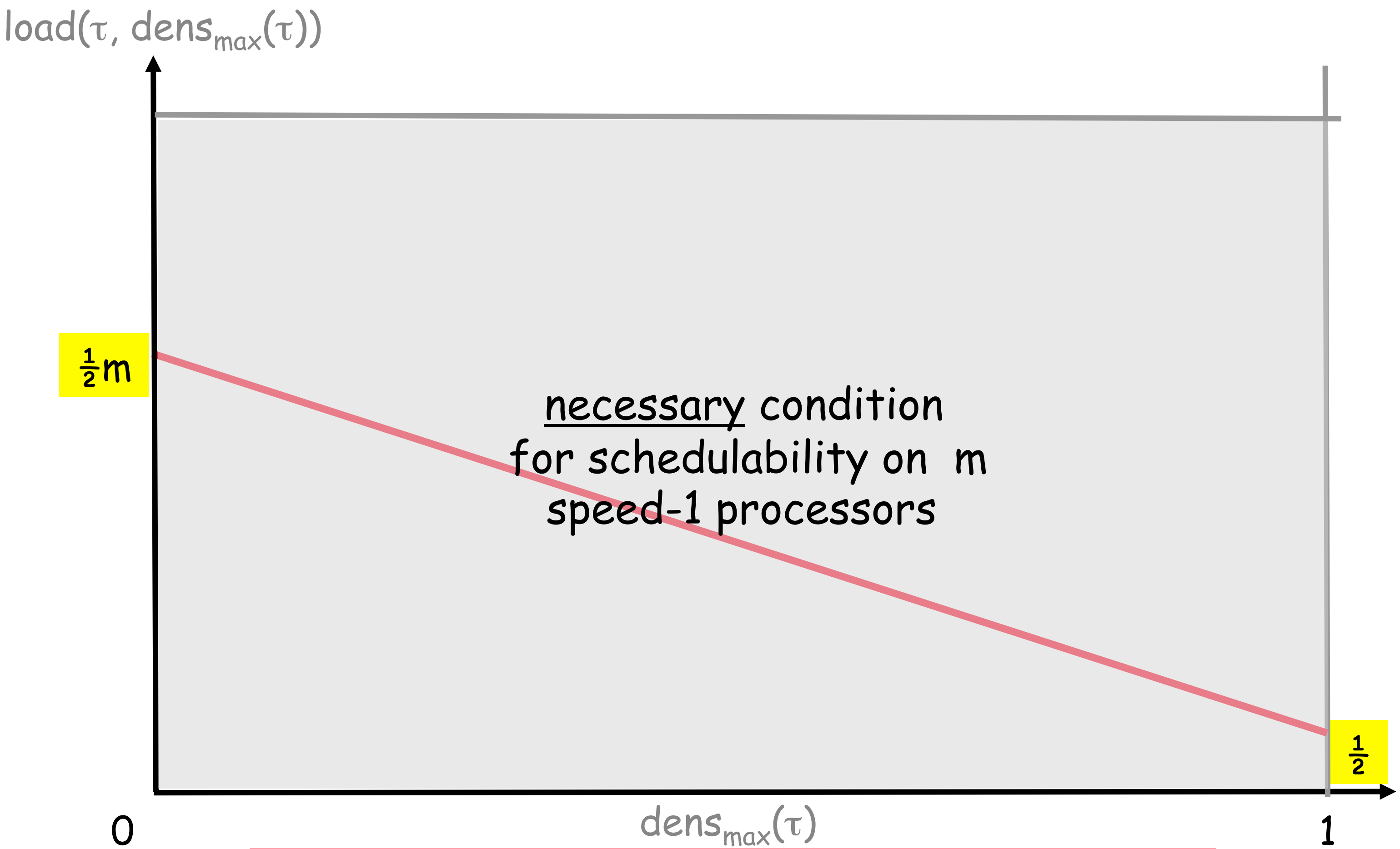
Runtime complexity: $O(n^2)$

Schedulability test: Run the partitioning algorithm!

A property: this algorithm schedules any system satisfying

$$\boxed{\mathrm{load}(\tau, \mathrm{dens}_{max}(\tau)) \leq \tfrac{1}{2}\,[m - (m-1) \times \mathrm{dens}_{max}(\tau)]}$$

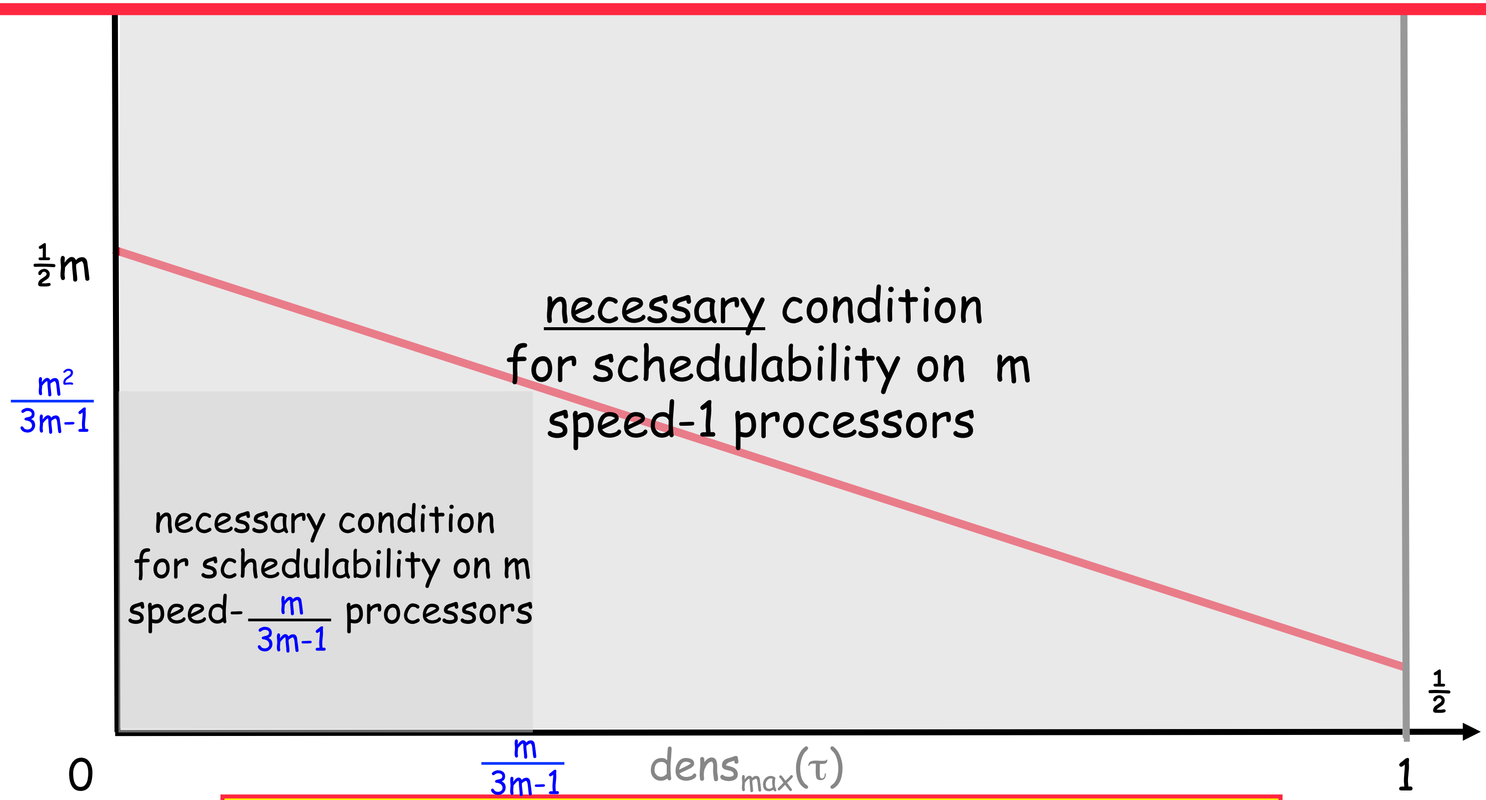
A partitioning algorithm



$$\text{load}(\tau, \text{dens}_{max}(\tau)) \leq \tfrac{1}{2}\,[m - (m-1) \times \text{dens}_{max}(\tau)]$$

A partitioning algorithm

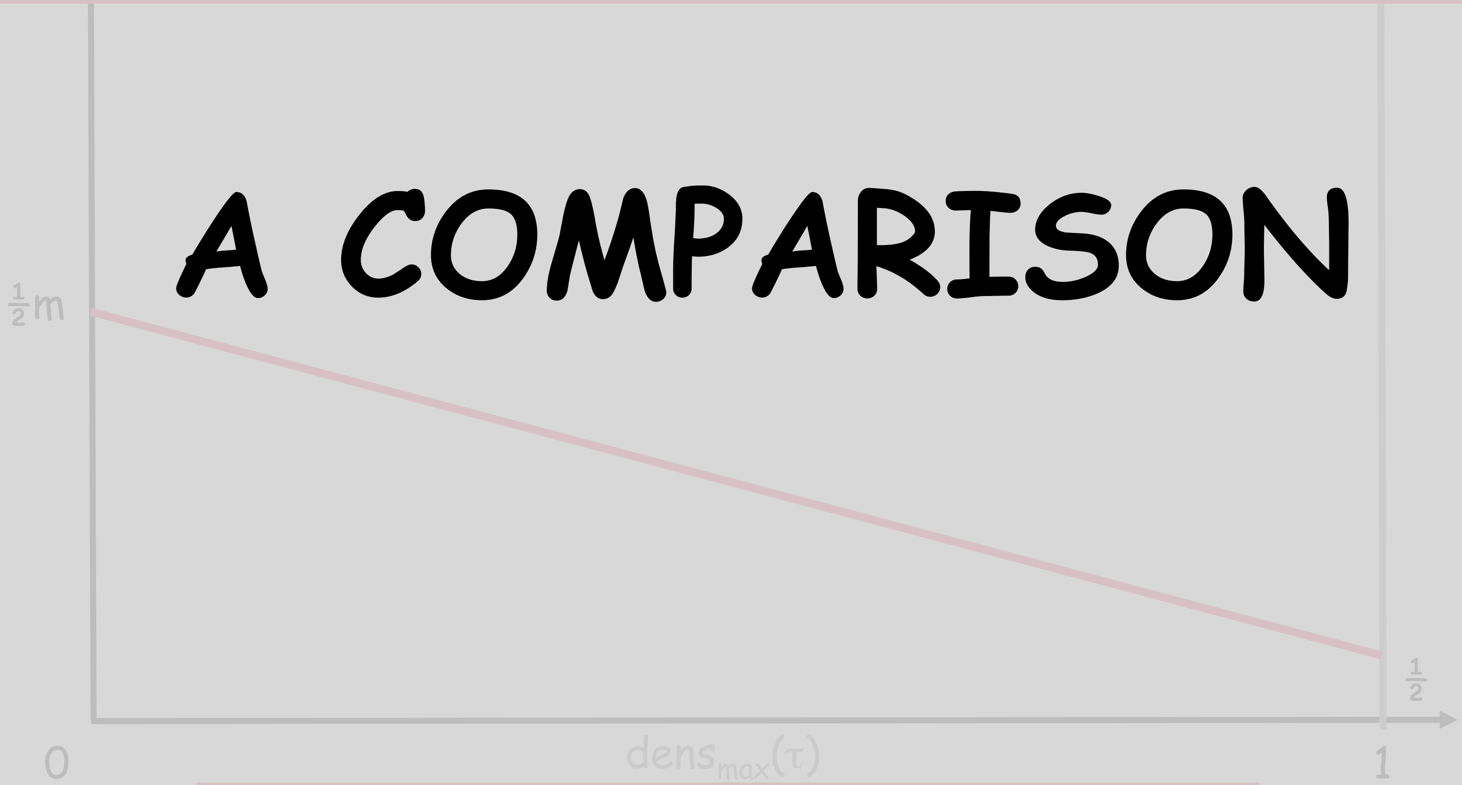
Any sporadic task system schedulable upon m processors is partitioned-EDF schedulable on m processors that are each (3 - 1/m) times as fast



$\text{load}(\tau, \text{dens}_{max}(\tau)) \leq \frac{1}{2}\,[m - (m-1) \times \text{dens}_{max}(\tau)]$

Any sporadic task system schedulable upon m processors is partitioned-EDF schedulable on m processors that are each (3 - 1/m) times as fast

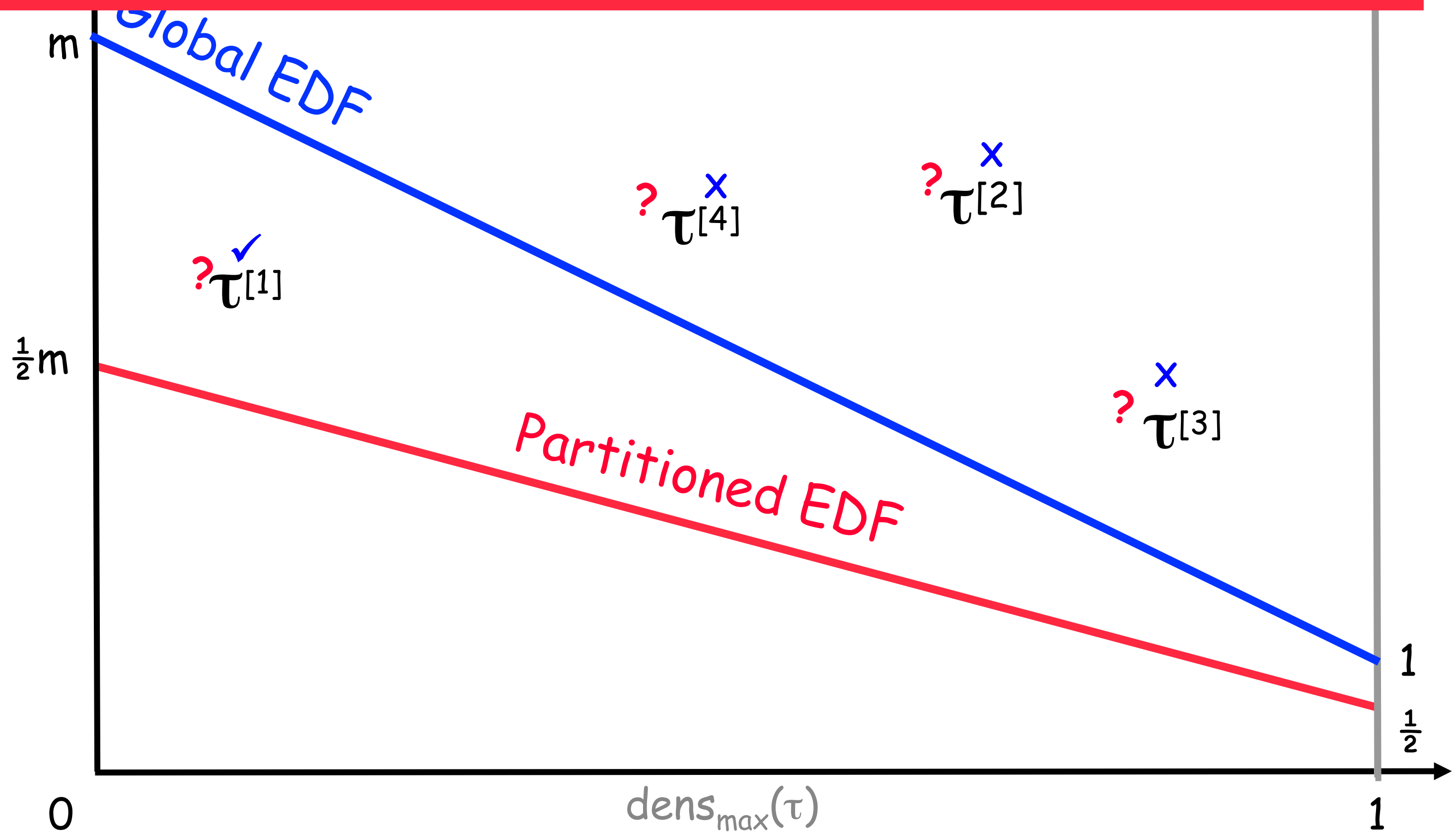
A COMPARISON



$load(\tau, dens_{max}(\tau)) \leq \frac{1}{2} [m - (m-1) \times dens_{max}(\tau)]$

Comparing Global and Partitioned EDF

BOTTOM LINE: Partitioned EDF and Global EDF offer comparable schedulability (to random tasksets)



Context and conclusions

Multiprocessor systems are increasingly important

⟹ need a theory of multiprocessor RT scheduling

Some breakthroughs recently…

- sporadic tasks on identical multiprocessors, scheduled using EDF
- other models, other algorithms

Multiprocessor RT scheduling theory today

≈

Uniprocessor RT scheduling theory in mid-1980's

⟹ significant progress soon??