UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Real-Time Systems Laboratory

EDF Scheduling for Identical Multiprocessor Systems

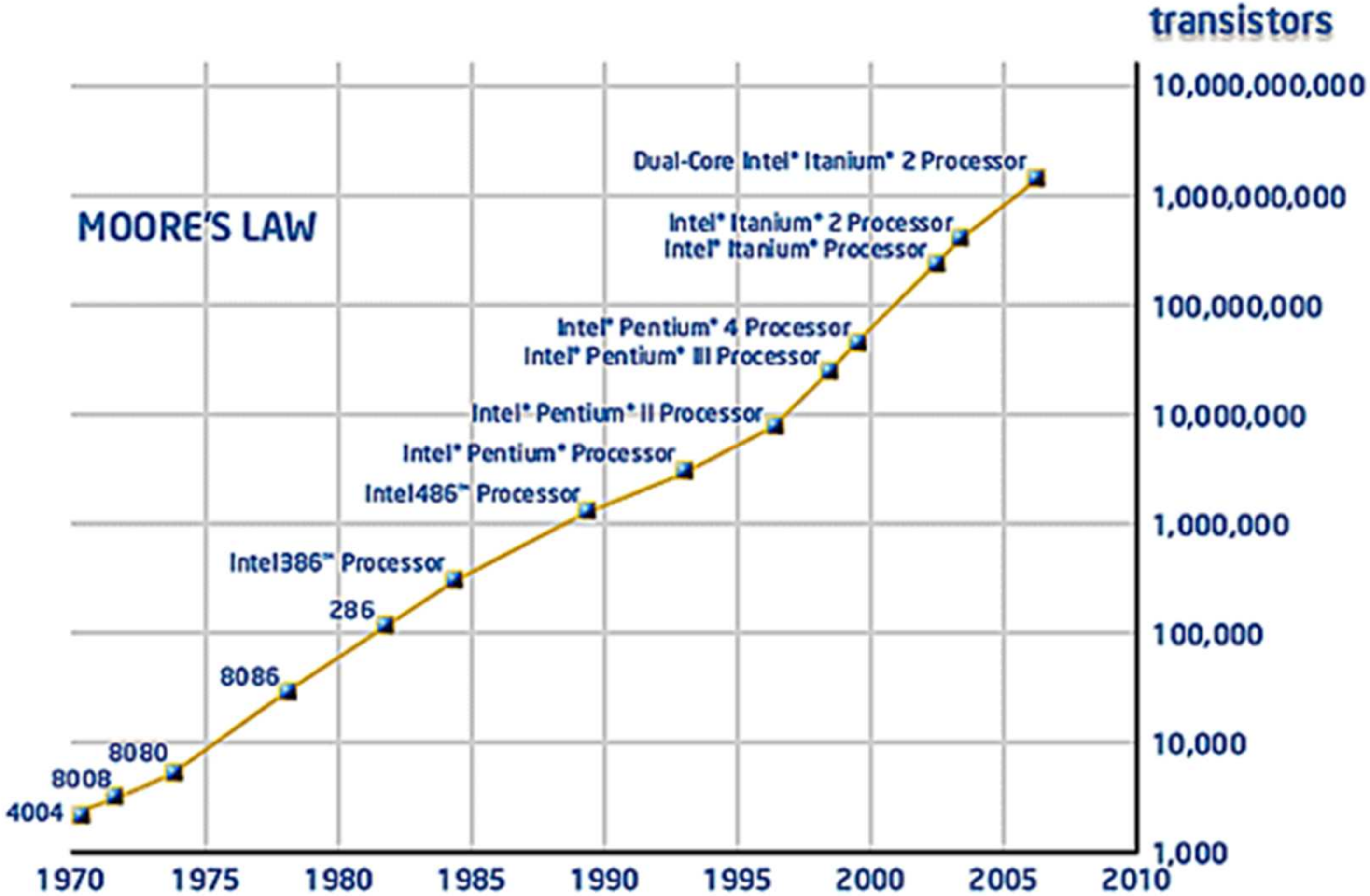
Marko Bertogna

University of Modena, Italy

June 19th, 2012, Luxembourg

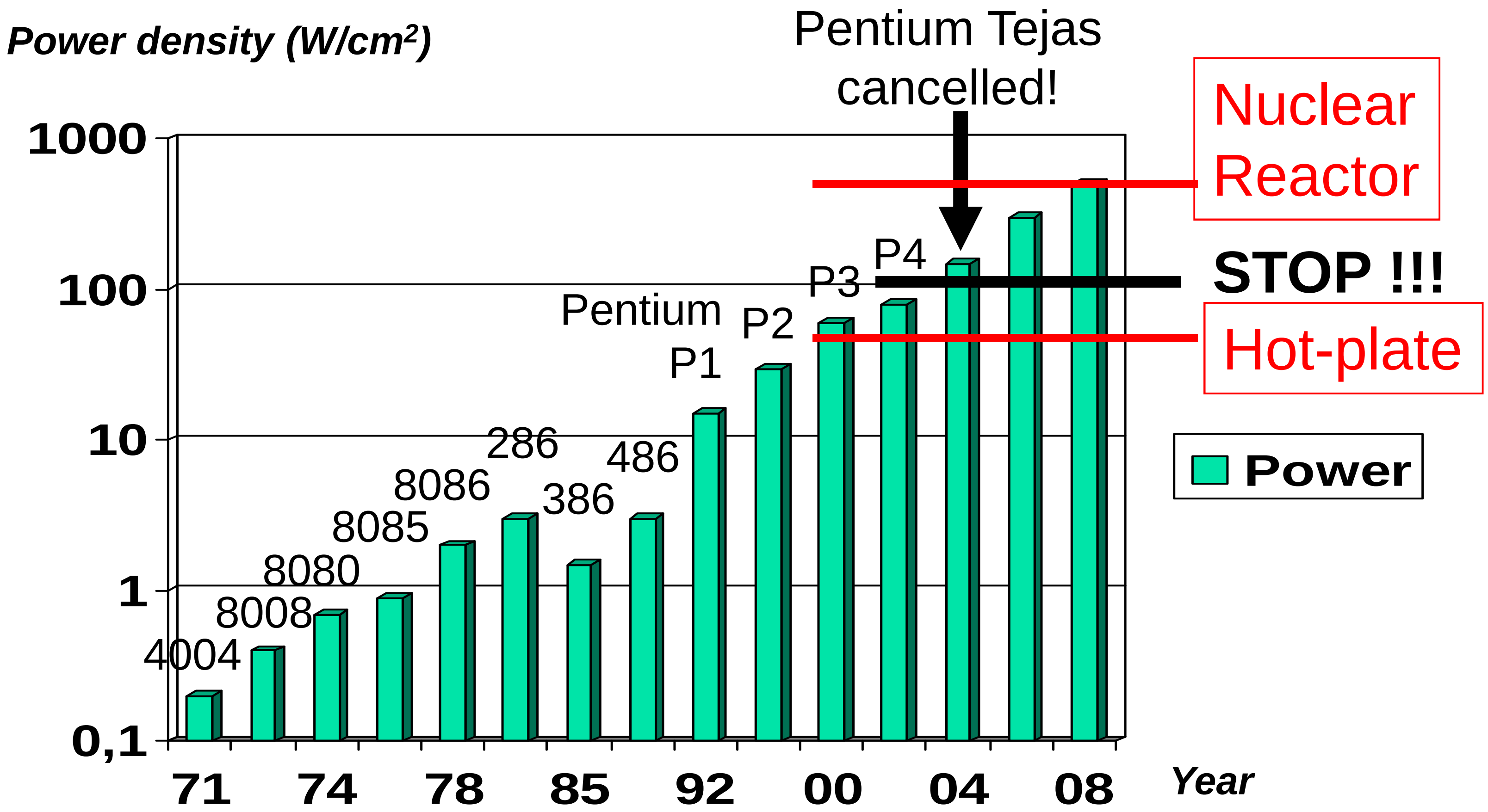
As Moore's law goes on…

- Number of transistor/chip doubles every 18 to 24 mm



…heating becomes a problem

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA



- P → V → f: Clock speed limited to less than 4 GHz

Technology trends

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Reduced gate sizes
- Higher frequencies allowed
- Larger number of transistors

BUT

- Physical limits of semiconductor-based microelectronics
- Larger dynamic power consumed
- Leakage current becomes important
- Higher density of transistors

Intel's timeline

UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Year	Processor	Manufacturing Technology	Frequency	Number of transistors
1971	4004	10 μm	108 kHz	2.300
1972	8008	10 μm	800 kHz	3.500
1974	8080	6 μm	2 MHz	4.500
1978	8086	3 μm	5 MHz	29.000
1979	8088	3 μm	5 MHz	29.000
1982	286	1,5 μm	6 MHz	134.000
1985	386	1,5 μm	16 MHz	275.000
1989	486	1 μm	25 MHz	1.200.000
1993	Pentium	0,8 μm	66 MHz	3.100.000
1995	Pentium Pro	0,6 μm	200 MHz	5.500.000
1997	Pentium II	0,25 μm	300 MHz	7.500.000
1999	Pentium III	0,18 μm	500 MHz	9.500.000
2000	Pentium 4	0,18 μm	1,5 GHz	42.000.000
2002	Pentium M	90 nm	1,7 GHz	55.000.000
2005	Pentium D	65 nm	3,2 GHz	291.000.000
2006	Core 2 Duo	65 nm	2,93 GHz	291.000.000
2007	Core 2 Quad	65 nm	2,66 GHz	582.000.000
2008	Core 2 Quad X	45 nm	3 GHz	820.000.000
2010	Core i3, i5, i7	32 nm	3,33 GHz	1.160.000.000
2012	Core i5, i7	22 nm	3,4 GHz	2.270.000.000
2014	?	16nm	?	

Power, frequency and voltage

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

$$P = \underbrace{A\,C\,V^2 f}_{\text{Dynamic power}} + \underbrace{V\,I_{leak}}_{\text{Static power (important below 100nm)}}$$

- A, C, f ↑ ⇒ Dynamic power increases exponentially
- Reducing V allows a quadratic reduction on dynamic P
- But clock frequency would decrease more than linearly since V~=0.3+0.7 f
  - unless V_{th} as well is reduced, but
    $I_{leak} \rightarrow I_{sub} + I_{gox} \rightarrow$ increases when V_{th} is low!

There is no way out for classic frequency scaling on single cores systems!

Keeping Moore's law alive

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Exploit the immense number of transistors in other ways
- Reduce gate sizes maintaining the frequency sufficiently low
- Use a higher number of slower logic gates
- In other words:

Switch to Multicore Systems!

How many cores in the future?

UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Patterson & Hennessy: *"number of cores will double every 18 months, while power, clock frequency and costs will remain constant"*
- More likely to be application dependent
- Many trade-offs
  - Technology limits
  - Transistor density
  - Amdahl's law

Amdahl's law

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- The total speedup that can be obtained increasing the number of processors is

$$\frac{1}{(1-P)+\frac{P}{N}}$$

P ← Parallel portion of application

N ← Number of processors/cores

Amdahl's law

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

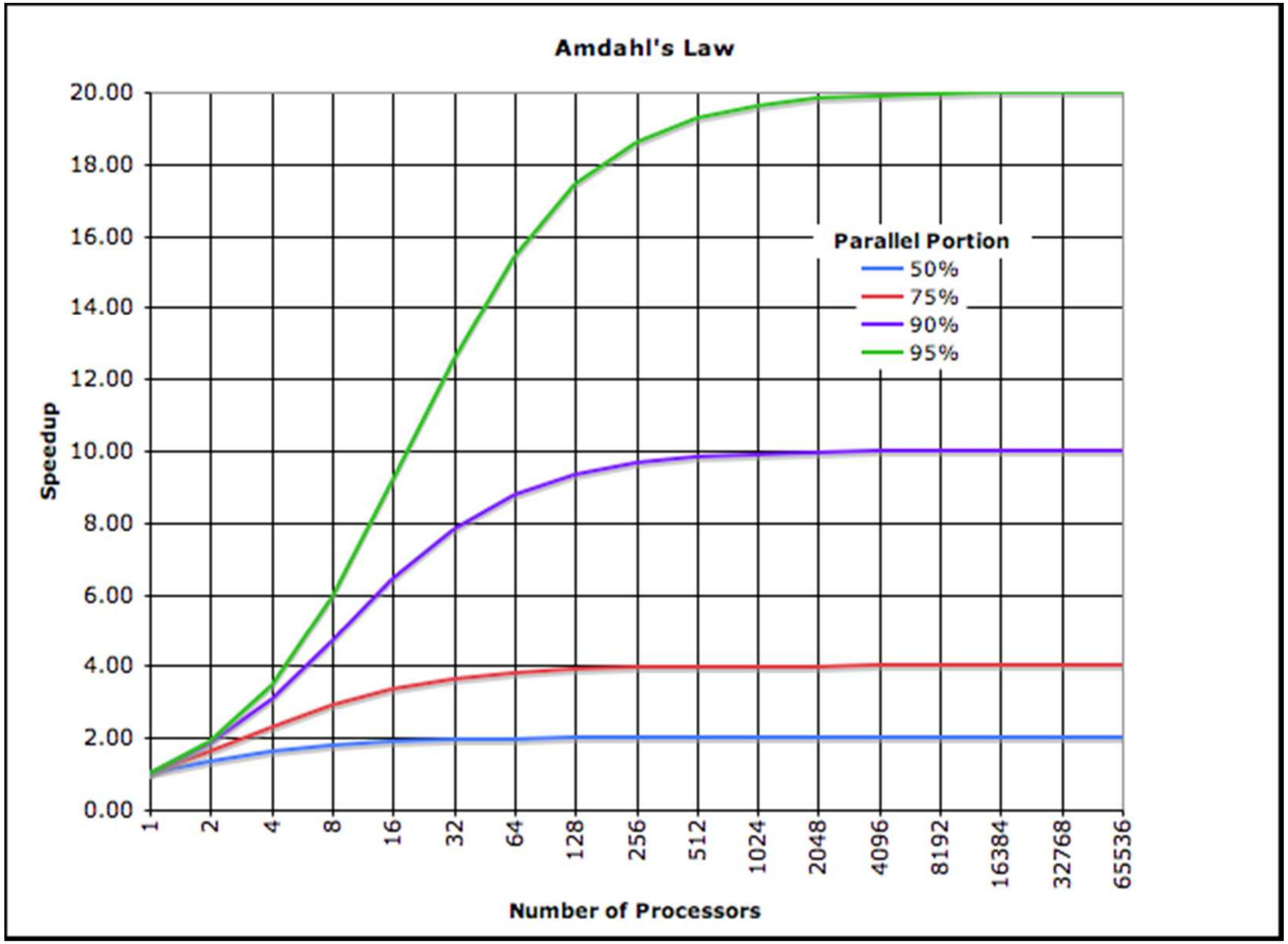
- The total speedup that can be obtained increasing the number of processors is

$$\frac{1}{(1-P)+\frac{P}{N}} \xrightarrow{N \to \infty} \frac{1}{1-P}$$

- In practice, performance/price falls rapidly as *N* is increased, even with a small *(1 – P)*
  - E.g.: P = 90% → (1 – P) = 10% → speedup < 10

Amdahl’s law

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA



$$\frac{1}{(1-P)+\frac{P}{N}}$$

Consequences

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Parallel computing is likely to be useful for
  - Small/medium number of processors, or
  - Embarrassingly parallel problems *(P → 1)*
    - Large number of parallel tasks with no dependencies
    - E.g., brute-force search in cryptography, 3D projection, GPU handled problems, etc.
    - 1 core @ 4 GHz = 2 cores @ 2 GHz
- Memory, bus and I/O bottlenecks impose further constraints

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

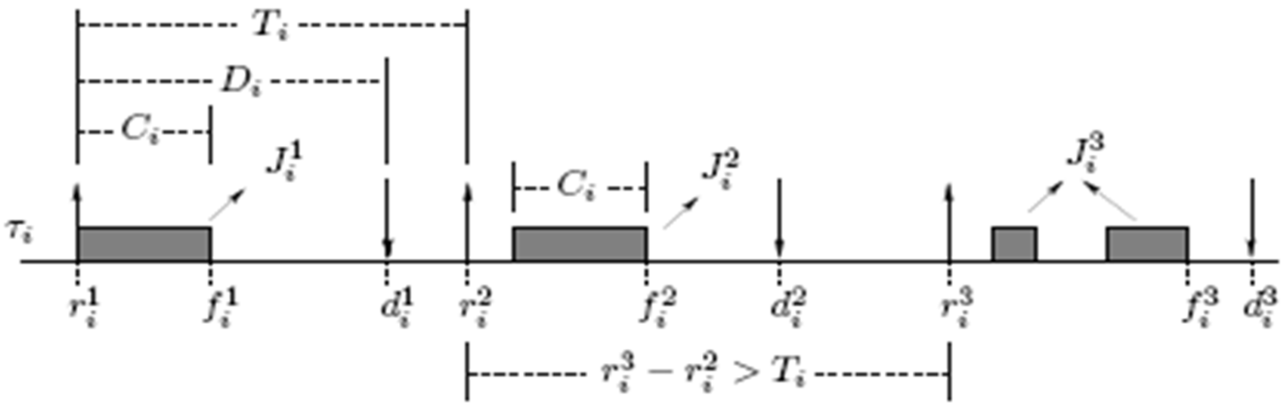
Retis
Real-Time Systems Laboratory

Real-time schedulability on identical multiprocessor systems

System model

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

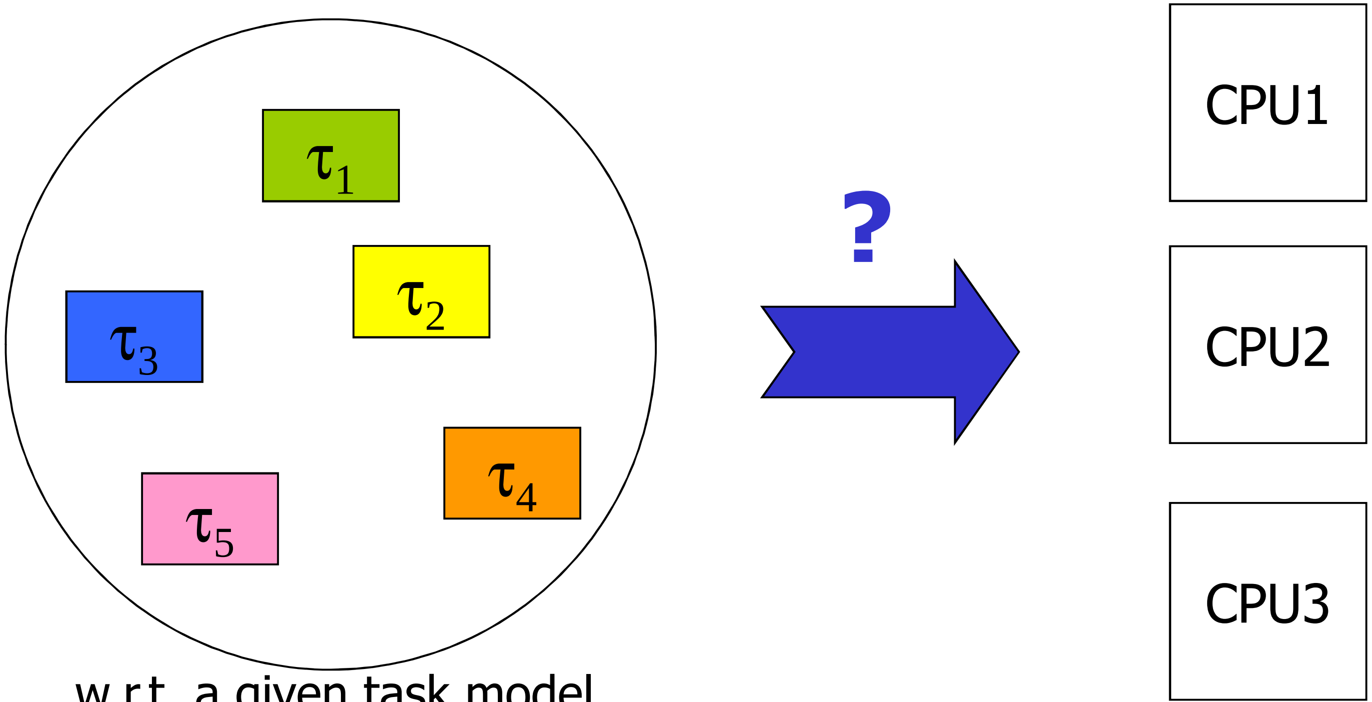
- Platform with m identical processors
- Task set τ with n independent sporadic tasks τ_i
  - Period or minimum inter-arrival time T_i
  - Worst-case execution time C_i
  - Deadline D_i
  - Utilization $U_i = C_i/T_i$, density $\lambda_i = C_i/min(D_i, T_i)$



Possible problems

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Feasibility problem
- Run-time scheduling problem
- Schedulability problem



w.r.t. a given task model

Uniprocessor RT Systems

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Solid theory (starting from the 70s)
- Optimal schedulers
- Tight schedulability tests for different task models
- Shared resource protocols
- Bandwidth reservation schemes
- Hierarchical schedulers
- RTOS support

Single processor EDF

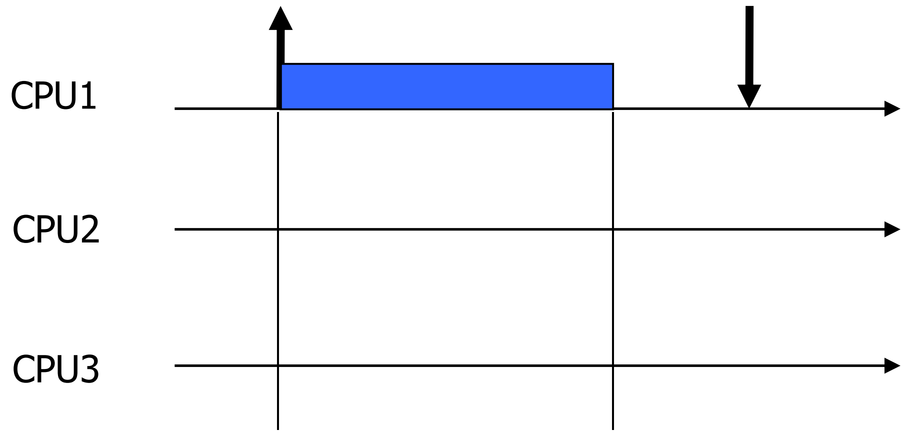
UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Optimal
  - if a set of jobs is feasible, than it can be successfully scheduled with EDF
- Bounded number of preemptions
- Efficient implementations
- Exact feasibility conditions
  - Linear test for implicit deadlines: $U_{tot} \leq 1$
  - Pseudo-polynomial test for constrained and arbitrary deadlines [Baruah et al. 90]

Muliprocessors are difficult

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- "The simple fact that a task can use only one processor even when several processors are free at the same time adds a surprising amount of difficulty to the scheduling of multiple processors" [Liu'69]



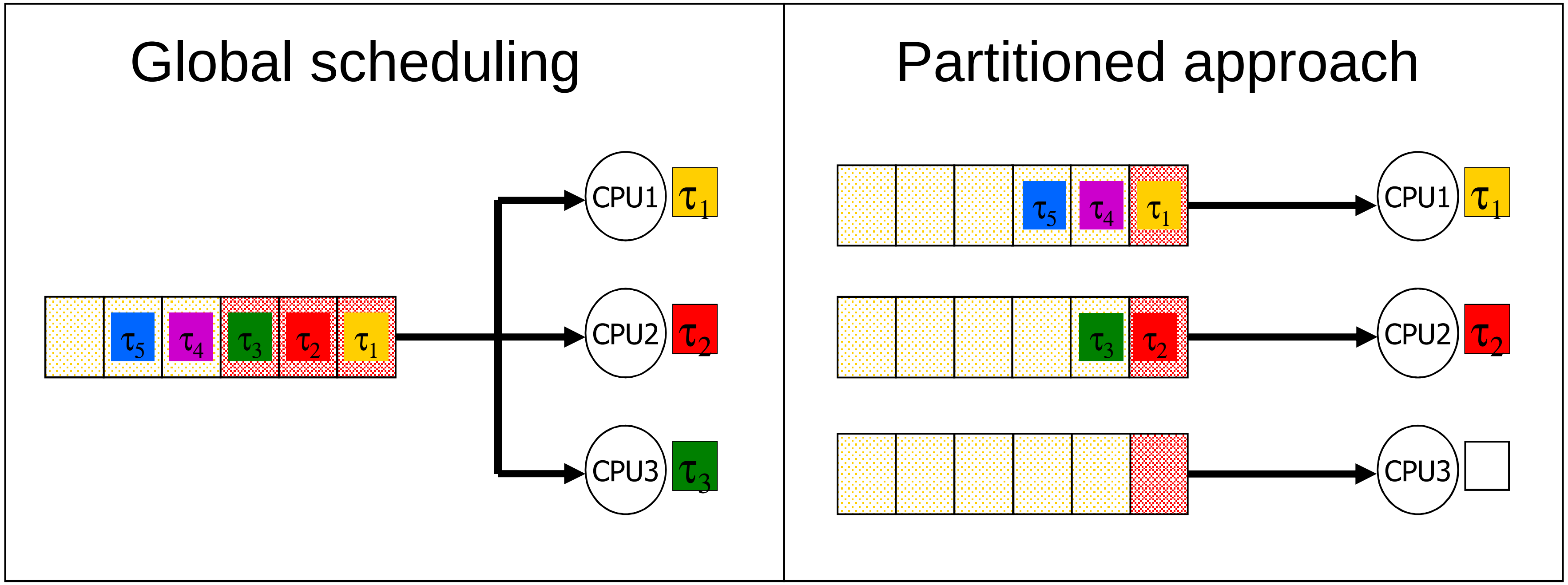
Multiprocessor RT Systems

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Many NP-hard problems
- Few optimality results
- Heuristic approaches
- Simplified task models
- Only sufficient schedulability tests
- Limited RTOS support

Global vs partitioned scheduling

- Single system-wide queue instead of multiple per-processor queues:



Partitioned scheduling

UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- The scheduling problem reduces to:

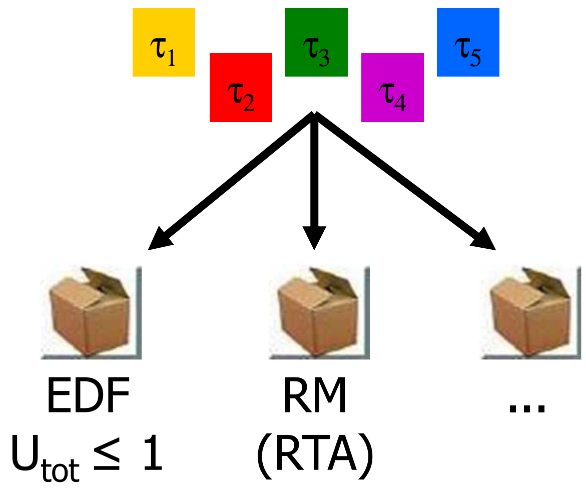
Bin-packing problem + **Uniprocessor scheduling problem**

NP-hard in the strong sense

Well known

⇓

Various heuristics used:
FF, NF, BF, FFDU, BFDD, etc.



Partitioned schedulability

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- [Lopez et al.] EDF-FF gives the best utilization bound among all possible partitioning methods:

$$U^{EDF-FF}(m) = 0.5\ (m+1)$$

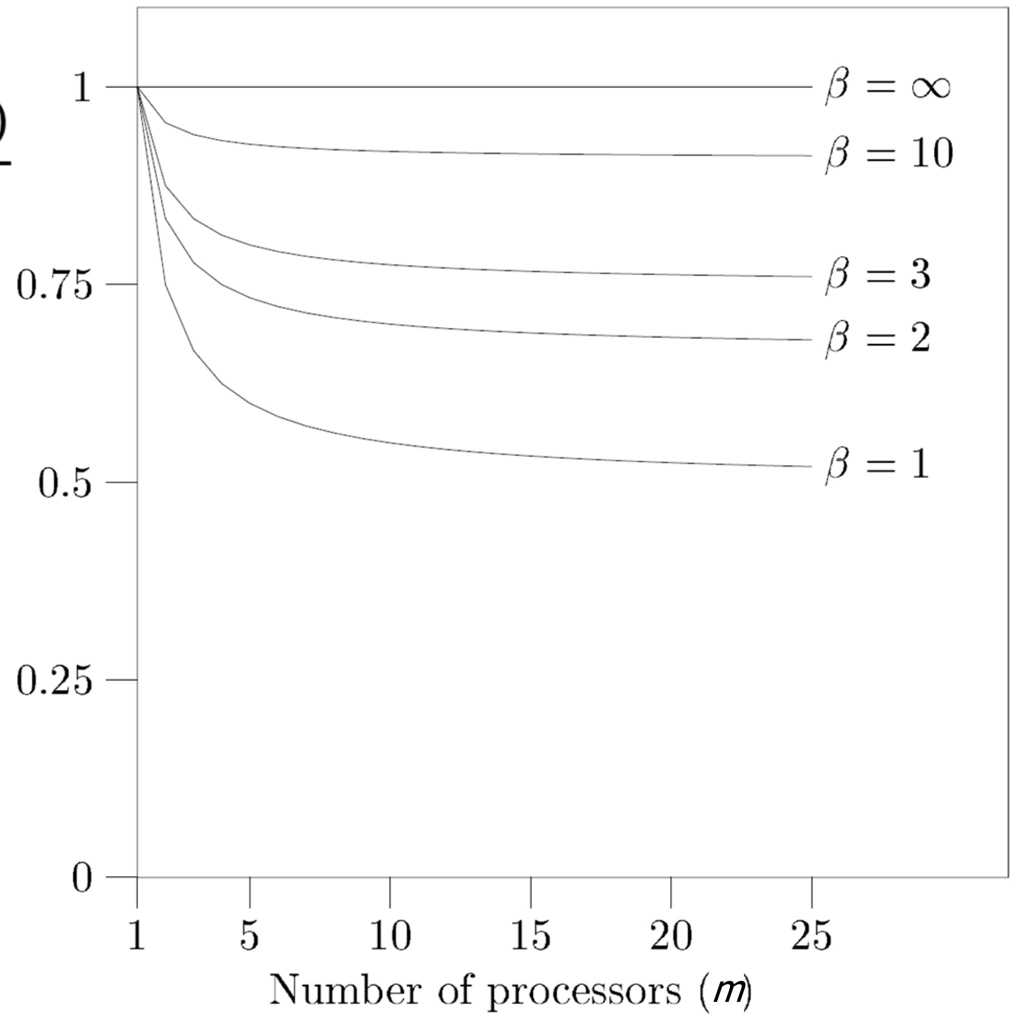
- A refined bound, when U_{max} is the maximum utilization among all tasks, is:

$$U^{EDF-FF}(m,\beta) = \frac{\beta m+1}{\beta+1}\text{, where } \beta = \lfloor 1/U_{max} \rfloor$$

Partitioned schedulability

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

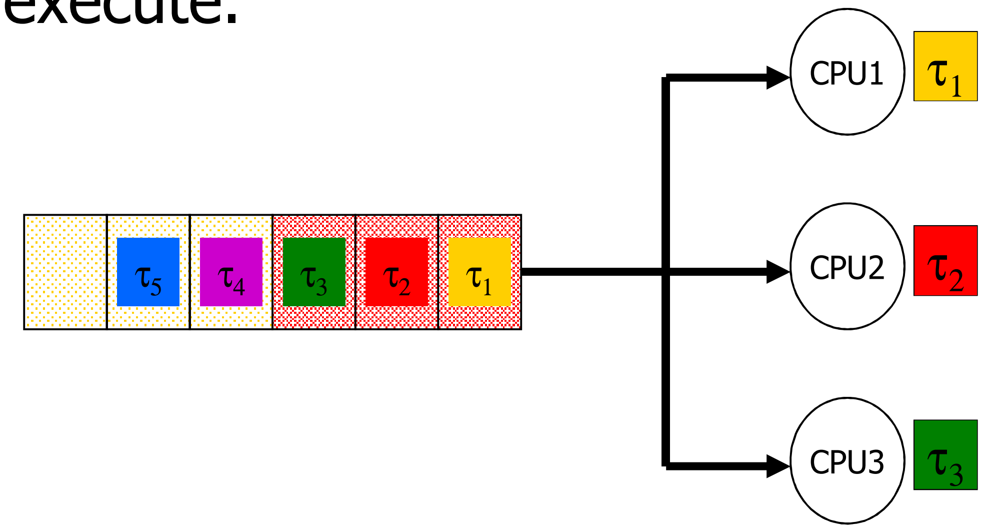
$$\frac{U^{EDF-FF}(m,\beta)}{m}$$



Global scheduling

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- The m highest priority ready jobs are always scheduled
- Work-conserving scheduler
  - No processor is ever idled when a task is ready to execute.



Global scheduling properties

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- ✓ Load automatically balanced
- ✓ Easier re-scheduling (dynamic loads, selective shutdown, etc.)
- ✓ Lower *average* response time (see queueing theory)
- ✓ More efficient reclaiming and overload management
- ✓ Smaller number of preemptions

- ✗ Migration cost: can be mitigated by proper HW (e.g., MPCore's Direct Data Intervention)
- ✗ Few schedulability tests → Further research needed

- Global and partitioned approaches are incomparable

Global scheduling problem

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

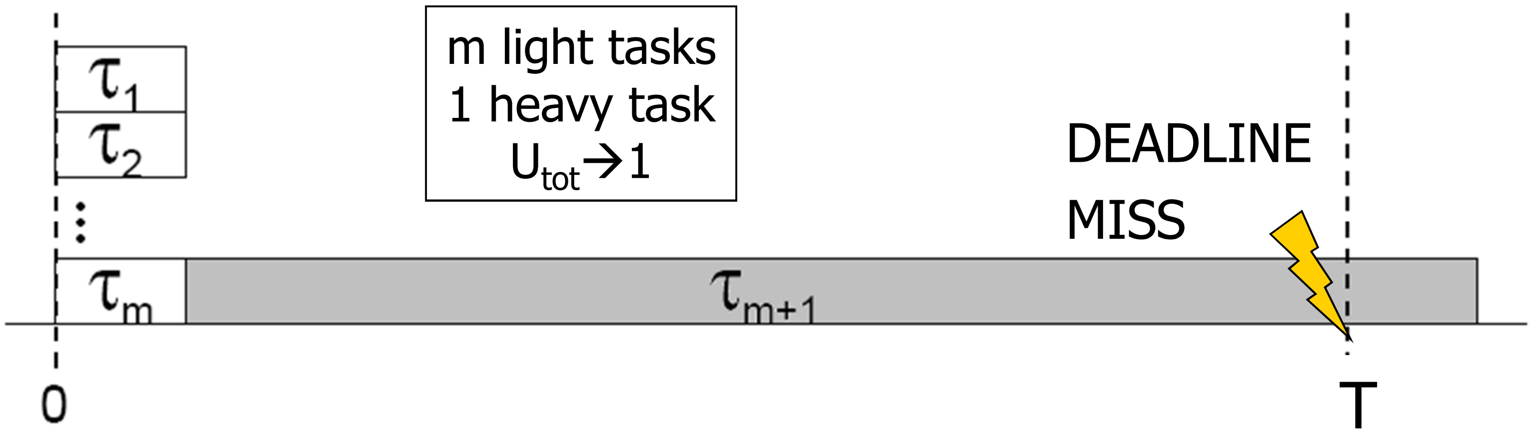
- Optimal algorithms ($U_{tot} \leq m$) are known only for implicit deadline systems:
  - PFair (PF, PD, PD^2), Boundary-Fair (DP-Wrap, LLREF, BF, …), EKG, RUN, U-EDF [ECRTS'12]
  - Preemption and synchronization issues
- No optimal scheduler known for more general task models
- Classic schedulers (e.g., EDF) are not optimal
  - Dhall's effect

Dhall's effect

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Example: **m** processors, **n**=**m+1** tasks, $D_i = T_i$

$\tau_1, \ldots, \tau_m = (1, T-1)$ $\tau_{m+1} = (T, T)$



EDF can fail at very low utilizations

Beyond implicit deadlines

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- No optimal algorithm is known for constrained or arbitrary deadline systems
- No optimal on-line algorithm is possible for arbitrary collection of jobs [Leung and Whitehead]
- Optimal algorithms for sporadic task system with constrained deadlines require clairvoyance [Fisher et al'09]

Global EDF scheduling

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Simple implementation
  - Intuitive priority assignment
  - Reduced scheduling overhead
  - Small number of preemptions/migrations
- Bounded tardiness (as long as $U_{tot} \leq m$)
- Good performance on average
- Many sufficient schedulability tests
  - But most of them are far from tightness
  - Exact tests are intractable

Global EDF: main results

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Many sufficient schedulability tests:

- GFB (RTSJ'01)
- BAK (RTSS'03 → TPDS'05)
- BAR (RTSS'07)
- LOAD (ECRTS'07,ECRTS'08,RTSJ'08 → RTSJ'09)
- BCL (ECRTS'05 → TPDS'09)
- RTA (RTSS'07)
- FF-DBF (ECRTS'09)

Most tests are incomparable

Critical instant

- A particular configuration of releases that leads to the largest possible response time of a task.
- Possible to derive exact schedulability tests analyzing just the critical instant situation.
- Uniprocessor FP and EDF: a critical instant is when
  - all tasks arrive synchronously
  - all jobs are released as soon as permitted

Multiprocessor anomaly

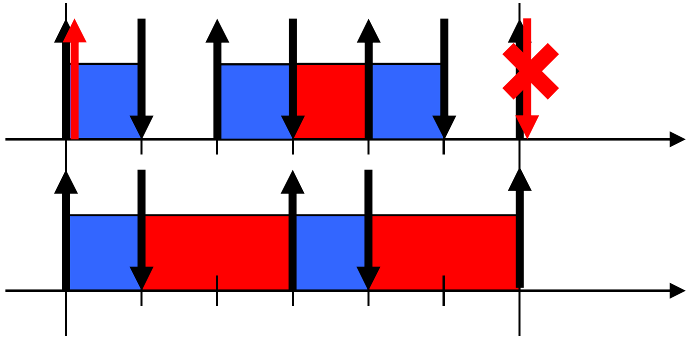
UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Synchronous periodic arrival of tasks is not a critical instant for multiprocessors:

$\tau_1 = (1,1,2)$
$\tau_2 = (1,1,3)$
$\tau_3 = (5,6,6)$

from [Bar07]

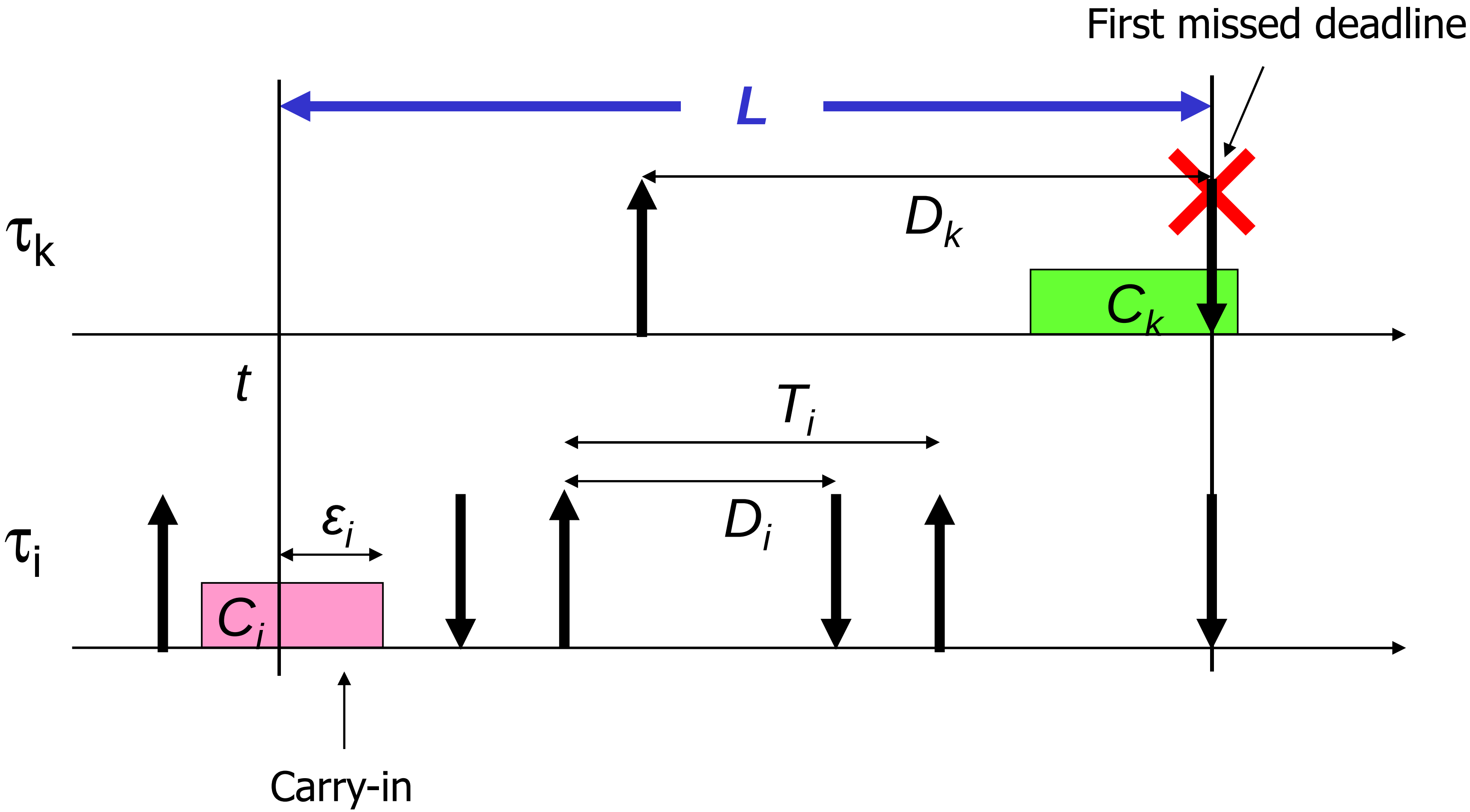
Second job of τ_2 delayed by one unit



Need to find pessimistic situations to derive sufficient schedulability tests

Problem window

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA



Adopted techniques

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

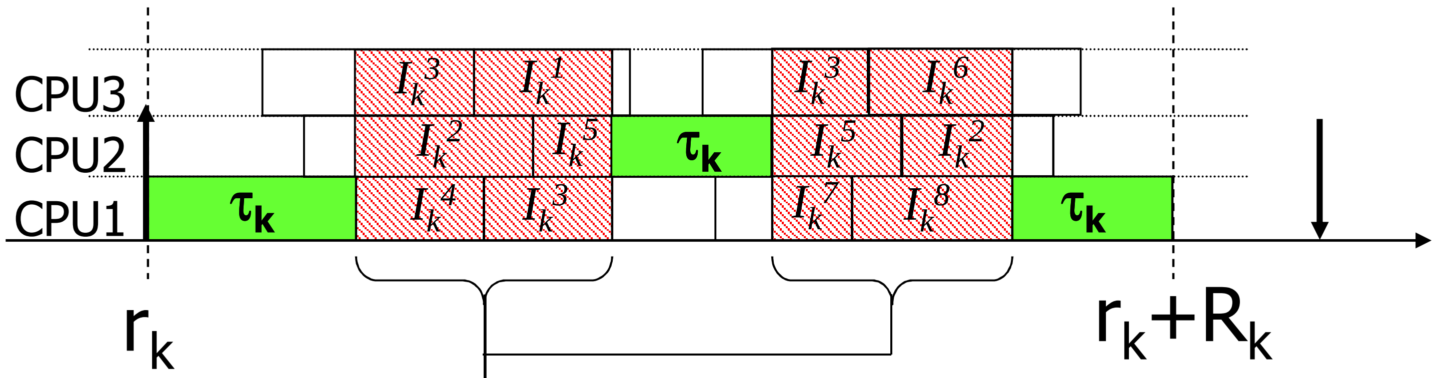
- Consider the interference on the problem job
- Bound the interference with the workload
- Use an upper bound on the workload
- Existing schedulability tests differ in
  - Problem window selection: L
  - Carry-in bound ε_i in the considered window
    - Amount of each contribution (BAK, LOAD, BCL, RTA)
    - Number of carry-in contributions (BAR, LOAD)
    - Total amount of all contributions (FF-DBF, GFB)

Introducing the interference

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

I_k = Total interference suffered by task τ_k

I_k^i = Interference of task τ_i on task τ_k



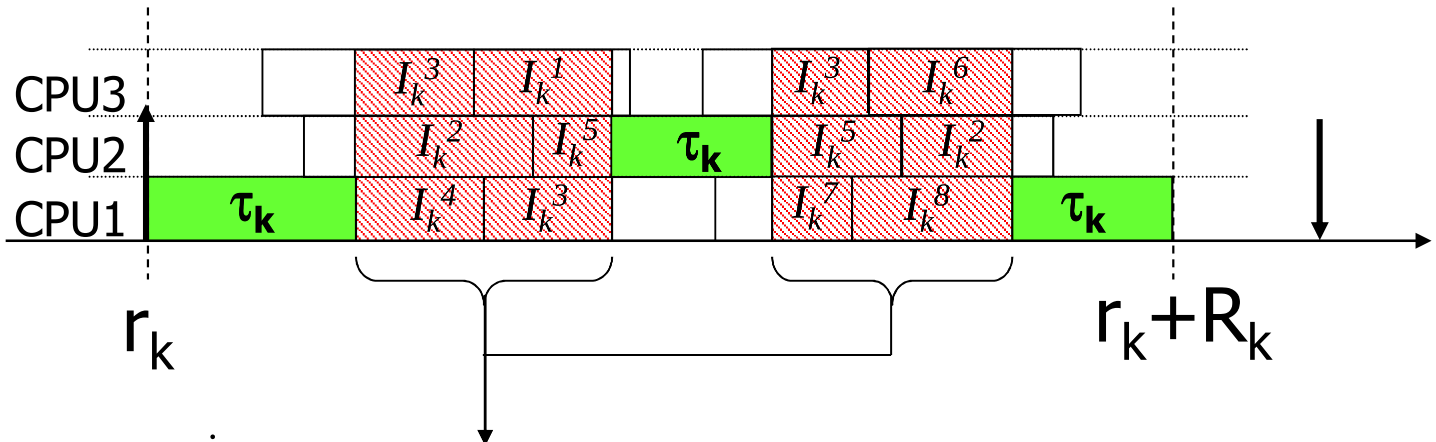
$$I_k(R_k) = \left\lfloor \frac{\sum I_k^i(R_k)}{m} \right\rfloor$$

$$R_k = C_k + I_k(R_k) = C_k + \left\lfloor \frac{1}{m} \sum_{i \neq k} I_k^i(R_k) \right\rfloor$$

Limiting the interference

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

It is sufficient to consider at most the portion (R_k-C_k+1) of each term I_i^k in the sum



$$I_k^i(R_k) \le I_k(R_k) < R_k - C_k + 1$$

It can be proved that $WCRT_k$ is given by the fixed point of:

$$R_k \leftarrow C_k + \left\lfloor \frac{1}{m} \sum_{i \neq k} \min(I_k^i(R_k), R_k - C_k + 1) \right\rfloor$$

Bounding the interference

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Exactly computing the interference is complex

Pessimistic assumptions:

1. Bound the interference of a task with the workload:

$$I_k^i(R_k) \leq W_i(R_k)$$

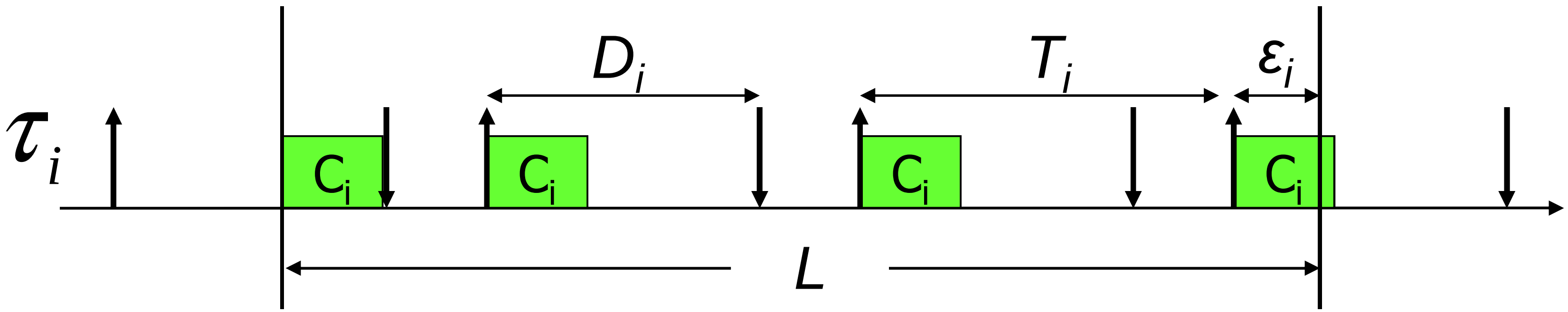
2. Use an upper bound on the workload.

Bounding the workload

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Consider a situation in which:

- The first job executes as close as possible to its deadline
- Successive jobs execute as soon as possible



$$W_i(L) \leq w_i(L) = N_i(L)C_i + \varepsilon_i(L)$$

where:

$$N_i(L) = \left\lfloor \frac{L + D_i - C_i}{T_i} \right\rfloor$$ (# jobs excluded the last one)

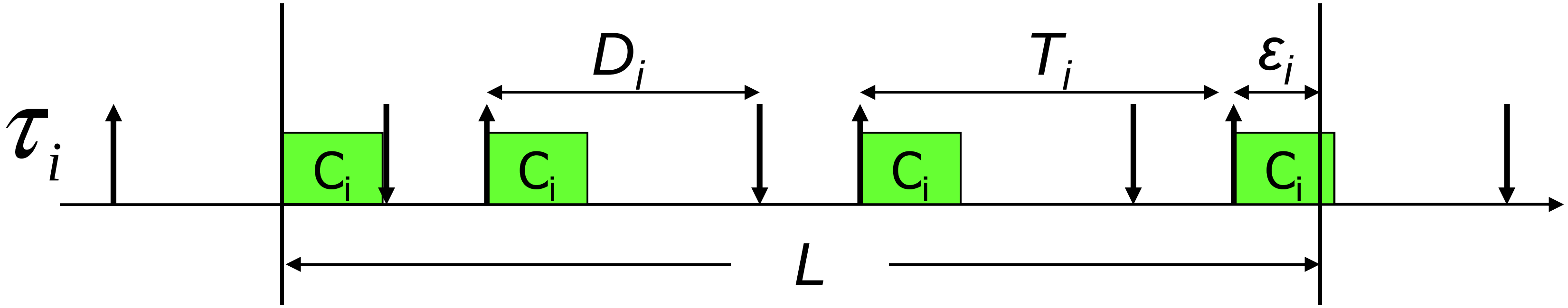
$$\varepsilon_i(L) = \min(C_i, L_i + D_i - C_i - N_i(L)T_i)$$ (last job)

Bounding the workload

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Consider a situation in which:

- The first job executes as close as possible to its deadline
- Successive jobs execute as soon as possible



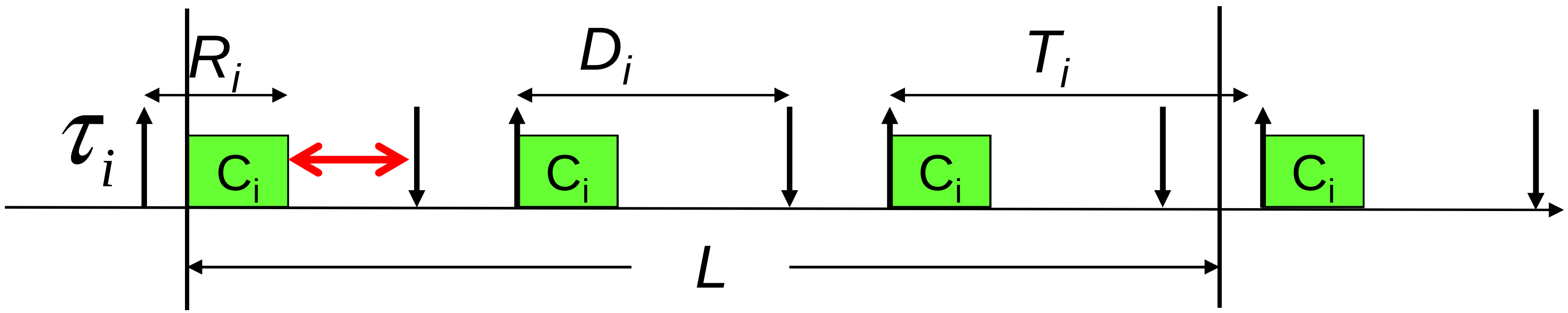
An upper bound on the WCRT of task k is given by the fixed point of R_k in the iteration:

$$R_k \leftarrow C_k + \left\lfloor \frac{1}{m} \sum_{i \neq k} \min(w_i(R_k), R_k - C_k + 1) \right\rfloor$$

Interference refinement

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

The computed bound R_i can be used to improve the interference estimation



$$W_i(L) \le w_i(L, R_i) = N_i(L, R_i)C_i + \varepsilon_i(L, R_i)$$

where:

$$\begin{cases} N_i(L, R_i) = \left\lfloor \dfrac{L + R_i - C_i}{T_i} \right\rfloor \\ \varepsilon_i(L, R_i) = \min(C_i, L_i + R_i - C_i - N_i(L, R_i)T_i) \end{cases}$$

Iterative schedulability test

1. All response times R_i initialized to D_i
2. Compute response time bound for tasks 1,…,n
   - if smaller than old value → update R_i
   - If $R_i > D_i$, mark as temporarily not schedulable
3. If all tasks have $R_i \leq D_i$ → return *success*
4. If no response time has been updated for tasks 1,…,n → return *fail*
5. Otherwise, return to point 2

Considerations

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Very good performances
  - Allows finding the largest number of EDF schedulable task sets for various load distributions
- Pseudo-polynomial complexity
  - A simpler version takes $O(n^2)$
- Fast average behavior

Conclusions

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

- Real-Time systems need to deal with the multicore revolution
- Multiprocessor Real-Time systems are "difficult"
  - No critical instant
  - Optimality often needs clairvoyance
- Many sufficient schedulability tests
  - Often far from tight conditions
  - Exact tests are intractable
- Further research is needed!

marko.bertogna@unimore.it

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA