



**Memo**  
about [INFO-F404](#)

## Real-Time Operating Systems

course by [Joël GOOSSENS](#), 2016

Master in Computer science 1  
Université Libre de Bruxelles  
Computer Science Department



Notes by  
Olivier PIRSON  
[olivier\\_pirson\\_opi@yahoo.fr](mailto:olivier_pirson_opi@yahoo.fr)  
<http://www.opimedia.be/>

August 21, 2017

Cover [picture](#) by David~O.

# 1 Introduction

	$T = D$	$C$	$load$
$\tau_1$	20	4	20%
$\tau_2$	10	2	20%
$\tau_3$	5	3	60%
			100%

Round Robin failed  
RM success

## 2 Uniprocessor real-time scheduling

Correctness: logical result of computations *and* time of the result

- **Hard-real time**: catastrophic consequences
- **Firm real-time**: some deadline may be missed
- **Soft real-time**: deadline missed not important

Job  $J_i = (a, e, d)$

- $a$ : release time
- $e$ : computer requirement (#CPU unit must be received in  $[a, d)$ )
- $d$ : absolute deadline

Periodic task  $\tau_i = (O_i, T_i, D_i, C_i)$

- $O_i$  offset: release time of the 1<sup>st</sup> job
- $T_i$ : period between 2 consecutive task releases
- $D_i$  relative deadline: time delay between time release and absolute deadline
- $C_i$  computation time: worst-case execution requirement

Induce infinite sequence of jobs:  $J_{i,1}, J_{i,2}, J_{i,3}, \dots, J_{i,k}, \dots$

$$a = O_i + (k-1)T_i \qquad e = C_i \qquad d = a + D_i$$

**Sporadic task**:  $T$  is the *minimum* between 2 consecutive task release

**Periodic/sporadic system**: finite set of periodic/sporadic tasks  $\tau = \{\tau_1, \dots, \tau_n\}$

$$\text{Task utilization } U(\tau_i) := \frac{C_i}{T_i} \qquad \text{System utilization } U(\tau) := \sum_{\tau_i \in \tau} U(\tau_i)$$

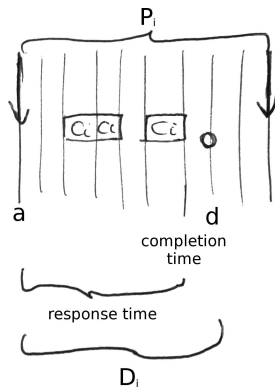
- **Implicit deadline**:  $\forall i : D_i = T_i$
- **Constrained deadline**:  $\forall i : D_i \leq T_i$
- **Arbitrary deadline**
- **Synchronous**:  $\forall i : O_i = 0$
- **Asynchronous**: not synchronous
  
- **Scheduling discipline**: algo. distribute resources
- **Preemptive scheduler**: capable interrupts process to allocate another
  
- **FTP Fixed Task Priority**: offline, one priority by task
- **FJP Fixed Job Priority**: online, one priority by job
- **DP Unrestricted Dynamic Priority**: priority may be change during lifetime of job

Assumptions on this chapter:

- uniprocessor

- hard real-time
- preemptible tasks/jobs
- **work-conserving**: CPU cannot be idle if exist active jobs
- tasks are independent (only CPU is shared)

**Job response time** = completion time - release time



**Lemma 6.**  $U(\tau) \leq 1 \iff \tau$  schedulable

## 2.1 FTP

WLOG:  $\tau_1 > \tau_2 > \tau_3 > \dots > \tau_n$  ( $\approx$ sort,  $O(n \log n)$ )

### 2.1.1 RM Rate Monotonic: lower period, higher priority

For synchronous, *implicit* deadline

- **System FTP-feasible**: if exists schedulable FTP-priority assignment
- **Assignment FTP-optimal**: if schedules *all* FTP-feasible system

**Critical instant of  $\tau_i$** : instant s.t. job release at this instant has maximum response time of all jobs in  $\tau_i$

**Lemma 10** (LIU, 2000).

$J_{i,k}$  release at same time  $t$  job of every higher priority task  
 $\iff t$  is critical instant of  $\tau_i$

**Lemma 11.**  $\tau$  periodic,  $S_1, S_2$  schedules of work-conserving schedulers for  $\tau$   
 $\forall t: S_1$  is idle at  $t \iff S_2$  is idle at  $t$

**Thm 12.** synchronous, *implicit* deadline  
 RM is an FTP-optimal priority assignment

$r_i^1$ : response time of the 1<sup>st</sup> request of  $\tau_i$

**Thm 13.** synchronous, *constrained* deadline  
 FTP-schedulable  $\iff \forall i: r_i^1 \leq D_i$

AUDSLEY & TINDELL, 1991:  $r_i^1 = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{C_j}{T_j} \right\rceil C_j$

Fixed point iteration (until  $w_{k+1} = w_k$  or  $w_k > D_i$ ):  $w_0 := C_i$

$$w_{k+1} := C_i + \sum_{j=1}^{i-1} \left\lceil \frac{w_k}{T_j} \right\rceil C_j$$

**Thm 14** (LEUNG, 2004). periodic, synchronous, implicit deadline

$$U(\tau) \leq n(\sqrt[n]{2} - 1) \implies \tau \text{ feasible with RM}$$

**Corollary 17.** periodic, synchronous, implicit deadline

$$U(\tau) \leq \ln 2 = 0.693\dots \implies \tau \text{ feasible with RM}$$

	$T_i = D_i$	$C_i$
$\tau_1$	10	4
$\tau_2$	15	3
$\tau_3$	20	7

not schedulable with RM

### 2.1.2 DM Deadline Monotonic: lower relative deadline, higher priority

For synchronous, *constrained* deadline (RM  $\subset$  DM)

**Thm 16.** synchronous, *constrained* deadline

DM is an FTP-optimal priority assignment

	$T_i$	$D_i$	$C_i$
$\tau_1$	100	110	52
$\tau_2$	140	154	52

RM:  $\tau_1 > \tau_2$  failed  
 DM:  $\tau_1 > \tau_2$  failed  
 but  $\tau_2 > \tau_1$  success

$\implies$  RM  
 DM are not optimal for arbitrary deadline

**Feasibility interval:**

finite interval s.t. if no deadline missing during this interval  $\implies$  system feasible

**Thm 18.** synchronous, *constrained* deadline

$[0, \max_i D_i]$  is a feasibility interval

$x \in \mathbb{N}$  is an **idle point** if all requests occurring  $< x$  are completed  $\leq x$

CPU is idle at  $x \implies x$  is an idle point

$[a, b)$  is an **elementary busy period** if  $\left\{ \begin{array}{l} a, b \text{ are idle points} \\ \text{idle point} \notin (a, b) \end{array} \right.$

**Level- $i$  busy period:**  $\left\{ \begin{array}{l} \text{elementary busy period for } \{\tau_1, \tau_2, \dots, \tau_i\} \\ \tau_i \text{ executes at least 1 job} \end{array} \right.$

**Thm 22.** synchronous:

$\lambda_j$  is the largest response time for a request of  $\tau_i \in$  first level- $i$  busy period  $[0, \lambda_j)$

$\lambda_j$  is the first idle point  $> 0$  for  $\{\tau_1, \tau_2, \dots, \tau_j\}$

$[0, \lambda_j)$  is the largest level- $i$  busy period

**Thm 23.** synchronous, *arbitrary* deadline:  
 $[0, \lambda_n)$  is a feasibility interval

$$\lambda_n \text{ is the smallest solution of } \lambda_n = \sum_{i=1}^n \left\lceil \frac{\lambda_n}{T_i} \right\rceil C_i$$

$$\text{Fixed point iteration: } w_0 := \sum_{i=1}^n C_i$$

$$w_{k+1} := \sum_{i=1}^n \left\lceil \frac{w_k}{T_i} \right\rceil C_i$$

### 2.1.3 Asynchronous

**Hyper-period**  $P := \text{lcm}_i T_i$

$O_{max} := \max_i O_i$

**Thm 24** (LEUNG & WHITEHEAD, 1982). asynchronous, *constrained* deadline:  
 $[O_{max}, O_{max} + 2P)$  is a feasibility interval

**Thm 25.** periodic task:  
feasibility schedules are periodic

**Thm 26.** asynchronous, *constrained* deadline,  $\tau_1 > \tau_2 > \dots > \tau_n$

$$S_1 := O_1$$

$$S_i := O_i + \left\lceil \frac{(S_{i-1} - O_i)^+}{T_i} \right\rceil T_i$$

$$x^+ := \max\{0, x\}$$

Schedule is feasible up to  $S_n + P \implies \left\{ \begin{array}{l} \text{is feasible} \\ \text{periodic from } S_n \\ \text{period } P \end{array} \right.$

$\implies [0, S_n + P)$  is a feasibility interval

	$O_i$	$T_i = D_i$	$C_i$
$\tau_1$	10	12	1
$\tau_2$	0	12	6
$\tau_3$	0	8	3

RM, DM:  $\tau_3 > \tau_1 > \tau_2$  failed  
but  $\tau_3 > \tau_2 > \tau_1$  success

$\implies \left. \begin{array}{l} \text{RM} \\ \text{DM} \end{array} \right|$  are not optimal for asynchronous

### 2.1.4 AUDSLEY algorithm

$\tau_i$  is **lowest-priority viable**  $\iff$  all jobs meet their deadline when  
 $\left\{ \begin{array}{l} \tau_i \text{ has the lowest priority} \\ \text{other tasks have any higher priority} \\ \text{when scheduling } \tau_j \neq \tau_i \text{ consider deadline softs} \end{array} \right.$

**Thm 28.**  $\tau_i$  lowest-priority viable:

$\exists$  feasible FTP-assignment for  $\tau \iff \exists$  feasible FTP-assignment for  $\tau \setminus \{\tau_i\}$

## AUDSLEY algorithm

```

procedure Audsley( $\tau$ ) {
  if (there is no lowest-priority viable task) {
    return infeasible
  }
  else {
     $\tau_i :=$  a lowest-priority viable task in  $\tau$ 
    assign lowest priority to  $\tau_i$ 
    Audsley( $\tau \setminus \{\tau_i\}$ )
  }
}

```

Checking if a task is lower-priority viable on  $[0, S_n + P_n)$   
 $O(n^2)$  distinct priority assignments  $\ll O(n!)$

## 2.2 FJP

**Earliest Deadline First EDF:** lower *absolute* deadline, higher priority

Job set  $J$  is **feasible** if  $\exists$  schedule which meets all the job deadlines

**Thm 30.** Job set  $J$  is feasible  $\implies$  EDF feasibly schedule  $J$

EDF is  $\left\{ \begin{array}{l} \text{FJP-optimal} \\ \text{FTP-optimal for } \left\{ \begin{array}{l} \text{asynchronous} \\ \text{arbitrary deadline} \end{array} \right. \end{array} \right.$

**Thm 31.** Periodic, implicit deadline:

$$U(\tau) \leq 1 \iff \tau \text{ schedulable}$$

**Corollary 32.** Periodic, implicit deadline:

$$U(\tau) \leq 1 \iff \tau \text{ is EDF-schedulable}$$

**Thm 33.**

$\tau \left\{ \begin{array}{l} \text{synchronous} \\ \text{periodic} \\ \text{arbitrary deadline} \\ \text{EDF-feasible} \end{array} \right. \implies \left\{ \begin{array}{l} \forall \text{ corresponding } \left\{ \begin{array}{l} \text{asynchronous} \\ \text{periodic} \end{array} \right\}: \text{EDF-feasible} \\ \forall \text{ corresponding sporadic: EDF-feasible} \end{array} \right.$

	$T_i = D_i$	$C_i$
$\tau_1$	4	2
$\tau_2$	7	3

response time: 5, 5, 5, **6** (the largest is not the first)

**Thm 34** (LIU, LEYLAND, 1973). periodic, synchronous, constrained deadline:  
 deadline miss in  $t \implies \nexists \text{ idle} < t$



**Thm 35** (GOOSENS, 1999). periodic, synchronous, constrained deadline: deadline miss in  $t \implies \nexists$  (except 0) idle point  $< t$

$\implies [0, L)$  is a feasibility interval  $L$  is the first idle point (except 0)

$$L (= \lambda_n) = \text{smallest solution of } L = \sum_{i=1}^n \left\lceil \frac{L}{T_i} \right\rceil C_i$$

Periodic system  $R$ , schedule  $S$ :

**Configuration**  $C_S(R, t) := ((\gamma_1(t), \alpha_1(t), \beta_1(t)), \dots, (\gamma_n(t), \alpha_n(t), \beta_n(t)))$

where  $\left\{ \begin{array}{l} \gamma_i(t) := \begin{cases} (t - O_i) \% T_i & \text{if } t \geq O_i \\ t - O_i & \text{if } t < O_i \end{cases} \\ \alpha_i(t) := \# \text{ active jobs of } \tau_i \\ \beta_i(t) := \text{cumulative CPU time used by oldest active job of } \tau_i \end{array} \right.$

**Thm 37.**  $R$  periodic, asynchronous, arbitrary deadline,  $S = \text{EDF}$ :

$R$  is feasible  $\iff \left\{ \begin{array}{l} \text{all deadline } \in [0, O_{\max} + 2P) \text{ met} \\ C_S(R, O_{\max} + P) = C_S(R, O_{\max} + 2P) \end{array} \right.$

	$O_i$	$T_i$	$D_i$	$C_i$	$U$	
$\tau_1$	0	4	4	2	$1/2$	all deadline $[0, 10)$ met but $\beta_2(6) = 2 \neq \beta_2(10) = 1$ and $\tau_2$ misses at $t = 21$
$\tau_2$	2	4	7	3	$3/4$	
					$5/4$	

**Corollary 38.** periodic, asynchronous, arbitrary deadline:

$U(\tau) \leq 1 \implies [0, O_{\max} + 2P)$  is a feasibility interval

## 2.3 DP

Job  $J = (a, e, d)$

$\epsilon_J(t) :=$  cumulative CPU time used by  $J$  since  $a$ .

**laxity**  $l_J(t) := d - t - (e - \epsilon_J(t))$

$> 0$ :  $J$  can wait  $l_J(t)$  time-units

$= 0$ :  $J$  must run immediately and until  $d$

$< 0$ :  $J$  will miss



$J$  running  $\implies l_J(t+1) = l_J(t)$

**LLF Least Laxity First**

**Thm 41** (MOK, 1983). LLF is DM-optimal

	$T_i$	$D_i$	$C_i$
$\tau_1$	10	8	4
$\tau_2$	10	9	5

EDF: preemptive free  
 LLF: 6 preemptions by 10 time-units

**Trashing:** large amount of resources are used to do a minimal work

$$l_{J_1}(t) = l_{J_2}(t) \mid \text{run } J_1 \implies l_{J_1}(t) = l_{J_2}(t) + 1$$

$\exists$  FJP-optimal  
 LLF cause lot of preemptions  $\mid$  so LLF useless

### 3 Multiprocessor real-time scheduling

Consider only limited parallel architecture. At each instant:

- one processor executes one task at most
  - one task is execute on one processor at most (no task or job parallelism)
  - strongly coupled systems (common time reference and shared memory)
  - identical parallel machines:  $\pi_1, \pi_2, \dots, \pi_m$
  - **Identical parallel machines:** all processors are identical (same computing power)
  - **Uniform parallel machines:** computing capacity  $s_i$  for each processor  $\pi_i$
  - **Unrelated parallel machines:** execution rate  $s_{i,j}$  for each job-processor  $(J_i, \pi_j)$
- Identical  $\subset$  Uniform  $\subset$  Unrelated
- **Partitioned scheduling:** divide system in subsystems associate to 1 processor
  - **Global scheduling:** jobs can to migrate during their lifetime

**Lemma 1** (BARUAH, 2007). Exists systems schedulable with global FJP algorithms but not partitioned.

	$T_i$	$D_i$	$C_i$
$\tau_1$	3	2	2
$\tau_2$	4	3	3
$\tau_3$	12	12	5

With 2 processors:  
all partitioned FJP failed  
global FJP with  $> \tau_3$  success

**Lemma 2** (BARUAH, 2007). Exists systems schedulable with partitioned FJP algorithms but not global.

	$T_i$	$D_i$	$C_i$
$\tau_1$	3	2	2
$\tau_2$	4	3	3
$\tau_3$	12	12	4
$\tau_4$	12	12	3

With 2 processors:  
all global FJP failed  
partitioned  $\{\tau_1, \tau_3\}, \{\tau_2, \tau_4\}$  FJP success

	$T_i$	$D_i$	$C_i$
$\tau_1$	4	2	1
$\tau_2$	5	3	3
$\tau_3$	2	3	7

**Density**  $\lambda_i := \frac{C_i}{\min(T_i, D_i)}$

**Total density**  $\lambda_{sum} := \sum_{\tau_i \in \tau} \lambda(\tau_i)$

**Maximal density**  $\lambda_{max} := \max_{\tau_i \in \tau} \lambda(\tau_i)$

sporadic tasks,  $m$  identical processors:

$$\left. \begin{array}{l} \lambda_{sum}(\tau) \leq m \\ \lambda_{max}(\tau) \leq 1 \end{array} \right| \iff \tau \text{ schedulable}$$

### 3.1 Partitioned scheduling

For *implicit* deadline

**Bin-Packing problem:**  $k$  objects of different sizes in a minimum of bins.

NP-complete, so heuristics.

Greedy algorithm, with optional pre-processing (sort by **DU** decreasing utilization or **IU** increasing utilization):

- **FF** First-fit
- Best-fit
- Worst-fit
- Next-fit

$$\text{RM: } U(\tau) \leq n_\pi (\sqrt[n_\pi]{2} - 1) \qquad n_\pi := \# \text{ tasks on processor } \pi$$

**Thm 5.** Sporadic, implicit deadline:

$$\left. \begin{array}{l} U(\tau) \leq \frac{m+1}{2} \\ U_{\max} \leq 1 \end{array} \right| \implies \tau \text{ schedulable with FFDU (with EDF local scheduler)}$$

### 3.2 Global scheduling

**Online scheduler:** take scheduling decisions at *runtime* with jobs already released and *without* knowledge of future

**Thm 7** (HONG & LEUNG, 1988).

$m > 1 \implies \nexists$  online and optimal scheduling

	$T_i = D_i$	$C_i$
$\tau_1$	2	4
$\tau_2$	2	4
$\tau_3$	4	8

**Intuitively positive change:** decrease  $U$  (increase  $T_i$ , decrease  $D_i$ ...)

**Anomaly** if schedulable system become unschedulable with intuitively positive change

Sporadic: no sufficient to check synchronous periodic sub-case (critical instant is an open question)

A schedulability test is **sustainable** if schedulable system remains schedulable with these possible changes:

- decreased  $C_i$  (**Sustainable with respect to the execution requirements**)
- later arrival times

- larger  $D_i$

**Thm 12** (CUCU-GROSJEAN & GOOSSENS, 2010). Unrelated platforms:  
FJP schedulers are sustainable with respect to the execution requirements

**Thm 13.** Periodic, implicit deadline:

$$\left. \begin{array}{l} U(\tau) \leq m \\ U_{\max}(\tau) \leq 1 \end{array} \right| \iff \tau \text{ feasible}$$

### 3.2.1 PFAIR scheduling

**Time slice** or **quantum**: period of time during a process is allowed to run

$$\begin{aligned} \text{Schedule } S: \tau \times \mathbb{N} &\rightarrow \{0, 1\} \\ (\tau_i, t) &\mapsto \begin{cases} 1 & \text{if } \tau_i \text{ scheduled in slice } [t, t+1) \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

For ideal **fair** schedule each  $\tau_i$  received  $U(\tau_i).t$  slices during  $[0, t)$  (implies all deadline are met)

$$\text{lag}(\tau_i, t) := U(\tau_i).t - \sum_{l=0}^{t-1} S(\tau_i, l)$$

Schedule has **PFAIR** property  $\iff -1 < \text{lag}(\tau_i, t) < 1, \forall \tau_i \in \tau, \forall t \in \mathbb{N}$   
 $\iff \forall \tau_i \in \tau: \tau_i$  receive  $\lfloor U(\tau_i).t \rfloor$  or  $\lceil U(\tau_i).t \rceil$  in  $[0, t)$

**Thm 15.** PFAIRness  $\implies$  all deadline are met

**Thm 16** (BARUAH, COHEN, PLAXTON & VARVEL, 1996).

Periodic, synchronous, implicit deadline:

$$\left. \begin{array}{l} U(\tau) \leq m \\ U_{\max}(\tau) \leq 1 \end{array} \right| \iff \exists \text{ PFAIR schedule}$$

$\implies$  PFAIR is optimal

**Thm 17.** Sporadic, arbitrary deadline:

$$\left. \begin{array}{l} \lambda_{\text{sum}}(\tau) \leq m \\ \lambda_{\max}(\tau) \leq 1 \end{array} \right| \implies \text{feasible}$$

**Thm 18** (SRINIVASAN & BARUAH, 2002). Sporadic, implicit deadline:

$$U(\tau) \leq m - (m-1)U_{\max}(\tau) \implies \text{schedulable with global EDF}$$

### 3.2.2 EDF<sup>(k)</sup>

Assume  $U(\tau_1) \geq U(\tau_2) \geq \dots \geq U(\tau_n)$

$\tau^{(i)} := \{\tau_i, \tau_{i+1}, \dots, \tau_n\}$

**EDF<sup>(k)</sup>**:  $\forall i < k$ : set maximal priority to  $\tau_i$  ( $d := -\infty$ )

$\forall i \geq k$ : set EDF priority to  $\tau_i$

EDF = EDF<sup>(1)</sup>

**Thm 20** (GOOSSENS, FUNK, & BARUAH, 2003). Sporadic, implicit deadline:

$m = (k - 1) + \left\lceil \frac{U(\tau^{(k+1)})}{1 - U(\tau_k)} \right\rceil \implies \text{EDF}^{(k)} \text{ schedulable}$

**Corollary 21**. Sporadic, implicit deadline

is schedulable with  $m_{min}(\tau) = \min_{k=1}^n \left\{ (k - 1) + \left\lceil \frac{U(\tau^{(k+1)})}{1 - U(\tau_k)} \right\rceil \right\}$  processors.

$k_{min} := k$  that minimize

## 4 Concurrency

...

## 5 Parallel algorithms

$t_s$ : execution time of uniprocessor (sequential) version

$t_p$ : execution time of multiprocessor (parallel) version with  $n$  processors

Speedup factor  $S(n) := \frac{t_s}{t_p}$  (relative performance difference)

**Linear speedup principle:**

maximum speedup factor that can be gained using  $n$  processors is  $n$

**Law 1 (AMDAHL's law).**

Let  $f$  the fraction of sequential computations of a parallel program.

$(1 - f)$  is the fraction of parallelizable computations.

$$S(n) = \frac{t_s}{f t_s + (1-f) \frac{t_s}{n}} = \frac{n}{1 + (n-1)f}$$

Upper bound of speedup factor (even with unlimited number of processors) :

$$\lim_{n \rightarrow \infty} S(n) = \frac{1}{f}$$

(If only 5% of the computations are sequential, the *potential* maximum speedup factor is 20, regardless of the number of processors.)

...



## References

[INFO-F404 2016] Joël GOOSSENS.

[Course INFO-F404](#) *Real-Time Operating Systems*.

Université Libre de Bruxelles, Computer Science Department, 2016

# Index

$>$ , 3

$a$ , 2

$\alpha_i(t)$ , 7

$\beta_i(t)$ , 7

$C_i$ , 2

$C_S(R, t)$ , 7

$d$ , 2

$D_i$ , 2

$e$ , 2

$\varepsilon_J(t)$ , 7

$\gamma_i(t)$ , 7

$J_i$ , 2

$\lambda_i$ , 9

$\lambda_{max}$ , 9

$\lambda_{sum}$ , 9

$l_J(t)$ , 7

$n_\pi$ , 10

$O_i$ , 2

$O_{max}$ , 5

$P$ , 5

$r_i^1$ , 3

$S$ , 11

$S(n)$ , 14

$\tau^{(i)}$ , 12

$T_i$ , 2

$\tau_i$ , 2

$t_p$ , 14

$t_s$ , 14

$x^+$ , 5

AMDAHL's law, 14

anomaly, 10

asynchronous, 2

AUDSLEY algorithm, 6

Bin-Packing problem, 10

computation time, 2

configuration, 7

critical instant, 3

deadline

arbitrary deadline, 2

constrained deadline, 2

implicit deadline, 2

relative deadline, 2

Deadline Monotonic, 4

density, 9

maximal density, 9

total density, 9

DM, 4

DU, 10

Earliest Deadline First, 6

EDF, 6

EDF<sup>(k)</sup>, 12

elementary busy period, 4

fair, 11

feasibility interval, 4

feasible

feasible job set, 6

system FTP-feasible, 3

FF, 10

fit

best-fit, 10

first-fit, 10

next-fit, 10

worst-fit, 10

Fixed Job Priority, 2

Fixed Task Priority, 2

FJP, 2

FTP, 2

hyper-period, 5

idle point, 4

intuitively positive change, 10

IU, 10

job, 2

lag, 11

laxity, 7

Least Laxity First, 7

- level- $i$  busy period, 4
- linear speedup principle, 14
- LLF, 7
- optimal
  - assignment FTP-optimal, 3
- parallel machines
  - identical parallel machines, 9
  - uniform parallel machines, 9
  - unrelated parallel machines, 9
- PFAIR, 11
- preemptive scheduler, 2
- quantum, 11
- Rate Monotonic, 3
- real-time
  - Firm real-time, 2
  - Hard real-time, 2
  - Soft real-time, 2
- response time, 3
- RM, 3
- schedule, 11
- scheduler
  - online scheduler, 10
- scheduling, 2
  - global scheduling, 9
  - partitioned scheduling, 9
- speedup factor, 14
- sustainable, 10
  - sustainable with respect to the execution requirements, 10
- synchronous, 2
- system
  - periodic system, 2
  - sporadic system, 2
- task
  - periodic task, 2
  - sporadic task, 2
- time slice, 11
- trashing, 8
- UD, 2
- Unrestricted Dynamic Priority, 2
- utilization
  - system utilization, 2
  - task utilization, 2
- work-conserving, 3

# List of Theorems

6	Lemma	3
10	Lemma (LIU, 2000)	3
11	Lemma	3
12	Thm	3
13	Thm	3
14	Thm (LEUNG, 2004)	4
17	Corollary	4
16	Thm	4
18	Thm	4
22	Thm	4
23	Thm	5
24	Thm (LEUNG & WHITEHEAD, 1982)	5
25	Thm	5
26	Thm	5
28	Thm	5
30	Thm	6
31	Thm	6
32	Corollary	6
33	Thm	6
34	Thm (LIU, LEYLAND, 1973)	6
35	Thm (GOOSENS, 1999)	7
37	Thm	7
38	Corollary	7
41	Thm (MOK, 1983)	7
1	Lemma (BARUAH, 2007)	9
2	Lemma (BARUAH, 2007)	9
5	Thm	10
7	Thm (HONG & LEUNG, 1988)	10
12	Thm (CUCU-GROSJEAN & GOOSENS, 2010)	11
13	Thm	11
15	Thm	11
16	Thm (BARUAH, COHEN, PLAXTON & VARVEL, 1996)	11
17	Thm	11
18	Thm (SRINIVASAN & BARUAH, 2002)	11
20	Thm (GOOSENS, FUNK, & BARUAH, 2003)	12
21	Corollary	12
1	Law (AMDAHL's law)	14

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Uniprocessor real-time scheduling</b>	<b>2</b>
2.1	FTP . . . . .	3
2.1.1	RM Rate Monotonic . . . . .	3
2.1.2	DM Deadline Monotonic . . . . .	4
2.1.3	Asynchronous . . . . .	5
2.1.4	AUDSLEY algorithm . . . . .	5
2.2	FJP . . . . .	6
2.3	DP . . . . .	7
<b>3</b>	<b>Multiprocessor real-time scheduling</b>	<b>9</b>
3.1	Partitioned scheduling . . . . .	10
3.2	Global scheduling . . . . .	10
3.2.1	PFAIR scheduling . . . . .	11
3.2.2	EDF <sup>(k)</sup> . . . . .	12
<b>4</b>	<b>Concurrency</b>	<b>13</b>
<b>5</b>	<b>Parallel algorithms</b>	<b>14</b>
	<b>References</b>	<b>15</b>
	<b>Index</b>	<b>16</b>
	<b>List of Theorems</b>	<b>18</b>
	<b>Contents</b>	<b>19</b>





⊗**TEX**<sub>ts</sub>

Last version (with booklet format and  $\LaTeX$  sources)

<https://bitbucket.org/OPiMedia/memo-info-f404-real-time-operating-systems>