Keynote: RTNS 2012

“Getting ones priorities right”

Robert Davis

Real-Time Systems Research Group, University of York

rob.davis@york.ac.uk

What is this talk about?

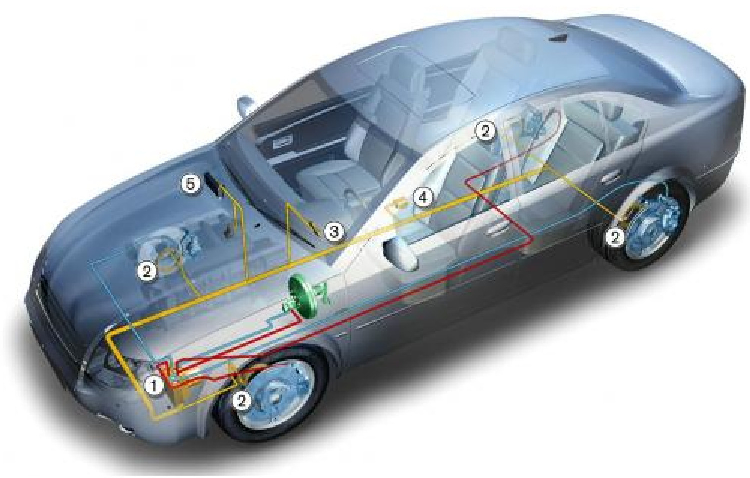
- Fixed Priority scheduling in all its guises
  - Pre-emptive, non-pre-emptive, deferred pre-emption
  - Single processor, multiprocessor
  - Sporadic tasks, mixed criticality, probabilistic execution times etc.

- Priority assignment
  - Why is it important?
  - What is an optimal assignment?
  - How do we find it?
  - Is Optimal Priority Assignment enough?
    Can we optimise other things as well?
  - Unsolved priority assignment problems

Priority assignment

- Why is priority assignment important
  - Achieve a schedulable system when it otherwise wouldn't be
  - Provide a schedulable system avoiding hardware overprovision / maximising use of hardware resources
  - Provide headroom for unforeseen interference or overruns

- Example
  - Controller Area Network (CAN)
  - Used for in-vehicle networks
  - Message IDs are the priorities



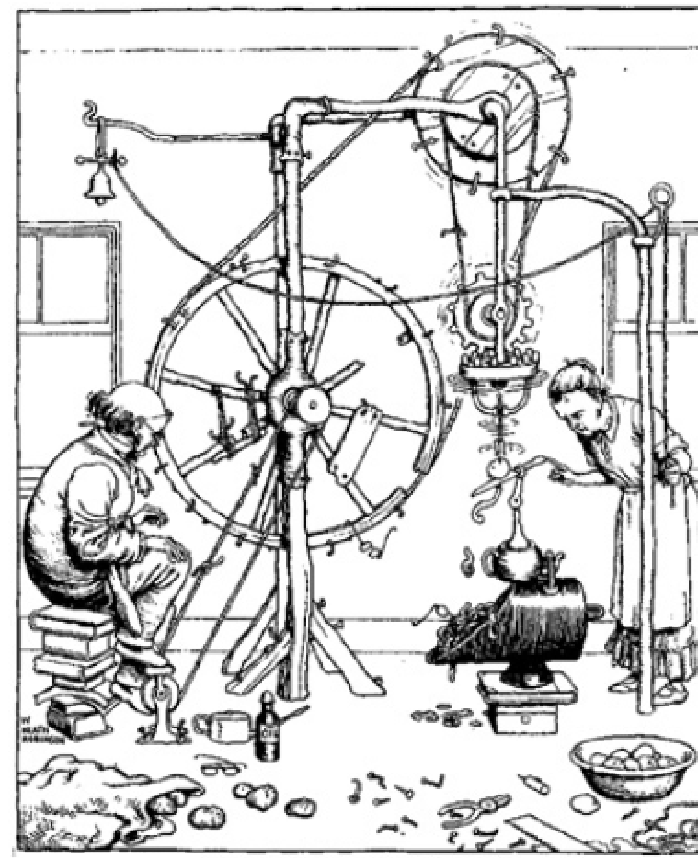
When priority assignment goes bad!

- From Darren Buttle's Keynote at ECRTS 2012

  The myth of CAN bus Utilisation – "You cannot run CAN reliably at more than 30% utilisation[1]"

  [1] Figures may vary but not significantly

- Why?
  - Message IDs i.e. priorities assigned in an ad-hoc way reflecting data and ECU supplier (legacy issues)
  - ...as well as many other issues, including device driver implementation



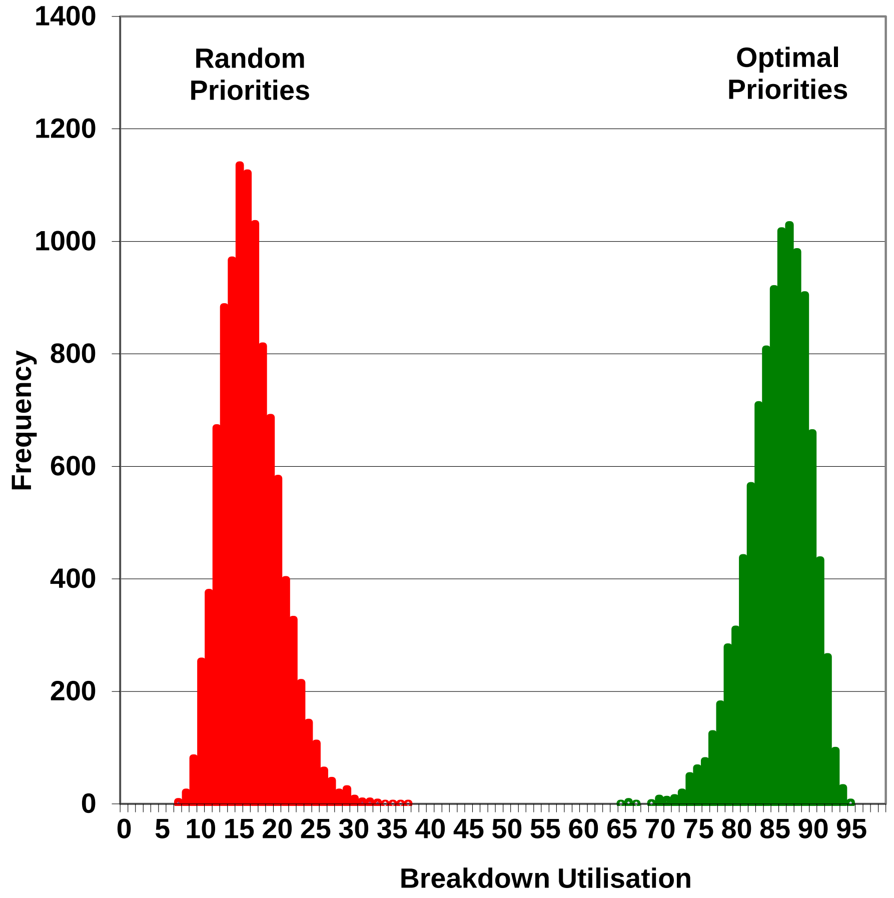
The Professor's invention for peeling potatoes.

When priority assignment goes bad!

- Example: CAN
  - Typical automotive config:
    - 80 messages
    - 10ms -1s periods
    - All priority queues
  - x10,000 message sets

- Breakdown utilisation
  - Scale bus speed to find util. at which deadlines are missed

80% v 30% or less



[R.I. Davis, S. Kollmann, V. Pollex, F. Slomka, "Schedulability Analysis for Controller Area Network (CAN) with FIFO Queues Priority Queues and Gateways". *Real-Time Systems,* 2012]

System model

- Single processor, fixed priority scheduling
  - Scheduler chooses the highest priority ready task to execute

- Periodic / Sporadic task model
  - Static set of n tasks. Each task τ_i has a unique priority i
    - C_i - Execution time (bound)
    - D_i - Relative deadline
    - T_i - Minimum inter-arrival time or period

- Variations
  - Implicit / constrained / arbitrary deadlines
  - Pre-emptive / non-pre-emptive / deferred pre-emption scheduling
  - Unique priorities or shared priority levels

Schedulability

- Schedulability tests
  - Determine if all jobs of a task (all tasks) can be guaranteed to meet their deadlines for all valid arrival patterns
  - Sufficient if all of the tasksets that the test deems to be schedulable are in fact schedulable
  - Necessary if all of the tasksets that the test deems to be unschedulable are in fact unschedulable
  - Exact implies both sufficient and necessary

- Worst-case response times
  - Schedulability tests often compute the worst-case response time R_i for each task and compare it with the task's deadline D_i to determine schedulability

Definition:
Optimal priority assignment policy

For a given system model, a priority assignment policy P is referred to as **optimal** if there are no systems, compliant with the model, that are schedulable using another priority assignment policy that are not also schedulable using policy P.

according to the test

according to the test

An optimal priority assignment policy can schedule any system that can be scheduled using any other priority assignment

May also consider priority assignment policies that are optimal with respect to a specific (sufficient) schedulability test

[N.C. Audsley, "Optimal priority assignment and feasibility of static priority tasks with arbitrary start times", Technical Report YCS 164, Dept. Computer Science, University of York, UK, 1991.]

[N.C. Audsley, “On priority assignment in fixed priority scheduling”, Information Processing Letters, 79(1): 39-44, May 2001.]

[R.I. Davis and A. Burns "Improved Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems”. *Real-Time Systems*, (2011) Volume 47, Number 1, pages 1-40]

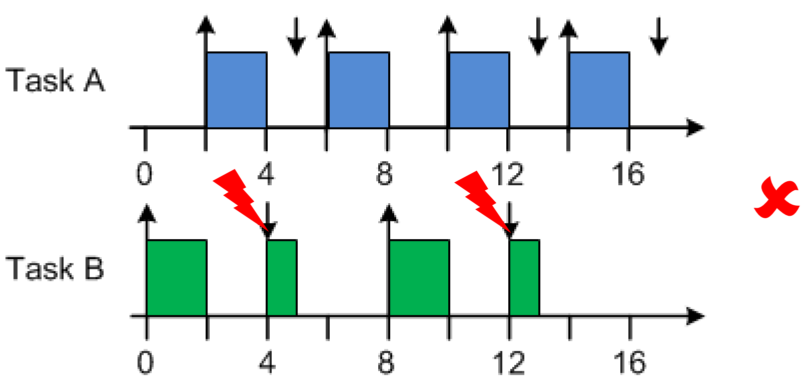
Early work on priority assignment

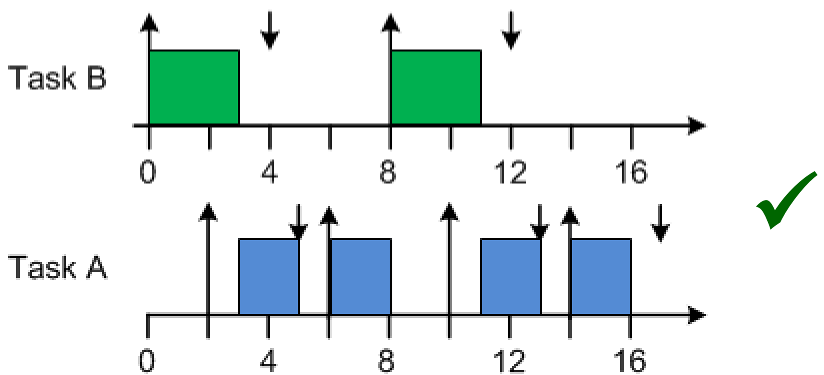
- 1967 Fineberg & Serlin
  - Two periodic tasks with implicit deadlines, better to assign the higher priority to the task with the shorter period
- 1973 Liu & Layland
  - Rate-Monotonic priority ordering is optimal for implicit deadline periodic tasksets (synchronous arrivals)
- 1982 Leung & Whitehead
  - Deadline-Monotonic priority ordering is optimal for constrained deadline tasksets (synchronous arrivals)
  - Deadline Monotonic not optimal for the asynchronous case (offsets)
- 1990 Lehoczky
  - Deadline Monotonic not optimal for arbitrary deadline tasksets
- 1994 Burns et al.
  - Deadline Monotonic not optimal for deadlines prior to completion
- 1996 George
  - Deadline Monotonic not optimal for non-pre-emptive scheduling

Deadline Monotonic: non-optimality

- Tasks with offsets

Task	Execution Time	Deadline	Period	Offset
A	2	3	4	2
B	3	4	8	0



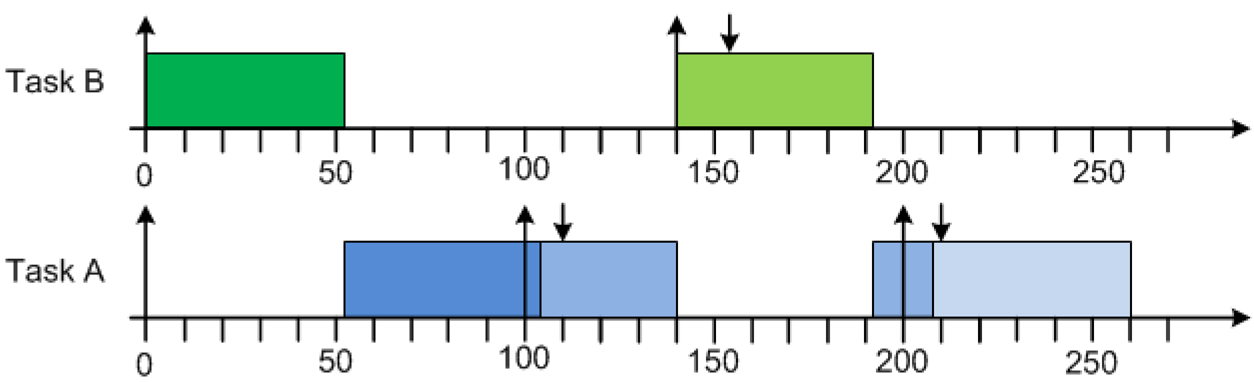


[J.Y.-T. Leung, J. Whitehead "On the complexity of fixed-priority scheduling of periodic real-time tasks, Performance Evaluation, 2(4): 237-250, 1982]

Deadline Monotonic: non-optimality

- Tasks with arbitrary deadlines

Task	Execution Time	Deadline	Period
A	52	110	100
B	52	154	140



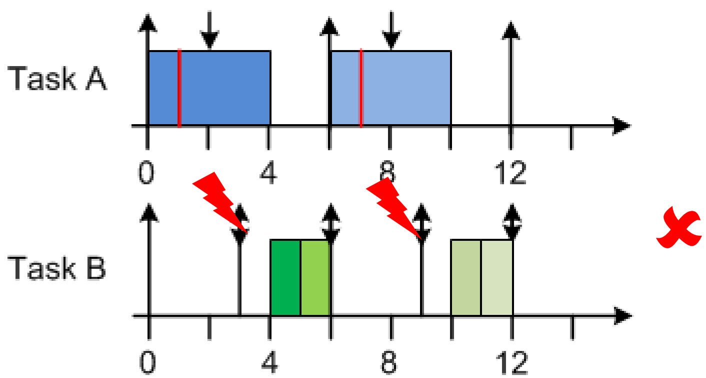
✓

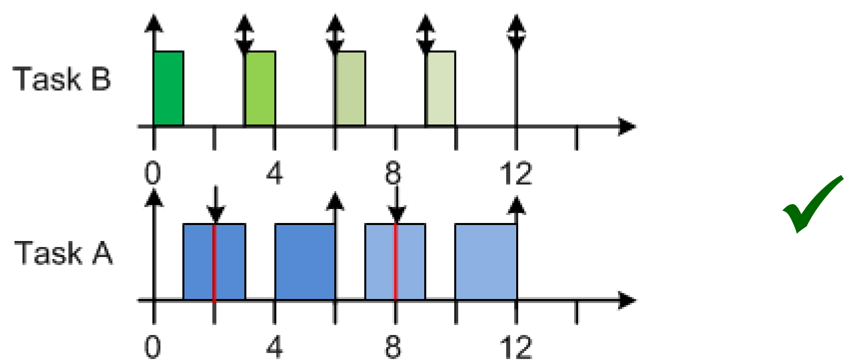
[Lehoczky J., "Fixed priority scheduling of periodic task sets with arbitrary deadlines". In proceedings Real-Time Systems Symposium, pages 201–209, 1990]

Deadline Monotonic: non-optimality

- Tasks with deadlines prior to completion

Task	Execution Time	Deadline	Period
A	1 + 3	2	6
B	1 + 0	3	3



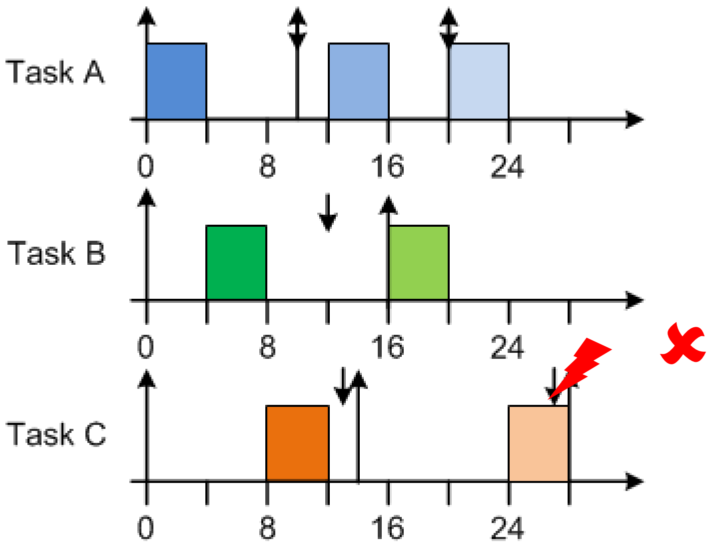


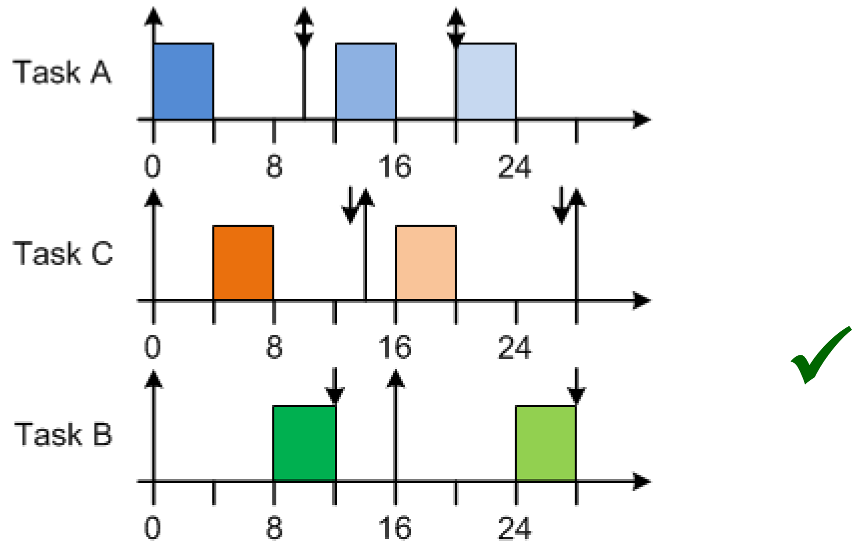
[A. Burns, K. Tindell, A.J. Wellings, "Fixed priority scheduling with deadlines prior to completion" In proceedings of the sixth Euromicro Workshop on Real-Time Systems. pp.138-142, 1994]

Deadline Monotonic: non-optimality

- Non-pre-emptive scheduling

Task	Execution Time	Deadline	Period
A	4	10	10
B	4	12	16
C	4	13	14



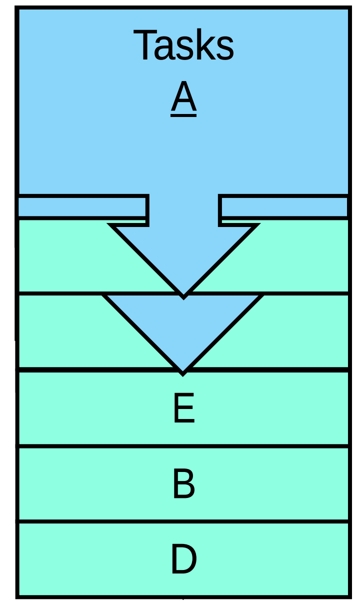


[L. George, N. Rivierre, M. Spuri, "Preemptive and Non-Preemptive Real-Time UniProcessor Scheduling", INRIA Research Report, No. 2966, September 1996]

Example derived from: [R.I. Davis and A. Burns "Robust priority assignment for messages on Controller Area Network (CAN)". Real-Time Systems, Volume 41, Issue 2, pages 152-180, February 2009]

Optimal Priority Assignment

```
for each priority level i, lowest first {
    for each unassigned task τ {
         if τ is schedulable at priority i
         assuming that all unassigned tasks are
         at higher priorities {
              assign task τ to priority level i
              break (exit for loop)
         }
    }
    if no tasks are schedulable at priority i {
         return unschedulable
    }
}
return schedulable
```



$n(n+1)/2$ schedulability tests rather than $n!$
by exploring all possible orderings
n = 25, that is 325 tests rather than 15511210043330985984000000

[N.C. Audsley, "Optimal priority assignment and feasibility of static priority tasks with arbitrary start times", Technical Report YCS 164, Dept. Computer Science, University of York, UK, 1991.]

[N.C. Audsley, “On priority assignment in fixed priority scheduling”, Information Processing Letters, 79(1): 39-44, May 2001.]

[K. Bletsas, and N.C. Audsley, “Optimal priority assignment in the presence of blocking”. *Information Processing Letters* Vol. 99, No. 3, pp83-86, August. 2006]

OPA algorithm appli

Powerful idea as we have said very little about the actual schedulability test hence broad applicability

- OPA algorithm provides optimal prio… schedulability test S for fixed priority sc… …ed that three conditions are met...
  - Condition 1: Schedulability of a task may, according to the test, be dependent on the set of higher priority tasks, but not on their relative priority ordering
  - Condition 2: Schedulability of a task may, according to the test, be dependent on the set of lower priority tasks, but not on their relative priority ordering
  - Condition 3: When the priorities of any two tasks of adjacent priority are swapped, the task being assigned the higher priority cannot become unschedulable according to the test, if it was previously deemed schedulable at the lower priority

Tests meeting these conditions referred to as **OPA-compatible**

[R.I. Davis, A. Burns "Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems". In proceedings Real-Time Systems Symposium pp 398-409, 2009.]

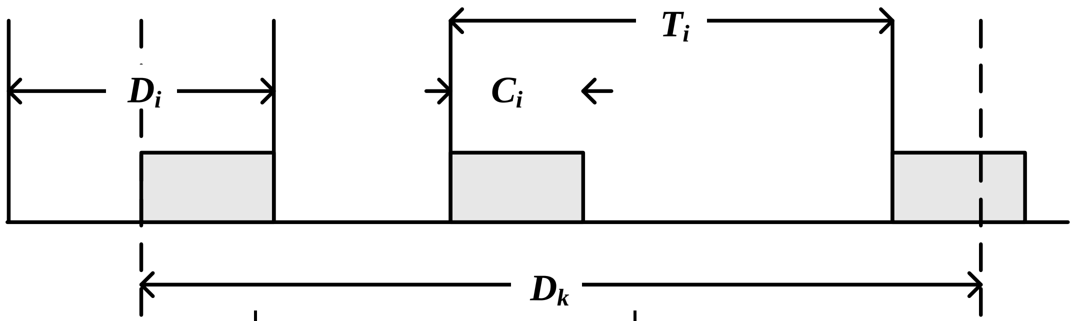
Multiprocessor: global FP scheduling

- Global FP scheduling
  - Single global run-queue fixed priority pre-emptive scheduling on multiple processsors
- Incompatible with OPA ✘
  - Any exact test (B. Andersson and Jonsson 2000) such as those for periodic tasksets given by Cucu and Goossens (2006, 2007).
  - Response time analysis (RTA test) of Bertogna and Cirinei (2007)
  - Improved RTA test of Guan et al. (2009)
- Compatible with OPA ✓
  - Deadline Analysis (DA test) of Bertogna et al. (2009)
  - Simple Response Time test of B. Andersson and Jonsson (2001)

[R.I. Davis and A. Burns "Improved Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems". Real-Time Systems, Vol. 47, No. 1, pp.1-40, 2011.]

Global FP schedulability tests #1

- Deadline Analysis “DA test” (Bertogna et al. 2009)



$$D_k \geq C_k + \left\lfloor \frac{1}{m} \sum_{\forall i \in hp(k)} I_i^D(D_k) \right\rfloor$$

Compatible with OPA ✓

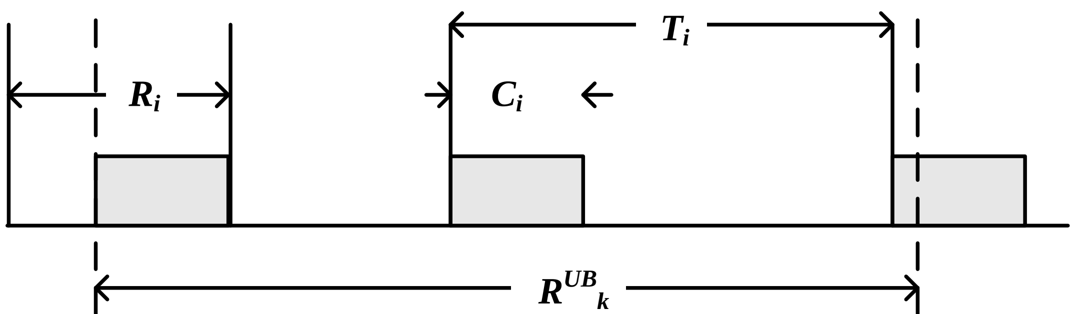
$$I_i^D(D_k) = \min(W_i^D(D_k), D_k - C_k + 1)$$

$$W_i^D(D_k) = N_i(D_k)C_i + \min(C_i, D_k + D_i - C_i - N_i(D_k)T_i)$$

$$N_i(D_k) = \left\lfloor \frac{D_k + D_i - C_i}{T_i} \right\rfloor$$

Global FP schedulability tests #2

- Response Time Analysis “RTA test” (Bertogna & Cirinei 2007)



$$R_k^{UB} \leftarrow C_k + \left\lfloor \frac{1}{m} \sum_{\forall i \in hp(k)} I_i(R_k^{UB}) \right\rfloor$$

Incompatible with OPA ✘

$$I_i(R_k^{UB}) = \min(W_i^R(R_k^{UB}), R_k^{UB} - C_k + 1)$$

$$W_i^R(L) = N_i^R(L)C_i + \min(C_i, L + R_i^{UB} - C_i - N_i^R(L)T_i)$$

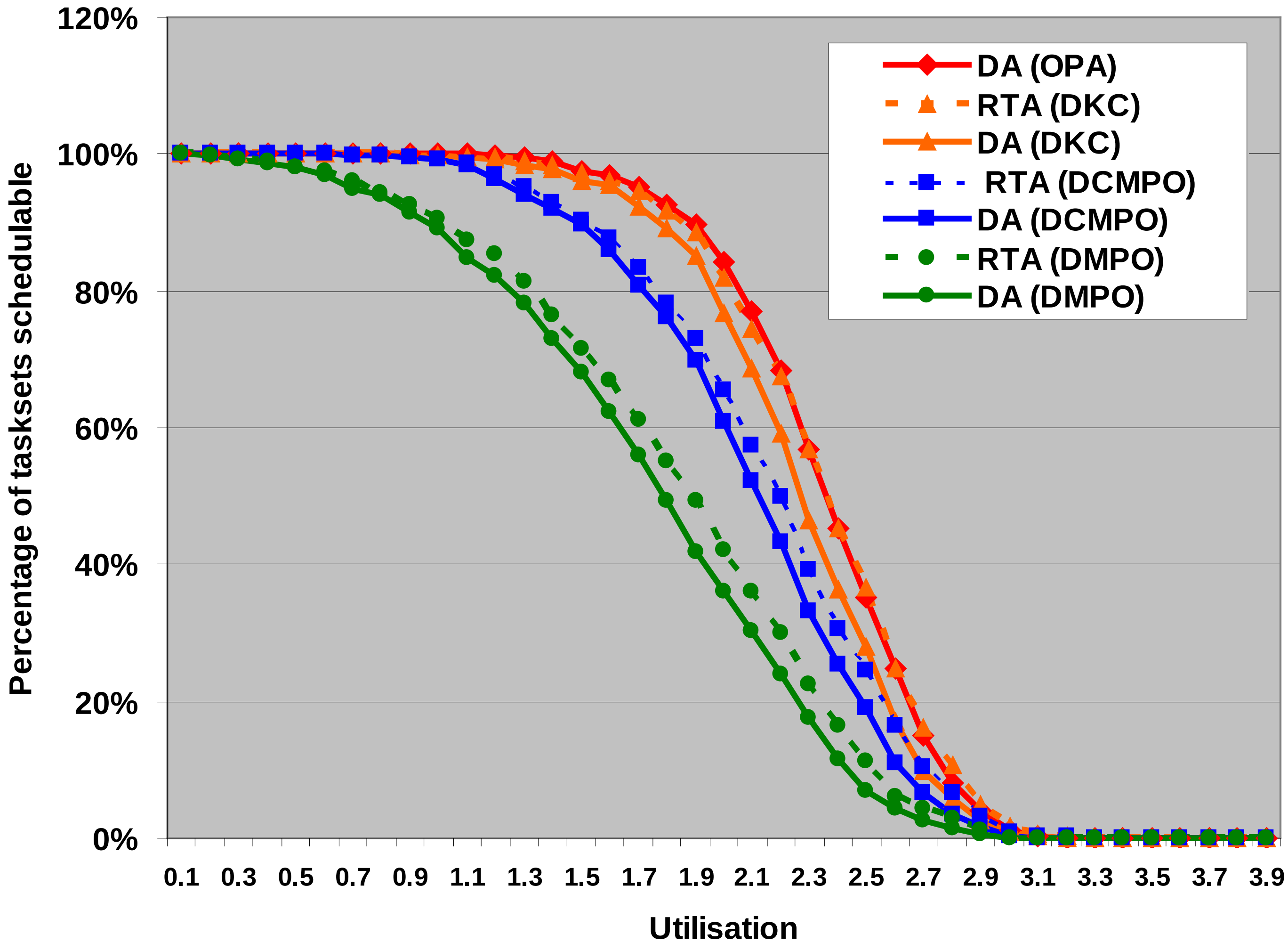
$$N_i^R(L) = \left\lfloor \frac{L + R_i^{UB} - C_i}{T_i} \right\rfloor$$

Multiprocessor: global FP scheduling

- RTA test dominates DA test
- Which is better?
  - RTA test + heuristic priority assignment
    - Deadline Monotonic
    - D – C Monotonic
    - DkC Monotonic (k is a factor that depends on the number of processors)
  - DA test + Optimal priority assignment

[R.I. Davis and A. Burns "Improved Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems". Real-Time Systems, Vol. 47, No. 1, pp.1-40, 2011.]

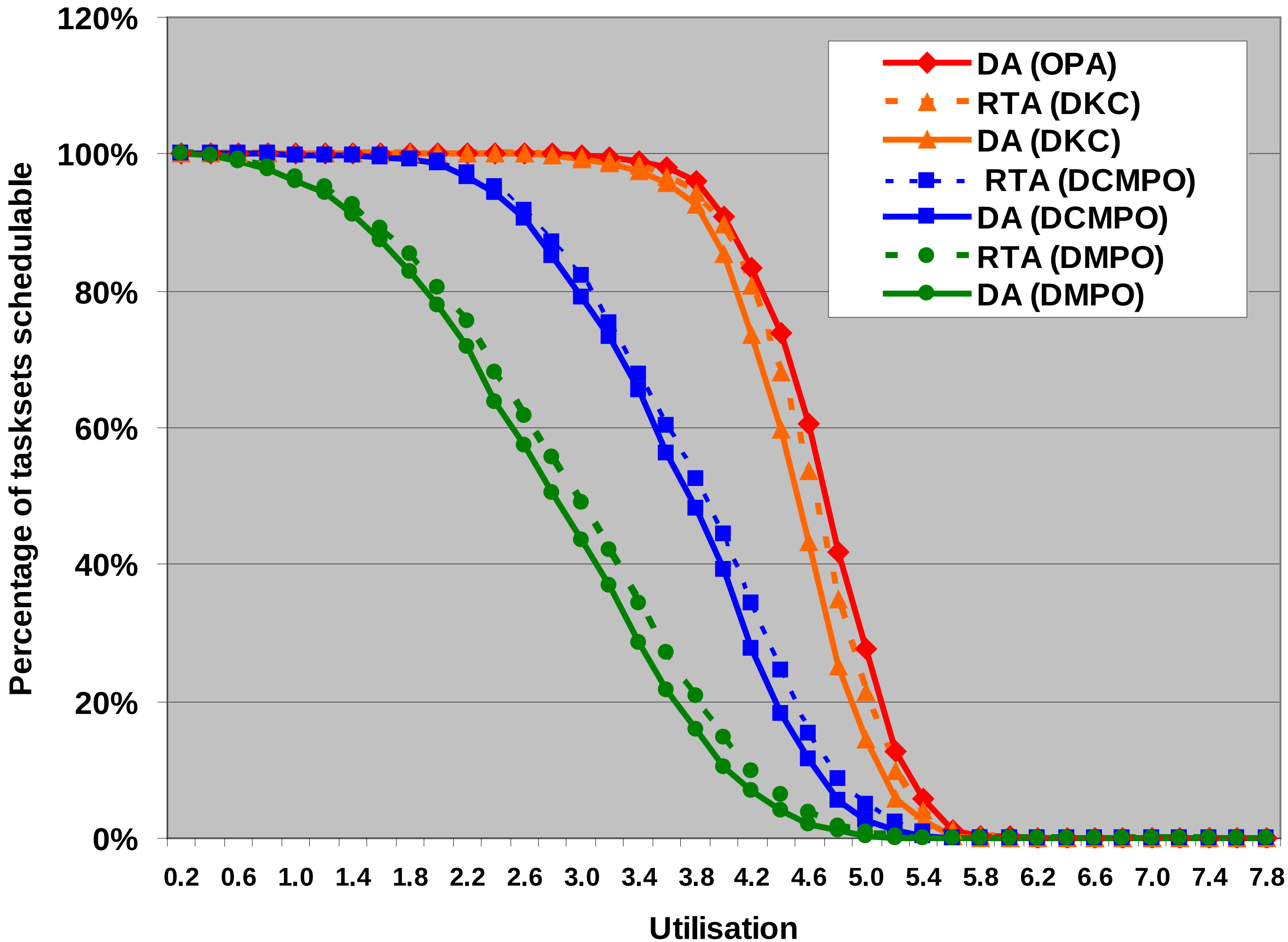
Global FP: Priority Assignment



4 Processors

20 tasks

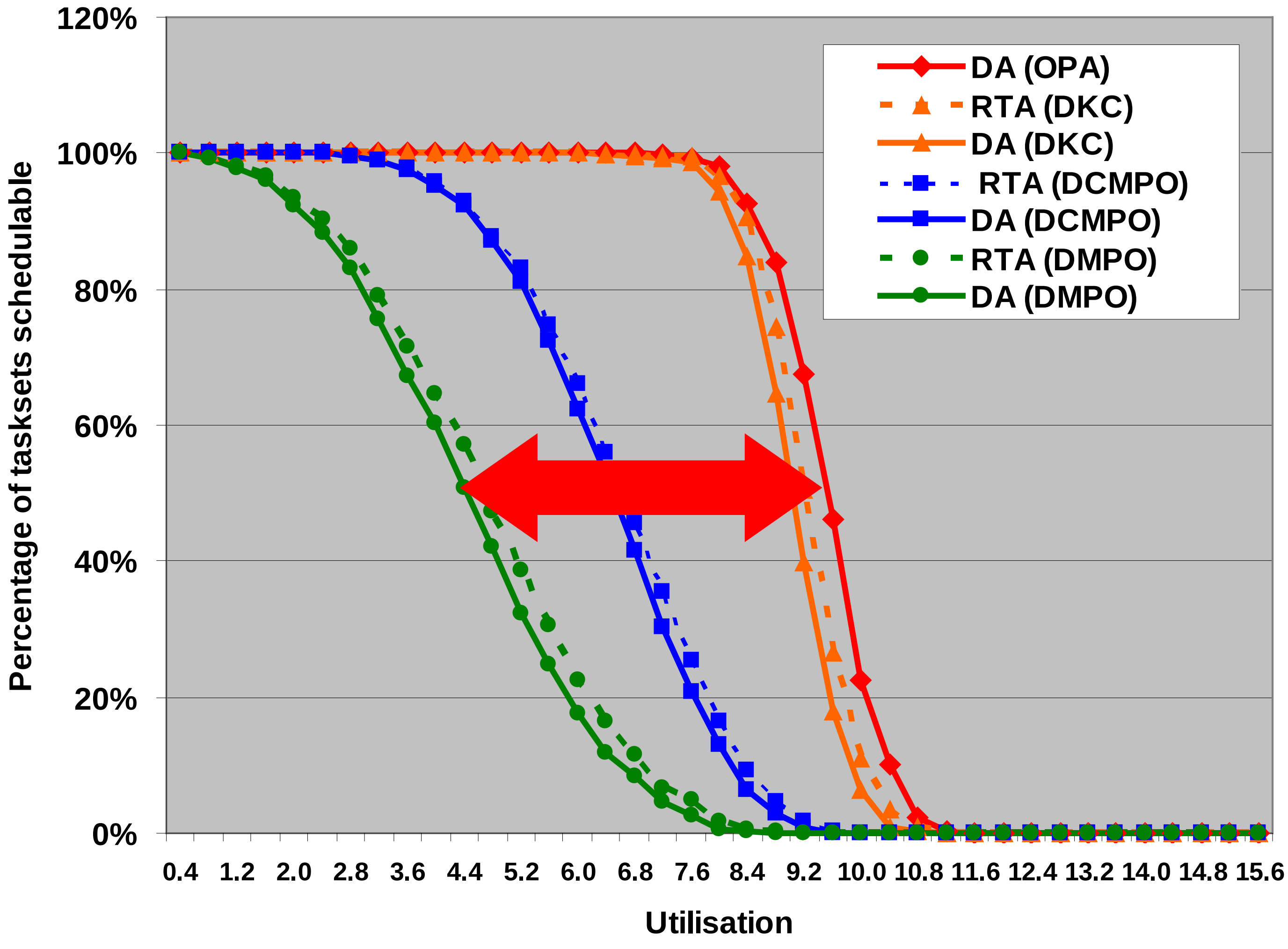
Global FP: Priority Assignment



8 Processors

40 tasks

Global FP: Priority Assignment

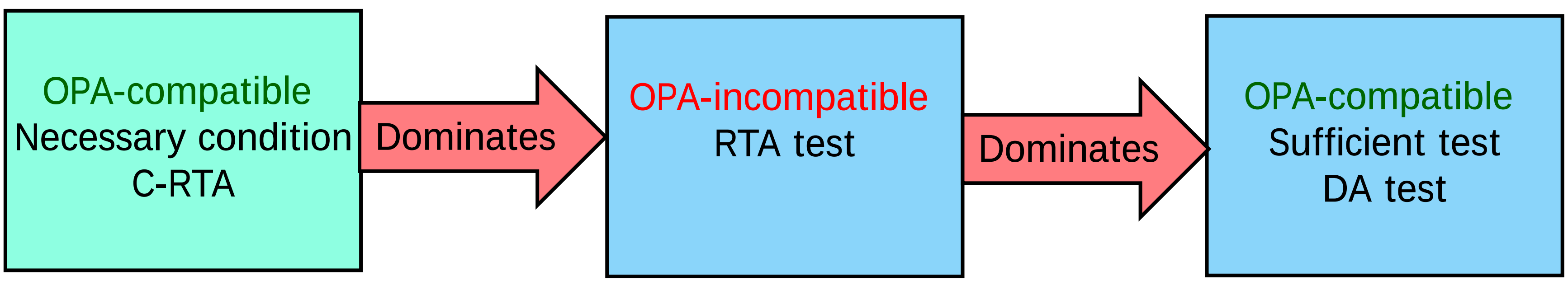


16 Processors

80 tasks

Beyond OPA

- What to do if the schedulability test is not OPA-compatible (e.g. RTA test for global FP scheduling)?
  - Search *n*! combinations?
- How to prune the search space?
  - Use dominance relationship between tests

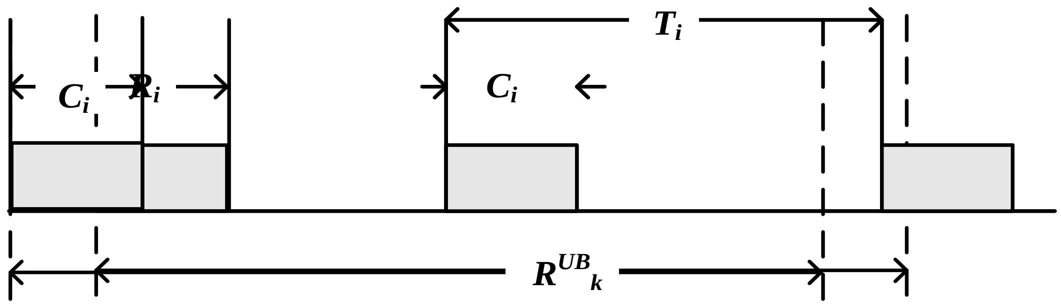


  - Use the sufficient test and the necessary condition to prune the choice of tasks at each priority level

[R.I. Davis and A. Burns, "On Optimal Priority Assignment for Response Time Analysis of Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Hard Real-Time Systems". University of York, Department of Computer Science Technical Report, YCS-2009-451, April 2010.]

C-RTA necessary test

- Based on Response Time Analysis “RTA test” (Bertogna & Cirinei 2007)



$$R_k^{UB} \leftarrow C_k + \left\lfloor \frac{1}{m} \sum_{\forall i \in hp(k)} I_i(R_k^{UB}) \right\rfloor$$

$$I_i(R_k^{UB}) = \min(W_i^R(R_k^{UB}), R_k^{UB} - C_k + 1)$$

$$W_i^R(L) = N_i^R(L) C_i + \min(C_i, L + R_i^{UB} - C_i - N_i^R(L) T_i)$$

$$N_i^R(L) = \left\lfloor \frac{L + R_i^{UB} - C_i}{T_i} \right\rfloor$$

Search with backtracking

Task Index

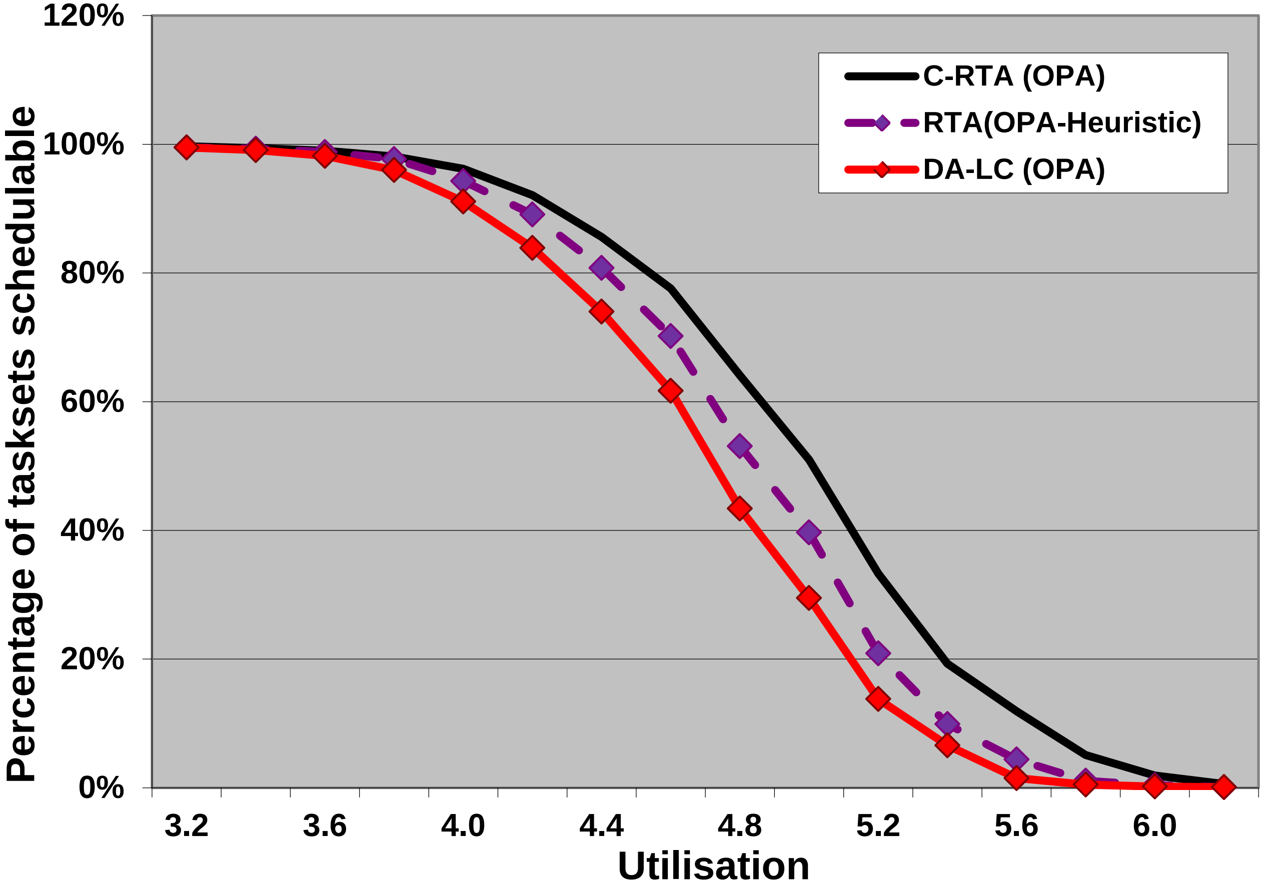
Priority Level		A	B	C	D	E
	1	✓	-	-	-	-
	2		-	-	✓	-
	3	×	?	-	?	-
	4	×	×	?	?	-
	5					✓

OPA-Compatible Sufficient test implies schedulable

OPA-Compatible Necessary test implies unschedulable

[R.I. Davis and A. Burns, "On Optimal Priority Assignment for Response Time Analysis of Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Hard Real-Time Systems". University of York, Department of Computer Science Technical Report, YCS-2009-451, April 2010.]

Global FP: Priority Assignment



8 Processors

40 tasks

[R.I. Davis and A. Burns, “On Optimal Priority Assignment for Response Time Analysis of Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Hard Real-Time Systems”. University of York, Department of Computer Science Technical Report, YCS-2009-451, April 2010.]

Minimising the number of Priority Levels with OPA

- Important for practical systems that may support only a limited number of priorities

```
for each priority level i, lowest first {
    Z = empty set
    for each unassigned task τ {
        if τ is schedulable at priority i  assuming that
        all unassigned tasks are  at higher priorities {
                add τ to Z
        }
    }
    if no tasks are schedulable at priority i {
        return unschedulable
    }
    else {
        assign all tasks in Z to priority i
    }
    if no unassigned tasks remain {
        break
    }
}
return schedulable
```

[N.C. Audsley, “On priority assignment in fixed priority scheduling”, Information Processing Letters, 79(1): 39-44, May 2001.]

Intermission

Robust Priority Assignment

- Drawback of OPA algorithm
  - Arbitrary choice of schedulable tasks at each priority
  - May leave the system only just schedulable – i.e fragile not robust to minor changes
- In practice tasks may be subject to additional interference
  - Execution time budget overruns; interrupts occurring in bursts or at ill-defined rates; ill-defined RTOS overheads; ill-defined critical sections; cycle stealing by peripheral devices (DMA) etc. etc.
- Want a **robust priority ordering**, able to tolerate the maximum amount of additional interference

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems”. In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Additional Interference

- Very general model of additional interference
- Additional Interference function $E(\alpha,w,i)$
  - α scaling factor – used to model variability
  - w time window – over which interference occurs
  - i priority level – at or below which the interference impinges on task response times
- Require that $E(\alpha,w,i)$ is a monotonic non-decreasing function of its parameters
  - In practice most sources of interference are
    - Greater in longer intervals of time than in shorter ones
    - Affect lower priorities if they also affect higher priorities
    - Guaranteed to be monotonic in α as this is the scaling factor

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems". In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Robust Priority Assignment

- Definition: **Robust Priority Assignment**

  (with an additional interference function $E(\alpha,w,i)$)

  For a given system model and additional interference function, a priority assignment policy *P* is referred to as **robust** if there are no systems, compliant with the system model, that are schedulable and can tolerate additional interference characterized by a scaling factor α using another priority assignment policy *Q* that are not also schedulable and can tolerate additional interference characterized by the same or larger scaling factor using priority assignment policy *P*.

  Of all feasible priority assignments, the robust priority assignment tolerates the most additional interference (largest α)

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems". In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Robust Priority Assignment (RPA) algorithm

- Based on OPA algorithm
- Same three conditions needed for compatibility

Condition 1: Schedulability of a task may, according to the test, be dependent on the set of higher priority tasks, but not on their relative priority ordering

Condition 2: Schedulability of a task may, according to the test, be dependent on the set of lower priority tasks, but not on their relative priority ordering

Condition 3: When the priorities of any two tasks of adjacent priority are swapped, the task being assigned the higher priority cannot become unschedulable according to the test, if it was previously deemed schedulable at the lower priority

As additional interference $E(\alpha, w, i)$ is monotonically non-decreasing in its parameters, the above conditions also hold when additional interference is considered

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems". In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

RPA Algorithm

```
for each priority level i, lowest first
{
    for each unassigned task τ
    {
         determine the largest value of α for which task τ
         is schedulable at priority i assuming that all
         unassigned tasks have higher priorities
    }
    if no tasks are schedulable at priority i
    {
         return unschedulable
    }
    else
    {
         assign the schedulable task that tolerates the
         max α at priority i to priority i
    }
}
return schedulable
```

Robust Priority Assignment

- Example 1: **Non-pre-emptive scheduling**
  - Additional interference from single invocation of an interrupt handler with unknown execution time
  - Additional interference $E(\alpha, w, i) = \alpha$

Task	C	D	T
τ_A	125	450	450
τ_B	125	550	550
τ_C	65	600	600
τ_D	125	1000	1000
τ_E	125	2000	2000

Robust Priority Assignment

- Computed values of α

Priority	Task				
	τ_A	τ_B	τ_C	τ_D	τ_E
5	NS	NS	NS	120	**354**
4	NS	NS	NS	**120**	-
3	10	**110**	74	-	-
2	135	-	**199**	-	-
1	**200**	-	-	-	-

- Robust priority ordering
  - Tolerates additional interference of up to **110** time units
- Deadline monotonic: neither optimal nor robust
  - Tolerates additional interference of up to **74** time units
- OPA: may be worse still
  - Might tolerate additional interference of only **10** time units

Robust Priority Assignment

- Example 2: **Pre-emptive scheduling, D >T**

Task	C	D	T
τ_A	42	118	100
τ_B	52	154	140

- Schedulable with priority orderings (τ_A, τ_B) and (τ_B, τ_A) with no additional interference

Robust Priority Assignment

- Case 1: $E(\alpha, w, i) = \alpha \left\lceil \frac{w}{100} \right\rceil$
  - (τ_A, τ_B) tolerates α = (58, **9**)
  - (τ_B, τ_A) tolerates α = (51, **10**) **Robust ordering**
- Case 2: $E(\alpha, w, i) = \alpha \left\lceil \frac{w}{200} \right\rceil$
  - (τ_A, τ_B) tolerates α = (76, **18**) **Robust ordering**
  - (τ_B, τ_A) tolerates α = (96, **15**)
- Case 3: $E(\alpha, w, i) = \alpha \left(\left\lceil \frac{w}{100} \right\rceil K + \left\lceil \frac{w}{200} \right\rceil L \right)$
  - Robust ordering depends on specific values of K and L
  - K=1, L=0: equivalent to Case 1: (τ_B, τ_A) is the **Robust ordering**
  - K=0, L=1: equivalent to Case 2: (τ_A, τ_B) is the **Robust ordering**

Robust Priority Assignment

- Result #1 (somewhat negative)

  In general, a Robust priority ordering can only be found if the form of the additional interference function is well defined (only α unknown).

  Often it can be well defined – e.g. robust to maximum amount of additional interference at the highest priority level, maximum number of transmission faults etc.

  But more to follow on specific system models...

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems". In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Robust Priority Assignment

- Mixed systems: two subsets of tasks
  - “DM tasks”
    - Satisfy the restrictions where Deadline Monotonic priority ordering is known to be optimal
    - Pre-emptable, D≤T, resource access according to SRP, no transactions or offsets
  - “Non DM tasks”
    - Don’t satisfy the restrictions where Deadline Monotonic priority ordering is known to be optimal
    - Pre-emptable with D>T, non-pre-emptable, co-operative scheduling with non-pre-emtable final sections, transactions, non-zero offset

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems”. In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Robust Priority Assignment

- Result #2

  For systems containing only DM tasks, Deadline Monotonic priority ordering is optimal and also **robust**, irrespective of task execution times and irrespective of the form of the additional interference $E(\alpha,w,i)$ provided only that the additional interference is monotonic in its parameters.

Result #3

For mixed systems containing both DM and non DM tasks, then there exists a **robust** priority order with the DM tasks in Deadline Monotonic partial order*

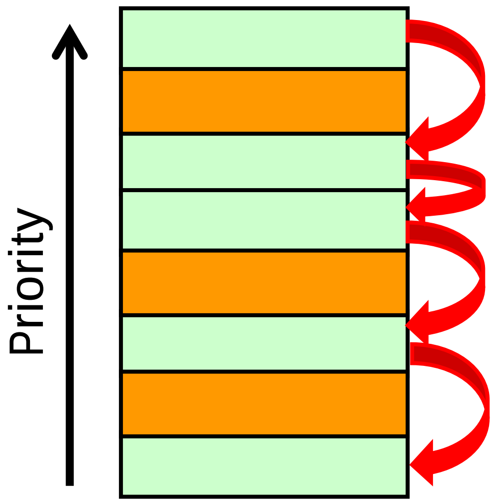
*This holds provided that the interference from non DM tasks is monotonically non-decreasing w.r.t. time intervals and priority levels, and not dependent on specific tasks

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems". In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Robust Priority Assignment

DM task
(e.g. constrained deadline)

Non DM task
(e.g. arbitrary deadline, part of a transaction etc.)



Deadline
Monotonic
Partial order

Robust Priority Assignment

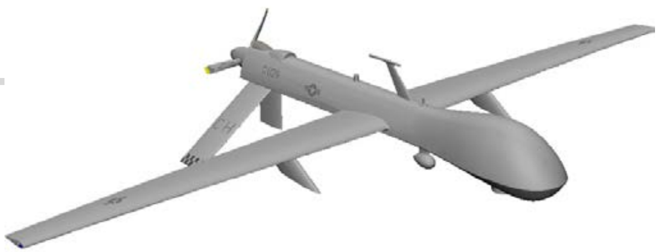
- Can improve efficiency of OPA and RPA algorithms
  - Of all the DM tasks, the one with the largest deadline is the one that can tolerate the most additional interference at a given priority level
  - Only one DM task need be checked at each priority level – the one with the largest deadline of all unassigned DM tasks
  - For n tasks, k of which are DM tasks:

    $(n(n+1)-k(k-1))/2$ task schedulability tests instead of $n(n+1)/2$

  - Example: 4 tasks in a transaction, 46 independent tasks
    max. of 240 schedulability tests instead of 1275

[R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems". In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.]

Mixed Criticality



- Examples
  - Aerospace: e.g. UAVs
  - Automotive: ASILs e.g. cruise control v. electronic steering assistance
- Task Model
  - Tasks have different criticality levels (e.g. HI and LO)
  - HI criticality tasks have different execution time bounds for the two criticality levels: C_i^{HI} and C_i^{LO}
  - When a HI task exceeds its LO criticality execution budget, then the system enters HI criticality mode
    - In HI criticality mode, all HI criticality tasks must meet their deadlines assuming HI criticality execution times, LO criticality tasks may be abandoned
    - In LO criticality mode, all tasks must meet their deadlines assuming LO criticality execution times

[S.K. Baruah, A. Burns, R.I. Davis "Response Time Analysis for Mixed Criticality Systems" . In proceedings 32nd IEEE Real-Time Systems Symposium (RTSS'11) , pages 34-43, Nov 29th - Dec 2nd, 2011]

Mixed Criticality FP Scheduling

- AMC-rtb
  - LO criticality mode: **High criticality tasks**

$$R_i = C_i(LO) + E(\alpha, w, i) + E(\alpha, w, i)$$

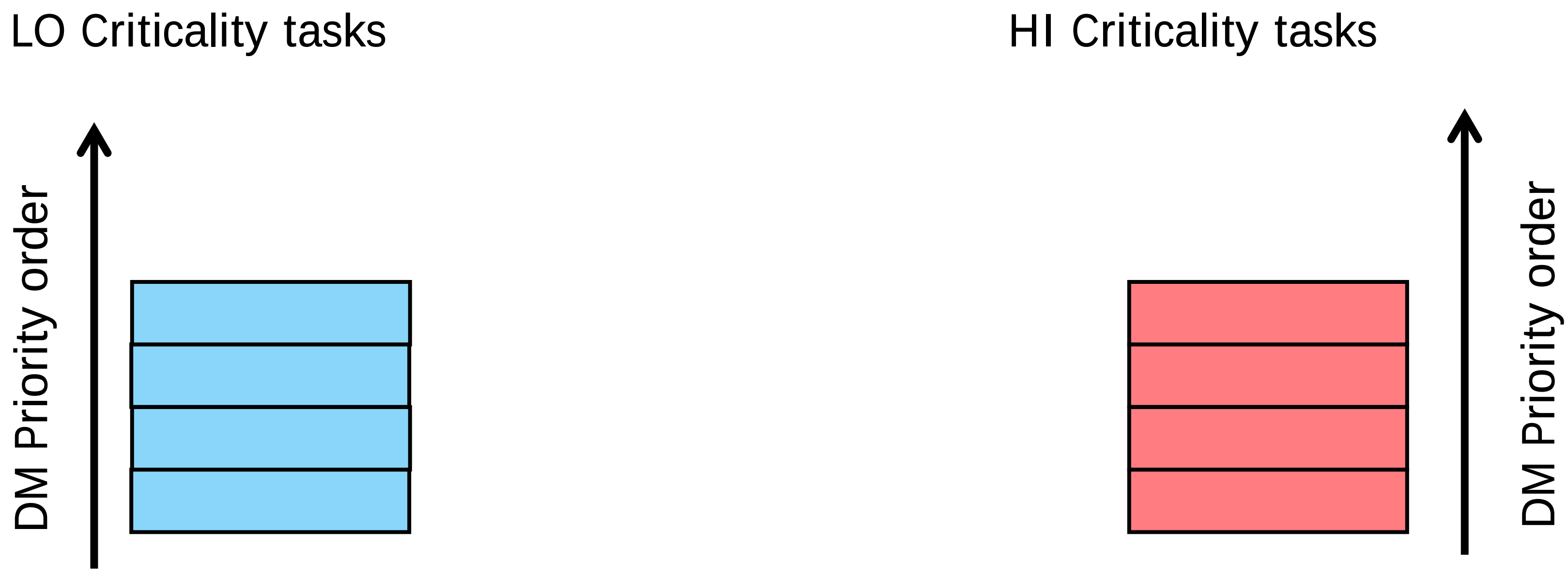
$$R_i^{LO} = R_i$$

  - HI criticality mode:

$$R_i = C_i(HI) + \sum_{\forall j \in hpH(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j(HI) + E(\alpha, w, i)$$

[S.K. Baruah, A. Burns, R.I. Davis "Response Time Analysis for Mixed Criticality Systems" . In proceedings 32nd IEEE Real-Time Systems Symposium (RTSS'11) , pages 34-43, Nov 29th - Dec 2nd, 2011]

Mixed Criticality FP Scheduling



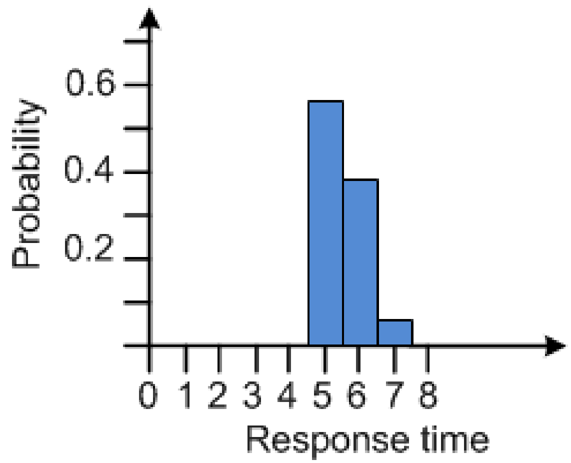
2n-1 schedulability tests rather than $n(n+1)/2$

[S.K. Baruah, A. Burns, R.I. Davis "Response Time Analysis for Mixed Criticality Systems" . In proceedings 32nd IEEE Real-Time Systems Symposium (RTSS'11) , pages 34-43, Nov 29th - Dec 2nd, 2011]

Priority assignment in probabilistic real-time systems

- Tasks with execution times modelled as independent random variables

Task	Execution Time	Deadline	Period	DMR threshold
A	$\begin{pmatrix} 2 & 3 \\ 0.7 & 0.3 \end{pmatrix}$	5	10	0.5
B	$\begin{pmatrix} 3 & 4 \\ 0.8 & 0.2 \end{pmatrix}$	6	10	0.05



- Deadline monotonic priority ordering not optimal
  - Task A at higher priority P($R_A > D_A$)=0 ✓ P($R_B > D_B$)=0.06 ✗
  - Task B at higher priority P($R_B > D_B$)=0 ✓ P($R_A > D_A$)=0.44 ✓

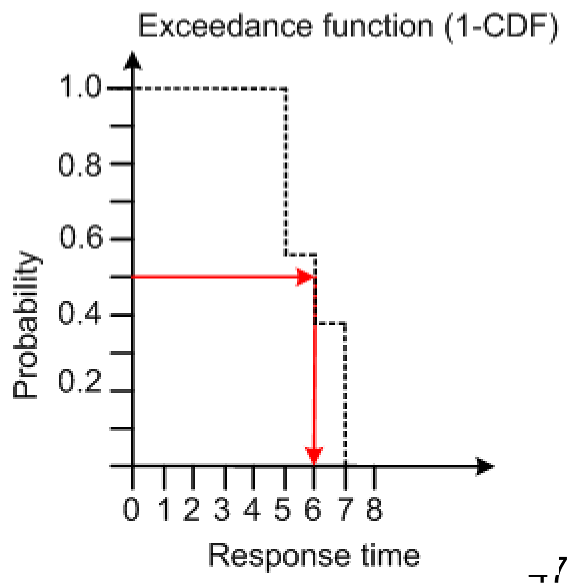
[D. Maxim, O. Buffet, L. Santinelli, L. Cucu-Grosjean, R. I. Davis “Optimal Priority Assignment Algorithms for Probabilistic Real-Time Systems” . In proceedings 19th International Conference on Real-Time and Network Systems (RTNS'11) , Sept 29-30th, 2011.]

Optimal Priority Assignment for probabilistic systems

- Same three conditions needed for OPA compatibility

Condition 1: Schedulability of a task may, according to the test, be dependent on the set of higher priority tasks, but not on their relative priority ordering
Condition 2: Schedulability of a task may, according to the test, be dependent on the set of lower priority tasks, but not on their relative priority ordering
Condition 3: When the priorities of any two tasks of adjacent priority are swapped, the task being assigned the higher priority cannot become unschedulable according to the test, if it was previously deemed schedulable at the lower priority

Definition of "schedulable" very different – based on probability of deadline failure (i.e. response time distribution and its exceedance function) compared to Dead Miss Ratio threshold



RTSS 2012

Please come along to my talk at
11:30am on 5th Dec 2012
RTSS 2012
San Juan, Puerto Rico

Optimal Fixed Priority Scheduling with
Deferred Pre-emption

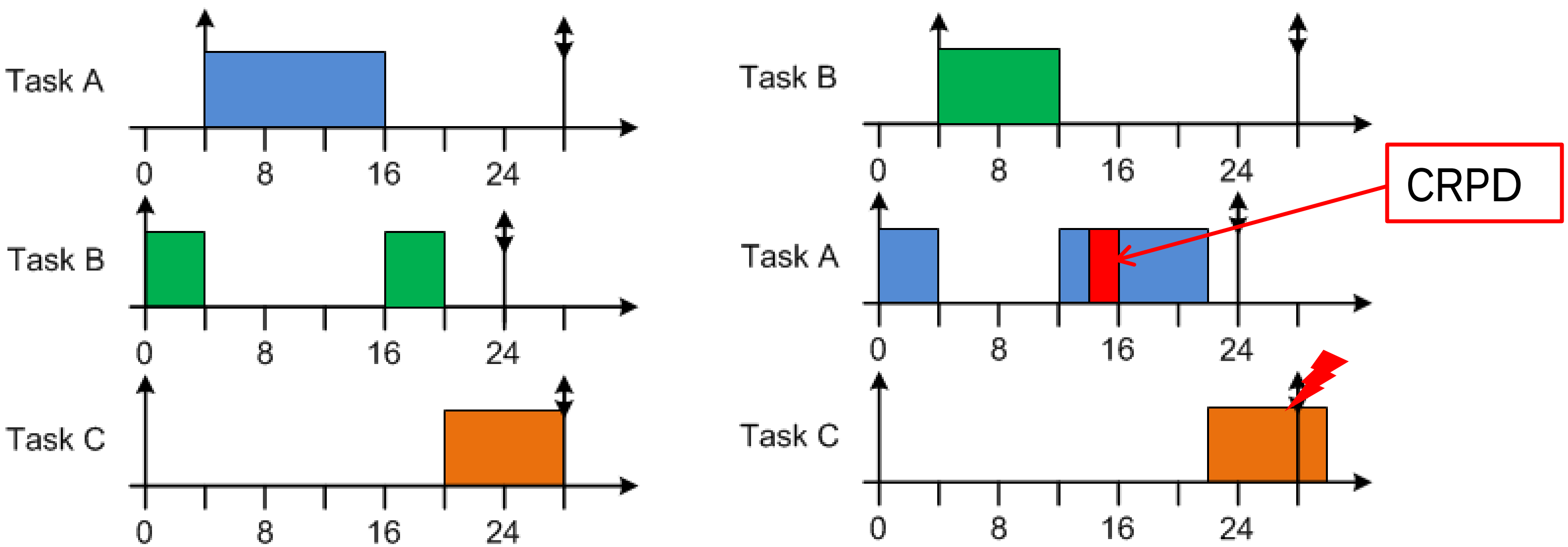
Rob Davis and Marko Bertogna

Interesting problems not obviously amenable to OPA

- FPDS: Minimising the number of pre-emptions through maximising blocking (Bertogna et al 2011)
  - Can be done from highest priority down rather than lowest priority up, but then requires a pre-defined priority ordering
- Probabilistic:
  - Minimising average/total probability of deadline failure across all tasks (Maxim et al 2011)
    - Swapping tasks at adjacent priorities may decrease the total, even if the larger of the two probabilities of deadline failure decreases
- NoC wormhole communication: Assigning priorities to network flows (Shi and Burns, 2008)
  - Response time of a network flow depends on the response times of higher priority flows
- Pre-emption thresholds: Assignment of base priorities and pre-emption thresholds (Wang and Saksena, 1999)
  - Pre-emption threshold assignment depends on the relative priority ordering of higher priority tasks

Interesting problems not obviously amenable to OPA

- Cache Related Pre-emption Delays (CRPD)
  - Response times depend upon the relative priority ordering of higher priority tasks



[S. Altmeyer, R.I. Davis, C. Maiza “Improved cache related pre-emption delay aware response time analysis for fixed priority pre-emptive systems” . Real-Time Systems, Volume 48, Issue 5, Pages 499-526, Sept 2012.]
[S. Altmeyer, R.I. Davis, C. Maiza “Cache related pre-emption delay aware response time analysis for fixed priority pre-emptive systems” . In proceedings 32nd IEEE Real-Time Systems Symposium (RTSS'11), pages 261-271, Nov 29th - Dec 2nd, 2011]

Questions?

References

S. Altmeyer, R.I. Davis, C. Maiza “Improved cache related pre-emption delay aware response time analysis for fixed priority pre-emptive systems” . Real-Time Systems, Volume 48, Issue 5, Pages 499-526, Sept 2012.

S. Altmeyer, R.I. Davis, C. Maiza “Cache related pre-emption delay aware response time analysis for fixed priority pre-emptive systems” . In proceedings 32nd IEEE Real-Time Systems Symposium (RTSS'11), pages 261-271, Nov 29th - Dec 2nd, 2011.

N.C. Audsley, "Optimal priority assignment and feasibility of static priority tasks with arbitrary start times", Technical Report YCS 164, Dept. Computer Science, University of York, UK, 1991.

N.C. Audsley, “On priority assignment in fixed priority scheduling”, Information Processing Letters, 79(1): 39-44, May 2001.

S.K. Baruah, A. Burns, R.I. Davis “Response Time Analysis for Mixed Criticality Systems” . In proceedings 32nd IEEE Real-Time Systems Symposium (RTSS'11) , pages 34-43, Nov 29th - Dec 2nd, 2011.

M. Bertogna, G. Buttazzo, G. Yao. "Improving Feasibility of Fixed Priority Tasks using Non-Preemptive Regions", In Proceedings Real-Time Systems Symposium, Vienna, Austria, December 2011.

K Bletsas, N Audsley, “Optimal priority assignment in the presence of blocking” Information processing letters 99 (3), 83-86, 2006.

A. Burns, K. Tindell, A.J. Wellings, "Fixed priority scheduling with deadlines prior to completion" In proceedings of the sixth Euromicro Workshop on Real-Time Systems. pp.138-142, 1994.

A. Burns, “Dual Priority Scheduling: Is the Processor Utilisation bound 100%” In proceedings RTSOPS, 2010.

References

R.I. Davis and A. Burns, “Optimal Priority Assignment for Aperiodic Tasks with Firm Deadlines in Fixed Priority Pre-emptive Systems”. *Information Processing Letters* 53(5). 10th March 1995.

R.I. Davis, A. Burns. "Robust Priority Assignment for Fixed Priority Real-Time Systems”. In proceedings IEEE Real-Time Systems Symposium pp. 3-14. Tucson, Arizona, USA. December 2007.

R.I. Davis, A. Burns "Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems”. In proceedings Real-Time Systems Symposium pp 398-409, 2009.

R.I. Davis and A. Burns "Robust priority assignment for messages on Controller Area Network (CAN)”. Real-Time Systems, Volume 41, Issue 2, pages 152-180, February 2009.

R.I. Davis and A. Burns, “On Optimal Priority Assignment for Response Time Analysis of Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Hard Real-Time Systems”. University of York, Department of Computer Science Technical Report, YCS-2009-451, April 2010.

R.I. Davis and A. Burns "Improved Priority Assignment for Global Fixed Priority Pre-emptive Scheduling in Multiprocessor Real-Time Systems”. Real-Time Systems, Vol. 47, No. 1, pp.1-40, 2011.

R.I. Davis, S. Kollmann, V. Pollex, F. Slomka, "Schedulability Analysis for Controller Area Network (CAN) with FIFO Queues Priority Queues and Gateways”.
Real-Time Systems, 2012.

R.I. Davis, M. Bertogna "Optimal Fixed Priority Scheduling with Deferred Pre-emption”. In proceedings 33rd IEEE Real-Time Systems Symposium (RTSS'12) , Dec 4th - 7th, 2012.

References

M.S. Fineberg and O. Serlin, "Multiprogramming for hybrid computation", In proceedings AFIPS Fall Joint Computing Conference, pp 1-13, 1967

George, L., Rivierre, N., Spuri, M., "Preemptive and Non-Preemptive Real-Time UniProcessor Scheduling", INRIA Research Report, No. 2966, September 1996.

Lehoczky J., "Fixed priority scheduling of periodic task sets with arbitrary deadlines". In Proceedings Real-Time Systems Symposium, pp 201–209, 1990.

J.Y.-T. Leung, J. Whitehead "On the complexity of fixed-priority scheduling of periodic real-time tasks, Performance Evaluation, 2(4): 237-250, 1982.

Liu C.L., Layland J.W., "Scheduling algorithms for multiprogramming in a hard-real-time environment", Journal of the ACM, 20(1) pp 46-61, 1973.

D. Maxim, O. Buffet, L. Santinelli, L. Cucu-Grosjean, R. I. Davis "Optimal Priority Assignment Algorithms for Probabilistic Real-Time Systems" . In proceedings 19th International Conference on Real-Time and Network Systems (RTNS'11) , Sept 29-30th, 2011.

S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In Proceedings of the Real-Time Systems Symposium, pp. 239–243, 2007.

Y. Wang and M. Saksena. Scheduling fixed-priority tasks with pre-emption threshold. In Proceedings RTCSA'99, Hong Kong, China, December 13-15, 1999.

S. Zheng, A. Burns, "Priority Assignment for Real-Time Wormhole Communication in On-Chip Networks". In Proceedings Real-Time Systems Symposium, pp. 421-430, 2008.

A. Zuhily and A. Burns, "Optimality of (D-J)-monotonic priority assignment". Information Processing Letters, Vol. 103 No. 6, 2007.

References

A. Zuhily, A. Burns: Exact scheduling analysis of non-accumulatively monotonic multiframe tasks. Real-Time Systems 43(2): 119-146 (2009)