Schedulability analysis of distributed real-time systems

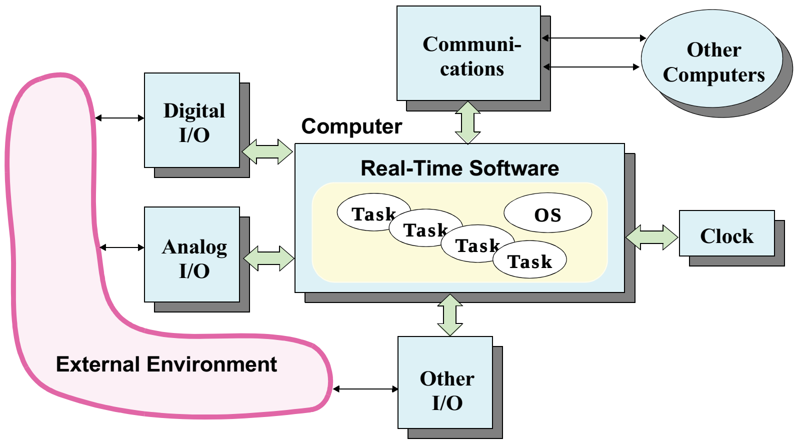
By: Michael González Harbour mgh@unican.es
http://www.ctr.unican.es
Grupo de Computadores y tiempo real, Universidad de Cantabria

Santander, February 2009

Schedulability analysis of real-time distributed systems

1. Introduction
2. Single processor systems
3. Distributed systems
4. Schedulability analysis: holistic approach
5. Schedulability analysis: offset-based approaches
6. Schedulability analysis: EDF
7. Modelling techniques: MAST
8. Conclusion and future work

Elements of a real-time system



Real-time systems

A Real-time system is a combination of a computer, hardware I/O devices, and special-purpose software, in which:

- there is a strong interaction with the environment
- the environment changes with time
- the system simultaneously controls and/or reacts to different aspects of the environment

As a result:

- timing requirements are imposed on software
- software is naturally concurrent

To ensure that timing requirements are met, the system timing behavior must be *predictable*

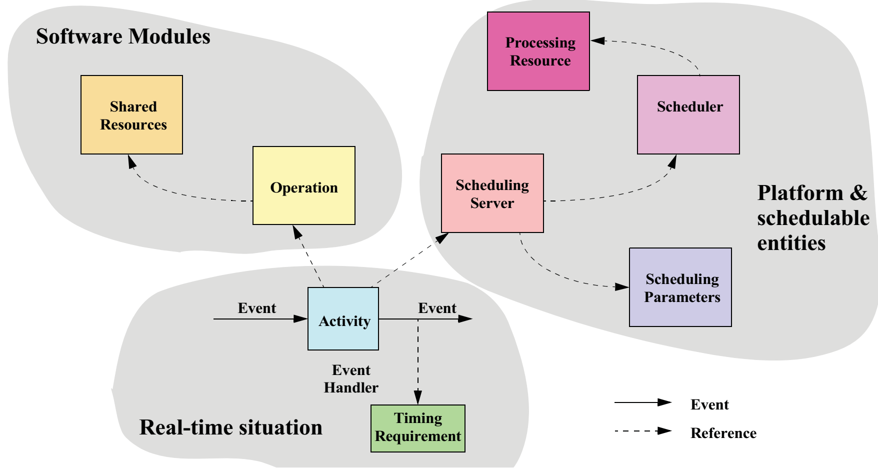
Real-time system model

The real-time system model contains five independent parts:

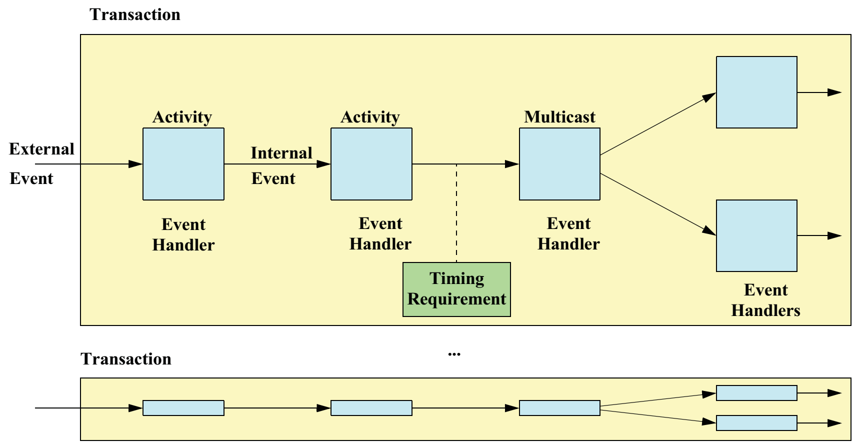
- platform and schedulable entities
  - *processing resources* and *schedulers*
  - threads and message streams (*scheduling servers*)
- Software modules
  - *operations* and messages
  - *shared resources*
- Real-time situation
  - representing a particular mode of operation of the system, composed of a set of *transactions*

A transaction contains a set of *activities* that will be executed by the system in response to *events*

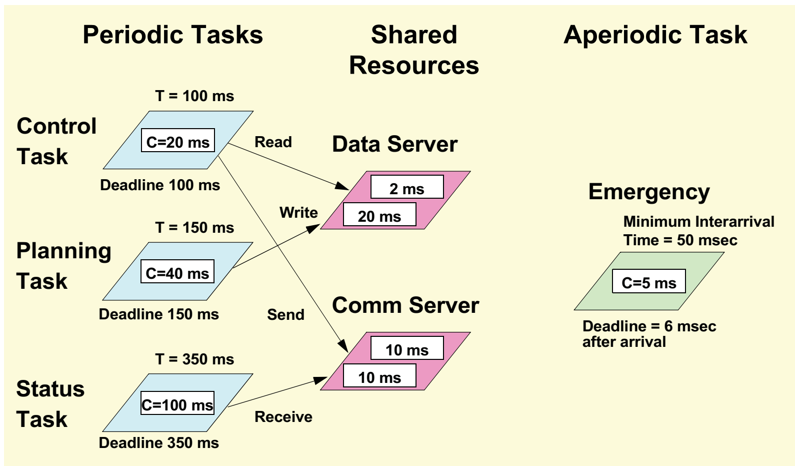
Real-time system model



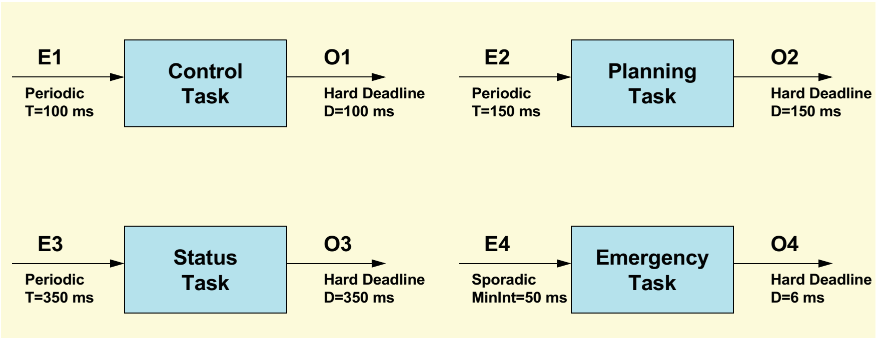
Real-time situation



A single-processor example



Transactions in this example



Schedulability analysis of real-time distributed systems

Definitions

Task parameters
- C_i = compute time (execution time) for task τ_i
- T_i = period (or minimum interarrival time) of task τ_i
- P_i = priority of task τ_i
- D_i = deadline of task τ_i
- R_i = worst-case response time of task τ_i
- ϕ_i = phase of task $\tau_{i,}$ (usually unknown)

Task utilization: $U_i = C_i / T_i$

CPU utilization for a set of tasks: $U = U_1 + U_2 + \ldots + U_n$

Basic principles of real-time analysis

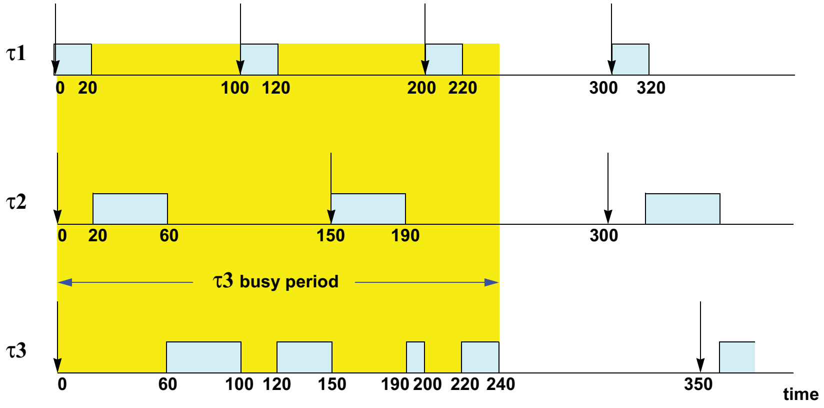
Two concepts help build the worst-case condition under fixed priorities with $D_i \le T_i$:

- *Critical instant*. The worst-case response time for all tasks in the task set is obtained when all tasks are activated at the same time
- *Busy period* for task τ_i. The interval during which the processor is busy executing τ_i or higher priority tasks
  - we only need to check the deadlines in the worst-case busy period
  - worst case busy period is initiated at a critical instant

Based on these concepts, several results arise:

- Optimality of deadline monotonic priorities when $D_i \le T_i$
- Utilization bound tests
- Response time analysis (exact test)

Example of a critical instant



Priority assignment

If $D_i \le T_i$, *deadline monotonic* assignment

- For a set of periodic independent tasks, with deadlines within the period, the optimum priority assignment is the deadline monotonic assignment
  - Priorities are assigned according to task deadlines
  - A task with a shorter deadline is assigned a higher priority

If $D_i > T_i$ for one or more tasks, Audsley's algorithm

- iteratively apply analysis, successively ordering tasks by priority (simulated annealing)
  - $O(n^2)$ times the analysis

Response time analysis (Harter, 1984; Joseph and Pandya, 1986)

Iterative test (pseudopolynomial time):

$$a_{k+1} = W_i(a_k) = \left\lceil \frac{a_k}{T_1} \right\rceil C_1 + \ldots + \left\lceil \frac{a_k}{T_{i-1}} \right\rceil C_{i-1} + C_i + B_i$$

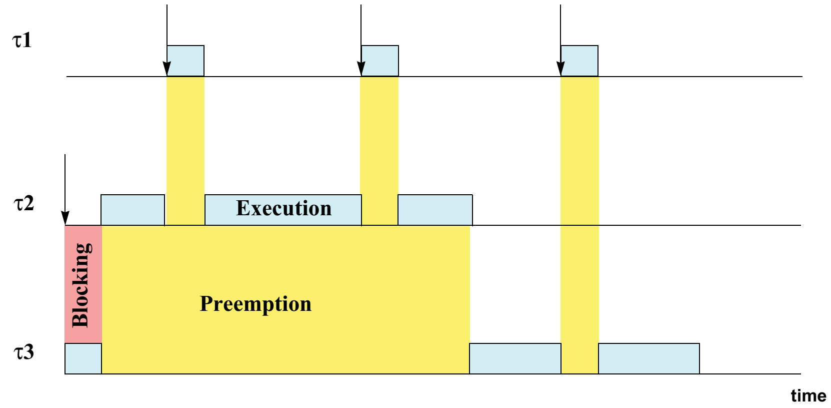
preemption — execution — blocking

- Start with any initial value < final value, for example

$$a_0 = C_1 + C_2 + \ldots + C_i$$

- Finish when two consecutive results are the same

Elements that influence the response time



Analysis with arbitrary deadlines (Lehoczky, 1990)

Iterative equation used for analysis of task-*i* in one resource:

$$w_i^{n+1}(p) = pC_i + B_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{w_i^n(p)}{T_j} \right\rceil C_j$$

This is carried out for p=1,2,3,..., until

$$w_i(p) \le pT_i$$

Then, the global worst-case response time is

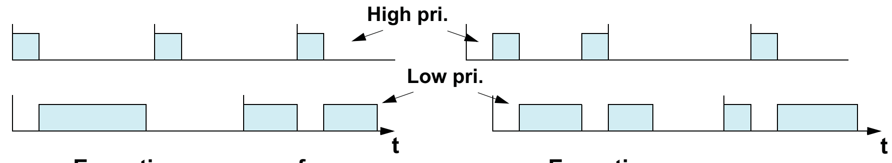
$$R_i = max(R_i(p)) \qquad R_i(p) = w_i(p) - (p-1)T_i + J_i$$

Effects of release jitter

Periodic events with jitter have an arrival time which may be early or late, within a bounded interval:

- events arrive at $t_0 + nT_i + j, \quad j \in [0, J_i]$

Jitter may have a delay effect on lower priority tasks:



Execution sequence for two periodic tasks. Worst case is one preemption

Execution sequence with jitter. Worst case is two preemptions

Analysis with arbitrary deadlines and jitter (Tindell, 1992):

Iterative equation used for analysis of task-*i* in one resource:

$$w_i^{n+1}(p) = pC_i + B_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{J_j + w_i^n(p)}{T_j} \right\rceil C_j$$

This is carried out for p=1,2,3,..., until

$$w_i(p) \le pT_i$$

Then, the global worst-case response time is

$$R_i = max(R_i(p)) \qquad R_i(p) = w_i(p) - (p-1)T_i + J_i$$

Schedulability analysis of real-time distributed systems

1. Introduction
2. Single processor systems
3. *Distributed systems*
4. Schedulability analysis: holistic approach
5. Schedulability analysis: offset-based approaches
6. Schedulability analysis: EDF
7. Modelling techniques: MAST
8. Conclusion and future work

Real-time networks

Very few networks guarantee real-time response

- many protocols are based on collisions
- no priorities or deadlines in most protocols
  - Wireless: IEEE 802.11e, Sensor networks,...
  - IEEE 802.1p, with 8 priority levels

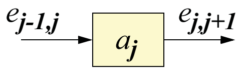
Some solutions

- CAN bus
- Priority-based token passing (e.g., RT-EP)
- Time-division multiplexed access
- Point to point networks

No commercial or standard EDF networks yet

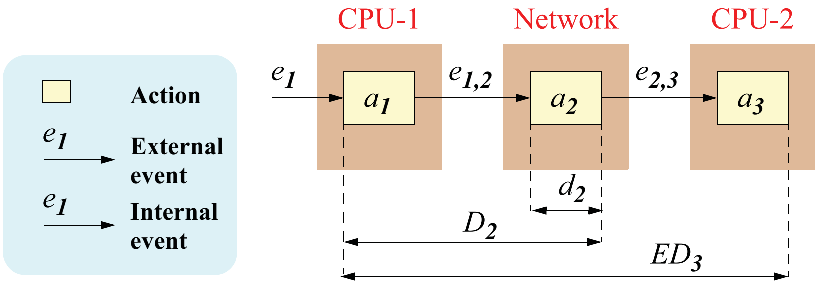
Distributed system model

Linear Action:



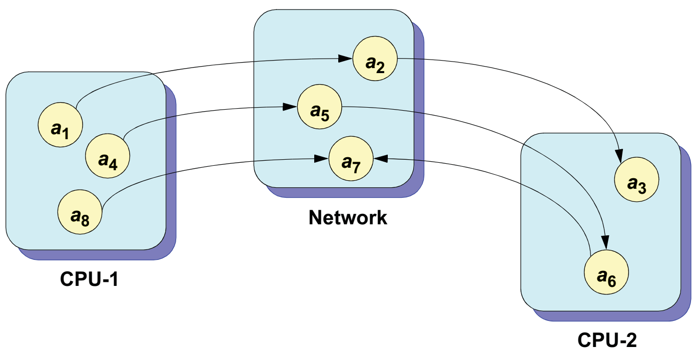
$T_{j-1,j} = T_j = T_{j,j+1}$

Linear Response to an Event:



Jitter in distributed systems

In the example below, actions a_2,a_3 and a_5,a_6,a_7,a_8 have jitter, even if a_1 and a_4 are purely periodic:



The problem

Mutual dependencies

- Jitter in one resource depends on the response times in other resources
- Response times depend on jitters

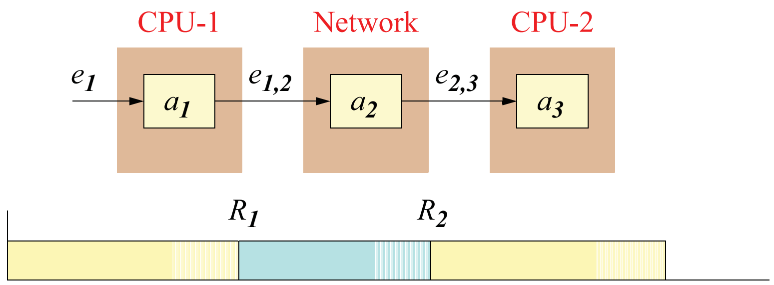
Solutions

- algorithms that can calculate jitter and response times
- change the scheduler
  - avoid jitter: phase control (requires global clocks)
  - eliminate the effects of jitter, with sporadic server scheduling

Avoiding jitter

Release each task (or message) at specific times in the schedule

- wait until the worst-case response time before releasing the next task or message
- requires global clocks, and OS and network driver cooperation



Eliminate the effects of jitter

Sporadic server scheduling

- guarantees an execution capacity (C_R)
- every replenishment period (T_R)

Features

- a minimum bandwidth for aperiodic events
- bounded preemption on lower priority tasks
  - effects like those of a periodic task
  - eliminates the effects of jitter

Not usually available in network schedulers

POSIX specifies sporadic server scheduling, but has a flaw

Schedulability analysis of real-time distributed systems

1. Introduction
2. Single processor systems
3. Distributed systems
4. *Schedulability analysis: holistic approach*
5. Schedulability analysis: offset-based approaches
6. Schedulability analysis: EDF
7. Advanced modelling techniques: MAST
8. Conclusion and future work

Holistic analysis technique

Mainly developed at the University of York

Each resource is analyzed separately (CPU’s and communication networks):

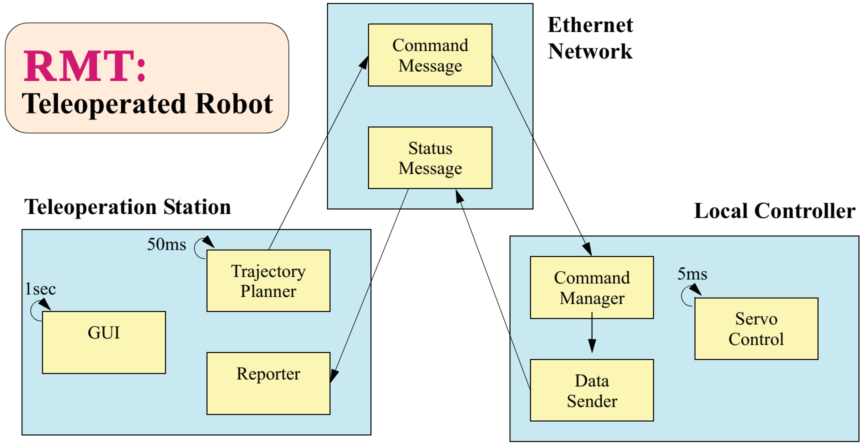
- all activations after the first have “*jitter*”
- *jitter* in one action is considered equal to the worst-case response time of the previous actions
- analysis in one resource affects *response times* in other resources
- the analysis is repeated *iteratively*, until a stable solution is achieved
- the method converges because of the *monotonic* relation between jitters and response times

Analysis in the distributed system

```
algorithm WCRT is
begin
    initialize jitter terms to zero
    loop
        calculate worst-case response times;
        calculate new jitters, equal to response
            times of preceding actions
        exit when not schedulable or
            no variation since last iteration;
    end loop;
end WCRT
```

Assumption: $J_i = R_i - R^b_i$, R^b_i = best-case response time=0

Example

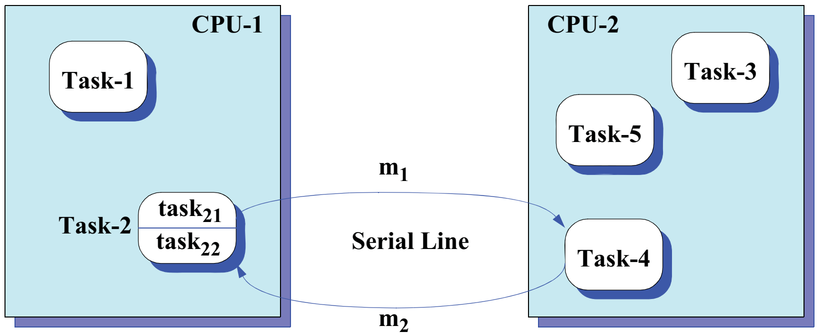


Schedulability analysis of real-time distributed systems

1. Introduction
2. Single processor systems
3. Distributed systems
4. Schedulability analysis: holistic approach
5. *Schedulability analysis: offset-based approaches*
6. Schedulability analysis: EDF
7. Advanced modelling techniques: MAST
8. Conclusion and future work

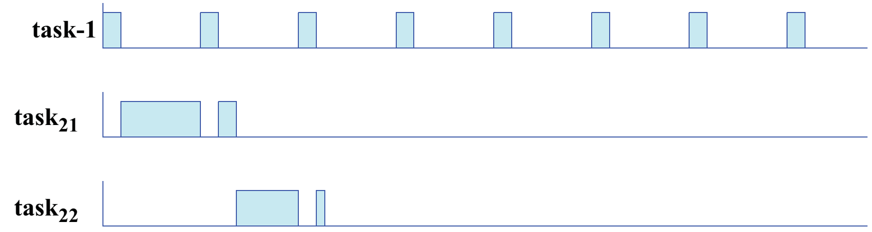
Pessimism in holistic analysis

Holistic analysis technique assumes independent task activations in each resource



Independent task analysis

Execution timeline for $task_{22}$ in previous example:

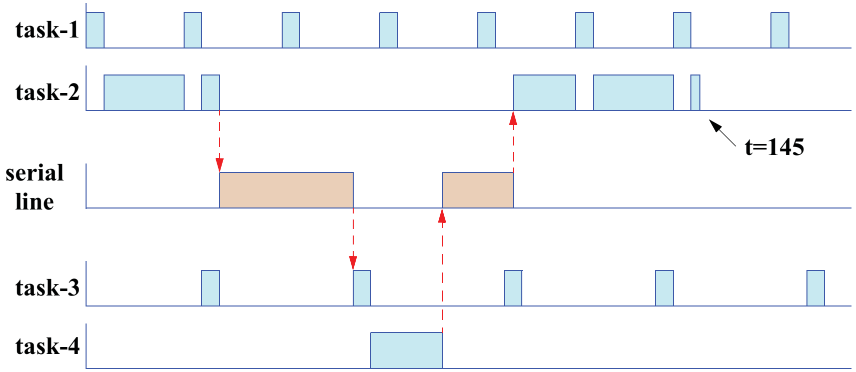


Response time for task-2:

- includes times for $task_{21}$, m_1, task-4, m_2 and $task_{22}$
- total is 270

Using offsets to reduce pessimism

Execution timeline for analysis of task-2:



Objectives of the offset-based techniques

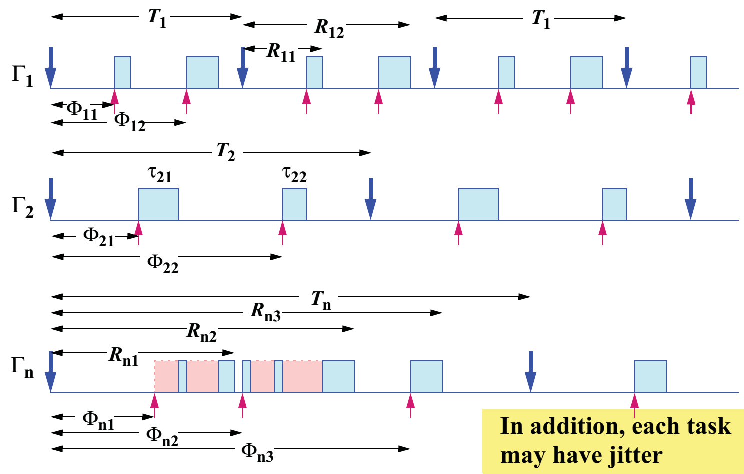
To reduce the pessimism of the worst-case analysis in multiprocessor and distributed systems:

- by considering offsets in the analysis
- offsets can be larger than task periods
  - this is important if deadlines > task periods
- offsets can be static or dynamic:
  - offsets are dynamic in distributed systems
  - also in tasks that suspend themselves

This enhancement comes "for free" as there is no change to the application

- although better results can be obtained if best-case execution times are measured

System model with offsets



Analysis with offsets

Developed by Tindell at the University of York:

- The exact analysis is intractable
- An upper-bound approximation provides good results (equal to exact analysis in 93% of tested cases)

Main limitations:

- Offsets are static
  - not applicable to general distributed transactions
- Offsets are less than the task periods
  - for distributed systems, deadlines would need to be smaller than or equal to the task periods

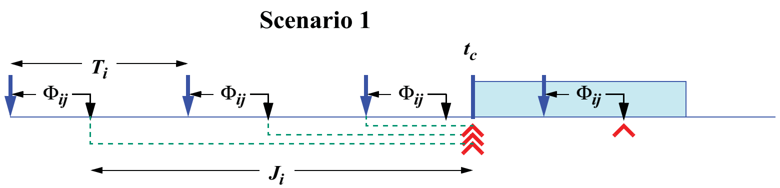
Extended in Cantabria to dynamic offsets and arbitrary deadlines

Exact analysis with static offsets

Contribution of task τ_{ij} to the response time of lower priority tasks:

- *Set 0*: activations that occur before the critical instant and that cannot occur inside the busy period
- *Set 1*: activations that occur before or at the critical instant, and that may occur inside the busy period
  - Theorem 1: worst-case when they all occur at the critical instant
  - Theorem 2: worst-case when the first one experienced its maximum jitter
- *Set 2*: activations that occur after the critical instant
  - Theorem 1: worst-case when they have zero jitter

Scenario for calculating the worst-case contribution of τ_{ij}



Upper bound approximation for worst-case analysis

Exact analysis is intractable:

- The task that generates the worst-case contribution of a given transaction is unknown
- The analysis has to check all possible combinations of tasks

Tindell developed an upper bound approximation:

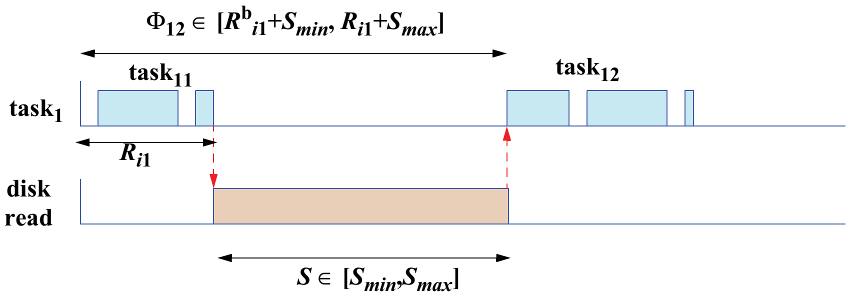
- For each transaction we consider a function that is the maximum of all the worst-case contributions considering each of the tasks of the transaction to be initiating the critical instant
- This technique is pessimistic, but pseudo-polynomial
- In 93% of the tested cases, the response times were exact

Analysis with dynamic offsets

In many systems the offset may be dynamic:

$$\Phi_{ij} \in [\Phi_{ij,\,min}, \Phi_{ij,\,max}]$$

Example: task with a suspending operation



Analysis with dynamic offsets (cont'd)

Dynamic offsets can be modeled with static offsets and jitter:

- **Equivalent offset:**

$$\Phi'_{ij} = \Phi_{ij,\,min}$$

- **Equivalent jitter:**

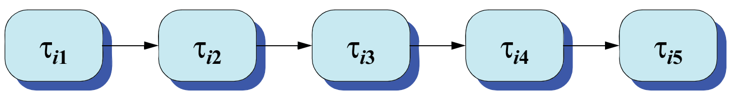
$$J'_{ij} = J_{ij} + (\Phi_{ij,\,max} - \Phi_{ij,\,min})$$

The problem is that now offsets depend on response times, and response times depend on offsets

- The solution is to apply the analysis iteratively, starting with response times = zero, until a stable solution is achieved
- We call this algorithm WCDO (Worst-Case Dynamic Offsets)

Analysis of multiprocessor and distributed systems

Distributed transaction Γ_i



Dynamic offsets in distributed transactions can also be modeled with static offsets and jitter:

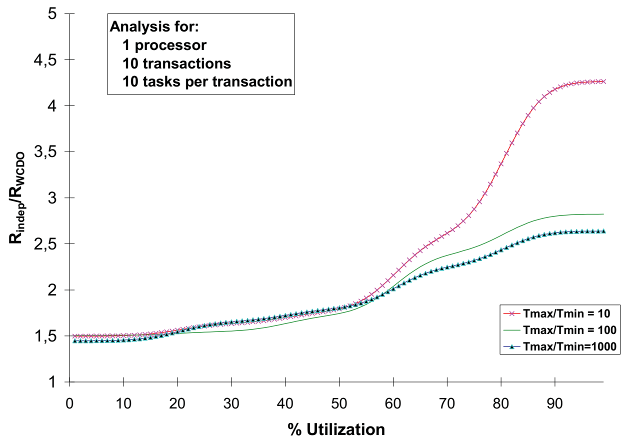
- **Equivalent offset:**

$$\Phi'_{ij} = \Phi_{ij,\,min} = R^b_{ij-1}$$

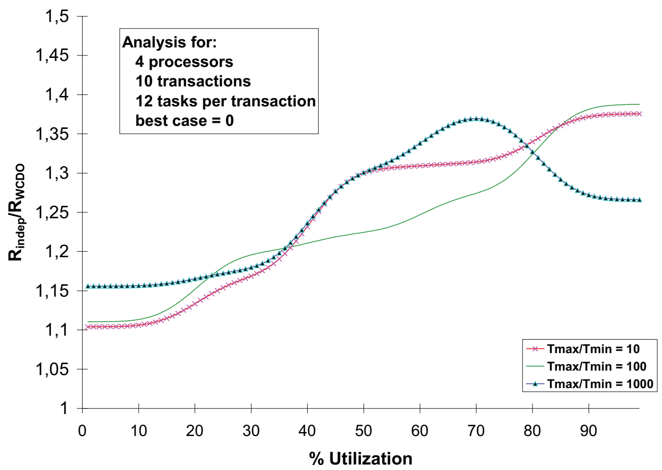
- **Equivalent jitter:**

$$J'_{ij} = J_{ij} + (\Phi_{ij,\,max} - \Phi_{ij,\,min}) = R_{ij-1} - R^b_{ij-1}$$

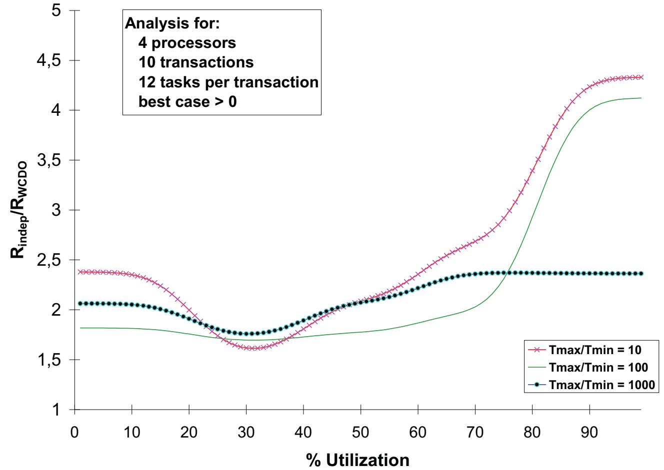
Comparison with existing technique: 1 processor



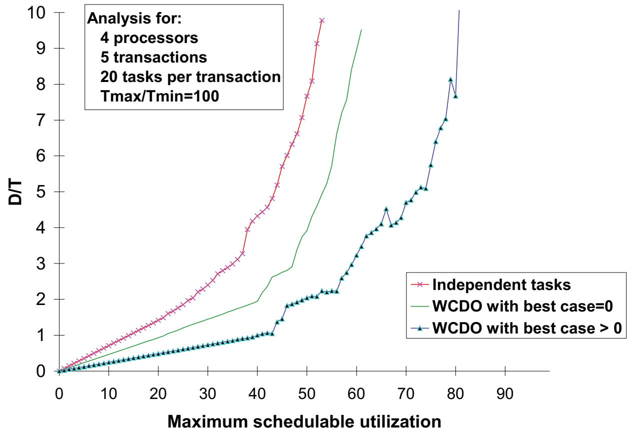
Comparison with four processors, and best-case=0



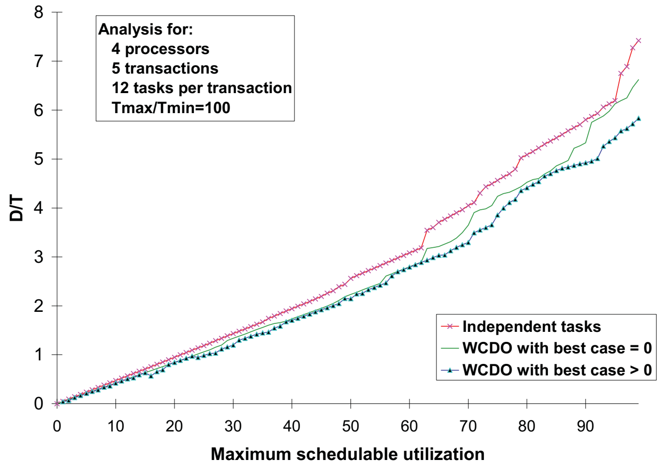
Comparison with four processors and best-case>0



Maximum utilization with 20 tasks per transaction



Maximum utilization with 12 tasks per transaction

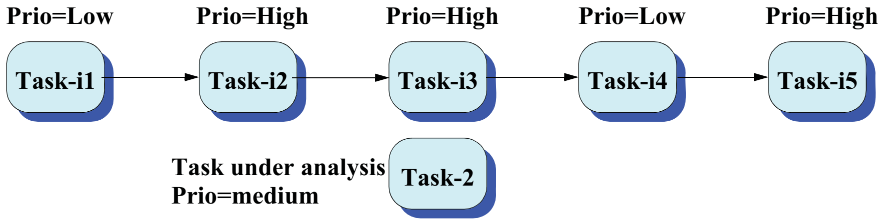


Room for improvement in offset-based analysis

Offset-based analysis produces results that are much less pessimistic than other methods (i.e., holistic analysis)

But offset-based analysis still has room for improvement

- a high priority task that is preceded by a low priority task may not be able to execute before a medium priority task under analysis



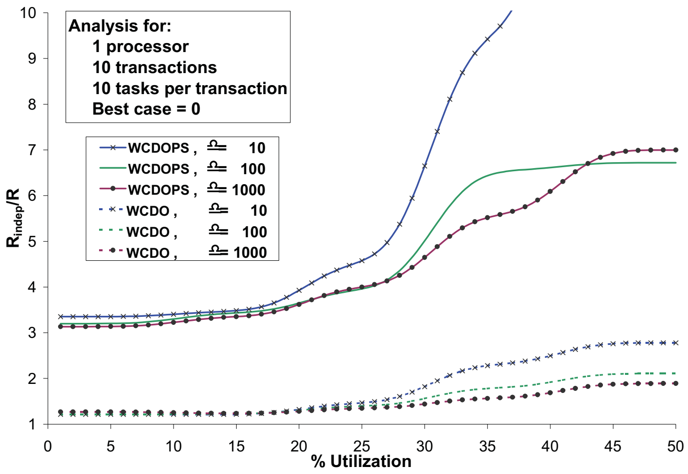
Objectives of optimized offset-based analysis

To enhance the offset-based schedulability analysis by:

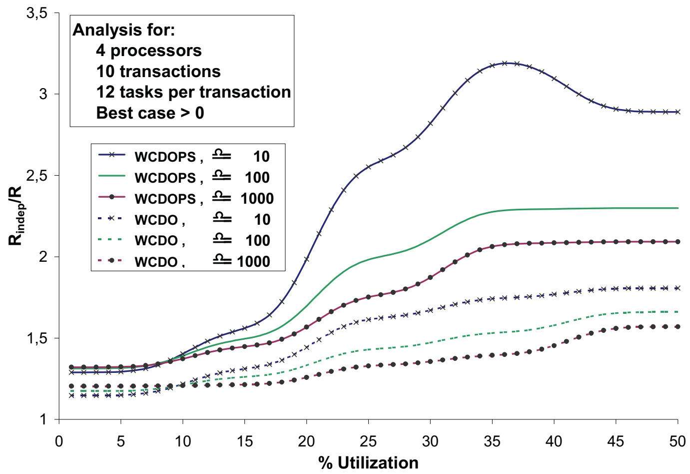
- Eliminating from the analysis the effects of higher priority tasks that cannot execute due to precedence constraints
- Eliminating the effects of the tasks that are preceded by the task under analysis

These enhancements reduce much of the pessimism in the analysis of distributed systems

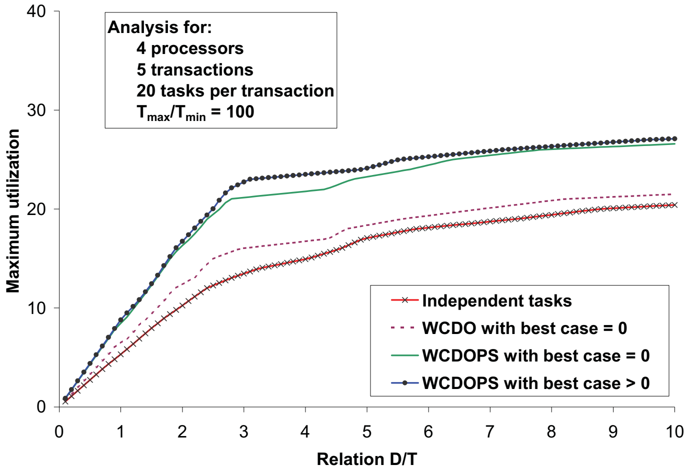
Simulation results: response times



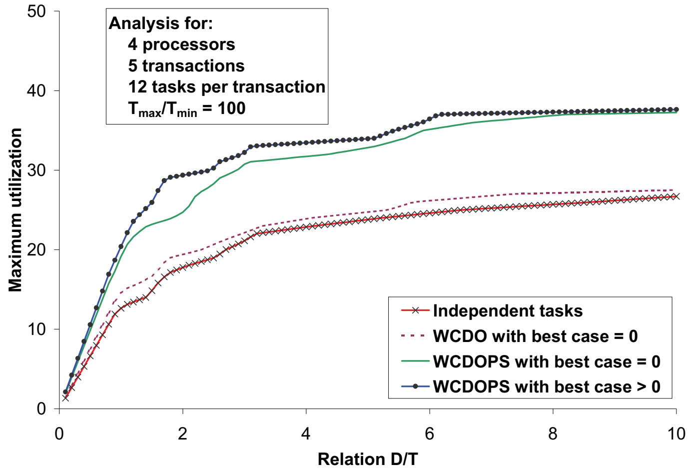
Simulation results: response times



Simulation results: utilization



Simulation results: utilization



Priority assignment techniques

No known optimum priority assignment

Simulated Annealing:

- Standard optimization technique
- Finds solutions by successively exchanging the priorities of a pair of tasks, and reapplying the analysis to determine if results are worse or better
- The probability of the change surviving is a function of the results
- Does not guarantee finding the solution

Priority assignment techniques (cont'd)

HOPA (Heuristic Optimized Priority Assignment)

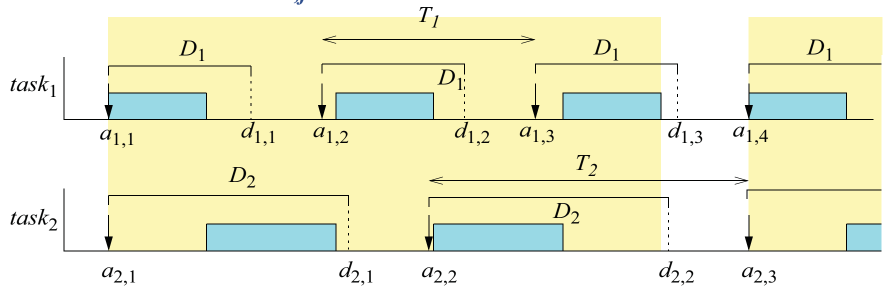
- Heuristic algorithm based on successively applying the analysis
- Much faster than simulated annealing
- Usually finds better solutions
- Does not guarantee finding the solution

Schedulability analysis of real-time distributed systems

7.2 EDF Scheduling Policy

Each task i has a relative period, T_i, and a relative deadline assigned: D_i

Each task-i job j has an absolute activation time, $a_{i,j}$, and an absolute deadline, $d_{i,j}$, used as the inverse of the job priority



Response time analysis in a single resource

Busy period: interval during which processor is not idle

Worst case ***response time*** of a task: found in a busy period in which all other tasks:

- are released at the beginning of the busy period
- and have experienced their maximum jitter

Differences with fixed priorities:

- The task under analysis does not necessarily start with the busy period
- The busy period is longer

Response time analysis in a single resource (cont'd)

Worst contribution of task τ_i to the busy period at time t, when the deadline of the analyzed task, τ_a, is D:

$$W_i(t, D) = min\left(\left\lceil \frac{t + J_i}{T_i} \right\rceil, \left\lfloor \frac{J_i + D - d_i}{T_i} \right\rfloor + 1\right)_0 \cdot C_i$$

Worst completion time of activation p, if first activation at A:

$$w_a^A(p) = pC_a + \sum_{\forall i \neq a} W_i(w_a^A(p), D^A(p))$$

Worst response time if first activation is A:

$$R^A(p) = w_a^A(p) - A + J_a - (p-1)T_a$$

Response time analysis in a single resource (cont'd)

Set of potential critical instants; L is the longest busy period:

$$\Psi = \bigcup\{(p-1)T_i - J_i + d_i\} \qquad \forall p = 1\ldots\left\lceil \frac{L-J_a}{T_a}\right\rceil, \forall i \neq a$$

Values of A to check:

$$\Psi^* = \{\Psi_x \in \Psi \mid (p-1)T_a - J_a + d_a \le \Psi_x < pT_a - J_a + d_a\}$$

$$A = \Psi_x - [(p-1)T_a - J_a + d_a]$$

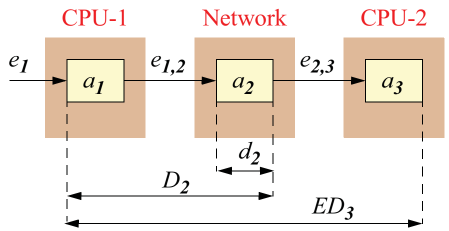
Worst-case response time

$$R_a = max[R^A(p)] \qquad \forall p = 1\ldots\left\lceil \frac{L-J_a}{T_a}\right\rceil, \forall A \in \Psi^*$$

EDF in distributed systems

Two scheduling schemes can be used in distributed systems

- **Global deadlines**
  - relative to the external event arrival, require global clock synchronization
- **Local deadlines**
  - relative to the internal event arrival, no global clocks needed



Holistic analysis for EDF systems

Developed by *Spuri* for global deadlines

- similar to the holistic analysis developed for fixed priorities at the University of York

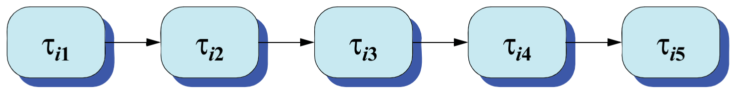
Each resource is analyzed separately (CPU's and networks):

- all activations after the first present "*jitter*"
- the analysis is repeated, until a stable solution is achieved
- the solution is pessimistic

Extended to local deadlines by Palencia (2009, submitted)

Offset-based analysis for EDF with global deadlines (Palencia, 2003)

Distributed transaction Γ_i



Dynamic offsets in distributed transactions can also be modeled with static offsets and jitter:

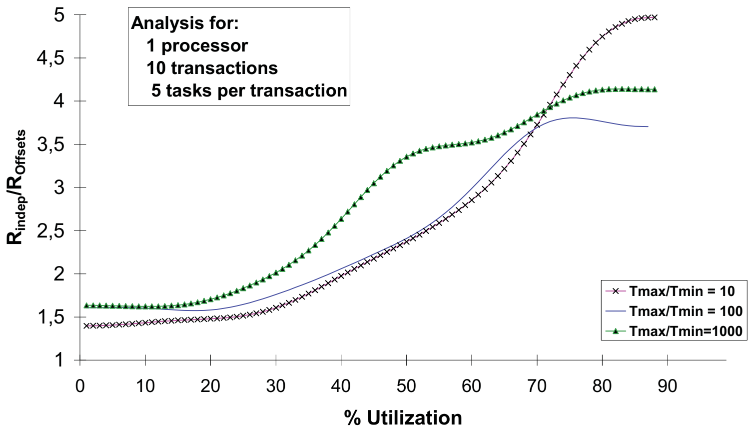
- **Equivalent offset:**

$$\Phi'_{ij} = \Phi_{ij,\,min} = R^b_{ij-1}$$

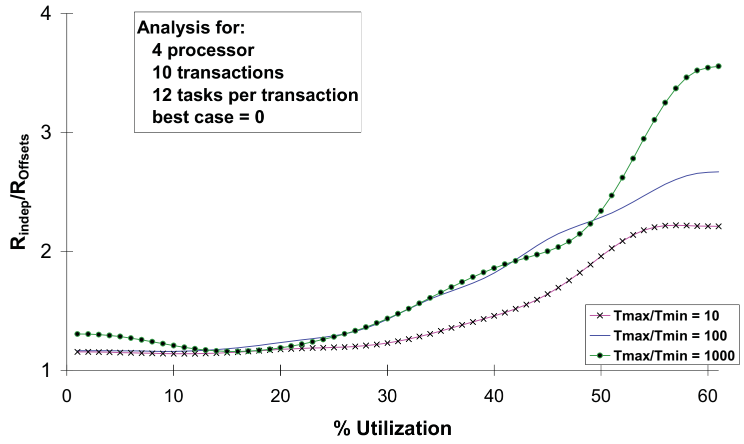
- **Equivalent jitter:**

$$J'_{ij} = J_{ij} + (\Phi_{ij,\,max} - \Phi_{ij,\,min}) = R_{ij-1} - R^b_{ij-1}$$

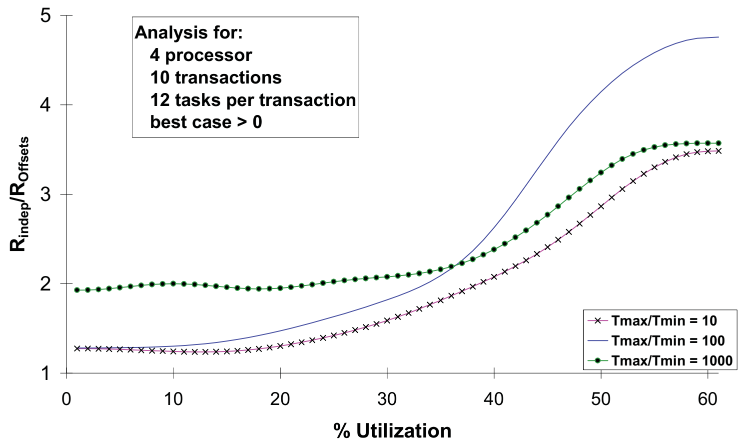
Comparison with holistic analysis: 1 processor



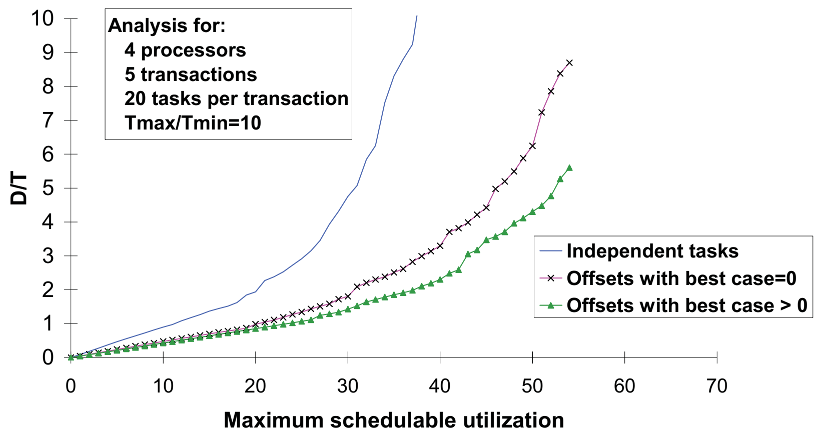
Comparison with four processors, and best-case=0



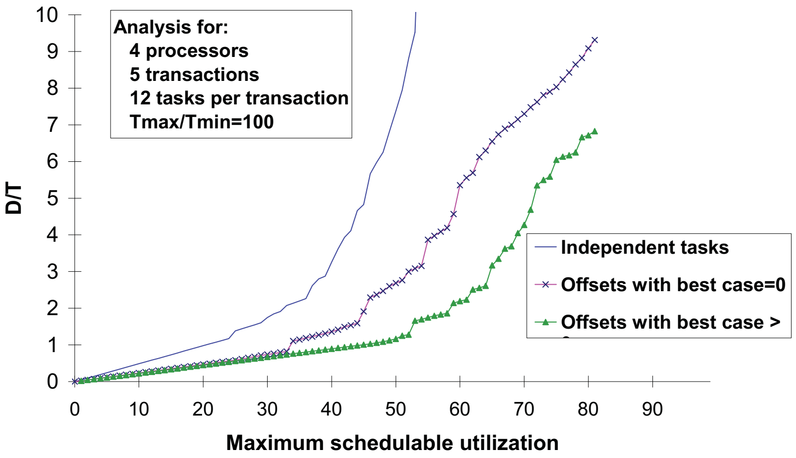
Comparison with four processors and best-case>0



Maximum utilization with 20 tasks per transaction



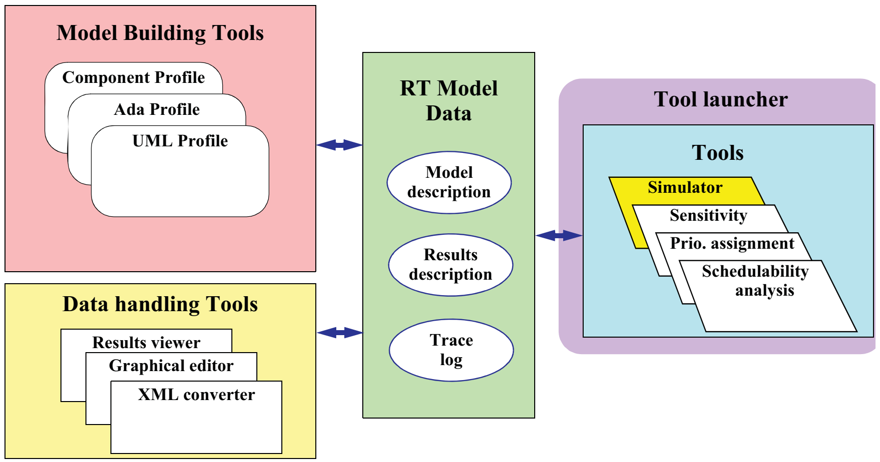
Maximum utilization with 12 tasks per transaction



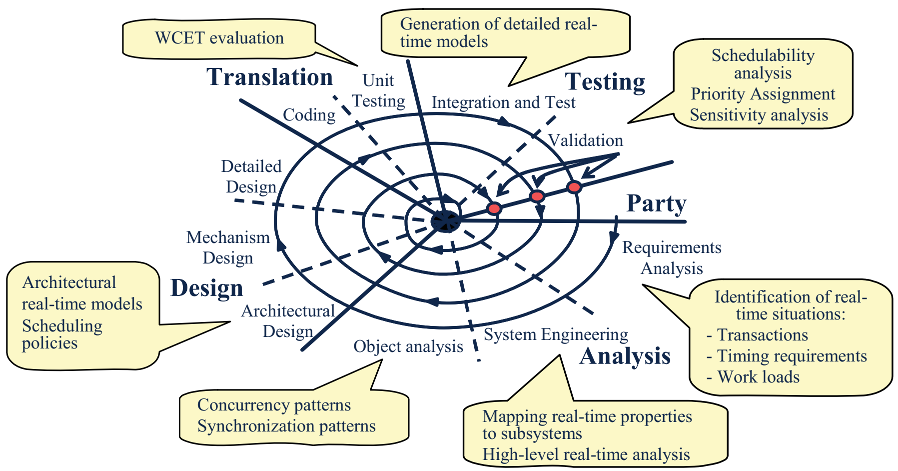
Schedulability analysis of real-time distributed systems

1. Introduction
2. Single processor systems
3. Distributed systems
4. Schedulability analysis: holistic approach
5. Schedulability analysis: offset-based approaches
6. Schedulability analysis: EDF
7. *Modelling techniques: MAST*
8. Conclusion and future work

MAST Environment



Real-time analysis: integration into the design process



Fixed Priority Response-Time Analysis

Technique	Single-Processor	Multi-Processor	Simple Transact.	Linear Transact.	Multiple Event T.
Classic Rate Mono-tonic	☑		☑		
Varying Priorities	☑		☑	☑	
Holistic	☑	☑	☑	☑	
Offset Based Unopti-mized	☑	☑	☑	☑	
Offset Based	☑	☑	☑	☑	
Multiple Event	☑	☑	☑	☑	☑

EDF Response Time Analysis Tools

Technique	Single-Processor	Multi-Processor	Simple Transact.	Linear Transact.	Multiple Event T.
Single Processor	☑		☑		
EDF_Within_Priorities	☑		☑		
Holistic - local	☑	☑	☑	☑	
Offset Based - local					
Holistic - global	☑	☑	☑	☑	
Offset Based - global	☑	☑	☑	☑	

An unsolved problem remains for distributed systems with SRP (Stack-based Resource assignment Protocol)

Priority assignment techniques

Technique	Single-Processor FP	Multi-Processor FP	Single-Processor EDF	Multi-Processor EDF
Monoprocessor	☑		☑	
HOPA / HOSDA	☑	☑	☑	☑
Simulated Annealing	☑	☑		

8. Conclusion

A solid analysis body exists for analyzing fixed priority distributed systems

- response time analysis
- offset-based techniques better than holistic analysis
  - but still pessimistic

Theoretical developments still needed for EDF distributed systems

- integration of SRP and response time analysis
- offset-based techniques for local deadlines
- precedence constraints in offset-based techniques

Technical developments still needed in

- real-time networks and distribution middleware

Conclusion (cont'd)

Modelling and analysis tools exist for distributed real time systems

MAST defines a model for describing real-time systems

- distributed and multiprocessor
- composable software modules
- independence of architecture, platform and modules

MAST provides an open set of tools

- schedulability analysis
- sensitivity analysis
- automatic blocking times
- priority assignment, ...

Future work: MAST

Finish current tools and add new ones

- integrate simulator
- Multiple-Event Analysis
- Various EDF tools
- Multiprocessor/multicore analysis
- Time-partitioned approaches
- Solve the RPC transaction model

Integration with MAST+

Integration with MARTE UML profile concepts

- MAST-2

Future work: Real-time analysis

Support for EDF:

- Incorporate aperiodic servers (i.e., CBS) into the response-time analysis for EDF
- Mixed EDF/FP analysis for distributed systems
- Solve the distributed EDF analysis with SRP blocking
- Extend the hierarchical scheduler analysis to distributed systems

Fixed priority systems

- Solve the multiple-event system analysis using offset-based analysis techniques

Multiprocessor and multicore scheduling policies

References

Books

[1] M.H. Klein, T. Ralya, B. Pollak, R. Obenza, and M. González Harbour. "A practitioner's Handbook for Real-Time Analysis". Kluwer Academic Pub., 1993.

[2] A. Burns and A. Wellings. "Real-Time Systems and Programming Languages". Second Edition. Addison-Wesley, 2002.

[3] G. Buttazzo. "Hard Real-Time Computing Systems". Kluwer Academic Pub., 1997.

[4] Liu J.S.W., "Real Time Systems". Prentice Hall, 2000.

Real-Time analysis in single processor systems

[5] J.P. Lehoczky, L. Sha, J.K. Strosnider, and H. Tokuda. "Fixed-Priority Scheduling for Hard Real-Time Systems". In *Foundations of Real-Time Computing: Scheduling and Resource Management*, Van Tilborg, Andre and Koob, Gary M., Ed., Kluwer Academic Publishers, 1991, pp. 1-30.

References (continued)

[6] L. Sha, M.H. Klein, and J.B. Goodenough. "Rate Monotonic Analysis for Real-Time Systems". In *Foundations of Real-Time Computing: Scheduling and Resource Management*, Van Tilborg, Andre and Koob, Gary M., Ed., Kluwer Academic Publishers, 1991, pp. 129-155.

[7] M.H. Klein, J.P. Lehoczky, and R. Rajkumar. "Rate-Monotonic Analysis for Real-Time Industrial Computing". IEEE Computer, Vol. 27, No. 1, January 1994.

[8] J.A. Stankovic. "Misconceptions About Real-Time Computing". IEEE Computer, Vol. 21, No. 10, October 1988.

[9] J.A. Stankovic and K. Ramamritham "What is Predictability for Real-Time Systems?". Real-Time Systems Journal, Vol. 2, 247-254, 1990.

[10] T.P. Baker, A. Shaw. "The Cyclic executive Model and Ada". *Proceedings of the IEEE Real-Time Systems Symposium*, December 1988.

References (continued)

[11] C.L. Liu, and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment". Journal of the ACM, 20 (1), pp 46-61, 1973.

[12] P.K. Harter, "Response times in level-structured systems". ACM Transactions on Computer Systems, vol. 5, no. 3, pp. 232-248, Aug. 1984.

[13] M. Joseph and P. Pandya, "Finding Response Times in a Real-Time System". The Computer Journal (British Computing Society) 29, 5, pp. 390-395, October 1986.

[14] J. Leung, and J.W. Layland. "On Complexity of Fixed-Priority Scheduling of Periodic Real-Time Tasks". *Performance Evaluation 2*, 237-50, 1982.

[15] J.P. Lehoczky, L. Sha, and Y. Ding. "The Rate Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behavior". *Proceedings of the IEEE Real-Time Systems Symposium*, 1989.

[16] J.P. Lehoczky. "Fixed Priority Scheduling of Periodic Tasks Sets with Arbitrary Deadlines". *IEEE Real-Time Systems Symposium*, 1990.

References (continued)

[17] M. González Harbour, M.H. Klein, and J.P. Lehoczky. "Fixed Priority Scheduling of Periodic Tasks with Varying Execution Priority". *Proceedings of the IEEE Real-Time Systems Symposium*, December 1991, pp. 116-128.

[18] M. González Harbour, M.H. Klein, and J.P. Lehoczky. "Timing Analysis for Fixed-Priority Scheduling of Hard Real-Time Systems". IEEE Trans. on Software Engineering, Vol. 20, No.1, January 1994.

[19] K. Tindell, "An Extendible Approach for Analysing Fixed Priority Hard Real-Time Tasks". Journal of Real-Time Systems, Vol. 6, No. 2, March 1994.

[20] M. Spuri. "Analysis of Deadline Scheduled Real-Time Systems". RR-2772, INRIA, France, 1996.

[21] M. González Harbour and J.C. Palencia, "Response Time Analysis for Tasks Scheduled under EDF within Fixed Priorities", Proceedings of the 24th IEEE Real-Time Systems Symposium, Cancun, México, December, 2003.

References (continued)

[22] M.H. Klein, and T. Ralya. "An Analysis of Input/Output Paradigms for Real-Time Systems". Technical Report CMU/SEI-90-TR-19, Software Engineering Institute, July 1990.

[23] Audsley, N.C., "Optimal Priority Assignment and Feasibility of Static Priority Tasks with Arbitrary Start Times", Dept. Computer Science, University of York, December 1991.

Aperiodic Scheduling

[24] B. Sprunt, L. Sha, and J.P. Lehoczky. "Aperiodic Task Scheduling for Hard Real-Time Systems". *The Journal of Real-Time Systems*, Vol. 1, 1989, pp. 27-60.

[25] M. González Harbour, J.J. Gutiérrez García, J.C. Palencia Gutiérrez. "Implementing Application-Level sporadic Server Schedulers in Ada 95". International Conference on Reliable Software Technologies: Ada-Europe'97. London, UK, 1997.

[26] L. Abeni and G. Buttazzo. "Integrating Multimedia Applications in Hard Real-Time Systems". Proceedings of the IEEE Real-Time Systems Symposium, Madrid, Spain, December 1998

References (continued)

Distributed Real-Time Systems

[27] L. Sha, and S.S Sathaye. "A Systematic Approach to Designing Distributed Real-Time Systems". IEEE Computer, Vol 26., No. 9, September 1993.

[28] K.W. Tindell, A. Burns, and A.J. Wellings. "Allocating Real-Time Tasks. An NP-Hard Problem Made Easy". *Real-Time Systems Journal*, Vol. 4, No. 2, May 1992. pp. 145- 166.

[29] K.W. Tindell, and J. Clark. "Holistic Schedulability Analysis for Distributed Hard Real-Time Systems". Microprocessing and Microprogramming, Vol. 50, Nos. 2-3, pp 117-134, April 1994.

[30] M. Spuri. "Holistic Analysis of Deadline Scheduled Real-Time Distributed Systems". RR-2873, INRIA, France, 1996.

[31] J.C. Palencia Gutiérrez, J.J. Gutiérrez García, M. González Harbour. "On the schedulability analysis for distributed hard real-time systems". 9th Euromicro Workshop on Real-Time Systems. Toledo, 1997.

References (continued)

[32] Gutiérrez García J.J. and González Harbour M.: "Optimized Priority Assignment for Tasks and Messages in Distributed Real-Time Systems". Proceedings of the 3rd Workshop on Parallel and Distributed Real-Time Systems, Santa Barbara, California, pp. 124-132, April 1995.

[33] Palencia Gutiérrez J.C., Gutiérrez García J.J., González Harbour M.: "Best-Case Analysis for Improving the Worst-Case Schedulability Test for Distributed Hard Real-Time Systems". Proceeding of the 10th IEEE Euromicro Workshop on Real-Time Systems, Berlin, Germany, June 1998.

[34] K. Tindell. "Adding Time-Offsets to Schedulability Analysis", Technical Report YCS 221, Dept. of Computer Science, University of York, England, January 1994.

[35] Palencia Gutiérrez J.C., González Harbour M.: "Schedulability Analysis for Tasks with Static and Dynamic Offsets". Proceedings of the 18th. IEEE Real-Time Systems Symposium, Madrid, Spain, December 1998.

References (continued)

[36] J.C. Palencia, and M. González Harbour, "Exploiting Precedence Relations in the Schedulability Analysis of Distributed Real-Time Systems". Proceedings of the 20th IEEE Real-Time Systems Symposium, 1999.

[37] J.J. Gutiérrez García, J.C. Palencia Gutiérrez, and M. González Harbour, "Schedulability Analysis of Distributed Hard Real-Time Systems with Multiple-Event Synchronization". Euromicro Conference on Real-Time Systems, Stockholm, Sweden, 2000.

[38] J.C. Palencia and M. González Harbour, "Offset-Based Response Time Analysis of Distributed Systems Scheduled under EDF". Euromicro conference on real-time systems, Porto, Portugal, June 2003.

[39] Redell, O. Response Time Analysis for Implementation of Distributed Control Systems, Doctoral Thesis, TRITA-MMK 2003:17, ISSN 1400-1179, ISRN KTH/MMK/R--03/17--SE, 2003

References (continued)

Synchronization

[40] L. Sha, R. Rajkumar, and J.P. Lehoczky. "Priority Inheritance Protocols: An approach to Real-Time Synchronization". IEEE Trans. on Computers, September 1990.

[41] Baker T.P., "Stack-Based Scheduling of Realtime Processes", Journal of Real-Time Systems, Volume 3, Issue 1 (March 1991), pp. 67–99.

[42] J.B. Goodenough, and L. Sha. "The Priority Ceiling Protocol: A Method for Minimizing the Blocking of High Priority Ada Tasks". *Proceedings of the 2nd International Workshop on Real-Time Ada Issues*, June 1988.

[43] R. Rajkumar, L. Sha, and J.P. Lehoczky. "Real-Time Synchronization Protocols for Multiprocessors". *IEEE Real-Time Systems Symposium*, December 1988.

[44] R. Rajkumar. "Real-Time Synchronization Protocols for Shared Memory Multiprocessors". *Proceedings of the 10th International Conference on Distributed Computing*, 1990.

[45] L. Sha, R. Rajkumar, J.P. Lehoczky. "Real-Time Computing with IEEE Futurebus+". *IEEE Micro*, June 1991, pp. 30-33, and 95-100.

References (continued)

Priority assignment

[46] J. Leung, and J.W. Layland. “On Complexity of Fixed-Priority Scheduling of Periodic Real-Time Tasks”. Performance Evaluation 2, 237-50, 1982.

[47] N.C. Audsley, “Optimal Priority Assignment and Feasibility of Static Priority Tasks with Arbitrary Start Times”, Dept. Computer Science, University of York, December 1991.

[48] K.W. Tindell, A. Burns, and A.J. Wellings. “Allocating Real-Time Tasks. An NP-Hard Problem Made Easy”. Real-Time Systems Journal, Vol. 4, No. 2, May 1992. pp. 145- 166.

[49] J.J. Gutiérrez García and M. González Harbour. “Optimized Priority Assignment for Tasks and Messages in Distributed Real-Time Systems”. Proceedings of 3rd Workshop on Parallel and Distributed Real-Time Systems, Santa Barbara, California, pp. 124-132, 1995.

References (continued)

Initial work on RMA

[50] C.L. Liu and J.W. Layland. “Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment”. Journal of the ACM, Vol. 20, No. 1, 1973, pp. 46-61.

[51] M. Joseph and P. Pandya, “Finding Response Times in a Real-Time System,” *BCS Computer Journal*, Vol. 29, no. 5, October 1986, pp 390-395.

[52] B.W. Lampson, and D.D. Redell. “Experience with Processes and Monitors in Mesa”. *Communications of the ACM* 23, 2, February 1980, pp. 105-107.

References (continued)

Real-Time operating systems and networks

[53] K.Tindell, A. Burns and A. Wellings. “Calculating Controller Area Network (CAN) message response times”. Proc. of the 1994 IFAC Workshop on Distributed Computer Control Systems (DCCS), Toledo, Spain.

[54] M. Aldea and M. González. “MaRTE OS: An Ada Kernel for Real-Time Embedded Applications”. Proceedings of the International Conference on Reliable Software Technologies, Ada-Europe-2001, Leuven, Belgium, Lecture Notes in Computer Science, LNCS 2043, May, 2001.

[55] Juan López Campos, J. Javier Gutiérrez and M. González Harbour. “CAN-RT-TOP: Real-Time Task-Oriented Protocol over CAN for Analyzable Distributed Applications”. 3rd International Workshop on Real-Time Networks (RTN), Catania, Sicily (Italy), July 2004.

[56] José María Martínez and Michael González Harbour. “RT-EP: A Fixed-Priority Real Time Communication Protocol over Standard Ethernet”. to appear in the Proceedings of the International Conference on Reliable Software Technologies, Ada-Europe-2005, York, UK, June 2005.

References (continued)

[57] ISO/IEC 9945-1:2003. Standard for Information Technology -Portable Operating System Interface (POSIX).

[58] IEEE Std. 1003.13-2003. Information Technology -Standardized Application Environment Profile- POSIX Realtime and Embedded Application Support (AEP). The Institute of Electrical and Electronics Engineers.

[59] CAN Specification Version 2.0. 1991, Robert Bosch GmbH, Postfatch 30 02 40, D-70442 Stuttgart.

[60] MaRTE OS home page. `http://marte.unican.es/`

[61] SHaRK home page. `http://shark.sssup.it/`

References (continued)

Ada

[62] Ada Language Reference Manual, ANSI, 1983.

[63] International Standard ISO/IEC 8652:1995, "Information Technology, Programming Languages, Ada Reference Manual", January 1995

References (continued)

POSIX: Standard Operating System Interface

[64] M. González Harbour y C.D. Locke. "Tostadores y POSIX". Novática, Vol. 129, Octubre 1997.

[65] B.O. Gallmeister, and C. Lanier. "Early Experience with POSIX 1003.4 and POSIX 1003.4a". *Proceedings of the IEEE Real-Time Systems Symposium*, December 1991, pp. 190-198.

[66] B.O. Gallmeister. "POSIX.4: Programming for the Real World". O'Reilly & Associates, Inc., 1995.

[67] B. Nichols, D. Buttlar and J. Proulx Farrell. "Pthreads Programming". O'Reilly & associates, Inc., 1996

[68] S. Kleiman, et al., "Programing with Threads". Prentice Hall, 1996.

References (continued)

[69] ISO/IEC Standard 9945-1:1996, "*Information Technology -Portable Operating System Interface (POSIX)- Part 1: System Application Program Interface (API) [C Language]*". Institute of Electrical and electronic Engineers, 1996.

[70] IEEE Standard 1003.1d:1999, "*Standard for Information Technology -Portable Operating System Interface (POSIX)- Part 1: System Application Program Interface (API) [C Language]. Realtime Extension*". The Institute of Electrical and Electronics Engineers, 1999.

[71] IEEE Standards Project P1003.1j, "*Draft Standard for Information Technology -Portable Operating System Interface (POSIX)- Part 1: System Application Program Interface (API) [C Language]. Advanced Realtime Extension*". Draft 9. The Institute of Electrical and Electronics Engineers, October 1999

[72] IEEE Standard 1003.13:1998, "*Standard for Information Technology -Standardized Application Environment Profile- POSIX Realtime Application Support (AEP)*". The Institute of Electrical and Electronics Engineers, 1998

References (concluded)

MAST

[73] J.M. Drake, M. González Harbour J.L. Medina: "MAST Real-Time View: Graphic UML tool for modeling object-oriented real time systems." Group of Computers and Real-Time Systems. University of Cantabria (Internal report), 2000.

[74] M. González Harbour, J.J. Gutiérrez, J.C. Palencia and J.M. Drake. "MAST: Modeling and Analysis Suite for Real-Time Applications". Proceedings of the Euromicro Conference on Real-Time Systems, Delft, The Netherlands, June 2001.

[75] MAST home page. http://mast.unican.es

FRESCOR project

[76] FRESCOR home page. http://frescor.org