INSTITUTE OF COMMUNICATION, INFORMATION AND PERCEPTION TECHNOLOGIES

Scuola Superiore Sant'Anna

Energy Saving Exploiting the Limited Preemption Task Model

M. Bambagini, G. Buttazzo, M. Bertogna, M. Marinoni

3rd International Real-Time Scheduling Open Problems Seminar (RTSOPS 12)
Pisa, Italy, July 10 2012

TeCIP Institute, Scuola Superiore Sant'Anna
Area della Ricerca CNR, Via Moruzzi, 1
56127 Pisa, ITALY

Retis
Real-Time Systems Laboratory

Outline

Introduction - Energy saving

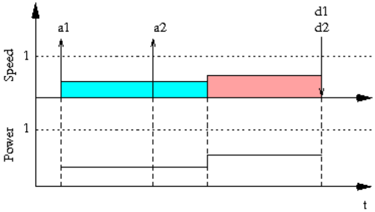
The energy saving techniques are divided in:

- DVFS, Dynamic Voltage and Frequency Scaling
  - Pillai (2001), Aydin (2004), Bini [1] (2005), Gong (2007)
- DPM, Dynamic Power Management
  - Huang [2] (2009), Rowe [4] (2010), Awan and Petters [3] (2011)

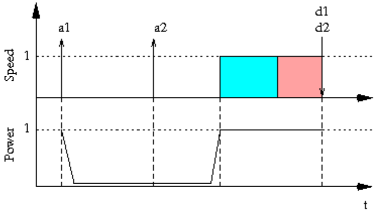
Introduction - Energy saving

The energy saving techniques are divided in:

- DVFS, Dynamic Voltage and Frequency Scaling
  - Pillai (2001), Aydin (2004), Bini [1] (2005), Gong (2007)
- DPM, Dynamic Power Management
  - Huang [2] (2009), Rowe [4] (2010), Awan and Petters [3] (2011)



DVFS

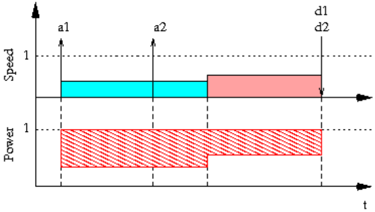


DPM

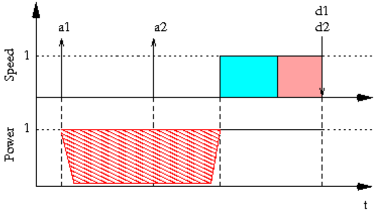
Introduction - Energy saving

The energy saving techniques are divided in:

- DVFS, Dynamic Voltage and Frequency Scaling
  - Pillai (2001), Aydin (2004), Bini [1] (2005), Gong (2007)
- DPM, Dynamic Power Management
  - Huang [2] (2009), Rowe [4] (2010), Awan and Petters [3] (2011)



DVFS



DPM

The technique usefulness depends on the HW's power function

Introduction - Deferred-Preemption model

Originally introduced by Burns in 1994 (Cooperative scheduling of Fixed Preemption Point model)

Bril et al. [6] presented an exact schedulability analysis for FP

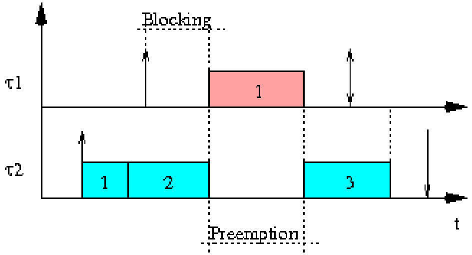
Introduction - Deferred-Preemption model

Originally introduced by Burns in 1994 (Cooperative scheduling of Fixed Preemption Point model)

Bril et al. [6] presented an exact schedulability analysis for FP

Tasks are divided into a set of non-preemptive chunks

Preemptions can happen only after the end of a chunk and before the starting of the next one



Introduction - Deferred-Preemption model

Deferred-Preemption benefits (with Fixed Priority scheduler):

1. a reduced number of preemptions → lower overhead
2. limited preemptive increases schedulability with respect to fully preemptive and non-preemptive, even with negligible preemption costs (Bertogna et al. [5])

Introduction - Deferred-Preemption model

Deferred-Preemption benefits (with Fixed Priority scheduler):

1. a reduced number of preemptions $\rightarrow$ lower overhead
2. limited preemptive increases schedulability with respect to fully preemptive and non-preemptive, even with negligible preemption costs (Bertogna et al. [5])

In other words:

- if Γ feasible under Fully-P $\implies$ Γ feasible under Limited-P

Introduction - Deferred-Preemption model

Deferred-Preemption benefits (with Fixed Priority scheduler):

1. a reduced number of preemptions $\rightarrow$ lower overhead
2. limited preemptive increases schedulability with respect to fully preemptive and non-preemptive, even with negligible preemption costs (Bertogna et al. [5])

In other words:

- if Γ feasible under Fully-P $\implies$ Γ feasible under Limited-P
- if Γ feasible under Limited-P $\not\Longrightarrow$ Γ feasible under Fully-P

Motivational example

Consider a system with two speeds ($s_1 = 0.5$ and $s_2 = 1$) and a task set Γ, scheduled using RM:

- τ_1: high priority, $C_1 = 30$ (at $s = 1$) and $T_1 = D_1 = 80$
- τ_2: low priority, $C_2 = 25$ (at $s = 1$) and $T_2 = D_2 = 200$

Motivational example

Consider a system with two speeds ($s_1 = 0.5$ and $s_2 = 1$) and a task set Γ, scheduled using RM:

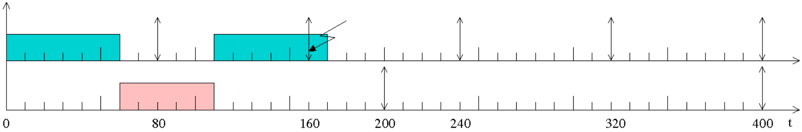
- τ_1: high priority, $C_1 = 30$ (at $s = 1$) and $T_1 = D_1 = 80$
- τ_2: low priority, $C_2 = 25$ (at $s = 1$) and $T_2 = D_2 = 200$

At $s = s_2 = 1$ ($U = 0.5$), the task set is feasible with the models:

- Non-Preemptive
- Fully-Preemptive
- Limited-Preemptive

Motivational example

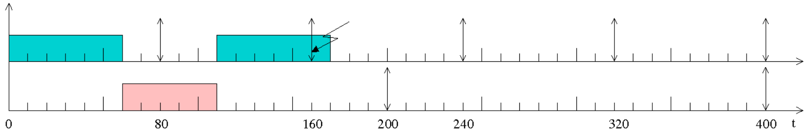
At $s = s_1 = 0.5$ ($\tau_1 : 60/80, \tau_2 : 50/200,\ U = 1$):



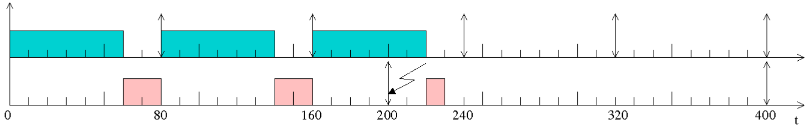
Non-Preemptive model

Motivational example

At $s = s_1 = 0.5$ ($\tau_1 : 60/80, \tau_2 : 50/200,\ U = 1$):



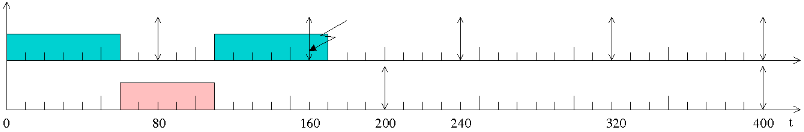
Non-Preemptive model



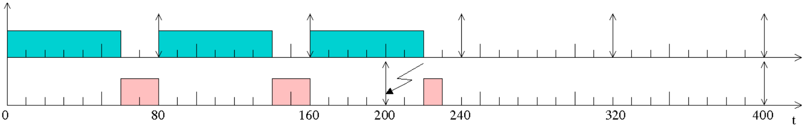
Fully-Preemptive model

Motivational example

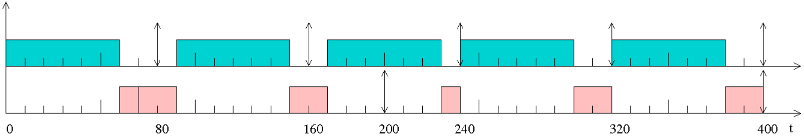
At $s = s_1 = 0.5$ ($\tau_1 : 60/80, \tau_2 : 50/200,\ U = 1$):



Non-Preemptive model



Fully-Preemptive model



Limited-Preemptive model

Open problem

The deferred preemption model optimizes the tasks execution, giving us extra slack time that is useful to save further energy

But, how can it be effectively exploited for reducing the overall energy consumption?

Possible approaches: DVFS and DPM

Open problem - DVFS approaches

Off-line: how can the speed be computed efficiently during PPP?

Open problem - DVFS approaches

Off-line: how can the speed be computed efficiently during PPP?

Online: what kind of algorithms can be used?

- A slower speed makes non-preemptive chunks longer

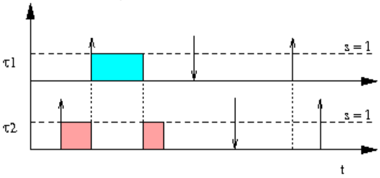
Open problem - DVFS approaches

Off-line: how can the speed be computed efficiently during PPP?

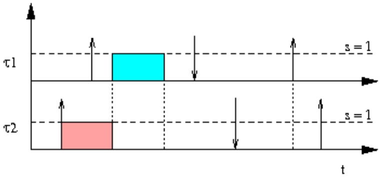
Online: what kind of algorithms can be used?

- A slower speed makes non-preemptive chunks longer

Fully-Preemptive



Limited-Preemptive



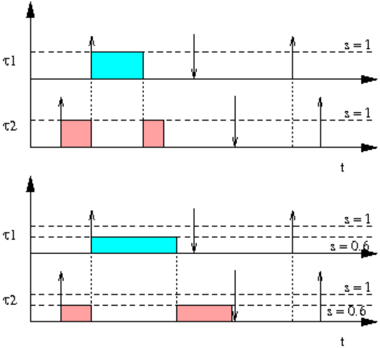
Open problem - DVFS approaches

Off-line: how can the speed be computed efficiently during PPP?

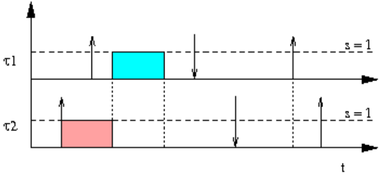
Online: what kind of algorithms can be used?

- A slower speed makes non-preemptive chunks longer

Fully-Preemptive



Limited-Preemptive



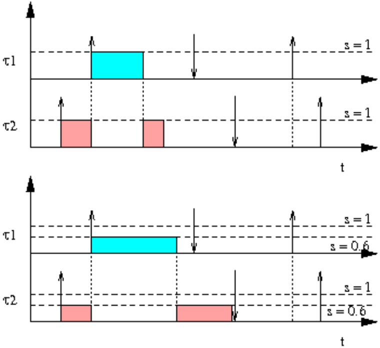
Open problem - DVFS approaches

Off-line: how can the speed be computed efficiently during PPP?

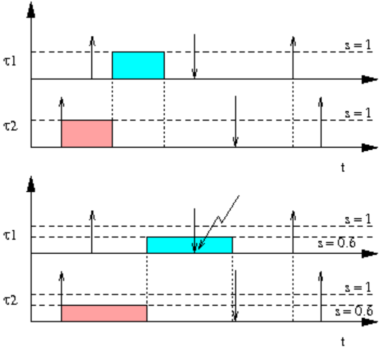
Online: what kind of algorithms can be used?

- A slower speed makes non-preemptive chunks longer

Fully-Preemptive

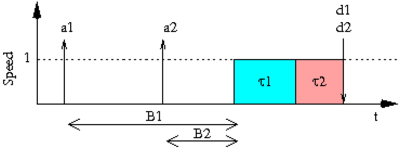


Limited-Preemptive



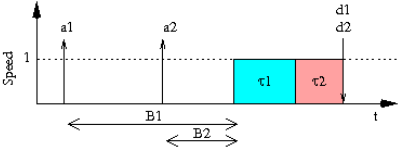
Open problem - DPM approaches

Task procrastination algorithms: forcing inactivity period blocks the execution of the tasks and so, reduces their blocking tolerance



Open problem - DPM approaches

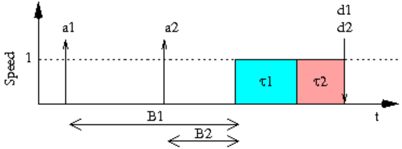
Task procrastination algorithms: forcing inactivity period blocks the execution of the tasks and so, reduces their blocking tolerance



Can algorithms for the fully-preemptive model be used without any deadline miss? If so, is any improvement guaranteed?

Open problem - DPM approaches

Task procrastination algorithms: forcing inactivity period blocks the execution of the tasks and so, reduces their blocking tolerance



Can algorithms for the fully-preemptive model be used without any deadline miss? If so, is any improvement guaranteed?

How is it possible to exploit Limited-Preemptive characteristics for further collecting idle intervals and compacting task executions?

Retis
Real-Time Systems Laboratory

thank you

m.bambagini@sssup.it