Optimal and Adaptive Multiprocessor Real-Time Scheduling: The Quasi-Partitioning Approach

Ernesto Massa [1] George Lima [2] Paul Regnier [2]
Greg Levin [3] Scott Brandt [3]

[1] State University of Bahia - Brazil

[2] Federal University of Bahia - Brazil

[3] University of California - USA

july,2014

Problem, Model and Notation

We address the problem of schedule a set of tasks in a multiprocessor environment

- tasks are sporadic
- deadline is implicit (deadline = minimum inter-arrival interval)
- a task instance has a worst case execution time (wcet)
- $R(\tau) = \text{wcet}/\text{period}$
- a task $\tau{:}(4, 8)$ has wcet $= 4$, deadline $= 8$ and $R(\tau) = 0.5$

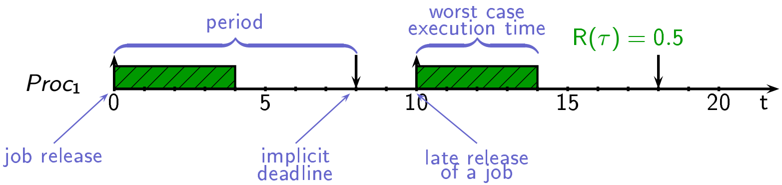

Figure 1: Real-Time System with tasks $\tau{:}(4, 8)$

Wonderful World

Assigning tasks to processors (First Fit Decreasing of Rate)

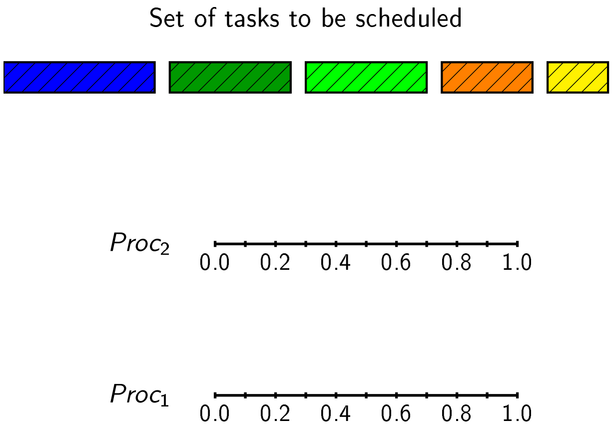

Figure 2: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

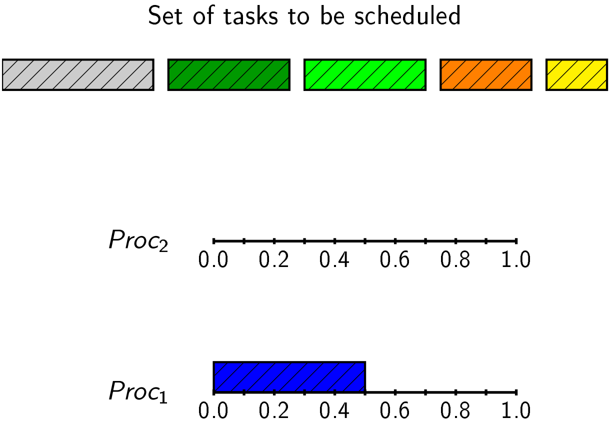

Figure 3: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

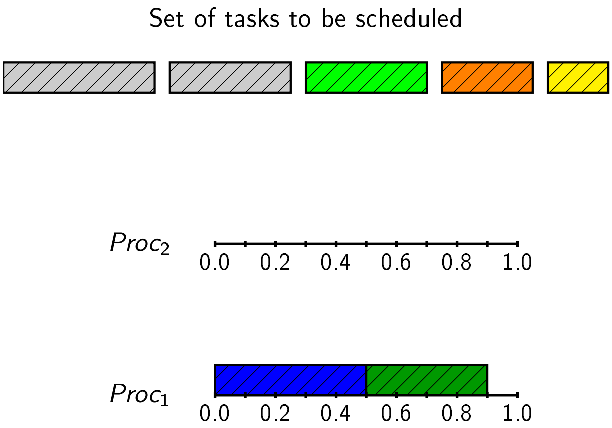

Figure 4: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

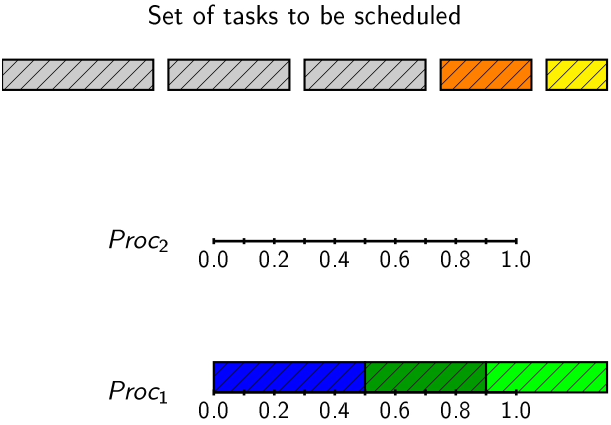

Figure 5: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

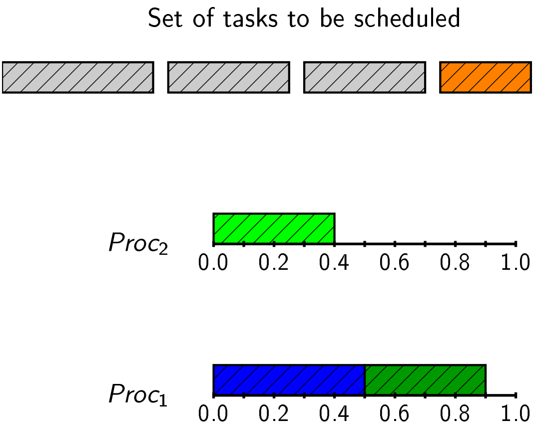

Figure 6: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

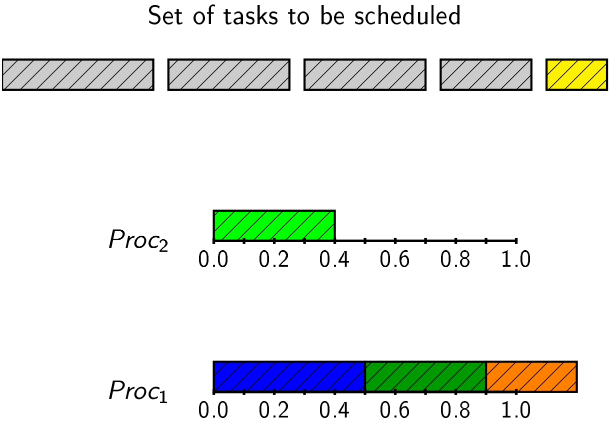

Figure 7: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

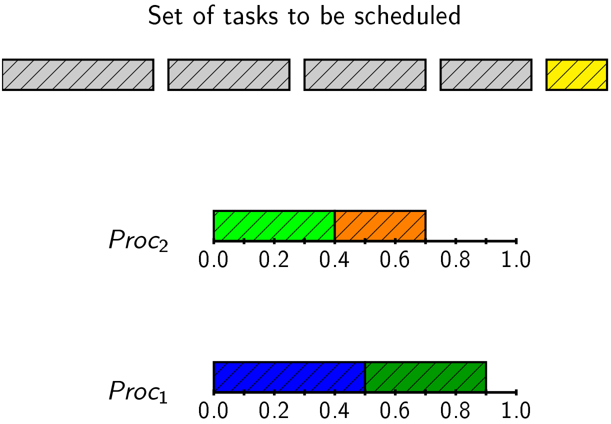

Figure 8: QPS example in two processors

Wonderful World

Assigning tasks to processors (First Fit Decreasing of task rate)

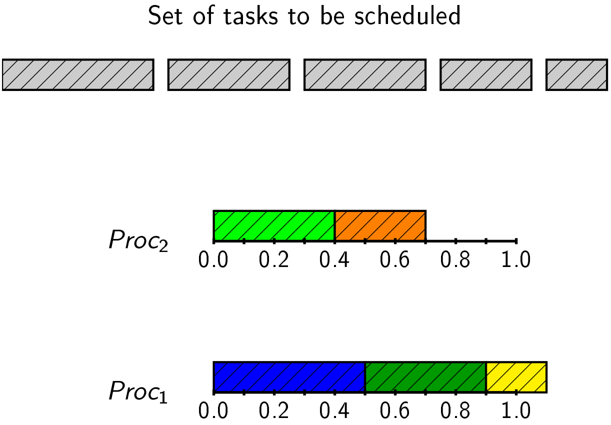

Figure 9: QPS example in two processors

Wonderful World (partitioned systems)

Assigning tasks to processors (First Fit Decreasing of task rate)

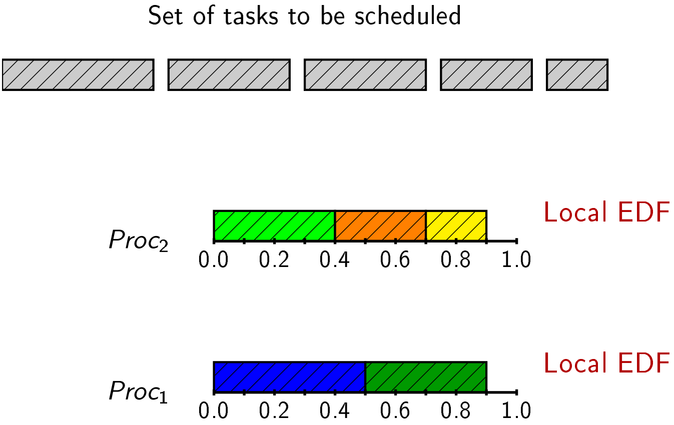

Figure 10: QPS example in two processors

But..

Assigning tasks to processors (First Fit Decreasing of task rate)

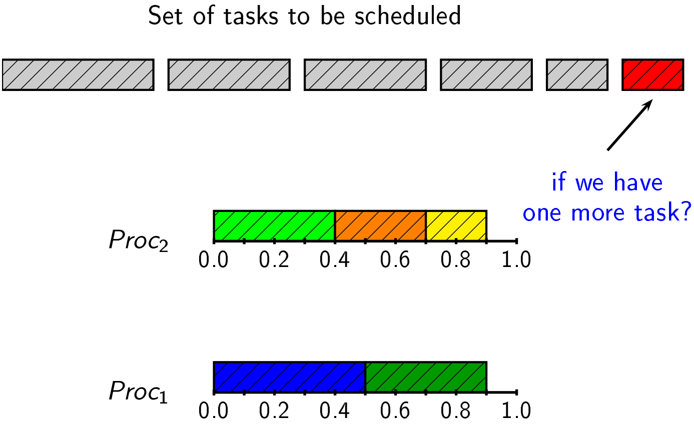

Figure 11: QPS example in two processors

But..

Assigning tasks to processors (First Fit Decreasing of task rate)

Set of tasks to be scheduled

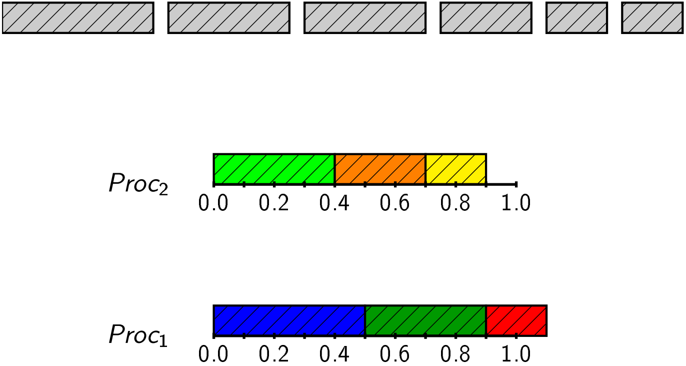

Figure 12: QPS example in two processors

But..

Assigning tasks to processors (First Fit Decreasing of task rate)

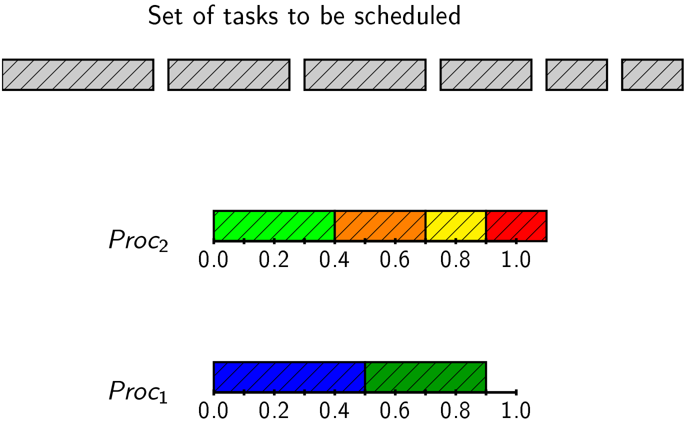

Figure 13: QPS example in two processors

Quasi-Partitioning

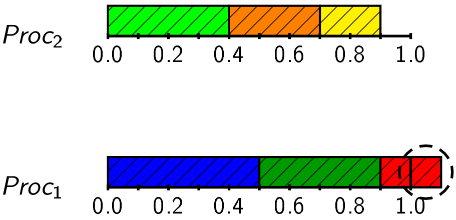

Figure 14: QPS example in two processors

Quasi-Partitioning Scheduling

Dealing with a group of tasks with rate greater than 1

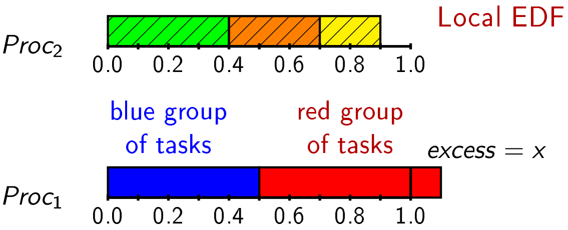

Figure 15: QPS example in two processors

Quasi-Partitioning Scheduling

Dealing with a group of tasks with rate greater than 1

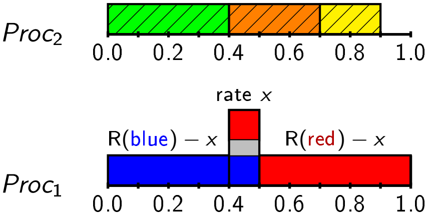

Figure 16: QPS example in two processors

Quasi-Partitioning Scheduling

Master, Slave and Dedicated Servers

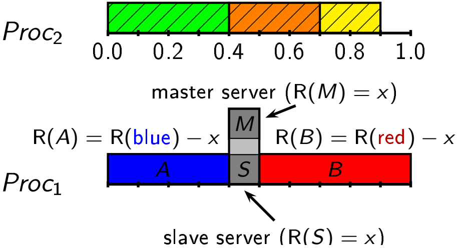

Figure 17: QPS example in two processors

Quasi-Partitioning Scheduling

Processor Hierarchy

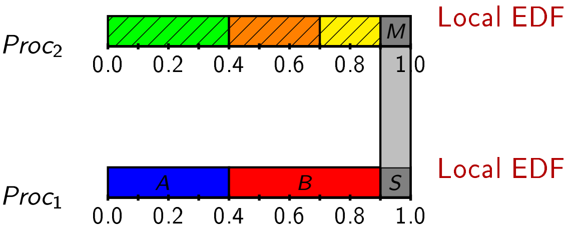

Figure 18: QPS example in two processors

Overview of Fixed-Rate Server

σ_1:$(R(\sigma_1) = 0.8, \text{cli} = \{\tau_1, \tau_2\})$
τ_1:$(2.4, 6) \Rightarrow R(\tau_1) = 0.4$
τ_2:$(4, 10) \Rightarrow R(\tau_2) = 0.4$

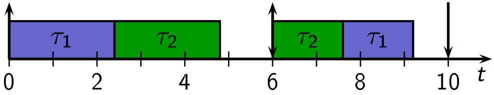

Figure 19: Tasks τ_1:$(2.4, 6)$, τ_2:$(4, 10)$ been served by fixed-rate server σ_1:$(0.8, \{\tau_1, \tau_2\})$.

Adaptation Strategy

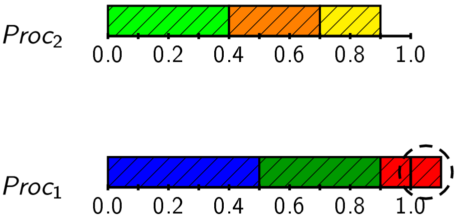

Figure 20: QPS example in two processors

Adaptation Strategy

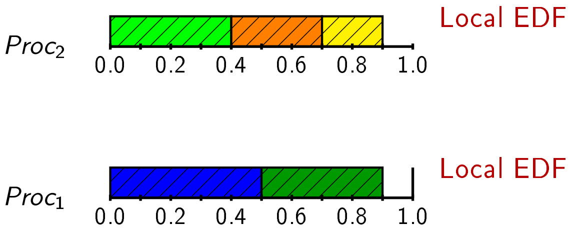

Figure 21: QPS example in two processors

Related Work (Optimal Algorithms)

Some techniques divide time into windows and execute tasks proportionally into each window (e.g. DPW,EKG,PFair), solving theoretically this problem, but imposing a large number of preemptions.

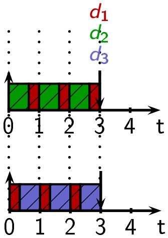

Figure 22: τ_1:(2, 3), τ_2:(2, 3) and τ_3:(2, 3) executing in windows

Other (RUN, U-EDF) use different approaches with lower number of preemptions.

Evaluation

QPS Evaluation

- Synthetic task sets generated according to Emberson Algorithm [1];
- Sporadic and periodic systems are considered;
- QPS is compared against DPW, EKG, RUN and U-EDF
- each simulation took into consideration $1,000$ task sets and run for $1,000$ time units.

Sporadic Systems

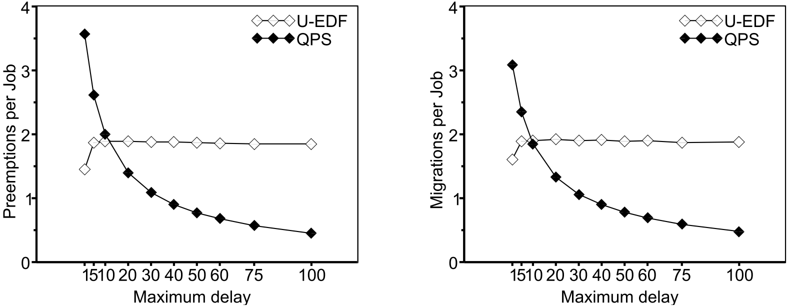

Figure 23: Average number of preemptions and migrations for systems with 16 sporadic tasks scheduled on 8 processors.

Periodic Systems

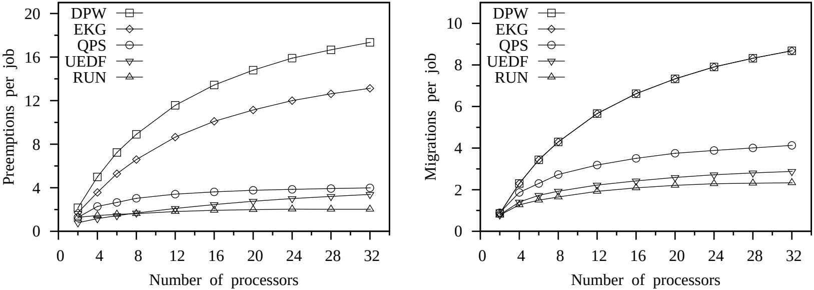

Figure 24: Average number of preemptions and migrations for periodic systems with $2m$ tasks that fully utilize m processors.

Periodic Systems

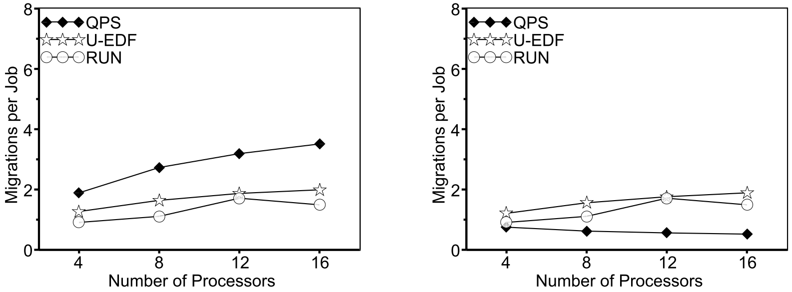

Figure 25: Average number of migrations for periodic systems with $2m$ tasks that utilize 100% and 98% of m processors.

Conclusions

- For sporadic task systems: QPS can take advantage of late tasks in the system and execute as a partitioned approach.
- For periodic task systems: QPS has better results, when the total system utilization is not greater than 98%.
- QPS is the first algorithm which goes from partitioned to global scheduling and vice-versa as a function of system load.

Thank you!

References: I

[1] P. Emberson, R. Stafford, and R. I. Davis.
Techniques for the synthesis of multiprocessor tasksets.
In *Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)*, pages 6–11, 2010.