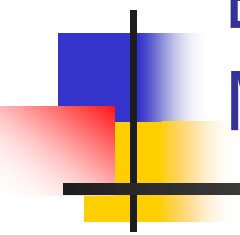


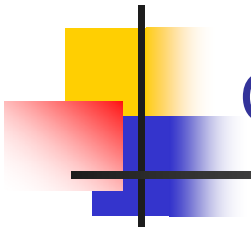
Energy-Aware Task Allocation for Rate Monotonic Scheduling

Tarek A. AlEnawy and Hakan Aydin

Prepared by:

Samuel Bushra

Introduction to Real-Time Systems

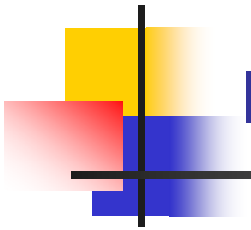


Outline

- Main idea
- Introduction
- System model and assumptions
  - Power and energy consumption model
  - Task and scheduling model
- Energy minimization procedure
- Framework
  - RMS Admission Control Algorithm
  - Partitioning Heuristics
  - Speed Assignment Schemes
- Experimental Results
- Conclusion

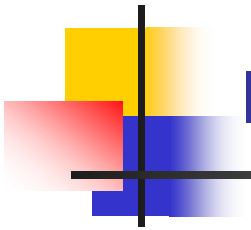
Main idea

- Problem of energy minimization
  - Periodic preemptive hard real-time tasks
  - Identical multiprocessor platform with dynamic voltage scaling capability
- Adopt partitioned scheduling and assume tasks are assigned rate-monotonic priorities.
- Propose integrated approach to the problem
  - RMS admission control test
  - The partitioning heuristic
  - The speed assignment algorithm
- Investigate impact of heuristics
  - Feasibility
  - Energy
  - Feasibility/Energy



Introduction

- Multiprocessor platform with DVS (Dynamic Voltage Scaling) capability. DVS reduces the system energy consumption by reducing the CPU supply voltage and the clock frequency simultaneously.
- 2 main approaches for multiprocessor RT scheduling:
  - Partitioned scheduling: admission control module permanently assigns each task to a processor. Each processor has its own ready queue and scheduler.
  - Global scheduling: scheduler selects from a single ready queue for executing the highest-priority ‘n’ tasks on ‘m’ processors. Tasks are allowed to migrate between processors.
- Partitioned and global scheduling are incomparable, in the sense that there are task sets that can be scheduled in a feasible manner by only one approach.
- Task allocation can have a significant impact on the overall energy consumption of the system.



Introduction

- Main contributions of this paper:
  - Problem of energy minimization in multiprocessor periodic RT scheduling with partitioning, and show that problem remains NP-Hard in the strong sense.
  - Evaluate well-known partitioning heuristics, RMS admission control algorithms, and speed assignment schemes in terms of feasibility performance and overall energy consumption.
  - Evaluate the effects of off-line partitioning and on-line partitioning.

System model and assumption

Power and Energy Consumption Model

- Multiprocessor platform M with m processors M_1, M_2, M_3, …., M_m.
- Number of processors m is fixed during the operation.
- All processors are identical in terms of processing power and speed-energy characteristics.
- Denote the maximum speed level available in system by S_{max}. Speed values are normalized with respect to S_{max}. (S_{max} = 1.0)
- Each processor has dynamic voltage scaling (DVS) capability to adjust its operating speed S.
- g(S) denotes CPU power dissipation function when running at speed S. g(S) is strictly convex and increasing function.
- Total energy consumption of processor: $E(t_1, t_2) = \int_{t1}^{t2} g(S(t))\, dt$

System model and assumptions
Task and Scheduling Model

- Set of n independent periodic hard real-time tasks $T = \{T_1, T_2, \ldots, T_n\}$
- Each task $T_i = (C_i, P_i)$ characterized by two parameters:
  - C_i: worst-case number of processor cycles required by T_i and a period P_i.
  - P_i: the period, which is equal to the relative deadline D_i.
- Worst-case execution time of task T_i when running at speed S is given by $c_i = C_i / S$.
- Preemptive scheduling model
- Utilization of task T_i under CPU speed S is given by $u_i(S) = C_i / (P_i S)$
- Aggregate utilization of the task set (under maximum speed) is given by $U_{tot} = \sum_i C_i / P_i$
- Partitioning-based approach to multiprocessor scheduling. Tasks are assigned permanently to processors.

System model and assumptions

Task and Scheduling Model

- On each processor, Rate Monotonic Scheduling policy is adopted: tasks are assigned static priorities that are inversely proportional to their periods.

- Utilization factor, denoted by α, determines the largest utilization among all the tasks in the set. Formally, $\alpha = \max(u_i)$, for all $T_i \in T$.

- A necessary (but not sufficient) condition for feasibility on system of *m* identical multiprocessors is to have a task set whose total utilization does not exceed the computing capacity. ($U_{tot} <= m$)

Review

- Necessary condition: a condition which must hold for a result to be true, but which does not guarantee it to be true.
- Sufficient condition: a condition, which if true, guarantees that a result is also true.

Energy-minimization procedure

- Objective: Under energy-constrained settings, allocate tasks and compute CPU speed assignments while minimizing the total energy consumption and preserving the feasibility.

- RMS-ENERGY-PARTITION: Given a set T of periodic hard real-time tasks and a set M of identical processors, find a task-to-processor assignment (partition) and compute task level speeds on each processor such that:

  - Workload can be scheduled by RMS in a feasible manner.
  - The total energy consumption on all processors M_1, M_2, M_3, …., M_m is minimum.

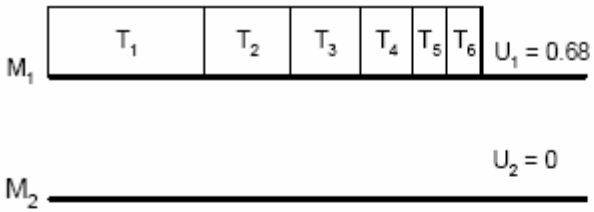
- Optimization problem where objective function is total energy consumption of M, subject to the constraint that the workload on each processor will be feasible when scheduled by rate-monotonic priorities.

Energy-minimization procedure

Motivational example

- Set of six periodic hard real-time tasks $T = \{T_1, T_2, \ldots, T_6\}$ to be scheduled using partitioning on two identical processors.
- Individual task utilizations $u_1 = 0.32$, $u_2 = 0.2$, $u_3 = 0.1$, $u_4 = 0.04$, $u_5 = 0.01$ and $u_6 = 0.01$.
- Example depicts 3 different partitions and the energy consumption patterns under RMS and EDF policies.
- Total utilization $U_{tot} = 0.68$ is less than the asymptotic Liu-Layland bound *ln 2* (0.693), any partitioning of these tasks is feasible with both EDF and RMS.

Energy-minimization procedure

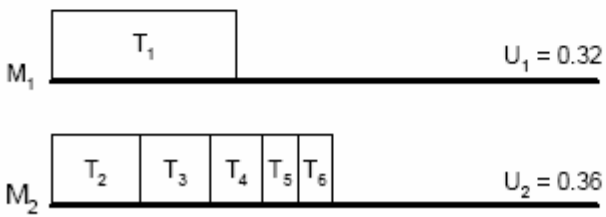


	RMS	EDF
S_1	0.925	0.68
S_2	Idle	Idle
E_{tot}	5818	3144

Figure 1. A feasible partition (Partition 1)

	RMS	EDF
S_1	0.436	0.34
S_2	0.436	0.34
E_{tot}	1295	786

Figure 2. A feasible and perfectly balanced partition (Partition 2)



	RMS	EDF
S_1	0.32	0.32
S_2	0.484	0.36
E_{tot}	1171	794

Figure 3. A feasible and slightly unbalanced partition (Partition 3)

Energy-minimization procedure

Analysis

- Partitioning shown in figure 1 corresponds to the schedule, where as many tasks as possible are packed on one processor while keeping other processors idle to accommodate tasks yet to be assigned.
- Figure 2 shows a perfectly balanced partition.
- Figure 3 shows the case where first processor is exclusively dedicated to task with largest utilization, while all other tasks are assigned to the second processor.

- $E(M_i) = P * U_i * g(S_i) / S_i$
  - U_i is total utilization of tasks assigned to M_i under CPU speed $S_i = 1.0$.
  - $P = lcm(P_1, P_2, \ldots, P_6)$. Take $P = 10000$ for illustration purposes.
  - $g(S_i)$ denotes CPU power dissipation function. $g(S) = S^3$.
  - $E(M_i)$ is energy consumption of processor when running at constant speed S_i during the interval $[0,P]$.

- For EDF, speed assignment scheme is optimal: $S_i = U_i$.
- For RMS, uniform slow-down approach: speed of processor with n_i tasks is $U_i / U_{bound}(n_i)$.
  $U_{bound}(k) = k * (2^{1/k} - 1)$ (Liu-Layland schedulability bound)

Energy-minimization procedure

Observations

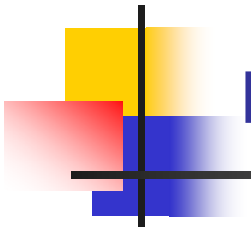
- Partition 1 has maximum energy among all three partitions considered for both EDF and RMS.
- Under EDF, Partition 2 has minimum energy consumption among all three partitions.
- Partition 3 has about 10% lower energy consumption than Partition 2.

- Although there is minor difference in processor utilizations, there is significant difference in number of tasks for each processor.

- **Feasible partitions can have significantly different energy characteristics.**

- Load balancing helps to reduce energy consumption, but is not necessarily the most energy-efficient option.

- Proposition: RMS-ENERGY-PARTITION is NP-Hard in the strong sense even when the feasibility is guaranteed a priori.



Framework

- What RMS admission control algorithm to use on each processor?
  - Uniprocessor RMS admission control algorithm adopted.
  - Two classes of algorithms:
    - Those using utilization and period information when making a decision.
    - Those based on *Time Demand Analysis*.
- What partitioning heuristic to use?
  - An efficient partitioning heuristic determines the processor to which the task will be assigned.
- What speed assignment scheme to adopt?
  - This step involves the computation of the CPU speed. The schedule must remain feasible even with the reduced speed.
- Provide a computational cost and performance benefit analysis of different schemes in terms of both feasibility and overall energy consumption.

Framework
RMS Admission Control Algorithms

- Selection of algorithm affects the feasibility performance of system.

Utilization-based feasibility tests

- Polynomial-time, yet approximate tests using information about utilization of task set.
- Sufficient condition: a task set is deemed feasible if the condition is not violated.

- Basic utilization-based tests: require only information about task utilizations.
  - Exact Liu-Layland test (ELL): a task set with n tasks is schedulable on one processor if $U_{bound}(n) = n * (2^{1/n} - 1)$
  - Hyperbolic test (HYP): provides tighter bounds than ELL by considering individual task utilizations:

$$\prod_{i=1}^{n}(1+u_i) <= 2$$

- Utilization-based tests exploiting period information: Liu and Layland observed that schedulability bound of RMS was 100% for harmonic task sets.
  - Burchard test: utilization bound is function of number of tasks and how close tasks are to being harmonic.
  - R-bound test: transforms given task set into one where ratio r of maximum period to minimum period does not exceed 2. $U_{bound}(n,r) = (n - 1)(r^{1/(n-1)} - 1) + 2/r - 1$

Framework
RMS Admission Control Algorithms

Time-demand-analysis-based tests

- Computationally more complex, but more accurate tests using information about *worst-case task execution times* and *periods*.
  - Time demand analysis (TDA): provides necessary and sufficient condition, runs in pseudo-polynomial time. The *time demand* function of task T_i :

$$w_i(t) = c_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{P_k} \right\rceil * c_k, \; 0 < t <= P_i$$

Task set is feasible if all tasks meet their deadlines under critical-instant phasing: for each task T_i, it is possible to find a time instant *t* where $w_i(t) <= t <= D_i$.

  - Pillai-Shin test: provides sufficient but not necessary condition for feasibility by checking whether time demand function $w_i(t)$ does not exceed relative deadline D_i at the task's period boundary $t = D_i = P_i$ for each task:

$$w_i(t = P_i) = \sum_{k=1}^{i} \left\lceil \frac{P_i}{P_k} \right\rceil * c_k <= P_i, \; \forall \, T_i \in T$$

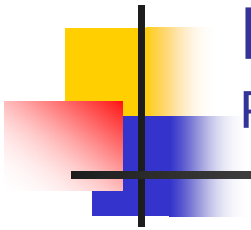
Framework

RMS Admission Control Algorithms

- All RMS feasibility tests, except for TDA, are inexact in the sense that they provide a sufficient but not necessary condition for feasibility.

Test Name	U_{tot}	u_i	P_i	c_i	Complexity	Exact
ELL	√	×	×	×	$O(n)$	×
Hyp	×	√	×	×	$O(n)$	×
R-Bound	√	×	√	√	$O(n)$	×
Burc	√	×	√	×	$O(n)$	×
PS	×	×	√	√	$O(n^2)$	×
TDA[3]	×	×	√	√	$O(n^2r)$	√

Table 1. RMS admission control algorithms used in this paper



Framework

Partitioning heuristics

Second part involves the selection of the partitioning scheme.

- Worst-Fit (WF) and Next-Fit (NF) tend to distribute the workload evenly among the available processors.
- First-Fit (FF) and Best-Fit (BF) attempt to greedily pack as many tasks as possible on one processor while keeping the other processors idle for tasks yet to be assigned.
- It is known that FF and BF tend to outperform NF and WF from the feasibility point of view.

Framework
Speed Assignment schemes

- Once feasible partition obtained with partitioning heuristic and admission control algorithm, perform speed assignment to minimize energy consumption on each processor.

Uniform slow-down technique

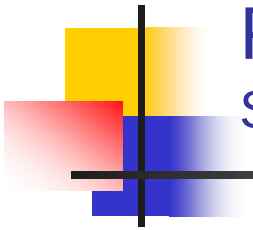
- Technique that computes unique speed across all tasks such that new effective utilization does not exceed utilization bound suggested by admission control algorithm.

  Example: $U_i / U_{bound}(n_i)$, where $U_{bound}(k) = k * (2^{1/k} - 1)$ (Liu-Layland schedulability bound)

Time-demand-analysis-based speed assignment techniques

- Pillai-Shin: single speed chosen for all tasks running on a given processor. A tentative target speed α_i is determined using the following equality:

$$\sum_{k=1}^{i} \left\lceil \frac{P_i}{P_k} \right\rceil * c_k = \alpha_i * P_i, \forall T_i \in T$$



Framework

Speed Assignment schemes

- Sys-Clock (SysC): chooses a single speed for all tasks running on same processor, and is optimal for fixed priority preemptive scheduling policies that use single speed.
- PM-Clock (PMC): allows different tasks running on same processor to have different speeds.

Both schemes use the idle time in a given schedule to reduce execution speed, and hence decrease the energy consumption, while maintaining the feasibility.

Experimental Results

Performance metrics

Given task set to be scheduled on multiprocessor platform, goal of algorithm selection:

- high feasibility performance
- low energy consumption
- low computational cost

- Feasibility metric (FH): percentage of task sets that are feasibly scheduled by heuristic *H* out of the total number of task sets generated during experiments.
- Energy consumption metric (ECH): average energy consumption for each task set scheduled by *H* in a feasible manner.
- Feasibility/energy metric (FEH): favors heuristics with high feasibility and low energy consumption.

Experimental Results

Simulation settings

- Function of two task set parameters: U_{tot} (total task set utilization) and α (task utilization factor)
- For a fixed number of processors *m*, U_{tot} is varied between m / 10 and m.
- $U_{tot} / n <= \alpha <= 1.0$
- Case of $\alpha = 1.0$ corresponds to having no constraint on upper bounds on individual task utilization.
- For each data point, 1000 task sets generated by varying task periods P_i and utilizations u_i.
- Each task has a uniform probability of having short (1-10ms), medium (10-100ms), or long (100-1000ms) period.
- Task periods are uniformly distributed in each range.
- Task utilizations u_i are generated uniformly in the interval $[0.001,\alpha]$.
- A task's worst-case execution time at maximum speed is then determined as $c_i = P_i * u_i$.
- 8 processors and 80 tasks are used.

Experimental Results

Results for off-line partitioning

- Off-line partitioning: all task characteristics are known in advance.

- Greedy behavior of FF and BF improves the feasibility, a result from the study of bin-packing problem.
- In balanced partition, load balancing tends to reduce energy consumption in general, but does not lead to minimum energy when RMS is used.
- In un-balanced partition, heavily loaded processors have to run at speed close to maximum to guarantee feasibility of task set, which increases total energy of system.

- FF and BF are identical in terms of feasibility and energy performance. Since BF has higher computational complexity than FF, it is not included in discussion.

Experimental Results

Results for off-line partitioning

Effect of partitioning schemes

- Figures 4-5 use Exact Liu-Layland test (ELL) schedulability bound and uniform slow-down technique.
- There is trade-off between feasibility and energy performances. It is not possible to determine best performing scheme just by considering feasibility and energy separately.
- For off-line partitioning, the best partitioning heuristic is Worst-Fit, followed by Next-Fit, then First-Fit.

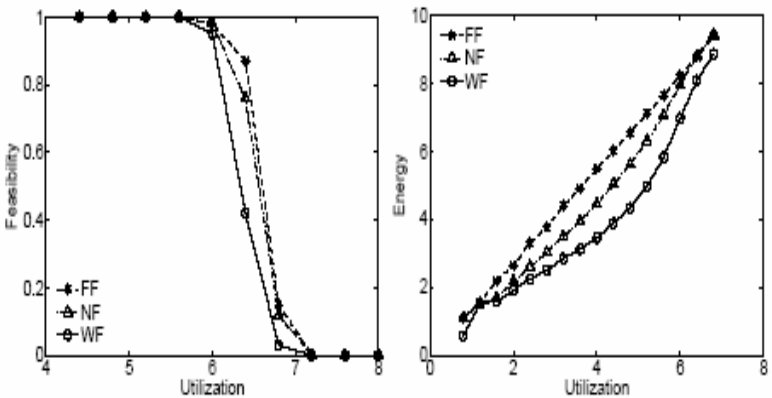


Figure 4. Feasibility and energy performance of offline partitioning heuristics using ELL for $\alpha = 1.0$

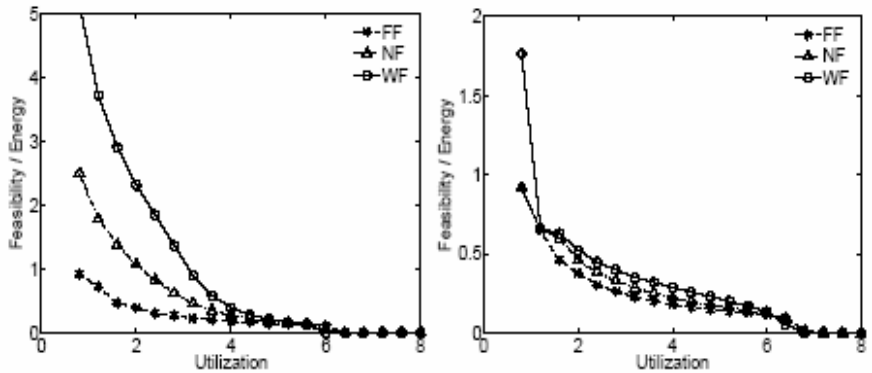


Figure 5. Feasibility/energy performance of offline partitioning heuristics using ELL: $\alpha = 0.5$ (left) and $\alpha = 1.0$ (right)

Experimental Results

Results for off-line partitioning

Effect of admission control and speed assignment schemes

- ELL, HYP, BURC, and R-BOUND use uniform-slowdown speed assignment technique.
- TDA uses the Sys-Clock speed assignment algorithm.
- The Worst-Fit partitioning heuristic is assumed to be used since it outperforms the other heuristics in terms of feasibility and energy consumption combined.

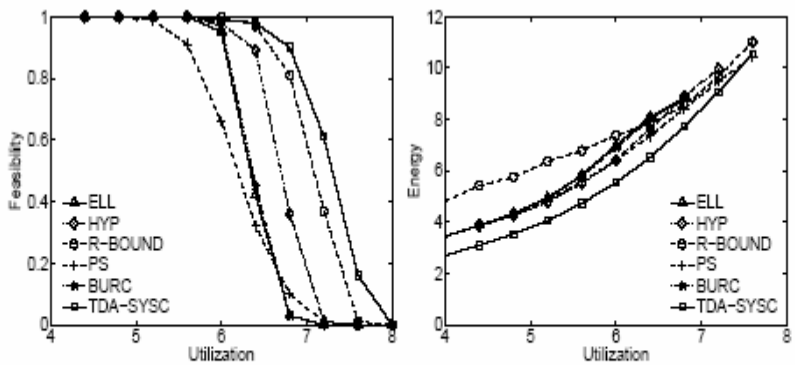


Figure 6. Feasibility and energy performance of the different techniques (off-line partitioning, $\alpha = 1.0$)

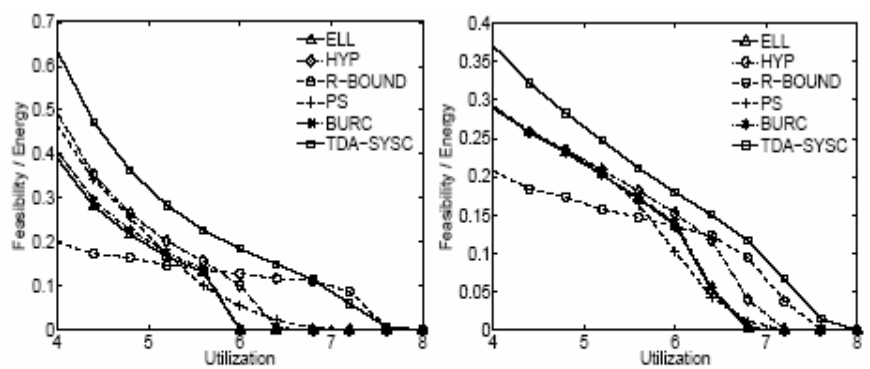


Figure 7. Feasibility/energy performance of the different techniques (off-line partitioning, $\alpha = 0.5$ (left) and $\alpha = 1.0$ (right))

Experimental Results

Results for on-line partitioning

- On-line partitioning: the task parameters are not known in advance.
- FF, BF and NF have good feasibility performance, but have poor energy and feasibility/energy performance, particularly at low utilization values.
- WF has good energy performance, but it has very low feasibility and feasibility/energy performance, especially at medium to high utilization. WF has poor performance since tasks with small utilization are distributed to multiple processors, preventing allocation of subsequent task with large utilization to separate processor.
- RESERVATION: algorithm reserves a pool of *k* processors (k <= m) for light tasks and the remaining m – k processors for heavy tasks. (light task means that its utilization does not exceed average utilization per processor)
- RESERVATION(k) maintains balance between good feasibility performance of FF/BF and the good energy performance of WF.

Experimental Results

Results for on-line partitioning

Effect of partitioning schemes

- ELL schedulability bound assumed for the results.
- RSVR2 has consistently good overall performance and is comparable to best scheme under each condition.
- For α = 0.5, RSVR4 provides best overall performance.

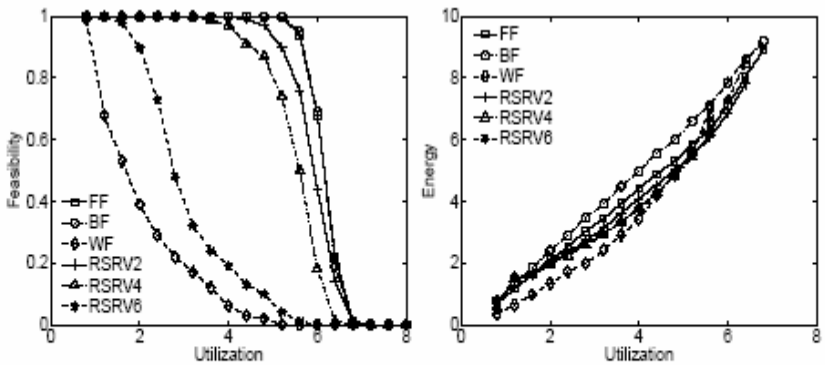


Figure 8. Feasibility and energy performance of on-line partitioning heuristics using ELL for $\alpha = 1.0$

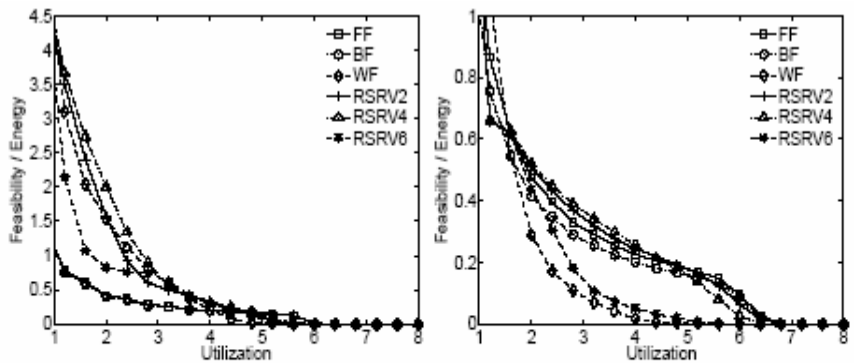


Figure 9. Feasibility/energy performance of on-line partitioning heuristics using ELL: $\alpha = 0.5$ (left) and $\alpha = 1.0$ (right)

Experimental Results

Results for on-line partitioning

Effect of admission control and speed assignment scheme

- RESERVATION(k=2) scheme used based on consistently good performance at different load conditions.
- R-BOUND not included since it mandates knowledge of the periods and task pre-ordering.
- In terms of hybrid performance metric, TDA-SYSC, with its sophisticated mechanism is the clear winner throughout the utilization.

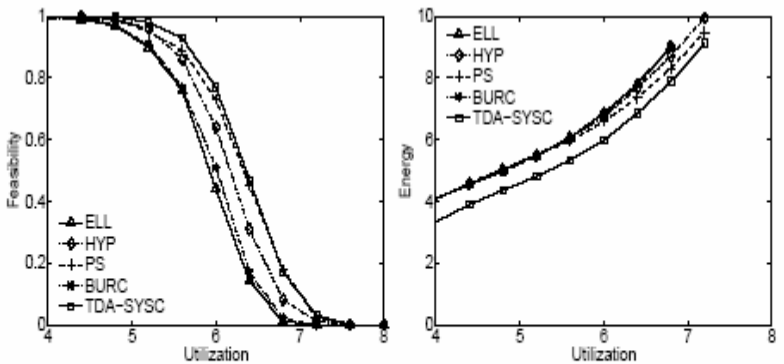


Figure 10. Feasibility and energy performance of different techniques (on-line partitioning, $\alpha = 1.0$)

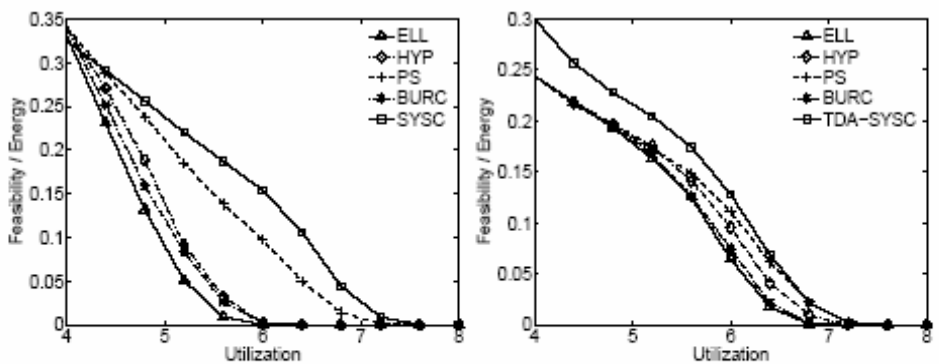


Figure 11. Feasibility/energy performance of different techniques (on-line partitioning, $\alpha = 0.5$ (left), $\alpha = 1.0$ (right))

Experimental Results

Effect of discrete speed levels

- In presence of finite number of discrete speeds, one selects a higher CPU speed level, leading to an increase in total energy consumption. This results in overall decrease of feasibility/energy performance.
- Relative performance of different feasibility tests unchanged compared to continuous speed case.

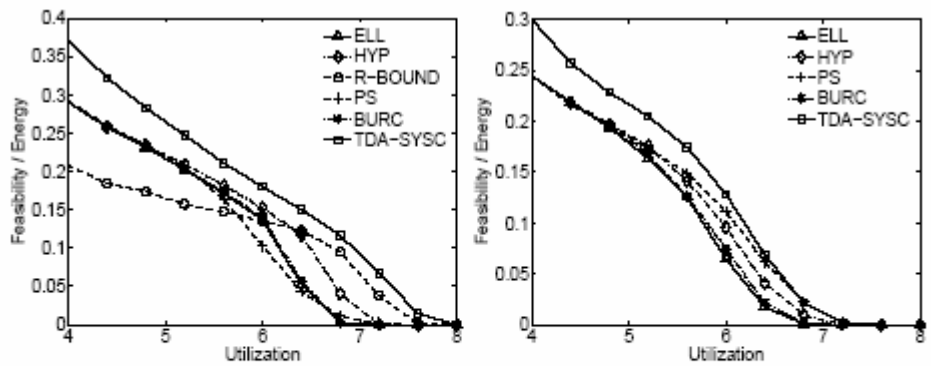
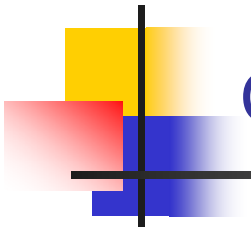


Figure 12. Feasibility/energy performance of the different techniques for α = 1.0 and Intel XScale specifications (off-line case (left) and on-line case (right)).



Conclusion

- First research effort addressing energy-aware scheduling of static-priority periodic RT task sets on multiprocessors with partitioned approach.
- Time Demand Analysis combined with Sys-Clock speed assignment has best overall performance in off-line and on-line settings.
- In off-line settings, Worst-Fit has best overall performance among partitioning heuristics.
- Hyperbolic test combined with uniform slow-down approach has best overall performance among polynomial-time schemes.
- In on-line settings, performance of Worst-Fit deteriorates significantly. However, RESERVATION(k) scheme exhibits a competitive overall performance.