An Integrated Approach for Applying Dynamic Voltage Scaling to Hard Real-Time Systems

-Yanbin liu and Aloysius Mok

-Presented by

-Manoj Sivakumar

Outline

- Motivation
- Dynamic Voltage Scaling
- Proposed Method
- Performance Evaluation
- Conclusions

Motivation

- Wireless and portable devices – limited power supply
- Maximize utilization while minimizing power used
- For Real-time systems – use as low power as necessary to complete all tasks before deadline

Dynamic Voltage Scaling

- DVS increases batter life by reducing the power consumption of the processor.
- Power consumed $P = kCV^2 f$
  - C – capacitance
  - V – Voltage
  - f – clock frequency
- But changing voltage supplied affects the clock frequency f

Dynamic Voltage Scaling

- Relation between voltage and frequency

$$f \propto (V - V_t)^2 / V$$

- Where Vt – threshold voltage
- Changing f will increase process completion times
- Hence a mechanism is necessary to determine what voltage should be supplied so that all the jobs complete on schedule

Proposed Method

- Proposed method tries to come up with an integrated approach for applying DVS to hard real-time systems that will be independent of the scheduling policy.
- Also after finding an offline schedule an online reclaim policy that readjusts the speed dynamically if job finishes in better than worst case time has been proposed.

System Model

- Periodic Tasks
- Preemptible
- Mutually independent
- Task Set T = {T1, T2 … Tn}
- Each task ‘i’ has a period p_i and a worst case execution time c_i
- All tasks start at time 0.

System Model

- H – Hyper period which is the LCM of all p_i
- Single Processor
- Normalized processor speed – 0 to 1

Definitions

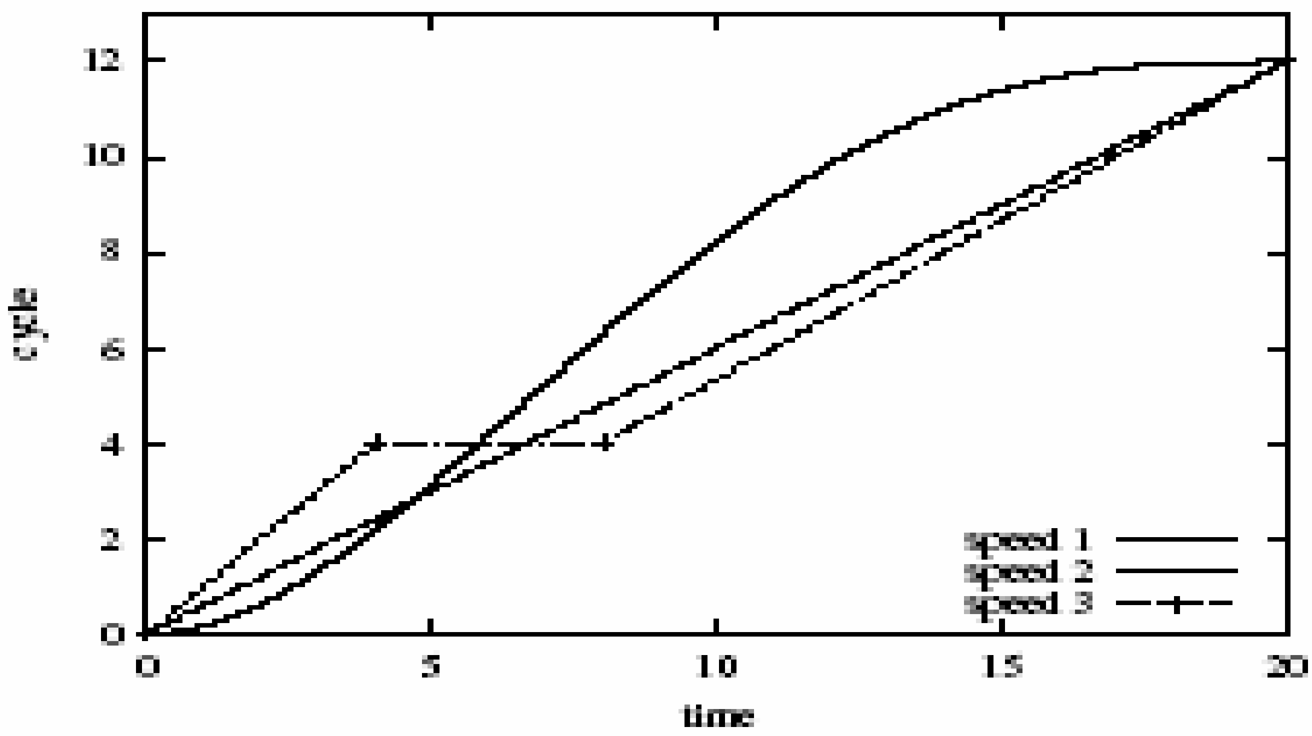
- Speed Function
- Available Cycle Function (ACF)
- Required Cycle Function (RCF)

Speed Function

- Speed Function s(t)
  - S(t) is the CPU speed, in cycles per time unit, at time t
  - Hence 0 <= S(t) <= 1
  - The cycles supplied by a processor during time period (t1, t2] is

$$\int_{t_1}^{t_2} S(t)dt$$

Speed Function



Speed Function

- The energy consume by a speed function is given as

$$\int_{t_1}^{t_2} P(S(t))dt.$$

- The objective of the optimization function would be to minimize the speed function S(t) in the interval (0,H]

ACF

- ACF(t) is defined as the upper bound on the cycles available for execution up to time t

$$ACF(t) = \sum_{i=1}^{n} \left(\left\lceil \frac{t}{p_i} \right\rceil * c_i \right).$$

Lemma 1 *If a speed function S optimizes energy consumption, then* $\int_0^t S(x)dx \le ACF(t)$ *for all time t.*

Required Cycle Function

- Basic RCF (BRCF(t)) is the minimal number of cycles that must be executed up to time t

$$BRCF(t) = \sum_{t=1}^{n} \left(\left\lfloor \frac{t}{p_i} \right\rfloor * c_i \right).$$

- Note that ACF and BRCF are independent of the scheduling policy
- Given a scheduler RCF(t) defines the CPU cycles that must be supplied to the tasks up to time t so that no job misses its deadline

Properties

1. $0 \leq S(t) \leq 1$ for $0 \leq t \leq H$.
2. $ACF(t)$, $BRCF(t)$, and $RCF(t)$ are non-decreasing step functions of time t. A step point of ACF is at the available time of each job; a step point of $BRCF$ is at the deadline of each job.
3. $ACF(t) \geq RCF(t) \geq BRCF(t)$ for $0 \leq t \leq H$.
4. $ACF(H) = RCF(H) = BRCF(H)$.

Properties

5. At most $ACF(t)$ cycles can be executed up to time t. If $\int_0^t S(x)dx > ACF(t)$, there are $\int_0^t S(x)dx - ACF(t)$ idle cycles.
6. At least $BRCF(t)$ cycles should be executed up to time t. If $\int_0^t S(x)dx < BRCF(t)$, there are missed deadlines.
7. If $ACF(t) \geq \int_0^t S(x)dx \geq RCF(t)$ at all time t, no job misses its deadline.
8. If $ACF(t) = RCF(t)$ during a time period $(t_1, t_2]$, then $S(t) = 0$ during the period.

Energy Optimization

- Suppose ACF and RCF are given. Then the optimization problem is

$$\begin{aligned} &\textit{minimize}: && E(S) = \int_0^H P(S(x))dx \\ &\textit{subject to}: && 0 \le S(t) \le 1, \\ & && RCF(t) \le \int_0^t S(x)dx \le ACF(t). \end{aligned}$$

- Since P(S) is a convex function of S we have the following results

Energy Optimization

- E(S) is a convex function of S
- The optimal speed function is unique
- Optimal speed function is a piece-wise linear function that changes speed only at the time when ACF or RCF changes. Hence the optimal speed function will change only when ACF or RCF increases.

Optimal Algorithm

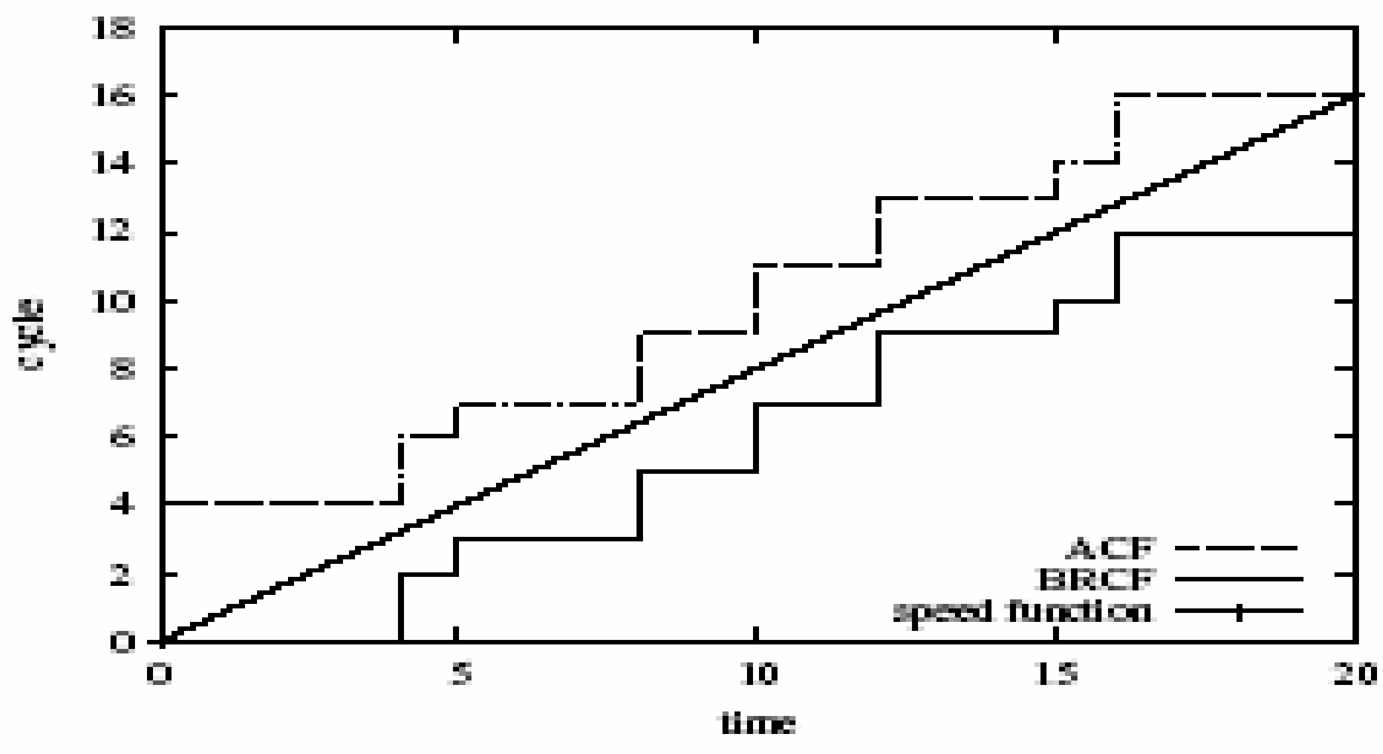
- Let a_0, …. a_m be the sorted sequences of time when ACF or RCF increases. Now the optimization can be transformed as

$$\begin{aligned}
&minimize: \quad E(S) = \sum_{j=1}^{m} P(S_j) * (a_j - a_{j-1}) \\
&subject\ to: \quad 0 \le S_j \le 1\ for\ 1 \le j \le m \\
&\quad RCF(a_k) \le \sum_{j=1}^{k} S_j * (a_j - a_{j-1})\ at\ deadline\ a_k, \\
&\quad ACF(a_l) \ge \sum_{j=1}^{l} S_j * (a_j - a_{j-1})\ at\ avail.\ time\ a_l.
\end{aligned}$$

Algorithm

- The algorithm basically tries to extend a straight line as long as possible from the coordinate (0,0) to (H,AFC(H)) that lies within the region delineated by ACF and RCF

Algorithm



18
16
14
12
10
8
6
4
2
0
cycle
0
5
10
15
20
time
ACF
BRCF
speed function

Online Reclaim

- The off-line algorithm is based on worst case budget. Hence at run time we might have slack cycles
- Solution – if a job finishes with ‘d’ cycles remaining then it can be interpreted as if the speed function increase by d and we can find the new optimal solution.

Online Reclaim

- FC – accumulated cycles up to t
- SC – scheduled cycles up to t
- FC – SC = slack cycles
- IF FC > SC
  - Reduce CPU speed
- IF FC = SC
  - Maintain same speed
- Note FC cannot be less than SC

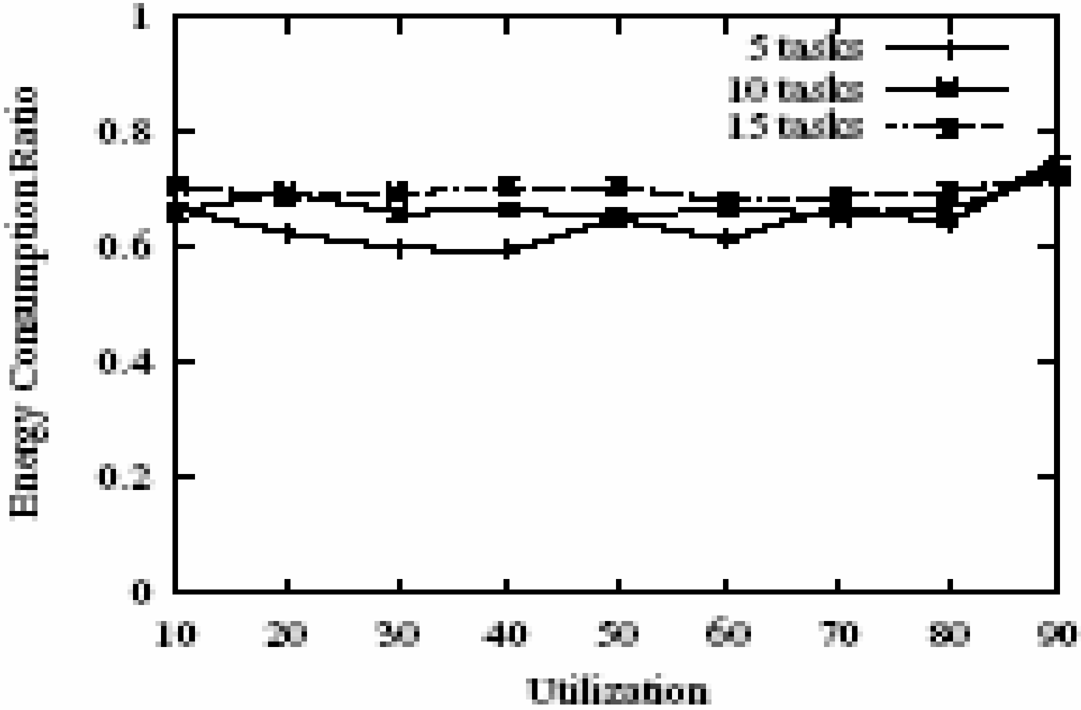
Online Reclaim

- To find new speed
  - Lookup ACF step points
  - The optimal solution is when we look at all ACF step points till H
  - Speed Vs Efficiency tradeoff
- Experiments were performed up to 3 lookup points

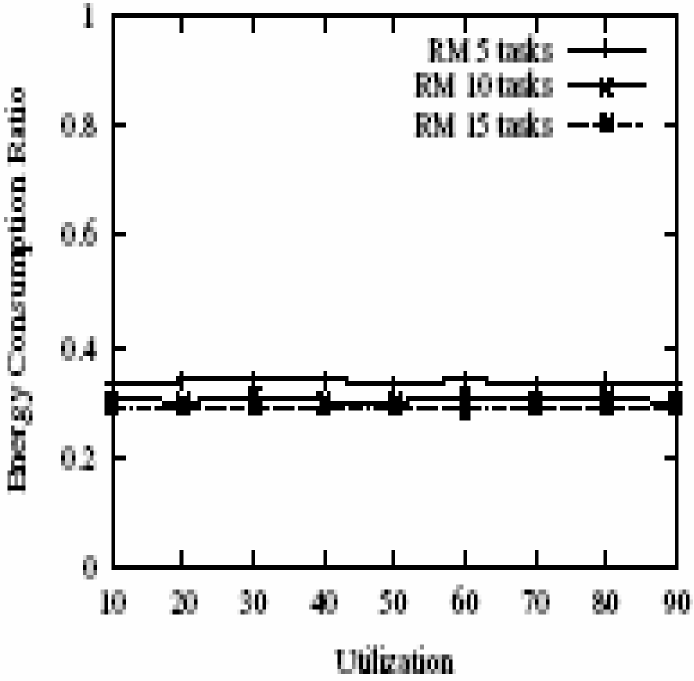
Performance Evaluation

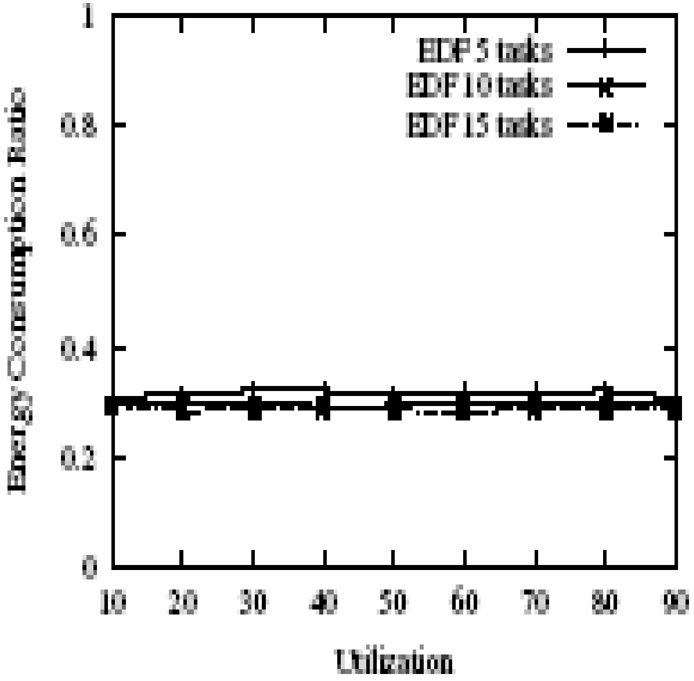
- Simulations were performed
- WCET - Worst case execution time
- BCET - Best case execution time
- BCET defined to be between 10 % to 100 % of WCET

Energy Consumption Ratio



Performance with Dynamic Reclaim





Conclusions

- An integrated solution independent of scheduling policy was proposed
- The algorithm also dynamically adjusts at run time
- Energy Savings of up to 40 % achieved without dynamic reclaim.