A Survey of Energy-Aware Real Time Scheduling Tools

Idawaty Ahmad
University Putra Malaysia
Faculty of Computer Science and Information Technology, University Putra Malaysia UPM Serdang
43400 Selangor
idawaty@upm.edu.my

ABSTRACT

This paper gives a brief survey on the existing simulation tools that are available to further investigate the performances of energy-aware real time scheduling algorithms. The previous work from Y. Chandarli et.al, suggested four properties for a real time scheduling tools to become a reference tool by the real time scheduling community. This paper compares the properties of the most recent real time scheduling tools considering to the above mentioned properties from simulation software perspective. Programming language is one of the important properties that influence the popularity of these tools. This criteria is compared between these tools. Java language is observed to dominate the development of these simulation tools.

KEYWORDS

Real time system, energy-aware real time scheduling, simulation tools, DVFS, DPM.

1 INTRODUCTION

Energy consumption is a critical design issue in real-time systems in which the system needs to continue meeting task deadline while staying within energy constraints especially in battery operated systems. The Dynamic Voltage and Frequency Scaling (DVFS) and Dynamic Power Management (DPM) are some of the basic technique that optimizes power consumption at the operating system level [2-6] in single core and multicore platforms.

2 ENERGY-AWARE REAL TIME SCHEDULING

In general, real time scheduling problems deal with the ordering of tasks to be executed in a way that the deadline constraints are satisfied. Typically, a task is characterized by its execution time, ready time, deadline and many other resource requirements. Some basic and popular algorithms for single core are Earliest Deadline First (EDF), Rate-Monotonic (RM), Preemptive Utility Accrual Scheduling (PUAS) [7, 8]. These algorithms are integrated with basic DVFS and DPM mechanism to achieve meeting the deadline constraint and minimizes energy consumed in real-time system.

2.1 DVFS

DVFS is an efficient technique for reducing CPU energy. DVFS adjusting supply voltage and frequency used by the CPU as shown in Figure 1. The DVFS framework aims at stretching out task executions through speed and voltage reduction. A higher speed reduces the execution time but increase the power consumption.

Figure 1. Optimization of consumption with DVFS [2]
Table 1 simplifies the notation used in Figure 1 and 2.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>A&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Power</td>
</tr>
<tr>
<td>C&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Execution time</td>
</tr>
<tr>
<td>D&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Deadline</td>
</tr>
<tr>
<td>T&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Period</td>
</tr>
</tbody>
</table>

Instead of lowering processor voltage and frequency as much as possible, energy-efficient real-time scheduling adjusts voltage and frequency according to some optimization criteria, such as low energy consumption and still meeting the deadline of real-time tasks. However, reduction of processor voltage and frequency causing slowdown in the task execution of programs, making a trade-off between energy saving and meeting deadline performance. Hence it is required to schedule the tasks properly, deciding where to change the processor voltage and frequency to have the best energy-efficient performance and yet does not miss any deadline. For real-time systems, the integrated DVFS schemes and real time scheduling algorithm such as EDF focus on minimizing energy consumption in the system, while still meeting the deadlines of real time tasks.

2.2 DPM

DPM is an effective technique for reducing power dissipation in which it has the selective shutdown of system components that are idle or underutilized as shown in Figure 2. The basic mechanism of DPM is to stop the processor when it is not required and to wake up when there are existing tasks [2].

3 EXISTING SIMULATION TOOLS

Many of the researchers in real time scheduling community evaluate the performances of new energy-aware real time scheduling algorithms by using simulation. Therefore, many simulation tools have been developed in last few years. The effort in comparing real time scheduling tools are done in [9, 10]. In [9], a comprehensive comparisons are made to 18 simulation tools based on eight criteria such as simulation, visualization, live-simulation, shared resources, multiple cores, sporadic tasks, open-source and programming languages. [10] summarizes over 20 simulation tools in the literature under four properties such as language, design, performance analysis scheduler profiling. However, none of them focuses specifically on a real time scheduling tools and energy awareness environment. This paper summarizes simulation tools in real time scheduling domain that support energy optimization as shown in Table 2.

<table>
<thead>
<tr>
<th>Tools</th>
<th>Year (doc)</th>
<th>Language (open-source)</th>
<th>Built-in Energy-aware algorithm / energy profile</th>
</tr>
</thead>
</table>
- PFP-ASAP  
- PDP-ST  
- PFPSlacktime  
- etc  
- Energy Profile |
- CC-EDF |
It is observed that most of the tools for energy aware real time scheduling are developed in Java.

3.1 Yartiss

Yartiss is a real-time multiprocessor scheduling simulator developed by researchers in Software, Network and Real-Time Team (LRT) research group in France [1]. Yartiss is an open source simulator and it is built from Java. The main interface of Yartiss is shown in Figure 4.

Yartiss allows researcher to model the energy harvesting technique from renewable energy sources such such as battery, solar and other sources.

<table>
<thead>
<tr>
<th>Source</th>
<th>Year</th>
<th>Language</th>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream [10]</td>
<td>2015</td>
<td>Java (not open-source)</td>
<td>CC-EDF with Total Bandwidth Server (TBS) algorithms</td>
</tr>
<tr>
<td>SimDVS [15]</td>
<td>2003</td>
<td>(not open-source)</td>
<td>InterDVS and IntraDVS family algorithms</td>
</tr>
</tbody>
</table>

Figure 4. Snapshot of Yartiss main interface

It consists of three part i.e., the harvester, storage unit and computing system as shown in Figure 5[16]. The harvester converts the energy from ambient surroundings into usable electrical power. The storage unit is a device to store the electrical power such as rechargeable battery or capacitor. The computing system is a real-time system that utilizes the energy in the battery to run the software [16].
Yartiss is the only simulator that has an energy profile specifically for energy harvesting purposes.

The energy profile consists of the energy source model and energy consumption model. Some of the energy harvesting scheduling algorithms built in Yartiss is shown in Table 3. Two type of scheduling algorithm i.e., fixed priority and dynamic priority such as EDF. The scheduling parameters of these algorithms are processor and the energy load, the amount of incoming energy and the energy level in the storage unit.

**Table 3.** Some of the energy-harvesting algorithm in Yartiss [16-18].

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EHDPDS5 (Preemptive Dynamic Priority)</td>
<td>Two thresholds $E_{\text{min}}$ and $E_{\text{max}}$ are configured. When energy fails under $E_{\text{min}}$, the system is paused for a maximal duration equal to the slack time. If energy state reaches $E_{\text{max}}$, the schedule is resumed.</td>
</tr>
<tr>
<td>PFP-Slacktime (Preemptive Fixed Priority)</td>
<td>If there is not enough energy, the system is paused for duration equal to the slack time. If no slack time, the system is paused 1 time unit. If the energy state reaches $E_{\text{max}}$, the schedule is resumed.</td>
</tr>
<tr>
<td>PFP-ASAP (Preemptive Fixed Priority)</td>
<td>If there is not enough energy, the system is paused for some time unit.</td>
</tr>
</tbody>
</table>

Figure 6 shows an example of source codes of PFP-ASAP in Java that is built in Yartiss.

![Source code of PFP-ASAP](image)
The developer has addressed some future works of Yartiss [1]

i. Implement the resource sharing protocols such as memory, cache and processor synchronization issues.

ii. Use the XML format for inputs and outputs.

iii. Develop the processors with DVFS capability.

3.2 SimSo

SimSo stands for Simulation of Multiprocessor Scheduling with Overheads is a scheduling simulator for real-time multiprocessor architectures that takes into account some scheduling overheads such as scheduling decisions, context switches and the impact of caches in real-time systems [11].

SimSo is developed by the researchers in Laboratory for Analysis and Architecture of System, (LAAR) research group in France [11]. SimSo is an open-source simulator developed based on a discrete event simulation using Python programming language. Currently, more than 25 popular schedulers are available including the basic energy-aware scheduling algorithm that is Static-EDF and CC-EDF. These are RT-DVFS family algorithms developed at early development of energy-aware real time scheduling.

Some of the scheduling algorithms built in SimSo is simplified in Table 4.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static-EDF</td>
<td>An offline algorithm that uses the static slack time technique to scale the CPU frequency. The execution speed is set equal to the lowest available one which guarantees the task set feasibility.</td>
</tr>
</tbody>
</table>
CC-EDF

Cycle conserving EDF uses the dynamic slack to scale the CPU frequency. Initially, tasks execute up to its worst case execution time, and the frequency is set accordingly. Upon completion, if the actual execution time is lesser, then the extra unused cycles are transferred to the remaining tasks. The remaining tasks get more cycles and the frequency can be scaled down.

Figure 10 shows an example of source codes of CC-EDF in Python programming language that is built in SimSo.

A recent survey from the Association for Computing Machinery (ACM) shows that Python surpassed Java as the top language used to introduce students to programming and computer science in the US [21]. This scenario will affect the used of Python as the most popular language in future.

3.3 Storm

Storm stands for Simulation TOol for Real time Multiprocessor scheduling. It is an open source tool developed in Java by the researchers in Parallel Heterogeneous Energy efficient Real-time Multiprocessor Architecture, (Pherma) research group in France [2]. It is a tool that allows multiprocessor simulation and analyzes energy consumption based on estimations. Storms can analyzed the system behavior and performances considering many features of
hardware architecture such as multicore design, multiprocessor architecture with shared memory, distributed architecture with communication network, memory architecture [2]. Figure 11 shows the main interface of Storm.

Storm provides support for DPM and DVFS techniques and can analyze the behavior and to evaluate the performances of the policies of scheduling while taking into account the algorithms of energy management. The processor type that can be used in Storm by the energy-aware scheduler is PXA270. It supports the dynamic adjustment of the power and the performance of the processor based on CPU demand.

3.4 Sparts

Sparts is an open source simulator implemented in Java developed by researchers from Polytechnic Institute of Porto in Portugal [12]. Sparts has the capability to simulate schedulers with various overheads such as preemption, energy consumption and migration for multicore environment. The power model in Sparts provides the information about the energy consumption in the current state of CPU execution such as MaxSpeed, SleepMode and IdleMode as shown in Figure 12. It calculates the overhead of these transitions including switching between different sleep states, by taking into account power properties of each state.
Sparts currently contains 5 uniprocessor scheduling algorithm i.e., EDF, RM, LLF including the leakage aware algorithms such as LC-EDF and ERTH. Leakage Control EDF (LC-EDF) is the first scheduling algorithm to minimize the leakage energy consumption in real-time systems [22-24]. Table 5 shows the leakage aware algorithms built in Sparts.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LC-EDF [22]</td>
<td>Leakage Control-EDF algorithm computes the time interval and delays the execution of the task when CPU is idle to extend the idle intervals and reduce the number of power transitions. As long as the total utilization is less than or equal to 1, the schedulability of the task set is guaranteed [22].</td>
</tr>
<tr>
<td>ERTH [25]</td>
<td>Enhanced Race-To-Halt algorithm tends to run the system at top speed with an aim to create long idle intervals, using slack management approach which are used to deploy a sleep state.</td>
</tr>
</tbody>
</table>

Figure 13 shows an example of source codes of LC-EDF in Java that is built in Sparts. Although Sparts provides the Java source file, the execution file is not provided thus not be able to be used directly by the researchers. The recent work that used Sparts is in [25] to measure the performance of a power aware thermal scheduling algorithm known as ERTH algorithm.

3.5 RTSIM

RTSIM stands for Real-Time system SIMulator is an open source framework to perform discrete event simulations of real-time control systems in distributed environment. RTSIM is written in C++ and the GUI is implemented in Java. The interface of RTSIM is shown in Figure 14.
The scheduling algorithms that are built in RTSIM is shown in Table 6. Most of the scheduling algorithms in RTSIM exploit the slack time for reducing energy consumption when tasks have a variable execution time.

**Table 6.** Energy-aware algorithms build in RTSIM

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RT-DVS</td>
<td>Refer to Table 4.</td>
</tr>
<tr>
<td>DVSST</td>
<td>Dynamic Voltage Scaling Algorithm for Sporadic tasks. DVSST schedules sporadic hard real-time tasks, reclaiming the unused bandwidth to lower the processor frequency. It keeps track on the total bandwidth used by all active sporadic tasks and CPU frequency is changed depending on this value.</td>
</tr>
<tr>
<td>DRA [28]</td>
<td>Dynamic Reclaiming Algorithm DRA is based on detecting early completions and adjusting (reducing) the speed of other tasks on-the-fly in order to provide additional power savings while still meeting the deadlines [28].</td>
</tr>
</tbody>
</table>

Figure 15 shows an example of source codes of RT-DVS in C++ that is built in RTSIM.

Figure 15. Source codes of RT-DVS in RTSIM

Although RTSIM is a bit old, it is used by the recent work in [14] that proposed energy-aware scheduling algorithm known as PAS-ES (Power Aware Scheduler under EDF Scheduling). In [26], the Java GUI is no longer available but instead a C++ implementation of the GUI [9].
3.6 Stream

Stream stands for Simulation Tool for Real-time Energy-efficient scheduling and Analysis for Multi-core processors. It is developed based on Java language but not an open source simulator. The software architecture of Stream is shown in Figure 16 [10]. For simulating energy-aware real-time scheduling, it consists of:

- energy profile with power and frequency voltage mapping
- energy controller with DVFS and DPM together with the existing algorithms EDF and RM integrated with aperiodic Total Bandwidth Server (TBS).
- DVFS trace
- Energy consumption analyzer

![Figure 16. Software architecture of Stream [10]](image)

Although Stream is published in year 2015 that is the most recent energy aware real-time scheduling tools, it is not an open source. Therefore, the documentation is very hard to find to further investigate the properties and scheduling algorithms available in Stream.

3.7 SimDVS

SimDVS is a unified simulation environment that provides a framework for objective performance evaluations of InterDVS and IntraDVS integrated with the EDF and RM real-time scheduling policy [15]. Figure 17 shows the SimDVS simulation environment. Similar to Stream, this tool is not an open source tool. Therefore, the documentation is very hard to find.

![Figure 17. SimDVS simulation environment [15]](image)
4 COMPARISONS

According to [1], there is no standard simulation tool approved by the real time community. [1] suggested four properties for a simulator to become a reference tool by the researchers:

a) The software must be under open source license.

b) API of the software must be well-documented.

c) Each part of the simulator such as task generator, scheduler, result analyzer must be independent from the other parts.

d) Easy to use in a way that a non-developer researchers can be able to use it easily.

This paper summarizes the properties of the most recent energy-aware real-time scheduling tools according to the criteria suggested in [1]. Table 7 shows the comparisons. Column (a) to (d) in summarizes the properties of these tools according to the criteria suggested in [1]. In addition, one more property that influence the selection criteria of energy aware real-time scheduling tools is the popularity of the programming language being used.

Table 7. Comparison of energy aware real time scheduling tools from simulation software perspective.

<table>
<thead>
<tr>
<th>Properties</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tools</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SimSo</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Yartiss</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Storm</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Spurs</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>RTSIM</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Stream</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>SimDVS</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

It is observed that Java dominates the language being used by the tools. However, as mentioned earlier, in the recent survey from ACM, it shows that Python surpassed Java as the top language used to introduce students to programming and computer science in the US [21]. This scenario will affect used of Python as the most popular language in future.

6 CONCLUSIONS

This paper gives a brief survey on the existing energy-aware real-time scheduling tools in the literature. This paper compares the properties of recent real time scheduling tools considering to the criteria proposed in [1] from simulation software perspective. Java language is observed to dominate the development of these simulation tools. Overall, Yartiss is easy to use and gives many properties to model an energy aware scheduling including its power/energy profile.

REFERENCES


