Optimal Procrastinating Voltage Scheduling for Hard Real-Time Systems

Yan Zhang, Zhijian Lu, John Lach, Mircea R. Stan, Kevin Skadron
University of Virginia
Charlottesville, VA 22904, U.S.A.
{zhangyan, zl4j, jlach, mircea, skadron}@Virginia.edu

ABSTRACT
This paper presents an optimal procrastinating voltage scheduling (OP-DVS) for hard real-time systems using stochastic workload information. Algorithms are presented for both single-task and multi-task workloads. Offline calculations provide real-time guarantees for worst-case execution, and online scheduling reclaims slack time and schedule tasks accordingly. The OP-DVS algorithm is provably optimal in terms of energy minimization with no deadline misses. Simulation results show up to 30% energy savings for single-task workloads and 74% for multi-task workloads compared to using a constant worst-case execution voltage. The complexity of our algorithm for multi-task workloads is linear to the number of tasks involved.

Categories and Subject Descriptors
D.4.1 [**Process Management**]: Scheduling; D.4.7 [**Organization and Design**]: Real-time systems and embedded systems

General Terms
Algorithms, Management, Experimentation.

Keywords
Power Management, Dynamic Voltage Scaling, Real-time Scheduling, Optimization Algorithm.

1. INTRODUCTION
In modern VLSI system design, power consumption is one of the most important design constraints. It is especially critical for portable systems due to their limited battery capacity. Meanwhile, mobile processors become more advanced and powerful but consume more energy. Thus, the fundamental tradeoff between performance and battery life remains critically important.

Applications that do not work under hard performance constraints can effectively use dynamic voltage scaling (DVS) to reduce energy consumption when desired or necessary, but applications with hard real-time deadlines must work under strict voltage scaling constraints. However, for such hard real-time systems, if a given task's required performance is lower than the system's maximum performance, the clock speed and its corresponding supply voltage can be reduced to the lowest possible level while still meeting the task's deadline.

DAC 2005 June 13 - 17, 2005, Anaheim, California, USA.

Significant research and development efforts have been expended on DVS [1, 2, 3]. Most of the previously published DVS algorithms use the minimum voltage and frequency allowable for worst-case execution to minimize energy consumption. As pointed out in [4, 5, 6], the optimal DVS approach for energy minimization when the task's execution time is unknown is to gradually increase the speed as the task progresses.

In this paper, we propose an optimal procrastinating DVS algorithm (OP-DVS) for frame-based, hard real-time applications, with the objective of minimizing energy consumption. The proposed algorithm utilizes task execution time distribution information, which can be obtained by profiling similar recently executed tasks, and procrastinates voltage increases as much as possible to minimize unnecessary energy expenditure. Unlike algorithms proposed in [4, 5, 6], we not only consider the single-task scenario but also take into account scenarios with multiple tasks. There are several papers that have proposed DVS algorithms for similar frame-based multiple tasks [7, 8, 9]. However, [7, 9] only change the order of the tasks to reduce energy, and [8] uses a constant voltage and frequency for each task. In this paper, we show mathematically that our OP-DVS algorithm can achieve global energy minimization for frame-based multiple tasks with known task execution time distributions.

2. SINGLE-TASK OP-DVS

2.1 System Model
We first introduce our application model, on which the OP-DVS algorithm is based. Consider a periodic multimedia task that has hard deadlines that are equal to the period. For a single-task scenario, we only schedule one task per period, and no task can be scheduled before the current task's period has expired. As pointed out in [6], although multimedia applications vary instantaneous CPU demands greatly, the probability distribution of their cycle demands is stable or changes slowly and smoothly. This observation makes it possible to schedule the task based on its workloads distribution.

We model the processor as follows: the processor can attain any speed between f_{min} and f_{max}. Although in real systems, only certain numbers of frequency and voltage settings can be chosen. Our algorithm can be modified easily to accommodate the discrete frequency/voltage setting. Given that the time for the processor to switch from one frequency to another is in the microseconds range, the execution for tasks is normally in the milliseconds and seconds range, we can ignore the frequency switching time.

If the threshold voltage V_{TH} is small enough compared to the supply voltage V, the relationship between the clock frequency f and the supply voltage V can be written as $f \approx K \cdot V^{\alpha-1}$, where K is the system constant and α is the velocity saturation factor, which varies from 1 to 2. For the sake of simplicity, in the rest of this paper we assume α=2, but the calculations remain valid for other values of α. The energy e is directly proportional to the square of the supply voltage: $e \sim V^2$ [5].

2.2 Single-Task OP-DVS Algorithm

We now describe our Single-Task OP-DVS algorithm. We assume that the workload distribution for a task S has been obtained in terms of clock cycles in ascending order as $\{c_1,c_2,c_3,...,c_k\}$ and their associated probabilities $\{p_1,p_2,p_3,...,p_k\}$.

Our OP-DVS algorithm will calculate a set of scheduling voltages $V(S)=\{V_1,V_2,...,V_k\}$ that are based on the workload distribution of task S and deadline T. During runtime, we will select different operating voltages and corresponding clock frequencies as the execution of task S progresses. We formulate OP-DVS as a constrained optimization problem as follows.

Find a set of scheduling voltage $V(S)=\{V_1,V_2,...,V_k\}$

Minimize:

$$e(V(S)) = p_1 c_1 V_1^2 + p_2 (c_1 V_1^2 + (c_2 - c_1) V_2^2) + ... + p_k (c_1 V_1^2 + (c_2 - c_1) V_1^2 + ... + (c_k - c_{k-1}) V_k^2) \quad (1)$$

Subject to:

$$\frac{c_1}{KV_1} + \frac{c_2 - c_1}{KV_2} + \frac{c_3 - c_2}{KV_3} + ... + \frac{c_k - c_{k-1}}{KV_k} = T \quad (2)$$

By solving the above optimization problem, we have results as follows.

$$V_1 = \frac{c_1 + (c_2 - c_1)(1 - p_1)^{1/3} + ... + (c_k - c_{k-1})\left(1 - \sum_{i=1}^{k-1} p_i\right)^{1/3}}{KT} \quad (3)$$

$$V_j = V_1 \left(\frac{1}{1 - \sum_{i=1}^{j-1} p_i} \right)^{1/3} \qquad j=2,...,k \quad (4)$$

Using the optimal $V(S)$, we can get the expected energy as follows:

$$e(V(S)) = \frac{\left(c_1 + (c_2 - c_1)(1 - p_1)^{1/3} + ... + (c_k - c_{k-1})\left(1 - \sum_{i=1}^{k-1} p_i\right)^{1/3} \right)^3}{K^2 T^2} \quad (5)$$

One important observation is that $V_1,V_2,...,V_k$ are in ascending order. Thus, optimal voltage scheduling is to begin executing a task at a low voltage and gradually increase it as the task progresses. Another important observation is that given a workload distribution of task S, $V(S)$ is proportional to $1/T$. For a different deadline T, we only need to scale $V(S)$ accordingly.

To consider the voltage limitation, we assume that supply voltage is limited in $[V_{min}, V_{max}]$. If $V_1<V_{min}$, we just set $V_1=V_{min}$ and calculate the rest of the period as a new deadline T'. Using this new deadline, we can reschedule the rest of the workloads. If $V_i>V_{max}$, we use a similar method: set $V_k=V_{max}$ and calculate the rest of period as a new deadline; reschedule the rest of workloads using this new deadline. To meet the discrete frequency/voltage setting, we just round up the voltage scheduling and ensure deadline is never missed. All of these calculations are done offline.

At runtime, since a task may finish before their worst-case execution time, we can set the processor to a low power mode for the rest of period. Figure 1 shows the proposed OP-DVS algorithm for a single task.

Offline:

1.Given task S, deadline T, workload distribution $\{c_1,c_2,c_3,...,c_k\}$ and corresponding probability $\{p_1,p_2,p_3,...,p_k\}$.

2.Calculate optimal schedule $V(S)=\{V_1,V_2,...,V_k\}$ using equations (3) and (4).

Online:

3.Initial_voltage_frequency(S): $V=V_1$, $f=KV_1$.

4.On number of clock cycles finished equal to c_{i-1}, change voltage and frequency to: $V=V_i$, $f=KV_i$, $i=2,...,k$

5.Upon task_finish: set processor to low power mode until T

6.back to step3 for next task.

Figure 1. Single-Task OP-DVS Algorithm.

The above discussion only considers scheduling one task per period. In the next section, we discuss multi-task workloads and propose two modified OP-DVS algorithms to achieve energy minimization.

3. MULTI-TASK OP-DVS

3.1 System Model

We examine the frame-based multi-task model introduced in [9, 10]. There are n tasks per period, all available at time zero. The task set is denoted by $S=\{S_n,S_{n-1},...,S_1\}$. All tasks in a frame have an identical deadline that is equal to their period. The mutual deadline/period (frame length) for n tasks is denoted by T. We also assume that the execution of tasks has been ordered so that S_n is the first task to be executed and S_1 is the last task to be executed. Each task may have its own workload distribution, denoted as $\{c_1,c_2,c_3,...,c_k\}_m$ and corresponding probabilities $\{p_1,p_2,p_3,...,p_k\}_m$, $m=n,n-1,...,1$. One application of this frame-based task model could be decoding MPEG video streams. Each frame actually involves a series of steps: entropy decoding, IDCT (inverse discrete cosine transform), motion compensation, and dithering.

3.2 Local OP-DVS Algorithm

In the Local OP-DVS algorithm, we extend the single-task OP-DVS in a straightforward way. First, we assign time budgets (deadlines) for each task in the frame based on its average workload length. At runtime, the tasks to be executed can utilize the slack time due to early termination of previous tasks.

Local OP-DVS provides a simple and effective solution to schedule multiple tasks for energy efficiency. However since it doesn't consider the interaction between each task in the frame, as explained in the next section, the solution is not globally optimal.

3.3 Global OP-DVS Algorithm

Before we detail our Global OP-DVS algorithm, let's have a look at one simple example. Assume we have two tasks, S_1 and S_2, to be executed sequentially and having a mutual deadline ($T=4.7$). Both tasks have the same distribution of execution cycles, for example, each task has only two possible execution times, $c_1=1$ and $c_2=2$, with the probability of 0.6 and 0.4, respectively.

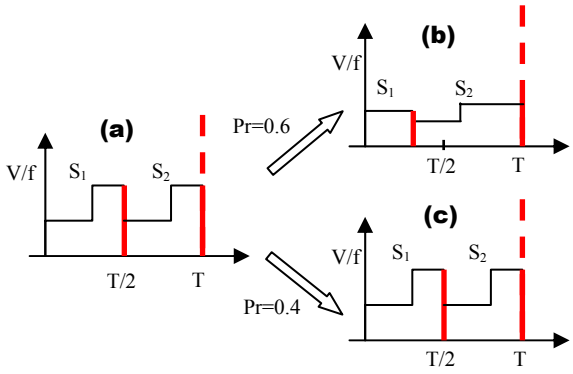

Figure 2. Voltage/frequency scheduling for two tasks with a mutual deadline using Local OP-DVS.

Figure 2 shows how to schedule both tasks using the Local OP-DVS. We assign half of the deadline to each task as its time budget. For each task, we apply Single-Task OP-DVS algorithm, and two voltage levels are assigned for each task. At run-time, S_1 will finish earlier than worst-case with probability of 0.6, and S_2 should be re-scheduled based on its newly extended time budget. With another probability of 0.4, S_1 will use its entire time budget and S_2 will be scheduled in the same way as that in the offline schedule. We obtain $e(S_1,S_2,T)$=1.608 by using Local OP-DVS algorithm.

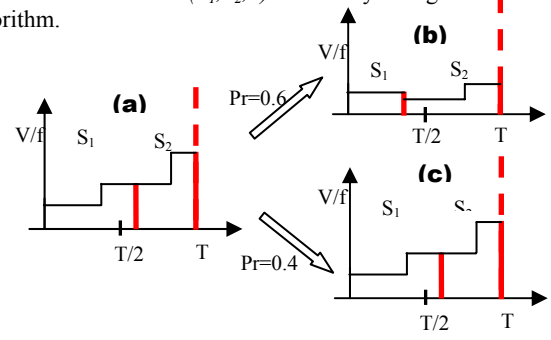

Figure 3. Optimal voltage/frequency scheduling for two tasks with a common deadline using Global OP-DVS algorithm.

The optimal voltage scheduling is demonstrated in Figure 3 using Global OP-DVS. Taking the above particular problem as an example, our Global OP-DVS gives $e(S_1,S_2,T)$=1.53. Notice that, compared with that shown in

Figure 2, the offline scheduling in the optimal solution does not let the two tasks share the deadline evenly, though these two tasks have exactly the same distribution in their execution time.

We derive our Global OP-DVS algorithm as follows. Examining equation (5), which is the calculation of minimum expected energy e for a single task, we can find that for a given workload distribution, eT^2 is a constant. We have the following theorem.

Theorem: Given task set $S=\{S_n,S_{n-1},\ldots,S_1\}$, their workloads distribution $\{c_1,c_2,c_3,\ldots,c_k\}_m$, $\{p_1,p_2,p_3,\ldots,p_k\}_m$ $(m=n-1,n-2,\ldots,1)$ and mutual deadline T, eT^2 is a constant determined only by the workload distributions, where e is the minimum expected energy to execute these n tasks.

This theorem can be proved by induction.

When $n=1$, it becomes the single-task case and the claim is obvious as shown in equation (5). We denote the minimum energy e as $e(1)$ and the constant as $A(1)$ for $n=1$. We can re-write equation (9) as follows.

Assuming that for $n-1$ tasks, our claim is true, we denote the minimum expected energy e for optimal scheduling $V(S_{n-1}),V(S_{n-2}),\ldots,V(S_1)$ as $e(n-1)$ and the constant as $A(n-1)$. Thus we have $e(n-1)T^2=A(n-1)$. Now we prove that our claim is also true for n tasks. Due to the space limit, we omit the detailed derivation

The general equations for optimal schedule $V(S_n)=\{V_j, j=1,2,\ldots,k\}$ can written as follows:

$$V_j = \frac{W}{T_0}\cdot U_j,\ W = \left(\frac{A(n-1)}{K}\right)^{1/3} \tag{6}$$

$$T_0 = \frac{T}{1+\frac{1}{K}\left(\frac{c_1}{WU_1}+\frac{c_2-c_1}{WU_2}+\ldots+\frac{c_k-c_{k-1}}{WU_k}\right)},$$

$$U_j = \left(\frac{p_j}{\left(1+\left(\frac{c_k-c_{k-1}}{W}+\frac{c_{k-1}-c_{k-2}}{WU_{k-1}}+\ldots+\frac{c_{j+1}-c_j}{WU_{j+1}}\right)\right)^3}+U_{j+1}^3\cdot\sum_{i=j+1}^{k}p_i\right)^{1/3}\left(\frac{1}{\sum_{i=j}^{k}p_i}\right)^{1/3}$$

$j=1,2,\ldots,k-1$

and $U_k=1$.

The optimal voltage schedule can be calculated offline according to equation (6). Online scheduling is done by scaling the offline schedule $V(S)$. Figure 4 shows the Global OP-DVS algorithm for frame-based task sets.

Offline:

1.Given task set $S=\{S_n,S_{n-1},\ldots,S_1\}$, workload distribution for each task $\{c_1,c_2,c_3,\ldots,c_k\}_m$,$\{p_1,p_2,p_3,\ldots,p_k\}_m$, $m=n,n-1,\ldots,1$, and mutual deadline T.

2.Schedule(S_1): calculate $V(S_1)$ using Single-Task OP-DVS algorithm under deadline T and obtain constant $A(1)$.

3.Schedule(S_2,$A(1)$): calculate $V(S_2)$ using equation (6) under deadline T and obtain constant $A(2)$.

4.Repeat until $V(S_n)$ and $A(n)$ are obtained.

Online:

5.Execute first task S_n using voltage schedule $V(S_n)$.

6.Upon S_n is finished: $V(S_{n-1})=V(S_{n-1})*T/(T-t(S_n))$

7.Repeat until all tasks in the frame are finished.

8. Set processor to low power mode until T.

9. Back to step 5 for next frame.

Figure 4. Global OP-DVS for frame-based task sets

For the discrete voltage/frequency setting, we use the similar method in single-task scenario.

4. SIMULATION RESULTS

Simulation results were obtained for OP-DVS under both the single-task and multi-task scenarios.

4.1 Single-Task OP-DVS

The baseline result for single-task DVS is for the approach of using the worst-case voltage V_{wc}, which is equal to $c_k/(KT)$. Figure 5 shows the energy savings of Single-Task OP-DVS compared with the execution at V_{wc}. First four benchmarks are Gaussian, Uniform, Exponential Decreasing (EXP(-)), and Exponential Increasing (EXP(+)) workload distributions, respectively. The last benchmark (MPEG) is the workload using *mpegplay* to decode an MPEG-1 video stream.

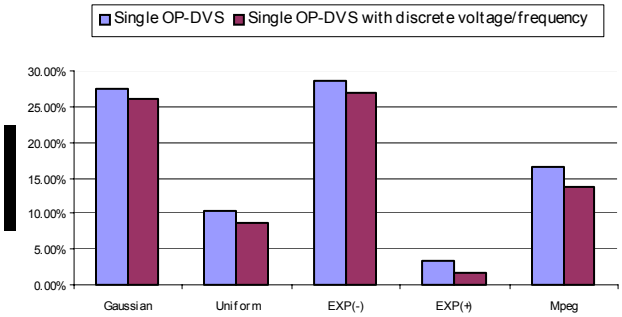

Figure 5. Energy savings provided by Single-Task OP-DVS.

From these results, we see that energy savings of up to 30% with Single-Task OP-DVS. We notice that when the workload distribution is exponential increasing, OP-DVS provides the lowest energy savings, as tasks have a higher probability to finish near the worst-case execution time. For system with discrete voltage/frequency setting, we see slightly smaller saving due to the limitation.

4.2 Multi-Task OP-DVS

Figure 6 shows the energy savings obtained from using Local and Global Multi-Task OP-DVS compared with a constant V_{wc}. These results assume that each frame contains five tasks. In benchmarks Gaussian, Uniform, EXP(-), EXP(+), and MPEG, all tasks in each frame have the same workload distribution

From the results shown in Figure 6 we can see that Global OP-DVS achieve as high as 74% energy savings over V_{wc}. Even with discrete voltage/frequency setting, we still obtain very close energy saving. Furthermore, Global OP-DVS outperforms Local OP-DVS for every benchmark. Therefore, we can draw the conclusion that interactions between tasks within a frame have to be taken into account in order to achieve global energy minimization.

5. CONCLUSION

This paper presents an optimal procrastinating DVS (OP-DVS) algorithm for hard real-time tasks with known execution time distributions, with the objective of minimizing energy consumption. For a single task, the Single-Task OP-DVS algorithm reduces energy consumption by gradually increasing the supply voltage and operating frequency until the task is completed while guaranteeing that the deadline is met. Simulation results show Single-Task OP-DVS achieves up to 30% energy savings over execution at the worst-case voltage.

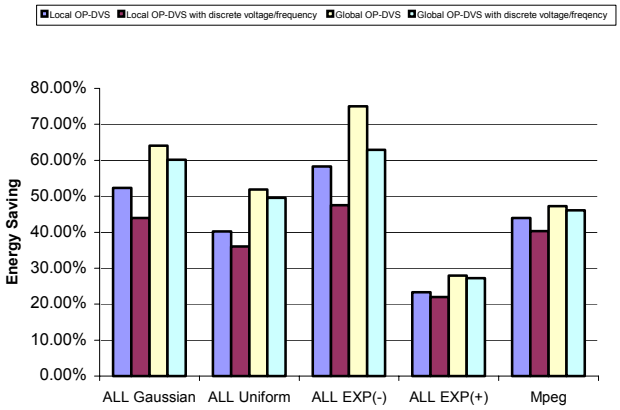

Figure 6. Energy savings provided by multi-task OP-DVS

For frame-based multiple tasks, Local OP-DVS and Global OP-DVS were presented. Local OP-DVS is an extension of Single-Task OP-DVS that utilizes the slack time from adjacent tasks. Global OP-DVS provides further energy reduction by taking into account the interactions of tasks within each frame. Global OP-DVS was mathematically proven to achieve global energy optimization for frame-based task sets, and simulation results show that it can reduce energy consumption up to 74% compared to execution at the worst-case voltage.

6. REFERENCES

[1] D. Grunwald, et al. Policies for Dynamic Clock Scheduling. In *Proc. of the 4th Symposium on OSDI*, 2000.

[2] Y-H. Lee and C. Krishna. Voltage-Clock Scaling for Low Power Energy Consumption in Real-Time Embedded Systems. In *Proc. of the 6th Intl. Conference RTCSA*, 1999.

[3] T. Pering, T. Burd, and R. Brodersen. Voltage Scheduling in the lpARM Microprocessor System. In *Proc. Of ISLPED*, 2000.

[4] J. R. Lorch and A.J. Smith. PACE: A New Approach to Dynamic Voltage Scaling. *IEEE Tran. on Computers*, July 2004.

[5] F.Gruian. Hard Real-Time Scheduling for Low-Energy Using Stochastic Data and DVS Processors. In *Proc. of ISLPED,* August, 2001.

[6] W. Yuan and K. Nahrstedt. Energy-efficient soft real-time CPU scheduling for mobile multimedia systems. In *Proc. of 19th ACM SOSP,* October, 2003.

[7] F. Gruian and K.Krzysztof. Uncertainty-based scheduling: energy-efficient ordering for tasks with variable execution time. In *Proc. of ISLPED,* August, 2003.

[8] C. Rusu, et al. Maximizing the System Value while Satisfying Time and Energy Constraints. In *Proc. RTSS'02,* Dec., 2002.

[9] L. Leung, C. Tsui, and W. Ki, Minimizing energy consumption of hard real-time systems with simultaneous tasks scheduling and voltage assignment using statistical data, *ASP-DAC*, Jan., 2004.