# Optimality and Improvement of Dynamic Voltage Scaling Algorithms for Multimedia Applications

Zhen Cao, Brian Foo, Lei He and Mihaela van der Schaar

Electronic Engineering Department, UCLA
Los Angeles, CA 90095

## ABSTRACT *

The time-varying workload for multimedia applications poses a great challenge for the efficient performance of dynamic voltage scaling (DVS) algorithms. While many DVS algorithms have been proposed for real-time applications, there does not yet exist a systematic method for evaluating the optimality of such DVS algorithms. In this paper, we propose an offline linear programming (LP) method to determine the minimum energy consumption for processing multimedia tasks under stringent delay deadlines. Based on this lower bound, we evaluate the efficiency of various existing DVS algorithms. Furthermore, we modify the LP formulation to construct an online robust sequential linear programming DVS algorithm for real-time multimedia processing. Simulation results from decoding over a wide range of video sequences shows that on average, our online algorithm consumes less than 1% more energy than the optimal lower bound while dropping only 0.1% of all scheduled decoding jobs, while the existing best algorithm consumes roughly 3% more energy at the same miss rate.

## Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-based Systems**]: Real-time and embedded systems

## General Terms

Algorithms, Performance

## Keywords

Multimedia, Dynamic Voltage Scaling, Power Management

## 1. INTRODUCTION

Due to the popularity of streaming multimedia applications on mobile and pervasive computing devices, computationally intensive multimedia applications must often be processed by energy-limited systems. Dynamic voltage scaling (DVS)-enabled processors are particularly attractive for such devices, since they can adapt their voltage level and associated clock frequency in real time to save energy while handling time-varying workloads and display deadlines [1][2]. In general, a DVS-enabled processor can conserve energy by reducing its voltage level; however, decreasing the voltage level will also slow the processor clock speed, thereby increasing the processing time, and hence the overall delay [2]. DVS algorithms therefore attempt to find a dynamic balance between the operating level (i.e. power and frequency) of the processor, and the quality-of-service for multimedia applications in terms of meeting stringent delay deadlines.

A wide variety of DVS algorithms have been proposed for delay-sensitive applications [3-8]. Some DVS algorithms perform optimization myopically over only one or two tasks, such that the processor power level is determined on the fly to meet imminent (soft) deadlines while considering either the worst case execution time (WCET), or the average case execution time (ACET) [4][6]. While such approaches have very low computational complexity, their performance is limited in that future tasks with imminent deadlines may require extremely high processing power to finish in time after the completion of the current task. On the other hand, more robust DVS algorithms schedule power based on multiple task deadlines [3], or using feedback control [5][7]. Nevertheless, such approaches can be highly suboptimal when jobs have highly time-varying workloads that cannot be accurately predicted by simple models. In [7], a combined leakage and DVS scheduling algorithm, based on earliest-deadline-first scheduling (EDF), is proposed for hard real-time systems. In [8], a queuing model-based DVS approach is proposed for video decoding, where different video frames and sequence types are classified into different sets of complexity distributions. Based on the encoding parameters, the delay for processing each frame can be analytically approximated for different processor operating levels, thus enabling the system to adapt the processor voltage in real-time and achieving the best reported energy savings. However, due to the online nature of most DVS algorithms, it is difficult to prove how far these algorithms are from the optimal power scheduling scheme without an optimal solution.

Moreover, leakage current in CMOS circuits today contributes a significant portion to the total power consumption, and is expected to increase five-fold with each generation [9]. Hence, technologies such as power gating are used to switch off unused resources to reduce leakage power when the workload is light. Under such conditions, zero power and frequency of sleeping mode should be considered in a DVS algorithm and it is possible that the power-frequency function for processors will no longer be convex. In this case, existing works [4][5][8] that attempt to minimize idle periods under the assumption of a convex power-frequency function will be no longer effective. Hence, a pervasive algorithm which is adaptive for both convex and non-convex power-frequency functions is needed for the DVS problem.

The contribution of this paper is as follows: first, we analyze the optimality of DVS algorithms by deriving an operational

* This work was partially supported by NSF CCR-0306682. Address comments to lhe@ee.ucla.edu.


*DAC 2008*, June 8–13, 2008, Anaheim, California, USA

lower bound for energy consumption, subject to processing all jobs before their delay deadlines (i.e. zero miss rate). We propose a linear programming (LP) DVS solution to obtain the optimal offline scheduling solution for both convex and non-convex power-frequency functions. Unlike the integer programming formulation presented in [11] for temperature-aware DVS scheduling, we consider the negligible voltage-switch overhead (compared to the high multimedia job complexities) to enable switching at any time during a job, and we show that this can be converted to a low complexity LP problem. Based on the workload traces collected during execution time, we solve the offline LP problem to obtain the lower energy bound for DVS algorithms. A thorough investigation of video decoding results (where many video sequences are decoded at many different bit rates) shows that, under the same zero miss rate, laEDF [3] consumes approximately 15% more energy than the optimal solution, and the queuing based algorithm in [8] consumes approximately 4% more than the optimal solution.

Based on the proposed LP formulation and accurate multimedia complexity modeling, we propose an online robust sequential linear programming approach to DVS, namely SLP/r, which significantly outperforms existing DVS solutions. Experimental results from real-time video decoding (where workloads are highly time-varying) indicate that SLP/r consumes less than 1% more energy than the optimal DVS solution while dropping only 0.1% of all scheduled decoding jobs, while the best existing algorithm (the queuing-based algorithm 2 in [8]) consumes roughly 3% more energy than SLP/r at the same miss rate. Importantly, SLP/r has only a small increase in complexity, and its relative complexity scales down when supporting increasingly computationally intensive future multimedia applications.

While we have used video decoding as an example in this paper for motivation and simulation, both the offline LP and online SLP/r approaches are applicable for the DVS problem concerning other delay-sensitive real-time applications with time-varying workloads.

The rest of this paper is organized as follows: Section 2 provides background on multimedia complexity and power modeling. Section 3 formally states the real-time DVS problem. Sections 4 and 5 present the optimal offline LP solution and the online SLP/r algorithm, respectively. Section 6 presents experimental results to validate our work. Section 7 concludes our work.

# 2. BACKGROUND AND MODELING

## 2.1 Multimedia Complexity

State-of-the-art coders often encode nearby video frames jointly to reduce the video transmission bit-rate. However, this leads to complicated group-of-pictures (GOP) structures, where some video frames require the reconstruction of many intermediate frames in order to be decoded, while other video frames require very few intermediate frames, resulting in very high workload variations between adjacent decoding jobs (Figure 1). Different sequences also exhibit different amounts of decoding complexity.

To mitigate the detrimental effects of highly time-varying workloads on DVS algorithms, in this work, we adopt the application-aware model for the video coding complexity described in [8] for the online algorithm proposed in this paper. In our work, *a job class* corresponds to a particular GOP frame type of a sequence type (e.g. we consider eight job classes for a GOP structure of eight decoding jobs). To model the (decoding) complexity within each class of jobs, offline training is used to obtain workload distributions of each class of jobs, as shown in Figure 2. These distributions enable us to collect important information about the complexity, such as the mean and variance for each class of jobs. As will be shown later, such information can be used by our proposed online DVS algorithm to achieve a tradeoff between energy consumption and quality-of-service.

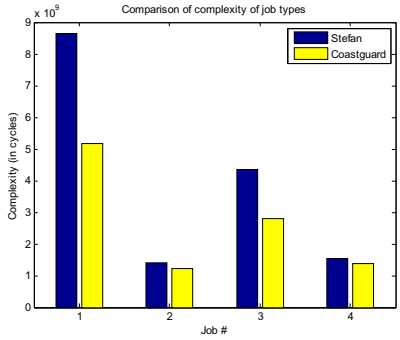



**Figure 1 Comparison of various decoding jobs for video sequences *Stefan* and *Coastguard*.**

Finally, note that the complexity of each video decoding job (which consists of at most 2 video frames each) is on the order of a billion cycles. Hence, overheads associated with voltage switch, which are on the order of less than one hundred clock cycles [12], are negligible compared to the processing complexity of multimedia tasks. Hence, as long as the DVS algorithm requires no more than a few voltage switches during a single job, we can ignore the voltage switching overhead.

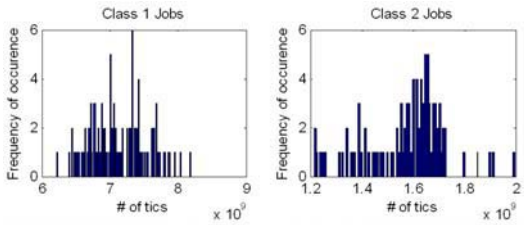



**Figure 2 The workload variance within the same types of decoding jobs.**

## 2.2 Dynamic and Leakage Power

In our work, we adopt the power model proposed in [10] and used in [9][11] for real-time applications. The dynamic power is:

$$P_d = C{V_{dd}}^2 F \tag{2.1}$$

where $C$ is the effective switching capacitance. We chose the leakage power model from [10], which includes the subthreshold and the reverse bias leakage power. For a given supply voltage $V_{dd}$, the leakage power $P_s$ and subthreshold leakage current $I_{sub}$ are:

$$P_s = L_g \left( V_{dd} I_{sub} + |V_{bs}| I_j \right) \tag{2.2}$$

$$I_{sub} = K_3 e^{K_4 V_{dd}} e^{K_5 V_{bs}} \tag{2.3}$$

where $C$, $V_{bs}$, $L_g$, $I_j$, $K_3$, $K_4$ and $K_5$ are constants for the 70nm node technology given in [10]. The clock frequency $F$ and threshold voltage $V_{th}$ are:

$$F = (V_{dd} - V_{th})^a / (L_d K) \quad (2.4)$$

$$V_{th} = V_{th1} - K_1 V_{dd} - K_2 V_{bs} \quad (2.5)$$

where $a$, $V_{th1}$, $K_1$, $K_2$, $L_d$, $K$ and $V_{bs}$=-0.7V are given for the 70nm node technology in [9]. Note that the algorithms proposed in this paper apply to any power model, whether the power-frequency function is convex or not. In general, the processor consumes both dynamic and leakage power for a given $V_{dd}$ level, and consumes no power when the $V_{dd}$ level is zero, i.e. in the power gating or sleep mode.

## 3. PROBLEM STATEMENT

For our real-time deadline-driven multimedia DVS problem, we are given a sequence of decoding jobs. For each job (which can be one frame or a pair of frames depending on the GOP structure), we are given its complexity, arrival time and display deadline. Because we are performing real-time media transmission and decoding, the arrival time can be influenced by the variance of the network bandwidth. On the other hand, a voltage/frequency configurable system can switch the frequency of its processor by dynamically adapting its voltage level. Hence, we have a set of active operating levels with frequencies and corresponding powers (sum of leakage power and dynamic power). Furthermore, if power gating is enabled, such that the processor can be shut down to save leakage power, we have an additional operating level for sleep mode.

The goal of a DVS algorithm is to find a scheduling solution, which consists of the time and the operating voltage for each switch, to minimize total energy consumption. The constraint is that the decoder can only start decoding a job after it arrives, and each job should be finished before its display deadline.

Given $M$ decoding jobs, let $C = \{C_1,\ldots,C_M\}$ , $T = \{T_1,\ldots,T_M\}$, $D = \{D_1,\ldots,D_M\}$ be the complexity, arrival time and display deadline of jobs, respectively; let $F = \{F_0,\ldots,F_k\}$, $P = \{P_0,\ldots,P_k\}$ ($F_0$ and $P_0$ for sleeping mode) be associated clock frequencies and powers for $K$ voltage levels, respectively; let $S = \{T_s, V_s, N\}$ be the scheduling solution, where $N$ is the number of voltage switches, $T_s=\{t_0,\ldots,t_N, t_0=0\}$ and $V_s=\{v_0,\ldots,v_N\}$ is the time and voltage level for each switch. When the precise complexity of each job is known, the constraints for the problem are given by deterministic $C_i$ and $T_i$. However, when uncertainties exist in the workload, $C_i$ and $T_i$ can be viewed as stochastic variables and DVS scheduling algorithms cannot guarantee that all jobs will be decoded before their deadlines. Hence, in the stochastic case, the hard deadline constraint can be replaced with the constraint of keeping the miss rate for jobs within a tolerable range.

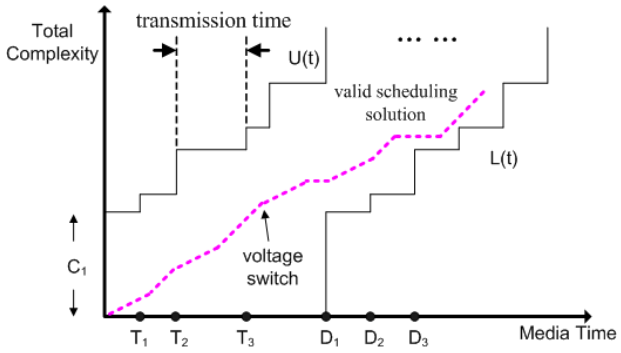



**Figure 3 DVS problem formulation**

We further illustrate the DVS problem in Figure 3. Here, the step function $U(t)$ is the accumulated complexity (i.e., total complexity in terms of clock circles) of decoding jobs transmitted since time zero, and the widths of the steps are transmission times of jobs over network. Another step function $L(t)$ indicates the total complexity that needs to be processed by time $t$ in order to meet display deadlines. I.e.,

$$U(t) = \sum_{j=1}^{k} (C_j), for\ T_{k-1} < t \le T_k,\ 1 \le k \le M,\ T_0 = 0 \quad (3.3)$$

$$L(t) = \sum_{j=0}^{k-1} (C_j), for\ D_{k-1} \le t < D_k,\ 1 \le k \le M, C_0 = 0, D_0 = 0 \quad (3.4)$$

Since the decoder cannot start decoding a job before it is completely received from network, and it must finish the job before its deadline, a valid DVS solution is a piecewise linear curve between $U(t)$ and $L(t)$. The slope of each piece indicates the associated clock frequency of the selected voltage level and a corresponding power is associated with each frequency.

The arrival time intervals can vary between each job. As shown in Figure 3, the transmission time for job 3 is larger than others. This can often occur when the network bandwidth is time-varying, which is common in wireless networks.

## 4. OPTIMAL OFFLINE SOLUTION

In this section, we show that the deadline-driven multimedia DVS problem can be mapped into a LP problem, which can be efficiently solved. Furthermore, if we know the precise complexity and arrival time of each decoding job, we can obtain the optimal scheduling solution.

We define a *transition point* as the time when a new job arrives (i.e. any $T_i$), or when a job deadline is reached (i.e. any $D_i$). We also define an *adaptation interval* as the time period between two adjacent transition points. The adaptation intervals for sample $U(t)$ and $L(t)$ curves are marked in dotted lines in Figure 4. We now prove an important result for voltage scheduling.

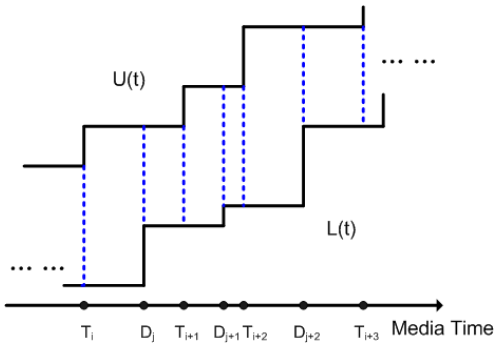



**Figure 4 Adaptation intervals**

*Theorem 1.* Within a single adaptation interval (i.e. when $U(t)$ and $L(t)$ are constant), an arbitrary ordering of any feasible voltage schedule is feasible and consumes the same amount of energy.

*Proof:* Suppose we have a voltage time allocation of each voltage level in this interval. Then, the total energy consumption is the sum of each allocated time slot multiplies the corresponding power. Similarly, the total complexity consumption is the sum of each allocated time slot multiplies the corresponding frequency. Then, if the time allocation is fixed, the energy and complexity consumptions are both fixed. Since the voltage scheduling

solution is valid and *U(t)* and *L(t)* are stable within this interval, the piecewise linear solution curve will not break the bound in spite of the ordering. As an example in Figure 5, the complexity consumption of sequence 2,0,1,3,4 and 0,1,2,3,4 (the numbers refer to the slopes) with the same time allocation is the same. ■

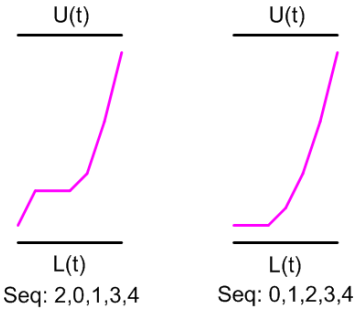



**Figure 5 Different voltage scheduling orderings**

Theorem 1 is the *key idea* to map the DVS problem into a tractable LP problem. Rather than finding the precise times for voltage switch, which would create an intractable integer linear programming (ILP) problem as in [11], we instead solve the *percentage* of time that the processor is operating at each voltage level within each adaptation interval. The LP problem can be formulated as follows:

*Problem Formulation 1*: label the transition points as an ordered set $I = \{I_0,...,I_L\}$, where $I_0=0$ and $I_L = T_{end}$, i.e. we have a total of $L$ adaptation intervals. For these $L$ intervals, we have voltage level allocation vectors given by $A = \{A_1,\ldots,A_L\}$, where $A_i=\{A_{i0},\ldots,A_{iK}\}$ and $A_{ij}$ is the allocation of voltage level $j$ in adaptation interval $i$. Then, the DVS problem is:

$$\min\ E = \sum_{i=1}^{N}\sum_{j=0}^{K}(A_{ij}\bullet P_j\bullet(I_i - I_{i-1})) \tag{4.1}$$

such that

$$0 \le A_{ij} \le 1,\ for\ 0 \le j \le K\ \ and\ \sum_{j=0}^{K} A_{ij} = 1 \tag{4.2}$$

$$L(I_n) \le \sum_{i=1}^{n}\sum_{j=0}^{K}(F_j\bullet A_{ij}\bullet(I_i - I_{i-1})) \le U(I_n), \forall\ 1 \le n \le L \tag{4.3}$$

Here, the unknown is the voltage level allocation vectors given by $A$. The constraint shown in (4.3) is the valid DVS solution between *U(t)* or *L(t)* which are defined in (3.3) and (3.4).

One can easily prove that the problem defined in (4.1) to (4.2) is a linear programming problem. Hence, with this formulation, solving this LP problem leads to the optimal solution for offline DVS problem. Note that this formulation is *pervasive*: the operating voltages can take on any set of discrete values, and there is no requirement for the power-frequency model (no need for a convex power-frequency function). Furthermore, this formulation is also applicable to other delay-sensitive DVS problems of real-time applications.

In our formulation, we have made the assumption that the values of *U(t)* and *L(t)* are precisely known, which means that we know the exact complexity and arrival time for each decoding job. This information can be obtained from the trace of the video decoding task. This approach can be used to determine the operational lower bound for energy consumption, as well as whether the utilized online DVS algorithm operates close to the optimal scheme.

# 5. EFFECTIVE ONLINE ALGORITHM

For online multimedia applications, where jobs are received through a network, we often do not know the precise complexity and arrival times of each decoding job. Nevertheless, the idea of mapping DVS into a linear programming problem in section 4 can still be used for online DVS. We solve the stochastic online DVS problem by sequentially solving robust linear program (rLP). We label our algorithm SLP/r.

The main idea of SLP/r is: we predict the stochastic complexity of decoding jobs in a future time window by using the means and variances of jobs, and solve an rLP problem to obtain the scheduling solution for the predicted decoding jobs in the window. After completing a decoding job (or several decoding jobs), we move the window forward based on the current time, adjust the predictions in the future window, and repeat the rLP based on possibly new statistics.

## 5.1 Consideration of Stochastic Complexity

The prediction of future decoding job complexities (in the sliding window) is crucial to our real-time DVS solution. In real time video transmission, this can be accomplished by having the encoder send complexity specifications, such as the mean and variance of each job class for each video scene, prior to transmitting the corresponding frames [8].

Using only the mean of each job class for prediction may lead to a high miss rate. To reduce the probability of misses, we incorporate the variance of each job class with mean to estimate the bounded "worst case" complexity in a probabilistic manner. Due to the central limit theorem, when uncertainties accumulate over many jobs (See Figure 2 for workload distribution examples.), the total workload tends toward a Gaussian distribution. In this case, the mean and variance for each job class can explicitly determine the miss rate probability under different adjustments of *U(t)* and *L(t)*. The adjustments are based on a confidence level $\alpha$ to adjust the new bounds. Note that for jobs far into the future of a prediction window, the accumulated variance over many jobs may be large. Hence, a scaled coefficient $\alpha$ (possibly 0, such that only the mean is considered) can be used to guarantee feasibility. Using a small coefficient for jobs far into the future does not necessarily increase the miss rate, since the rLP solution will only determine the DVS schedule for imminent jobs, after which the rLP is solved again for the future jobs based on the decoding results.

The rLP problem for a given prediction window is as follows:

*Problem Formulation 2*:

$$\min\ E = \sum_{i=1}^{W}\sum_{j=0}^{K}(A_{ij}\bullet P_j\bullet\varphi) \tag{5.1}$$

such that

$$0 \le A_{ij} \le 1,\ for\ 0 \le j \le K\ \ and\ \sum_{j=0}^{K} A_{ij} = 1 \tag{5.2}$$

$$L(I_n) \le \sum_{i=1}^{n}\sum_{j=0}^{K}(F_j\bullet A_{ij}\bullet\varphi) \le U(I_n), \forall\ 1 \le n \le W \tag{5.3}$$

Where $\varphi$ is the display interval, $W$ is the prediction window size. Adaptation intervals $I$, *U(t)* and *L(t)* are defined as follows (detailed description is in section 5.2):

$$I = \{I_0,\ldots,I_W\}, I_i = i\bullet\varphi \tag{5.4}$$

$$U(t) = \sum_{j=1}^{k} (\widetilde{C}_{W_0+j}), I_{k-1} < t \le I_k,\ 1 \le k \le W \tag{5.5}$$

$$L(t) = \max(0, U(t - \theta \bullet \varphi)) \tag{5.6}$$

Where $W_0$ is the current adaptation interval and $\widetilde{C}_i$ is stochastic complexity of job $i$ based on [8]. Specifically we have:

$$\widetilde{C}_{W_0+j} \le \rho_{W_0+j} + \alpha_j \bullet \sqrt{v_{W_0+j}} \tag{5.7}$$

$$\alpha_j = \max(0, \alpha \bullet (R - j + 1)/R), 1 \le j \le W \tag{5.8}$$

where $\rho_i$ and $v_i$ is the mean and variance of stochastic complexity of job $i$, $\alpha$ is the confidence level set by the user and $R$ is a constant. (5.8) indicates that, $\alpha_j$ is linearly scaled between $\alpha$ and 0 over time. Note that a tradeoff between miss rate and energy consumption could be achieved by tuning $\alpha$. For example, increasing $\alpha$ will make the bounds tighter and lead to more energy consumption but a lower miss rate.

One can easily show that the problem defined by equations (5.1) to (5.8) is an rLP problem. Note that with stochastic complexity model, the proposed online algorithm applies to other real-time applications although we only use video decoding as an example in this paper.

After we finish decoding one job, we need to adjust *U(t)* and *L(t)* dynamically. The idea is shown in Figure 6. The gray area indicates the variance part of prediction, the dotted line indicates the adaptation intervals and the dotted area indicates the bound adjustments. Figure 6(a) shows the solution from robust linear programming using mean and variance of each job class as a prediction. The real complexity of each job is shown in Figure 6(b). In this particular case, we used three adaptation intervals to finish decoding job No.1. As our granularity for recalculating the solution is one adaptation interval, we may consume more complexity than that of the specific job. As shown in Figure 6(b), after finishing job No.1, we also finished job No.2 and part of job No.3. Hence, we need to adjust the prediction dynamically to give more accurate prediction. As shown in Figure 6(c), the dotted area indicates that part of job No.3 is already finished in the previous interval, and when we move the window forward, we need to adjust *U(t)* and *L(t)* accordingly.

## 5.2 Extension to Variable Communication

For SLP/r, another problem is that we need to deal with variance of network bandwidth, because we do not know the exact arrival time of each job. The idea is that we assume that a network buffer at the decoding side collects packets and dispatches jobs to the decoder according to the display frame rate. We fix the dispatch time as $\theta$ display intervals before the display deadline of the job. This means that we predict adaptation intervals using only display intervals. In this fashion, we can reduce the number of adaptation intervals (hence the size of the rLP problem). In this case, the adaptation intervals *I, U(t)* and *L(t)* are defined as (5.4) to (5.6). If a job arrives before our scheduling solution (i.e. the real *U(t)* is higher than the complexity consumption line), we just switch voltages as guided by rLP. Sometimes, a job may arrive very late due to insufficient bandwidth of network, which may occur in unreliable wireless networks. In this case, if a job arrives too late to be processed, power gating can be used to shut down the processor until a new job arrives, based on which *U(t)* and *L(t)* are adjusted for the next rLP.

# 6. SIMULATIONS AND RESULTS

## 6.1 Experimental Setup

For experiments, we adopted the power and frequency model for the 70nm technique node in [9][11]. We considered discrete voltage levels $V_{dd}$ between 0.6V and 1.0V with voltage step sizes of 0.1V. The clock frequencies and power for different $V_{dd}$ levels are presented in Table 1.

**Table 1. Frequency and power for different $V_{dd}$ levels**

| **Vdd (V)** | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
|---|---|---|---|---|---|
| **Frequency (GHz)** | 0.79 | 1.27 | 1.81 | 2.42 | 3.09 |
| **Dynamic Power ($10^{-5}$W)** | 0.12 | 0.27 | 0.50 | 0.84 | 1.33 |
| **Leakage Power ($10^{-5}$W)** | 0.21 | 0.29 | 0.40 | 0.54 | 0.72 |
| **Total Power ($10^{-5}$W)** | 0.33 | 0.56 | 0.90 | 1.38 | 2.05 |

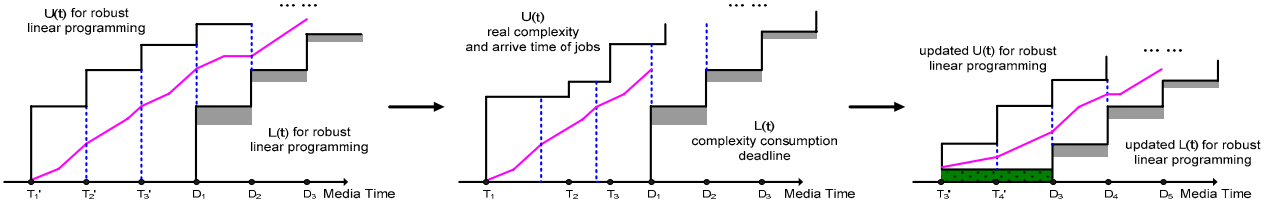
We combined 10 video sequences into a long sequence, which was then decoded. We measured the complexity for each decoding job in terms of clock cycles and used the measurement for offline scheduling. We also tuned the stochastic model using the measurement for the proposed online algorithm SLP/r.

Furthermore, we generated more data based on Monte Carlo method to present a more general simulation. The experiment observation was almost the same as the result from real data. Hence, we only present the result from the real data here.

To simulate a real-time video decoding environment with sequences that have a frame rate of 30Hz, we fix the hard display deadlines. We assume that the (soft) frame arrivals from the network follow norm distribution as in [13] to simulate a wireless network, and we applied the same generated arrival times of jobs for all algorithms in our experiments. For all algorithms, we calculate energy with the same power model considering leakage power. Since the absolute value of energy is not important, we report normalized energy with respect to energy consumed by the optimal solution.

## 6.2 Optimality Study

In our experiment, we revised power models of laEDF [3] and queuing based algorithms [8] to consider leakage power. Also we revised these algorithms to consider sleep mode for a fair comparison.




(a) obtain solution from rLP (b) adopt DVS solution for job 1 (c) adjust prediction and repeat rLP

**Figure 6 Dynamic adjust of prediction for rLP**

For queuing based algorithms 1 and 2 in [8], we selected algorithm 2 for comparison as it outperforms algorithm 1 experimentally. We tuned the parameters to obtain different trade off points for energy and miss rate. For the queuing based algorithm, we tuned the delay sensitivity parameter $\varepsilon$, and for laEDF, we used different WCETs. The result is shown in Figure 7. The energy achieved by the optimal offline LP solution (e.g. the lower bound) is normalized to 1. Note that based on our formulation, the optimal solution always has zero miss rate. The result shows that for a zero miss rate, laEDF consumes approximately 15% more than the optimal and queuing based algorithm 2 consumes approximately 4% more than optimal.

## 6.3 Effectiveness of SLP/r

We also compared the proposed online algorithm SLP/r with the optimal solution and existing DVS algorithms. For SLP/r, we tuned the confidence level $\alpha$ to obtain different trade off points for energy and miss rate. The sliding window size of SLP/r is set to 16 jobs (2 GOPs).

From Figure 7, one can see that SLP/r has only about 1% more energy consumption than the optimal solution while keeping the miss rate below 0.1%. The existing best algorithm (the queuing-based algorithm 2 in [8]) consumes roughly 3% more energy than SLP/r under the same missing rates (0.1%), while laEDF consumes approximately 13% more than SLP/r. Compared with queuing based algorithm 2, SLP/r shrinks the gap between online algorithm and the optimal offline algorithm by nearly 75%.

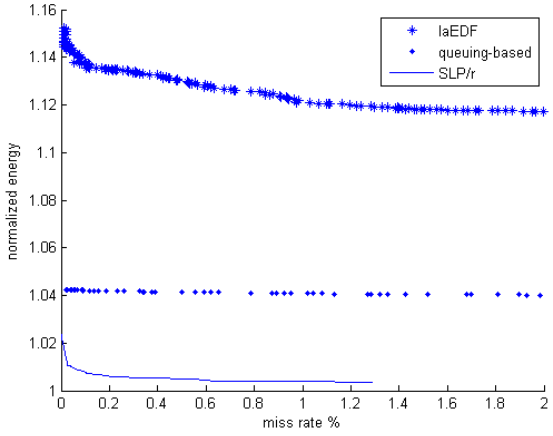



**Figure 7 Energy and miss rate**

The total runtime of SLP/r for the combined 512s long video sequence is 36s, which means that the overhead of the online scheduling algorithm is approximately 7% of the video decoding workload, which is acceptable. We expect this relative runtime overhead to decrease in the future with a more careful implementation. The associated energy overhead of scheduling will also decrease relatively to the more computationally intensive multimedia algorithms for higher resolution video decoding in the future.

# 7. CONCLUSIONS AND FUTURE WORK

In this paper, we have analyzed the optimality of online DVS algorithms by formulating the optimal off-line DVS into a linear program (LP). We show that at a zero miss rate, existing works consume at least 4% more energy than the optimal. We have also developed an effective online DVS algorithm using robust sequential linear programming (SLP/r), which significantly outperforms existing online DVS solutions and is merely 1% away from the optimal. Furthermore, we plan to develop algorithms that are more efficient than SLP/r. For example, we can provide more precise prediction of the jobs by exploiting video sequence characteristics and coding parameters in state-of-the-art coding algorithms. In this way, we can reduce the overhead by reducing the frequency of solving the LP problem, and increasing the performance of the scheduling solution. Preliminary results (refer to [14] for details) show that we can speed up the runtime to 8x compared with SLP/r while reducing energy consumption at the same miss rate.

# 8. REFERENCES


[1] L. Benini, and G. De Micheli. Dynamic power management: design techniques and CAD tools. Kluwer Academic Publishers, Norwell, MA, 1997.

[2] D. Marculescu. On the use of microarchitecture-driven dynamic voltage scaling. *Proceedings of the Workshop on Complexity-Effective Design*, 2000.

[3] P. Pillai, and K. Shin. Real-time dynamic voltage scaling for low-power embedded operating systems. *Proceedings of the 18th ACM symposium on Operating Systems*, 2001.

[4] W. Yuan, K. Nahrstedt, S. Adve, D. Jones, and R. Kravets. GRACE: cross-layer adaptation for multimedia quality and battery energy. *IEEE Transactions on Mobile Computing*, 2006.

[5] Y. Zhu, and F. Mueller. Feedback EDF scheduling exploiting dynamic voltage scaling. *Proceedings of the 11th international conference on Computer Architecture*, 2004.

[6] K. Choi, K. Dantu, W. Cheng, and M. Pedram. Frame-based dynamic voltage and frequency scaling for a MPEG decoder. *Proceedings of ICCAD*, 2002.

[7] Y. Zhu, and F. Mueller. DVSleak: combining leakage reduction and voltage scaling in feedback EDF scheduling. *Proceedings of LCTES*, 2007.

[8] B. Foo, and M. van der Schaar. A queuing theoretic approach to processor power adaptation for video decoding systems. *IEEE Trans. Signal Process*, to appear.

[9] R. Jejurikar, C. Pereira, and R. Gupta. Leakage aware dynamic voltage scaling for real-time embedded systems. *Proceedings of DAC*, 2004.

[10] S. Martin, K. Flautner, T. Mudge, and D. Blaauw. Combined dynamic voltage scaling and adaptive body biasing for low power microprocessors under dynamic workloads. *Proceedings of ICCAD*, 2002.

[11] S. Zhang, and K. S. Chatha. Approximation algorithm for the temperature-aware scheduling problem. *Proceedings of ICCAD*, 2007.

[12] J. Dunning, G. Garcia, J. Lundberg, and E. Nuckolls. An all-digital phase-locked loop with 50-cycle lock time suitable for high-performance microprocessors. *IEEE Journal of Solid-State Circuits*, Volume 30, Issue 4, Apr 1995 Page(s):412 – 422.

[13] A. Adas. Traffic Models in Broadband Networks. *IEEE Communications Magazine*, Vol. 35, Issue 7, July 1997.

[14] Z. Cao, B. Foo, L. He, and M. van der Schaar. Optimality and Improvement of Dynamic Voltage Scaling Algorithms for Multimedia Applications. *Technical Report UCLA*, 08-267, 2008.