1.4p

Dynamic Power Management for Streaming Data

Nathaniel Pettis, Le Cai, and Yung-Hsiang Lu
School of Electrical and Computer Engineering, Purdue University
npettis,lc,yunglu@purdue.edu

ABSTRACT

This paper presents a method that uses data buffers to smoothen request variations and to create long idleness for power management. This method considers the power consumed by the buffers and assigns an energy penalty for buffer underflow. Our approach provides analytic formulas for calculating the optimal buffer sizes and the amount of data to store in the buffers. We use video prefetching as a case study and obtain power savings of more than 74% for MPEG-1 and 34% for MPEG-2 videos.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Performance attributes

General Terms

Design, Performance

Keywords

Dynamic power management, streaming video, prefetching, quality of service

1. INTRODUCTION

Dynamic power management (DPM) [1] changes a component's power states when it is idle or under-utilized. For example, when a hard disk is idle, the disk can be shut down ("turned off" or "asleep") to save power. The main challenge in DPM is to correctly predict when a component is idle. This is due to the overhead of DPM: shutting down a component takes time and additional energy. Hence, a component should be shut down only if the overhead can be amortized across a long period of idleness with sufficient energy savings. Some methods use scheduling to explicitly create idle periods [9, 11]. Some other methods model the interactions between a service provider and a service requester as stochastic processes [2, 4, 13]. A queue is inserted between the provider and the requester, as shown in Figure 1 (a). These methods shut down the provider when the queue is empty and expected to remain empty for an extended period of time.

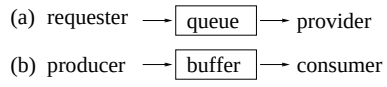

Figure 1: Two different models for DPM.

In this paper, we adopt a different model as shown in Figure 1 (b). This model considers a producer and a consumer. The main difference is that the producer-consumer model allows prefetching, a commonly used technique for improving performance. In contrast, the provider is reactive and cannot prefetch before the requester sends a request. For example, a video player may prefetch many frames and store them in the buffer. The consumer retrieves the frames and displays them on the screen and meets the timing constraints (30 frames per second). If the producer is a hard disk, it may be shut down to save power when sufficient frames are stored in the buffer. Before the buffer empties, the disk is turned on (also called "awakened") to fill the buffer. In order to amortize the overhead of turning on the disk, it is preferred to store more data in the buffer so that the disk can remain idle longer. However, this requires a larger buffer memory. When the power consumption of the buffer memory is considered, a large buffer may no longer be advantageous. Hence, we need to find an optimal buffer size that is large enough to amortize the overhead of DPM but small enough to actually save power.

This paper presents a new approach for DPM using data buffers. In our previous work [3], we calculate the optimal buffer sizes for constant producing rates and consuming rates. This paper extends the previous work in two ways: (a) This paper considers uncertain consuming rates. Because of this uncertainty, the buffer may underflow and fail to satisfy the consumer. (b) This paper considers the delay of DPM. Because of the time needed to awaken a disk, it should be awakened early enough to prevent buffer underflow, a violation of the timing constraints of the consumer. We assign an energy penalty whenever underflow occurs and use this penalty to quantify the quality of service (QoS). A major contribution of this paper is providing an analytic model for computing the optimal buffer sizes that includes the essential parameters. Unlike existing heuristics, it can precisely determine the effects of parameter changes. We

ISLPED'04, August 9–11, 2004, Newport Beach, California, USA.

use a disk as the example of the producer in this paper but our method is applicable to any two interacting components with a buffer between them. Our approach is based on the concept of inventory management in operations research [5]. We demonstrate the close relationship between power management and inventory management by modeling the buffer's power as the holding cost in a warehouse. To demonstrate the effectiveness of our approach, we use video prefetching as case studies and obtain 74.5% power savings for MPEG-1 and 34.8% for MPEG-2 videos.

2. RELATED WORK

DPM methods have been proposed to manage different types of components, such as processors [8, 9, 11], hard disks [7, 14] and wireless network interface cards [15]. The basic concept of DPM is predicting the time when a component is idle or under-utilized. Then, the component is shut down or its performance is scaled down. The main differences among existing methods are their approaches to predict these idle or under-utilized periods. Scheduling [8, 9, 11] arranges the execution time of tasks to keep processors at low-power states (i.e. lower frequencies) while meeting all timing constraints. Hwang et al. [7] use the lengths of recent idle periods to predict the length of future idle periods. If the predicted length is longer than the break-even time of the component being managed, the component is shut down. The break-even time of a component is the minimum length of an idle period to save this component's power [1]. Ramanathan et al. [14] present an adaptive algorithm that improves prediction by assuming the lengths of idle periods follow a distribution function. Simunic et al. [15] use time-indexed semi-Markov models to predict the idle periods of wireless network interface cards. Qiu et al. [13] use continuous-time Markov models so that DPM decisions can be made asynchronously. Chung et al. [4] present a DPM strategy that can handle idle periods which do not follow the same stationary distributions. Some methods use data buffers to create and prolong idle periods. For example, Im et al. [8] use buffers to exploit the slack time when multimedia applications do not use the worst-case execution time. When the processor has slack time, the processor fills the buffers so that the processor's frequency and voltage can be scaled down later. Lu et al. [10] use buffers to smoothen the variations of MPEG frames' execution time. Their method constructs a graph whose vertices represent the buffers' status and the processor's frequency. They find a sequence of the vertices to minimize the sum of the frequencies. Existing methods, however, do not consider the power consumed by the buffers and can provide only heuristic rules to estimate the required buffer size. Our analytic approach can calculate the buffer sizes for optimal energy savings.

3. PREFETCHING FOR DPM

3.1 Problem Description

Figure 2 shows the basic problem in power management using a data buffer. The producer stores data into the buffer at rate α (MB/s) and the consumer removes data from the buffer at rate β (MB/s). During t_1, the amount of data stored in the buffer grows at rate $\alpha - \beta$. At the end of t_1, the producer is shut down. Totally, αt_1 MB are produced; let S be this amount. Let Q be the maximum amount of data stored in the buffer. The consumer continues removing data so the amount of data in the buffer declines at rate β. It takes t_2 to remove all the data in the buffer. The producer is turned on at the end of t_2 to restart producing. Our goal is to find the values of S, Q, t_1, and t_2 to achieve the maximum power savings of the producer. We assume that $\alpha > \beta$; otherwise, buffer underflow occurs.

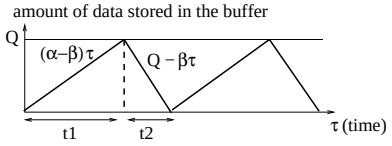

Figure 2: The amount of data stored in the buffer.

There are three components consuming power: the producer, the buffer, and the consumer. Since the consumer continues removing the data from the buffer, its power consumption is not affected by DPM. We assume the buffer's power consumption is proportional to the amount of data stored in the buffer. This assumption is valid when unused portions of memory are turned off [6, 12]. In their work, unused banks of memory or cache lines enter a sleep state to reduce power. As a result, the power consumption is proportional to the amount of data stored in the memory.

3.2 Uncertain Consuming Rate

Suppose the production rate, α, is a constant. This assumption is valid because α represents the bandwidth of the producer, for example, a data bus. Our previous work [3] assumes a constant consuming rate β and no awakening delay. In reality, the consuming rate may vary and the awakening delay may be significant. Let λ be the awakening delay for the producer. We assume that λ is a constant; it is a component-specific parameter, ranging from several milliseconds for a processor to a few seconds for a hard disk. For an uncertain consuming rate, β is a random variable. Suppose $\psi(\beta)$ is the density function of β and $\Psi(\beta)$ is the distribution function, $\Psi(\beta) = \int_0^\beta \psi(\eta)d\eta$. Let γ be the expected value of β, $\gamma = \int_0^\infty \psi(\eta)\eta d\eta$. We assume that $\alpha > \gamma$; otherwise, the buffer will eventually underflow.

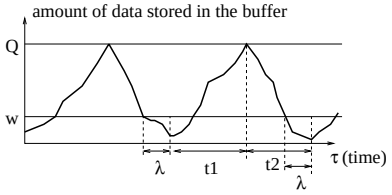

Figure 3: Buffer storage for random variable β.

Figure 3 shows the strategy when β is uncertain and λ is nonzero. The producer is awakened when there are still w MB of data in the buffer. The value of w is called the *awakening point*. After λ, the producer starts producing data at rate α and stores the excess data into the buffer. We assume that $\lambda < t_2$. Suppose S MB of data are produced, $S = \alpha t_1$. In steady-state operation, the consumer removes all prefetched data in a period. The expected length of a period is $\frac{S}{\gamma}$, so we obtain $\frac{S}{\gamma} = t_1 + t_2$. When the producer starts producing at the beginning of t_1, the expected amount of data in the buffer is $w - \lambda\gamma$. The producer fills the buffer

at the expected rate of $\alpha-\gamma$. At the end of t_1, the expected amount of data in the buffer is $Q = w - \lambda\gamma + (\alpha-\gamma)t_1$. The average amount of data stored in the buffer during t_1 is $\frac{(w-\lambda\gamma)+(w-\lambda\gamma+(\alpha-\gamma)t_1)}{2} = \frac{2(w-\lambda\gamma)+(\alpha-\gamma)t_1}{2}$. A similar analysis shows that the average amount of data stored in the buffer during t_2 is also $\frac{2(w-\lambda\gamma)+(\alpha-\gamma)t_1}{2}$.

3.3 Energy Penalty for Underflow

A large w keeps more data in the buffer after λ so more energy is consumed by the buffer. On the other hand, a smaller w increases the probability of buffer underflow. Buffer underflow means that the consumer cannot obtain the data it needs. For video prefetching, buffer underflow means disrupted frames. Hence, w should be sufficiently large to prevent underflow. We quantify the quality of service (QoS) by assigning an energy penalty when buffer underflow occurs. Let ρ be the energy penalty for each MB of underflowed data. This value represents the perceived reduction in service; it is a subjective measure dependent upon applications. If underflow is unacceptable, we can choose a very large value for ρ. The advantage of using an energy penalty for QoS is that we can explicitly trade-off between the energy consumed by the buffer and buffer underflow. The penalty occurs only if the consumed data exceeds w, i.e. $\lambda\beta > w$, over the interval λ. The amount of overflow is the data consumption in excess of w multiplied by the probability of data consumption at that level. Let $f(w)$ be the penalty energy: $f(w) = \rho\int_{\frac{w}{\lambda}}^{\infty}\psi(\eta)(\lambda\eta - w)d\eta$.

3.4 Optimal Buffering

The total expected energy consumption in a period is the sum of the energy consumed by the producer: $p_p t_1 + k$, the buffer: $p_b\frac{2(w-\lambda\gamma)+(\alpha-\gamma)t_1}{2}(t_1+t_2)$, and the underflow penalty: $f(w)$. The expected length of a period is $t_1 + t_2 = \frac{S}{\gamma}$. Also, $S = \alpha t_1$. The expected power consumption in one period is $\frac{p_p t_1 + k}{t_1+t_2} + \frac{p_b\frac{2(w-\lambda\gamma)+(\alpha-\gamma)t_1}{2}(t_1+t_2)}{t_1+t_2} + \frac{f(w)}{t_1+t_2}$. Therefore, the expected power consumption p_e in one period can be rewritten as

$$\begin{aligned} p_e &= \frac{p_p\frac{S}{\alpha} + \frac{k}{\frac{S}{\gamma}} + \frac{f(w)}{\frac{S}{\gamma}} + p_b\frac{2(w-\lambda\gamma)+(\alpha-\gamma)\frac{S}{\alpha}}{2}}{} \\ &= \frac{p_p\gamma}{\alpha} + \frac{k\gamma+f(w)\gamma}{S} + p_b(w-\lambda\gamma) + \frac{p_b(\alpha-\gamma)S}{2\alpha} \end{aligned} \quad (1)$$

We need to determine two variables, S and w, to minimize the average power in each period. We take the partial derivatives with respect to S and w and set them to zero:

$$\frac{\partial p_e}{\partial w} = p_b - \frac{\rho\gamma\int_{\frac{w}{\lambda}}^{\infty}\psi(\eta)d\eta}{\lambda S} = 0 \quad (2)$$

$$\frac{\partial p_e}{\partial S} = \frac{p_b(\alpha-\gamma)}{2\alpha} - \frac{k\gamma+f(w)\gamma}{S^2} = 0 \quad (3)$$

The condition to have a minimum average power is $\frac{\partial^2 p_e}{\partial S^2}\frac{\partial^2 p_e}{\partial w^2} - (\frac{\partial^2 p_e}{\partial S\partial w})^2 > 0$; otherwise, the solutions S and w represent a saddle point of p_e. We can simplify the expression as $2\frac{k\gamma+f(w)\gamma}{S^3}\frac{\rho\gamma\psi(w)}{\lambda^2 S} - (\frac{\rho\gamma\int_{w/\lambda}^{\infty}\psi(\eta)d\eta}{\lambda S^2})^2 = \frac{\rho\gamma^2}{\lambda^2 S^4}[2\psi(w)(k+f(w)) - \rho(\int_{\frac{w}{\lambda}}^{\infty}\psi(\eta)d\eta)^2]$. Because this is not necessarily positive, a minimum average power may not exist. Suppose a minimum power does exist. Let S^* and w^* be the solutions for this minimum average power. Because $Q = w^* - \lambda\gamma + (\alpha-\gamma)t_1$ and $\alpha t_1 = S^*$, we can also compute the required buffer size Q^*. The derivation is similar to finding the optimal reorder points as explained in [5], where ordering cost, holding cost, and shortage cost are analogous to producer power, buffer power, and underflow penalty, respectively.

$$\int_{\frac{w^*}{\lambda}}^{\infty}\psi(\eta)d\eta = \frac{p_b\lambda S^*}{\rho\gamma} \quad (4)$$

$$S^* = \sqrt{\frac{2\alpha\gamma(k+\rho\int_{\frac{w^*}{\lambda}}^{\infty}\psi(\eta)(\lambda\eta - w^*)d\eta)}{p_b(\alpha-\gamma)}} \quad (5)$$

$$Q^* = w^* - \lambda\gamma + (\alpha-\gamma)\frac{S^*}{\alpha} \quad (6)$$

3.5 Analysis

The density function $\psi(\beta)$ is non-negative, so $\int_{\frac{w}{\lambda}}^{\infty}\psi(\eta)d\eta$ decreases as w grows. From (4), we can see that when p_b increases, w^* should decrease. This is intuitive: when p_b is large, the buffer memory consumes more power. Thus, w^* should be small to keep the amount of data in the buffer lower. When ρ increases, w^* also increases to reduce the probability of underflow. When γ is large, the consumer removes data from the buffer quickly. As a result, w^* should be larger. Because $\int_{\frac{w^*}{\lambda}}^{\infty}\psi(\eta)d\eta \le 1$, $\frac{p_b\lambda S^*}{\rho\gamma}$ must not exceed one. This is a restriction of ρ. The value of ρ should be sufficiently large as a "penalty."

4. EXPERIMENTS

We use video prefetching as our case study. An IBM 1GB microdrive acts as a producer that generates data for a Micron 64MB SDRAM buffer. The parameters are extracted from manufacturer datasheets and physical measurements in our previous work [3] as shown in Table 1. We assume the average consumption rate is 1.5 Mbps (0.1875 MB/s) for MPEG-1 video with a uniformly distributed variation of ±50%. Using formula (4), the minimum value of ρ is 0.035 J/MB. We choose 10 J/MB for ρ to indicate small tolerance of data underflow. Using (4) and (5), we can find the optimal values of S^*, w^*, and Q^* as 2.29 MB, 44.6 kB, and 2.06 MB, respectively.

λ	0.238 s	α	1.824 MB/s	p_p	0.221 W
ρ	10 J/MB	γ	1.5 Mbps	p_b	0.012 W/MB
k	0.151 J				

Table 1: System parameters.

Figure 4 compares the power consumption without prefetch and with buffered prefetch. Without prefetch, the microdrive is kept on continuously and consumes constant power 0.221 W. With the buffer, the power varies. As the buffer is filled from the producer, the power consumption increases to point A because more power is consumed as data fills the buffer. Subsequently, the energy consumption exceeds that of the unbuffered system. At point A, the disk is turned off, considerably reducing the power dissipation. The power gradually decreases to point B as the buffered data is consumed. When the buffer reaches its awakening point at point B, the power dissipation increases for a short time due to the overhead associated with spinning up the hard disk. At this point, one period has been completed, and the power savings between the two systems can be evaluated. The buffered system consumes 74.5% less average

power than the unbuffered system. The percentage of power savings is defined as the ratio of the reduced power consumption and the producer's (i.e. the microdrive's) static power. The power savings are calculated by assuming we can turn off each byte of the buffer memory. If we use 256 KB as the unit of memory, the power savings are slightly lower, 73.6%. This is because 256 KB of memory is turned on even when we need only one more byte. We can also vary the consumption rate to determine the power savings in different situations. For example, the percentage power savings decreases as the consumption rate increases. When we increase the rate from 0.1875 MB/s for MPEG-1 video to 0.75 MB/s for MPEG-2, the power savings are reduced from 74.5% to 34.8%. The S^*, w^*, and Q^* values for MPEG-2 are 5.66 MB, 178.6 kB, and 3.33 MB, respectively.

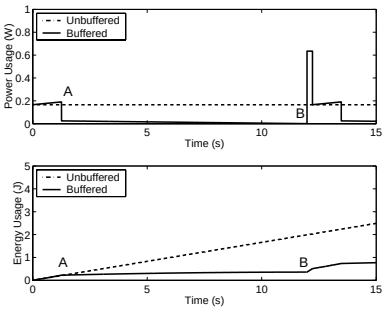

Figure 4: Comparison of power and energy in buffered and unbuffered systems.

We now compare the power consumption of our method against heuristically determined buffer sizes. We compare the buffer sizes of 512 kB, 1 MB, 2 MB, 4 MB, and 8 MB. The awakening points are also heuristically determined as 8 kB, 16 kB, 32 kB, 64 kB, or 128 kB. Figure 5 depicts the savings in average power for each buffer size with respect to an unbuffered system as the awakening point varies. In this figure, a higher curve is better because it indicates more power savings. In all cases, the power savings are lower than the optimal buffer. Two general trends are observed in this figure. When the awakening point is less than the optimal w^*, power savings can be achieved by increasing the buffer size, thereby amortizing the effects of underflow and awakening overhead. For awakening points higher than the optimal value, we notice that the buffer size has little effect upon average power savings, as the 1-8 MB buffers converge to within 10% of each other. The 512 kB buffer saves much less power due to the shorter period $(t_1 + t_2)$ and high overhead. Increasing the buffer size does not necessarily increase power savings when the awakening point is too large.

5. CONCLUSION

This paper presents a method for reducing power consumption by adding data buffers between a producer and a consumer. The method calculates optimal buffer sizes for probabilistic rates of data consumption and application-specific penalty of buffer underflow. The power savings are compared to an unbuffered system and heuristically sized buffers. The results demonstrate over 74% reduction for MPEG-1 video and 34% for MPEG-2 video.

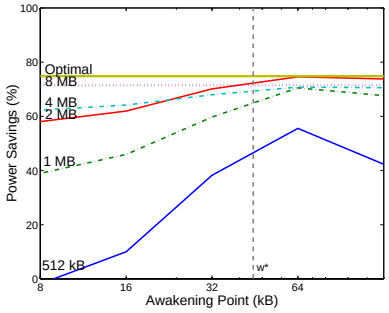

Figure 5: Power savings for heuristic sizes as compared to an unbuffered system.

6. ACKNOWLEDGMENTS

Nathaniel Pettis is supported by the GAANN Fellowship. Le Cai is supported by the Purdue Research Foundation.

7. REFERENCES

[1] L. Benini, A. Bogliolo, and G. D. Micheli. A Survey of Design Techniques for System-Level Dynamic Power Management. *IEEE Trans. VLSI*, 8(3):299–316, June 2000.

[2] L. Benini, A. Bogliolo, G. A. Paleologo, and G. D. Micheli. Policy Optimization for Dynamic Power Management. *IEEE Trans. CAD*, 18(6):813–833, June 1999.

[3] L. Cai and Y.-H. Lu. Dynamic Power Management Using Data Buffers. In *DATE*, pages 526–531, 2004.

[4] E.-Y. Chung, L. Benini, A. Bogliolo, Y.-H. Lu, and G. D. Micheli. Dynamic Power Management for Nonstationary Service Requests. *IEEE Transactions on Computers*, 51(11):1345–1361, November 2002.

[5] F. S. Hillier and G. J. Lieberman. *Introduction To Stochastic Models in Operations Research*. McGraw Hill, 1990.

[6] Z. Hu, S. Kaxiras, and M. Martonosi. Let Caches Decay: Reducing Leakage Energy Via Exploitation of Cache Generational Behavior. *ACM Transactions on Computer Systems*, 20(2):161–190, May 2002.

[7] C.-H. Hwang and A. C.-H. Wu. A Predictive System Shutdown Method for Energy Saving of Event-driven Computation. *ACM TODAES*, 5(2):226–241, April 2000.

[8] C. Im, H. Kim, and S. Ha. Dynamic Voltage Scheduling Technique for Low-power Multimedia Applications Using Buffers. In *ISLPED*, pages 34–39, 2001.

[9] T. Ishihara and H. Yasuura. Voltage Scheduling Problem for Dynamically Variable Voltage Processors. In *ISLPED*, pages 197–202, 1998.

[10] Y.-H. Lu, L. Benini, and G. D. Micheli. Dynamic Frequency Scaling With Buffer Insertion for Mixed Workloads. *IEEE Trans. CAD*, 21(11):1284–1305, November 2002.

[11] J. Luo and N. Jha. Static and Dynamic Variable Voltage Scheduling Algorithms for Real-time Heterogeneous Distributed Embedded Systems. In *Asia and South Pacific Conference on VLSI Design*, pages 719–726, 2002.

[12] J. Pisharath and A. Choudhary. An Integrated Approach To Reducing Power Dissipation in Memory Hierarchies. In *CASES*, pages 89–97, 2002.

[13] Q. Qiu and M. Pedram. Dynamic Power Management Based on Continuous-time Markov Decision Processes. In *DAC*, pages 555–561, 1999.

[14] D. Ramanathan and R. Gupta. System Level Online Power Management Algorithms. In *DATE*, pages 606–611, 2000.

[15] T. Simunic, L. Benini, P. Glynn, and G. D. Micheli. Dynamic Power Management for Portable Systems. In *MobiCom*, pages 11–19, 2000.