2003

Stochastic Learning Feedback Hybrid Automata for Power Management in Embedded Systems

Pushkin Kachroo
University of Nevada, Las Vegas, pushkin@unlv.edu

S. K. Shukla

T. Erbes

H. Patel

SMCia/03
IEEE International Workshop on Soft Computing in Industrial Applications
Binghamton University, Binghamton, New York, June 23-25, 2003

Stochastic Learning Feedback Hybrid Automata for Power Management in Embedded Systems

Pushkin Kachroo Sandeep K. Shukla Teodora Erbes Hiren Patel

Bradley Department of Electrical & Computer Engineering
Virginia Tech., Blacksburg, VA 24061-0111
{pushkin,shukla,terbes,hiren}@vt.edu

Abstract - In this paper we show that stochastic learning automata based feedback control switching strategy can be used for Dynamic Power Management (DPM) employed at the system level. DPM strategies are usually incorporated at the operating system of embedded devices to exploit multiple power states available in today's ACPI compliant devices. The idea is to switch between power states depending on device usage, and since device usage times are not deterministic, probabilistic techniques are often used to create stochastic strategies, or strategies that make decisions based on probabilities of device usage spans. Previous work [1] has shown how to approximate the probability distribution of device idle times and dynamically update them, and then use such knowledge in controlling power states. Here, we use stochastic learning automaton (SLA) which interacts with the environment to update such probabilities, and then apply techniques similar to [1] to optimize power usage with minimal effect on response time for the devices.

I. INTRODUCTION

The technical focus of the work presented here is on strategies that can be employed by the operating system, networking software and to some extent application software in effectively managing power usage. Due to constraints on power budgets of electronic systems, Dynamic Power Management (DPM) is increasingly gaining importance, as evidenced in the research literature [6][8][2][3][14][4][23][12], as well as concerted industry efforts such as Microsoft's OnNow [9] and ACPI [10]. Significant work has been done in the area of DPM in search of strategies that yield the most reduction in power/energy with the least amount of runtime computational effort. These include heuristic shutdown policies [5], prediction based shutdown policies [6][8], multiple voltage scaling [2], and stochastic modeling based policy optimization [18][11]. Most strategies are designed using experimental intuition and are then experimentally evaluated to be effective. In our work, we are trying to create a foundation on which to build strategies in a more systematic manner. As discussed in [3], the design and analysis of power management policy optimization techniques are still wide open fields of research. Our framework for analysis is that of design of closed loop dynamic control by treating this problem as a stochastic feed back control design problem.

Industry initiatives such as ACPI [10] and OnNow [9], have tried to develop standard APIs for operating system developers to access power state information of devices which are endowed with such states, and to change them under the operating system control. ACPI also develops specification for different classes of devices, how many power states such devices should have and their general characteristics. For example, for storage class devices (e.g. disk drives), ACPI prescribes 4 inactive power states. Before power became a first class parameter for system design, devices were manufactured with only ON and OFF states. However, bringing devices back from OFF state to ON state causes penalties in terms of delay and energy. Hence, unless the devices are idle for a long enough time, turning off a device does not result in power savings. In fact, it may result in energy loss, and unwarranted delay and reduction in system throughput with negative or no gain in battery life. Introduction of various kinds of shutdown power states by ACPI standards led the manufacturers to develop devices with multiple power states. For example, most disk drives manufactured today have 4 shutdown power states. Whereas ACPI does a good job of classifying the system devices into various classes (e.g., Storage class, Audio Device class, Communication Device class etc.), they do not prescribe how to optimally use these power states to

0-7803-7855-5/03/$17.00 © 2003 IEEE

enable the system to reduce energy expenditure while not compromising performance. That is left to the designers of the operating system components and services for power management. *Since this is a multi-dimensional optimization problem and the pattern of device usage by applications running on the system are unknown a priori, these lead to on-line optimization problems.*

While adjusting (or scheduling) ON or OFF states of a device with the goal of minimizing energy usage is a well-studied problem, formal understanding and scheduling techniques for device power management with multiple operating *and* shutdown states is an ongoing research problem. Moreover, devices can also have multiple active power states. For example, most CPUs including Intel's most recent mobile processor code named BANIAS has multiple frequency and voltage settings which can be changed under O/S control. A communication class device may have multiple power levels to transmit data on wireless links, and different such levels may be selected based on the criticality of the communication etc. Devising power management strategies for such devices (with multiple active states and multiple shutdown state) which optimize energy expenditure without undue cost on performance is an even more challenging problem.

One way to address this issue is to characterize typical input sequences by a probability distribution and optimize the online algorithm for that distribution. This has the advantage that algorithms are tuned for the input sequences that are thought to be more typical of those arising in the given application. Of course, the success of this method depends on the ability to accurately describe the input sequence by a probability distribution. When the input sequence characteristic changes, one must be able to switch to a different control strategy. However, discovering dynamically that the probability distribution has changed by itself is a challenge, which was taken into consideration in [1]. An approximation strategy for input sequence probability distribution and dynamic updating of the approximation was done there. A probability based control strategy that makes use of this approximate probability distribution was formulated. This paper provides an alternative approach to this inference and/or approximation of the probability distribution via SLAs.

II. OTHER RELEVANT WORK

There is an extensive body of research [11][17][18][19] on the probabilistic formulation of the DPM problem in which the power state changes are modeled by Markov/Semi-Markov processes and the input request arrivals are modeled by well known distributions (e.g. Binomial or Poisson, and inter-arrival times are then distributed as Geometric, Exponential etc). The formulation is then turned into an optimization problem, which is then solved to obtain system parameters to tune the DPM strategy. However, it is a strong assumption that inter-arrival times are distributed according to some special case distribution, especially when the arrivals are very much application dependent. This concern was addressed in [19] in which a method was developed that was independent of the distribution generating the idle period length. Unfortunately, this approach requires solving a large instance of linear programming at every idle period, which is clearly impractical for runtime determined in embedded systems.

An online DPM strategy can be characterized by a threshold on the idle time interval for transitioning from the active to the sleep state. For multiple state systems, there is a sequence of thresholds each of which indicates when to transition to the next lower power consumption state. Previous work on prediction based dynamic power management can be categorized into two groups: adaptive and non-adaptive. A non-adaptive strategy sets its thresholds once and for all and does not alter them. Adaptive strategies, on the other hand, use the history of idle periods to guide the decisions of the algorithm for future idle periods. There have been a number of adaptive strategies proposed in the literature [6][7][8][13][15][16]. Many adaptive dynamic power management strategies use a sequence of past idle period lengths to predict the length of the next idle period. These strategies typically describe their prediction for the next idle period with a single value. Given this prediction, they transition to the power state that is optimal for this specific idle period length. In the case that their prediction is wrong, they transition to the lowest power state if the idle period extends beyond a fixed threshold value. For the sake of comparison with other approaches, we shall call these predictive DPM schemes single-value prediction schemes (SVP). In particular, Chung, Benini and De Micheli [16] address multiple idle state systems using a prediction scheme based on adaptive learning trees. Their method has an impressively high hit ratio in its prediction.

One of the chief limitations of the single-valued prediction approach is that it fails to capture uncertainty in the prediction for upcoming idle period length. For example, if a very short idle period and a very long idle period are equally likely, these methods are forced to pick a single prediction, and pay a penalty in the likely event that the prediction is wrong. We address this limitation by using a probability distribution to describe the upcoming idle period. The distribution allows for a much richer prediction so that the algorithm can optimize in a way that takes the nature of this additional information into account.

The idea of modeling idle period lengths by a probability distribution leads to two distinct questions:

- If the upcoming idle period length will be generated according to a probability distribution known to the algorithm, how can this information be used to optimize power consumption?

Embedded devices conforming to ACPI with multiple modes of operation for power management purposes can be modeled the as follows. Let us say that there are n different power modes an embedded system can be in. There is a running cost associated with operating in various modes. There is also a cost associated with switching from one state to the

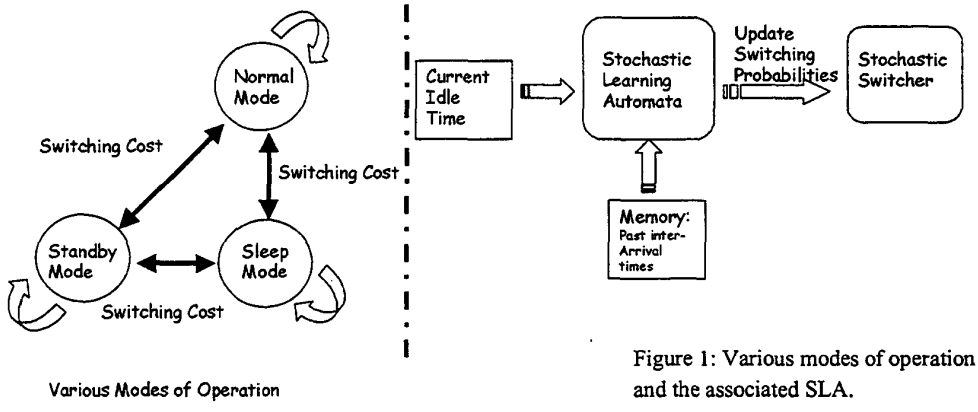

Figure 1: Various modes of operation and the associated SLA.

- Given historical data for previous idle periods, how can we use this information to reliably construct a probability distribution describing future idles periods? Note that, this probability distribution changes dynamically depending on applications running on the system. Hence, this probability distribution needs to be learned dynamically as well.

III. OUR APPROACH

In [12], we have addressed the first question, and devised strategy for DPM, and in [20] we have addressed the second question. In answer to the first question, an algorithm is given which determines thresholds to transition from one state to the next given that the next idle period will be generated by a fixed, known distribution. Analytical as well as empirical results are given which demonstrate that this algorithm does very well when the idle periods follow an a priori known probability distribution. In the second work, we developed a method to use recent history to form an estimate for a distribution governing future usage patterns for the device. Here we revisit them again with a different control theoretic approach as follows:

other. The set of states is a linear ordered set, the "normal set" being the one that is the largest member (implying that it has the highest power consumption). The modes are ordered on the basis of the power consumption. The cost of switching in general is not symmetric, since the cost to switch from A to B might not be same as from B to A. Also, the cost of switching is generally proportional to how many states are between two states in the ordering of the sets. As an example we consider an embedded system with three states, namely, normal, standby and sleep states.

The overall problem statement can be stated as follows. The system dynamics are given in terms of a timed automaton as:

$$\begin{aligned} q^{\#} &= \psi(q, i(q,x)) \\ \dot{x} &= 1 \\ \rho(q,x) &= 0 \end{aligned} \qquad (1)$$

These dynamics show the discrete transition of the discrete state q that can take three values corresponding to the three modes of operation; they show the clock as a continuous variable in terms of a stop watch, since it gets reset by the jump transition during every switch. The

problem formulation is in terms of designing a feedback controller $i(q,x)$ that attempts to keep the following cost objective low.

$$\min J = Lt_{T\to\infty}\frac{1}{T}\int_0^T C(q,t)dt + Lt_{N\to\infty}\frac{1}{N}\sum_{i=0}^{N} S_i \quad (2)$$

The objective shows that the total cost in an infinite execution is obtained by tracking all the cost of staying in any states for some time, as well as by including the costs associated with all the switching involved.

We show in this paper a stochastic learning automata based feedback control switching strategy that is aimed at keeping the overall average cost low. The scheme is shown on the right hand side of Figure 1. The stochastic learning automaton (SLA) interacts with the environment to update its probabilities. The actions are based on probabilities of various switches that are updated by the response from the environment. The response from the environment is quantified in terms of how much time elapsed before a new data (or program) to be served comes into the system. The SLA can use all or some of the old inter-arrival values/patterns for updating the action probabilities.

We combine the action probabilities scheme of the SLA with the timed automaton model of the system to obtain an overall closed loop stochastic learning feedback hybrid automaton for the problem. We study various SLA schemes in this framework such as reward-penalty and reward inaction in terms of performance of the overall system, and also in terms of verifiable properties, such as liveness and safety.

The SLA control design as presented above is based on a heuristic strategy motivated from the connections of SLA to mathematical psychology. However, the aim of achieving (2) using SLA design can be made more analytic by viewing the SLA feedback design as a solution of a value iteration or policy iteration scheme of dynamic programming [21][22]. This detailed relationship will be pursued in a future paper.

The SLA Design

The SLA design architecture is illustrated using the following example, which is a restricted version of the general case.

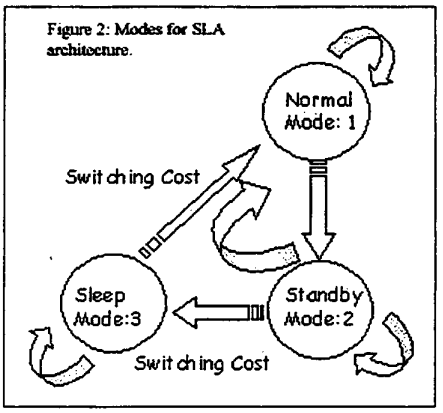

Figure 2: Modes for SLA architecture.

In this example there are three modes: normal, standby and sleep. They are numbered 1, 2 and 3 as shown. The control design is a randomized algorithm in the sense that when the system is in any state, there are probabilities of switching to through the allowable transition curves. These probabilities are updated when a new service request arrives, or in other words, the last idle time interval is known. The probabilities can be updated using various SLA schemes of reward-penalty, reward-inaction, etc. The general update law for the probabilities is given below.

$$p_{ij}^{\#} = p_{ij} + f_{ij}(p_{ij}, T, S) \quad (3)$$

Equation (3) shows that the updated probability from mode i to j is a function of the previous value of the probability, the latest idle time T, and a vector S that contains all the times spent in each mode during that cycle. After the idle time is over, we can compare the components of vector S with the values of another vector S_o that contains the times that would be spent in each mode if we used an "offline" optimal controller that would know the actual idle times beforehand. The probabilities are updated based on the difference between S and S_o. For instance, let us say that the time spent in mode 1 was less than the corresponding time of the offline optimal controller, then the updated probability from mode 1 to other modes will have a positive increment that can be a linear function of the time difference. More detailed description of this SLA and the experimental results will be described in the journal version.

Reference

[1] S. Irani, S. Shukla and R. Gupta, " Competitive Analysis for Dynamic Power Management for systems with Multiple Power Saving States", UCI-ICS Technical report 01-50, September 2001.

[2] L. Benini and G. De Micheli, "*Dynamic Power Management: Design Techniques and CAD Tools*", Kluwer Academic Publishers, 1998.

[3] L. Benini, G. De Micheli, and E.Macii, "*Designing low-power circuits: practical recipes*", IEEE Circuits and Systems Magazine, 1(1):6--25, Mar. 2001.

[4] D. Ramanathan, S. Irani, and R. K. Gupta, "*Latency Effects of System Level Power Management Algorithms*", in Proceedings of the IEEE International Conference on Computer-Aided Design, Nov. 2000..

[5] D. Ramanathan, "*High-Level Timing and Power Analysis for Embedded Systems*", PhD thesis, University of California at Irvine, Sept. 2000.

[6] Chi-Hong Hwang, C. Allen and H. Wu, "*A Predictive System Shutdown Method For Energy Saving of Event-Driven Computation*", in Proceedings of the IEEE/ACM International Conference on Computer Aided Design, pages 28--32, Nov. 1996.

[7] Karlin, M. Manasse, L. McGeoch, and S. Owicki, "*Randomized competitive algorithms for non-uniform problems*", in 1st Annual ACM-SIAM Symposium on Discrete Algorithms, pages 301-309, Jan. 1990.

[8] M. B. Srivastava, A. P. Chandrakasan, and R. W. Broderson, "*Predictive Shutdown and Other Architectural Techniques for Energy Efficient Programmable Computation*", IEEE Trans. on VLSI Systems, 4(1):42--54, Mar. 1996.

[9] Microsoft, "*OnNow Power Management Architecture for Applications*", available at http://www.microsoft.com/hwdev/pcfuture/onnowapp.HTM, Aug. 1997.

[10] Intel, Microsoft, and Toshiba, "*Advanced Configuration and Power Interface specification*", available at http://www.intel.com/ial/powermgm/specs.html, Dec. 1996.

[11] Q. Qiu, Q. Wu, and M. Pedram, "*Stochastic Modeling of Power Managed Systems: Construction and Optimization*", in Proceedings of the International Symposium of Low Power Electronics and Design, pp. 194-199, Aug. 1999.

[12] S. Irani, S. Shukla, R. Gupta, "Competitive Analysis of Dynamic Power Management Strategies for Systems with Multiple Power Saving States", In proceedings of Design Automation and Test, Europe, Conference, March 2002.

[13] Ramanathan, S. Irani, R. Gupta, "An Analysis of System Level Power Management Algorithms and their effects on Latency", Accepted to be published at the IEEE Transactions on Computer Aided Design.

[14] S. K. Shukla and R. K. Gupta, "A Model Checking Approach to Evaluating System Level Power Management Policies for Embedded Systems", In the Proceedings of IEEE Conference on High Level Design Validation, 2001.

[15] L. Benini, A. Bogliolo, G. Paleologo, and G. D. Micheli. "Policy Optimization for Dynamic Power Management", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, 18(6): 813-833, 1999.

[16] E. Y. Chung, L. Benini, and G. D. Micheli, "Dynamic Power Management using Adaptive Learning Tree", In proceedings of ICCAD, 1999.

[17] Y. Lu, E. Chung, T. Simunic, L.Benini, and G. De Micheli, "Quantitative Comparison of Power Management Algorithms", Proceedings of Design Automation and Test, Europe, 2000.

[18] Q. Qiu, and M. Pedram, "Dynamic Power Management Based on Continuous Time Markov Decision Processes", In Proceedings of the Design Automation Conference (DAC), 1999.

[19] Tajana Simunic: "Energy Efficient System Design and Utilization", Ph.D Thesis, Stanford University, 2001.

[20] S. Irani, S. Shukla, R. Gupta, "Online Strategies for Dynamic Power Management in Systems with Multiple Power Saving States", ACM Transactions on Embedded Computing Systems, special issue on Power Aware Embedded Computing, to appear.

[21] Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 1995.

[22] Bertsekas and Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, 1996.

[23] E.Y. Chung, L. Benini, A. Bogliolo, and G. De Micheli, "*Dynamic Power Management for Non-Stationary Service Request*", In Proceedings of the Design Automation and Test Europe, 1999.