Enhanced Q-Learning Algorithm for Dynamic Power Management with Performance Constraint

Wei Liu, Ying Tan, Qinru Qiu
Department of Electrical and Computer Engineering
Binghamton University, State University of New York
Binghamton, New York 13902, USA

Abstract - This paper presents a novel power management techniques based on enhanced Q-learning algorithms. By exploiting the submodularity and monotonic structure in the cost function of a power management system, the enhanced Q-learning algorithm is capable of exploring ideal trade-offs in the power-performance design space and converging to a better power management policy. We further propose a linear adaption algorithm that adapts the Lagrangian multiplier λ to search for the power management policy that minimizes the power consumption while delivering the exact required performance. Experimental results show that, comparing to the existing expert-based power management, the proposed Q-learning based power management achieves up to 30% and 60% reduction in power saving for synthetic workload and real workload, respectively while in average maintain a performance within 7% variation of the given constraint.

I. Introduction

System level power management must consider the uncertainty and variability that comes from the environment, the application and the hardware. Statically optimized resource and power management are not likely to achieve the best performance when the input characteristics are changing. Good power management controllers should be able to observe, learn and adapt to different hardware systems and different working environments.

Many existing works focus on the stochastic nature of the power management problem [1]~[4]. However, their techniques require offline system modeling and policy optimization and hence are not adaptive. Reference [5] proposes a user-based adaptive management technique that considers user annoyance as a performance constraint. However, this approach requires offline training, which is not suitable in a changing environment.

An on-line learning algorithm in [6] dynamically selects the best DPM policies from a set of candidate policies called experts. Each expert has a weight factor, the value of which indicates the benefit gained if the correspondent expert was chosen during the last idle period. The one with the highest value will control the device for the next idle period. Reference [7] proposes a similar approach using a different learning algorithm. The expert-based machine learning algorithm is able to find an appropriate DPM policy in short time without any prior workload information. However, it cannot explore the power performance trade-offs effectively.

In this paper, we propose a novel approach for system level power management using enhanced Q-learning. The traditional Q-learning algorithm provides a model-free solution for the Markov Decision Problem. It provides provable convergence to the optimal solution in a Markovian environment. Some research works have investigated the feasibility of applying the traditional reinforcement learning to solve the problem in a non-Markovian environment or a partially observable Markovian environment [10][11]. Their results show that Q-learning is capable to achieve the same performance as the other reference learning algorithms at the cost of slower convergence speed. This paper focuses on improving the convergence speed of the Q-learning for the DPM problem.

Scheduling and management of queuing system has been studied in many domains including multiuser wireless communication. It is proven that given a system with a submodular and monotonic cost function, the actions in the optimal policy are nondecreasing to the number of waiting requests [12]. That is, higher processing speed should be used when there are more requests waiting in the system. A structured learning algorithm has been proposed to schedule the packets in a MIMO (multiple input multiple output) wireless communication system [8].

In this paper, we adopt the method in [8] and enhance the performance of traditional Q-learning by exploiting the submodularity and monotonic structure in the cost function of a power management system. The enhancement confines the Q-learning algorithm to search only the policies whose action is none decreasing to the number of waiting requests, which improves the search speed. Based on the enhanced Q-learning algorithm, we further propose a linear adaption algorithm that adapts the Lagrangian multiplier λ to search for the power management policy that minimizes the power consumption while delivering the exact required performance. This property is important to autonomous power management because it ensures that the power management algorithm can adapt itself during the runtime to maintain a constant performance when the workload varies or to meet a changing QoS (quality of service) requirement.

The main characteristics of the proposed learning algorithm are summarized as follows.

1. The power manager does not require any prior knowledge of the workload neither does it depend on any pre-designed experts, or any DPM policies.
2. The search space of the power management controller is restricted. Therefore it converges to a better policy in less time. Compared to the power management using traditional Q-learning, the proposed learning algorithm provides 40% and 90% reduction in power and latency respectively.
3. The power management controller is able to adapt itself to the changing performance constraint during runtime. The experimental results show that it can converge to a policy that delivers just enough performance with minimum power consumption within 50 updates.
4. Experimental results show that, comparing to the existing expert-based power management, the proposed Q-learning based power management achieves up to 30% and 60% reduction in power saving for synthetic workload and real workload respectively.

The outline of the paper is as follows. Section II gives our problem formulation. Section III presents the enhanced Q-learning algorithm and the Lagrangian multiplier updating algorithm. The experimental results and analysis are presented in Section IV. We conclude the work in Section V.

II. Problem Formulation

The power management system is similar to many previous works in stochastic power management, which consists of a service requestor ($\mathcal{SR}$), a service provider ($\mathcal{SP}$), and a service queue ($\mathcal{SQ}$). The service requestor generates different types of requests to be processed by the service provider. These requests are first buffered in a FIFO queue ($\mathcal{SQ}$) before being processed. The state space of the environment is the composite space comprising of $\mathcal{SR}$ states, $\mathcal{SQ}$ states and $\mathcal{SP}$ states.

The power manager (PM) observes the system through a noisy

*This work is supported in part by NSF under grant CNS- 0845947

channel therefore it only receives the partial observation. As reference [4] pointed out, such partial observation can be caused by delayed information or hidden states in $\mathcal{SP}$, $\mathcal{SR}$ or $\mathcal{SQ}$. For example, $\mathcal{SR}$ has many request generation modes which are not distinguishable to the power manager. We use a finite set $\boldsymbol{\mathcal{B}} = \{o_1, o_2, o_3, \ldots, o_n\}$ to denote the set of observations. If the system is Markovian, then the observation is a hidden Markov process.

The object of power management is to minimize the average power consumption with respect to a given performance constraint. The action space can be denoted as a finite set $\boldsymbol{\mathcal{A}} = \{a_1, a_2, a_3, \ldots, a_n\}$, with n representing the number of all power modes that the $\mathcal{SP}$ can switch to. The power manager chooses an action for $\mathcal{SP}$ every time the system leaves one state and enters another. A discrete-time slotted model is used throughout this work, which means all the decision making and system state updating occur on a cycle basis.

III. Enhanced Q-Learning for Policy Learning

We modified the traditional Q-learning from four perspectives to provide better solution for the DPM problem.

A. *Modified Cost Function with Latency Constraint*

Our goal is to optimize the power while maintaining a certain level of overall performance. To extend the Q-learning to address our problem, we define a Lagrangian cost

$$C(s,a;\lambda) = c(s,a) + \lambda d(s,a) \qquad (1)$$

where $c(s,a)$ is the power consumption when action a is taken in state s, and $d(s,a)$ is the delay caused by the action. We set different delay costs for actions go_active and go_sleep that are described as following:

$$d(s, go\_active) = q; \quad d(s, go\_sleep) = q * (1 + T_{tran})$$

Where q is the number of requests in queue and T_{tran} is the average power mode switching time.

B. *Learning in the Observation Domain*

During each cycle, the agent obtains an observation of current system state, which may not be the real system state due to the noisy channel between the agent and the environment. The learner keeps a set of Q-values for each observation-action pair, which is updated in the same way as traditional Q-learning algorithm.

$$Q(o,a;\lambda) = \big(1 - \epsilon_{(o,a)}\big)Q(o,a;\lambda) + \epsilon_{(o,a)}(c(o,a;\lambda) + min_{a'}\, Q(o',a';\lambda)) \quad (2)$$

Next time when the same observation is re-observed, the action with the smallest Q-value is chosen and a new Q-value is again calculated. In another word, each observation cycle, we update the Q value for the observation-action pair of the previous cycle.

C. *Structure in Cost Function to Reduce Search Space*

Before presenting the enhanced Q-learning algorithm, we need to introduce the definition of submodular function, which has been widely used in recent years in combinatorial optimization [13].

Definition 1: A function $f: A \times X \times P \to R$ is submodular in (a, x) if for all $p \in P$, $a' \geq a$, and $x' \geq x$, $f(a',x';p) - f(a,x';p) \leq f(a',x,;p) - f(a,x;p)$ .A policy π is nondecreasing in the $\mathcal{SQ}$ state sq if the index of the action $a = \pi([sp,sq,sr])$ taken in the state $[sp,sq,sr]$ is nondecreasing in $\mathcal{SQ}$ state sq for each $\mathcal{SP}$ state sp and $\mathcal{SR}$ state sr. It is proven that if the cost function $C(sq, sp, sr, a; \lambda)$ is submodular and convex of (sq, a), and if it is also an increasing function of sq, then optimal policy is a nondecreasing function of sq [12]. The original proof is made for transmission scheduling in a wireless communication system. However, it can be extended to our DPM system because of the analogy between these two. This property indicates that with the increase of waiting requests in $\mathcal{SQ}$, the optimal policy should process requests from $\mathcal{SQ}$ at a higher speed, and hence a higher probability to keep $\mathcal{SP}$ active.

Theorem 1. The Q function $Q(sq,sp,sr,a;\lambda)$ of a nondecreasing policy is submodular in the domain of (a, sq), that is:

$$Q(a',sq';sp,sr) - Q(a,sq';sp,sr) \leq Q(a',sq;sp,sr) - Q(a,sq;sp,sr) \quad (3)$$

$\forall a, a' \in \boldsymbol{\mathcal{A}}$, $a' \geq a$, $\forall sq, sq' \in \mathcal{SQ}$, $sq' \geq sq$, $\forall sp \in \mathcal{SP}$ and $\forall sr \in \mathcal{SR}$.

Theorem 1 gives an important property of a nondecreasing policy that can be used to enhance the performance of the traditional Q learning algorithm.

Based on the above analysis, we confine our policy search to the nondecreasing policies whose Q values satisfy equation (3). By reducing the search space, we will find the better policy in less time.

Also, by rewriting the Bellman equation we have the following equation for the general Q-learning:

$$\mathbb{E}_{s'}[c(o,a;\lambda) + \min_{a'} Q(o',a';\lambda) - Q(o,a;\lambda)] = 0 \qquad (4)$$

The goal of the enhanced Q-learning is to search for the policy that satisfies both (3) and (4). To achieve this goal we use Primal-Dual method. First, based on fixed Lagrangian multiplier λ, we form the new Q-function parameterization to guarantee that Q factors are submodular. Let $\varphi_{sq,a}^{sp,sr}$ for $sq = 0, \ldots, \mathcal{SQ}; a = 0, \ldots, A; sp \in \mathcal{SP}$ and $sr \in \mathcal{SR}$ be the linear combination of 4 related Q values as follows.

(1) If $sq = 1, \ldots, \mathcal{SQ}; a = 1, \ldots, A$, then

$$\varphi_{sq,a}^{sp,sr} = Q([sq-1,sp,sr],a) - Q([sq-1,sp,sr],a-1) - \big(Q([sq,sp,sr],a) - Q([sq,sp,sr],a-1)\big).$$

(2) If $sq = 0$ or $a = 0$, then

$$\varphi_{sq,a}^{sp,sr} = Q([sq,sp,sr],a).$$

Meanwhile, we can define the two row vectors as follows:

$$\boldsymbol{Q} = [Q([sq,sp,sr],a)]; \qquad \boldsymbol{\Psi} = \left[\varphi_{sq,a}^{sp,sr}\right]$$

The relation between $\boldsymbol{Q}$ and $\boldsymbol{\Psi}$ subject to the constraint on submodularity of Q factors can be written as below:

$$\boldsymbol{\Psi} = \boldsymbol{Q}\boldsymbol{\mathcal{D}} \geq \boldsymbol{0}$$

where $\boldsymbol{\mathcal{D}}$ is a $A \times \mathcal{SQ} \times \mathcal{SP} \times \mathcal{SR}$ dimensional block diagonal transformation matrix.

Equation (4) holds for any $o \in O, a \in A$ and its next observation state and action pair $o' \in O, a' \in A$. Applying the new row vector $\boldsymbol{Q}$, we rewrite this equation compactly as

$$\mathbb{E}[g(\boldsymbol{Q};i,i')] = 0$$

where $i = sq + (a-1)\cdot\mathcal{SQ} + (sp-1)\cdot A\cdot\mathcal{SQ} + (sr-1)\cdot A\cdot\mathcal{SQ}\cdot\mathcal{SP}$ is the index of state-action pair $([sq,sp,sr],a)$ and i' is the index of next state-action pair $([sq',sp',sr'],a')$ in vector $\boldsymbol{Q}$. $g(\boldsymbol{Q};i,i')$ is a vector-valued function and the lth element of it can be described as

$$g_l(\boldsymbol{Q};i,k) = \begin{cases} 0, & if\ l \neq i \\ c(o,a;\lambda) + \min\limits_{j \in \boldsymbol{Q}_k} \boldsymbol{Q}_j - \boldsymbol{Q}_i & if\ l = i \end{cases}$$

where $\boldsymbol{Q}_i$, is the set of all possible next state-action pairs (o', a').

The Lagrangian of our optimization problem is described as

$$L = f(\boldsymbol{Q};i,i') + \boldsymbol{Q}\boldsymbol{\mathcal{D}}v^T$$

where the gradient function of $f(\boldsymbol{Q};i,i')$ is $g(\boldsymbol{Q};i,i')$ and v is a $A \times \mathcal{SQ} \times \mathcal{SP} \times \mathcal{SR}$ dimensional row vector of Lagrange multipliers for primal-dual method. This problem can be solved by iteratively updating the following formulas [9]:

$$\boldsymbol{Q}^{(t+1)} = \boldsymbol{Q}^{(t)} + \epsilon_{([sq,sp,sr],a)}[g(\boldsymbol{Q}^{(t)};i,i') + v^{(t)}\boldsymbol{\mathcal{D}}] \qquad (5)$$

$$v^{(t+1)} = \max\,[v^{(t)} - \epsilon_{([sq,sp,sr],a)}\boldsymbol{Q}^{(t)}\boldsymbol{\mathcal{D}}, 0] \qquad (6)$$

where $\epsilon_{([sq,sp,sr],a)}$ is the learning rate of our algorithm.

D. *Lagrangian Multiplier Adaption*

The power manager using the enhanced Q-learning algorithm can provide ideal tradeoff between power and performance. This is achieved by tuning the Lagrangian multiplier λ in $C(s, a; \lambda)$. Increasing λ will monotonically increase the performance and

decrease the power saving. However the exact relation between λ and system performance cannot be quantified in advance. For a given performance constraint, we propose an iterative algorithm that adapts the λ to the most suitable value.

Let $D(\pi_\lambda^*)$ denote the average latency of the optimal policy. We know that $D(\pi_\lambda^*)$ is a decreasing function of λ. The following function is used to adaptively update λ per fixed time period.

$$\lambda_{t+1} = \lambda_t + k(D(\pi_\lambda^*) - \tilde{C}) \quad (7)$$

The function calculates the new λ_{t+1} as the summation of previous λ_t and the updating value. In this equation, k is the adapting coefficient which has the most important impact.

Calculate $D(\pi_\lambda^*)$;
If flag = $'not\ converge'$ /* still searching for a suitable λ */
  If $|D(\pi_\lambda^*) - \tilde{C}| > 5\% \cdot \tilde{C}$
    Updating_λ: $\lambda_{t+1} = \lambda_t + k(D(\pi_\lambda^*) - \tilde{C})$;
  Else
    Set flag = $'converge'$;
Else if flag = $'converge'$ /*already find out the suitable λ */
  If $|D(\pi_\lambda^*) - \tilde{C}| > 10\% \cdot \tilde{C}$ /*λ_t not suitable any more*/
    Set flag = $'not\ converge'$;
    $\lambda_{t+1} = \lambda_t + k(D(\pi_\lambda^*) - \tilde{C})$;
End

Figure 1 Algorithm for Lagrangian Multiplier λ Adaption.

Figure 1 gives the linear adaption algorithm. The value of λ converges when the relative difference between the average latency $D(\pi_\lambda^*)$ and the latency constraint $\tilde{C}$ is less than 5%. Once the value of λ converges, we will continue using this policy until the $D(\pi_\lambda^*)$ deviates from $\tilde{C}$ by more than 10% which indicates that the service request pattern has been changed. After that the linear adaption algorithm will start the updating equation again.

Although the adaptive algorithm is a general technique that can be used by any power manager which can explore the performance and power tradeoffs, its feasibility is enabled by properties that are pertinent to the enhanced Q-learning. Firstly, the enhanced Q-learning is capable of providing ideal power-performance tradeoffs. Therefore for a given $\tilde{C}$, we can find a λ that gives performance close to $\tilde{C}$. Secondly, the enhanced Q-learning converges to the optimal policy very quickly.

IV. Experimental Results and Analysis

We evaluate the performance of the enhanced Q-learning algorithm using both synthetic workloads and real workloads. The results are compared against the traditional Q-learning algorithm and the expert-based machine learning algorithm [6]. In the rest of the paper, we refer to these three techniques as enhanced Q-learning, traditional Q-learning and expert-based DPM. Three algorithms are adopted as experts in the expert-based DPM: fixed timeout policy, adaptive timeout policy, and exponential predictive policy. TABLE I shows the characteristics of mentioned policies.

A. Experiment Using Synthetic Workload

In this section, we implement our simulation based on synthesized workload. The requests are generated by an $\mathcal{SR}$ model that has three states, s1, s2, and s3 whose incoming request rates are 0.01, 0.10, and 0.25 respectively. At each cycle, a state has 0.01 probabilities switching to any of the other two states. Note that the power manager does not know the $\mathcal{SR}$ model. Furthermore, since our system is partially observable, we assume that the power manager is not able to distinguish the two modes, s1 and s2, which generate requests at relatively low speeds. Hence, the power manager will receive two observations about $\mathcal{SR}$, one is high workload mode and the other is low workload mode.

The $\mathcal{SP}$ used in the simulation is a hard disk drive (HDD) which has two power modes, Active and Sleep. Its power consumption and switching time are shown in TABLE II. The unit for T_{tran} and T_{be} is cycle. The service rate of $\mathcal{SP}$ is 0.5/cycle. The $\mathcal{SQ}$ can hold up to 5 waiting requests; therefore there are 7 different states of $\mathcal{SQ}$ including empty queue state and overflow state.

TABLE I. **Characteristics of compared policies.**

Policy	Characteristics
Fixed Timeout	Timeout = $3 \times T_{be}$
Adaptive Timeout	Initial timeout = $3 \times T_{be}$, Adjustment = +/-1 cycle
Exponential Predictive [6]	$I_{t+1} = \alpha \cdot i_t + (1-\alpha) \cdot I_t$, $\alpha = 0.5$
Expert-based Learning	Uses the above three policies as experts.

We develop a cycle-based simulator that reports the average power consumption and latency. We run the simulation for 50,000 cycles. All of the results reported in this paper are the averages over 20 different simulations with different random seeds.

TABLE II **Characteristics of Service Provider.**

Device	P_{active}	P_{sleep}	P_{tran}	T_{tran}	T_{be}
HDD	1.6	0.4	2.4	2.5	4.2

. The Lagrangian multiplier λ controls the power-latency tradeoff in our enhanced Q-learning algorithm. We vary the value of λ from 0.01 to 50 in order to generate a power-performance tradeoff curve in design space. We fix learning rate to $\epsilon = 0.25$

For the expert-based DPM [6], the agent learns how to choose the most proper expert based on the workload. We evaluate three different versions of expert-based DPM by setting the timeout of the fixed timeout expert to T_{be}, $3T_{be}$, and $5T_{be}$,. The other experts remain to be the same. The expert-based DPM also has a parameter $\alpha \in (0,1)$ that can be used to control the tradeoff between the power and performance.

Figure 2 gives the power-latency tradeoff curves for the enhanced Q-learning, the traditional Q-learning, and the expert-based DPM. It shows that the enhanced Q-learning algorithm finds better policy with more evenly distributed power-latency tradeoff points and wider tradeoff range. It consumes lower power while maintaining the same average latency as the expert-based DPM.

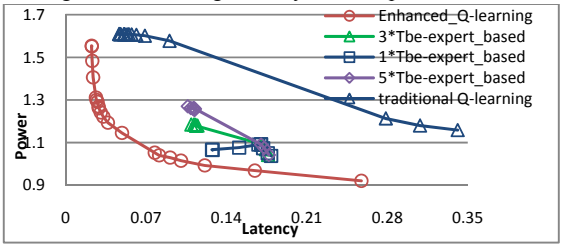

Figure 2. Power/Latency tradeoff curves for synthetic workload.

The next experiment is to evaluate the performance of the linear adaptation algorithm to find the most suitable Lagrangian multiplier λ with respect to the given latency constraint $\tilde{C}$. We choose $k = log10/(0.05 \times \tilde{C})$ as the adapting coefficient in Equation (7).

We do λ adaption every 200 cycles and begin to calculate average power-latency values 50 cycles after the new updated λ is applied to our Q-learning algorithm. The 50 cycles here is considered to be adaption period for the new λ_{t+1}.

TABLE III **Results for λ Adaption in Synthetic Workload.**

Constraints	0.973	0.123	0.079	0.049	0.026	0.022
Actual	1.028	0.136	0.071	0.046	0.026	0.023
Difference	5.6%	10.7%	9.1%	5.5%	0.7%	4.7%
Updates	48	7	11	13	1	22

TABLE III shows some of the results of λ adaption method. The first row gives the constraints represented by the average number of

requests in SQ. The second row gives the average latency that is measured in the system after the linear adaptation converges. The third row shows the relative difference between these two. According to our experiment, for about 90% of times the linear adaptation algorithm can converge to a policy whose performance is within 7% of the given constraint in less than 50 updates.

B. *Experiment Using Real Workload*

Experiments in this section are performed using real workloads. Using Windows Performance Monitor, we collected hard disk read/write request sequences from two desktop workstations.

Instead of using the number of requests, we use the aggregated transaction size of the requests to represent the state $\mathcal{SR}$ and $\mathcal{SQ}$. The current state of $\mathcal{SR}$ is represented by the total transaction size of all of the incoming requests in this cycle while the current state of $\mathcal{SQ}$ is represented by total transaction size of all of the waiting requests in the queue. We further discretize the state space of $\mathcal{SR}$ and $\mathcal{SQ}$. Both $\mathcal{SR}$ and $\mathcal{SQ}$ are divided into 7 different states: 0(empty state), 1(0~1MB), 2(1~2MB), 3(2~3MB), 4(3~4MB), 5(4~5MB), 6(> 5MB, overflow state).The $\mathcal{SP}$ used in the simulation is TOSHIBA MK6006AH hard disk drive (HDD) [14]. Its power consumption and switching time are reported in TABLE IV. The service rate for both read and write of $\mathcal{SP}$ is 16.6MB/sec.

TABLE IV **Characteristics of Service Provider.**

$P_{active}(W)$	$P_{sleep}(W)$	$P_{tran}(W)$	$T_{tran}(sec)$	$T_{be}(sec)$
1.1	0.12	1.2	3.0	6.6

The first trace was collected at night when only two applications were running and it took more than 200 minutes. We label this trace as "$trace\_low$". The other one was collected at noon with a set of applications running simultaneously and high disk I/O activities. We label this trace as "$trace\_high$".

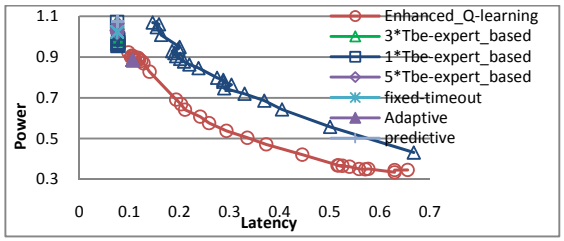

Figure 3. Performance of different algorithms under low workload.

Figure 3 shows the power-latency tradeoff curves for $trace\_low$. As shown in the figures, for different expert-based algorithms, $\mathcal{SP}$ keeps active most of the time in order to achieve high performance. The proposed enhanced Q-learning finds better policy that achieves lower power consumption with the cost of slightly higher latency. The enhanced Q-learning is able to explore the power-latency tradeoff more effectively.

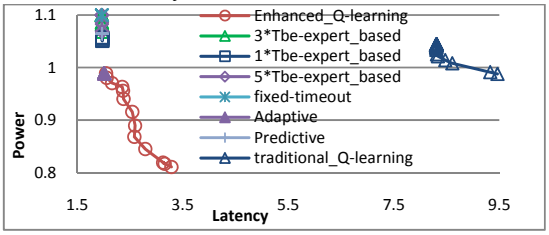

Figure 4. Performance of different algorithms under high workload.

Figure 4 gives the experimental results for $trace\_high$. The proposed Q-learning algorithm can reach up to 27% power saving compared to expert-based algorithm.

The real workload is different from the synthetic workload because its incoming rate is time varying. Therefore, the value of λ has to be adjusted from time to time during the runtime in order to satisfy the same performance constraint. For both traces, the latency/performance variation during the entire simulation can be controlled within 10% of the given constraint.

At the end, we varied the performance constraint and tested the linear adaption algorithm. TABLE V gives us the results in the main range of latency constraints. The average latency is the value for the entire simulation time. The average difference between latency constraint and actual latency is 6.88%.

TABLE V **Results for λ Adaption in Real Workload.**

Constraints	0.365	0.244	0.150	0.111	0.097	0.086
Actual	0.321	0.241	0.147	0.111	0.096	0.088
Difference	11.9%	1.5%	1.4%	0.7%	1.0%	2.9%

V. Summary and Conclusions

In this paper, we present a novel model-free on-line algorithm for dynamic power management with performance constraint. By restricting the search space of power manager, the proposed algorithm converges to a better policy quickly. Furthermore, our power manager is able to adapt itself to the changing performance constraint during runtime.

References

[1] L. Benini, G. Paleologo, A. Bogliolo, and G. De Micheli, "Policy optimization for dynamic power management," *IEEE Trans on Computer-Aided Design*, Vol. 18, pp.813-833, Jun. 2001.

[2] T. Simunic, L. Benini, and G. De Micheli, "Event-driven power management," *IEEE Trans on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 20, pp.840-857, Jul. 2001.

[3] H. Jung, and M. Pedram, "Dynamic power management under uncertain information," in *Proceedings of Design Automation & Test in Europe (DATE'07)*, Apr. 2007.

[4] Q. Qiu, Y. Tan and Q. Wu, "Stochastic Modeling and Optimization for Robust Power Management in a Partially Observable System," in *Proceedings of Design Automation & Test in Europe (DATE'07)*, Apr. 2007.

[5] G. Theocharous, S. Mannor, N. Shah, P. Gandhi, B. Kveton, S. Siddiqi, and C-H. Yu, "Machine Learning for Adaptive Power Management," *Intel Technology Journal*, Vol. 10, pp. 299-312, Nov. 2006.

[6] D. Dhiman and T. Simunic Rosing, "Dynamic power management using machine learning," *Proceedings of International Conference on Computer-Aided Design (ICCAD'06)*, Nov. 2006.

[7] V.L. Prabha and E. C. Monie, "Hardware Architecture of Reinforcement Learning Scheme for Dynamic Power Management in Embedded Systems," EURASIP Journal on Embedded Systems, Vol. 2007.

[8] D. V. Djonin and V. Krishnamurthy, "V-BLAST power and rate control under delay constraints in Markovian fading channels—Optimality of randomized monotonic policies," *SIEEE Trans. Signal Process.*, 2007.

[9] D. P. Bertsekas, *Nonlinear Programming,* 2nd ed. Belmont, MA: Athena Scientific, 1999.

[10] M. Pendrith, "On reinforcement learning of control actions in noisy and nonMarkovian domains" *Technical Report NSW-CSE-TR-9410,* School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia, 1994.

[11] K. Sikorski and T. Balch, "Model-Based and Model-Free Learning in Markovian and Non-Markovian Environments," *Proceedings of Agents-2001 Workshop on Learning Agents*, May 29, 2001.

[12] D. V. Djonin and V. Krishnamurthy, "Structural results on the optimal transmission policies and costs for correlated sources and channels," in Proc. of CDC, 2005.

[13] S. Fujishiqe, "Submodular Functions and Optimization," Vol.58, Elsevier Science, September 2005.

[14] "TOSHIBA Hard Disk Drive Specification 1.8 inch Hard Disk Drive MK6006GAH/MK4006GAH/MK3006GAL".