An Online Algorithm Optimally Self-tuning to Congestion for Power Management Problems

Wolfgang Bein[1], Naoki Hatta[2], Nelson Hernandez-Cons[3], Hiro Ito[2], Shoji Kasahara[3], and Jun Kawahara[4]

[1] Center for Information Technology and Algorithms, School of Computer Science, University of Nevada, Las Vegas
beinw@unlv.nevada.edu
[2] Department of Communications and Computer Engineering, Graduate School of Informatics, Kyoto University
{nhatta,itohiro}@kuis.kyoto-u.ac.jp
[3] Department of Systems Science, Graduate School of Informatics, Kyoto University
{shoji,nelson}@i.kyoto-u.ac.jp
[4] JST ERATO MINATO Project
jkawahara@erato.ist.hokudai.ac.jp

Abstract. We consider the classical power management problem: There is a device which has two states ON and OFF and one has to develop a control algorithm for changing between these states as to minimize (energy) cost when given a sequence of service requests. Although an optimal 2-competitive algorithm exists, that algorithm does not have good performance in many practical situations, especially in case the device is not used frequently. To take the frequency of device usage into account, we construct an algorithm based on the concept of "slackness degree." Then by relaxing the worst case competitive ratio of our online algorithm to $2 + \varepsilon$, where ε is an arbitrary small constant, we make the algorithm flexible to slackness. The algorithm thus automatically tunes itself to slackness degree and gives better performance than the optimal 2-competitive algorithm for real world inputs. In addition to worst case competitive ratio analysis, a queueing model analysis is given and computer simulations are reported, confirming that the performance of the algorithm is high.

1 Introduction

Consider an electric light which is turned on automatically when someone passes by, say, for example at the entrance of a building. Or consider a device which can enter a "sleep mode" – a state for energy saving when not used for a certain period of time (e.g. a server or a copy machine). We abstract such a situation to the automatic operation of a two-state device, which has an *ON-state* and an *OFF-state*. In this paper we equate the sleep mode with the OFF-state, which consumes no power. If a user utilizes the device, the state of it must be ON during usage. When the user has finished, if another user needs it almost immediately, it is wasteful to turn it off, because additional power consumption

R. Solis-Oba and G. Persiano (Eds.): WAOA 2011, LNCS 7164, pp. 35–48, 2012.

occurs when switching the state. For example, a copy machine consumes extra electrical power when it comes back from the sleep mode. In the case of a compact fluorescent light, switching states frequently shortens the life of the bulb and although the switching cost may be negligible one can amortize the shorter lifespan appropriately.

It is, of course, not known in advance when and how many users will request service. Since we have to control switching the states of the device without knowledge of future requests, the situation can be formulated as an online optimization problem. More formally, there is a two-state device. Users request the device one after another and use it for an arbitrary period of time. When it is used, the device must be kept in the ON-state. Once it is not used it can be turned off at an arbitrary time. In the ON-state the device uses an amount of power proportional to usage time, one unit of cost per unit of time. We call this the *running cost*. There is no cost in the OFF-state. There is also a constant *switching cost* $a > 0$ for turning the device on and no cost for turning it off. An optimal online algorithm is well known for this problem in the context of the worst case competitive analysis [7,11,12]. We call this the optimal worst case competitive ratio algorithm, or *OWCR*, for short. *OWCR* turns it on when a user requests service. After use *OWCR* waits for the next service request for time a. If another user requests service within the period, *OWCR* does not turn it off; otherwise, *OWCR* turns it off after time a. The *OWCR* is 2-competitive, which is optimal.

However, *OWCR* does not perform well in various natural situations. If the arrival interval of users is spaced long enough, *OWCR* keeps the device in the ON-state for an extra a units for each user. This action seems to be quite wasteful. If the device usage time is infinitesimal the cost for *OWCR* is $2a$ (the switching cost and running cost are both a). On the other hand, the behavior of the optimal offline algorithm "OPT" is to turn off immediately after the user has finished. OPT pays only switching cost a (and the infinitesimal running cost). Therefore, the competitive ratio for such a request sequence is 2. The fact is that *although OWCR does not seem clever at all, under worst case competitive analysis this algorithm has the best performance.*

Basic Concept of Our Algorithm. Since *OWCR* is optimal it appears as if there is not much hope for improvement. However, if we allow the worst case competitive ratio to increase by only a small positive constant ε from 2 (say, for example, by 0.01), we can design various algorithms that have lower cost than *OWCR* for real world inputs. For example, we can decrease the duration of keeping the ON-state ("*standby time*") gradually when the frequency of the device usage becomes low: If the *waiting time* – the time from the preceding user leaving to the latest user arriving – was more than a (something like an "off-peak" situation), the system may be slack and the algorithm may elect to make the standby time shorter. On the other hand, if the preceding waiting time was less than or equal to a (similar to a "rush hour"), the algorithm may reset the standby time.

In other words, standby time may decrease as a sequence $x_1 > x_2 > \cdots$ while off-peak arrivals continue. Such an algorithm may have better performance.

However we cannot evaluate this easily in the context standard competitive analysis. On the other hand, we are not satisfied only with heuristics or experimental analysis and we seek a rigorous theoretical analysis.

For the above aim, we introduce a parameter called "slackness degree," which represents the frequency of arrivals. We set the parameters $x_1, x_2, \ldots$ to optimize the competitive ratio for each slackness degree. The proposed algorithm adaptively reacts to fluctuation of inputs and works optimally in the sense of the competitive ratio according to slackness degrees. Moreover, an important property of our algorithm is that it need not know the actual slackness degree. Its worst case competitive ratio is at most $2 + \varepsilon$ and it is close to the optimal offline algorithm for real world inputs. E.g., if we set $\varepsilon = 0.01$ and if the slackness degree of an input is 10, 20, 50 and 100, then the competitive ratio is 1.58, 1.29, 1.12 and 1.06, respectively.

Related Work. As mentioned above it is well known that the optimal competitive ratio of this problem is 2 [9,11,12,17]. In the randomized model where the algorithm uses a probability distribution the best competitive ratio improves to $e/(e-1) \approx 1.582$ (the expected cost of the algorithm for all inputs is within a factor of $e/(e-1)$ of the optimal cost). Furthermore, it is known that this bound is tight [9,10,11]. Systems with multiple power saving states have also been studied, and it is clear that the additive model, where costs at any states are cumulative, is reduced to the two state model. Thus, it is sufficient to consider the two state model [1,8,15,18].

Strategies for this problem are categorized into two groups: adaptive and non-adaptive. Non-adaptive strategies set a threshold only once at first on the idle time interval for switching from the active state to the sleep state [8]. Adaptive strategies use the history of idle periods to make the decisions for future inputs [6,8].

There are several lines of investigation for evaluating algorithms more adequately by considering alternatives to the competitive ratio or analysis. In substitution for the competitive ratio, the accommodation ratio [4], the Max/Max ratio [2] and the random order ratio [13] were proposed. To analyze more realistic situations, competitive analysis with restriction to the adversary or using parameters have also been considered. In the access graph model [3] and the diffuse adversary model [14], the competitive ratio is evaluated by using weak adversaries whose action is restricted (see the survey [5]). Panagiotou et. al. [16] analyzed the LRU algorithm for paging by introducing parameters α and β which characterize the degree of the locality of reference. They showed the competitive ratio is bounded by a function of α and β.

Organization. Section 2 gives the basic statement of the problem. Section 3 presents our algorithm and gives the concept of slackness degree and competitive ratio analysis under this concept. Section 4 gives an analysis using queueing theory with computer simulations. Conclusions are given in Section 5.

2 Problem Statement

In this paper we consider a device with infinite capacity that has two states, an *ON-state* and an *OFF-state* (simply ON and OFF), for which we design a control algorithm for changing between ON and OFF. Let $t_1^s, \ldots, t_n^s, t_1^e, \ldots, t_n^e$ be non-negative real values that represent the service times $t\_^s$ and end-of-service times $t\_^e$ for n requests and which satisfy $0 \leq t_1^s < t_1^e < t_2^s < t_2^e < \cdots < t_n^s < t_n^e$. The input for this problem is given as $\sigma = \langle (t_1^s, t_1^e), (t_2^s, t_2^e), \ldots, (t_n^s, t_n^e) \rangle$. For this input, the device must be kept ON between times t_i^s and t_i^e for each $i = 1, \ldots, n$.

The state of the device can be switched at an arbitrary time. There is no cost for switching from ON to OFF, while there is a *switching cost* $a\,(> 0)$ when switching from OFF to ON. For keeping the ON-state, it takes *running cost* of one unit per unit time.

The strategy of the optimal offline algorithm (OPT) for this problem is clear: If the period between the current request and next one is less than a, the device is kept ON. Otherwise, it is turned off immediately. Therefore OPT's total cost for the input $\sigma = \langle (t_1^s, t_1^e), (t_2^s, t_2^e), \ldots, (t_n^s, t_n^e) \rangle$ is $a + \sum_{i=1}^{n} (t_i^e - t_i^s) + \sum_{i=1}^{n-1} \min\{t_{i+1}^s - t_i^e, a\}$.

For some i-th request (t_i^s, t_i^e), we consider the sum of the i-th running cost and the next $(i+1)$-th switching cost. Let ALG be any algorithm. Let u be the period between the end of the request and the start of the next request (i.e., $u = t_{i+1}^s - t_i^e$). Then the optimal offline algorithm pays $t_i^e - t_i^s + \min\{u, a\}$. If ALG turns the device off after $v\,(< u)$ standby time, the cost is $t_i^e - t_i^s + v + a$. Otherwise, it must pay $t_i^e - t_i^s + u$. In each case, the smaller $t_i^e - t_i^s$ is, the worse the competitive ratio becomes. Therefore, from the standpoint of competitive analysis, it is enough to consider that usage times of the device (i.e., $t_i^e - t_i^s$ for each i) are tiny. On the basis of the above discussion, we redefine this problem as follows.

Let $t_1, \ldots, t_n$ be non-negative real values satisfying $0 \leq t_1 < \cdots < t_n$ representing the time of service of the device for n requests. An input is given as $\sigma = \langle t_1, t_2, \ldots, t_n \rangle$. We do nothing if the state is ON at t_i $(i = 1, \ldots, n)$, and should turn a device ON if it is OFF at that time. For a given input $\sigma = \langle t_1, t_2, \ldots, t_n \rangle$ (n may be ∞), the action of an algorithm is determined by a sequence $\langle w_1, w_2, \ldots, w_n \rangle$, where w_i is standby time after ith request is leaving. In other words, the problem is how to determine w_i from $\langle t_1, t_2, \ldots, t_i \rangle$. For each $i = 2, \ldots, n$, let $u_i = t_i - t_{i-1}$ be an *idle period*. OPT's cost for this redefined problem is $\mathrm{OPT}(\sigma) = a + \sum_{i=1}^{n-1} \min\{t_{i+1} - t_i, a\}$. We denote ALG's cost for input σ by $\mathrm{ALG}(\sigma)$ and the competitive ratio of ALG for σ by $\mathcal{R}_{\mathrm{ALG}}(\sigma) = \mathrm{ALG}(\sigma)/\mathrm{OPT}(\sigma)$. Let Σ be the set of whole inputs σ. For a subset $\Sigma' \subseteq \Sigma$, we define $\mathcal{R}_{\mathrm{ALG}}(\Sigma') = \sup_{\sigma \in \Sigma'} \mathcal{R}_{\mathrm{ALG}}(\sigma)$. And we represent $\mathcal{R}_{\mathrm{ALG}} = \mathcal{R}_{\mathrm{ALG}}(\Sigma)$, which is the (worst case) competitive ratio of ALG.

3 Our Algorithm

3.1 Decrease and Reset Algorithm (DRA)

We propose an algorithm which decreases the standby time gradually when the frequency of the device usage becomes low.

Decrease and Reset Algorithm (DRA).
Let $x_1, x_2, \ldots,$ be an infinite non-increasing sequence of non-negative values. In DRA, $w_i = x_{f(i)}$ such that

$$f(i) = \begin{cases} f(i-1)+1 & \text{if } u_i \geq a \text{ and } i \neq 1, \\ 1 & \text{otherwise.} \end{cases}$$

If $x_i = a$ for all i, DRA is equivalent to *OWCR*. Setting x_i be larger than a is clearly wasteful, and hence we consider cases such that $x_i \leq a$ for all $i = 1, 2, \ldots$. From a simple observation we see that x_1 gives a lower bound of $\mathcal{R}_{\mathrm{DRA}}$:

Observation 1. $\mathcal{R}_{DRA} \geq 1 + a/x_1$.

Proof. Let m be an integer. For an input $\sigma = \langle x_1, 2x_1, \ldots, mx_1 \rangle$, OPT's total cost is $a + (m-1)x_1$ and DRA's total cost is $m(a + x_1)$. Thus the competitive ratio of them is the following:

$$\mathcal{R}_{\mathrm{DRA}} \geq \mathcal{R}_{\mathrm{DRA}}(\sigma) = \frac{m(a+x_1)}{a+(m-1)x_1} \overset{(m\to\infty)}{\to} 1 + \frac{a}{x_1}.$$

□

From this observation, it follows that x_1 cannot be much smaller than a, otherwise $\mathcal{R}_{\mathrm{DRA}}$ becomes very large. In other words, if the difference between a and x_1 is small, the effect to $\mathcal{R}_{\mathrm{DRA}}$ is not so large. Thus we relax the worst case competitive ratio from 2 to $2 + \varepsilon$ for small $\varepsilon > 0$, i.e., we let $x_1 = a/(1+\varepsilon)$.

The above observation is easily extended to the other values $x_2, x_3, \ldots,$ as follows.

Observation 2. *For any integer k,*

$$\mathcal{R}_{DRA} \geq \frac{ka + \sum_{i=1}^{k} x_i}{(k-1)a + x_k}.$$

Proof. Let m be an integer and $t_1, t_2, \ldots, t_{mk}$ be a sequence such that $t_1 = a$ and if $i = 1 \bmod k$ then $t_i = t_{i-1} + x_k$, otherwise $t_i = t_{i-1} + a$. For input $\sigma = \langle t_1, t_2, \ldots, t_{mk} \rangle$, DRA sets $w_i = x_{g(i)}$ for all i, where $g(i) = ((i-1) \bmod k) + 1$. OPT's total cost is $a + m(k-1)a + (m-1)x_k$ and DRA's total cost is $mka + m\sum_{i=1}^{k} x_i$. Thus the competitive ratio of them is the following:

$$\mathcal{R}_{\mathrm{DRA}} \geq \mathcal{R}_{\mathrm{DRA}}(\sigma) = \frac{mka + m\sum_{i=1}^{k} x_i}{a + m(k-1)a + (m-1)x_k} \overset{(m\to\infty)}{\to} \frac{ka + \sum_{i=1}^{k} x_i}{(k-1)a + x_k}.$$

So this observation is satisfied. □

Our upper bound of the competitive ratio is $2+\varepsilon$, and thus, the following inequalities must be satisfied for every $k = 1, 2, \ldots$:

$$\frac{ka + \sum_{i=1}^{k} x_i}{(k-1)a + x_k} \leq 2 + \varepsilon.$$

Solving this equation for x_k, we have

$$x_k \geq \frac{1}{1+\varepsilon}\left((2+\varepsilon)a + \sum_{i=1}^{k-1} x_i\right) - ka.$$

By elementary induction, we obtain

$$x_k \geq -\varepsilon\left(\frac{2+\varepsilon}{1+\varepsilon}\right)^k a + (1+\varepsilon)a. \tag{1}$$

This is a necessary condition for keeping the competitive ratio less than or equal to $2+\varepsilon$. But this condition is not sufficient to guarantee optimality within the ε bound. We propose next an algorithm that sets exact values for $x_2, x_3, \ldots$, to guarantee optimality within the ε bound.

3.2 How to Set the Coefficients for "Optimality"

Before turning to this problem, we need to define what "optimal" means here. Our motivation is to give a better algorithm for slack systems. Thus we introduce a measure, "slackness degree" for representing the slackness of input sequences. For an input sequence $\sigma = \langle t_1, t_2, \ldots, t_n \rangle$, request i is called a *busy request* if $u_i \leq a$ or a *slack request*, otherwise. The first request is neither busy nor slack one. We denote the number of slack requests in σ by $s(\sigma)$, and that of busy requests in σ by $b(\sigma)$.

Definition 1. *For an input σ, if $s(\sigma)/b(\sigma) \geq d$ $(b(\sigma) \neq 0)$ for a real number $d \geq 0$, σ is called d-slack. The slackness degree $d(\sigma)$ is defined as the maximum d such that σ is d-slack.*

The slackness degree describes how busy the inputs are. Clearly if $d(\sigma)$ is larger, σ has more slack. We will optimize DRA under the assumption that an input is d-slack without knowing the value of d.

We consider asymptotic performance, and assume that σ is large enough. In other words σ has a sufficient number of busy requests, i.e., $b(\sigma) = \omega(1)$ if $b(\sigma) \neq 0$.

Note that for $b(\sigma) = 0$ (i.e., all arrivals are slack), we can easily get the upper bound of the competitive ratio of $1 + \frac{\sum_{i=1}^{|\sigma|} x_i}{|\sigma| a}$, which is close to 1 when $|\sigma|$ is large and $\lim_{i\to\infty} x_i = 0$. This case is so particular that we ignore it in the following.

We will show that it is sufficient to consider inputs which end with a busy request:

For a detailed analysis, let us separate an input $\sigma = \langle t_1, t_2, \ldots, t_n \rangle$ into some ($b(\sigma)$ or $b(\sigma)+1$) blocks as follows. Assume that $t_{b_1}, t_{b_2}, \ldots, t_{b_{b(\sigma)}}$ ($0 \le b_1 < b_2 < \cdots < b_{b(\sigma)} \le n$) are the busy requests in σ. The blocks are defined as $B_1 = \{t_1, \ldots, t_{b_1}\}$, $B_2 = \{t_{b_1+1}, \ldots, t_{b_2}\}$, $\ldots$, $B_{b(\sigma)} = \{t_{b_{b(\sigma)-1}+1}, \ldots, t_{b_{b(\sigma)}}\}$. If $b_{b(\sigma)} < n$, then $B_{b(\sigma)+1} = \{t_{b_{b(\sigma)}+1}, \ldots, t_n\}$ also exists. For analyzing the worst case competitive ratio we will show that the final block $B_{b(\sigma)+1}$ can be ignored even if it exists. Let $\beta(\sigma)$ be the number of blocks in σ. (Then $\beta(\sigma) = b(\sigma)$ or $b(\sigma)+1$.) Let $s(B_i)$ be the number of slack requests in block B_i.

Lemma 1. *If $s(B_{b(\sigma)}) \le s(B_{b(\sigma)+1}) - 2$ holds, then $\mathcal{R}_{DRA}(\sigma) \le \mathcal{R}_{DRA}(\sigma')$, where σ' is obtained from σ by exchanging $t_{b_{b(\sigma)}}$ with $t_{b_{b(\sigma)}+1}$, i.e., $\sigma' = \langle t_1, \ldots, t_{b_{b(\sigma)}-1}, t_{b_{b(\sigma)}+1}, t_{b_{b(\sigma)}}, t_{b_{b(\sigma)}+2}, \ldots, t_n \rangle$. (Note that $t_{b_{b(\sigma)}+1}$ is a slack request from $s(B_{b(\sigma)})$ (≥ 2).)*

Proof. The competitive ratio of DRA for σ' is

$$\mathcal{R}_{\mathrm{DRA}}(\sigma') = \frac{DRA(\sigma) - x_{s(B_{b(\sigma)+1})} + x_{s(B_{b(\sigma)})+2}}{OPT(\sigma) - x_{s(B_{b(\sigma)})+1} + x_{s(B_{b(\sigma)})+2}}.$$

Since $x_1, x_2, \ldots$ is a non-increasing sequence and $s(B_{b(\sigma)}) \le s(B_{b(\sigma)+1}) - 2$, $x_{s(B_{b(\sigma)})+2} \ge x_{s(B_{b(\sigma)+1})}$ and $x_{s(B_{b(\sigma)})+1} \ge x_{s(B_{b(\sigma)})+2}$ hold. Thus we get

$$\mathcal{R}_{\mathrm{DRA}}(\sigma') = \frac{DRA(\sigma) - x_{s(B_{b(\sigma)+1})} + x_{s(B_{b(\sigma)})+2}}{OPT(\sigma) - x_{s(B_{b(\sigma)})+1} + x_{s(B_{b(\sigma)})+2}} \ge \frac{DRA(\sigma)}{OPT(\sigma)} = \mathcal{R}_{\mathrm{DRA}}(\sigma).$$

□

Lemma 2. *For any $d \ge 0$ and sufficiently long inputs, there exists an input which finishes with a busy request and gives the worst competitive ratio in the same slackness degree d.*

Proof. By Lemma 1 for a d-slack input σ we can shift the last busy request later as long as the last two blocks satisfy $s(B_{b(\sigma)}) \le s(B_{b(\sigma)+1}) - 2$ without decreasing the competitive ratio (Operation 1). We can clearly exchange the two subsequences in σ which begin with a slack request and end with a busy request without changing the competitive ratio (Operation 2).

When we apply Operation 1 and Operation 2 for a sufficiently long input σ repeatedly and let the result be σ^*, which is d-slack and gives the worst competitive ratio, approximately we can assume that σ^* finishes with a busy request. □

Lemma 3. *For any input σ, if each x_i satisfies inequality (1) then $\mathcal{R}_{DRA}(\sigma) \le 2 + \varepsilon$.*

Proof. In Observations 1 and 2, the given input is clearly the worst for the competitive ratio among one-block input σ (i.e., σ includes one busy requests at the end) and the slackness degree is fixed. This means $\mathcal{R}_{\mathrm{DRA}} \le 2 + \varepsilon$ for any one-block input. From Lemma 2, $\mathcal{R}_{\mathrm{DRA}} \le 2 + \varepsilon$ for any long enough input. □

Lemma 4. *For a sufficiently long input σ, if σ is d-slack ($d > 0$) and $x_1, x_2, \ldots$ ($x_i \leq a$) satisfy inequality (1), the following inequality holds:*

$$\mathcal{R}_{DRA}(\sigma) \leq 1 + \frac{1}{d} + \frac{\sum_{i=1}^{\infty} x_i}{ad}. \tag{2}$$

And the equality holds for $d \geq h - 1$ where $h = \min\{i \mid x_i = 0\}$.

Proof. To analyze the worst case input, we define $\sigma(k)$ as an input in which one busy request arrives after $(k-1)$ slack requests, where $k = 1, 2, \ldots$, is any positive integer. Then we find that all the worst case input instances are described as the combination of $\sigma(k)$ by Lemma 2. Let the combination of them be σ_w, which can be represented by a sequence of $\sigma(\cdot)$, i.e., $\sigma_w = \sigma(f(1))\sigma(f(2))\cdots\sigma(f(n))$ where $n = b(\sigma_w)$, and each $f(i)$ is a positive integer ($i = 1, \ldots, n$).

Against this input, DRA must pay the switching cost for all the requests. The cost of DRA for σ_w is $a + \sum_{i=1}^{n} \left\{ \sum_{j=1}^{f(i)} (x_j + a) \right\} + x_1$. OPT keeps the ON-state during $x_{f(i)}$ for the last input in each $\sigma(f(i))$ and switches to OFF immediately for the other inputs. The cost is $a + \sum_{i=1}^{n} x_{f(i)} + \sum_{i=1}^{n} (f(i) - 1)a$. Therefore the competitive ratio is

$$\begin{aligned}
\mathcal{R}_{\mathrm{DRA}}(\sigma_w) &= \frac{a + \sum_{i=1}^{n} (\sum_{j=1}^{f(i)} (x_j + a)) + x_1}{a + \sum_{i=1}^{n} x_{f(i)} + \sum_{i=1}^{n} (f(i) - 1)a} \\
&\leq \frac{a + \sum_{i=1}^{n} \sum_{j=1}^{f(i)-1} x_j + \sum_{i=1}^{n} f(i)a + x_1}{a + \sum_{i=1}^{n} (f(i) - 1)a} \\
&\leq \frac{a + n \sum_{i=1}^{\infty} x_i + \sum_{i=1}^{n} (f(i) - 1)a + an + x_1}{a + \sum_{i=1}^{n} (f(i) - 1)a}.
\end{aligned}$$

Since $\sum_{i=1}^{n} (f(i) - 1)/n = s(\sigma_w)/b(\sigma_w) \geq d$ and σ_w is d-slack, we have

$$\begin{aligned}
\mathcal{R}_{\mathrm{DRA}}(\sigma) \leq \mathcal{R}_{\mathrm{DRA}}(\sigma_w) \quad &\leq \quad 1 + \frac{\sum_{i=1}^{\infty} x_i + a + x_1/n}{a/n + ad} \\
&\overset{(n \to \infty)}{\to} 1 + \frac{1}{d} + \frac{\sum_{i=1}^{\infty} x_i}{ad}.
\end{aligned}$$

The inequalities are tight when $\sum_{i=1}^{n} x_{f(i)} = 0$ and the slackness degree of σ_w is just d, and such input exists only when $\sum_{i=1}^{n} f(i)/n \geq h$. Thus for a sufficiently long input when $d \geq h - 1$, we find that the bound is tight. □

Note that even for 0-slack inputs ($s(\sigma) = 0$), if $x_1, x_2, \ldots$ satisfy (1), the competitive ratio is guaranteed to be $2 + \varepsilon$ according to Observation 1.

We get the upper bound of the worst competitive ratio with parameter d. To minimize it, we should minimize each x_i such that they satisfy (1).

Theorem 1. *We set the coefficients x_i as $x_i = \max\{-\varepsilon\,((2+\varepsilon)/(1+\varepsilon))^i\, a + (1+\varepsilon)a,\ 0\}$. Then for any sufficiently long d-slack input σ, DRA guarantees the following competitive ratio:*

$$\mathcal{R}_{DRA}(\Sigma_d) = \min\left\{1 + \frac{1}{d} + \frac{\sum_{i=1}^{h-1} x_i}{ad},\ 2 + \varepsilon\right\}, \tag{3}$$

where Σ_d is the set of sufficiently long d-slack inputs, and $h = \lfloor(\log(1+\varepsilon)-\log\varepsilon)/(\log(2+\varepsilon)-\log(1+\varepsilon))\rfloor + 1$.

Proof. Let h be defined as in Lemma 4. Then the value of h is obtained as shown above. From Lemmas 4 and 3 we get

$$\mathcal{R}_{\mathrm{DRA}}(\Sigma_d) \le \min\left\{1 + \frac{1}{d} + \frac{\sum_{i=1}^{\infty} x_i}{ad},\ 2+\varepsilon\right\}.$$

To optimize the competitive ratio we should minimize each x_i in range of satisfying inequality (1). So we get

$$x_i = \begin{cases} -\varepsilon\left(\dfrac{2+\varepsilon}{1+\varepsilon}\right)^i a + (1+\varepsilon)a & \text{if } i < h, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

This means $\sum_{i=1}^{\infty} x_i = \sum_{i=1}^{h-1} x_i$.

Furthermore, from Lemma 4, when $d \ge h-1$ there are inputs which hold the equation in (2) tightly. On the other hand, from Lemma 3, when $d < h-1$, there are inputs such that DRA uses only $x_1, \ldots, x_d$ (i.e., they satisfy (1) tightly.) and then achieve $2+\varepsilon$-competitive ratio tightly. Therefore we obtain the desired equation (3). □

From this, we will call the *DRA* satisfying the condition of Theorem 1 *the optimal DRA* (*ODRA*).

Corollary 1. *For the value that* $0 < \varepsilon < 0.2$,

$$\mathcal{R}_{ODRA} \le \min\left\{1 + \frac{(1+\varepsilon)^2 + 2(1+\varepsilon)\log\frac{1}{\varepsilon}}{d},\ 2+\varepsilon\right\}.$$

We also get such a heuristic bound, but skip the details of proof. If $d \to \infty$ then $\mathcal{R}_{\mathrm{ODRA}} \to 1$. Therefore we confirm that the competitive ratio is close to 1 when the frequency of requests within any time period is small enough.

Note that this algorithm works certainly without information of the input σ. Since we can define the d-slackness from some period from the entire input, we can evaluate the competitive ratio considering a part when the slackness degree changes.

4 Queueing Analysis

4.1 Analysis

In this section, we analyze the cost performance of DRA using queueing theory. We assume that customers arrive at the system according to a Poisson process with rate λ. The sojourn time of a customer is independently and identically distributed (i.i.d.) with a general distribution with mean $1/\mu$. As we mentioned before, the system capacity is infinity. Then the system we consider here is an M/G/∞ queueing model.

In the M/G/∞ model, the busy period is defined as the time interval during which the number of customers in the system is greater than zero, while in the idle period, no customers are in the system. For analytical simplicity, we assume that the system is in equilibrium at time 0, and that the first busy period starts at time 0. Let B_n and I_n denote the nth busy period of the system and the nth idle period, respectively. Note that both busy periods and idle periods are i.i.d., and hence independent of n. The mean busy period and the mean idle period of the M/G/∞ system are given by

$$E[B_n] = \frac{e^{\rho} - 1}{\lambda} (\equiv E[B]), \quad E[I_n] = \frac{1}{\lambda} (\equiv E[I]), \tag{5}$$

respectively, where $\rho = \lambda/\mu$. We define the nth cycle as the time interval consisting of B_n and I_n.

The power control process under DRA with coefficients given by (4) evolves as follows. When the first busy period B_1 starts, the initial power cost a is required. During the busy period, the power cost per unit time is one. When B_1 ends, the system is kept in the ON-state for the standby time of x_1. Note that x_1 is the power cost of the first idle period I_1. If $I_1 > a$, the next standby time for I_2 is set to x_2. If $I_1 \leq a$, then the standby time for I_2 is initialized to x_1. Similarly, if $I_1 > a$ and $I_2 > a$, then the standby time for I_3 is set to x_3, while if $I_1 > a$ and $I_2 \leq a$, the standby time for I_3 is initialized to x_1, and so on. In the following, the time interval from the beginning of the busy period with x_1 standby time to the end of the idle period which is smaller than a is referred to as the reset interval.

Let L (≥ 1) denote the number of cycles in a reset interval. Consider the amount of power consumption during a reset interval. When the number of cycles in the reset interval is $L = k$, the amount of power consumption is given by the power consumption for k busy periods and k standby times. Let T_k denote the total amount of power consumption of standby times in the reset interval consisting of k cycles. We obtain

$$T_k = \begin{cases} \sum_{i=1}^{k-1} x_i + I_k \cdot 1_{\{I_k \leq x_k\}}, & k = 1, 2, \ldots, h-1, \\ \sum_{i=1}^{h-1} x_i, & k \geq h, \end{cases}$$

where 1_χ is the indicator function of event χ. Then we have the following lemma.

Lemma 5. *The mean of the total amount of power consumption of standby times in a reset interval $E[T_L]$ is given by*

$$\begin{aligned} E[T_L] = {} & \epsilon a(2+\epsilon)e^{-\lambda a}(1-e^{-\lambda a})^{h-2} \\ & -\epsilon a(2+\epsilon)(1-e^{-\lambda a})\frac{(2+\epsilon)e^{-\lambda a}}{1+\epsilon-(2+\epsilon)e^{-\lambda a}} \left\{1 - \left(\frac{2+\epsilon}{1+\epsilon} \cdot e^{-\lambda a}\right)^{h-2}\right\} \\ & +(1+\epsilon)a \cdot \frac{e^{-\lambda a}}{1-e^{-\lambda a}} \left\{1 - (h-1)e^{-\lambda a(h-2)} + (h-2)e^{-\lambda a(h-1)}\right\} \\ & +\frac{1}{\lambda} \cdot \frac{1-e^{-\lambda a(h-1)}}{1-e^{-\lambda a}} - \sum_{k=1}^{h-1} e^{-\lambda a(k-1)} \left(\frac{1}{\lambda} + x_k\right) e^{-\lambda x_k} \end{aligned}$$

$$+\left[\epsilon a(2+\epsilon)\left\{1-\left(\frac{2+\epsilon}{1+\epsilon}\right)^{h-1}\right\}+(1+\epsilon)a(h-1)\right]\cdot e^{-\lambda a(h-1)}. \quad (6)$$

From the Poisson arrival assumption we can obtain this formula, however, we skip the details of proof.

Let Q_{ODRA} denote the mean power-consumption cost per unit time. Then we obtain the following theorem.

Theorem 2. *Q_{ODRA} is given by*

$$Q_{\mathrm{ODRA}}=\frac{1}{e^{\rho}}\left[\lambda a\left\{(1-e^{-\lambda a})\cdot\sum_{k=1}^{h-1}e^{-\lambda\{a(k-1)+x_k\}}+e^{-\lambda a(h-1)}\right\}\right.$$
$$\left.+e^{\rho}-1+\lambda(1-e^{-\lambda a})E[T_L]\right], \quad (7)$$

where $E[T_L]$ is given by (6).

Proof. Let R denote the reset interval. Using L, the number of cycles in a reset interval, we obtain $R=\sum_{n=1}^{L}(B_n+I_n)$. We skip the details of analysis, but we can obtain the mean reset interval $E[R]$ as follows.

$$E[R]=E\left[\sum_{n=1}^{L}B_n\right]+E\left[\sum_{n=1}^{L}I_n\right]=\frac{e^{\rho}}{\lambda(1-e^{-\lambda a})}.$$

Let $W(k)$ denote the total amount of power consumption for a reset interval which consists of k cycles. $W(k)$ is given by $W(k)=(k-1)a+a\cdot 1_{\{x_k<I_k\le a\}}+\sum_{i=1}^{k}B_i+T_k$. Taking the mean of $W(L)$ yields

$$E[W(L)]=a(E[L]-1)+aE[1_{\{x_k<I_k\le a\}}]+E[L]E[B]+E[T_L].$$

Note that reset intervals are i.i.d. and that the amount of power consumption during the reset interval is also i.i.d. Therefore, we have $Q_{\mathrm{ODRA}}=E[W(L)]/E[R]$ from the renewal-reward theorem [19]. □

4.2 Numerical Examples

In this section, we present some numerical examples using the analysis shown in the previous section. In order to validate the analysis, we also perform some simulations with the algorithms and compare the results obtained with the ones from the analysis.

Algorithms Compared to ODRA. For comparing with *ODRA* we consider the following algorithm, which is a simple variant of DRA.

Given parameter k let ALG(k) be the DRA algorithm where $x_1=x_2=\ldots=x_k=a$, $x_{k+1}=x_{k+2}=\ldots=0$.

Table 1. Basic parameters

Parameter	Value
Value of a	1, 3, 10 [unit]
ALG(10) parameter k	10
Value of ϵ	0.1, 0.01, 0.001
Consuming cost while in ON-state	1 [unit]
Customer arrival rate λ	0.001, 0.01, 0.1, 0.5, 0.99
Mean sojourn time $1/\mu$	1
Number of events	100000
Number of simulations	100

The worst case competitive ratio of $ALG(k)$ is $2+\frac{1}{k}$, which can be obtained easily. We consider three cases: $ALG(\infty)$ (all x_i are equal to a), $ALG(1)$, $ALG(10)$. Note that $ALG(\infty)$ is equal to $OWCR$. Let Q_ξ denote the mean of the power-consumption cost per unit time when the algorithm of $\xi \in \{OPT, ALG(\infty), ALG(1), ALG(10)\}$ is employed. Q_ξ's can be derived in a straightforward manner, and we obtain

$$
\begin{aligned}
Q_{\mathrm{OPT}} &= 1 - e^{-(a\lambda+\rho)}, \\
Q_{\mathrm{ALG}(\infty)} &= 1 + (a\lambda - 1)e^{-(a\lambda+\rho)}, \\
Q_{\mathrm{ALG}(1)} &= 1 + 2(a\lambda - 1)e^{-(a\lambda+\rho)} - (a\lambda - 1)e^{-(2a\lambda+\rho)}, \\
Q_{\mathrm{ALG}(10)} &= 1 + (a\lambda - 1)e^{-(a\lambda+\rho)} + (a\lambda - 1)e^{-(ka\lambda+\rho)} - (a\lambda - 1)e^{-((k+1)a\lambda+\rho)}.
\end{aligned}
$$

We omit detailed derivations of the above equations due to the page limitation.

Competitive Ratios. We calculated the average power consumption per unit time of each algorithm and the competitive ratio with Q_{opt}. We use the basic set of parameters shown in Table 1. The analytical results are shown in Table 2. We also conducted experiments with Monte Carlo simulation, in order to validate the results obtained through the analysis. (Skipping the details of simulation results.) The analytical results exhibit good agreement with simulation, and this validates the analytical derivations for Q_ξ's.

We can observe that the performance of these algorithms is almost the same when the system is congested, e.g., $\rho \geq 0.5$. The reason is clearly that it never be OFF. The difference appears when the system becomes slack. Especially $ALG(\infty)$, which must be the optimal worst competitive ratio algorithm ($OWCR$), shows very bad average competitive ratio (e.g., it is around 1.9 for $\rho = 0.01, a = 10$). $ALG(1)$ shows the best average competitive ratio in every case. However it's worst competitive ratio is 3, i.e., it may perform badly for adversary inputs. Our algorithm $ODRA$ performs almost the same as $ALG(1)$ in every case. From these results, we can observe that $ODRA$ has good performance not only in the worst case but also in average case.

Table 2. Analytical results of competitive ratios

ρ	a	$Q_{ALG(\infty)}/Q_{\mathrm{OPT}}$	$Q_{ALG(1)}/Q_{\mathrm{OPT}}$	$Q_{ALG(10)}/Q_{\mathrm{OPT}}$	$Q_{\mathrm{ODRA}}/Q_{\mathrm{OPT}}$		
					$\epsilon = 0.1$	$\epsilon = 0.01$	$\epsilon = 0.001$
CR_{worst}		2	3	2.1	2.1	2.01	2.001
0.001	1	1.49950	1.00075	1.00052	1.001271	1.002656	1.004223
0.001	3	1.74850	1.00336	1.00232	1.005697	1.011871	1.018820
0.001	10	1.90410	1.01350	1.09015	1.022692	1.046865	1.073527
0.01	1	1.49501	1.00739	1.04936	1.012424	1.025660	1.040258
0.01	3	1.73510	1.03264	1.19886	1.053683	1.108154	1.164850
0.01	10	1.86001	1.12344	1.15605	1.188589	1.350484	1.488426
0.1	1	1.45167	1.06483	1.29439	1.099045	1.184070	1.256516
0.1	3	1.60997	1.24109	1.58518	1.304441	1.472030	1.555845
0.1	10	1.49896	1.49896	1.49896	1.400979	1.479744	1.496660
0.5	1	1.29099	1.17649	1.28972	1.189663	1.260917	1.284193
0.5	3	1.23478	1.29557	1.24777	1.200283	1.228261	1.234040
0.5	10	1.02052	1.03682	1.02052	1.019213	1.020304	1.020495
0.99	1	1.15672	1.15662	1.15672	1.127199	1.152416	1.157843
0.99	3	1.05614	1.09169	1.05614	1.052462	1.056840	1.057616
0.99	10	1.00017	1.00032	1.00017	1.000182	1.000183	1.000184

5 Conclusions

We have introduced the concept of slackness degree, which reflects the frequency of requests, and developed the "optimal" online algorithm under this concept. We strongly believe that it is important to consider inputs of problems based on the real world and to design more practical algorithms in online problems. In future work, we plan to consider the randomized version and the multi-state version of this problem.

Acknowledgments. This research has been carried out in collaboration with the "Consumer Electronics Network Eco Management" project sponsored by Panasonic Corporation. We would like to thank the project members, Mr. Toshiya Naka, Mr. Hideyuki Yoshida and Mr. Kazuhiro Aizu. We also would like to thank Prof. Hiroshi Fujiwara of Toyohashi University of Technology for his valuable comments on power consumption problems and helpful discussions.

References

1. Augustine, J., Irani, S., Swamy, C.: Optimal power-down strategies. In: Proc. 45th Symp. Foundations of Computer Science (FOCS), pp. 530–539. IEEE (2004)
2. Ben-David, S., Borodin, A.: A new measure for the study of on-line algorithms. Algorithmica 11, 73–91 (1994)
3. Borodin, A., Irani, S., Raghavan, P., Schieber, B.: Competitive paging with locality of reference. J. Comput. Systems Sci. 50, 244–258 (1995)

4. Boyar, J., Krarup, S., Nielsen, M.N.: Seat reservation allowing seat changes. J. Algorithms 52, 169–192 (2004)
5. Chrobak, M.: Sigact news online algorithms column 8. SIGACT News 36, 67–81 (2005)
6. Chung, E., Benini, L., Bogliolo, A.: Dynamic power management for non-stationary service requests. In: Proceedings of the Design and Automation and Test in Europe Conference and Exhibition, pp. 77–81 (1999)
7. Eggers, S.J., Katz, R.H.: Evaluating the performance of four snooping cache coherency protocols. In: Proc. 16th International Symp. on Computer Architecture (ISCA). IEEE (1989)
8. Irani, S., Gupta, R., Shukla, S.: Competitive analysis of dynamic power management strategies for systems with multiple power savings states. In: DATE 2002: Proceedings of the Conference on Design, Automation and Test in Europe, p. 117. IEEE Computer Society, Washington, DC, USA (2002)
9. Irani, S., Pruhs, K.R.: Algorithmic problems in power management. ACM SIGACT News (2005)
10. Karlin, A.R., Kenyon, C., Randall, D.: Dynamic tcp acknowledgement and other stories about $e/(e-1)$. In: Proc. 33rd STOC, pp. 502–509. ACM (2001)
11. Karlin, A., Manasse, M., McGeoch, L., Owicki, S.: Competitive randomized algorithms for nonuniform problems. Algorithmica 11, 542–571 (1994)
12. Karlin, A., Manasse, M., Rudolph, L., Sleator, D.: Competitive snoopy caching. Algorithmica 3, 79–119 (1988)
13. Kenyon, C.: Best-fit bin-packing with random order. In: Proc. 7th Symp. on Discrete Algorithms (SODA), pp. 359–364. ACM/SIAM (1996)
14. Koutsoupias, E., Papadimitriou, C.: Beyond competitive analysis. SIAM J. Comput. 30, 300–317 (2000)
15. Lotker, Z., Patt-Shamir, B., Rawitz, D.: Rent, lease or buy: Randomized algorithms for multislope ski rental. In: Albers, S., Weil, P. (eds.) 25th International Symposium on Theoretical Aspects of Computer Science (STACS 2008). Leibniz International Proceedings in Informatics (LIPIcs), vol. 1, pp. 503–514. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2008)
16. Panagiotou, K., Souza, A.: On adequate performance measures for paging. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, STOC 2006, pp. 487–496. ACM, New York (2006)
17. Phillips, S., Westbrook, J.: Competitive analysis and beyond. In: Algorithms and Theory of Computation Handbook, ch.10. CRC Press (1999)
18. Ramanathan, D., Irani, S., Gupta, R.: Latency effects of system level power management algorithms. In: Proceedings of the IEEE International Conference on Computer Aided Design (2000)
19. Wolff, R.W.: Stochastic modeling and the theory of queues. Prentice-Hall (1989)