COLLABORATIVE POWER MANAGEMENT IN WPANs USING RATIONAL CONTROLLERS: A CASE STUDY

J. FLINN, P. T. KABAMBA, W.-C. LIN, S. M. MEERKOV, AND C. Y. TANG

Received 27 January 2005

This paper addresses the issue of battery power conservation in wireless personal area networks (WPANs). Specifically, we consider a WPAN, which contains a processor and a disk drive, and develop a collaborative power management technique, which minimizes the total WPAN power consumption. Our approach is based on the theory of rational behavior, which leads to a collaborative architecture where devices in the WPAN are equipped with cooperating rational controllers (RCs). Using, as an example, the Intel 80200 XScale processor and the Hitachi 1 GB microdrive, we show that collaborative power management using RCs offers substantial power saving compared to selfish operation, where each device attempts to minimize only its own power consumption.

1. Introduction

In recent years, the field of mobile computing has seen two important emerging trends. The first trend is the development of powerful specialized computing, communication, and storage devices, such as the Apple iPod [2], handheld computers with Intel 80200 XScale processors [9], and 3G cell phones [19]. The second trend is the development of short-range radio technologies, such as Bluetooth [7], that offer low-power, high-bandwidth connectivity for mobile devices.

These two trends are enabling the development of *wireless personal area networks* (WPANs) that aggregate multiple specialized devices worn or carried by an individual user. The distributed nature of a WPAN offers several important advantages over a traditional monolithic mobile computer. First, each device in a WPAN can be placed at its most convenient location. For example, displays and microphones can be placed near the eyes and mouth, while other devices such as a mobile disk drive can be placed on one's belt or in a pocket. Second, a WPAN can easily add new capabilities by dynamically networking with additional devices. For instance, a laptop or stationary computer could provide additional computing power for demanding applications, while a digital camera or RFID reader could be networked to provide specialized I/O capacity.

The distributed nature of a WPAN also creates important challenges that must be addressed. Chief among such challenges is the conservation of battery power. Indeed, since

Mathematical Problems in Engineering 2005:5 (2005) 491–501
DOI: 10.1155/MPE.2005.491

each device in a WPAN typically utilizes its own energy supply, the productivity of the entire network can be seriously compromised if even a single device runs out of battery power. Hence, power management techniques that extend the operating lifetime of the WPAN as a whole must be developed.

To date, a large body of work is available on power management of *individual* devices in a WPAN. Specifically, for processors, dynamic voltage scaling algorithms that adaptively modify CPU speed have been studied in [12, 23, 25]. These algorithms exploit the fact that the processor power consumption is approximately proportional to the square of the supply voltage and, thus, reducing the voltage or CPU speed by a factor of two results in a reduction of power consumption by a factor of four. For storage devices, power management schemes that control the transitions among active, idle, and sleep modes have been designed and analyzed in [3, 13, 18, 24]. The largest possible power saving that can be achieved using these schemes has been investigated in [4]. Finally, for communication devices, power-efficient protocols that periodically or adaptively enable/disable client receivers in a wireless local area network have been proposed in [1, 8, 10, 17].

The above-mentioned power management techniques are *selfish* in the sense that each device attempts to minimize only its own power consumption. When these devices are connected to form a WPAN, interactions among them can actually *increase* the power consumed by the WPAN as a whole, compared to not utilizing any power management. Indeed, we show in this paper that such a phenomenon can be observed in a WPAN as simple as one consisting of only a processor and a disk drive. Therefore, it is necessary to develop *collaborative* power management techniques, which allow the devices to cooperate and account for the interactions among them, so that the total WPAN power consumption is reduced.

In this paper, we focus on a WPAN containing a processor and a disk drive and develop a collaborative power management technique, whereby the two devices cooperate, rather than compete, to ensure that the total power consumed by the WPAN is minimized. Our approach is based on the *theory of rational behavior* (TRB) [5, 6, 11, 16, 20, 21, 22], which offers methods for designing decentralized systems where each component adjusts its individual decision so that the system's penalty function is optimized. This theory leads to a collaborative architecture where the devices are equipped with *rational controllers* (RCs), the interaction of which minimizes the total WPAN power consumption. Using, as an example, the Intel 80200 XScale processor and the Hitachi 1 GB microdrive, our analysis shows that collaborative power management using RCs yields up to 38% power saving in comparison with selfish operation of the devices.

The outline of this paper is as follows: Section 2 briefly reviews the main results of TRB. In Section 3, we present a case study on collaborative power management using RCs for a WPAN having only a processor and a disk drive. The case study includes a power-efficiency comparison between selfish and collaborative power management. Finally, the conclusion and future work are formulated in Section 4.

2. Theory of rational behavior

The *theory of rational behavior* (TRB) emerged in the 60's in the work of mathematicians and physicists, primarily from the former USSR (see [21, 22] for a comprehensive

treatment). Initially, this theory was intended to explain the simplest form of animal behavior [20] and collective behavior of social insects modeled as automata [5, 6, 16]. The main ideas and structure of TRB are summarized below.

2.1. Individual rational behavior. Although there are several ways of defining individual rational behavior in the framework of TRB, the one described here is the simplest [11].

Consider a decision space X and a scalar penalty function $\varphi(x) > 0$, $x \in X$, defined on it. Introduce a dynamical system with trajectories depending on the function $\varphi(x)$ and a positive integer N. Denote these trajectories as $x_{\varphi(x),N}(x_0, t_0, t)$, where x_0 is the initial decision at time t_0. We say that the dynamical system with trajectories $x_{\varphi(x),N}(x_0, t_0, t)$ exhibits *rational behavior* if the following two properties hold.

(a) *Ergodicity*. For every initial decision x_0 and any set B in X, there exists a time moment t' such that $x_{\varphi(x),N}(x_0, t_0, t') \in B$ (or, for probabilistic systems, $\mathrm{P}\{x_{\varphi(x),N}(x_0, t_0, t') \in B\} > 0$).

(b) *Rationality*. During a sufficiently long time interval, the ratio of residence times in decision $x_1 \in X$ and $x_2 \in X$ is greater than 1, if $\varphi(x_1) < \varphi(x_2)$. Moreover, this ratio tends to infinity as $N \to \infty$.

These properties imply that the behavior is rational if all possible decisions are explored and those with smaller penalties are selected more often than those with larger ones. Clearly, parameter N can be viewed as a measure of rationality.

The literature offers many examples of dynamical systems with trajectories exhibiting rational behavior [11, 21, 22]. One of them, referred to as the *ring element*, can be described as follows. Let $X = [0, 1)$ and let $\varphi(x) > 0$ be a continuous scalar function. Consider the following dynamical system:

$$\dot{x} = \varphi^N(\{x\}), \tag{2.1}$$

where $\{x\}$ denotes the fractional part of x and N is a positive integer. Clearly, trajectories of this system satisfy both ergodicity and rationality properties. The ergodicity is satisfied because all subsets of X are visited. The rationality property is met since the residence time in the vicinity of each decision x_1 can be evaluated as

$$T_{N,x_1} \approx \frac{1}{\varphi^N(x_1)}. \tag{2.2}$$

Therefore, for any $x_1, x_2 \in X$,

$$\frac{T_{N,x_1}}{T_{N,x_2}} = \left(\frac{\varphi(x_2)}{\varphi(x_1)}\right)^N > 1 \quad \text{if } \varphi(x_1) < \varphi(x_2) \tag{2.3}$$

and

$$\frac{T_{N,x_1}}{T_{N,x_2}} \longrightarrow \infty \quad \text{as } N \longrightarrow \infty. \tag{2.4}$$

Note that the behavior of this rational element is akin to simulated annealing [14].

2.2. Collective behavior of rational elements. Consider a group of M rational elements with level of rationality N. Assume each of the elements has two decisions in its decision space: x_1 and x_2. Assume also that the collective penalty function is given by $f(\nu)$, where ν is the fraction of elements in state x_1. The individual penalty for each element is given by $f(\nu)$ if the element is in state x_1 and it becomes $f(\nu - 1/M)$ if it changes its state to x_2. The question addressed is: will the collective converge to the point ν^*, where $f(\nu)$ reaches its global minimum? The answer to this question is as follows [11, 16]. If $\lim_{N,M\to\infty} N/M > 0$, convergence to ν^* takes place. On the other hand, if $\lim_{N,M\to\infty} N/M = 0$, the group converges to $\nu = 1/2$, no matter how large $f(1/2)$ may be. Note that $\nu = 1/2$ corresponds to the situation where entropy is maximized and, thus, if N is not large enough in comparison with M, the collective behaves like a statistical mechanical gas, rather than a purposeful group. This result indicates that parameters of the rational controllers must be carefully selected so that the behavior of the group is rational.

3. Case study: processor and disk drive

This Section presents a case study on collaborative power management in wireless personal area networks (WPANs) using rational controllers (RCs). Although our eventual goal is to treat WPANs having multiple devices, in this case study we consider a WPAN containing only a processor and a disk drive that, together, have to perform computing and read/write tasks (see Figure 3.1). We first describe the models for job arrival, the processor, and the disk drive. We then introduce two architectures for power management using RCs and evaluate their efficacy under several job arrival scenarios.

3.1. Modeling.

Job arrival. We assume a sequence of jobs, identified by the index k, such that the kth job requires $w_p(k)$ cycles of computing and $w_d(k)$ bytes of reading from/writing to the disk drive (see Figure 3.1). The sequences $\{w_p(k),\ k = 1,2,\ldots\}$ and $\{w_d(k),\ k = 1,2,\ldots\}$ are modeled as random processes taking positive integer values.

Processor. We assume that the processor is capable of operating at n_p pairs of frequencies and voltages, denoted $(f_1,v_1),(f_2,v_2),\ldots,(f_{n_p},v_{n_p})$. For each pair (f_i,v_i), let p_i denote the power consumed. We also assume that both frequency and voltage are kept constant during the processing of each job, and that they are allowed to change upon the completion of a job. Thus, we denote the frequency, voltage, and power used during the processing of the kth job as $f(k)$, $v(k)$, and $p(k)$, respectively. Therefore, the energy consumed by the processor to complete the kth job is given by

$$e_p(k) = p(k)\frac{w_p(k)}{f(k)}. \tag{3.1}$$

Disk drive. The disk drive is assumed to have three modes of operation: *active*, *idle*, and *sleep*. Whenever a read/write job is requested, the disk drive switches to the active mode, reading/writing at a fixed data rate of s bytes per second. Once the job is completed, it switches to the idle mode, stays there for τ seconds, and then switches to the sleep

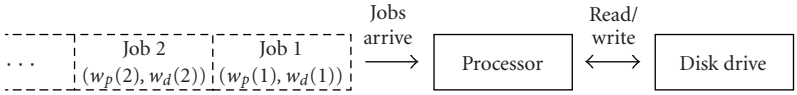

Figure 3.1. WPAN containing a processor and a disk drive.

mode. The parameter τ, referred to as the *threshold*, is allowed to take n_d values, denoted $\tau_1, \tau_2, \ldots, \tau_{n_d}$. We also assume that the threshold is kept constant during the processing of each job, and is allowed to change upon the completion of a job. Thus, we denote the threshold used during the processing of the kth job as $\tau(k)$.

Labeling the active, idle, and sleep modes by 1, 2, and 3, respectively, the power/energy consumption of the disk drive is characterized by the matrix

$$\mathbf{\Phi} = \begin{bmatrix} \phi_1 & \Phi_{12} & \Phi_{13} \\ \Phi_{21} & \phi_2 & \Phi_{23} \\ \Phi_{31} & \Phi_{32} & \phi_3 \end{bmatrix}, \tag{3.2}$$

where $\phi_i > 0$ is the power used in mode i, and $\Phi_{ij} > 0$ is the energy used to switch from mode i to j. It is assumed that

$$\begin{aligned} \phi_1 > \phi_2 > \phi_3, \qquad & \Phi_{13} \le \Phi_{12} + \Phi_{23}, \\ \Phi_{31} \le \Phi_{32} + \Phi_{21}, \qquad & \Phi_{12} + \Phi_{21} < \Phi_{13} + \Phi_{31}. \end{aligned} \tag{3.3}$$

The second and third inequalities in (3.3) imply that switching directly between the active and sleep modes requires no more energy than going through the idle mode. The last inequality in (3.3) implies that switching from the active mode to idle and back to active consumes less energy than switching from the active mode to sleep and back to active. As it follows from the above, the energy consumed by the disk drive during the processing of the kth job is given by

$$e_d(k) = \begin{cases} \phi_1 \dfrac{w_d(k)}{s} + \Phi_{12} + \phi_2 \left(\dfrac{w_p(k)}{f(k)} - \dfrac{w_d(k)}{s} \right) + \Phi_{21}, & \text{if } \dfrac{w_p(k)}{f(k)} - \dfrac{w_d(k)}{s} \le \tau(k), \\ \phi_1 \dfrac{w_d(k)}{s} + \Phi_{12} + \phi_2 \tau(k) + \Phi_{23} + \cdots + & \\ \phi_3 \left(\dfrac{w_p(k)}{f(k)} - \dfrac{w_d(k)}{s} - \tau(k) \right) + \Phi_{31}, & \text{if } \dfrac{w_p(k)}{f(k)} - \dfrac{w_d(k)}{s} > \tau(k), \end{cases} \tag{3.4}$$

where the first alternative in (3.4) does not trigger the sleep mode, and the second one does.

3.2. Control architectures. We assume that the processor and the disk drive each has an RC. Upon completion of each job, the processor controller selects the pair of frequency and voltage to be used for the next job. Hence, the processor controller determines the sequence $\{(f(k), v(k)),\ k = 1, 2, \ldots\}$. Similarly, upon completion of each job, the disk drive

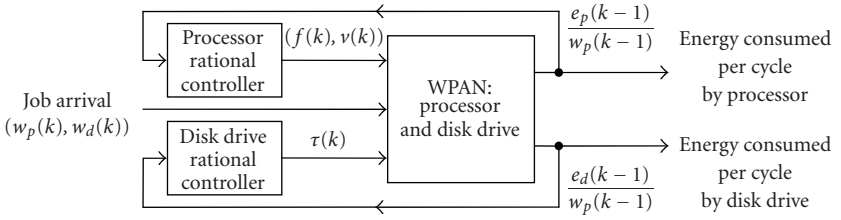

Figure 3.2. Selfish architecture.

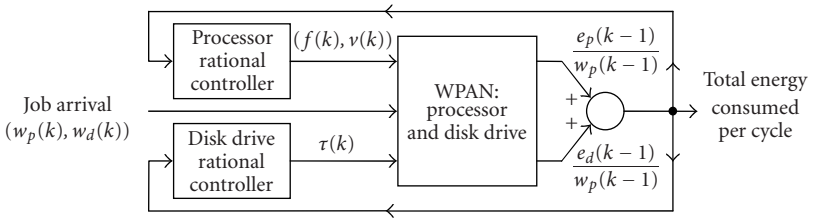

Figure 3.3. Collaborative architecture.

controller selects the threshold to be used for the next job and, thus, determines the sequence $\{\tau(k),\ k = 1,2,\ldots\}$.

These two controllers can be used in two different architectures, namely: *selfish* and *collaborative*. Figure 3.2 illustrates the selfish architecture, where each controller takes its decision based on the energy per cycle *consumed by an individual device* over the just completed job. Specifically, the processor controller selects $(f(k), v(k))$ based on $e_p(k-1)/w_p(k-1)$, whereas the disk drive controller selects $\tau(k)$ based on $e_d(k-1)/w_p(k-1)$.

In contrast, Figure 3.3 illustrates the collaborative architecture, where each controller takes its decision based on the energy per cycle *consumed by the WPAN as a whole* over the just completed job. Specifically, the processor and disk drive controllers select $(f(k), v(k))$ and $\tau(k)$, respectively, based on $e_p(k-1)/w_p(k-1) + e_d(k-1)/w_p(k-1)$. Note that, in general, the selection can be based on $c_p(e_p(k-1)/w_p(k-1)) + c_d(e_d(k-1)/w_p(k-1))$, where c_p and c_d are weighting coefficients representing the relative importance of the processor and disk drive, respectively.

3.3. Rational controllers. As shown in Figures 3.2 and 3.3, the RCs have energy per cycle as their inputs and decisions as their outputs. The dynamics of each controller are described by a stochastic automaton with n_p or n_d states for the processor or disk drive controller, respectively. The transition diagram for each automaton is fully connected, that is, transitions from any state to all other states are possible. The automaton leaves its state, occupied at time $k-1$, with probability $P^N(e(k-1)/w_p(k-1))$, or remains in it with probability $1 - P^N(e(k-1)/w_p(k-1))$; upon leaving its current state, the automaton transits to all other states equiprobably. Here, $P : (0,\infty) \to (0,1)$ is a strictly increasing *penalty function*, positive integer N is the *level of rationality*, and $e(k-1)$ is the energy

consumed which depends on the architecture. Specifically, for the selfish architecture, $e(k-1) = e_p(k-1)$ and $e(k-1) = e_d(k-1)$ for the processor and disk drive controllers, respectively; and for the collaborative architecture, $e(k-1) = e_p(k-1) + e_d(k-1)$ for both the processor and disk drive controllers.

The penalty function P and the level of rationality N are the parameters of the controllers to be selected so that the dynamics and steady states of the closed-loop system are desirable. Note that these controllers exhibit rational behavior in the sense of Section 2. Indeed, ergodicity is ensured by the fully connected nature of the transition diagrams, and rationality is ensured by the monotonically increasing nature of the penalty function.

3.4. Scenarios. To evaluate the efficacy of the selfish and collaborative architectures as well as the behavior of the RCs, we consider a sequence of 10 000 jobs and four scenarios for computing job arrival: (1) *high load*, characterized by $w_p(k) = 10^9$; (2) *medium load*, with $w_p(k) = 5 \times 10^8$; (3) *low load*, with $w_p(k) = 10^8$; and (4) *variable load*, defined as high load for the first 3334 jobs, medium load for the next 3333 jobs, and low load for the last 3333 jobs. For each of these four scenarios, we assume that $w_d(k) = 4 \times 10^5$.

The processor under consideration is the Intel 80200 XScale processor [9]. Although this processor can operate at seven frequencies, we consider only the lowest and highest frequencies, that is, 333 MHz and 733 MHz, respectively. For the 333 MHz frequency, the voltage is 1.0 V, whereas for the 733 MHz frequency, the voltage is 1.5 V. As indicated in [15], the power consumption at (333 MHz, 1.0 V) is 0.174 W, and at (733 MHz, 1.5 V) is 0.8295 W.

The disk drive considered is the Hitachi 1 GB microdrive. This disk drive is capable of reading/writing at a fixed data rate of 4 MB/s. Although it is capable of operating in four modes, we consider only the active, idle, and sleep modes. For this device, the power/energy consumption matrix $\mathbf{\Phi}$ defined in (3.2) has the following numerical values [4]:

$$\mathbf{\Phi} = \begin{bmatrix} 1.18\,\mathrm{W} & 0\,\mathrm{J} & 0.3\,\mathrm{J} \\ 0.09\,\mathrm{J} & 0.76\,\mathrm{W} & 0.3\,\mathrm{J} \\ 1\,\mathrm{J} & 0.9\,\mathrm{J} & 0.08\,\mathrm{W} \end{bmatrix}. \tag{3.5}$$

Although this disk drive can operate with arbitrary thresholds, we consider only two: 0.2 second and 1.0 second.

As specified in Section 3.3, the processor and disk drive are each equipped with an RC. Here, we consider two levels of rationality, $N = 3$ and $N = 9$, and choose the penalty function for both of the controllers as

$$P\left(\frac{e(k-1)}{w_p(k-1)}\right) = \frac{1 + \operatorname{erf}\left(1100\left(e(k-1)/w_p(k-1)\right) - 0.0029\right)}{2}. \tag{3.6}$$

3.5. Results. For the purpose of comparison, we evaluate the performance of constant parameter strategies, defined by keeping constant, throughout all the jobs, the frequency and voltage of the processor, and the threshold of the disk drive. Table 3.1 summarizes the simulation results, showing the energy consumption, in joules, for each of the four loads

Table 3.1. Energy consumption for constant parameter strategies.

Power management strategies		High load $w_p(k) = 10^9$	Medium load $w_p(k) = 5 \times 10^8$	Low load $w_p(k) = 10^8$	Variable load
Constant parameter strategy	(333 MHz, 1.0 V) & 0.2 s	**23087.6**	19273.8	16222.8	19528.4
	(333 MHz, 1.0 V) & 1.0 s	28527.6	24713.8	4124.8	19123.0
	(733 MHz, 1.5 V) & 0.2 s	27867.9	21664.0	**3488.5**	17674.5
	(733 MHz, 1.5 V) & 1.0 s	33307.9	**12162.4**	**3488.5**	**16321.3**

Table 3.2. Probability of residence for selfish and collaborative architectures.

Power management strategies		High load $w_p(k) = 10^9$	Medium load $w_p(k) = 5 \times 10^8$	Low load $w_p(k) = 10^8$
Selfish RCs	$N = 3$	$\begin{bmatrix} 0.98 & 0.02 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0.71 & 0.29 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0.39 & 0.61 \\ 0 & 0 \end{bmatrix}$
	$N = 9$	$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0.93 & 0.07 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0.21 & 0.79 \\ 0 & 0 \end{bmatrix}$
Collaborative RCs	$N = 3$	$\begin{bmatrix} 0.85 & 0.04 \\ 0.1 & 0.01 \end{bmatrix}$	$\begin{bmatrix} 0.02 & 0.01 \\ 0.02 & 0.95 \end{bmatrix}$	$\begin{bmatrix} 0.18 & 0.19 \\ 0.32 & 0.31 \end{bmatrix}$
	$N = 9$	$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0.01 & 0 \\ 0 & 0.99 \end{bmatrix}$	$\begin{bmatrix} 0.07 & 0.09 \\ 0.42 & 0.42 \end{bmatrix}$

and each of the four constant parameter strategies. For each load, the smallest energy consumption is shown in bold. For instance, for high load, the smallest energy consumption is 23087.6 J, achieved by using the combination of parameters (333 MHz, 1.0 V) and 0.2 second. The results in Table 3.1 will be used below as a benchmark against which to compare the efficacy of the selfish and collaborative architectures.

Table 3.2 summarizes the simulation results in terms of probability of residence in the various states of the stochastic automata associated with the controllers. The results are displayed for high, medium, and low loads, for the selfish and collaborative architectures, and for $N = 3$ and $N = 9$. Each entry of the table is a 2×2 matrix, with rows corresponding to the states of the processor controller ((333 MHz, 1.0 V) and (733 MHz, 1.5 V), resp.), and columns corresponding to the states of the disk drive controller (0.2 second and 1.0 second, resp.). The entries of each of these 2×2 matrices are probabilities of residence in the corresponding states. For instance, the $(1,2)$ entry of each of the matrices is the probability of choosing the combination of parameters: (333 MHz, 1.0 V) for the processor, and 1.0 second for the disk drive.

Table 3.3. Energy consumption for selfish and collaborative architectures.

Power management strategies		High load $w_p(k) = 10^9$	Medium load $w_p(k) = 5 \times 10^8$	Low load $w_p(k) = 10^8$	Variable load
Selfish RCs	$N = 3$	23169.2	20871.0	8784.9	17691.6
	$N = 9$	23087.6	19678.5	6643.6	16479.1
Collaborative RCs	$N = 3$	23903.1	12598.2	5820.3	14173.9
	$N = 9$	23087.6	12165.1	4420.3	13316.1

Based on Tables 3.1 and 3.2, the following observations can be made.

(a) *Selfish RCs do exhibit rational behavior*. Indeed, it is well known in the voltage scaling literature that processors operating at low frequency and low voltage consume less power. Also, note from (3.1) that the threshold selected by the disk drive does not affect the energy consumed by the processor. Thus, for the processor to be rational, it should select (333 MHz, 1.0 V). Subsequently, for the disk drive to be rational, it should select the threshold that yields the smallest energy consumption, given the parameters selected by the processor. We see that this is indeed the case. For instance, consider high load with $N = 3$. The processor almost always selects (333 MHz, 1.0 V) and the disk drive most frequently selects 0.2 second, as predicted by Table 3.1.

(b) *Collaborative RCs also exhibit rational behavior*. For collaborative RCs to be rational, they should most frequently select the combination of parameters that corresponds to the bold entries in Table 3.1. We see that this is indeed the case. For instance, for medium load, according to Table 3.1, the best combination of parameters is (733 MHz, 1.5 V) and 1.0 second. For $N = 3$, Table 3.2 shows that this combination is selected with probability 0.95.

(c) *Regardless of architecture, increasing N increases the probability of residing in an optimal state*. This is true even when the optimal state is not unique. For instance, for low load and collaborative architecture, Table 3.1 shows that selecting (733 MHz, 1.5 V) for the processor minimizes energy consumption regardless of the threshold of the disk drive. Table 3.2 shows that, for $N = 3$, the probability of selecting (733 MHz, 1.5 V) for the processor is 0.63, split equally between the two thresholds. However, for $N = 9$, this probability of residence increases to 0.84, and is still split equally.

Table 3.3 summarizes the simulation results in terms of energy consumption for high, medium, low, and variable loads, for selfish and collaborative architectures, and for $N = 3$ and $N = 9$.

Comparing Tables 3.1 and 3.3, the following observations can be made.

(d) *Regardless of architecture, increasing N improves power efficiency*. Obviously, this is a consequence of observation (c) above, since residing more often in an optimal state decreases energy consumption.

(e) *For variable load, collaborative RCs outperform all constant parameter strategies*. This is because collaborative RCs are capable of adapting to a better frequency and threshold when the load varies. The energy saving achieved by collaborative RCs compared to constant parameter strategies is at least 18.4% and up to 31.8%.

(f) *For all loads, collaborative RCs outperform their selfish counterparts.* The energy saving can be as high as 38.2%.

4. Conclusion and future work

In the above case study, we have shown that RCs do exhibit rational behavior. Moreover, increasing the level of rationality increases the probability of residence in an optimal state, which in turn improves power efficiency of the WPAN. Finally, we have shown that collaborative RCs offer significant power-efficiency improvement over both constant parameter strategies and selfish RCs.

Future work on collaborative power management using RCs will focus on extending the results of this paper to a general WPAN consisting of multiple heterogeneous devices, operating under various kinds of inter-device traffics. The extension will include both theoretical analysis and experimental verification.

References

[1] M. Anand, E. B. Nightingale, and J. Flinn, *Self-tuning wireless network power management*, Proc. IEEE/ACM International Conference on Mobile Computing and Networking, California, 2003, pp. 176–189.

[2] Apple Corporation, *Apple iPod technical Specifications*, http://www.apple.com/ipod/, 2005.

[3] F. Douglis, P. Krishnan, and B. Marsh, *Thwarting the power-hungry disk*, Proc. Winter USENIX Conference, California, 1994, pp. 292–306.

[4] J. Flinn, P. T. Kabamba, S. M. Meerkov, and C. Y. Tang, *Power-improvement capacity of computing and communication devices*, Tech. Rep. CCR 04-01, Department of Electrical Engineering and Computer Science, University of Michigan, Michigan, January 2004.

[5] I. M. Gel'fand, I. I. Pyatetsky-Shapiro, and M. L. Tsetlin, *On some classes of games and games of automata*, Dokl. Akad. Nauk SSSR **152** (1963), no. 4.

[6] S. L. Ginsberg and M. L. Tsetlin, *On modeling of collective behavior of automata*, Problems of Information Transmission **1** (1965), no. 2.

[7] J. C. Haartsen, *The Bluetooth radio system*, IEEE Personal Communications **7** (2000), no. 1, 28–36.

[8] IEEE Computer Society Local Metropolitan Area Network Standards Committee, *Wireless LAN medium access control (MAC) and physical layer (PHY) specifications*, IEEE Standard 802.11, 1997.

[9] Intel Corporation, *Intel 80200 processor based on Intel XScale microarchitecture*, http://www.intel.com/design/iio/manuals/273411.htm, 2005.

[10] R. Krashinsky and H. Balakrishnan, *Minimizing energy for wireless web access with bounded slowdown*, Proc. IEEE/ACM International Conference on Mobile Computing and Networking, Georgia, 2002, pp. 119–130.

[11] S. M. Meerkov, *Mathematical theory of behavior-individual and collective behavior of retardable elements*, Math. Biosci. **43** (1979), no. 1-2, 41–106.

[12] P. Mejia-Alvarez, E. Levner, and D. Mossé, *Adaptive scheduling server for power-aware real-time tasks*, ACM Transactions on Embedded Computing Systems **3** (2004), no. 2, 284–306.

[13] H. Okamura, T. Dohi, and S. Osaki, *On the effect of power saving by auto-sleep function of a computer system I—Modeling by a renewal process*, Transactions of the Information Processing Society of Japan **39** (1998), no. 6, 1858–1869.

[14] R. H. J. M. Otten and L. P. P. P. van Ginneken, *The Annealing Algorithm*, Kluwer Academic, Massachusetts, 1989.

[15] C. Pereira, V. Raghunathan, S. Gupta, R. Gupta, and M. Srivastava, *A software architecture for building power aware real time operating systems*, Technical Report #02-07, Department of Information and Computer Science, University of California, California, 2002.
[16] B. G. Pittel, *On a probabilistic model of collective behavior*, Probl. Inf. Transm. **3** (1967), no. 3.
[17] T. Simunic, L. Benini, P. Glynn, and G. De Micheli, *Dynamic power management for portable systems*, Proc. IEEE/ACM International Conference on Mobile Computing and Networking, Massachusetts, 2000, pp. 11–19.
[18] ———, *Dynamic power management of laptop hard disk*, Proc. Design, Automation and Test in Europe Conference and Exhibition, Paris, 2000, p. 736.
[19] The 3rd Generation Partnership Project, *3GPP home page*, http://www.3gpp.org, 2005.
[20] M. L. Tsetlin, *Automata and modeling of simplest forms of behavior*, Soviet Mathematic Uspekhi **18** (1963), no. 4.
[21] ———, *Research on Automata Theory and Simulation of Biological Systems*, Nauka, Moscow, 1969.
[22] V. I. Varshavsky, *Collective Behavior of Automata*, Nauka, Moscow, 1973.
[23] G. F. Welch, *A survey of power management techniques in mobile computing operating systems*, Operating Systems Review **29** (1995), no. 4, 47–56.
[24] F. Zheng, N. Garg, S. Sobti, C. Zhang, R. E. Joseph, A. Krishnamurthy, and R. Y. Wang, *Considering the energy consumption of mobile storage alternatives*, Proc. 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Florida, 2003, pp. 36–45.
[25] D. Zhu, R. Melhem, and B. R. Childers, *Scheduling with dynamic voltage/speed adjustment using slack reclamation in multi-processor real-time systems*, IEEE Trans. Parallel Distrib. Syst. **14** (2003), no. 7, 686–700.

J. Flinn: Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA
E-mail address: jflinn@eecs.umich.edu

P. T. Kabamba: Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA
E-mail address: kabamba@umich.edu

W.-C. Lin: Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA
E-mail address: wenchiao@umich.edu

S. M. Meerkov: Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA
E-mail address: smm@eecs.umich.edu

C. Y. Tang: Honeywell Labs, Minneapolis, MN 55418, USA
E-mail address: choon.yik.tang@honeywell.com