Power-Aware Optimal Checkpoint Intervals for Mobile Consumer Devices

Sung-Hwa Lim, Se Won Lee, Byoung-Hoon Lee, and Seongil Lee, *Member*, IEEE

**Abstract — ***It is highly desired to employ a checkpoint and rollback scheme on a mobile consumer device for fault tolerance, because spatial fault-tolerant schemes such as hardware replications cannot be used. We can reduce the loss of computation of a process in the presence of failures by periodically saving the process's state on stable storage as a checkpoint and rolling back to the latest checkpoint when a failure has occurred. However, a mobile consumer device is not considered to have sufficiently large and stable storage to store its checkpoint data. Therefore, a remote checkpoint technique is preferred in wireless environments in which the checkpoint data of a mobile device is kept in a remote checkpoint server instead of the mobile device. Dense checkpoints incur frequent wireless transmissions whereas coarse checkpoints increase the loss of computation. Many checkpoint research papers have tried to minimize the expected execution time. However, an effective solution which minimizes the energy expenditure should be also provided, because battery power is one of the most critical resources of a mobile device. In this paper, we propose the energy-aware optimal checkpoint interval in wireless remote checkpoint environments. We develop a stochastic model with Markov chain and then derive the optimal value. For the practical evaluation, we conducted not only analytical estimation and simulation but also experimental measurements by implementing on a real test-bed.*[1].

Index Terms — Mobile Consumer Device, Fault-tolerance, Energy Conservation, Wireless Networks.

I. INTRODUCTION

In ubiquitous environments, mobile consumer electronic devices are getting smarter and smaller to provide interactive intelligent services to users by collaborating with internet services. For example, ubiquitous well-being services can be provided by employing various wearable biological sensing devices, which collect user's heartbeat, body heat, and so on, and transmit the collected information to the user's smart phone or internet health-care servers through wireless communications [1].

Although computer hardware is more reliable in this decade compared to previous decades, smart mobile consumer devices such as wearable smart sensors (e.g., smart pendants [1]), computational RFIDs, or smart phones tend to fail for a variety of reasons. The high complexity of emerging software architectures increases the occurrence of transient faults during the computation. Memory allocation faults can easily occur on a mobile device that usually has a small volume of memory, because small memory tends to be fully occupied by memory leakage bugs quicker than large memory does. External damage or battery drain also easily occurs on mobile devices.

Therefore, an effective fault tolerant strategy should be employed in mobile consumer devices. However, employing hardware-based fault-tolerant schemes (e.g., hardware replications) to mobile devices is not realistic [2], though they are quite simple and effective. Therefore, software-based fault-tolerance techniques that do not require additional hardware resources are desirable to mobile devices. A checkpoint and rollback recovery technique is an efficient software-based fault tolerance strategy that does not require additional hardware resources [3]. We can reduce the loss of computation of a process by periodically saving the process's state on stable storage as a checkpoint and rollback to the most recent checkpoint when a failure has occurred [4], [5].

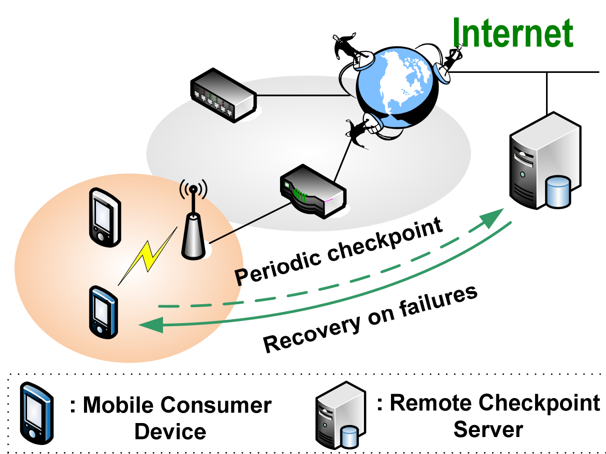

Fig. 1. An example of a remote checkpoint strategy.

In wireless environments, a mobile consumer device is not considered to have sufficiently large and stable storage to store its checkpoint data, because mobile devices have a lack of stable storage, small memory, limited battery capacity, and are prone to external damage. Therefore, it is better to save the checkpoint data of a mobile device into the stable storage of a remote server through a wireless interface and download the last checkpoint from the server when the device has failed [6], [7], [8], as shown in Fig. 1.

[1] This work was supported by Key Research Institute Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2011-0018394)

Sung-Hwa Lim, and Seongil Lee are with Department of Systems Management Engineering, Sungkyunkwan University, Suwon, Korea (e-mail: sunghwa@skku.edu, silee@skku.edu).

Se Won Lee is with Department of Business Administration, Dongguk University-Seoul, Seoul, Korea (e-mail: swlee94@dongguk.edu).

Byoung-Hoon Lee is with the Institute for Information and Electronics Research, Inha University, Incheon, Korea (e-mail: componer@inha.ac.kr).

Contributed Paper
Manuscript received 10/15/11
Current version published 12/27/11
Electronic version published 12/27/11.

0098 3063/11/$20.00 © 2011 IEEE

However, remote checkpointing incurs wireless transmissions that greatly increase the energy expenditure of a mobile device. It is clear that much greater energy is expended in transmitting the data through a wireless interface than in accessing local memory.

Dense checkpoints incur large energy dissipation because frequent wireless transmissions are required. Coarse checkpoints also bring out large energy expenditures because the loss of computation increases, which lengthens the total execution time. Therefore, we need to find an optimal point that minimizes the energy expenditure.

Previous research papers tried to minimize the expected execution time that includes the time expended for the checkpoint establishment, the rollback recovery, and useful computation. However, energy expenditure is a big concern for mobile devices nowadays, because battery power of mobile devices is now one of the most crucial resources. In this paper, we propose a method for determining an approximation to the optimal checkpoint interval that minimizes the energy expenditure in wireless remote checkpoint environments. To evaluate the proposed solution, we conducted performance analyses and simulations. For the practical evaluation, we also conducted experimental measurements by implementing on a real test-bed. As far as we know, it is the first paper that deals with the energy optimal checkpoint interval for a mobile device in wireless remote checkpoint environments.

The outline of the paper is organized as follows. Section II discusses some related work and Section III presents our system model. Then, in Section IV we propose an approximated solution to find out the energy optimal checkpoint interval. The performance of the proposed solution was evaluated with analytical estimates, simulations and experimental measurements in Section V. Finally, we conclude our work in Section VI.

II. Related Work

Checkpoint and rollback recovery techniques have been widely studied for the fault tolerance of systems, which is a cost-effective software-based method against transient and intermittent faults. A checkpoint is a snapshot of the state of a process which is saved in stable storage and is loaded into the memory when the system fails [4]. By restarting from the point of the last saved checkpoint instead of the beginning, the loss of useful computation incurred by the failure can be substantially reduced [5].

Employing a checkpoint and rollback scheme on a mobile device is inevitable for fault tolerance, because spatial fault-tolerant schemes such as hardware replications cannot be applied [2]. However, a mobile device generally does not have sufficiently large and stable storage to keep checkpoint data. Therefore, many algorithms use the storage of a remote checkpoint server as a stable storage area so that a mobile device may store its checkpoint data into the servers via the wireless communication and restore it from the server in the presence of failures [6], [7], [8].

We name this strategy *remote checkpointing* in this paper. It is obvious that the storage of a fixed server is more reliable than that of a mobile device as well as the server has much larger capacity. Moreover, when a mobile device has failed permanently, its checkpoint data stored in the server can be migrated to another mobile device and restarted. Most research on the remote checkpoint strategy have been focused on reducing message overhead between mobile hosts and the remote checkpoint server [6], [7], [9], [10].

Establishing or loading a checkpoint expends overhead time and energy. Therefore, it has been one of the main goals of the research on checkpoint and rollback schemes to reduce the total overhead which consists of the loss of computation and the checkpoint overhead [4], [3], [11]. Finding out an appropriate time interval between consecutive checkpoints (i.e., checkpoint interval) is very critical to reduce the total overhead. If the checkpoint interval is too long, the loss of computation incurred by failures will be greatly increased. On the other hand, if the intervals are too short (i.e., frequent checkpoints), the checkpoint overhead will become too high.

Many research papers have proposed approximated methods to find out the optimal checkpoint interval that minimizes the expected execution time. Young has proposed a first order approximation to compute the optimal checkpoint interval according to the failure rate and the checkpoint overhead [3] for traditional checkpoint systems. Daly proposed a perturbation solution providing a higher order approximation [11] than Young's approximation [3]. Zhan et Al. presented a heuristic method for remote checkpoint systems which dynamically adjusts the checkpoint frequency according to variety of failures when the failure rate is assumed to change over time in wireless networks [2]. Men et al. proposed a remote checkpoint method to obtain the appropriate checkpoint interval by considering not only the failure rate and checkpoint overhead but also the handoff rate of a mobile device in cellular networks when the mobile device frequently moves across the cellular areas [12]. George et al. proposed an aperiodic remote checkpoint scheme in which a mobile device takes checkpoints only when its handoff rate exceeds a predefined threshold value instead of taking checkpoints periodically [13].

Nowadays, battery power is one of the most crucial resources of mobile devices. The previous checkpoint research to find out the optimal checkpoint interval is focused on reducing the expected execution time whereas nowadays energy expenditure is also very critical to the battery life of a mobile device. Therefore, we present an approximation of the optimal checkpoint interval to minimize energy expenditure of a mobile device in a wireless remote checkpoint environment.

III. System Model

When a mobile device fails, its local state is corrupted and all the useful computation done before the failure occurred is lost [4]. To relive this loss of computation, the mobile device periodically saves the current state of its running process as checkpoints, and recovers from a failure by a rollback to the most recent checkpoint. A checkpoint is a copy of current process's state, which is stored on stable storage. An example of the checkpoint and rollback process is shown in Fig. 2. In this paper, uni-process application model is assumed.

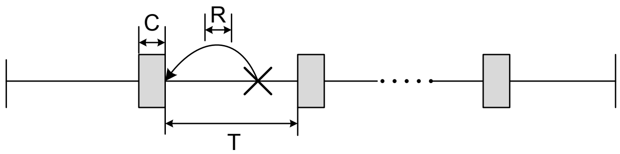

Fig. 2. An example of a checkpoint and rollback scheme.

A checkpoint is said to have been established, if the execution of the process can start over from this checkpoint by a rollback and recovery in the presence of failure. Establishing a checkpoint incurs time overhead denoted by C, and rolling back to the last checkpoint when a failure has occurred requires time overhead denoted by R. We assume that C and R are constants for simplicity, though a more elaborate model may assume C and R to be some function of time.

The execution of a process can be considered as a series of intervals, each of which begins immediately after a checkpoint is established and ends after the next checkpoint is successively established. We assume that every interval between consecutive checkpoints for a process has the same time length T of useful computation as presented in [4]. At least, time T+C is required to complete one checkpoint interval is even if any failures are not occurred.

A service failure of a system is caused by one or more faults. Although a failure does not always happen on a machine whenever a fault occurs [14], we assume that it occurs once a fault has occurred for purposes of simplicity. Faults that occur on a mobile device are assumed to be transient, and are governed by a Poisson process with rate λ. In case of failure, the mobile device is assumed to lose current states of running processes in the memory. The mobile device may also fail during useful computation, checkpoint establishment, or recovery operation. If a failure occurs during checkpoint establishment, the rollback point will be the previous checkpoint.

The mobile computing system employed in our paper mainly consists of mobile consumer devices, remote checkpoint servers, and access points. A mobile device can access the Internet by connecting its nearest access point (or base station) through the wireless interface[2]. We employ a simple remote checkpoint mechanism for mobile devices, in which an application of the mobile device periodically saves its checkpoint data into storage of the remote checkpoint server, and load the last checkpoint from the server when a failure has occurred as shown in Fig. 1.

We assume that wireless communication, as well as local memory access, is required during checkpoint establishment C or rollback recovery R, whereas only local memory access is required during normal local computation. Let E_N denote the expended energy per second for normal local computation, which includes only the energy expended by local memory. We also denote the power consumption for checkpoint or recovery operations as E_C, which includes the power consumed by both local memory and wireless interfaces. Since E_C includes E_N and the power consumed by a wireless interface is greater than zero, E_C is always greater than E_N. Actually, our power consumption model is similar to the ones used in [16], [17]. Table I summarizes the notations used in this paper.

TABLE I
NOTATIONS

Notation	Description
λ	Failure rate of a mobile device
T	Checkpoint interval (seconds)
C	Time (seconds) required to establish a checkpoint (through a wireless channel)
R	Time (seconds) required to load the last checkpoint (through a wireless channel)
E_N	Power (watts) consumed for useful computation that requires only local memory access
E_C	Power (watts) consumed for checkpoint or rollback recovery operations that require both wireless communication and local memory access
m	E_C / E_N

IV. Power-Aware Optimal Checkpoint Interval

Determining an appropriate checkpoint interval T is very important to minimize the time and energy expended to complete a mobile device task. If T is too long, the loss of computation will increase when failures occur. On the other hand, if T is too short, i.e., frequent checkpoints, the overhead costs establishing the checkpoints will be greatly enlarged.

In our system, mobile failure process is a Poisson process with parameter λ and each mobile failure epoch is a regeneration point. Therefore, the power-aware optimal checkpoint interval can be analyzed by a stochastic analysis. We use Markov chain and renewal reward theory [18]. Let us denote X_n as the n-th mobile failure, then the stochastic process $\{X_n, n=0,1,2,\ldots\}$ is a Markov chain. Fig. 3 shows the state transition diagram of our Markov chain.

Since the energy expenditure can be different according to each case when a failure occurs during useful computation (T), checkpointing (C) or recovering (R), 3 states for mobile failure(state 2, 3, and 4) are needed. We assume that useful computation does not require wireless transmissions but only local memory accesses.

State 0 is an initial state, and State 1 is a termination state. State 2 is a state in which a failure has occurred during useful computation. States 3 and 4 are states in which a failure has occurred during checkpoint establishment and during rollback recovery, respectively. For example, when a failure has occurred during the useful computation in a checkpoint interval, then a transition from state 0 to state 2 occurs.

After that, if another failure occurs while a checkpoint is being established after completing the useful computation of the interval, then a transition from state 2 to 3 occurs. If any failures occur during the rollback recovery, a transition from

[2] The wireless interface can be WiFi, 3G/4G cellular data communication, or Bluetooth, etc.

state 3 to 4 occurs. After that, if useful computation and checkpoint establishment of the interval have been successfully completed without further failures, then a transition from state 4 to state 1 occurs.

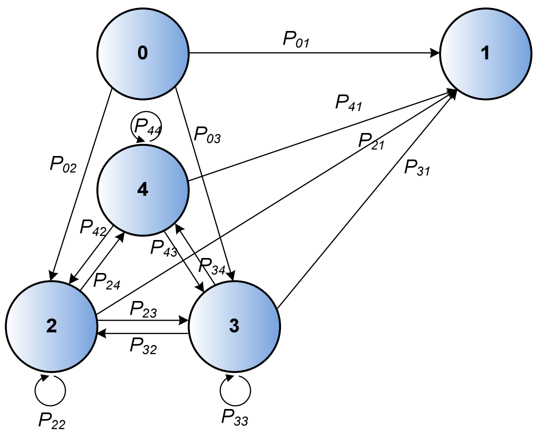

Fig. 3. State transition diagram of Markov chain.

We can calculate transition probabilities (P_{xy}) as (1). One step transition probability matrix $\boldsymbol{P}$ is

$$\boldsymbol{P} = \{P_{ij}\},\ i,j \in \{0,1,2,3,4\}$$

where, $P_{ij} = \Pr$ [next state is j | current state is i].

$$\begin{aligned}
& & P_{21} &= P_{31} = P_{41} = e^{-\lambda(R+T+C)} \\
P_{01} &= e^{-\lambda(T+C)} & P_{22} &= P_{32} = P_{42} = e^{-\lambda R} - e^{-\lambda(R+T)} \\
P_{02} &= 1 - e^{-\lambda T} & P_{23} &= P_{33} = P_{43} = e^{-\lambda(R+T)} - e^{-\lambda(R+T+C)} \\
P_{03} &= e^{-\lambda T} - e^{-\lambda(T+C)} & P_{24} &= P_{34} = P_{44} = 1 - e^{-\lambda R}
\end{aligned} \tag{1}$$

The transition cost is the expected energy expended in the mobile device. Note that the total energy expended during the useful computation of a checkpoint interval, the checkpoint establishment and the rollback recovery is $E_N T$, $E_C C$, and $E_C R$, respectively.

The cost K_{01} is the expended energy during the useful computation of an interval and the checkpoint establishment (T+C). K_{02} is the expended energy when a failure has occurred during the useful computation. The cost K_{03} is the expended energy when a failure has occurred during the checkpoint establishment, given that a failure doesn't occur during the useful computation. K_{22}, K_{32} and K_{42} are the expended energy when a failure has occurred during T after previous failure had occurred during the useful computation (T), checkpoint establishment (C), and the rollback recovery (R), given that any failure had not occurred during the rollback recovery for the previous failure, respectively. K_{23}, K_{33} and K_{43} are the expended energy when a failure has occurred during C after the previous failure had occurred during T, C, and R, respectively. K_{24}, K_{34} and K_{44} are the expended energy when a failure has occurred during the rollback recovery (R) after the previous failure had occurred during T, C, and R, respectively. K_{21}, K_{31} and K_{41} are the expended energy when any failure has not occurred during T nor C after the previous failure had occurred during T, C, and R, respectively.

Now we can obtain conditional expected energy cost of each transition as (2).

$$\begin{aligned}
K_{01} &= E_N T + E_C C \\
K_{02} &= \int_0^T \left(E_N t \cdot \frac{\lambda e^{-\lambda t}}{1-e^{-\lambda T}} \right) dt = E_N \left(\frac{1}{\lambda} - \frac{T e^{-\lambda T}}{1-e^{-\lambda T}} \right) \\
K_{03} &= \int_T^{T+C} \left((E_N T + E_C (t-T)) \frac{\lambda e^{-\lambda t}}{e^{-\lambda T} - e^{-\lambda(T+C)}} \right) dt \\
&= E_N T + E_C \left(\frac{1}{\lambda} - \frac{C e^{-\lambda C}}{1-e^{-\lambda C}} \right) \\
K_{22} = K_{32} = K_{42} &= \int_R^{R+T} \left((E_C R + E_N (t-R)) \frac{\lambda e^{-\lambda t}}{e^{-\lambda R} - e^{-\lambda(R+T)}} \right) dt \\
&= E_C R + E_N \left(\frac{1}{\lambda} - \frac{T e^{-\lambda T}}{1-e^{-\lambda T}} \right) \\
K_{23} &= K_{33} = K_{43} \\
&= \int_{R+T}^{R+T+C} \left((E_C R + E_N T + E_C (t-R-T)) \frac{\lambda e^{-\lambda t}}{e^{-\lambda(R+T)} - e^{-\lambda(R+T+C)}} \right) dt \\
&= E_N T + E_C R + E_C \left(\frac{1}{\lambda} - \frac{C e^{-\lambda C}}{1-e^{-\lambda C}} \right) \\
K_{24} = K_{34} = K_{44} &= \int_0^R \left(E_C t \cdot \frac{\lambda e^{-\lambda t}}{1-e^{-\lambda R}} \right) dt = E_C \left(\frac{1}{\lambda} - \frac{R\ e^{-\lambda R}}{1-e^{-\lambda R}} \right) \\
K_{21} = K_{31} = K_{41} &= E_C R + E_N T + E_C C
\end{aligned} \tag{2}$$

In (1) and (2), we can see that the transition probability and cost are equal from any of states 2, 3 or 4 to state 1, and it is the same as to state 2, 3, and 4, respectively, as shown in (3).

$$\begin{aligned}
&P_{21} = P_{31} = P_{41}, \quad && P_{22} = P_{32} = P_{42}, \\
&K_{21} = K_{31} = K_{41}, \quad && K_{22} = K_{32} = K_{42}, \\
&P_{23} = P_{33} = P_{43}, \quad && P_{24} = P_{34} = P_{44}, \\
&K_{23} = K_{33} = K_{43}, \quad && K_{24} = K_{34} = K_{44}.
\end{aligned} \tag{3}$$

The expected energy expenditure during a checkpoint interval is the transition cost from state 0 to state 1, and can be given as (4)

$$\Gamma_E = \Gamma_{01} = P_{01} K_{01} + P_{02} \left(K_{02} + \Gamma_{21} \right) + P_{03} \left(K_{03} + \Gamma_{31} \right) \tag{4}$$

where,

$$\Gamma_{21} = \frac{P_{21}K_{21} + P_{22}K_{22} + P_{23}K_{23} + P_{24}K_{24} + P_{23}\Gamma_{31} + P_{24}\Gamma_{41}}{1-P_{22}},$$

$$\Gamma_{31} = \frac{P_{31}K_{31} + P_{32}K_{32} + P_{33}K_{33} + P_{34}K_{34} + P_{32}\Gamma_{21} + P_{34}\Gamma_{41}}{1-P_{33}},$$

$$\Gamma_{41} = \frac{P_{41}K_{41} + P_{42}K_{42} + P_{43}K_{43} + P_{44}K_{44} + P_{42}\Gamma_{21} + P_{43}\Gamma_{31}}{1-P_{44}}.$$

From (1) and (2), we can simplify (4) as (5).

$$\Gamma_E = \frac{E_C \left(e^{\lambda(C+R+T)} + e^{\lambda C} - e^{\lambda R} - e^{\lambda(C+T)} \right) + E_N \left(e^{\lambda(C+T)} - e^{\lambda C} \right)}{\lambda} \tag{5}$$

Let us define $G(t)$ as the expected amount of energy (joules) required to complete t time units of useful computation for a job in a mobile device. $G(t)$ includes the energy expended for useful computation, wasted computation by faults, and other overhead. Then the energy overhead ratio r is as (6), by the renewal reward theorem.

$$
\begin{aligned}
r &= \lim_{t\to\infty}\frac{G(t)-E_N t}{t} \\
&= \frac{\Gamma_E - E_N T}{T} = \frac{\Gamma_E}{T} - E_N \\
&= \frac{E_C\left(e^{\lambda(C+R+T)} + e^{\lambda C} - e^{\lambda R} - e^{\lambda(C+T)}\right) + E_N\left(e^{\lambda(C+T)} - e^{\lambda C}\right)}{\lambda T} - E_N.
\end{aligned} \tag{6}
$$

Practically, C, R and T are always greater than zero. Since E_C is the summation of E_N and the expended energy for wireless communication per second, E_C is always greater than E_N. Therefore, we can define $m = E_C / E_N$ where m is always greater than one. Then, (6) is simplified as (7).

$$
r = E_N\left[\frac{m\left(e^{\lambda(C+R+T)} + e^{\lambda C} - e^{\lambda R} - e^{\lambda(C+T)}\right) + \left(e^{\lambda(C+T)} - e^{\lambda C}\right)}{\lambda T} - 1\right] \tag{7}
$$

The checkpoint interval which minimizes the overhead ratio r can be obtained by differentiating (7) respect to T, and

$$
\frac{\partial r}{\partial T} = E_N\left[\frac{e^{\lambda C}(m-1) - e^{\lambda R}m + e^{\lambda(C+T)}(m-1)(\lambda T - 1) - e^{\lambda(C+R+T)}m(\lambda T - 1)}{\lambda T^2}\right] = 0.
$$

After simplification, we can obtain (8).

$$
e^{\lambda T}(\lambda T - 1) = \frac{m - 1 - e^{\lambda(C+R)}m}{e^{\lambda R}m - m + 1}, \quad \text{for } T > 0. \tag{8}
$$

To solve (8), we use a 2nd-order Taylor series expansion respect to λT. After some simplifications, we get the following 3rd-order polynomial equation:

$$
T^2\lambda^2 + T^3\lambda^3 + \frac{2\left(1 - e^{-C\lambda}\right)e^{R\lambda}m}{\left(1 - e^{-R\lambda}\right)m - 1} = 0 \tag{9}
$$

The power-aware optimal checkpoint interval (T_E^*), is a real solution of (9), and is shown as (10).

$$
T_E^* = \frac{1}{3\lambda}\left(\sqrt[3]{B} + \frac{1}{\sqrt[3]{B}} - 1\right) \tag{10}
$$

$$
\text{where, } A = \frac{\left(1 - e^{-C\lambda}\right)e^{R\lambda}m}{\left(1 - e^{-R\lambda}\right)m - 1},\ B = 3\sqrt{6A + 81A^2} - 27A - 1
$$

As we can see (10), the power-aware optimal value is dependent on λ, m, C, and R. Although the result seems to be quite complicated, it is quite easy to apply in practice by just using a portable calculator. Note that we can also obtain the time optimal interval from (10) by setting m to 1 (i.e. E_C=E_N=1 in (5) and (6)).

V. Performance Evaluation

In this section, we evaluate the performance of our method that finds out power-aware optimal checkpoint intervals computed using (10) through analytic estimations, simulations, and experimental measurements. The system parameters and assumptions follow the system model described in Section III.

To obtain more practical energy expenditure parameters (i.e., E_N and E_C) for analytical estimations and simulation, we refer to the power consumption values of an off the shelf WiFi [15] and a mobile DRAM [19]. In our scenario, we assume that a wireless link spends 70% of its time in idle mode, 20% of its time in receiving mode, and 10% of its time transmitting as employed in [20], [21]. As a result, we obtain E_N=0.02 watts and E_C=0.4 watts.

A. Analytical Estimates

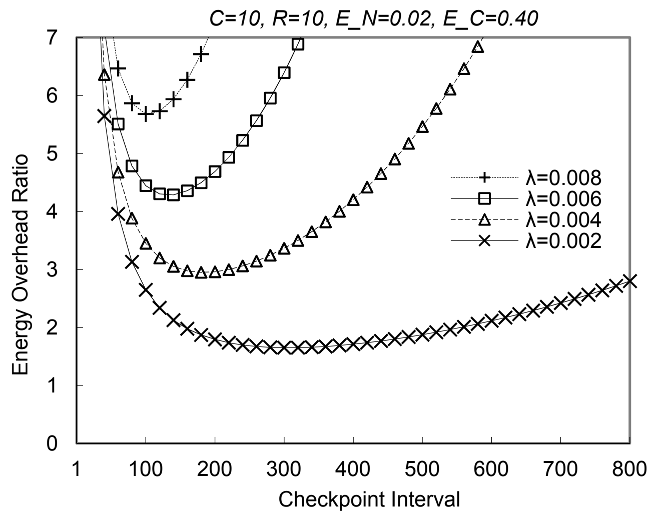

Fig. 4. Energy overhead ratio *r* by varying checkpoint interval *T* with respect to the failure rate λ.

Fig. 4 shows the energy overhead ratio r of the expended energy during one checkpoint interval, which is computed by (7), by varying checkpoint interval T with respect to failure rate λ when C=10, R=10. As we can see, r shows the minimized value where T has the power-aware optimal value (T_E^*), each of which is shown in Table II. We can also see that r is very sensitive to λ. As λ is getting higher, T_E^* should decrease, because the loss of useful computation increases according to λ. For example, as λ converges to 1, T_E^* should also converge to 1.

Table II
Power-Aware Optimal Values of *T* for Various Failure Rates

λ	0.002	0.004	0.006	0.008
T_E^*	103.38	131.12	180.59	299.89

* C=10, R=10, E_N=0.02, and E_C=0.40

Fig. 5 shows overhead ratio r during a checkpoint interval by varying the checkpoint interval with respect to m which is E_C/E_N. In the experiment, we vary m by changing E_C while E_N is fixed to 0.02 watts. As we can see, r is minimized when the checkpoint interval is the energy optimal value (T_E^*), each of which is shown in Table III. We can see that the energy optimal interval is also dependent on m. As m is getting higher, T_E^* should increase accordingly, because dense checkpoints are inefficient when the power consumption for storing a checkpoint is high.

TABLE III
ENERGY OPTIMAL VALUES OF *T* BY VARYING E_C/E_N.

E_C/E_N (m)	1	10	100
T_E^*	62.6	150.4	228.9

* C=10, R=10, E_N=0.02, and λ =0.004

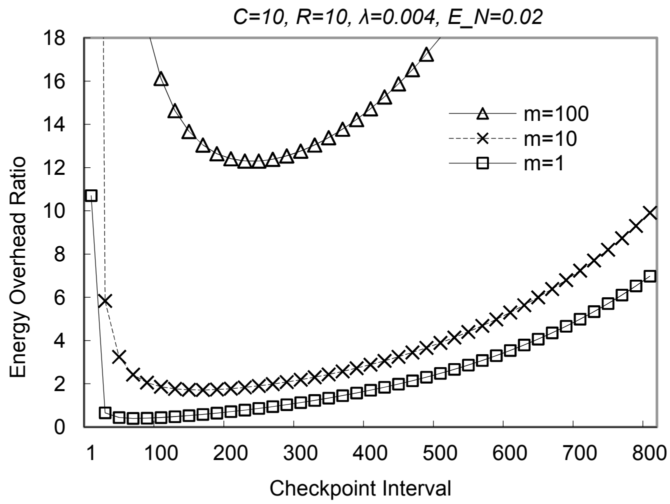

Fig. 5. Energy overhead ratio *r* with respect to *m*.

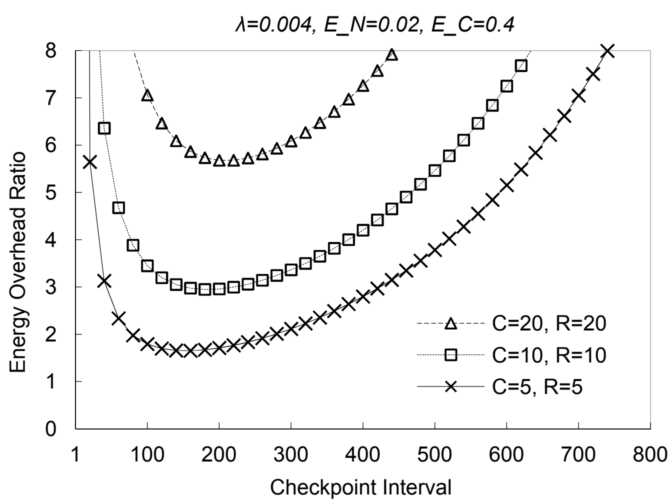

Fig. 6. Energy overhead ratio *r* with respect to *C* and *R*.

Fig. 6 shows energy overhead ratio during a checkpoint interval by varying *T* with respect to *C* and *R*. As we can see, the energy expenditure is also minimized when the checkpoint interval is the energy optimal value (T_E^*), each of which is presented in Table IV. As *C* is getting larger, T_E^* should increase accordingly, because dense checkpoints are inefficient when the checkpoint overhead time is large.

TABLE IV
ENERGY OPTIMAL VALUES OF *T* BY VARYING *C* AND *R*

C and *R*	5	10	20
T_E^*	149.9	180.6	206.8

* E_N=0.02 E_N=0.40, and λ =0.004.

Fig. 7 shows the overhead ratio *r* with respect to checkpoint interval policies. *Young's Time Optimal* denotes the first order approximation of the time optimal checkpoint method proposed by Young [3] as shown in (2). *Daly's Time Optimal* denotes a higher order approximation proposed by Daly [11] as shown in (3), which shows higher accuracy than Young's. *Power-aware Optimal* denotes our power-aware optimal checkpoint interval method presented in (10). Obviously, we can see that employing the energy optimal interval expends less energy than employing the time optimal interval. Daly's method has shown the less-expected execution time compared to Young's method, which is because Daly's has a more accurate approximation [11]. However, Daly's method does not always show less energy expenditure compared to Young's method. This means that the more accurate approximation for the time optimal checkpoint interval does not guarantee the lower energy expenditure of a mobile device.

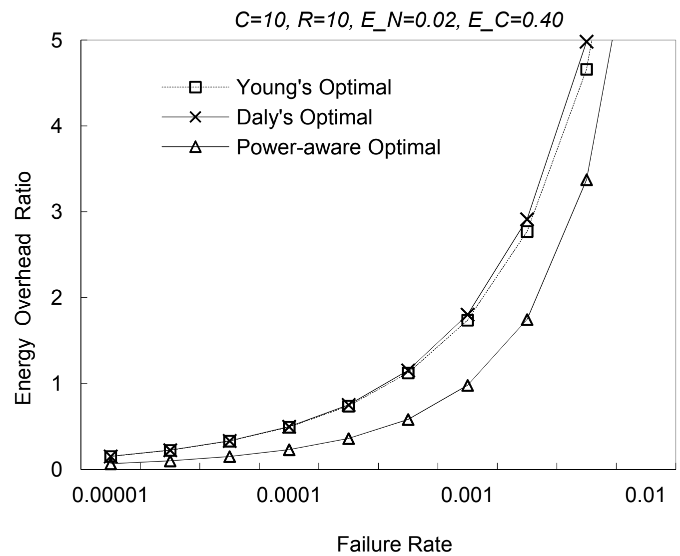

Fig. 7. Comparison of energy overhead ratio *r* between power-aware optimal and time optimal intervals.

B. Simulations

For the more practical performance evaluation, we conducted simulations with practical energy expenditure parameters, which are from Table 3. The total energy (joules) expended to complete a job is measured while the total time for the useful computation of the job is set to 10,000 seconds. We used MATLAB v.7.4 as a simulation tool, and simulations for each scenario are repeated 10,000 times and averaged.

Fig. 8 shows the total energy expended to complete a job by varying checkpoint interval *T* with respect to failure rate λ where E_N=0.02 watts and E_C=0.40 watts. As we can see, the energy expenditure is minimized when the checkpoint interval is the energy optimal value (T_E^*), each of which is shown in Table II.

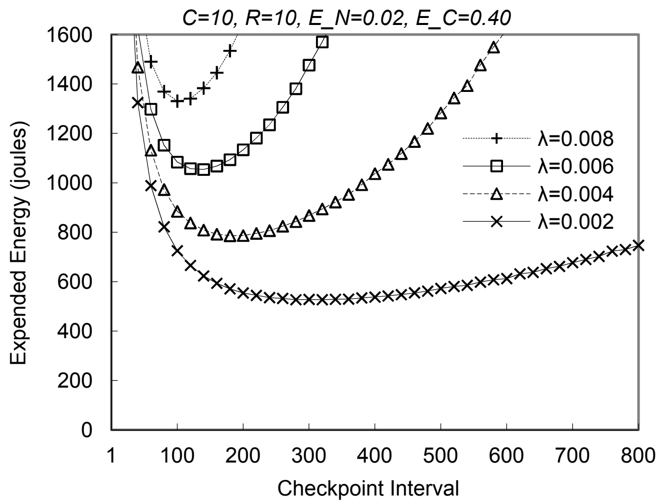

Fig. 8. Energy expenditure with respect to λ

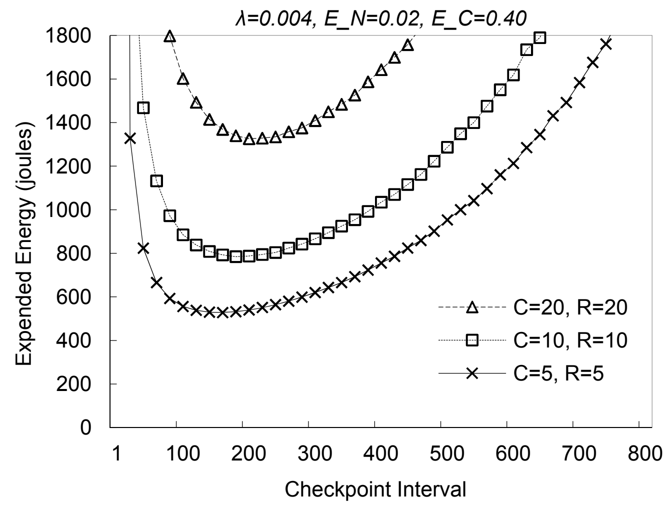

Fig. 10. Energy expenditure with respect to *C* and *R*.

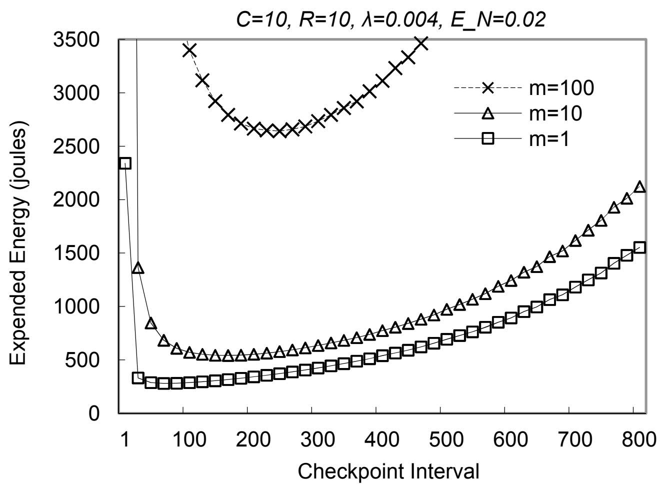

Fig. 9. Energy expenditure with respect to *m*.

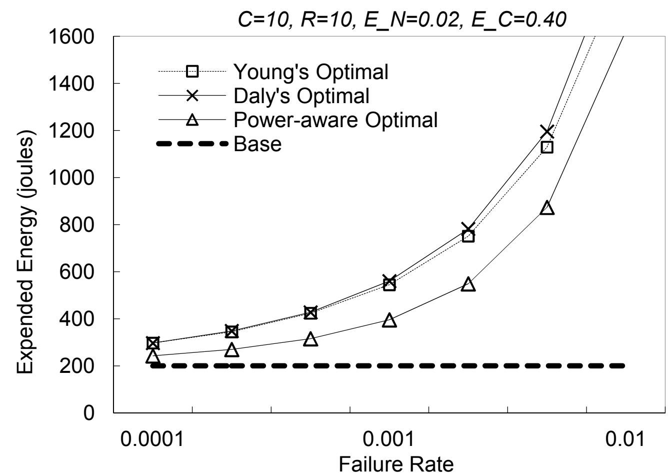

Fig.11. Comparison of energy expenditure between power-aware optimal and time optimal interval policies.

Fig. 9 shows the total energy expended to complete a job by varying checkpoint interval with respect to m which is E_C/E_N. In the experiment, we varied m by changing E_C while E_N was fixed to 0.02. As we can see, the energy expenditure was minimized when the checkpoint interval was the energy optimal value (T_E^*), each of which was shown in Table III.

Fig. 10 shows the total energy expended to complete a job by varying checkpoint intervals with respect to C and R. As we can see, the energy expenditure is also minimized when the checkpoint interval is the energy optimal value (T_E^*), each of which are presented in Table IV.

Fig.11 shows the total expended energy to complete a job by varying λ with respect to checkpoint interval. Please note that at least 200 joules[3] of energy are required to complete the job even though there are no failures and no checkpoints. Obviously, we can see that employing our power-aware optimal intervals can expend less energy than employing other time optimal interval policies.

[3] 0.02 watts × 10,000 seconds = 200 joules

C. Experimental Measurements from the Implementation

For experimental measurements, we implemented a test program on real test-bed, and assessed its performance. The test-bed consists of a mobile consumer device (a smart phone with a 1 GHz CPU, a 512MB RAM, an 802.11 b/g/n WiFi interface, and a 3.7V and 1500 mAh battery), a WiFi access point, and a remote server. The mobile device is saving and loading its checkpoint data into/from the remote server through the WiFi access point. Fig. 12 shows the test bed deployed for our experiments.

The test application was programmed in JAVA. While saving the checkpoint data (i.e., during C), the mobile device transmits data into the server via the WiFi interface intensively. The same conditions are employed while loading the checkpoint data (i.e., during R). Wireless communication does not occur during useful computations (i.e., during T). To obtain clear results, during experiments, the test-bed ran no other application processes except system related processes, and wireless interfaces other than WiFi were disabled.

Fig. 12 Test bed for our experiments

The total energy expended to complete a job is measured while the total time for the useful computation of the job is set to 1,200 seconds. Device failures were simulated by a Poisson process with rate λ. The experiments were repeated 10 times for each scenario and the results were averaged. To compute the power-aware optimal value (T_E^*), we observed that E_C=0.00812% and E_N=0.00279% of the battery capacity from the mobile device. Fig. 13 shows a screenshot of the test application running on the mobile device for the experiments.

Fig. 14 shows the total energy expended to complete a job by varying checkpoint interval T with respect to λ where C=5 and R=5. As we can see, the energy expenditure is minimized when the checkpoint interval is the power-aware optimal value (T_E^*), each of which is shown in Table V.

Fig. 13. Screenshot of the test program running on the mobile consumer device

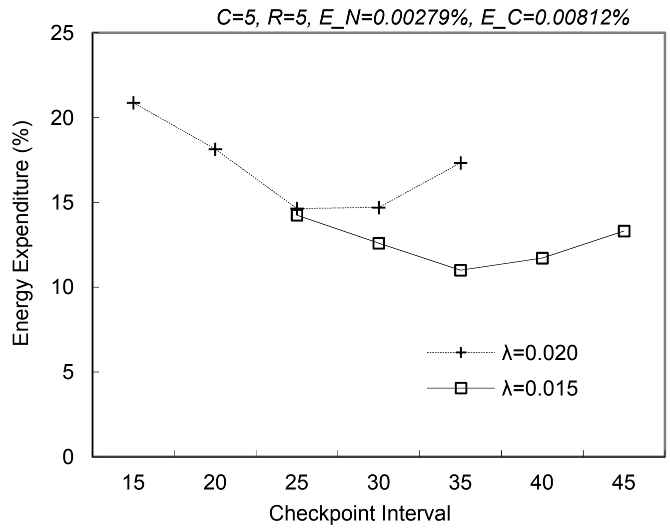

Fig. 14. Energy expenditure by varying the checkpoint interval with respect to λ.

TABLE V
POWER-AWARE OPTIMAL VALUES WITH RESPECT TO FAILURE RATES

λ	0.015	0.02
T_E^*	33.11	27.48

* C=5, R=5, E_N=0.00279%, and E_C=0.00812%

IX. CONCLUSION

In this paper, we proposed a solution for finding out the optimal checkpoint interval which minimizes the energy expenditure of a mobile device in remote checkpointing wireless environments. We evaluated our proposed solution by analysis, simulation, and experimental measurements by implementing on a Smartphone. From the results of the evaluation, we can see that our proposed solution is able to provide the almost optimal checkpoint interval minimizing the energy expenditure of mobile devices. Our results also show that the power-aware optimal point is deeply dependant on the ratio of E_C to E_N as well as the failure rate and the checkpoint overhead.

REFERENCES

[1] H.-N. Lee, Sung-Hwa Lim, J.-H. Kim, "UMONS: Ubiquitous Monitoring System in Smart Space," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 3, pp.1056-1064, Aug. 2009.
[2] Z. Zhan, Z. De-chaeng, C. Yi-wei, and Y. Xian-zong, " The Checkpoint Interval Optimization of Kernel-level Rollback Recovery based on the Embedded Mobile Computing System," *IEEE International Conference on Computer and Information Technology Workshops*, Australia, July 2008.
[3] J. W. Young, "A first order approximation to the optimum checkpoint interval," *Communications on the ACM*, vol. 17, pp. 530-531, Sept. 1974.
[4] N. H. Vaidya, "On checkpoint latency," *Pacific Rim International Symposium on Fault-Tolerant Systems*, pp. 60-65, Dec. 1995.
[5] K. M. Chandy, J. C. Browne, C. W. Dissly, and W. R. Uhrig, "Analytic Models for Rollback and Recovery Strategies in Data Base Systems," *IEEE Transactions on Software Engineering*, vol. 1, March 1975.
[6] K.-S. Byun, Sung-Hwa Lim and J.-H. Kim, "Two-Tier Checkpointing Algorithm Using MSS in Wireless Network," *IEICE transactions on Communications*, vol. E86-B, no. 7, pp. 1-7, July 2003.

[7] G.-H. Li, H. Wang, “A Novel Min-process Checkpointing scheme for Mobile Computing Systems,” *Journal of Systems Architecture*, vol. 51, pp. 45-61, 2005.
[8] I.-R. Chen, B. Gu, S. E. George, and S.-T. Cheng, “On Failure Recoverability of Client-Server Applications in Mobile Wireless Environments,” *IEEE Transactions on Reliability*, vol. 54, no. 1, pp. 115-122, 2005.
[9] S. Biswas and S. Neogy, “A Low Overhead Checkpointing Scheme for Mobile Computing Systems,” *IEEE International Conference on Advanced Computing and Communications*, India, Dec. 2007.
[10] P. Kumar, P. Gupta, and A. K. Solanki, “Dealing with Frequent Aborts in Minimum-process Coordinated Checkpointing Algorithm for Mobile Distributed Systems,” *International Journal of Computer Applications*, vol.3, no. 10, July 2010.
[11] J. T. Daly, “A Higher Order Estimate of the Optimum Checkpoint Interval for Restart Dumps,” Future Generation Computer Systems, Vol. 22, pp. 303-312, 2006.
[12] C. Men, and Z. Xu, “Performance Analysis of Rollback Recovery Schemes for the Mobile Computing Environment,” *International Conference on Internet Computing in Science and Engineering*, Aug. 2008.
[13] S. E. George, I.-R. Chen, Y. Jin, “Movement-based checkpointing and logging for recovery in mobile computing systems,” *ACM International Workshop on Data Engineering for Wireless and Mobile Access*, U.S.A., June 2006.
[14] A. Avizienis, J.-C. Laprie, B. Randell and C. Landwehr, “Basic Concepts and Taxonomy of Dependable and Secure Computing”, *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11-33, 2004.
[15] “CW1200 : 802.11n dual-band WLAN system-on-chip,” Data Sheet, ST Ericsson.
[16] W. Rabiner H., A. Chandrakasan, H. Balakrishnan. “Energy-Efficient Communication Protocol for Wireless Microsensor Networks.” *Hawaii International Conference on System Sciences*, Hawaii, Jan. 2000.
[17] K.-H. Han, J.-H. Kim, Y.-B. Ko and W.-S. Yoon. “An Energy Efficient Broadcasting for Mobile Devices Using a Cache Scheme.” *Lecture Notes in Computer Science (Proc. ICCS 2004)*, vol. 3036, 2004.
[18] S. M. Ross, “Stochastic Processes,” 2nd ed., John Wiley & Sons, 1996.
[19] “Mobile DRAM Power-Saving Features and Power Calculations,” Technical node, TN-46-12, Micron, May 2009.
[20] “Power Consumption and Energy Efficiency Comparisons of WLAN Products,” White Paper, Atheros Communications, Inc, 2003.
[21] B. McFarland and M. Wong, “The Family Dynamics of 802.11,” *ACM Queue*, vol. 1, no. 3, May 2003.

BIOGRAPHIES

Sung-Hwa Lim received his B.S., M.S., and Ph.D. degree in computer engineering from Ajou University, Korea, in 1999, 2001, and 2008, respectively. He was a post doctoral researcher at University of Illinois at Urbana-Champaign from 2008 through 2009. He has been a research professor at the department of Systems Management Engineering at Sungkyunkwan University, Korea, since 2009. His research interests include mobile and ubiquitous computing, wireless networks, and power-aware computing.

Se Won Lee received his B.S., a M.S., and Ph.D. degree in Industrial engineering from Sungkyunkwan University, Korea, in 1998, 2003, and 2008, respectively. He was a research professor at the department of Systems Management Engineering at Sungkyunkwan University from 2009 through 2011. He is currently a visiting professor at the department of Business Administration, Dongguk University-Seoul, South Korea. His research interests include queuing theory, operations research, and applied stochastic processes.

Byoung-Hoon Lee received his B.S. and M.S. degrees in Computer Engineering, Chungbuk National University, Cheongju, South Korea, in 1998 and 2000, and his Ph.D degree in Information and Communication, Ajou University, Suwon, South Korea, in 2009. He is currently a research professor of the Institute for Information and Electronics Research at Inha University. His research interests include ubiquitous computing, distributed systems and embedded programming.

Seongil Lee (M’06) received a B.S. and a M.S. from Seoul National University, Korea, in Industrial Engineering in 1985 and 1987, respectively. He received the Ph.D. from University of Wisconsin-Madison in 1995 with works on haptic interface design for people with disabilities. He taught at Chonnam National University from 1987 to 1988, and has been a faculty at Sungkyunkwan University, Suwon, Korea, since 1998. He is currently a professor at the department of Systems Management Engineering. His research interests include human factors and ergonomics, universal design and accessibility, Human-Computer Interaction, accessible computing, and user interface design in mobile and ubiquitous computing systems.