Analytic Model for Optimal Checkpoints in Mobile Real-time Systems

Sung-Hwa Lim[1], Byoung-Hoon Lee[2] and Jai-Hoon Kim[3]
[1] Department of Multimedia, Namseoul University
Choenan, Republic of Korea
[e-mail: sunghwa@nsu.ac.kr]
[2] Solution Development Team, NSE
Daejeon, Republic of Korea
[e-mail: componer@hotmail.com]
[3] Department of Cyber Security, Ajou University
Suwon, Republic of Korea
[e-mail: Jaikim@ajou.ac.kr]
*Corresponding author: Jai-Hoon Kim

Received May 5, 2016; revised June 7, 2016; accepted July 1, 2016; published August 31, 2016

Abstract

It is not practically feasible to apply hardware-based fault-tolerant schemes, such as hardware replication, in mobile devices. Therefore, software-based fault-tolerance techniques, such as checkpoint and rollback schemes, are required. In checkpoint and rollback schemes, the optimal checkpoint interval should be applied to obtain the best performance. Most previous studies focused on minimizing the expected execution time or response time for completing a given task. Currently, most mobile applications run in real-time environments. Therefore, it is extremely essential for mobile devices to employ optimal checkpoint intervals as determined by the real-time constraints of tasks. In this study, we tackle the problem of determining the optimal inter-checkpoint interval of checkpoint and rollback schemes to maximize the deadline meet ratio in real-time systems and to build a probabilistic cost model. From this cost model, we can numerically find the optimal checkpoint interval using mathematical tools. The performance of the proposed solution is evaluated using analytical estimates.

Keywords: Real-time systems, Fault tolerance, Probabilistic theory, Mobile Computing

This research is partly supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2015R1D1A1A01060034 and NRF- 2014R1A1A2057796)

http://dx.doi.org/10.3837/tiis.2016.08.015 **ISSN : 1976-7277**

1. Introduction

Mobile smart devices play key roles in recently developed computing environments, because they constitute the interface between the Internet and human users. Because mobile smart devices are very vulnerable to faults, which occur for various reasons in mobile environments, effective fault-tolerant strategies should be applied in mobile consumer devices. However, the application of hardware-based fault-tolerant schemes, such as hardware replication, is not practically feasible [1]. Therefore, software-based fault-tolerance techniques should be applied in mobile smart devices. The checkpoint and rollback recovery technique is a widely used software-based fault tolerance strategy that does not require additional hardware resources [5]. The loss of a process' computation by failures can be reduced by periodically storing the process' current state in a stable storage as a checkpoint, and rolling back to the most recent checkpoint when a failure occurs during the execution of the process [2, 3]. **Fig. 1** illustrates an example of the checkpoint and rollback strategy for mobile devices presented in [8, 9].

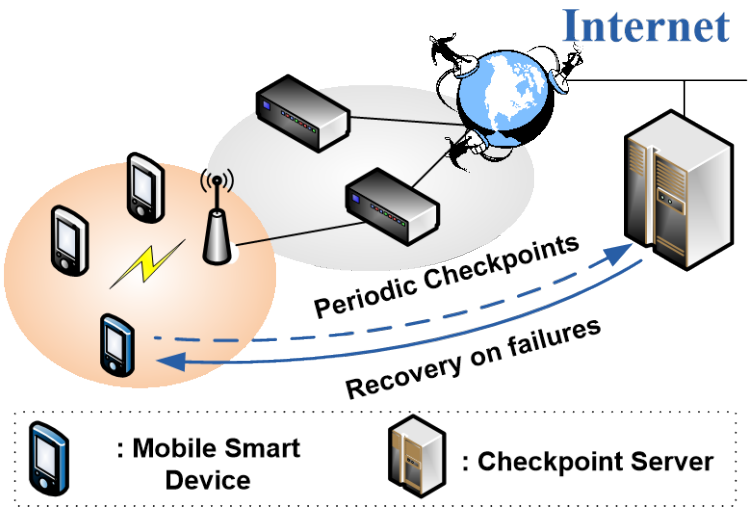

Fig. 1. An example of a mobile checkpoint and rollback strategy

Finding the optimal checkpoint interval (i.e., the inter-checkpoint interval) is crucial to system performance, because dense checkpointing increases the overhead cost for checkpointing, whereas coarse checkpointing increases the loss of useful computation upon failure [2, 5]. Many studies have been conducted to determine the optimal checkpoint interval according to various parameters [4, 5, 8, 9]. However, most previous studies did not consider real-time environments in which every task has its deadline, which is especially important as current mobile applications usually run the most important criterion is to complete a task before its deadline. Therefore, it is strongly required that mobile smart devices employ the optimal checkpoint interval in real-time environments. In real-time systems, policy that is

determined by considering real-time constraints of given tasks. In order to obtain the optimal checkpoint interval, we should build a cost model for the real-time system.

The contributions of this paper are as follows:

- **Cost model in terms of the deadline meet ratio**: We present a cost model for the checkpoint and rollback system in terms of execution time and deadline meet ratio for real-time tasks. From the proposed cost model, we can numerically determine the optimal checkpoint interval using mathematical tools.
- **The optimal checkpoint intervals examples**: We provide optimal checkpoint intervals from the cost model presented in Section 4 by using a mathematical tool to explore some example scenarios.
- **Performance evaluation**: For the performance evaluations, we conduct analytical estimates and simulations. In the performance evaluations, we show that the deadline meet ratio is maximized when the checkpoint interval is near the optimal value.

This paper reports a theoretical study of our previous work [11], in which the optimal checkpoint intervals for real-time tasks were empirically studied. A preliminary version of this paper appeared in EEECS 2016 [10]. This version includes a concrete analysis about the deadline meet ratio, and the performance evaluation with simulations.

This paper is organized as follows. In Section 2, we discuss some related work and in Section 3 our system model is presented. Then, in Section 4 we propose a cost model for the optimal checkpoint interval that maximizes the deadline meet ratio. The performance of the proposed solution is evaluated using analytical estimates in Section 5. Finally, we conclude our paper in Section 6.

2. Related Work

The checkpoint and rollback recovery scheme is one of the most widely used software-based fault-tolerant techniques, in particular, as a defense against intermittent and transient faults. The checkpoint is a snapshot of the state of a currently running process and is periodically saved in stable storage. By loading the checkpoint into its memory, when a failure occurs the system can restart its process from the point of the last saved checkpoint instead of the beginning. Thus, the loss of useful computation can be effectively reduced [2, 5].

The determination of the optimal checkpoint interval has been a primary objective of studies on checkpoint schemes, because a tradeoff exists for the length of the interval: if the interval is too long, upon failure the loss of useful computation will increase; otherwise, if the interval is too short, the overhead incurred by establishing checkpoints will increase. Young proposed a very simple first-order approximate solution to obtain the optimal checkpoint interval based on failure rate and checkpoint overhead [5], while Daly proposed a perturbation solution that provides a higher order approximation [4] than that of Young [5]. In these studies, the authors attempted to minimize the expected execution time for completing a given task.

In some research studies, attempts were made to minimize energy consumption, given that energy is a crucial resource for mobile devices. Lim *et al.* proposed a second-order approximation for the optimal checkpoint interval, minimizing the energy consumption of mobile devices according to device failure rate, checkpoint overhead, and communication failure rates [8, 9].

Currently, most software applications have real-time features, and therefore, tasks for these applications should be processed according to time constraints (i.e., prescribed deadlines). In

real-time systems, meeting the deadline of each task is the most important goal. Punnekkat *et al.* [12, 13] presented a schedulability analaysis of the checkpoint system with real-time tasks to reduce response time. Shin *et al.* developed analytic models of real-time checkpoint systems to obtain optimal checkpoint intervals for minimizing mean task-execution time.

However, previous studies did not provide cost models and solutions for the optimal inter-checkpoint interval in terms of the deadline meet ratio, while the deadline meet ratio of each task is the most important metric for evaluating the performance of real-time systems.

3. System Model

We employed a system model similar to the one used in [10]. When a system fails, its local state is corrupted and all useful computation executed before the failure is lost. To mitigate this loss, the system periodically saves the current state of its running processes as a checkpoint, and recovers from a failure by rolling back to the most recent checkpoint. A checkpoint is a copy of a current process' state, which is stored in stable storage. An example of the checkpoint and rollback process is shown in **Fig. 2**. T is the checkpoint interval, and R is recovery cost. We assume that all faults are detectable and transient and occur according to a Poisson process at rate λ. For simplicity, we assume that a fault leads to a system's failure, although in practice this does not always occur [7]. Establishing a checkpoint incurs an overhead of C sec.

In this study, a uni-process (i.e., task) application model is assumed. The required time to complete a useful computational task is E. The task must be completed within its relative deadline, where the relative deadline of a task is the summation of E and the length of its laxity [6]. In this study, we assumed that the length (i.e., in second) of a task's deadline is determined by following an exponential distribution and its mean value is $\frac{1}{l}$.

Table 1 describes the system parameters.

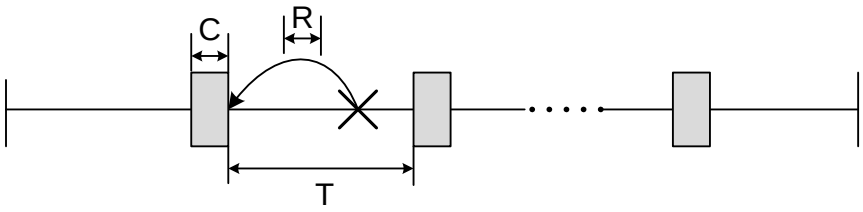

Fig. 2. Example of a checkpoint and rollback scheme

Table 1. System Parameters

Notation	Description
λ	Failure rate
E	Total time to complete a useful computational task (time unit)
C	Time required to establish a checkpoint (time unit)
n	Total number of checkpoints
$\frac{1}{l}$	Mean value of the length of a task's deadline

4. Optimal Checkpoint Interval for Real-time Tasks

In this section, we develop a cost model (i.e., the expected deadline meet ratio) for the checkpoint and rollback scheme for real-time tasks in terms of execution time. Using the proposed cost model, it is possible to compute the optimal checkpoint interval that maximizes the deadline meet ratio.

During the task execution, there are two possible cases:

1) Without failure

The probability that the task is completed successfully without failure is

$$P_{no\_Fail} = e^{-\lambda(E+nC)} \qquad (1)$$

We assume that a task can be completed successfully if no failure occurs during its execution. Therefore, the probability of a task being completed before its relative deadline (i.e., meeting the deadline) without failures is expressed as

$$R_{meet\_without\_Fail} = 1 \qquad (2)$$

2) With failure

A failure may occur during the execution of the task. Then, the probability that a failure occurs during the execution of the task is

$$P_{Fail} = 1 - e^{-\lambda(E+nC)} \qquad (3)$$

To compute the deadline meet ratio in the case where a failure occurs, we need to track the following process.

The probability density function that a failure occurs at time t, where $0 \le t \le \frac{E}{n} + C$, is expressed as

$$\frac{\lambda e^{-\lambda t}}{1 - e^{-\lambda\left(\frac{E}{n}+C\right)}} \qquad (4)$$

The deadline meet ratio in the case where a failure occurs at time t_0 is expressed as

$$\int_{t_0}^{\infty} l e^{-lt} dt = e^{-l(t_0+nC)} \qquad (5)$$

Employing Equation (4) and (5), we can compute the expected deadline meet ratio $R_{meet\_with\_Fail}$ when a failure occurs as follows:

$$
\begin{aligned}
R_{meet\_with\_Fail} &= \int_0^{\frac{E}{n}+C} \left(\frac{\lambda e^{-\lambda t}}{1-e^{-\lambda\left(\frac{E}{n}+C\right)}} e^{-l(t+nC)} \right) dt \\
&= \int_0^{\frac{E}{n}+C} \left(\frac{\lambda e^{-nlC} e^{-(\lambda+l)t}}{1-e^{-\lambda\left(\frac{E}{n}+C\right)}} \right) dt \\
&= \frac{\lambda e^{-nlC}}{1-e^{-\lambda\left(\frac{E}{n}+C\right)}} \left(-\frac{1}{\lambda+l} \right) \left[e^{-(\lambda+l)t} \right]_0^{\frac{E}{n}+C} \\
&= \frac{\lambda e^{-nlC}}{1-e^{-\lambda\left(\frac{E}{n}+C\right)}} \frac{1}{\lambda+l} \left(1-e^{-(\lambda+l)\left(\frac{E}{n}+C\right)} \right) \\
&= \frac{\lambda e^{-nlC}}{\lambda+l} \frac{1-e^{-(\lambda+l)\left(\frac{E}{n}+C\right)}}{1-e^{-\lambda\left(\frac{E}{n}+C\right)}}
\end{aligned}
\tag{6}
$$

Therefore, using Equation (1), (2), (3), and (6), we can compute the deadline meet ratio R_D as Equation (7).

$$
\begin{aligned}
R_D &= \left(P_{no\_Fail} \times R_{meet\_without\_Fail} \right) + \left(P_{Fail} \times R_{meet\_with\_Fail} \right) \\
&= e^{-\lambda(E+nC)} + \left(1-e^{-\lambda(E+nC)} \right) \frac{\lambda e^{-nlC}}{\lambda+l} \frac{1-e^{-(\lambda+l)\left(\frac{E}{n}+C\right)}}{1-e^{-\lambda\left(\frac{E}{n}+C\right)}}
\end{aligned}
\tag{7}
$$

Obtaining the exact optimal value *n* from Equation (7) is not practically feasible. However, we can numerically obtain the optimal checkpoint value, *n*, which can maximize the deadline meet ratio, by employing mathematical computation tools.

5. Performance Evaluation

5.1 Analytical Estimates

In this section, we describe the performance analysis using Equation (7) by estimating the deadline meet ratio. The system parameters and assumptions follow the system model described in Section 3. We assume that the total useful computation time E for a task is 1500 time units.

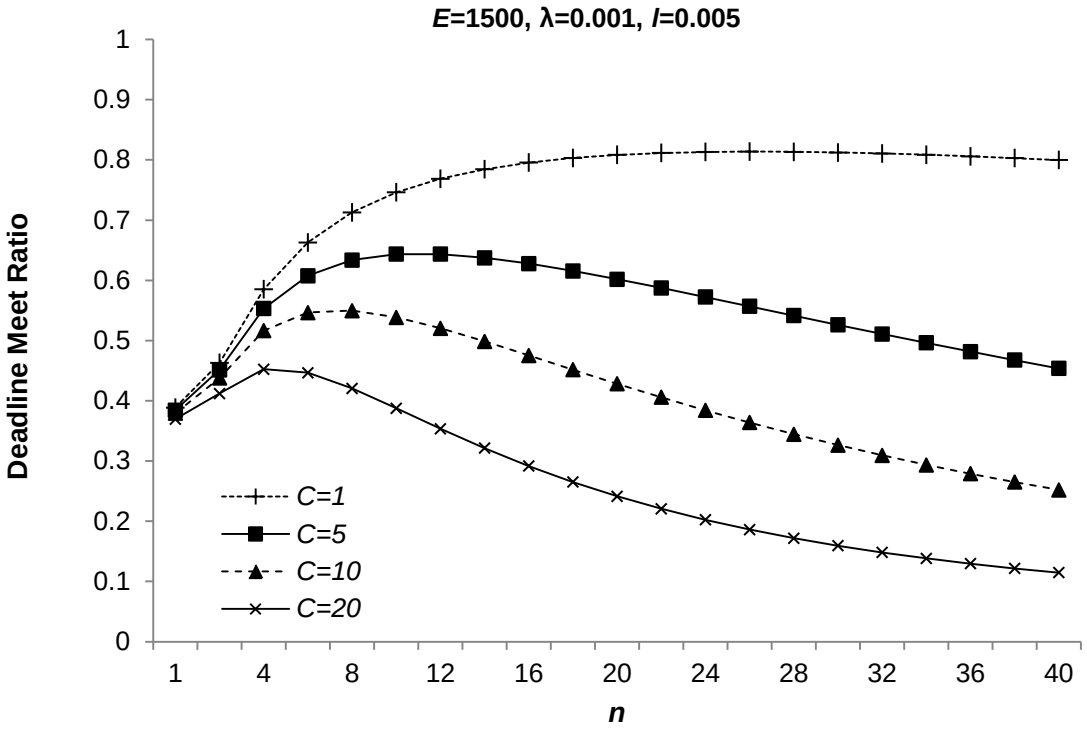

Fig. 3. Deadline meet ratio R_D versus the number of checkpoints n with respect to the checkpoint overhead C

Table 2. Sub-optimal values of T according to varying C

C	1	5	10	20
OPTIMAL n	22.7	11	7.3	4.2

E = 1500, λ = 0.001, and l = 0.005

Fig. 3 shows the deadline meet ratio R_D for completing the useful computational task in time E according to a varied number of checkpoint intervals n with respect to the checkpoint

overhead C. The failure rate λ is set to 0.001, and $\frac{1}{l}$, which is the mean value of the length of a task's deadline, is set to 200 (i.e., l = 0.005). Table 2 shows the sub-optimal number of checkpoints to maximize the deadline meet ratio with respect to the value of C, which is numerically obtained using a mathematical computation tool[1]. As we can see, R_D is maximized when n is close to the optimal value, each of which is presented in Table 2. We can also see that R_D is very sensitive to C.

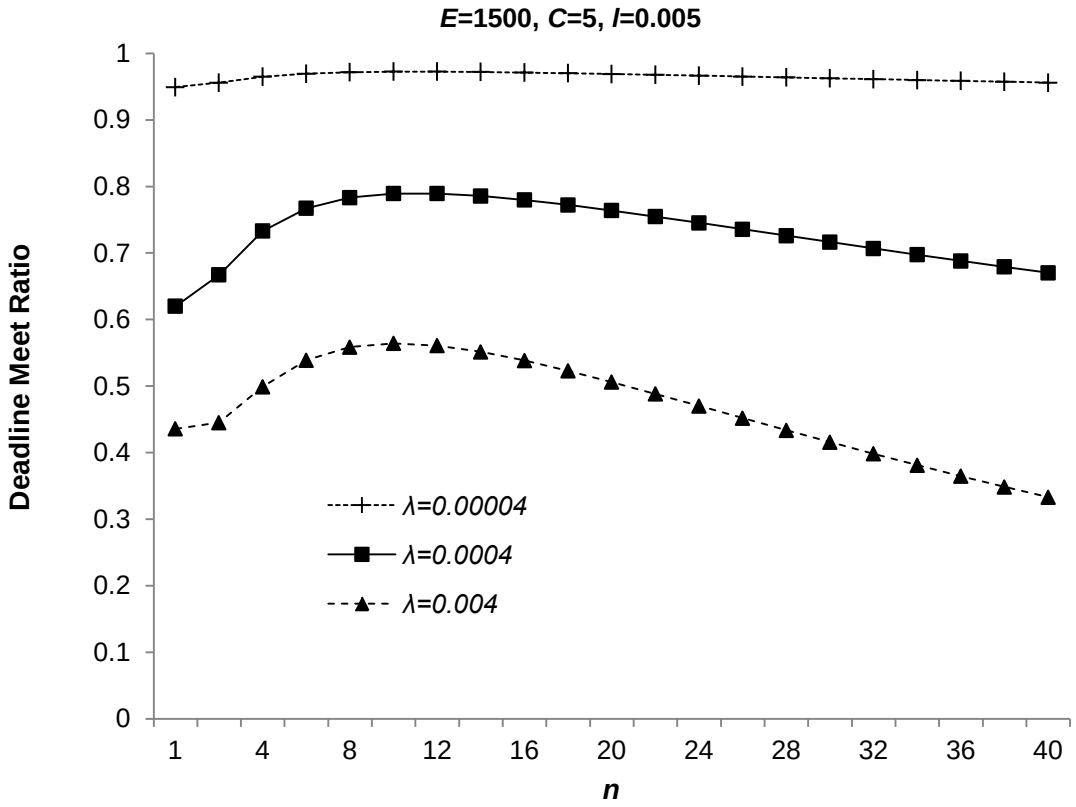

Fig. 4. Deadline meet ratio R_D versus the number of checkpoints n with respect to failure rate λ

Table 3. Energy-optimizing values of n for various failure rates

Λ	0.00004	0.0004	0.004
T_E^*	N/A	10.9	10.1

E = 1500, C = 5, and *l* = 0.005

[1] Wolfram Mathematica 7 [15].

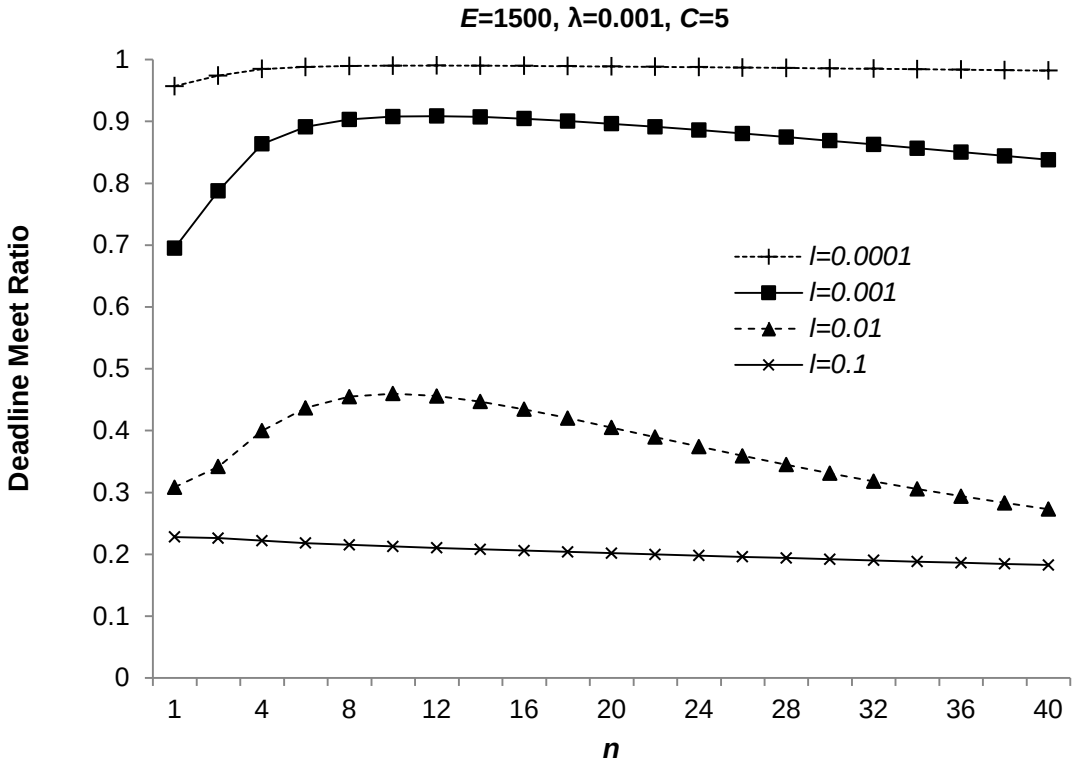

Fig. 5. Deadline meet ratio R_D versus the number of checkpoints n with respect to deadline arrival rate l

Table 4. Energy optimizing values of n according to varying *deadline arrival rate*

L	0.0001	0.001	0.01	0.1
optimal n	n/a	11.62	9.98	n/a

E = *1500,* λ = 0.001 and C = 5

Fig. 4 shows the deadline meet ratio R_D when n is varied with respect to λ. C is set to 5; l is set to 0.005. **Table 3** shows the sub-optimal value of n for maximizing the deadline meet ratio with respect to λ values, which is also numerically obtained by the mathematical computation tool. As we can see, R_D is maximized when n is near the optimal value presented in **Table 3**. The graph does not include the optimal point where λ is too small (e.g., λ = 0.0004).

To show the effect of l, we show R_D according to various values of n with respect to l in **Fig. 5**. λ is set to 0.001, and C is set to 5. **Table 4** shows the sub-optimal value of n with respect to l values, which is also numerically obtained by the mathematical computation tool. R_D is maximized when n is near the optimal values, each of which is presented in **Table 4**. The plot does not include the optimal point if l is too small (e.g., l = 0.0001) or too big (e.g., l = 0.1)

5.2 Simulation

For a more practical performance evaluation, we conducted simulations. We utilized MATLAB v. 7.4 as a simulation tool. Simulations for each scenario were repeated 5,000 times and averaged. **Fig. 6** shows the deadline meet ratio R_D for completing the useful computational task in time E according to a varied number of checkpoint intervals n, with respect to the checkpoint overhead C. The failure rate λ is set to 0.001, which means that the mean value of the length of a task's deadline, is set to 200 (i.e., $l = 0.005$). If n is smaller than the optimal value (i.e., coarse checkpoints), the loss of computation will increase when failures occur. Therefore, the deadline meet ratio decreases. On the other hand, if n is larger than the optimal value (i.e., frequent checkpoints), the overhead costs of establishing the checkpoints are greatly increased. Therefore, the deadline meet ratio also decreases. As we can see, R_D is maximized when n is near the optimal value, each of which is presented in **Table 3**.

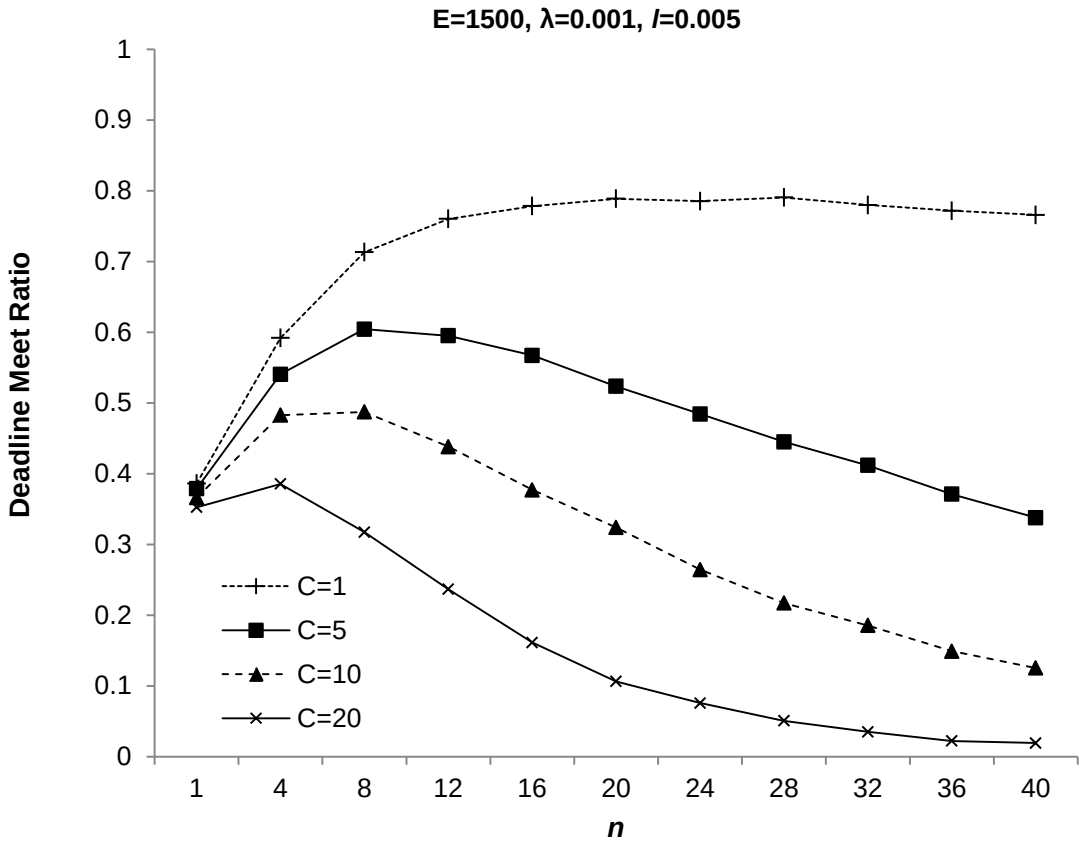

Fig. 6. Deadline meet ratio R_D versus the number of checkpoints n with respect to the checkpoint overhead C by simulation

5. Conclusion

It is extremely essential for mobile devices to employ the optimal checkpoint interval, as determined by considering the real-time constraints of tasks. In order to obtain the optimal checkpoint interval, cost models for real-time systems is highly desired. In this paper, we present a cost model of the checkpoint and rollback system in terms of the deadline meet ratio for real-time tasks. Using the proposed cost model, we can numerically obtain the sub-optimal number of checkpoints (i.e., n) that can maximize the deadline meet ratio by employing mathematical computation tools. The performance of the proposed solution was evaluated using analytical estimates and simulations. For future work, we will extend our cost model to a multi-process real-time task model.

References

[1] 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," in *Proc. of IEEE International Conference on Computer and Information Technology Workshops*, Australia, July 2008. Article (CrossRef Link)

[2] Nitin H. Vaidya, "On checkpoint latency," in *Proc. of Pacific Rim International Symposium on Fault-Tolerant Systems*, pp. 60-65, Dec. 1995.

[3] 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. Article (CrossRef Link)

[4] John T. Daly, "A Higher Order Estimate of the Optimum Checkpoint Interval for Restart Dumps," *Future Generation Computer Systems*, Vol. 22, pp. 303-312, 2006. Article (CrossRef Link)

[5] John W. Young, "A first order approximation to the optimum checkpoint interval," *Communications on the ACM*, vol.17, pp.530-531, Sept.1974. Article (CrossRef Link)

[6] Jane Liu, "Real-time Systems," Prentice Hall, 2000.

[7] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell and Carl 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 Article (CrossRef Link)

[8] Sung-Hwa Lim, Se Won Lee, Byoung-Hoon Lee and Seongil Lee, "Power-Aware Optimal Checkpoint Intervals for Mobile Consumer Devices," *IEEE Transactions on Consumer Electronics*, Vol. 57, No. 4, pp. 1637-1645, Nov. 2011. Article (CrossRef Link)

[9] Sung-Hwa. Lim, Se Won. Lee, Byoung-Hoon. Lee, Seonil Lee, Ho Woo Lee, "Stochastic method for Power-Aware Checkpoint Intervals in Wireless Environments: Theory and Application," *Journal of Industrial and Management Optimization*, vol. 8, no. 4, pp. 969-986, November 2012. Article (CrossRef Link)

[10] Sung-Hwa Lim, Byoung-Hoon Lee, and Jai-Hoon Kim, "Cost Model for the Checkpoint and Rollback Scheme in Mobile Real-time Systems," *EEECS 2016*, Phuket, Thailand, Jan. 2016.

[11] Kyue-Sup Byun and Jai-Hoon Kim, "Determining Checkpointing Intervals for Fault Tolerant Real-Time Systems," *PDPTA 2000*, Las Vegas, June 2000.

[12] Sasikumar Punnekkat and Alan Burns, "Analysis of checkpointing for schedulability of real-time systems," *Real-Time Computing Systems and Applications*, 1997. Article (CrossRef Link)

[13] Sasikumar Punnekkat, Alan Burns and Robert Davis, "Analysis of Checkpointing for Real-Time Systems," *Real-Time Systems*, vol. 20, pp. 83-102, 2001
Article (CrossRef Link)
[14] Kang G. Shin, Tein-Hsiang. Lin and Yann-Hang Lee, "Optimal checkpointing of real-time tasks," *IEEE Trans. Computers*, vol. C-36, pp. 1328–1341, Nov. 1987.
Article (CrossRef Link)
[15] Wolfram Mathematica official website, http://www.wolfram.com.

Sung-Hwa Lim is currently an assistant professor in the Department of Multimedia at Namseoul University. He received B.S., M.S., and Ph.D. degrees in Computer Engineering from Ajou University, South Korea, in 1999, 2001, and 2008, respectively. He was a post-doctoral researcher in the Coordinated Science Lab. at the University of Illinois, Urbana Champaign (UIUC) during 2008–2009. His research interests include the Internet of Things, information-centric networking, and real-time systems.

Byoung-Hoon Lee is currently a principal researcher at NSE Technology Inc. He received B.S. and M.S. degrees in Computer Engineering from Chungbuk National University, in Cheongju, South Korea, in 1998 and 2000, respectively, and Ph.D. in Information and Communication from Ajou University, South Korea, in 2009. He was a research professor at the Institute for Information and Electronics Research at Inha University. His research interests include real-time systems, distributed systems, and embedded programming

Jai-Hoon Kim received the B.S. degree in Control and Instrumentation Engineering from Seoul National University, South Korea, in 1984, M.S. degree in Computer Science from Indiana University, USA, in 1993, and a Ph.D. in Computer Science, from Texas A&M University, USA, in 1997. He is currently a professor in the Cyber Security Department at Ajou University, South Korea. His research interests include distributed systems, cyber-physical systems, and mobile computing.