Wireless Netw (2009) 15:991–998
DOI 10.1007/s11276-008-0097-2

An analysis of location record checkpointing interval for mobility database in PCS networks

Hung-Hsin Chang · Ming-Feng Chang · Chien-Chao Tseng

Published online: 31 January 2008

Abstract Mobility database that stores the users' location records is very important to connect calls to mobile users on personal communication networks. If the mobility database fails, calls to mobile users may not be set up in time. This paper studies failure restoration of mobility database. We study per-user location record checkpointing schemes that checkpoint a user's record into a non-volatile storage from time to time on a per-user basis. When the mobility database fails, the user location records can be restored from the backup storage. Numeric analysis has been used to choose the optimum checkpointing interval so that the overall cost is minimized. The cost function that we consider includes the cost of checkpointing a user's location record and the cost of paging a user due to an invalid location record. Our results indicate that when user registration intervals are exponentially distributed, the user record should never be checkpointed if checkpointing costs more than paging. Otherwise, if paging costs more, the user record should be always checkpointed when a user registers.

H.-H. Chang (✉)
Department of MIS, Chin-min Institute of Technology, MiaoLi, Taiwan, R.O.C.
e-mail: hhchang@csie.nctu.edu.tw

M.-F. Chang · C.-C. Tseng
Department of Computer Science, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C.

M.-F. Chang
e-mail: mfchang@csie.nctu.edu.tw

C.-C. Tseng
e-mail: cctseng@csie.nctu.edu.tw

Keywords Personal communications services · Mobility database · Location register (HLR/VLR) · Failure restoration · Checkpoint

1 Introduction

To set up a call in time to a mobile user in a cellular network, such as GSM (Global System for Mobile Communications) and UMTS (Universal Mobile Telecommunications System), it is necessary to constantly keep track of the mobile user's location. In GSM and UMTS, user location records are stored in a two-level database that consists of HLR (Home Location Register) and VLR (Visitor Location Register) [1]. The HLR resides in the user's home network and maintains mobile users' profile information and the current visited VLRs. For each visiting user in the location areas managed by a VLR, the VLR stores the user's subscription information and current location. When a mobile user crosses a location area, the user needs to register to the VLR and/or the HLR. Thus, the mobility database, HLRs and VLRs, are frequently modified for location tracking and queried for call delivery. If the mobility database fails, calls to mobile users may not be set up in time because of invalid location records.

Many mobility database restoration schemes have been studied. ETSI (European Telecommunications Standards Institute) recommends periodically autonomous registration [2] where a mobile user is required to register its location with the mobility database periodically even if the user does not cross a location area. Therefore, after a location database fails, an invalid location record can be restored sooner by the autonomous registration, and the number of calls lost due to invalid location records is reduced. Haas and Lin [3] considered the tradeoff between the cost of autonomous registration and the penalty of lost calls due to invalid location records. They suggested that the autonomous registration interval should be chosen to be approximately equal to the call inter-arrival time. Fang et al. [4] considered the same cost function, and their study concluded that the optimal choice of autonomous registration interval may not be unique. They also showed that the optimal value can be found under certain traffic conditions. In addition, Fang et al. [5] showed that the optimal autonomous registration interval depends on the weighting ratio between the registration signaling cost and the lost-call cost. To further reduce the time to restore invalid location records, Haas and Lin [6] proposed a demand re-registration scheme where mobile users are requested to re-register after the database fails. This scheme reduces the time to restore the location database. However, since user registration requires radio contact, this demanded re-registration from a large number of mobile users may cause repeated channel collisions, and thus waste wireless resources. Lin and Lin [7] studied a similar problem, the registration interval of badge-based location tracking system. Their results indicated that the channel collisions can be reduced by using exponential registration intervals without increasing the probability of losing calls due to invalid location records.

Checkpointing has also been use to enhance the reliability of a mobility database in presence of failures. UMTS recommends that the mobility database is periodically checkpointed to a non-volatile storage [8]. After a mobility database failure, the user location information can be restored from the non-volatile storage. Checkpointing mobility database is more cost-effective than autonomous registration, because accessing a local non-volatile storage is in general cheaper and faster than accessing a radio channel. If a user's location record is not checkpointed every time when it is updated, the restored record may be out-of-date. In this case, to set up a call, the network can page the user at the location areas around the out-of-date location. Lin [9] derived the optimal checkpointing interval to balance the checkpointing cost against the paging cost, and showed that a user record need not be checkpointed if the checkpointing frequency is higher than 10 times or lower than 0.1 times of the user's moving rate. Wang et al. [10] proposed an aperiodic checkpointing scheme where checkpointing of location database is not performed periodically but is triggered by a threshold on the number of uncheckedpointed location records. They also showed that aperiodic checkpointing outperforms periodic checkpointing when the threshold value is not large. Lin [11] proposed a per-user checkpointing algorithm where a user record is checkpointed only if the user record is modified when the checkpointing timer for the user expires. Otherwise, checkpointing is performed when the user registers for the next time. Since mobile users exhibit different characteristics in terms of registration and calling behavior, per-user checkpointing schemes can serve each user better than a whole-system scheme, but the system has to maintain a checkpointing timer for each user. This timer maintenance job seems to be a large overhead to the system, but the hashed and hierarchical timing wheels, designed by Varghese and Lauck [12], take constant ($O(1)$) time to maintain n outstanding timers, i.e., the time complexity is independent of the number of timers.

In summary, per-user checkpointing schemes can serve each user best without much overhead. However, no analysis has been done on the choice of the checkpointing intervals for per-user checkpointing scheme. In this paper, we study three per-user checkpointing schemes and consider a cost function consisting of the checkpointing cost and the paging cost. Numeric analysis was used to derive the optimal checkpointing frequency when user registration interval is exponentially distributed.

Note that checkpointing and rollback-recovery has long been used to reduce the expected execution time of long-running computation [13–16]. A job run on a computer can be modeled as a transition through a sequence of computation states enabled by computing of the processor(s). When a failure occurs, the computation of the job can be restarted from the most recent checkpoint (an error-free computation state saved before). The transition of computation state is extremely rapid given today's computation power, typically trillions of times per second. On the other hand, the checkpointing of program state is costly due to its size and the relatively slow speed of nonvolatile memory. The question is how often to checkpoint so that the total computation time is minimized. This checkpointing and rollback-recovery problem becomes more complicated if one considers parallel processing issues of multi-processor system [17, 18], or the checkpoint location issue of a moving server system [19].

In our work, the checkpointing of the location database of personal communications services (PCS) is different from that of the computation state in two aspects. First, the current user location is not computed from the previous location. When the database fails and the backup has an invalid location record, the valid location can only be restored by paging the user or by a latter user registration. Second, a user location changes at a very slow rate, typically once in dozens of minutes; on the other hand, the checkpointing of a location record can be done in a flash, typically tens of miniseconds, since it is only a couple of bytes. Therefore, when a user registers a new location (this occurs once in a couples of minutes), we consider whether the user location record needs to be checkpointed or not. If we decide to checkpoint, and since it is unnecessary to checkpoint more than once in an inter-registration interval, the remaining question is when to do it. The location checkpointing can be described as a delayed event after the user registers. The question is how long the delay is. In this paper, the delay is determined by two types of timers (fixed and exponential) and the expected expiration interval of the timers varies in the range of $0-\infty$.

The rest of the paper is organized as follows. Section 2 describes three per-user checkpoint algorithms, and their analytic models are presented in Sect. 3. Numeric results are discussed in Sect. 4. Conclusions are given in Sect. 5.

2 Three per-user checkpointing algorithms

To simplify our discussion, all the events that lead to location update of a mobile user, such as registration, call origination, and crossing of location areas, will be referred to as registration. Since accessing a radio channel is more expensive than accessing a local storage, we assume that no autonomous registration is performed. Note that for a per-user database checkpointing algorithm, the checkpointing timer and the registration interval for each user may be different.

Three per-user database checkpointing algorithms are depicted in Fig. 1. The notation used in the figures is described as follows. t_r denotes the interval between two consecutive registrations and T_C denotes the checkpointing timer. In general, when T_C expires, the user record is checkpointed if it has been updated.

- Periodically checkpointing a modified record (FIXED)

The first scheme is essentially the same as the UMTS checkpointing method except that it is performed on a per-user basis and that only a modified location record is checkpointed. It works as follows,

1. When a user record is checkpointed, a timer, T_C, of fixed expiration interval is set on (see t_0 in Fig. 1(a)).
2. When T_C expires, if the user record has been modified, the user record is checkpointed (see t_3 and t_6 in Fig. 1(a)). Otherwise, if the user record has not been modified when T_C expires, T_C is restarted (see t_4 in Fig. 1(a)) and the process repeats.

This scheme will be referred to as FIXED, because a timer of fixed expiration interval is used.

- Lin's per-user checkpointing algorithm with an exponential timer (LINEXP)

Lin presented a per-user checkpointing algorithm [11], which is illustrated in Fig. 1(b). His algorithm assumes that timer T_C is exponentially distributed with mean $1/\lambda$. The algorithm is described as follows,

1. T_C is started when a user record is checkpointed (see t_0 in Fig. 1(b)).
2. When T_C expires, if the user record has been updated (see t_3 in Fig. 1(b)), it is checkpointed. Otherwise, if the user record has not been updated (see t_4 in Fig. 1(b)), the user record is checkpointed at the next user registration (see t_5 in Fig. 1(b)).

Lin's algorithm differs from the FIXED scheme in that when the timer expires and the user record is not modified, the user record is checkpointed at the next user registration, but scheme FIXED waits until the timer expires after the next user registration. This algorithm will be referred to as LINEXP.

- Lin's per-user checkpointing algorithm with a fixed checking interval (LINFIX)

To study the effects of exponential timers and fixed timers, we apply fixed timers to Lin's per-user checkpointing algorithm. The algorithm is identical to LINEXP except that timer T_C is of fixed expiration interval. An example user registration and checkpointing scenario can be found in Fig. 1(c). This algorithm will be referred to as LINFIX.

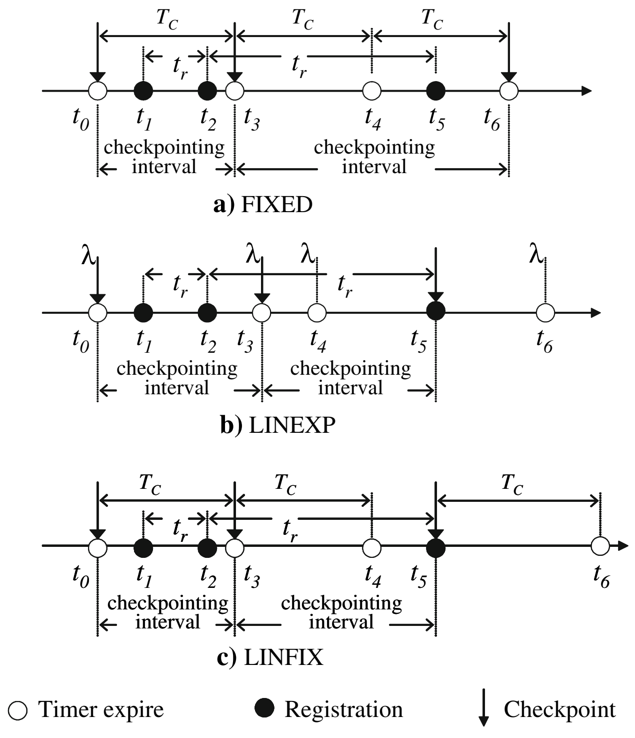


Fig. 1 Three per-user checkpointing algorithms

3 Numeric analysis

The cost function we consider in the paper includes the cost of paging a user with an invalid location record and the cost of checkpointing a user's location record. Let P_{ib} denote the probability that a user record in the backup database is invalid when the main database fails. When an invalid user record is encountered by an incoming call, the network pages the user. Let t_f denote the average database failure interval. It can be shown that the paging cost is proportional to P_{ib}/t_f. Let I denote the expected length of the checkpointing interval. The checkpointing cost is proportional to $1/I$. Let c_b denote the cost of checkpointing a location record, and c_p denote the expected cost to page a user with an invalid location record due to mobility database failure. For the cost function we consider, the checkpointing cost equals to $c_b(1/I)$, and the paging cost equals to $c_p(P_{ib}/t_f)$. The cost function is given as follows

$$C = c_b \cdot 1/I + c_p\left(P_{ib}/t_f\right) \tag{1}$$

We will study the effects of changing the expiration interval of T_C on the total cost, and try to find the optimal timeout interval to minimize the total cost. In our analysis, t_r is assumed to be exponentially distributed with mean $1/u$. The failure of mobility database is assumed to be a random distribution with a mean value an order larger than $1/u$ (the expected inter-registration interval).

- FIXED

Let T denote the expiration interval of timer T_C. Consider two consecutive checkpoints, checkpoints A and B, as shown in Fig. 2. At checkpoint A, the user record is checkpointed and timer T_C is activated. Since the user registers after T_C expires for the $(i-1)$th time and before the ith time, the user record is checkpointed when T_C expires for the ith time, at checkpoint B.

Let Q_i denote the probability that the interval between two consecutive checkpoints is of length iT, i.e., the user registers between time $(i-1)T$ and iT. We have

$$Q_i = \int_{(i-1)T}^{iT} ue^{-ut}dt = e^{-u(i-1)T}\left(1 - e^{-uT}\right)$$

The expected checkpointing interval can be obtained as follows.

$$I_{FIXED} = \sum_{i=1}^{\infty} iTQ_i = \frac{T}{1 - e^{-uT}} \tag{2}$$

Since the inter-registration interval has an IID (independent identically distribution). The user registrations can be modeled as a renewal process. The behavior of checkpointings is also a renew process; because at each checkpoint, timer T_C is restarted and the

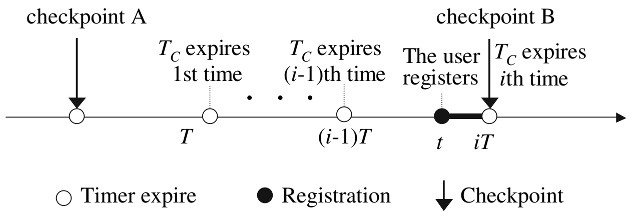


Fig. 2 Two consecutive checkpoints (FIXED)

registration interval is exponentially distributed. For a reliable mobility database, we expect the interval between two consecutive database failures is significantly larger than the user registration interval and the checkpointing interval. In this situation, the time when the database fails can be seen as a random observer to the renew process of user registration and that of checkpointing. The backup user record is invalid only after the user registers and before the record is checkpointed. If the main database fails during this period, the system restores an invalid backup record. Thus, we have

$$P_{ib\_FIXED} = \left(\sum_{i=1}^{\infty} \int_{(i-1)T}^{iT} ue^{-ut}(iT - t)dt\right) \Big/ I_{FIXED}$$
$$= 1 - \frac{1 - e^{-uT}}{uT} \tag{3}$$

From (1) to (3) the cost function can be obtained as follows,

$$C_{FIXED} = c_b \cdot \frac{1 - e^{-uT}}{T} + \frac{c_p}{t_f}\left(1 - \frac{1 - e^{-uT}}{uT}\right)$$
$$= \frac{c_p}{t_f} + \frac{1}{u}\left(uc_b - \frac{c_p}{t_f}\right)\left(\frac{1 - e^{-uT}}{T}\right) \tag{4}$$

Our goal is to minimize the cost by choosing an appropriate T.

$$\frac{d}{dT}\left(\frac{1 - e^{-uT}}{T}\right) = \left(-1 + \frac{1 - uT}{e^{uT}}\right) \Big/ T^2$$

Since $e^{uT} = 1 + uT + (uT)^2/2! + (uT)^3/3! + \cdots$ and $uT > 0$, we have $\frac{1+uT}{e^{uT}} \le 1$ and $\left(-1 + \frac{1+uT}{e^{uT}}\right)/T^2 \le 0$ for $T \ge 0$, $u \ge 0$. This leads to that $\frac{1-e^{-uT}}{T}$ is a monotonic decreasing function of T. From (4), we can draw the conclusions below,

1. If $uc_b < \frac{c_p}{t_f}$, C_{FIXED} is a monotonic increasing function of T. C_{FIXED} can be minimized when $T = 0$, i.e., the expiration interval of the timer is of length 0. At each user registration, since the timer must have expired, the user record should be checkpointed. In this case, $C_{FIXED} = uc_b$.
2. If $uc_b > \frac{c_p}{t_f}$, C_{FIXED} is a monotonic decreasing function of T. C_{FIXED} can be minimized when T $= \infty$, i.e., the expiration interval of the timer is of infinite length. Since the timer never expires, the user record should never be checkpointed. In this case $C_{FIXIED} = \frac{c_p}{t_f}$.
3. If $uc_b = \frac{c_p}{t_f}$, $C_{FIXIED} = uc_b = \frac{c_p}{t_f}$; $\mathrm{C_{FIXED}}$ is a constant independent of T. T can be any value, i.e., at a user registration, the user record can be either checkpointed or not checkpointed. The cost of checkpointing the record and the cost of not checkpointing (the expected paging cost) are the same.

Note that the minimum cost that scheme FIXED can achieve equals to $Min\left(uc_b, \frac{c_p}{t_f}\right)$.

- LINEXP

The analysis of the LINEXP is similar to that of the FIXED. Considering two consecutive checkpoints, there are two possible conditions as shown in Fig. 3. For Case I, shown in Fig. 3(a), the user registers before timer T_C expires, so that the checkpointing interval is equal to the expiration interval of timer T_C (s in Fig. 3(a)). For Case II, shown in Fig. 3(b), the user registers after timer T_C expires, so that the checkpointing interval is equal to the user registration interval (t in Fig. 3(b)).

Since the registration interval and the checkpointing timer are both exponentially distributed, the expected length of checkpointing interval can be obtained by adding the intervals of both conditions.

$$\begin{aligned} I_{LINEXP} = & \int_0^{\infty} \int_{t=0}^{s} s \cdot ue^{-ut} \cdot \lambda e^{-\lambda s} dtds \\ & + \int_0^{\infty} \int_{t=s}^{\infty} t \cdot ue^{-ut} \cdot \lambda e^{-\lambda s} dtds \\ = & \frac{1}{\lambda} + \frac{\lambda}{u(\lambda+u)} \end{aligned} \tag{5}$$

For Case I, the backup user record is invalid only after the user registration at time t. For Case II, the backup user record is always up-to-date because when the user registers, the record is also checkpointed. From the random observer property, P_{ib} can be obtained as follows.

$$\begin{aligned} P_{ib\_LINEXP} = & \left(\int_0^{\infty} \int_{t=0}^{s} (s-t) \cdot ue^{-ut} \cdot \lambda e^{-\lambda s} dtds \right) \Big/ I_{LINEXP} \\ = & \frac{u^2}{u^2 + \lambda u + \lambda^2} \end{aligned} \tag{6}$$

From (1), (5), and (6), the cost function can be obtained as follows,

$$\begin{aligned} C_{LINEXP} = & c_b \cdot \left(\frac{1}{\lambda} + \frac{\lambda}{u(\lambda+u)} \right) + \frac{c_p}{t_f} \cdot \frac{u^2}{u^2 + \lambda u + \lambda^2} \\ = & uc_b + \left(\frac{c_p}{t_f} - uc_b \right) \cdot \frac{u^2}{u^2 + \lambda u + \lambda^2} \end{aligned} \tag{7}$$

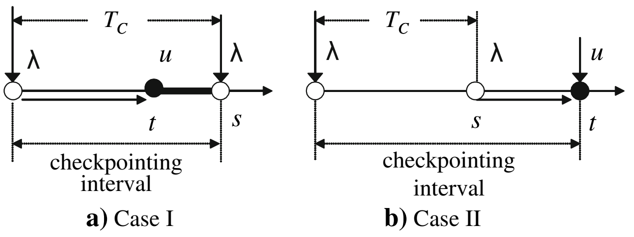


Fig. 3 Two possible cases of checkpointing (LINEXP)

Since $\frac{d}{d\lambda}\left(\frac{1}{u^2+\lambda u+\lambda^2}\right) = \frac{-(u+2\lambda)}{\left(u^2+\lambda u+\lambda^2\right)^2} \le 0$ for $\lambda \ge 0, u \ge 0$, $\frac{1}{u^2+\lambda u+\lambda^2}$ is a monotonic decreasing function of λ. We also obtain the following results, which are essentially the same as those obtained from FIXED. Note that the expected timeout interval of the exponential timer is $1/\lambda$.

1. If $uc_b < \frac{c_p}{t_f}$, C_{LINEXP} is a monotonic decreasing function of λ. The optimum $C_{LINEXP} = uc_b$, when $\lambda = \infty$, i.e., the timeout interval of the checkpointing timer is of length 0.
2. If $uc_b > \frac{c_p}{t_f}$, C_{LINEXP} is a monotonic increasing function of λ. The optimum $C_{LINEXP} = \frac{c_p}{t_f}$, when $\lambda = 0$, i.e., the timeout interval of the checkpointing timer is of infinite length.
3. If $uc_b = \frac{c_p}{t_f}$, $C_{LINEXP} = uc_b = \frac{c_p}{t_f}$, a constant independent of λ. λ can be any value.

- LINFIX

Since this algorithm is identical to algorithm LINEXP except that it utilizes a checkpointing timer with fixed expiration interval. The two checkpointing cases of LINEXP shown in Fig. 3 can also be used to analyze LINFIX. For Case I, the checkpointing interval is equal to the expiration interval of the timer, which is T. For the Case II, the checkpointing interval is equal to the user registration interval (t). The expected length of checkpointing interval can be obtained as follows.

$$I_{LINFIX} = \int_0^{T} T \cdot ue^{-ut} dt + \int_T^{\infty} t \cdot ue^{-ut} dt = T + \frac{e^{-uT}}{u} \tag{8}$$

P_{ib} equals to the probability that the main database fails in Case I after the user registration.

$$\begin{aligned} P_{ib\_LINFIX} = & \left(\int_0^{T} (T-t) \cdot ue^{-ut} dt \right) \Big/ I_{LINFIX} \\ = & 1 - \left(\frac{1}{uT + e^{-uT}} \right) \end{aligned} \tag{9}$$

From (1), (8) and (9), the cost function can be obtained as follows,

$$\begin{aligned} C_{LINFIX} = & \frac{c_b}{T + \frac{e^{-uT}}{u}} + \frac{c_p}{t_f} \cdot \left(1 - \frac{1}{uT + e^{-uT}} \right) \\ = & \frac{c_p}{t_f} + \left(uc_b - \frac{c_p}{t_f} \right) \cdot \frac{1}{uT + e^{-uT}} \end{aligned} \tag{10}$$

Since $\frac{d}{dT}\left(\frac{1}{uT+e^{-uT}}\right) = \frac{-u(1-e^{-uT})}{(uT+e^{-uT})^2}$ and $e^{-uT} \le 1$ for $T \ge 0$, $u \ge 0$, we get $\frac{-u(1-e^{-uT})}{(uT+e^{-uT})^2} \le 0$. Thus, $\frac{1}{uT+e^{-uT}}$ is a monotonic decreasing function of T. We can obtain same results as in the FIXED and LINEXP.

1. If $uc_b < \frac{c_p}{t_f}$, C_{LINFIX} is a monotonic increasing function of T. The optimum $C_{LINFIX} = uc_b$, when $T = 0$.

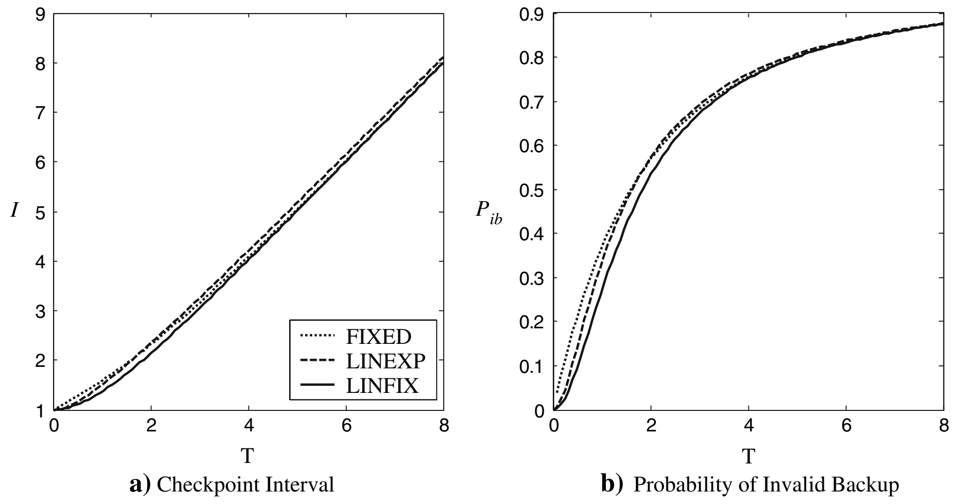


Fig. 4 Comparison of checkpointing algorithms for exponential registration interval

2. If $uc_b > \frac{c_p}{t_f}$, C_{LINFIX} is a monotonic increasing function of T. The optimum $C_{LINFIX} = \frac{c_p}{t_f}$, when $T = \infty$.
3. If $uc_b = \frac{c_p}{t_f}$, $C_{LINFIX} = uc_b = \frac{c_p}{t_f}$. T can be any value.

It is important to note that the analyses of three algorithms all lead to the same conclusions. If the checkpointing cost out-weights the paging cost $(uc_b > \frac{c_p}{t_f})$, we should never checkpoint a user record. On the other hand, if $\frac{c_p}{t_f} > uc_b$, we should use a duplicated database.

4 Numeric results

Without loss of generality, we let the expected user registration rate, u, to be 1 per unit-of-time. This can be interpreted as one registration per x minutes. A small x means the user registers frequently. We consider exponential registration interval and examine the effects of the timeout interval (T) on the expected checkpointing interval (I) and on the probability of invalid backup record at database failure (P_{ib}). The expiration interval of timer T_C used in FIXED and LINFIX varies in the range 0.2–8 unit-of-time. In addition, the expected expiration interval of the exponential timer in LINEXP also varies in the range 0.2–8. The curves in Fig. 4(a) are obtained from Eqs. 2, 5, and 8, and those in Fig. 4(b) from Eqs. 3, 6, and 9. The results indicate that all three algorithms obtain similar results; both the expected checkpointing interval and the probability of invalid backup record increase as the timeout interval increases. The differences between the three algorithms are small, but for a given timeout interval, LINFIX has the smallest P_{ib} at the cost of the shortest checkpointing interval, I. When the timeout interval is larger than 4 (i.e., four times the registration interval), all checkpointing algorithms act much the same. This is because when a long checkpointing timer expires, the user record is most likely modified and needs to be checkpointed for all algorithms.

Figure 5 shows the cost functions at various paging costs; the user registration interval is exponential distributed. The curves are obtained from Eqs. 4, 7, and 10. Without loss of generality, let $u = 1$, $c_b = 1$, and $\frac{c_p}{t_f}$ vary in the range of 0.5–1.5. The results indicate when $uc_b = \frac{c_p}{t_f}$,

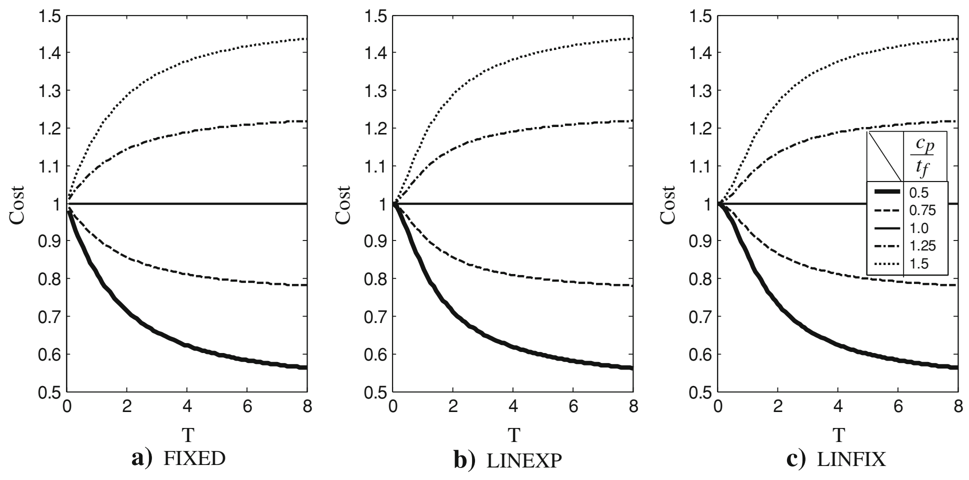


Fig. 5 Cost functions for various paging costs (exponential registration interval)

the cost of all algorithms equals to 1 $(=\frac{c_p}{t_f}=uc_b)$, and the total cost is independent of the timer expiration interval. Furthermore, when $\frac{c_p}{t_f}>uc_b$, the cost increases as the timeout interval, T, increases, and when $uc_b>\frac{c_p}{t_f}$, the cost decreases as T increases.

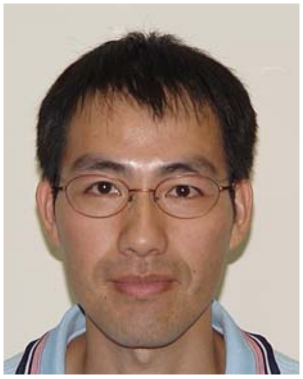
5 Conclusions

Checkpointing can be used to enhance the reliability of the location database of PCS networks, since each user exhibits a unique calling and moving behavior, per-user checkpointing schemes can serve the users, as well as the operators, better. In this paper, we have analyzed three per-user location database checkpointing algorithms using numeric analysis. The costs that we considered include the checkpointing cost and the paging cost. Our results indicate that when inter-registration times are exponentially distributed, a user location record should either be always checkpointed at registration, or be never checkpointed at all, depending on the weighting ratio between the checkpointing cost and the paging cost. If the checkpointing cost is of more concern, the user record should never be checkpointed; otherwise, the user record should be always checkpointed (duplicated) at registration. In this paper, we did not investigate the effects of incoming call arrivals on the optimal choice of the checkpointing frequency directly; we assumed that the expected paging cost is known. Further study is needed to obtain the paging cost in the PCS networks. In addition, if paging a user with an invalid location record cannot be done in time, the caller may hang up and the call is lost. It may be meritorious to consider a cost function consisting of the checkpointing, paging and lost-call costs.

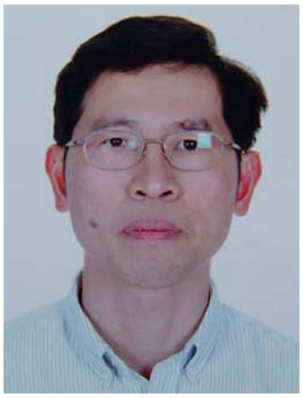
References

1. EIA/TIA. (1991). Cellular radio-telecommunication intersystem operation: Automatic roaming. BEIA/TIA Technical Report IS-41.3.
2. ETSI/TC. (1993). Restoration procedures, version 4.2.0. ETSI Technical Report Recommendation GSM 03.07.
3. Haas, Z. J., & Lin, Y.-B. (1998). On the optimizing the location update costs in the presence of database failures. *ACM-Baltzer Wireless Networks, 4*, 419–426.
4. Fang, Y., Chlamtac, I., & Fei, H. B. (2000). An active mobility database failure recovery scheme and performance analysis for PCS networks, In *Conference of the IEEE Communications Society (March 2000)* (pp. 757–764).
5. Fang, Y., Chlamtac, I., & Fei, H. B. (2000). Analytical results for optimal choice of location update interval for mobility database failure restoration in PCS networks. *IEEE Transactions on Parallel and Distributed Systems, 11*, 615–624.
6. Haas, Z. J., & Lin, Y.-B. (1999). Demand re-registration for PCS database restoration. In *Proceedings Military Communications Conference 2 (October–November, 1999)*. (pp. 887–892).
7. Lin, Y.-B., & Lin, P. (1998). Performance modeling of location tracking systems. *ACM Mobile Computing and Communications Review, 2*, 24–27.
8. 3GPP. (June 2001). 3rd Generation partnership project: Technical specification group services and system aspects; General packet radio service (GPRS); Service description; Stage 2, Technical Specification 3G TS 23.060 version 4.1.0.
9. Lin, Y.-B. (1995). Failure restoration of mobility database for personal communication networks. *ACM-Baltzer Wireless Networks, 1*, 365–372.
10. Wang, T.-P., Tseng, C.-C., & Chou, W.-K. (1997). An aggressive approach to failure restoration of PCS mobility databases. *ACM Mobile Computing and Communications Review, 1*, 21–28.
11. Lin, Y.-B. (2005). Per-user checkpointing for mobility database failure restoration. *IEEE Transactions on Mobile Computing, 4*, 189–194.
12. Varghese, G., & Lauck, A. (1997). Hashed and hierarchical timing wheels: Efficient data structures for implementing a timer facility. *IEEE/ACM Transactions on Networking, 5*, 824–834.
13. Young, J. W. (1974). A first order approximation to the optimum checkpoint interval. *ACM communications, 17*, 530–531.
14. Gelenbe, E. (1979). On the optimum checkpoint interval. *Journal of the Association for Computing Machinery, 26*, 259–270.
15. Plank, J. S., & Elwasif, W. R. (1998). Experimental assessment of workstation failures and their impact on checkpointing systems. In *28th International Symposium on Fault-Tolerant Computing* (pp. 48–57).
16. Plank, J. S., Li, K., & Puening, M. A. (1998). Diskless checkpointing. *IEEE Transactions on Parallel and Distributed Systems, 9*, 972–986.
17. Bruno, J. L., Coffman, E. G. (1997). Optimal fault-tolerant computing on multiprocessor systems. *Acta Informatica, 34*, 881–904.
18. Bruno, J. L., Coffman, E. G., Lagarias, J. C., Richardson, T. J., & Shor, P. W. (1999). Processor shadowing: Maximizing expected throughput in fault-tolerant systems. *Mathematics of Operations Research, 24*, 362–382.
19. Coffman, E. G., Flatto, L., & Wright, P. E. (1993). A stochastic checkpoint optimization problem. *SIAM Journal of Computing, 22*, 650–659.

Author Biographies



Hung-Hsin Chang received the B.S., M.S. and Ph.D. degree in Computer Science and Information Engineering from the National Chiao-Tung University, Taiwan, R.O.C., in 1990, 1995 and 2005. He is currently a teacher in Chin-Min Institute of Technology, Taiwan. His current research interests include design and analysis of wireless network, Internet communications, and personal communications network.



Ming-Feng Chang received the Ph.D. degree in Computer Science from the University of Illinois at Urbana-Champaign in 1991. He is currently a Professor in the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan, R.O.C. His current research interests include design and analysis of Internet communications, personal communications network, mobile payment, and performance modeling.

Chien-Chao Tseng is currently a professor in the Department of Computer Science at National Chiao-Tung University, Hsin-Chu, Taiwan. He received his B.S. degree in Industrial Engineering from National Tsing-Hua University, Hsin-Chu, Taiwan, in 1981; M.S. and Ph.D. degrees in Computer Science from the Southern Methodist University, Dallas, Texas, USA, in 1986 and 1989, respectively. His research interests include Wireless Internet, Handover Techniques for Heterogeneous Networks, and Mobile Computing.