モバイルコンピューティングとワイヤレス通信 20－38
(2002. 3. 8)

Hybrid Checkpoint Protocol in Wireless LAN Environment

Yoshinori Morita, Masamitsu Miyazaki and Hiroaki Higaki
Department of Computers and Systems Engineering
Tokyo Denki University
E-mail: {mine,miya,hig}@higlab.k.dendai.ac.jp

For supporting mission-critical applications in a mobile network system, hybrid checkpointing has been proposed. In a recent mobile network, wireless LAN protocols such as IEEE 802.11 and HIPERLAN are getting popular and communication with mobile computers is realized by using Mobile IP in the Internet. This paper proposes a novel hybrid checkpoint protocol. Here, message logging for mobile computers is achieved based on broadcast property of wireless LAN protocols. In addition, by extending Mobile IP, network overload in recovery is avoided. For both checkpointing and recovery in the proposed protocol, all required information is piggied back to messages. That is, no additional message is required.

無線LAN環境における複合チェックポイントプロトコル

東京電機大学 理工学部 情報システム工学科
森田 義徳　宮崎 正光　桧垣 博章
E-mail: {mine,miya,hig}@higlab.k.dendai.ac.jp

信頼性のあるネットワークシステムを実現する方法の一つにチェックポイントリカバリがある。ノートPCやPDAなどのモバイルコンピュータが、IEEE 802.11などの無線LANプロトコルを用いて通信を行なうモバイルネットワークシステムにおいて耐故障性を高めるために、複合チェックポイントが提案されている。本論文では、無線通信セル内に存在する2つのモバイルコンピュータが基地局を介さない通信(アドホックモード通信)を行なうという仮定のもとで新しい複合チェックポイントプロトコルを設計し、その評価を行なった。同一セルに属するモバイルコンピュータ間で送受されるブロードキャストメッセージを基地局が受信する方法により、チェックポイントリカバリに必要なメッセージログを作成するための制御メッセージ数が削減されている。また、Mobile IPへの対応を行なうことにより、リカバリ時に輻湊が発生することを防止している。

1 Background and Objective

According to the advances of computer and communication technologies, many kinds of mobile computers like notebook computers and personal data assistants (PDAs) are widely available. Intelligent Transport Systems (ITS) with mobile communication are now being developed. A mobile network system is composed of *fixed computers* and *mobile computers* interconnected by communication networks. A fixed computer is located at a fixed location in the network. A mobile computer moves from one location to another in the network. The mobile network is divided into multiple *wireless cells*. A mobile computer moves from one wireless cell to another and sometimes out of any wireless cell. There is an *access point* in each wireless cell. Access points and fixed computers are interconnected by a high-speed wired network. A mobile computer communicates directly with another mobile computer in the same wireless cell. It communicates through an access point supporting it with a fixed computer and a mobile computer in another wireless cell. This is realized by using wireless LAN protocols such as IEEE 802.11 [15] and HIPERLAN [14].

In a network system, applications are realized by cooperation of multiple computers. Usually, computers and networks are developed by using widely available products including personal computers, mobile computers, engineering workstations, Ethernets, routers, repeaters, switches and so on. Mission-critical applications cannot always be implemented in such a system. Hence, it is important to discuss how to make and keep the system reliable and available. Checkpoint-recovery [2, 5, 13] is one of the well-known methods to achieve reliable network systems. Each computer s_i takes a local checkpoint c_i where local state information of s_i is stored into a stable storage. If some computer fails, s_i restarts from c_i. A global checkpoint which is a set of local checkpoints is required to be *consistent* [2]. Fixed computers take consistent checkpoints by using *synchronous checkpoint protocols* [2, 5] with low synchronization overhead since they communicate with each other through a high-speed wired network and they have enough amount of stable storages to store state information [3, 6]. Some papers [7, 10] discuss synchronous checkpoint protocols for mobile computers. However, it requires high communication and synchronization overhead for mobile and fixed computers to take checkpoints synchronously due to mobility and lack of battery capacity of mobile computers. Moreover, it is difficult for mobile computers to take local checkpoints by themselves since they have neither enough volume of stable storages nor so much battery capacity as to frequently access the stable storage [7]. In a protocol in [7], mobile computers in a wireless cell take a consistent global checkpoint without communication by using synchronized realtime clocks and state information of a mobile computer is stored into a stable storage in a fixed computer. However, it is difficult to achieve synchronized realtime clocks since message transmission delay among mobile computers is unpredictable. Mobile computers may fail to take local checkpoints due to lack of battery capacity or movement to outside of any wireless cell. In a synchronous checkpoint protocol, every computer has to give up to take a consistent global checkpoint if a certain mobile computer fails to take a local checkpoint. Hence, *asynchronous checkpoint protocols* for mobile computers [1, 9] have been proposed. However, in these protocols, each access point is required to take a local checkpoint for a mobile computer each time a message is transmitted between them. Thus, high synchronization and storage access overheads are required.

In order to solve this problem, *hybrid checkpointing* where local checkpoints are asynchronously taken by mobile computers while synchronously taken by fixed computers has been proposed [4]. Mobile computers take local checkpoints by storing state information into stable storages in access points. Here, local checkpoints of mobile computers are taken when they send a checkpoint request message to an access point. By combining synchronous and asynchronous checkpoint protocols, number of checkpoints is reduced. Hence, frequency of accesses to stable storages is also reduced. Thus, hybrid checkpointing makes mobile systems so reliable that mission-critical applications are implemented with less overhead. In a proposed protocol in [4], every message from a mobile computer is assumed to be transmitted through an access point supporting the wireless cell. Thus, even if a message is exchanged between mobile computers within a wireless cell, the message is required to be forwarded by the access point. However, in wireless LAN protocols such as IEEE 802.11 and HIPERLAN, every message transmitted by a mobile computer is broadcasted and mobile computers within a wireless cell communicate directly. This paper proposes another hybrid checkpoint protocol for supporting wireless LAN protocols and evaluates the performance.

2 System Model

A network system $\mathcal{S} = \langle \mathcal{V}, \mathcal{L} \rangle$ is composed of a set $\mathcal{V} = \{s_1, \ldots, s_n\}$ of computers and a set $\mathcal{L} \subseteq \mathcal{V}^2$ of communication channels. An execution of an application is realized by cooperation of multiple computers communicating with each other by exchanging messages through communication channels. $\langle s_i, s_j \rangle \in \mathcal{L}$ indicates a communication channel from a computer s_i to another computer s_j. Each $\langle s_i, s_j \rangle$ is assumed to be reliable with the help of underlying protocols. A *state* of s_i is updated at each *event* in s_i. There are two kinds of events; *local events* and *communication events*. At a local event, s_i updates state by local computation without exchanging a message. At a communication event, s_i communicates with another computer by exchanging a message and updates state. There are two kinds of communication events; a *message sending event* $s(m)$ and a *message receipt event* $r(m)$ for a message m.

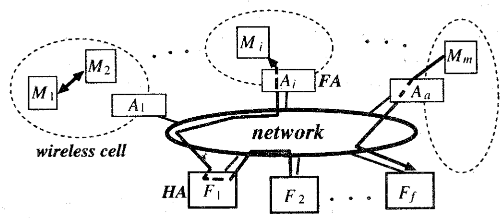

Figure 1: Mobile network system.

In a mobile network system, which is a kind of $\mathcal{S}$, there are three kinds of computers; *fixed computers* $F_1, \ldots, F_f$, *mobile computers* $M_1, \ldots, M_m$ and *access points* $A_1, \ldots, A_a$ as shown in Figure 1. F_i is connected at a fixed location in the network. M_i moves from one location to another. M_i communicates with another computer by using a wireless LAN protocol like IEEE 802.11 [15]. Here, each M_i is included in a single wireless cell. A message exchanged between M_i and M_k in the same wireless cell is transmitted directly by using CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) protocol. Reliable communication is achieved by using acknowledgment messages and retransmission timers. In addition, since a wireless LAN protocol is intrinsically broadcast-base, a message transmitted from M_i is received by all the computers including an access point in the same wireless cell. Messages exchanged between M_i and a mobile computer in another wireless cell and between M_i and a fixed computer are transmitted through an access point. These messages are also transmitted by using a broadcast-based communication protocol.

In the Internet, Mobile IP [8] is widely available for supporting communication of mobile computers. For each mobile computer M_i, there are two agent processes; a home agent HA_i and a foreign agent FA_i. HA_i is at a fixed location in a communication network, i.e. in a certain access point. FA_i is in an access point supporting a wireless cell including M_i. Hence, FA_i moves according to the movement of M_i. Each time M_i moves out of a wireless cell, FA_i sends a registration message for release of a connection to HA_i. In addition, each time M_i moves into a wireless cell, FA_i sends a registration message for establishment of a connection to HA_i. A message m destined to M_i sent by a fixed computer or a mobile computer in another wireless cell is transmitted to HA_i regardless of the wireless cell including M_i. Since HA_i knows a location of FA_i, HA_i forwards m to FA_i by using IP tunneling. On receipt of m, FA_i forwards m to M_i.

3 Hybrid Checkpointing

Synchronous checkpoint protocols have an advantage that computers restart from the most recent local checkpoints without domino effect. However, it is difficult for multiple mobile computers to take local checkpoints synchronously for high synchronization and communication overhead due to mobility and lack of resources. Hence, *hybrid checkpointing* in Figures 2 and 3 has been proposed [4].

[Hybrid checkpointing]

- Each fixed computer F_i takes a local checkpoint c_{F_i} by using a synchronous checkpoint protocol. A set $\tilde{C} = \{c_{F_1}, \dots, c_{F_f}\}$ of local checkpoints taken by the fixed computers is referred to as a *coordinated checkpoint.*
- Each mobile computer M_i takes a local checkpoint c_{M_i} by using an asynchronous checkpoint protocol. □

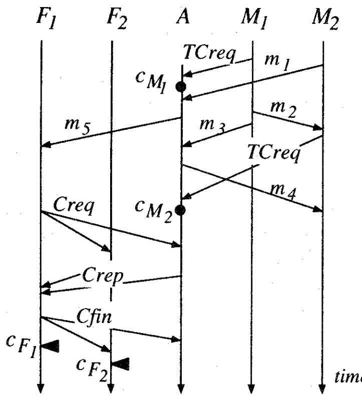

Figure 2: Checkpoint in hybrid protocol.

At c_{M_i}, state information of M_i is stored into a stable storage in an access point A_j where M_i is included in a wireless cell supported by A_j. M_i fails to take c_{M_i} if M_i moves out of the wireless cell or battery power in M_i is exhausted. Thus, M_i takes c_{M_i} only if M_i does not move out of a wireless cell and has enough battery power for taking c_{M_i}. Hence, M_i asynchronously takes c_{M_i}, i.e. independently of the other computers. M_i has to restart from a local state consistent with $\tilde{C}$. However, c_{M_i} is not always consistent with $\tilde{C}$ since M_i takes c_{M_i} independently of the fixed computers taking $\tilde{C}$. Hence, a kind of log-based restart protocols [11, 12] is designed as shown in Figure 3. Messages exchanged between M_i and other computers

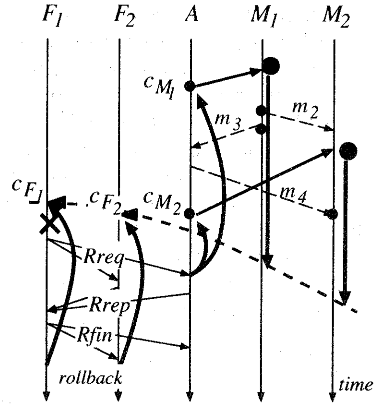

Figure 3: Restart in hybrid protocol.

after taking c_{M_i} are stored into a stable storage in A_j. In recovery, M_i restores the state information at c_{M_i} and the messages from the stable storage in A_j. After replaying a sequence of events according to the messages, M_i gets a state consistent with $\tilde{C}$ and restarts. While M_i replays, M_i does not exchange messages with other computers. There is a *checkpoint agent process* p_{ij} for M_i in A_j. p_{ij} stores state information of M_i at c_{M_i} into a stable log sl_{ij}. p_{ij} also stores m into a message log ml_{ij} on behalf of M_i.

4 Message logging in wireless LAN

In a hybrid checkpoint protocol proposed in [4], every message transmitted and received by a mobile computer M_i is assumed to be transmitted through an access point A_j supporting communication of M_i. That is, a message m destined to a computer S from M_i is sent to A_j and A_j forwards m to S directly if S is a mobile computer in the same wireless cell or through a wired network if S is a fixed computer or a mobile computer in another wireless cell. A message m destined to M_i from S is received by A_j and A_j forwards m to M_i directly. In this communication model, a checkpoint agent process p_{ij} for M_i in A_j stores all the messages exchanged by M_i into a stable storage in A_j on behalf of M_i since p_{ij} resides in A_j. However, in wireless LAN protocols such as IEEE 802.11 and HIPERLAN, mobile computers within a wireless cell communicate directly without the help of an access point. Hence, it is critical how p_{ij} stores the messages into a stable storage for M_i to achieve a consistent state for recovery. Here, wireless LAN protocols are broadcast-based protocols. A message sent by a mobile computer or an access point in a wireless cell is received by all the mobile computers and the access point in the wireless cell. Thus, a message m from a mobile computer M_i is received by an access point A_j even if m is destined to another mobile computer M_j in the same wireless cell. Thus, a checkpoint agent process p_{ij} for M_i in A_j stores sufficient state information and messages to get a consistent local state with a coordinated checkpoint

$\tilde{C}$ in recovery. Here, the order of events sending and receiving the stored messages in recovery is significant. Only by receiving messages broadcasted from M_i by a wireless LAN protocol, the order of events observed by A_j might be different from that really occurred in M_i.

Now, we consider the following two cases for achieving a consistent order of events in a mobile computer M_i:

- A message m is transmitted between M_i in a wireless cell of an access point A_j and a mobile or fixed computer out of the wireless cell [case 1].
- A message m is transmitted between two mobile computers M_i and M_k within a wireless cell of A_j [case 2].

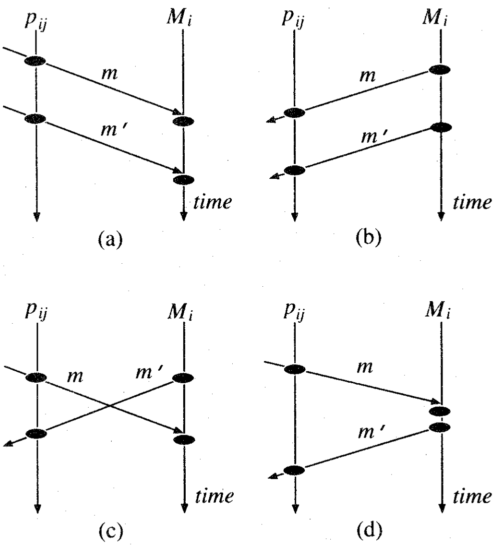

Figure 4: Message Logging [case 1].

[case 1] In A_j, there exists a checkpoint agent process p_{ij} for M_i. First, suppose that p_{ij} forwards two messages m and m' to M_i in this order as shown in Figure 4(a). Here, p_{ij} knows occurrences of $r(m)$ and $r(m')$ in M_i and stores m and m' into a message log for recovery. Since p_{ij} identifies an order of occurrences of $r(m)$ and $r(m')$ in M_i, p_{ij} gets an ordered message log for recovery. Next, suppose that p_{ij} forwards two messages m and m' from M_i in this order as shown in Figure 4(b). Here, p_{ij} knows occurrences of $s(m)$ and $s(m')$ in M_i and stores m and m' into a message log for recovery. Since p_{ij} identifies an order of occurrences of $s(m)$ and $s(m')$ in M_i, p_{ij} gets an ordered message log for recovery. Finally, suppose that p_{ij} forwards m to M_i and m' from M_i as shown in Figures 4(c) and 4(d). Here, p_{ij} knows occurrences of $r(m)$ and $s(m')$ in M_i and stores m and m' into a message log for recovery. However, p_{ij} does not identify an order of occurrences of $r(m)$ and $s(m')$ in M_i. □

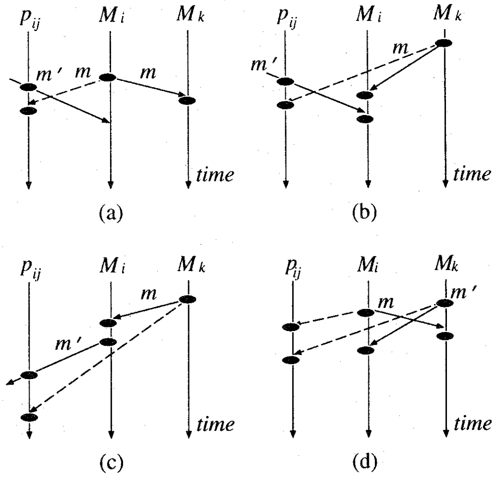

Figure 5: Message Logging [case 2].

[case 2] In A_j, there exists a checkpoint agent process p_{ij} for M_i. First, suppose that M_i sends a message m to M_k and p_{ij} forwards another message m' to M_i as shown in Figure 5(a). Here, since m is transmitted by a wireless LAN protocol, m is broadcasted in the wireless cell and A_j receives m. Hence, p_{ij} knows occurrence of $s(m)$ in M_i and stores m into a message log for recovery. However, p_{ij} does not identify an order of occurrences of $s(m)$ and $r(m')$ in M_i. Next, suppose that M_i receives a message m from M_k and p_{ij} forwards another message m' to M_i as shown in Figure 5(b). Here, since m is transmitted by a wireless LAN protocol, m is broadcasted in the wireless cell and A_j receives m. Hence, p_{ij} knows occurrences of $s(m')$ and $r(m)$ in M_i and stores m into a message log for recovery. However, p_{ij} does not identify an order of occurrences of $r(m)$ and $r(m')$ in M_i. As shown in Figure 5(c), suppose that M_i receives a message m from M_k and p_{ij} forwards another message m' from M_i. Here, since m is transmitted by a wireless LAN protocol, m is broadcasted in the wireless cell and A_j receives m. Hence, p_{ij} knows occurrence of $r(m)$ in M_i. However, p_{ij} does not identify an order of occurrences of $s(m')$ and $r(m)$ in M_i. Finally, suppose that M_i sends a message m to M_k and M_k sends another message m' to M_i as shown in Figure 5(d). Here, since m and m' are transmitted by a wireless LAN protocol, m and m' are broadcasted in the wireless cell and A_j receives m and m'. Hence, p_{ij} knows occurrences of $s(m)$ and $r(m')$ in M_i. p_{ij} stores m and m' into a message log for recovery. However, p_{ij} does not identify an order of occurrences of $s(m)$ and $r(m')$ in M_i. □

Though p_{ij} receives all the messages transmitted and received by M_i, it is impossible for p_{ij} to store messages into a message log in the same order as M_i transmits and receives them by only receiving them broadcasted by using a wireless LAN protocol. One idea to get correctly ordered message log for M_i is that M_i stores messages into a volatile storage temporarily and transmits them to p_{ij} for storing them into a stable storage. However, a mobile computer does not always have enough capacity of storage to store the messages. Moreover, additional messages are required to be transmitted for carrying the messages from a mobile computer to an access point. Thus, communication overhead is increased.

In order to solve this problem, we design the following protocol:

- A message transmitted in a wireless cell is stored into an unordered message buffer $mbuf_i$ for M_i in an access point temporarily. Even if the message is transmitted between mobile computers in a wireless cell, the access point receives the message due to broadcast property of a wireless LAN protocol.
- Order information of message sending and receipt events in M_i is piggied back to another message transmitted from M_i. Even if the message is transmitted between mobile computers within a wireless cell, an access point can get the order information due to broadcast property of a wireless LAN protocol.

As a consequence of separation of a message itself and its sending and receipt order information, no additional message is transmitted to store the messages in a consistent order into a stable storage in an access point. In this protocol, a unique identifier is assumed to be assigned to every message. In addition, M_i has a variable $Rseq_i$ which holds a sequence of identifiers of messages sent by M_i. Initially, $Rseq_i = \phi$.

[Message logging in M_i]

- At a message receipt event $r(m)$ for a message m, a message identifier $id(m)$ of m is added to the end of $Rseq_i$. m is delivered to an application.
- At a message sending event $s(m)$ for a message m, $Rseq_i$ is piggied back to m and m is transmitted. $Rseq_i = \phi$.

[Message logging in p_{ij}]

- On receipt of a message m destined to M_i, p_{ij} stores a copy of m into $mbuf_i$. p_{ij} forwards m to M_i if m is from a computer out of the wireless cell.
- On receipt of a message m transmitted from M_i,
  1) p_{ij} takes messages, whose identifiers are included in $Rseq_i$ which is a sequence of message identifiers piggied back to m, out of $mbuf_i$. If all the required messages have not yet stored into $mbuf_i$, this procedure is suspended. If p_{ij} receives a message destined to M_i, this procedure is restarted after storing the message into $mbuf_i$.
  2) p_{ij} stores the messages into a tentative message log tml_{ij} in a volatile storage according to the order of message identifiers in $Rseq_i$.
  3) p_{ij} stores m into tml_{ij}.
  4) p_{ij} forwards m to a destination computer if m is destined to a computer out of the wireless cell. □

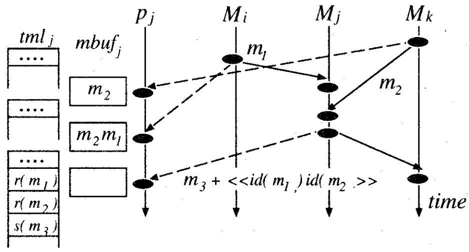

Figure 6: Message logging.

5 Checkpoint protocol

In our hybrid checkpoint protocol, fixed computers $F_1, \ldots, F_f$ take a coordinated checkpoint $\tilde{C} = \{c_{F_1}, \ldots, c_{F_f}\}$ by using a synchronous checkpoint protocol while mobile computers $M_1, \ldots, M_m$ take local checkpoints $c_{M_1}, \ldots, c_{M_m}$ by using an asynchronous one. $F_1, \ldots, F_f$ take a consistent coordinated checkpoint $\tilde{C}$ according to the following protocol:

[Coordinated checkpoint $\tilde{C}$]

1) A *coordinator computer* CS sends a checkpoint request message *Creq* to $F_1, \ldots, F_f$ and $A_1, \ldots, A_a$ through a wired network.
2) On receipt of *Creq*, each F_i takes a tentative local checkpoint tc_{F_i} by storing state information into a volatile storage.
3) Each F_i and A_j sends back a reply message *Crep* to CS.
4) If CS receives all the *Creps*, CS sends a final message *Cfin* to $F_1, \ldots, F_f$ and $A_1, \ldots, A_a$.
5) On receipt of *Cfin*, each F_i makes tc_{F_i} stable, i.e. takes c_{F_i} by storing the state information in step 2) into a stable storage. □

In order to avoid orphan messages, each computer suspends transmission of application messages while the computer has a tentative checkpoint, i.e. from step 2) to step 5).

Next, we discuss how each mobile computer M_i takes a local checkpoint c_{M_i}. Here, suppose that M_i is supported by an access point A_j. A checkpoint agent process p_{ij} in A_j takes a *tentative local checkpoint* tc_{M_i} independently of the other computers. State information required for M_i to restart from tc_{M_i} is carried by a tentative checkpoint request message *TCreq*. On receipt of *TCreq*, p_{ij} stores state information of M_i into a *tentative state log* tsl_{ij} in a volatile storage of A_j.

[Tentative checkpoint tc_{M_i} in p_{ij}]

1) M_i sends *TCreq* to p_{ij}. *TCreq* carries state information of M_i.
2) On receipt of *TCreq*, p_{ij} takes tc_{M_i} by storing the

state information of M_i into tsl_{ij}. If another checkpoint agent process p_{ik} had taken a tentative local checkpoint tc'_{M_i} of M_i, p_{ij} requests p_{ik} to discard tc'_{M_i}. □

Let $\langle p_i^1, \ldots, p_i^c \rangle$ be a sequence of checkpoint agent processes supporting M_i where p_i^1 has tc_{M_i}, i.e. A_i^1 receives *TCreq* from M_i, and p_i^c in an access point A_i^c is a current checkpoint agent process of M_i. If A_i^c receives *Creq* for taking $\tilde{C}$, p_i^1 updates tc_{M_i} to a *stable local checkpoint* c_{M_i} by storing the state information in a tentative state log tsl_i^1 into a *stable state log* sl_i^1. Moreover, each p_i^k $(1 \leq k \leq c)$ stores the messages in a tentative message log tml_i^k into a *stable message log* ml_i^k. The stable logs are stored in a stable storage while the tentative logs are stored in a volatile storage.

[Stable checkpoint c_{M_i} in p_i^k]

- If A_i^1 with tc_{M_i} receives *Creq*, p_i^1 stores the state information in tsl_i^1 into sl_i^1 before sending back *Crep*. $tsl_i^1 = \phi$.
- If A_i^k $(1 \leq k \leq c)$ receives *Creq*, p_i^k stores the messages in tml_i^k into ml_i^k. If there is another stable local checkpoint c'_{M_i} when c_{M_i} is taken, stable logs for c'_{M_i}, i.e. a stable state log sl'^k_i and a stable message log ml'^k_i, are discarded from the stable storage. Here, since *Creq* messages are sent to all access points in the coordinated checkpoint protocol, no control message among the checkpoint agent processes is required. □

According to the discussed message logging and checkpoint protocols, state information and message logs are stored into stable storages of multiple access points according to movement of a mobile computer. These are gathered to the mobile computer in recovery. In [4], the authors have proposed a recovery protocol implementing this distributed logging. However, the protocol requires so many message transmissions to gather the logged information that both wired and wireless networks may be overloaded and network congestion may occur. In order to reduce messages transmitted in recovery, each time a mobile computer M_i moves out of a wireless cell and is disconnected from an access point A_j, a foreign agent process FA_{ij} in A_j sends state information and message log in volatile and stable storages, i.e. tsl_{ij}, tml_{ij}, sl_{ij} and ml_{ij}, to a home agent process HA_i of M_i. Since these information is carried by a registration message for release of a connection in Mobile IP, no additional message is required.

[Disconnection from FA_{ij}]

1) Before moving out of a wireless cell supported by an access point A_j, a foreign agent process FA_{ij} sends a registration message *Mreg* to HA_i according to Mobile IP. If state information and/or message log are stored in volatile and/or stable storages, these are piggied back to *Mreg*.
2) On receipt of *Mreg*, HA_i stores the information into volatile and/or stable storages. If message logs have already been stored, newly received logs are added to the end of the existing logs.
3) HA_i sends back an acknowledgment message *Mack* to FA_{ij} according to Mobile IP.
4) On receipt of *Mack*, FA_{ij} discards the state information and the message log.
5) In case of receipt of *Creq*, the information stored in a volatile storage of HA_i is stored into a stable storage according to a message logging protocol. □

By using this protocol, state information and message logs are stored at most two computers. One is an access point in which HA_i is located and the other is an access point A_j currently supporting M_i and in which FA_{ij} exists. Hence, recovery of M_i is realized by cooperation of HA_i and FA_{ij}. In addition, the functions of a checkpoint agent process p_{ij} are also supported by FA_{ij} since all messages destined to M_i is forwarded to FA_{ij} by using IP tunneling.

The message logging and checkpoint protocols proposed here have the following properties:

[Property 1]
Each M_i has one stable checkpoint c_{M_i} for consistent recovery with $\tilde{C}$ most recently taken by fixed computers. □

[Property 2]
Each M_i has at most one tentative checkpoint tc_{M_i}. □

[Property 3]
Necessary and sufficient messages for achieving a consistent local state of M_i with $\tilde{C}$ are stored into a stable message log in the same order as M_i exchanged. □

6 Restart Protocol

By using the checkpoint protocols in section 5, state information of both fixed and mobile computers are stored into stable storages. For mobile computers to get consistent state with a coordinated checkpoint taken by fixed computers, a message logging protocol is designed in section 4. For avoiding network overload, state information and message logs are gathered to an access point with a home agent process. Now, a restart protocol is designed as follows:

[Restart protocol in F_i]

1) A *coordinator computer* CS sends a restart request message *Rreq* to $F_1, \ldots, F_f$ and $A_1, \ldots, A_a$.
2) On receipt of *Rreq*, each F_i and A_j send back a reply message *Rrep* to CS.
3) After receipt of all the *Rreps*, CS sends a final message *Rfin* to $F_1, \ldots, F_f$ and $A_1, \ldots, A_a$.
4) On receipt of *Rfin*, F_i restarts from c_{F_i}.

[Restart protocol in M_i]

1) On receipt of *Rreq*, FA_{ij} in A_j sends a state information request message *SIreq* to HA_i.
2) On receipt of *SIreq*, HA_i sends back a state information reply message *SIrep* with state information at tc_{M_i} and a message log in a stable storage to FA_{ij}.
3) On receipt of *SIrep*, FA_{ij} sends a restart request mes-

sage *Rreq* with all the needed state information and a message log to M_i.

4) M_i gets a consistent state with $\tilde{C}$ by using the information in *Rreq*. M_i restores state information and replays ordered messages in the log without exchanging messages with other computers. □

7 Evaluation

In this section, we show evaluation of performance of our protocol. The evaluation is achieved from the following two view points: stable storage access overhead and number of messages for recovery.

In the conventional mobile asynchronous checkpoint protocol [7], each time a mobile computer communicates with another computer, a message is stored into a message log in a stable storage to keep the system consistent. However, in the hybrid checkpoint protocol proposed in this paper, only when a checkpoint agent process receives a *Creq* message, a set of state information and message logs are stored into a stable storage. Otherwise, i.e. on receipt of a *TCreq* message or at a communication event, state information and/or messages are stored into a volatile storage with lower access overhead. Figure 7 shows numbers of accesses to a stable storage in these protocols. Here, in the conventional protocol, an average duration between two successive communication events in a mobile computer is assumed 30sec. On the other hand, in the hybrid checkpoint protocol, the number of communication events is the same as in the conventional protocol and a duration between two successive receipt of *Creq* messages is assumed 120sec. According to this evaluation, our proposed protocol achieves fault-tolerance with less stable storage access overhead than the conventional one.

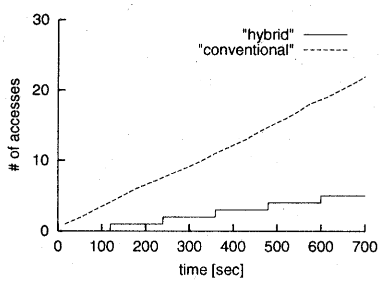

Figure 7: Stable storage access overhead.

Next, we evaluate recovery overhead by comparing with the conventional hybrid checkpoint protocol without supporting mobile IP [4]. Here, suppose that there are 10 mobile computers in a mobile network system. Each mobile computer moves from one wireless cell to another according to the Poisson process whose average time duration between two successive movements is 180sec. In addition, time duration between two successive recovery is also assumed to be Poisson process with an average 600sec. Figure 8 shows the number of messages transmitted for our protocol in the simulation results. Each time a mobile computer moves from a wireless cell to another, registration messages with state information and a message log is exchanged between a foreign agent and a home agent. In recovery, the gathered state information and message logs stored in a home agent is transmitted to the mobile computer. Though continuous network load is observed, only 20 messages are exchanged in recovery. On the other hand in the conventional protocol, no message is exchanged in failure-free execution as shown in Figure 9. However, in recovery, many messages are exchanged between current foreign agent processes and obsolete foreign agent processes to gather distributed state information and message logs to mobile computers. Hence, it is difficult to avoid network overload and congestion. Therefore, our protocol higher performs than the conventional protocol.

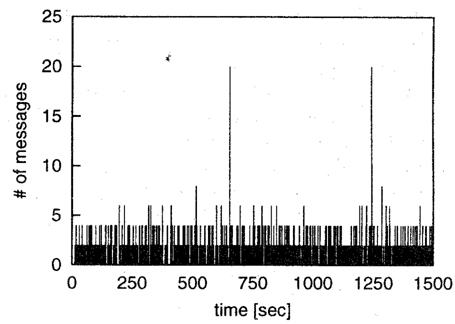

Figure 8: Number of messages in our protocol.

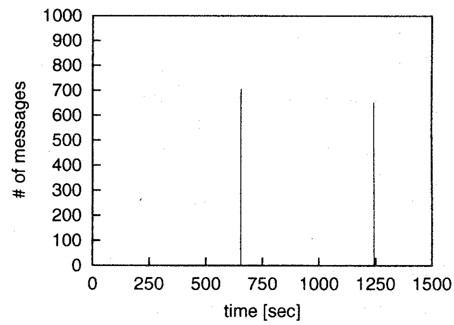

Figure 9: Number of messages in conventional protocol.

Next, performance of the proposed hybrid checkpoint protocol P_p is compared with that of a conventional protocol P_c [4]. Here, average message transmission delay is evaluated. In the both checkpoint protocols, messages exchanged between two mobile computers have to be logged for recovery. Hence, additional delay is required for transmitting control messages and storing message logs into storages due to the hybrid checkpointing. Table 1 shows some assumptions for the simulations.

Table 1: Assumptions for simulation.

wireless LAN protocol	IEEE 802.11b
Number of mobile computers	2∼ 20
average size of messages	5 kbyte
average gap of transmissions	10, 50, 100 msec

Figure 10 shows a simulation result. In P_p, even if the number of mobile computers in a wireless cell is increased, the average transmission delay of a message is almost constant.

On the other hand, in P_c, the average transmission delay of a message is rapidly increased according to the increase of number of mobile computers. Here, for storing message logs into an access point, all the messages are transmitted through the access point. Hence, two-hop message transmission within a wireless cell is required and more contentions and collisions in CSMA/CA occurs. In addition, message transmission and storing of a message log occurs sequentially in P_c and concurrently in P_p. Therefore, shorter message transmission delay is required in P_p than P_c. The longer a gap of two successive message transmission is, the more the difference of message transmission delay between P_p and P_c is.

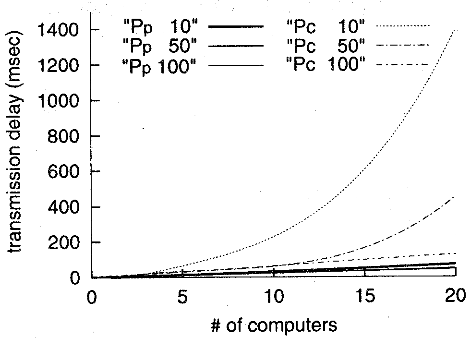

Figure 10: Message transmission delay.

8 Concluding Remarks

It is significant to discuss how to make mobile network systems including mobile and fixed computers more reliable and available. This paper has discussed how the mobile computers and fixed ones take consistent checkpoints and restart from them. We have newly proposed an implementation of hybrid checkpointing for supporting wireless LAN protocols such as IEEE 802.11 and HIPERLAN. Based on the broadcast property of message transmission, access points store messages exchanged by mobile computers for recovery. By separation of message content and order information, the proposed protocol achieves consistent message logs without additional messages. Compared with a conventional protocol, less stable storage access overhead and network load are required, and shorter delay is required for application message transmission.

References

[1] Acharya, A. and Badrinath, B.R., "Checkpointing Distributed Applications on Mobile Computers," The 3rd International Conference on Parallel and Distributed Information Systems, pp. 73–80 (1994).

[2] Chandy, K.M. and Lamport, L., "Distributed Snapshots: Determining Global States of Distributed Systems," ACM Trans. on Computer Systems, Vol. 3, No. 1, pp. 63–75 (1985).

[3] Elnozahy, E.N., Johnson, D.B. and Wang, Y.M., "A Survey of Rollback-Recovery Protocols in Message-Passing Systems," Technical Note of Carnegie Mellon University, CMU-CS-96-181 (1996).

[4] Higaki, H. and Takizawa, M., "Checkpoint-Recovery Protocol for Reliable Mobile Systems," The 17th International Symposium on Reliable Distributed Systems, pp. 93–99 (1998).

[5] Koo, R. and Toueg, S., "Checkpointing and Rollback-Recovery for Distributed Systems," IEEE Trans. on Software Engineering, Vol. SE–13, No. 1, pp. 23–31 (1987).

[6] Muller, G., Hue, M. and Peyrouz, N., "Performance of consistent checkpointing in a modular operating system: Result of the FTM experiment," Lecture Notes in Computer Science: Dependable Computing -EDCC-1, pp. 357–365 (1994).

[7] Neves, N. and Fuchs, W.K., "Adaptive Recovery for Mobile Environments," Communications of the ACM, Vol. 40, No. 1, pp. 69–74 (1997).

[8] Perkins, C., "IP Mobility Support," RFC 2002 (1996).

[9] Pradhan, D.K., Krishna, P.P. and Vaidya, N.H., "Recovery in Mobile Wireless Environment: Design and Trade-off Analysis," The 26th International Symposium on Fault-Tolerant Computing Systems, pp. 16–25 (1996).

[10] Prakash, R. and Singhal, M., "Low-Cost Checkpointing and Failure Recovery in Mobile Computing Systems," IEEE Trans. on Parallel and Distributed Systems, Vol. 7, No. 10, pp. 1035–1048 (1996).

[11] Smith, S.W., Johnson, D.B and Tygar, J.D., "Completely asynchronous optimistic recovery with minimal rollbacks," The 25th International Symposium on Fault-Tolerant Computing, pp. 361–370 (1995).

[12] Smith, S.W. and Johnson, D.B., "Minimizing timestamp size for completely asynchronous optimistic recovery with minimal rollback," The 15th International Symposium on Reliable Distributed Systems, pp. 66–75 (1996).

[13] Wang, Y.M., "The maximum and minimum consistent global checkpoints and their applications," The 14th International Symposium on Reliable Distributed Systems, pp. 86–95 (1995).

[14] "Radio Equipment and Systems(RES); HIPERLAN," ETSI Functional Specifications (1995).

[15] "Wireless LAN Medium Access Control(MAC) and Physical Layer(PHY) Specifications," Standard IEEE 802.11 (1997).