

# Hybrid Checkpoint Protocol in Wireless LAN Environment

Yoshinori Morita and Hiroaki Higaki  
Department of Computers and Systems Engineering  
Tokyo Denki University  
E-mail: {mine,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 piggybacked 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 桧垣 博章  
E-mail: {mine,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、リカバリ時に輻濫が発生することを防止している。

### 1 Introduction

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. An access point is a fixed computer with a wireless network interface. Fixed computers including access points are interconnected by a 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 [23] and HIPERLAN [22].

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 c-

omputers, 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 [2-8, 11, 13, 14, 18, 19, 21] 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* [5]. Fixed computers take consistent checkpoints by using *synchronous checkpoint protocols* [5, 7, 18] with low synchronization overhead since they communicate with each other through a wired network and they have enough amount of stable storages to store state information [6, 10]. Some papers [2-4, 11, 14] 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 [11]. In a protocol

in [11], 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 [13, 19] 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 overhead 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 [8]. 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 [8], every message from a mobile computer is assumed to be transmitted through an access point supporting the mobile computer. 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  $\mathcal{V} = \{s_1, \dots, s_n\}$  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 help of protocols in underlying layers. 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  $m$ .

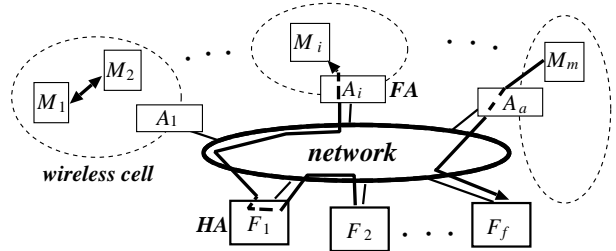


Figure 1: Mobile network system.

In a mobile network system, which is a kind of  $\mathcal{S}$ , there are three kinds of computers; *fixed computers*  $F_1, \dots, F_f$ , *mobile computers*  $M_1, \dots, M_m$  and *access points*  $A_1, \dots, A_a$  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 [23]. Here, each  $M_i$  is included in a single wireless cell. A message exchanged between  $M_i$  and  $M_k$  in a wireless cell is transmitted directly by using CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) protocol. 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 a 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 between  $M_i$  and the access point by using a broadcast-based wireless LAN protocol.

## 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 [8].

### [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. Here,  $tc_{M_i}$  is referred to as a *tentative local checkpoint*.  $M_i$  also stores messages sent and received after  $c_{M_i}$  into a message log  $log_{M_i}$  to achieve a consistent local state  $vc_{M_i}$  with a coordinated checkpoint  $\tilde{C}$ . Thus local state is referred to as a *virtual local checkpoint*.  $\square$

### [Recovery]

- Each fixed computer  $F_i$  restores from a local checkpoint  $c_{F_i}$  of the most recent coordinated checkpoint  $\tilde{C}$ .
- Each mobile computer  $M_i$  restores the state of a

tentative local checkpoint  $tc_{M_i}$  and replays events according to a message log  $log_{M_i}$  to get state of a virtual local checkpoint  $vc_{M_i}$ . Then,  $M_i$  restarts from  $vc_{M_i}$ .  $\square$

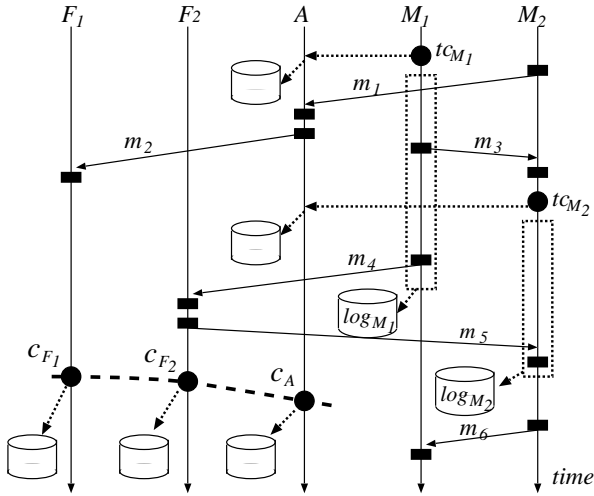


Figure 2: Checkpointing in hybrid protocol.

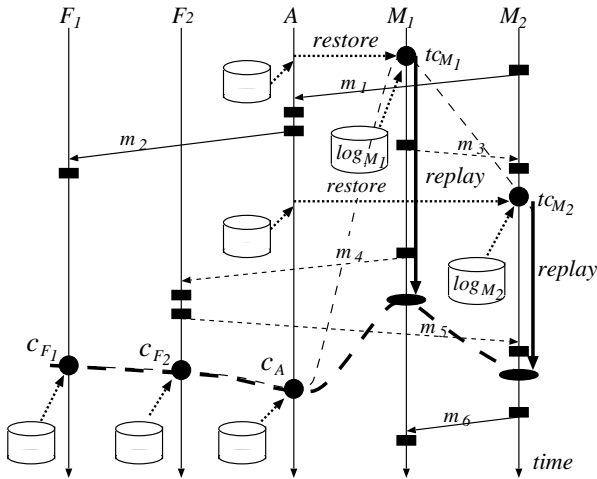


Figure 3: Restart in hybrid protocol.

At  $tc_{M_i}$ , 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  $tc_{M_i}$  if  $M_i$  moves out of the wireless cell or battery power in  $M_i$  is exhausted. Thus,  $M_i$  takes  $tc_{M_i}$  only if  $M_i$  does not move out of a wireless cell and has enough battery power for taking  $tc_{M_i}$ . Hence,  $M_i$  asynchronously takes  $tc_{M_i}$ , i.e. independently of the other computers.  $M_i$  has to restart from a local state consistent with  $\tilde{C}$ . However,  $tc_{M_i}$  is not always consistent with  $\tilde{C}$  since  $M_i$  takes  $tc_{M_i}$  independently of the fixed computers taking  $\tilde{C}$ . Hence, a kind of log-based restart protocols [1, 9, 15–17, 19–21] is designed as shown in Figure 2. Messages exchanged between  $M_i$  and other computers after taking  $tc_{M_i}$  are stored into a message log  $log_{M_i}$  in a stable storage in  $A_j$ . In recovery,  $M_i$  restores the state information at  $tc_{M_i}$

and the logged messages in  $log_{M_i}$  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 really 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  $tc_{M_i}$  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 [8], 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  $M_i$ . In this communication model, a checkpoint agent process  $p_{ij}$  for  $M_i$  in  $A_j$  stores all the messages sent and received 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 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 virtual local checkpoint for recovery. Here, wireless LAN protocols are broadcast-based protocols. A message sent by a mobile computer and an access point is received by all the mobile computers and the access point in a 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 a wireless cell. Thus, a checkpoint agent process  $p_{ij}$  for  $M_i$  in  $A_j$  stores sufficient state information and messages to achieve a virtual local checkpoint  $vc_{M_i}$  consistent 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 communication events observed by  $A_j$  might be different from that really occurred in  $M_i$ .

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

- A message  $m$  sent or received by  $M_i$  is stored into an unordered message buffer  $m_{buf_{ij}}$  in an access point  $A_j$  temporarily. Even if  $m$  is transmitted between  $M_i$  and another mobile computer in a wireless cell,  $A_j$  receives  $m$  due to broadcast property of a wireless LAN protocol.
- Order information of communication events in  $M_i$  is piggybacked to another message  $m'$  sent by  $M_i$ . Even if  $m'$  is destined to another mobile computer within a wireless cell,  $A_j$  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 in order 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 received by  $M_i$ . Initially,  $Rseq_i = \langle \rangle$  (empty).

[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 piggybacked back to  $m$  as  $m.Rseq_i$  and  $m$  is transmitted.  $Rseq_i = \langle \langle \rangle \rangle$ .

#### [Message logging in $p_{ij}$ ]

- On receipt of a message  $m$  destined to  $M_i$ ,  $p_{ij}$  stores a copy of  $m$  into  $mbuf_{ij}$ .  $p_{ij}$  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  $m.Rseq_i$  which is a sequence of message identifiers piggybacked back to  $m$ , out of  $mbuf_{ij}$ . If all the required messages have not yet stored into  $mbuf_{ij}$ , this procedure is suspended. If  $p_{ij}$  receives a message destined to  $M_i$ , this procedure is resumed after storing a copy of the message into  $mbuf_{ij}$ .
  - 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  $m.Rseq_i$ .
  - 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.  $\square$

Suppose that  $M_i$  moves from a wireless cell supported by an access point  $A_i^k$  to another one supported by  $A_i^{k+1}$  ( $1, 2, \dots, c-1$ ) and  $M_i$  is currently supported by  $A_i^c$ . In each  $A_i^k$ , there exists a checkpoint agent process  $p_i^k$  of  $M_i$ . Each  $p_i^k$  stores messages sent and received by  $M_i$  to a *tentative message log*  $tml_i^k$  in a volatile storage of  $A_i^k$  while  $M_i$  is supported by  $A_i^k$ . Hence, messages for which events are required to be replayed in recovery are stored in distributed manner, i.e. in a sequence of tentative message logs  $\langle tml_i^1, \dots, tml_i^c \rangle$ .

## 5 Checkpoint protocol

Fixed computers  $F_1, \dots, F_f$  take a consistent coordinated checkpoint  $\tilde{C}$  by using the following protocol:

#### [Coordinated checkpoint $\tilde{C}$ ]

- 1) A *coordinator computer*  $CS$ , which might be one of the fixed computers, sends a checkpoint request message  $Creq$  to  $F_1, \dots, F_f$  and  $A_1, \dots, A_a$  through a wired network.
- 2) On receipt of  $Creq$ , each  $F_i$  takes a tentative local checkpoint  $tc_{F_i}$  by storing the current 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  $F_1, \dots, F_f$  and  $A_1, \dots, A_a$ .
- 5) On receipt of  $Cfin$ , each  $F_i$  takes  $tc_{F_i}$  by using  $tc_{F_i}$  stable. Here,  $F_i$  stores the state information at  $tc_{F_i}$  in step 2) into a stable storage.  $\square$

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

2) and step 5).

Next, we discuss how each mobile computer  $M_i$  takes a local checkpoint. 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 the 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 the current state information of  $M_i$ .
- 2) On receipt of  $TCreq$ ,  $p_{ij}$  stores the state information of  $M_i$  carried by  $TCreq$  into  $tsl_{ij}$ .  $\square$

Let  $\langle p_i^1, \dots, p_i^c \rangle$  be a sequence of checkpoint agent processes supporting  $M_i$  where  $p_i^1$  in an access point  $A_i^1$  stores state information at the most recent tentative checkpoint  $tc_{M_i}$ , i.e.  $A_i^1$  receives the most recent  $TCreq$  from  $M_i$ , and  $p_i^c$  in  $A_i^c$  is a current checkpoint agent process of  $M_i$ . When fixed computers take a coordinated checkpoint  $\tilde{C}$ , a checkpoint request message  $Creq$  is received every access point. On receipt of  $Creq$  in  $A_i^1$ ,  $p_i^1$  stores the state information at  $tc_{M_i}$  in a tentative state log  $tsl_i^1$  into a *stable state log*  $sl_i^1$ . In addition, on receipt of  $Creq$  in  $A_i^k$  ( $k = 1, 2, \dots, c$ ), each  $p_i^k$  stores the messages in a tentative message log  $tml_i^k$  into a *stable message log*  $ml_i^k$ . The stable logs  $sl_i^k$  and  $ml_i^k$  are in a stable storage while the tentative logs  $tsl_i^k$  and  $tml_i^k$  are in a volatile storage.

#### [Virtual checkpoint $vc_{M_i}$ in $p_i^k$ ]

- If  $A_i^1$ , an access point at  $tc_{M_i}$ , receives  $Creq$ ,  $p_i^1$  stores the state information in  $tsl_i^1$  into  $sl_i^1$  before sending back  $Crep$ . Then,  $tsl_i^1$  is cleared.
- If  $A_i^k$  ( $k = 1, 2, \dots, c$ ) receives  $Creq$ ,  $p_i^k$  stores the messages in  $tml_i^k$  into  $ml_i^k$ .

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 [8], 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  $Mreq$  to  $HA_i$  according to Mobile IP. If state information and message log are stored in volatile and stable storages, these are piggybacked to  $Mreq$ .
- 2) On receipt of  $Mreq$ ,  $HA_i$  stores the information into volatile and 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 the message logging protocol.  $\square$

By using this protocol, 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 in Mobile IP.

## 6 Restart Protocol

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  $F_1, \dots, F_f$  and  $A_1, \dots, A_a$ .
- 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  $F_1, \dots, F_f$  and  $A_1, \dots, A_a$ .
- 4) On receipt of  $Rfin$ ,  $F_i$  restarts from  $c_{F_i}$ .

### [Restart protocol in $M_i$ ]

- 1) On receipt of  $Rreq$  in  $A_j$ ,  $FA_{ij}$  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 between  $tc_{M_i}$  and  $vc_{M_i}$  in a stable storage to  $FA_{ij}$ .
- 3) On receipt of  $SIrep$ ,  $FA_{ij}$  sends a restart request message  $Rreq$  with the state information and a message log carried by  $SIrep$  to  $M_i$ .
- 4)  $M_i$  gets the state at  $vc_{M_i}$  consistent with  $\tilde{C}$  by using the information in  $Rreq$ .  $M_i$  restores state information and replays ordered communication events according to the message log without really exchanging messages with other computers.  $\square$

## 7 Evaluation

In this section, we show evaluation of performance of our protocol. The evaluation is achieved in the following two points: stable storage access overhead and communication overhead.

In the conventional mobile asynchronous checkpoint protocol [11], each time a mobile computer communicates with another computer, a message is stored into a message log in a stable storage to keep the sys-

tem 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 messages are stored into a volatile storage with lower access overhead. Figure 4 shows numbers of accesses to a stable storage in these protocols. Here, in both protocols, message sending events are assumed to be occurred in accordance with an average duration between two successive message sending events is 300msec. In addition, in the hybrid checkpoint protocol, an access point receives a  $Creq$  message for a coordinated checkpoint in accordance with Poisson process where average durations between two successive receipts of a  $Creq$  message are 300msec, 500msec, 1000msec, 1500msec. As shown in the evaluation result, our proposed protocol achieves fault-tolerance with less stable storage access overhead than the conventional one.

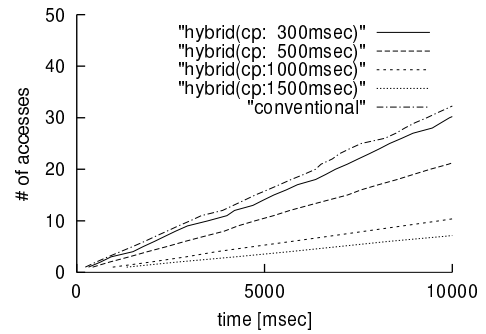


Figure 4: Stable storage access overhead.

Next, we evaluate communication overhead by comparing with the conventional hybrid checkpoint protocol with the distributed logging, i.e. without supporting mobile IP [8]. Here, suppose that there are 20 mobile computers. Switching between wireless cells occurs in accordance with Poisson process where an average duration between two successive switches is 180sec. In addition, recovery from a failure is also in Poisson process with 600sec average duration. Figure 5 shows the number of messages. In our protocol, 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 40 messages are exchanged in recovery. On the other hand in the conventional protocol, no message is exchanged in failure-free execution. 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.

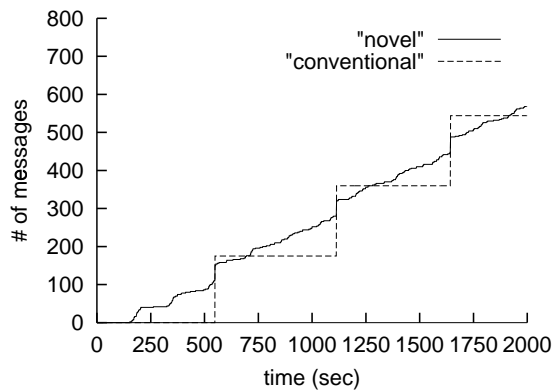


Figure 5: Number of messages in recovery.

## 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 communication overhead are required.

## References

- [1] Alvisi, L. and Marzullo, K., "Message logging: pessimistic, optimistic, causal and optimal," *IEEE Trans. on Software Engineering*, Vol. 24, No. 2, pp. 149-159 (1998).
- [2] Cao, G. and Singhal, M., "On the impossibility of min-process non-blocking checkpointing and efficient checkpointing algorithm for mobile computing systems," *The 12th International Conference on Parallel Processing*, pp. 37-44 (1998).
- [3] Cao, G. and Singhal, M., "Low-cost checkpointing with mutable checkpoints in mobile computing systems," *The 18th International Conference on Distributed Computing Systems*, pp. 464-471 (1998).
- [4] Cao, G. and Singhal, M., "Mutable checkpoints: a new checkpointing approach for mobile computing systems," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 12, pp. 157-172 (2001).
- [5] 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).
- [6] Elnozahy, E.N., Alvisi, Y.M., Wang, Y.M. and Johnson, D.B., "A Survey of Rollback-Recovery Protocols in Message-Passing Systems," *Technical Report of Carnegie Mellon University, CMU-CS-99-148* (1999).
- [7] Gendelman, E., Bic, L.F. and Dillencourt, M.B., "An efficient checkpointing algorithm for distributed systems implementing reliable communication channels," *The 18th International Symposium on Reliable Distributed Systems*, pp. 290-291 (1999).
- [8] Higaki, H. and Takizawa, M., "Checkpoint-Recovery Protocol for Reliable Mobile Systems," *The 17th International Symposium on Reliable Distributed Systems*, pp. 93-99 (1998).
- [9] Lee, B., Park, T., Yeom, H.Y. and Cho, Y., "An efficient algorithm for causal message logging," *The 17th International Symposium on Reliable Distributed Systems*, pp. 19-25 (1998).
- [10] 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).
- [11] Neves, N. and Fuchs, W.K., "Adaptive Recovery for Mobile Environments," *Communications of the ACM*, Vol. 40, No. 1, pp. 69-74 (1997).
- [12] Perkins, C., "IP Mobility Support," *RFC 2002* (1996).
- [13] 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).
- [14] 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).
- [15] Rao, S., Alvisi, L. and Vin, H.M., "The cost of recovery in message logging protocols," *The 17th International Symposium on Reliable Distributed Systems*, pp. 10-18 (1998).
- [16] 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).
- [17] Ssu, K.F., Bin Yao, B. and Fuchs, W.K., "An adaptive checkpointing protocol to bound recovery time with message logging," *The 18th International Symposium on Reliable Distributed Systems*, pp. 244-252 (1999).
- [18] Taesoon, P. and Yeom, H.Y., "Communication pattern based checkpointing coordination for fault-tolerant distributed computing systems," *The 12th International Conference on Information Networking*, pp. 559-562 (1998).
- [19] Taesoon, P. and Yeom, H.Y., "An asynchronous recovery scheme based on optimistic message logging for mobile computing systems," *The 20th International Conference on Distributed Computing Systems*, pp. 436-443 (2000).
- [20] Yao, B. and Fuchs, W.K., "Message logging optimization for wireless networks," *The 20th International Symposium on Reliable Distributed Systems*, pp. 182-185 (2001).
- [21] Zambonelli, F. and Netzer, R.H.B., "An efficient logging algorithm for incremental replay of message-passing applications," *The 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing*, pp. 392-398 (1999).
- [22] "Radio Equipment and Systems (RES); HIPERLAN," *ETSI Functional Specifications* (1995).
- [23] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *Standard IEEE 802.11* (1997).