Hybrid Checkpoint Protocol with Unreliable Channels

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

For supporting mission-critical applications in a mobile network system, hybrid checkpointing has been proposed. In order to apply hybrid checkpointing to recent wireless LAN networks, we classified wireless LAN protocols into four types according to the communication model. Conventional hybrid checkpoint protocol supports centralized protocols as Bluetooth and cell-dependent infrastructured protocols as IEEE802.11 with an assumption of reliable broadcast. However, in CSMA/CA based protocols including IEEE802.11, messages may be lost due to noisy wireless environment and a hidden terminal problem. In this paper, we propose a newly designed hybrid checkpoint protocol by which lost broadcast messages are detected by pigging back IDs of causally preceding messages and retransmitted according to requests from an access point.

低信頼な無線通信を含むモバイルネットワークのための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
宮崎 正光　森田 義徳　桧垣 博章
E-mail: {miya,mine,hig}@higlab.k.dendai.ac.jp

移動コンピュータ環境において、耐故障性を高めるための技術として複合チェックポイントプロトコルが提案されている。また、無線 LAN プロトコルをサポートするために、我々は、無線 LAN プロトコルを 4 つに分類した。従来の複合チェックポイントプロトコルは、Bluetooth のように移動コンピュータ間の通信も必ず基地局を経由するプロトコルと、信頼性のあるブロードキャストを想定した IEEE802.11 のような、セル依存直接通信プロトコルをサポートするように設計されている。しかし、IEEE802.11 のような、CSMA/CA に基づくプロトコルでは、雑音のある無線環境や、隠れ端末問題により、メッセージが紛失することがある。本論文では、送信メッセージに、因果先行するメッセージの ID を付加することで、基地局が紛失したブロードキャストメッセージを検出し、紛失したメッセージの再送信を行なう、新しい複合チェックポイントプロトコルを提案する。

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. In addition, applications based on cooperation of multiple autonomous robots are getting developed, and intelligent transport systems (ITSs) with mobile communication are also being implemented.

A mobile network system is composed of *fixed computers* and *mobile computers* interconnected by a communication network. A fixed computer is located at a fixed location and communicates through a wired network like Ethernet. A mobile computer moves from one location to another and communicates through a wireless communication channel with other mobile computers within a transmission range.

This is realized by using wireless communication protocols such as Bluetooth [1] and wireless LAN protocols, e.g. IEEE802.11 [2] and HIPERLAN [3]. An *access points* supports communication of mobile computers. It is a fixed computer connected not only to a wired network to communicate with fixed computers and other access points but also to a wireless networks to communicate with mobile computers.

In a network system, applications are realized by cooperation of multiple computers. Usually, a network system is composed of widely available products including personal computers, mobile computers, engineering workstations, Ethernets, routers, repeaters, switches and so on. Hence, a mission-critical application is not always realized in such a system. Checkpoint-recovery [5, 9, 15] is one of the well-known methods for achieving reliable and available network systems. Each computer v_i takes a local checkpoint c_i where local state information of v_i is stored into a stable storage. If a certain computer fails and recovers, v_i restarts from c_i. A global checkpoint, which is a set of local checkpoints, is required to denote a *consistent global state* [5].

Fixed computers take consistent checkpoints by using *synchronous checkpoint protocols* [5, 12] with low synchronization overhead by communication through a wired network [7, 14]. However, it requires high communication and synchronization overhead to take checkpoints synchronously in a mobile computing system due to mobility and lack of battery capacity of mobile computers. Moreover, it is difficult for a mobile computer to store state information into its unstable disk storage whose volume is limited [15]. In order to solve this problem, the authors have proposed *hybrid checkpointing* where local checkpoints are asynchronously taken by mobile computers while synchronously taken by fixed computers [9]. Mobile computers take local checkpoints by storing state information into stable storages in fixed computers and access points. In addition, in order to restart a mobile computer from a consistent state with other computers, not only the state information but also messages sent and received by the mobile computer and communication events for the messages are required to be logged.

In a hybrid checkpoint protocol in [9], every message from a mobile computer included in a transmission range of an access point is assumed to be forwarded by the access point. Hence, the access point stores the message into a message log. This protocol is designed for such a centralized wireless communication protocol as Bluetooth [1]. In another hybrid checkpoint protocol in [13], a message between mobile computers included in a transmission range of an access point is directly transmitted without help of the access point. According to the broadcast property of wireless communication, every message is also received and stored into a message log by an access point. This protocol is designed for such a cell-dependent wireless communication protocol as IEEE802.11 [2]. In this protocol, reliable message transmission is assumed. Though messages may be lost since wireless communication channels are unreliable and the hidden terminal problem [8, 11] occurs, by using acknowledgment messages and retransmission timers, reliable message transmission between wireless computers is achieved. However, it is not certain for an access point to receive the messages transmitted between the mobile computers. This paper proposes a novel hybrid checkpoint protocol for cell-dependent wireless networks with unreliable communication environments.

2 Hybrid Checkpointing

2.1 Conventional Checkpointing

A network system $\mathcal{N} = \langle \mathcal{V}, \mathcal{L} \rangle$ is composed of a set $\mathcal{V} = \{v_1, \ldots, v_n\}$ 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. $\langle v_i, v_j \rangle \in \mathcal{L}$ is a communication channel from a computer v_i to another computer v_j. A *state* of v_i is updated at each *event* in v_i. There are two kinds of events; *local events* and *communication events*. At a local event, v_i updates its state by local computation without exchanging a message. At a communication event, v_i communicates with another computer by exchanging a message and updates its 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. Among events in a network system, *happen before* relation is defined [5].

[Happen before relation]
An event e_i happens before another event e_j, which is denoted by $e_i \to e_j$, iff one of the following conditions is satisfied:

- e_i occurs before e_j in a computer.
- e_i and e_j are $s(m)$ and $r(m)$, respectively, for a message m.
- For a certain event e_k, $e_i \to e_k$ and $e_k \to e_j$. □

If e_i happens before e_j, e_j *causally depends on* e_i.

For checkpointing in a network system $\mathcal{N}$, it is impossible to store the state information for recovery of a whole system in a centralized manner due to unpredictable message transmission delay in communication channels. Hence, each computer $v_i \in \mathcal{V}$ takes a *local checkpoint* c_i by storing state information of v_i into a stable storage. A *global checkpoint* $C_{\mathcal{V}}$ is a set of local checkpoints taken by all the computers in $\mathcal{V}$, i.e. $C_{\mathcal{V}} = \{c_1, \ldots, c_n\}$. A global checkpoint denotes a global state of the system. Hence, in order to restart $\mathcal{N}$ from a global state denoted by $C_{\mathcal{V}}$, each computer $v_i \in \mathcal{V}$ restarts from c_i.

If a computer v_i takes a local checkpoint c_i and restarts execution of an application from c_i independently of the other computers in $\mathcal{V}$, there may exist two kinds of *inconsistent messages*; *lost messages* and *orphan messages*. Here, suppose that a message m is transmitted through a communication channel $\langle v_i, v_j \rangle$ and computers v_i and v_j take local checkpoints c_i and c_j, respectively. Let $C_{\{v_i,v_j\}} = \{c_i, c_j\}$ be a set of local checkpoints. m is a lost message for $C_{\{v_i,v_j\}}$ iff $s(m)$ occurs before taking c_i in v_i and $r(m)$ occurs after taking c_j in v_j. On the other hand, m is an orphan message for $C_{\{v_i,v_j\}}$ iff $s(m)$ occurs after taking c_i in v_i and $r(m)$ occurs before taking c_j in v_j, that is $c_i \to c_j$. A global checkpoint $C_{\mathcal{V}}$ is defined to be *consistent* iff there is neither lost nor orphan message in any communication channel in $\mathcal{L}$ [5]. If there exists an orphan message m_o in a communication channel $\langle v_i, v_j \rangle \in \mathcal{L}$ for $C_{\{v_i, v_j\}}$, execution of an application in $\mathcal{N}$ is restarted incorrectly. Though m_o has been already received by v_j, v_i might not send m_o after recovery due to non-deterministic property of execution of an application in v_i. On the other hand, if there exists a lost message m_l in a communication channel $\langle v_i, v_j \rangle \in \mathcal{L}$ for $C_{\{v_i, v_j\}}$, execution of an application in $\mathcal{N}$ is also restarted incorrectly. Though m_l has not yet been received by v_j, v_i has finished $s(m_l)$ and does not send m_l any more after recovery. However, by storing the state information of $\langle v_i, v_j \rangle$, i.e. by using a message log, m_l is received by v_j after recovery.

Conventionally, two kinds of protocols for taking consistent global checkpoints in $\mathcal{N}$ have been proposed; *asynchronous* and *synchronous* checkpoint protocols. In asynchronous checkpoint protocols [10, 16], each computer takes local checkpoints independently of the other computers. If a certain computer fails and recovers, the computers cooperate to find a consistent global checkpoint for recovery, i.e. a set of local checkpoints taken independently for which there is no inconsistent message in any communication channel. An asynchronous checkpoint protocol implies less communication and synchronization overhead for taking checkpoints because of no communication among the computers. However, it takes longer time for the computers to restart since the computers have to exchange messages carrying information of local checkpoints for detecting a consistent global checkpoint, i.e. a set of local checkpoints to denote a consistent global state. Moreover, if there is no consistent global checkpoint, the computers have to restart execution of an application from the initial state. It is called *domino effect* [18].

On the other hand, in synchronous checkpoint protocols [5, 6, 12, 17, 19] multiple computers cooperate to take a consistent global checkpoint. In synchronous checkpoint protocols, each computer always restarts execution of an application from the most recent local checkpoint since the checkpoints are surely consistent. Thus, no domino effect occurs. Though communication and synchronization overhead for taking consistent global checkpoints is higher, it is acceptable in recent high-speed wired networks [6, 7, 14].

2.2 Hybrid Checkpointing

A mobile network system $\mathcal{MN} = \langle \mathcal{MV}, \mathcal{ML} \rangle$, which is a kind of $\mathcal{N}$, consists of the following three kinds of computers; *fixed computers* $F_1, \ldots, F_f$, *mobile computers* $M_1, \ldots, M_m$ and *access points* $A_1, \ldots, A_a$. F_i is connected at a fixed location in the network and communicates with other computers through a high-speed wired network. In addition, F_i has enough resources such as processing power, disk storage to achieve stable storage for storing the state information at local checkpoints.

On the other hand, power supply in M_i is restricted since M_i has only limited battery capacity. Computation resources in M_i is also limited. Processing power in M_i is lower than that in F_i and M_i does not have stable storage to store the state information at local checkpoints since M_i does not have enough disk storage capacity and the storage is unstable due to the movement of M_i. M_i moves from one location to another. M_i communicates with another mobile computer or an access point in a transmission range by using a wireless communication protocol, e.g. Bluetooth [1] and wireless LAN protocols such as IEEE802.11 [2] and HIPERLAN [3] based on CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance).

A_i is connected at a fixed location in the network. If A_i communicates with a fixed computer or another access point, it communicates through a high-speed wired network. A_i also communicates with mobile computers in a transmission range by using a wireless communication protocol.

A wireless communication media is intrinsically unreliable and its bandwidth is lower than that of a wired communication media. For reliable transmission of a message m from a computer v_i (a mobile computer or an access point) to another computer v_j, an acknowledgment message a for m is transmitted from v_j to v_i on receipt of m. If a retransmission timer in v_i is expired without receiving a, v_i retransmits m. In addition, since a wireless communication media is broadcast-base, if a computer v_i sends a message m to another computer v_j, all computers in the transmission range of v_i receives m.

As discussed in the previous subsection, synchronous checkpoint protocols have an advantage that computers restart from the most recent local checkpoints without domino effect in recovery. In a high-speed wired network, required communication overhead to take cooperated checkpoints is acceptable. However, it is difficult for multiple mobile computers to take local checkpoints synchronously since synchronization and communication overhead is high due to lower bandwidth and reliability in wireless communication channels and mobility of mobile computers.

Hence, the authors have proposed *hybrid checkpointing* as shown in Figures 1 and 2 [9]. Here, a synchronous checkpoint protocol and an asynchronous one is combined based on the properties of fixed computers and mobile ones.

[Hybrid checkpointing]

- Each fixed computer F_i takes a local checkpoint c_{F_i} by using a synchronous checkpoint protocol. A

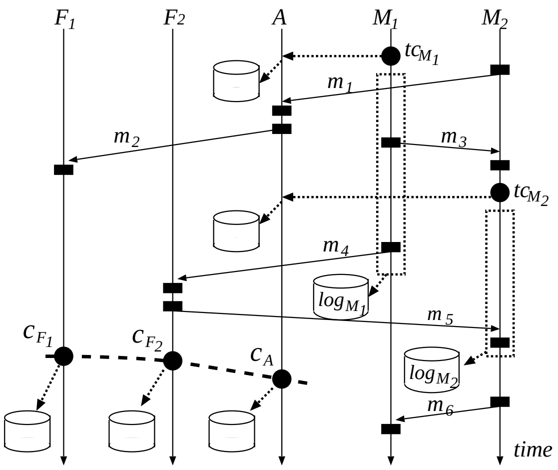

Figure 1: Checkpoint in hybrid protocol.

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. □

At a local checkpoint c_{M_i} in a mobile computer M_i, state information of M_i is stored into a stable storage. Since the disk storage in M_i has only limited capacity and is unstable, the state information of M_i is stored into a stable storage in a fixed computer F_l or an access point A_k. M_i fails to take c_{M_i} if M_i moves out of transmission range of any access point and the state information is not transmitted to any fixed computer and access point. In addition, if battery power in M_i is exhausted, it is also impossible for M_i to take c_{M_i}. Thus, M_i takes c_{M_i} only if M_i communicates with F_l or A_k 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 execution of an application 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. Hence, a kind of log-based restart protocols [4, 16, 20–22] is applied as shown in Figure 2. Messages transmitted between M_i and other computers after taking c_{M_i} are stored into a stable storage. In recovery, M_i restores the state information at c_{M_i} and the logged messages from the stable storage. From the state at c_{M_i}, M_i replays a sequence of events for the logged messages and gets a state consistent with $\tilde{C}$. During the replay, M_i does not exchange messages with other computers.

2.3 Wireless Communication Model

A mobile network system consists of mobile computers, fixed computers and access points. According to

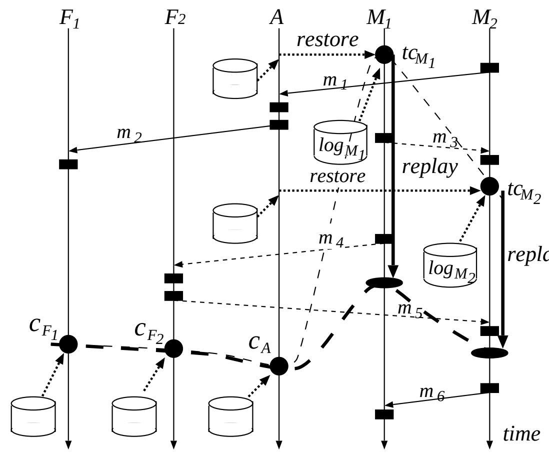

Figure 2: Restart in hybrid protocol.

restrictions for communication of a mobile computer, there are the following four communication models; a *centralized communication model*, a *cell-dependent infrastructured communication model*, a *cell-independent infrastructured communication model* and an *ad-hoc communication model.* Here, a *wireless cell* of an access point A_k is a transmission range of A_k.

In a centralized communication model, all messages transmitted from a mobile computer in a wireless cell of an access point are forwarded by the access point. Even if two mobile computers M_i and M_j in a wireless cell of an access point A_k are in a transmission range of each other, a message m from M_i to M_j is transmitted to A_k then forwarded by A_k. Bluetooth [1] is a wireless communication protocol based on this model.

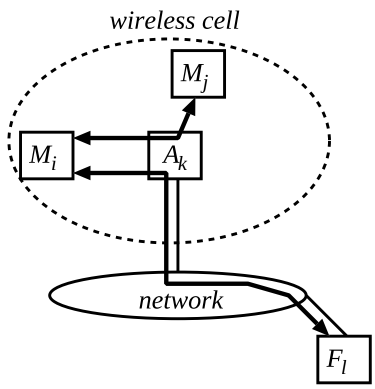

Figure 3: Centralized communication model.

In a cell-dependent infrastructured communication model, a wireless network system is decomposed into multiple wireless cells each of which is supported by an access point. If a mobile computer M_i in a wireless cell of an access point A_k sends a message m to another mobile computer M_j in the same wireless cell, m is directly transmitted to M_j. On the other hand, if M_j is out of the wireless cell, m is forwarded by A_k. m is transmitted to M_j through a high-speed wired network. In addition, if M_i sends m to a fixed computer F_l, m is also forwarded by A_k. IEEE802.11 [2], which is currently the most widely available wireless LAN protocol, is based on this model.

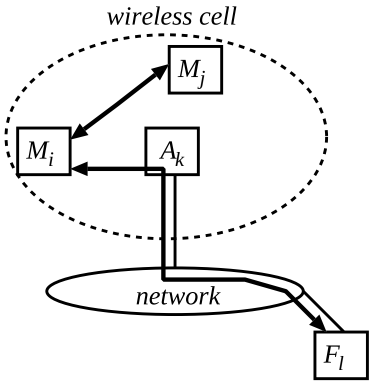

Figure 4: Cell-dependent infrastructured communication model.

In a cell-independent infrastructured communication model, two mobile computers in a transmission range communicate directly and independently of wireless cells. If mobile computers M_i and M_j are in a transmission range of each other, a message m from M_i to M_j is directly transmitted without help of an access point. If M_i and M_j are impossible to communicate directly and M_i is in a wireless cell of an access point A_k, m is forwarded by A_k. In addition, if M_i sends m to a fixed computer F_l, m is also forwarded by A_k. HIPERLAN [3] is based on this model.

In an ad-hoc communication model, there is neither fixed computer nor access point. A mobile network system consists of only mobile computers. If mobile computers M_i and M_j are in a transmission range of each other, a message m from M_i to M_j is directly transmitted. On the other hand, if M_i and M_j are impossible to communicate directly, m is transmitted with the help of other mobile computers. That is, all mobile computers work as routers.

In order to store state information and a message log of a mobile computer into a stable storage of an access point or a fixed computer, the mobile computer is required to communicate with an access point whenever it communicates with other computers. Hence, hybrid checkpointing is applicable to network systems based on the centralized communication model [9] and the cell-dependent infrastructured communication model [13]. In [13], reliable message transmission is assumed. For logging a message m transmitted from a mobile computer M_i to another one M_j in a wireless cell of an access point A_k, A_k receives m which is broadcasted in the wireless cell. For storing the logged messages into a stable storage by A_k in the same order as that M_j has received, order information of receipt events in M_j is pigged back to another message later transmitted from M_j. For reliable transmission of m between M_i and A_k, an acknowledgment message transmission and retransmission timer is required, i.e. M_i waits for acknowledgment messages from M_j and A_k. Hence, modification of a wireless communication protocol is required and many MAC messages are exchanged.

3 Protocol

3.1 Overview

Same as the conventional hybrid checkpoint protocol in [13], a message m exchanged between mobile computers M_i and M_j and the order information of communication events for m are separately transmitted to an access point A_k. m is received by A_k when m is exchanged between M_i and M_j since m is broadcasted in a wireless cell of A_k. m is stored into an unordered message buffer $mbuf_i$ in a volatile storage in A_k temporarily. Transmission of m between M_i and M_j is reliable by using acknowledgment messages and a retransmission timer. If the timer is expired without receiving an acknowledgment message for m, m is retransmitted. However, it is not certain whether A_k receives m since A_k is not a destination of m and does not send an acknowledgment message for m. Each message m carries sequences of communication events on which a message receipt event $r(m)$ causally depends. If a message m is sent by a mobile computer to A_k for forwarding m to a fixed computer or a mobile computer out of the wireless cell of A_k, m is surely received by an acknowledgment message and a retransmission timer. In addition, m carries the order information of all messages m' whose sending and receipt events causally depends on $r(m)$. That is, m' is required for the mobile computers in the wireless cell of A_k to get consistent local states with a coordinated checkpoint $\tilde{C}$ for which a checkpoint request message is received by A_k after $r(m)$. Hence, if a communication event of a message m' in M_i is carried by m and m' is not stored in $mbuf_i$ due to loss of m, A_k requires M_i to retransmit m.

3.2 Message Logging Protocol

[Message Transmission from M_i to M_j]

1. At a message sending event $s(m)$, in a mobile computer M_i, $m.src = M_i$, $m.dst = M_j$, $m.event\_ids = event\_ids_i$, $event\_ids_i = \phi$, $m.prec\_mes\_ids = prec\_mes\_ids_i$ and an identifier of m is added to $prec\_mes\_ids_i$.
2. M_i broadcasts m to all mobile computers within a wireless cell of A_k.
3. On receipt of m (here, $m.dst = M_j$), $prec\_mes\_ids_j = prec\_mes\_ids_j \cup m.prec\_mes\_ids$ and M_j appends an identifier of m and of $r(m)$ to $prec\_mes\_ids_j$ and $event\_ids_j$, respectively. M_j

sends back an acknowledgment for m to M_i. Without receipt of the acknowledgment, if a re-transmission timer for m in M_i is expired, M_i re-transmits m.

4. On receipt of m (here, $m.dst \neq A_k$), A_k looks for a message which causally precedes m and has not yet stored in an unordered message buffer $mbuf_i$. If a message m' is not stored in $mbuf_i$ and a message log for M_i and an identifier of m' is in $m.prec\_mes\_ids$, A_k requests a sender of m' to transmit m' to A_k according to a later discussed re-transmission protocol.
5. A_k takes out a message m'' which is received by M_i at a message receipt event whose identifier is in $m.event\_ids$ out of $mbuf_i$ and stores m'' into a message log according to the order in $m.event\_ids$. Finally, A_k stores m into a message log and into $mbuf_j$. □

[Message Transmission from M_i to C out of a cell of A_k]

1. At a message sending event $s(m)$ in a mobile computer M_i, $m.src = M_i$, $m.dst = C$, $m.event\_ids = event\_ids_i$, $event\_ids_i = \phi$, $m.prec\_mes\_ids = prec\_mes\_ids_i$ and $prec\_mes\_ids_i = \phi$.
2. M_i transmits m to an access point A_k. Without receipt of an acknowledgment from A_k, if a re-transmission timer for m in M_i si expired, M_i re-transmits m.
3. On receipt of m, A_k sends back an acknowledgment to M_i, forwards m to a next hop for C and looks for a message which causally precedes m and has not yet stored in an unordered message buffer $mbuf_i$. If a message m' is not stored in $mbuf_i$ and a message log for M_i and an identifier of m' is in $m.prec\_mes\_ids$, A_k requests a sender of m' to transmit m' to A_k according to a later discussed re-transmission protocol.
4. A_k takes out a message m'' which is received by M_i at a message receipt event whose identifier is in $m.event\_ids$ out of $mbuf_i$ and stores m'' into a message log according to the order in $m.event\_ids$. Finally, A_k stores m into a message log. □

[Message Transmission from C out of a cell of A_k to M_j]

1. On receipt a message m from a computer C out of a wireless cell of A_k to a mobile computer M_j, A_k stores m into $mbuf_j$ and forwards m to M_i. Without receipt of an acknowledgment from M_i, if a re-transmission timer for m in A_k is expired, A_k re-transmits m.
2. On receipt of m, M_i appends an identifier of m and of $r(m)$ to $prec\_mes\_ids_j$ and $event\_ids_j$, respectively. M_j sends back an acknowledgment for m to A_k.

[Re-transmission from M_i to A_k]

1. If A_k detects a message m' which is not stored in $mbuf_i$ and a message log for M_i and an identifier of m' is in $m.prec\_mes\_ids$ for a received message m, A_k requests a sender of m' to transmit m' to A_k. This re-transmission procedure is concurrently executed with other procedures such as message transmission and reception. Without receipt of m, if a re-transmission timer for m' in A_k is expired, M_i retransmits the request.
2. On receipt of the request, M_i sends m to A_k. □

3.3 Checkpoint protocol

Fixed computers $F_1, \ldots, F_f$ 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 $F_1, \ldots, F_f$ 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 the current state information into a volatile storage.
3. Each F_i and A_k 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 takes c_{F_i} by making tc_{F_i} stable. Here, F_i stores the state information at tc_{F_i} 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. 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_k. A_k 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*, A_k stores the state information of M_i into a *tentative state log* tsl_i in a volatile storage of A_k.

[Tentative checkpoint tc_{M_i} in A_k]

1. M_i sends *TCreq* to A_k. *TCreq* carries the current state information of M_i.
2. On receipt of *TCreq*, A_k stores the state information of M_i carried by *TCreq* into tsl_i. □

When fixed computers take a coordinated checkpoint $\tilde{C}$, a checkpoint request message *Creq* is received by every access point. On receipt of *Creq*, A_k stores the state information at tc_{M_i} which is stored in a tentative state log tsl_i into a stable state log sl_i, in a stable storage. In addition, A_k stores a sequence of messages for achieving a local state consistent with $\tilde{C}$. Some of the messages are stored in a tentative message log tml_i and the others are required to be re-transmitted due to the concurrent re-transmission procedure. If all the required messages are stored in tml_i, these messages are taken from tml_i and stored into a stable message log ml_i.

[Checkpoint C_{M_i} in A_k]

1. On receipt of $Creq$, A_k stores the state information in tsl_i into sl_i. $tsl_i = \phi$.
2. After receiving all the messages causally precedes the receipt event of $Creq$ and is required to be re-transmitted, A_k stores these messages into ml_i. $tml_i = \phi$.
3. A_k sends back $Crep$ to a coordinator CS. □

3.4 Evaluation

Here, average number of MAC messages transmitted among mobile computers and an access points for transmission of an application message from a mobile computer M_i to another one M_j in a wireless cell of an access point A_k is evaluated. For evaluation, the proposed protocol P_p is compared to an extended conventional protocol [13] P_a for reliable message transmission in which an acknowledgment message is transmitted not only from M_j but also A_k. Let f be probability of message loss in a wireless communication channel. M_a and M_p are evaluated average numbers of MAC messages for P_a and P_p, respectively, where $g = 1 - (1 - f)^2$.

$$M_a = 3\sum_{k=1}^{\infty} k \cdot \{(1-g^k)^2 - (1-g^{k-1})^2\}$$

$$M_p = \left(2 + \frac{f(1-f)^2}{1-f^2(2-f)}\right)\sum_{k=1}^{\infty} k \cdot g^{k-1}(1-g)$$

As shown in Figure 5, M_p is less than M_a for any $0 \le f < 1$.

4 Concluding Remarks

This paper proposes a novel hybrid checkpoint protocol for supporting wireless LAN protocol such as IEEE802.11. The proposed protocol is applicable in a mobile network system based on a cell-dependent infrastructured model with losses of messages due to unreliable communication channels and existence of hidden terminals. Compared with a conventional protocol extended by adding an acknowledgment message transmission and retransmission timer for achieving reliable message transmission, our protocol requires less MAC message.

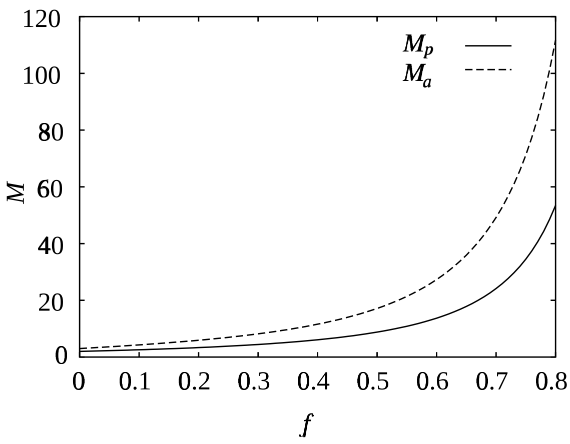

Figure 5: Numbers of MAC messages.

References

[1] "Specification of the Bluetooth Systems," http://www.bluetooth.com (1999).

[2] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," Standard IEEE802.11 (1997).

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

[4] 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).

[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., Johnson, D.B and Zwaenepoel, W., "The Performance of Consistent Checkpointing," Proc. of the 11th IEEE Symposium on Reliable Distributed Systems, pp. 39–47 (1992).

[7] 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).

[8] Fullmer, C.L. and Garcia-Luna-Aceves, J.J., "Solutions to Hidden Terminal Problems in Wireless Networks," Proc. of the ACM SIGCOMM'97, pp. 14–18 (1997).

[9] Higaki, H. and Takizawa, M., "Checkpoint-Recovery Protocol for Reliable Mobile Systems," Proc. of the 17th IEEE Symposium on Reliable Distributed Systems, pp. 93–99 (1998).

[10] Juang, T.T.Y. and Venkatesan, S., "Efficient Algorithms for Crash Recovery in Distributed Systems," Proc. of the 10th Conference on Foundations of Software Technology and Theoretical Computer Science, pp. 349–361 (1990).

[11] Kern, P., "MACA – A New Channel Access Method for Packet Radio," Proc. of the ARRL/CRRL Amateur Radio 9th Computer Networking Conference, pp. 134–140 (1990).

[12] 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).

[13] Morita, Y. and Higaki, H., "Fault-Tolerant Mobile Communication Network based on IEEE 802.11 and Mobile IP," Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications, pp. 972–978 (2001).

[14] Muller, G., Hue, M. and Peyrouz, N., "Performance of Consistent Checkpointing in a Modular Operating System: Result of the FTM Experiment," Proc. of the 1st European Dependable Computing Conference, pp. 357–365 (1994).

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

[16] Taesoon, P. and Yeom, H.Y., "An asynchronous recovery scheme based on optimistic message logging for mobile computing systems," Proc. of the 20th International Conference on Distributed Computing Systems, pp. 436–443 (2000).

[17] Ramanathan, P. and Shin, K.G., "Use of Common Time Base for Checkpointing and Rollback Recovery in a Distributed System," IEEE Trans. on Software Engineering, Vol. 19, No. 6, pp. 571–583 (1993).

[18] Randell, B., "System Structure for Software Fault Tolerance," IEEE Trans. on Software Engineering, Vol. SE–1, No. 2, pp. 220–232 (1975).

[19] Silva, L.M. and Silva, J.G., "Global Checkpointing for Distributed Programs," Proc. of the 11th IEEE Symposium on Reliable Distributed Systems, pp. 155–162 (1992).

[20] Smith, S.W., Johnson, D.B and Tygar, J.D., "Completely Asynchronous Optimistic Recovery with Minimal Rollbacks," Proc. of the 25th International Symposium on Fault-Tolerant Computing, pp. 361–370 (1995).

[21] Smith, S.W. and Johnson, D.B., "Minimizing Timestamp Size for Completely Asynchronous Optimistic Recovery with Minimal Rollback," Proc. of the 15th IEEE Symposium on Reliable Distributed Systems, pp. 66–75 (1996).

[22] Yao, B. and Fuchs, W.K., "Message logging optimization for wireless networks," Proc. of the 20th International Symposium on Reliable Distributed Systems, pp. 182–185 (2001).