2003－MBL－26　(13)
2003／9／26

アドホックネットワークのための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
小野 真和　桧垣 博章
E-mail: {masa,hig}@higlab.k.dendai.ac.jp

ネットワーク環境においてミッションクリティカルアプリケーションを実現する手法として、チェックポイントリカバリプロトコルがある。従来の有線ネットワークを対象としたプロトコルでは、状態情報を格納するための安定記憶が存在すること、一貫性のないメッセージ (紛失メッセージと孤児メッセージ) の検出、回避をメッセージの送信元コンピュータと送信先コンピュータの同期によって実現できる程度に十分な通信帯域幅が存在することが前提となっている。本論文では、これらの前提が成立しないアドホックネットワークにおけるチェックポイントプロトコルを提案する。状態情報は複数の隣接移動コンピュータに記憶する。このとき、紛失メッセージとなる可能性のあるメッセージを中継移動コンピュータの状態情報の一部として記憶することにより、状態情報とメッセージログを同一の移動コンピュータに保存することができる。最後に、コンピュータの移動のために孤児メッセージや紛失メッセージが発生する問題について述べる。

Checkpoint Protocol for Mobile Ad-hoc Networks

Masakazu Ono and Hiroaki Higaki
Department of Computers and Systems Engineering
Tokyo Denki University
E-mail: {masa,hig}@higlab.k.dendai.ac.jp

For achieving mission critical network applications, checkpoint recovery protocols have been researched and developed. In coventional protocols for wired networks, stable storages to store state information are assumed and enough bandwidth is assigned to synchronize a sender and a receiver computers of a message in order to avoid that the message becomes inconsistent, i.e. neither orphan nor lost. In this paper, we propose a novel checkpoint protocol in ad-hoc networks without stable storage and enough communication bandwidth.

1　はじめに

ノート型コンピュータやPDAなどの移動コンピュータや、IEEE802.11 [2] やHIPERLAN [1] などの無線通信プロトコルの研究開発が進み、広く普及している。また、無線基地局を介して有線ネットワークと接続されたインフラストラクチャネットワークだけでなく、移動コンピュータだけで構成されるアドホックネットワークへの要求が高まっている。アドホックネットワークの適用例として、一時的に構築されるイベント会場や災害現場などでのネットワーク、危険地帯で無線基地局が設置できない場所における自律移動型ロボットの協調動作のためのネットワーク、センサネットワークなどがある。このようなアドホックネットワークにおいて、ミッションクリティカルアプリケーションの実行を考えたとき、耐故障性を実現するためのチェックポイントリカバリ手法を適用することが考えられる。しかし、従来のチェックポイント手法は安定記憶がネットワーク上に存在することを前提としており、また、一貫性のないメッセージ (孤児メッセージ、紛失メッセージ) を送信元コンピュータと送信先コンピュータの同期によって検出することが可能な帯域幅がネットワークによって提供されているとしている。そのため、アドホックネットワーク上の移動コンピュータは安定記憶を持てない問題や、狭帯域幅で低信頼なネットワークによる同期のための通信オーバヘッドの拡大の問題を解決する必要がある。そこで、本論文ではアドホックネットワークにおいて安定記憶を実現し、送受信コンピュータ間における同期のための通信オーバヘッドを回避する新たなチェックポイントプロトコルを提案する。

2 従来手法

2.1 チェックポイントプロトコル

チェックポイントプロトコルによって、各移動コンピュータ $M_i \in V$ が設定したローカルチェックポイント c_i の集合であるグローバルチェックポイント C_V が一貫性を持つとは、次の性質を満たすことをいう [3]

[定義]

1) M_s から M_r へ配送されるメッセージ m が紛失メッセージであるとは、グローバルチェックポイント C_V に対して、$Send(m)$ が c_s に先行し、c_r が $Receive(m)$ に先行することである。なお、$Send()$ と $Receive()$ は、アプリケーションにおけるメッセージ送受信イベントである。
2) メッセージ m が孤児メッセージであるとは、C_V に対して、c_s が $Send(m)$ に先行し、$Receive(m)$ が c_r に先行することである。
3) 紛失メッセージ、孤児メッセージを含まないグローバルチェックポイントは、一貫性が保たれているという。□

ただし、紛失メッセージをリカバリ回復時に再送信することができれば、システム状態の一貫性を維持することが可能である。そこで、一貫性のあるグローバルチェックポイントを以下のように再定義する。

[定義]

4) 一貫性のあるグローバルチェックポイントとは、孤児メッセージを含まず、すべての紛失メッセージをリカバリ回復時に再送信可能であるものである。□

従来のチェックポイントプロトコルにおいては、m が C_V に対して紛失メッセージや孤児メッセージとなることを M_r でのみ判定することを前提としている。そのため、これらの発生を回避するには、システム全体での同期を必要としていた。例えば、Koo [6] のプロトコルにおいては、チェックポイント要求メッセージ $CReq$ を受信してから、チェックポイント終了メッセージ $CFin$ を受信するまでの間、アプリケーションメッセージの送信を禁止している。ところが、アドホックネットワークにおいては、無線通信の狭帯域幅、低信頼性といった性質により、同期のコストが大きく、アプリケーションの停止時間が長くなるという問題がある。そこで、アドホックネットワークにおいては、m が紛失メッセージあるいは孤児メッセージとなる可能性を、m の配送経路上にある移動コンピュータが判定し、必要に応じて m を記憶したり、m の転送を遅延させたりすることが望まれる。ここでは、隣接する移動コンピュータ間における同期のみが必要であることから、アプリケーションの停止時間を短縮することが可能である。

2.2 モバイルチェックポイントプロトコル

モバイルチェックポイントプロトコルの実現にあたり、論文 [7] では、移動コンピュータを含むネットワークを以下の 4 つのモデルに分類している。

1) Centralized Networks
2) Cell Dependent Infrastructured Networks
3) Cell Independent Infrastructured Networks
4) Ad-hoc Networks

1)–3) は、ネットワークの構成要素に固定コンピュータを含んでいる。そこで、固定コンピュータに実現した安定記憶に移動コンピュータの状態情報を保存することにより、チェックポイントを設定することができる。論文 [5] では、同期チェックポイント手法と非同期チェックポイント手法を組み合わせた複合チェックポイント手法を提案している。[5,8] および [7] において、それぞれ 1)、2) に対するプロトコルを設計している。ところが 4) においては、ネットワークの構成要素が移動コンピュータのみであることから、安定記憶の実現が困難である。そこで、各移動コンピュータのチェックポイントの設定を、複数の移動コンピュータに状態情報を記憶させることによって実現する。

3 提案プロトコル

以下の条件のもとでプロトコルを構成する。

[前提条件]

1) アドホックネットワークに含まれるすべての移動コンピュータは、チェックポイントプロトコルの実行中、マルチホップ配送により互いにメッセージ交換が可能である。
2) 隣接する移動コンピュータ間の通信リンクは、動的に切断および確立されることがある。
3) 各移動コンピュータは、隣接する移動コンピュータのリストを保持している。
4) 隣接する移動コンピュータ間の通信リンクは双方向であり、半二重通信が行なわれる。また、ユニキャスト通信は、受信確認と再送機構により、メッセージの紛失なく実現されているものとする。□

チェックポイントプロトコルの基本形を示す。チェックポイントプロトコルの開始は、任意の移動コンピュータが任意のタイミングで行なうことができる。チェックポイント設定要求の伝達と、チェックポイント間の同期は、チェックポイント設定要求メッセージ $CReq$ の

フラッディングによって実現される (図 1) [4]。

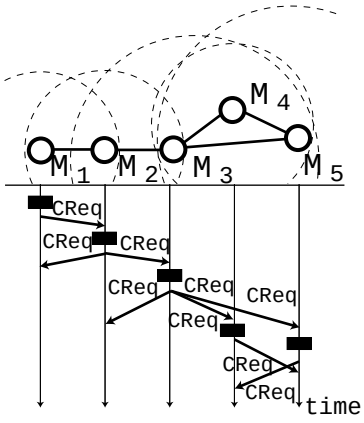

図 1: チェックポイントプロトコル

$CReq$ を受信した移動コンピュータ M_i は、現在の状態情報 S_i を獲得することによってローカルチェックポイント c_i を設定するとともに、$CReq$ を隣接する移動コンピュータ群、すなわち、M_i の無線信号到達範囲内に存在するすべての移動コンピュータにブロードキャストする。これを繰り返すことによって、前提条件 1) により、すべての移動コンピュータにおいて、ローカルチェックポイントを設定することができる。

ここで、移動コンピュータ M_i の状態情報 S_i を安定記憶に保存するために、S_i を複数の隣接移動コンピュータに記憶させる手法を用いる。各移動コンピュータは、ローカルチェックポイント c_i における状態情報 S_i を獲得してから $CReq$ のブロードキャストを行なうことから、S_i を $CReq$ によって配送することにより、追加のメッセージを要することなく S_i の配送が実現される。

[アドホックチェックポイントプロトコル (基本形)]

1) 任意の移動コンピュータ M_0 が、M_0 の状態情報 S_0 を獲得することでローカルチェックポイント c_i を設定するとともに、S_0 を含み、M_0 が生成した ID が付与されたチェックポイント設定要求メッセージ $CReq$ を M_0 の無線信号到達範囲内にブロードキャストする。このとき、タイマ T_0 をセットする。
2) 移動コンピュータ M_i が送信したチェックポイント設定要求メッセージ $CReq$ を受信した移動コンピュータ M_j は、以下の処理を行なう。
   2-1) M_i から同一の ID を持つ $CReq$ を受信していないならば、受信した $CReq$ に含まれる M_i の状態情報 S_i を保存する。
   2-2) M_j が一度も $CReq$ を受信していないならば、M_j の状態情報 S_j を獲得するとともに、S_j を含み、受信した $CReq$ と同一の ID を付与した $CReq$ を M_j の無線信号到達範囲内にブロードキャストする。このとき、タイマ T_j をセットする。
3) 移動コンピュータ M_j が近隣する移動コンピュータリスト L_j に含まれるすべての移動コンピュータから $CReq$ を受信する以前にタイマ T_j が時間切れとなったならば、M_j は、同じ $CReq$ を再度ブロードキャストする。□

ここで、チェックポイントプロトコルの実行と並行に送受信されたメッセージは、紛失メッセージや孤児メッセージとなる可能性がある。紛失メッセージは、いずれかの移動コンピュータに保存し、リカバリ回復時に、保存されたメッセージを再送信することによって、システム状態の一貫性を維持することができる。一方、孤児メッセージは、リカバリ再実行時に送信元移動コンピュータが同一のメッセージを再度送信する保障がないことから、その発生を回避しなければならない。移動コンピュータの移動による通信路の切断と接続が発生しない場合には、以下の性質が成り立つ。

[性質]

1) 紛失メッセージ m_l は、その配送経路上で以下のいずれかの条件を満足する。
   1-a) m_l の配送経路上にある 1 台以上の移動コンピュータ M_i において、$receive(m_l) \rightarrow c_i \rightarrow send(m_l)$ が成り立つ。
   ただし、$\rightarrow$ は、イベント間の happen before の関係を表すものとする。
   1-b) m_l の配送経路上にある 2 台の移動コンピュータ M_i,M_j において、M_i,M_j は互いに直接通信可能であり、$send(m_l) \rightarrow c_i$ かつ $c_j \rightarrow receive(m_l)$ が成り立つ。
2) 孤児メッセージ m_o は、その配送経路上で以下の条件を満足する。
   2-a) m_o の配送経路上にある 2 台の移動コンピュータ M_i,M_j において、M_i と M_j は互いに直接通信可能であり、$c_i \rightarrow send(m_o)$ かつ $receive(m_o) \rightarrow c_j$ が成り立つ。□

性質 1) により、紛失メッセージとなる可能性があるメッセージ m_l を中継移動コンピュータ、すなわち m_l の送信元でも送信先でもない移動コンピュータが検出することができる。もし、この検出が送信先移動コンピュータ M_r でのみ検出可能であるならば、(例えば、論文 [3,6] では、M_r で m_l が紛失メッセージとなることを検出している。) M_r が m_l を受信した時点、すなわち $Receive(m_l)$ においては、M_r がすでに $CReq$ を

隣接移動コンピュータへブロードキャスト送信済みであることが考えられる。この場合、m_l を隣接移動コンピュータに記憶させるために、m_l を含む制御メッセージをブロードキャストする必要がある。しかし、これによって、以下の問題が発生する。

(1) M_r がリカバリで必要とする情報を記憶させるためのチェックポイントプロトコルで送受信されるメッセージが増加する。

(2) M_r の状態情報 S_r を保存した隣接移動コンピュータがこの制御メッセージの送信時点においても M_r の無線信号到達範囲内に存在することは保証できない。すなわち、リカバリで必要とする情報が複数の移動コンピュータに分散することによって、リカバリ時に送受信されるメッセージ数が増加する。

(3) チェックポイントプロトコルが終結したことを各移動コンピュータが検出するためには、新しい同期メッセージの導入が必要となる。

そこで、紛失メッセージとなる可能性のあるメッセージ m_l を $CReq$ を送信する前に検出可能な移動コンピュータの存在が不可欠である。ここで、条件 1-a) を満たすメッセージ m_l については、移動コンピュータ M_i が m_l を $CReq$ の送信前に検出することができる。

[紛失メッセージとなる可能性のあるメッセージの検出 (1)]

移動コンピュータ M_i が中継メッセージ (M_i を送信元、送信先としないメッセージ m_l) を $CReq$ の送信前に受信し、$CReq$ 送信時までに送信していないならば m_l は紛失メッセージとなる可能性のあるメッセージである。M_i は m_l を $CReq$ に含めて送信することができる。□

一方、条件 1-b) を満たすメッセージ m_l については、m_l が紛失メッセージとなる可能性があるメッセージであることを検出できるのは M_j であり、検出したとき M_j はすでに $CReq$ メッセージを送信済みであることがある。そこで、以下の方法により m_l の検出を M_j で行ない、その結果を M_i が $CReq$ を送信する前に通知する。

[紛失メッセージとなる可能性のあるメッセージの検出 (2)]

$CReq$ を送信していない移動コンピュータ M_i から送信されたメッセージ m_l を移動コンピュータ M_j が $CReq$ 送信後に受信したならば、m_l の受信確認応答メッセージに m_l が紛失メッセージとなる可能性があることを示す情報を付加する。M_i は、これを受信したならば、$CReq$ を送信するときに状態情報 S_i とともに m_l をブロードキャストする。この手法を適用するために、各移動コンピュータは、メッセージ送信時から受信確認応答メッセージの受信までの間は $CReq$ を送信することを禁止することができない。□

条件 1-a)、条件 1-b) はメッセージ m_l が紛失メッセージとなる必要条件であり、十分条件ではない。例えば、図 2 において、m は M_1 から送信される $CReq$ に含まれるがこれは紛失メッセージではない。

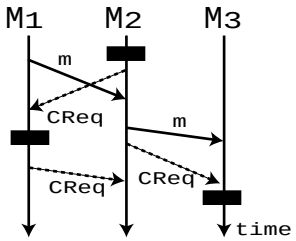

図 2: 紛失メッセージとならない場合

もし、グローバルチェックポイント $C_V = \{c_1, c_2, c_3\}$ から、リカバリしたならば、m は M_1 によって再送信されるが、M_3 では送信済みである。ここで、m が紛失メッセージとなる可能性のあるメッセージであることを検出した移動コンピュータ (m を含む $CReq$ を送信した移動コンピュータではない) は、m がリカバリ時に再送信されるメッセージであることを示す情報を m に付与する。m を受信した送信先移動コンピュータでは、m の受信が $CReq$ の送信前であるならば、再送信時に m を受信しても破棄することを示す情報を $CReq$ に含めることができる。また、m の受信が $CReq$ の送信後であるならば、m は紛失メッセージであるので、リカバリ後に再送信されたものを受信する必要がある。

また図 3 のように複数の移動コンピュータが、1 つのメッセージを紛失メッセージとなる可能性のあるメッセージであると検出することがある。

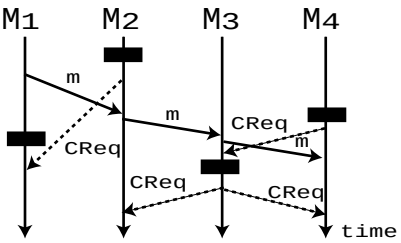

図 3: 紛失メッセージとなる場合

例えば、m は M_2 および M_4 において紛失メッセージとなる可能性のあるメッセージとして検出される。も

し、これらの移動コンピュータがそれぞれの $CReq$ に m を含めるならば、リカバリ時に送信先移動コンピュータ M_4 は複数の m を受信することになる。この問題を解決するために、前段で述べた再送信時に破棄すべきメッセージであることを示す情報を用いる。もし、この情報を含むメッセージを紛失メッセージの可能性があるメッセージであると検出しても、これを以降に送信される $CReq$ に含めないことによって、最初に検出した移動コンピュータ (または、この移動コンピュータにこのメッセージを送信した移動コンピュータ) の送信する $CReq$ にのみこのメッセージを含ませることができる。

孤児メッセージとなる可能性のあるメッセージは、中継移動コンピュータでは対処を行わない。送信先移動コンピュータで受信を遅延させることによってのみ対処すれば十分である。

[メッセージ通信プロトコル]

メッセージ m には、$m.source\_creq\_sent$、$m.creq\_set$、$m.logged$ という 3 つのフラグが含まれる。

(送信 $Send(m)$)

1 $m.logged := false$ とする。

2 送信元移動コンピュータ M_s が $CReq$ を送信済みであるならば、$m.creq\_sent := true$, 未送信であるならば $m.creq\_sent := false$ とする。

3 $m.source\_creq\_sent := m.creq\_sent$ とする。

4 $send(m)$ により m を送信する。m の受信確認応答メッセージ $ack(m)$ を受信するまで $CReq$ の送信を禁止する。

5 $ack(m)$ を受信し、$m.logged = false$ かつ $ack(m).logged = true$ であるならば、以降に送信される $CReq$ に m を含める。

(中継 $receive(m)$)

1 $m.logged = false$ かつ $m.creq\_sent = false$ かつ中継移動コンピュータ M_t が $CReq$ を送信済みであるならば、$m.logged := true$、$ack(m).logged := true$ として $ack(m)$ を返送する。

2 それ以外の場合は、$ack(m).logged := m.logged$ として $ack(m)$ を返送する。

(中継 $send(m)$)

1 送信元移動コンピュータ Ms が $CReq$ を送信済みであるならば、$m.creq\_sent := true$, 未送信であるならば $m.creq\_sent := false$ とする。

2 $send(m)$ により m を送信する。m の受信確認応答メッセージ $ack(m)$ を受信するまで $CReq$ の送信を禁止する。

3 $ack(m)$ を受信し、$m.logged = false$ かつ $ack(m).logged = true$ であるならば、以降に送信される $CReq$ に m を含める。

(受信 $Receive(m)$)

1 $m.logged = false$ かつ $m.creq\_sent = false$ かつ送信先移動コンピュータ M_d が $CReq$ を送信済みであるならば、$m.logged := true$、$ack(m).logged := true$ として $ack(m)$ を返送する。

2 それ以外の場合は、$ack(m).logged := m.logged$ として $ack(m)$ を返送する。

3 もし、$m.source\_creq\_sent = true$ かつ M_d が $CReq$ を未送信であるならば、$CReq$ の送信が終了するまで一時停止する。

4 $Receive(m)$ により m を受理する。

5 $m.logged = false$ かつ、チェックポイント設定のための状態情報 S_d が獲得済みで $CReq$ 送信前ならば、以降に送信される $CReq$ に m を含める。□

各移動コンピュータ M_i は、$CReq$ に状態情報 S_i とともに含ませるメッセージの集合をメッセージログ ML_i として保存することとする。

[アドホックチェックポイントプロトコル]

1) 任意の移動コンピュータ M_0 が、M_0 の状態情報 S_0 を獲得することでローカルチェックポイント c_i を設定するとともに、S_0 とメッセージログ ML_0 を含み、M_0 が生成した ID が付与されたチェックポイント設定要求メッセージ $CReq$ を M_0 の無線信号到達範囲内にブロードキャストする。このとき、タイマ T_0 をセットする。

2) 移動コンピュータ M_i が送信したチェックポイント設定要求メッセージ $CReq$ を受信した移動コンピュータ M_j は、以下の処理を行なう。

2-1) M_i から同一の ID を持つ $CReq$ を受信していないならば、受信した $CReq$ に含まれる M_i の状態情報 S_i とメッセージログ ML_i を保存する。

2-2) M_j が一度も $CReq$ を受信していないならば、

M_j の状態情報 S_j とメッセージログ ML_j を獲得するとともに、S_j を含み、受信した $CReq$ と同一の ID を付与した $CReq$ を M_j の無線信号到達範囲内にブロードキャストする。このとき、タイマ T_j をセットする。

3) 移動コンピュータ M_j が近隣する移動コンピュータリスト L_j に含まれるすべての移動コンピュータから $CReq$ を受信する以前にタイマ T_j が時間切れとなったならば、M_j は、同じ $CReq$ を再度ブロードキャストする。

4) $ML_i := \phi$ として終了する。□

4 まとめと今後の課題

本論文では、アドホックネットワークにおけるチェックポイントプロトコルの概略を示した。紛失メッセージとなる可能性のあるメッセージを、エンド–エンドではなく、ホップバイホップで検証し、メッセージログに保存する機構を導入することにより、各移動コンピュータが隣接移動コンピュータに対して状態情報とメッセージログを含むチェックポイント要求メッセージを一度だけ送信することにより、同期オーバヘッドを削減することができる。本論文では、チェックポイントプロトコルの実行中のコンピュータの移動については考慮していない。

しかし、移動コンピュータ間のリンクが切断、確立することにより、図 4 に示すように、性質 1)2) を満たさない紛失メッセージ、孤児メッセージが発生する。そこで、以下の条件を満たすメッセージは、紛失メッセージ、孤児メッセージとなる可能性のあるメッセージであると判定する可能性がある。

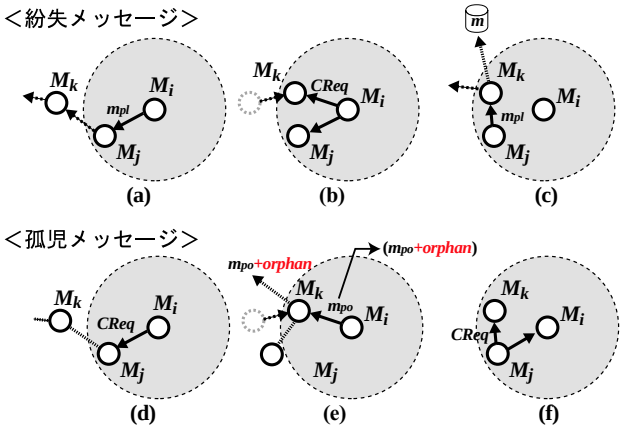

図 4: 紛失/孤児メッセージになり得るメッセージ

[性質 (スケッチ)]

3) m_{pl} を M_j から受信した M_k において m_{pl} を受信する直前のチェックポイント c_k を設定したとき、M_i が M_k の隣接する移動コンピュータリストに含まれていなかった場合、m_{pl} は紛失メッセージとなる可能性がある。

4) m_{po} を M_k に送信した M_i において m_{po} を送信する直前のチェックポイント c_i を設定したとき、M_k が M_i の隣接する移動コンピュータリストに含まれていなかった場合、m_{po} は孤児メッセージとなる可能性がある。□

紛失メッセージとなる可能性のある m_{pl} は、M_k によって保存される (図 4(c))。一方、孤児メッセージとなる可能性のある m_{po} には、孤児メッセージとなる可能性があることを示す情報を添付し (図 4(d))、送信先移動コンピュータで、ローカルチェックポイント設定手続きを終えるまでアプリケーションでの受信を遅延することで対処が可能であると考えられる。

今後は上記事項を検証するとともに、リカバリプロトコルの設計を行なう。

参考文献

[1] “Radio Equipment and Systems (RES); HIPERLAN,” ETSI,Functional Specifications (1995)

[2] “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” Standard IEEE 802.11 (1999).

[3] 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).

[4] Corson, M.S. and Ephremides, A., “A Distributed Routing Algorithm for Mobile Wireless Networks,” ACM Journal of Wireless Networks, vol. 1, No. 1, pp. 61–81 (1995).

[5] Higaki, H. and Takizawa, M., “Checkpoint-Recovery Protocol for Reliable Mobile Systems,” Proc. of the 17th SRDS, pp.93–99 (1998).

[6] 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).

[7] Miyazaki, M., Morita, Y. and Higaki, H., “Hybrid Checkpoint Protocol for Mobile Networks with Unreliable Wireless Communication Channels,” Proc. of the 2nd AMOC, pp.164–171 (2002).

[8] Morita, Y. and Higaki, H., “Checkpoint-Recovery for Mobile Computing Systems,” Proc. of the 21st ICDCS Workshops, pp.479–484 (2001).