셀룰라 네트워크를 위한 조정된 2-계층 체크포인팅 알고리즘

정회원 변 계 섭*, 김 재 훈*

Two-Tier Coordinated Checkpointing Algorithm for Cellular Networks

Kyue-Sup Byun*, Jai-Hoon Kim* ***Regular Members***

요 약

이동 컴퓨팅 환경에서는 충분하지 못한 자원으로 인해 기존의 분산 알고리즘이 비효과적인 경우가 많다. 특히 이동 호스트의 낮은 가용성으로 인하여 중단없는 서비스를 보장받기 힘들다. 이들의 원인은 낮은 대역폭과 호스트들의 이동성, 작은 저장장치 및 안전하지 않은 저장장치, 이동 호스트와 이동 호스트, 이동 호스트와 기지국간의 통신 오버헤드 그리고 충분하지 못한 배터리 수명들을 들 수 있다. 특히 통신 오버헤드(이동 호스트를 찾는 오버헤드, 이동 호스트의 정보를 저장하는 오버헤드 등)로 인해 호스트의 배터리 수명을 단축시키고 통신 비용을 증가시킨다. 이러한 통신 오버헤드를 최소화함으로써 이동 호스트의 배터리 수명이나 고장으로부터 좀 더 안전한 시스템을 구성할 수 있다. 본 논문에서는 셀룰라 네트워크를 기반으로 하는 이동 컴퓨팅 환경에서 통신비용과 이동 호스트의 작업을 최소화시키는 조정된 2-계층 체크포인팅(coordinated checkpointing) 알고리즘을 제안하고 수학적으로 비용을 분석하였으며 기존 알고리즘과 성능을 비교하였다. 기존의 조정된 체크포인팅 알고리즘에서 통신 비용(메시지 수)의 복잡도는 이동 호스트의 수의 제곱($O(n^2)$)이지만, 제안한 알고리즘은 기지국의 수 더하기 이동 호스트의 수의 제곱($O(n+m^2)$)이다. 일반적으로 기지국의 수에 비해 이동 호스트의 수가 상대적으로 훨씬 많기 때문에 ($n>>m$) 기존의 알고리즘에 비해 제안한 알고리즘은 상대적으로 많은 통신 비용을 감소시킬 수 있다.

ABSTRACT

Mobile computing raises many new issues, such as lack of stable storage, low bandwidth of wireless channel, high mobility, and limited battery life. These new issues make traditional coordinated checkpointing algorithms unsuitable. In this paper, we present the concept of two-tier coordinated checkpointing algorithm in which the messages are requested by the mobile host are handled by the appropriate MSS. Also, the broadcast messages are handled by MSS instead of relaying the messages to all the mobile hosts directly as in the previous algorithms. In this way, the communication cost can be reduced and one of the MSS works as a coordinator and maintains the overall system consistency. Mobile computing based on two-tier coordinated checkpointing algorithm reduces the number of synchronization messages. Performance comparisons by parametric analysis depict that a two-tier coordinated checkpointing algorithm can reduce communication cost greatly compared to the previous algorithms in which the messages are directly sent to the mobile hosts.

* 아주대학교 정보통신전문대학원 분산이동컴퓨팅 연구실(jaikim@madang.ajou.ac.kr)
논문번호 : K01095-0308, 접수일자 : 2001년 3월 8일
※ 본 연구는 한국과학재단 목적기초연구(2001-1-30300-016-2) 지원으로 수행되었음.

Ⅰ. 서 론

이동 컴퓨팅 환경은 낮은 대역폭과 호스트들의 이동성, 작은 저장공간 및 안전하지 않은 저장장치[13], 호스트간, 호스트와 기지국간의 통신오버헤드 그리고 충분하지 못한 배터리 수명 등으로 인해 기존의 분산 알고리즘을 변경 없이 그대로 이용하기에 적합하지 않다. 그러므로 짧은 배터리 수명이나 호스트의 이동간에 일어날 수 있는 장애들을 고려한 새로운 알고리즘이 필요하다. 특히 호스트 장애 발생시 호스트와 기지국간의 통신이 필요하게 되는데, 이 때 통신간의 큰 오버헤드로 장애가 발생할 수 있다. 그러므로 이런 장애에 대처하기위해 적은 메시지 로깅이나 통신 회수를 줄이는 알고리즘이 필요하다.

본 논문에서는 이동 호스트와 기지국간의 통신 비용을 최소화할 수 있는 조정된 체크포인팅(coordinated checkpointing) 알고리즘을 제안한다. 조정된 체크포인팅은 분산 시스템에서 일부 노드에 장애가 발생했을 때, 모든 노드가 일치성을 유지한 과거의 상태로 되돌아가(Roll-back) 다시 수행할 수 있도록 상태를 저장하는 기법이다. 이 때, 모든 노드가 일치성을 유지하도록 각 노드의 상태를 저장하고 전송중인 메시지를 로깅하게 된다. 각 노드는 다른 모든 노드로 일치성을 유지를 위한 제어 메시지를 보내야 하므로 통신 비용이 많이 소요된다. 특히, 무선 통신망을 포함하는 이동 컴퓨팅환경에서는 통신 대역폭이 충분하지 않고 이동 호스트의 컴퓨팅 자원 및 전원이 부족하므로 기존의 조정된(coordinated) 기법[11,15]을 그대로 사용하는 것은 부적절하다. 본 논문에서는 조정된 체크포인팅 알고리즘[11,15]을 이동 컴퓨팅 환경에 적합하도록 적은 통신 비용과 이동 호스트의 작업을 줄이는 방법을 제안하여 장애에 대해 안전하고 호스트가 갖는 정보를 보다 안전하게 저장할 수 있는 시스템을 구성할 수 있도록 한다. 제안하는 협조적인 체크포인팅을 통하여 일관된 시스템 정보를 저장하고 2-계층으로 알고리즘[2]을 구성해 비용을 최소화한다.

Ⅱ. 관련 연구

1. 일치성 있는 시스템 상태(Consistent System States)

메시지 패싱 시스템의 상태는 시스템에 참가하는 모든 프로세스 각각의 상태와 통신채널 상태로 구성된다. 일치성있는 시스템 상태는 분산 처리를 위한 실행 중에 발생하는 상태 중 하나이다. 일치성있는 시스템 상태란 수신된 모든 메시지는 송신자의 상태에서 보면 이미 보내진 것으로 보여지는 것이다[11]. 예를 들어, 그림 1은 일치성있는 단면과 일치성이 없는 단면(snap shot)으로 인해 발생하는 세 개의 프로세스 간의 일치성있는 상태와 일치성이 없는 시스템의 상태를 보여준다. 그림 1에서 알 수 있듯이 일치성이 없는 시스템에서의 프로세스 p2가 메시지 m1을 받았음에도 불구하고 프로세스 p1의 상태는 아직 메시지를 송신하지 못하였다. 반면 일치성있는 단면은 수신된 메시지를 항시 송신측에서 보낸 것으로 나타난다.

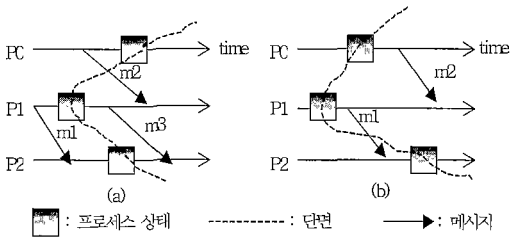


그림 1. (a) 일치성있는 단면 (b) 일치성없는 단면

2. 조정된 체크포인팅(Coordinated Checkpointing)

조정된 체크포인팅에서 프로세스는 전체적인 시스템의 일치성있는 상태를 위해서 서로 간의 체크포인트를 조정한다. 조정된 체크포인팅은 모든 프로세스가 최근의 조정된 체크포인트로부터 항상 다시 시작할 수 있는 일치성 유지를 위하여 계속 과거의 상태로 연속하여 되돌아 가야 하는 도미노 효과를 일으키지 않는다. 또한 조정되지않은 체크포인팅 기법과는 달리 일치성을 유지한 시스템 상태를 찾기 위한 오버헤드가 거의 없다. 그러나 각각의 프로세스는 자신만의 체크포인트를 할 수 없고 동시에 체크포인팅을 해야 하므로 비효율적인 가능성이 있다. 또한 일치성 유지를 위한 제어 메시지 오버헤드가 소요되며, 어떤 작업에 앞서 조정자 역할을 하는 프로세스가 있어야 한다. 일치성있는 체크포인팅을 위해서는 체크포인팅 프로토콜이 실행되기 전까지 프로세스간 통신이 블록되어야 한다[3]. 이것은 2-phase 블록킹 프로토콜로 구현할 수 있다[15]. 2-phase 블록킹 프로토콜은 조정자(coordinator)가 체크포인트를 할 것에 관한 메시지를 브로드캐스트하는 것이다. 프로세스가 이 메시지를 받았을 때, 체크포인트를

수행하고, 이 때는 어플리케이션 메시지를 중지한다. 그런 다음, 체크포인트가 끝났음을 조정자에게 알린다. 다른 프로세스로부터 체크포인트 완료 메시지를 받은 조정자는 전체 프로세스에게 전체적으로 체크포인트가 이루어졌음을 알린다. 이 메시지를 받은 프로세스들은 이전에 하고 있었던 어플리케이션 메시지를 다시 보내게 된다. 이 경우엔 전체적인 일관된 시스템을 유지하기 위해 기존의 어플리케이션 메시지를 중지하고 먼저 체크포인트 메시지를 받아서 일관된 상태를 유지한 다음, 이 작업이 끝나고 다시 작업을 수행하게 된다. 그런데 이 때, 결함이 발생하였다면, 이미 저장되어있는 가장 최근의 체크포인트 지점으로 롤백(Roll-back)하여 작업을 계속 수행하게 된다.

3. 비 블럭킹 체크포인트 조정(Nonblocking Checkpoint Coordination)

블럭킹(blocking checkpoint)기법은 도중에 프로세스간 메시지 전달이 블록킹(중단)되므로 성능 저하의 우려가 있다. 이를 개선하기 위하여 메시지 전달에 체크포인팅 프로토콜 수행도중 중단되지않는 비블럭킹 체크포인트(nonblocking checkpoint)기법이 제안되었다. 블록킹을 하지않는 체크포인트 조정은 체크포인트 요청 메시지를 다른 프로세스들에게 보내게 되는데, 그림 2의 (a)처럼 p가 체크포인트 요청 메시지를 받은 후에 어플리케이션 메시지(m)를 보냈을 때, q는 체크포인트 요청 메시지가 도착하기 전에 어플리케이션 메시지(m)를 받게 되기 때문에 일치성이 유지되지 못한다. 이 경우에 FIFO채널을 이용하면, 체크포인트 요청 메시지가 보내지기 전에 어플리케이션 메시지가 항상 먼저 발생하여서 이 문제를 해결할 수 있다. 이럴 경우엔 첫번째 체크포인트 요청 메시지가 도착하였을 때, 각 프로세스들은 체크 포인트를 수행하게 된다(그림 2의 (b)). Chandy와 Lamport[11]가 제안한 분산 순간 상태 알고리즘(distributed snapshot algorithm)은 이런 비블럭킹 체크포인트 조정 프로토콜(nonblocking checkpoint coordination protocol)이다. FIFO채널이 아닐 때는 그림 2의 (c)와 같이 어플리케이션 메시지(m)에 체크포인트 요청을 함께 실어 보내어 일치성을 유지할 수 있다[4].

4. Global-State-Detection Algorithm[11]

전체적인 시스템의 일관성을 유지하기 위해 Chandy와 Lamport는 marker라고 하는 일종의 체크

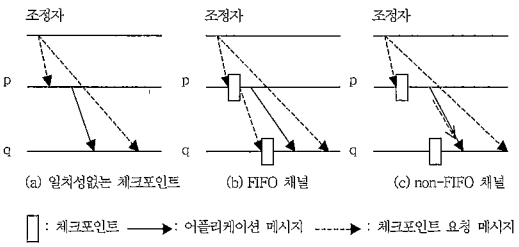


그림 2. 조정된 비블럭킹 체크포인팅
(Nonblocking coordinated checkpointing)

포인트 요청 메시지를 보낸다. 이 알고리즘은 시스템의 전체적인 일관성을 지키기 위한 것으로 이는 결함이 발생하였을 때 최소한의 비용을 들여 이전 체크포인트한 지점으로 롤백하여 다시 시스템을 정상화시키는데 효율적이다. 이를 알고리즘 1에 요약하였다.

5. Two-tier Principle[2]

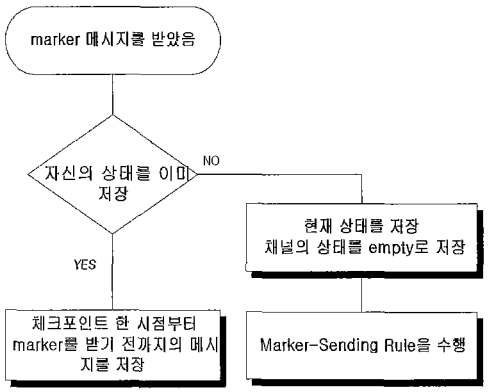
셀룰라 네트워크를 기반으로 한 이동컴퓨팅 환경은 이동호스트의 컴퓨팅 자원이 부족하고 무선 통신망의 대역폭이 충분하지 않기 때문에 기존의 분산 알고리즘을 그대로 이용하기에 부적합하다. 이동호스트의 컴퓨팅 작업 및 메시지 송수신을 최소화하기 위하여 되도록 유선 통신망과 고정 호스트(기지국)의 자원을 이용하여 작업을 수행하고 이동 호스트는 해당 기지국과 최소한의 통신만 수행하도록 2-계층기법의 분산 알고리즘을 제안하였다. 2-계층 기법의 사용 예로써 기존의 토큰 링 상호배제(Mutual Exclusion) 알고리즘을 변형하여 셀룰라 네트워크에 기반 한 이동 컴퓨팅 환경에 적용하였다. 토큰 링 상호배제 알고리즘은 논리적 토큰 링으로 네트워크가 구성되어 있으며 토큰을 받은 이동 호스트만이 자원을 이용할 수 있게 된다. 이는 상호배제를 보장하며, 각각의 자원들에 공평하게 자원을 분배할 수 있는 이점이 있다. 기존의 토큰 링 상호배제 알고리즘을 그대로 적용하면 토큰이 모든 이동호스트를 논리적 링의 순서대로 방문하게 되어 무선 통신망과 이동 호스트의 자원낭비가 심하게 된다. 2-계층기법을 이용하여 이를 개선하면 토큰은 기지국을 방문하게 되고 자원사용을 원하는 이동 호스트는 해당 기지국에 요청을 하며 토큰이 해당 기지국에 도착 시 자원사용을 허가 받게 된다. 이와 같이 2-계층 기법은 토큰이 불필요하게 모든 이동 호스트를 방문하는 낭비를 막아 비용을 줄일 수 있다. Badrinath의 논문 [2]에서는 이동 호스트의 이

동에 따른 위치 정보를 관리하는 기법에 따라 Search Strategy, Inform Strategy 그리고 Proxy Strategy로 구분하여서 2-계층 기법의 성능을 분석하였다. 본 논문에서는 2-계층 기법을 기반으로 하여 조정된 체크포인팅 알고리즘을 제안한다.

알고리즘 1: Chandy & Lamport 알고리즘[11]

● Marker-Sending Rule for a Process p.
프로세스 p는 먼저 자신의 상태를 저장하고 p로부터 외부로 나가는 모든 채널 c를 통해 marker message를 보낸다.

● Marker-Receiving Rule for a Process q.
if(저장하지 않았다면)
begin
현재 상태를 저장.
채널의 상태를 empty로 저장.
Marker-Sending Rule을 수행.
end
else
체크포인트한 시점부터 marker을 받기 전까지의 메시지를 저장



6. 이동성에 대한 고려

위치 이동성을 가진 이동 호스트(mh)에게 메시지를 전달하는 것은 상당히 어렵고 복잡한 일이다. 한 노드에서 다른 노드에게 메시지를 보낼 때, 목적 노드가 이동하였다면 메시지를 다시 전달하여야 한다. 왜냐하면, 목적 노드는 이미 이동을 하여 이전의 기지국(MSS)에서 벗어나 다른 새로운 MSS에 연결되어 위치 상태가 변경되었기 때문이다. 그러나 이전의 MSS에서 다른 MSS로 이동한 호스트의 정보를 바꾸는 일정한 시간동안 이동 호스트는 네트워크로부터 끊겨져 있게 된다. 네트워크 계층을 위한 이동성을 고려한 라우팅 프로토콜에 대한 많은 연구가 진행되어 왔다[8,12,14].

이동 호스트가 네트워크로부터 끊겨져 있는 경우, 어플리케이션 레벨에서는 끊겨져 있는 이동 호스트에 대하여 체크포인트 알고리즘을 통한 체크포인트 요청이 발생하게 되고 기지국에 다시 연결될 때까지 이 체크포인트 요청에 대한 결과는 지연된다. 이런 과정에서 시스템의 일치성을 위해 소비되는 전체적인 시간이 증가하게 된다. 이런 단점을 보완하기 위한 이동성에 대한 연구가 진행되었다[8,14]. Prakash의 논문[14]과 Cao의 논문[8]에서는 이동 호스트가 기지국과 연결이 단절된 상태에서도 시스템의 일치성을 유지하기위한 알고리즘을 제안하였다. 이동 호스트가 현재 연결되어있던 기지국 MSS p에서 다른 기지국 MSS q로 이동하여 연결이 단절된 시간동안 이동 호스트는 이전에 연결되어있던 기지국 MSS p에 연결이 단절되어있는 상태(local snapshot)를 알린다. 만약 이동 호스트가 단절된 상태에서 체크포인트를 요청하면, 기지국 MSS p는 단절된 상태에서의 체크포인트라는 것을 알고 연결된 상태의 체크포인트와 구별하여 저장하고, 체크포인트 요청을 전달하기 위해 메시지 의존 정보를 사용한다. 또한 이동 호스트는 이동 호스트와 기지국의 채널을 통해 단절된 상태에서 메시지의 시퀀스를 보낸다. 이 때, 기지국 MSS p에서는 단절되어있는 동안 이동 호스트의 메시지를 버퍼에 저장하며 모든 메시지 시퀀스와 마지막 메시지를 알고 있다. 이동 호스트가 MSS p로부터 이동하여 MSS q에 연결되고 이동 호스트가 이전 기지국의 정보를 가지고 있다면, 이동 호스트는 MSS q를 통해 MSS p에 이동 호스트가 MSS q에 연결되었다는 메시지를 보내고, 이동 호스트가 이전 기지국의 정보를 가지고 있지 못했다면, 이동 호스트는 연결된 네트워크에 브로드캐스트하게 된다. 연결된 메시지를 받은 MSS p는 이동 호스트가 속한 기지국 MSS q에 이동한 이동 호스트에 대한 모든 정보(체크포인트, 저장된 메시지 등)를 보내고 단절된 상태에서 저장되었던 이동 호스트의 모든 정보를 없앤다. 이동 호스트에 관한 정보를 받은 MSS q는 이 정보를 이동 호스트에 보낸다. 이렇게 함으로써 네트워크에서 단절된 상태에서의 이동 호스트의 이동에 따른 시스템의 일치성을 유지할 수 있다.

Ⅲ. 모바일 환경을 위한 조정된 체크포인트 알고리즘

기존의 분산환경에서의 알고리즘은 이동성을 비롯한 이동 호스트의 안전하지 않은 저장 공간이나 짧은 배터리 수명으로 인해 이동환경에서 적합하지 않다[13]. 또한 이를 이동 환경에 바로 적용하게 되면 상당한 비용이 요구된다. 그러므로 이동 환경에 적합한 새로운 알고리즘이 필요하다. 본 논문에서는 Badrinath의 논문[2]에서 제안한 2-계층 기법을 기반으로 이동 환경에 적합한 조정된 체크포인트 기법을 제안한다.

1. 시스템 모델(System Model)

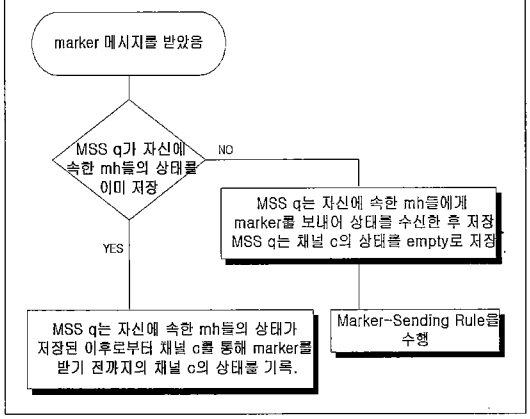
시스템은 여러 개의 기지국(MSS: Mobile Support Station)로 구성되어 있으며, 각각의MSS는 여러 개의 이동 호스트(mh)를 가지고 이를 관리한다. MSS에는 mh의 위치 정보와 상태정보를 저장한다. 이 mh들은 어느 한 순간에 이동성을 가진다. mh가 이동하였을 때, MSS는 이 정보를 가지고 있으며, 기존 MSS의 위치에서 이동한 MSS에는 이들의 정보를 저장한다. MSS간에는 유선 네트워크로 구성되어 있으며, 서로는 안정된 FIFO채널을 통해 통신을 한다. 또한 MSS와 mh간에는 일정한 시간이내에 직접적인 통신이 가능하다. 각각의 mh는 자신들의 상태를 저장하는 주기를 알고 있으며, 동기적인 저장을 한다. 또한 MSS간의 시스템의 일치성을 유지 하기 위해 조정된 체크포인트를 이용한다.

2. 모바일 컴퓨팅 환경을 위한 조정된 체크포인팅 알고리즘

MSS와 MSS간의 marker신호는 다음과 같은 메커니즘을 갖는다. 여기에서 mh는 자신들의 상태저장주기를 알고 있으며 marker 메시지는 송신자 mh1과 수신자 mh2를 포함한다. MSS는 자신에 속한 mh들에게 marker신호를 보내어 mh의 상태를 수신 받아 저장한다. mh는 MSS로부터 marker신호를 받으면 자신의 상태를 MSS로 전송한다. 채널 c의 상태 저장은 mh로 전달되는 메시지가 MSS를 통하게 되므로 가능하다.

알고리즘 2 : MSS에 의한 2-계층 알고리즘

```
I. Sending Rule for a MSS p.
   marker를 모든 채널 c를 통해 다른 MSS에게
   전송.
II. Receiving Rule for a MSS q.
 if (MSS q 가 자신에 속한 mh들의 상태를 아직
     저장하지 않았음) then
     begin
        MSS q는 자신에 속한 mh들에게 marker를 보
        내어 상태를 수신한 후 저장.
        MSS q는 채널 c의 상태를 empty로 저장.
        Marker-Sending Rule을 수행.
     end
else
     MSS q는 자신에 속한 mh들의 상태가
     저장된 이후로부터 채널 c를 통해
     marker를 받기 전까지의 채널 c의 상태를
     기록.
```



알고리즘 2는 MSS에 의한 2-계층 알고리즘을 나타낸다. 그림 3과 같이 2-계층 알고리즘이 적용되는 셀룰라 네트워크에서 기지국들(MSS p, MSS q, MSS r)은 유선 통신망으로 서로 연결되어 있으며, 각 기지국은 자신에 속한 무선 통신망으로 연결된 호스트들을 관리한다. 상위 레벨에서는 marker 신호가 각 기지국 사이에서 전달되고 하위 레벨에서는 각 기지국이 처음 marker를 받았을 때, 자신들에 속한 이동 호스트들의 정보를 저장하게 된다.

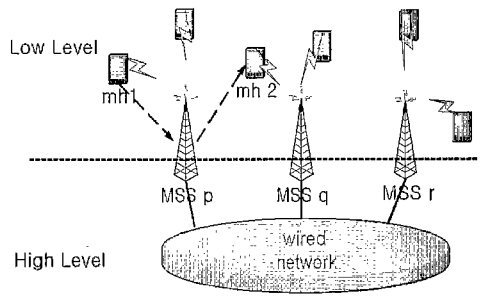


그림 3. 2-계층 알고리즘이 적용되는 셀룰라 네트워크

3. 이동성을 고려한 조정된 체크포인팅 알고리즘

알고리즘 2는 이동 호스트가 이동했을 때의 상황을 고려하지 않았다. 즉 이동 호스트가 기존의 영역

에서 벗어나 다른 지역으로 이동했을 때를 고려한 알고리즘이 필요하다. 알고리즘 3은 이를 위한 알고리즘을 나타내었다. mh의 이동성을 고려하기 위해서 자신의 셀에 존재하는 mh들은 MSS에 의해 관리(모든 mh의 상태와 메시지 로그를 저장, 관리)된다. 이동한 후에도 이동한 호스트를 관리한다. mh가 이동한 경우엔 가장 중요한 문제는 전체적인 시스템상태의 일치성을 유지해야 한다는 것이다. MSS는 이동한 mh의 위치정보를 알고 있기 때문에 이동한 mh로 보내온 메시지를 mh가 이동한 지역을 담당하는 MSS를 통해 전달할 수 있다. MSS는 marker 메시지를 처음 받을 때 자신에 속한 mh의 상태를 저장하기 위하여 자신(HA:Home Agent)의 영역에 있는 mh에게는 상태 요청을 하여 응답을 받지만 다른 MSS(FA:Foreign Agent) 영역으로 이동한 mh의 상태 저장을 위하여 FA에게 별도의 marker를 보내어 FA가 대신 이동한 mh의 상태를 저장하도록 한다. 메시지 로깅을 위해서도 다른 지역의 MSS(FA)로 이동한 mh에 대해서는 FA가 메시지 로깅을 수행하도록 한다. 그림 4는 MSS p가 조정자 역할을 수행하며, 체크포인트를 위해 marker를 보내는 것을 보여준다. 이때 MSS q(Home Agent)에 속해 있던 mh2가 MSS r(Foreign Agent)로 이동했을 때, MSS q는 별도의 marker를 MSS r에 전달하여 MSS q가 mh2의 상태를 저장하도록 한다. 그러나 아직 자신의 영역에 있는 mh1에 대해서는 직접 상태를 요청하고 응답받는다.

4. 이동성에 따른 일치성 유지를 위한 고려사항

전체적인 시스템의 일치성을 유지하는 것은 복잡하고 어려운 일이다. 특히 해당 기지국에 속해있던 이동 호스트가 이동하는 경우 시스템의 일치성에 문제를 일으킬 수 있다.

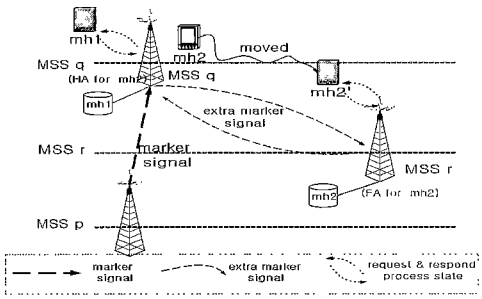


그림 4. 이동하는 mh의 상태 저장

Marker-Sending Rule에 의해 HA에서 marker를 받고 지역내의 이동 호스트의 상태를 저장하고 다른 기지국에 marker를 보냈다고 가정하자. 이미 상태를 저장한 이동 호스트(mh1)가 FA로 이동하고 다른 기지국(MSS r)에 marker가 도착하기 이전에 애플리케이션 메시지를 보내려 한다면 전체적인 일치성이 지켜지지 않는다. 이를 해결하기 위해 MSS r은 marker가 도착하여 상태를 저장할 때까지 애플리케이션 메시지를 전달하지 않고 연기를 시키면 일치성을 유지할 수 있다. 예를들면, 그림 5에서 marker를 받은 MSS p가 자신에 속한 이동 호스트들의 상태를 저장하였다. 그러나 MSS p에 속해 있던 이동 호스트 mh1이 marker가 MSS r에 도착하기 전에 MSS q로 이동한 상황에서 애플리케이션 메시지를 MSS r에 전달하기를 원하다면 전체적인 시스템의 일치성을 유지할 수 없다. 이 경우에 MSS q에 이동한 mh1가 보낸 애플리케이션 메시지는 전달하지 않고 지연시키다가 도착하면 지연된 메시지를 전달한다.

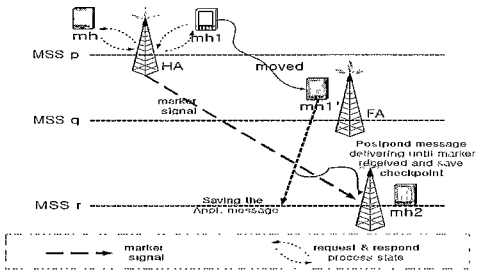


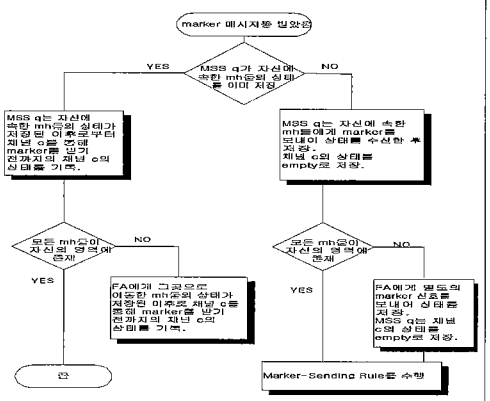
그림 5. 이동성을 고려한 조정된 체크포인트 알고리즘

알고리즘 3 : 이동성을 고려한 알고리즘

```
I. Sending Rule for a MSS p.
   marker를 모든 채널 c를 통해 다른 MSS에게
   전송.
II. Receiving Rule for a MSS q.
 if (MSS q 가 자신에 속한 mh들의 상태를 아직
     저장하지 않았음) then
     begin
        For all(자신에 속한 mh)
          if(자신의 영역에 존재)
             mh들에게 marker를 보내어 상태를
             수신한 후 저장.
             채널 c의 상태를 empty로 저장.
I. Sending Rule for a MSS p.
   marker를 모든 채널 c를 통해 다른 MSS에게 전송.
II. Receiving Rule for a MSS q.
 if (MSS q 가 자신에 속한 mh들의 상태를 아직
     저장하지 않았음) then
```

```
    begin
        For all(자신에 속한 mh)
          if(자신의 영역에 존재)
            mh들에게 marker를 보내어 상태를
            수신한 후 저장.
            채널 c의 상태를 empty로 저장.

          else if( 다른 영역에 이동)
            FA에게 별도의 marker를 보내어 대신
            mh의 상태를 저장하고, 채널 c의
            상태를 empty로 저장하도록 요청.
        Marker-Sending Rule을 수행.
    end
else
    begin
        For all(자신에 속한 mh)
          if(자신의 영역에 존재)
            MSS q는 자신에 속한 mh들의 상태가
            저장된 이후로부터 채널 c를 통해
            marker를 받기 전까지의 채널 c의
            상태를 기록.
          else if( 다른 영역에 이동)
            FA에게 그곳으로 이동한 mh들의
            상태가 저장된 이후로 채널 c를 통해
            marker를 받기 전까지의 채널 c의
            상태를 기록하도록 요청.
    end
```



Ⅳ. 비용 분석 및 성능 비교

mh와 MSS간에 일어나는 비용은 무선과 유선환경으로 나눌 수 있다. 무선 환경에서는 유선환경에 비하여 일반적으로 통신 오버헤드가 많기 때문에 이를 고려한 비용 분석이 필요하다. 비용 분석에 사용된 페러미터는 다음과 같다.

- n : mh(mobile host: 이동 호스트)의 개수
- m : MSS(Mobile Support Station: 기지국)의 개수
- Cwireless : 무선 통신망을 통해 MSS와 mh사이에서 메시지를 보내는데 소요되는 비용
- Cfixed : 기지국간에 메시지를 보내는데 소요되는 비용
- p : 이동율(mh가 FA로 이동했을 가능성)

일반적으로 이동 호스트인 mh의 개수가 더 많으므로 n >> m라고 가정한다. 기존의 조정된 체크포인팅 알고리즘을 그대로 이용하여 셀룰라 네트워크에 적용하면 다음과 같은 비용이 필요하다.

$$n(n-1)*(2C_{wireless}+C_{fixed})+n(n-1)*p*C_{fixed}$$

기존의 조정된 체크포인팅 알고리즘은 이동 호스트가 네트워크상에 존재하는 모든 이동 호스트들에게 marker신호를 보내야 하므로 자신을 제외한 모든 이동 호스트에게 메시지를 보내게 된다. 이 때, 송신측 이동 호스트에서 기지국으로 $C_{wireless}$, 송신측 기지국에서 수신측 기지국으로 C_{fixed}, 수신측 기지국에서 수신측 이동 호스트로 $C_{wireless}$의 비용이 필요하므로 유선 통신 비용은 $n(n-1)*(2C_{wireless}+C_{fixed})$ 와 같다. 이동 호스트가 이동할 경우, 이동 호스트는 자신이 속한 기지국뿐만 아니라 모든 이동 호스트에게 메시지를 보내게 된다. 이때 드는 비용은 $n(n-1)*p*C_{fixed}$와 같다. 기존의 조정된 체크포인트 알고리즘은 이동 호스트의 수의 제곱에 비례하여 통신 비용이 증가함을 알 수 있다. 반면에 제안한 2-계층기법의 알고리즘을 이용할 때의 비용은 다음과 같다.

$$m(m-1)*C_{fixed}+2*n*C_{wireless}+2(m-1)*n*p*C_{fixed}$$

제안한 2-계층 알고리즘은 상위 계층(High Level)에서 기지국 사이의 유선 통신망을 이용하여 marker 신호를 보내고 하위 계층(Low Level)에서 각 기지국마다 자신의 영역에 있는 이동 호스트에게만 무선 통신망을 통하여 marker신호를 보내어 통신 비용을 줄인다. $m(m-1)*C_{fixed}$는 기지국간의 유선 통신 비용을 나타내며, $2*n*C_{wireless}$ 는 기지국과 기지국 내의 이동 호스트간의 무선 통신 비용이다. 이동 호스트로 통신을 할 경우, 기지국은 marker라는 일종의 체크포인트 요청 메시지를 이동 호스트에게 보내고 이동 호스트는 이에 요청에 대한 답변을 보내

는데 $2*n*C_{wireless}$의 무선 통신 비용이 필요하다. 이동 호스트가 HA에서 FA로 이동할 경우에 드는 비용은 $2(m-1)*n*p*C_{fixed}$와 같다. 이동 호스트의 이동성이 클수록 통신 비용은 증가함을 알 수 있다.

기존 알고리즘과 제안하는 2-계층 기법 알고리즘의 성능을 비교하면 그림 6과 같다. 그림 6은 이동 호스트가 증가함에 따른 통신 비용을 비교한 것이다. 이 때, MSS의 수는 10으로, $C_{wireless}$는 5.0, C_{fixed}는 1.0, 이동율 p는 0.5로 가정하였고 이동 호스트의 수는 $1 \le n \le 20$으로 하였다. 가정에서 이동율 p를 0.5로 가정한 이유는 그림 9에서 나타난 결과에서 일반적인 값을 선택하였고, 이동 호스트의 개수와 기지국의 개수는 임의로 증가하면서 시뮬레이션한 결과 성능의 많은 차이를 보여 일부분만을 나타내었다. 그림 6의 결과를 보면 기존의 환경에서는 상당히 많은 비용이 필요로 함을 알 수 있다. 그러나 이동 호스트가 적을 때 제안한 알고리즘 성능이 떨어진다. 그 이유로 기존의 알고리즘은 이동 호스트의 수가 적을 때, 보내야 할 메시지의 수는 적다. 그러므로 통신 비용이 적게 든다. 그러나 제안한 2-계층 알고리즘에서는 이동 호스트의 수가 적었을 때에도 기지국과 통신을 해야 하고 다시 기지국은 다른 이동 호스트가 위치한 셀 내의 기지국과 통신을 수행하며, 이 기지국은 다시 목적 이동 호스트에 통신을 하게 되므로 상대적으로 많은 통신 비용이 소요된다. 그러나 이런 상황은 일반적으로 이동 호스트의 수가 기지국에 비해 훨씬 많으므로 거의 발생하지 않는다. 그림 7은 MSS의 수와 mh의 수의 상대적인 비율에 따른 메시지의 수를 보여주며, 이동율 p는 0.5, $C_{wireless}$는 5.0 그리고 C_{fixed}는 1.0으로 가정하였다. 이 경우에도 기존의 알고리즘은 상당히 많은 통신 오버헤드가 필요하다. 그러나 제안한 알고리즘인, 저비용 조정된 체크포인팅 알고리즘에서는 유선 통신망인 기지국을 이용함으로써 통신 오버헤드를 줄일 수 있다.

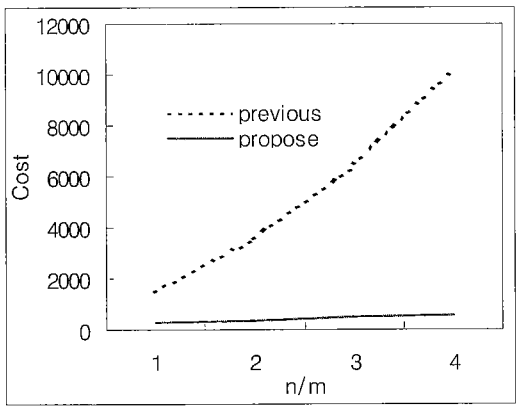


그림 7. n/m에 따른 통신 비용

그림 8은 무선 통신 비용과 유선 통신 비용의 상대적인 비율에 따른 통신 비용를 보여주며, mh의 수는 20, MSS의 수는 10, 이동확률은 0.5로 그리고 이동율 p는 0.5로 가정하였다. 무선 통신과 유선 통신의 상대적 통신 비용이 증가할수록 기존의 알고리즘은 통신 비용이 급격히 증가되지만 제안된 2-계층 방식은 상대적으로 완만히 증가함을 알 수 있다. 이동 환경과 고정 네트워크 환경에서의 차이가 비용율이 증가할수록 많이 나는데 이는 무선 통신망의 비용이 상대적으로 커지면 커질수록 통신 비용이 증가함을 알 수 있다. 그림 9는 이동확률을 $0 \le p \le 1$로 가정하였으며, $C_{wireless}$는 5.0, C_{fixed}는 1.0, mh의 수는 20, MSS의 수는 10으로 가정하였다. 이동할 확률이 높을수록 많은 비용이 듦을 알 수 있다.

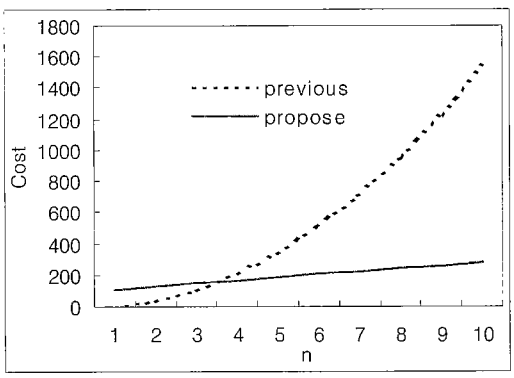


그림 6. n(mh의 개수) 에 따른 통신 비용

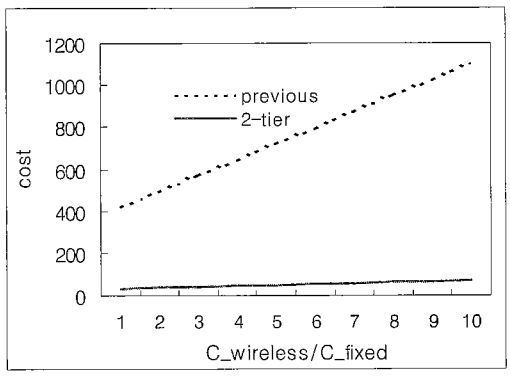


그림 8. $C_{wireless}$ / C_{fixed} 에 따른 통신 비용

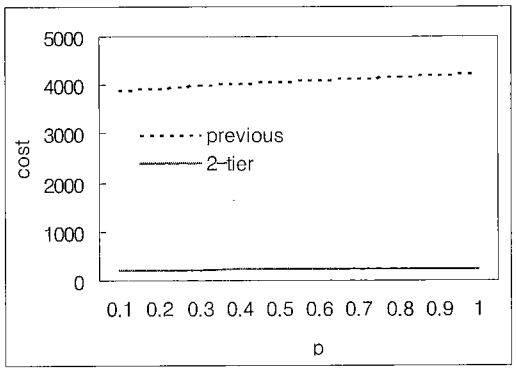


그림 9. p(mobility ratio)에 대한 통신 비용

V. 결 론

모바일 컴퓨팅 시스템은 이동 호스트들(mobile hosts)과 기지국(mobile support station)으로 구성되어 있으며 이들 서로는 네트워크 상에 무선으로 통신을 한다. 모바일 컴퓨팅은 핸드오프나 부족한 저장공간, 그리고 낮은 대역폭 등의 해결해야 될 많은 새로운 문제에 직면해있다. 이들의 문제로 인해 전통적인 유선환경에서의 체크포인팅 알고리즘은 적합하지 않다. 그러므로 모바일 컴퓨팅 시의 문제점을 해결하기 위한 새로운 알고리즘이나 전통적인 알고리즘의 변형이 필요하다. 특히 모바일 환경에서 빈번하게 발생하는 통신 오버헤드를 줄이기 위한 새로운 알고리즘의 필요성이 절실하다.

본 논문에서는 모바일 환경에 적합하고 무선 통신망에서 통신 오버헤드를 최소화할 수 있는 새로운 알고리즘을 제안하였다. 이것은 2-계층 알고리즘으로 기지국에 속한 이동 호스트들이 요청하는 메시지를 해당 기지국에서 서비스를 하며 기존 알고리즘에서 이동 호스트들에게 모두 보냈던 메시지들을 기지국에서 대신 관리하게 된다. 이렇게 함으로써 통신 오버헤드와 통신 비용을 줄일 수 있으며, 또한 기지국 중에 조정자(coordinator)를 두어 전체적인 시스템의 일치성을 유지한다. 기존의 조정된 체크포인팅 알고리즘은 각 노드마다 메시지를 송수신하므로 체크포인팅 메시지 수의 복잡도가 이동 호스트의 제곱($O(n^2)$)이지만, 제안된 2-계층 알고리즘에서는 메시지수의 복잡도가 기지국의 수와 이동 호스트의 수의 제곱의 합($O(n+m^2)$)이다. 그러므로 기존의 알고리즘에 비해 제안한 알고리즘은 상대적으로 많은 통신 비용을 감소시킬 수 있다.

참 고 문 헌

[1] Arup Acharya, B. R. Badrinath, "Checkpointing Distributed Application on Mobile Computers", the 3rd Intl, *Conf. On Parallel and Distributed Information Systems*, Sept. 1994.

[2] B. R. Badrinath, Arup Acharya and Tomasz Imielinski. "Designing distributed algorithms for mobile computing networks", *Computer Communications*, Vol. 19, No. 4, 1996.

[3] C. Critchlow and K. Taylor. "The inhibition spectrum and the achievement of causal consistency". *Technical Report TR 90-1101, Cornell University*, Feb. 1985.

[4] E.N. Elnozahy, D.B. Johnson and W. Zwaenepoel. "The Performance of Consistent Checkpointing", *In Proc. of the Eleventh Symposium on Reliable Distributed Systems*, pp 39-47, Oct. 1992.

[5] Elnozahy, D. B. Johnson and Y. M Wang, A "Survey Roll-Recovery Protocols in Message-Passing Systems", *Technical Report CMU-CS-96-101, Carnegie Mellon University*, 1996.

[6] F. Teraoka, Y. Yokote, and M. Tokoro. "A Network Architecture Providing Host Migartion Transparency", *Proc. Of ACM SIGCOMM '91*, Sep. 1991.

[7] G. Janakiraman and Y. Tamir. "Coordinated Checkpointing-Rollback Error Recovery for Distributed Shared Memory Multicomputers", *In Proc. of the 13th Symp. on Reliable Distributed Systems* (SRDS'94), pp 42-51, Oct. 1994.

[8] Guohong Cao and Mukesh Singhal, "Low-Cost Checkpointing with Mutable Checkpoints in Mobile Computing Systems", *Proc. of the The 18th International Conference on Distributed Computing Systems*, 1998.

[9] J. Ioannidis, D. Duchamp, and G. Q. Maguire "IP-based protocols for mobile internetworking", *In Proc. Of ACM SIGCOMM Symp. on Communication, Architectures and Protocols*, pp 235-245, Sep. 1991.

[10] Kyue-Sup Byun and Jai-Hoon Kim, "Determining Checkpointing Intervals for Fault Tolerant Real-Time Systems", *2000 International Conference on Parallel and Distributed Processing Techniques and Applications* (PDPTA'2000), Las Vegas, Nevada, Jun. 2000.

[11] K. Chandy and L. Lamport. "Distributed Snapshots: Determining Global States of Distributed Systems", *ACM Transactions on Computer Systems*, Vol. 3, No 1, , pp 63-75, Feb. 1985.

[12] P. Bhagwat and C.E. Perkins, "A Mobile Networking System Based on Internet Protocol(IP)", *In Proc. USENIX Symp. Mobile and Location-Independent Computing*, pp. 69-82, Aug. 1993.

[13] P. Krishna, N.H Vaidya, and D.K. Pradhan. "Recovery in Distrubuted Mobile Environments", *IEEE Workshop on Advances in Parallel and Distributed System*, Oct. 1993.

[14] R. Parakash and M. Singhal. "Low-Cost Checkpointing and Failure Recovery in Mobile Computing Systems", *IEEE Trans. on Parallel and Distributed System*, pages 1035-1048, Oct. 1996.

[15] R. Koo and S. T. H. Yang. "Checkpointing and rollback-recovery for distributed system", *IEEE Trans. Software Eng.*, SE-13(1):23-31, Jan. 1987.

변 계 섭(Kyue-Sup Byun) 학생회원

1999년 2월 : 아주대학교 정보 및 컴퓨터 공학부 졸업(공학사)

2000년 2월 : 아주대학교 정보통신전문대학원 석사과정

<주관심 분야> 이동 컴퓨팅, 결함허용 시스템, 실시간 시스템

김 재 훈(Jai-Hoon Kim) 정회원

1984년 : 서울대학교 제어계측공학과 (학사)

1993년 : Indiana University, Computer Science (석사)

1997년 : Texas A&M University, Computer Science (공학박사)

1984년~1991년 : 대우통신㈜ 컴퓨터연구실, 팀장

1995년~1997년 : Texas A&M University, Graduate Research Assistant

1997년~1998년 : 삼성전자㈜ 컴퓨터시스템팀, 수석연구원

1998년~현재 : 아주대학교 정보통신전문대학원, 교수

<주관심 분야> 분산시스템, 이동컴퓨팅, 실시간시스템