이동 컴퓨팅 환경에서 갱신가능 메시지를 이용한 캐쉬 일관성 유지 정책

박 성 배[†] · 황 부 현[††]

요 약

미래의 이동 컴퓨팅 환경에서는 휴대 가능한 랩탑 등의 이동호스트가 무선 통신 채널을 통하여 데이타베이스에 접근하는 것이 일반화될 것이다. 이동호스트는 무선 통신의 낮은 대역폭으로 인한 문제를 해결하기 위하여 자주 사용하는 데이타를 캐쉬하며, 캐쉬 데이타의 정확성을 위하여 캐쉬 일관성을 유지하여야 한다. 캐쉬 일관성은 이동호스트의 무제한적인 이동과 이동호스트 지원 서버와의 빈번한 통신 단절로 인하여 위배될 수 있다. 따라서 이동호스트를 효율적으로 지원하는 캐쉬 일관성 유지 정책이 요구된다.

본 논문에서는 2단계 완료 프로토콜을 지원하는 중복 데이타베이스의 이동 컴퓨팅 환경에서 갱신가능 메시지를 이용한 정책을 제안한다. 이 정책은 이동호스트가 다른 셀로 이동할 때 캐쉬 일관성이 위배되는 문제를 해결하기 위하여 갱신가능 메시지를 사용한다. 갱신가능 메시지의 이용은 수신하지 못한 무효화 메시지의 갱신 데이타만을 찾아 다시 캐쉬하도록 지원한다. 결과적으로 제안된 정책은 임의의 데이타 변경만으로 모든 캐쉬 데이타의 제거없이 캐쉬 일관성을 유지하기 때문에 대역폭을 효율적으로 사용하는 장점을 지닌다.

A Strategy using Updatable Message for Retaining the Cache Consistency in the Mobile Computing Environment

Seongbae Park[†] · Buhyun Hwang[††]

ABSTRACT

In the future mobile computing environment, it will be generalized that a mobile host with portable Laptop or Palmtop accesses to database through wireless communication channel. For solving the problems which are caused from low bandwidth of wireless communication, mobile hosts cache the datum used frequently on them. Thus the mobile hosts must always retain the cache consistency for the correctness of the cached datum. The cache consistency is mainly affected from the unrestricted host mobility and the disconnection of the communication between mobile hosts and mobile support station. In this situation, the strategy to retain efficiently cache consistency is required.

In this paper, we propose a strategy using updatable message for retaining the cache consistency, in the mobile computing environment, with replicated database supporting 2 phase commit protocol. This strategy makes use

※이 논문은 1997년도 한국과학재단의 학술 조성비 지원에 의하여 연구되었음.

† 정 회 원:순천공업전문대학 전자계산과 부교수
†† 정 회 원:전남대학교 전산학과 교수

논문접수:1997년 3월 26일, 심사완료:1997년 5월 26일

of a updatable message to resolve the cache consistency problem, since mobile host crosses the boundary of cell. In using of this updatable message, it find out and cache only the update data of missing invalidation message. As a result, the proposed strategy utilizes network bandwidth efficiently because it is not need to delete all the cached datum with any datum change.

1. 서 론

휴대용 컴퓨터의 대중화와 무선 통신의 급격한 발전은 현재의 유선 컴퓨팅 환경이 이동 컴퓨팅 환경으로 바뀔 것을 예고하고 있다[2, 6, 10]. 이동 컴퓨팅 환경은 기존의 유선 컴퓨팅 환경과 비교할 때 다음과 같은 제약이 따른다. 첫째, 무선 통신은 낮은 대역폭을 갖는다. 낮은 대역폭으로 인하여 이동호스트가 MSS(Mobile Support Station)에게 빈번히 데이타를 요청할 때 데이타 통신 채널에 대한 점유 경쟁이 발생된다. 그러므로 낮은 대역폭의 효율적인 활용하기 위하여 자주 사용하는 데이타를 캐쉬하는 것이 바람직하다[3, 5, 12, 13]. 둘째, 이동호스트의 자유로운 이동 및 통신 단절은 데이타 관리의 어려움을 가중시킨다. 이동호스트의 이동 때문에 MSS는 이동호스트의 위치와 상태에 관한 정확한 정보를 유지할 수 없다. 또한 이동호스트의 사용 기간을 연장하기 위해서 사용자가 의도적으로 통신을 단절시키거나, 예측할 수 없는 통신 단절이 발생할 수 있다. 이와 같은 이동호스트의 이동과 통신 단절은 캐쉬 일관성에 영향을 미친다[1, 4, 7, 8].

이동호스트의 낮은 대역폭을 효율적으로 사용하기 위하여 중복 데이타베이스를 기반으로 한 이동 컴퓨팅 환경에서 여러 가지 캐쉬 일관성 유지 정책들이 제안되었다. [4, 7]은 이동호스트가 MSS와 통신이 단절된 후 다시 연결을 시도할 때, [8]은 이동호스트가 다른 셀로 이동하였을 때, 캐쉬 일관성을 유지하기 위한 정책을 제안하였다. 이 정책들은 갱신 데이타에 대한 무효화 메시지를 주기적으로 방송하며, 중복 데이타의 모든 사본들은 동시에 갱신될 수 없음을 전제로 하였으며, 캐쉬 일관성이 위배될 때 모든 캐쉬 데이타를 제거하였다.

이 논문에서는 2PC(2 Phase Commit Protocol)를 지원하는 중복 데이타베이스에서 이동호스트가 다른 셀로 이동할 때 캐쉬 일관성의 위배에 대한 문제를 해결하기 위하여 갱신가능 메시지(UMRCC: Updatable Message for Retaining the Cache Consistency)를 이용하는 정책을 제안한다. 여기에서 MSS는 무효화 메시지, 갱신가능 메시지 또는 응답 메시지를 방송하며, 이동호스트는 다른 셀로 이동할 때 무효화 메시지를 수신하지 못할지라도 갱신가능 메시지를 수신한다.

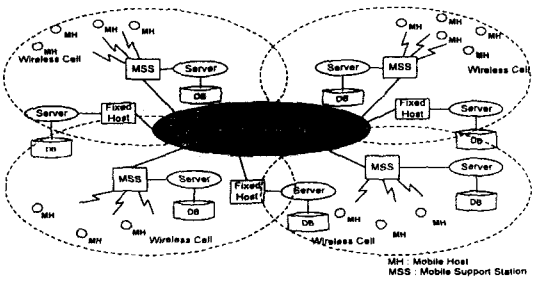
UMRCC에서는 이동호스트가 다른 셀로 이동할 때마다 무효화 메시지의 수신 여부를 MSS에게 확인하는 문제를 해결하고, 이동호스트가 무효화 메시지를 수신하지 못하여 캐쉬 일관성이 위배되었을 때 모든 캐쉬 데이타 제거하는 문제를 해결한다. 이동호스트가 다른 셀로 이동하면 갱신가능성이 있는 데이타만을 MSS에게 요청한다. 이동호스트는 무효화 메시지를 수신하지 못하여도 갱신가능 메시지를 이용하여 갱신 데이타에 대한 정보를 다시 받을 수 있으므로 캐쉬 데이타를 제거하는 경우가 발생하지 않는다. 이러한 관점에서 갱신가능 메시지의 이용은 캐쉬 일관성 위배를 예방할 수 있고, 캐쉬 데이타의 제거로 인하여 다시 캐쉬하는 오류를 범하지 않게 된다. 그러므로 UMRCC는 대역폭을 보다 효율적으로 사용할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 제안한 정책에 대한 시스템 모델을 제시하고, 3장에서는 캐쉬 일관성을 위배하는 경우와 불필요하게 캐쉬 데이타를 제거하는 경우의 문제를 정의한다. 4장에서는 UMRCC 정책을 제안하고 5장에서는 제안한 정책의 비용 평가를 한다. 6장 관련 연구에서는 기존 연구와 비교하여 UMRCC의 차이점을 설명한다. 마지막으로 7장에서는 결론을 기술한다.

2. 이동 컴퓨팅 모델

이동 컴퓨팅 모델은 (그림 1)과 같으며 이동호스트와 고정호스트를 갖는다. MSS는 이동호스트와 통신할 수 있는 인터페이스를 가지고 있는 고정호스트이

며, 한 지리적 통신 영역에 위치하는 이동호스트들과 통신이 가능하다. 셀은 MSS와 통신할 수 있는 지리적 영역을 의미한다. 셀은 물리적으로 다른 셀과 중첩될 수 있으며, 비록 셀이 중첩되었을지라도 이동호스트는 단지 하나의 MSS와 통신이 가능하다[2, 3, 6, 8].



(그림 1) 이동 컴퓨팅 모델
(Fig. 1) The mobile computing model

모델에서의 이동호스트는 무선 통신의 낮은 대역폭을 효율적으로 사용하기 위하여 캐쉬를 사용한다. 또한 이동호스트는 캐쉬 데이타의 일관성을 유지하기 위하여 MSS로부터 주기적으로 방송된 무효화 메시지에 의해 캐쉬 데이타를 갱신하고, 캐쉬되지 않은 데이타는 MSS에게 요청하여 캐쉬한 후 연산에 이용한다. 고정호스트는 중복 데이타베이스에 기반을 두고 낙관적인 스케줄링 기법과 2PC 프로토콜을 사용한다고 가정한다. 중복 데이타베이스는 데이타 항목의 다수 사본이 다중 사이트에 기억된 분산 데이타베이스이다. 중복 데이타베이스를 사용하는 주요 이유는 위험성이 높은 데이타를 다수 사이트에 기억시켜 임의의 사이트에서 고장이 발생하여도 연산을 수행할 수 있기 때문에 시스템의 가용성을 증가하기 때문이다[9].

2.1 분산 중복 데이타베이스의 동시성 제어 기법

중복 데이타베이스에서 동시성 제어 기법은 ROWA (Read One Write All) 프로토콜과 정족수 합의 프로토콜(QCP:Quorum Consensus Protocol)이 있으며, 트랜잭션 스케줄링 기법에는 낙관적인 방법과 비관적인 방법이 있다[9, 11].

ROWA 프로토콜에서 판독 연산은 자신의 사이트에 있거나 가장 가까운 사이트에 위치한 하나의 데이타 사본을 판독하고, 기록 연산은 모든 데이타 사본에 대한 갱신을 요구한다. QCP는 판독 연산, 기록 연산이 모두 정족수를 만족하여야 하며, 이들 정족수의 합은 전체 사이트의 수보다 크다. QCP는 기록 연산 시 전체 사본을 갱신할 필요가 없기 때문에 ROWA보다는 시간적 부담은 적으나 판독 연산을 위하여 정족수에 해당하는 사본들을 읽어야만 정확한 데이타를 찾을 수 있다.

스케줄링의 비관적인 방법은 트랜잭션의 각 기록 연산을 데이타 사본이 존재하는 모든 사이트로 전송하여 해당 지역 트랜잭션 관리자에 의해 수행하는 방법이고 낙관적인 방법은 한 사이트에서 트랜잭션의 수행을 완료한 후 그 결과를 각 사이트에 vot-req (vote-request)와 함께 전송하여 갱신 결과를 반영하는 방법이다.

2.2 원자적인 완료 프로토콜

트랜잭션 완료 프로토콜에는 2PC 방법이 있다. 2PC 프로토콜은 조정자가 어떤 트랜잭션의 수행을 종료한 후 각 사이트에 vot-req(vote request)하며, 각 사이트는 직렬성에 위배되지 않으면 "yes"를 응답하고 대기 상태(ready state)에 있게 된다. 조정자는 모든 사이트로부터 "yes"를 응답받으면 데이타베이스에 갱신 결과를 반영하고 각 사이트에 완료 메시지를 보낸다. 각 사이트는 완료 메시지를 받은 후 그 결과를 데이타베이스에 반영한다. 그러므로 2PC는 트랜잭션의 원자적인 수행을 지원한다. 또한 모든 고정호스트의 데이타베이스에 갱신을 반영하는 시점은 다를 수 있다[9].

3. 문제 정의

이동호스트의 캐쉬 일관성이 위배되는 경우는 다음 두가지를 고려할 수 있다. 첫 번째는 이동호스트가 셀의 경계를 벗어나 다른 셀로 이동하여 무효화 메시지를 수신하지 못하였을 때이다. 두 번째는 고정 통신망에서 전송 지연으로 인하여 고정호스트는 서로 다른 시간에 갱신 데이타의 정보를 받게되므로써 다른 무효화 메시지를 방송할 때 이다.

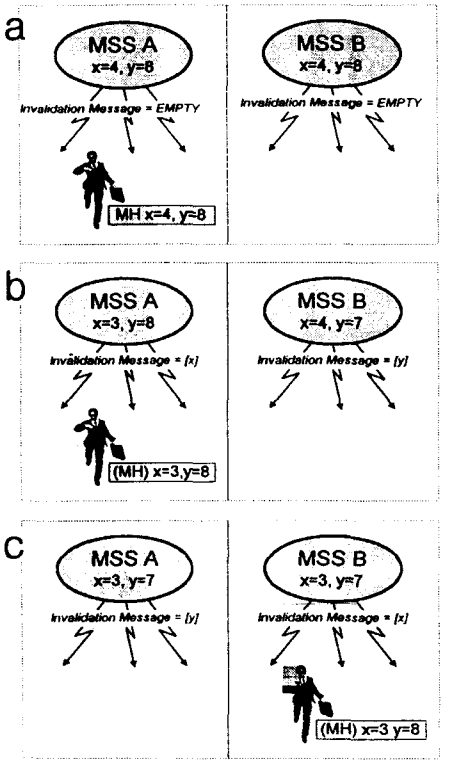
이 장에서는 2PC를 지원하는 중복 분산 데이타베이스를 기반으로 하고 MSS가 주기적으로 무효화 메

시지를 방송하는 이동 컴퓨팅 환경에서 캐쉬 일관성이 위배되는 문제를 정의한다.

3.1 캐쉬 일관성 위배(문제점 1)

이동호스트가 다른 셀로 이동하여 자신이 캐쉬하고 있는 데이타가 갱신되었음을 나타내는 무효화 메시지를 수신하지 못한다면 캐쉬 일관성이 위배된다.

(그림 2)의 시간 a에서 MSS A, B의 데이타베이스와 이동호스트 MH의 캐쉬가 x와 y에 대해 일관성을 유지하고 있으며, 방송 시점에서 데이타에 대한 갱신이 없으므로 각 MSS의 무효화 메시지는 〈EMPTY〉이다. 시간 b에서 서로 다른 트랜잭션이 MSS A에서는 x를, MSS B에서는 y를 갱신하고, 각 MSS는 그 결과를 방송한다. MH는 MSS A로부터 x에 대한 무효화 메시지를 받은 후 MSS B의 셀로 이동한다. 시간 c에서 각 MSS는 다른 MSS로부터 전송받은 갱신 결과



(그림 2) 기존 정책의 캐쉬 일관성 위배

(Fig. 2) The violation of cache consistency on existing strategy

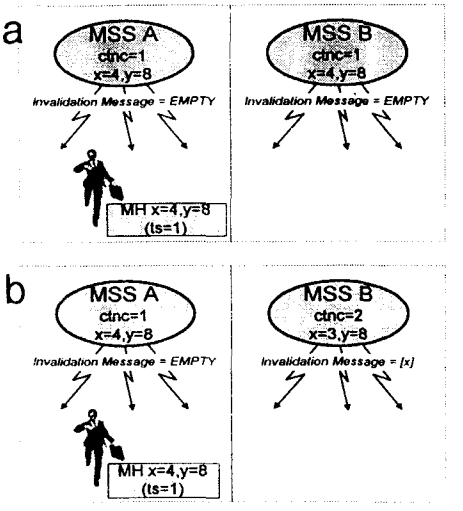
를 데이타베이스에 반영한 후 방송한다.

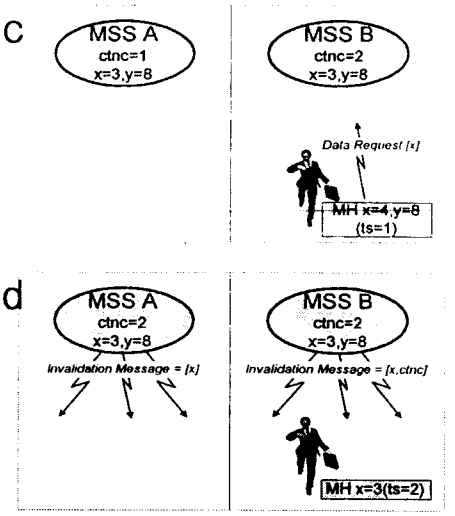
고정 통신망의 중복 분산 데이타베이스가 2PC를 지원하면 모든 고정호스트에서 데이타에 대한 갱신이 전송 지연의 차이로 인하여 동시에 이루어지지 않는다. 만약 완료 메시지가 한 방송 주기 내에 모든 MSS에 전파된다면 각 MSS는 갱신 결과를 데이타베이스에 반영하고 그 결과를 방송한다. 그러므로 모든 MSS의 무효화 메시지는 같고 이동호스트의 캐쉬 일관성은 유지된다. 그러나 어떤 MSS가 방송 주기 내에 완료 메시지를 받지 못하면 그 MSS와 다른 MSS의 무효화 메시지의 내용이 서로 다를 수 있다((그림 2)의 b와 c 참조). 이 때 이동호스트의 캐쉬 일관성은 위배된다.

3.2 불필요한 캐쉬 제거(문제점 2)

이동호스트가 캐쉬 일관성이 위배되었음을 인식하면 모든 캐쉬 데이타를 제거하므로 시스템 성능이 저하된다. 이 문제점은 [8]에서 제안한 정책에 기준을 두고 기술한다.

(그림 3)의 시간 a에서 MSS A와 B의 데이타베이스 그리고 MH의 캐쉬는 x와 y에 대해 일관성을 유지한다. 각 MSS는 갱신 카운터 ctnc를 유지하고 MH는 ctnc와 동일한 값의 타임스탬프 ts를 갖는다. 시간 b의 MSS B에서 어떤 트랜잭션이 x를 갱신하면 MSS B는 ctnc를 1 증가시키고 무효화 메시지를 방송한다. 이 때 MSS A에서는 〈EMPTY〉를 방송한다. 시간 c





(그림 3) 기존 정책의 불필요한 캐쉬 제거
(Fig. 3) Unnecessary cache dropping with existing strategy

에서 MH가 다른 셀로 이동한 후 무효화 메시지의 수신 여부를 확인하기 위해 MSS에게 어떤 데이타를 요청한다. 시간 d에서 MSS B가 ctnc와 x를 MH에게 보내주면 MH는 ts를 ctnc와 일치시키고 x를 캐쉬한다. MH는 자신의 ts와 MSS B의 ctnc가 일치하지 않기 때문에 무효화 메시지를 수신하지 못했음을 인식하고 캐쉬 데이타를 모두 제거한다. 이 때 MH는 캐쉬 데이타 y가 일관성을 유지하고 있음에도 불구하고 캐쉬로부터 제거한다. 또한 MH가 새로운 셀로 이동한 후 그 셀의 MSS에게 데이타를 요청하지 않으면 3.1절에서 제시된 것과 동일한 문제점이 발생한다.

4. 캐쉬 일관성 유지 정책

이 장에서는 3장에서 제시한 문제점을 해결하기 위하여 캐쉬 일관성을 유지하기 위하여 갱신가능 메시지(UMRCC)를 이용한 정책을 제안한다. 제안할 정책을 위하여 다음과 같이 가정한다.

트랜잭션은 갱신 트랜잭션과 판독 전용 트랜잭션(질의:query)으로 구분된다. 갱신 트랜잭션은 트랜잭션 내에 기록 연산이 하나 이상 포함된 경우이고, 판독 전용 트랜잭션은 오직 읽기 연산들로만 구성된다. 본 논문의 이동호스트는 판독 전용 트랜잭션만을 수행하고, 갱신 트랜잭션은 고정호스트에서 수행된다고 가정한다.

이동호스트가 데이타를 캐쉬하는 방법에는 MSS에게 데이타를 요청하면 MSS가 바로 그 데이타 값을 전송하는 방법과 MSS가 무효화 메시지를 방송할 때 함께 보내는 방법이 있다. 후자의 경우 MSS의 방송 시점에 병목현상을 발생할 수 있으므로 이 논문에서는 전자의 경우를 가정한다.

또한 MSS가 무효화 메시지를 방송하는 방법에 따라 다음과 같이 두 가지 경우로 분류할 수 있다. 첫째, 갱신 결과를 모두 방송하는 방법과, 둘째, 갱신 결과에 대한 정보(비트 열 또는 갱신 항목 목록)를 방송하면 각 이동호스트는 자신에 해당되는 정보를 MSS에게 보내며, MSS는 이들을 종합하여 해당 데이타들을 방송하는 방법이 있다. 이 논문에서는 대역폭을 효율적으로 사용할 수 있는 둘째 방법을 적용하는 것을 가정한다.

4.1 메시지 정의

캐쉬 일관성을 유지하기 위하여 MSS는 이동호스트에게 무효화 메시지를 방송한다. 무효화 메시지의 정의는 다음과 같다.

【정의 1】 무효화 메시지(Invalidation Message)

무효화 메시지는 한 방송 주기내에 MSS에서 수행된 트랜잭션이 갱신한 결과를 이동호스트의 캐쉬에서도 갱신될 수 있도록 MSS가 방송하는 메시지이다.

【정의 2】 〈EMPTY〉

〈EMPTY〉는 방송 주기 동안에 갱신 데이타가 존재하지 않을 때 MSS가 방송하는 무효화 메시지이다.

기존 연구에서는 방송 주기 동안에 임의의 데이타가 갱신되면 그 결과를 이동호스트에 방송하고 이동호스트는 무효화 메시지 중에서 자신이 캐쉬한 데이타가 있으면 캐쉬 내용을 갱신하여 캐쉬 일관성을 유지한다. 만약 방송 주기 동안에 갱신 데이타가 없으면 MSS는 〈EMPTY〉를 방송한다.

이 논문에서는 방송 시점에 무효화 메시지 외에도 다음 정의와 같은 갱신가능 메시지를 포함한다.

【정의 3】 갱신가능 메시지 (updatable message)

갱신가능 메시지는 완료 프로토콜을 수행중에 참가자가 "yes" 메시지를 조정자에게 보낸 후 완료하고자 하는 트랜잭션이 갱신할려는 데이타의 리스트이

다. 즉, 리스트 내의 데이타가 갱신가능성이 있음을 미리 이동호스트에게 알리는 메시지이다.

또한 MSS는 방송 주기 동안에 다음과 같은 응답 메시지를 이동호스트에 전송할 수 있다.

【정의 4】 응답 메시지 (*response message*)

응답 메시지는 이동호스트가 MSS에게 데이타를 요청할 때 MSS가 이동호스트에게 해당 데이타를 전송하는 메시지이다.

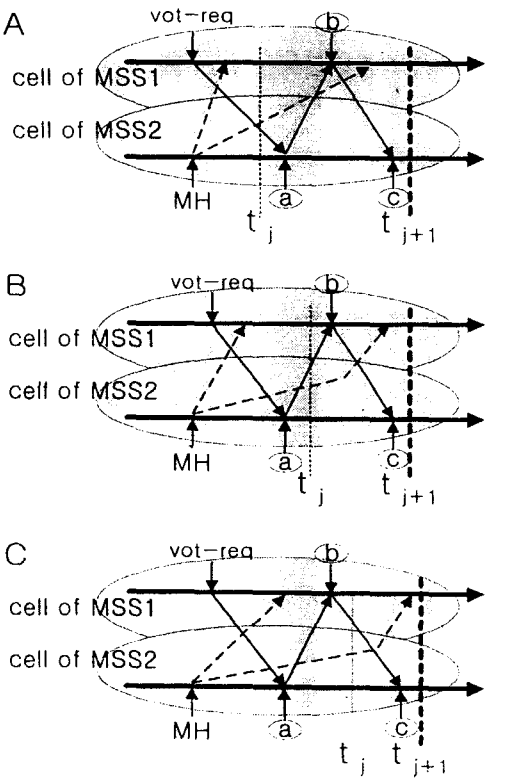
이 논문에서 MSS로부터 방송되는 내용은 무효화 메시지와 갱신가능 메시지를 모두 포함할 수 있으며, 경우에 따라 이들 메시지 중에서 일부만을 방송할 수 있다. 만약, 위 두 메시지 중 어느 것도 방송할 필요가 없으면 MSS는 〈EMPTY〉를 방송한다. 또한 응답 메시지는 이동호스트의 연산에 필요한 데이타가 자신의 캐쉬에 없어서 MSS에게 데이타를 요청할 때와 이동호스트가 갱신가능한 데이타의 정확한 값을 캐쉬하기 위하여 MSS에게 데이타를 요청할 때에 따라 구분된다. 전자의 경우, MSS는 이동호스트로부터 응답 메시지 요청이 오면 즉시 응답한다. 후자의 경우, MSS는 해당 트랜잭션의 완료 여부를 검토하여 트랜잭션이 그 결과를 전송한다. 만약 트래잭션이 아직 대기 상태이면 완료나 철회될 때까지 기다린 후 응답하며, 완료된 상태이면 갱신 결과를, 철회된 상태이면 철회 메시지를 전송한다.

갱신 트랜잭션은 모든 고정호스트에 있는 데이타 사본들을 동시에 갱신할 수 없기 때문에 각 MSS의 무효화 메시지가 서로 다를 수 있다((그림 1, 2) 참조). 이와같은 이유로 이동호스트가 다른 셀로 이동할 때 캐쉬 일관성이 보장되지 못하는 문제가 발생한다. 이때 이동호스트는 모든 캐쉬를 제거하므로써 일관성을 유지할 수 있다. 그러나 이동호스트는 빈번히 사용하는 데이타를 캐쉬하므로 제거된 데이타의 대부분은 다시 캐쉬될 것이며 이는 낮은 대역폭에 부담을 가중시킨다. 그러므로 단순히 이동호스트가 메시지 수신에 실패하였다는 이유로 모든 캐쉬를 버린다는 것은 비효율적이다. 따라서 가능하면 갱신된 항목만을 찾아 무효화시키므로써 불필요한 통신 부담을 줄이는 것이 효율적이다.

4.2 UMRCC 정책

이제부터 이동호스트가 다른 셀로 이동할 때 모든 캐쉬 데이타를 제거하지 않고 갱신된 항목만을 찾아 무효화시키는 방법인 UMRCC를 제안한다. 이를 위하여 갱신 트랜잭션이 2PC 프로토콜에 의해 완료되는 과정과 MSS의 방송 시점, 그리고 이동호스트의 이동을 고려한다.

(그림 4)에서 점선은 MH 의 이동을 나타내고, 화살표를 가진 실선은 2PC 프로토콜에서 메시지 이동을 나타내고, 이동호스트 MH는 MSS2에서 MSS1으로 이동한다고 하자. (그림 4)의 A, B 그리고 C에서 갱신 트랜잭션은 MSS1을 조정자로 하여 수행을 종료한 후 트랜잭션의 완료를 위하여 2PC 프로토콜을 수행한다고 할 때 MSS1은 MSS2에 vot-req 메시지를 보낸다. MSS2는 ⓐ의 시점에 *vot_req*를 수신한 후 MSS1에게 "yes"를 응답하고 데이타 x가 갱신가능함을 인식한다. MSS1은 모든 고정호스트로부터 "yes"를 받으면 그 트랜잭션을 완료하기 위하여 각 사이트에 완



(그림 4) MSS의 방송과 캐쉬 데이터의 갱신
(Fig. 4) Update of cache datum and broadcasting of MSS

료 메시지를 보내고 데이타베이스에 갱신 결과를 반영한다. MSS2도 조정자 MSS1으로부터 완료 메시지를 받은 후 그 결과를 데이타베이스에 반영한다. 따라서 실제 데이타베이스에 반영되는 시점은 MSS1의 경우에는 ⓑ이고, MSS2의 경우에는 ⓒ가 된다. 그러나 MSS2가 "no"를 응답하게 되면 조정자는 갱신 트랜잭션을 철회하고 모든 고정호스트에 철회 메시지를 보낸다. 철회 메시지를 받은 각 고정호스트는 해당 트랜잭션을 철회한다.

i)(그림 4)의 A 경우

MH가 방송 시점 t_j에서 MSS2의 셀에 있을 때 〈EMPTY〉를 수신하고, MSS1의 셀에 있을 때 갱신가능 메시지를 수신한다. 이 경우 방송시점 t_{j+1}에서 모든 MSS는 갱신 결과를 방송하므로 캐쉬 일관성이 유지된다.

ii)(그림 4)의 B 경우

MH가 방송시점 t_j에 MSS1이나 MSS2의 셀에 있으면 갱신가능 메시지를 수신한다. 이 경우도 방송시점 t_{j+1}에서 모든 MSS는 갱신 결과를 방송하므로 캐쉬 일관성이 유지된다.

iii)(그림 4)의 C 경우

MH가 방송시점 t_j에서 MSS1의 셀에 있다면 무효화 메시지를 수신한다. 그러나 MH가 방송시점 t_j에서 MSS2의 셀에 있다면 갱신가능 메시지를 수신하고 MSS1의 셀로 이동하며 다음 방송 t_{j+1}에서 MSS1으로부터 갱신 결과를 방송바디 못한다. 이 때 MH는 MSS2로부터 이미 해당 데이타에 대한 갱신가능 메시지를 이미 수신하고 있으므로 그 데이타에 관한 정보를 MSS1에게 요구하여 수신하지 못한 무효화 메시지의 갱신 결과를 얻을 수 있으므로 캐쉬 일관성이 위배되는 것을 방지할 수 있다.

지금까지 설명된 UMRCC 정책을 종합하면 이동호스트가 무효화 메시지를 수신하면 캐쉬를 갱신하고, 갱신가능 메시지를 수신한 상태에서 다른 셀로 이동하면 이동호스트는 무효화 메시지를 수신할 기회를 상실하게 되고 캐쉬 일관성은 위배된다. 이 때 이동호스트는 갱신가능한 데이타의 정확한 값을 MSS에게 요구한다. 그리고 MSS는 응답 메시지에 의하여 그 결과를 이동호스트에게 전송한다.

기존 정책[8]과 비교할 때 UMRCC 정책은 이동호스트가 다른 셀로 이동할 때마다 MSS에게 갱신 여부를 확인하는 절차를 줄일 수 있고, 캐쉬 데이타를 모두 제거하지 않고도 캐쉬 일관성을 유지할 수 있다.

4.3 메시지 교환 알고리즘

4.1절에서는 갱신가능 메시지의 이용하여 캐쉬 데이타의 제거없이 캐쉬 일관성을 유지할 수 있는 정책인 UMRCC 정책에 관하여 기술하였다. 여기에서는 기술된 정책을 기반으로 하여 MSS가 메시지를 방송하고 이동호스트가 그 메시지를 수신하는 과정을 보인다. BROADCASTING_to_MH는 MSS가 메시지를 방송하고 RESPONSE_to_MH는 MSS가 응답 메시지를 전송하는 알고리즘이다.

```
/* MSS가 무효화 메시지를 방송하는 알고리즘 */
Procedure BROADCASTING_to_MH;
BEGIN
    CASE msg of /* for invalidation message */
          1 : broadcasting
              update message, updatable message;
          2 : broadcasting
              update message;
          3 : broadcasting
              updatable message
    otherwise : broadcasting 〈EMPTY〉
  END;
END
```

```
/* MSS가 응답 메시지를 전송하는 알고리즘 */
Procedure RESPONSE_to_MH;
  IF request data from MH THEN
     IF for caching THEN response response-message
     ELSE IF for requestion of the updatable data
     THEN IF related transaction is committed
             THEN response response-message
             IF related transaction is active
             THEN delay transmission until
                     the transaction is commited;
                     IF related transaction is aborted
                     THEN transmit abort message;
END.
```

RECEIVE_from_MSS는 BROADCASTING_to_MH를 이용하여 MSS가 무효화 메시지를 방송하면 이를 수신한 이동호스트가 캐쉬 일관성을 유지하기 위하여 메시지를 처리하는 알고리즘이다.

```
/* 이동호스트에서 메시지 처리를 위한 알고리즘 */
Procedure RECEIVE-from-MSS;
BEGIN
  IF MH received update message THEN
  BEGIN
    update cache_data of MH
    IF the active transaction accessed data
        in update message
      THEN active transaction is aborted
      ELSE continue operation of active transaction
  END;
  IF MH received updatable message
  THEN BEGIN
        IF across boundary of cell of MH
        THEN BEGIN
              request value of the updatable
               data to MSS;
              blocking operation of active
               transaction until accept response
               message;
            END;
        END;
  IF MH received response message
  THEN BEGIN
        cache or update data on MH
        IF updated data and active transaction of
          MH accessed the data
        THEN abort the transaction
        ELSE continue the execution of active
          transaction;
       END;
END.
```

【정리 1】 UMRCC는 알고리즘 BROADCASTING_to_MH와 RESPONSE_to_MH, 그리고 RECEIVE_from_MSS에 의하여 캐쉬 일관성을 유지한다.

(증명) 일반적으로 이동호스트는 MSS로부터 방송된 무효화 메시지를 이용하여 캐쉬 일관성을 유지한다. UMRCC에서 MSS는 ①무효화 메시지와 갱신가능 메시지를 함께 방송하거나 ②무효화 메시지만을 방송하며, ③갱신가능 메시지만을 방송할 수 있다. 또한 ④갱신 데이타와 갱신가능한 데이타가 없으면 〈EMPTY〉를 방송한다. 이동 컴퓨팅 환경에서 셀은 물리적으로 다른 셀과 중첩될 수 있으며((그림 3) 참조), 비록 셀이 중첩될지라도 이동호스트는 단지 하나의 MSS와 통신이 가능하다.

이동호스트가 다른 셀로 이동하면 적어도 위의 네 가지 경우 중에서 적이도 한 메시지를 받게 된다. BROADCAS TING_to_MH는 이를 나타낸다. 이동호스트가 ①의 메시지를 수신하면 캐쉬를 갱신하고, MSS에게 응답 메시지를 요청한다. ②의 경우는 캐쉬를 갱신하고, ③의 경우는 MSS에게 응답 메시지를 요청한다. 그리고 ④의 경우는 이동호스트에게 어떠한 조치를 요구하지 않는다. 이를 반영한 과정이 RECEIVE_from_MSS이다. 마지막으로 MSS가 이동호스트로부터 응답 메시지 요청이 있으면 MSS는 관련 트랜잭션이 완료되었으면 해당 데이타를, 철회되었으면 철회 메시지를 전송하고, 아직도 그 트랜잭션이 수행 중이면 완료까지 지연한다. 이를 반영한 과정이 RESPONSE_to_MH이다. 이동호스트는 갱신 데이타에 대하여 무효화 메시지를 이용하여 그리고 트랜잭션이 아직 완료되지 않았으나 갱신가능성이 있는 데이터에 대하여 갱신가능 메시지와 응답 메시지를 이용하여 캐쉬 일관성을 유지한다.

5. 비용 평가

이 장에서는 UMRCC와 기존 정책[8]의 통신비용(메시지의 량)을 비교 분석한다. 다음은 어떤 시간 주기 L 동안에 어떤 트랜잭션의 연산을 위하여 소요되는 통신비용을 산출하는데 필요한 기호들이다.

λ^{w}_{x}: 데이타 x가 고정호스트에 의해 갱신될 확률

λ^{r}_{ix}: 데이타 x가 이동호스트 i에 의해 판독될 확률

λ^{mo}_{i}: 이동호스트 i가 다른 셀로 이동할 확률

λ^{up}_{ix}: 데이타 x를 캐쉬한 이동호스트 i가 MSS로부터 갱신가능 메시지를 받을 확률

δ: 이동호스트 i가 한 데이타를 캐쉬 요청할 때 MSS에 보내는 메시지의 크기

δ': 이동호스트 i가 제거된 모든 데이타들을 다시 캐쉬 요청할 때 MSS에 보내는 메 시지의 크기

δ'': 이동호스트 i가 갱신가능한 데이타를 MSS에 확인 요청하는 메 시지의 크기

α: 이동호스트 i가 요청한 한 데이타에 대하여 MSS가 방송하는 응답 메시지의 크기

α': 이동호스트 i가 요청한 모든 데이타에 대하여 MSS가 방송하는 응답 메시지의 크기

α'': MSS가 방송하는 갱신가능 메시지의 크기

n: 이동호스트 i가 제거된 데이타들 중에서 필요한 데이타들을 다시 캐쉬 요청한 데이타의 수

이제 이동호스트가 캐쉬를 사용하며 캐쉬 일관성이 위배되었을 때 해당 데이타들을 캐쉬하기 위하여 소요되는 비용이 기존 정책과 UMRCC에서 어떻게 다른가를 비교 분석한다.

기존 정책에서는 이동호스트가 다른 셀로 이동하여 무효화 메시지의 수신 기회를 상실하면 이동호스트 i의 모든 캐쉬를 제거한다. 그 후에 이동호스트 i가 제거된 데이타를 캐쉬한다. 여기에서 이동호스트 i가 모든 캐쉬 데이타를 제거할 확률은 다음과 같다.

$$\lambda^{mo}_i \sum_{x \in cache(i)} \lambda^{up}_{ix}$$

이 때 제거된 데이타를 다시 캐쉬하는 두 가지 경우를 고려하여 소요되는 통신비용을 산출하자.

i) 제거된 데이타를 모두 다시 캐쉬하는 경우

$$C_{all\text{-}cache}(i) = \lambda^{mo}_i (\delta' + \alpha') \sum_{x \in cache(i)} \lambda^{up}_{ix} \qquad ①$$

ii) 제거된 데이타를 필요에 따라 캐쉬하는 경우

$$C_{need\text{-}cache}(i) = \lambda^{mo}_i(\delta + \alpha)\Big(\sum_{x \in cache(i)} \lambda^{r}_{ix}\Big)\Big(\sum_{x \in cache(i)} \lambda^{up}_{ix}\Big)$$

$$= n\,\lambda^{mo}_i (\delta + \alpha) \sum_{x \in cache(i)} \lambda^{up}_{ix} \qquad ②$$

식 ①과 ②로부터 $C_{all\text{-}cache}(i)$과 $C_{need\text{-}cache}(i)$를 비교하여 보자. 즉, $(\delta' + \alpha')$와 $n(\delta + \alpha)$를 비교한다. 이를 위해 다음과 메시지 항목들을 가정한다.

i) $\delta = \langle MH\text{-}id,\ data\text{-}item \rangle$

ii) $\delta' = \langle MH\text{-}id,\ data\text{-}item_1,\ data\text{-}item_2,\ ...,\ data\text{-}iten_n. \rangle$

iii) $\delta'' = \langle$MH-id, data-item$_1$, data-item$_2$, ..., data-iten$_k$.$\rangle$

iv) $\alpha = \langle MSS\text{-}id,\ MH\text{-}id,\ (data\text{-}item,\ value,\ timestamp) \rangle$

v) $\alpha' = \langle MSS\text{-}id,\ MH\text{-}id,\ (data\text{-}item,\ value,\ timestamp)_1,$
$(data\text{-}item,\ value,\ timestamp)_2,\ ...,$
$(data\text{-}item,\ value,\ timestamp)_n \rangle$

vi) $\alpha'' = \langle MSS\text{-}id,\ MH\text{-}id,\ (data\text{-}item,\ value,\ timestamp)_1,$
$(data\text{-}item,\ value,\ timestamp)_2,\ ...$
$(data\text{-}item,\ value,\ timestamp)_k \rangle$

여기에서 n=1이면 $\delta = \delta'$, $\alpha = \alpha'$이고, n>1이면 δ'는 하나의 MH-id와 데이타에 관한 정보를 가지고 있으나 δ의 경우는 δ를 n번 반복하여 전송하게 된다. 따라서 δ'는 $n\delta$보다 n-1개의 MH-id를 절약할 수 있으므로 $\delta' < n\delta$과 $\alpha' < n\alpha$가 성립한다. 결과적으로 $C_{all\text{-}cache}(i) \le C_{need\text{-}cache}(i)$이 된다. 그러므로 UMRCC 정책의 통신비용과 비교하기 위하여 기존 정책의 통신비용으로 $C_{all\text{-}cache}(i)$을 이용한다.

다음은 UMRCC 정책에서의 통신비용($C_{updatable}(i)$)이다.

$$C_{updatable}(i) = \lambda^{mo}_i(\delta'' + \alpha'') \sum_{x \in cache(i)} \lambda^{up}_{ix} \qquad ③$$

기존 정책과 SUMRC와의 비용을 비교하면 다음과 같다.

$$C_{all\text{-}cache}(i) - C_{updatable}(i) = \lambda^{mo}_i(\delta' + \alpha') \sum_{x \in cache(i)} \lambda^{up}_{ix}$$

$$-\lambda^{mo}_i(\delta'' + \alpha'') \sum_{x \in cache(i)} \lambda^{up}_{ix} \qquad ④$$

식 ④의 결과가 양수이면 제안한 정책 UMRCC 정책의 통신 비용이 절약되고 대역폭을 효율적으로 사용한 결과가 된다. 식 ④에서 $C_{all\text{-}cache}(i)$는 제거된 데이타 모두를 다시 캐쉬하므로 제거된 데이타가 n개라면 α'와 δ'는 n개의 데이타에 대한 메시지의 크기이다. 식 ④에서 $C_{updatable}(i)$는 갱신가능한 데이타에 대해서만 캐쉬하므로 갱신가능한 데이타의 수를 k라 하면 δ''와 α''는 k개의 데이타에 대한 메시지이다. 이 때, k는 n개의 캐쉬 데이타 중에서 갱신가능한 데이타에 해당하므로 k<n이 성립한다. 따라서 $(\delta' + \alpha') > (\delta'' + \alpha'')$이 성립되므로 $C_{all\text{-}cache}(i) > C_{updatable}(i)$가 된다. 지

금까지 설명한 내용에 따르면 통신 비용인 메시지의 량이 가장 적은 기존 정책보다도 UMRCC 정책의 통신 비용이 더 절약된다.

또한 기존 정책은 제거된 모든 데이타를 다시 캐쉬하여야 하므로 통신채널에 대한 많은 점유 경쟁을 야기하며, 이들 데이타를 다시 캐쉬한 후 연산을 해야 하므로 트랜잭션 수행은 데이타들이 캐쉬될 때까지 지연된다. 그러나 UMRCC 정책은 갱신가능한 데이터만을 수정하므로 통신 채널의 사용 부담이 줄고, 제거된 모든 데이타가 캐쉬될 때까지 기다릴 필요가 없으므로 전체적인 트랜잭션 수행 시간이 줄어 든다.

6. 관련 연구

기존 연구의 이동 컴퓨팅 환경에서 낮은 대역폭을 효율적으로 활용하기 위하여 자주 사용되는 데이타를 미리 해당 호스트의 기억공간에 캐쉬하고, 이동호스트의 캐쉬 일관성을 유지하기 위한 정책들이 제안되었다[1, 4, 7, 8]. 여기에서 MSS는 갱신 데이타에 대한 무효화 메시지를 주기적으로 방송할 때 캐쉬 데이타를 가지고 있는 이동호스트가 무효화 메시지를 수신하면 캐쉬 데이타를 갱신한다. 그러나 이동호스트가 다른 셀로 이동하거나 통신 단절로 인해 무효화 메시지를 수신하지 못하면 캐쉬 데이타의 갱신 여부를 인식하지 못하게 되어 캐쉬 일관성이 위배된다. 이 때 이동호스트가 가지고 있는 모든 캐쉬 데이타를 제거하므로써 캐쉬 일관성을 유지할 수 있으나 이는 매우 비효율적이다[12].

[1]에서는 무선 통신비용을 데이타 요청과 방송에 이용되는 메시지 크기로 표현하였으며, 데이타에 대한 판독/갱신률을 고려하여 캐쉬 데이타를 선정하는 알고리즘을 제안하였다. [13]에서는 이동호스트를 기능에 따라 다음과 같은 세 가지 역할로 분류하였다. 첫 번째, 단순한 단말기의 역할을 한다. 두 번째, 데이타베이스의 서버 역할을 한다. 마지막으로 캐쉬를 사용하는 클라이언트 역할을 한다. 이 논문에서는 이동 트랜잭션의 올바른 수행을 위하여 이동호스트의 캐쉬 일관성을 유지하는 것이 주요 과제임을 보였다. [8]에서는 이동호스트가 다른 셀로 이동하여 무효화 메시지를 수신하지 못할 때 갱신 카운터와 타임스탬프를 이용하여 캐쉬 일관성이 위배되는 문제를 해결하였다. 이동호스트가 다른 셀로 이동하여 MSS에게 데이타를 요청하면 MSS는 해당 데이타와 갱신 카운터를 방송한다. 이동호스트는 전송된 무효화 메시지의 갱신 카운터가 자신이 유지한 타임스탬프보다 크면 무효화 메시지를 수신하지 못했음을 인식한다. 이 때 이동호스트는 MSS에 요청한 데이타를 제외한 모든 캐쉬 데이타를 제거하여 캐쉬 일관성을 유지한다. 따라서 이 정책은 이동호스트가 새로운 셀로 이동하면 무효화 메시지의 수신 여부를 확인하기 위하여 MSS에게 데이타를 요청하여야 하며, 모든 캐쉬 데이타를 제거한 후 다시 캐쉬하므로 대역폭에 대한 부담을 가중시킨다. [4]에서는 이동호스트가 MSS와 통신이 단절된 후 다시 통신이 연결되었을 때 캐쉬 일관성 위배에 대한 문제를 TS(broadcasting TimeStamps) 방법과 AT(Amnesic Terminals) 방법, 그리고 시그니쳐(signatures) 방법을 이용하여 해결하였다. 여기에서는 통신이 단절된 동안에 갱신 데이타에 대한 정보를 MSS에 유지하고, 이동호스트가 다시 통신 연결을 요청해 오면 MSS가 그 정보를 방송하여 캐쉬 일관성을 유지하였다. AT에서는 MSS가 갱신 정보를 유지하는 기간을 한 방송 주기로 정하고, TS에서는 MSS가 별도의 창(window)을 이용하며, 창의 크기를 동적으로 조정하여 갱신 정보를 유지할 수 있도록 하였다. [7]에서는 [4]와 같은 경우에 비트 열을 이용하는 방법을 제안하여 해결하였다. 이 방법은 통신이 단절된 동안에 MSS가 갱신 데이타들에 대한 정보를 비트 열을 이용하여 표현하고, 이동호스트가 다시 통신 연결을 요청해 오면 무효화 메시지로써 비트 열을 방송하므로써 대역폭을 보다 효율적으로 사용하였다.

본 논문에서 이동호스트가 캐쉬를 사용하고, [4, 7]에 의하여 통신 단절로 인한 캐쉬 일관성 위배는 해결될 수 있다. UMRCC 정책에서는 이동호스트가 다른 셀로 이동할 때 캐쉬 일관성이 위배되는 문제를 해결하였다. [8]에서는 이동호스트가 다른 셀로 이동하면 무효화 메시지의 수신 여부를 확인하기 위하여 MSS에게 정보를 항상 요청하여야 한다. UMRCC 정책에서는 이동호스트가 다른 셀로 이동한 후 갱신 가능성이 있는 데이타만을 찾아 MSS에게 확인 요청하므로 대역폭의 부담을 줄일 수 있다. 또한 기존 정책[1, 4, 7, 8, 12]에서는 캐쉬 일관성이 위배되면 모든 캐쉬 데이타를 제거한다. 이때 제거된 데이타는 이동호스트

에서 자주 사용되기 때문에 대부분 다시 캐쉬될 것이므로 대역폭의 부담을 가중시킨다. UMRCC 정책에서는 이동호스트가 무효화 메시지를 수신하지 못하여도 갱신가능 메시지를 이용하여 갱신 데이타에 대한 정보를 다시 받을 수 있는 기회를 부여 받는다. 그러므로 갱신가능 메시지의 이용은 캐쉬 일관성 위배를 야기하지 않으므로 모든 캐쉬 데이타를 제거할 필요가 없기 때문에 대역폭의 부담은 더욱 줄어든다.

7. 결 론

이동 컴퓨팅 환경에서 캐쉬 일관성은 이동호스트의 무제한적인 이동과 MSS와의 통신 단절로 인해 위배될 수 있다. 또한 무선 통신 채널의 낮은 대역폭은 캐쉬 일관성을 유지하는데 많은 제약을 준다. 따라서 캐쉬 일관성을 유지하고 낮은 대역폭을 효율적으로 사용하는 정책이 요구된다.

이 논문에서는 이동호스트가 다른 셀로 이동하여 무효화 메시지를 수신할 기회를 상실하여 캐쉬 일관성이 위배되는 문제와 대역폭을 효율적으로 사용하기 위한 UMRCC 정책을 제안하였다. 여기에서 갱신가능 메시지 이용은 캐쉬 일관성이 위배되었을 때 모든 캐쉬 데이타를 제거하지 않고 갱신 데이타만을 찾아 다시 캐쉬하기 때문에 보다 효율적으로 대역폭을 사용할 수 있도록 보장한다. 또한 이동호스트가 다른 셀로 이동할 때마다 무효화 메시지 수신 여부를 판별하기 위하여 MSS에 항상 데이타를 요청할 필요가 없이 캐쉬 일관성을 유지한다.

앞으로의 연구방향은 MSS의 방송 주기가 캐쉬 일관성에 미치는 영향 및 이동호스트에서 갱신 트랜잭션이 수행될 때 캐쉬의 일관성을 고려하는 것이다.

참 고 문 헌

[1] Ada Fu et al., "Dynamic Policies in Selecting a Caching Set for a Distributed Mobile Computing Environment", Technical Report, Department of Computer, the Chinese University, Hong Kong, May, 1995.

[2] B. R. Badrinath, A. Acharya, T. Imielinski, "Structuring Distributed Algorithms for Mobile Hosts", in Proc. of the 14th International Conference on Distributed Computing Systems, pp. 21-28, Jan., 1994.

[3] B. R. Badrinath and T. Imielinski, "Replication and Mobility", In second Workshop on the management of Replicated Data, pp. 9-12, 1992.

[4] D. Barbara et al., "Sleepers and Workaholics: Caching Strategies in Mobile Environments", in Proc. ACM SIGMOD, June, 1994.

[5] Evaggelia Pitoura et al., "Revising Transaction Concepts for Mobile Computing", in Proc. IEEE Workshop on Mobile Systems and Applications, Dec., 1994.

[6] G. H. Forman and J. Zahorjan, "The Challenges of Mobile Computing", IEEE Computers, 27(6), pp. 38-47, Apr., 1994.

[7] J. Jing, Omran Bukhres, Ahmed Elmagarmid, Rafael Alonso, "Bit-Sequences: A New Cache Invalidation Method in Mobile Environments", pp. 1-25.

[8] M. H. Wong and W. M. Leung, "A Caching Policy to Support Read-Only Transactions in a Mobile Computing Environment"

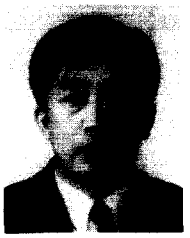
[9] P. A. Bernstein, V. Hadzilacos and N. Goodman, "Concurrency Control and Recovery in database Systems", Addison Wesley, Reading, Massachusetts, 1987.

[10] R. Alonso and H. F. Korth, "Database Systems Issues in Nomadic Computing", in Proc. of the 1993 SIGMOD Conference, pp. 388-392, May, 1993.

[11] S. Ceri, G. Pelagatti, "Distributed Databases Principles and Systems", McGraw-Hill, 1984.

[12] T. Imielinski and B. R. Badrinath, "Quering in Highly Mobile-distributed Environments", in Proc. of the 18th VLDB Conference Vancouver, British Columbia, Canada, pp. 41-52, 1992.

[13] V. R. Narasayya, "Distributed Transactions in a Mobile Computing System", CSE552, March, 1994., pp. 1-13.



박 성 배

1985년 2월 전남대학교 계산통계학과(학사)
1987년 8월 전남대학교 대학원 계산통계학과(이학석사)
1995년 8월 전남대학교 대학원 전산통계학과(박사과정 수료)
1988년 4월~현재 순천공업전문대학 전자계산과 부교수
관심분야:분산시스템, 분산멀티미디어, 이동 컴퓨팅 등

황 부 현

1978년 숭실대학교 전산학과(학사)
1980년 한국과학기술원 전산학과(공학석사)
1994년 한국과학기술원 전산학과(공학박사)
1980년~현재 전남대학교 전산학과 교수
관심분야:분산 시스템, 분산 데이타베이스 보안, 객체지향 시스템