단순화된 메타데이타를 이용한 고스트 기반 정족수 동의 기법의 개선

(Quorum Consensus Method based on Ghost using Simplified Metadata)

조 성 연 † 김 태 윤 ††

(Song-Yean Cho) (Tai-Yun Kim)

요 약 중복 데이타는 분산 시스템에서 결함을 포용하는 방법 중 하나로 여러 복사본들의 일관성을 유지하기 위하여 복제 제어 프로토콜(replica control protocol)을 필요로 한다. 복제 제어 프로토콜 중 정족수 동의 기법(quorum consensus method)은 정족수 이상의 동의를 얻어서 데이타에 접근하는 방법이다. 그런데 사이트나 통신 링크의 실패로 인하여 정족수 이상의 동의를 얻을 수 없으면 데이타 접근이 불가능하게 되므로 부족한 정족수를 고스트(ghost)로 대치하는 방법이 필요하다. 고스트는 메타 데이타(meta data)에 상태 정보만을 저장한 프로세스이기 때문에 생성과 관리의 비용을 줄이려면 메타 데이타를 간략화 시키는 것이 중요하다. 따라서 본 논문에서는 동료 집합을 이용하여 메타 데이타를 구성하는 방법을 제안한다. 제안된 방법은 2N+logN 비트만으로 메타 데이타를 구성할 수 있게 하고 동료 집합만을 이용하는 프로토콜이나 동적 선형 보우팅 프로토콜보다 향상된 가용성을 나타낸다. 제안된 방법의 가용성은 마코브 모델(Markov model)을 이용하여 측정하고 그 결과를 다른 프로토콜들과 비교 분석한다.

Abstract Replicated data that is used for fault tolerant distributed system requires replica control protocol to maintain data consistency. The one of replica control protocols is quorum consensus method which accesses replicated data by getting majority approval. If site failure or communication link failure occurs and any one can't get quorum consensus, it degrades the availability of data managed by quorum consensus protocol. So it needs for ghost to replace the failed site. Because ghost is not full replica but process which has state information using meta data, it is important to simplify meta data. In order to maintain availability and simplify meta data, we propose a method to use cohort set as ghost's meta data. The proposed method makes it possible to organize meta data in 2N+logN bits and to have higher availability than quorum consensus only with cohort set and dynamic linear voting protocol. Using Markov model we calculate proposed method's availability to analyze availability and compare it with existing protocols.

1. 서 론

하나의 컴퓨터가 가지는 성능과 신뢰성의 한계를 극복하기 위하여 네트워크를 이용하여 여러 개의 컴퓨터들을 연결하는 것을 분산 시스템(distributed system)이라 한다. 분산 시스템은 다수의 컴퓨터 노드와 통신 링크로 구성되기 때문에 일부 사이트나 통신 링크에서 실패가 발생할 수 있다. 이러한 부분적 실패를 마스킹(masking)하고 전체 시스템의 가용성을 향상시키기 위하여 데이타의 복사본(*replica*)을 여러 사이트에 저장하는 중복 데이타가 사용된다. 중복 데이타의 사용은 시스템 전체의 통신 비용을 감소시키고 가용성을 증가시키지만 여러 복사본들 사이의 일관성을 유지하기 위하여 복제 제어 프로토콜을 필요로 한다. 현재까지 [1,2,3,5,10]과 같은 다양한 복제 제어 프로토콜이 제안되고 개발되어 왔다. 그 중 정족수 동의 기법은 어떠한 오퍼레이션을 수행하기 위해서는 정족수 이상 복사본들의 동의

† 학생회원 : 고려대학교 컴퓨터학과
sycho@netlab.korea.ac.kr
†† 종신회원 : 고려대학교 컴퓨터학과 교수
tykim@netlab.korea.ac.kr
논문접수 : 1999년 6월 7일
심사완료 : 1999년 11월 11일

를 얻어야 하는 방법으로 개념이 간단하고 사이트나 통신 링크 복구 후 프로토콜 적용을 위한 특별한 메커니즘을 필요하지 않는 장점을 가진다. 그러나 이러한 장점과 동시에 사이트나 통신 링크의 실패가 반복되어 정족수 이하의 복사본들만의 가용상태에 있으면 데이타 접근이 불가능해지는 단점도 가진다. 이러한 단점을 해결하기 위하여 [1]에서는 실패가 발생한 복사본들을 재생산하는 방법을 제안하였다. 그런데 완전 복사본을 재생산하는 것은 비용이 많이 들기 때문에 복사본의 상태 정보만 저장하여 정족수 동의 결정 과정에 참여하도록 하여 가용성을 유지하면서 비용을 줄이는 방법이 필요하다. 상태 정보만을 저장한 고스트[2]를 이용하는 방법이 그것인데 이 때 상태 정보를 저장하는 메타 데이타를 효율적으로 구성할 수록 비용 감소 효과가 높아진다. 따라서 본 논문에서는 [6]에서 제안된 동료 집합(cohort set)으로 메타 데이타를 구성하는 고스트 메커니즘을 구현하여 비용을 낮추고 가용성을 향상시키는 방법을 제안한다.

제안된 방법은 고스트 이용한 정족수 동의 기법에 기반하고 있으므로 2장에서는 관련 연구로 정족수 동의 기법과 고스트에 대하여 설명한다. 3장에서는 [7]에서 제안된 동료 집합의 개념을 기술하고 동료 집합을 이용할 때 최신 데이타 값을 가지는 복사본을 찾는 알고리즘을 제안한다. 4장에서는 고스트 메커니즘의 메타 데이타를 동료 집합으로 구성하는 방법을 제안하고 제안된 메타 데이타를 이용하여 읽기, 쓰기, 복구의 실행을 위한 알고리즘을 제안한다. 5장에서는 마코프 모델(Markov model)을 이용하여 제안된 메커니즘의 가용성을 분석하여 평가하고 마지막으로 6장에서 결론과 향후 과제를 제시한다.

2. 정족수 동의 기법 기반 고스트 메커니즘

본 장에서는 분산 시스템 환경에서 중복 데이타의 일관성 유지를 위한 복제 제어 프로토콜에 대하여 설명한다. 중복 데이타가 일관성을 잃어버리는 원인은 분산 환경 내의 사이트나 통신 링크에서 실패가 발생하기 때문이다. 본 논문에서는 이러한 분산 시스템의 실패 모델을 사이트에서 실패가 발생하면 사이트는 동작을 멈추고 다른 사이트에게 오류 메시지를 전달하지 않는 fail stop 실패 모델로 전제한다. 그리고 사이트들 사이에서 송수신 되는 메시지는 통신 링크를 지나는 동한 변형되지 않으며 통신 링크에서 발생한 실패는 연결되어 있던 사이트의 실패와 동일시한다.

2.1 복제 제어 프로토콜

중복 데이타를 다룰 때 가장 중요한 것을 여러 개의 복사본들 사이의 일관성을 유지하는 것이기 때문에 복제 제어 프로토콜을 이용하여 일관성을 유지한다. 복제 제어 프로토콜 중 가장 간단한 것은 Available Copy Protocol[10,11]이다. ROWA-A(read one write all available)이라고도 불리는 이 방법은 쓰기 동작을 가용 상태의 모든 복사본을 대상으로 실행하여 가용 상태의 모든 복사본들은 최신 데이타 값을 가진다고 전제하는 방법이다. 따라서 읽기는 가용 상태인 어떠한 복사본을 대상으로도 실행 가능하고 쓰기도 가용 상태인 복사본이 하나라도 존재하면 실행 가능하므로 읽기와 쓰기 모두에 대하여 높은 가용성을 보장한다. 그러나 통신 링크의 일시적인 실패로 인하여 네트워크 분할이 발생하면 다른 네트워크 분할에 존재하는 가용 상태의 모든 복사본들을 대상으로는 쓰기의 실행이 불가능하므로 데이타의 일관성을 보장하기가 어렵다. 이러한 단점을 보완한 것이 정족수 동의 기법이다. 정족수 동의 기법은 데이타에 접근하기 위하여 [정의 1]로 정의된 정족수 집합내의 복사본들 중 정족수 이상의 동의를 얻어야 하는 방법이다.

[정의 1] 정족수 집합 : 프로토콜을 P, 복사본들의 전체 집합을 U, |U| = n, 오퍼레이션 O를 수행하기 위하여 필요한 정족수를 Q_O라고 할 때 O에 대한 정족수 집합 Q_O는 $Q \subseteq U$이고 $|Q| > Q_O$를 만족하는 Q중에서 P에 의하여 Q_O를 얻을 수 있는 집합이다.

중복 데이타를 대상으로 읽기 O_R, 쓰기 O_W를 실행할 때 읽기는 마지막 실행된 쓰기에 의하여 갱신된 데이타 값을 읽도록 보장하므로 읽기와 쓰기의 정족수 집합은 다음 공리를 만족한다.

[공리 1] 교집합 법칙 : 읽기 정족수 집합 Q_R와 쓰기 정족수 집합 Q_W는 적어도 하나 이상의 공통 원소를 가진다. 즉 $Q_R \cap Q_W \neq \phi$이다.

복사본이 3개 존재할 때 사이트나 통신 링크의 연속된 실패로 인하여 가용 상태의 복사본이 불가용 상태로 변화되는 과정은 (그림 1)과 같다.

(그림 1)과 같이 복사본이 3개 존재하고 읽기 정족수 Q_R와 쓰기 정족수 Q_W가 2일 때 State0는 사이트나 통신 링크의 실패가 전혀 발생하지 않은 상태로 모든 복사본이 가용 상태이다. State0는 읽기와 쓰기 정족수가 모두 만족되기 때문에 읽기와 쓰기가 모두 실행 가능하다. State0에서 사이트나 통신 링트의 실패가 발생하면 가용 복사본이 2개인 State1로 전이된다. State1는 읽기와 쓰기 정족수가 모두 만족되기 때문에 읽기와 쓰기가 모두 실행 가능하다. 또 다시 실패가 발생하여 Sate2로

전이되면 가용 복사본이 하나만 존재하므로 읽기와 쓰기의 실행이 모두 불가능하다. 가용 복사본이 하나도 존재하지 않는 State3에서도 State2 상태와 같이 읽기와 쓰기의 실행이 불가능하다.

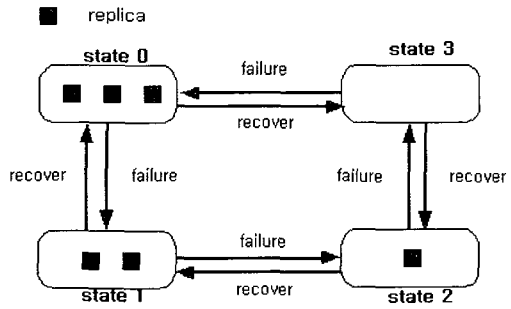

그림 1 실패 발생에 따른 복사본의 가용성 변화

위에서 설명한 바와 같이 정족수 동의 기법은 네트워크 분할에서도 높은 가용성을 제공하지만 사이트나 통신 링크에서 실패가 발생하면 정족수 이상의 동의를 얻는 것이 불가능해질 수 있다. 이것을 막기 위하여 동적 선형 보우팅 기법(Dynamic Linear Voting)[12,13]이 제안되었다. 동적 선형 보우팅 기법은 복사본들에게 가중치를 부여하고 가용 복사본들의 가중치 합이 정족수 이상이면 오퍼레이션이 실행되도록 하는 방법에 기반한다. 만일 사이트나 통신 링크의 실패로 가용 복사본의 수가 줄어들면 가용 상태의 복사본들의 가중치를 조정하여 읽기나 쓰기가 실행되도록 하여서 중복 데이타의 가용성을 향상시킨다. (그림 1)처럼 복사본이 3개 존재할 때 초기 상태 State0에서는 모든 복사본들의 가중치를 1로 설정한다. 그러나 연속적인 실패가 발생하면 State2 상태로 전이되므로 쓰기와 읽기의 실행이 불가능해진다. 그러한 상황이 발생하면 가용 상태의 복사본의 가중치를 2로 조정하여 쓰기의 실행이 가능하도록 만든다. 그러나 이 방법은 가중치 할당을 위한 특별한 알고리즘을 필요로 할뿐만 아니라 [12,13]에서 설명된 바와 같이 가중치의 조절이 전체 복사본들의 수가 아니라 가중치가 조정되기 전 단계의 가용 복사본의 수와 연관되어 조정되기 때문에 복구된 복사본들이 정족수 결정 과정에서 제외되는 현상이 발생할 수 있다. 따라서 본 논문에서는 실패가 발생한 복사본을 고스트로 대치하여 오퍼레이션의 가용성을 높이는 방법을 이용한다[2].

2.2 정족수 동의 기법 기반 고스트 메커니즘

고스트는 복사본의 상태 정보만을 메타 데이타에 저장하여 실패가 발생한 복사본 대신 정족수 동의 결정 과정에 참여하는 프로세스이다. 고스트는 데이타 값을 가지지 않기 때문에 읽기에는 참여하지 못하고 쓰기에만 참여하여 쓰기 가용성을 Available Copy Protocol과 같은 수준으로 향상시킨다.

복사본들의 가용 상태를 감시하는 boot service는 완전 복사본의 실패가 감지되었을 때 잉여 사이트에 고스트를 생성한다. 본 논문에서는 [2]처럼 boot service는 충분히 복제되어 있고 실패가 거의 일어나지 않으며 고스트를 생성하기 위한 잉여 사이트들은 [2]처럼 충분히 많다고 가정한다. 이러한 가정 하에서 사이트나 통신 링크의 실패로 인하여 가용 상태의 복사본들이 고스트로 대치되는 과정은 (그림 2)와 같다.

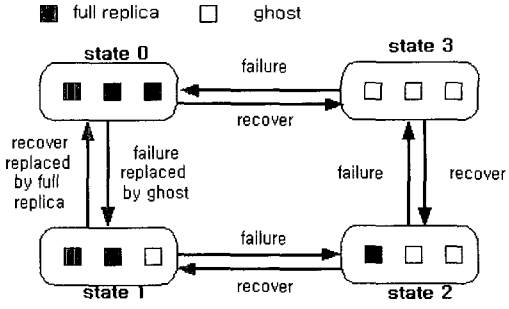

그림 2 실패 발생에 따른 고스트의 생성 변화

(그림 2)는 복사본이 3개 존재하고 읽기 정족수 Q_R와 쓰기 정족수 Q_W가 2일 때를 표현한 것이다. 여기에서 복사본은 (그림 1)의 가용 복사본과 같은 의미인 완전 복사본과 실패가 발생한 완전 복사본을 대치하는 메타 데이타만을 가지는 고스트 두 가지 상태로 존재한다. State0은 사이트나 통신 링크의 실패가 전혀 발생하지 않은 단계로 모든 복사본이 완전 복사본의 상태이다. State0은 읽기와 쓰기 정족수가 모두 만족되기 때문에 읽기와 쓰기의 실행이 모두 가능하다. 사이트나 통신 링트에서 실패가 발생하면 잉여 사이트에 고스트가 생성되어 완전 복사본을 대치하여서 한 개의 고스트와 두 개의 완전 복사본이 존재하는 State1로 전이된다. State1은 읽기와 쓰기 정족수가 모두 만족되므로 읽기와 쓰기의 실행이 모두 가능하다. State1에서 다시 실패가 발생하여 완전 복사본 하나가 고스트로 대치되면 Sate2로 전이된다. State2 상태는 고스트를 이용하지 않았을 때에는 (그림 1)의 State2와 같은 읽기와 쓰기 정족수를 모두 만족시키지 못하므로 읽기와 쓰기의 실행이 모두 불가능하다. 그러나 메타 데이타를 가지는 고스트를 이용하여 쓰기 정족수를 보충하면 쓰기의 실행이

가능해 진다. 따라서 Available Copy Protocol과 같이 가용 상태의 복사본이 하나 이상만 존재하면 쓰기가 가능한 수준으로 쓰기 가용성이 향상된다. State2에서 또 다시 실패가 발생하면 고스트들만 존재하는 혼수상태(comatose)인 State3으로 전이된다. State3에서는 완전 복사본이 하나도 없기 때문에 읽기와 쓰기가 모두 불가능하다. 그리고 혼수 상태 발생 이후 복사본이 복구될 때 복구 순서에 따라서 읽기 정족수 이상의 복사본들이 복구되어도 읽기가 불가능한 상황이 발생할 수 있다. 이러한 상황은 최신 데이타 값을 가진 복사본이 복구되지 않았기 때문이므로 혼수 상태 발생 이후 마지막으로 갱신된 데이타 값을 가진 복사본이 복구되거나 쓰기가 실행되어서 데이타 값이 갱신되거나 특별할 복구 오퍼레이션이 실행되기 이전에는 읽기 정족수를 만족하더라도 읽기를 실행할 수 없다.

3. 동료 집합 기반 최신 복사본 검색

2장에서 설명한 고스트를 이용한 정족수 동의 기법에서 동료 집합을 이용하여 메타 데이타를 구성하면 필요한 메타 데이타의 크기를 줄일 수 있다. 따라서 본 장에서는 제안한 방법에서 이용되는 동료 집합의 정의와 특징을 설명하고 고스트 메커니즘에서 동료 집합을 이용할 때 최신 데이타 값을 가진 복사본을 찾는 알고리즘을 제안한다.

3.1 동료 집합

메타 데이타는 복사본의 갱신 상태를 저장하기 위하여 사용되는데 현재까지는 주로 쓰기가 실행될 때마다 하나씩 증가되는 버전 넘버를 사용하였다. 버전 넘버는 개념상 이해가 쉽고 구현이 간단하지만 [8]에서 지적한 것처럼 다음과 같은 단점을 지닌다. 첫째 고정된 크기의 정수를 이용하므로 단순 증가 방법을 사용하면 언젠가는 오버플로우가 발생하기 때문에 이를 막기 위한 특별한 메커니즘이 필요하다. 둘째 복사본 집합의 상태에는 변화가 없더라도 쓰기가 실행될 때마다 버전 넘버를 증가 시켜야 한다. 예를 들면 정족수 집합이 {A, B, C}일 때 쓰기가 복사본 A, B, C 모두의 데이타 값을 갱신시키면 갱신 후에 A, B, C 모두 최신 데이타 값을 가지므로 복사본 집합의 상태는 불변이지만 복사본들의 버전 넘버는 증가되어야 한다. 마지막으로 버전 넘버의 타입에 따라서 버전의 범위에 한계가 달라진다. 정수로 표현하는 경우라도 구현되는 플랫폼에 따라서 정수가 32비트인 경우에는 2^{32}번의 갱신을 표현할 수 있고 64비트인 경우에는 2^{64}번의 갱신을 표현할 수 있다. 그러므로 버전 넘버를 표현하는 변수의 타입과 실행되는 플랫폼에 따라서 표현할 수 있는 갱신의 횟수가 달라지는 문제점을 가진다. 본 논문에서는 위와 같은 문제점들은 복사본들 집합의 상태를 고정된 크기의 메타 데이타로 저장하는 동료 집합을 이용하여 해결한다. [7]에서 제시한 동료 집합의 정의는 다음과 같다.

[정의 2] 동료 집합 : 복사본 R의 동료 집합 C_R은 가장 최근 쓰기나 복구 오퍼레이션에 참여한 복사본들의 집합이다.

[공리 2] 복사본 R의 동료 집합 C_R은 항상 자기 자신을 포함한다.

복사본은 쓰기나 복구 오퍼레이션에 참여 할 때마다 자신의 동료 집합을 갱신하기 때문에 [정의 2]와 [공리 2]에 따라서 오퍼레이션에 참여한 복사본들의 동료 집합은 같다. 그러므로 동일한 동료 집합을 가진 복사본들은 동일한 데이타 값을 가진다고 보장할 수 있어서 버전 넘버를 동료 집합으로 대치할 수 있다.

동료 집합은 복사본들의 상태를 표현하는 것이므로 복사본이 M 개의 다른 상태를 가지면 logM 비트로 상태를 표현할 수 있다. 즉 N개의 복사본이 존재할 때 동료 집합은 N*logM 비트로 표현될 수 있다. 따라서 전체 메타 데이타의 양은 N개의 복사본들이 존재할 때 복사본들을 구별하기 위한 ID를 표현하는 logN 비트와 동료 집합의 비트를 더한 N*logM + logN 비트이다. 아래 (표 1)는 버전 넘버와 동료 집합을 사용하였을 때 특징들의 비교이다.

표 1 버전 넘버와 동료 집합의 비교

	버전 넘버를 사용했을 때	동료 집합을 사용했을 때
오버 플로우	발생 특별한 메커니즘 필요	발생 안 함
갱신 횟수	단순한 증가 알고리즘이므로 제한됨	무제한
저장 장소 사용량	표현 데이타 타입과 구현된 언어 그리고 플랫폼에 따라서 다름 ex) 정수, C언어, Unix 시스템이면 16비트	N*logM + logN으로 고정 N: 복사본의 수 M: 복사본의 변화 가능한 상태 수

3.2 최신 데이타 값 검색 알고리즘

고스트 메커니즘의 메타 데이타를 동료 집합으로 구성할 때 읽기나 쓰기를 제공하기 위해서는 [2]에서 가장 큰 버전 넘버를 가진 복사본의 데이타 값을 최신 값으로 확신하는 것과 같이 최신 데이타 값을 가지는 복사본을 구별하는 방법이 필요하다. 따라서 본 장에서는 동료 집합의 상태를 분석하여 최신 데이타 값을 가지는

완전 복사본을 찾는 알고리즘은 제안한다.

최신 데이타 값을 가진 복사본을 찾기 위하여 동료 집합의 특징을 분석해야 한다. [보조 정리 2]는 고스트 메커니즘에서 동료 집합을 이용했을 때 항상 만족되는 특성이다.

[보조 정리 1] 중복 데이타 D가 N개의 복사본들로 이루어지고 그 중 M개가 완전 복사본이고 N-M이 고스트일 때 M개의 완전 복사본들이 읽기 정족수 집합을 구성하고 그 복사본들의 동료 집합이 모두 동일하면 M개의 완전 복사본들은 모두 최신 데이타 값을 가진다.

<증명>

(1) [공리 1]의 교집합 법칙에 따라서 M개의 완전 복사본으로 구성된 읽기 정족수 집합 중 적어도 하나의 완전 복사본은 최신 데이타 값을 가진다.

(2) [정의 2]와 [공리 2]에 의하여 쓰기 실행 후 참여 복사본들의 동료 집합은 같아진다. 그런데 복사본 M개 중 적어도 하나의 완전 복사본이 가장 최근에 실행된 쓰기나 복구에 참여하여 최신 갱신 상태를 표현하는 동료 집합을 가진다. 따라서 그 복사본과 동일한 동료 집합을 가지는 M-1개의 복사본들도 그 복사본이 마지막으로 참여한 쓰기나 복구에 참여한 것이다.

(3) 따라서 복사본들 중 하나가 최신 데이타를 가지고 동료 집합이 모두 동일하면 복사본들은 모두 최신 데이타를 가진다. □

[보조 정리 1]을 적용함으로써 복사본들의 동료 집합이 모두 같은 경우 최신 데이타 값을 갖는 복사본이라고 확신할 수 있다. 다수의 실패와 복구 이벤트의 발생으로 복사본들의 동료 집합이 달라지는 경우에는 [보조 정리 2]를 이용하여 최신 데이타 값을 가진 복사본을 찾는다.

[보조 정리 2] 중복 데이타가 N개 복사본을 가질 때 그 중 M개가 완전 복사본이고 N-M개가 고스트이다. M개 완전 복사본이 읽기 정족수 집합 Q_R을 형성하면 $R \in Q_R$인 모든 복사본 R중에서 C_R내의 고스트들의 집합을 C_{GR}로 표시할 때 $|C_R - C_{GR}|$이 가장 적거나 $C_R - C_{GR}$이 C_R내의 다른 복사본들의 동료 집합의 부분집합이면 복사본 R은 최신 데이타 값을 가진다.

<증명>

(1) [공리 1]에 의하여 M개의 완전 복사본이 읽기 정족수 집합 Q_R을 형성하면 Q_R의 원소 중 적어도 하나는 최신 데이타 값을 가지고 있다.

(2) [정의 2]와 [공리 2]에 의하여 최신 쓰기나 복구의 실행 이후 실행에 참여한 복사본들이 동료 집합은 같다. 만약 복사본들의 동료 집합이 다른 경우는 마지막 쓰기나 복구 오퍼레이션 이후 동료 집합의 원소 중 일부에서 실패가 발생하여 고스트로 대치된 경우이다.

(3) 실패가 발생하여 완전 복사본이 고스트로 대치되면 동료 집합의 완전 복사본의 수가 줄어든다.

(4) 완전 복사본이 줄어든 다음 실패가 발생한 복사본이 복구된 후 복구나 쓰기 오퍼레이션이 실행되면 동료 집합이 갱신된다. 동료 집합이 갱신되면 복사본들의 동료 집합 원소들 중 완전 복사본의 수가 모두 하나씩 증가되므로 동일한 동료 집합을 가지게 된다.

(5) 따라서 Q_R에 속하는 복사본들 중에서 동료 집합내의 완전 복사본 원소 수가 가장 적거나 완전 복사본 집합이 다른 복사본들의 동료 집합들 중에서 완전 복사본 집합의 부분집합이면 복사본 R은 최신 데이타 값을 가진다. □

[보조 정리 2]에 의하여 최신 데이타 값을 가진 복사본을 확신하는 과정을 <예제 1>을 통하여 설명하면 다음과 같다.

<예제 1> 세 개의 복사본 A, B, C가 존재하고 읽기와 쓰기 정족수가 2($|Q_R| = |Q_W| = 2$)일 때, 이벤트들의 발생과 오퍼레이션들의 실행에 따라서 복사본들 동료 집합의 내용이 변화된다. 각각의 복사본들의 동료 집합을 Ca, Cb, Cc로 표현한다. 고스트는 G로, 고스트의 동료 집합은 [A, B, C]로 표현하며 최신 데이타 값을 가진 복사본의 동료 집합은 A, B, C로 표현한다.

(a) State 0 (초기 상태) : Ca = A, B, C Cb = A, B, C Cc = A, B, C

어떠한 오퍼레이션도 실행되기 전인 초기 상태이므로 모든 복사본들이 최신 데이타 값을 가진다. 증명된 [보조 정리1]에서 제시한 것과 같이 최신 데이타 값을 가지는 모든 복사본 A, B, C의 동료 집합이 동일한 것을 확인할 수 있다.

(b) State 1 (C 실패, 쓰기 실행) : Ca = A, B, G Cb = A, B, G Cc = [A, B, C]

복사본 C가 있는 사이트에서 실패가 발생하여 복사본 C가 비가용(unavailable)상태가 되면 고스트로 대치된다. 대치된 후 쓰기 오퍼레이션 Ow가 실행되면 동료 집합들은 State 1이 된다. State1에서 복사

본 C는 상태 정보만을 가진 고스트로 대치되므로 쓰기의 실행에 따른 결과 값을 가질 수 없으므로 복사본 A와 B 만이 최신 데이타 값을 가진다. 이것은 [보조 정리 1]에서 제시한 것과 같이 최신 데이타 값을 가지는 고스트 상태가 아닌 복사본 A와 B의 동료 집합이 동일한 것을 알 수 있다.

(c) State 2 (B 실패, C 복구) : Ca =A, B, G Cb = [A, B, G] Cc = A, B, C
복사본 B가 있는 사이트에서 실패가 발생하여 비가용 상태가 되고 복사본 C가 복구되어서 다시 완전 복사본으로 대치되면 동료 집합들은 State 2가 된다. 복구된 복사본 C는 아직 쓰기의 실행 결과가 반영되지 못한 상태이므로 최신 데이타 값을 가지지 못한다. 따라서 복사본 A만이 최신 데이타 값을 가지므로 [보조 정리 2]에서 증명된 바와 같이 복사본 A와 C의 동료 집합 중 복사본 A가 동료 집합에서 고스트를 뺀 수가 가장 작은 것을 확인할 수 있다. 여기서 복구는 복사본의 사이트 복구로 정족수 동의 결정에 참여하기 위한 복구 오퍼레이션이 실행되기 전이다.

(d) State 3 (쓰기 실행) : Ca =A, G, C Cb = [A, B, G] Cc = A, G, C
복사본 C에서 대하여 복구 오퍼레이션이 실행되기 전에 쓰기가 실행되면 동료 집합들은 State 3으로 갱신된다. State3는 최근에 실행된 쓰기에 의한 데이타 갱신이 복사본 A와 C에 모두 반영된 상태이므로 복사본 A와 C가 모두 최신 데이타 값을 가진다. [보조 정리 1]에서 증명된 바와 같이 최신 데이타 값을 가지는 복사본 A와 C의 동료 집합이 모두 같은 것을 확인할 수 있다.

[보조 정리 1]과 [보조 정리 2]를 바탕으로 최신 데이타 값을 가지는 복사본을 찾는 함수 GetLatestValue를 디자인하면 알고리즘은 (그림 3)과 같다.

```
Let QC be the set of quorum.
Let C_R be the cohort set of R, R∈QC.

temp = R', R'∈QC;
num = 0;

repeat
if ( R∈QC, R's cohort set C_R == C_R' )
    QC = QC - R;
    num++;
until QC == ∮;
if ( num == |QC| )
   return R1_value;

num = 0;
original_QC = temp_QC = QC;
repeat
   temp = R', R'∈temp_QC;
   temp_QC = temp_QC -R';
   repeat
      if ( C_R & C_R' == C_R' )
         QC = QC - R;
         num++;
   until QC == ∮;
   if ( num == |temp_QC| )
      return R'_value;
   QC = original_QC;
until temp_QC == ∮;
```

그림 3 제안된 GetLatestValue의 알고리즘

4. 동료 집합 기반 고스트 메커니즘

본 장에서는 3장에서 제시한 최신 데이타 값 복사본 검색 알고리즘을 이용하여 읽기, 쓰기, 복구 오퍼레이션의 알고리즘을 제안한다. 그리고 동료 집합을 이용하여 메타 데이타를 구성할 때 고스트 메커니즘에서 이용되는 메타 데이타의 구성도 제안한다.

4.1 동료 집합 기반 메타 데이타

고스트 메커니즘을 이용하는 복제 제어 프로토콜에서 복사본은 완전 복사본과 고스트 두 가지 상태를 가진다. 고스트는 완전 복사본에서 실패가 발생하였을 때 정족수 동의 결정을 대신하는 것으로 상태 정보만을 가진 프로세스이다. 만약 실패가 연속적으로 발생하면 완전 복사본이 하나도 존재하지 않는 혼수 상태가 되는데 이러한 상태는 복사본의 복구 순서에 따라서 일관성이 유배될 수 있으므로 오퍼레이션을 실행하기 전에 특별한 처리를 해야 한다. 이러한 사항들을 고려하여 고스트 메커니즘에서 이용하는 메타 데이타를 구성하면 다음과 같다.

Ghost's meta data

Identifier	current cohort set	fail cohort set	comatose bit
logN	N	N	1

full replica's meta data

Identifier	current cohort set	comatose bit
logN	N	1

그림 4 제안된 메타 데이타의 구성

복사본의 상태가 고스트와 완전 복사본 두 가지이므로 log2 비트로 상태 표현이 가능하다. 따라서 복사본이 N개일 때 동료 집합은 log2* N 비트 즉 N비트로 표현할 수 있다. 고스트 메커니즘에서는 혼수 상태가 발생하

므로 이를 표시하기 위하여 comatose 1비트를 이용한다. 또한 고스트가 쓰기 오퍼레이션에 참여하면서 복사본들의 갱신 상태를 구별에 도움이 주어야 하므로 고스트의 메타 데이타는 두 개의 동료 집합으로 구성한다. 버전 넘버처럼 데이타 값의 상태 변화를 유지하는 $C_{current_cohort_set}$과 실패가 발생한 당시의 상황을 저장하는 $C_{fail_cohort_set}$이다. 위와 같이 구성되는 고스트의 메타 데이타는 (그림 4)와 같다.

4.2 제안된 메타 데이타 기반 오퍼레이션

본 장에서는 (그림 4)과 같이 구성된 메타 데이타를 이용하여 중복 데이타에 실행되는 읽기, 쓰기, 복구의 알고리즘을 제안한다.

중복 데이타의 값을 갱신하는 쓰기가 실행되면 데이타 값뿐만 아니라 복사본의 갱신 상태를 표현하는 동료 집합도 갱신되어야 한다. 또한 쓰기가 실행되면 완전 복사본들이 모두 최신 데이타 값을 가지이므로 정족수만 만족되면 읽기를 실행 할 수 있다. 즉 혼수 상태가 발생한 후라도 쓰기가 실행된 후면 복구된 복사본들이 모두 최신 데이타 값을 가지므로 읽기 정족수만 만족되면 읽기를 실행할 수 있다. 따라서 쓰기를 실행할 때 데이타 값을 갱신한 후 comatose 비트를 0으로 세팅하여 최신 데이타 값을 가진 완전 복사본이 가용 상태임을 표시한다. 제안된 쓰기 알고리즘은 (그림 5)과 같다.

```
Let QC be the set of quorum.
Let C_R be the cohort set of R, R∈QC.
Let C_full be cohort set of full replica.
Let C_ghost be cohort set of ghost replica.

num = 0;
repeat
if ( R∈QC, R's cohort set C_R is C_full )
    QC = QC − R;
    num++;
until QC == ∮;

if (num == 0)
   abort .
else if (num == 1)
        R's comatose bit=1 and Update C_R, all R∈QC.
     else if (num > 1)
            execute write to R.
            R's comatose bit = 0, all R∈QC.
      update C_R, all R∈QC.
```

그림 5 제안된 쓰기 오퍼레이션의 알고리즘

읽기는 마지막 쓰기에 의하여 갱신된 데이타 값을 읽는 것을 보장해야 하므로 정족수를 만족해야하는 조건 외에 혼수 상태에 대한 처리가 필요하다. 혼수 상태에 대한 처리는 다음과 같다. comatose를 비트를 체크하여 1이면 혼수 상태 발생 가능성이 있는 것이고 0이면 없는 것이다. 따라서 comatose 값이 0이고 정족수만 만족되면 읽기를 수행하고 1이면 정족수가 만족되더라도 최신 데이타 값을 가진 복사본이 가용 상태가 아님을 의미하므로 읽기를 실행하지 못한다. 제안된 읽기 알고리즘은 (그림 6)과 같다.

```
Let Q_Read be quorum number for reading.

num = check = 0;
repeat
if ( R∈QC, R's cohort set C_R is C_full )
    QC = QC − R;
    num++;
  if ( R's comatose bit ==1 )
        check++;
until QC == ∮;

if (num >Q_Read && check == 0)
    GetLatestValue(QC);
else abort;
```

그림 6 제안한 읽기 오퍼레이션의 알고리즘

정족수 동의 기법은 특별한 복구 과정을 필요로 하지 않는 특징을 가지는데 고스트 메커니즘을 이용하면 혼수 상태라는 특수 상황이 발생하므로 복사본이 있는 사이트가 복구되었을 때 특별한 오퍼레이션을 필요로 한다. 즉 혼수 상태가 발생한 이후 쓰기가 한 번도 실행되지 않았거나 최신 데이타 값을 가진 복사본이 아직 복구되지 않았는지를 조사할 수 있는 방법이 필요하다. 조사 방법은 comatose 비트를 이용하는 것인데 값이 1이면 현재 최신 데이타 값을 가진 완전 복사본이 없는 상황이므로 자신이 최신 데이타 값을 가지고 있는지 체크한다. 체크는 고스트의 동료 집합인 $C_{current_cohort_set}$과 $C_{fail_cohort_set}$를 비교하는 것으로 두 동료 집합이 동일하면 자신이 최신 데이타 값을 가진 복사본임을 의미한다. 자신이 최신 데이타 값을 가진 경우에는 다른 완전 복사본들에게 자신의 데이타 값을 전송하고 comatose 비트를 0으로 세팅하여 읽기 가능 상태로 만든다. 반면에 동료 집합이 동일하지 않으면 자신은 최신 데이타 값을 소유하지 못했고 다른 완전 복사본들도 마찬가지인 경우이다. 이 때에는 다음에 실행될 쓰기나 복구 오퍼레이션이 복사본들의 데이타를 최신 데이타 값으로 갱신시킬 것이라고 예측할 수 있으므로 특별한 오퍼레이션을 취하지 않는다. (그림 7)는 복구 오퍼레이션 알고리즘을 나타낸 것이다.

```
Let R_value be the data value of recovered replica R.
Let D_value be the latest value of replicated data D.
Let C_ghost_current be current cohort set of ghost.
Let C_ghost_fail be fail cohort set of ghost.
Let CCB_R be the comatose check bit of R, R∈QC.

num = check = 0;

repeat
if ( R∈QC, R's cohort set C_R is C_full )
    QC = QC - R;
    num++;
    if (R's comatose bit == 1)
        check++;
until QC == φ;

if (check >0 && C_ghost_current == C_ghost_fail)
   CCB_R = 0;
   Execute write R_value to all R∈QC.
else if (check == 0 )
        R_Value = GetLatestValue(QC).
update C_R, all R∈QC.
```

그림 7 복구 오퍼레이션의 알고리즘

5. 가용성 분석

중복 데이타의 성능은 데이타가 이용 가능한 상태에 있을 확률인 가용성으로 판단된다. 본 장에서는 제안된 방법의 읽기와 쓰기에 대한 가용성을 구하고 이것을 기존의 프로토콜들과 비교 분석한다.

본 논문에서는 복사본이 N개 존재할 때 오퍼레이션 O에 대한 가용성은 Ao(n)로 표현한다. 가용성을 계산하기 위하여 N개의 복사본이 N개의 사이트에 분리되어 존재하며 고스트를 생성시키는 boot service는 충분히 중복되어서 거의 실패가 일어나지 않는다고 가정한다. 또한 boot service는 복사본이 있는 사이트에서 실패가 발생하는 즉시 이를 감지할 수 있다. 이러한 전제 아래에서 가용성을 분석하기 위하여 [2,3,6,7]의 논문들처럼 다음을 가정한다. 실패는 평균 실패율(mean failure rate) λ로 지수 분포를 가지고 복구는 평균 복구율(mean repair rate) μ로 지수 분포를 가진다. 시스템은 대기 확률 분포(equilibrium distribution)를 가지며 이산 상태 마코프 프로세스(discrete-state Markov process)로 특징 지워진다.

(그림 8)은 복사본의 수 N이 3이고 읽기와 쓰기 정족수가 2일 때($|Q_W|=|Q_R|=2$) 실패와 복구의 발생에 따른 동료 집합의 상태 전이를 나타낸 것이다. 3개의 복사본이 존재할 때 각각의 복사본 R의 동료 집합 C_R은 xy로 표현한다. 이때 x는 완전 복사본의 수이고 y는 고스트의 수이다. 고스트의 동료 집합은 [xy]로 표현한다.

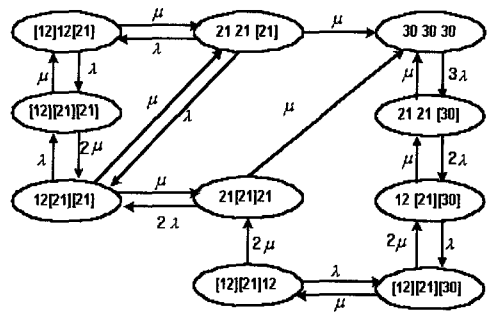

그림 8 복사본이 3개일 때 동료 집합의 상태 전이 다이어그램

(그림 8)에서 초기 상태는 <예제 1>의 State0과 같다. 복사본들의 동료집합은 모든 복사본들이 완전 복사본임을 표시하고 있으므로 동료 집합에 표시되는 완전 복사본의 수는 3이고 고스트의 수는 0이므로 동료집합은 30으로 표시된다. 따라서 복사본들 각각의 동료집합은 30, 30, 30으로 나타낼 수 있다. 세 개의 완전 복사본들 존재하는 사이트들 중 하나에서 실패가 발생할 확률은 하나의 사이트에서 실패가 발생할 평균 실패율이 λ이고 사이트가 3개이므로 3λ가 된다. 따라서 State0에서 3λ의 확률로 실패가 발생하면 <예제 1>의 State1과 같은 상태인 State1로 전이된다. State1은 실패가 발생한 사이트에 존재하는 복사본이 고스트로 대치되고 다른 사이트에 존재하는 복사본들의 동료 집합이 이것을 반영한 상태이다. 즉 가용 상태의 완전 복사본들의 동료 집합은 완전 복사본 2개와 고스트 1개가 존재함을 나타내어서 21로 표시된다. 이 때에 사이트나 통신 링크의 실패가 발생했을 때를 나타내는 고스트의 동료집합은 갱신되지 않으므로 실패가 발생 전의 상태인 30으로 남고 고스트의 동료 집합임을 나타내기 위하여 [30]으로 표시된다. 그러므로 State1에서의 복사본들 각각의 동료집합은 21, 21, [30]으로 나타낼 수 있다. State1에서 실패가 발생하였던 사이트가 복구되면 그 확률은 실패가 발생하였던 하나의 사이트가 복구되는 것이므로 평균 복구율 μ가 된다. 실패가 발생하였던 사이트가 복구되고 복구 오퍼레이션이 실행되면 고스트로 대치되었던 복사본은 완전 복사본으로 대치되면 <예제 1>의 State0와 같은 State0로 전이된다. 즉 이 때의 복사본들의 동료 집합의 상태는 30, 30, 30이 된다.

중복 데이타의 읽기 정족수 Q_R이 2이므로 읽기가 가

능한 상태는 (그림 8)에서 완전 복사본이 2개 이상인 상태인 P(30 30 30) , P(21 21 [30]), P(21 [21] 21), P(21 21 [21])이다. 따라서 ρ는 λ/μ로 평균 실패와 복구의 비율일 때 읽기 가용성 Ar(3)은 아래 식과 같다.

$$Ar(3) = P(30\ 30\ 30) + P(21\ 21\ [30]) + P(21\ [21]\ 21) + P(21\ 21\ [21]) = \frac{3\rho+1}{(\rho+1)^3}$$

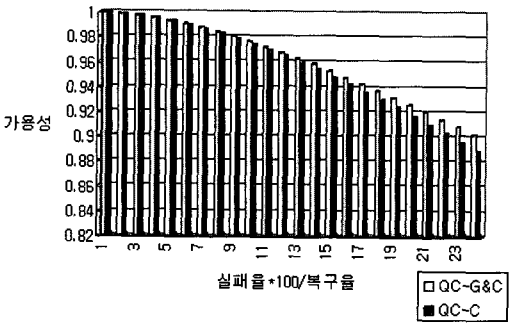

그림 9 읽기 오퍼레이션에 대한 가용성 그래프

위의 식으로 표현되는 읽기 가용성을 실패와 복구 비율의 변화에 따른 그래프로 표현한 것이 (그림 9)이다. 그래프의 세로축은 가용성 값으로 1이면 데이타가 항상 가용 상태인 것을 의미하고 작아질수록 가용성이 낮아짐을 의미한다. 그래프의 가로축은 평균 실패와 복구의 비율인 ρ값인데 실패가 복구에 비하여 발생 빈도가 높을수록 ρ값이 작아진다. [9]에서 실험한 결과에 따라 ρ는 0에서 0.25사이의 값으로 계산하였다. (그림 9)에 나타난 두 개의 그래프 중 QC-G&C는 본 논문에서 제시한 방법이고 QC-C는 동료 집합만을 사용한 방법이다. 두 개의 그래프들은 ρ의 값이 0에서 0.9사이일 때는 가용성이 비슷하고 0.11이상이 되면 가용성의 차가 0.25일 때는 0.03이상이 된다. 그러므로 (그림 9)에 나타난 바와 같이 제안한 방법은 실패의 발생에 대한 복구의 비율이 0.11에서 0.25 사이일 때 많은 가용성 향상을 보였다. 그런데 제안된 방법은 혼수 상태를 포함하기 때문에 쓰기가 충분히 자주 실행될 때, 즉 실패 빈도보다 평균 쓰기 오퍼레이션의 빈도가 더 높을 때 높은 가용성을 나타낸다.

제안된 방법의 쓰기 가용성은 고스트 메커니즘의 이용으로 많은 향상을 보였다. 쓰기 정족수 부족을 고스트가 채워주기 때문에 하나 이상의 완전 복사본만 가용 상태이면 쓰기가 가능하다. 따라서 제안된 방법을 이용하면 (그림 8)에서 완전 복사본이 하나 이상 존재하는 상태인 P(30 30 30), P(21 21[30]), P(12[21][30]), P([12] [21]12), P(12[21][21]), P([12]12[21]), P(21 21[21]), P(21[21]21)에서 쓰기를 실행 할 수 있으므로 Available Copy Protocol와 같은 수준의 가용성을 나타낸다.

$$Aw(3) = P(30\ 30\ 30) + P(21\ 21[30]) + P(12[21][30]) + P([12]\ [21]12) + P(12[21][21]) + P([12]12[21]) + P(21\ 21[21]) + P(21[21]21) = \frac{2\rho^4+11\rho^3+17\rho^2+9\rho+2}{(\rho+1)^3\times(2\rho^2+3\rho+2)}$$

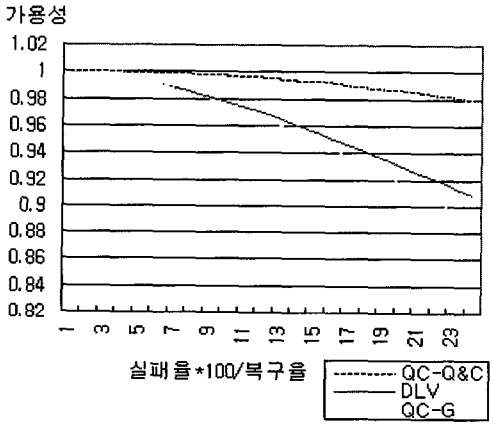

그림 10 쓰기 오퍼레이션에 대한 가용성 그래프

(그림 10)은 실패와 복구의 비율 변화에 따른 제안한 방법의 쓰기 오퍼레이션에 대한 가용성을 나타낸 것으로 그래프의 가로축과 세로축의 의미는 (그림 9)와 같다. 쓰기 오퍼레이션에 대한 가용성은 (그림 10)에 나타난 바와 같이 평균 0.97이상의 높은 값을 나타낸다. 고스트를 사용하지 않은 경우인 QC-C의 가용성은 정족수를 모두 완전 복사본으로 채워야하기 때문에 제안된 방법보다 0.02에서 0.08 정도 낮고 기존의 방법인 DLV의 가용성은 최신 데이타 값을 가진 복사본에 대한 낮은 변별도가 쓰기 오퍼레이션을 제약하기 때문에 제안한 방법보다 0.02에서 0.09 정도 낮다.

6. 결 론

중복 데이타의 사용은 사이트나 통신 링크에서 실패가 발생했을 때 이를 마스킹하고 데이타의 가용성을 향상시키는 반면 여러 복사본들을 두어야 하기 때문에 일관성 유지를 위한 복제 제어 프로토콜을 필요로 한다.

복제 제어 프로토콜들은 복사본의 갱신 상태를 메타 데이타에 저장하여 일관성 유지에 이용하므로 비용을 줄이기 위하여 메타 데이타를 효율적으로 구성해야 한다. 본 논문에서는 고스트 메커니즘을 이용하여 가용성을 향상시키면서 동료 집합으로 메타 데이타를 구성하여 비용을 줄이는 방법을 제안하였다. 제안된 방법은 2N+logN비트만으로 메타 데이타를 구성하면서 쓰기 가용성을 Available Copy Protocol 수준으로 향상시킬 수 있었다.

향후 연구 과제는 사이트나 통신 링크의 실패율과 쓰기 오퍼레이션의 빈도의 변화에 따른 읽기 오퍼레이션의 가용성 변화를 분석하여 읽기 오퍼레이션에 대한 가용성을 향상시킬 수 있는 방법을 연구하는 것이다.

참 고 문 헌

[1] J.F. Paris. "Efficient management of replicated data," In Proc. Int'l Conf. on Database Theory, 1988.

[2] A.S Tanenbaum and R. van Renesse, H. van Streveren, G.J Sharp, S.Mullender, J. Jensen, and G van Rossum, "Experiences with the Ameoba distributed operating system," Communications of the ACM, 33(12):46-63, December 1990.

[3] P.A Bernstein, V. Hadzilacos, and N. Goodman, *Concurrency Control and Recovery in Database Systems*, Addison -Wesley, 1987.

[4] B.M Oki and B.H Liskov, "Viewstamped replication: A general primary copy method to support highly available distributed systems," In Proc. 7th ACM Symp. On Principles of Distributed Computing, Toronto, Canada, Aug. 1988.

[5] R.H Thomas, "A majority consensus approach to concurrency control for multiple copy database," ACM transactions on Database Systems, 4(2):180-209, June 1979.

[6] J.F.Paris and Qun Rose Wang, "On the Performance of Voting with Ghosts," Proc. International Symposium on Applied Computing, pp. 75-84, 1993.

[7] Darrell D. E. Long and J. F. Paris, "A Leaner, More Efficient, Available Copy Protocol," Proc. 8th IEEE Symposium on Parallel and Distributed Processing, pp. 400-407, 1996.

[8] Darrell D. E. Long and J. F. Paris, "Voting without Version Numbers," Proc. 1997 International Phoenix Conference on Computer and Communication, pp. 139-145, 1997.

[9] Darrell D. E. Long , A. Muir, and G. Golding, " A Longitudinal Study of Internet Host Reliability," Proc. 14th Symp. On Reliable Distributed Systems, pp 2-9, 1995.

[10] P.A. Bernstein and N. Goodman, "The Failure And Recovery Problem For Replicated Database," ACM transaction on Database Systems, 9(4):596-615, December 1984

[11] P.A. Bernstein and N. Goodman, "Serializability Theory For Replicated Database," Journal of Computer and System Science, 31(3):355-374, December 1986

[12] D. Davcev and W.A. Burkhard, "Consistency and Recovery Control for Replicated Files," Proc. 10th ACM Symposium on Operating System Principles, pp 87-96, 1985

[13] S. Jajodia and D. Mutchler, "Enhancements to the Voting Algorithm," Proc. 13th VLDB Conference, pp 399-405, 1989

[14] 유헌창, 손진곤, 황종선, "분산 시스템에서 보우팅을 위한 계층 구조의 변환 방법", 정보과학회논문지, 22(3), pp. 371-378, 1995.

조 성 연

1998년 고려대학교 컴퓨터학과 (이학사). 1998년 ~ 현재 고려대학교 컴퓨터학과 석사과정 재학중. 관심분야는 분산 객체 시스템, 결합 포용 시스템, 네트워크 보안, 이동 에이전트 등

김 태 윤

1981년 고려대학교 산업공학과 졸업(공학사). 1983년 미국 Wayne State University 전산학과 석사 졸업. 1987년 미국 Auburn University 전산과학과 박사 졸업. 1988년 ~ 현재 고려대학교 컴퓨터학과 교수. 관심분야는 컴퓨터그래픽스, 컴퓨터네트워크, EDI시스템, ISDN, 이동통신, 위성통신 등