

이동 컴퓨팅 환경에서 효율적인 예측 가능한 자동 저장 방식 설계

김남광, 서진모*, 신승훈, 박승규

아주대학교 정보통신전문대학원

E-mail : {keywest, seo007, sihsh, sparky}@madang.ajou.ac.kr

A Study on Efficiently Predictive File Hoarding In the Mobile Computing Environment

Nam-Kwang Kim, Jin-Mo Seo*, Seung-Hun Sihh, Seung-Kyu Park
Professional Graduate School for Information & Communication Technology

요 약

본 논문에서는 이동 컴퓨팅 환경에서 사용자의 데이터 사용 유형을 분석하여, 네트워크의 영향을 감소시켜 사용자 작업을 계속 유지하는 방법에 대해 연구하였다. 이를 위해서는 데이터의 특성에 맞는 선택적 자동 저장 크기를 선별해야 하며, 기존의 판단 알고리즘 외에 추가 판단기준을 도입하면 자동 저장의 적중률이 향상됨을 보인다. 또한, 자동 저장 리스트의 크기에 대한 고찰을 통해 적정 수준의 적중률을 유지하기 위한 최소한의 리스트 크기를 알아보았다.

1. 서론

컴퓨팅 환경과 컴퓨터 네트워크의 발달에 따른 유 무선 디지털 네트워크 기술이 발전되면서, 이동 컴퓨팅 환경에서는 이동 중 원거리 작업과 동기적 그룹웨어에 대한 요구가 증가되고 있다. 하지만 이동성에 따른 가용자원의 부족, 이동환경에서의 네트워크 환경에 대한 신뢰도, 분실이나 데이터 손실의 위험, 갱신 자동저장정책의 필요, 제한된 전력 및 응용 프로그램의 부족 등이 문제점으로 제기되고 있다.

본 논문에서는 위의 문제점들 중에서, 이동 컴퓨팅 환경에서 보다 효과적이고 안정적인 작업 수행 유지에 필수 요소인 효율적인 자동저장방식을 설계하였다. 이는 고정된 통신 네트워크 환경에 이동 단말기가 추가되는 형태로 전개될 때, 적용 가능하도록 설계되어 있다

2. 관련연구

네트워크의 단절 상황에서도 사용자가 작업을 진행할 수 있도록 지원하는 시스템모델이 연구되어 왔는데, 그 중 카네기 멜론 대학의 AFS[1]에서는 단절된 상황에서 사용자가 단절을 대비하여 응용 작업에 적절한 파일을 지역 저장매체에 임시 저장(caching)하는 방법을 사용한다. 또한 Coda[2]에서는 사용자가 Hoard Profile 내에 자동 저장된 파일과 디렉토리를 직접 지정해주는 방법을 취한다. 하지만 네트워크의 단절이 자주 발생하게 되면 변동 상황에 대한 추적 작업이 계속 발생하게 되므로 네트워크에 부하가 심하고 재 연결 후 대응 조치가 신뢰도가 높지 못하다는 단점을 갖는다. 또한 Transparent Analytical Spying은 사용자의 파일 참조 형태를 관찰하여 작업그룹을 만들어 사용하는 것, 사용자 데이터와 응용 프로그램에 대한 데이터를 확장자나 디렉토리 이름으로 구분하는 것, 그리고 타임스탬프를 사용하는 세 가지 방법을 사용하고 있다.

이들 시스템의 특징은 두 가지로 분류될 수 있는데, 그 하나는 사용자의 의존도가 높다는 점이고, 이에 따라 필요한 파일을 적절히 저장하지 못한다는 문제점들을 가지고 있으며, 다른 하나는 작업시간이나 메모리 요구가 너무 많기 때문에 이동 컴퓨팅 환경에서 사용하기에는 부족함이 많다는 점이다.

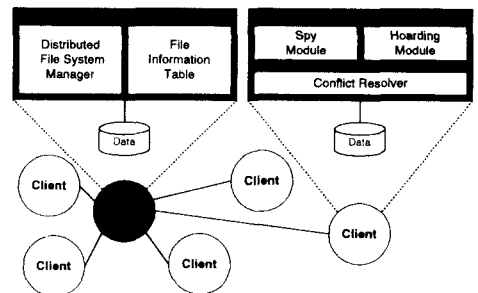
3. 시스템설계

3.1 개요

본 논문에서는 네트워크 단절 기간 동안의 사용자의 계속된 작업 수행을 지원하기 위한 파일 자동 저장 방식을 보다 효율적으로 지원하는 시스템을 제안한다. 기존의 시스템과 달리 사용자 의존 비용을 최소화 시키면서 자동저장을 위해 발생하는 추가적인 부하는 이동 컴퓨팅에 적합할 정도로 낮추고자 한다. 특히, 프로세스를 추적하는 방법을 통해 파일이 참조되는 정보를 보다 정확히 획득하고, 파일간 관계를 추가적인 다양한 범위에서 판단하여 자동 저장 시 발생 가능한 오류 확률을 최소화 하고자 한다.

3.2 시스템

본 논문에서의 시스템은 기존의 분산 시스템 모델을 근간으로 하고 있다. 즉, 각각의 서버간에는 유선의 네트워크로 연결되어 있고, 각 서버가 담당하는 셀 내의 클라이언트 간에는 무선/유선 LAN으로 연결됨을 가정한다.



<그림 1> 시스템 전체 구성도

제안한 시스템은 클라이언트/서버 시스템으로 이루어진다. <그림 1> 서버에는 분산 파일을 관리하는 분산 파일 시스템 관리자와 파일의 각종 정보 및 작업 중 사용 형태를 기록하는 파일 정보 테이블이 기존의 클라이언트에 추가되는 형태이다. 클라이언트는 미리 가져와야 할 파일 혹은 데이터를 관리하는 부분인 Hoard Module과, 현 클라이언트의 사용유형을

통계적으로 관찰하는 Spy module, 데이터간 충돌 해결 기능을 가진 Conflict Resolver로 구성된다.

4. 예측 가능한 파일 자동 저장 방안

4.1 자동 저장 모듈

일반적으로 데이터 파일의 캐쉬 미스의 경우 발생하는 문제보다 시스템 파일에 대한 캐쉬 미스의 경우 치명적인 오류를 유발시키기 때문에 이를 작업그룹을 형성하는데 정정보로 활용한다. 또한 시스템 파일을 상주 프로세스에 대한 것파, 사용자의 요청에 따라 주기적으로 실행되는 프로세스에 대한 것으로 구분한다. 전자는 캐쉬전략에 상관없이 100% 캐쉬되어야 하므로, 작업 그룹 결정을 위한 전략은 후자의 경우만을 대상으로 한다.

4.2 자동 저장 키(Hoarding Key)

본 시스템의 가장 큰 특징은 단절 중 작업 시 자동 저장된 데이터에 대한 통계적 자료가 'Spy Module'에 의해서 기록되고 보관된다는 점이다. 여기에 보관된 내용을 토대로 Hoarding Module이 자동 저장 시 판단기준으로 삼아, 다음 사용할 확률이 가장 높은 데이터를 선택하여 자동 저장하게 된다. <표 1>은 제안된 시스템에서의 저장키에 대한 리스트이다. 주요 키 중에서 'Working Set'키는 전적으로 'Spy Module'에 의해 결정된다. 따라서 결정된 정보가 완전히 획득되기 전까지는 다른 자동 저장 키에 의존하여 자동 저장된 파일들과, 이때 캐쉬오류가 발생되어 요구된 파일 정보를 통해 작업을 수행한다.

Main Key	분류기준
Working Set	사용자의 파일 참조 형태에 따른 log기록 및 tree형성
User Based	파일 소유권(owner, group, others)을 통한 분류
Time Based	최근 또는 특정 시간을 설정하여 분류
Application	각 응용 프로그램에 필요한 파일들의 그룹 형성

<표 1> 저장 키 리스트

4.3 선택 가능한 자동 저장 방법

파일 복제 방법에는 여러 가지가 있으나, 본 연구에서는 특정 형태의 복제 방법을 지양하고, 선택 가능한 복제 방법을 제안한다. 이동 컴퓨팅 환경에서의 저장공간간의 협소성과 네트워크의 신뢰도를 고려할 때, 모든 파일에 대해 동일 크기의 복제를 적용하는 것에는 부리가 있기 때문이다. 그러므로 복제 대상 파일에 대한 해당 특징을 기술하여, 자동 저장 시 여러 복제 단위 중 하나를 선정, 효율적인 복제를 가능하게 한다.

이동 컴퓨팅 환경의 열악함을 고려할 때, 이의 해결방안의 하나가 바로 볼륨을 이용한 파일 자동 저장 및 관리 방안인데, 이는 지역적으로 연결된 파일을 물리적 저장공간에 묶을 수 있다는 점과 볼륨의 단위를 나누어 실행한 작업을 통해 볼륨간의 통합적 관리가 용이하게 되는 것 및 연관 관계가 높은 데이터를 자동 저장하여 자동 저장 적중률을 높일 수 있는 가능성이 높다고 할 수 있다. 또한, 클라이언트에 저장된 볼륨들에 대해서는 각각 하나의 버전 벡터(Version Vector)를 부여한다. 이는 모든 복제 파일에 버전 벡터를 부여해 작업을 수행하는 Ficus[3]시스템과 비교할 때, 각 파일에는 버전 벡터보다 작은 크기의 단순한 상태 벡터를 부여하고, 파일의 수정이 발생할 경우에만 버전 벡터를 갱신하는 방법을 통해 불필요한 과부하를 줄이도록 하였다.

파일 복제에 다른 일관성 유지 정책을 위한 중요 도구로

파일 정보 테이블에 저장되는 각 파일 당 ID와 캐쉬 된 파일에 대한 태그(tag)가 있다. <표 2>과 같이 상태 벡터는 캐쉬 된 파일의 버전과 서버의 원 파일 간의 버전을 확인함으로써 캐쉬 된 파일이 자신의 소유인지 여부를 확인하여 작업 권한에 제한을 두었다. 서버번호는 그 파일에 대한 사용 권한을 가진 서버를 나타내고, C-flag는 파일의 일관성 유지 형태를 결정한다. C-flag가 1인 경우는 엄격한 일관성 유지를 의미하며, 0인 경우는 낙관적 일관성을 채택하여 일정 시간 경과 후 일관성을 유지한다.

Replication Schemes는 해당 파일의 자동 저장 시 알맞은 복제 방법을 선택할 수 있게 되어 있어 블록/파일/볼륨 단위 중 하나를 선택적으로 사용하여 보다 효율성을 기하였다.

	Client ID	Status Vector	C-Flag	Server #	Replication Scheme	Owner
File A	2, 3, 10	2(2, 3)	0	1	File	2
File B	5, 7	0	1	1	File	7
File C	4, 6, 10	1(6)	0	1	Volume	10
File D	2,3,4, 5,6,7, 8,9,10, 11	0	3	1	Block	1
...

<표 2> 파일 정보 테이블

그리고 시스템 파일인 경우에는 별도로 분류하여(C-flag : 3) 읽기만 가능한 상태가 된다. Client ID는 일관성 유지 문제가 발생 시, 해당 클라이언트를 참조하여 수정 혹은 그 외의 적당한 조치를 취하는데 이용한다. 서버로부터 자동 저장이 수행되면 각 해당 데이터에 tag가 붙게 된다. 플래그는 기존의 디스크 캐쉬에서 사용되는 플래그들(dirty bit, invalid bit, delay bit 등)과 함께 일관성을 결정하는 플래그(C-flag)가 첨가된다.

5. 모의 실험

5.1 모의 실험의 매개 변수 및 구조

모의 실험에서의 시스템의 매개변수는 동일 셀 내에서의 작업을 고려하였으므로 서버는 1개이고, 서버에 등록된 이동 클라이언트 개수는 10개로 가정하였다. 이동 클라이언트 그룹은 5개의 클라이언트를 하나의 그룹으로 하여 2개를 두었으며, 서버 내 데이터 개수는 시스템 관련 파일이 10개, 개인사용 파일이 각각 3개씩 30개, 그룹관련 파일은 각각 30개씩 60개를 사용 총 100개의 파일로 구성되어 있다. 데이터의 참조 형태(선택적 자동 참조의 경우)는 시스템 관련파일은 블록으로 개인사용 파일은 파일형태로, 그룹관련 파일은 선택적 볼륨을 이용하도록 하였으며, 하나의 이동 클라이언트의 사용주기 당 참조 데이터 수는 10개 설정하였다. 모의 실험 시스템의 구조는 다음과 같다. 서버와 클라이언트를 초기화한 후 각 클라이언트마다 서버에 데이터를 요청한다. 서버로부터 요청한 데이터가 이상 없이 클라이언트에 전송이 되면 Spy Module은 자동 저장 리스트에 기록한다. 이 리스트는 계속 보관되며 Hoarding Module의 자동 저장 작업이 발생하면 기존설정에 제공된다. Hoarding Client는 다음 요청 데이터를 예상하기 위해 Read Spy Module을 통해 Spy Module이 기록하고 있는 자동 저장 리스트를 넘겨받고 정해진 자동 저장 방법에 따라 다음

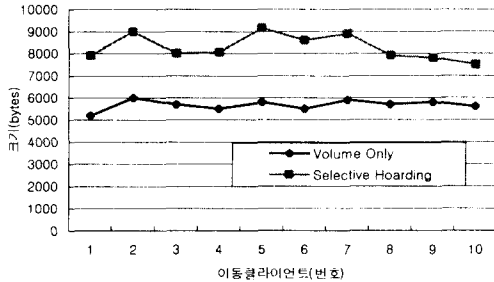
사용이 예상되는 데이터를 서버로 요청하게 되고 이 과정은 관찰 시간이 종료될 때까지 반복된다. < 표 3> 은 모의 실험에서의 조건을 나타내고 있다.

조건	값
시스템 관찰 시간	100~10,000sec
서버 및 네트워크 장애 발생	Negative exponential distribution (1/0.0001 sec)
장애 발생 후 복구 소요 시간	Negative exponential distribution (1/0.01 sec)
이동 클라이언트의 최초 데이터 요청 시간	Negative exponential distribution (1/0.0128 sec)
데이터 요청 후 서버의 응답 시간	Uniform distribution [3, 5] sec
이동 클라이언트의 데이터 수신 후 사용 시간	Uniform distribution [40, 60] sec

<표. 3> 실험조건

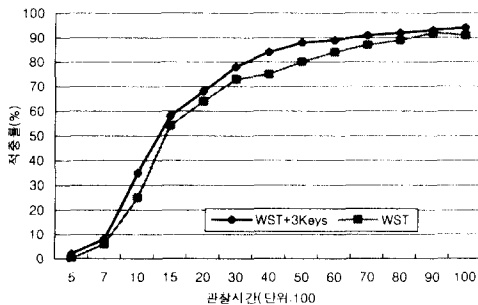
5.2 모의 실험 결과

모의 실험한 내용은 자동 저장 시 데이터의 크기를 결정하는 문제이다. 본 모의 실험에서는 자동 저장에 대해 데이터 유형에 따른 차별 적용을 통해 기존의 일률적 적용과 비교하였다. <그림 3> 은 선택적 불륨만 사용하는 경우보다 3가지의 데이터 유형 별 자동 저장 크기로 가져오는 경우가 적은 양의 데이터를 가져오음을 보인다.



<그림 3> 자동 저장 크기에 따른 비교

다음 모의 실험은 자동 저장 방법에 따른 적중률 비교이다. 사용된 저장 리스트의 크기는 100 개이며 관찰 시간을 늘려가면서 각각의 방법의 적중률을 비교했다.

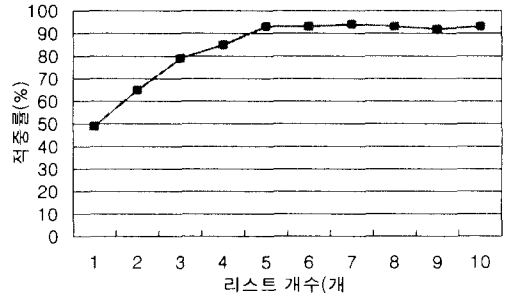


<그림 4> 자동 저장 방법에 따른 적중률 비교

<그림 4> 는 Working Set Tree 알고리즘만 사용하는 경우보다 추가로 3 가지(파일 소유권, 최근 사용내역, 파일의 그룹형성, < 표 1> 참조)을 사용하여 자동 저장한 결과 성능이

향상됨을 보인다. 여기서의 적중률은 이동 클라이언트가 현재 사용하는 데이터를 사용하는 중에 다음 사용 예상 데이터를 가져오는 경우만 고려하였다. 그러므로 현 사용 후 다음 사용할 데이터를 캐쉬에서 살펴보고 이미 자동 저장된 경우 카운터를 증가시켰다. 여기서는 적중률이 향상된 반면 자동 저장 데이터를 선정하기 위해 추가로 수행해야 하는 작업이 수반된다.

마지막으로 이동 클라이언트가 Spy Module 에서 사용하는 리스트의 크기에 대한 실험 결과이다. 열악한 이동 컴퓨팅 환경에서 어느 정도의 적중률을 보장한다면 자동 저장 리스트의 크기도 줄이는 것이 필요하기 때문이다. 관찰 시간은 충분한 사용 유형을 파악하기 위해 10,000 초를 수행하였으며, 자동 저장 방법은 4 가지 모두를 사용하였다. <그림 5> 는 90% 이상의 적중률을 유지하기 위해서는 최소한 50개의 리스트를 보유하고 있어야 한다는 결과가 산출되었다. 따라서 이동 클라이언트 사용자가 일정한 주기로 일정한 작업을 수행하는 일을 대부분 수행한다면 50 개의 리스트를 갖고 충분히 관찰 사용할 수 있다는 결과이다. 본 모의 실험에서 사용된 리스트의 크기는 1 개당 20byte 를 차지한다.



<그림 5> 자동 저장 리스트의 크기

6. 결론 및 향후 과제

본 논문에서는 상대적으로 열악한 이동 컴퓨팅 환경에서 사용자의 데이터 사용 유형을 분석하여, 네트워크의 영향을 감소시켜 사용자 작업을 계속 유지하는 방법에 대해 연구하였다. 이를 위해서는 데이터의 특성에 맞는 선택적 자동 저장 크기를 선택해야 하며, 기존의 판단 알고리즘 외에 추가 판단기준을 도입하면 자동 저장의 적중률이 향상됨을 보인다. 또한, 자동 저장 리스트의 크기에 대한 고찰을 통해 적정 수준의 적중률을 유지하기 위한 최소한의 리스트 크기를 알아야겠다.

향후 과제로서 본 논문의 실험내용을 실제 시스템으로 구현하여 실제 상황과 모의 실험의 상관관계를 알아볼 필요가 있으며, 4 가지 자동 저장 방법 외에 추가적인 자동 저장 방법을 고안하여 적정 수준의 적중률에 도달하는 사용 회수를 줄이는 연구가 수반되어야 할 것이다.

7. 참고문헌

[1] L.B.Houston and P.Honeyman, "Disconnected Operation for AFS", In Proc. Of the 1993 USENIX Symposium on Mobile and Location-Independent Computing, Cambridge, MA, Aug. 1988
 [2] J.J.Kistler and M. Satyanarayanan, "Disconnected Operation in the Coda File System", ACM Trans. On Computer Systems, Vol.10, No.1, Feb.1992
 [3] P.Reiher, J.S.Heidemann, D.Ratner, G. Skinner, and G.J.Popek."Resolving File Conflicts in the Ficus File System ", In USENIX Conference Proceedings, pp.183 -195. USENIX, June, 1994