모바일 클라우드에서 소셜 네트워크 기반 신뢰적인 자원 관리 기법

박지수, 정대용, 임종범, 유헌창†
고려대학교 대학원 컴퓨터교육과
e-mail:bluejisu@korea.ac.kr

Social Network Based Reliable Resource Management Technique in Mobile Cloud

JiSu Park, DaeYong Jung, JongBeom Lim, HeonChang Yu
Dept of Computer Science Education, Korea University

요 약

최근 스마트폰 및 타블릿PC 등 모바일 장치의 성능(쿼드코어)이 급격히 성장하고, 사용자의 증가, 무선 통신 환경(LTE, Wibro 등)의 발전으로 모바일 장치들을 연산 가능한 자원으로 고려하는 모바일 클라우드 연구가 이루어지고 있다. 그러나 자원 제공자인 모바일 장치의 자율적 연산 참여와 탈퇴가 가능하기 때문에 기존의 시스템들은 수행중인 연산이 중단되어 연산의 완료 시간이 지연되며, 시스템의 결함으로 이어지게 된다. 따라서 모바일 장치의 안정적인 연산 수행을 위한 안정적인 모바일 자원을 제공해야 한다. 본 논문에서는 신뢰적인 모바일 자원을 제공하기 위해 소셜 네트워크 정보를 이용하여 그룹화 함으로써 모바일 자원 관리를 제공한다.

1. 서론

모바일 클라우드 컴퓨팅은 유선 통신망 기반인 클라우드 컴퓨팅과 무선 통신망 기반인 모바일 컴퓨팅이 결합된 것으로 모바일 장치를 연산이 가능한 자원 제공자로 사용하기 위한 클라우드 환경을 지원한다. 모바일 장치는 스마트폰뿐만 아니라 일반 피처 폰에서부터 노트북, 넷북, PDA, 태블릿PC 등 이동성을 가지는 모든 기기들을 포함한다. 기존의 모바일 장치는 CPU, 메모리, 배터리들이 성능 면에서 제약이 있으나 최근 스마트폰과 태블릿PC의 급격한 성장과 4G(LTE, Wibro 등)와 같은 무선 통신 환경의 발전으로 모바일 장치의 처리 속도가 빨라졌다. 또한 배터리 용량도 점점 증가하여 이동이 편리해짐으로 모바일 장치를 사용하는 유저들이 증가하고 있다. 이로 인해 클라우드에서 모바일 장치를 연산 가능한 자원으로 활용하기 위한 연구들이 진행 되고 있다[1][2][3]. 그러나 모바일 장치를 연산 가능한 자원으로 사용하기 위해선 해결해야할 문제점이 있다.

모바일 장치는 자율적 연산 참여와 탈퇴가 가능하기 때문에 수행중인 연산이 중단되어 연산의 완료 시간이 지연되며 시스템의 결함으로 이어지게 된다. 이는 연산 수행에 대한 신뢰성이 저하되고 연산의 완료를 보장하지 못하게 되어 모바일 장치를 연산 가능한 자원으로 사용하는데 제한 사항을 만든다. 또한 클라우드와 모바일 자원은 이질적인 특성을 가짐으로써 클라우드에서 처리하던 연산을 모바일 자원에서 처리 할 수 없다.

본 논문에서는 모바일 장치의 안정적인 자원 참여를 위해 소셜 네트워크 정보를 이용하여 그룹화를 함으로써 모바일 자원 관리를 제공한다. 또한 클라우드와 모바일 자원의 이질적인 특성을 고려하여 중간에 프록시를 두어 양쪽의 자원을 모두 활용할 수 있도록 한다.

2. 모바일 클라우드 컴퓨팅

모바일 클라우드 컴퓨팅에서 모바일 장치를 이용하는 것에 관한 연구는 크게 3가지로 나눌 수 있다[4]. 첫 번째는 씬 클라이언트이다. 모바일 장치를 이용하여 클라우드 서비스를 이용하는 것으로 구글의 Gmail 같은 것이다[5]. 현재까지 대부분의 모바일 클라우드 연구는 모바일 장치를 클라우드 시스템에 원격 접속을 하거나, 웹브라우저에서 이메일, 오피스 작업, 자동 백업 등과 같은 인터페이스로만 이용하는 방법이었다. 두 번째는 모바일 장치를 자원 제공자로서 이용하는 것이다[1][2][3][6][7]. 이는 모바일 그리드 및 ad-hoc 환경에서 모바일 장치를 이용한 연구가 클라우드 환경으로 이동되면서 모바일 장치의 자원 활용이 대두되고 있다. 특히 MapReduce와 같은 연산처리를

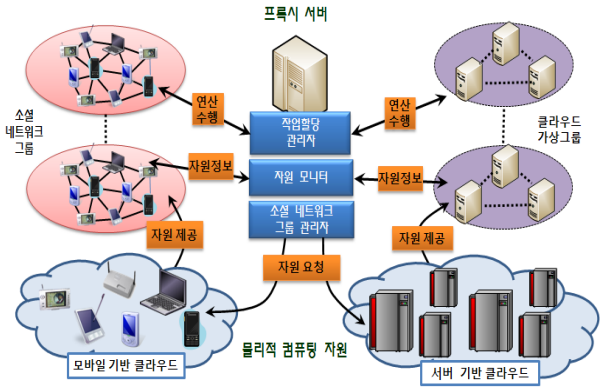
* 본 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임
(No. 20120007429)
† 교신저자

하는 기법에선 작업을 작게 태스크로 나누므로(64MB 수준) 모바일 장치에서도 연산이 가능하다[3]. 세 번째는 모바일과 클라우드 자원들 사이에서 Local Cloudlt을 둠으로써 워드로드를 줄이는 SheevaPlug와 같은 연구가 있다[8][9]. SheevaPlug를 Plug Computer라고도 부르며, 리눅스 기반으로 운용되는 저전력 시스템이다. 이와 같은 컴퓨터는 USB등을 통해 프린터나 외장하드디스크를 연결하여 가정용 미디어 서버, 혹은 사진이나 음악, 영화등 공유할 수 있는 웹서버 등의 가정용 홈 서버용으로 사용한다. 모바일과 클라우드 사이에 Plug 컴퓨터를 두어 모바일에선 Plug 컴퓨터에 접속을 하여 서비스를 이용하고, Plug 컴퓨터는 클라우드에 연결하여 서비스를 제공받는다. 본 논문은 두 번째인 모바일 장치를 자원 제공자로써 이용하는 연구를 위해 자원 관리를 위한 기법을 제안한다.

3. 모바일 클라우드에서 신뢰적인 자원 관리 기법

3.1 시스템 환경

모바일 클라우드 컴퓨팅은 기존의 클라우드 컴퓨팅과 모바일이 결합된 것으로 다양한 모바일 장치를 통해 클라우드 컴퓨팅 환경을 제공한다. 그러나 자율적 연산 참여와 탈퇴가 가능하기 때문에 수행중인 연산이 중단되어 연산의 완료 시간이 지연되며 시스템의 결함으로 이어지게 된다. 또한 클라우드와 모바일 자원은 이질적인 특성을 가짐으로써 클라우드에서의 연산처리를 모바일 자원에서 할 수 없다. 따라서 이를 중재하기 위해 프록시 서버를 둔다. 프록시 서버에서는 클라우드 플랫폼과 모바일 장치의 연결뿐만 아니라 모바일 장치의 제한적 자원과 성능을 보완하고, 모바일 장치들을 관리하기 위한 기능들을 가진다. 제안하는 프록시 기반 모바일 클라우드 시스템에서 모바일 클라우드와 모바일 장치의 구조는 그림 1과 같으며, 자원관리를 위해 3개의 구성요소를 둔다.



(그림 1) 프록시 기반 모바일 클라우드 아키텍처

자원 모니터링은 모바일 장치로부터 이용패턴과 특성정보, 그리고 자원 정보들을 수집한다. 이 수집된 정보들은 작업할당 관리자와 소셜 네트워크 그룹 관리자에 보낸다. 소셜 네트워크 그룹 관리장에서는 모바일 장치의 이용패턴과 특성정보를 분석하여 소셜 네트워크 그룹을 생성하고 관리한다. 작업할당 관리자는 작업이 상태에 따라 모바일 자원 및 클라우드 자원에 작업을 할당하여 관리한다.

3.2 자원 모니터링

모바일 장치의 상태는 자원 정보인 CPU, 메모리, 네트워크, 배터리에 따라 동적으로 변화한다. 따라서 변화 하는 정보를 정확하게 수집하여 분석을 해야 모바일 장치의 상태를 파악할 수 있다. 그러므로 본 논문에서는 모바일 장치의 동적인 상태 패턴을 예측하기 위해 마코브 체인 기반의 모니터링 기법을 제안한다. 이는 시간 상태에 따른 모바일 장치의 동적 상태 변화를 확률 모델링함으로써 정확한 상태를 예측한다. 이를 위해 각각의 상태를 결함이 발생하지 않고 안정적으로 연산이 가능한 상태 S_s(Stable State), 연산이 가능하지만 언제 결함이 발생될지 모르는 불안정한 상태 S_u(Unstable State), 결함이 발생하여 연산이 불가능한 상태 S_d(Disable State)로 구분하여 정의한다. 3개의 상태를 이용한 마코브 체인은 다음 행렬표(그림 2)와 같이 모델링된다. 행렬표에서 S_s^i는 i시간에서 S_s인 상태를 의미한다. 따라서 i시간일 때 각 상태의 발생 확률은 $P_{S_s^i}$, $P_{S_u^i}$, $P_{S_d^i}$ 같이 표현할 수 있다.

$$\mathbf{P} = \begin{array}{c|ccc|c} & S_s^j & S_u^j & S_d^j & \\ \hline & P_{S_s^i S_s^j} & P_{S_s^i S_u^j} & P_{S_s^i S_d^j} & S_s^i \\ & P_{S_u^i S_s^j} & P_{S_u^i S_u^j} & P_{S_u^i S_d^j} & S_u^i \\ & P_{S_d^i S_s^j} & P_{S_d^i S_u^j} & P_{S_d^i S_d^j} & S_d^i \end{array}$$

(그림 2) P 행렬표

$P_{S_s^i S_s^j}$는 i시간의 에서 j시간의 S_s로 전이하는 확률을 나타낸다. 즉, i시간에서 j시간의 상태 전이 확률 값은 다음과 같이 계산할 수 있다.

$$P_j = \sum_{n=1}^{m} P_i \times P_{i-1}$$

m은 상태의 수, $P_i \geq 0$이고, $\sum_{n=1}^{m} P_i = 1$이다.

다음 상태를 예측하기 위해선 이전 상태의 확률 값이 필요하다. 이전 상태의 확률 값을 구하기 위해선 기존 데이터를 이용한다. 기존 상태 정보로 상태 전이 확률을 구하고, 초기상태확률(Π)을 구한다.

$$\Pi = \{\pi_1, \ldots, \pi_n\}$$

π_i는 초기 상태 확률($\pi_i = S_s$이면 S_s가 초기 상태인 확률), n은 상태의 수, $1 \leq i \leq n$, $\sum_{i=1}^{n} \pi_i = 1$이다.

수집된 정보를 이용하여 예측한 모바일의 상태에 따라 모니터링 시간 간격을 조절하여 동적으로 모니터링을 한

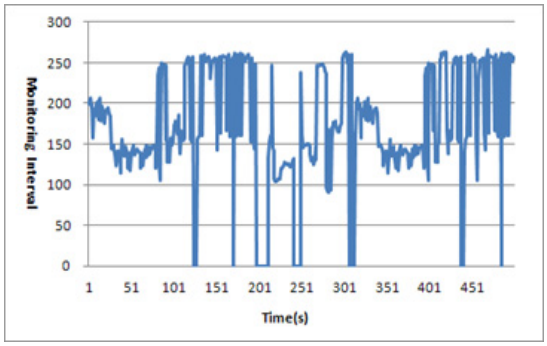
다. S_s에서 모니터링 인터벌을 짧게 하면 불필요한 정보들이 쌓이고, 정보 수집으로 인해 오버헤드가 발생한다. 따라서 예측되는 값이 S_s로 나온다면 모니터링 인터벌은 작업 처리 시간과 S_s의 사용 제한 비율로 계산하여 결정한다. 이러한 S_s에서의 인터벌 시간을 $T_{S_s}^{i \to j}$ (Stable Job Processing Time Interval)이라 한다. T_{job}는 작업 처리 시간이며, $R_x (x = s,u,d)$은 상태의 사용 제한 비율이다.

$$T_{S_s}^{i \to j} = T_{job} \times (1 - R_s)$$

S_u 에서는 자원 정보가 어떤 상태로 변화할 지 알 수가 없으며, S_d로 변경될 수 있다. 따라서 S_s 보다 모니터링 인터벌을 짧게 하여 모니터링을 해야 한다. S_u 에서 인터벌 시간을 $T_{S_u}^{i \to j}$(Unstable Job Processing Time Interval)이라 하며, 작업 처리 시간, S_u의 사용 제한 비율 그리고 S_d의 사용 제한 비율로 계산한다.

$$T_{S_u}^{i \to j} = T_{job} \times (1 - R_u) \times R_d$$

배터리 정보의 경우 작업을 하지 않더라도 시간의 흐름에 따라 계속적으로 변화하게 된다. 이와 같이 정적으로 변화하는 시간에 따른 인터벌을 T_{static}(Static Time Interval)이라 하며, T_{static}는 시스템 관리자에 의해 인터벌을 결정할 수 있다. 그림 3은 모니터링 인터벌 시간을 측정한 것이다.



(그림 3) 모니터링 인터넷 시간 측정

3.3 소셜 네트워크 그룹 관리

소셜 네트워크 그룹은 모바일 장치의 이용 패턴과 모바일 장치의 특성을 분석하여 비슷한 패턴 및 특성을 가진 장치 간에 그룹화를 한다.

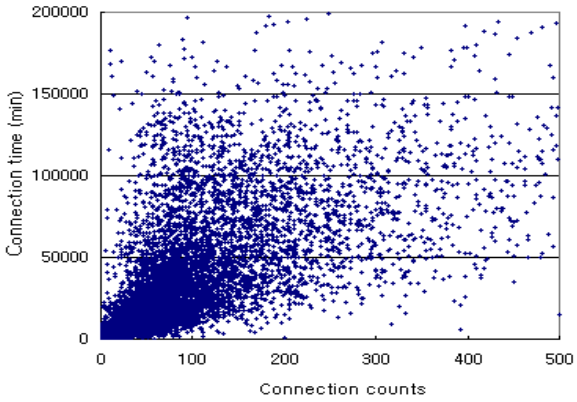
모바일 자원의 이용패턴은 모바일 사용자 측면에서 모바일 장치를 이용하는 패턴에 따라 분석한다. 이용 패턴을 도출하기 위해 무선 네트워크 이용 데이터를 이용하여 다음과 같은 정보들을 분석해야 한다.

- 연결 위치 : 모바일 자원이 사용된 장소이며, 이동 경로에 대한 정보
- 연결 횟수 : 일간 평균 연결 횟수로 유효한 연결의 빈도
- 연결 지속시간 : 각각의 연결 시작시간부터 연결 끊김 시간까지의 시간

기존의 모바일 자원 정보는 CPU, 메모리, 네트워크 대역폭, 배터리 등의 이용률 정보만을 사용하였다. 그러나 이용률은 모바일 자원을 사용할 수 있는 신뢰성 분석에는 사용 가능하지만, 소셜 네트워크를 구성하기 위한 요소로는 사용할 수가 없다. 따라서 모바일 자원을 소셜 네트워크로 구성하기 위해서 다음과 같은 모바일 자원의 특성을 분석한다.

- CPU 타입 : 모바일 자원에서 사용되는 CPU 종류 (ARM, Palm, Intel 등)
- OS 타입 : 모바일 자원에서 사용되는 운영체제의 종류 (Mobile windows, iOS, Android 등)

이와 같은 특성들은 두 가지 의미로 계층적인 그룹 분류에 사용한다. 첫 번째는 연결 위치, 연결 횟수와 연결 지속시간은 모바일 자원의 신뢰성을 위한 그룹화 요소로서, 같은 위치와 함께 많은 연결과 오랜 지속시간을 가지는 자원이 신뢰성이 높다. 따라서 그림 4와 같이 분석된 자료를 이용하여 3개의 그룹($G_{High}, G_{Middle}, G_{Low}$)을 생성한다. 그림 4는 다트머스 대학(Dartmouth college)에서 2005.12 ~ 2006.6 기간 동안 수집된 무선 네트워크 이용 데이터를 분석하였다[10].



(그림 4) 장치별 접속회수와 접속시간의 분포

알고리즘 1. 소셜 네트워크 그룹 생성

```
1   social_group( ) {
2     while(true) {
3       collect resource information through
        monitoring();
4       analyze connection count and connection
        continuous time;
5       specify bounds of connection position;
6       for each mobile device {
7         assign the mobile device to each
          CPU/OS_type group;
8         for each CPU/OS_type group
9           assign the mobile device to 3 group;
            // G_High, G_Middle, G_Low
10      }
```

두 번째는 CPU 타입, OS 타입은 모바일 자원의 이질적인 특성(이질성)을 분류하여 동질의 자원을 그룹화 하기 위한 요소이다. 각 타입별 그룹화를 통해 이절성의 문제를 해결한다.

3.4 작업 할당 관리

모바일 자원을 클라우드에서 활용하기 위해 모바일 하둡 기반의 연산 처리 방법을 제안한다[3]. 하둡 기반의 연산은 모든 태스크의 처리가 끝나야 연산을 마무리 할 수 있다. 그러나 모바일 환경에서는 신뢰성, 이질성 등으로 연산 수행 시 모바일 자원의 결함 문제가 발생한다. 특히 하둡은 한 자원(노드)에서 태스크 처리를 못할 경우 작업을 완료할 수 없으며, 다른 작업 처리를 할 수 없게 되므로 클라우드 환경에서 하둡은 3개의 자원에 복제하여 처리하도록 한다. 그러나 모바일 자원 그룹에서는 3개만으로 잦은 결함의 문제를 해결할 수 없다. 따라서 모바일 자원에 작업 할당할 시 그룹의 결함 수를 이용하여 복제 수를 결정하여 할당한다. 본 논문에서 그룹 j의 복제 수(N_R^j)는 기본 복제 수(N_R^d ; 3개 복제)와 그룹 j의 결함 수(N_F^{gj})를 계산하여 결정한다.

$$N_R^j = N_R^d + \frac{N_F^{gj}}{\sum_{i=1}^{n} N_F^{g_i}} \times 100$$

알고리즘 2. 작업 할당

```
1  job_allocation( ) {
2    while(true) {
3      collect group information through monitoring();
4      analyze task size;
5      for each task {
6        select group; // G_High, G_Middle, G_Low
7        calculate number of replica; // N_R^j
8        assign task to the mobile device;
9      }
```

4. 결론 및 향후 연구 과제

본 논문에서는 모바일 클라우드 환경에서 신뢰적인 자원 관리를 위해 3가지 관리 방안을 제시하였다. 첫 번째로 모바일 장치의 동적 자원 정보를 수집하기 위해 동적 모니터링 인터벌 시간을 적용하였다. 모바일 장치의 상태를 예측하여 모니터링 인터벌 시간을 동적으로 변화 시켜 신뢰적인 정보를 수집 및 분석할 수 있도록 하였다. 두 번째로 소셜 네트워크를 이용하여 계층적인 그룹화 기법을 제시하였다. 모바일 장치간의 소셜 정보를 크게 신뢰성을 위한 정보와 이질성을 위한 정보로 분리하였다. 신뢰성을 위한 연결 위치, 연결횟수 및 지속시간과 이질성을 위한 CPU/OS 타입을 이용하여 그룹을 계층적으로 생성하였다. 세 번째로 신뢰성을 높이기 위해 각 그룹별 복제 수를 결정하여 작업을 할당 하도록 하였다.

향후 연구에서는 복제 수에 따른 작업 할당 알고리즘을 적용하여, 다른 알고리즘과 비교 분석을 한다.

참고문헌

[1] Ghosh, P., Roy, N., Das, S.K.: Mobility-Aware Efficient Job Scheduling in Mobile Grids. In: Proc. of Cluster Computing and Grid (2007)

[2] JongHyuk Lee, SungJin Choi, Taeweon Suh, and HeonChang Yu, “Mobility-aware Balanced Scheduling Algorithm in Mobile Grid Based on Mobile Agent,” The Knowledge Engineering Review, accepted for publication

[3] E. Marinelli, "Hyrax: Cloud Computing on Mobile Devices using MapReduce,", Master Thesis Draft, Computer Science Dept., CMU, September 2009.

[4] Niroshinie Fernando, Seng W. Loke , Wenny Rahayu, Mobile cloud computing: A survey, Future Generation Computer Systems 29, 2013, pp84-106

[5] http://www.google.com/mobile/mail/. (last visit : 2012. 9. 27)

[6] Joni. F., Frank S., Fabio. F., A Virtual Cloud Computing Provider for Mobile Devices, 2010

[7] M. Black and W. Edgar, "Exploring mobile devices as Grid resources: Using an x86 virtual machine to run BOINC on an iPhone," 2009 10th IEEE/ACM International Conference on Grid Computing, IEEE, 2009, pp. 9-16.

[8] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for VM-based cloudlets in mobile computing, IEEE Pervasive Computing 8 (2009) 14 - 23.

[9] http://en.wikipedia.org/wiki/SheevaPlug (last visit : 2012. 9. 27)

[10] Sungjin Song, HeonChang Yu, “The Scheduling method considering the Using Pattern of the Mobile device in Mobile Grid”, Journal of Korea Association of Computer Education, 2008