모바일 그리드 환경에서의 작업 이주 기법[1)]

정대용[o+], 최숙경[+], 박지수[+], 정광식[++], 유헌창[+2)]
[+] 고려대학교 대학원 컴퓨터교육학과,
{karat[o], csukyong, bluejisu, yuhc}@comedu.korea.ac.kr
[++] 한국방송통신대학교 컴퓨터과학과
kchung0825@knou.ac.kr

Job Migration in Mobile Grid

DaeYong Jung[o+], SookKyong Choi[+], JiSu Park[+], KwangSik Chung[++], HeonChang Yu[+]
[+] Dept. of Computer Science Education, Korea University
[++] Dept. of Computer Science, Korea National Open University

1. 서 론

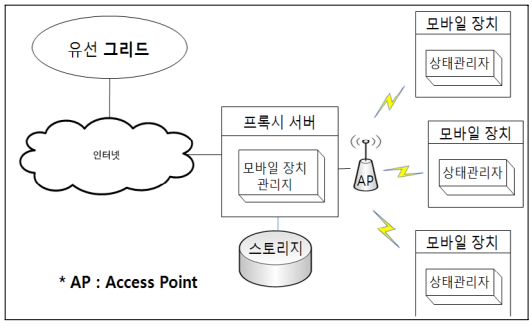
그리드 컴퓨팅은 지리적으로 분산된 컴퓨터 자원을 가지고 대량의 데이터 처리나 복잡한 문제를 해결하는 분산 시스템 환경이다. 최근 노트북, PDA 등과 같은 모바일 장치의 보급으로 이 장치들을 그리드 자원으로 활용하려는 모바일 그리드에 대한 연구가 많이 이루어지고 있다. 그러나 모바일 그리드는 기존 그리드에 비해 모바일 장치의 낮은 성능, 배터리 및 무선 신호의 제약으로 작업을 완료시키지 못하는 문제점을 가진다. 모바일 장치에서 작업 수행 중에 무선 접속이 끊어지거나 배터리가 부족하여 작업을 완료할 수 없는 현상 즉, 결함이 발생할 수 있다. 이런 문제를 해결하기 위해 이용되는 방법들 중 하나가 작업 이주(job migration) 기법이다. 작업 이주 기법은 하나의 모바일 장치에 결함이 발생하더라도 다른 모바일 장치에서 결함이 발생한 시점부터 작업을 이어서 수행할 수 있다[1].

[2]에서는 모바일 에이전트 환경에서 모바일 장치에 메일박스를 구성해서 체크포인팅 하는 방법을 제안하였다. 이 방법에서는 체크포인팅 정보를 가진 모바일 장치에 결함이 발생했을 경우, 체크포인팅 정보가 메일박스와 함께 없어지는 문제점이 있다.

본 논문에서는 모바일 그리드에서 작업 수행 시 발생하는 문제점을 개선하기 위한 기법을 제안한다. 이를 위해 모바일 그리드 환경에서 모바일 장치의 배터리 부족이나 무선 신호세기 약해져서 작업 중 접속이 끊어지는 경우들에 대해 미리 예측하고 작업을 이주할 수 있도록 한다. 작업을 이주하기 위해 현재 작업 상태를 저장해야 하며 이를 위해 체크포인팅 기법을 사용한다. 그 중에서도 사용자 시스템에 독립적인 파일형식으로 작업을 저장하는 사용자 정의 체크포인팅 기법을 사용한다[3]. 이는 일정한 간격으로 체크포인팅을 하는 시스템 레벨 체크포인팅 기법에 비해 사용자가 정한 상태에서 체크포인팅을 하기 때문에 모바일 장치에 더 적합한 방법이다.

2. 시스템 구성

본 논문에서 제안하는 시스템 구성은 다음과 같다. 모바일 장치와 AP(Access Point) 사이에서 무선 신호세기와 모바일 장치의 배터리 용량을 관리하는 상태 관리자를 모바일 장치에 둔다. 그리고 모바일 장치들의 접속 상태와 모바일 장치 등록을 관리하는 모바일 장치 관리자를 프록시 서버에 둔다. 두 관리자를 통해서 모바일 장치의 현재 접속 상태, 무선 신호세기와 배터리 상태를 확인하여 결함이 발생할 수 있는 경우를 감지한다.



[그림 1] 시스템 구성

두 관리자의 기능은 다음과 같다. 상태 관리자는 AP와 모바일 장치에서의 무선 신호세기, 배터리 용량을 확인하여 체크포인팅하는 시점을 결정한다. 모바일 장치 관리자는 모바일 장치의 접속 상태를 확인한다. 그리고 모바일 장치에 결함이 발생하면 스토리지에 저장된 체크포인팅 정보를 이용하여 남은 작업 종료 시간 안에 작업할 수 있는 다른 모바일 장치를 선택하여 작업을 이주시킨다.

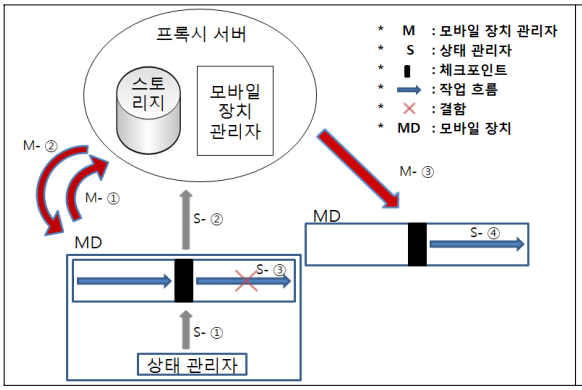
3. 작업 이주 및 작업 이주 조건

체크포인팅 및 작업이주를 위해 두 가지 조건 즉, 모바일 장치의 무선 신호세기와 배터리 용량을 정하고 이들에 대한 임계치를 설정하였다. 이 임계치는 체크포인팅하는 시점을 결정하게 되고, 신호세기나 배터리 용량이 임계치 이하로 내려가면 체크포인팅하게 된다. 또 신호세기나 배터리 용량이 임계치 이하의 설정 값보다 낮아지면 모바일 장치에 결함이 발생한 것으로 판단하여 체크포인팅한 작업을 다른 모바일 장치로 이주해서 계속 작업을 수행한다.

1) 이 논문은 2006년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임 (KRF-2006-311- D00173)
2) 교신저자

모바일 장치에서 다른 모바일 장치로 작업을 이주할 때 새로운 모바일 장치의 선택 기준은 남은 작업 종료 시간이 된다. 스토리지에서 체크포인팅 정보를 가져오고, 모바일 장치들에게 예상되는 남은 작업의 종료 시간을 계산하여 모바일 장치를 선택해서 작업을 이주시킨다.

[그림 2]는 상태 관리자와 모바일 장치 관리자로 구성된 시스템과 그들 간의 작업의 흐름을 보여 준다. 모바일 장치 관리자(M)는 등록된 모바일 장치의 접속 상태를 확인하고, 상태 관리자(S)는 모바일 장치의 무선 신호세기와 배터리 용량을 확인한다. 두 관리자를 통해 결함이 발견되었을 경우, 이를 감지하고 작업을 이주시킬 수 있다.



모바일 장치 관리자(M)

① 모바일 장치를 등록시킴
② 등록된 모바일 장치의 접속 상태를 확인함
③ 남은 작업 종료 시간을 계산하여 작업을 이주할 다른 모바일 장치를 선택함

상태 관리자(S)

① 신호세기와 배터리 용량을 확인한 후 체크포인팅을 결정함
② 모바일 장치에서 체크포인팅 하여 스토리지에 저장함
③ 결함 발생
④ 작업을 이주 받은 모바일 장치는 체크포인트 시점부터 작업을 수행함

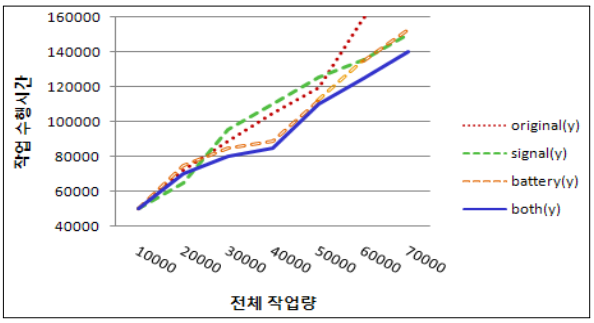
[그림 2] 작업 이주 흐름

4. 성능 평가

성능 평가는 JAVA를 이용하여 작업 이주 기법을 시뮬레이션 하였다. 성능 평가를 하기 위한 가정은 다음과 같다. 첫째, 작업 하나당 배터리 용량 소모를 1셀(cell)로 가정한다. 둘째, 모바일 장치의 초기 위치와 이동하는 위치는 임의로 구성한다.

작업 수행 시간을 다음의 각 경우에 대해 반복 수행하여 결과를 비교하였다.

① 작업 이주 없이 다른 모바일 장치에 작업을 재수행하는 경우(original)
② 신호세기 조건을 사용해서 다른 모바일 장치에 작업을 이주하는 경우(signal)
③ 배터리 용량 조건을 사용해서 다른 모바일 장치에 작업을 이주하는 경우(battery)
④ 신호세기, 배터리 용량 조건을 모두 사용해서 다른 모바일 장치에 작업을 이주하는 경우(both)



[그림 3] 성능 평가 결과

[그림 3]은 시뮬레이션 결과를 보여준다. 이는 본 논문에서 제안한 모바일 장치의 신호세기와 배터리 용량을 모두 고려한 작업 이주 기법이 나머지 경우들보다 전체 작업 수행시간이 줄어들었음을 보여준다. 성능평가 결과, 작업량이 적은 경우에는 4가지 조건에서 비슷한 성능을 보이지만 작업량이 많이 증가하게 되면 ①의 경우에는 더 이상 작업을 수행할 수 없고, ②, ③, ④의 경우에는 작업 이주를 통해서 계속 연산이 가능하다. 또한 ④의 경우는 ①의 경우에 비해 작업량이 많아질수록 더 좋은 성능을 보였다.

5. 결론

본 논문에서는 모바일 그리드 환경에서의 제약으로 발생될 수 있는 사항을 해결하기 위한 작업 이주 기법을 제시하였다. 모바일 장치의 결함을 예측하고, 현재 작업한 데이터를 체크포인팅을 한 후, 결함이 발생했을 경우 체크포인팅 정보를 이용하여 다른 모바일 장치에서 작업을 이어서 수행하도록 한다. 이를 통해 모바일 장치의 제약 사항인 배터리 용량, 무선 신호세기에 대한 문제를 해결할 수 있다. 성능 평가를 통해서 본 논문에서 제시한 작업 이주 기법이 작업 수행에 있어 효율성 및 신뢰성을 향상시킴을 증명하였다.

참고문헌

[1] P.Roe and C.Szyperski. "Transplanting in Gardens : Efficient Heterogeneous Task Migration for Fully Inverted Software Architectures", Proceedings of the Fourth Autralasian Computer Architecture Conference, January 1999.

[2] Jin Yang, Jiannong Cao, Weigang Wu, "CIC: An Integrated Approach to Checkpointing in Mobile Agent Systems", Proceedings of the Second International Conference on Semantics, Knowledge, and Grid (SKG'06), 2006.

[3] Luís Moura Silva, João Gabriel Silva, "System-Level versus User-Defined Checkpointing", Symposium on Reliable Distributed Systems, pp.68-74, 1998.