Journal of Korea Multimedia Society Vol. 13, No. 1, January 2010(pp. 1-9)

대용량 소프트웨어 실행을 위한 모바일 런타임 라이브러리 설계 및 구현

이예인[†], 이종우[††]

요 약

모바일 통신의 발전으로 데스크탑과 같이 위치가 고정적인 시스템 외에도 이동이 편리한 휴대폰 등의 모바일 컴퓨팅시스템이 많이 이용되고 있다. 모바일 단말기의 컴퓨팅 성능이 발전하고 있지만, 많은 계산이나 처리를 요구하는 소프트웨어를 휴대폰 같은 모바일 단말기에서 이용하기는 힘들다. 이 같은 단점을 해소하기 위해 모바일 클러스터 컴퓨팅을 활용하기로 하고, 본 연구에서는 기존 모바일 컴퓨팅 시스템을 분석하였지만 기존의 모바일 클러스터 컴퓨팅 연구들에서는 구현보다는 시스템 구조 제안에 머물고 있거나, 실제 휴대폰 등의 단말기로 구현한 예가 없는 등, 현실에서 활용하기에는 무리가 있는 실정이다. 이에 본 논문에서는 기존의 JPVM 클러스터 시스템에 휴대폰이 참여하도록 하고 클러스터 시스템에서 대용량 소프트웨어를 처리하여 휴대폰에서 그 결과를 볼 수 있도록 하였다. JPVM 클러스터에 참여한 휴대폰 상에서는 병렬 응용의 실행과 종료뿐만 아니라 그 실행 결과도 원하는 형태로 볼 수 있다. 구현된 시스템은 휴대폰이 클러스터 시스템에 참여할 수 있다는 측면에서 Mobile-JPVM이라고 할 수 있으며, 성능평가를 통해 Mobile-JPVM이 대용량 소프트웨어를 실행시키는데 문제가 없음을 확인하였다.

Design and Implementation of a Mobile Runtime Library for Execution of Large-scale Application

Ye-In Lee[†], Jong-Woo Lee[††]

ABSTRACT

Today's growth of the mobile communication infrastructure made mobile computing systems like cellular phones came next to or surpassed the desktop PCs in popularity due to their mobility. Although the performance of mobile devices is now being improved continuously, it is a current common sense that compute intensive large-scale applications can hardly run on any kind of mobile handset devices. To clear up this problem, we decided to exploit the mobile cluster computing system and surveyed the existing ones first. We found out, however, that most of them are not the actual implementations but a mobile cluster infrastructure proposal or idea suggestions for reliable mobile clustering. To make cell phones participated in cluster computing nodes, in this paper, we propose a redesigned JPVM cluster computing engine and a set of WIPI mobile runtime functions interfacing with it. And we also show the performance evaluation results of real parallel applications running on our Mobile-JPVM cluster computing systems. We find out by the performance evaluation that large-scale applications can sufficiently run on mobile devices such as cellular phones when using our mobile cluster computing engine.

Key words: Mobile Computing(모바일 컴퓨팅), Mobile Runtime Library(모바일 런타임 라이브러리), WIPI(위피), Mobile Software(모바일 소프트웨어), Performance Evaluation(성능평가), Mobile Cluster Computing(모바일 클러스터 컴퓨팅)

※ 교신저자(Corresponding Author) : 이종우, 주소 : 서울특별시 용산구 청파2가(608-743), 전화 : 02)710-9952, FAX : 02)710-9704, E-mail : bigrain@sm.ac.kr
접수일 : 2009년 5월 14일, 수정일 : 2009년 11월 30일
완료일 : 2010년 1월 18일
[†] 정회원, 숙명여자대학교 멀티미디어과학과 석사 (E-mail : stickglue@sm.ac.kr)
[††] 종신회원, 숙명여자대학교 정보과학부 멀티미디어과학전공 부교수
※ 본 연구는 숙명여자대학교 2008학년도 교내연구비 지원에 의해 수행되었음

1. 서 론

모바일 단말기의 활용도가 높아지고 모바일 단말기가 처리할 수 있는 소프트웨어 응용 범위가 확장되면서 모바일 단말기가 하나의 컴퓨팅 시스템으로 자리 잡고 있다. 하지만 아직도 모바일 단말기의 하드웨어에는 여러 성능적 한계가 있기 때문에 모바일 단말기에서 실행시킬 수 있는 소프트웨어는 제한적일 수밖에 없는 형편이었다. 본 논문에서는 이와 같은 문제점을 해결하기 위해 모바일 단말기가 기존 PC들로 구성되는 클러스터 컴퓨팅에 참여하게 함으로써 모바일 단말기에서도 대용량 소프트웨어를 실행시킬 수 있도록 하였다. 클러스터 컴퓨팅은 병렬 컴퓨팅의 일종으로써 네트워크로 연결된 여러 이기종의 컴퓨터를 하나의 대형 컴퓨터처럼 활용하는 고성능 컴퓨팅 기술인데, 지금까지는 PC와 같은 워크스테이션으로 구성되는 것이 일반적이었다. 클러스터 컴퓨팅 환경에 모바일 단말기가 참여하기 위해서는 모바일 단말기에서 프로그래밍이 가능한 언어를 고려하지 않을 수 없다. 본 논문에서 가정하고 있는 모바일 단말기는 휴대폰인데 현재 국내에서 사용되고 있는 휴대폰 상에서는 대부분 자바로 작성된 소프트웨어가 실행되고 있어서 자바로 구현된 클러스터 컴퓨팅 환경이 필요했으며 또, 자바는 WIPI Jlet 모바일 소프트웨어를 이식할 수 있고 일관된 API를 제공하며 플랫폼에 독립적인 병렬프로그래밍 환경을 제공하므로 자바로 구현된 JPVM을 클러스터 컴퓨팅 기반 플랫폼으로 사용하였다.

대용량 소프트웨어를 휴대폰 상에서도 실행시키는 것이 가능하게 만든 본 논문의 의의는 기술적 측면 이외에도 다음과 같은 필요성이 있기 때문이었는데, 모바일 인터넷 사용자의 급증과 모바일 데이터 통신 속도의 급속한 발전, 하지만 이에 부응하지 못하는 모바일 소프트웨어의 발전 속도 등이 그것이다.

현재 모바일 장치의 기술이 컴퓨팅 분야와 통신 분야에서 급속하게 발전하고 있다. 영국의 ARC Group 예측에 의하면 2010년 내에 약 10억명 이상의 인터넷 접속자가 생길 것이며, 이중 7억 5천만 명이 휴대형 이동 단말기를 사용할 것이라고 예상되는 반면, 유선 사용자는 6억 7천만 명이 될 것으로 예측하고 있다[1]. 이는 휴대폰이 정적인 통화 단말기에서 네트워크가 가능한 컴퓨팅 요소(CE: Computing Element)로 발전하고 있음을 의미한다. 이같은 현재 상황으로 볼 때 휴대폰이 클러스터 컴퓨팅의 한 노드로 참여하는 것은 휴대폰의 하드웨어적 한계를 극복하는 좋은 현실적 대안이 될 것으로 보인다. 기존에도 Moset[2], MCC[3]와 같이 모바일 단말기를 클러스터의 한 노드로 포함하는 연구가 진행되었었지만, 이들은 실제적으로 모바일 단말기를 클러스터의 노드로 포함한 시스템 구현을 제안한 것이 아니라, 이상적인 환경에서 적용할 수 있는 하나의 패러다임을 제시한 것에 불과했다. 본 논문에서는 휴대폰이 JPVM의 노드로 참여하여 휴대폰에서 병렬 응용의 모든 실행 상황을 실행부터 종료에 이르기까지 볼 수 있게 하였으며, 이를 위해 기존 JPVM을 확장 재설계하였고, 모바일 노드를 위한 WIPI Jlet 라이브러리를 구현하였다.

2. 관련연구

2.1 모바일 클러스터 관련 기존 연구

Moset[2]은 유무선 네트워크로 연결되는 클러스터로써 모바일 노드와 정적 노드를 통합하기 위해 운영체제 및 네트워크 프로토콜 수준에서 모바일 단말기가 갖는 제약과 이질성을 해결하고자 하였다. Moset의 시스템 구조는 그림 1과 같은데, 이 그림에서 알 수 있듯이 Moset에서는 모바일 호스트(MH)들과의 통신 신뢰도를 높이는 데에 그 주안점이 있다.

인접한 몇몇 모바일 호스트들을 묶어 CELL로 엮어두고 중재자를 거쳐 서로 통신하게 하고 있다. 데이터전달의 신뢰도를 위해 모바일 단말기의 제약을 고려하여, Moset은 신뢰성있는 멀티캐스트 프로토콜을 제안하고 있다. 따라서 모바일 노드와 정적 노드간의 통신 프로토콜에 초점을 맞춘 연구라고 할

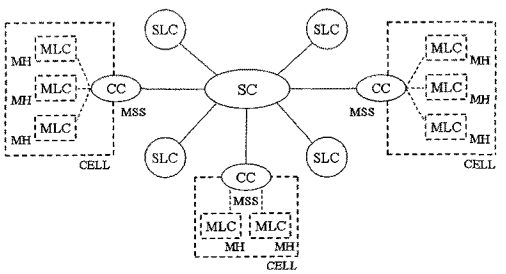

그림 1. Moset 시스템 구조

수 있으므로, 모바일 클러스터 컴퓨팅을 위한 런타임 라이브러리를 실제 구현하고 있는 본 논문과는 그 성격이 다르다 하겠다.

하이브리드 클러스터 컴퓨팅[4]에서는 유무선으로 연결된 모바일 장치와 정적 컴퓨터로 구성되는 클러스터를 하이브리드 클러스터라고 칭하고 있는데, 하이브리드 클러스터를 구성하기 위한 자바 모바일 객체와 모바일 IP 시스템을 제안하고 있다. 전체 시스템은, 컴퓨터와 하이브리드 클러스터 컴퓨팅을 지원하는 네트워크와 무선 모바일 IP와 모바일 객체를 지원하는 클러스터 미들웨어, 그리고 병렬 분산 응용프로그램으로 구성된다. 이 연구는 이 세 부분 중 클러스터 미들웨어를 중점으로 설명하고 있고 모바일 장치로는 휴대폰이 아닌 노트북을 대상으로 하고 있다. 노트북과 휴대폰은 우선 운영체제가 다르고 노트북은 고정IP가 할당되지만, 본 논문에서 가정하는 휴대폰은 고정IP가 할당되지 않으므로 구현 방식이 달라질 수밖에 없다. 따라서 이 연구도 역시 본 논문과는 초점이 다르다고 하겠다.

2.2 병렬 프로그래밍 환경 관련 기존 연구

PVM(Parallel Virtual Machine)[5,6]은 메시지 전달 라이브러리를 기반으로 분산된 이기종 워크스테이션들을 단일 가상자원 또는 단일 병렬 시스템으로 보이게 하여, 병렬 응용 프로그램의 수행과 개발을 손쉽게 하기 위한 소프트웨어 도구이다. C 또는 Fortran으로 작성된 PVM 라이브러리는 점대점 데이터 전송, 메시지 브로드캐스팅, 상호 배제, 프로세스 제어, 프로세스 동시화 등을 통해 병렬 응용프로그램 개발자에게 손쉬운 프로그래밍 환경을 제공한다. PVM 라이브러리를 이용하여 작성된 하나의 PVM 응용은 여러 개의 PVM 태스크들로 구성되는데 PVM 태스크들은 워크스테이션 클러스터에 참여하는 여러 호스트들에 투명하게 분산되어 동시에 병렬로 수행되는 프로세스들이다. PVM은 C 또는 Fortran으로 작성된 라이브러리이므로 본 논문에서 이를 기반 플랫폼으로 사용하기에는 적절치 않았다.

TPVM(Thread-oriented PVM)[7]은 저용량 처리기들 간의 병렬처리와 스케줄링을 스레드단위로 제공하는 PVM 유사 서브시스템이다. TPVM은 데이터 의존성에 기반하여 계산을 분해하는 데이터 스케줄링 뿐만 아니라 전통적인 태스크기반 메시지 전달 라이브러리를 제공한다. 스레드 기반 분산 컴퓨팅 인터페이스와 스레드 관련 서비스를 제공하는 스레드 인터페이스 모듈, 그리고 스케줄링과 시스템 데이터 관리를 수행할 수 있는 스레드 서버 모듈로 구성된다. 하지만 TPVM은 개발이 완료되지 않은 라이브러리이고, 특정 응용 프로그램에 대해서만 실험해 봤을 뿐 모든 PVM 응용프로그램이 다 실행되는 것은 아니라고 밝히고 있어 본 논문에서 기반시스템으로 사용하기에는 무리가 있다고 할 수 있다.

2.3 JPVM

JPVM[8]에서 제공하는 프로그래밍 인터페이스는 PVM의 인터페이스와 비슷하지만 네트워크 병렬 프로그래밍을 위한 언어로는 자바를 사용한다. 즉, PVM의 자바 버전인 셈이다. JPVM API가 PVM과 완전히 같지는 않지만 PVM에 익숙한 프로그래머가 PVM 코드를 자바용으로 변환하고 JPVM 시스템을 쉽게 구축하도록 하는 환경을 제공한다. 동시에, JPVM은 PVM에서는 제공되지 않던 스레드 안정성, 테스크 당 다중 통신 종단점, 직접 메시지 루틴과 같은 새로운 특징을 제공한다. JPVM은 전부 자바로 구현되어 있어서 자바가상머신을 제공하는 플랫폼 사이에서 높은 이식성을 나타낸다.

JPVM의 시스템 구조는 그림 2와 같은데, 크게 마스터 노드와 슬레이브 노드로 나눌 수 있다. 마스터 노드는 클러스터에서 태스크 분배, 결과 수집, 슬레이브 노드관리 등의 역할을 한다. 슬레이브 노드는 마스터 노드로부터 태스크를 할당받아 작업을 수행하고 결과를 마스터 노드로 전송한다.

마스터 노드는 데몬, 콘솔, 응용프로그램의 세부분

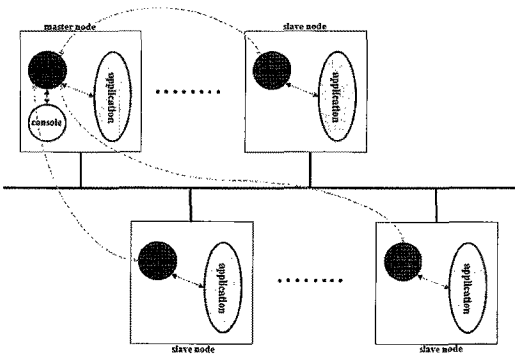

그림 2. JPVM 시스템 구조

으로 나눌 수 있다. 데몬은 JPVM에서 마스터 노드와 슬레이브 노드에 관련된 모든 연결기능을 담당하는 백그라운드 프로세스로서 데몬이 하는 작업은 다음과 같다.

- 태스크 생성
- 콘솔에서 추가한 호스트를 호스트 목록에 추가
- 미리 정해 놓은 메시지태그로 요청 내용을 구분하여 적절한 함수를 호출, 즉 서비스.

콘솔은 각 노드의 JPVM 데몬과 상호작용하는 명령 라인 프로그램으로 콘솔은 사용자의 입력을 받아 적절한 함수를 호출한다. 콘솔이 담당하는 작업은 다음과 같다.

- 호스트의 추가
- 클러스터를 구성하는 호스트의 상태정보 출력
- 데몬의 강제 종료

응용프로그램은 JPVM에서 제공하는 PVM 유사 API를 이용해 작성된다. 응용프로그램에서 사용하는 API는 jpvmEnvironment 클래스에 구현되어 있다.

jpvmEnvironment 클래스는 JPVM 시스템에서 가장 중요한 태스크 생성 기능을 담당한다. 병렬 응용 프로그램 시작 초기에 생성되어 데몬과 응용 사이, 또는 데몬과 콘솔 사이의 연결을 담당한다. jpvmEnvironment 클래스에서 응용프로그램이 사용하도록 제공하는 중요 API들을 자세히 살펴보면 표 1과 같다.

3. Mobile-JPVM 설계 및 구현

3.1 Mobile-JPVM의 전체적 실행 설계

휴대폰이 노드로 참여한 JPVM은 그림 3과 같이 작업지시를 휴대폰에서 서버의 JPVM 마스터로 전송하면(①), 마스터 노드는 휴대폰과의 통신을 위해 실행한 starserver에서 휴대폰이 보낸 메시지를 수신한 후 그 메시지를 해당 응용 프로그램에 전달한다(②). 해당 응용 프로그램은 메시지가 지시하는 작업

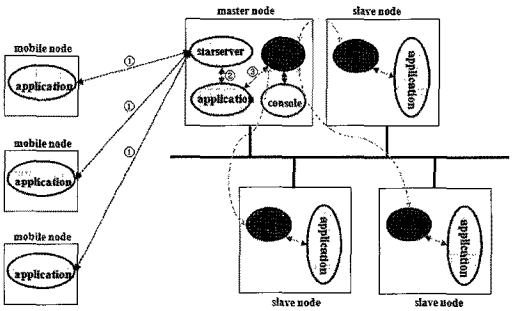

그림 3. 모바일 노드가 참여한 JPVM

표 1. jpvmEnvironment 클래스 메소드 중 응용 프로그램에서 사용하는 중요 함수들

기능요약	메소드 이름	설 명
태스크 생성	pvm_spawn(String task_name, int num, jpvmTaskId tids[])	새로운 JPVM 프로세스들을 생성한다. 매개변수 task_name을 통해 전달된 파일 명칭을 가진 실행 가능한 파일들을 매개 변수 num만큼 복제하고 미리 생성한 태스크 식별자인 tids[]를 할당한다.
메시지 송수신	pvm_send(jpvmBuffer buf, jpvmTaskId tid, int tag)	buf에 저장되어 있는 메시지를 태스크 식별자가 tid인 태스크(프로세스)에게 송신한다. tag는 서로 다른 메시지들을 구별하기 위하여 프로그래머가 제공하는 정수이다.
	pvm_mcast(jpvmBuffer buf, jpvmTaskId tids[], int ntids, int tag)	buf에 저장되어 있는 메시지를, 태스크 식별자 배열 tids에 등록된 모든 프로세스에게 송신한다. ntids는 수신할 태스크 식별자의 개수이고, tag는 서로 다른 메시지들을 구별하기 위하여 프로그래머가 제공하는 정수이다.
	pvm_recv(jpvmTaskId tid, int tag) / pvm_recv(jpvmTaskId tid) / pvm_recv(int tag)/ pvm_recv()	메시지 태그인 tag를 가진 메시지가 프로세스 tid로부터 도착할 때까지 기다리다가, 수신되면 jpvmMessage 객체에 저장한다. 이는 태스크가 어떤 종류의 메시지든 요청할 수 있도록 하기 위함이다.
태스크 아이디 확인	pvm_mytid()	처음 호출되었을 때에는 호출 프로세스를 데몬에 등록하고, 매번 호출될 때마다 프로세스의 태스크 식별자 tid를 돌려준다.
	pvm_parent()	호출한 프로세스를 생성한 프로세스(즉, 자신의 부모 프로세스)의 태스크 식별자 tid를 돌려준다.

표 2. 모바일 지원을 위해 마스터노드에 추가한 클래스들

추가한 클래스	기 능
jpvmMenvironment	jpvmEnvironment 클래스의 기능을 감싸고(wrapping) 모바일로부터의 요청을 받기 위해 소켓통신을 초기화하고 작업요청을 수신한다.
jpvmDataCommunication	모바일과의 통신을 동기화하기 위해 스레드를 생성한다.
jpvmStarServer	모바일과의 통신을 위한 소켓을 생성하고 모바일의 요청을 대기한다.
jpvmClientManager	모바일로부터 요청을 수신하고 요청결과를 모바일로 송신한다.

을 본 연구에 의해 수정·확장된 JPVM 라이브러리를 이용하여 병렬 처리한다(③). 병렬 처리 후 생성된 결과를 starserver를 통해(②) 휴대폰으로 전송하여(①) 휴대폰에서 출력결과를 확인할 수 있도록 한다.

원래의 JPVM은 정적인 노드만을 지원하는 병렬 클러스터 라이브러리이기 때문에 휴대폰의 요청을 처리하는 부분이나 결과를 휴대폰으로 전송하는 부분, 또는 휴대폰에서 사용할 JPVM 대응 WIPI 라이브러리 부분 등이 지원되지 않는다. 기존 JPVM에 모바일 노드를 참여시키기 위하여 표 2와 같이 마스터 노드에 모바일 노드와의 통신을 위한 jpvmStarServer 클래스와 jpvmClientManager 클래스를 추가하였고, 모바일 노드로부터의 병렬 처리 요청을 지원하기 위해 jpvmMenvironment 클래스를 추가하였다. jpvm Menvironment에서 추가한 기능인 모바일과의 통신 연결, 메시지 수신, 결과의 전송 기능 모두가 PC와의 작업이 아닌 클러스터에 추가된 휴대폰과의 작업을 위한 것이다. 슬레이브 노드는 모바일과 직접적인 통신을 하지 않기 때문에 변경되는 부분이 없다.

3.2 마스터 노드 확장

앞서 언급한 바와 같이 마스터 노드에는 모바일과의 통신을 위해 jpvmMenvironment, jpvmStarServer, jpvmClientManager, jpvmDataCommunication 클래스가 추가되었다. 그림 4는 이렇게 확장된 마스터 노드의 클래스 구조를 보이고 있다.

이들 각 클래스들에 대해 살펴보면 다음과 같다.

3.2.1 jpvmMenvironment 클래스

모바일 노드와 통신하기 위해 추가한 클래스이다. jpvmEnvironment 클래스의 기능을 감싸고 모바일에서 요청을 받기 위해 소켓통신을 초기화하고 작업요청을 수신한다. 또, 작업결과를 저장하고 모바일로 전송하는 함수를 제공한다. 표 3은 jpvmMenvironment

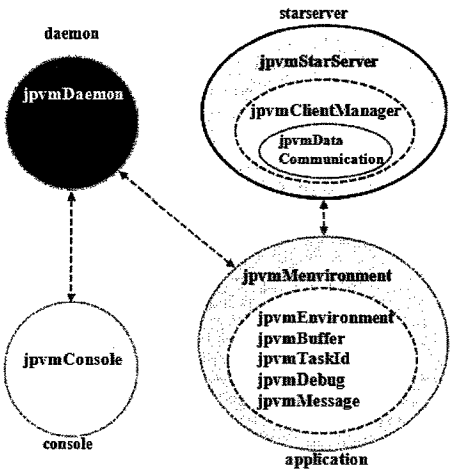

그림 4. 모바일 노드를 위해 확장된 마스터 노드 구조

클래스의 메소드들에 대하여 자세히 보이고 있다.

3.2.2 jpvmStarServer 클래스

여러 모바일 노드들에서 마스터 노드로 동시에 접속을 요청할 경우 각 요청별로 소켓을 생성하고 jpvmClientManager를 생성하여 모바일 노드의 연결 상황을 관리한다. 모바일 노드와의 접속이 종료될 경우 remove()를 호출하여 생성된 소켓을 제거한다. 표 4는 jpvmStarServer의 메소드들을 보이고 있다.

3.2.3 jpvmClientManager 클래스

모바일 노드에서 마스터 노드로 데이터를 전송할 경우 jpvmClientManager에서 recvData()를 이용하여 데이터를 수신한다. 수신한 후 jpvmData-Communication 클래스에서 생성한 스레드를 깨워 데이터가 수신된 것을 알린다. 모바일로 데이터를 전송하는 경우 sendData()를 이용한다. 표 5에서 jpvmClientManager의 주요 메소드들을 설명하고 있다.

표 3. jpvmMenvrionment 클래스 메소드 설명

기능요약		메소드	설 명
생성자		jpvmMenvironment()	모바일로부터 요청을 받기 위한 스레드를 생성하고 통신을 위한 소켓연결을 초기화한다. 모바일로부터 요청이 수신되면 해당 메시지를 읽는다.
메시지 수신		String recvMsg()	모바일로부터 들어오는 메시지를 대기하고 메시지가 오면 메시지를 수신한다. 받은 메시지를 String 타입으로 바꾸어 리턴한다.
		byte[] recvByteMsg()	모바일로부터 들어오는 메시지를 대기하고 메시지가 오면 메시지를 수신한다. 받은 메시지를 바이트배열 타입으로 리턴한다. 보통, 모바일에서 이미지 같은 바이너리 데이터를 수신할 경우에 사용한다.
결과 전송	메시지	void sendMsg(String msg)	모바일로 String 타입의 메시지를 송신한다. 보통, 모바일노드로 단순한 메시지를 전송할 경우에 사용한다.
	텍스트	void sendText(String textfile)	결과파일이 생성된 후 모바일로 텍스트파일의 내용을 전송한다. 결과파일이 생성된 후 전송하기 위해 jpvmDataComunication 클래스를 이용하여 동기화시킨다.
	이미지	void sendImg(File ff)	결과파일이 생성된 후 모바일로 이미지 파일을 전송한다. 결과파일이 생성된 후 전송하기 위해 jpvmDataComunication 클래스를 이용하여 동기화시킨다.

표 4. jpvmStarServer 클래스 메소드 설명

기능요약	메소드	설 명
생성자	jpvmStarServer(jpvmDataComunication dc)	8000번 포트로 소켓을 생성한다.
소켓 생성	void run()	모바일 노드들로부터의 통신요청을 받아들이고 각 요청마다 jpvmClientManager를 생성하고 실행한다.
소켓 제거	void remove(jpvmClientManager sck)	생성된 소켓을 제거한다.

표 5. jpvmClientManager 클래스 메소드 설명

기능요약	메소드	설 명
메시지 수신	byte[] recvData()	모바일 노드에서 보내는 메시지를 수신한다. 수신된 메시지를 바이트 배열 형태로 반환한다.
메시지 전송	void sendMsg(String go)	모바일 노드로 메시지를 송신한다. String 타입의 메시지를 송신할 경우에 사용하는데, 보통 텍스트 메시지를 송신하는 경우에 사용한다.
	void sendMsg(byte[] go)	모바일로 메시지를 송신한다. byte 배열 타입의 메시지를 송신할 경우에 사용하는데 보통 이미지 등의 바이너리 파일을 송신할 경우에 사용한다.
동기화	void run()	모바일 노드로부터의 메시지를 수신하기 위한 스레드를 생성하고 메시지를 수신하면 jpvmDataCommunication 객체에 메시지가 수신되었다고 알린다.

3.2.4 jpvmDataCommunication 클래스

메시지 송수신 시에 응용 프로그램들간의 동기화를 위한 클래스이다. 그림 5와 같이 해당 응용프로그램에서 메시지를 대기하고 있는데, 모바일에서 메시지를 송신하면 jpvmClientManager 객체가 메시지를 수신하고 jpvmDataCommunication 객체에 알려준다. 그러면 jpvmDataCommunication 객체는 해당 응용 프로그램에 메시지를 전달한다.

3.3 모바일 노드

모바일 노드는 그림 6과 같이 서버의 마스터 노드와 소켓통신을 초기화하고 마스터 노드와 메시지를

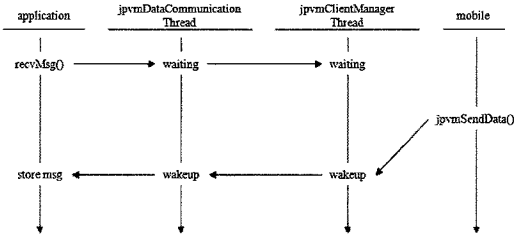

그림 5. 응용프로그램과 모바일간의 메시지 흐름

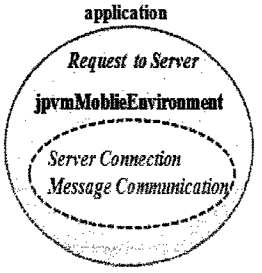

그림 6. 모바일노드 구조

주고받는다. 메시지에는 실행할 응용 프로그램 이름과 돌려받을 결과파일의 종류 및 개수 등이 포함된다. 실행이 끝난 후에는 결과 데이터의 종류에 따라 적절한 함수를 호출하여 데이터를 수신한다. 결과 데이터 유형이 텍스트인 경우 스트링의 형태로 수신하고, 이미지인 경우에는 바이트 형태로 수신하여 모바일 기기 내 특정 디렉토리에 이미지 파일의 형태로 저장한다. 마스터 노드와 통신 시에 모바일에서 사용하는 함수들은 jpvmMobileEnvironment 클래스로 구현하였다. 표 6은 jpvmMobileEnvironment 클래스의 멤버함수들을 자세히 보이고 있다.

4. 성능평가

본 논문에서 구현한 Mobile-JPVM의 정상 동작 여부와 성능을 평가하기 위해 다음과 같은 성능평가 프로그램을 구현, 실행하고 그 결과를 측정, 분석하였다.

- 16개의 태스크로 나눈 256×256 행렬 곱셈 프로그램
- 40개의 태스크로 나눈 정적태스크 할당 방법을 이용한 600×600 프렉탈 이미지 생성 만델브로트집합 프로그램
- 모바일 노드의 요청에 따라 8개의 프레임을 갖

표 6. jpvmMobileEnvironment 클래스의 함수 설명

기능요약	함수	설 명
소켓연결	jpvmServerConnect(String url)	매개변수 url에 전달된 마스터 노드의 주소로 소켓연결을 시도한다.
데이터 전송	jpvmSendData(String str)	String 타입의 메시지를 마스터 노드로 전송한다.
	jpvmSendData(byte[] buff)	byte 배열 타입의 메시지를 마스터 노드로 전송한다.
	jpvmSendImage(File file)	마스터 노드로 전송할 이미지를 File 객체로 생성한 후, 그 객체를 마스터 노드로 전송한다. 전송 중에는 다이얼로그를 생성해 사용자에게 전송상황을 알려준다.
결과 수신	jpvmRecvMsg()	마스터 노드로부터 간단한 메시지를 전송받는다. 받은 메시지를 string 타입으로 리턴한다.
	jpvmRecvText()	마스터 노드로부터 결과 메시지를 전송받는다. 받은 메시지를 string 타입으로 리턴한다.
	jpvmRecvImage(String img)	마스터 노드로부터 이미지 파일을 전송받아 휴대폰 내의 특정 디렉토리에 저장한다.
결과 뷰잉	jpvmTotalPage(String text, int width, int height)	text를 휴대폰 화면의 출력 사이즈인 width×height로 휴대폰에 출력할 페이지 수를 계산한다.
	jpvmViewPage(String text, int page)	페이지 번호에 맞는 텍스트를 한 화면에 출력할 라인의 수만큼 배열을 생성하여 리턴한다.
	jpvmReadImage(String filePath)	매개변수 filePath의 이미지를 읽어 이미지를 생성한다. 생성한 이미지를 리턴한다.
	jpvmScaleImage(Image src, int dstW, int dstH)	이미지를 출력하고자 하는 크기에 맞추어 재조정한다.

는 다수의 animatedGif 이미지를 생성하는 animatedGif 실시간 생성 및 뷰잉 프로그램

그림 7은 병렬 행렬 곱셈 프로그램을 클러스터 노드의 종류와 수를 변화시키면서 측정한 결과이다.

t_s : Execution time using one processor

t_p : Execution time using multiprocessor with processors

$$S(n)=\frac{t_s-t_p}{t_s}\times 100 \qquad \text{수식 (1) 성능 향상율}$$

수식 (1)을 이용하여 성능 향상율을 계산하면, 휴대폰+PC 2대 클러스터의 경우는 약 15%, 휴대폰+PC 4대 클러스터의 경우는 약 48%의 성능향상을 보인다. Mobile-JPVM을 이용하여 실행할 경우에 1초 ~ 2초정도의 시간만 소요되기 때문에 사용자가 빠른 실행 결과를 얻을 수 있다.

그림 8은 만델브로트집합 연산 프로그램 실행 시간을 클러스터 노드의 종류와 수를 변화시키면서 측정한 결과이다. 수식 1을 이용하여 성능 향상율을 계산하면 휴대폰+PC 2대 클러스터의 경우는 약 23%, 휴대폰+PC 4대 클러스터의 경우는 약 25%의 성능향상을 보인다. 만델브로트 집합 설계 시 태스크를 40개로 분할계산하기 때문에 각 태스크간의 통신시간이 많이 발생한다. 이와 같은 이유로 병렬 행렬 곱셈 프로그램에 비해서 만델브로트 집합 연산 프로그램의 성능 향상율이 낮게 측정된 것으로 판단된다.

그림 9는 animatedGif 생성 및 뷰잉 프로그램을 클러스터 노드의 수를 변화시키면서 측정한 결과이다. animatedGif 생성 및 뷰잉 프로그램은 휴대폰이 가지는 컴퓨팅 능력의 한계로 인하여 단독으로 실행할 수 없어서 휴대폰만의 실행결과는 얻지 못하였다. 실행시간을 기준으로 성능 향상율을 계산하면 휴대

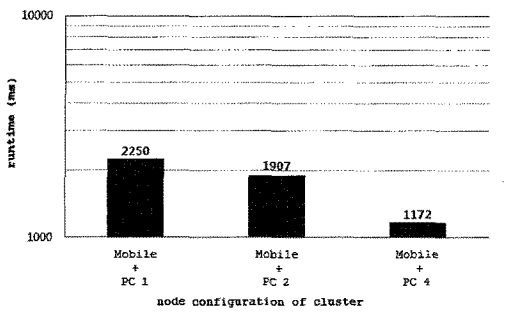

그림 7. 병렬 행렬 곱셈 프로그램 실행시간 측정 결과

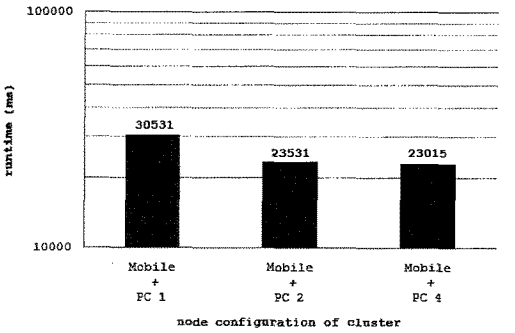

그림 8. 만델브로트집합 연산 프로그램 실행시간 측정 결과

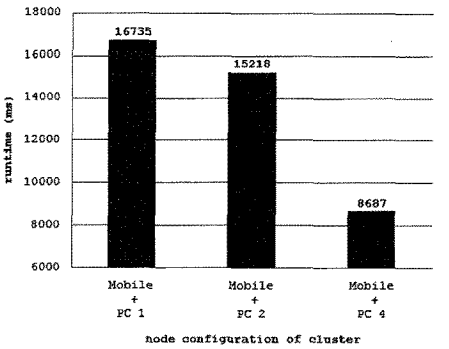

그림 9. animatedGif 생성 및 뷰잉 프로그램 실행시간 측정 결과

폰+PC 1대 클러스터 상에서보다 휴대폰+PC 2대 클러스터의 경우에 약 9%, 휴대폰+PC 4대 클러스터의 경우는 약 92%의 성능향상을 보인다. 이와 같이 클러스터를 4대 이상으로 구성할 경우 휴대폰에서 작업지시를 내린 후 10초 이내에 결과가 생성되므로 사용자가 불편없이 실시간으로 이미지를 얻을 수 있다.

5. 결론 및 향후 연구

본 논문에서는 컴퓨팅 능력이 부족한 모바일 단말기의 한계를 극복하기 위해 플랫폼 이식성이 높은 Java로 구현된 JPVM을 이용하여 Mobile-JPVM 라이브러리를 확장설계, 구현하고 이를 이용하여 여러 병렬 응용 프로그램을 구현, 실행하였다. 응용 프로그램의 실행 결과를 바탕으로 연구 성과를 평가해보면 다음과 같다. 첫째, 모바일 단말기의 부족한 컴퓨팅 능력으로 인하여 실행할 수 없었던 대용량의 소프트웨어를 클러스터로 구성된 서버와 모바일이 연동되게 하여 모바일 단말기에서 실행할 수 있도록 하였

다. 이것으로 모바일 단말기가 가지는 부족한 컴퓨팅 능력을 높여 모바일 단말기에서 실행할 수 있는 소프트웨어의 한계를 상당부분 해결할 수 있다. 둘째, Mobile JPVM 라이브러리를 이용하여 기존의 PC들로만 이루어진 클러스터에서 사용되었던 병렬 응용 프로그램을 쉽게 모바일이 참여하는 클러스터에로 이식할 수 있게 되었다.

다만, 본 논문에서 실험한 병렬 응용 프로그램들 중 animatedGif 생성 및 뷰잉 프로그램을 제외하면 실용적이라기 보다는 성능 평가를 위한 프로그램들이다. 향후 실험만을 위한 프로그램보다는 실용적이고 모바일에서 구현되었을 경우 더 많은 이점을 가지는 여러 병렬 응용 프로그램들을 구현하는 것이 목표이다.

또한 모바일 환경에서 발생할 수 있는 통신 단절이나 모바일 노드들의 이동 등으로 인해 발생할 수 있는 여러 문제에 대한 대안을 제시하는 것도 향후에 해결해야할 연구이고, 개발한 모바일 노드용 WIPI Jlet 라이브러리의 범용성을 높이는 것도 향후 해결해야 할 연구이다.

참 고 문 헌

[1] ARC Group, "Future Mobile Computing: Device Trends and Wireless Solutions 2002-2007,"

[2] M.A. Maluk Mohamed, A. Vijay Srinivas and D. Janakiram, "Moset:An anonymous remote mobile cluster computing paradigm," *Journal of Parallel and Distributed Computing*, Vol.65, No.10, pp. 1212-1222, 2005.

[3] Haihong Zheng, Rajkumar Buyya and Sourav Bhattachaya, "Mobile Cluster Computing and Timeliness Issues," *Informatica: An International Journal of Computing and Informatics*, Vol.23, No.1, pp. 5-17, 1999.

[4] L. Cheng, A. Wanchoo and I. Marsic, "Hybrid Cluster Computing With Mobile Objects," The Fourth International Conference/Exhibition, Vol.2, pp. 909-914, 2000.

[5] A. Geist, A. Beuelin, J.Dongarra, W.Jiang, R.Manch, and V. Sunderamm, "PVM 3 User's Guide and Reference manual," Oak Ridge National Laboratory, Oak Ridge, Tennesee 37831, 1994.

[6] R. J. Manchek, "Design and Implementation of PVM Version 3," Master Thesis, University of Tennessee, Knoxville, 1994.

[7] A. Ferrari and V.S. Sunderam, "TPVM: distributed concurrent computing with lightweight processes," Fourth IEEE International Symposium on High Performance Distributed Computing, pp. 211, 1995.

[8] Adam J. Ferrari, "JPVM: Network Parallel Computing in Java," Technical Report:CS-97-29, 1997.

이 예 인

2007년 숙명여자대학교 정보과학부 멀티미디어과학전공(이학사)

2007년~현재 숙명여자대학교 멀티미디어과학과(석사)

관심분야 : Mobile System Software, Embedded System Software, Mobile Computing

이 종 우

1990년 서울대학교 컴퓨터공학과(학사)

1992년 서울대학교 컴퓨터공학과대학원(석사)

1996년 서울대학교 컴퓨터공학과대학원(박사)

1996년~1998년 현대전자㈜ 정보시스템사업본부 파장

1998년~1999년 현대정보기술㈜ 책임연구원

1999년~2002년 한림대학교 정보통신공학부 조교수

2002년~2003년 광운대학교 컴퓨터공학부 조교수

2003년~2004년 아이넉스소프트㈜ 개발이사

2004년~현재 숙명여자대학교 정보과학부 멀티미디어과학전공 부교수

2008년 뉴욕주립대 스토니브룩 Research Scholar

관심분야 : Mobile System Software, Storage Systems, Computational Finance, Cluster Computing, Parallel and Distributed Operating Systems, Embedded System Software