韓國 컴퓨터情報學會 論文誌
第12卷 第6號, 2007. 12.

2007-12-6-1-1

그리드 컴퓨팅 환경에서 확장 가능한 분산 스케줄링

이 준 동*, 이 무 훈**, 최 의 인***

Scalable Distributed Scheduling in Grid Computing Environment

Joon Dong Lee *, Moo Hun Lee **, Eui In Choi ***

요 약

본 논문에서는 분산된 자원 관리를 위한 새로운 프레임워크를 제안한다. 제안된 프레임워크는 새로운 특징들을 가진다. 첫째로, 자원 관리 시스템은 시스템 속성에 의해 특징지어지는 자원 컨텐츠 정보를 사용하여 분산되어진다. 자원 컨텐츠에 의거하는 분산 시스템은 그리드를 통해 워크로드 균형을 고려한다. 또한 Quality of Service(QoS)를 위한 특수한 스케줄링 요청을 만족시킨다. 두 번째로, 분산 시스템은 계층적 peer-to-peer 네트워크를 구성한다. 이러한 peered 네트워크는 효과적인 메시지 라우팅 메커니즘을 제공한다. 제안된 프레임워크가 시뮬레이션을 통하여 분산 환경에서 QoS를 만족한다는 것을 증명하고 있다.

Abstract

We propose a novel framework for distributed resource management. The framework has the following novel features. First, the resource management system is distributed using resource content information that is characterized by system properties. We argue that a distributed system based on resource content is sufficient to satisfy specific scheduling requests for Quality of Service(QoS) considering workload balance across a grid. Second, the distributed system constructs a hierarchical peer-to-peer network. This peered network provides an efficient message routing mechanism. The simulation results demonstrate that the proposed framework is proficient to satisfy QoS in distributed environment.

▸ Keyword : 그리드 컴퓨팅(Grid Computing), 스케줄링 시스템(Scheduling System), 분산 스케줄링(Distributed Scheduling), 서비스의 질(Quality of Service)

• 제1저자 : 이준동
• 접수일 : 2007.9.28, 심사일 : 2007.10.31, 심사완료일 : 2007.12.20.
* 강릉대학교 컴퓨터공학부 교수, ** 한남대학교 컴퓨터공학과 박사과정
*** 한남대학교 컴퓨터공학과 교수
※ 이 논문은 2007년도 한남대학교 학술연구 조성비 지원에 의하여 연구되었음.

Ⅰ. 서 론

그리드 컴퓨팅은 많은 데이터의 집약과 과학적 응용을 위해 높은 성능을 가진 컴퓨팅의 한 방법으로 자리잡고 있다. 이러한 컴퓨팅 패러다임은 경쟁적이고 협력적이고 그 각각의 조직을 고르게 분포된 데이터와 계산을 위한 계산집중적인 어플리케이션[1, 2, 3]을 위해 공동의 자원의 공유를 허가 한다. 공동의 자원은 문서, 소프트웨어, 컴퓨터, 데이터, 센서를 포함하여 컴퓨팅에 필요한 다양한 자원을 필요로 한다. 공동의 연구에서 다양하게 변화하고 별개의 정책과 다양한 조직에 의해 제어되는 다수의 이질적인 자원들은 이를 관리하기 위한 적절한 메커니즘을 필요로 한다.

단지, 지리학적 위치에 의해 분산 스케줄링 서비스를 접목시킨 기존의 스케줄링 기법들은 과학적 컴퓨팅의 특정한 스케줄링 요청을 원활히 할 수 없으며, 그러한 자원을 보호할 수도 없다. SDSS, LIGO, CMS, ATLAS[8, 9, 10, 11]와 같은 기술적 프로젝트들에서 다수의 실험들은 10 페타바이트(peta-byte)의 속도로 빠르고 거대한 양의 데이터를 생성되고 기록된다. 우리는 공동으로 연구하는 팀들의 세계 전역에 흩어져 있는 과학자들이 빠르고 정확하게 데이터를 접근, 분석하도록 하기 위해 자원 관리 시스템을 분산시키는데 있어 효율적이고 정밀한 방법들이 필요하게 된다. 효과적인 메시지 라우팅(message routing) 이슈의 획일적인 정의 또한 큰 규모의 그리드 환경에서 꼭 필요한 중요한 기술 요소이다.

확장 가능하고 효과적인 자원 관리 및 요청 스케줄링은 그리드 컴퓨팅에서의 가장 기초적인 기능이다. 메시지 라우팅 이슈[4, 5, 6, 7]는 Petascale Virtual Data Grid(PVDG)[1]에서 특징을 충분히 지원해야할 고려사항이다. 전형적인 그리드는 지리학적으로 분리되어진 조직들이나 개개인들로부터 이질적인 자원들의 공정한 결과로 이루어져 있다. 그 때문에, 환경으로의 확장 가능을 위한 분산 자원 관리 서비스를 필요로 한다. 우리는 분산된 서비스를 위해 다음과 같은 새로운 특징들을 지닌 프레임워크(framework)를 개발한다. 첫째로, 자원 관리 시스템은 시스템 속성에 의해 간주되는 자원 컨텐츠 정보를 이용하여 계층적 peer-to-peer 네트워크에서 분산되어진다. 두 번째로, peered 네트워크는 과다한 메시지와 성능 병목현상을 예방하기 위해 효과적인 메시지 라우팅 메커니즘을 QoS 질의를 표현하는 프로세스에게 제공한다.

본 논문의 구성은 다음과 같다. 2장에서는 현재 그리드 컴퓨팅과 관련하여 스케줄링 서버 분산과 메시지 전달에 대한 관련 연구에 대해 설명한다. 3장은 효율적인 분산 스케줄링과 메시지 라우팅에 대해 설명하고, 4장에서는 컨텐츠 기반 그리드 자원 클러스터링 방법에 대해 설명한다. 5장에서는 Peer to peer 스케줄링 서비스에 대해 설명하고, 6장에서는 제안한 스케줄링 기법의 실험 평가에 대하여 설명한다. 끝으로 7장에서 결론과 향후 연구에 대해 설명한다.

Ⅱ. 관련 연구

스케줄링 서버 분산과 메시지 전달은 그리드 컴퓨팅의 과학적 어플리케이션을 위해 특히 고려해야할 문제들이다. 지리학적 위치에 의거하여 서비스를 분산하고 다음 또는 랜덤 서버를 간단하게 요청들이 경과하는 naïve 방법은 다양한 컴퓨팅 환경에서 시간 결정적이거나 시스템 특유의 어플리케이션을 위해 동작할 수는 없다.

2.1. Hash based peer-to-peer network

peer-to-peer 분산 시스템[12]의 독립 노드는 인접한 peer-to-peer 분산 시스템의 작은 집합에 연결한다. 질의들은 이에 부합하는 데이터를 찾기 위해 네트워크 내의 모든 연결을 따른 다른 노드들을 재귀적으로 순환한다. Peer-to-peer 시스템은 키와 관련된 value값을 위해 하나 또는 그 이상의 네트워크 노드들의 응답으로 주어진 키들에 부합되는 검색 메커니즘을 명시적으로 또는 함축적으로 제공한다. 키 검색을 위한 단순한 질의는 이를테면 CAN, Chord, Tapestry[16, 17, 18]와 같은 거의 모든 peer-to-peer 시스템에서 사용된다. 앞에서 말한 시스템들은 매우 적은 수의 메시지나 노드 호핑(hopping)으로 데이터를 찾을 수 있음을 보증한다. 하지만, 이러한 테크닉들은 명확한 네트워크 구조와 완전한 데이터의 위치 제어를 요구한다. 그 시스템들은 분산된 해시(hash) 테이블 검사에 초점을 둔 고도의 집중된 컨텐트 어드레스블(content-addressable) 네트워크이기 때문에 매치메이킹(matchmaking) [13, 14]과 같은 일반적인 목적을 지닌 질의들을 지원을 할 수 없다.

2.2. Content based network

커뮤니케이션 프레임워크와 같은 content-based 네트워크는 큰 규모와 약결합된(loosely coupled) 분산 어플리케이션을 위한 커뮤니케이션 메커니즘을 지원한다. content-based 네트워크는 메시지 발송자가 명시한 주소들 보다 메

시지의 컨텐츠에 의해 실행된 네트워크를 통한 메시지 흐름을 위해 제공된 하위구조에 의해 기존의 유니캐스트(unicast)와 멀티캐스트(multicast) addressed-based 네트워크를 보충한다. content-based 네트워크에서 메시지 수신자가 네트워크로 간단하게 메시지를 보내는 동안, 'predicates'[19]에 의해서 네트워크를 통한 메시지 타입들에 관계하고 있는 메시지들을 명확하게 한다. 기존의 address-based 네트워크에서 전송 함수는 네트워크에서 노드들 사이에서 통과하는 메시지들에 의해 증가되어 실행된다.

메시지 라우팅에서 content-based 네트워크는 메시지 초과를 회피하고 최소한으로 진행된 메시지의 경로를 제공하는 주요 이슈를 가지고 있다[20, 21, 22]. 명시된 주소들을 포함한 메시지들이 목적지로 전송되는 address-based 네트워크와 달리, 라우터들의 경로 테이블 내의 속성에 대해서 메시지들의 컨텐츠를 검토하고, 메시지를 수신하는 근방에 위치한 라우터를 결정한다. 라우팅 의사결정(decision-making)과 경로 테이블 크기 유지는 다량의 복잡한 속성들과 대용량의 메시지를 포함하는 주요 병목현상이다.

III. 효율적인 분산 스케줄링과 메시지 라우팅

분산 자원 관리와 스케줄링이 이질적인 그리드 환경의 관리 내에서 다양한 QoS를 지닌 과학적 어플리케이션들을 위해 컴퓨팅 성능을 향상시켰다. 스케줄링을 위한 노력은 서비스 분산과 메시지 라우팅을 위해 독특한 하위구조를 갖는다. 제안된 프레임워크는 요청에 따른 신속한 접근을 고려하고 있으며 이는 몇몇 분산된 컴퓨팅의 특징을 이용한다.

계층적으로 분산된 조작, 저장 또는 네트워크 자원의 가상조직(VO; Virtual Organization)으로의 기여는 CPU, 메모리(memory), 저장소(storage), 대역폭(bandwidth)과 같은 시스템 속성에 의해 특징지어진다. 자원은 선택되어진 속성을 위해 value값의 집합과 같이 묘사되어지고, value값의 집합은 d 차원의 속성 공간에서 노드에 해당한다. 공간 내의 좌표는 공간내의 좌표들에 의거하여 밀집되어질 수 있다. 클러스터링(clustering) 절차는 유사한 속성 value값들이 함께 그룹화 되어지는 자원들을 의미한다. 각각의 자원 클러스터로 할당된 하나 또는 그 이상의 자원 관리 시스템에 의해, 우리는 자원 컨텐츠 정보에 의거한 충분하게 분산된 시스템을 이루었다. 할당된 스케줄러들은 클러스터 내에서 원격 또는 로컬 스케줄링 요청을 배정하는 자원을 독립적으로 실행한다.

자원 클러스터를 통한 분산된 시스템은 peered 네트워크를 구성한다. 네트워크는 그룹과 메시지 라우팅의 지역 내에서 검사 기능성을 지닌 독립적인 자원 관리를 보장한다. 특히, 프레임워크는 자원 속성에 특정한 요청을 전달하는 효과적인 메커니즘을 제공한다. 네트워크내의 스케줄링 서버는 다른 자원 클러스터 내에서 서비스의 정보를 포함하는 요청 라우팅 테이블을 유지한다. 요청의 목적지를 결정하기 위하여 서버는 테이블에서 클러스터를 표현하는 요청과 속성 정보 사이의 논리적 사상을 실행한다. 프레임워크는 content-based clustering network에서 표현하는 QoS 요청의 효과적인 전달을 돕는다.

자원 관리 시스템의 다중 객체는 비동기하게 공유된 자원들에 접근한다. 이것은 사용자의 자원 사용이 분산된 시스템 사이에서 조화되지 않는다고 간주하는 것과 같이 자원 상태와 다른 정보의 원인이 될 수 있다. 게다가, 모순을 야기하는 자원 제한이 단일 자원과 충돌이 일어날 수 있다. 이 프레임워크는 로킹(locking) 메커니즘을 사용하기여 공유된 자원의 동시접근을 제어한다. 의도되는 자원의 요청 서브미션은 두 단계로 이루어진다. 첫째로, 요청은 자원의 로크(lock)를 획득하기 위해 경합한다. 둘째로, 한 번의 로크로 실제의 서브미션은 자원을 독립적으로 이룬다. 로킹은 다중 서버들 사이에서 중앙 집중적인 제어를 하지 않고 정보의 일관성을 유지하는 독립적인 스케줄링을 제공한다.

IV. 컨텐츠 기반 그리드 자원 클러스터링

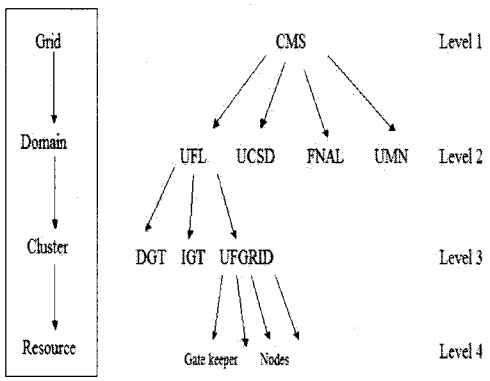
이 장에서 우리는 그것들의 속성을 따른 VO에서의 클러스터 자원 할당 기법을 설명한다. 그리드 자원들은 계층적인 하위구조에서 조직되어진다. 분산된 스케줄링 네트워크는 자원 그룹핑(grouping)에 의거하여 구성되어진다.

4.1. 그리드 자원의 계층적 구조

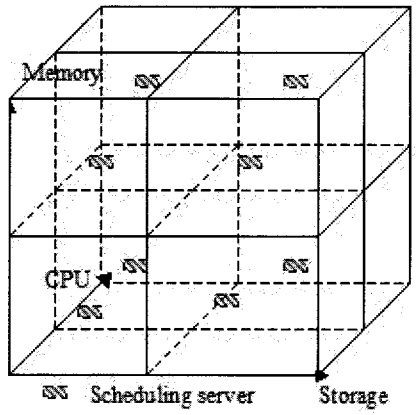
그리드 VO는 연산력(computing), 저장(storage), 네트워크 자원(network resources)의 계층적 구조를 유지한다. 그림 1-(a)는그리드 자원 구조의 예를 보여준다. 자원 제공자는 계층적 트리에서 전역으로 스케줄링되는 자원 레벨을 정의한다. 레벨 이하의 자원에서, 참가자의 로컬 스케줄러는 스케줄링 결정을 한다. 계층적 구조는 트리의 leaf

레벨에서 전역으로 스케줄링된 자원들을 제출할 수 있다. 구조는 전역으로 스케줄링된 자원들이 트리에서 다른 레벨로 표시되기 때문에 동일하지 않은 leaf 레벨을 갖는다.

중간물이나 구조의 leaf 레벨의 자원은 시스템 속성들의 집합으로 특징지어진다. 속성들의 그룹은 CPU, memory, bandwidth, benchmarking 정보와 같은 물리적인 정보를 포함할 수 있다. 중간 자원이 계층의 낮은 레벨에서 자원들의 적당한 정보를 모으는 동안, 자원은 자기 자신의 고유한 값들을 속성들과 동일한 것으로 간주한다.



(a) 계층적 자원 구조의 예
(a) A hierarchical resource structure example



(b) 자원 클러스터링 예
(b) A resource clustering example

그림 1. 그리드 자원 계층과 특정 공간에서 자원 클러스터링
Fig 1. grid resource hierarchy and resource clustering in property space

4.2. 자원 클러스터링

VO 내의 이질적인 자원들은 타입들 혹은 속성들의 용량에 의하여 분류되어진다. 분류법은 동적으로 변화하는 그리드 환경 내에서 이질적인 자원들의 특징을 이루는 방법이다.

선택된 시스템 속성들의 집합은 자원이 그것들의 value 값에 따라서 속성들로 표현 되어질 수 있는 d-차원의 공간에서 구조되어진다. 자원 속성 공간에서 우리는 단순 속성을 지닌 자원들을 모두 그룹화 하는 클러스터링 알고리즘을 적용한다. 그림 1-(b)는 3차원 공간에서 클러스터링의 예를 보여주고, 이는 CPU, memory, storage로 구성되어진다. 몇몇의 알고리즘이 그래프 분할과 클러스터링[25, 26]을 위해 이용되어지는 동안, 우리는 속성 공간에서 자원들의 접근을 따른 그룹을 만든다.

V. Peer to peer 스케줄링 서비스

이 장에서 우리는 peer-to-peer 네트워크에서 분산된 스케줄링 서비스를 기술한다. 우리는 VO를 통한 스케줄링 서비스 분산과 스케줄러들 사이에서 메시지 라우팅 메커니즘에 대해 설명한다. 우리는 또한 분산된 스케줄링 서비스 하위구조의 변화에 따른 자원 클러스터 유지와 라우팅 테이블 업데이트를 기술한다.

5.1. 스케줄링 서비스 배포

다중의 자원 그룹들로 할당되어진 스케줄링 서비스 모듈들은 분산된 스케줄링 서비스 네트워크로 구성되어진다. 그림 1의 그래프 (b)는 스케줄링 서비스 분산의 예를 보여준다. $S_c = \{s_i \mid \text{scheduler}, 1 \ i \ l\}$ (l은 스케줄러들의 수) 로 정의되어진 자원 클러스터를 위한 스케줄러의 집합은 클러스터 내부에 자원 상태와 사용 정책들에 의거한 독립적인 스케줄링 결정을 만든다. 스케줄링 프레임워크는 VO 내의 자원 클러스터들을 모두 분산한 스케줄링을 지닌 peered 네트워크를 구성한다. 일반적으로, 이는 VO 내의 (2^d x l) 스케줄링 서비스 모듈, d 는 속성들의 수, l은 클러스터의 스케줄러들의 수, 이다. 그것은 스케줄 가능한 자원들의 네트워크인 그리드에 기반 한 오버레이(overlay) 네트워크이다. 각 서버는 클러스터의 전형적인 자원 속성 정보와 클러스터에 부합하는 스케줄링 서비스를 포함하는 스케줄링 서비스 라우팅 테이블을 갖는다.

자원 클러스터에서 스케줄링 서버의 수는 단일 서버에 의해 야기되는 스케줄링 실패와 혼잡을 회피하는 것 이상이 가능하다. 클러스터내의 서버들이 지역 내에 할당된 상태와 정책 정보에 의한 자원들의 동일한 집합으로 스케줄링 결정을 하기 때문에 어떤 서비스 모듈들은 스케줄링 원격 요청

을 위한 영역을 표현할 수 있다. 그것은 각 스케줄러의 라우팅 테이블 사이즈를 클러스터의 수와 같은 2^d로 제한한다. 전형적인 스케줄러 모듈은 클러스터에서 가능한 서버들 사이에서 랜덤하게 선택되어진다.

5.2. 분산 스케줄링 서비스에서 메시지 라우팅

한 사용자가 스케줄러에 요청을 제출할 때, 스케줄러는 그것의 스케줄링 방법에 따라 자원 할당 절차를 실행한다. 스케줄러가 요청에 의해 제출된 QoS 요구사항을 만날 수 있다면 스케줄링 작업이 완료되었고, 일 서브미션 절차가 시작될 것이다. 하지만, QoS가 현 지역에서 자원들을 만족할 수 없다면, 스케줄러는 다른 자원 그룹의 스케줄러에게 요청을 전달하는 것을 필요로 할 것이다. 프레임워크는 서비스 peered 네트워크에서 요청 라우팅 메커니즘을 고안한다.

스케줄링 서버가 자원 할당 결정을 요청에 내릴 수 없는 세 가지 경우를 정의한다. 첫째, 스케줄러는 클러스터 안에서 자원들은 사용한 요청의 QoS 요구사항을 만족시킬 수 없다. 둘째, 클러스터 내의 모든 자원들은 작업 실행 시 과부하가 일어난다. 셋째, 스케줄러들 자신은 요청 스케줄링 작업에서 과부하가 일어난다. 자원들 혹은 스케줄러들에 요청이 과부하가 일어나는 경우는 근접한 클러스터내의 스케줄러들에게 단순하게 전달되어진다. 하지만, 전달 결정은 요청이 자원 명세 QoS를 요구하는 첫 번째 경우와 같이 평이하지는 않다. 스케줄러들은 안정된 요구가 가능한 자원 그룹으로 요청을 전달하는 스마트한 결정을 내릴 수 있다. 결정은 자원 클러스터들 사이에서 넘치는 요청을 회피할 수 있어야 하고, 그리드 컴퓨팅 성능에서나 어느 쪽에서든 병목현상이 없어야 한다.

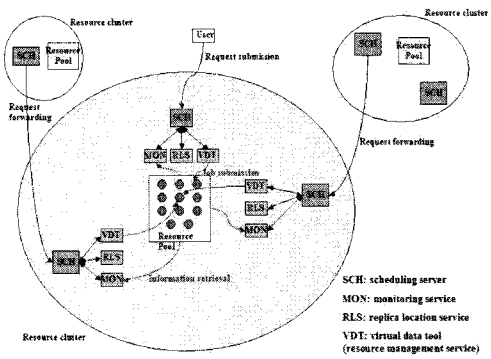
사용자는 서버가 전달 함수의 결과에 따른 요청을 다른 자원 그룹 라우팅 테이블의 다른 서버로 전달하는 동안, 스케줄링 요청을 지리학적 구역 내의 인접한 서버로 제출한다. 스케줄링 서버는 요청 라우팅 테이블을 유지한다. 테이블 내의 각 엔트리는 클러스터와 클러스터 내의 스케줄러의 주소에서 자원의 속정 정보를 모은다. 자원들은 그것들의 속성 값을 따라 클러스터링 되어 지기 때문에 OS 혹은 아키텍처와 같은 불연속적 속성을 위한 단일 값이 되고, memory 혹은 CPU speed와 같은 지속적인 속성을 위한 범위 값이 된다.

스케줄러가 명세 QoS 요구사항을 지닌 요청을 전달하는 목적지의 발견을 요구할 때 스케줄러는 QoS와 테이블의 속성 정보 사이에서 논리적 매핑을 실행한다. QoS 요구사항을 만족할 수 있는 자원 클러스터를 찾은 후에 요청은 라우팅 테이블에 명시된 스케줄러로 전달되어 진다. 스케줄러 서버는 QoS 요구 레벨을 결정하는 사용자와 상호작용을 할 수 있다. [6]은 QoS를 만족하는 사용자 상호작용을 기술하고 있다.

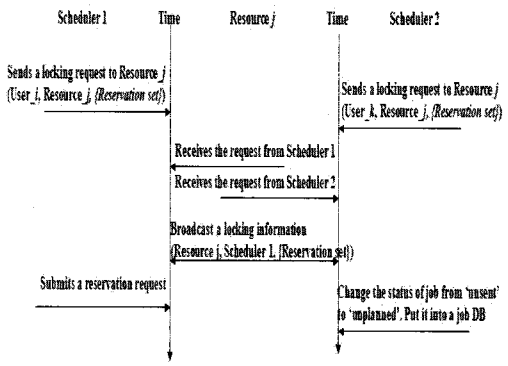
5.3. 자원 클러스터와 라우팅 테이블

peer-to-peer 스케줄링 네트워크를 유지할 목적으로, 프레임워크는 자원 특성에 이용되는 다수의 속성에 따라 자원 클러스터를 유지하기 위한 연산을 정의한다. VO에서 자원이 d 속성으로 식별되어질 때, 다수의 자원 클러스터는 2^d이다. 이전 절의 클러스터링 알고리즘에 이어 자원이 2^{d+m} 자원 클러스터로 재그룹되어 진다. 여기에서 우리는 다수의 자원이 다수의 속성보다 더 크다고 가정한다. 하나 또는 그 이상의 스케줄링 서비스 모듈은 각 자원 그룹으로 할당되어진다. 하나의 속성이 리스트로부터 제거되어질 때, 클러스터링은 속성을 추가하는 것과 더불어 동일한 순서로 재구성되어진다.

변환된 속성 집합과 더불어 새로운 자원 클러스터를 구성한 후에, 라우팅 테이블은 각 클러스터에 할당되어진다. 클러스터는 다중 스케줄러를 가짐에도 불구하고, 테이블은 클러스터로 표현되어지는 단일 스케줄러를 가진다. 스케줄러는 클러스터 안의 다중 스케줄링 객체들 사이에서 임의로 선택되어진다. 어떤 스케줄러는 그것의 자원 클러스터 내에서 공유된 자원 상태 정보를 기반으로 동일한 스케줄링 결정을 내린다. 스케줄러는 자원 그룹 내에서 자원 상태와 이용정책에 기반을 두고 스케줄링 결정을 내린다. 그림 2는 분산 서비스의 구조와 일관된 공유 자원 접근의 예를 보여주고 있다.



(a) 분산 서비스 구조
(a) Distributed service architecture



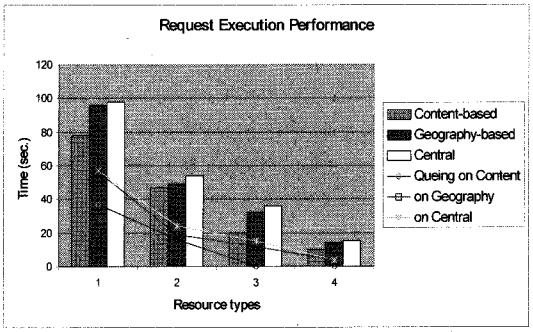
(b) 일관된 공유 자원 접근의 예
(b) An example of consistent shared access to a resource
그림 2. 일관된 분산 스케줄링 서비스
Fig 2. Consistent distributed scheduling service

VI. 실험 평가

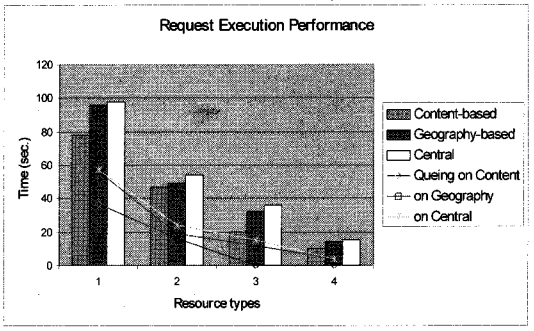
실험 평가는 content-based, geography-based, centralizes scheduling networks의 세 가지의 다른 네트워크 토폴로지(topology)에서 요청 스케줄링의 성능을 보여준다. geography-based가 지리학적 구역에 기반 한 기존의 분산된 네트워크이긴 하지만 앞 장에서 다룬 논의들에 따라 content-based network를 구성한다. 서비스에 집중된 단일 서버는 그리드 내의 모든 자원들에 대한 스케줄링 요청들을 관리한다.

실험에서는 자원 클러스터링에 따른 content-based network를 구성한다. 스케줄링 서버는 각 그룹들로 이루어진다. 서버는 다른 스케줄러들과 클러스터에 부합하는 속성 정보의 IP 주소를 포함한 메시지 라우팅 테이블을 유지한다. geography-based network에서 서버는 네트워크 지역 내에서 다양한 속성을 지닌 자원들에게 스케줄링 결정을 내린다. 서버가 불완전한 요청을 지니고 있을 때 서버는 단순하게 그것들을 이웃한 클러스터로 전달한다. 집중되어진 스케줄링 서비스 아키텍처에서, 단일 서버는 모든 자원들의 속성 정보를 가진다. 사용자들은 스케줄링 결정에서 자원들을 사용하는 스케줄러에게 요청을 제출한다.

이 실험에서 세 가지 네트워크 토폴로지에서 두 가지 측면으로 스케줄링 성능을 평가한다. 첫 번째로, 각각 자원의 작업 완료 시간을 측정한다. 효율적인 요청 스케줄링은 자원에 대한 작업량을 줄여준다. 그리고 자원에 대한 큐잉(queuing) 시간에 의해 표현되어진다. 보다 나은 실행 성능은 정확한 예약에 있어서 QoS 요구사항을 만족하기 위한 가능성을 증가시킨다. 두 번째로 토폴로지 상에서 서로 다른 다수의 요청사이의 QoS miss rate을 조사한다.



(a) 작업 실행 성능(300 요청)
(a) Task execution performance (300 requests)



(b) QoS 요청 간과율
(b) QoS request missing rates
그림 3. 스케줄링 네트워크 토폴로지 사이에서 성능 비교
Fig 3. Performance comparison among scheduling network topologies

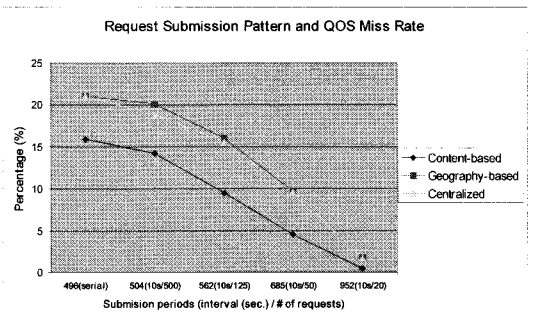
그림 3의 그래프 a는 네트워크 토폴로지가 요청 완료 시간에 어떤 영향을 미치는 지를 보여준다. 우리는 각 속성에 따라 자원을 4가지 종류로 분류하였다. 그래프는 각각의 자원 분류에서 평균 작업 완료와 큐잉 또는 대기 시간을 보여준다. 우리가 스케줄링 네트워크를 구성할 때, 큐잉 시간이 동일한 자원 성능에 있어서 차이점을 유발하는 것을 알 수 있다. 그래프 a에서 content-based 상의 자원은 다른 것에 비하여 더 적은 큐잉 시간을 갖는다. 이러한 사실은 네트워크상의 자원이 다른 것에 비하여 더 적은 작업 부하를 가지는 것을 알 수 있다. content-based 스케줄링 네트워크는 모든 VO 상의 자원에서 QoS 만족을 용이하게 한다. 반면에 스케줄러는 그 범위 안의 자원이나 geography-based에서 이웃하는 자원에 대해 스케줄링 결정을 내린다. 제한된

사용은 오랜 작업 완료 시간을 야기하게 하고, 자원이 쉽게 과부하 걸리게 한다.

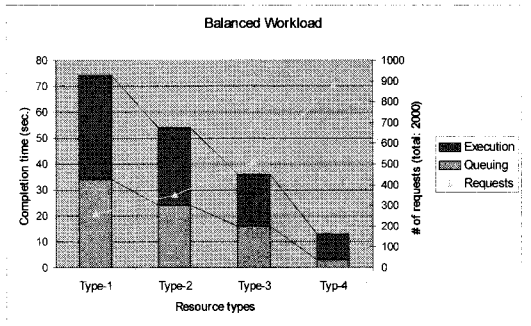
그림 3의 그래프 b는 스케줄링 서비스가 요청의 모든 합계에서 QoS 요구사항을 만족하기에 어려움이 있다는 것을 보여준다. 그래프는 각 스케줄링 네트워크에서 자원 할당 결정 시간에 실패된 QoS 만족의 비율을 나타낸다. 다수의 요청으로 인해 자원이 과부하가 걸리게 됨으로써, 스케줄러는 QoS 요청에 대처하지 못하게 된다. 그래프는 content-based 네트워크에서 스케줄러가 다른 네트워크 보다 낮은 miss 율을 갖는 것을 보여준다. 그래프 a에서 보여준 바와 같이, 네트워크에서 자원은 property-based 자원 클러스터링 때문에 다른 것보다 적은 과부하를 가지게 된다.

다양한 요청 서브미션 패턴은 자원 워크로드(workload)에서 다르게 형성될 수 있다. 그림 3에서 볼 수 있듯이, QoS 요구사항의 큰 비율이 스케줄링 네트워크에서 정체된 그리드 자원들을 만족시킬 수 없다. 이 실험에서 우리는 단지 on-time 자원 제한만을 실행한다. QoS 만족은 자원들의 통용하고 있는 필요조건에 의해서만 확정되어진다. 요청들의 연속이 짧은 시간 내에 제출되어질 때, 제한된 자원들의 수는 과부하에 쉽게 이른다. 하지만, 작업들의 동일한 수가 좀 더 긴 시간동안 제출되어질 때, QoS 만족의 비율은 짧은 시간에 제출되어지는 것보다 높아진다. 그림 4의 그래프 (a)는 QoS miss rate는 서브미션 시간의 증가에 따라 감소한다.

on-time 알고리즘에 의거한 자원 할당은 총체적인 자원 사용과 QoS 만족 내에서 제한을 갖는다. 진보된 자원 제한 개념은 요청 스케줄링을 쉽게 한다. 스케줄러는 가까운 미래 시간 프레임에서 예상된 자원 워크로드를 이용하여 제한을 생성할 수 있다. 몇몇 워크로드를 예측하는 테크닉도 있긴 하지만, 가장 단순한 것은 자원 큐(queue) 또는 스택(stack)을 조사하는 것이다. 과부하 된 자원에 요청을 제출하는 것 대신에, 스케줄러는 자원 사용을 확보하고, 서브미션을 지연시킨다. 자원들 가운데 균형이 잡힌 워크로드는 분산된 스케줄링 시스템에서 제한을 쉽게 한다. 그룹 내의 자원들은 그들의 용량에 적당한 다른 워크로드를 갖는 반면에, 워크로드 균형(workload balance)은 할당된 작업들의 수를 제어함으로서 이루어진다. 그림 4의 그래프 (b)는 자원의 다른 종류에서 조합이 잘된 워크로드의 예를 보여준다. 우리는 2000개의 요청들을 4개의 다른 자원들의 종류로 제출한다. 평균 요청 완결 시간이 자원 그룹들 사이에서 다른데도, 완결되는 큐잉 시간의 비율은 Type-1, Type-2, Type-3, 세 가지 타입 사이에서 유사하게 45%이다. 제출된 요청의 수가 전체적으로 모든 타입에 빈번한 자원들을 만들기에 충분할 정도로 크지 않기 때문에, Type-4 그룹에서 자원들은 다른 그룹에서보다 적은 큐잉 시간을 갖는다. 노란 선은 각 자원 그룹에서 할당된 요청들의 수를 말한다.



(a) QoS 만족률과 요청 제출
(a) QoS satisfaction rate and request submission



(b) 워크로드 균형과 요청 할당
(b) Workload balance and request assignment
그림 4. 자원 워크로드와 QoS 만족
Fig 4. Resource workload and QoS satisfaction

VII. 결론

본 논문에서 분산 스케줄링 서비스 프레임워크를 설명했다. 네트워크 자원들은 지리학적 위치들을 대신한 그들의 특징들을 기반 하여 클러스터링 되어지고, 서비스는 content-based 클러스터링을 따라 분산되어지기 때문에 효율적인 접근이 고려되어야 한다. 분산된 서비스들은 peer-to-peer 네트워크에서 연결되어진다. peered 네트워크에서 메시지 전달은 스케줄링 요청의 QoS 요구사항을 만족하기 위하여 라우팅 함수에 기반을 둔 해시를 사용하여 실행한다.

향후 연구는 작업부하가 현재와 동일하다고 가정한 상태에서 일반적이고 실제적인 어플리케이션들을 반영하기 위한 작업적재의 다른 형태들을 처리하는 스케줄링 방법에 대하여 연구하는 것이다. 또한 다양한 서비스의 질 요구와 스케줄 작업들의 방법을 발전시켜야 할 것이다.

참고문헌

[1] 박다혜, 이종식, "계산 그리드 컴퓨팅에서의 자원 성능 측정을 통한 그리드 스케줄링 모델," 한국컴퓨터정보학회 논문지 제11권 제1호, 2006.

[2] 조수현, 김영학, "계산 그리드 상에서 프로그램의 특성을 반영한 작업 프로세스 수의 결정에 관한 연구," 한국컴퓨터정보학회 논문지 제11권 제1호, 2006.

[3] 황선태, 심규호, "계산 그리드에서 워크플로우 기반의 사용자 환경 설계 및 구현," 한국컴퓨터정보학회 논문지 제10권 제4호, 2005.

[4] R. Min, M. Maheswaran, Scheduling Co-Reservations with Priorities in Grid Computing Systems, Proceedings of the 2ndIEEE/ACM International Symposium on Cluster Computing and the Grid, 2002

[5] D. P. Spooner, J. Cao, J. D. Turner, H. N. Lim Choi Keung, S. A. Jarvis, G. R. Nudd, Localised Workload Management Using Performance Prediction and QoS Contracts, Eighteenth Annual UK Performance Engineering Workshop, 2002

[6] J. In, A. Arbree, P. Avery,R. Cavanaugh, S. Katageri, S. Ranka, Sphinx: A Scheduling Middleware for Data Intensive Applications on a Grid, GriphyN Project Technical Report, GriPhyN 2003-17, 2003.

[7] J. In, P. Avery, R. Cavanaugh, S. Ranka, Policy Based Scheduling for Simple Quality of Service in Grid Computing, 18th International Parallel & Distributed Processing Symposium, New Mexico, USA, 2004

[8] Sloan Digital Sky Survey, http://www.sdss.org, 2004

[9] Laser Interferometer Gravitational Wave Observatory, http://ligo.caltech.edu, 2004

[10] The Compact Muon Solenoid, an experiment for the Large Hadron Collider at CERN, http://cmsinfo.cern.ch/Welcome.html/, 2004

[11] The ATLAS Experiment, http://atlasexperiment. org, 2004

[12] D. Doval, D. O'Mahony, Overlay Networks, A Scalable Alternative for P2P, IEEE Internet Computing, August 2003.

[13] W. Hoschek, A Unified Peer-to-Peer Database Framework for Scalable Service and Resource Discovery, Proc. of the International IEEE/ACM Workshop on Grid Computing, Baltimore, USA, Nov. 2002. Springer Verlag.

[14] Dan Bradley, Condor-G Matchmaking in USCMS, Condor technical report, University of Wisconsin, Nov. 2003

[15] A. Crespo, H. Garcia-Molina, Semantic Overlay Networks for P2P Systems, Technical report, Stanford University, Jan. 2003.

[16] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, A scalable content-addressable network, ACM SIGCOMM, 2001.

[17] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishman, Chord: A scalable peer-to-peer loopup service for internet applications, ACM SIGCOMM, 2001.

[18] B. Zhao, J. Kubiatowicz, A. Joseph, Tapestry: An infrastructure for fault-tolerant wide-area location and routing, Technical report, U. C. Berkeley, 2001

[19] A. Carzaniga, A.L. Wolf, Content-based Networking: A New Communication infrastructure, NSF Workshop on an infrastructure for Mobile and Wireless Systems, Scottsdale, AZ, October, 2001

[20] A. Carzaniga, M.J. Rutherford, A.L. Wolf, A Routing Scheme for Content-Based Networking, Proceedings of IEEE INFOCOMM 2004, Hong Kong China, March, 2004.

[21] R. Chand, P. Felber, A Scalable Protocol for

Content-Based Routing in Overlay Networks, Proceedings of the IEEE International Symposium on Network Computing and Applications, Cambridge, MA, April, 2003.

[22] M. Aron, D. Sanders, P. Druschel, W. Zwaenepoel, Scalable Content-aware Request Distribution in Cluster-based Network Servers, Proceedings of the 2000 Annual Usenix Technical Conference, San Diego, CA, June, 2000.

[23] T. Sandholm, J. Gawor, Globus Toolkit 3 Core A Grid Service Container Framework, Globus Toolkit 3 Core white paper, Globus technical reference, July 2003

[24] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, Grid Information Services for Distributed Resource Sharing, Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing, IEEE Press, August 2001.

[25] Chao-Wei Ou, Sanjay Ranka, Parallel Incremental Graph Partitioning, IEEE Transaction on Parallel and Distributed Systems, 8, 8, 884-896, 1997.

[26] Vipin Kumar, Graph Partitioning for Multi-phase and Multi-physics Computations, IEEE International Conference on Cluster Computing, Newport Beach, California, October 2001.

저 자 소 개

이준동(Lee Joon Dong)

2001년 2월 : 홍익대학교 전자계산학과 이학박사

1997년 ~ 2006년 : 원주대학 교수

2007년 ~ 현재: 강릉대학교 교수

관심분야 : 시스템 소프트웨어, 유비쿼터스, 실시간처리, 데이터베이스, Grid Computing

이무훈(Lee Moo Hun)

2002년 : 한남대학교 컴퓨터공학과 공학사

2004년 : 한남대학교 컴퓨터공학과 공학석사

2004년 ~ 현재 : 한남대학교 컴퓨터공학과 박사과정

관심분야 : Web search engine, Semantic Web, Web Service, Distributed Scheduling System, Grid Computing

최의인(Choi Eui In)

1995년 : 홍익대학교 전자계산학과 이학박사

1992년 ~ 1996년 : 명지전문대학 전자계산과 조교수

1996년 ~ 현재 : 한남대학교 컴퓨터공학과 교수

2003년 : UCLA visiting scholar

관심분야 : Ubiquitous Computing, Web search engine, Semantic Web, Context Modeling, Grid Computing