클라우드 환경에서 가용 자원 활용도를 고려한 워크플로우 작업 클러스터링 기법†

명노영*, 정대용*, 정광식+, 유헌창*
*고려대학교 컴퓨터교육학과
+한국방송통신대학교 컴퓨터학과
e-mail : *{mry1811, karat, yuhc}@korea.ac.kr, +Kchung0825@knou.ac.kr

Workflow Task Clustering Method Considering Available Resources in Cloud Environments

Rohyoung Myung*, Daeyong Jung*, KwangSik Chung, Heonchang Yu*
*Distributed and Cloud Computing Lab., Korea University
+Dept. of Computer Science, Korea National Open University

요 약

워크플로우 매니지먼트시스템은 오늘날의 어플리케이션들의 처리를 위한 효율적인 워크플로우 설계와 수행을 가능하게 한다. 그러나 천체물리학, 생물학, 지질학과 같이 과학탐구에 목적을 둔 어플리케이션들의 경우 대용량의 데이터를 연산해야 하기 때문에 단일 컴퓨팅 자원으로는 단 시간내에 작업을 완료하기 어렵다. 클라우드 환경에서 워크플로우를 효율적으로 수행하기 위해서는 여러 자원을 효율적으로 활용하기 위한 분산 병렬처리가 필수적이다. 일반적으로 시스템의 마스터노드에서는 클러스터의 원격노드들에게 어플리케이션 수행을 위해 설계된 워크플로우에 맞게 작업들을 분배하게 되는데 이때 마스터노드와 원격노드의 큐에서의 대기시간과 원격노드에서 할당된 작업들을 위한 스케줄링 시간은 성능을 좋지 않게 만드는 원인이 된다. 따라서 본 논문은 클라우드 환경에서 원격노드에서 작업수행이전까지의 지연시간을 줄이기 위한 최적화 방법으로 컴퓨팅 자원 활용도를 고려한 작업들의 병합 기법을 적용해서 워크플로우의 처리 속도를 향상시킨다.

1. 서론

워크플로우 매니지먼트 시스템[1]은 어플리케이션의 작업을 미세한 작업들과 작업들에 필요한 입출력 파일들을 가시적인 형태의 DAG[2]로 표현한다. 또한, 어플리케이션의 수행환경을 고려한 작업과 자원의 조합을 통해 효율적인 어플리케이션 처리를 가능하게 한다.

오늘날의 과학 탐구적 성격을 갖는 어플리케이션들[3,4,5,6,7]의 경우 수많은 작업들과 이에 필요한 다수의 대용량 입출력 파일들로 구성되기 때문에 다수의 컴퓨팅 자원을 활용한 병렬처리를 통해 단시간내 처리가 가능하다. 이를 위해서는 클라우드 환경에서 어플리케이션들을 수행할 수 있는 워크플로우의 설계가 필요하다. 일반적으로 클라우드 환경에서 워크플로우를 통해 어플리케이션을 수행할 경우 가용자원 확인, 가용자원 및 데이터 지역성을 고려한 원격지 노드 선택, 워크플로우 수행 성능 향상을 위한 작업 클러스터링, 그리고 실제 워크플로우 실행 등의 4 단계로 수행된다[8]. 두 번째 단계에서 마스터노드는 워크플로우를 구성하는 작업들의 수행시간과 작업들의 수행에 필요한 입출력 파일들의 지역성을 고려해서 실제 작업을 수행할 원격노드들을 선정한다. 또한 동일한 파일의 생성을 방지하기 위해 워크플로우의 입출력파일들을 분석해서 수행과정을 줄이는 최적화 작업을 수행한다. 세 번째 단계에서는 마스터노드와 작업노드에서 발생하는 시스템부하를 줄이기 위해 작업들을 클러스터링한다. 시스템부하의 감소는 원격노드에서 작업들의 실질적인 수행시작시간을 앞당기고 결과적으로 워크플로우 수행성능을 향상시킨다. 마지막으로 클러스터링 된 작업집합들을 원격지 노드에 보낼 때는 DAG 의 동일한 단계에 있는 작업집합들을 전송하고 한 단계가 끝나면 다음 단계의 클러스터링된 작업들을 원격지 노드로 전송하는 방식으로 실행한다. 그러나 작업들을 클러스터링 함으로써 시스템부하는 감소시켰지만 기존의 세 번째 단계에서 시스템을 구성하는 노드들의 컴퓨팅 성능을 고려한 각각의 작업 배치는 무시되게 된다.

따라서, 본 논문의 2 장에서는 워크플로우 매니지먼

† “이 논문은 2015 년도 정보(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2012RIA2A2A02046684).”

트의 실행흐름을 고려하고, 클러스터링 기법의 전처리에 필요한 비용을 줄이면서 시스템 환경의 컴퓨팅 성능을 고려한 클러스터링 기법을 소개한다. 3 장에서는 실제 워크플로우 매니지먼트 시스템에서 본 논문에서 제시한 클러스터링 기법을 지명도가 높은 과학 탐구 어플리케이션 Montage 에 적용해 봄으로써, 기법의 성능을 평가한다.

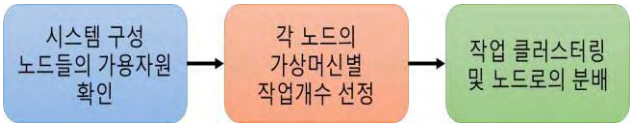
2. 워크플로우 작업 병합 기법

기존의 클러스터링 기법으로는 Horizontal Clustering (HC)[9], Horizontal Runtime Balancing algorithm(HRB)[8], Adaptive Fine-grained Job Scheduler(AFJS)[10] 등이 있다. 첫 번째 HC 는 워크플로우 매니지먼트 시스템 Pegasus 의 기본 작업 클러스터링 알고리즘으로 사용자로부터 몇 개의 작업들을 하나의 작업집합으로 클러스터링할지를 입력받는다. 이를 통해 마스터노드는 워크플로우의 각 단계별로 작업들을 사용자로부터 입력받은 작업의 개수만큼 작업집합에 클러스터링해서 작업집합들을 생성하고 모든 단계가 완료될 때까지 반복한다. 그러나 이는 시스템 환경뿐만 아니라 어플리케이션의 특성을 고려하지 못한 클러스터링을 하기 때문에 비효율적이다.

두 번째 HRB 는 기존의 HC 와는 다르게 같은 단계의 작업들을 클러스터링할 때 작업집합에 분산되는 작업들의 수행시간을 고려한 클러스터링을 수행한다. 이를 위해 마스터노드는 각 단계별로 각각의 작업집합들에 포함된 작업들의 수행시간을 합산해서 현재까지 수행시간이 가장 작은 작업집합에 작업을 클러스터링한다. 최종적으로 모든 단계가 완료될때까지 클러스터링을 반복한다. 또한 [11]을 통해 이전보다 효율적인 작업집합당 작업개수를 선정함으로써 기존의 HC 보다 성능을 향상시켰다. 그러나 워크플로우 매니지먼트 시스템에 구동되는 어플리케이션이 이미 충분히 분석되지 않았을 경우 수행시간을 위한 별도의 분석이 필요하다는 문제가 존재한다.

세 번째 AFJS 는 수행시간 뿐만아니라 작업집합에 포함된 입출력데이터의 크기를 고려한 클러스터링을 수행한다. 이를 위해 마스터노드는 각 단계별로 각각의 작업집합들에 포함된 작업들의 수행시간과 입출력 데이터크기를 합산해서 클러스터링될 작업이 작업집합의 최대수행시간을 넘지 않거나 수용 데이터 크기를 넘지 않으면 작업집합에 포함시킨다. 그러나 AFJS 는 시스템 환경의 컴퓨팅 능력과 데이터 크기를 모두 고려했지만 앞선 HRB 처럼 작업들의 수행시간을 알아야 하고, 또한 작업집합의 효율적인 최대수행시간을 구해야만 클러스터링이 가능하다는 단점을 갖는다. 최적의 최대수행시간을 도출하기 위해서는 어플리케이션과 시스템 환경을 모두 고려한 세밀한 분석이 필요하다.

워크플로우 매니지먼트 시스템에서 어플리케이션의 처리성능 향상을 위한 워크플로우의 단계별 작업 클러스터링은 필수불가결하다. 그러나 [8]에서처럼 시스템 환경을 고려해서 작업들이 진행될 노드를 정하고 클러스터링 기법을 적용할 경우 시스템의 성능과 가용자원을 반영한 작업처리 노드 선정이 훼손된다. 본 논문에서는 작업들을 클러스터링할 때 작업이 처리될 각 노드들의 성능과 가용자원에 적합하게 클러스터링하고 클러스터링된 작업 집합들을 선정된 노드에 분배함으로써 기존의 문제점을 해결한다. 이에 대한 순서도는 그림(1)과 같다.



(그림 1) 워크플로우 작업 병합기법 순서도

본 논문의 병합기법은 (그림 1)과 같이 첫 번째 단계에서 시스템을 구성하는 노드들의 가용자원을 확인한다. 가용자원 확인시 시스템 구성 노드들의 CPU 성능 및 노드들에 위치한 가상머신(VM)들의 VCPU 성능을 확인한다. 그리고 VCPU 별 현재 사용량을 확인한다. 두 번째 단계에서는 워크플로우의 작업들을 단계별로 노드들의 CPU 개수에 비례하게 그루핑한다. 노드별로 나눠진 그룹들은 다시 VM 들의 VCPU 의 개수와 현재 사용량을 고려해서 VM 별 작업개수를 선정한다. 마지막 단계에서는 선정된 작업개수에 맞게 클러스터링을 통해 작업집합들을 생성하고 VM 들에게 분배한다.

$$node_i.CA = \frac{node_i.CN * node_i.CU}{\sum_{k=1}^{n}(node_k.CN * node_k.CU)} \quad (1)$$

$$VM_i.VA = \frac{VM_i.VN * VM_i.VU}{\sum_{k=1}^{n}(VM_k.VN * VM_k.VU)} \quad (2)$$

$$node_i.TN = TN * node_i.CA \quad (3)$$

$$VM_i.TN = TN * node_i.CA * VM_i.VA \quad (4)$$

(그림 2) 노드와 가상머신별 할당 작업개수 계산식

<표 1> 계산식 (1), (2), (3), (4)의 변수목록

변수명	의미
$node_n.CA$	노드 n의 CPU가용률
$node_n.CN$	노드 n의 CPU개수
$node_n.CU$	노드 n의 CPU사용률
$VM_n.VA$	가상머신 n의 VCPU가용률
$VM_n.VN$	가상머신 n의 VCPU개수
$VM_n.VU$	가상머신 n의 VCPU사용률
TN	전체 작업개수
$node_n.TN$	노드 n에 할당된 작업개수
$VM_n.TN$	가상머신 n에 할당된 작업개수

(그림 2)에서는 각각의 노드에 할당될 작업개수와 노드에 위치한 가상머신 각각에 할당될 작업개수를 위한 계산식이다. (1), (2)번 계산식에서는 각각 특정 노

드의 CPU 가용률과 특정 VM 의 VCPU 가용률을 계산한다. 그리고 CPU 가용률을 사용해서 전체 작업개수 중에 해당 노드로 분배할 작업의 개수를 선정한다. 또한 VCPU 가용률을 사용해서 노드에 분배된 작업의 개수중에 해당 VM 으로 분배할 작업의 개수를 선정한다. 계산식에 사용된 변수들은 <표 1>과 같다.

3. 성능평가

성능평가를 위한 시뮬레이터의 실험환경은 <표 2>와 같다.

<표 2> WorkflowSim 수행환경

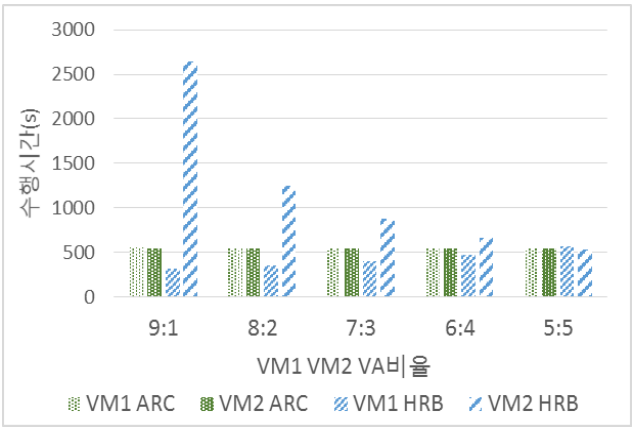
운영체제	Windows7(64bit)
CPU	Core I5(3.4GHz)
RAM	8GB
SDD	256GB

워크플로우의 작업 클러스터링 및 실행에는 WorkflowSim[12]을 사용했다. WorkflowSim 은 그리드 환경 및 클라우드 환경에서 사용자의 기호에 맞게 노드와 노드안에 생성되는 가상머신들의 성능 조작을 지원한다. 또한, 다양한 작업 병합 기법의 적용이 가능하고 대중적인 과학탐구 어플리케이션들의 성능측정 및 비교평가를 위해 제작된 시뮬레이터이다. 또한 클라우드 환경의 데이터센터, Host Machine(HM), VM 에서 워크플로우 수행을 통해 발생하는 지연들을 세밀하게 프로그래밍함으로써 시뮬레이터의 신뢰성을 향상시켰다. 본 논문에서 WorkflowSim 을 사용해서 정의한 수행환경은 <표 3>과 같다.

<표 3> 워크플로우 수행환경

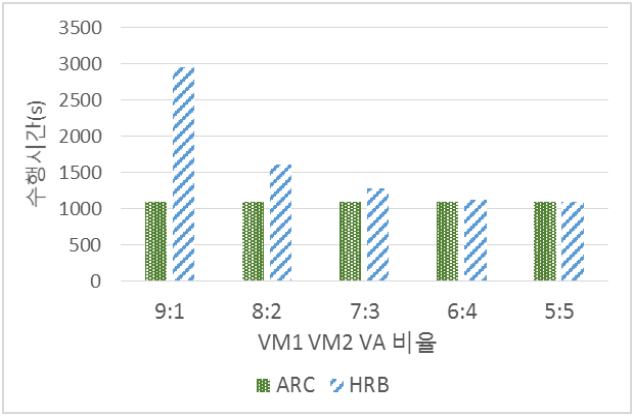
Host Machine	
운영체제	Linux(x86)
CPU	MIPS(2.0GHz*4)
RAM	4GB
HDD	256GB
Virtual Machine	
운영체제	Xen
CPU	MIPS(0.2~1.8GHz*2)
RAM	1GB
HDD	64GB

<표 2>, <표 3>을 기반으로 WorkflowSim 에서 대표적인 과학탐구 어플리케이션인 Montage[4]를 사용해서 HRB 와 본 논문에서 제시한 기법 using Available Resource Clustering(ARC)간 성능을 비교한다. HRB 를 사용한 이유는 WorkflowSim 수행 결과에서 작업들의 수행시간을 구하는 시간을 제외하면 HRB 가 가장 좋은 성능을 보이기 때문이다. 하나의 HM 은 두 개의 VM 을 가지며 VM 의 CPU 코어 개수, RAM 의 총합은 HM 과 같다. VM 들의 사용가능 자원합이 HM 의 자원 총합 이하일 경우 HM 의 가용률은 워크플로우의 수행시간에 큰 영향을 주지 못하기 때문에 실험에서는 ARC 와 HC 를 사용했을 때 VA 에 따른 VM 들의 성능을 측정하고 비교한다. VA 차이는 HM 에서 제공하는 자원내에서 VCPU 코어의 성능제어를 통해 발생시킨다.



(그림 3) VM 가용비율에 따른 VM 별 ARC 와 HC 의 수행시간

(그림 3)에서는 VM1 과 VM2 간 VA 의 차이를 발생시켰을때 ARC 와 HRB 의 수행시간을 나타낸다. ARC 의 경우 VM1 과 VM2 의 VA 차이가 심해져도 이를 반영한 차등 작업 분배를 통해 워크플로우 수행시간의 차이가 거의 발생하지 않는 사실을 확인할 수 있다. 반면 HRB 의 경우 VM 들의 VA 차이가 심해질수록 VM 의 성능차이가 심하게 발생한다는 사실을 확인할 수 있다. 또한 VA 가 높아졌다고해도 작업분배가 적합하게 이뤄지지 않았기 때문에 비효율적인 워크플로우 수행을 하게 되고 (그림 4)에서처럼 전체 워크플로우를 수행하는데 소요되는 시간도 큰 차이를 보이게 된다.



(그림 4) VM 가용비율에 따른 ARC 와 HC 의 전체 수행시간

동일한 VA 가 적용될 경우(5:5)는 ARC 와 HRB 의 성능이 유사하지만 VA 차이가 벌어지면 성능차이가 기하급수적으로 커지는 결과를 확인할 수 있다. 또한 HRB 뿐만이 아니라 HC 와 AFJS 역시 VA 차이가 발생하면서 ARC 와 성능차이가 크게 발생하는 사실을 확

인할 수 있었다.

4. 결론 및 향후연구

워크플로우 매니지먼트 시스템은 그리드, 클라우드 환경과 같은 다양한 환경에서 사용자에게 어플리케이션 수행을 위한 워크플로우의 설계와 처리를 지원한다. 그러나 현재의 과학탐구 성격을 갖는 어플리케이션들의 경우 대용량의 데이터를 처리해야 하기 때문에 보다 효율적인 시스템 자원의 사용을 통한 신속한 처리를 필요로 한다. 이를 위해 워크플로우를 분산 컴퓨팅 환경에서 효율적으로 처리할 수 있도록 워크플로우를 구성하는 작업들의 클러스터링 방식에 대한 연구가 진행되어 왔다. 본 논문에서는 기존의 비효율적인 워크플로우 작업 클러스터링의 방식과는 다르게 작업환경에 적합한 워크플로우 수행을 통해 작업들이 수행될 최선의 노드를 선정한다. 또한 해당 노드의 가용자원을 고려한 작업분배를 통해 워크플로우 매니지먼트 시스템을 통한 과학탐구 어플리케이션의 수행 성능을 향상시켰다. 추후 연구에서는 다수의 HM 들이 보다 더 많은 VM 들을 갖고 상호작용하는 환경에서 ARC 의 성능을 측정해서 기법의 타당성을 보일 것이다.

참고문헌

[1] Deelman, Ewa, et al. "Pegasus: A framework for mapping complex scientific workflows onto distributed systems." *Scientific Programming* 13.3 (2005): 219-237.

[2] Foster, Ian, et al. "Chimera: A virtual data system for representing, querying, and automating data derivation." *Scientific and Statistical Database Management, 2002. Proceedings. 14th International Conference on*. IEEE, 2002.

[3] Abramovici, Alex, et al. "LIGO: The laser interferometer gravitational-wave observatory." *Science* 256.5055 (1992): 325-333.

[4] Berriman, G. Bruce, et al. "Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand." *Astronomical Telescopes and Instrumentation*. International Society for Optics and Photonics, 2004.

[5] Graves, Robert, et al. "CyberShake: a physics-based seismic hazard model for Southern California." *Pure and Applied Geophysics* 168.3-4 (2011): 367-381.

[6] USC Epigenome Center,http://epigenome.usc.edu.

[7] SIPHT, http://pegasus.isi.edu/applications/sipht.

[8] Chen, Weiwei, et al. "Using imbalance metrics to optimize task clustering in scientific workflow executions." *Future Generation Computer Systems* (2014).

[9] Deelman, Ewa, et al. "Pegasus: Mapping scientific workflows onto the grid."*Grid Computing*. Springer Berlin Heidelberg, 2004.

[10] Liu, Quan, and Yeqing Liao. "Grouping-based fine-grained job scheduling in grid computing." *Education Technology and Computer Science, 2009. ETCS'09. First International Workshop on*. Vol. 1. IEEE, 2009.

[11] Chen, Weiwei, et al. "Balanced task clustering in scientific workflows."*eScience (eScience), 2013 IEEE 9th International Conference on*. IEEE, 2013.

[12] Chen, Weiwei, and Ewa Deelman. "Workflowsim: A toolkit for simulating scientific workflows in distributed environments." *E-Science (e-Science), 2012 IEEE 8th International Conference on*. IEEE, 2012.