작업 이력의 통계 분석을 통한 적응형 그리드 자원 선택 기법
(An Adaptive Grid Resource Selection Method Using Statistical Analysis of Job History)

허 신 영 [†]　　김 윤 희 [††]
(Cinyoung Hur)　　(Yoonhee Kim)

요 약 다양한 과학 분야에서 대규모의 계산집중적인 어플리케이션들이 많은 그리드 자원을 활용해감에 따라 그 실행 관리와 제어의 어려움도 증가하였다. 어플리케이션의 반복되는 실행으로 축적된 작업 이력을 참조하여 어플리케이션의 특성을 파악하고 그리드 자원 선택 정책을 결정하였다. 본 논문은 그리드 컴퓨팅 환경과 이를 활용한 어플리케이션의 이력을 분석하기 위해 통계적 기법인 PBDF(Plackett-Burman with Fold-Over)계획법을 적용하였다. PBDF는 그리드 환경과 어플리케이션에서 주요한 요인들을 파악하고, 그것들이 얼마만큼 영향을 미치는 가를 수치화한다. 영향력 큰 요인은 작업 이력에서 참조 프로파일을 찾고 적절한 자원을 선택하는데 사용하였다. 응용의 수행 결과를 다시 작업 이력에 포함시키고 인자의 신뢰도를 조정하였다. 본 연구는 항공우주 연구 그리드의 작업 이력을 분석하여 적응형 자원 선택 알고리즘을 제안하였다. 주요한 요인들의 영향력을 계산하고 자원 선택 정책에 반영하는 실험을 하였다. 또한, 수행이 끝난 후 인자의 신뢰도를 평가해 그리드 환경 변화에 적응하는 알고리즘의 유효성을 검증하였다. 오류가 빈번한 그리드 환경에서 자원 선택 기법을 평가하기 위해 다양한 시나리오에서 그 적응력을 실험하였다.
키워드 : 적응형 자원 선택, 작업 이력, 그리드 스케줄링, e-Science, Plackett-Burman

Abstract As large-scale computational applications in various scientific domains have been utilized over many integrated sets of grid computing resources, the difficulty of their execution management and control has been increased. It is beneficial to refer job history generated from many application executions, in order to identify application's characteristics and to decide selection policies of grid resource meaningfully. In this paper, we apply a statistical technique, Plackett-Burman design with fold-over (PBDF), for analyzing grid environments and execution history of applications. PBDF design identifies main factors in grid environments and applications, ranks based on how much they affect to their execution time. The effective factors are used for selecting reference job profiles and then preferable resource based on the reference profiles is chosen. An application is performed on the selected resource and its execution result is added to job history. Factor's credit is adjusted according to the actual execution time. For a proof-of-concept, we analyzed job history from an aerospace research grid system to get characteristics of grid resource and applications. We built JARS algorithm and simulated the algorithm with the analyzed job history. The simulation result shows good reliability and considerable performance in grid environment with frequently crashed resources.
Key words : Adaptive Resource selection, Job history, Grid computing, e-Science, Plackett-Burman

· 본 연구는 숙명여자대학교 SRC여성질환연구센터 특별연구비 지원으로 수행되었음
† 학생회원 : 숙명여자대학교 컴퓨터과학과
hurcy@sookmyung.ac.kr
†† 종신회원 : 숙명여자대학교 컴퓨터과학과 교수
yulan@sookmyung.ac.kr
(Corresponding author임)
논문접수 : 2009년 11월 11일
심사완료 : 2010년 2월 23일

정보과학회논문지: 시스템 및 이론 제37권 제3호(2010.6)

1. 서 론

그리드 컴퓨팅은 분산된 컴퓨팅 자원 및 데이터, 어플리케이션 등을 네트워크로 연결하여 단일한 자원으로 제공함으로써 다양한 분야에 활용되어왔다. 특히, 과학분야에서 그리드를 통해 분산된 과학기술 자원을 공유하여 연구 역량을 높일 수 있었기 때문에 많은 도메인에서 그리드를 활용하였다. 그러나 단순히 대 규모의 자원을 단일하게 이용하는 것에서 나아가 어플리케이션의 특성을 파악하여 그리드를 활용하는 것이 필요하다. 일반적인 과학 어플리케이션은 방대한 양의 데이터, 장시간의 수행 등의 특성이 있는데, 세부적인 도메인에 따라 어플리케이션의 특성이 다르기 때문에 필요한 그리드의 특성도 다르다. 본 논문에서 대상으로 하는 유체해석은 수치해석과 가시화 중심적인 어플리케이션으로서 계산 그리드가 필요하다. 이러한 특성을 고려하여 자원을 구성한다면 연구 효율성을 높일 수 있다.

어플리케이션의 특성과 더불어 어플리케이션의 그리드 사용 이력은 여러 분야에서 효과적으로 사용될 수 있다. 사용 이력을 다각도에서 분석하여 얻은 정보는 그리드 자원 관리의 연구, 그리드 관리 및 운용, 그리드 설계 및 성능 평가 등에 쓰인다. 이러한 사용 이력 분석에 다양한 통계적인 기법들을 적용하는 연구가 활발히 진행되고 있다. 통계적 기법들은 어플리케이션의 특성과 그리드의 속성을 인자(factor)들로 정의하고 각 인자의 영향력과 인자들 간의 상호작용을 파악한다. 이러한 인자들을 고려하여 자원을 선택하면 그리드를 효율적으로 이용할 수 있다. 하지만, 인자들이 너무 많으면 인자의 영향력 및 상호작용을 파악하기 어렵다. 이에, Plackett-Burman(PB) 계획법을 사용하여 그리드 환경과 어플리케이션에서 유의한 인자들을 파악하고, 각 인자가 얼마만큼 영향을 미치는 가를 수치화하였다. 영향력 높은 인자는 자원 선택 시 활용하였다.

본 논문은 어플리케이션의 특성 및 그리드의 사용 이력을 통계적 기법으로 분석하여 어플리케이션 수행 성능에 영향을 미치는 인자를 파악하였다. 그 인자를 활용해 적확한 자원 선택을 하고 안정적인 작업 실행을 도모하였다. 이를 위해, 그리드에서 작업의 수행 결과 및 어플리케이션의 특성을 작업 프로파일로 정의하고, 이로 구성된 작업 이력을 분석하였다. 작업 이력 분석 시, 어플리케이션의 수행시간에 가장 큰 영향을 미치는 인자를 선별하며, 여기에 PB 계획법을 사용하였다. 자원 선택 정책은 영향력 큰 인자를 바탕으로 자원과 참조할 작업 프로파일을 선택하였다. 어플리케이션의 종료 후, 참조 프로파일에 명시된 수행시간과 실제 수행시간의 오차를 다음 자원 선택에 반영하였다. 새로 생성된 작업 프로파일은 작업 이력에 포함시켜 다음 자원 선택에 활용하였다.

본 논문 구성은 다음과 같다. 2장에서는 작업 이력과 적응형 자원 선택의 관련 연구를, 3장은 본 연구가 대상으로 하는 그리드 환경 분석 및 PB 계획법 소개를, 4장과 5장에서는 본 연구가 제안하는 작업 이력 기반 자원 선택 알고리즘과 그 시스템에 대한 설계 및 구조를 소개하며, 6장에서는 작업 이력 및 실험과 그 결과를 분석해 제안한 방법의 성능을 검증하고, 7장에서 결론을 정리하였다.

2. 관련 연구

그리드 컴퓨팅[1]은 다수의 조직과 관리 도메인에 걸쳐 다양한 자원들을 네트워크로 연결하여 단일한 자원으로 사용할 수 있도록 고안된 시스템이다. 또한 그리드는 소규모 지역부터 인터넷 규모까지 크기가 다양하고 새로운 조직을 추가시킬 수 있는 확장성이 있다. 본 논문에서 주 대상으로 하는 것은 HTC(High Throughput Computing) 그리드로서 배치 어플리케이션, 워크플로우, 파라미터 스윕과 같은 특성을 갖는 과학기술분야의 연구수행에 주로 활용된다. 파라미터 스윕(parameter sweep)은 많은 과학과 공학 분야에서 활용되는 어플리케이션으로, 실험의 집합으로 구성되며 각 실험은 개별적인 파라미터 값들을 갖는다. 이와 같은 시스템은 실제로 e-Science[2]라는 패러다임에 적용되고 있으며, 분산된 과학기술자원-연구자, 연구시설 및 장비, 연구정보 등-을 공동 활용함으로써 연구역량을 향상시켰다. e-Science는 대용량의 협업연구가 요구되는 순수과학 연구 분야에 적용되고 산업기술과 교육, 국방 경제 등의 분야까지 폭 넓게 활용되고 있다.

GWA[3,4](The Grid Workloads Archive)는 작업 이력을 축적하여 활용하는 연구이다. 그리드를 관리하는 기관별로 자원 관리 정책이 다르기 때문에, 그리드 연구자나 사업자들이 그리드를 활용하고 있음에도 그리드 작업 부하의 기록들이 전무 하다시피 하다. 이러한 배경에서 GWA는 그리드 작업 부하 저장소를 제안하고 작업 부하 데이터를 교환하는 동시에 그리드 커뮤니티가 만나는 접점을 제공하였다. 작업 부하 저장소를 세우는데 필요한 요구사항들을 정의하고 GWA와 이러한 요구사항들을 충족시키기 위한 접근방안들을 서술하였다. 또한, 그리드 작업 부하 정보를 공유하기 위한 포맷과 이와 관련된 툴을 소개하였다. 이 툴을 이용하여 GWA는 잘 알려진 9개의 그리드 환경으로부터 데이터를 수집하고 분석하였다. 최종적으로, 그리드 자원 관리의 연구, 그리드 설계, 동작, 유지보수에 GWA의 이용 가능성을 제안하였다.

NIMO(NonInvasive Modeling for Optimization)[5]는 대규모의 그리드 컴퓨팅 환경에서 계산집중적인 과학 어플리케이션들을 수행 할 때 그 실행시간을 예측하기 위한 비용 모델을 자동적으로 학습하는 시스템이다. 일반적으로 과학 어플리케이션은 하나 이상의 배치 작업들이다. 이러한 작업들을 효율적으로 실행하기 위해 적합한 자원을 찾는 일이 중요하다. 이를 위해 학습 샘플들을 생성하고, 통계적인 학습 기법을 적용하여 시스템 모델을 학습하였다. NIMO는 다양한 조건 하에서 능동적으로 어플리케이션을 배치시키고 감시하지만, 운영체제나 어플리케이션 자체에 어떠한 변경 없이도 수동적인 결과 수집으로 학습 데이터를 얻는다. 학습 시스템의 검증을 위해 DSCR(Shared Production Cluster at Duke)상에서 자주 수행된 생물학 관련 어플리케이션들 모델의 적확성과 효율성을 평가하는 실험을 하였고, 다양한 시나리오에서 자원 계획을 유도하는 모델을 사용하였다. NIMO와 본 연구의 시스템 파라미터를 비교할 때, 본 논문은 시스템의 동작 특성의 수행성공률과 자원 특성의 위치를 추가적으로 고려하였다. NIMO는 생물의학 분야의 대용량의 이미지 처리·데이터 검색과 고성능의 계산·병렬 어플리케이션들을 실험하였고, 본 논문은 항공우주 연구 분야의 계산 집중적이고 HTC를 요하는 유체 해석 어플리케이션에 적용하고 검증하였다.

시스템 모델을 만들지 않고 그리드 환경에 적응하기 위해, Gosia W.[6]는 어플리케이션의 수행 결과에 따른 자가 적응(Self-adaptive) 기법을 제안하였다. 이 연구는 어플리케이션의 특성을 파악하고 모델링하기 위한 비용문제를 지적하였다. 이를 해결하기 위하여 우선 임의의 자원을 선택한 후 수행 과정을 살펴 자원 할당을 조절하였다. 다양한 시나리오로 성능 병목 현상을 감소시키는 결과를 증명하였다. 하지만, 그리드 자원의 성능 병목 현상 해결에 초점을 맞췄기 때문에 오류가 빈번한 그리드의 특성을 고려한 실험이 부족하였다. 본 논문은 자원 오류 시 적응력 있는 알고리즘을 제시하고 다양한 시나리오에서 성능을 검증하였다.

시스템 모델의 바탕이 되는 지식 베이스(Knowledge Base)를 구축하고 검증하기 위해, 이론적인 배경이 확고하고 다양한 기법들을 제공하는 실험계획법(Design of Experiment)[7]이 적용되었다. 실험계획법은 시스템을 대상으로 실험을 하여 원하는 데이터를 수집한다. 그 중에 실험인자의 유의성을 검사하기 위한 실험 방법들 가운데, 일부실시법(Fractional Factorial Design)과 선별계획법이 효율적이다. 일부실시법은 실험변수들이 많아 전체 실험이 불가능할 때 적은 횟수의 실험으로 실험변수들의 주 영향과 중요한 상호작용들을 판단한다. 선별 계획법[7]은 수많은 실험 인자들이 존재할 때 원래의 실험을 설계하기에 앞서 중요한 실험 변수들을 선별하는 방법으로 주로 2-수준(2-level) 일부실시법과 PB 계획법 등이 사용된다. 실제로 PB 계획법은 그리드 스케줄러의 성능 평가[8] 및 작업 이력 분석[5] 등 다양한 분야에 적용되었다.

3. 그리드 환경 분석

3.1 대상 어플리케이션의 특성

그리드를 이용한 과학 분야의 어플리케이션으로서 다양한 분야에서 활용되는 전산 유체 역학(Computational Fluid Dynamics)[9]의 해석은 HTC(High Throughput Computing)를 수행하며, 크게 세 과정을 거쳐 이루어진다. 초기에는 해석의 대상인 격자를 제작하고 그 다음 단계로 생성된 격자를 기반으로 전산 유체 역학의 수치해석이 수행되며 최종적으로 해석된 결과는 별도의 가시화 소프트웨어를 통해 확인한다. 전산 유체 역학 분야에서는 해석 단계에서 계산 자원이 많이 필요하다. 해석단계는 연구자가 생성한 격자를 재처리하고 해석자를 사용하여 최종적으로 가시화하는 과정을 말한다. 수치해석 단계 이전에는 격자와 더불어 해석 조건과 범위를 다양하게 배치하여 다수의 작업을 동적으로 동시에 생성한다. 이 조건에 따라서 연구자가 풀고자 하는 작업의 수행시간이 결정되기 때문에, 본 연구는 이러한 전산 유체 역학의 수치 해석을 유체 해석이라 명명하고 그 특성을 어플리케이션의 특성으로 설정하였다. 특히, 유체 해석은 장시간 동안 수치 해석을 수행하기 때문에 반드시 안정적으로 작업을 수행할 수 있는 자원 선택이 요구되고, 이를 위해 이 단계의 작업의 생성 및 실행 과정 파악이 필요하다.

일반적인 유체 해석 작업은 먼저 작업을 수행할 계산 자원에 수치해석 작업에 이용될 입력파일들-격자, 매개변수 정보가 담긴 파일, 격자의 해석자-을 전송한다. 점선은 파일이나 입력 조건의 생성을 의미하고, 실선은 네트워크를 통한 파일의 전송을 의미한다(그림 1 참고). 유체 해석을 위한 실제 시스템은 항공우주 통합연구 환경인 e-AIRS(e-Science Aerospace Integrated Research System) 2.0[10]으로서, 한국과학정보기술원[11]의 클러스터들과 PRAGMA 그리드 자원을 이용한다. PRAGMA (Pacific Rim Application and Grid Middleware Assembly)[12]는 태평양 지역 어플리케이션과 그리드 미들웨어를 위한 연합으로 35개의 기관이 Globus 미들웨어[13]로 묶여진 대규모의 그리드를 제공한다. 그리드 자원을 효율적으로 사용하기 위해 그리드 스케줄러를 사용하여 연구자가 요청하는 작업을 어느 기관의 자원에 할당시킬지 결정한다. 그리드 자원은 오류가 빈번함에도 관리 권한, 규모, 종류 등이 매우 다양하기 때문에 그리

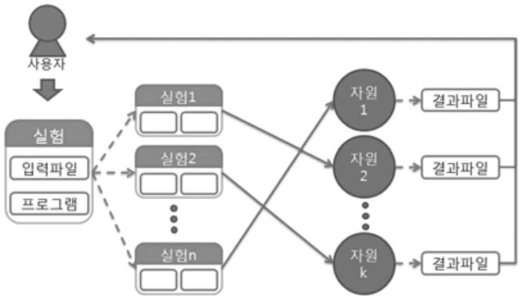

그림 1 작업의 생성 및 수행 과정

드 스케줄러가 각 자원에 대한 상태정보를 실시간으로 수집하기 어렵다. 사용 불가능한 자원에 작업을 제출하면 연구의 효율성을 저하시키게 된다. 따라서 오류가 빈번한 그리드 환경을 고려한 스케줄러가 필요하다.

3.2 인자의 영향력

P.Shivam[14]은 인자와 인자의 영향력에 대해서 엄격하게 정의를 내리고 있는데, 이는 실험 계획법 적용을 위하여 반드시 전제되어야 한다. 인자를 그리드 어플리케이션의 수행시간에 영향을 주는 요인들로 정의하고 각 인자가 성능에 얼마만큼 영향을 미치는지 판단한다. 예를 들어, 작업과 관련된 인자(작업의 특성, 개수 등)와 자원과 관련된 인자(CPU 속도 및 개수, 메모리 크기 등)가 그리드 어플리케이션의 성능을 결정짓는 인자라고 볼 수 있다. 식 (1)로 인자를 표현하며 성능에 가장 큰 영향을 미치는 인자는 식 (2)로 표현한다.

$$F = \{f_1, f_2, \cdots, f_k\} \quad (1)$$

$$f_{max} = \{f \mid \text{MostEffective } f \in F\} \quad (2)$$

유체해석 어플리케이션은 실행의존성 없는 실험들의 파라미터를 다양하게 변화시켜 그 결과를 관찰하기 때문에 특정 파라미터에 따라 수행시간이 크게 영향을 받는다. 또한 그리드 자원의 상태와 성능에 따라 수치해석의 수행 시간이 결정되므로 그리드와 관련된 인자를 고려하는 것도 중요하다. 본 연구에서 정의한 인자들은 유체해석 어플리케이션, 자원, 네트워크와 관련한 주요 인자들이다. 어플리케이션 인자로는 물체 주위의 공기의 유동 속도와 음속의 비를 나타내는 마하 수(MA), 주어진 유동 조건에서 관성에 의한 힘과 점성에 의한 힘(Viscouse force)의 비를 나타내는 레이놀즈 수(RE), 날개 면과 바람의 공기흐름 방향과 각도를 나타내는 받음각(AOA), 실험 반복 횟수(ITMAX)를 선정하였다. 자원과 관련한 인자로는 CPU 성능, 메모리, 안정성을 대상으로 하였고, 네트워크 관련한 인자는 대역폭, 지연, 거리를 선정하였다.

4. 작업 이력 기반 자원 선택 알고리즘

4.1 시스템의 작업 프로파일과 작업 이력

작업 프로파일은 어플리케이션의 특성, 수행에 필요한 데이터, 작업을 수행한 자원의 정보, 작업 수행시간을 포함한다. 각 요소들 간의 상호작용으로 인해 작업 수행시간, 작업 성공률 등이 영향을 받으므로, 작업 프로파일로 작업이 수행되었던 기록을 구조적으로 표현할 수 있다. 또한, 자원의 가용 성능과 상태가 시간에 따라 동적으로 변하는 그리드 환경에서 작업 프로파일의 수행시간 정보는 간접적으로 자원의 상태 정보를 제공한다.

$$P = \langle j, r, T_j, s \rangle \quad (3)$$

$$j \in J = \{j_1, j_2, \cdots, j_n\},$$

$$r \in R = \{r_1, r_2, \cdots, r_m\},$$

$$T_j > 0, \; s = \{\text{DONE, FAILED}\}$$

작업 프로파일은 식 (3)과 같이 표현한다. 요청된 작업들 중 하나의 작업(j)이 그리드의 특정 자원(r)에 할당되어 수행된 시간(T_j)으로써 단일 작업 프로파일을 구성한다. 표 1은 실제 작업 프로파일의 예이다. 예를 들어, 고유한 작업 아이디 '1000000233'으로 구별되는 작업 프로파일은 어플리케이션 인자인 {RE, AOA, MA, ITMAX}의 값이 {5, 7, 0.7, 10000}이고, Status가 완료(Done)이고, rocks-153 클러스터에서 1701초간 수행한 정보를 나타낸다. 다시 말해, 유체해석 어플리케이션의 수행시간은 사용자가 입력한 파라미터 값과 그리드 스케줄러에 의해 선택된 자원에 결정된다.

작업 이력은 표 1의 P와 같은 개별 작업 프로파일들이 축적된 집합으로, 식 (4)로 표현한다. 작업 프로파일을 구성하는 정보를 바탕으로 작업 이력이 갖는 정보, 경향, 규칙을 도출하고 그리드 자원을 관리하는데 활용하였다. 작업 프로파일을 구성하는 항목(C_j)은 필요에 따라 바뀔 수 있다. 작업이 실행될 때마다 새 작업 프로파일이 생성되어 작업 이력에 추가하였다. 축적된 작업 이력 중에서 유의한 작업 프로파일을 선별하여 활용하는 것이 중요하기 때문에, 식 (5)와 같이 유의한 작업 프로파일들은 PR(Profile Reference) 집합으로 선별하였다. 작업 프로파일을 선별하는 기준은 PB 계획법으로

표 1 작업 프로파일과 작업 이력의 예

JobID	RE	AOA	MA	ITMAX	Status	Resource	Execution Time(sec)
1000000233	5	7	0.7	10000	DONE	rocks-153	1701
1000000241	5	7	0.7	10000	DONE	sakura	1458
1000000246	5	7	0.7	10000	DONE	nucleus	1321

구한 인자(f_{max})를 기준으로 하였다.

$$P \in JobHistory \quad (4)$$

$$PR = \{\forall P \mid f_{max} \text{ of } P == C_j\} \quad (5)$$

4.2 시스템의 작업 프로파일과 작업 이력

그림 2의 알고리즘은 영향력이 가장 큰 인자와 참조 작업 프로파일들 그리고 작업 특성에 적합한 자원을 구하고, 그림 3의 알고리즘으로 참조 작업 프로파일 집합을 확보하였다. 영향력이 큰 인자가 없다면, 한번도 PBDF가 적용되지 않았다고 간주하고 PB 계획법으로 시스템에 영향을 미치는 인자들을 평가하였다. 영향력이 가장 큰 인자를 f_{max}로 지정한 후, JobHistory(F,j) 함수를 통하여 참조 작업 프로파일 집합을 확보하였다. 참조 프로파일 집합에서 작업 j 가 수행되었던 이력에서 자원 요소들의 집합을 R 로 관리하였다. 이 자원들을 f_{max}를 기준으로 평가하였다. 예를 들어 f_{max}가 안정성이라면 안정성이 가장 높은 자원을 $r_{selected}$에 저장하고, 선택된 자원에 작업을 수행시켰다. 어플리케이션과 관련된 인자 중 영향력이 높은 것을 중심으로 작업 이력에서 프로파일을 참조하였다.

```
Given j ∈ J, T_j
    f_max = null, PR = ∅, r_selected = null
    if f_max not exists then
        PBFD(F, T_j)
        Rating F in deceasing order
        f_max = one of F related to resource
    endif
    PR = JobHistory(F, j)
    R = PR(r)
    r_selected = r of f_max(R)
    Dispatch j on r_selected
```

그림 2 작업 이력 기반 자원 선택 알고리즘

```
JobHistory(F, j)
    PR = null
    repeat each Profile
        Find P where C_j == f_max
        PR = PR ∪ P
    until P == empty
    return PR
```

그림 3 참조 작업 프로파일 검색 알고리즘

4.3 시스템의 작업 프로파일과 작업 이력

$r_{selected}$에서 수행된 작업은 실패할 수도 있고 성공할 수도 있기 때문에, 그림 4의 알고리즘을 통해 오차를 고려하여 상황에 맞게 적응하였다. 작업이 수행된 후 새로운 작업 프로파일(P_{new})을 작업 이력에 포함시켰다. 작업이 실패하면, 그 작업이 수행된 자원의 안정도를 조정하였다. 작업이 성공하면, 새로운 작업 프로파일과 참조하였던 프로파일의 시간 요소를 비교하였다. 두 프로파일의 시간 차이는 작업 프로파일 참조 알고리즘의 오차이다. 오차가 한계치(Threshold) 보다 작으면 인자에 신뢰도를 높이고 그렇지 않으면 신뢰도를 낮춘 후 다시 PBDF를 수행하였다. 한계치는 어플리케이션의 특성을 바탕으로 사용자가 지정할 수 있다.

```
After finish the j on r_selected
    P_new = ⟨j, r_selected, T_j, s⟩
    JobHistory = P_new ∪ JobHistory
    if P_new.s == Failed then
        decrease Stability of r_selected
    else
        Err = |T(P_new) − T(P_selected)|
        if Err < Threshold Then
            Increase Credit f_max
        else
            Decrease Credit f_max
            Repeat PBFD(F, T_j)
        endif
    endif
```

그림 4 적응형 알고리즘

5. 작업 이력 기반 자원 선택 시스템 설계 및 구현

본 논문에서는 그리드를 이용하는 어플리케이션의 특성을 파악하여 필요한 자원의 특성도 선별하여 활용하고, 자원을 선택 하는데 사용자로 하여금 다양한 정책을 적용할 수 있도록 한다. 분산 되어있는 자원들을 그리드 미들웨어를 통해 VO(Virtual Organization)로 구성하여 다양한 자원을 이용한다. 그림 5에서와 같이 그리드 미들웨어를 통해 알맞은 자원 선택, 자원의 이용 상황 모니터링, 작업 부하 모니터링, 분산되어 있는 자원 사용을 위한 공유 서비스 등을 이용할 수 있다. 이러한 환경을 기반으로 시스템 미들웨어에서 작업 이력 기반 자원 선택 시스템의 주요 기능들을 제공하였다.

시스템 미들웨어에서는 사용자가 정의한 어플리케이션의 특성과 그에 필요한 자원의 특성이 정의된 프로파일 스키마에 기초하여 작업 프로파일을 모은다. 프로파

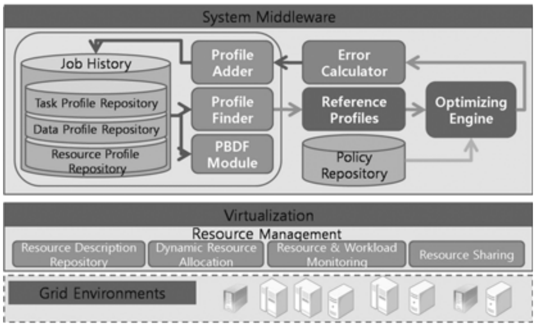

그림 5 작업 이력 분석 기반 그리드 자원 선택 시스템 구조

일들은 각각 프로파일 저장소(Task, Data, Resource Profile Repository)에서 관리되고 특정 어플리케이션이나 자원 상태의 추이를 분석하기 위하여 활용된다. Plackett-Burman 계획법을 제공하는 모듈(PBDF module)은 프로파일 및 작업 이력(Job History)에서 여러 가지 참조 작업 프로파일들을 추출한다. 자원 선택 정책(Policy Repository)을 바탕으로 최적화 엔진(Optimizing Engine)이 최종적으로 어플리케이션 특성에 적합한 자원을 선택한다. 생성된 작업 프로파일은 프로파일 추가기(Profile Adder)에 의해 순환적으로 작업 이력에 모아지며 주기적으로 오차 계산(Error Calculation)을 한다.

6. 실험 및 결과 분석

6.1 작업 이력 분석

작업 이력 분석을 통하여 시스템에 대해 다양한 정보를 파악할 수 있다. 유체해석 그리드의 작업 이력을 분석하면, 그리드를 구성하는 클러스터 중 f32와 sakura의 이용 빈도가 월등히 많음을 알 수 있다(그림 6 참조).

그림 7은 2008년 8월부터 2009년 3월에 걸친 작업의 분포로서, 2008년 11월과 12월에 집중적으로 작업이 제출되었음을 보여준다. 그림 8은 유체해석 그리드에서 수행된 작업들은 대체로 3750분 이내로 종료함을 보여준다. 본 연구는 실험을 위하여 분석한 내용 중 자원 별 이용 빈도와 수행시간을 실험 데이터로 선정하였다.

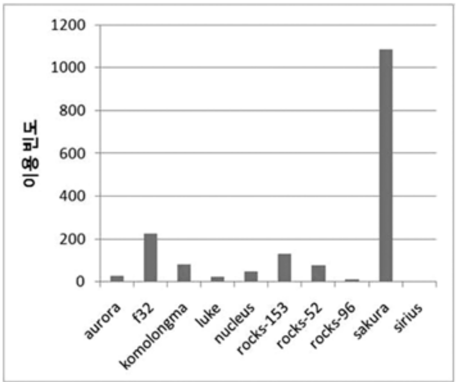

그림 6 자원 별 이용 빈도

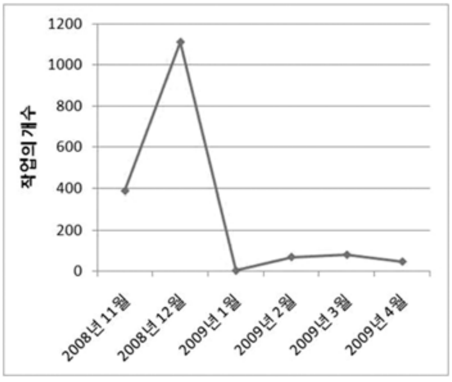

그림 7 날짜에 따른 작업 분포

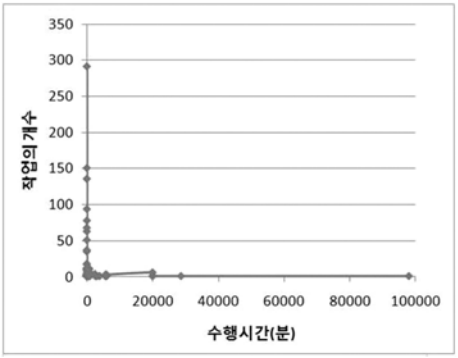

그림 8 수행시간의 히스토그램

6.2 인자의 영향력 분석

실험에 앞서, 위에서 정의한 인자들을 PB 계획법으로 분석하였다. PB 계획법에서 (+), (-)값은 일반적으로 인자가 가질 수 있는 값 보다 더 크거나 작은 값을 설정해야 한다. 본 연구는 실험을 위해 작업 이력에서 각 인자의 (+), (-) 값을 분석하였다. 표 3의 MA와 AOA는 표준 편차가 작고 정규 분포를 보였기 때문에 중위수(median)를 기준으로 1사분위수와 3사분위수를 각각 (+)값과 (-)값으로 설정하였다. 반대로 RE와 ITMAX는 표준편차가 크게 나타나서 특정 값을 지정하기보다 평균(mean)을 기준으로 범위를 나누었다.

3.2절에서 언급한 자원 인자들 중 실험을 위하여 CPU 클럭, 안정도, 네트워크 인자를 선택하였고 각 인자 값에 따라 자원들을 분류하였다. 자원은 PRAGMA 그리드 모니터링 서비스[15]를 참고하여 실제 작업 이력에 나타난 자원들만을 대상으로 하였다. 안정도[16]는 전체 수행된 작업 중 성공한 작업의 비율이다. 대역폭에 대한 정보가 부족한 클러스터는 위치를 고려하여 네트워크 인자의 값을 설정하였다. 모든 자원 인자에 대하여 (+)와 (-)로 나누는 경계 근처의 값을 가지는 자원은 양쪽에 모두 활용하였다.

표 2의 PBDF(Plackett-Burman Design with Foldover)는 실제의 작업 이력을 분석하여 다양한 인자들이 수행시간에 미치는 영향을 수량화한 것이다. -1은 인자의 (-)값을 뜻하고, 1은 인자의 (+)값을 뜻한다. 실험 계획에서 일부실시법의 일종인 Plackett-Burman(PB) 계획법은 인자의 유의성(significance)를 검사하기 위해 Plackett-Burman에 의해 제안되었다. k=n-1개의 인자

표 2 실제 작업이력 기반으로 PBDF를 적용한 결과

	자원			어플리케이션				
	CPU 클럭	안정도	네트워크	RE	AOA	MA	ITMAX	수행시간(s)
1	1	1	1	−1	1	−1	−1	1507
2	−1	1	1	1	−1	1	−1	2401
3	−1	−1	1	1	1	−1	1	47889.5
4	1	−1	−1	1	1	1	−1	898.33
5	−1	1	−1	−1	1	1	1	279.5455
6	1	−1	1	−1	−1	1	1	128
7	1	1	−1	1	−1	−1	1	6786
8	−1	−1	−1	−1	−1	−1	−1	195.52
9	−1	−1	−1	1	−1	1	1	14373
10	1	−1	−1	−1	1	−1	1	25267
11	1	1	−1	−1	−1	1	−1	1266
12	−1	1	1	−1	−1	−1	1	7504
13	1	−1	1	1	−1	−1	−1	625
14	−1	1	−1	1	1	−1	−1	26517
15	−1	−1	1	−1	1	1	−1	363.8
16	1	1	1	1	1	1	1	1811.083
영향력	−61235	−41668.5	−13353	64790.05	71254.74	−94770.3	70264.48	
순위	1	2	3	4	2	1	3	

표 3 인자의 신뢰도

f_{max}	MA	AOA	RE	ITMAX	총 작업 수
MA	24	10	10	10	561

를 n번의 실험으로 실시할 때 사용하기 위하여 제안되었고, 여기서 n은 4의 배수이다. PB 계획법의 설계방법은 다음과 같다[7,17]. PB 계획법을 위해 정의된(+1, −1) 기호 행들을 나열하여 총 16행의 계획행렬을 만든다. 표 3에서 각 행은 인자들의 특정 값으로 구성된 실험을 뜻하는데, 인자의 부호와 그 결과 값을 곱하여 모든 행에 걸쳐 합산하여 인자의 영향력을 구한다. 여기서 합산한 값의 절대값을 영향력으로 본다. 예를 들면, CPU 클럭의 영향력은 $|1507 - 2401 - 47889.5 \cdots - 363.8 + 1811.083|$이다. 최종적으로, 수행시간에 미치는 영향력에 따라 각 인자들의 순위를 매길 수 있다.

분석 결과, 자원과 관련된 인자로는 CPU 클럭이 수행시간에 가장 영향을 많이 미쳤고, 두 번째로 안정도가 영향력이 높았으며 네트워크는 영향력이 낮았다. 어플리케이션과 관련된 인자로는 MA 파라미터가 가장 영향력이 높았다. 이렇게 분석한 각 인자의 영향력에 따라 실험에 활용하였다. 첫 번째 시나리오는 자원의 오류가 없다는 가정 하에, 세 가지 자원 선택 알고리즘을 비교한다. 두 번째 시나리오는 자원에 오류가 났을 때 차선적인 자원 선택과 오류 난 자원에 대한 평가를 조정하는 경우를 살펴본다. 마지막 시나리오는 작업이력이 풍부해짐에 따라 자원 선택의 신뢰도가 증가함을 보여준다.

각 실험에 쓰인 가상 작업 부하는 유체 해석을 위한 항공우주 통합연구 환경인 e-AIRS 2.0의 작업 이력에서 작업 부하의 정규분포를 참고하였다[18]. 이하의 두 실험을 위하여 Java 기반의 그리드 시뮬레이터인 GridSim[19]을 사용하였고, 실제 자원의 속성을 바탕으로 가상 그리드 자원을 구성하였다.

6.3 시나리오 1: 여러 자원 선택 방법의 성능 비교

이 시나리오에서는 본 논문에서 제시한 방법과 성능 우선 자원 선택, 무작위 자원 선택 방법을 비교하였다. 그림 9는 자원 선택 방법 별 평균 수행 시간이다. 본 논문에서 제시하는 작업 이력 기반 자원 선택 알고리즘은 가장 짧은 수행시간을 보장하는 성능우선 자원선택 방법보다는 평균 수행 시간이 길었으나, 무작위로 자원을 선택하는 방법에 비해서는 짧은 수행 시간을 보장할 수 있다. 그러나 그리드의 일부 자원이 갑자기 오류가 발생하여 작업의 수행이 불가능 한 경우가 많다. 이러한 그리드의 속성을 배제하였기 때문에 성능 우선 자원 선택 방법은 실제 상황에서 한계가 있다.

6.4 시나리오 2: 오류를 포함한 환경의 실행시간 비교

이 시나리오는 그리드의 일부 자원이 갑자기 오류가 발생하여 요청되는 작업의 수행이 불가능 한 경우를 고려하였다. 본 논문에서 제안한 알고리즘은 작업 수행 이력에 따라 조정되는 자원의 안정도를 고려하여 자원 선택을 하므로 오류가 나는 상황에 적응력 있게 대처하였다. 그림 10은 사용자와 자원의 거리와 오류 발생 환경을 고려한 자원 선택 실험 토폴로지이다. 오류발생기가 임의의 자원에 오류를 발생시키고, 오류가 발생한 자원에 제출된 작업은 다른 자원에게 제출하였다. 그림 11은

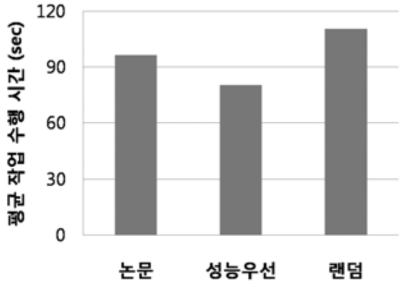

그림 9 자원 선택 방법 별 평균 수행시간

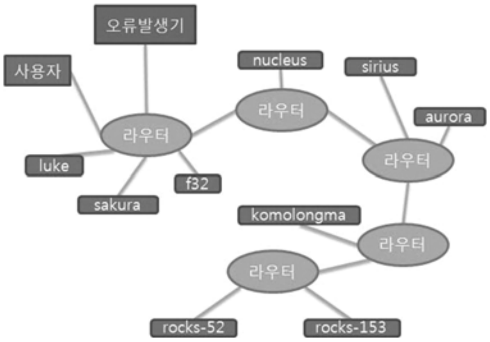

그림 10 오류 발생 환경 하의 자원 선택 실험 토폴로지

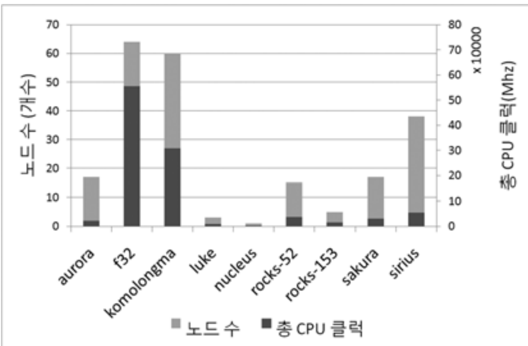

그림 11 유체해석 그리드의 성능(CPU 클럭)과 규모(총 노드 수)

그리드 자원 별로 총 CPU 클럭과 노드의 수가 비례하지만, 기관 별 규모와 CPU 성능에 따라 차이 남을 보여준다. 본 실험은 사용자와 자원의 거리, 자원의 성능 및 규모를 고려한 토폴로지를 구성하였고, 시나리오 1의 가상 작업 부하를 사용하여 총 5회 실험했다.

그림 12에서 자원 별로 제공하는 총 머신 수와 평균 오류 머신 수를 비교하였다. f32와 komolongma와 같이 많은 머신을 가지는 자원은 오류 머신이 많이 발생해도 자원의 전체적인 성능 저하가 일어나지 않지만, nucleus

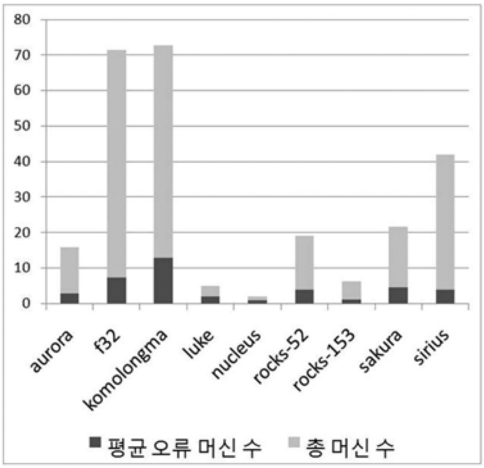

그림 12 자원 별 총 머신 수와 평균 오류 머신 수

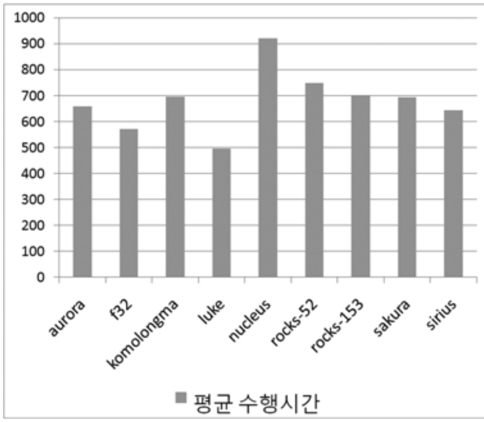

그림 13 자원 별 평균 수행시간

와 같이 규모가 작은 자원은 오류 발생 상황에 취약하다.

그러한 이유로 그림 13에서 다른 자원에 비해 nucleus가 평균 수행 시간이 길게 나타났다. 오류가 발생한 자원에 제출되었던 작업이 다시 다른 자원에서 수행되기 때문에 전체적인 수행시간이 증가한다. 그림 14에서 총 작업 수에 비해 실패한 작업의 수가 비교적 적은 f32와 rocks-153이 안정도가 높게 나타났고, 그림 13에서 그 자원들의 평균 수행 시간도 비교적 우수하게 나타났다. 이렇게 수행된 작업 이력들은 자원의 성능뿐만 아니라 안정도에 대한 정보도 제공하므로 자원 선택에 효과적으로 활용된다. 표 3은 시나리오 2에서 총 561개의 작업이 수행되면서 참조한 작업 프로파일과의 수행시간 오차를 계산한 결과를 보여준다. 각 인자의 신뢰도를 10으로, 한계치를 180초로 설정하고 실험하였다. 작업 이력에서 참조 작업 프로파일 집합을 확보할 때 검색 기준으로 활용한 인자가 비교적 정확한 수행시간을 제공

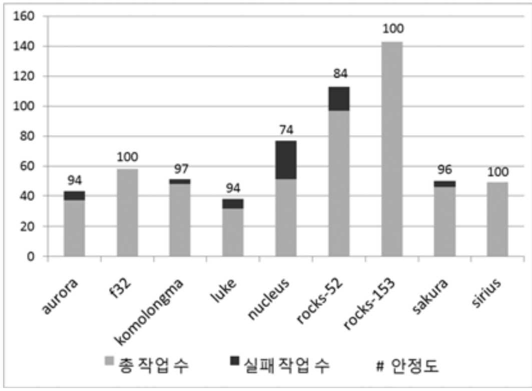

그림 14 자원 별 안정도

할수록 신뢰도를 증가시켰다. 결과적으로 표 2에서 결정된 MA의 신뢰도가 10에서 24로 증가한 것을 통해 비교적 정확한 참조 작업 프로파일을 제공하고 있음을 알 수 있다.

6.5 시나리오 3: 유체해석 그리드의 스케줄러와의 비교

이 시나리오에서는 유체해석 그리드에서 사용되는 실제 그리드스케줄러와 본 논문에서 제안한 작업 이력 기반 적응형 알고리즘으로 동작하는 스케줄러의 성능을 비교하였다. 과학 어플리케이션은 그 동작 시간이 길고 고성능의 자원을 필요로 하기 때문에, 오류 발생 시 작업을 처음부터 반복해야 하는 위험성이 따른다. 그렇기 때문에 안정성을 보장하는 자원 제공이 필수적이다. 따라서 본 시나리오에서 작업 이력을 기반으로 자원을 선택할 시 안정적으로 작업을 수행하는 능력을 검증하였다. 한 가지 작업을 100회씩 시나리오 2의 토폴로지 상에서 수행하였고 총 5회 반복하였다. 자원 오류로 인해 작업이 수행되지 못한 경우, 재 할당하여 모든 작업이 완전히 끝날 때까지 실험을 수행하였다.

시나리오 2의 결과 분석을 통하여, 유체해석 어플리케이션에 적합한 자원은 고성능이거나 오류에 잘 견디는 자원이다. 자원의 규모가 클수록 개별 노드의 오류 상황에 강하고, 규모가 작을수록 오류 상황에 취약하였다. 유체해석 그리드에 동원된 자원의 성능과 규모는 그림 11과 같다. 분석 결과, f32, komolongma, sirius이 상대적으로 적합하였다. 유체해석 그리드의 특성 상 자원의 상태 정보를 실시간으로 보고 받기 어렵고 기관 별로 자원 관리 정책이 다르기 때문에, 그리드 스케줄러는 자원의 규모를 고려한 라운드 로빈 기법으로 자원을 선택한다. 따라서 노드 수가 많은 자원은 더 많은 작업을 할당 받아 수행하게 된다. 그러나 실제 유체 해석 그리드 스케줄러의 운영은 그리드 자원의 안정도[16]를 고려하므로, 본 연구의 실험에 반영하였다.

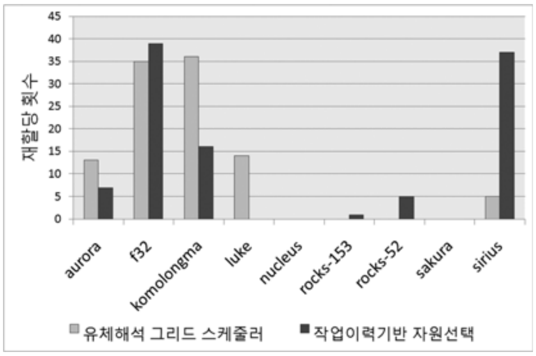

그림 15 오류 시 작업 재할당 횟수

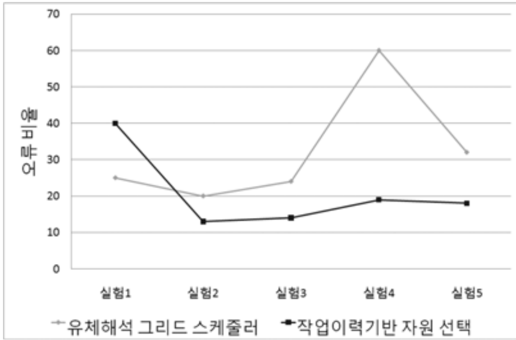

그림 16 자원 선택 방법 별 오류 비율

그림 15는 오류 발생 후, 동일한 작업을 다른 자원에 제출한 횟수이다. 실험 결과, 유체해석 그리드에서 오류로 재할당된 작업들은 주로 f32, komolongma, aurora, luke와 같은 자원이었다. luke의 규모보다 sirius가 더 크지만 실제의 안정도는 luke가 훨씬 높았다[16]. 유체해석 그리드의 실험은 이러한 안정도를 참고하였기에, 보다 많은 작업을 luke에 할당하였다. 반면에, 작업 이력 기반 자원 선택은 오류 시 주로 선택된 자원이 f32, sirius, komolongma, aurora이었다. 결과적으로, 작업 이력 기반 자원 선택이 어플리케이션의 재할당을 충분히 수용할 수 있는 성능과 안정도를 제공하였다.

그림 16은 총 5회 수행된 실험 별로 두 가지 자원 선택 방법의 오류 비율을 나타낸다. 오류 비율이란, 실패한 작업에게 다시 자원을 제공할 시 안정성을 보장하는 자원을 제공하는 비율이다. 오류 비율이 높을수록 작업이 다시 실패할 확률이 높고, 나아가 연구의 효율성을 저하시킬 수 있다. 작업 이력 기반 자원 선택은 실험 1에서 누적된 작업 이력이 없어서 오류 비율이 높았다. 그러나 실험을 거듭할수록 누적되는 이력을 활용함으로써 오류 비율이 안정되었다. 반면에 유체해석 그리드의 경우, 자원의 규모를 고려한 라운드 로빈 기법이기 때문

에 실험에 따라 편차가 크게 나타났다. 특히, 실험 4에서 규모가 큰 자원인 komolongma를 오류로 인하여 사용할 수 없었고 그로 인해 오류 비율이 급격히 높아졌다. 결론적으로, HTC를 수행하는 유체해석은 반복적으로 수행되는 특성을 갖기 때문에, 작업 이력이 풍부해짐에 따라 유체해석 스케줄러보다 향상된 성능과 안정성을 제공한다.

7. 결론 및 향후 연구

본 논문은 어플리케이션의 특성 및 그리드의 사용 이력을 통계적 기법으로 분석하여 어플리케이션 수행 성능에 영향을 미치는 인자를 파악함으로써 보다 효율적인 자원 선택을 하고 그리드의 활용도를 증대시키고자 하였다. 이러한 목적 하에 실험 계획법인 Plackett-Burman 계획법을 적용하여 항공우주 연구 분야의 유체해석 어플리케이션의 특성 및 그리드의 속성을 파악하고 작업 이력을 분석하였다. 시스템의 성능에 큰 영향을 미치는 인자로 작업 이력을 분석하여 어플리케이션을 위한 자원을 선정하였다. 작업을 수행한 후 작업 이력 분석의 정확도를 재평가함으로써 적응력 있는 자원 선택 기법을 제공하였다. 또한 작업 이력이 풍부해짐에 따라 자원 선택의 신뢰도를 높였다.

향후 연구로 다양한 어플리케이션이 그리드를 활용할 수 있도록 각 어플리케이션의 특성을 손쉽게 정의할 수 있는 서비스를 연구할 것이다. 어플리케이션에 맞춤형 자원을 제공하는 동시에 여러 어플리케이션들이 그리드를 공유함으로써 그리드 활용도 증대를 기대할 수 있다. 또한 영향력 있는 인자에 현실적인 가중치를 부여하고 실제 작업 이력과 비교연구를 수행함으로써 본 논문에서 제안한 알고리즘을 다듬어 나갈 것이다. 나아가 어플리케이션의 수행시간 외에 사용자의 다양한 요구사항에 대해서 본 논문이 제시하는 방법을 적용하여, 그리드 사용 및 관리 연구에 대한 활용을 확대시키고 세부적인 실험을 통해 평가할 것이다.

참 고 문 헌

[1] Klaus Krauter, Rajkumar Buyya, Muthucumaru Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," Software-Practice & Experience, vol.32 no.2, pp.135-164, February 2002.

[2] 김종암, 김윤희, 김병수, 안재완, 김진호, 이준석, 최중근, 이근배, 조정현, 김지영, 허신영, 강상현, 류근영, "다분야 유체해석을 위한 e-Science 기술개발 및 활용연구," 보고서 I-08-GG-05-01R-1, 한국과학기술정보원, 2008.

[3] Alexandru Iosup, Hui Li, Mathieu Jan, Shanny Anoep, Catalin Dumitrescu, Lex Wolters, Dick H. J. Epema, "The Grid Workloads Archive," Future Generation Computer Systems, v.24 n.7, pp.672-686, July 2008.

[4] The Grid Workloads Archive, http://gwa.ewi.tudelft.nl

[5] Piyush Shivam, Shivnath Babu, Jeff Chase, "Active and Accelerated Learning of Cost Models for Optimizing Scientific Applications," International Conference on Very Large Data Bases (VLDB), pp.535-546, Seoul, Korea, September 2006.

[6] Gosia Wrzesinska, Jason Maassen, Henri E. Bal, "Self-adaptive applications on the grid," Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming, pp.121-129, San Jose, California, USA, March 2007.

[7] Montgomery Douglas C., "Design and Analysis of Experiments Fifth Edition," John Wiley & Sons, Inc. 2000.

[8] Daniel C. Vanderster, Nikitas J. Dimopoulos, Randall J. Sobie, "Improved Grid Metascheduler Design using the Plackett-Burman Methodology," Proceedings of the 21st International Symposium on High Performance Computing Systems and Applications, p.9, May 2007.

[9] Computational Fluid Dynamics, http://www.cfd-online.com/

[10] e-AIRS 2.0, http://repository.kisti.re.kr:8080/gridsphere/gridsphere

[11] 한국과학기술정보원, http://www.kisti.re.kr

[12] PRAGMA(Pacific Rim Application and Grid Middleware Assembly), http://www.pragma-gird.net/about/

[13] Globus Alliance, http://www.globus.org/

[14] Piyush Shivam, "Proactive Experiment-Driven Learning for System Management," Ph.D. Dissertation, Department of Computer Science, Duke University, 2007.

[15] PRAGMA Grid Monitoring home page, http://pragma-goc.rocksclusters.org/scmsweb/

[16] Cinyoung Hur, Yoonhee Kim, "Adaptive Grid Resource Selection based on Job History Analysis using Plackett-Burman Designs," APNOMS 2009 LNCS 5787, pp.133-142, 2009.

[17] 박성현, "현대실험계획법(개정판)," 민영사, 2005.

[18] Hui Li, Rajkumar Buyya, "Model-Driven Simulation of Grid Scheduling Strategies," e-Science and Grid Computing, IEEE International Conference, pp.287-294, Bangalore, Dec 2007.

[19] GridSim 4.2 beta (released on Oct 12, 2008), http://www.gridbus.org/gridsim/

허 신 영

2007년 숙명여자대학교 컴퓨터과학과(학사). 2009년 숙명여자대학교 컴퓨터과학과(석사). 2009년～현재 숙명여자대학교 컴퓨터과학과 연구원. 관심분야는 그리드 컴퓨팅, 클라우드 컴퓨팅, 지식기반시스템 등

김 윤 희

1991년 숙명여자대학교 전산학과(학사) 1996년 Syracuse University 전산학과(석사). 2000 Syracuse University 전산학과(박사). 1991년～1994년 한국전자통신연구원 연구원. 2000년～2001년 Rochester Institute of Technology 컴퓨터공학과 조교수. 2001년～2004년 숙명여자대학교 컴퓨터과학과 조교수. 2004년～2009년 숙명여자대학교 컴퓨터과학과 부교수. 2009년～현재 숙명여자대학교 컴퓨터과학과 교수. 관심분야는 그리드 컴퓨팅 환경(PSE), 워크플로우 제어, 분산 어플리케이션/서비스 관리 등