최적포트폴리오 기법을 이용한 개방형 전산 환경의 안정성 확보에 관한 연구

김학진
연세대학교 경영대학
(hakjin@yonsei.ac.kr)

박지현
서울대학교 기술경영경제정책대학원
(april3@snu.ac.kr)

차세대 전산환경은 사용자들이 필요한 전산자원을 네트워크를 통해 공급받는 그리드 컴퓨팅 기반의 개방형 전산환경으로 진화할 것으로 예상된다. 개방형 전산환경의 도입은 전산자원 활용의 효율성을 높이고 협업을 증진시키며 공급의 유연성과 비용 절감 등의 효과를 가져올 수 있다. 그러나 네트워크를 통해 공급되는 특성으로 인해 개방형 전산자원의 성능에는 변동성이 수반된다. 전산자원의 성능에 변동성이 있는 경우, 주어진 예산과 시간만 고려하여 전산자원 서비스를 구성하는 단순 최적화 방법을 사용했을 때는 서비스의 최종 성능과 실행 시간 등을 규정한 서비스 수준 계약(Service Level Agreement, SLA)을 만족시키지 못할 위험이 높다. 따라서 개방형 전산환경의 서비스 브로커는 전산자원 공급의 안정성을 높이기 위해서 서비스를 구성하는 개별 전산자원의 성능 변동성을 고려하여 위험을 최소화하는 전산자원 포트폴리오를 구성할 것이 요구된다. 본 연구에서는 평균절대편차(Mean-Absolute Deviation, MAD) 포트폴리오 최적화 기법을 이용하여 서비스 브로커의 공급 안정성을 향상시키는 방법을 제시하였다. 제시된 최적화 기법의 효과를 알아보기 위한 방법으로 가상의 개방형 전산환경을 모델링하고, 고객의 제약 조건과 개방형 전산환경의 변동성 정도에 따라 전산자원 공급 서비스의 성공률을 시뮬레이션 하였다. 시뮬레이션 결과로서, 첫째, 단순 최적화 방법보다 변동성을 감안한 MAD 포트폴리오 최적화 기법을 이용했을 때 공급의 안정성이 뛰어난 것을 확인할 수 있었다. 둘째로는 특히 개별 전산자원의 변동성 예측의 정확성이 높아질수록 성능도 더욱 향상되는 결과를 가져왔다. 셋째, 측정된 변동성을 이용하여 개방형 전산자원의 가격을 할인하는 정책을 추진할 경우 서비스 공급 범위의 확대에도 효과가 클 것으로 예상되었다.

논문접수일 : 2008년 12월 29일 논문수정일 : 2009년 04월 05일 게재확정일 : 2009년 05월 16일 교신저자 : 박지현

1. 서론

차세대 전산 환경은 하드웨어, 소프트웨어, 데이터 등의 전산 자원을 개인별로 '소유하던' 기존의 방식을 벗어나 가상화된 조직(Virtual Organization) 내에서 '공유하고 필요에 따라 서비스 형태로 공급받는' 방식의 네트워크 기반의 그리드 컴퓨팅이 될 것으로 예측되고 있다(Foster et al., 2002; Hwang and Kesselman, 2003). 이러한 환경에서는 네트워크에 연결하면 언제 어디서나 필요한 양과 기능의 하드웨어, 소프트웨어 및 데이터를 공급받을 수 있게 되어, '네트워크가 곧 컴퓨터'라는 공식이 성립하게 되어(Minoli, 2004), 개인은 물론이고 기업도 조직의 울타리를 넘어 네트워크

* 본 연구는 방송통신위원회 및 정보통신연구진흥원의 방송통신정책연구센터운영지원사업의 연구결과로 수행되었음(IITA-2009-(C1091-0901-0003)).

상에 존재하는 여러 서비스 제공자(Service Provider, SP)들이 공급해주는 다양한 종류의 전산 자원을 서비스 형태로 이용할 수 있게 된다(Foster et al., 2002). 이른바 개방형 전산 환경이 도래하는 것이다.

기업이 개방형 전산 환경을 선택하는 이유는 크게 3가지로 정리된다(Thanos et al., 2007). 첫째는 기업 내부에 지리적으로 산재되어 있는 전산 자원을 통합하여 성능 효율을 최대화 하는 것이다. 이러한 기술은 최근 그린 IT 추세와 맞물려 대대적인 인터넷데이터센터(Internet Data Center, IDC)의 통폐합 등으로 이미 진행 중에 있다. Thanos et al.(2007)에 따르면 EU에서 시행하고 있는 그리드 컴퓨팅을 이용한 비즈니스 모델 구축 프로젝트에 참여하는 기업의 70%가 사내 전산 자원을 최적화 하는 엔터프라이즈 그리드(Enterprise Grid)를 구축하고 있다. 둘째는 협력업체간 정보 및 자원 공유 기능을 향상시키기 위한 것이다. 고가의 전산 장비의 경우 파트너쉽을 맺은 기업들이 함께 공유함으로써 개별적으로 장비를 구입하고 보유함으로써 가지는 비용 부담을 줄일 수 있다. 또한 협력업체 사이에 구축된 파트너 그리드(Partner Grid)를 통해 협업환경을 향상시킬 수 있다(Boden, 2004). 그러나 이 경우 협업 기업들 간에 기여도에 대한 명확한 원칙을 세워놓지 않으면 무임승차로 인해 전체 전산 시스템의 효율이 낮아질 위험이 있다. 마지막으로 유틸리티 컴퓨팅 시스템을 구축하여 전산 자원을 보유하고 필요로 하는 개인이나 기관이 전산 자원의 공급자가 되거나 구매자가 되는 경우이다. 유틸리티 컴퓨팅 시스템은 전기, 수도, 가스와 같은 공공 기반시설처럼 전산 자원도 유틸리티 형태로 소비자에게 공급되며 소비자는 사용량만큼의 가격만 지불하면 되는 차세대 전산 시스템의 이상적인 모델이다(Rappa, 2004). 기업들은 내부에 전산 시스템을 구축하지 않고, 유틸리티 컴퓨팅 시스템에 접속하여 네트워크를 통해 공급되는 전산자원을 사용할 수 있으며, 설령 내부 전산시스템을 갖추고 있더라도 피크타임시 부족한 자원을 네트워크를 통해 외부로부터 공급받을 수 있다. 그러나 공급처와 시스템의 복잡성을 효과적으로 제어하여 양질의 서비스를 안정적으로 공급해주는 기술의 부족으로 아직까지 업체들에 의해 선뜻 도입되지는 못하고 있다(Thanos et al., 2007).

개방형 전산 환경에서는 공급자와 시스템에 상관없이 전산 자원이 기능에 따라 규격화된 제품 서비스로 제공된다(Kenyon and Cheliotis, 2004). 이러한 환경에서는 소비자가 동일한 성능을 구현함에 있어 선호에 따라 공급자와 시스템을 다양하게 선택할 수 있는 이점이 있다. 그러나 다른 한편으로는 네트워크를 통해 공급받는 전산자원의 성능은 필연적으로 네트워크의 성능에 영향을 받을 수밖에 없고, 네트워크상에서 발생하는 다양한 문제들로 인해 공급되는 성능에 변동성을 가질 수밖에 없다. 또한 공급자 별 시스템 관리의 정도 및 시스템 별 특성에 따라 각 전산자원의 안정성에 차이가 발생하여 원하는 전산 자원 성능을 정확히 보장하는 것이 더욱 어려워질 수 있다. 이와 같은 이유로 고객에게 정확하고 안정적인 서비스를 제공하는 것은 개방형 전산환경의 논의 초기 단계부터 꾸준히 지적되어온 문제이며, 특히 이에 대한 해결 없이는 비즈니스 영역에서의 개방형 컴퓨팅 환경 구축은 어렵다고 여겨진다(Boden, 2004; Foster et al., 2002; Kenyon and Cheliotis, 2004).

본 논문에서는 개방형 전산자원의 변동성을 효과적으로 제어하여 고객에게 안정적인 서비스를 제공하기 위한 방안으로 평균절대편차(Mean-Absolute Deviation, MAD) 포트폴리오 최적화 기법을 소개

한다. 본 논문은 MAD 포트폴리오 최적화 기법의 효과를 알아보기 위해 가상의 개방형 전산환경을 모델링하고, 요청된 작업의 제약 조건과 개방형 전산환경의 변동성 정도에 따라 전산자원 공급 서비스의 성공률을 시뮬레이션 방법을 통해 측정한다. 시뮬레이션 결과, 개별 전산자원의 변동성은 고려하지 않고 고객이 요청한 완료시간과 예산만으로 전산자원 공급을 최적화하는 방법을 제시했던 Menasce and Casalicchio(2004)의 모델보다 MAD 포트폴리오 최적화 기법을 이용했을 때 공급의 안정성이 뛰어난 것을 확인할 수 있다. 또한 변동성을 감안한 전산자원공급 최적화를 도입할 경우, 측정된 변동성에 의거하여 전산자원의 가격을 할인할 수 있게 되어 더 많은 소비자에게 서비스를 제공하는 것이 가능하다.

본 논문의 구성은 다음과 같다. 제 2장에서는 개방형 전산자원의 안정성을 향상시키기 위한 기존의 연구들을 살펴보고, 제 3장에서는 본 논문에서 제시하는 MAD 포트폴리오 방법론을 설명한다. 제 4장에서는 성능 분석을 위한 시뮬레이션 환경 및 시나리오를 소개하고, 제 5장에서는 시나리오에 따른 시뮬레이션 결과를 보여준다. 마지막으로 제 6장에서 시사점 및 향후 연구에 대해 논의한다.

2. 기존 연구

개방형 전산환경은 초기 SETI@home으로 대표되는 연구 네트워크에서 시작하였다(SETI@home). SETI@home의 개방형 전산환경은 대규모 CPU연산이 필요한 프로젝트를 지원하는 고성능 컴퓨팅을 구현하기 위해 자발적 공여 차원에서 개인 PC의 자원을 공유하는 형태이다. 자발적인 공유를 기반으로 하는 환경에서는 개인 공여자의 상황에 따라 자원 공급이 갑자기 중단되어 작업이 중단되거나 악의적인 자원공급자가 결과를 조작하는 등의 문제가 발생할 수 있다. 이에 대한 대처방법으로 동일한 작업을 여러 컴퓨터에 동시에 보내어 결과를 비교하는 중복 수행 방법이 해결책으로 제시되었다. 그러나 개방형 전산환경을 상업용으로 사용하고자 할 경우 이 방법은 자원사용이 효율적이지 못하고, 추가되는 자원만큼 소비자가 더 많은 비용을 지불해야 하므로 적용이 부담스럽다(Domingues et al., 2007). 기업에서 개방형 전산자원 환경을 도입할 때는 개별 서버 및 네트워크에서 발생할 수 있는 위험 상황을 미리 파악하고 이를 최소화하는 환경을 구축하는 것에 더욱 노력을 기울여왔다(Kleban and Clearwater, 2004). 그 결과, 위험을 수반하는 외부네트워크로의 확장은 피하게 되었고, 클러스터 컴퓨팅 및 엔터프라이즈 그리드 정도가 상업화에 이용될 수 있었다(Minoli, 2004; Plaszczak and Wellner, 2005).

다른 한편으로는 네트워크를 통해 공급되는 개방형 전산자원이 성능에 불확실성을 내포하고 있기는 하지만, 그 변동성을 정확히 측정할 수 있다면 개방형 전산 환경을 상업화하는데 장애요소가 되지 않을 것으로 보는 시도들이 생겨났다. 액세스 그리드(Assess Grid)에서는 전산자원 공급자가 고객이 요청한 작업을 처리한 후, 서비스 브로커에게 각 전산자원이 서비스수준 계약(Service Level Agreement, SLA)에 명시된 성능을 내지 못했던 확률을 보고하게끔 하였다(Battre et al., 2007). 서비스 브로커는 각 전산자원의 성능과 함께 SLA 위반 확률을 미리 소비자에게 공지함으로써 다음번 작업 요청 시 공급자와 소비자 모두 보다 정확한 정보를 바탕으로 계약을 할 수 있고, 성공률을 높여 시장을 유지할 수 있게 된다. 그러나 이 때, 전산자원 공급자는 소비자를 유인하기 위해 위반 확률을 낮추어 보고할 유인이 있으므로 제 3의 기

관(여기서는 서비스 브로커)의 검증 절차를 거치도록 하는 방법이 제안 된 것이다. Song et al.(2006)의 연구에서는 전산자원을 소비자에게 서비스 형태로 공급하는 브로커가 전산자원 공급자의 신뢰도를 객관적 지표 및 기존 거래 실적을 바탕으로 점수를 매겨 관리한다. 소비자는 작업을 요청할 때 완료시간과 예산뿐 아니라 보안 레벨까지 지정할 수 있다. 서비스 브로커는 전산자원 공급자의 공급 기록을 바탕으로 소비자가 지정한 보안 레벨에 맞는 적절한 전산자원 공급자를 선택하여 고객의 작업을 수행한다. 이 방법은 전산자원 공급자의 행동양식을 관찰하므로 미래의 공급을 예측하는데 있어 액세스그리드 방법보다 일관적인 값을 기대할 수 있다. 그러나 전산자원이 시장기반에서 거래될 수 있는 상품이 되기 위해서는 각 전산자원 수준의 변동성이 측정되어야만 한다(Kenyon and Cheliotis, 2004).

본 논문은 개방형 전산 자원의 변동성을 측정하고 고객에게 제공하는 것에서 더 나아가 서비스 브로커가 전산자원 공급 서비스를 더욱 안정적으로 운영할 수 있는 방안을 연구한다. 개방형 전산환경과 마찬가지로 투자에 대한 수익에 불확실성을 가지고 있는 주식 시장에서는 위험 관리의 방법으로 투자 포트폴리오를 구성한다. 포트폴리오는 고수익 고위험 주식과 저수익 저위험 주식을 혼합하여 동일한 수익률이면 위험을 낮게, 동일한 위험 수준에서는 수익률을 높게 하는 투자를 할 수 있게 만들어 준다(Markowitz, 1952). 마찬가지로 개방형 전산환경에서 네트워크를 통해 공급되는 전산자원의 성능도 평균과 분산을 구할 수 있다면 전산자원 공급의 위험을 최소화하는 최적 포트폴리오를 구성할 수 있다. 본 논문에서는 계산의 편리함을 위해 전통적으로 포트폴리오를 최적화하는데 사용되는 마코위츠(Markowitz)의 모형 대신 MAD 모형을 이용한다(Konno and Yamazaki, 1991). MAD 모형은 Konno and Yamazaki(1991)가 처음 제안하였다. 기존의 마코위츠의 모형은 분산을 위험을 측정하는 측도로서 사용함으로써 최적 포트폴리오 문제를 이차식의 최적화 모형(Quadratic Programming)으로 표현하는데, 이 모형은 계산상의 어려움이 있다. MAD 방법은 이를 해결하기 위해 위험의 측도로 평균절대편차(MAD)를 사용함으로써 모형을 선형계획 모형으로 표현하게 되어 대규모의 최적 포트폴리오를 효과적으로 풀 수 있는 길을 마련하였다. 이후 모형의 수식에서 절대치를 보다 적절한 형태로 재구성하는 방법이 제안되었다(Feinstein and Thapa, 1993). 본 논문에서는 Feinstein and Thaph(1993)의 개량된 MAD 모형을 적용하여 고객의 작업 요청에 대해 최소한의 위험 수준을 갖는 전산 자원 포트폴리오를 공급하여 공급의 안정성을 보장하는 서비스 브로커 모델을 구현하였다.

3. 변동성을 고려한 최적화 모델

개방형 전산환경에서 전산자원의 소비자는 수행하고자 하는 작업의 필요량만큼 네트워크를 통해 서비스 브로커에게 자원을 요청한다. 작업량은 CPU 사이클 수 등으로 표시되며 이때 소비자는 예산과 완료 시간을 함께 제약조건으로 명시한다. 서비스 브로커는 이 정보를 바탕으로 주어진 예산하에서 정해진 완료 시간까지 고객의 작업을 완료할 수 있도록 미리 공급계약을 맺은 전산자원들 중에서 적절한 전산자원을 선택하여 고객의 작업을 할당한다. 서비스 브로커와 전산자원 공급자 간의 공급계약은 각 전산자원의 성능, 엑세스 가격(access price)과 SLA을 기준으로 이루어진다. 이때 성능은 각 전산자원의 물리적 용량으로, 시스템

정보 등을 통해 읽어올 수 있으며, 장비의 최대 성능을 의미한다. 그리고 엑세스 가격은 단위 시간 동안 해당 전산자원을 사용할 수 있는 권리를 사는 가격이고(Buyya et al., 2001), SLA는 전산자원의 가용성, 최소 성능 수준 등의 계약 조건을 명시한 것이다. 서비스 브로커는 매 기 해당 전산자원이 실제로 실현한 성능을 모니터링하고, SLA 위반 여부를 감시한다. <그림 1>은 이상에서 설명한 개방형 전산환경의 서비스 프로세스를 요약적으로 보여주고 있다.

서비스 브로커의 이윤을 모형화하기 위해, 서비스 브로커가 고객의 요청을 주어진 예산과 시간 내에 성공적으로 완료하면 각 거래에 대해 일정 정도의 수수료를 받고, 작업수행에 있어 주어진 조건을 만족하지 못하면 위약금을 지불한다고 가정하자. 그리고 모형의 단순화를 위해 모든 고객 거래에 대해 수수료는 p로 동일하다는 전제하에 상황을 분석한다. 위약금은 수수료의 η배라고 가정하고, 일정 기간 동안 서비스 브로커가 중개한 고객 거래의 전체 수가 Q, 이 중에서 성공적으로 완료한 고객 거래의 비율을 q라고 했을 때, 이 기간 동안 서비스 브로커의 이윤 극대화 식은 식 (1)과 같다. 따라서 $q=1$일 때 서비스 브로커의 이윤은 최대화 된다.

$$Max\ pqQ-\eta p(1-q)\ Q \qquad (1)$$

전산자원 성능의 변동성으로 인해 발생하는 작업 실패 비율 $(1-q)$를 최소한으로 줄이기 위해, 이 논문에서 제안하는 바는 서비스 브로커가 고객의 요청이 왔을 때 위험을 최소화하는 전산자원 포트폴리오를 구성하여 고객에게 제공하는 것이다.

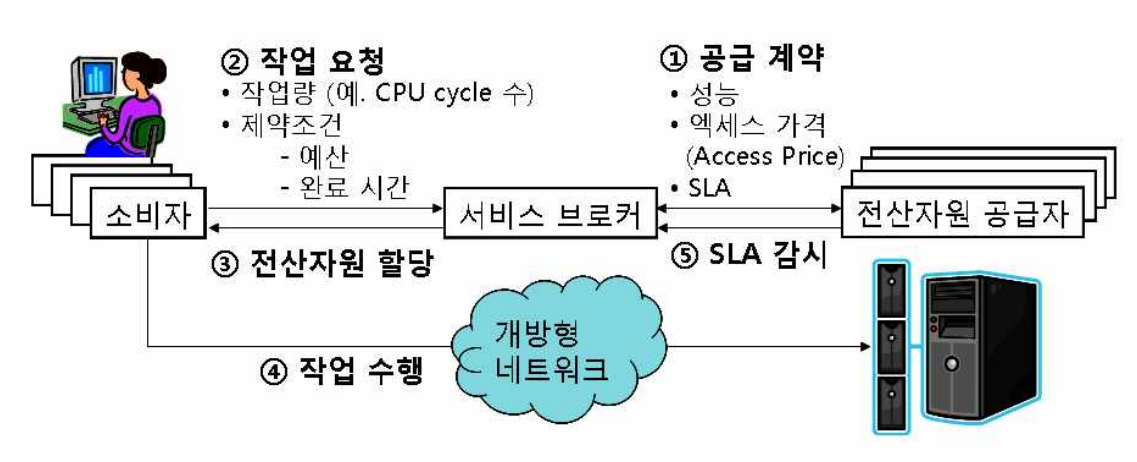
전체 N개의 전산자원 중에서 위험을 최소화하는 포트폴리오의 구성은 다음 모형에 의해 얻어진다.

$$Min\ \Sigma_{h=1}^{H} \mid \Sigma_{j=1}^{N}\ (r_j^h - r_j)\ x_j \mid \qquad (2)$$

$$s.t. \quad \Sigma_{j=1}^{N}(r_j x_j) \geq M_0/T_{max} \qquad (3)$$

$$\Sigma_{j=1}^{N}(c_j x_j) \leq C_{max}/T_{max} \qquad (4)$$

$$0 \leq x_j \leq a_j,\ j=1,\ \cdots,\ N \qquad (5)$$



<그림 1> 개방형 전산환경의 서비스 프로세스

여기서 H는 서비스 브로커가 포트폴리오를 구성하기 위해 개별 전산자원의 변동성을 관측하여 저장해 둔 히스토리 데이터의 전체 개수이며, h는 저장된 각 데이터의 색인을 표시한다. 따라서 H의 값에 따라 서비스 브로커의 전산자원 모니터링의 정도를 표시할 수 있다. N은 브로커가 동시에 가용할 수 있는 전체 전산자원의 수이며, j는 각 전산자원의 색인이다. 위 모델에서 서비스 브로커는 개방형 전산자원을 규격화된 단위에 따라 상품화하여 고객에게 판매한다. 따라서 N은 물리적 자원의 개수를 의미하는 것이 아니라, 고객의 작업 한 단위를 수행하도록 서비스 브로커가 나누어 놓은 가상 머신(virtual machine)의 수이다. r_j^h는 j의 변동성을 측정하기 위해 과거 실현되었던 전산자원 j의 실제 성능을 관측한 값으로, 히스토리 데이터 색인 h에 저장되어 있는 값이다. r_j는 전산자원, 즉 한 단위의 가상 머신 j의 평균 성능으로($\Sigma_{h=1}^{H} r_j^h$)/H이며, 실제 단위는 시간당 CPU 사이클 수이나, 이 논문에서는 규격화된 단위 가상 머신의 성능에 대한 상대적 비율로 표현된다. 브로커는 완료 시간과 가격에 대한 다양한 고객의 요청에 유연하게 대응할 수 있도록 r_j를 다양하게 구비할 수 있다. x_j는 서비스 브로커가 이 최적화 식을 통하여 도출하고자 하는 값으로, 고객의 작업을 수행하기 위해 사용되는 개별 전산자원 j의 양이다. 본 모델의 개방형 전산환경은 분산 컴퓨팅(distributed computing)에서 일반적으로 사용하는 시분할(time slicing) 작업 스케쥴링 방식을 채택하고 있다고 가정하므로, 본 논문에서 x_j는 단위시간 내에서 사용된 시간 비율로 표시된다. T_{max}와 C_{max}는 각각 고객이 서비스 브로커에게 작업을 요청할 때 계약조건으로 주는 완료 시간과 예산이고 이때 규격화된 작업 단위로 계산된 고객의 총 작업량(필요한 총 CPU 사이클 수)을 M_0로 나타낸다. c_j는 이 전산자원에 대한 단위 시간당 엑세스 가격이고, a_j는 이번 단위 시간 내에서 서비스 브로커가 사용할 수 있는 전산자원 j의 최대 사용 가능 시간 비율(availability)을 의미한다.

위의 모델에서 목적식 (2)는 포트폴리오에 소속된 전산자원의 평균 값 r_j로부터 개별 성능 r_j^h의 변동의 절대 값을 합산하여 이를 최소화하는 전산자원 조합을 구하려고 하는 것임을 알 수 있다. 이는 전통적으로 투자자산의 최적 포트폴리오을 구하기 위한 마코위츠의 모형에서 변동성의 측도로 분산을 이용한데 반해 계산적인 어려움을 극복하기 위해 평균 절대편차(Mean Absolute Deviation)를 이용한 MAD 모델의 기법을 보여준다. 식 (3)은 포트폴리오의 성능에 대한 제약식이다. 식에서 구성된 포트폴리오의 성능은 각 전산자원 성능의 합 $\Sigma_{j=1}^{N}(r_j x_j)$로 볼 수 있다. 이 값은 적어도, M_0를 T_{max}내에 완료하기 위해 이번 단위 시간에 필요한 최소 포트폴리오 성능인 M_0/T_{max}만큼은 되어야 한다. 식 (4)는 고객의 예산에 관한 제약식으로, 단위시간당 사용한 총 전산자원에 대해 지불한 값 $\Sigma_{j=1}^{N}(c_j x_j)$이 고객의 예산제약조건인 C_{max}/T_{max}보다 작아야 한다. 마지막으로 식 (6)은 포트폴리오에 속한 전산자원 j의 사용량은 0보다 크며 주어진 최대 사용 가능한 양을 넘을 수 없다는 조건이다.

위 모델은 목적함수에 절대값을 포함하고 있어 이를 풀 수 없다. 따라서 위 모델을 풀기 위해서는 다음의 선형계획 문제로의 변환이 필요하다. Feinstein and Thaph(1993)가 제안한 방법에 따라 여기서는 음이 아닌 두 변수 v_h와 w_h를 이용하여 목적함수에서 절대값을 없애주는 방법을 이용하였다.

$$Min\ \Sigma_{h=1}^{H}(v_h+w_h), \quad (6)$$

$$s.t.\quad v_h-w_h-\Sigma_{j=1}^{N}(r_j^h-r_j)x_j=0,\ h=1,\ \cdots,\ H \quad (7)$$

$$\Sigma_{j=1}^{N}(r_j x_j) \geq M_0/T_{max} \quad (8)$$

$$\Sigma_{j=1}^{N}(c_j x_j) \le C_{max}/T_{max} \quad (9)$$

$$0 \le x_j \le a_j,\ j=1,\ \cdots,\ N \quad (10)$$

$$v_h \ge 0,\ w_h \ge 0,\ h=1,\ \cdots,\ H \quad (11)$$

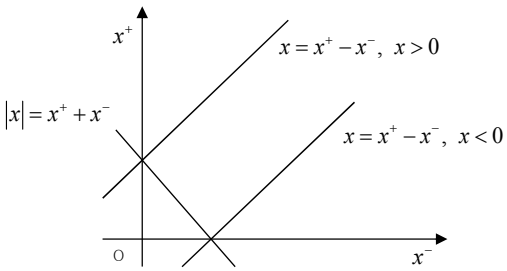
보는 바와 같이 식 (2)에서의 절대값은 식 (6)에서 v_h+w_h의 각 h에 대해 두 개의 변수의 합으로 표현되었다. 그리고 이 두 변수를 식 (6)과 보완적으로 설명하기 위해 제약식 (7)이 추가되었고 식 (11)에서 두 변수가 비부수 조건식을 가짐을 표현하였다. 즉 식 (2)에서 절대값 안에 있는 식은 두 변수의 차이로 표현되고 그 절대치는 합으로 표현되는 것이다. 그리고 h에 대한 절대치의 합은 식 (2)에서와 마찬가지로 구성된 전산 자원 포트폴리오의 평균 절대편차로서 포트폴리오의 위험도를 측정한다. 이와 같은 변환은 다음의 명제를 통해 이루어진다.

정리

$$\begin{aligned} min \quad & x^+ + x^- \\ \text{s.t} \quad & x^+ - x^- = x \\ & x \in X \\ & x^+,\ x^- \ge 0 \end{aligned}$$

은 $min\ \{|x| : x \in X\}$와 동치이다.

증명 : 우선 위의 첫번 최적화 문제에서 다음의 사실을 확인할 수 있다.



$x > 0$ 이면 $x^+ > 0,\ x^- = 0$

$x = 0$ 이면 $x^+ = 0,\ x^- = 0$

$x < 0$ 이면 $x^+ = 0,\ x^- = 0$

왜냐하면 그래프에 보는 바와 같이 임의의 x에 대해 목적함수식을 최소화하는 방법은 $x^+ + x^-$값이 x^+축이나 x^-축에서 만날 때이고 이 곳에서 앞의 사실들이 실현된다. 또한 이는 $x^+ = max\{x, 0\}$과 $x^- = max\{-x, 0\}$라는 사실과 동치이다. 따라서 이 결과로부터 우리는 $x = x^+ - x^-$이고 $|x| = x^+ + x^-$라는 사실을 얻어낼 수 있게 된다. 첫 문제의 해(x^+, x^-, x)는 최소의 $|x|$값을 갖는 해로서 두 번째 문제의 조건 $x \in X$를 만족시킨다. 마찬가지로 두 번째 문제의 해 x는 $x^+ = max\{x, 0\}$과 $x^- = max\{-x, 0\}$로 정의할 때 첫째 문제의 조건을 만족시킨다. 그러므로 두 문제는 동치이다. W

따라서 증명에서처럼 각 v_h와 w_h는 $\Sigma_{j=1}^{N}(r_j^h - r_j)x_j$의 양의 부분과 음의 부분의 크기를 표시하는 변수라고 볼 수 있다. 또한 최적화에 의해 이 두 변수는 하나가 양의 값을 가지면 다른 하나는 0의 값이 되는 성질을 갖게 됨으로써 두 변수의 합을 통해 절대값을 표현하게 된다. 이 과정을 통해 식 (2)~식 (5)의 모형은 식 (6)~식 (11)에 나타나는 것처럼 선형계획 모형으로 표현이 가능하다.

4. 성능 분석 방법

제 3장에서 제시된 전산자원 공급 모델의 성능을 분석하기 위해 본 논문에서는 가상의 개방형 전산환경을 시뮬레이션 하였다. 변동성을 감안하지 않은 단순 최적화 방법과 비교하여 동일 조건에서 각 서비스 브로커가 얻은 이윤을 바탕으로 본 논문에서 제시한 MAD 포트폴리오 최적화 모델의 우수성을 보인다. 시뮬레이션 프로그래밍에

는 자바(JAVA)언어를 사용하였고, 최적화 문제를 푸는 솔버로는 ILOG CPLEX 11.0을 이용하였다. 대규모 시스템에 적용되는 알고리즘에 대한 모형을 설계하는 경우, 모형의 적절성을 검증하기 위하여 이를 실제 시스템을 구축하는 것이 가능하지 않는 경우가 많다. 이러한 경우 제안한 모형의 성능을 확인하는 방법으로 시뮬레이션 기법을 이용하는 것이 효과적이다. 강재호 외(2005)는 자동화 컨테이너 터미널에서 운영되는 무인유도차량의 효율적 주행경로 알고리즘을 제안하면서 시뮬레이션으로 그 효용성을 검증하였다. 본 논문의 모델 환경과 같은 유비쿼터스 네트워크 환경에서 사용자와 네트워크 상에서 제공되는 자원 간의 다양한 상호작용을 분석하고 새로운 모델을 설계하는 경우에도 시뮬레이션 기법을 사용한 여러 사례를 찾아볼 수 있다(박아름과 이경전, 2008; 권오병 외, 2007; 권오병 외, 2005). 이러한 논문들을 살펴봤을 때, 시뮬레이션 방법은 결과에 영향을 줄 것으로 예상되는 중요한 변수들을 통제하고 값을 변경시켜가면서 각각의 변수가 가지는 영향을 파악하고, 다양한 상황 별로 도출된 결과들을 비교해 봄으로써 가장 적절한 대안을 찾아내는 용도로 유용하게 사용될 수 있음을 확인 할 수 있다.

4.1 시뮬레이션 가정

먼저, 서비스 브로커는 개방형 전산 환경 내에서 $N=100$개의 전산자원을 보유하고 있으며 각 전산자원은 고객의 작업요청 한 단위를 처리하는데 사용된다. 서비스 브로커가 비즈니스를 수행하는 개방형 전산 환경은 그 속에 속해있는 전산자원들이 일정한 성능을 보이는 것이 아니라 불확실한 변동성을 가지며 이 불확실성의 최대값 u는 외생변수로 서비스 브로커에게 주어진다.

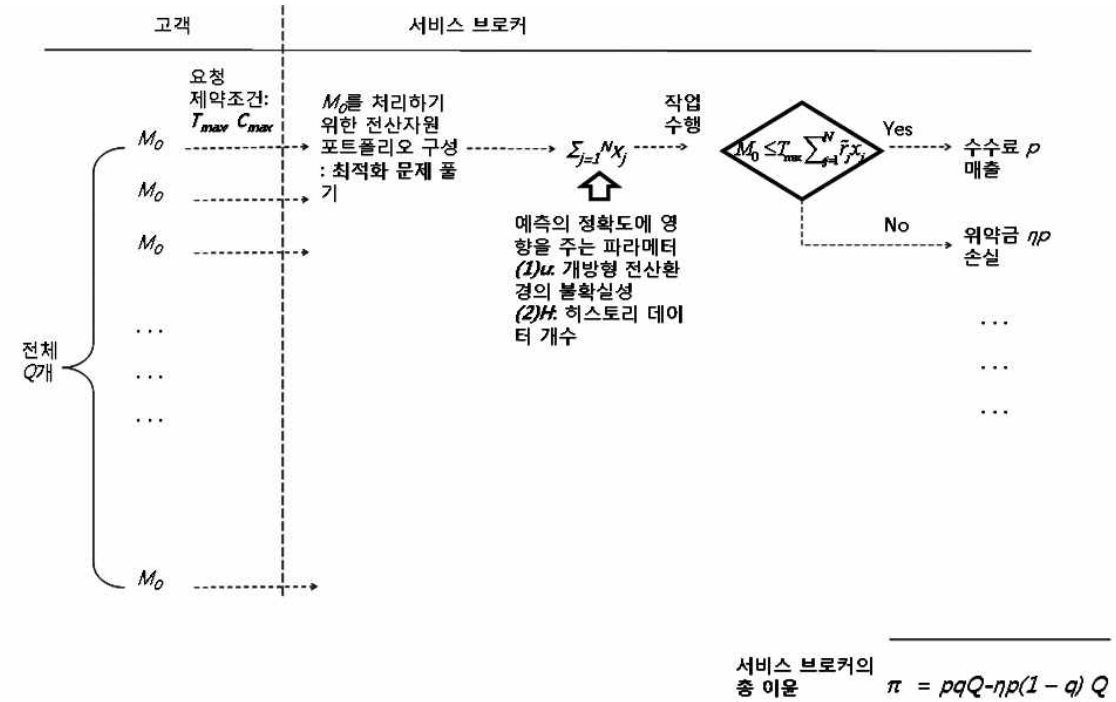
서비스 브로커가 보유한 모든 전산자원은 상호간 확장성이 보장된다고 가정한다(Boden, 2004; Minoli, 2004). 따라서 브로커는 모든 전산자원을 활용하여 자유로운 포트폴리오 구성이 가능하다. 전산 자원의 성능, 즉 고객의 한 단위 작업 요청을 수행하는 한 단위의 가상 머신의 최대 성능 R_j는 고객의 한 단위 작업을 1로 표준화 했을 때 1과 2 사이의 값을 가지도록 설정하였다. 또한 전체 N개 전산자원의 평균 성능 분포는 지수함수 분포를 가진다고 가정하였다. 즉, 빠른 완료시간을 요구하는 고객들을 지원하기 위해 상대적으로 높은 성능을 가진 전산자원도 보유하고 있으나, 성능이 높아질수록 그 수는 상대적으로 적어진다. 전산 자원의 가격 c_j는 성능 R_j에 따라 결정된다. 본 논문에서는 c_j와 R_j가 비례한다고 가정하였다.

각 전산자원의 위험 수준 σ_j는 같은 성능(CPU 수치, 메모리 용량 등)이라도 공급자와 시스템 종류(예 : 운영체제가 윈도우 XP, 윈도우 Vista, Linux 등으로 달라질 경우)에 따라 달라질 수 있다(Chen et al., 2005; Song et al., 2006). 따라서 본 논문에서는 전산자원의 성능 R_j과 위험 수준 σ_j는 서로 독립변수로 가정한다. 개별 전산자원의 위험 수준 σ_j는 외생변수인 현재 개방형 전산환경 전체의 최대 위험수준 u안에서 임의로 선택된다. 한편 R_j은 전산자원의 물리적 최대 용량을 나타내는 고정된 값으로 실제 실현되는 성능 r_j^h는 R_j보다 적게 될 것으로 생각할 수 있다. 본 논문에서는 r_j^h값은 시뮬레이션의 매 기마다 R_j에서 $R_j(1-\sigma_j)$ 사이의 값 중 임의의 값으로 실현된다고 가정한다. 서비스 브로커는 r_j^h값을 H개 관측하여 각 전산자원의 평균 성능과 변동성 측도를 구하고, 이를 이용하여 고객의 요청이 들어왔을 때 가장 안정적인 전산자원 포트폴리오를 구성할 수 있다. 이 논문에서는 변동성 σ_j의 측도 로서 MAD를 이용한다.

포트폴리오 최적화 문제를 풀어서 나온 (x_1, …, x_N)을 실제 고객의 작업 처리에 적용하였을 때, 전산자원 j의 실제 실현된 성능을 $\tilde{r}_j$라고 하면 $M_o/\sum_{j=1}^{N}\tilde{r}_j x_j$가 고객에게 제공하는 실제 작업 처리 시간이 되고 $T_{\max}\sum_{j=1}^{N}c_j x_j$가 고객이 실제로 전산자원 이용을 위해 사용한 비용이 된다. 따라서 $M_o/\sum_{j=1}^{N}\tilde{r}_j x_j$가 $T_{\max}$보다 작거나 같은 경우 고객의 작업에 필요한 전산자원을 성공적으로 공급한 것이 되고 그렇지 못한 경우 전산자원의 성능 변동성으로 인해 고객이 필요로 하는 전산자원 공급에 차질이 생긴 경우라고 볼 수 있다.

4.2 시뮬레이션 시나리오

개방형 전산환경을 이용하는 고객은 서비스 공급자와 SLA 계약을 통해 서비스의 품질(Quality of Services)을 보장받을 수 있을 것으로 기대한다(Boden, 2004; Kenyon and Cheliotis, 2004). Menasce and Casalicchio(2004)의 모델에서는 서비스 브로커가 고객과 맺은 SLA를 만족시킬 수 있도록 하는 방안으로 고객의 예산 내에서 가장 빨리 작업을 완료하는 전산자원 조합을 구하는 최적화 문제를 풀도록 하고 있다. 그러나 이러한 방법에서는 종종 전산자원 공급자 측의 요인으로 인해 고객과 맺은 SLA를 지키지 못하는 경우가 발생한다(Battre et al., 2007). 본 논문에서는 공급되는 전산자원의 성능이 불확실성을 가지는 개방형 전산환경에서 Menasce and Casalicchio(2004)의 단순 최적화 방법에 비하여 본 논문에서 제시하는 변동성을 감안한 최적화 방법이 얼마나 전산 자원



<그림 2> 시뮬레이션 흐름도

공급의 안정성을 개선시킬 수 있는지 시뮬레이션을 통해 비교한다.

성능 분석 요소는 단순히 총 서비스 공급량 대비 성공률이 아니라 식 (1)에 근거하여 서비스 브로커의 이윤이 된다. 계산을 간편하게 하기 위해 고객이 서비스 공급에 대한 수수료로 서비스 브로커에게 지불하는 비용은 $p=1$, 서비스 브로커가 서비스 공급에 실패했을 때 지불하는 위약금 비율은 $\eta=1$로 고정하고, 서비스 브로커의 이윤 $\pi = qQ-(1-q)Q=(2q-1)Q$를 계산한다. <그림 2>는 전체적인 시뮬레이션의 흐름도를 보여준다.

시뮬레이션의 인자는 2가지로, 주어진 개방형 전산환경의 최대 불확실성 u와 서비스 브로커가 변동성을 계산하기 위해 관측하는 개별 전산자원 성능의 히스토리 데이터 개수 H이다. u에 대해서는 비교적 불확실성이 낮은 경우와 높은 경우 2가지 경우를 시뮬레이션 하였으며 각각 0.1과 0.5값을 대입하였다. 또한 변동성을 감안한 최적화 방법에서는 H를 몇 개로 두느냐에 따라 각 전산자원의 평균 성능 및 변동성을 계산하는 정확도가 달라지게 되므로 H를 1부터 10까지 변화시켜가면서 그 영향을 살펴보았다. 정확도가 높아지면, 브로커의 이윤에 양의 영향을 미친다. 모든 시나리오에서 전산자원 성능 R_j의 분포는 동일하도록 난수 발생을 조정하였으며, σ_j의 분포는 u에 따라 고정되도록 하였다.

시뮬레이션의 매 기마다 서비스 브로커에게 작업을 요청하는 고객의 수는 전산자원의 최대 개수와 같은 100으로 정하고, 각 고객이 요청하는 작업의 크기 M_0는 1로 표준화하였다. 작업요청과 관련된 고객의 제약조건(T_{max}, C_{max})에 대해서는 이들 제약조건이 실제로 포트폴리오 구성에 있어 직접적인 제한요소로 작용하는 경우와 그렇지 않은 경우 최적화 성능이 달라질 수 있을 것으로 예상할 수 있다. 비교대상인 단순최적화 기법이 고객의 예산만을 제약조건으로 하고 있기 때문에 시뮬레이션에서도 고객의 예산 제약이 포트폴리오 구성에 있어 직접적인 제한 요소로 작용하는 경우($T_{max}=1$, $C_{max}=1$)와 그렇지 않은 경우($T_{max}=1$, $C_{max}=2$) 를 나누어 결과를 살펴보았다. 마지막으로, 전산자원의 변동성을 계속 모니터링하고 있는 서비스 브로커의 경우, 전산자원 공급자와 가격 협상을 통해 전산자원 별로 변동성 정도에 따라 엑세스 가격 c_j를 할인 받을 수 있는 가능성이 있다. c_j는 브로커의 이윤과 직접적 연관은 없지만, c_j가 낮아짐으로써 더 많은 고객을 끌어들일 수 있어 Q를 증가시킬 수 있다. 따라서 (1)의 식에 의해 전산자원 가격 협상은 브로커의 이윤에 긍정적인 효과가 있을 것으로 기대할 수 있다.

이상의 변수들을 모두 조합하여 본 논문에서는 총 64개의 시나리오를 구성하여 시뮬레이션을 수행하였다(<표 1>). 각 시나리오에 대해서는 50회 시행의 평균값을 서비스 브로커의 최종 성능으로 취하였다.

5. 성능 분석 결과

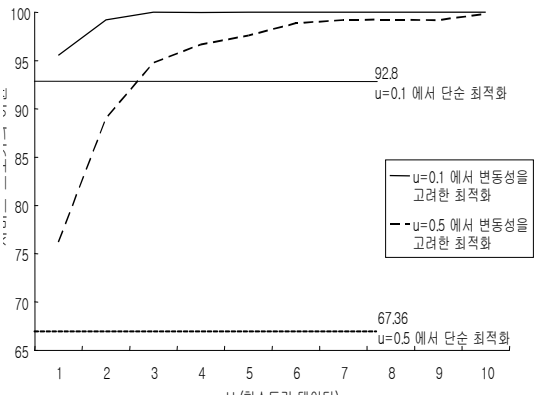
먼저 고객의 예산이 제한요소가 되지 않는 경우에 대한 22개 시나리오를 시뮬레이션한 결과가 <그림 3>에 나타나 있다. 단순 최적화를 하는 서비스 브로커의 이윤은 $u=0.1$에서 92.8, $u=0.5$에서 67.36으로 개방형 전산환경의 변동성이 커질수록 이윤에 심각한 손실이 발생하는 것을 알 수 있다. 변동성을 고려한 최적화를 적용했을 경우 H에 따라 결과가 달라지는데, $u=0.1$에서는 $H=3$에서부터($H=4$일 때 99.96 제외) 이윤의 손실이 하나도 없이 서비스 공급이 가능하고, $u=0.5$에서는 $H=1$에서는 이윤이 76.2에 불과하나 $H=7$부터는

<표 1> 시뮬레이션 시나리오

예산 제한	u	서비스 브로커의 모델	H	c_j 할인 여부	시나리오 수
있음 ($T_{max}=1$, $C_{max}=1$)	$u=0.1$	단순 최적화 모델	해당 없음	할인하지 않음	1
		변동성을 고려한 최적화 모델	1부터 10까지 10단계	할인하지 않음	10
				할인 적용	10
	$u=0.5$	단순 최적화 모델	해당 없음	할인하지 않음	1
		변동성을 고려한 최적화 모델	1부터 10까지 10단계	할인하지 않음	10
				할인 적용	10
없음 ($T_{max}=1$, $C_{max}=2$)	$u=0.1$	단순 최적화 모델	해당 없음	생략(할인 여부가 브로커의 이윤에 미치는 영향이 거의 관찰되지 않음)	1
		변동성을 고려한 최적화 모델	1부터 10까지 10단계		10
	$u=0.5$	단순 최적화 모델	해당 없음		1
		변동성을 고려한 최적화 모델	1부터 10까지 10단계		10
총 합					64

이윤이 99를 상회한다. H가 커질수록 각 전산자원의 성능에 대한 예측오차를 줄여 고객의 작업을 수행하는데 있어 성공률을 높일 수 있기 때문으로 보인다. u가 클수록 H에 더 크게 영향을 받는다는 것은, 개방형 전산환경의 변동성이 커질수록 개별 전산자원에 대한 성능 모니터링을 더욱 강화하여 변동성을 정밀하게 계산할 필요가 있다는 의미로 해석할 수 있다.

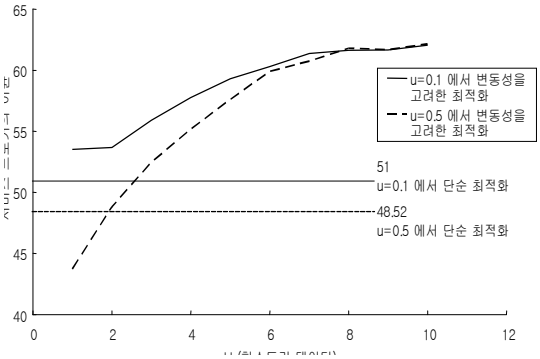
한편, H의 증가에 따른 브로커의 이윤 변화 형태를 살펴보면, 이윤은 H가 증가할수록 증가하나, 한계이윤은 H가 증가할수록 감소하는 수확체감의 형태를 보여준다. H를 증가시키는데 추가적인 비용이 들지 않는다면 이윤이 가장 높은 지점에서 H를 결정하는 것이 최적이다. 그러나, H는 개방형 전산환경에 흩어져있는 다양한 전산자원의 성능을 모니터링하고 감시하는 것과 관련된 변수로, 개방형 전산환경의 범주가 커지고, 전산자원의 수가 늘어날수록 H증가에 따른 비용부담이 커질 것으로 예상할 수 있다. 이윤극대화 문제의 균형조건에 의하면 한계수익은 한계비용과 일치한다. 그러므로 서비스 브로커는 H의 증가에 따라 증가된 한계이윤이 H를 증가시키는 한계비용보다 클 때까지만 H를 증가시킬 것이다(Cheliotis et al., 2003). 예를 들어 H를 증가시키는 한계비용이 1이라고 가



<그림 3> 고객의 예산 제약이 제한 요소로 작용하지 않는 경우 서비스 브로커의 이윤

정하면 가장 효율적인 수준의 H는 u =0.1에서 2(이윤 99.2)이고 u =0.5에서는 6(이윤 98.88)이다.

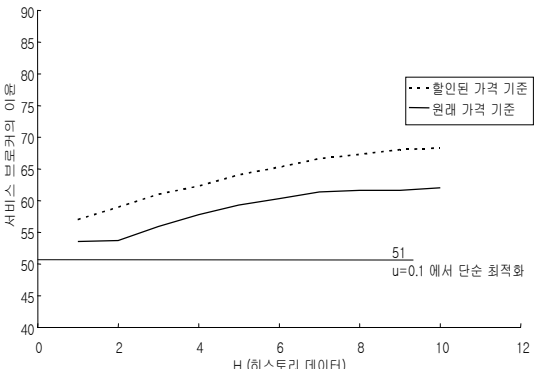
<그림 4>에서는 고객의 예산이 직접적인 제한 요소로 작용할 경우 서비스 브로커의 이윤이 어떻게 변화하는지 보여준다. 이 경우에도 서비스 브로커가 변동성을 고려한 최적화 기법을 채택했을 경우 단순 최적화보다 더 높은 이윤을 얻을 수 있다(단, u =0.5에서는 H =1 제외). H의 증가에 따른 서비스 브로커의 이윤을 살펴보면 <그림 3>에서와 마찬가지로 u에 상관없이 H가 증가할수록 같은 수준으로 수렴하고 있음을 알 수 있다. 즉, u가 증가하더라도 각 전산자원 별로 정확히 측정된 변동성을 바탕으로 위험을 최소화하는 전산자원 포트폴리오를 구성하면 주어진 고객의 요청에 대해 최대 이윤을 달성하는 것이 가능하다는 것을 보여준다. 이번에도 H를 증가시키는 한계비용을 1이라고 가정하고 최적 H를 구해보면 u =0.1에서 7(이윤 61.38)이고 u =0.5에서는 8(이윤 61.8)이 된다.



<그림 4> 고객의 예산 제약이 직접적인 제한 요소로 작용하는 경우 서비스 브로커의 이윤

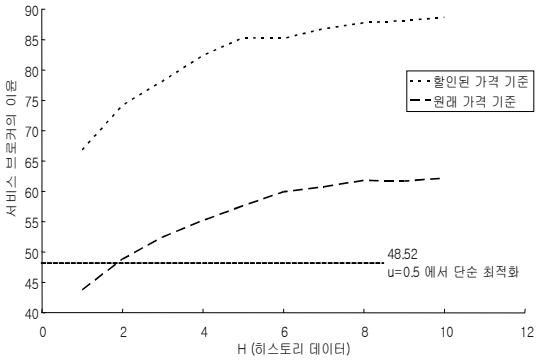
한편 고객이 예산 제약에 묶여 있는 경우, 서비스 브로커가 H에 제약이 없다고 가정한다면 브로커의 이윤은 총 거래량 Q에 따라 제약을 받는다. 즉, 서비스 브로커에게 전산자원 서비스를 요청하는 고객은 변함없이 100명이지만 예산 제약 때문에 브로커가 서비스 공급을 결정한 고객의 수 Q는 그보다 훨씬 적은 숫자가 된다. 서비스 공급을 결정한 Q명의 고객에 대해서는 u에 상관없이 높은 H값을 통해 품질을 보장할 수 있게 되었으므로, 이제 이윤을 늘리기 위한 서비스 브로커의 과제는 Q를 늘리는 것이 된다. 고객들이 예산 제약에 묶여 있으므로 Q를 늘리려면 전산자원의 가격을 낮출 수 있어야 한다. 따라서 서비스 브로커는 전산자원의 엑세스 가격을 측정된 변동성에 비례하여 할인하도록 전산자원 공급자와 협상해 볼 수 있다. 본 시뮬레이션에서는 간단하게 각 전산자원의 평균 변동성 $\sigma_j/2$을 가격 할인율로 적용하였다.

<그림 5>와 <그림 6>을 통해 이렇게 가격 할인을 받게 되었을 때 서비스 브로커의 이윤이 어떻게 변화할 수 있는지 알아볼 수 있다. <그림 5>는 u =0.1에서 전산자원 가격 할인을 통한 서비스 브로커의 이윤 증가를 나타낸 것이고, <그림 6>은 u =0.5에서의 변화를 표시한 것이다. 전산자원의 가격 c_j가 원래는 고정된 값인 R_j에 비례하여 결정



<그림 5> 고객이 예산 제약에 묶여 있는 경우 가격 할인을 도입했을 때 서비스 브로커의 이윤 변화(u = 0.1)

되었으나 이제는 평균 성능인 r_j에 비례하여 결정되기 때문에 u가 증가할수록 c_j가 낮아지게 된다. 따라서 u가 더 클수록(<그림 6>) 더 많은 고객이 예산 제약에 제한을 받지 않게 되어 Q가 더욱 증가하게 되고, 결과적으로 서비스 브로커의 이윤도 더 큰 폭으로 증가한다.



<그림 6> 고객이 예산 제약에 묶여 있는 경우 가격 할인을 도입했을 때 서비스 브로커의 이윤 변화(u = 0.5)

6. 시사점 및 향후 연구

본 연구는 개방형 전산환경을 기업 및 상업용으로 확대하고자 할 때 걸림돌이 되고 있는 성능의 불확실성을 해소하는 방안으로 MAD 포트폴리오 최적화 기법을 소개하였다. 개방형 전산환경에서 제공되는 전산 서비스의 성능이 불확실성을 가지는 것은 서비스를 구성하고 있는 개방형 전산자원들이 변동성을 가지기 때문이다. 지금까지는 단순히 주어진 예산과 시간 내에 서비스를 수행할 수 있도록 전산자원을 선택하여 공급했다면, 본 논문에서는 각 개방형 전산자원의 변동성을 모니터링하고, 이 데이터를 바탕으로 위험을 최소화하는 전산자원 포트폴리오를 구성함으로써 서비스의 안정성을 향상시킬 수 있었다.

또한 제시한 모델의 성능을 서비스 브로커의 이윤 관점에서 평가하여 첫째, 고객의 요청을 성공적으로 수행하지 못했을 경우 부담하게 되는 위약금까지 고려함으로써 성공률을 과대평가하지 않도록 하였고, 둘째, 서비스 브로커가 이윤을 최대화하기 위해서는 성공률 뿐 아니라 성사된 거래의 총 수 Q도 중요하게 작용하는 것을 확인하였고, 셋째, 개별 전산자원의 변동성을 정확히 계산하는 정도를 나타내는 H가 증가할수록 이윤 또한 증가하지만 H를 증가시키는데 따를 비용 증가분을 감안하여 최적 H는 한계비용이 한계이윤을 초과하지 않는 지점에서 결정되어야 함을 설명하였다.

마지막으로 본 논문에서는 개방형 전산자원의 엑세스 가격이 물리적 성능에 의해서가 아니라 변동성을 고려하여 평균적으로 실현되는 성능을 기준으로 책정될 때 개방형 전산환경에서 더 많은 비즈니스 기회가 창출될 수 있음을 보였다. 이것은 비단 서비스 브로커 및 예산 제약에 묶여있는 소비자뿐 아니라 예산 제약이 없는 소비자와 전산자원 공급자에게도 이익이 될 수 있다. 전산자원 공급자 입장에서는 1회 사용에 대한 단위 가격은 낮아졌지만, 총 거래량이 증가하여 장기적으로는 더 높은 수익을 기대할 수 있다. 또한 변동성에 기반한 가격 정책은 전산자원 공급자에게 품질에 대한 피드백으로 작용하여 자원 관리를 강화하도록 하는 유인으로도 작용한다(Song et al., 2006). 이렇게 될 경우 전반적으로 전산 자원의 변동성이 줄어들게 되어 예산 제약에 묶여있지 않는 소비자도 더 안정적인 서비스의 혜택을 누릴 수 있게 된다.

본 연구는 몇 가지 한계점을 지니고 있다. 첫째로 소비자의 효용 및 전산자원 공급자의 이윤과 관련해서는 아직 분석하고 있지 않다는 한계를 가지고 있다. 서비스 브로커의 이윤을 분석하는데 사

용되었던 각각의 시나리오 하에서 소비자의 효용 및 전산자원 공급자의 이윤 또한 어떻게 변하는지를 분석하여 개방형 전산 환경에 참여하는 모든 주체들의 이익을 다 같이 극대화시킬 수 있는 전략을 연구하는 것이 추가적으로 필요하다. 둘째, 모델의 적용 부분에 있어서는, 네트워크에 흩어진 전산자원을 사용함에 있어 위치 요소가 고려되지 않았다는 한계가 있다. 네트워크를 사용하는 구간이 많을 수록 고객의 작업을 전송하고 그 결과를 전송 받는 과정에서 추가 비용과 시간이 더 발생할 수 있는 소지가 있다(Stuer et al., 2007). 따라서 동일한 업무를 처리하더라도 이용하는 개방형 전산자원의 수가 많아질 수록, 처리비용이 증가하게 된다. 마지막으로 비즈니스 워크플로우를 순차적으로 따라야 하는 작업일 경우에는, 작업을 나누어 처리할 수 있는 개수가 한정되어 포트폴리오 구성에 있어 자유도가 낮아질 수 밖에 없다는 문제가 있다. 이러한 문제점에 대한 한가지 대안은, H를 포트폴리오에 포함되는 전산자원의 수를 조정하는 통제변수로 활용하는 것을 생각해 볼 수 있는데(Konno and Yamazaki, 1991; Feinstein and Thapa, 1993), 시스템 환경과 고객의 작업 특성을 고려하여 각 상황에서 적절한 자원의 분산도 (H)를 결정하는 방법에 대해서는 추후 연구에서 다루게 될 것이다.

참고문헌

강재호, 최이, 강병호, 류광렬, 김갑환, "자동화 컨테이너 터미널에서의 AGV 충돌 방지 및 교착 해결 방안", *한국지능정보시스템학회논문지*, 11권 3호(2005), 25~43.

권오병, 최근호, 최성청, "Rescorla-Wagner 모형을 활용한 다중 에이전트 웹 서비스 기반 욕구인지 상기 서비스 구축 및 성능분석", *한국지능정보시스템학회논문지*, 11권 3호(2005), 1~23.

권오병, 최석제, 박태환, "존대등분 계산법과 사례기반추론을 활용한 상황 인식형 모바일 인터페이스 시스템", *한국지능정보시스템학회논문지*, 13권 3호(2007), 141~160.

박아름, 이경전, "U-Publication 시스템과 비즈니스 모델의 설계와 분석", *지능정보연구*, 14권 3호(2008), 41~57.

Battre, D., et al., "Gaining Users' Trust by Publishing Failure Probabilities", *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*(2007).

Boden, T., "The Grid Enterprise-Structuring the Agile Business of the Future", *BT Technology Journal*, Vol.22, No.1,(2004), 107~117.

Buyya, R., et al., "A Case for Economy Grid Architecture for Service Oriented Grid Computing", *Proceedings of the 15th International Parallel and Distributed Processing Symposium*(2001).

Cheliotis, G., et al., *Grid Economics : 10 Lessons from Finance*, Joint technical report, IBM Research Zurich and GRIDS Lab, Melbourne, 2003.

Chen, P.-Y., et al., "Software Diversity for Information Security", *Workshop on the Economics of Information Security(WEIS)*(2005).

Domingues, P. et al., "Sabotage-Tolerance and Trust Management in Desktop Grid Computing", *Future Generation Computer Systems*, Vol.23, No.7(2007), 904~912.

Feinstein, C. D. and M. N. Thapa, "A Reformulation of a Mean-Absolute Deviation Portfolio Optimization Model", *Management Science*, Vol.39, No.12(1993), 1552~1553.

Foster, I., et al., "Grid Services for Distributed System Integration", *Computer*, Vol.35, No.6 (2002), 37~46.

Hwang, S. and C. Kesselman, "A Flexible Framework for Fault Tolerance in the Grid", *Journal of Grid Computing*, Vol.1, No.3(2003), 251~272.

Kenyon, C. and G. Cheliotis, "Grid Resource Commercialization : Economic Engineering and Delivery Scenarios", *Grid Resource Management : State of the Art and Future Trends* Kluwer Academic Publishers, 2004.

Kleban, S. D. and S. H. Clearwater, "Computation-at-Risk : Assessing Job Portfolio Management Risk on Clusters", *Proceedings of the 18th International Parallel and Distributed Processing Symposium*(2004).

Konno, H. and H. Yamazaki, "Mean-Absolute Deviation Portfolio Optimization Model and Its Applications to Tokyo Stock Market", *Manage. Sci.*, Vol.37, No.5(1991), 519~531.

Markowitz, H., "Portfolio Selection", *The Journal of Finance*, Vol.7, No.1(1952), 77~91.

Minoli, D., *A Networking Approach to Grid Computing*, Wiley-Interscience, 2004.

Plaszczak, P. and R. Wellner, *Grid Computing : The Savvy Manager's Guide*, Elsevier/Morgan Kaufmann, San Francisco, 2005.

Rappa, M. A., "The Utility Business Model and the Future of Computing Services", *IBM Syst. J.*, Vol.43, No.1(2004), 32~42.

Seti@Home, http://setiathome.berkeley.edu/.

Song, S., et al., "Risk-Resilient Heuristics and Genetic Algorithms for Security-Assured Grid Job Scheduling", *Computers, IEEE Transactions on*, Vol.55, No.6(2006), 703~719.

Stuer. G., et al., "A commodity market algorithm for pricing substitutable Grid resources", *Future Generation Computer Systems*, Vol. 23(2007), 688~701.

Thanos, G., et al., "Adopting the Grid for Business Purposes : The Main Objectives and the Associated Economic Issues", *Grid Economics and Business Models*, 2007.

Abstract

Minimizing the Risk of an Open Computing Environment Using the MAD Portfolio Optimization

Hak-Jin Kim* · Jihyoun Park**

The next generation IT environment is expected to be an open computing environment based on Grid computing technologies, which allow users to access to any type of computing resources through networks. The open computing environment has benefits in aspects of resource utilization, collaboration, flexibility and cost reduction. Due to the variation in performance of open computing resources, however, resource allocation simply based on users' budget and time constraints often fails to meet the Service Level Agreement(SLA). This paper proposes the Mean-Absolute Deviation(MAD) portfolio optimization approach, in which service brokers consider the uncertainty of performance of resources, and compose resource portfolios that minimize the uncertainty. In order to investigate the effect of this approach, we simulate an open computing environment with varying uncertainty levels, users' constraints, and brokers' optimization strategies. The simulation result concludes threefolds. First, the MAD portfolio optimization improves the success ratio of delivering the required performance to users. Second, the success ratio depends on the accuracy in predicting the variability of performance. Thirdly, the measured variability can also help service brokers expand their service to cost-critical users by discounting the access cost of open computing resources.

Key Words : Open Computing Environment, Grid Computing, Service Level Agreement, Uncertainty, Risk Management, Portfolio Optimization, Simulation

* School of Business, Yonsei University
** Corresponding Author, Technology Management, Economics and Policy Program, Seoul National University

저 자 소 개

김학진

연세대학교에서 경영학 학사(1987)와 석사(1989)를 취득하고 미국 University of Illinois at Urbana-Champaign에서 수학으로 석사를 받았다(1994) 이후 미국 Carnegie Mellon University의 Tepper School of Business에서 OR로 경영학 박사를 취득하였으며(2001) 현재 연세대학교 조교수로 재직 중이다. 관심 영역으로는 논리 기반하의 최적화 문제, Constraint Programming, Combinatorial Optimization, Portfolio Selection, Risk management, Product Design Problem, Semantic Web 기술의 경영학 문제에 적용, Grid Computing, 기술 경영 정책 문제들에 관심을 갖고 있다.

박지현

서울대학교에서 컴퓨터공학 학사(2002)를 취득하고, 현재 서울대학교 기술경영경제정책대학원 석박사통합과정에 재학 중이다. 현재 ITRC 통신방송 융합기술 정책연구센터 연구원으로 활동하면서, 융합환경에 따른 인터넷 산업 및 정책 발전방향에 관한 KT 프로젝트에 참여하고 있다. 그 이전에는 KT 비즈니스 그리드 모델발굴 프로젝트 및 EU의 FP6 프로젝트의 일환인 GridEcon(Grid Economics and Business Models) 프로젝트에 참여하였으며, 호주 University of Melbourne의 Grid computing and Distributed Systems Laboratory에서 교환연구원으로 활동하였다. 대학원 과정 시작 전, 한국IBM IT Service 부서에서 근무한 바 있다. 주 관심분야는 비즈니스 그리드 환경에서 자원의 위험 관리와 소비자 후생, 과금체계 등이다.