글로벌 컴퓨팅 시스템에서 그룹 기반 정확성 검사를 위한 스케줄링 기법

김홍수[O] 백맹순[†] 유헌창[‖] 황일선[‡] 황종선[†]
고려대학교 컴퓨터학과 분산시스템 연구실[†], 고려대학교 컴퓨터교육과[‖], 한국과학기술정보연구원[‡]
{hera[O], msbak[†], hwang[†]}@disys.korea.ac.kr, yuhc@comedu.korea.ac.kr[‖], his@kisti.re.kr[‡]

Scheduling Mechanism to Group-based Result Checking in Global Computing Systems

Hong-Soo Kim[O], Maeung-Soon Baik, Chong-Sun Hwang
Dept. of Computer Science & Engineering, Korea University
Hyun-Chang Yoo
Dept. of Computer Science@Education, Korea University
il-Sun Hwang
Supercomputing center, KISTI

요 약

글로벌 컴퓨팅 시스템 환경에서 자원 제공자에 의해 수행하는 연산 결과에 대한 정확성을 제공하기 위한 결과 검사 기법은 중요한 고려사항이다. 자원 제공자들은 인터넷에 연결되어 인증 없이 자율적으로 연산에 참여할 수 있기 때문에 이들에 의해 수행된 연산 결과에 대한 정확성을 보장해야만 한다. 기존 연구에서 결과에 대한 정확성을 보장하기 위해 다수 투표법과 결점 검사법을 사용하지만 정확성 검사 기법을 위한 스케줄링 기법을 사용하고 있지 않아 높은 연산 지연 시간과 부하가 발생 한다. 따라서, 본 논문에서는 결과 검사에 대한 연산 지연 시간과 부하를 줄일 수 있는 신용도 기반 그룹 구성을 통한 스케줄링 기법을 제안한다.

1. 서론

글로벌 컴퓨팅은 네트워크에 연결된 유휴 컴퓨팅 자원들을 이용해 고 처리 컴퓨팅(high throughput computing) 응용을 수행하는 컴퓨팅 패러다임이다. 인터넷 기반 고 처리 컴퓨팅 분야는 크게 그리드 컴퓨팅 시스템과 글로벌 컴퓨팅 시스템으로 구별할 수 있는데 글로벌 컴퓨팅 시스템의 구별되는 특징은 전 세계에 분산되어 인터넷에 연결된 컴퓨팅 자원의 유휴 시간을 이용하여 거대한 연산과 분산 응용을 수행하는 것이다.

그리드 컴퓨팅 시스템과 글로벌 컴퓨팅 시스템은 연산에 대한 신뢰성에 대해서 각기 다른 가정을 한다. 그리드 컴퓨팅 시스템은 비교적 신뢰할 수 있는 컴퓨팅 자원들을 가정하지만, 글로벌 컴퓨팅 시스템은 자원 제공자가 연산에 대한 간섭을 받거나 악의적으로 잘못된 결과를 도출할 수 있는 방해 행위에 노출되어 있다. 그래서, 악의적인 자원 제공자가 잘못된 연산 결과를 반환하고 잘못된 연산 결과가 전체 연산에 반영된다면 전체 연산 결과는 더 이상 신뢰할 수 없게 된다. 실제로 SETI@home에서 악의적인 자원 제공자들이 원래 코드를 변형하여 잘못된 결과를 반환하였다[2]. 따라서, 글로벌 컴퓨팅 시스템에서 연산 결과에 대한 정확성 검사 기법은 중요한 고려 사항이다.

기존 연구에서 연산 결과에 대한 정확성 검사를 위해 크게 **다수 투표법**(majority voting)과 **결점 검사법**(spot checking)을 사용한다. 다수 투표법[ε]은 최소 두개 이상의 여분을 통해 수행된 결과와 비교하는 투표방식으로 과반수 이상의 결과가 같으면 채택한다. 하지만, 이 방식은 결함률이 클 경우 여분이 2k+1개로 늘어나기 때문에 최소 2개 이상의 여분이 필요해 비효율적이다. 결점 검사법[3]은 미리 신뢰성 있는 노드를 선정한 후 실제 수행될 노드에서 수행된 결과와 비교하여 같은 결과이면 채택하는 방식이다. 다수 투표법보다 낮은 여분으로 비교적 효과적이지만 신용도(Credibility) 있는 검사자 그룹을 미리 선정해야하는 문제점이 있다. 기존 연구에서는 정확성 검사를 위한 스케줄링 기법을 사용하고 있지 않아 높은 연산 지연 시간과 부하가 발생한다. 따라서, 본 논문에서는 신용도 기반 그룹 구성을 통한 스케줄링 기법을 통해 전체 연산 지연 시간과 부하를 줄일 수 있는 신용도 기반 그룹 구성을 통한 스케줄링 기법을 제안한다.

본 논문은 다음과 같은 순서로 구성되어 있다. 2장에서는 글로벌 컴퓨팅 시스템 분야에서 결과에 대한 신용도를 보장하기 위한 기존 연구들을 소개한다. 3장에서는 글로벌 컴퓨팅 시스템 모델과 동작 방식에 대해서 논의한다. 4장에서는 본 논문의 제안 기법인 신용도 기반 그룹 구성을 통한 스케줄링 기법을 설명한다. 5장에서는 성능 분석을 통해 본 논문의 기법과 기존 기법을 비교 분석하고 6장에서 결론을 맺는다.

2. 관련연구

SETI@home[2]에서는 악의적인 자원 제공자들이 원래 코드를 변형하여 잘못된 결과를 도출함을 알 수 있는데 이를 해결하기 위하여 많은 시간을 들여 전체 연산을 재 수행하는데 다수 투표법을 사용함으로써 많은 여분과 연산 지연이 발생된다.

Bayanihan[3]에서는 다수 투표법과 결점 검사법을 이용해 신용도 기반 결함포용이라는 새로운 기법을 제안하였다. 그러나, 선행 처리자 우선 할당 기법(eager scheduling)을 사용하여 작업을 무조건 빨리 끝나는 자원 제공자에게 할당하고, 하나의 연산이 끝난 후 결과에 대한 신용도를 검사해 재 할당하는 방식에서 연산 결과 검사에 대한 연산 지연 시간이 발생하고 투표에 대한 부하가 발생한다.

XtremWeb[4]에서는 결과에 대한 신뢰성을 제공하기위해 허락 임계값과 거절 임계값을 이용해 순차적 테스팅(sequential testing)을 함으로써 평균 샘플 크기를 정한다. 정해진 평균 샘플 크기는 실제 에러율을 나타내는 것으로 하나의 배치를 수행할 때마다 적응적으로 계산되는데 여기서 전체 연산 지연이 발생하게 된다.

3. 시스템 모델

본 논문에서는 작업풀 기반 **마스터-워커 모델**을 가정한다. 이 모델은 많은 인터넷 기반 병렬 컴퓨팅 시스템들이 채택하고 있다. 이 모델에서 연산은 서로 독립적인 작업 객체들로 구성된 배치(batches)가 순서대로 수행이 된다. 그림 1과 같이 초기에 배치는 작업 할당 서버에서 수많은 작업 객체로 분리되어 작업풀에 위치된다. 작업 객체는 작업풀에서 그룹 기반의 선행 처리자 우선 할당 기법에 따라 각 자원 제공자에게 할당되고 연산을 수행한 후 결과를 결과 저장 서버에 되돌려 준다. 자원 제공자가 하나의 작업 객체를 수행한 후에는 작업풀에서 새로운 작업을 요청할 수 있고 모든 작업 객체가 할당된 이후에 요청된 것들은 느린 작업을 수행하는 자원 제공자의 작업 객체를 가로채는 작업 가로채기(work stealing)[5]에 의해 하나의 배치 수행이 끝난다.

본 논문에서는 다음과 같은 **결함 모델**을 가정한다. 결함의 종류를 크게 세 가지로 분류한다. 첫째는 **정지 결함**(stop failure)으로 자원 제공자가 연산 수행 중에 멈추거나 프로세스가 멈추어 버리는 경우이다. 해결 방안으로는 모니터링 서버나 중간 관리자가 결함을 탐지하여 새로운 노드를 선정해 연산을 다시 시작할 수 있다[7]. 둘째는 **일시적 결함**(transient failure)으로 연산을 수행하던 자원 제공자가 탈퇴(leave)하거나 참여(join)하는 경우이다. 이 경우에는 연산 탈퇴 시 취해진 검사점을 이용해 다시 연산에 참여할 때 최근 검사점부터 연산을

계속 수행할 수 있다. 마지막으로 **악의적인 결함**(malicious failure)에 의한 방해 행위(sabotage)에 대해 포용할 수 있는 방해 행위 포용(sabotage tolerance) 기법이 필요하다. 기존에 제안 되었던 다수 투표법이나 결점 검사법 그리고 본 논문에서 제안하는 기법을 통해 이를 해결한다.

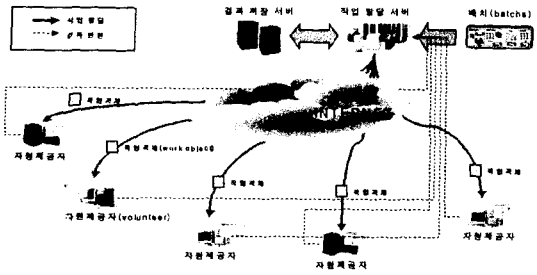

그림 1 글로벌 컴퓨팅 시스템 연산 모델

4. 그룹 기반 정확성 검사를 위한 스케줄링 기법

본 논문에서 제안하는 신용도 기반 그룹 구성 기법과 그룹 기반 스케줄링 기법을 이용한 그룹 기반 정확성 검사를 위한 스케줄링 기법을 설명한다. 기존에 제안되었던 결점 검사법과 다수 투표법을 이용한 신용도 기반 결함포용 기법[3]과는 달리 본 논문의 제안 기법에서는 각 자원 제공자들을 신용도에 기반하여 그룹 구성을 통해 스케줄링 함으로써 전체 연산에 대한 연산 지연 시간과 투표로 인한 부하를 줄일 수 있는 방법을 제안한다.

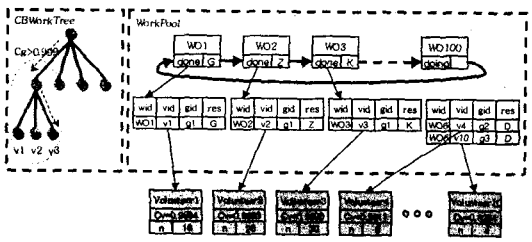

그림 2 신뢰성 기반 그룹 지정 기법의 작업풀과 작업트리

4.1 소개

그룹 기반 정확성 검사 기법에서는 그림 2와 같이 자원 제공자의 신용도 값을 기반으로 그룹을 구성하여 그룹 기반 스케줄링 기법을 적용한다. 그러기 위해서 각 자원 제공자마다 결점 검사를 통해서 신용도C_{v_i}를 측정한다. 자원 제공자가 연산을 수행한 후 결과가 만족할 만한 최소 임계값θ를 만족하면 받아들인다. θ는 응용에 따라 다른 값을 갖는 에러율 ϵ에 따라 $\theta = 1 - \epsilon$로 계산된다. 자원 제공자에 의해 수행된 결과가 θ를 만족하려면 다수 투표법을 통해 연산에 대한 신뢰성을 검증 받아야한다. 단, 초기에 작업 할당 시 스케줄러는 해당 그룹의 신뢰성에 따라 투표 그룹을 지정하여 동시에 할당하고 투표법을 실시할 때에는 자신이나 상위의 그룹에게 동시에 할당한다. 예를 들어, 하나의 여분을 갖한 다수 투표법을 통해 올바른 결과를 얻지 못하였다면 두개의 여분을 이용한 다수 투표법을 시행한다.

4.2 신용도

자원 제공자들이 수행된 연산 결과에 대한 정확성을 보장하기 위하여 자원 제공자마다 신용도를 측정한다. 본 논문에서의 신용도는 다음과 같이 정의 한다.

[정의 1] 신용도. 자원 제공자들에 의해 수행된 연산 결과의 정확성을 판단하는 근거인 자원 제공자 신뢰 지수이다. 작업 객체를 할당하기 전에 결점 검사를 통해 정확한 결과를 되돌려준 횟수n에서 결함률f와 연산 참여도CPD값의 비율로써 다음과 같이 계산한다.

$$C_{v_i} = 1 - \frac{f}{n} \cdot CPD_k(v_i) \quad (\text{만약, } n>0) \ (1)$$

$$C_{v_i} = 1 - f \cdot CPD_k(v_i) \quad (\text{만약, } n=0) \ (2)$$

수식 (1)에서C_{v_i}는 자원 제공자v_i에 대한 신용도, n은 결점 검사법에 의해 올바른 값을 되돌려준 횟수, f는 선출된 자원 제공자가 방해자일 확률인데 만약, n=0일 경우 즉, 결점 검사한 수가 없을 때에는 수식(2)를 통해 계산한다. 그리고 연산 참여도인 $CPD_k(v_i)$는 자원 제공자 v_i가 실제 연산에 참여한 시간과 탈퇴한 시간에서 실제 연산 참여시간의 비율로써 계산한다. 연산 참여도$CPD_k(v_i)$는 다음과 같이 계산한다.

$$CPD_k(v_i) = 1 - \frac{CJT_k(v_i)}{CJT_k(v_i) + CLT_k(v_i)} \quad (3)$$

수식 (3)에서 $CJT_k(v_i)$는 v_i의 연산 참여 시간을 나타내고 $CLT_k(v_i)$는 v_i의 연산 탈퇴 시간을 나타낸다. 따라서 실제로 연산에 참여하는 시간보다 탈퇴한 시간이 많다면 신용도가 낮아지는데 즉, 연산 참여도가 좋을수록 좋은 신용도를 가진다.

4.3 신용도 기반 그룹 구성 기법(CBGM)

신용도 기반 그룹 구성 기법은 자원 제공자의 신용도에 따라 그룹이 구성된다. 작업 객체를 할당하기 전에 결점 검사법에 의해 계산된 신용도 지수 C_{v_i}를 가지고 *CBGM*에 의해 비슷한 신용도를 갖는 자원 제공자끼리 그룹을 구성하여 작업트리를 완성한다. 그룹의 구성 시 최소 임계값 보다 높은 신용도를 갖는 그룹은 최상위 그룹에 위치한다. 그리고 자원 제공자가 연산 수행 후 결과를 되돌려줄 때마다 *CBGM*에 의해 작업트리는 갱신된다. 그림 2에서와 같이 신용도 기반 작업트리(*CBWT*)는 작업 풀의 순환 연결 구조에서 선택된 다음 작업 객체를 어느 자원 제공자에게 할당할지를 결정한다. 작업 트리의 자원 제공자의 상태는 연산이 가능한 상태인 "idle(i)", 탈퇴하여 다른 작업을 하거나 연산 수행중인 상태인 "busy(b)" 그리고 정지 결함이 발생한 "die(d)"값을 갖는다.

신용도를 기반한 작업 그룹을 구성한 후 각 그룹에 작업을 할당할 때 스케줄러는 작업 객체에 대해 어느 작업 객체가 어느 자원 제공자에게 할당이 되고 어느 그룹에 속해 있는지를 알기위해 아래와 같이 식별자를 부여한다. 각 작업 객체 WO의 구성 요소는 다음과 같다.

$$WO(v_{id}, w_{id}, g_{id})$$

v_{id}는 자원 제공자 식별자이고 w_{id}는 작업 객체 식별자 그리고 g_{id}는 그룹 식별자 즉, *CBWT*에서 분리되어질 각 그룹의 식별자이다.

4.4 그룹 기반 스케줄링 기법(GBSM)

본 논문에서의 작업 스케줄링은 신용도 그룹 기반 선행 처리자 우선 할당 기법을 적용한다. 4.3에서와 같이 먼저 작업을 할당하기 전에 자원 관리자에 대한 신용도를 측정하기 위해 결점 검사법을 실시하여 신용도 그룹을 만든다. 그리고, 그림 3과 같이 작업 할당 서버(*TAS*)는 *GBSM*을 통해 자원 제공자에게 작업w_i을 할당한다. 만약 할당된 자원 제공자의 신용도가 최소 임계값 보다 작다면 다수 투표법을 실시하는데 이때, *GBSM*은 같은 작업 객체를 작업 트리를 통해 현재 그룹이나 자신보다 상위에 있는 그룹의 자원 제공자에게 동시에 할당한다. 이후에 두 자원 제공자에 의해 수행된 결과R_i는 *CBGM*을 통해 결과를 확인하고 올바른 값이면 결과를 *TAS*에 되돌려 준다. 스케줄링 시 고려해야할 것은 각 작업풀에서 작업 객체의 상태를 다음과 같이 고려해야만 한다. 작업풀의 작업 객체 상태는 세 가지로 분류되는데 수행이 끝

나 올바른 값을 되돌려 주었을 경우 "done", 작업을 할당하기 전 상태나 작업 수행이 끝난 상태를 "undone", 작업을 할당하여 수행중인 상태인 "doing" 상태를 가진다. 그림 4에서 *GBSM* 알고리즘을 표현한다.

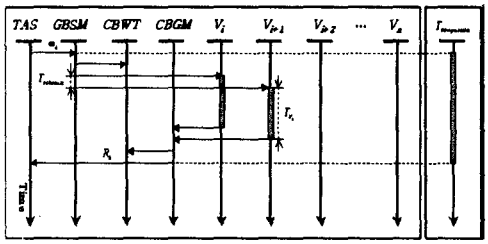

그림 3 GBSM 순차 다이어그램

```
var
  wo : WorkObject;
  wp : WorkPool;
  wt : CBWT;
  wid : WorkObjectID;
  vid : VolunteerID;
  gid : GroupID;

while wp.nextWorkPoolObject() != null &
      wp.nextWorkObjectState == "undone" do
    wo := wp.nextUndoneWorkObject();
    if initial computation then
      allocationWorkObject(wo);
    else
      for wt.everyCredibilityGroup do
        if( wt.gid > wo.gid )
          vid := wt.searchVolunteer("idle");
        endif
      endfor;
    endif
    wo.gid := wt.getGroupID(vid);
    allocationWorkObject(wo);
endwhile;
```

그림 4 GBSM 알고리즘

5. 성능 평가

본 논문에서 제안한 그룹 기반 정확성 검사를 통한 스케줄링 기법은 연산 지연 시간의 분석을 통해 기존 기법보다 적은 연산 지연 시간과 부하를 보인다. 첫째로, **연산 지연 시간은 정확성** 검사로 인해 수행될 연산이 지연되는 시간이다. 기존 기법에서는 투표 시 하나의 연산이 끝난 후 신용도를 측정해 임계값을 넘지 않으면 재 할당하는 방식이지만 그림 3에서처럼 제안 기법에서는 *GBSM*이 *CBWT*를 통해 먼저 신용도를 측정하여 재 할당이 필요하다면 동시에 스케줄링 함으로써 연산 지연 시간을 줄일 수 있다. 둘째로, **부하**는 자원 제공자의 신용도에 따라 추가로 수행되는 여분의 양이다. 제안 기법에서 여분의 양을 줄이기 위해 *GBSM*은 스케줄링 시 재작업 할당의 경우 자신의 그룹이나 상위 그룹에 작업을 할당함으로써 자신보다 신용도가 높은 자원 제공자에게 할당할 확률이 높아 여분을 줄일 수 있다.

- **연산 지연 시간**. 기존 기법의 연산 지연 시간에 비해 본 논문에서 제안한 그룹 기반 정확성 검사 기법의 연산 지연 시간을 비교 분석한다. m개의 추가 여분의 연산을 수행할 때의 전체 연산 수행 시간 $T_{computation}$은 아래와 같다.

기존 기법에서, $T_{computation} = mT_{v_i} + mT_{schedule} + \alpha$ (4)

제안 기법에서, $T_{computation} = T_{v_i} + mT_{schedule} + \beta$ (5)

수식 (4)에서 T_{v_i}는 자원 제공자가 하나의 작업 객체를 수행하는 시간, $T_{schedule}$는 투표를 위해 재작업을 할당하기 위한 스케줄링 시간, m은 투표 시 여분의 갯수 그리고 α, β는 통신 시간으로 전체 연산 지연 시간에서 아주 작기 때문에 고려하지 않는다. 두 수식에서, 기존 기법에서는 하나의 연산이 수행된 결과에 따라 신용도를 측정해 투표 그룹이 설정되기 때문에 제안 기법 보다 m배의 연산 수행 시간이 걸린다. 따라서, 기존 기법보다 제안 기법의 전체 연산 지연 시간이 적은을 알 수 있다.

- **부하**. 부하는 기존 기법에서의 결과에 대한 정확성 검사를 위한 여분의 양보다 제안 기법에서의 여분의 양이 적음을 보인다. 따라서, 신용도가 높은 자원 제공자를 선출해야하고 선출된 자원 제공자는 비교적 악의적이지 않은 자원 제공자이다. 따라서, 선의의 자원 제공자를 선출할 확률은 **베르누이 확률을** 따른다. 여분의 양은 결과 검사에서 신용도가 낮을수록 많은 여분이 필요하므로 제안 기법이 신용도가 높은 자원 제공자를 선출함을 보인다.

기존 기법에서, $P_X = \binom{n}{s}[\frac{F}{N}]^s[1-\frac{F}{N}]^{n-s}$ (6)

제안 기법에서, $P_X = \binom{n}{s}[\frac{F}{N-\sum_{j=k}^{n} N_{G_j}}]^s[1-\frac{F}{N-\sum_{j=k}^{n} N_{G_j}}]^{n-s}$ (7)

수식 (6), (7)에서 P_X는 선의의 자원 제공자를 선출할 확률, n은 n번 시행할 횟수, s는 s번 성공할 횟수, F는 악의적인 자원 제공자 수, N은 전체 자원 제공자 수이다. 수식 (7)에서 N_{Gi}는 i번째 그룹에 자원 제공자 수이다. 두 수식을 비교해보면, 기존 기법에서는 전체 자원 제공자 중에 작업이 먼저 끝난 자원 제공자에게 할당하지만, 제안 기법에서는 *GBSM*에 의해 신용도가 자신 보다 높은 그룹의 자원제공자를 선출하여 할당하기 때문에 확률적으로 적은 여분이 필요함을 알 수 있다.

6. 결론 및 향후 연구 방향

본 논문에서 제안한 그룹 기반 정확성 검사 기법으로 기존 기법보다 적은 연산 지연 시간으로 연산 결과에 대한 정확성을 보장할 수 있었다. 본 논문에서 제안한 결과 검사 기법은 어느 글로벌 컴퓨팅 시스템이나 적용가능하다. 가까운 미래에 본 기법을 글로벌 컴퓨팅 시스템에 구현하여 적용할 계획이다.

참고 문헌

[1] M. O. Neary, P. Cappello. Advanced Eager Scheduling for Java Based Adaptively Parallel Computing. JGI'02, November 3-5, 2002.

[2] D. Molnar. The SETI@home Problem.

[3] L. Sarmenta. Sabotage-Tolerance Mechanism for Volunteer Computing Systems. FGCS, 18(4). 2002.

[4] C. Germain-Ren명, N. Playez. Result Checking in Global Computing Systems. ICS'03, June 23-26, 2003.

[5] R. D. Blumofe, C. F. Joerg, B. C. Kuszmaul, C. E. Leiserson, K. H. Randall, and Y. Zhou. Cilk: An Efficient Multithreaded Runtime System. In 5th ACM SIAPLAN Symposium on Principles and Practice of Parallel programming (PPOPP '95), pages 207-216, Santa Barbara, CA, July 1995.

[6] L. LAMPORT, R. SHOSTAK, M.PEASE. The Byzantine Generals Problem. ACM Transactions on Programming Languages and Systems, Vol. 4, No. 3, July 1982.

[7] 김홍수, 강인성, 최성진, 황일선, 황종선, 유헌창. 인터넷 기반 병렬 컴퓨팅에서 중간 관리자 구성과 결함포용 기법, 한국정보과학회 가을 학술발표논문집(3), 제 30권 2호, pp. 643-645, 2003.