Journal of Korea Multimedia Society Vol. 11, No. 3, March 2008(pp. 331-340)

랜덤 네트워크에 적용 가능한 SBIBD기반의 부하 균형 알고리즘

이옥빈[†], 이여진[††], 최동민[†††], 정일용[††††]

요 약

부하균형을 위해서는 각 노드의 부하상태 정보가 모든 노드에게 알려져야 하는데 O.Lee[15]가 제안한 SBIBD(Symmetric Balanced Incomplete Block Design) 기반의 부하균형 알고리즘은 노드의 수가 v일 때 2라운드 메시지 교환과 $O(v\sqrt{v})$의 통신오버헤드에 의해 각 노드가 네트워크상의 모든 노드로부터 부하상태 정보를 수신한다. 이 때 각 노드의 통신오버헤드는 $O(\sqrt{v})$로서 각 노드가 균등한 오버헤드를 가지며 각 노드가 수신한 정보는 중복이 없다는 점 또한 이 알고리즘의 장점이다. 기술한 바와 같이 이 알고리즘은 매우 효율적이지만 임의의 소수 p에 대하여 노드의 수가 $v=p^2+p+1$일 때만 수행될 수 있다. 이 논문에서는 네트워크상의 노드의 수가 임의의 양의 정수일 때도 이 알고리즘이 동작할 수 있도록 특수한 결합구조를 생성하고자 한다. 이 제안의 성과를 평가하기 위해 노드의 수가 $w(5 \le w \le 5{,}000)$인 네트워크를 가정하고 각각의 경우에 두 번의 라운드 정보교환으로 최소 80% 이상의 부하상태정보를 수신하도록 실험을 수행한 결과 트래픽오버헤드는 $O(w\sqrt{w})$ 보다 낮으며 각 노드의 트래픽 오버헤드가 균등하지는 않으나 그 편차가 크지 않은 것으로 나타났다.

An Algorithm of SBIBD based Load Balancing Applicable to a Random Network

Okbin Lee[†], Yeojin Lee[††], Dongmin Choi[†††], Ilyong Chung[††††]

ABSTRACT

In order to make load balancing, workload information of nodes should be informed to the network. In a load balancing algorithm[13] based on the SBIBD(Symmetric Balanced Incomplete Block Design), each node receives global workload information by only two round message exchange with $O(v\sqrt{v})$ traffic overhead, where v is the number of nodes. It is very efficient but works well only when $v=p^2+p+1$ for a prime number p. In this paper, we generate a special incidence structure in order for the algorithm works well for an arbitrary number of nodes. In the experiment with w,($5 \le w \le 5{,}000$), nodes and more than 80% of receiving workload information, traffic overhead was less than $O(w\sqrt{w})$and the result for standard deviation of traffic overhead showed that each node has largely balanced amount of traffic overhead.

Key words: Load Balancing(부하균형), Block Design(블록 디자인)

※ 교신저자(Corresponding Author) : 정일용, 주소 : 광주광역시 동구 서석동(501-759), 전화 : 062)230-7712, FAX : 062)230-7754, E-mail : iyc@chosun.ac.kr
접수일 : 2007년 10월 18일, 완료일 : 2008년 1월 29일
[†] 정회원, 조선대학교 컴퓨터공학과
(E-mail : lobin@mogent.com)
[††] 준회원, 조선대학교 컴퓨터공학과
(E-mail : smile96@naver.com)
[†††] 준회원, 조선대학교 컴퓨터공학과
(E-mail : cdm1225@hanafos.com)
[††††] 종신회원, 조선대학교 컴퓨터공학과
※ 본 연구는 산업자원부의 지역혁신 인력양성사업의 연구결과로 수행되었음

1. 서 론

분산시스템에서 어떤 시스템은 과부하 되는가 하면 또 어떤 시스템은 부하가 적거나 유휴상태를 나타내기도 한다. 그러므로 이렇게 불균형한 부하를 갖는 시스템 간에 부하균형이 유지되도록 부하균형을 유지함으로써 시스템의 활용도를 높이고 응답시간을 줄일 수가 있다. 따라서 부하분산 기법은 부하상태에 대한 정보의 유지 및 전파, 프로세스이동시점의 결정, 이동시킬 프로세스 선택, 프로세스가 이동될 컴퓨터의 결정, 전송된 프로세스의 상태복구 등을 고려해야한다. 이때 컴퓨터의 부하상태를 측정하거나 부하상태정보를 전파하기 위한 방식은 최소의 오버헤드를 가져야한다. 컴퓨터의 작업량을 나타내는 지수는 CPU의 대기열에 의해 표현되는 것이 가장 효과적이라고 알려져 있다[1]. 부하균형 알고리즘은 정적(static)이거나 동적(dynamic)일 수 있다. 분산시스템이 v개의 컴퓨터로 구성되어 있을 때 정적인 분산시스템에서는 i번째 작업을 i(mod v)번째 컴퓨터에 할당한다. 동적인 분산시스템에서는 각 시스템의 부하상태 정보를 이용하여 부하가 적은 컴퓨터에 작업을 할당한다. 또한 작업량 분산 방식은 특정 시스템에서 모든 노드에 대한 부하정보를 가지고 있으면서 불균형된 노드간에 부하를 분산하도록 중재하는 집중형과 각각의 노드가 다른 노드에 대한 부하정보를 가지고 부하분산을 실행하는 분산형으로 구분될 수 있다. 분산시스템에서 부하 분산을 위해서는 분산형의 동적인 알고리즘을 사용하는 것이 바람직하다[2-6]. 부하 분산을 위해 각각의 노드에게 주어지는 부하상태정보는 최신의 정보이어야 할 필요가 있다. 유효시간이 지난 정보는 각각의 노드에서 시스템의 전체에 대한 상황을 올바로 파악할 수 없게 하고 더불어 올바른 부하분산을 수행할 수 없게 한다. 이렇게 최신의 정보를 유지하기 위해서는 주기적으로 갱신되는 정보전송을 위한 통신비용이 심각한 문제가 될 수 있다.

[7-9] 등의 연구에서는 전체 노드 간에 각 노드들의 부하상태 정보를 공유하기 위한 과다한 통신비용을 줄이기 위해 인접한 노드 사이에만 부하상태정보를 교환하고 국부적으로 부하균등화 수행을 반복함으로써 전체적으로 균등부하에 수렴하게 됨을 주장하고 있다, 그러나 만일 적은통신비용으로 네트워크상의 모든 노드의 부하상태를 알 수 있다면 전체적으로 부하균등화를 이루는 시간은 상당히 단축될 수 있다.

노드의 수가 v인 네트워크를 가정할 때, 완전연결 그래프의 경우 모든 노드에 대한 부하상태정보 수집을 위해서는 1라운드 메시지 교환과 $O(v^2)$의 통신이 요구된다. 하이퍼큐브상에서 동작하도록 설계된 CWA(Cube Walking Algorithm)[10]의 경우 $\log_2(v)$ 라운드의 메시지교환과 $v\times(\log_2 v)^2$의 통신량을 요구한다[11,12]. 토폴로지에 독립적인 방법으로 각 노드간의 통신패턴을 형성하는 방법을 제안한 SBN (Symmetric Broadcast Networks) 또한 $\log_2(v)$ 라운드의 메시지 교환과 $2v\times(\log_2 v)^2$ 의 통신량을 요구한다[13,14]. 이들은 또한 $v=2^d$ 이라는 노드 수의 제약을 가지고 있다. SBIBD 기반의 알고리즘은 2라운드 메시지교환과 $2v\sqrt{v}$보다 작은 통신량을 갖는 효율적인 방법이지만 마찬가지로 임의의 소수 p에 대하여 $v=p^2+p+1$이라는 노드 수의 제약을 갖는다[15].

이 논문에서는 [15]의 노드수의 제약을 없애고 임의의 수의 노드로 구성된 네트워크에서 [15]의 부하균형 알고리즘이 동작할 수 있도록 특수한 결합구조를 생성하는 방법을 제안한다. 제안된 방법의 성능평가를 위하여 w,($5\le w\le 5{,}000$), 개의 노드가 존재하는 네트워크를 가정하고 각각의 경우에 2라운드 정보교환으로 최소 80% 이상의 부하상태정보를 수신하도록 실험을 수행한 결과 트래픽오버헤드는 $O(w\sqrt{w})$ 보다 낮으며 각 노드의 트래픽 오버헤드가 균등하지는 않으나 그 편차가 크지 않은 것으로 나타났다.

이 논문의 구성은 다음과 같다.

2장에서는 SBIBD를 정의하고 SBIBD 기반의 부하균형 알고리즘을 살펴본다. 3장에서는 이 알고리즘이 적용 가능한 임의 노드 수 w로 이루어진 특수한 결합구조를 생성하는 방법을 제안하다. 4장에서 실험결과를 제시하고 5장은 결과를 논한다.

2. SBIBD 기반의 부하균형

2.1 SBIBD의 정의

SBIBD(Symmetric Balanced Incomplete Block Design(SBIBD)는 다음과 같이 정의될 수 있다.

V={0,1,...v-1}를 v개의 원소를 갖는 집합이라고 하자. 그리고 집합족 B=B_0, B_1, .., B_{b-1}는 V의 원소들로 이루어진 부분집합들로 구성되어 있다고 하자. 유한 결합구조 σ=A,B에 대하여 σ가 다음조건을 만족할 때 이는 BIBD(Balanced Incomplete Block Design)이다. 그리고 이를 (b,v,r,k,λ)디자인이라 한다[16].

(1) B는 블럭이라 불리는 b개의 부분집합으로 구성되며 각각의 부분집합은 V에 속하는 k개의 원소를 갖는다.

(2) V의 원소 각각은 r개의 블록에만 존재한다.

(3) V의 원소 중 임의의 두 원소 쌍(pair)은 B에서 λ번 나타난다.

(4) k < v

임의의 (b,v,r,k,λ)디자인에 대하여 만일 다음 두 가지 조건을 만족하면 이는 SBIBD(symmetric balanced incomplete block design)이고 (v,k,λ)디자인이라 한다.

(1) k=r

(2) b=v

BIBD를 구성하기 위한 파라미터 b,v,r,k,λ 간에는 특별한 관계가 존재한다. 예를 들면 (b,v,r,k,λ) 디자인에서는 bk=vr이 성립하고 r(k-1)=λ(v-1)이 성립한다. 그리고 (v,k,λ)디자인의 경우 모든 두 블록 사이에는 λ 개의 교집합이 존재한다. 그렇지만 임의의 (b,v,r,k,λ)디자인 또는 (v,k,λ)디자인이 존재하기 위한 조건 및 생성하기 위한 일반적인 방법은 찾기 어려운 것으로 알려져 있다. r(k-1)=λ(v-1)에 의하면 λ=1인 경우에 (v,k,1)디자인 구성이 가능한 v는 $v=k^2-k+1$로 한정됨을 알 수 있다. [15]에서는 임의의 소수 p에 대하여 (v.p+1,1)디자인을 생성하는 방법을 제시하였다.

2.2 SBIBD 기반의 부하균형 알고리즘

[15]에서는 소수 p에 대하여 (v,p+1,1)디자인을 기반으로 2단계 메시지교환과 각 노드 당 균등한 p의 통신오버헤드에 의해 각 노드가 망 전체의 노드에 대한 부하상태정보를 수신할 수 있는 알고리즘을 제시하였다. 표 1의 예를 사용하여 이 알고리즘을 설명하기로 하자.

표 1의 예에서 보는 바와 같이 각각의 블록 B_i는 원소 i를 포함한다. 이 알고리즘에서 각각의 원소 i는 노드 i에 매핑된다. 여기에서 우리는 블록번호와 일치하는 노드를 블록대표노드라 부르기로 한다.

표 1. (13,4,1)디자인의 예

V = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 }
B_0 = { 0, 1, 2, 3 }
B_1 = { 1, 6, 9, 12 }
B_2 = { 2, 5, 9, 10 }
B_3 = { 3, 5, 7, 12 }
B_4 = { 1, 4, 7, 10 }
B_5 = { 1, 5, 8, 11 }
B_6 = { 0, 4, 5, 6 }
B_7 = { 2, 6, 7, 11 }
B_8 = { 2, 4, 8, 12 }
B_9 = { 0, 7, 8, 9 }
B_{10} = { 3, 6, 8, 10 }
B_{11} = { 3, 4, 9, 11 }
B_{12} = { 0, 10, 11, 12 }

메시지교환 제 1라운드에서 각각의 블록대표노드는 동일 블록의 노드들로부터 그들의 부하상태정보를 수신한다. 메시지교환 제 2라운드에서 각각의 블록대표노드는 1라운드에 수신한 정보와 자신의 정보 묶음을 동일 블록의 노드들에게 전송한다. 블럭 B_i={ i, a, b, c} 에 대하여 노드 i는 첫 번째 라운드에 노드 a, b ,c로부터 부하상태정보를 수신한다. 두 번째 라운드에서 노드 i는 노드 a에게 {i, b, c} 의 정보를, 노드 b에게는 {i, a, c} 그리고 노드 c에게는 {i, a, b}의 정보를 전송한다. (v,p+1,1)디자인의 각 블록은 동일한 크기 (p+1)을 가지므로 각 라운드 당 각 노드는 동일하게 p의 트래픽 오버헤드를 분담하면서 중복없이 망 전체의 노드에 대한 정보를 수신하게 된다. 여기에서 각 노드가 망 전체의 노드에 대한 정보를 수신할 수 있는 이유는 (v,p+1,1)디자인에서 두 원소의 쌍이 동일 블록에 존재하는 경우는 정확히 한번 발생하기 때문이다. 예를 들어 노드 0의 경우, 제 1 라운드에서 노드 1, 2, 3으로부터 그들의 부하상태정보를 수신하고 제 2라운드에서 노드 6으로부터 노드집합 4, 5, 6, 노드 9로부터 노드집합 7, 8, 9 그리고 노드 12로부터 노드집합 10, 11, 12의 부하상태정보를 수신하게 된다.

이 방법은 설명한 바와 같이 명백하고도 효율적인 방법이지만 $v=p^2+p+1$이라는 노드 수의 제약을 가지고 있다. 그러므로 우리는 이 제약을 완화할 수 있는 방법에 대하여 연구할 필요가 있다.

3. 특수 결합구조 생성

이 장에서는 [15]의 알고리즘을 랜덤 네트워크에 적용 가능하도록 하기 위한 특수 결합구조를 생성하는 알고리즘을 제안한다. 이는 (v,p+1,1)의 특징을 최대한 보존하면서 임의의 블록 수를 갖는 결합구조를 의미한다. 이 논문에서 제안하는 방법은 임의의 노드 수 w를 위한 디자인을 구성하고자 할 때 (v,p+1,1)디자인으로부터 필요한 수만큼의 블록을 지움으로써 w블록을 가진 디자인을 생성하도록 하고 있다. 이때 어떤 블록을 지울 것인지를 결정하는 문제는 대단히 어려운 문제이다. 표 1의 (13,4,1)디자인으로부터 3개의 원소와 그 블록을 삭제함으로써 10개의 블록으로 구성된 디자인을 얻을 수 있게 된다. 여기에서 삭제된 원소의 집합을 DEL_BLOCKS라고 부르기로 한다. 만약 원소 10, 11, 12를 삭제하기로 결정한다면 즉, DEL_BLOCKS={10,11,12}인 경우라면, 블록 10, 11, 12를 삭제하고 남은 10 개의 블록에서 원소 10, 11, 12를 삭제한다. 그 결과 수신되지 못한 정보의 개수는 12가 된다. 왜냐하면 B_{10}의 구성요소에 의해 노드 3은 정보교환 제 2라운드에서 노드 10으로부터 노드 6과 8에 대한 정보를 수신해야하지만 더 이상 노드 10이 존재하지 않으므로 정보 수신이 불가능하다. 노드 6과 8의 경우도 마찬가지의 이유로 동일한 결과를 갖게 된다. 또한 B_{11}의 구성요소에 의해 노드 3, 4, 9는 자신 이외의 두개의 노드에 대한 정보를 얻을 수 없게 된다. 하지만 B_{12}의 경우는 정보 미수신의 결과를 가져오지 않는데 이는 B_{12}와 DEL_BLOCKS의 차가 오직 하나의 원소뿐이기 때문이다. 이와 같이 DEL_BLOCKS={10,11,12}일 때 미수신되는 정보의 수는 6+6+0=12 개가 된다.

우리는 미수신되는 정보의 수를 줄이기 위해 다른 조합을 선택할 수 있다. 예를 들어 DEL_BLOCKS={0, 1, 6}인 경우라면, 미수신되는 정보의 수는 2+2+2=6 개가 된다.

이와 같이 (v,p+1,1)디자인으로부터 몇 개의 블록과 그에 해당하는 원소를 삭제한 후의 결합구조에 의해 [15]의 알고리즘을 적용한 결과는 망 전체의 노드에 대한 정보 수신을 불가능하게 한다. 그 이유는 특정 블록을 삭제함으로써 "서로 다른 두 원소의 쌍이 정확히 한번 존재한다." 라고 하는 글로벌 정보수신을 가능하게 하는 필수 조건이 더 이상 성립하지 않기 때문이다. 그러므로 DEL_BLOCKS에 포함된 블록들이 다음과 같은 특성을 가질 때 미 수신 정보의 수를 최소화할 수 있음을 알 수 있다. 첫째, DEL_BLOCKS의 원소들이 해당 블록에 최대한 많이 존재하도록 해야 한다. 둘째, 해당 블록에서 DEL_BLOCKS의 원소를 삭제한 결과 각각의 블록의 크기는 최대한 작아야 하면 각 블록의 크기가 균등할수록 바람직한 원하는 결과를 얻을 수 있다. 하지만 이 문제에 대한 최적의 해를 찾기는 매우 어렵다.

알고리즘1에서는 효율적으로 DEL_BLOCKS를 구성하여 랜덤 네트워크상에서 효율적인 부하상태 정보 교환이 이루어질 수 있기 위한 결합구조를 생성하는 방법을 제안한다. 이 알고리즘은 원소 쌍의 부재율을 최소화함으로써 미수신율을 줄이고 완성된 결합구조를 구성하는 각 블록이 가능한 한 균등한 크기를 가지도록 함으로써 일부 노드에 통신오버헤드가 편중되지 않도록 하였다. 이 알고리즘은 또한 원하는 수신율을 지정할 수 있도록 하였다. 만일 완성된 결과가 지정된 수신율에 미치지 못하는 경우에 일부의 블록의 링크를 추가함으로써 수신율을 높이도록 하고 있다.

삭제가 이루어진 후 수신율은 삭제된 블록이 얼마나 많은 삭제된 블록번호를 포함하느냐에 좌우된다. 즉, 미 수신율은 B_i의 크기에 비례하게 된다. $B_i''= B_i\text{-}DEL\_BLOCKS,\ i \in DEL\_BLOCKS$. 이 알고리즘에서는 가장 많은 원소를 가진 B_i에 포함 되지만 비교적 그 크기 작은 B_i에는 포함되지 않는 원소 중에서 DEL_BLOCKS에 포함될 원소를 선택한다. 만일 지정된 수신율에 미달하여 수신율을 조정할 필요가 있는 경우에는 가장 큰 B_i를 선택하여 B_i의 원소 x중 B_x''의 크기가 가장 작은 블록에 B_i''의 원소를 포함시킨다. 이렇게 함으로서 각 노드가 보다 균등한 트래픽 오버헤드를 분담하도록 하는 결과를 얻을 수 있다.

이 알고리즘에서 어떤 블록을 지울 것인지를 결정하는 함수는 find_next choice(BS)이다. 여기에서 BS는 이미 삭제하기로 결정된 블록들 즉 B_i'', $i \in$ DEL_BLOCKS들의 집합족을 말한다. 이 함수는 DEL_BLOCKS에 추가될 원소를 반환하는데 그 선택 기준은 BS의 가장 큰 블록에 속하면서 동시에 비교적 그 크기가 작은 BS의 블록에는 속하지 않는 원소이다. '비교적 작은' 이라는 표현은 명료하지 않게

알고리즘 1. 특수결합구조 생성

입력 : 1. 임의의 양의 정수 w
　　2. 수신율
출력 : 1. 특수결합구조 $Z = \{V,B\}$
(1) 다음 조건을 만족하는 정수 v를 선택한다.
　$X = \{x | x=p^2+p+2,\ x \geq w\}$
　$v = min(X)$
(2) (v,p+1,1)디자인 $Z=\{V,B\}$를 생성한다.
(3) $x>w$이면, 다음과 같이 Z의 원소와 블록을 삭제한다.
　(a) 집합 DEL_BLOCKS와 집합족 BS을 다음과 같이 정의한다.
　　DEL_BLOCKS={v-1}
　　BS(1) = B_{v-1} - {v-1}
　　V=V-{v-1}
　(b) V의 크기가 w가 될 때까지 삭제한다.

```
while (|V | > w){
   nc = find_next_choice(BS);
   DEL_BLOCKS=DEL_BLOCKS ∪ {nc };
   BS(size)= Bnc - DEL_BLOCKS;
    // size는 DEL_BLOCKS의 크기
   V = V - {nc};
   B = B - Bnc ;
   }
for (each Bi)
   Bi = Bi - DEL_BLOCKS;
//------------------
find_next_choice(BS){
   DO_NOT_DELS =
     find_short_blocks();
       // BS(i)의 크기가 비교적 적은
       // i들의 집합
   Long_BLOCK = find_x(BS) ;
       //BS중 그 크기가 가장 큰 집합.
   nc=find_next();
       //Long_BLOCK 에 포함되지만
       //DO_NOT_DELS에는 포함되지 않는 원소
   return nc ;
   }
```

(4) 수신율이 지정된 수신율보다 작으면 다음을 수행한다.

```
whie (수신율 < 지정수신율) {
   x = find_x(BS) ;
   // BS중 의 크기가 가장 큰 블록
   // x를 찾는다.
   i= find_i(x );
     // i∈Bx 에 대하여 가장 작은 크기를
     //  갖는 Bi를 찾는다.
   Bi =  Bi ∪ Bx ;
   BS = BS-BS(x);
}
```

들릴 수 있지만 이는 상황에 따라 조정할 수 있으며 이 논문의 실험에서는 그 크기가 p/4보다 작거나 같은 블록을 비교적 작은 블록으로 규정하였다. 여기에서 p+1은 블록의 삭제가 이루어지기 전 각 블록의 크기이다. 여기에서는 보다 신속한 결과를 얻기 위해 삭제된 블록에 빈번하게 나타나는 원소를 찾기 위한 탐색은 하지 않지만 위와 같은 선택 방법은 각 원소의 출현 횟수가 동일하므로 선택된 원소가 비교적 큰 크기를 갖는 블록에 포함될 확률을 증가시킬 것으로 기대된다.

이 알고리즘에 의해 w = 9인 경우의 특수결합구조를 생성하는 과정을 가정해보자. 이 경우에 표 1의 예제로부터 4개의 원소와 해당 블록을 삭제하게 되며 이 때 선택된 조합은 DEL_BLOCK = {12, 0, 1, 11} 이다. 그리하여,

BS(1) = {10},
BS(2) = {2, 3},
BS(3) = {6, 9},
BS(4) = {3, 4, 9}

에 의해 미수신되는 정보의 수는 10 이고 수신율은 RR = 87.7%가 된다.

8개의 노드를 위한 특수결합구조를 새성하기를 원한다면 위의결과에 의해 다음 선택은 원소 3이 된다. 왜냐하면 원소 3은 가장 큰 블록 BS(4)의 첫 번째 원소이면서 비교적 작은 블록 예를 들면, 그 크기가 1인 BS(1)에 속하지 않기 때문이다. 그 결과

DEL_BLOCKS = {12, 0, 1, 11, 3} 그리고
BS(1) = {10},
BS(2) = {2},
BS(3) = {6, 9},
BS(4) = {4, 9}.
BS(5) = {5, 7}

에 의해 미수신되는 정보의 수는 6, 수신율은 RR = 90.625%가 된다 .

알고리즘의 4단계에서는 수신율의 결과가 지정된 수신율에 미치지 못할 경우, 수신율 향상을 위해 임의의 블록에 원소를 추가하도록 하고 있다. 위의 9개 노드로 이루어진 결합구조(RR = 87.7%)에서 수신율을 95% 이상으로 향상시키기를 원한다면 어떤 블록에 원소를 추가해야 한다. 이 알고리즘에 의하면 추가할 원소는 그 크기가 가장 큰 BS(4)={3,4,9}가 되고 추가되는 장소는 B_3, B_4, B_9 중 B_3가 그 크기가 가장 작다면 B_3에 {3, 4, 9}를 추가한다. B_i의 크기는 노드 i의 통신량을 의미하기 때문이다.

이렇게 2의 원소를 추가한 결과, 미수신되는 정보의 수는 6으로 수신율 95%를 달성하며 블록의 크기

는 가능한 작은 편차를 유지한다.

표 2는 알고리즘 1에 의해 새성된 9 개 노드를 위한 특수결합구조를 보인 것이다. 여기에서 수신율은 87.7%, 각 노드가 수신한 정보의 개수의 평균은 7.889 그리고 정보 수신율에 대한 표준편차는 1.269 이다. 각 노드의 트래픽 오버헤드는 평균 4.22 이며 트래픽 오버헤드의 표준편차는 0.833으로 대체로 균등한 오버헤드를 갖는다.

표 3은 9 노드를 위한 특수결합구조를 생성한 후 수신율 향상을 위해 블록에 원소를 추가하는 경우를 보여준다. 수신율 87.7%에서 95%로의 향상을 위해 2개의 원소를 추가한 결과 수신율의 표준편차는 0.441이다. 노드 당 평균 트래픽 오버헤드는 4.66으로서 1회당 오버헤드는 $\sqrt{9}$보다 작고 표준편차는 0.5로서 모든 노드가 동일한 오버헤드를 갖는 것은 아닐지라도 대체로 균등한 분배가 이루어지고 있음을 알 수 있다.

그렇지만 원소 추가에 의해 수신정보의 중복이 발생하게 되는데 이는 원소의 쌍이 중복하여 존재하는 경우가 발생하기 때문이다. 표 4는 알고리즘 1에 의해 생성된 표 3의 특수결합구조에 따라 정보교환을

표 2. 알고리즘 1의 수행 결과 1

알고리즘 1에 의해 9 노드를 위한 특수결합구조 생성
DEL_BLOCKS= {12, 0, 1, 11} DEL ELES1 = {10} DEL ELES2 = {3, 2} DEL ELES3 = {6, 9} DEL ELES4 = {3, 4, 9} B_2 = {2, 5, 9, 10} B_3 = {3, 5, 7} B_4 = {4, 7, 10} B_5 = {5, 8} B_6 = {6, 4, 5} B_7 = {7, 2, 6} B_8 = {8, 2, 4} B_9 = {9, 7, 8} B_{10} = {10, 3, 6, 8}
미수신 정보의 수 = 10 수신율=87.7% 수신율의 표준편차 = 1.269 노드당 평균 트래픽 오버헤드 = 4.22 (2.11 per round) 트래픽 오버헤드의 표준편차= 0.833 [0.782(round 1), 0.601(round 2)]

표 3. 알고리즘 1의 수행 결과 2

알고리즘 1에 의해 9 노드를 위한 특수결합구조 생성 후 수신율 향상
B_2 = {2, 5, 9, 10} B_3 = {3, 5, 7} [{4, 9} B_4 = {4, 7, 10} B_5 = {5, 8} B_6 = {6, 4, 5} B_7 = {7, 2, 6} B_8 = {8, 2, 4} B_9 = {9, 7, 8} B_{10} = {10, 3, 6, 8}
미수신 정보의 수 = 4 , 수신율=95% 수신율의 표준편차 = 0.441 노드당 평균 트래픽 오버헤드 = 4.66 (2.33 per round) 트래픽 오버헤드의 표준편차 = 0.5 [(0.707(round 1), 0.866(round 2)]

표 4. 알고리즘 1의 수행 결과 3 : 각 노드에서 수신하는 정보

노드 ID	제 1라운드 수신정보
	제 2라운드 수신정보
2	{5, 9, 10}
	{7, 6} {8, 4}
3	{5, 7, 4, 9}
	{10, 6, 8}
4	{7, 10}
	{ 3,5,7,9 } {6, 5}{8, 2}
5	{8}
	{2, 9, 10} {3, 7, 4, 9} {6, 4}
6	{4, 5}
	{7, 2} {10, 3, 8}
7	{2, 6}
	{3, 5, 4, 9} {4, 10} {9, 8}
8	{2, 4}
	{5 }{9, 7} {10, 3, 6}
9	{7, 8}
	{2, 5, 10} {3, 5, 7, 4}
10	{3, 6, 8}
	{2, 5, 9} {4, 7}

수행한 결과 수신정보의 중복이 있음을 보여준다. 노드 5는 노드 4와 9에 대한 정보를 중복 수신하고 있다.

4. 실험 및 성능 평가

이 논문에서는 임의의 노드수를 갖는 랜덤네트워크에 SBIBD 기반의 효과적인 부하상태정보교환 알고리즘을 적용 가능하도록 하기 위해 임의의 수의

블록을 갖는 특수한 결합구조를 생성하였다. 이장에서는 제안된 특수 결합구조 생성 알고리즘의 평가 및 제안된 방법으로 새성된 결합구조를 사용한 부하상태정보교환의 결과에 대한 성능 평가를 수행한 결과를 제시한다.

실험을 위해서는 인텔 펜티엄 프로세서 1.86GHz/RAM 1MB/위도우즈 XP 시스템에서 C 프로그램을 이용하여 알고리즘을 구현하고 각각의 노드 수의 경우에 대하여 결과를 생성하기까지의 경과시간을 측정하였다. 측정 결과는 그림 1과 같다. 노드의 수가 $w=v=p^2+p+1$인 경우 짧은 시간 내에 결과를 생성한다. 하지만 w와 $v=p^2+p+1$의 차가 클수록 점차 결과를 생성하는데 필요한 시간이 증가함을 알 수 있다. 이는 차가 클수록 탐색공간의 크기가 증가하기 때문이다. 노드의 수가 많을수록 디스크 액세스에 소모되는 시간이 증가하는 것 또한 주요 원인으로 작용하였다.

다음은 생성된 결합구조의 효용성에 대하여 평가하였다. 실험의 범위는 임의의 수의 노드가 존재하는 네트워크 환경으로 노드의 수 w는 5부터 5000의 경우까지로 정하였다. 그리고 임의의 노드 수의 경우마다 2라운드 정보교환으로 80% 이상의 부하상태정보를 수신하도록 할 때 각 노드수의 경우에 대하여 노드의 정보수신율의 평균과 표준편차 그리고 통신오버헤드의 평균과 표준편차를 구하였다.

그림 2는 노드의 수에 따른 정보수신율을 나타내고 있다. $w=p^2+p+1$일 때, 수신율을 100%이고 삭제되는 불럭의 수가 증가함에 따라 수신율이 차차 감소하여 지그재그 모양의 그래프를 그리고 있다. 이 실험에서 우리는 수신율 80%로 설정하였고 수신율이 80%에 못 미치는 경우에 알고리즘 1-4와 같은 방법

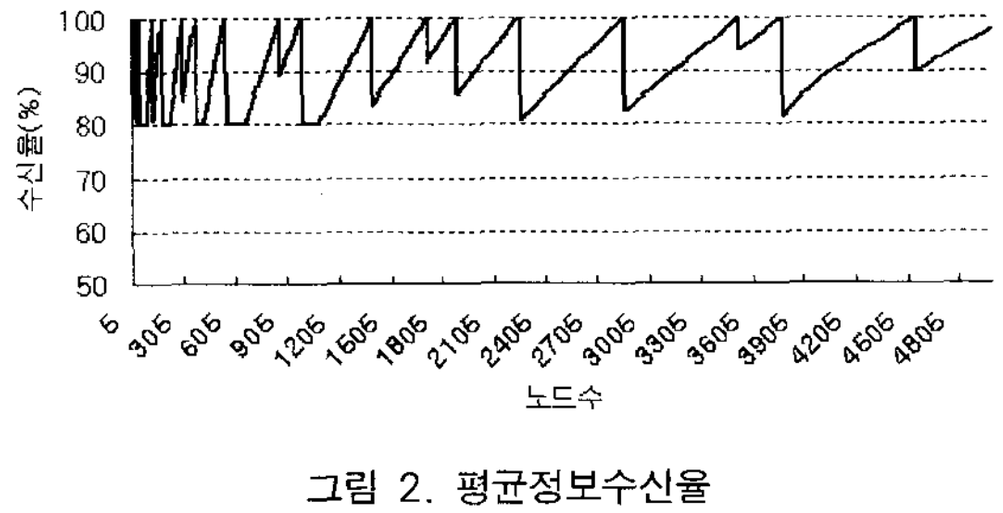


그림 2. 평균정보수신율

으로 특정 블록에 원소를 추가하여 80%이상의 수신율을 가지도록 하였다.

그림 3은 그림 2와 같은 평균 수신율을 가질 때 각 노드의 정보수신율에 대한 표준편차를 나타낸다. $w=p^2+p+1$일 때, 표준편차는 0이 되고 삭제되는 불럭의 수가 증가함에 따라 표준편차 차차 증가하는 경향을 보이는 것을 알 수 있다. 하지만 표준편차의 크기는 평균 수신정보 수의 5%보다 낮은 것으로 나타났다. 이는 정보가 특정 노드에 편중되지 않았음을 의미한다.

그림 4는 그림 2와 같은 수신율을 가질 때 노드들의 평균통신오버헤드를 나타낸 것이다. 여기에서 보는 바와 같이 이상적인 경우의 수인 $w=p^2+p+1$인 경우 통신량은 노드 당 $\sqrt{w}$가 된다. 그리고 삭제되는 불럭의 수가 증가함에 따라 수신율이 감소하는데 이에 따라 유사한 비율로 트래픽 오버헤드 또한 감소하는 것을 알 수 있다. 이는 제안된 결합구조를 사용한 결과가 SBIBD 기반의 부하상태정보교환과 유사한 통신량을 가짐을 의미한다. 이와 같이 평균 통신오버헤드가 크지 않으나 통신오버헤드가 일부노드에 편중되는지 알아보기 위해 트래픽오버헤드에 대한 표준편차를 구하였다.

그림 5는 각 노드의 통신량에 대한 표준편차를 보

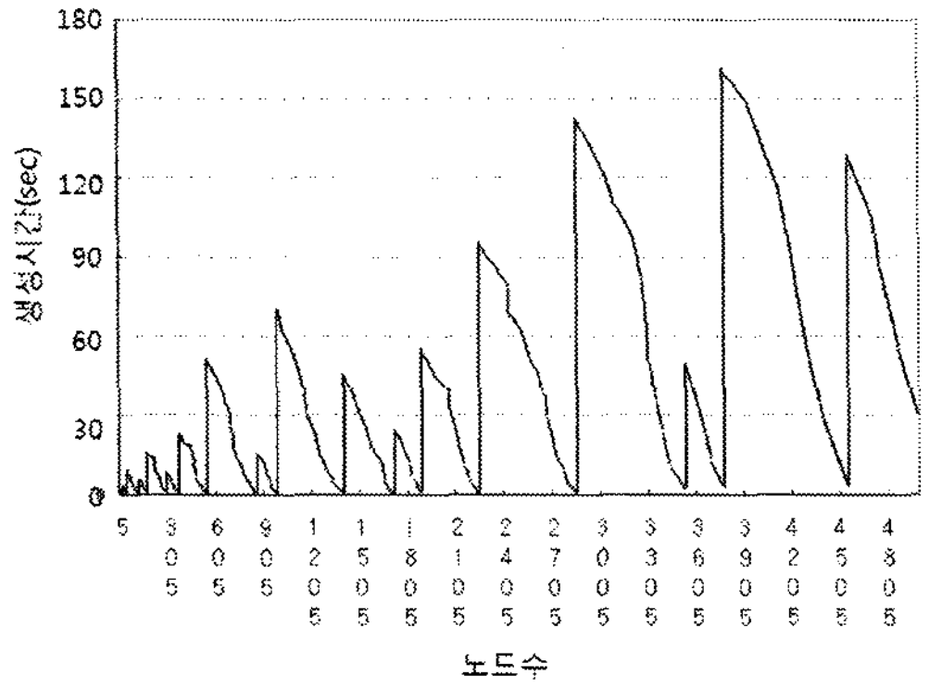


그림 1. 특수결합구조 생성시간

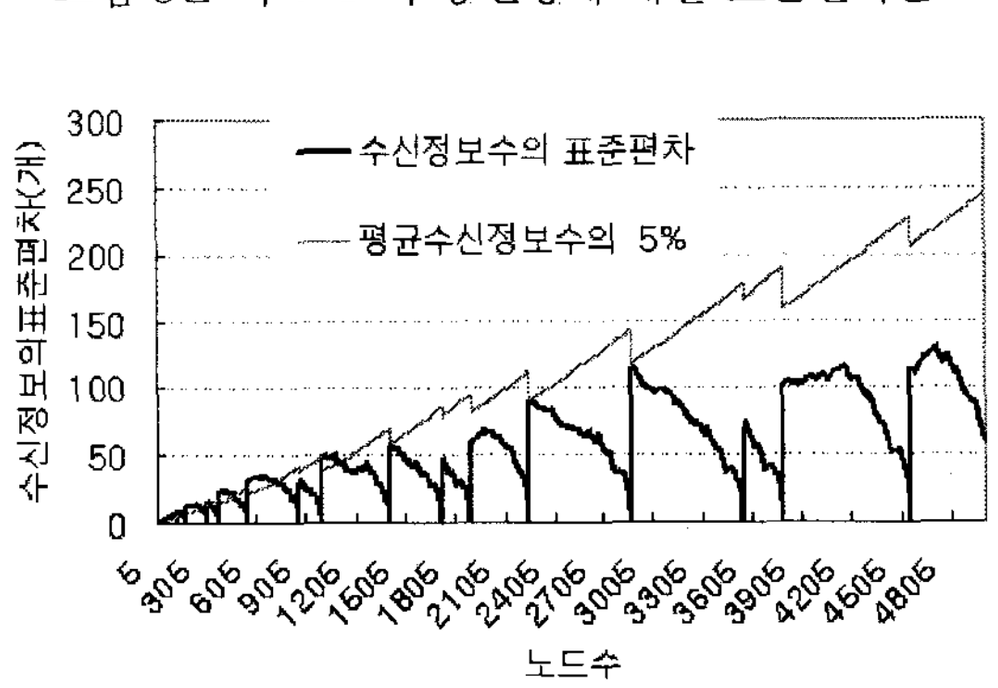


그림 3. 각 노드의 정보수신율에 대한 표준편차

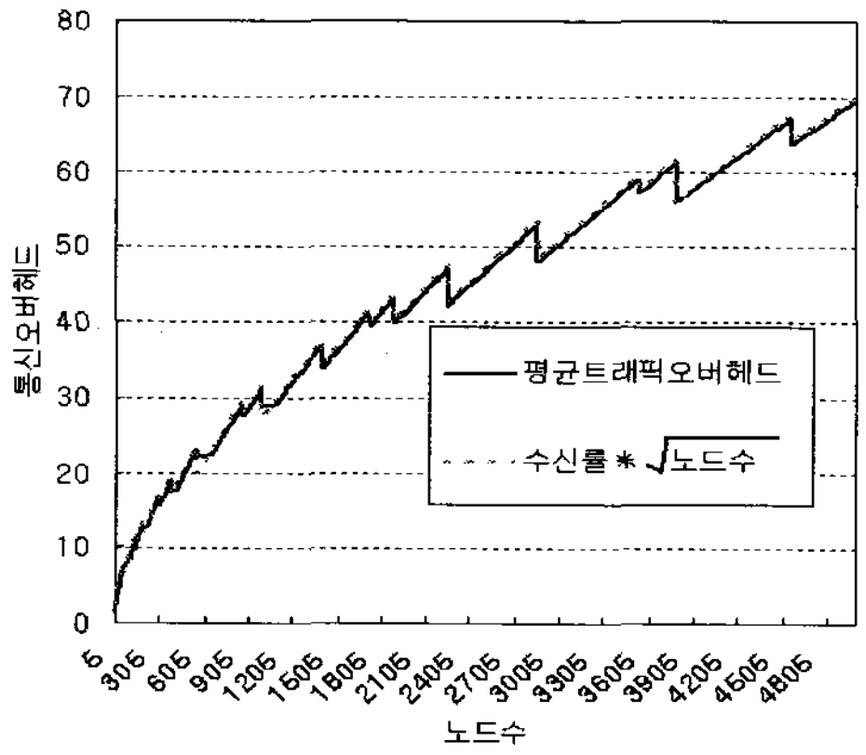


그림 4. 노드의 평균 트래픽오버헤드

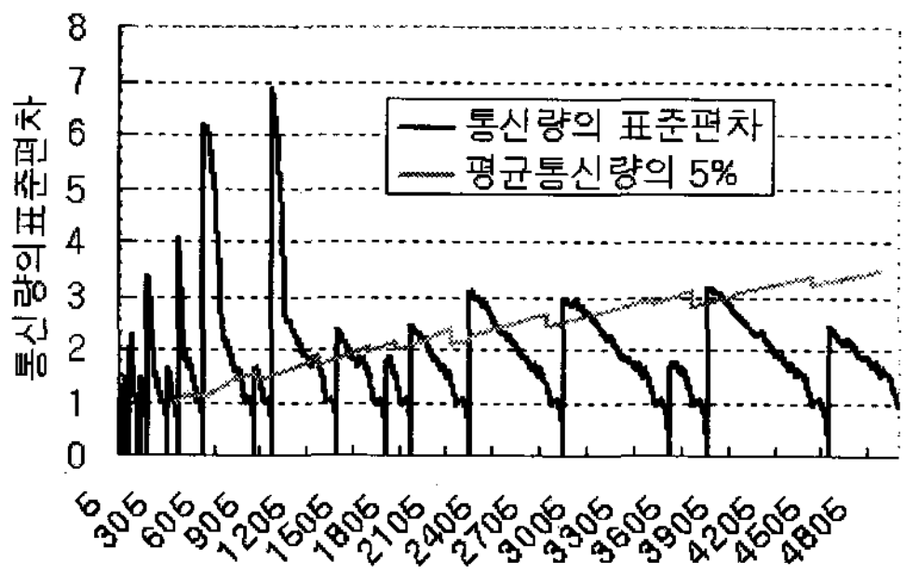


그림 5. 노드의 트래픽오버헤드에 대한 표준편차

여준다. 노드 수가 1500 이상의 경우 표준편차는 평균통신량의 5%보다 작은 것으로 나타났다. 하지만 노드의 수가 1500보다 작은 경우 일부구간에서는 통신량의 20%에까지 이르는 상당히 높은 표준편차를 가졌다.

5. 결 론

시스템의 활용도를 높이고 응답시간을 줄이기 위해서는 부하 균형이 이루어져야 한다. 부하균형을 위해서는 각 노드의 부하상태 정보를 수집할 수 있어야 하며 이 정보에 근거하여 부하균등화 정책에 따라 과 부하된 노드에서 적은 부하를 갖는 노드로의 타스크 이동이 이루어져야한다. 이 논문에서는 각 노드가 글로벌 상태정보를 얻기 위한 방법에 대하여 연구하였다. [15]에서는 SBIBD(Symmetric Balanced Incomplete Block Design)의 부분집합인 (v,p+1,1)디자인을 사용하여 2라운드 정보교환과 $v\sqrt{v}$의 트래픽 오버헤드에 의해 각 노드가 글로벌한 부하상태정보를 수신하는 방법을 제안하였다. 이 방법은 매우 효율적이지만 임의의 소수 p에 대하여 노드의 수가 $v=p^2+p+1$일 때 될 수 있다는 노드수의 제약을 가지고 있다. 이 논문에서는 네트워크상의 노드의 수가 임의의 양의 정수일 때도 이 알고리즘이 동작할 수 있도록 특수한 결합구조를 생성하였다.

이 제안의 성과를 평가하기 위해 임의의 양의 정수 w,$(5 \le w \le 5{,}000)$ 개의 노드가 존재하는 네트워크를 가정하고 각각의 경우에 2라운드 정보교환으로 최소 80% 이상의 부하상태정보를 수신하도록 실험을 수행한 결과 트래픽오버헤드는 $O(w\sqrt{w})$ 보다 낮으며 각 노드의 트래픽 오버헤드가 균등하지는 않으나 대체로 통신오버헤드는 노드들 사이에 잘 분산되는 것으로 측정되었다. 하지만 일부 구간에서 통신오버헤드의 편차가 평균 통신량의 20% 까지 증가하고 있다. 이 문제는 v와 w의 차가 클수록 심화되는 특성을 갖는다. 각 노드들에게 통신오버헤드가 잘 분산될 수 있도록 하기 위한 방법에 대한 연구가 이루어 질 필요가 있다. 또한 수신율 향상을 위한 링크 추가의 결과로 발생하는 수신정보의 중복에 대한 분석과 개선방법이 연구되어야 할 것이다.

참 고 문 헌

[1] R.Knuz, "The Influence of Different Workload Description on a Heuristic Load Balancing Scheme," IEEE Trans. on Software Eng., Vol. 17, No.7, pp. 725-730, 1991.

[2] Menno Dobber, Ger Koole, and Rob van der Mei, "Dynamic Load Balancing Experiments in a Grid," Proc. of CCGrid, pp. 1063-1070, 2005.

[3] G.D. Fitta, and M.R. Berthold, "Dynamic Load Balancing for Distributed Mining of Moleculr Structures," IEEE Trans. on Parallel and Distributed Systems, Vol.17, No.8, pp. 773-785, 2006.

[4] Jie Hu, and Raymond Klefstad, "Decentralized Load Balancing on Unstructured Peer-2-Peer Computing Grids," Fifth IEEE International Symposium on Network Computing and

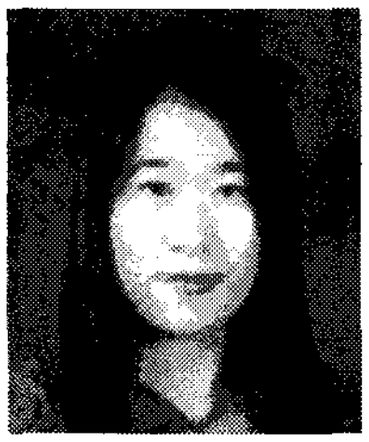
Applications, pp. 247-250, 2006.
[5] B.A. Shirazi, Scheduling and load balancing in parallel and distributed systems, IEEE Computer Society Press, 1995.
[6] M. Willebeek-Lemair, and A. P. Reeves, "Strategies for dynamic load-balancing on highly parallel computers," IEEE Trans. on Parallel and Distributed Systems, Vol.4, No. 9, pp. 979-993, 1993.
[7] Tomas L. Casavant, and Jon G.Kuhl, "Effects of response and stability on scheduling in distributed computing systems," IEEE Trans. on Software Engineering, Vol.14, No.11, pp. 1578-1588, 1988.
[8] S.H.Hosseini, B.Litow, and M.Malkawi, "Analysis of a graph coloring based distributed load balancing algorithm," Journal of Parallel and Distributed Computing, Vol.10, No.2, pp. 160-166, 1990.
[9] C.Hui, and S.Chanson "Hydrodynamic Load Balancing," IEEE Trans. on Parallel and Distributed System, Vol.10, No.11, pp. 1118-1137, 1999.
[10] M. Wu, "On Runtime Parallel Scheduling for processor Load balancing," IEEE Trans. on Parallel and Distributed System, Vol.8, No.2, pp. 173-186, 1997.
[11] K. Nam, and J. Seo, "Synchronous Load balancing in Hypercube Multicomputers with Faulty Nodes," Journal of Parallel and Distributed Computing, Vol.58, pp. 26-43, 1999.
[12] H. Rim, J. Jang, and S. Kim, "Method for Maximal Utilization of Idle links for Fast Load Balancing," Journal of KISS, Vol.28, No.12, pp. 632-641, 2001.
[13] S. Das, D. Harvey, and R. Biswas, "Adaptive Load-Balancing Algorithms Using Symmetric Broadcast Networks," Journal of parallel and Distributed Computing, Vol.62, No.6, pp. 1042-1068, 2002.
[14] S. Das, D. Harvey, and R. Biswas, "Parallel Processing of Adaptive Meshes with Load Balancing," IEEE Trans. on Parallel and Distributed Systems, Vol.12, No.12, pp. 1269-1280, 2001.
[15] O. Lee, M.Anshel, I.Chung, "Design of an efficient load balancing algorithm on distributed networks by employing symmetric balanced incomplete block design," IEE Proceedings-Communications, Vol.151, pp. 535-538, 2004.
[16] C.L.Liu, Block Designs in Introduction to Combinatorial Mathematics, McGraw-Hill, 1968.

정 일 용

1983년 한양대학교 공과대학 졸업(공학사)
1987년 City University of New York 전산학과(전산학 석사)
1991년 City University of New York 전산학과(전산학 박사)
1991년~1994년 한국전자통신연구소 선임연구원
1994년~현재 조선대학교 컴퓨터공학부 교수
관심분야 : 네트워크 보안, 전자상거래, 분산시스템 관리, 코딩이론, 병렬 알고리즘, 모바일 애드혹 네트워크

이 옥 빈

1990년 조선대학교 전자계산학과 (이학사)
1993년 조선대학교 전자계산학과 (이학석사(
2004년 충북대학교 전자계산학과 (이학박사)
2006년~현재 조선대학교 박사후 연구원
관심분야 : 분산시스템, 그리드컴퓨팅, 정보보안, 무선 센서 네트워크



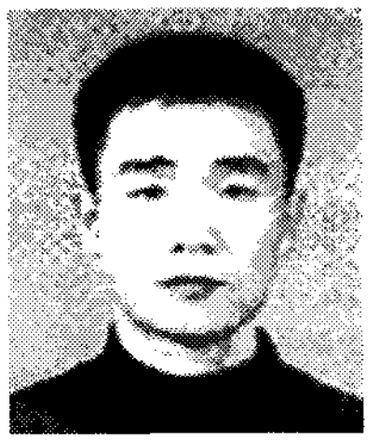
이 여 진

2000년 조선대학교 전자계산학과 졸업(이학사)

2003년 조선대학교 전자계산교육학과 졸업(교육학석사)

2006년 조선대학교 일반대학원 전자계산학과 박사수료

관심분야 : 정보보안, 네트워크보안, 전자상거래, 무선 센서 네트워크



최 동 민

2003년 경희대학교 공과대학 졸업(공학사)

2007년 조선대학교 교육대학원 정보.컴퓨터교육학과 졸업(교육학석사)

2008년 현재 조선대학교 일반대학원 컴퓨터공학과 박사과정

관심분야 : 정보보안, 무선 센서 네트워크, 애드 혹 네트워크