논문 2009-46CI-3-12

이더넷 다중 클러스터에서 GHT의 병렬 분산 구현

(Parallel Distributed Implementation of GHT on Ethernet Multicluster)

김 영 수*, 김 명 호*, 최 홍 문**

(Yeong-Soo Kim, Myung-Ho Kim, and Heung-Moon Choi)

요 약

이더넷 클러스터에서 그 분산처리 규모를 확장하려면 스위치 당 최대포트 수(현재 48포트)에 의해 물리적 제약을 받는다. 본 연구에서는 MPI기반 이더넷 클러스터에서 일반화 허프변환(generalized Hough transform: GHT)의 분산처리 규모를 확장하기 위해 다수의 이더넷 스위치들로 다중 클러스터를 구현하고, 확장에 따른 통신 부담을 병렬분산 시간분석 모델 및 통신성능 모델로 분석한 후 고속화 구현하였다. 다중 클러스터 분산처리환경에서 가능한 작업분할 정책들에 대해 평가하고, 허프공간 누산기 배열분할(accumulator array partitioning: AAP)정책을 수정 적용하여 노드간의 통신회수와 통신시간을 최소화하였고, 노드 수의 증가에 따라 AAP 정책의 분할 데이터 범위를 크게 하고 그에 부합하는 부하균형 알고리즘도 구현하였다. 단일링크 병목을 갖는 클러스터간(intercluster) 통신지연을 최대한 줄이기 위하여 일감 분배에는 선형 파이프라인 방송을 사용하고, 작은 결과 메시지들의 수합(gathering)에는 선형 플랫트리(flat tree)를 사용함으로써 총체적으로 계산과 통신을 최대한 시간 중첩시켰다. 제안한 병렬분산 GHT를 이더넷 다중 클러스터 상에서 그 성능을 점근해석하고 실험하여, 4개 고속 이더넷 스위치로 128 노드의 MPI 기반 다중 클러스터를 구현하여 거의 선형에 가까운 속도제고율(speedup)을 확인하였다.

Abstract

Extending the scale of the distributed processing in a single Ethernet cluster is physically restricted by maximum ports per switch. This paper presents an implementation of MPI-based multicluster consisting of multiple Ethernet switches for extending the scale of distributed processing, and a asymptotical analysis for communication overhead through execution-time analysis model. To determine an optimum task partitioning, we analyzed the processing time for various partitioning schemes, and AAP(accumulator array partitioning) scheme was finally chosen to minimize the overall communication overhead. The scope of data partitioned in AAP was modified to fit for incremented nodes, and suitable load balancing algorithm was implemented. We tried to alleviate the communication overhead through exploiting the pipelined broadcast and flat-tree based result gathering, and overlapping of the communication and the computation time. We used the linear pipeline broadcast to reduce the communication overhead in intercluster which is interconnected by a single link. Experimental results shows nearly linear speedup by the proposed parallel distributed GHT implemented on MPI-based Ethernet multicluster with four 100Mbps Ethernet switches and up to 128 nodes of Pentium PC.

Keywords : GHT, MPI, 다중 클러스터, 병렬처리, 속도제고율

I. 서 론

GHT는 투영기반(projection based) 물체 검출 알고리즘으로서 잡음에 강하며, 다른 물체에 겹쳐지거나 일부가 가려진 물체까지도 검출할 수 있어서 공업용 검사[1], 지문정합[2], 위성영상 기반 물체탐지[3], 의료용 영상정합[4] 및 세그먼테이션[5] 등에 널리 사용되는 유용한 알고리즘이다. 그러나 GHT는 다양한 크기 변화나 회전된 물체까지 검출하기 위한 복잡과정 때문에 매우 계산집약적인 알고리즘이다. 이에 대규모 연산처리를 해결하기 위한 고속 병렬처리의 HT (Hough transform) 및 GHT에 관한 연구들[6~11]이 많이 진행되어 왔다.

기존 연구들 중 SIMD 메쉬[6], SIMD 하이퍼큐브[7],

* 정회원, 영남이공대학 컴퓨터정보계열
(Div. of Computer Information, Yeungnam College of Science & Technology)
** 정회원, 경북대학교 전자전기컴퓨터학부
(School of Electrical Engineering & Computer Science, Kyungpook National University)
※ 이 논문은 2008학년도 영남이공대학 연구조성비 지원에 의하여 연구되었음.
접수일자: 2008년3월22일, 수정완료일: 2009년5월4일

SIMD 피라미드[8] 등은 실제 구현 없이 알고리즘만 제안한 연구들로서, 영상을 실제 프로세서 어레이에 분할 할당하는 병렬처리를 전제로 한 바, 실제로는 영상크기에 따른 프로세서 어레이구성에 제약을 받는 한계점이 있으며, 이론적인 토폴로지에 의존하고 있으므로 실제 구현에는 여러 가지 해결되어야할 문제들이 남아 있다.

또한 Encore Multimax 520[9]와 같은 MIMD 공유메모리 다중프로세서 상에서 다양한 영상 분할 정책들과 누산기 분할 정책이 구현될 수 있음을 보였으나, 밀결합 공유메모리 구조의 병목현상으로 인하여 병렬 HT의 성능 개선에 있어서 노드 수에 제약이 있음을 보였다. MIMD 다중컴퓨터 CM-5[10]에서는 영상을 임의의 프로세서 어레이에 병렬분할 처리하는 정책을 사용하여 GHT를 구현하였으나, 프로세서 어레이 구성의 최적화가 어려워 성능개선에 한계가 있음을 보였다.

한편 AP100[11]과 같은 MIMD 분산메모리 다중프로세서 상에서는 HT의 병렬구현에서 누산기 분할정책과 영상 분할정책을 실험한 결과, 누산기 분할 정책이 더 나은 성능향상을 보였다. 그러나 이들 시스템들[10~11]은 방송(broadcast), 취합(reduction) 및 동기화(barrier)와 같은 협동연산(cooperative operation)을 위한 고속의 전용 상호접속망과 전용 메시지전달 라이브러리에 기반하고 있다. 이에 저자는 이더넷 단일클러스터를 사용한 지난 연구[14]에 이어, 본 연구에서는 여러 개의 클러스터로 확대하고 성능 분석을 하였다. 단일 이더넷 스위치에 접속 가능한 최대 포트수가 48포트로 제한되어 있어서 그 이상의 분산처리 규모로 확대하고자 할 때 필연적으로 따르는 다중 스위치를 이용한 다중 클러스터의 구성과 그에 따른 통신부담의 해석과 그 감축방법, 그리고 부하균등을 전제로 한 최선의 태스크분할 방식 및 최선의 통신방식 등에 대한 연구가 이루어져야 하겠다.

본 연구에서는 MPI기반 이더넷 단일 클러스터에서 일반화 허프변환의 분산처리 규모를 확장 구현하기 위해 다수의 이더넷 스위치들로 다중 클러스터를 구현하고, 확장에 따른 통신부담을 병렬분산 및 통신성능 모델로 시간분석한 후 고속화 구현하였다. 다중 클러스터의 분산처리환경에서 다양한 작업분할 정책과 그 스케줄링 알고리즘에 대해 평가하고, 최선의 작업분할 정책으로 허프공간 누산기배열 분할처리 정책을 개선 적용하여 노드간 통신회수와 통신시간을 최소화하고, 증가된 노드 수에 부합하는 부하균형 알고리즘을 구현하였다. 이웃 클러스터로의 메시지 전송시점에서 추가되는 스위칭지연은 전체 통신지연에 비해 매우 적음을 확인하고, 단일링크 병목을 갖는 클러스터간(intercluster) 통신지연을 최대한 줄이기 위하여 일감 분배에는 선형 파이프라인 방송(linear pipeline broadcast)방식을 사용하고, 작은 결과 메시지들의 수합에는 선형 플랫트리(flat tree) 수합방식을 사용함으로써 다중 클러스터에서도 총체적으로 계산과 통신을 최대한 시간 중첩시켰다. 제안한 병렬분산 GHT를 이더넷 다중 클러스터 상에서 그 성능을 점근해석한 후 실험하여, 4개 고속이더넷스위치로 128 노드의 MPI 기반 다중 클러스터를 구현하였을 때 단일 클러스터에서와 같이 거의 선형에 가까운 속도제고율(speedup)을 확인하였다.

II. 다중 클러스터를 위한 병렬분산 GHT

1. 다중 클러스터 토폴로지

본 논문에서는 GHT의 분산처리 규모를 확장하기 위하여 그림 1과 같이 2 계층 구조의 MPI기반 이더넷 다중 클러스터를 구현하였다.

그림 1에서 보는 바와 같이 본 연구에서는 통신지연과 처리능력이 동일한 FE (Fast Ethernet; 100Mbps) 스위치와 동질의 PC들로 구성된 4개의 단일 클러스터들로 다중 클러스터를 구성하였다. 다중 클러스터 전체에 걸쳐서 한 대의 PC를 마스터(master) 노드로 하고, 나머지 PC들을 워커(worker) 노드로 하였다. 마스터 노드는 마스터 프로세스와 워커 프로세스를 같이 수행한

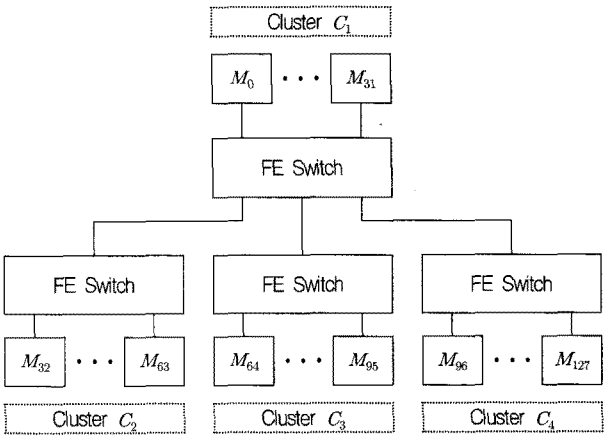


그림 1. GHT의 병렬분산 구현을 위한 다중 클러스터의 네트워크 토폴로지

Fig. 1. Network topology of the Multicluster for the proposed parallel distributed GHT implementation.

다. 마스터 프로세스는 작업(task)을 초기화하고, 워커 프로세스들에게 일감(workload)을 분배하고, 일감 처리 결과를 수합하여 종합 결과를 산출하는 등 전체 작업을 관리하는 일을 담당한다. 워커는 마스터로부터 각자의 일감을 분배받아, 상호 독립적으로 일감을 처리하고 그 결과를 마스터에게 전달하도록 하였다.

2. 다중 클러스터에서의 작업 분할 정책

다중 클러스터에서의 작업 분할은 다음과 같이 하였다. GHT의 병렬 분산처리 모델은 지난 연구[14]에서 제안한대로 T_{pre}로 표현했던 전처리과정을 제외하고, 전처리 이후의 과정들을 차례로 나열하면, T_{br}로 표현했던 p개의 워커들로의 일감방송 과정, T_{ght}로 표현했던 $(1/p)$분할 GHT연산 및 지역최고치 병렬분산 계산과정, T_{ga}로 표현했던 지역최고치(local peak) 수합과정, 마지막으로 T_{gpk}로 표현했던 마스터의 전역최고치(global peak) 계산과정으로 구성된다. 그러므로 전처리 과정을 제외하고, 어떤 작업분할 정책도 통식방식도 적용하지 아니한 순수한 병렬 GHT 수행시간을 T_{pght0}라고 한다면, 이는

$$T_{pght0} = T_{br} + T_{ght} + T_{ga} + T_{gpk} \quad (1)$$

로 표현될 수 있으며, $(1/p)$분할 GHT연산 및 지역최고치 병렬분산 계산시간인 T_{ght}를 줄이기 위하여, 여러 작업분할 처리정책들을 세우고, 다중 클러스터에서 노드간 통신회수와 통신지연을 최대한 줄이는 정책을 선택해야 한다.

반복연산 분석에 의한 순차처리 GHT의 계산 복잡도는 O($EKSR$)이다. 템플릿 물체정보 K를 분할처리 대상에서 배제하면, 목표영상의 에지 수 E을 분할하는 영상 분할처리 정책과 허프공간의 스케일 양자화 수 S 또는 회전각 양자화 수 R을 분할하는 누산기배열 분할처리 정책이 있을 수 있다.

지난 연구[14]에서는 이들 두 가지 정책들에 대하여 통신 및 계산 복잡도 측면에서 거시적으로 비교 분석하였지만, 다중 클러스터 분산처리 환경에서 통신과 계산을 포함한 전체처리시간 측면에서 상세히 분석하였다. 다중 클러스터에서는 노드 수 증가에 대처하기 위하여 기존의 단일 클러스터에서와는 달리 S 또는 R 분할 대신에 그림 2에서와 같이 $(S\times R)$을 분할할 수 있도록 누산기배열 분할처리 정책을 개선 적용하였고, 또한

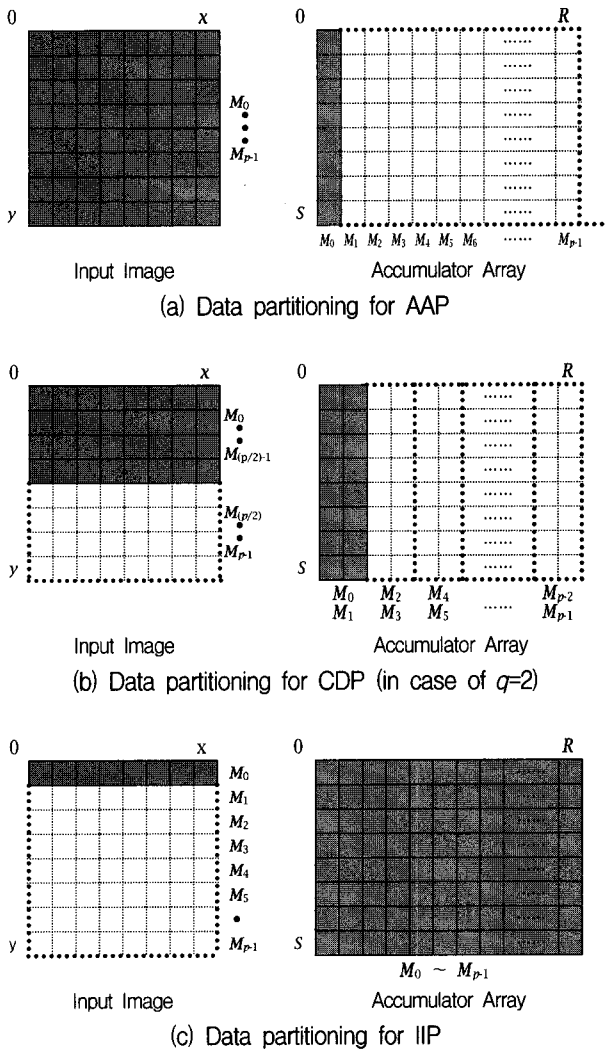


그림 2. 제안된 병렬분산 GHT를 위한 데이터 분할 정책

Fig. 2. Data partitioning schemes for the proposed parallel distributed GHT.

목표영상의 E와 누산기 배열의 $(S\times R)$을 조합하여 분할할 수 있는 새로운 데이터 조합분할 처리정책을 분석대상에 추가하였다.

(1) AAP(accumulator array partitioning) 정책

첫째, 허프공간 H(X, Y, S, R)의 누산기 배열을 분할처리하기 위하여 그림 2(a)와 같이 $(S\times R)$을 분할하는, 기존 AAP정책과 다른 클러스터를 위한 AAP정책을 수립할 수 있다. 이 정책에서는 각 노드는 전체 영상에 대하여 자기에게 할당된 $(S\times R)/p$에 대한 투표계산과 함께 지역 최고치 계산을 병행한다. 그 후에 그림 3(d)와 같은 지역 최고치들의 수합 과정을 진행한다. AAP 정책을 사용하는 경우 식(1)에서 (1/p)분할 GHT

및 지역최고치를 계산하는 시간 T_{ght}는

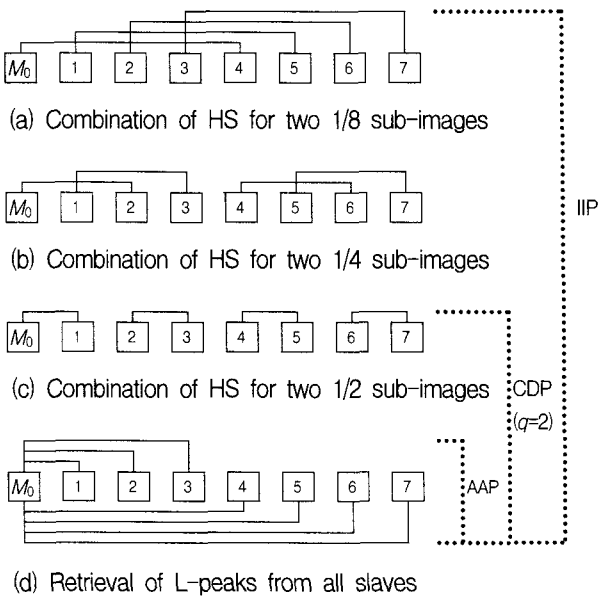
$$T_{ght} = ((E \times K \times \alpha) + \beta) \times \frac{S \times R}{p} \quad (2)$$

으로 표현할 수 있다. 여기서, α는 특정한 S_i와 R_j에서 하나의 투표를 계산하는 데 걸리는 시간, β는 특정한 S_i와 R_j에서 지역 최대치를 찾는데 걸리는 시간이다.

(2) CDP(combinational data partitioning) 정책

둘째로는 목표영상 에지 E 및 허프공간 $(S \times R)$의 분할을 조합하는 CDP정책을 수립할 수 있다. 만약 $1 < q < p$이고 2의 누승수인 q를 분할인수라고 한다면, 목표영상 에지 E를 q로 분할하면 각 노드는 $(1/q)$분할 영상을 처리하게 되며, 동시에 각 노드는 허프공간 $(S \times R)$은 (p/q)로 분할된 $q \times (S \times R)/p$의 허프공간을 처리하게 된다. 분할 인수 $q = 2$인 경우의 분할영상과 분할 허프공간을 그림 2(b)에 나타내었다. 1/2로 분할된 서로 다른 영상을 M_0에서 $M_{(p/2)-1}$까지 절반의 노드들과 $M_{(p/2)}$에서 M_{p-1}까지 절반의 노드들에게 각각 할당하고, AAP에서의 허프공간 분할의 2배 분량인 $2 \times (S \times R)/p$를 각 노드에게 할당한다.

CDP정책에서는 GHT 계산과정을 투표계산과 집계계산의 두 단계로 나눈다. 1단계에서 각 노드는 $(1/q)$분할 영상과 $q \times (S \times R)/p$분할 허프공간에 대하여 H(X, Y, S, R)에 투표계산을 완료한다. 그 후 2단계에서는 두 이웃노드 간에 누산기 배열들을 $\log_2 q$ 회수만큼 서로 교환 합산한 후에 각 노드는 완전한 누산기 배열들을 가진다. 그 후에 각 노드는 누산기 배열들을 $(1/p)$분할하여 지역 최고치를 계산한 후 수합 과정을 진행한다. 그림 3(c)에서는 p=8, q=2인 경우에 누산기 배열들의 교환 합산 과정을, 그림 3(d)에서는 완전한 누산기 배열들을 (1/8)분할하여 지역 최고치를 계산하고 이들을 수합하는 과정을 보였다.



(a) Combination of HS for two 1/8 sub-images

(b) Combination of HS for two 1/4 sub-images

(c) Combination of HS for two 1/2 sub-images

(d) Retrieval of L-peaks from all slaves

그림 3. p=8, q=2인 경우에 대한 AAP, CDP 및 IIP에서 노드간 통신

Fig. 3. Interprocessor communication in AAP, CDP, and IIP for p=8 and q=2.

그러므로 분할인수 q로 목표영상을 분할할 때 $(1/q)$분할영상과 $q \times (S \times R)/p$분할 허프공간을 각 노드가 처리하며 $\log_2 q$ 회수의 누산기 배열에 대한 교환합산 과정이 요구되므로, CDP정책을 사용하는 경우 식(1)에서의 T_{ght}는

$$T_{ght} = ((\frac{E}{q} \times K \times \alpha) + (\gamma + \delta) \times \log_2 q + \beta) \times \frac{q \times S \times R}{p} \quad (3)$$

로 표현할 수 있다. 여기서 영상크기를 N^2이라고 한다면 γ는 N^2의 H(Xc, Yc)를 두 이웃 노드간 교환 전송하는 데 걸리는 시간, δ는 두 개의 H(Xc, Yc)들을 합산하는 데 걸리는 시간을 각각 나타낸다.

(3) IIP(input image partitioning) 정책

셋째로 목표영상 에지 E를 분할 처리하기 위하여 S와 R을 분할하지 않고 목표영상을 분할 할당하는, 기존의 IIP정책과 다른 클러스터를 위한 IIP 정책을 수립할 수 있다. IIP정책에서는 $(1/p)$분할 영상에서 완전한 영상이 될 때까지 $\log_2 p$ 회수의 허프공간의 교환 조합 및 합산하는 과정이 요구되며, 모든 노드는 합산한 완전한 누산기 배열들을 $(1/p)$분할하여 지역 최고치를 계산한 후 수합 과정을 진행해야 한다. 그러므로 IIP정책을 사용하는 경우 식(1)에서의 T_{ght}는

$$T_{ght} = ((\frac{E}{p} \times K \times \alpha) + (\gamma + \delta) \times \log_2 p + \beta) \times S \times R \quad (4)$$

로 표현할 수 있다. 그림 3(a), (b) 및 (c)는 p=8에 대한 IIP 정책에서 (1/8)분할 영상에서 완전한 영상이 될 때까지의 허프공간의 조합 및 합산 과정을 보여주며. 그 후에 그림 3(d)와 같이 모든 노드는 합산된 완전한 누산기 배열들을 $(1/p)$씩 분할하여 지역 최고치를 계산한 후 수합 과정을 진행한다.

위와 같이 본 연구에서는 다중 클러스터 상에서의 다양한 작업 분할 정책들의 T_{ght}에 대하여 모든 통신시간과 계산시간을 포함시켜 구한 식(2), (3) 및 (4)를 비교하였다. 여기서 식(4)와 같은 IIP 정책은 식(3)의 CDP 정책에서 $q = p$가 되는 특별한 경우로 볼 수 있다. IIP 정책의 식(4)에서 p는 CDP정책의 식(3)의 q보다 항상 크므로 CDP정책이 IIP정책보다 효율적이며, CDP 정책의 식(3)은, AAP정책의 식(2) 보다, $H(Xc, Yc)$를 이웃 노드간 조합 합산하기 위한 통신과 계산 시간인 $(\gamma+\delta)\times\log_2 q\times(q\times S\times R)/p$를 더 포함하고 있어 AAP정책이 제일 우수하다.

또 $(1/p)$분할 GHT연산 과정에서 AAP정책에는 노드간 통신이 필요 없지만, IIP정책과 q가 큰 경우의 CDP 정책에는 노드간 및 클러스터간 통신이 요구되므로, 클러스터간 통신의 경합문제가 발생할 수 있다. 이들을 종합하여 다중 클러스터 상에서는 통신회수 및 통신지연이 최소가 되는 AAP정책을 작업분할 처리 정책으로서 선택 사용하였다.

다중 클러스터 상에서의 병렬분산처리 GHT의 시간분석 모델은 다중 이더넷 스위치 경로에서의 스위칭 지연을 T_{br}에 포함시키면 그림 4와 같은 기존 단일 클러스터에서의 모델[14]을 그대로 활용할 수 있다.

3. 다중 클러스터 상에서의 부하 균형

S와 R의 범위 및 양자화 수는 GHT 응용 분야에 따라 적절한 값으로 설정될 수 있으나, 대체적으로 공업물체검사[1]에서 S는 0.8과 1.3사이에서 0.01 간격으로 양자화 되므로 약 50, R은 0°부터 359°사이에서 1°간격으로 양자화 되므로 360, 의료영상정합[3]에서 S는 0.76과 1.26사이에서 0.02 간격으로 양자화 되므로 26이며, R은 +90°에서 +90°까지이므로 약 180임을 밝힌 바 있다. 이에 따르면 GHT 알고리즘의 S와 R의 내포루프(nested loop)의 처리회수 $(S\times R)$은 공업물체검사의 경우 18000, 의료영상정합의 경우 4680이 되어 한 기관이 소유한 PC들에 분배 할당하기에 충분하다.

본 연구에서는 기존 AAP정책에 따라 허프공간 $H(X, Y, S, R)$의 S와 R 중에서 어느 하나를 분할 처리할 수 있지만, 다중 클러스터 기반에서는 보다 큰 값인 $(S\times R)$을 분할 처리함으로써, 증가된 노드 수 p에 의해 가급적 정수 분할이 가능하게 하였다. 이는 당연히 $(S\times R)\geq p$이라는 가정하에서다.

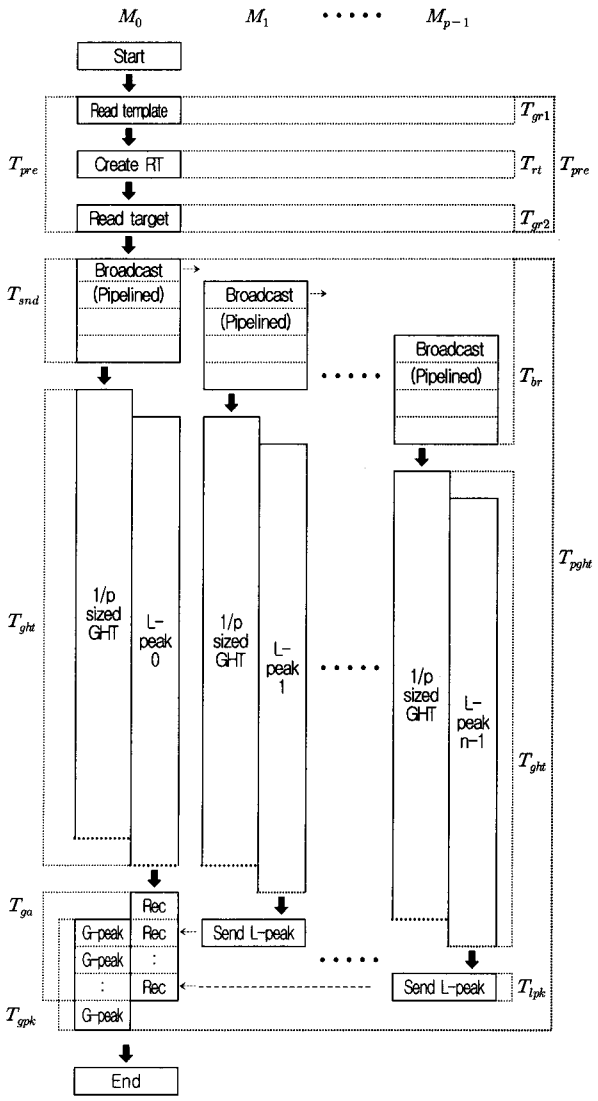


그림 4. 이더넷 다중 클러스터 상에서의 병렬분산처리 GHT의 시간분석 모델

Fig. 4. Time analysis model for the proposed parallel distributed GHT on Ethernet multicluster.

각 노드에서 부하의 균형은 S와 R 모두가 2의 누승수일 때, 즉 $S=2^a$, $R=2^b$일 때 자동적으로 이루어진다. 여기서 a와 b는 1, 2, 3, ..., $(\log_2 p-1)$ 등의 정수이다. 이 경우 $(S\times R)=2^{(a+b)}$이 되어, 2의 누승수가 되고, 이는 $(S\times R)=k\cdot p$가 되어, 2의 누승수인 p의 k 정수배가 되기 때문에 $(S\times R)$이 p로 쉽게 k 정수배로 분할되어 부하균형이 쉽게 이루어진다. 그러나 S와 R 어느 쪽도 2의 누승수가 아닐 때는 p로 $(S\times R)$을 정수배 분할하기 어려워 부하균형이 어렵다.

본 연구에서는 $(S\times R)$이 p 노드들에게 동일 정수배로 나뉘어 분배되도록 3가지 전제 조건을 가정 하였다. 첫 번째는 $R\geq S$이며, 두 번째는 R은 짝수(even

number), 세 번째로 $S=2^n$, $n=1, 2, 3, ..., (\log_2 p-1)$이다. 다음은 본 논문에서 구현한 부하균형 알고리즘이다.

```
if ((R ≥ p) && (R mod p == 0)) {
    Prepare the scale and rotation factors for dividing R by p
}
else if ((R×S) mod p == 0) {
    Find a large c for S to be divided by 2^c (2^c ≤ S)
    Prepare the scale factors for dividing S by 2^c,
     and the rotation factors for dividing R by 2^((log2 p) - c)
}
else {
    Find a large c for S to be divided by 2^c (2^c ≤ S)
    Set p1=2^c, p2=2^((log2 p) - c), where p = p1 + p2
    Find a integer r = (R mod p2) as partitioned rotation
    Find m from R = ((r+1)×m + (r×(p2 - m))
    Prepare the scale factors for dividing S by p1,
    Prepare the rotation factors for partitioning (r+1)
     degrees to m nodes, and then r degrees to (p2 - m)
     nodes
}
```

부하 균형 알고리즘에서는, 제일 먼저 R을 p로 정수 분할 시도하여 사용하지만, 만약 이 시도에서 실패하면 $(S\times R)$을 p로 적절하게 분할하여 사용한다. 그러나 $(S\times R)$을 p로 정수 분할할 수 없으면, S를 먼저 최대한 분할한 다음에 R을 분할하여 노드들에게 할당한다. 이때 그림 4에서 보는 바와 같이 확률적으로 번호가 낮은 노드들이 병렬분산 GHT수행을 더 빨리 완료할 것이므로 더 높은 부하인 $(r+1)$각을 번호가 낮은 노드들에게 먼저 배당하고, 더 낮은 부하인 r각을 번호가 높은 노드들에게 할당하였다.

4. 다중 클러스터에서의 일감 분배

본 다중 클러스터의 분산처리 환경에서는 일감 분배를 위한 통신시간을 줄이기 위하여 선형파이프라인 방송을 택하였다. 기존의 병렬처리 환경에서와는 달리 속도제고율을 높이기 위해서, 통신에 비하여 단위일감(grain size)을 최대한 크게 해야 한다. 이에 일감 방송 방식을 선정함에 있어서, 클러스터간의 단일 연결선로에 의한 연결포트 경합을 최대한 줄이고 효율적인 일감 분배를 위하여, 구성할 수 있는 파이프라인 방송 알고리즘들 중에서, 충분히 큰 방송메시지에서는 이진(binary) 트리나 3-ary 트리 파이프라인보다 더 효율적인 선형트리 파이프라인 방송방식[12]을 선택하고, 이를 초기 일감분배에 반영하였다. 여기서 초기 일감은 목표 영상과 R-table 정보를 합한 것을 의미한다.

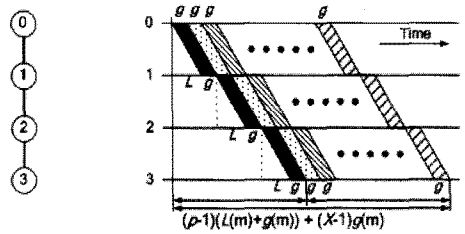
한편 선형 파이프라인 방송에서는 방송 메시지크기, 노드 수 및 세그먼트 크기가 주어지면 이론적 방송 통신시간을 산정할 수 있다. 파이프라인 방송을 위하여 $msize$바이트의 송신메시지를 $\frac{msize}{X}$바이트의 세그먼트로 X번 나누어 전송할 때, 파이프라인 방송의 통신시간 T_{br}은, $(X-p-1)\times T(\frac{msize}{X})$로 주어지며, X가 p보다 충분히 클 때 $(X-p-1)\times T(\frac{msize}{X})\approx T(msize)$[12]이 되어, 최소의 통신시간을 달성할 수 있다.

이더넷 다중 클러스터에서의 노드간 통신은 PC 외부 네트워크 및 여러 이더넷 스위치들을 경유하므로 큰 지연시간이 존재한다. 이에 본 연구에서는 이더넷 다중 클러스터 환경에서 일감 방송시간을 분석하기 위하여, 큰 방송 메시지에 적합하고 방송 메시지를 바이트 단위가 아닌 세그먼트 단위로 취급할 수 있는 통신의 성능 모델인 PLogP (Parameterized LogP) 성능 모델[12~13]을 사용하였다. 그림 5는 이더넷 다중 클러스터에서 선형 파이프라인 방송을 위한 통신의 성능 모델링이다.

이 모델에서 $L(m)$과 $g(m)$은 모두 메시지의 함수로서 $L(m)$은 m바이트 메시지의 프로세스 대 프로세스간의 지연시간 (latency)이며, $g(m)$는 연속으로 송신 또는 수신하는 m바이트 메시지들 간의 시간 간격(gap per messages)이라 할 때, 파이프라인 방송통신 시간은

$$T_{comm}=(p-1)(L(\frac{msize}{X})+g(\frac{msize}{X}))+(X-1)g(\frac{msize}{X}) \quad (5)$$

이 된다. $L(\frac{msize}{X})$은 $(\frac{msize}{X})$바이트 세그먼트의 첫 번



(a) Linear Tree (b) Total Communication Time

그림 5. 이더넷 다중 클러스터에서 파이프라인 방송을 위한 통신의 성능 모델링

Fig. 5. Performance modeling of the communication for the pipelined broadcast on the Ethernet multicluster.

째 1 bit가 송신 프로세스에서 출발하여 수신 프로세스에 도착하는 시간이므로, 이는 전파지연(propagation delay), 버퍼 복사(buffer copy), 네트워크 설정(set-up) 및 이더넷 스위치 지연 등을 포함한다. $g(\frac{msize}{X})$는 하나의 세그먼트를 전송하는 데 걸리는 시간이므로, 송신/수신 오버헤드 및 메시지 전송시간을 포함한다. 그림 5와 식(5)에서 p가 커지면 노드 M_0부터 M_{p-1}까지의 통신지연이 커지며, 충분히 큰 일감에서 X가 커지면 파이프라인 효과가 커진다. 이 모델을 사용하기 위해서는 사용할 세그먼트 크기에 대한 L과 g를 먼저 측정하여야 한다. 본 연구에서는 이를 위하여 포트들 간에 1024바이트 세그먼트에 대한 왕복(roundtrip) 시간을 측정함으로써, $L(1024)$와 $g(1024)$를 구하여 통신성능 분석에 사용하였다.

이들과 동시에 구한 이더넷스위치 내부지연 $L_{sw}(1024)$는 0.090ms이었다. 클러스터간 거리 $70m$에 대한 전파지연 T_p은 약 0.00035ms이다. 클러스터가 추가될 때, $L_{sw}(1024)$와 T_p이 전체지연시간 $(p-1) \cdot L(1024)$에 미치는 영향을 알아보기 위하여, p=128인 경우 노드M_0에서 M_{127}까지 파이프라인 전송되는 동안, $L_{sw}(1024)+T_p$이 추가되는 회수를 산정해보았다. 그림 1에서 클러스터 C_1에서 C_2로 통과할 때 1회, C_2에서 C_3으로 통과할 때 2회, C_3에서 C_4로 통과할 때 2회이다. 그러므로 C_2까지 $(L_{sw}+T_p)$, C_3까지 $3(L_{sw}+T_p)$ 및 C_4까지 $5(L_{sw}+T_p)$ 등의 추가적 지연이 있으며, 이들은 전체지연 $(p-1) \cdot L(1024)$에 비하면 매우 작은 값이어서 전체 통신지연에 영향을 거의 미치지 않음을 확인하였다.

본 연구에서는 물리적 2계층 트리 형태로 구성된 이더넷 다중 클러스터 위에서, 파이프라인 방송을 위한 경합회피(contention free) 방송트리를 구성하기 위하여 그림 1에서와 같이 클러스터 번호와 노드 번호 M_i, i = 0, 1, 2 의 순서대로 노드들을 배치하여 사용하였다.

5. 다중 클러스터 환경에서의 결과 수합

본 연구에서와 같은 다중 클러스터에서의 결과수합에는 작은 결과 메시지 수합이 기본을 이루므로 이에 가장 적합한 선형 플랫트리[15]를 사용하였다. 그림 4의 결과수합 시간 T_{ga}는 모든 워커 프로세스들이 GHT 수행을 끝내고 마스터 프로세스에게 동일한 40바이트의 결과 메시지를 전달하는 시간이다.

결과수합에서는 워커 프로세스마다 사정에 따라 일감을 받은 순서대로 결과를 출력하지 못할 경우가 발생할 수 있으므로, 최악의 경우의 T_{ga}은, 한 노드 M_i가 이웃 노드 M_{i+1}에게 차례로 결과 메시지를 전달하는 시간 $T_{i,i+1}$ 들의 합과 같다고 볼 수 있으며, 이는 T_{ga}의 상한선(upper bound)이 되며, 이를 T_{ga_upper}라고 표현한다면,

$$T_{ga\_upper} = \sum_{i=0}^{p-1} (T_{i,i+1}) \tag{6}$$

이 된다. 한편 T_{ga}의 하한선(lower bound)은, 파이프라인 방송이 M_1부터 M_{127}까지 차례로 수신 완료될 때, 노드 M_{j-1}로부터 노드 M_j가 마지막 세그먼트 1024바이트를 수신하는 데 걸리는 시간 $T_{j-1,j}$들의 합과 같다고 볼 수 있으며, 이를 T_{ga_lower}라고 표현한다면, 식(7)과 같다.

$$T_{ga\_lower} = \sum_{j=1}^{p-1} (T_{j-1,j}) \tag{7}$$

한편 파이프라인 전송과 무관하게 모든 워커 노드가 연이어 차례로 마스터 노드에게 동일한 결과 메시지를 보내는 시간을 산정하고, 이 시간이 식(7)의 T_{ga_lower}보다 작은지를 알아보았다. 클러스터C_1 내에서 하나의 수집시간을 T_c라고 한다면 $T_c = L(40)+g(40)$이다. 그러므로 클러스터 C_1에서의 수집시간 합계는 $32 \cdot T_c$가 되며, 클러스터간 통신을 감안한, 클러스터 C_2, C_3 및 C_4의 수집시간의 총합계는 $3 \cdot 32 \cdot (T_c + L_{sw}(40) + T_p)$가 되므로, 이들을 모두 합한 총수집시간은 $4 \cdot 32 \cdot T_c + 3 \cdot 32 \cdot (L_{sw}(40)+T_p) \approx 8.192$ms이 된다. 여기서 T_c는 측정값 0.055ms를, $L_{sw}(40)$은 측정값 0.012ms를 적용하였다. 파이프라인 방송과 무관하게 연속적인 총 수집시간 8.192ms는, 파이프라인 방송에 의거한 식(7)의 총수집시간인 $127 \cdot (L(1024)+g(1024)) \approx 38.354$ms보다 더 작으므로, 식(7)에 의해 수집시간이 지배됨을 알 수 있다.

6. 속도 제고율

다중 클러스터에서 속도제고에 가장 큰 영향을 미치는 처리 시간중첩은, 초기일감 방송과 GHT 연산의 시

간중첩이다. 그림 4에서 보는 바와 같이, 일감방송 완료 시간은 T_{br}이며, 노드 M_0의 일감방송 시간은 T_{snd}이므로, 일감방송과 GHT연산의 중첩시간 T_{ov_br}은 $T_{ov_br} = T_{br} - T_{snd}$가 된다. 이는 결과수집 시간 T_{ga}의 하한선을 결정하며, 노드 수 p가 커지면 이 시간도 커지므로 속도제고는 감소한다.

단일 노드에서의 순차처리 GHT 수행시간 T_{seq}는 식 (8)로 주어진다.

$$T_{seq} = T_{pre} + T_{sght} \tag{8}$$

p개의 노드들로 구성된 다중 클러스터에서의 병렬분산처리 GHT의 수행시간 T_{par}은 $T_{par} = T_{pre} + T_{pght}$로 주어진다. 여기서 T_{pght}는 그림 4에서 보는 바와 같이 전처리 과정을 제외한 병렬 GHT 수행 시간이며, 노드 M_0와 노드 M_{p-1} 양쪽에서 보았을 때 T_{pght}는 $T_{pght} = T_{br} + T_{ght} + (1/p)T_{ga} + (1/p)T_{gpk}$로 표현할 수 있다. 여기서 T_{ght}는 $(1/p)T_{sght}$로 근사할 수 있으므로, T_{pght}는

$$T_{pght} \approx T_{br} + (\frac{1}{p})T_{sght} + (\frac{1}{p})T_{ga} + (\frac{1}{p})T_{gpk} \tag{9}$$

이 된다. 식에서 $(1/p)T_{sght}$는 병렬분산 GHT의 계산복잡도가 순차처리에서 O(*EKSR*)인 것을 병렬분산처리의 경우 최악의 경우라도 O(*EKS*) 또는 O(*EKR*)로 줄인 결과이며, $(1/p)T_{ga}$와 $(1/p)T_{gpk}$는 취합통신의 복잡도를 AAP정책에서 O(1)로 줄이고 통신과 계산을 중첩시킨 결과이다. 식에서 T_{br}은 일감방송 시간이며, PLogP 모델을 적용한 식(5)의 T_{comm}을 의미한다. 따라서 제안한 병렬분산 GHT의 속도제고율 Sp는

$$Sp = \frac{T_{sght} + T_{pre}}{(\frac{1}{p})T_{sght} + T_{pre} + T_{br} + (\frac{1}{p})T_{ga} + (\frac{1}{p})T_{gpk}} \tag{10}$$

로 정의된다. 충분한 일감이 주어진 상태에서는 T_{sght}에 비해 T_{pre}, T_{br}, T_{ga} 및 T_{gpk} 등은 아주 작은 값이므로 속도제고율 Sp는 점근해석 (asymptotic analysis)하여 $Sp = O(p)$로 근사화 될 수 있다. 따라서 제안한 병렬분산 GHT 알고리즘은 어느 한계의 p까지는 거의 선형적으로 p에 비례하는 속도제고율을 기대할 수 있다.

III. 실험 및 고찰

본 실험에서는 3.4 GHz Pentium 프로세서와 1GB 메모리를 가진 HP DX7300 PC를 사용하였다. 모든 PC들은 32-bit Windows XP Professional를 사용하고, 각 PC의 이더넷카드로는 100Mbps 속도와 full duplex 통신모드를 가진 Intel 82566DM를 사용하였다.

모든 클러스터의 스위치들은 CISCO Catalyst WS-C2960-48TC-L (48 Ethernet 10/100Mbps ports, 2 dual-purpose uplinks (one 10/100/1000Mbps Ethernet port, and one SFP-based Gigabit Ethernet port))이며, 이는 축적전송(store & forward) 스위칭모드와 full duplex 통신모드로 동작한다. 그림 1에서 모든 클러스터들은 지역적으로 떨어져 위치하며, 클러스터 2, 3 및 4의 이더넷스위치들은 클러스터 1의 이더넷스위치와 각각 70 m의 UTP CAT-6 케이블로 연결되어 있다. MPI 미들웨어(middleware)는 MPICH 2를 사용하였으며, 이와 관련된 관리도구들을 사용하였다. 병렬분산 GHT 알고리즘을 위한 응용프로그램은 Microsoft Visual C++로 구현하였다.

실험에 사용된 영상들은 템플릿과 목표 영상 각각에 대하여 128×128, 256×256, 512×512 인공영상 (artificial image)들과 512×512 실영상(real image)이다. 단일 클러스터의 연구결과와 현재의 실험결과를 비교하기 위하여 이전 연구와 동일한 영상들[14]과 동일한 영상부하를 사용하였다.

표 1은 p=16, 32, 64, 128인 경우 512×512영상에서의 일감크기 272K바이트에 대한 방송시간의 이론치와 실측치의 비교 결과이다. 이는 PLogP 모델에 의거 1024바이트 크기의 세그먼트에 대한 $g(1024)$=0.095ms, $L(1024)$=0.207ms의 측정결과를 기반으로 하였으며, 실측치와 이론치가 매우 근사함을 보였다. 이때 메시지간 시간간격 $g(1024)$=0.095ms는, 세그먼트 1024바

표 1. 512×512 목표영상의 일감방송 시간의 이론치 및 실측치의 비교

Table 1. Comparison of theoretical and measured values for the time of broadcasting the workload of the 512×512 target image. [msec]

T_{comm} \ p	16	32	64	128
Theoretical value	30.275	35.107	44.771	64.099
Measured value	30.484	35.408	44.902	64.735

표 2. 512×512 목표영상의 일감에 대한 세그먼트 크기에 따른 T_{br} 측정값

Table 2. Measured value of T_{br} for varying segment size in the workload of 512×512 target image.[msec]

Seg_size \ p	16	32	64	128
256	62.605	64.028	75.588	96.719
512	34.692	39.429	48.826	72.238
1024	30.484	35.408	44.902	64.735
2048	30.829	37.619	50.434	79.191
4096	34.012	43.215	62.631	116.875

이트에, IFG 12바이트, 프리앰블 8바이트, 프레임 헤더+트레일러 18바이트, IP헤더 20바이트 및 TCP헤더 20바이트를 추가한 실제 프레임 길이 1102바이트에 대한 시간이므로, 트리구조의 다중클러스터의 최대 bandwidth는 92.8Mbps이었다.

표 2는 512×512영상에서의 일감크기 272K바이트에 대한 세그먼트 크기 256, 512, 1024, 2048, 4096바이트에 대한 T_{br}의 측정값이다. 실험 결과 세그먼트 크기가 작을 때는 노드 CPU의 통신 부담이 늘어나고, 클 때는 MTU (maximum transmission unit)에 의해 노드 CPU의 세그먼테이션/조립 부담이 늘어나므로, 세그먼트 크기가 1024일 때 가장 작은 T_{br} 값을 보였다.

표 3에는 각 영상크기 및 노드 수의 변화에 따른 GHT의 전체 실행시간을 나타내었다. 각 PC에 별도의 작업부하가 없는 실험실 환경에서 실험하였으므로 전체적으로 3회의 실험에서 실험편차가 거의 없었다.

그림 6은 구현한 병렬분산 GHT에 대한 실험 결과 속도제고율을 그래프로 나타내었다. 노드 수가 증가함에 따라 통신시간이 증가하며 단위일감이 줄어들어 어

표 3. 노드 수 변화에 따른 제안된 병렬분산 GHT의 실행 시간 [초]

Table 3. Execution time of the proposed parallel distributed GHT for varying number of processors [sec]

Category	Artificial			Real
Size \ p	128×128	256×256	512×512	512×512
1	20.320	80.486	320.249	492.293
2	10.167	40.246	160.125	246.147
4	5.086	20.144	80.188	123.103
8	2.555	10.086	40.156	61.702
16	1.318	5.189	20.627	31.703
32	0.721	2.799	11.054	16.850
64	0.393	1.462	5.692	8.592
128	0.215	0.735	2.796	4.229

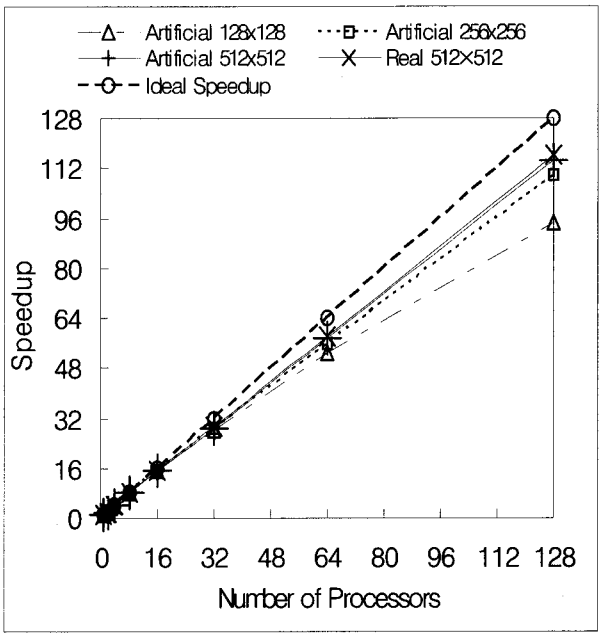


그림 6. 128 노드를 가진 다중 클러스터에서 병렬분산 GHT 알고리즘에 대한 속도제고율

Fig. 6. Speedup for the parallel distributed GHT algorithm on multicluster with 128 nodes.

느 영상에서나 속도제고율은 감소추세를 보였다. 한편 영상크기가 클수록 속도제고율이 커짐은 통신에 비하여 상대적으로 단위일감이 커지기 때문에 파이프라인 방송이 통신부담을 줄여 노드 수 증가에도 속도제고율의 감축율이 꽤 작은 것으로 보인다.

512×512 실영상은 동일한 크기의 인공영상보다 목표영상의 에지 점들의 수가 46% 더 많지만, 단일 클러스터에서와 같이 다중 클러스터에서도 속도제고율은 거의 같은 성능을 보였다.

다중 클러스터에서는 노드의 CPU 속도가 지난 연구[14]인 단일 클러스터에서보다 13% 높으나, 32 노드까지의 비교에서 거의 같은 속도제고율을 보였다.

IV. 결 론

2계층 트리구조로 구성된 100Mbps 이더넷 다중 클러스터 상에서 MPI 기반 병렬분산처리 GHT를 구현하여 모델링하고 시간 분석한 후 고속화하여 구현하였다. 다중 클러스터들을 연결하는 단일 연결선로의 병목현상을 해결하기 위하여 충분히 큰 전송 메시지에 적합한 선형 파이프라인 방송방식을 채용하고, 수합에는 작은 결과메시지에 적합한 선형 플랫트리 수합을 채용하여 통신지연을 최대한 줄였고, 다양한 작업분할 정책들을

분석한 후, 통신회수와 통신시간을 최대한 줄이기 위하여 AAP 정책을 수정 적용하고 그에 대한 부하균형 알고리즘을 구현하였다. 실험결과, 4개 이더넷 고속 스위치와 128 노드로 구성한 다중 클러스터에서 제안된 병렬분산 GHT는 512×512 크기의 실영상에 대하여 *Sp*=116.409까지의 속도제고율을 얻었으며, 클러스터 컴퓨팅에서도 원만한 병렬분산처리가 가능함을 확인하였다.

참 고 문 헌

[1] Beinglass, A. and Wolfson, H. J., "Articulated Object Recognition, or: How to Generalize the Generalized Hough Transform", *Proc. of IEEE CVPR,* pp. 461-466, June 1991.

[2] Chaoqiang Liu, Tao Xia, and Hui Li, "A Hierarchical Hough Transform for Fingerprint Matching", *Lecture Notes in Computer Science*, vol. 3072, 2004.

[3] Fujii, K. and Arikawa, T., "Urban object reconstruction using airborne laser elevation image and aerial image", *IEEE Trans. on Geoscience and Remote Sensing*, vol. 40, Iss. 10, pp. 2234-2240, Oct. 2002.

[4] Chmielewski and Leszek, "Choice of the Hough transform for image registration", *Proc. of the SPIE*, vol. 5505, pp. 122-134, 2004.

[5] B. Howe, A. Gururajan, H. Sari-Sarraf, and L. R. Long, "Hierarchical segmentation of cervical and lumbar vertebrae using a customized generalized Hough transform and extensions to active appearance models", *6th IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 182-186, March 2004.

[6] C. Guerra and S. Hambrusch, "Parallel algorithms for line detection on a mesh," *Journal of Parallel and Distributed Computing*, vol. 6, no. 1, pp. 1-19, Feb 1989.

[7] Y. Pan and Y. H. Chuang, "Parallel Hough transform algorithms on SIMD hypercube arrays," *Proc. of ICPP*, vol. 3, pp. 83-86, Aug. 1990.

[8] M. Atiquzzaman, "Pipelined implementation of the multiresolution Hough transform in a pyramid multiprocessor," *Pattern Recognition Letters*, vol. 15, no. 9, pp. 841-851, Sep. 1994.

[9] A. N. Choudhary and R. Ponnusamy, "Implementation and evaluation of Hough algorithms on a shared-memory multiprocessor," *Journal of Parallel and Distributed Computing*, vol. 12, no. 2, pp. 178-188, June 1991.

[10] D. Baumann and S. Ranka, "The Generalized Hough Transform on an MIMD Machine," *Journal of Undergraduate Research in High-Performance Computing*, 2, 1992

[11] A. Underhill, M. Atiquzzaman, and J. Ophel, " Performance of the Hough transform on a distributed memory multiprocessor," *Microprocessors and Microsystems*, vol. 22, no. 7, pp. 355-362, Jan. 1999.

[12] P. Patarasuka, X. Yuan, and A. Farajb, "Techniques for pipelined broadcast on ethernet switched clusters ", *Journal of Parallel and Distributed Computing*, vol. 68, Iss. 6, pp. 809-824, June 2008.

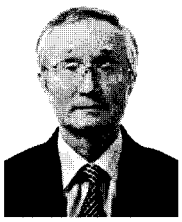
[13] T. Kielmann, H. E. Bal, and K. Verstoep, "Fast Measurement of LogP Parameters for Message Passing Platforms," *Proc. of IPDPS Workshop on Parallel and Distributed Processing*, pp. 1176-1183, May 2000.

[14] Y. S. Kim, J. S. Kim, and H. M, Choi, "Parallel Distributed Implementation of GHT on MPI-based PC Cluster", *Journal of IEEK*, vol. 44-CI, no. 3, May 2007.

[15] S. S. Vadhiyar, G. E. Fagg, and J. J. Dongarra, "Towards an Accurate Model for Collective Communications", *International Journal of High Performance Computing Applications*, Vol. 18, No. 1, pp. 159-167, 2004.

저 자 소 개

김 영 수(정회원)
1980년 경북대학교 전자공학과 학사 졸업
1982년 경북대학교 전자공학과 석사 졸업
1999년 경북대학교 전자공학과 박사 수료
1986년~현재 영남이공대학 컴퓨터정보계열 교수
<주관심분야 : Parallel Distributed Processing, Computer Networks, Ubiquitous & Embedded System>



김 명 호(정회원)
1969년 서울대학교 섬유공학과 학사 졸업
1981년 영남대학교 전자공학과 석사 졸업
1999년 대구카톨릭대학교 컴퓨터공학과 박사 수료
1983년~현재 영남이공대학 컴퓨터정보계열 교수
<주관심분야 : Distributed Operating Systems, Parallel Computing, Concurrent Programming>

최 흥 문(정회원)
대한전자공학회 논문지
제39권 CI편 제1호 참조
현재 경북대학교 전자전기컴퓨터학부 교수