# 대규모 중력장 다체계 전산모사를 위한 새로운 트리코드

안철오[0] 이상환
한양대학교 기계공학과 병렬연산연구실, 한양대학교 기계공학부
coahn@hanyang.ac.kr shlee@hanyang.ac.kr

# A New Treecode for Massive Gravitational N-Body Simulation

Cheol O Ahn[0] Sang Hwan Lee
Parallel Computing Lab., Mechanical Eng., Hanyang University
School of Mechanical Eng., Hanyang University

물리학에서의 많은 문제는 해석하고자 하는 계를 수많은 입자들의 집합체로 모델링하여 이들의 움직임과 상호작용을 연구하는데, 이러한 연구방법을 흔히 다체계 문제라고 부른다. 이와 같은 다체계 문제는 기본적으로 모든 개개의 입자가 자신을 제외한 다른 모든 입자와 가지는 상호 작용을 연산해야 하기 때문에, 입자의 수(N)가 증가함에 따라 연산 횟수는 그 제곱에 비례한다는데 그 어려움이 있다. 이러한 점을 극복하기 위한 방법은 계층적 트리데이터 구조를 이용하는 고속화 알고리즘을 사용하는 것으로서, 이러한 방법을 보통 트리코드로 부른다. 이는 공간에 분포하고 있는 입자들을 그 위치정보에 근거하여 계층적 트리 데이터 구조를 만들고, 이를 연결 리스트로 구성하여 입자계 상호작용 연산에 이용하는 것으로, 멀리 떨어진 입자들의 덩어리 효과를 멱급수의 초기 일부 항만을 이용하여 계산함으로써 연산량을 $O(N^2)$에서 $O(NlogN)$으로 감소시킨다. 다체계 문제의 고속 해석기술에 있어서 Barnes등에 의해 개발된 octtree[1]가 가장 널리 사용되고 있는데, 이 방법은 입자들의 기하학적 정보를 바탕으로 매 분할마다 공간상에 동일한 체적의 여덟 개의 정육면체 공간을 재귀적으로 생성하여 트리구조를 만든다.

본 연구에서는 좌표계에 독립적이고 입자 분포에 고도의 적응성을 가지는 새로운 트리코드(k-tree)를 개발하였다. 이 방법은 입자가 분포하고 있는 공간을 임의의 정수 k개로 분할하기 위해서 k-means 알고리즘을 이용하여 매 분할마다 k개의 부공간을 생성한다. 이러한 분할을 재귀적으로 하나의 셀에 하나 이하의 입자가 존재할 때 까지 실시하여 완전한 계층적 트리구조를 만든다. 이 방법은 트리의 차수 k를 달리할 수 있다는 특징이 있으며, 입자간 상호작용을 계산하기 위한 트리순환은 기존과 동일하다.

트리코드의 성능을 비교하기 위해서, 트리순환에 있어서 공통적이며 트리에 독립적인 서브루틴을 사용한다. 연산은 AMD Athlon64 3000+ 와 1GB Ethernet을 사용하는 32노드의 리눅스 클러스터에서 수행되었다. 병렬화된 연산환경에서 계산을 수행하기 위한 영역분할기법으로는 ORB(Orthogonal Recursive Bisection Method)를 사용하였다. ORB에 의해 공간적으로 이웃하게 되는 입자들의 집합이 각각의 프로세서로 할당된 후, 프로세서마다 트리들이 구성되고, 완전한 상호작용을 계산하기 위해 입자들이 프로세서 간에 교환된다. LET는 영역분할 결과를 이용하여 하나의 프로세서가 다른 프로세서들에게서 자신이 가지는 모든 입자들이 완전한 힘 계산을 하는데 필요한 트리의 노드들만을 전송 받아 완전한 트리를 구성함으로써 힘 계산단계에 있어서는 아무런 통신도 일어나지 않도록 하는 방법이다[2]. 본 연구에서는 LET를 사용하지 않는 경우와 LET를 적용한 경우에 대하여 각각을 비교하였다. 대상으로 하는 입자계 분포는 128만개의 입자가 중앙부 원점에 고도로 입자가 응집되어 있는 Plummer 모델을 택하였다. 이는 천체물리학에서 구형성단의 모형에 많이 사용하는 것으로서, 동일한 입자수를 가지는 다른 분포에 비해 동일 오차수준에서 훨씬 많은 연산을 필요로 하는 고비용 연산의 특성을 가진다. 오차는 병렬화된 입자-입자간 직접 상호작용 연산 프로그램을 작성하여 개개 입자가 가지는 가속도를 구한 후, 이 값을 참값으로 하여 각각의 입자가 가지는 가속도를 비교하여 상대오차를 구하고 이들의 r.m.s. 값을 비교하였다[3].

그림1은 트리의 차수 k를 변화시켜가며 단일 타임스텝에서의 입자당 $N_E$(시험입자와 노드간 거리비교 횟수)와 $N_I$(실제 상호작용 연산 횟수)를 동일 오차수준에서 비교한 것으로서, $N_E$의 경우 k가 2인 경우 가장 높게 측정된 반면 $N_I$는 비교적 가장 낮게 측정되었다. 가장 바람직한 트리의 차수 k는 3~5 부근으로 추정되었으며 기존의 octtree대비 최소 30%이상 연산성능이 향상되었음을 알 수 있다. 이러한 차이는 보다 작은 오차를 수반하는 정밀한 연산의 경우 더욱 증가하였다.

그림2는 LET의 사용여부에 따라 단일 타임스텝에 소요된 연산시간을 비교한 것이다. LET사용여부에 관계없이 역시 k가 3~5부근에서 가장 우수한 성능을 보이는 것으로 조사되었다. 그러나 선택한 입자분포와 연산환경에 대해서 LET는 긍정적인 효과가 나타나지 않았다. LET에 의한 성능향상은 다른 종류의 입자분포, 아마도 더욱 고성능의 네트워크장치가 동원되는 연산환경에서 그 효과가 부각될 것으로 기대한다.

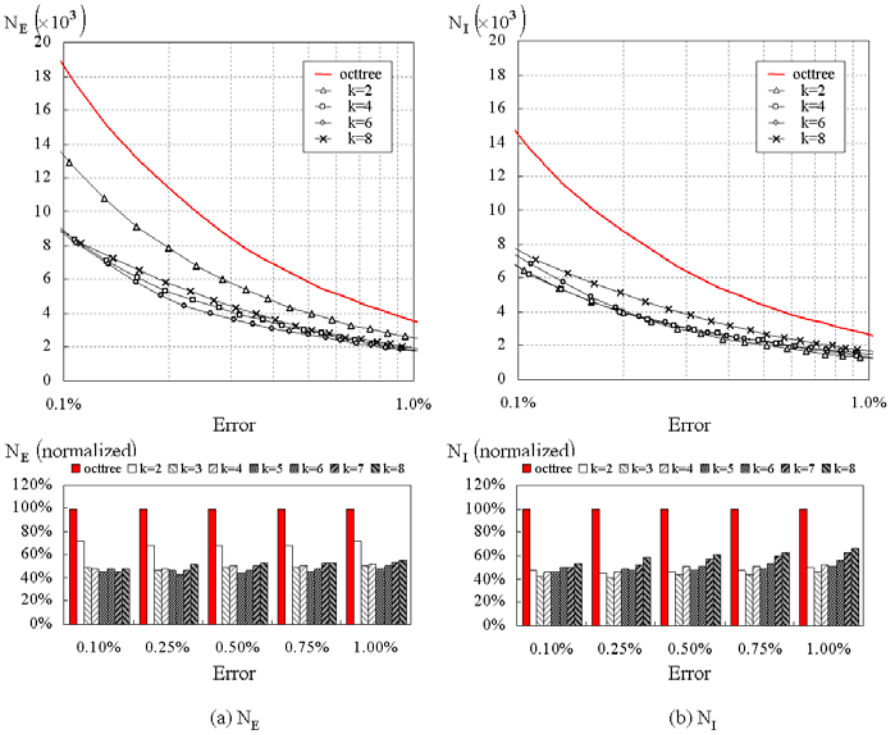


**그림 1** 단일 타임스텝에서 k값에 따른 $N_E$와 $N_I$의 비교

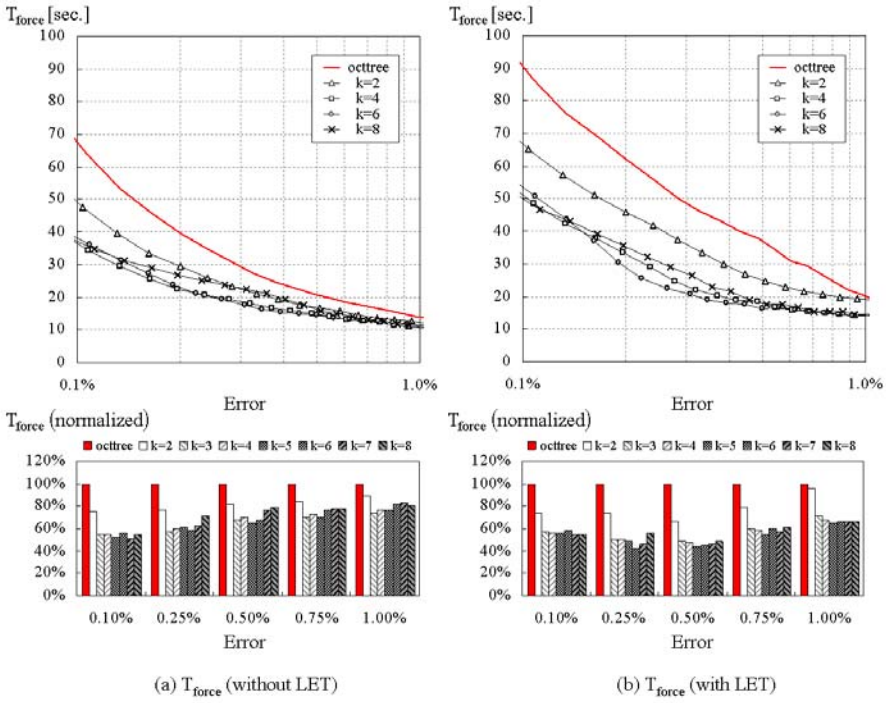


**그림 2** 단일 타임스텝 연산에 측정된 시간

참고문헌

[1] J. Barnes and P. Hut, A hierarchical O(NlogN) force-calculation algorithm, Nature 324, 446 (1986).

[2] J. Dubinski, A parallel tree code, New Astron. 1, 2 (1996).

[3] J. Makino, A comparison of two different tree algorithms, J. Comput. Phys. 88, 393, (1990).