ハイパフォーマンス
コンピューティング 69−11
（1997. 12. 12）

分子動力学法プログラム AMBER と Barnes-Hut tree code の並列化と性能評価

斎藤 稔[1] 三十尾 潔高[2] 志澤 由久[2]
丸川 一志[3] 秋山 泰[1]
野口 保[1] 鬼塚 健太郎[1]

分子動力学シミュレーションのプログラム AMBER と Barnes-Hut の tree code とを、分散メモリ型並列計算機 SR2201 上で MPI 通信ライブラリを使って並列化し実行スピードの評価を行った。AMBER の実行速度は、AMBER のホームページで公表されている速度に比べて飛躍的に向上した。また、Barnes-Hut tree code の実行速度は、独自の並列化手法によって目覚ましく加速した（128PUs で 116 倍、256PUs で 170 倍）。

Parallelization and Performace Evaluation of MD Program AMBER and Barnes-Hut Tree Code

MINORU SAITO,[1] KIYOTAKA MISOO,[2] YOSHIHISA SHIZAWA,[2]
KAZUSHI MARUKAWA,[3] YUTAKA AKIYAMA,[1] TAMOTSU NOGUCHI[1]
and KENTARO ONIZUKA[1]

We parallelized MD program AMBER and Barnes-Hut tree code using the MPI library and evaluated those performance data on SR2201. The speed of AMBER was remarkably increased by our improvements compared with the performance data of AMBER home page. On the other hand, the speed of Barnes-Hut tree code was also accelelated by our parallelization approach on SR2201 (116 and 170 times faster for 128 and 256 PUs, respectively).

1. 序　論

1.1 蛋白質の分子動力学シミュレーション

分子動力学シミュレーションは、固体から液体そして蛋白質に至るまで様々な多体系の性質を理論的に研究するために発展してきた。分子動力学シミュレーションの適用範囲が広がるためには、分子動力学シミュレーションの方法の改良と電子計算機の処理能力の向上の両方が必要であった。例えば、蛋白質は数千から数万個の原子から成る巨大な分子であるが、固体や液体のように均一でなく、むしろ複雑で階層的な構造を持っている。このような構造は、原子間の数種類の複雑な相互作用によってバランスしている。更に、蛋白質は水溶液中に存在するために、蛋白質を取り巻く数千個の水分子をまとめてシミュレーションしなければならない（図 1）。そのために、蛋白質の分子動力学シミュレーションは、蛋白質と水の原子から成る数万原子の巨大な系について行わなければならない。原子間の相互作用が複雑であり原子数が数万個に達するために、分子動力学シミュレーションを実行するには膨大な計算を処理しなければならない。蛋白質の本格的な分子動力学シミュレーションが行えるようになるためには、ベクトル型のスーパーコンピューターが登場するのを待たなければならなかった。ベクトル型の CPU の能力が頭打ちとなった今は、並列計算機による処理能力の向上が望まれている。

1.2 AMBER[1)]

蛋白質の分子動力学シミュレーションを行うためのプログラムは、すでに幾つか存在する。AMBER、DISCOVER、CHARM、GROMOS が、普及しているプログラムとして良く知られている。このなかでも、AMBER は原子間の相互作用のパラメターが論文で公表されており、そのソースコード (Fortran で数万行) に改良を加えることが許されているために並列化の検討も以前からなされている。すでに今から 5

1 新情報処理開発機構: Real World Computing Partnership. Correspondence should be addressed to Minoru Saito (E-mail:msaito@trc.rwcp.or.jp).
2 (株) 情報数理研究所: Information and Mathematical Science Laboratory Inc.
3 (株) アロー: Arrow Co.

年前に、佐藤らは、AMBER を AP1000 上で DMA 通信の方法に基づいて並列化して速度データを計測した[2]。彼らの結果は非常に良い台数効果を示している(約2万原子の系を、16、128、256PUs で 15、89、129 倍)。しかし、当時の AP1000 のプロセッサ単体の性能が劣っていたために、MD を一ステップ実行するための計算時間は、約2秒(512PUs)もかかっていた。したがって、シミュレーションを実際に必要なステップ(数百万ステップ)行うのは不可能であった。近年、AMBER の MPI による並列版(AMBER ver.4.1)が作成され、その SR2201 上でのスピードの測定結果が AMBER のホームページに掲示されている。それによると、SR2201 上での台数効果が非常に悪いことが分かる(1997 年 11 月現在、16PUs で 3 倍)。すなわち、AMBER4.1 の MPI による並列版は、SR2201 の潜在能力を十分に発揮してないことが分かる。我々は、この原因がどこにあるのかを解明し、SR2201 の能力を十分に引き出すように AMBER の MPI 版を改良することにした。

1.3 Barnes-Hut tree code

Barnes-Hut tree code は、数万体の重力多体系を長距離万有引力をカットオフせずに高速に計算しシミュレートする最初のプログラムである[3]。このプログラムでは、空間を疎視化することによって万有引力の計算を高速に行う。すなわち、まず、遠くの空間をセルに区切る。つぎに、セルの中にある質量をまとめて一つの質量と見なしセルの重心に置く。セルの重心の総質量からの引力を、セル中の沢山の質量分布からの引力のかわりに計算する。筆者の一人(斎藤)は、この方法にヒントを得て水溶液中の蛋白質の長距離クーロン相互作用の高速計算法(PPPC;Particle-Particle and Particle-Cell 法)を開発し、水溶液中の蛋白質を長距離クーロン力をカットオフせずにシミュレートすることを初めて可能にした[4),5]。Barnes-Hut tree code は、その後、アメリカの幾つかのグループによって並列化された[6]。我々は、Barnes-Hut の NlogN アルゴリズムによる空間の疎視化が、SR2201 のような分散メモリ型の並列計算機上でどの程度のスピードが出るのかを、独自の方法で並列化を行って調べることにした。そのために、Barnes によって公開されているオリジナルの tree code を性能評価に用いた。

2. プログラムの構造

2.1 AMBER の構造(図2)

蛋白質の分子動力学シミュレーションの一ステップは、原子間の各種相互作用の計算と運動方程式を解いた後の座標と速度の更新とに大きく分けることができる。後者に要する計算時間は、相互作用の計算に比べて非常に小さい。原子間の相互作用は、結合性の力(共有結合 bond、結合角 angle、二面角 dihedral に起因した力)と非結合性の力 nonbond(原子間の

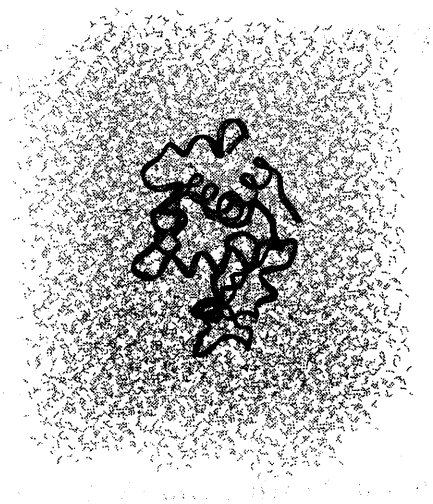

図1 水中のタンパク質(ヒトリゾチーム,lzmwat)

AMBER	Barnes-Hut tree code
<force>	<maketree>
pairlist	calcmkey
balance	sortmkey
nonbond	localtree
bond	calcchg
angle	combtree
dihedral	<calcforce>
barrier	<eq. of motion>
allreduce(F)	<allgather>
<eq. of motion>	
shake	
allreduce(E)	
allgather(x;v)	

図2 AMBER と Barnes-Hut tree code の計算の流れ

Lennard-Jones 力、原子電荷間のクーロン力)からなる。このうち、結合性の力は、ほぼ原子数のオーダーの計算量になる。一方、Lennard-Jones 力とクーロン力は、原子の対の間に働く力であるため結合性の力よりも計算量は多い。Lennard-Jones 相互作用は、距離の6乗に逆比例して急激に減衰するのに対して、クーロン相互作用は万有引力と同様に距離の1乗に逆比例してゆっくり減衰する。そのため、遠距離まで相互

作用が到達し、相互作用の数は原子数の2乗に比例して膨大になる。一般に、長距離相互作用を含む系のシミュレーションを行うためには、長距離相互作用を高速に処理する計算アルゴリズムと高速な計算機が必要である。

蛋白質には、正味の電荷をもつ原子団（アミノ酸側鎖）が存在するにもかかわらず、長距離クーロン力は計算が困難であるために10オングストローム前後（蛋白質のサイズより小さい）でカットオフされていた。しかし、筆者の一人（斎藤）は、蛋白質のクーロン力をPPPC法によりカットオフせずにシミュレーションを行い、従来のカットオフによるシミュレーションと比較して、クーロン力のカットオフが結果に深刻な悪影響を及ぼすことを明らかにした[7]。AMBERは以前からカットオフに基ずいて計算を行っていたため、並列化の性能評価もカットオフの計算についてなされている。我々も比較のために、クーロン力の計算をカットオフして測定することにした（prowatは、12A cutoff）。カットオフによってクーロン力を計算するには、まず、カットオフの半径内に存在する原子団をリストする（アミノ酸残基ベースのカットオフ）必要がある（pairlist）。次に作成したリストを参照しながら、粒子が相互作用する相手を限定して力を計算する(nonbond)。

2.2 Barnes-Hut tree codeの構造（図2）

Barnes-Hut tree codeは、空間を入れ子になった大小のセルに分ける部分（maketree）と大小のセルからの力を計算する部分（calcforce）とからなる。入れ子になった大小のセルは木構造をとる。すなわち、系全体を含むroot cellが木の幹に相当し、root cellに含まれる大小のセルは、幹から伸びる枝の節に相当する。粒子（あるいは、原子）は木の葉に対応する。粒子がセルや他の粒子の質量(あるいは電荷）から受ける力は、木構造を利用しながら計算する。Barnes-Hut tree codeは、Fortran版とC版が公開されている。我々は、Fortran版およびC版から書き換えたC++版を並列化した。計測には、AMBERの出力する電荷分布と独自に準備した電荷分布の両方を用いた。

3. 並列化の方法

3.1 AMBERの並列化の方法

分子動力学シミュレーションのプログラムを並列化するには、大きく分けて空間分割とデータ分割の方法がある。AMBER ver.4.1のMPIによる並列版は、データ分割によって並列化されている。分割の方法としては、粒子や相互作用をブロックに分割しプロセッサに割り当てる方法(バンド分割）と一個一個の粒子をプロセッサに順番に割り振る方法(サイクリック分割）がある。AMBER ver.4.1のMPI版では、結合性相互作用のループをバンドで分割し、非結合性相互作用のループをサイクリックに分割している。並列実行の直後には、粒子に働く力のデータは、それを計算したプロセッサのみがメモリに持っている（分散メモリ型並列計算機の場合）。したがって、次のステップに進む前に、それらのデータを互いに通信することによって完全なデータを全てのプロセッサが持つ必要がある（allreduce(F)）。粒子に働く力から運動方程式を解くことによって、次の時間ステップの粒子の座標を求めることができる。運動方程式の計算も粒子についての単純ループであるから並列実行することができる。並列に実行した後は、プロセッサ間の通信によって全ての粒子の更新された座標データを、全てのプロセッサが次のステップの力の計算に移る前に持つ必要がある（allgather(x,v)）。

3.2 Barnes-Hut tree codeの並列化の方法

Barnes-Hut tree codeの木構造の作成に要する時間は、それに基づく力の計算時間に比べて二桁小さい。しかし、数百倍の高速化を達成するためには無視できない時間である。したがって、木構造の作成も並列化しなければならない(maketree)。木構造の作成を並列に行うために、我々はまず、幹から先の枝を幾つかの枝に束ねてプロセッサに分割することにした。次に、それぞれの枝から先の細い枝と葉からなる木構造を各プロセッサが作成する(localtree)。そして、各プロセッサが作成した部分木に関する情報をプロセッサ間で通信し、それらの情報から各プロセッサが完全な木構造を完成させる(combtree)。ここで、木構造を並列に効果的に取り扱うためにmorton keyを用いた。次に、粒子へ働く力を計算するために、粒子と相互作用する大小の全てのセルを木構造を探索することによって決定する。Fortranでは、それらのセルの全てをあらかじめ表にリストアップしている。木構造の探索のアルゴリズムは、FortranとC++とで異なる。木構造の探索と粒子に働く力を計算するループをプロセッサで分割することによって並列実行することが容易にできる(calcforce)。

4. 性能評価

4.1 AMBERの性能評価

AMBER(ver.4.1)のMPI版の性能評価は、AMBERに付随しているテストデータ(prowat)と我々が準備したテストデータ（lzmwat）について行った。prowatとは、98個のアミノ酸から成る一個の蛋白質plastocyaninが立方体の水に沈められている系である。蛋白質の原子数は、1460個、水の分子数は、3375個、全原子数は、11585個である。lzmwatは、130個のアミノ酸から成るヒトリゾチームという蛋白質が、比較的大きなサイズの立方体の水に沈められている系である(図1)。蛋白質の原子数は、2023個、水の分子数は、5685個、全原子数は、19078個である。

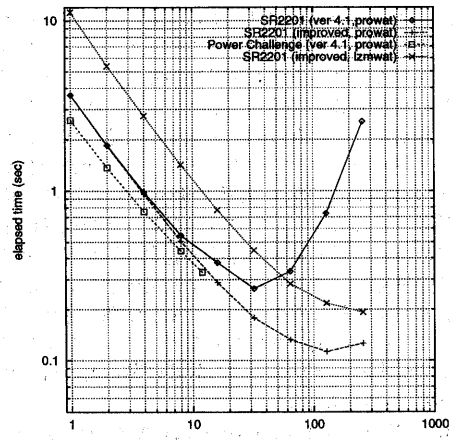

図3　AMBER の 1 ステップの平均実行時間 (sec/step)

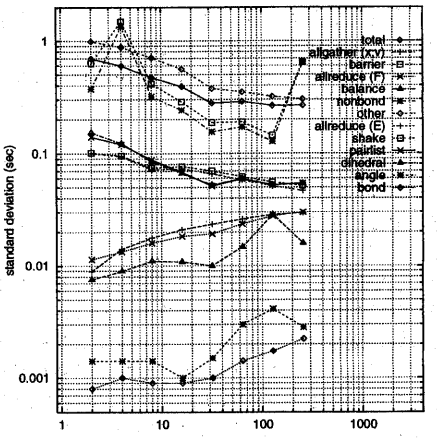

図6　図 5の実行時間のプロセッサ間のばらつき (sec/step)

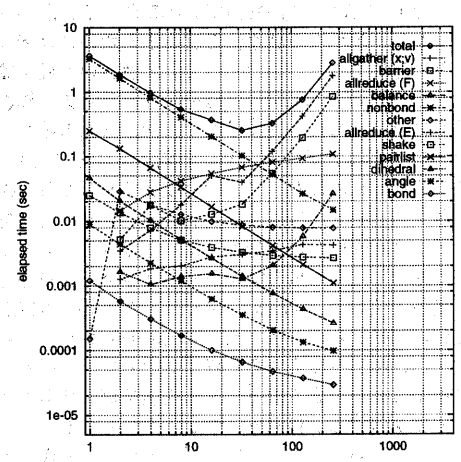

図4　AMBER ver.4.1 の主要部分の実行時間 (sec/step)

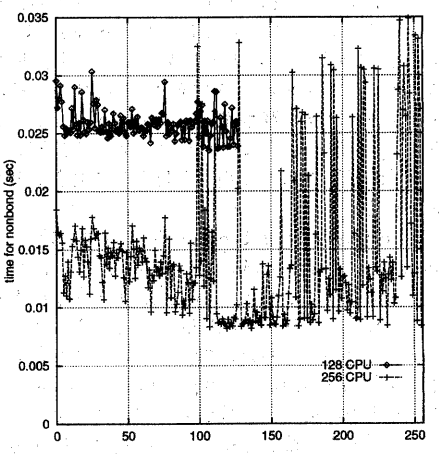

図7　各プロセッサが nonbond の計算に要する時間 (sec/step)

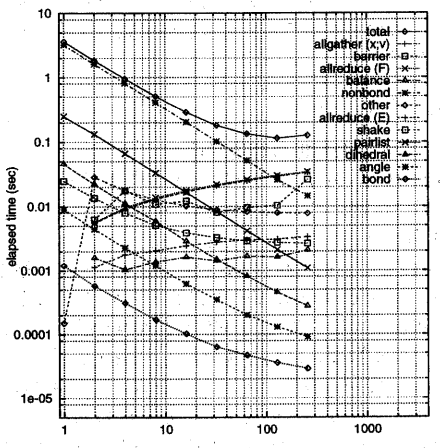

図5　AMBER 改良版の主要部分の実行時間 (sec/step)

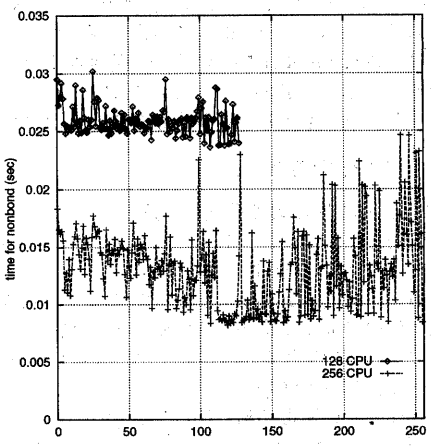

図8　各プロセッサが nonbond の計算に要する時間（改良後）

AMBER(ver.4.1) の SR2201 上でのスピードは、AMBER のホームページに公開されている。我々の測定した結果は、それを再現している (図 3)。この図から分かるように AMBER(ver.4.1) の SR2201 上での並列化効率は非常に悪い。この原因を明らかにするために、我々は AMBER の主要部分について分けて時間測定を行った。その結果を図 4に示す。この図から、プロセッサ台数が増えるにしたがって、プロセッサ間の通信に要する時間が急激に増大していることが分かった。この通信には、MPI 関数の allgatherv と allreduce を利用している。そこで、SR2201 上のこれらの関数に頼らず、我々が自作したツリー通信の関数を利用して計測を行った。その結果、図 3に示すように並列化効率は驚くほど向上した。しかし、プロセッサ台数が更に増えて 128 台になると、並列化による加速は飽和する。

計算の主要部分に要する時間のうちで、最も時間のかかるのは非結合性相互作用の計算である。最も計算時間を必要とする非結合性相互作用の計算は、プロセッサ台数に対して比例して加速されている (図 5の nonbond)。一方、通信に要する時間は、ツリー通信によって減ったものの、プロセッサの台数が増えると他の部分の計算時間が短縮するためしだいに目立ってくる (図 5)。台数効果の飽和の原因は、主に通信に要する時間であることが分かる。lzmwat(15 A カットオフ) の台数効果を図 3に示した。prowat(12 A カットオフ) よりも 1 ステップの計算時間が増えるとともに台数効果が良くなっていることが分かる。これは、nonbond の計算時間が増えたために相対的に通信の時間が目立たなくなったためである。カットオフの距離を恣意的に長くすれば台数効果を良く見せることができるが、計算時間は急激に増えるためにプロセッサ台数を増やしても計算時間を短縮することができなくなる。

次に、各プロセッサへの負荷の分散が、どの程度均等に行なわれているかを調べた。各プロセッサがそれぞれの主要部分の計算に要する時間のばらつきをプロットした (図 6)。非結合性相互作用の計算時間のプロセッサ間の標準偏差が、256 プロセッサになって増大していることが分かる。各プロセッサが要した時間を図 7にプロットした。128 プロセッサの負荷分散はそれほど悪くないのに比べて、256 プロセッサでは約 100 番目のプロセッサから 256 番目のプロセッサの計算時間にばらつきがあることが分かる。この原因を調べた結果、水分子 (TIP3P モデル) に特化された高速計算ルーチン (qiktip) が、負荷分散 (balance) の副作用で高速に動作しなくなるためであることが分かった。qiktip の高速化が犠牲にならないように改良した結果、図 8に示されるように 99 から 256 番目までのプロセッサの負荷は改善された。ここで示した非結合性相互作用は最も時間のかかっている部分であることから、更に負荷分散を改善することによって、全体の性能を上げることができると思われる。

4.2 Barnes-Hut tree code の性能評価

Barnes-Hut tree code の Fortran 版と C++版の並列化パフォーマンスを図 9,11に示す。図 9は、AMBER の計測で用いた prowat の電荷分布に対する計算時間である。我々の C++の tree code は 256PUs まで加速している。一方、Fortran の tree code は、32PUs までは C++の code よりも速いが 64PUs で減速して C++と同じスピードになる。また、Fortran 版は、SR2201 の疑似ベクトル化によって全てのプロセッサ台数にわたって計算速度が向上していることが分かる。

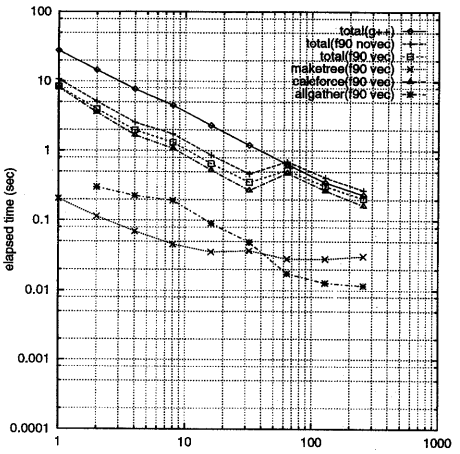

図 9　tree code の 11585 原子 (prowat) に対する実行時間 (sec/step)

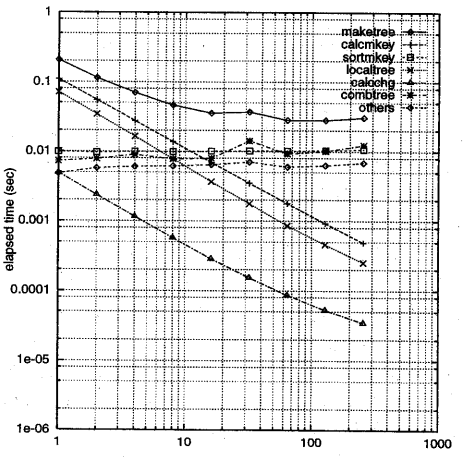

図 10　図 9の maketree 内部の実行時間 (sec/step)

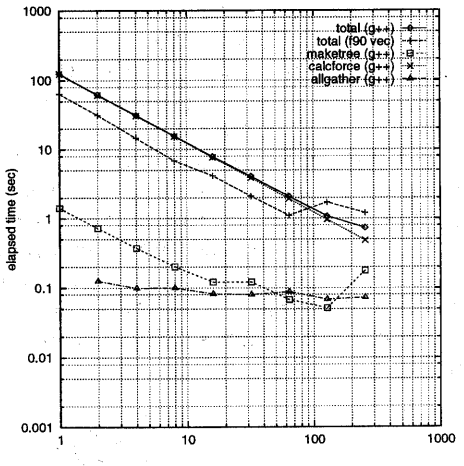

図 11 tree code の 50000 個の電荷分布に対する実行時間

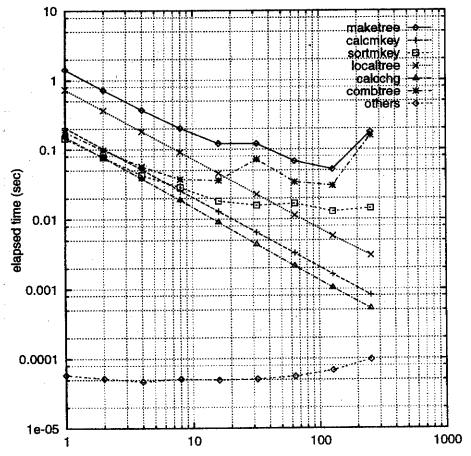

図 12 図 11の maketree 内部の実行時間 (sec/step)

並列化による性能を、更に大きな粒子系で評価するために、5 万粒子の電荷分布を準備して計測した (図 11)。Fortran 版 tree code の計算速度は、64PUs まで加速し 128PUs で一旦減速することが分かる。この原因を調べたところ、calcforce 中の木構造を探索する部分に起因することが分かった。現在、改良を検討している。一方、C++版の計算時間は、台数に反比例して理想的に減少していることが分かる。粒子へのセルからの力の計算 (calcforce) は、粒子についての単純ループをプロセッサに分割しているので理想的な加速を示している。ところで、木構造を作る maketree は、256 台で計算時間が増大している。そのため、tree code 全体の加速が 256 台になって鈍っている。この maketree の時間の増加は、各プロセッサが構築した部分木の繋ぎ合わせ (combtree) に起因しているように見える (図 12)。しかし、更に解析を進めた結果、combtree に至るまでの処理での同期の乱れが、combtree 内部での全通信で表面化したことが分かった。したがって、256 プロセッサでの負荷の等分配を再検討することによって、256 台での並列化効率は改善されると思われる。

5. 結　　論

AMBER ver4.1 の MPI 版は、通信が並列化効率を悪化させていることが明らかになったので、SR2201 の MPI 関数 (allgather と allreduce) を用いずにツリー通信の関数を自作して用いた。その結果、AMBER 全体の計算スピードを目覚しく向上させることができた。Barnes-Hut tree code は、我々の並列化のアプローチ (木構造を並列に構築) によって非常に良い並列化効率を実現することができた。我々の並列化 AMBER と並列化 tree code を組み合わせることにより、並列化 PPPC 法を組み込んだ COSMOS90[7] に迫る高速シミュレーションを実現できると期待される。

謝辞 g++を SR2201 に移植した新情報処理開発機構 佐藤三久 博士に感謝します。

参 考 文 献

1) D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham III, D. M. Ferguson, G. L. Seibel, U. C. Singh, P. K. Weiner and P. A. Kollman, AMBER4.1, University of California, San Francisco (1995).

2) H. Sato, Y. Tanaka, H. Iwama, S. Kawakita, M. Saito, K. Morikami, T. Yao, S. Tsutsumi, "Parallelization of AMBER molecular dynamics program for the AP1000 highly parallel computer" Proceedings of SHPCC-92, pp.113-120(1992).

3) J. Barnes and P. Hut, "A hierarchical O(NlogN) force-calculation algorithm", Nature, 324, pp.446-449(1986).

4) 斎藤 稔, "蛋白質分子の分子動力学計算", 科学、岩波書店,61,pp.822-824(1991).

5) M.Saito, "Molecular dynamics simulations of proteins in water without the truncat ion of long-range Coulomb interactions", Molecular Simulation,8,pp.321-333(1992).

6) D. H. Bailey, P. E. Bjorstad, J. R. Gilbert, M.V. Mascagni, R. S. Schreiber, H. D. Simon, V. J. Torczon, L. T. Watson (Eds.), "Parallel processing for Scientific computing", Proceedings of the seventh SIAM conference (1995).

7) M.Saito, "Molecular dynamics simulations of proteins in solution: Artifacts caused by the cutoff approximation", J.Chem.Phys.,101,pp.4055-4061(1994).

8) 斎藤 稔, 河野秀俊、佐谷野健二、皿井明倫、"高並列化分子動力学／自由エネルギー摂動計算による蛋白質の転移温度変化の導出", 第 11 回分子シミュレーション討論会 (1997).