社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－ARC－172 (1)
2007－HPC－109 (1)
2007／3／1

コンパクトクラスタを利用した分子軌道計算の性能評価

早川 潔* 佐々木 徹** 梅田 宏明***, † 長嶋 雲兵***, †

ハイパフォーマンスアプリケーションであるHF法を用いた分子軌道計算プログラムにおいて，全体の実行時間の約99%の時間をFock行列生成に費やされている．分子軌道計算プログラムをはじめとするハイパフォーマンスアプリケーションを低消費電力かつ省スペースで実行するために，我々は組み込み部品を利用したコンパクトクラスタ（EMDCシステム）を開発し，その上で並列Fock行列生成プログラムを実装するための通信ライブラリを開発した．EMDCシステム上で，並列Fock行列プログラムを2種類（動的負荷分散および静的負荷分散）を実行した結果，どちらのプログラムもFock行列生成において90%以上の高い並列化効率を示した．特に，動的負荷分散では，EMDCシステムのようなヘテロクラスタシステムで有効であることが実証された．

Implementation and Evaluation of Molecular Orbital Calculations Program on a Compact Cluster

Kiyoshi Hayakawa, *Tohru Sasaki***, *Hiroaki Umeda****, †, and *Umpei Nagashima****, †

In molecular orbital calculations, one of the high performance applications, execution time of Fock matrix construction is approximately 99% of all execution time. We are developing a compact embedded cluster (called EMDC), and a communication method on it for a parallel Fock matrix construction program . The parallel Fock matrix construction program was developed on EHPC project, and EMDC has been developing for low power, small space and rapid prototyping. Network architecture (tree-base network) of EMDC conforms to parallel Fock matrix construction program. In order to communicate among the system, we developed a communication library for tree-base network. In evaluations, two parallelizing methods were executed on EMDC. Specially, the dynamic task distribution method shows high paralellization efficiency on hetero cluster like EMDC system.

1 はじめに

近年，汎用CPUの低消費電力化が進み，そのCPUを使用したPCクラスタの研究も盛んに行われている[1][2]．低消費電力CPUを使用することにより,CPUファンなどの冷却装置が小型化され（または省かれ），より高密度に実装可能である．

高密度実装クラスタとして，*Green Destiny*[1]や*MegaProto*[2]というクラスタが知られている．いずれのクラスタも1シャーシに20個前後の低消費電力型のCPUを使用している．本研究室でも，低消費電力でコンパクトなクラスタとして，EMDC[1]システムを開発中である．

EMDCシステムは1シャーシに3台のノードが搭載されおり，ネットワークをシャーシ内とシャーシ外に分けて，データ転送をする．つまり，シャーシ間ネットワーク(Inter-Chassis Network)とシャーシ内ネットワーク（Intra-Chassis Network)に分け，効率の良い転送を考えている．コレクティブ通信は，これらのネットワークでtreeを構成し，データ転送を行うことを考えている．

一方，ハイパフォーマンスアプリケーションの1つとして，分子軌道計算が知られている．分子軌道計算法の1つであるHF（Hartree-Fock法）法では，処理全体の約99%がFock行列生成に費やされており，より効率のよいFock行列生成が必要となる．EHPCプロジェクト[3]では，分子軌道計算などのハイパ

*:大阪府立工業高等専門学校 Osaka Prefectural College of Technology
**:アプリオリマイクロシステムズ A-Priori Microsystems Ltd
***:産業技術総合研究所 National Institute of Advanced Industrial Science and Technology
†:CREST 科学技術振興機構 CREST Japan Science and Technology Agency

[1]Embedded Middle Density Cluster

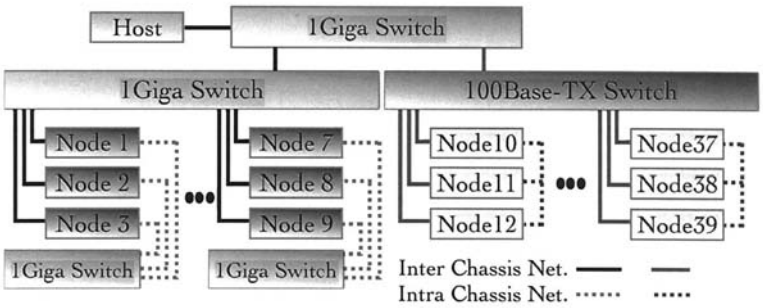

図 1: EMDC システムの構成

フォーマンスアプリケーションを効率よく処理するために，コンピュータシステムのプラットフォームを提案し，そのプラットフォーム上で効率のよい並列 Fock 行列生成アルゴリズムも開発されている [4]. このプラットフォームでは，各ノードを Tree 上に結合させており，EMDC のネットワークとの親和性も高い.

そこで，本稿では，EMDC システムの 2 つのネットワークを使った仮想 tree ネットワークを考え，そのネットワーク上での通信フレームワークを提案し，そのフレームワークを基にした通信ライブラリ（H/L-comm) を開発した．また，H/L-comm を利用して，EHPC プロジェクトで開発された分子軌道計算プログラムを EMDC システム上に実装し，その性能評価を行った.

2 EMDC システム

図 1 に EMDC システムを構成を示す．EMDC システムでは，FA などで使用する組み込み機器のマザーボードを採用し，そのマザーボードにインテルが供給している EmbeddedCPU を搭載することで長期間運用を可能にしている．組み込み機器のマザーボードにも Pentium M などのノート PC で使用される低消費電力型 CPU が搭載できる製品もでてきており，より低消費電力なクラスタが実現可能である.

2.1 シャーシ内部の構成

本システムでは，PentiumM(2.0GHz) 搭載のノードが 9 台，PentiumIII 搭載のノード 30 台，およびホストコンピュータで構成されている．PentiumIII 搭載のノードでは，3 ノードのうち 1 ノードが動作周波数 600MIIz の CPU で残りの 2 ノードが 700MHz の CPU である．PentiumIII や PentiumM などの比較的低動作周波数ではあるが低消費電力である CPU を利用して，コンパクトなクラスタを目指している．今後，FPGA などを利用してアプリケーションをハードウェア化し，より高速に処理できるシステムにすることを考えている．PentiumM ノードには 1Gbyte(デュアルチャネル)，PentiumIII には 256Mbyte のメモリが搭載されている.

HDD は，コンパクトフラッシュメモリ（PentiumM・PentiumIII ノードともに 1Gbyte）を採用している．このことにより，機械稼働部品が減り，より長期運用・低消費電力なシステムになると考えられる.

2.2 ネットワーク構成

ネットワーク構成は，前述したとおり，Intra-Cahssis Network および Inter-Chassis Network で構成されている．PentiumIII ノードの Intra-Chassis Network は，SCC ボードと呼ばれる低レイテンシ通信用ボードを使用して構築し，PentimuM ノードの Intra-Chassis Network は，Gigabit Ethernet で構築する．ただし，本稿の評価では，SCC の代わりに，Ethernet を使用している．PentiumIII ノードの Inter-Chassis Network は，100Base-TX の Ethernet で構築し，PentiumM ノードの Inter-Chassis Network は，Gigabit Ethernet で構築する.

3 仮想 tree ネットワーク

EMDC システムにおけるネットワークの枠組みとして，仮想 tree ネットワークを考える（図 2 参照）．Intra/Inter-Chassis Network をうまく利用することにより，仮想的に tree ネットワークを構成す

る．シャーシ内の通信を Intra-Chassis Network で密に行うことができるので，1 シャーシで 3 つのネットワークポート（Inter-Chassis Network のポート）があると仮想的に考え，それらのポートを使って，binary tree を構成する．

3.1 仮想 Tree ネットワークおけるノード間結合

本システムでは，シャーシ間接続として，シャーシ内の中央に位置するノード（Center ノード）が上位レベルのノードと接続し，シャーシ内の右側および左側に位置するノード (Right ノードおよび Left ノード）が下位レベルのノードと接続する[2]．つまり，レベル n とレベル $(n+1)$（$n = 0, 2, 4, ...$:2 の倍数）のノードは Inter-Chassis Network で通信する．

シャーシ内接続では，Center ノードが Right および Left ノードより 1 つ上のレベルとして定義し，それらの通信を Intra-Chassis Network で通信する．つまり，レベル $(n+1)$ とレベル $(n+2)$ のノードは Intra-Chassis Network で通信する．

本システムで実行する分子軌道計算プログラムでは，n を 2 までの tree ネットワークにした．つまり，Level 1 のノードが Center ノードとなり，Level 2 のノードは Right または Left となる．

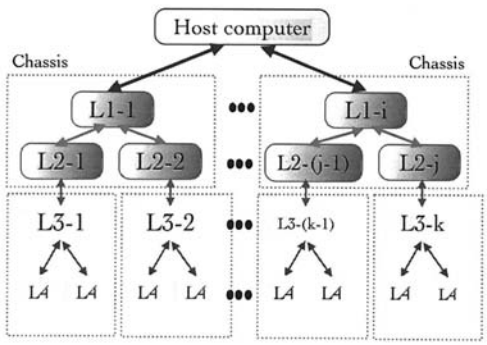

図 2: 仮想 tree ネットワーク

4 通信フレームワーク

仮想 tree ネットワーク上での通信フレームワークを提案する．本フレームワークは後述するアプリケーションフレームワークの下で適用される．

[2]ホストコンピュータをルートノード（レベル 0）とする．

4.1 アプケーションフレームワーク

アプリケーションフレームワークを図 3 に示す．

アプリケーションフレームワークは，Appli_Control 処理部，Communication Library 処理部，Communication Bridge 処理部，および Compute Program 処理部で構成される．

Appli_Control 処理部では，計算ノードの計算データを生成，各計算ノードに分配，および計算ノードが処理した計算結果の収集処理などアプリケーション全体のコントロールが行われる．Communication Library 処理部は後述する通信ライブラリ (H_comm および L_comm) の処理を行う．Communication Bridge 処理部は，通信のリンクレイヤ層が異なる場合に，その間のインターフェースの役割を行い，コレクティブ通信を制御する役割も行う．Compute Program 処理部は，Appli_Control 処理部から送られてきたデータを使用して計算を行う．分子軌道計算プログラムでは，Compute Program 処理部は Fock 行列生成処理が行われる．

Level 0 のノードには，Appli_Control および Communication Library(H_comm) を割り当て，Level 1 のノードには，Communication Library(H_comm および L_comm) および Communication Bridge を割り当て，Level 2 のノードには，Communication Library(L_comm) および Compute Program を割り当てた．

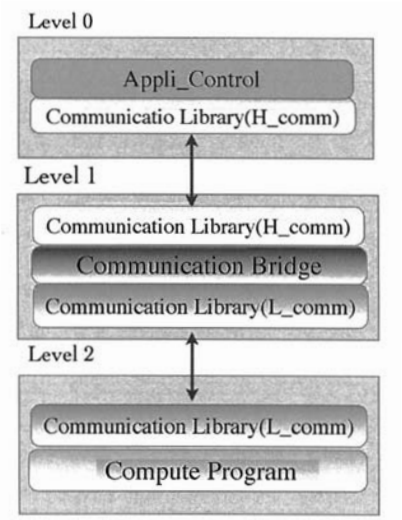

図 3: EMDC システムにおけるアプリケーションフレームワーク

4.2 通信フレームワーク

通信フレームワークの構成例を図4に示す．3つの連続するレベルに属しているノード集合が，通信を形成する1つの集合（MPIでいうCommunicator)として構成される．それらの集合をレベル$3n$からレベル$3n+2(n=0,2,4...:2$の倍数）に属しているノード集合，レベル$3n+2$からレベル$3n+4$に属しているノード集合，およびレベル$3n+4$からレベル$3(n+1)$に属しているノード集合というように，ある集合の最下位レベルのノードがその下のレベルのノード集合の最上位レベルのノードと重なるように集合を構成する．重なったレベルにCommunication Bridgeを挿入し，上位のレベルのノード集合と下位レベルのノード集合を結合させる．このことにより，より深いレベルのノード間通信を可能にする．

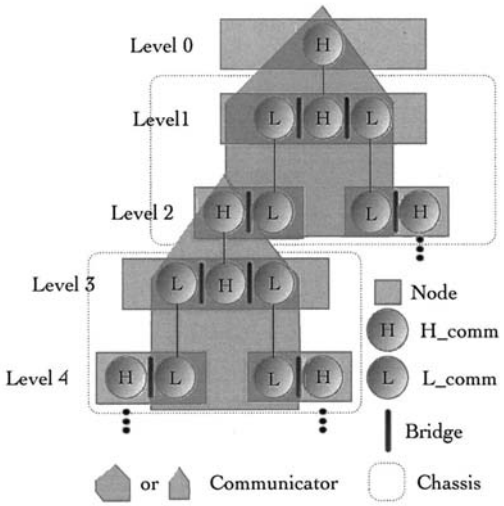

図4: 通信フレームワークの構成例（Level 4の場合）

4.3 通信ライブラリ

通信ライブラリのレイヤ構成を図5に示す．

通信ライブラリは，シャーシ内ネットワーク側のレイヤおよびシャーシ外ネットワーク側のレイヤ，それを結ぶコレクティブ通信レイヤで構成される．今回の実装では，シャーシ内，シャーシ外ネットワークのPhy.およびLinkレイヤは，Ethernetとし，それに対応するH_commおよびL_commを実装した．

H_commおよびL_commでは，send/recieveなどのatomicな命令を実装し，それを利用してCommunication Bridgeレイヤが通信を行う．H_commおよびL_commは，基本的に，1つ上（または1つ下）のレベルへのノード間通信のみを実現する．1つより上（または下）のレベルへのノード間通信は，Communication Bridgeレイヤを介して行われる．コレクティブ通信もCommunicaiton Bridgeレイヤが担当する．

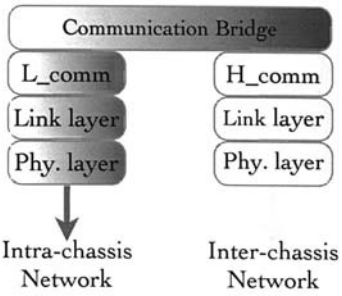

図5: 通信ライブラリのレイヤ構成

5 分子軌道計算における通信手順

本通信フレームワークを使用して実装した分子軌道計算は，静的負荷分散および動的負荷負荷分散の2つの負荷分散方式[4]である．HF法では，Fock行列生成時に，SCF（Self-Consistent Field:自己無撞着場）計算を繰り返す．静的負荷分散では，Appli_Contorl処理部があらかじめ計算タスクの計算量を決め，全計算ノードに一斉に同じ量の計算を実行させる．一方，動的負荷分散では，Appli_Contorl処理部が計算タスクを細かく分割し，空いている計算ノードに割り当て，計算を実行させる．

6 性能評価

分子軌道計算の性能評価として，C端末をOHキャップしたグリシンの5量体$(Glycine)_5$のHF/6-31G(d,p) (ガウス型基底の数は400)を計算した．文献[4]に示されているスクリーニングを行い，動的負荷分散で計算した場合と静的負荷分散で計算した場合を比較した．分子軌道プログラムにはGAMESS[5]をAppli_Control処理部として利用した．

本稿では，EMDCシステムの一部分を使用して計算した予備評価，およびEMDCシステム全体を使用した全体評価を行った．予備評価では，PentiumIII ノードのみで構成したクラスタの実行時間とPentiumMのみで構成したクラスタとの性能比較を

行い，全体評価では，PentiumIII および PentiumM で構成されたヘテロクラスタ環境での評価を行った．

本評価では，36 台のノード（PentiumIII ノード 30 台，PentiumM ノード 6 台）を使用して行った．各ノードには，LinuxOS(version 2.6) が搭載されている．また，本評価では，PentiumIII の Intra-Chassis Network は Ethernet を使用している．

6.1 PentiumIII ノードと PentiumM ノードとの性能比較

予備評価として，EMDC システムを PentiumIII ノードと PentiumM ノードに分け，それぞれで分子軌道計算を実行させた．実行結果を図 6 に示す．図 6 において，「static」は静的負荷分散における分子軌道計算全体の実行時間であり，「dynamic」は動的負荷分散における分子軌道計算全体の実行時間である．

PentiumM ノードの処理速度は，PentiumIII のそれと比べて，約 3 倍高速であることがわかった．この数値は，動作動作周波数の差とほぼ一致する．分子軌道計算では，メモリアクセスが少なく，浮動小数点処理部の性能が動作時間に大きく影響するので，PentiumM ノードに実装されているデュアルチャネルメモリなどの高速化技術が実行時間に影響を与えられなかったと思われる．

PentiumM のシャーシ（2 ノード実行）の消費電力（静的負荷分散）は 127W であり，PentiumIII のそれは 103W であった．積算電力（消費電力×計算時間）で比較すると，PentiumM シャーシは PentiuIII シャーシより，4.91 倍よい結果となった．また，Pentium4(3.0GHz:Northwood, Hyperthreding, FSB800MHz) 搭載のパソコンで，同様の計算を行い積算電力を計測した．その場合，PentiumM 筐体は Pentium4 パソコンより 1.96 倍よい結果となった．

6.2 EMDC 全体での評価

図 7 および図 8 に，並列化効率 (速度向上率を計算ノード台数で割った値) を示す．

ノード数 2 台での動的負荷分散の並列化効率は，静的負荷分散の並列化効率より低い値を示した．この原因の 1 つとして，タスクの粒度が細かすぎたため，通信オーバーヘッドが静的負荷分散よりも大幅に上昇してしまった事が挙げられる．しかし，4 台以降，動的負荷分散の並列化効率がそれほど悪化しない．よって，計算ノード 2 台から 6 台までは，静的負荷分散の並列化効率が動的負荷分散のそれに比

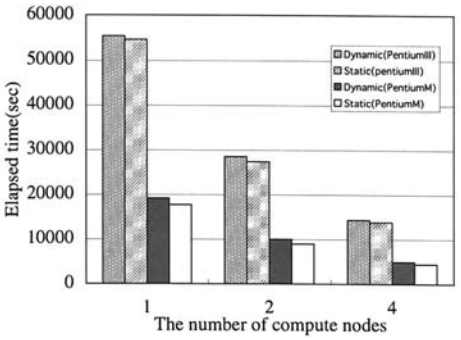

図 6: PentiumIII および PentiumM クラスタにおける $\alpha\text{-}helix(Glycine)_5$ の実行時間

べ，高い数値を示しているが，8 台から 14 台までは，ほぼ同じ値を示し，16 台以降は，動的負荷分散の並列化効率が高い数値を示した．計算ノード 22 台目から PentiumM ノードが加わり，動的負荷分散の並列化効率が Fock 行列生成部分で 0.11 ポイント，全体でも 0.09 ポイント上昇した．

PentiumM ノードが加わって得られた（分子軌道計算全体の）並列化効率の上昇について考察する．並列化効率と直接対応する値として，利用効率（PentiumM が加わったことによって得られる理想的な実行時間／実際の実行時間）を考える．予備評価の結果から，PentiumIII と PentiumM との性能差は，動的負荷分散で約 2.8 倍である．ノード数 22 台（PentiumM ノード 2 台を含む）における動的負荷分散の理想的な実行時間は，「PentiumIII ノード 1 台での静的負荷分散における実行時間／PentiumIII に換算したノード数」である．PentiumIII に換算したノード数を 25.6(20+2.8x2) とすると，ノード数 22 台における動的負荷分散の実行時間の理想値は，2133[sec] となる．よって，利用効率は 0.85 となり，ノード数 20 台における並列化効率（0.92) より 0.07 低下した．この低下の原因は，通信オーバーヘッドによる影響が大きいと推測される．PentiumM は，PentiumIII より多くのタスクを処理するため，PentiumM への通信量が増加し，PentiumIII 2 台追加の並列化効率の低下（約 0.01) より大きく低下してしまったと推測される．

7 おわりに

EMDC システムを構築し，その上で分子軌道計算プログラムを実装し，性能評価を行った．分子軌道

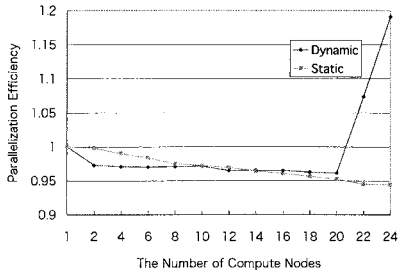

図 7: α-$helix(Glycine)_5$ の並列化効率（Wfock)

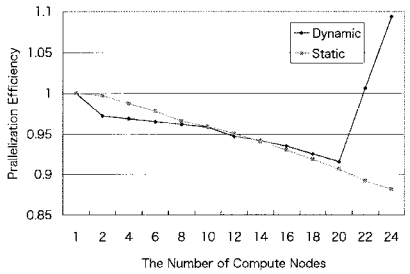

図 8: α-$helix(Glycine)_5$ の並列化効率（Wtotal)

計算プログラムを実装するために，EMDC システムのネットワークに適した通信フレームワークおよび通信ライブラリを提案し，実装した．実装した通信ライブラリおよび分子軌道計算プログラムは，36 台システム構成において正常に動作しており，性能評価では，動的負荷分散・静的負荷分散のどちらとも，Fock 行列生成処理において，0.9 ポイント以上の高い並列化効率を得る事ができ，全体処理においても 0.88 ポイント以上の並列化効率を得た．特に，動的負荷分散において，PentiumM が計算に加わった場合，並列化効率が 1 ポイントを超える値になった．静的負荷分散ではそのような効果が得られなかったことから，動的負荷分散は，ヘテロクラスタ環境においても計算ノードの能力を十分引き出せる能力を持っていることが実証できた．

今後の課題としては，L1 ノードにも Compute Program 処理部を割り当て処理した場合の性能評価をする必要がある．本稿では，Fock 行列生成処理のみを行うノードでの並列化効率を調べるために，評価 L2 ノードのみに Compute Program 処理部を割り当てたが，L1 ノードも計算資源が余っており，それを利用する事により，より高速に処理する事が可能になると推測される．また，PentiumM が計算に加わった時の上昇率が理想値より低いので，原因を解析し，改善する必要がある．さらに，他のハイパフォーマンスアプリケーションでも，今回提案した実行モデルおよび通信ライブラリをうまく適用して効率よく計算できる方法を検討する．ハイパフォーマンスアプリケーションの開発スタイルとして，まず，研究室内にあるコンパクトクラスタを使って，ソフトウェアでラピッドプロトタイピング型の開発をして，その後，そのソフトをハード化し，より高速に実行することを考えている．その場合，ソフトからハードへのスムーズに以降できるように，実行モデルや通信ライブラリを改善する必要がある．

この研究の一部は，平成 18 年度科学研究補助金（基盤研究 C:課題番号 18500044）「低消費電力コンパクトクラスタの研究」によって行われた．

参考文献

[1] Warren, W., Weigle, E., Feng, W. :High-Density Computing : A 240-Node Beowulf in One Cube Meter. Super Computing. CD-ROM(2002).

[2] 中島 浩, 中村 宏, 佐藤 三久, 朴 泰祐, 松岡 聡, 高橋 大介, 堀田 義彦, "高性能計算のための低電力・高密度クラスタ MegaProto", 情報処理学会論文誌コンピューティングシステム, Vol.46, No. SIG12 (ACS 11), pp. 46–61(2005).

[3] 佐々木 徹, 村上 和彰 :科学計算専用計算機のプラットフォームシステム，日本コンピュータ化学会，Vol. 4 No. 4,pp. 139–145(2005).

[4] 梅田 宏明, 稲富 雄一, 本田 宏明, 長嶋 雲兵：分子軌道計算専用計算機のためのフォック行列並列計算アルゴリズムの開発，Vol. 4 No. 4 ,pp. 179–187(2005).

[5] Schmidt, M., Baldridge, K., Boatz, J., Elbert, S., Gordon, M., Jensen, J., Koseki, S., Matsunaga, N., Nguyen, A., Su, S., Windus, T., Dupuis, M., Montgomery, J. :General atomic and molecular electronic structure system. Journal of Computational Chemistry Vol. 14, Issue 11, pp 1347–1363(1993).

[6] 林 徹生, 本田 宏明, 稲富 雄一, 井上 弘士, 村上 和彰, "Cell プロセッサへの分子軌道法プログラムの実装と評価", 情報処理学会研究報告, 2006-HPC-103, pp. 103-108(2006).