革新的シミュレーションソフトウェアの性能測定

加藤千幸

東京大学生産技術研究所
革新的シミュレーション研究センター

１．はじめに

革新的シミュレーション研究センター（以下、当センター）では，HA8000 クラスタシステムを活用し,ものづくりからバイオテクノロジ,ナノテクノロジまで幅広い分野のアプリケーションソフトに対するベンチマークテストおよび検証計算を実施した。使用したアプリケーションソフトは,文部科学省次世代 IT 基盤構築のための研究開発「革新的シミュレーションソフトウェアの研究開発プロジェクト」で開発された。これらのアプリケーションソフトウェアは，当センターがフリーソフトウェアとして公開しており，現在は、次世代 IT 基盤構築のための研究開発「イノベーション基盤シミュレーションソフトウェアの研究開発」プロジェクトのもと，開発が継続されている。本稿では，流体解析ソフト FrontFlow/blue，FMO 法に基づく量子化学計算プログラム ABINIT-MP,Kohn-Sham-Roothaan 法に基づく密度汎関数法プログラム ProteinDF,ならびに第一原理シミュレーションソフト PHASE に関して，それぞれ 2 章，3 章，4 章，5 章において紹介する。

２．流体解析コードのベンチマークテストおよび基礎検証

２．１　流体解析コード FrontFlow/blue の概要

FrontFlow/blue（FFB）は文部科学省次世代 IT 基盤構築のための研究開発「革新的シミュレーションソフトウェアの研究開発プロジェクト」において開発された流体解析ソフトウェアである。FFB の最大の特長は，乱流現象を高精度に予測できることにある。従来の流体解析では，時間平均に基づく Reynolds Averaged Navier-Stokes（RANS）が主に用いられていたため，流れの非定常性が現象の本質を支配する問題（例えば、振動や騒音）を取り扱うことができなかった。一方，FFB は非定常流れを高精度に予測することができる Large Eddy Simulation（LES）をベースとしているため，これまで予測することができなかった，機械内部の圧力変動スペクトルや空力騒音スペクトルの高精度予測が可能である。ここで，LES とは計算格子により解像することができる計算格子スケールよりも大きな渦（**Large Eddy**）の非定常の挙動を直接計算（**Simulation**）する手法である。計算格子スケールよりも小さな渦の挙動はサブグリッドスケールモデルによりモデル化される。

FFB のもうひとつの特長は“高速性”である。FFB はスカラマシン，ベクトルマシンにおいて，それぞれの特徴に応じて，マシンの性能を最大限に引き出すように設計されている。これにより大規模な解析を現実的な計算時間（例えば，ターボマシン 1 回転の計算が数時間）で実行することが可能となっている。現状では，スカラマシンにおいて，約 100CPU を用いて，1 千万点規模の六面体計算格子を用いた計算が可能である。また，ベクトルマシンにおいては約 4,000CPU を用いて，10 億点規模の六面体格子を用いた計算が可能である。LES は RANS と比較し，細かな計算格子が求められるため，実用上，その計算コストが問題になることがしばしば

ある。FFB が LES の実用的な問題への適用において多くの実績がある理由は FFB の“高速性”によるところが大きい。

２．２　流体解析コード FrontFlow/blue のベンチマークテスト

HA8000 システムに FrontFlow/blue ver.5.2 (FFB) をインストールし、六面体ソルバーles3c のベンチマークテストを行なった。使用したコンパイラは日立製コンパイラ mpif90 で、コンパイラオプションには-Oss を使用した。ノード内のコアの使用方法としては、全てのコアにプロセスを割り当てるフラット MPI と、CPU (4 コア) にプロセスを割り当て、CPU 内 (コア間) には自動並列化機能を用いるハイブリッド MPI の二通りをテストした。詳細は割愛するが、上記テストにおいて適切に NUMA 最適化を施す必要あることが確認された。テストデータは、ノードあたり百万要素に固定した。すなわち、全体のデータサイズは、ノード数に比例して大きくなる。ノード数として、1、2、4、8、16、32、64、128、256 をテストした。テストデータの全体要素数は最大で約 2.6 億点である。ステップあたりの計算時間を測定し、測定時間およびソルバーの演算回数より、実効性能を算出した。第 1 表および第 1 図に、ベンチマークテストの結果を示す。本ベンチマークテストの結果を以下にまとめる。

■単体の実効性能は約 3%であった。

■フラット MPI を用いたほうが、ハイブリッド MPI より若干性能が高かった。

■フラット MPI では 256 ノード(4096 コア)まで、ハイブリッド MPI では 128 ノード (2048 コア) まで高い並列性能を確認できた。

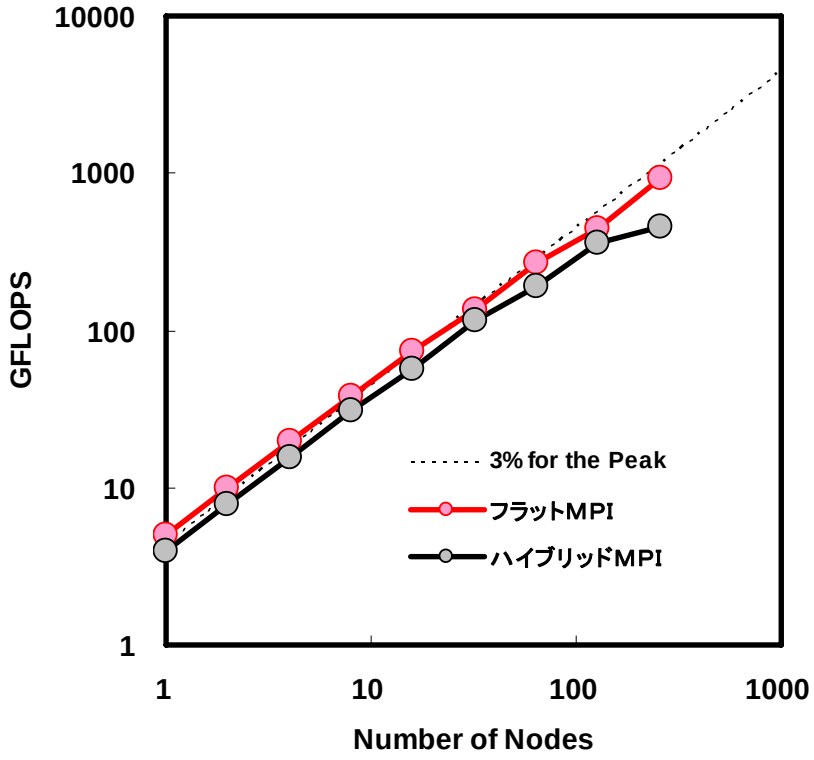
■プロセス数が多くなると，入出力に費やされる時間が 20 分程度となり，無視できなくなることが確認できた (第 2 図参照)。

今後の課題としては、単体実行効率および実用問題への応用が挙げられる。

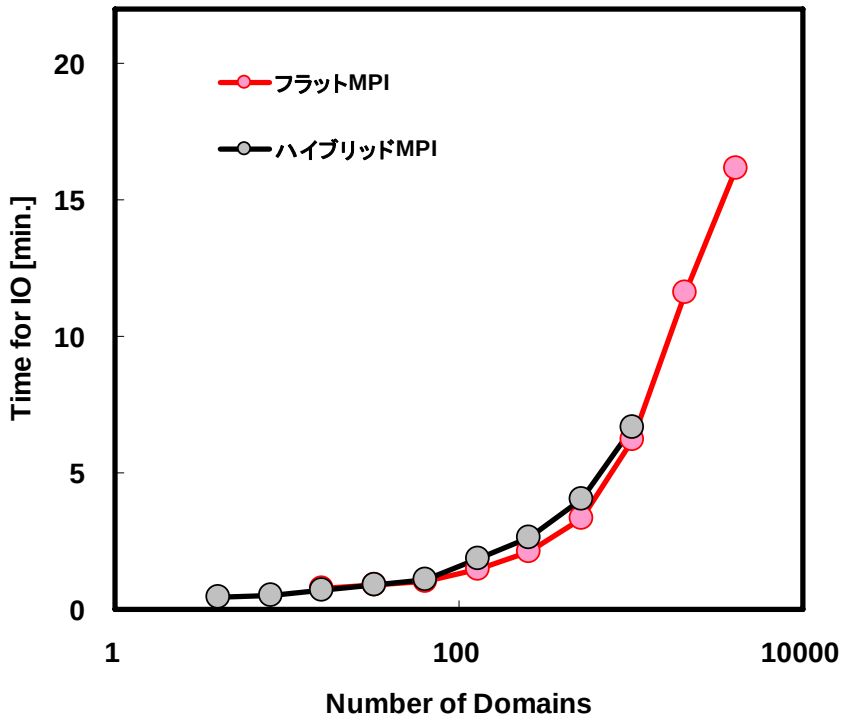
第 1 表：　HA8000 システムにおける FFB ベンチマークテスト結果

テストモード	ノード数	プロセス数	計算時間　(sec)		実行性能	
			0step	100step	GFLOPS	対ピーク (%)
フラット MPI	1	16	47	442	5.1	3.4
	2	32	55	454	10.0	3.4
	4	64	63	473	19.5	3.3
	8	128	87	509	37.9	3.2
	16	256	128	556	74.8	3.2
	32	512	201	678	134.2	2.8
	64	1024	374	855	266.1	2.8
	128	2048	698	1,277	442.1	2.3
	256	4096	968	1,525	919.2	2.4
ハイブリッド MPI	1	4	27	535	3.9	2.7
	2	8	30	540	7.8	2.7
	4	16	41	552	15.7	2.7
	8	32	53	577	30.5	2.6
	16	64	66	624	57.3	2.4
	32	128	111	666	115.3	2.4
	64	256	157	827	191.0	2.0

	128	512	244	964	355.6	1.9
	256	1024	399	1,525	454.7	1.2



第 1 図： HA8000 における FFB ベンチマークテスト結果



第 2 図： FFB ベンチマークテストにおいてファイル IO に費やした時間

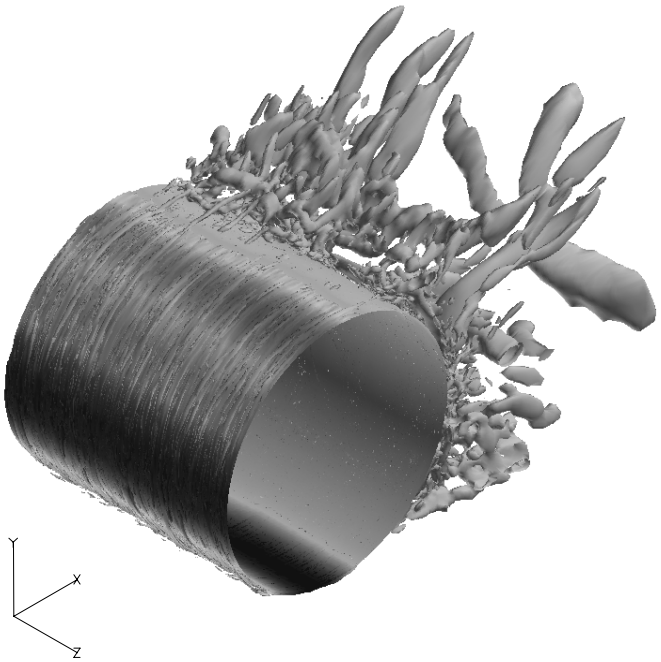
２．３　円柱周り流れ解析

HA8000 におけるFFBの検証計算のため，FFBを用いて円柱まわり流れ解析を実施した。円柱直径および主流流速をベースとするレイノルズ数は 8.5x10^{5}であり、このレイノルズ数では、流れははく離前に乱流遷移する。上記の高レイノルズ数における数値計算例はまれであり、流れの構造やメカニズムを理解するうえで本検証計算が有用である。本検証計算の計算条件をにまとめる。

第 2 表：　FFB による円柱周り流れ解析の計算条件

項目	内容
ソルバー	FFB 六面体ソルバー
計算格子	六面体　約 10 万要素
コンパイラ	日立製フォートランコンパイラ
使用ノード数	HA8000　16 ノード　（ハイブリッド MPI）
乱流モデル	Detached Eddy Simulation[(1)]

2000 ステップ計算するための計算時間は約 4 時間で，実効性能は約 28.5GFLOPS であった。計算結果の一例として、第 3 図に DES により計算された瞬時流れ場の渦構造を示す。渦構造は 2 次不変量の等値面として表現されている。



第 3 図：　レイノルズ数 8.5x10^{5}の円柱まわり流れ瞬時流れ場の渦構造（2 次不変量等値面）

２．４　空力騒音の直接計算

空力騒音の予測手法は分離計算と直接計算のふたつに大別される。分離計算では，音場が流れ場に影響を無視することにより流れ場と音場を別々に計算する。すなわち，はじめに音源データを得るために流れ場の計算を行い，その後に，音の伝播を計算する。一方，直接解法では，流れ場と音場を同時に計算する。FFB では分離計算による空力騒音予測を行う。分離計算は，直接解法と比べ計算コストが低いため，流れの速さが音速と比べ比較的小さい低マッハ数流れから発生する空力騒音予測では有効である。しかしながら，音場が速度場に与える影響が無視できないキャビティ音等の現象では，直接解法による空力騒音予測が必要となる。筆者らは，空力騒音の直接計算にも取り組んでいる。本節では，HA8000 システムを用いて計算を実施した空力騒音の直接計算の結果について紹介する。

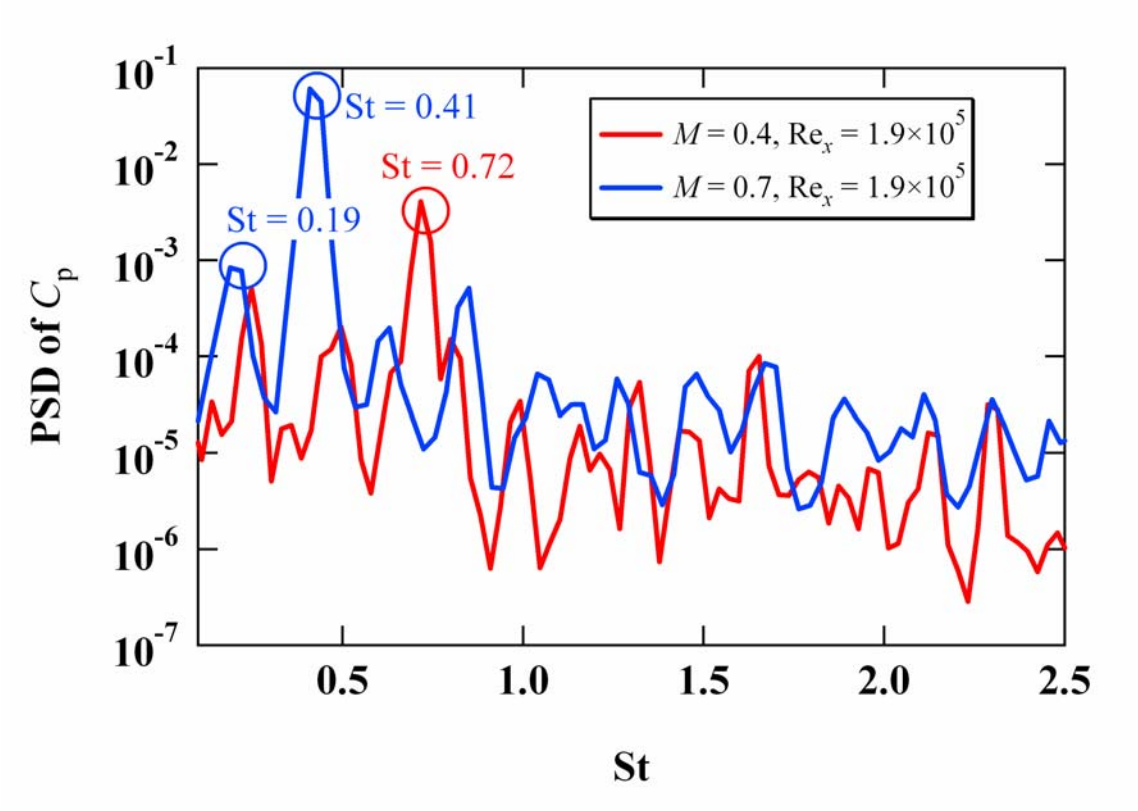
空力騒音は空気の流れ中の非定常運動から発生する音であり，音のパワーは速度の 4〜8 乗に比例し増大する。このため，近年の輸送機関の高速化において，空力騒音は大きな問題となっている。 特に，キャビティ上の流れでは，ピーク性の強い音が発生し，160dB に達する場合もある[(2)]。 こうしたキャビティ音は飛行機の着陸装置格納部や新幹線の車両車間部などから発生し，設計段階での予測が求められている。

キャビティ音の発生機構について，Rossiter[(3)] はキャビティ前縁で発生した渦が後縁にぶつかる際音が発生し，発生した音が前縁での渦形成を誘発するフィードバックループ(Fluid-acoustics interaction) を提案している。 Rossiter はこのフィードバック機構に基づくピーク音の周波数予測式を提案し，乱流境界層中のキャビティから発生するピーク音の周波数と一致することを確かめている。 また，Forestier は高速度シュリーレン法および Laser-Doppler Velocimetry (LDV) をもちい，乱流境界層中のキャビティのせん断層内に乱流の微小渦にくらべ大規模な渦が形成されることを明らかにしている。しかし，音波による大規模渦構造の形成機構や音波の発生機構は明らかにされていない。 本研究の目的は， 乱流境界層中のキャビティにおける Fluid-acoustics interaction を詳細に明らかにすることである。

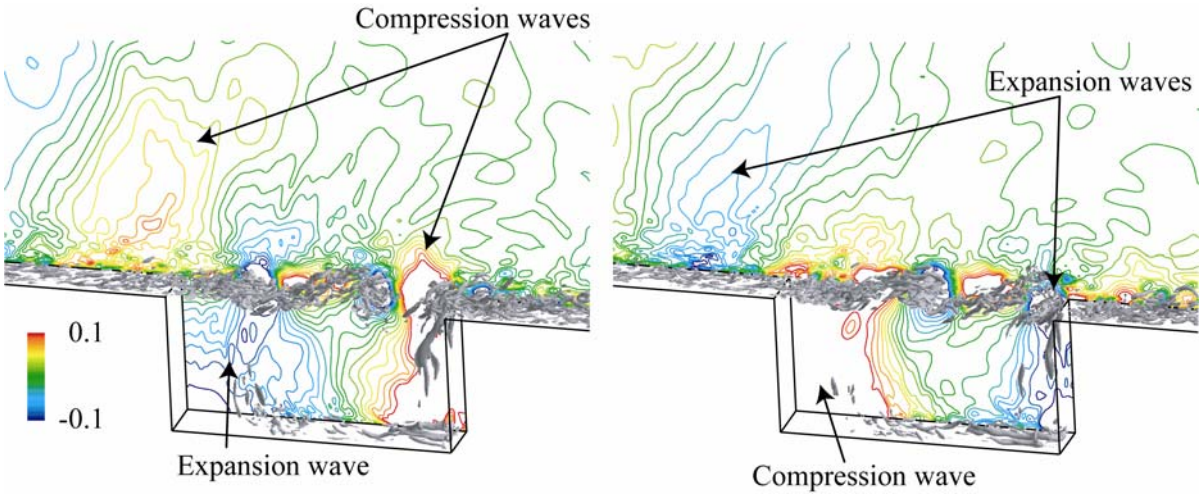
キャビティ底部におけるスパン方向に平均した圧力係数 のPower Spectral Density (PSD) をマッハ数 M= 0.4, 0.7 について第 4 図に示す.ここで，フーリエ変換にもちいたデータ長さは各マッハ数の主要なピーク周波数の 16 周期分とした。 各マッハ数において周波数 がSt = $fD/u_{1\infty}$ = 0.72 (M = 0.4)およびSt = 0.41 (M = 0.7)である強いピーク音が発生することがわかった。

第 5 図にマッハ数M = 0.4 における第 2 不変量の等値面および圧力係数Cp’の各地点における時間平均値と瞬時値の差を等高線にあらわす。乱流境界層中の微少な縦渦構造はキャビティ前縁においてはく離し，x_2方向の速度こう配により発生するx_3方向渦構造と重なり，渦構造は活発になる。また，せん断層内では，二次元的な大規模渦構造が生じおり，渦構造内では低圧領域が形成される.その結果，微小渦構造は大規模渦構造内で密になり，高圧領域では引き伸ばされリブ構造を形成する。

キャビティ内部のせん断層部以外での圧力変動は渦構造による変動は小さく，音波を表している。第 5 図より二次元大規模構造が後縁にぶつかる際に膨張波が発生することがわかる。 発生した音波はキャビティ内部を上流へ伝播し，前縁においてキャビティ外部へ放出される。



第 4 図：　圧力スペクトル



第 5 図：　圧力変動および渦構造

３．FMO 法に基づく量子化学計算プログラム ABINIT-MP の並列計算性能評価

３．１ はじめに

ABINIT-MPは「革新的シミュレーションソフトウェアの開発(RSS21)[1]」プロジェクトにおいて開発されたタンパク質-化学物質相互作用解析システムBioStationの中核となる量子化学計算プログラムである。ABINIT-MPの最大の特徴は，タンパク質-化学物質間の相互作用を量子力学に基づいて計算する点である。このため，古典力場では精度のよい計算が困難なタンパク質-化学物質間の相互作用を精密に計算できる。

通常タンパク質の量子化学計算は非常に困難であるが，非経験的フラグメント分子軌道法(Fragment Molecular Orbital Method:以下 FMO 法と略)[4-10]を採用しているため，ABINIT-MP はタンパク質の量子化学計算も現実的な時間(30 並列計算で半日～1 日程度)で可能となっている。FMO 法を採用したため，化学物質-タンパク質の結合様式をアミノ酸残基単位で解析できるようになっているのも ABINIT-MP の大きな特徴である。結合様式の詳細な情報は，医薬品等の分子設計において非常に有用である。このため，BioStation は量子論に基づく高精度なタンパク質-化学物質の相互作用解析システムとして創薬の現場において役立ちつつある[11,12]。

今回は，2008 年３月に公開された ABINIT-MP ver.4.1(最新版)の並列計算性能を調査するために，グリシン 128 量体(899 原子)に対して FMO-MP2/6-31G 計算を HA8000 上で実行した。その結果を報告する。以下,3.2 節「FMO 法に基づく相互作用解析」において FMO 法の概略と ABINIT-MP での並列計算方法ついて解説し，3.3 節「HA8000 における ABINIT-MP の性能測定結果」で今回の性能調査結果を報告する。3.4 節「まとめ」では，調査結果をまとめる。

３．２ FMO 法に基づく相互作用解析

３．２．１ FMO 法計算の流れ(FMO-Hartree-Fock 計算)

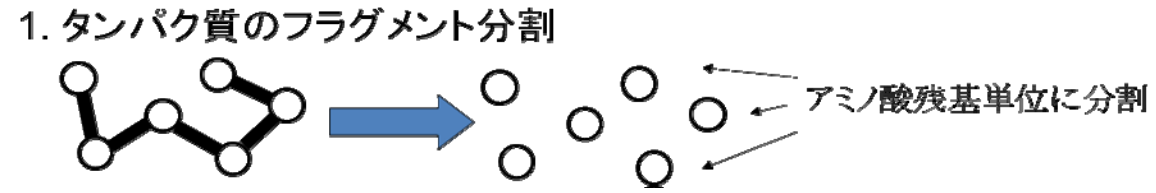
タンパク質の FMO 法計算は，3 段階に分けて行う。まず，タンパク質を構成要素であるアミノ酸残基単位に分割する(第 6 図参照)。

続いて，各フラグメントにおいて Hartree-Fock(HF)計算を実行する(Monomer Self-Consistent Field 計算;以下 Monomer SCF 計算と略)。この際，周囲フラグメントからの静電ポテンシャルを考慮していることに注意する。全てのフラグメントに対し分子軌道計算を実行し，分子全体の電子密度が自己無撞着になるまで iteration を行う(第 7 図参照)。

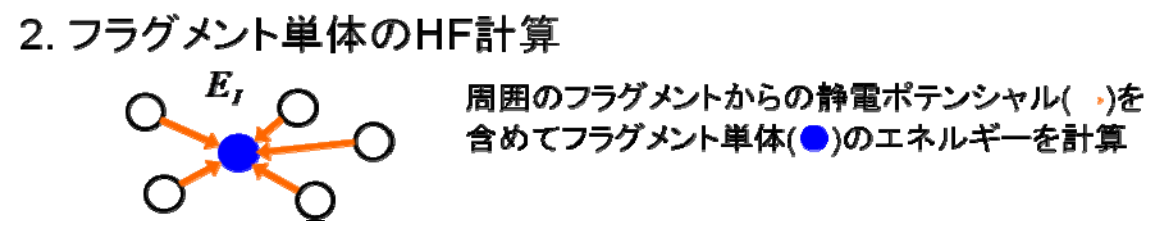
最後に，フラグメントペアに対しても HF 計算を行う(Dimer Self-Consistent Filed 計算;以下 Dimer SCF 計算と略)。この際，周囲のフラグメント(モノマー)からの静電ポテンシャルを考慮する(第 8 図参照)。

Dimer SCF計算は互いに近距離にあるフラグメントペアに対してのみ行い，遠く離れたフラグメント間ペアに対しては,静電相互作用項のみを考慮した近似計算(Dimer静電近似)を実行する。このため，FMO法全体としての計算時間は，ほぼO(N^1～N^2)と非常に高速になっている。

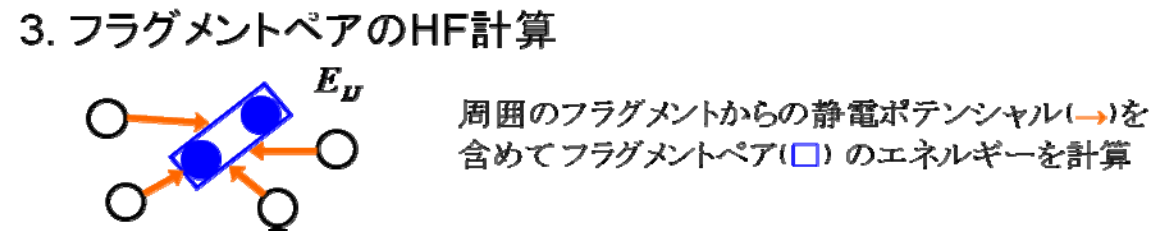
[1] http://www.ciss.iis.u-tokyo.ac.jp/rss21/



第 6 図:タンパク質のフラグメント分割



第 7 図:モノマーに対する SCF 計算



第 8 図:ダイマーに対する SCF 計算

３．２．２ MP2 計算による分散力の記述

今回並列性能調査を行う際に用いる計算手法は，MP2 (Møller-Plesset の２次摂動)計算である。MP2 計算とは，HF 波動関数を無摂動波動関数，静電相互作用から HF 平均場を引いたものを摂動項として取り扱った二次摂動計算のことである。FMO 法のフレームワークでは，フラグメントおよびフラグメントペア に対して MP2 計算を行う[13-15]。

タンパク質中にはトリプトファン，フェニルアラニンなどといった環状側鎖を持つアミノ酸残基が存在することが多い。このような環状側鎖中のπ軌道と化学物質との間には，π-π相互作用や CH/π相互作用といった分散力が働き，タンパク質-化学物質の結合において重要な役割を果たしている。分散力の記述は HF 計算ではできないため，電子相関を考慮した計算手法が必要となる。電子相関を考慮した計算の中でも MP2 計算は，計算コストが他の手法に比して軽いため，タンパク質という大規模分子における分散力の記述に有効であると考えられている。

３．２．３ ABINIT-MP における FMO 法の並列化実装

ABINIT-MP は，FMO 法の実装の際に二段階の並列化を行っている。即ち，フラグメント(もしくはフラグメントペア)に対する並列化とフラグメント(もしくはフラグメントペア)内の並列化である。ABINIT-MP では，各フラグメント独立に monomer SCF 計算および monomer MP2 計算を行うことが可能なので，非常に高い並列計算性能を出すことができる。また、各フラグメントペアに対しても同様に独立に dimer SCF 計算，dimer 静電近似計算を実行することが可能である。このため，FMO 法全体としても非常に高い並列計算性能を示す(3.3 節参照)。

３．３ HA8000 における ABINIT-MP の性能測定結果

３．３．１ 計算条件

ABINIT-MP のコンパイル方法、計算対象分子や計算条件について説明する。
ABINIT-MP のコンパイルには,日立製最適化フォートランを使用し,最適化レベル O4 でコンパイルした。また,並列化ライブラリは MPICH1.2 を用いた。コンパイル条件を第 3 表にまとめる。

第 3 表:使用したコンパイラ及びコンパイルオプション

コンパイラ	日立製作所最適化フォートラン f90
最適化オプション	O4
並列化ライブラリ	MPICH 1.2

計算対象分子としてグリシン 128 量体(全 899 原子:第 9 図参照)を採用し,計算手法/基底関数は MP2/6-31G 計算を採用した。フラグメント分割の際，フラグメントあたり 1 アミノ酸残基を割り当て，8 コアを割り当てた。また，1 コアあたり 1500MB のメモリ容量を使用した。第 4 表に計算条件をまとめる。

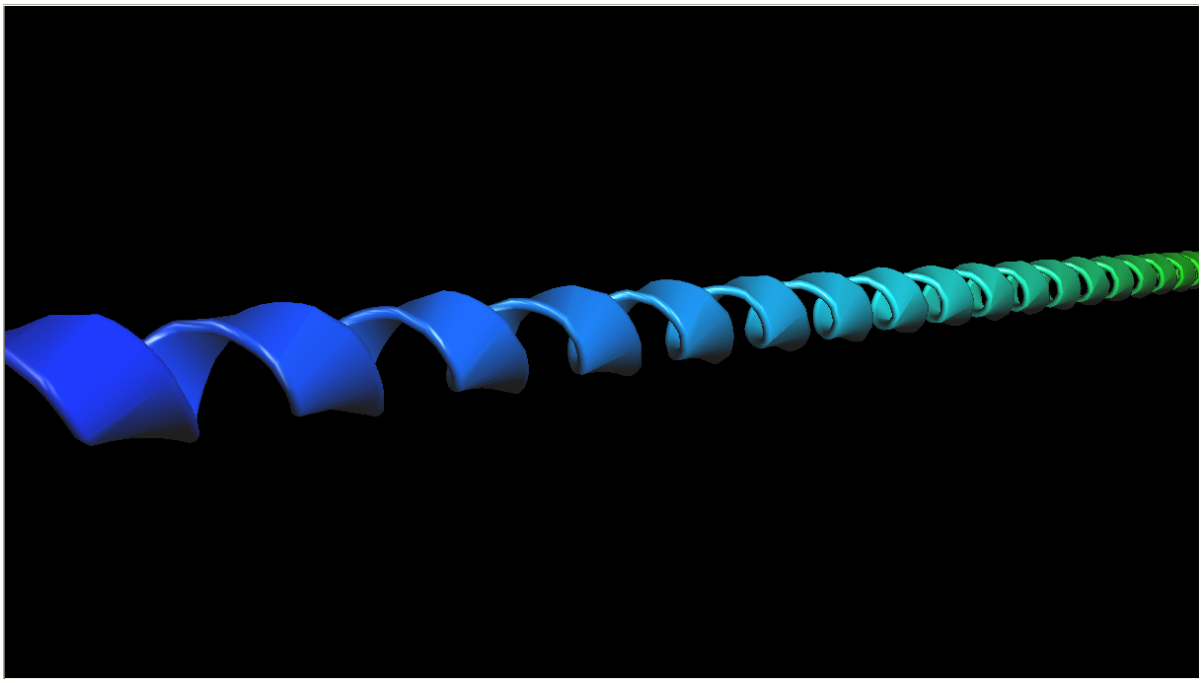
第 4 表: 計算条件

計算手法/基底関数	MP2/6-31G
フラグメントのサイズ	1 アミノ酸残基
コア数/フラグメント	8
メモリ容量/コア	1500MB

また,参考のため実際に使用したジョブ投入スクリプト例を以下(第 5 表)に示す。

第 5 表: 16 ノード 128 コア使用時の計算投入スクリプト

```
#!/bin.bash
#@$ -q b0n_16
#@$ -N 16  ←――――  16 ノード使用
#@$ -J T8  ←――――   8 コア/ノードを使用
#@$ -lT 1:00:00
cd /home/kobayasi/JOB
mpirun -np 128  abinitmp.exe < ./input/Gly128_MP2_6-31G.ajf > ./output/Gly128.out
```



第 9 図：性能調査に用いたグリシン 128 量体(α-helix)。リボン表示で図示する。

３．３．２ ABINIT-MP の性能測定結果

３．３．１の計算条件で 1,2,4,8 ノードを使用した計算を実行し，第 6 表の測定結果を得た。並列化率 α は，コア数 p の時の計算時間 $T(p)$ がアムダールの法則，即ち

$$\frac{T(p)}{T(1)} = 1 - \alpha + \alpha/p$$

に従うと仮定し，測定された計算時間を用いて最小二乗フィットした。この結果，得られた並列化率は $\alpha = 0.9994$ (99.94%)であった。

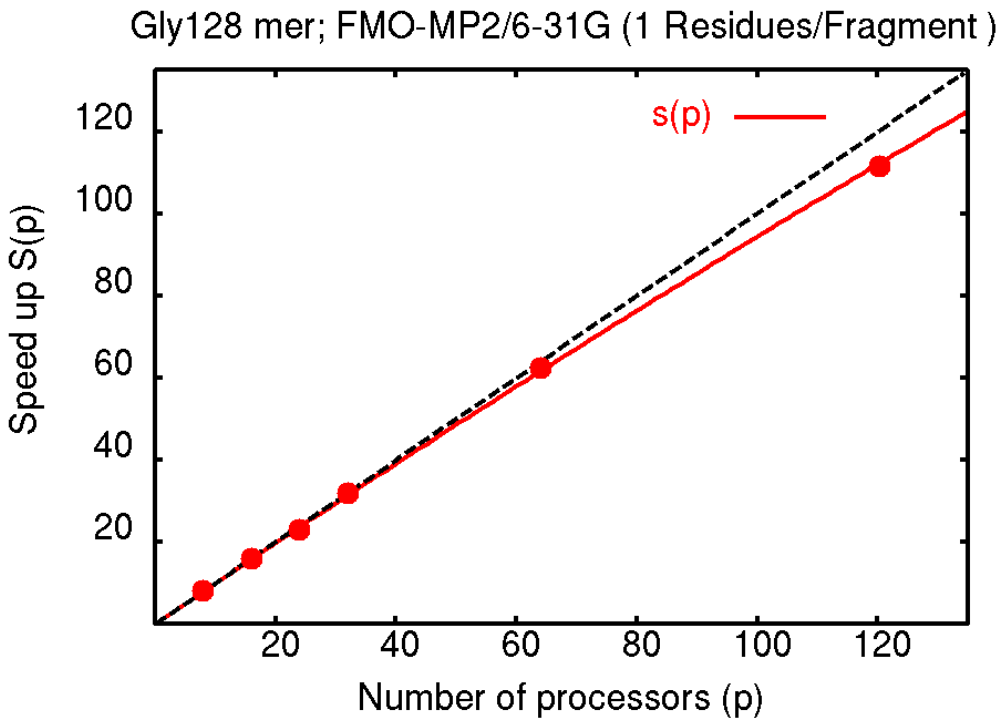
第 6 表：Gly128 量体の FMO-MP2/6-31G 計算

Gly128 残基(原子数=899,原子軌道=5389)の FMO-MP2/6-31G 計算		
ノード数	コア数	計算時間
1	8	11240.3
2	16	5638.9
4	32	2833.1
8	64	1444.8
16	128	740.0
ABINIT-MP の並列化率 99.94%		

また,測定結果から得られた高速化率を図示する(第 10 図参照)。図中には高速化 $S(p)$ 率もアムダールの法則に従うと仮定し，

$$S(p) = \frac{T(1)}{T(p)} = \frac{1}{1 - \alpha + \alpha/p}$$

となる曲線を赤字で示している。



第 10 図:HA8000 における ABINIT-MP の高速化率。横軸はプロセッサ数,縦軸は高速化率を示す。

また，測定されたFlops数を第 7 表にまとめる。ピーク性能比は，測定されたFlops数と理論性能(=9.2GFlops×コア数)との比から算出した。

第 7 表:ABINIT-MP の Flops 数とピーク性能比

Gly128 残基(原子数=899,原子軌道=5389)の FMO-MP2/6-31G 計算			
ノード数	コア数	GFlops	ピーク性能比(%)
1	8	3.69	5.02%
2	16	7.35	4.99%
4	32	14.63	4.99%
8	64	28.82	4.89%
16	128	56.36	5.02%

３．４ まとめ

ABINIT-MP の HA8000 上における並列計算性能を調査するため,Gly128 量体の FMO-MP2/6-31G 計算を 1(8 コア),2(16 コア),4(32 コア),8(64 コア),16(128 コア)ノードを用いて行った。
測定された結果から，ABINIT-MP の HA8000 上での並列化率は 99.94%であり,演算性能はピーク性能の 5%程度であることが判明した。

このことから，現在の ABINIT-MP は,1,000 並列(100 ノード規模の計算)という大規模並列計算においても十分な性能(並列化効率 50%以上)を出すと期待される。

今後 ABINIT-MP は，10,000～100,000 並列計算で十分な計算性能を発揮するため，更なる並列化率向上を目指したチューニングを行う予定である。

４．　ProteinDF

ProteinDF は，タンパク質全電子計算を目的とした Kohn-Sham-Roothaan 法に基づく密度汎関数法プログラムである。Kohn-Sham-Roothaan 法は分子積分演算，交換相関積分演算と一般行列演算からなる行列方程式である。巨大生体分子であるタンパク質の電子状態を高速かつ高精度にシミュレーションするために，ProteinDF では様々な高速化・並列化を施している。開発はオブジェクト指向言語 C++を利用し，行列演算ライブラリとして LAPACK，ScaLAPACK を利用している。

現在，生体分子の構造データベース PDB に登録されているタンパク質の 99%以上は 1,000 残基以内に収まる。すなわち，1,000 残基程度のタンパク質の全電子シミュレーションが可能になれば，ほとんどのタンパク質を網羅することができることになる。

このサイズのタンパク質の全電子計算を，スプリットバレンス型の基底関数を用いて実行すると行列次元数は約 100,000 になる。これを倍精度の密行列で保持するために必要なメモリサイズは，行列一つにつき 80 GB 程になる。行列演算，例えば行列積(A×B=C)には最低 3 つの行列が必要であるので，240 GB のメモリが必要ということになる。ProteinDF では RI 法を用いるため，さらに 2～3 倍次元程度の大きな補助基底関数を必要とする。到底，分散メモリ型並列計算機 1 ノードに収まるメモリサイズではない。

このような大規模メモリが必要な行列演算に対して，ProteinDF では巨大行列を並列計算機の各ノードへ分散保持させる戦略を採った。行列データの分配方法として ScaLAPACK で採用されている方式を利用した。この行列保持方式を採用にするにあたり，独自の分子積分演算リーチンの並列化方法を見直す必要があったが，一般行列演算には ScaLAPACK を利用することができた。ScaLAPACK は様々な計算機ごとに適したチューニングが行われ，ライブラリとして利用できるため，ScaLAPACK の採用は開発コスト・性能の両面から利点があった。

上記方法によりいくつかの巨大行列を並列計算機内で分散保持することが可能になったが，これでも量子化学計算で用いられる SCF 繰り返し計算に必要な行列を全てメモリ上に確保することはできない。そこで ProteinDF では，直近の演算に必要な行列以外はメモリ以外のデバイス，すなわちディスク上に退避することで巨大分子の量子化学計算を可能にしている。したがって，行列操作・演算が必要になるたびにディスクへの I/O と各ノード間通信とが発生することになり，計算効率はこれらの性能に大きく左右される。

ProteinDF プログラムは HA8000 システム上にてトラブルなくビルドし，シミュレーションを行うことができた。大小いくつかのデータセットにてシミュレーションを実行した結果、残念ながら今回はディスク I/O が足枷となり本システムのパフォーマンスを発揮するに至らなかった。一方，他システムの ProteinDF ベンチマークから，10,000 並列でも性能を発揮できると予想される良好な結果が得られている。HA8000 システムでは，各ノードに実装された高速な一時ディスク領域を利用することができる。今後，分散行列を各ノードの一時ディスク領域へ退避する等の仕組みを ProteinDF に導入することにより計算効率を大幅に改善できると考えられる。

５．第一原理シミュレーションソフト PHASE

密度汎関数法と擬ポテンシャル平面波手法に基づく第一原理シミュレーションソフトPHASE は様々な物質の性質の解析を可能にする。特に，半導体デバイス材料のシミュレーションにその威力を発揮してきた。たとえば，トランジスター向けの誘電体材料として期待されているア

モルファス・ハフニウム・アルミニウム酸化物(HfAlO)やセリウム酸化物の誘電物性解析を世界に先駆けて行ってきた[16,17]。また，半導体不純物系の 1 万原子を越えるモデルの大規模電子状態計算を地球シミュレータの 512 ノードを用いて行い，16.2 Tflop/sの性能を達成し，SC07のゴードンベル賞の最終選考に残った実績[2]がある。地球シミュレータとは異なるアーキテクチャーのHA8000 で，同様な大規模電子状態計算を行うためにはプログラムコードの最適化が必要である。今回は，コンパイラの選定とキャッシュチューニングの調整を行い，SR11000 との性能比較を行ったので，それらについて報告する。

PHASE コードは Fortarn90 で書かれており，HA8000 で利用できるフォートランコンパイラは日立製最適化フォートランコンパイラとインテルフォートランコンパイラである。日立製コンパイラを用いる場合は，高速フーリエ変換ライブラリとして MATRIX/MPP ライブラリを採用し，LAPACK ライブラリは情報基盤センターが提供するものを使用した。使用したメイクファイルは SR11000 向けものとほぼ同じであるが,MPI ライブラリのディレクトリの指定と C 言語の関数の名前変換方法を変更したので，変更した箇所を以下に示す。

```
F90FLAGS = -64 -Oss -i,P -msg=w -parallel -I/opt/mpich-mx/include
F77FLAGS = -64 -Oss -i,P -msg=w -parallel -I/opt/mpich-mx/include
CFLAGS = -O0 -DINTEL -64
```

インテルコンパイラを用いる場合は，EM64T/AMD64 向けのメイクファイルを書き換えて使用した。高速フーリエ変換ライブラリとして情報基盤センターが提供するFFTW3 ライブラリ[3]とインテルマスカーネルライブラリに含まれるLAPACKライブラリを用いた。変更した箇所を以下に示す。

```
F90FLAGS = -w90 -w95 -W0 -traceback -I/opt/itc/mpi/mpich-mx-intel/include
F77FLAGS = -w90 -w95 -W0 -traceback -I/opt/itc/mpi/mpich-mx-intel/include
LIBS = -Wl,-rpath,/opt/intel/mkl/10.0.3.020/lib/em64t
       -L/opt/intel/mkl/10.0.3.020/lib/em64t -Wl,--start-group
       /opt/intel/mkl/10.0.3.020/lib/em64t/libmkl_intel_lp64.a
       /opt/intel/mkl/10.0.3.020/lib/em64t/libmkl_sequential.a
       /opt/intel/mkl/10.0.3.020/lib/em64t/libmkl_core.a -Wl,--end-group
       -L/opt/itc/lib -lfftw3 -L/opt/itc/mpi/mpich-mx-intel/lib -lmpich
       -Wl,-rpath,/opt/mx/lib64,-rpath,/opt/mx/lib -L/opt/mx/lib64 -L/opt/mx/lib/
       -lmyriexpress -lrt -lfmpich -Bdynamic -lpthread
```

半導体デバイス材料として用いられるアモルファス酸化物$HfSiO_2$(第 11 図)を計算対象とし，HA8000 においてPHASEの性能調査を行った。比較的小規模である 192 原子系の計算における 1 回の波動関数更新に要する時間の計測結果を第 12 図に示す。日立製コンパイラとインテルコンパイラを用いて作成したプログラムをMPI並列で実行したが，日立製コンパイラの場合はMPIの

[2] http://sc07.supercomputing.org/schedule/event_detail.php?evid=11132
[3] http://www.fftw.org/

みの並列実行のほかにCPU内でコア間スレッド並列化を行うハイブリッド 4x4 並列実行も行った。MPI並列のみで1ノード(16MPI並列)で実行した結果を比較すると，日立製コンパイラよりもインテルコンパイラの方が約20秒速い。ノード数を増やすと両方とも実行時間が減少する。8ノードでの並列加速率は日立製コンパイラが3.6倍で,インテルコンパイラが3.9倍である。日立製コンパイラのハイブリッド4x4並列実行は1ノードでMPI並列のみの実行よりも約10秒遅いが,スケールが良いため8ノードでインテルコンパイラのMPI並列のみ実行よりも速くなり,その並列加速率は5.4倍である。

PHASEはベクトル計算機向けに開発されたが,公開されている最新コードではキャッシュチューニングが施されている。たとえば，波動関数$\phi_n(\mathbf{r})=\Sigma_j c(n,j)\exp(i\mathbf{g}_j\cdot\mathbf{r})$と擬ポテンシャルのプロジェクター関数$\beta_p(\mathbf{r})=\Sigma_j d(p,j)\exp(-i\mathbf{g}_j\cdot\mathbf{r})$の内積$\langle\beta_p|\phi_n\rangle$の計算はオリジナルコードでは以下に示すような3重ループで計算されている。

```
DO p = 1, Nprojector
        DO n = 1, Nband
                DO j = 1, Nplanewave
                        <βp|φn> = <βp|φn> + d(p,j) * c(n,j)
                END DO
        END DO
END DO
```

$N_{projector}$はプロジェクターの数,N_{band}は電子状態の数,$N_{planewave}$は平面波(基底関数)の数である。最内側ループはベクトル計算機ではベクトル化される非常に長いループである。キャッシュを使用するスカラー計算機では,このままのコードではキャッシュミスが頻発するためCPUの性能を十分に引き出せない。そこで，キャッシュ利用効率を上げて性能を引き出せるように最内側のループ長を調節した。部分配列d(p,j1:j2)とc(1:N_{band},j1:j2)がキャッシュに乗った状態で演算が行なわれるように変更したコードを以下に示す。

```
DO j1 = 1, Nplanewave, Ncache
        j2 = j1 - 1 + Ncache
        DO p = 1, Nprojector
                DO n = 1, Nband
                        DO j = j1, j2
                                <βp|φn> = <βp|φn> + d(p,j) * c(n,j)
                        END DO
                END DO
        END DO
END DO
```

N_{cache}は部分配列の合計サイズがキャッシュサイズの 3/4 になるように調整している。Itaniumプロセッサーでは実際のキャッシュサイズを用いるようにしてあるが，ベクトル機以外の他の

アーキテクチャーではキャッシュサイズを 1MBに固定している。キャッシュブサイズは入力ファイルのcontrolブロック内の変数cachesizeにkB単位で設定することができる。たとえば，キャッシュサイズを 1MBに設定するには，

```
Control{
    cachesize = 1024
}
```

とする。今回紹介した日立製コンパイラ向けメイクファイルでプログラムを作成すると，キャッシュサイズ変数 cachesize のデフォルト値はゼロになってしまうので，かならずキャッシュサイズ変数を適切に設定していただきたい。

HA8000 はコアあたり 512kB の L2 キャッシュをもつため，すべてのコンパイラと並列化手法の組み合わせでキャッシュサイズ変数を 512kB に設定した。しかし，それぞれの組み合わせに最適なブロッキングサイズがあるはずなので，8 ノードでプログラムを実行し，最適なブロッキングサイズの探索を行った。第 13 図にその結果を示す。MPI のみの並列化では日立，インテルともキャッシュサイズ変数が 256kB で実行時間が最小になる。一方，ハイブリッド 4x4 並列ではキャッシュサイズ変数を L2 キャッシュサイズ 512kB よりも大きく取った方が速くなる。キャッシュサイズ変数が 1MB と 2MB とで実行時間の差は殆どなく，キャッシュサイズ変数が 512kB と 1MB とで実行時間の差はわずか 1 秒である。従って，MPI 並列化のみの場合と異なりハイブリッド並列化ではブロッキングサイズにあまり依らずに良い性能が得られると結論できる。まとめると，HA8000 における最適な並列化手法は日立製コンパイラを用いたハイブリッド 4x4 並列化であり，キャッシュサイズ変数を 1MB 程度に設定すると良い。

擬ポテンシャル平面波法コードには平面波に関する長いループ構造があるため，ベクトル計算機に向いている。SR11000 はスカラー計算機であるが大きな L3 キャッシュを備えている上にバンド幅が太いため，あたかもベクトル計算機のように振る舞う。実際，HA8000 と SR11000 の 1 ノードあたりの理論性能値は同じであるにもかかわらず，96 原子系で HA8000 と SR11000 の性能を比較すると SR11000 の方が 2〜3 倍良い性能を示す（第 14 図）。ノード数が増えると性能差はわずかに縮まるが，SR11000 の優位性は変わらない。540 原子系では SR11000 はスケールが良く，2(8)ノードで HA8000 の 2.6(3.1)倍の性能を示す（第 15 図）。540 原子系で４k 点を用いる場合は，第 16 図に示すように HA8000 のスケールも良く，32 ノードを利用すれば，SR11000 の 8 ノード利用時の性能 231 Gflop/s よりも 83 Gflop/s 高い性能が得られる。

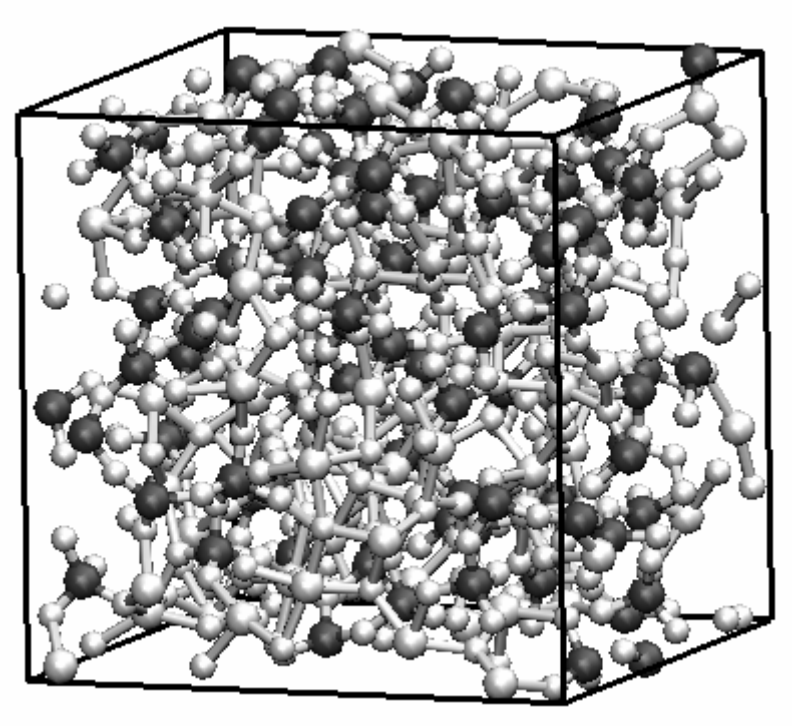
今回は個別ルーチンの性能評価まで踏み込んで行わなかった。また，詳細な報告は控えるが，数千原子系の大規模計算を HA8000 で実行するのは，現在のコードでは困難であることが判明した。地球シミュレータのようなベクトル並列計算機とは異なり，1000 MPI プロセスを超える並列実行を考慮したプログラム設計が必要であり，キャッシュ利用効率向上を目指して個別ルーチンの詳細な性能評価を行い，HA8000 のような超並列計算機向けにも PHASE コードを最適化していきたい。

最後に参考として， HA8000 ８ノードを用いてハイブリッド 4x4 並列で PHASE を実行するときのバッチスクリプトを紹介する。

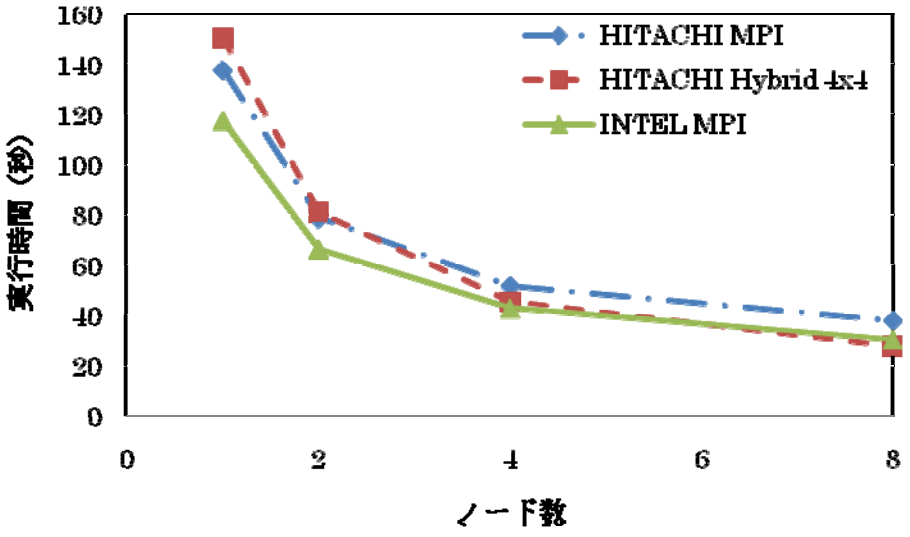
```
#!/bin/bash
#@$-q parallel
#@$-N 8
#@$-J T4
#@$-lT 8:00:00
export HF_PRUNST_THREADNUM=4

install=~/phase_v701/bin

idir=~/test/input
pdir=~/test/pp
odir=~/test/output
wdir=/tmp/work
bdir=/tmp/bin

cd $odir
cat <<here >stg.cfg
in $install/phase $bdir/phase
in $idir/nfinput.data $wdir/nfinput.data
in $idir/file_names.data $wdir/file_names.data
in $idir/continue.data $wdir/continue.data
here

echo 'start:' `date`
/opt/itc/bin/stagein $odir/stg.cfg
cp $pdir/* $wdir
cp $idir/continue_bin.data $wdir
cp $idir/nfchgt.data $wdir
cp $idir/zaj.data $wdir
cd $wdir
mpirun ~/numarun $bdir/phase
cp $wdir/* $odir
echo 'end:' `date`
```

このバッチスクリプトを用いれば，プログラム実行中の入出力はノード内で行われるので，ネットワークファイルシステムを用いるよりも良い性能が得られる。これをそのまま使うには，ディレクトリ input に入力ファイル nfinput.data と file_names.data を置き，ディレクトリ pp に必要な擬ポテンシャルファイルを置く必要がある。また，PHASE の実行ファイルは通常インストール位置~/phase_v701/bin になければならない。HA8000 利用の手引きに説明されている NUMA コントロールを行うスクリプト numarun がホームディレクトリ直下に置かれているとして

いる。継続計算を行うためには 4 つの出力ファイル continue.data, continue_bin.data, nfchgt.data,zaj.data を継続実行時に入力ファイルとしてワークディレクトリ$wdir に置かなければならない。上に示したバッチスクリプトにおいて，継続実行時の操作は太字で記述されているので，初期実行時には削除して使用していただきたい。PHASE を HA8000 で実行するときに，このバッチスクリプトを役立てていただければと思う。

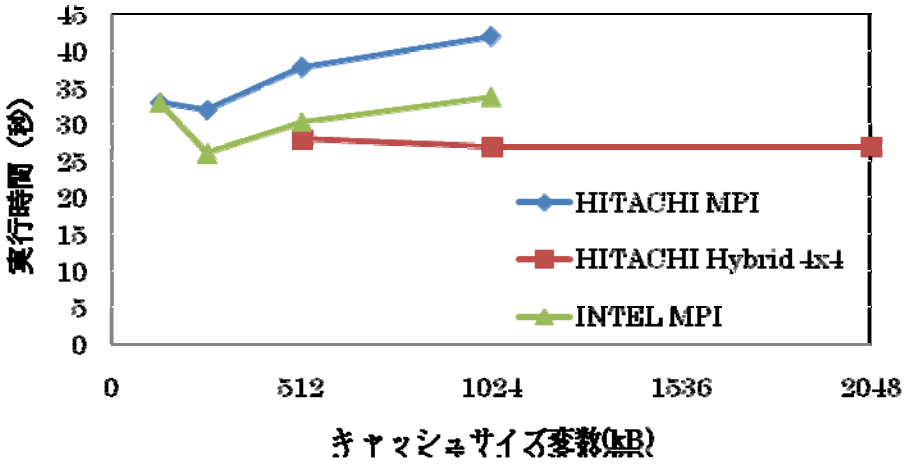


第 11 図　アモルファス$HfSiO_2$(540 原子)の構造モデル。
小さい白い球が酸素，大きい白い球がハフニウム，大きい灰色の球がシリコンである。

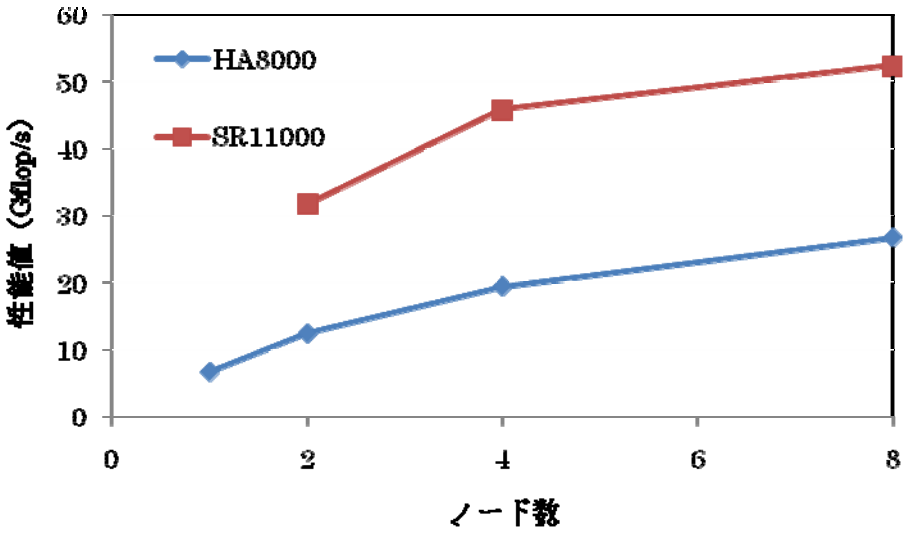


第 12 図：　$HfSiO_2$系(192 原子)の直線探索最小化の実行時間の並列度依存性。
コンパイラ(HITACHI と INTEL)，並列化方法(hybrid 4x4 と flat MPI)の組み合わせ毎にノード数に対する実

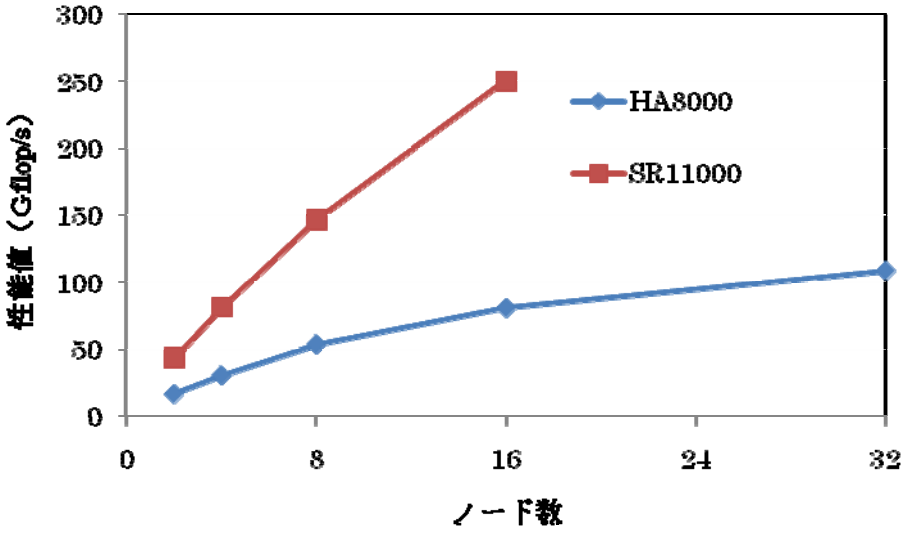
行時間の変化が示されている。キャッシュサイズ変数を 512kB に設定している。



第 13 図： $HfSiO_2$系(192 原子)の直線探索最小化の実行時間のキャッシュブロッキング依存性。
コンパイラ(HITACHI と INTEL)，並列化方法(hybrid 4x4 と flat MPI)の組み合わせ毎にキャッシュサイズ変数に対する実行時間の変化が示されている。８ノードで実行している。

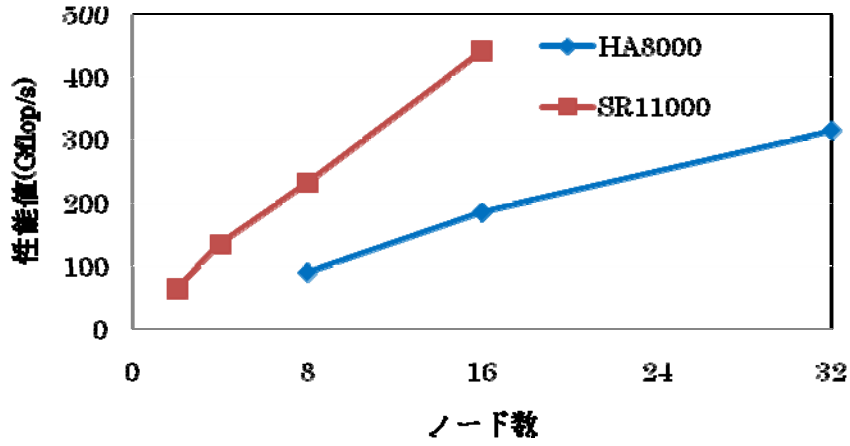


第 14 図　$HfSiO_2$系(96 原子，Γ点のみ)の電子状態計算の性能。
HA8000 と SR11000 の性能値を利用ノード数に対して示している。HA8000 では日立製コンパイラによるハイブリッド 4x4 並列化を採用し，キャッシュブサイズ変数を 1MB に設定している。



第 15 図： $HfSiO_2$系(540 原子, Γ点のみ) の電子状態計算の性能。

HA8000 と SR11000 の性能値を利用ノード数に対して示している。HA8000 では日立製コンパイラによるハイブリッド 4x4 並列化を採用し，キャッシュブロッキングのサイズを 1MB に設定している。



第 16 図： $HfSiO_2$系(540 原子, ４k点) の電子状態計算の性能。

HA8000 と SR11000 の性能値を利用ノード数に対して示している。HA8000 では日立製コンパイラによるハイブリッド 4x4 並列化を採用し，キャッシュサイズ変数を 1MB に設定している。

参 考 文 献

（１）Spalart, P. R., Jou, W.-H., Strelets, M., and Allmaras, S. R., 1997, “Comments on the Feasibility of LES for Wings, and on a Hybrid RANS/LES Approach”, In: Liu, C. and Liu, Z. (eds), Advances in DNS/LES, Proceedings of 1st AFOSR International Conference on DNS/LES, Ruston, Greyden Press, pp. 137-147.

（２）Krishnamurty, K., National Advisory Committee for Aeronautics Technical Report, No. 3487 (1955)

（３）Rossiter, J. E., Aeronautical Research Council, No. 3438 (1964)

（４）D. G. Fedorov, K. Kitaura, “Theoretical development of the fragment molecular orbital (FMO) method”, in Modern methods for theoretical physical chemistry of

biopolymers, E. B. Starikov, J. P. Lewis, S. Tanaka, Eds., Elsevier (2006) pp. 3-38.
（５）T. Nakano, Y. Mochizuki, K. Fukuzawa, S. Amari, S. Tanaka, “Developments and applications of ABINIT-MP software based on the fragment molecular orbital method”, in Modern methods for theoretical physical chemistry of biopolymers, E. B. Starikov, J. P. Lewis, S. Tanaka, Eds., Elsevier (2006) pp. 39-52.
（６）D. G. Fedorov, K. Kitaura, “Extending the Power of Quantum Chemistry to Large Systems with the Fragment Molecular Orbital Method”, *J. Phys. Chem.* **A 111**, 6904-6914 (2007).
（７）北浦和夫, “フラグメント分子軌道法の概要”, CBI 学会計算化学研究会ワークショップ(2008/5/22), http://www.cbi.or.jp/cbi/jigyou/FMO_gaiyo.pdf
（８）中野達也, 望月祐志, 甘利真司, 小林将人, 福澤薫, 田中成典, “フラグメント分子軌道法に基づいた生体巨大分子の電子状態計算の現状と今後の展望”, *J. Comput. Chem. Jpn.* **6**, 173-184 (2007).
（９）福澤薫, 中野達也, 加藤昭史, 望月祐志, 田中成典, “フラグメント分子軌道法による生体高分子の応用計算”, *J. Comput. Chem. Jpn.* **6**, 185-198 (2007).
（１０）佐藤文俊, 中野達也, 望月祐志 編著, “プログラムで実践する生体分子量子化学計算”,森北出版, 2008.
（１１）T. Ozawa, K. Okazaki, “CH/π Hydrogen Bonds Determine the Selectivity of the Src Homology 2 Domain to Tyrosine Phosphotyrosyl Peptides: An Ab Initio Fragment Molecular Orbital Study”, *J. Comput. Chem.* **29**, 2656-2666 (2008).
（１２）T. Ozawa, E. Tsuji, M. Ozawa, C. Handa, H. Mukaiyama, T. Nishimura, S. Kobayashi, K. Okazaki, “The importance of CH/π hydrogen bonds in rational drug design: An ab initio fragment molecular orbital study to leukocyte-specific protein tyrosine (LCK) kinase”, *Bioorg. Med. Chem.* **16**, 10311-10318 (2008).
（１３）Y. Mochizuki, T. Nakano, S. Koikegami, S. Tanimori, Y. Abe, U. Nagashima, K. Kitaura, “A parallelized integral-direct second-order Møller-Plesset perturbation theory method with fragment molecular orbital scheme”, *Theor. Chem. Acc.* **112**, 442-452 (2004).
（１４）Y. Mochizuki, S. Koikegami, T. Nakano, S. Amari, K. Kitaura, “Large scale MP2 calculations with fragment molecular orbital scheme”, *Chem. Phys. Lett.* **396**, 473-479 (2004).
（１５）Y. Mochizuki, K. Yamashita, T. Murase, T. Nakano, K. Fukuzawa, K. Takematsu, H. Watanabe, S. Tanaka, “Large scale FMO-MP2 calculations on a massively parallel-vector computer”, *Chem. Phys. Lett.* **457**, 396-403 (2008).
（１６）H. Momida, T. Hamada, Y. Takagi, T. Yamamoto, T. Uda and T. Ohno: ″Dielectric constants of amorphous hafnium aluminates: First-principles study″, *Phys. Rev. B* **75**, 195105-1-10 (2007).
（１７）T. Yamamoto, H. Momida, T. Hamada, T. Uda and T. Ohno, “First-Principles Study of Dielectric Properties of Cerium Oxide”, *Thin Solid Films* **486,** 136-140 (2005).