Advance/FrontSTRの固有値解析機能のベンチマーク解析

松原　聖*　　大家　史*

Benchmarking the Eigen Solver of Advance/FrontSTR

Kiyoshi Matsubara* and Fumito Ohya*

汎用構造解析ソフトウェア Advance/FrontSTR の固有値解析機能に関して報告する。当社では、アドバンスシミュレーション Vol.3 で紹介したように、大規模で高速な固有値解析のアルゴリズム等を実用化すべく開発を進めている。これらの開発は現在進行中である。本稿では、その途中段階として、「中規模データ」に対する固有値解析のベンチマークを行った内容を報告する。本稿では、中規模とは百万自由度から数百自由度前後の問題、小規模とは数十万自由度の問題、大規模とは一千万自由度を超える問題のことを呼ぶものとする。本ベンチマークでは、中規模問題（10 万自由度から 240 万自由度の大きさのデータ）に対して、その精度および並列化による高速化の効果を測定した。その結果、十分に大きいサイズの問題では、16CPU では 12 倍程度の処理速度（12 分の 1 の処理時間短縮）を得ることができたことを報告する。

Key word: 構造解析、固有値解析、並列化、大規模、高速化、精度

1. はじめに

近年の製造業における品質保証への要求に応えるためより精度の高い振動解析等のニーズが高なっている。または、背景として、計算機の性能向上および価格の低下により、大規模な計算により精度の高い解析が可能となってきている。そのひとつの手段として、当社では、並列化を中心とした、解析の大規模化および高速化を実現するためのソフトウェアの開発に取り組んできた。

本稿では、そのひとつとして、汎用構造解析ソフトウェア Advance/FrontSTR の固有値解析機能に関して報告する。アドバンスシミュレーション Vol.3 で紹介したように、当社では、大規模で高速な固有値解析アルゴリズム等を実用化すべく開発を進めている。それらの開発は進行中ではあるが、今回は、その途中段階として、中規模のベンチマークを行った。

数値計算におけるデータの規模は、分野や取り扱う問題により大きくことなるが、ここでは、小規模とは数十万自由度の問題、中規模とは百万自由度から数百自由度前後の問題、大規模とは一千万自由度を超える問題のことを呼ぶとする。

*アドバンスソフト株式会社　技術第 5 部
5th Technical Division, AdvanceSoft Corporation

2. ベンチマーク問題

2.1. 解析対象

汎用構造解析ソフトウェア Advance/FrontSTR の固有値解析機能のベンチマークを行う。利用した問題は、図 1 に示す容器の固有値問題である。境界条件は設定せず、剛体モードの出る問題とした。ここでは、本問題に対して、次のような大きさのサイズのモデルを作成して、解析を行った。

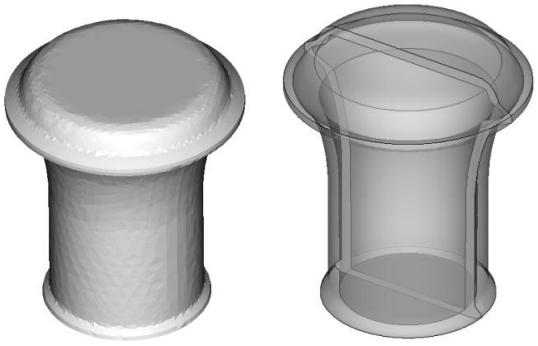

図 1　ベンチマークモデル
（左；ソリッドモデル、右；透視図）

2.2. 計算機環境

ベンチマークを実施した計算機の環境は表 1 の通りである。この計算機環境は、特殊な計算機環境ではなく、比較的安価に企業で導入できる程度の計算機環境である。

表 1　解析に利用した計算機環境

項目	内容
OS	CentOS release 4.4 (Final)
CPU	Dual Core Opteron280(2.4GHz)
構成	16nodes(4Core/node) 2nodes(interactive node) Total 72 Cores
メモリ	16GB/node(4GB/Core)

2.3. 解析条件

メッシュ数、求める固有値数、および計算機の並列度については、次の通りである。

- 上記のモデルをもとに作成した 5 種類のメッシュを利用して、固有値を求める。
- 20 個、100 個、1000 個の固有値を求める
- 1CPU から 16CPU の数のプロセッサを用いた並列計算を行う。

表 2 ベンチマークモデルサイズ（その 1）

名称	Case A	Case B	Case C
要素	4 面体	4 面体	4 面体
次数	2 次	2 次	2 次
要素数	31, 474	66, 159	135, 265
節点数	54, 558	110, 681	214, 136
自由度数	163, 674	332, 043	642, 408

表 3 ベンチマークモデルサイズ（その 2）

名称	Case D	Case E
要素	4 面体	4 面体
次数	2 次	2 次
要素数	251, 792	529, 272
節点数	386, 074	795, 552
自由度数	1, 158, 222	2, 386, 656

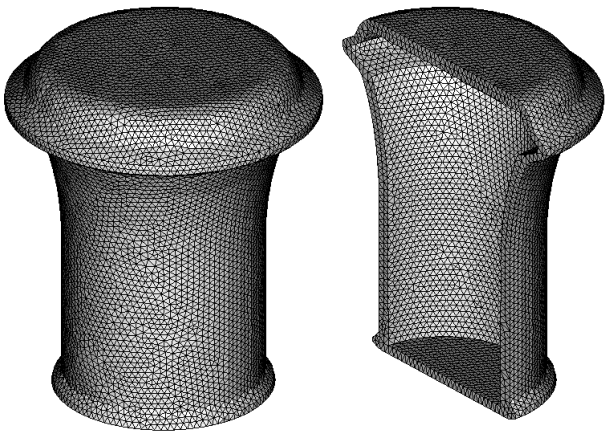

図 2　計算メッシュ(Case D)と断面図

2.4. 評価内容

本ベンチマークでは、次の 2 項目について評価を行った。

- 固有値の精度
- 処理時間

このうち、処理時間では、実行時に利用する使用記憶容量とともに、並列計算の速度向上の割合、処理時間の問題規模への依存性、処理時間の求める固有値数への依存性、処理時間の問題サイズへの依存性について、検討した。

3. 解析結果

3.1. 結果概要

設定した問題に関し、汎用構造解析ソフトウェア Advance/FrontSTR の固有値解析を利用して解析を実行した。実施した計算ケースは、50 ケース程度である。ここで、計算機環境については、独占する環境を利用したものではなく、通常のバッチジョブでひとつのノード内に別のジョブがあることもあった。これは、通常の企業の計算機運用でも発生する状況であるため、特にこの計算のために計算機を独占する環境としなかった。このようなことから、使用するメモリが大きくなる問題については、同じノード内での他のジョブとメモリの競合が起こり、処理時間が遅くなったと推定できる事例もあった。本稿では、それも含めた全体の経過時間（wall clock での Elapsed Time）でソフトウェアの処理時間を

評価した。

また、本評価と合わせ、いくつかのケースに関し、別のソフトウェアとの固有値解析の比較も実施した。その結果、まったく同じメッシュを利用した場合には、同程度の処理時間（dual core と本解析の2CPU の処理時間が同程度）および精度（0.1%以内）であったことをまず報告しておく。

3.2. 処理時間

処理時間については、wall clock で測定した。まず、代表的な処理時間について示す。横軸に自由度数および縦軸に処理時間を示す。ここでは、代表的に、最も処理時間の速い 16CPU を利用したケースについて示した。240 万自由度の問題が 16CPU を利用して 3 時間程度で解けていることが分かる。ここでは、100 固有値を求めるケースを示した。

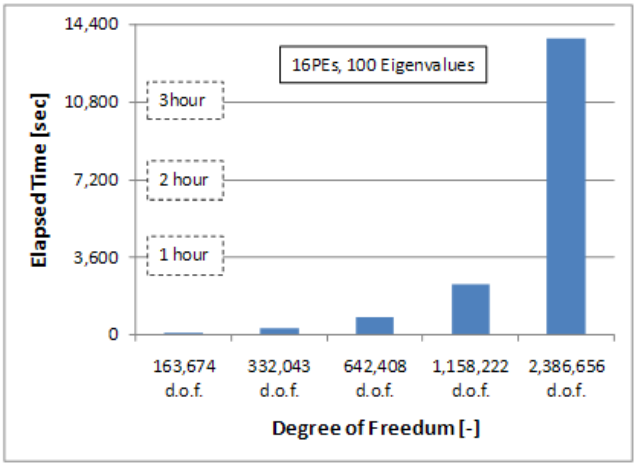

図 3　代表的なケースの処理時間

次に、代表的なケースにおける処理時間がどのように CPU に依存しているかとその速度向上比を示した。このケースでは、16CPU で 12 倍の処理速度となっているため、有効に並列計算機の資源を利用しているといえる。

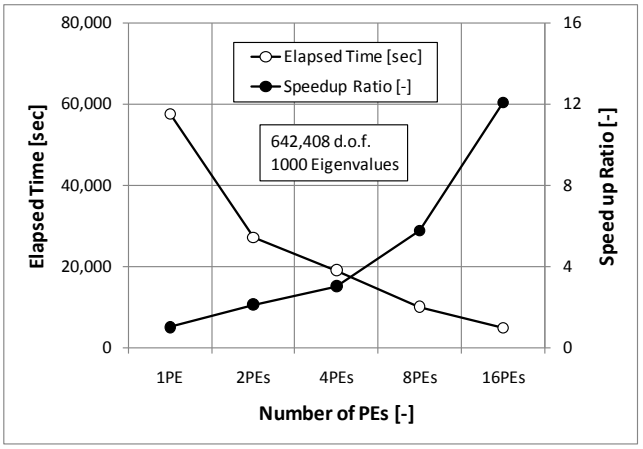

図 4　並列処理と処理時間

以下では、各ケースにおける使用したプロセッサ毎の処理時間、求めた固有値毎の処理時間について実測値をまとめた。まず、ケース A について示す。ケース A については自由度数が少ないため 1000 固有値を求めるケースは実施していない。

3.3. 使用メモリのサイズ

まず、解析に必要な使用記憶容量について述べる。固有値解析では、通常の応力解析等よりもかなり大きなメモリが必要である。それぞれのケースについて、本ソフトウェアでは、次のような量のメモリが必要である。

表 4　解析に必要な記憶容量（単位:GB）

Case A	Case B	Case C	Case D	Case E
2.2	2.9	15.9	32.0	62.7

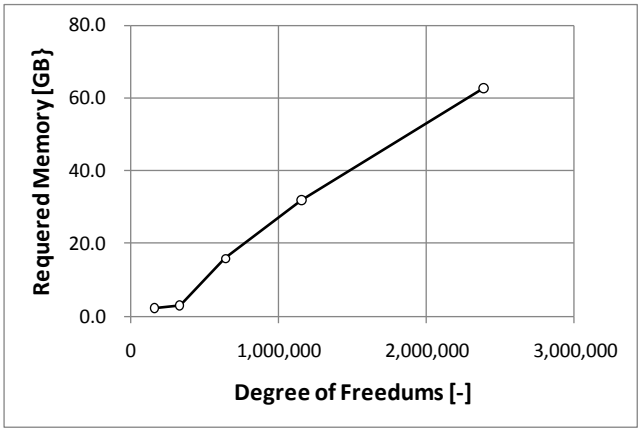

図 5　解析に必要な記憶容量

3.4. 固有値の精度

解析では、それぞれ、20 個の固有値、100 個の固有値、1000 個の固有値を求めた。まず、固有値を求める個数により、その結果は変化しなかったことを確認した。

図 7 では、図 6 の低次モードの部分のみについて拡大した図を示す。図 6 により、得られた固有値については、ほとんど一致していることが分かる。また、参考までに、市販のソフトウェアとの比較を行っているが、同じメッシュの場合には、0.1%以内で一致していることを確認している。

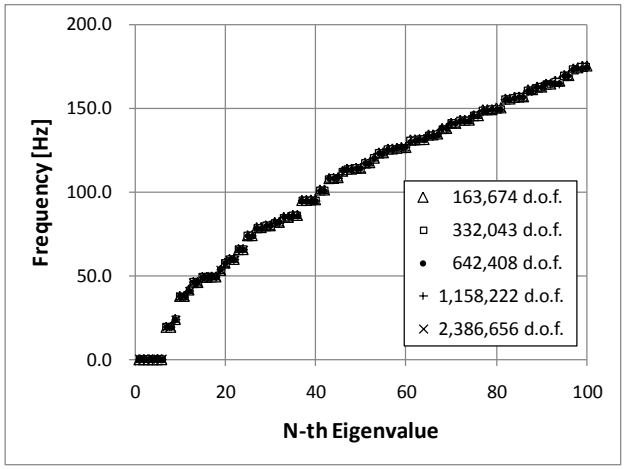

図 6　計算で得られた固有値（全モード）

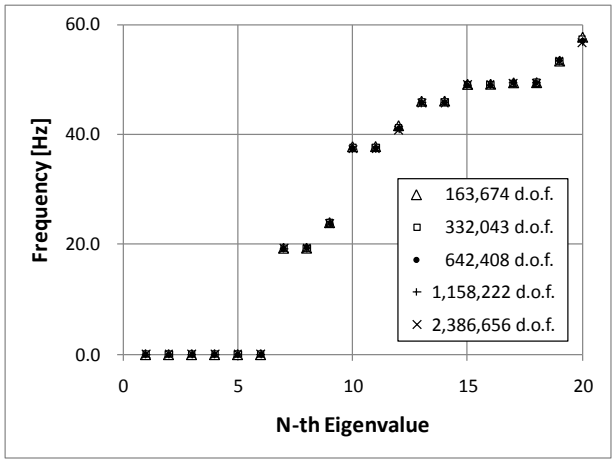

図 7　計算で得られた固有値（低次モード）

これらの結果に対して、最もメッシュの細かいケース E を正として、固有値の小さい順にその値の相対誤差を比較した。このモデルでは、100 万自由度を超えるあたりで、その誤差は平均 0.1%程度に収まっている。もちろん、この基準は、モデルの複雑さに依存するが、メッシュ数に対する感度の目安は得られたと考えている。

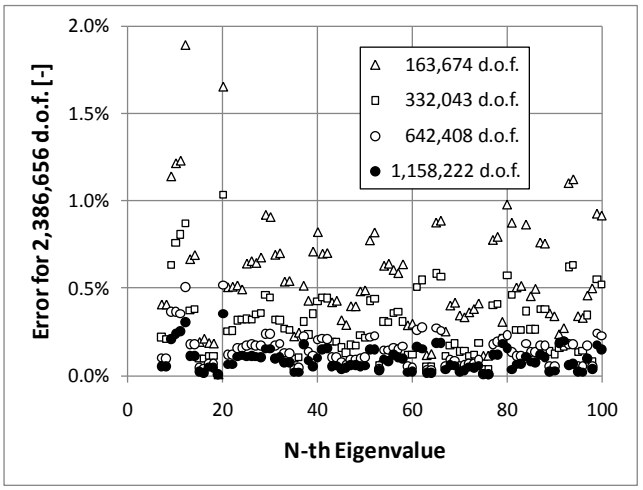

図 8　計算で得られた固有値の誤差

また、次のページ以降の表 1～表 3 に、モード変形図を示す。ここでは、剛体モードの 1 から 6 番目までの固有値のモードは除外して図示した。例えば、図 7 から、7 番目および 8 番目は重複固有値に近く、そのモードは類似した形状となっている。また、10 番目および 11 番目のモードについても同様である。それぞれ、X 軸の正の方向（正面図）から、Y 軸の正の方向（上面図）および Z 軸の正の方向（側面図）からの変形モードを示す。

この変形モードについては、節点座標に質量マトリクスで正規化された固有ベクトル（を実数したもの）を加えたものである。ただし、この実数については、変形が図に見えるように乗じているが、すべてのモードについて一定の実数を乗じていることを補足しておく。

表 5 モード変形図（その 1）

	上面図	正面図	側面図
第7モード			
第8モード			
第9モード			
第10モード			
第11モード			

表 6 モード変形図（その 2）

	上面図	正面図	側面図
第12モード			
第13モード			
第14モード			
第15モード			
第16モード			

表 7 モード変形図（その 3）

	上面図	正面図	側面図
第17モード			
第18モード			
第19モード			
第20モード			
第21モード			

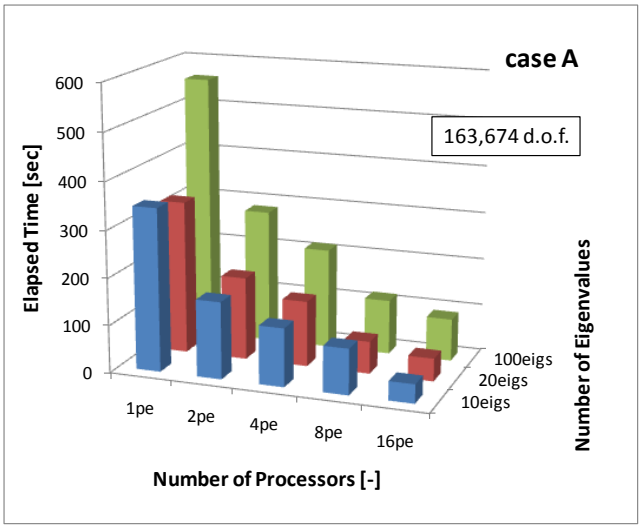

図 9　処理時間（ケース A）

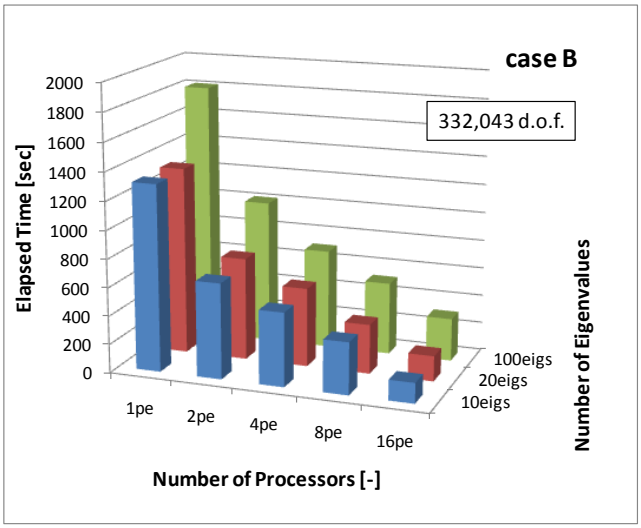

図 11　処理時間（ケース B）

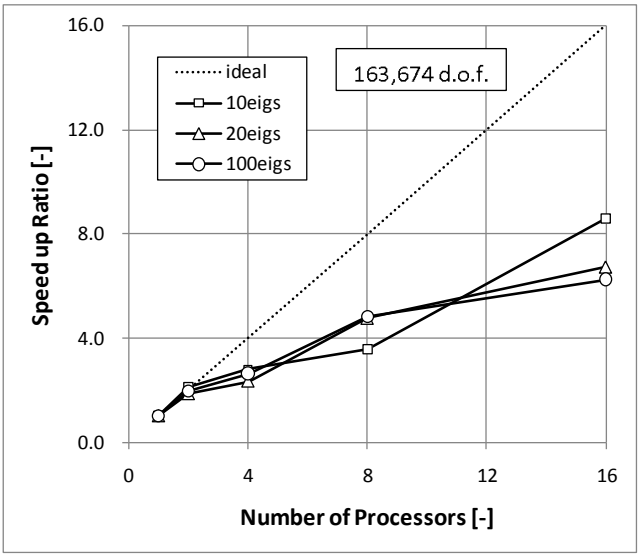

図 10　並列化による速度向上（ケース A）

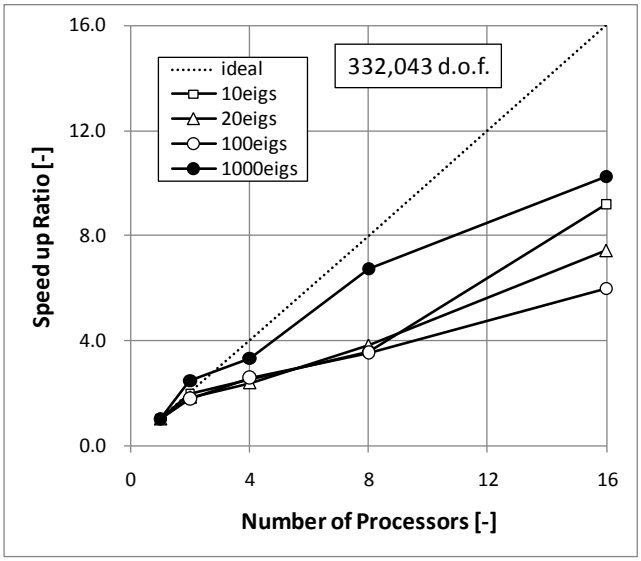

図 12　並列化による速度向上（ケース B）

表 8 処理時間一覧（ケース A）

PE 数	10eigs	20eigs	100eigs
1pe	345.2	324.8	560.1
2pe	163.1	175.2	283.1
4pe	123.5	138.8	212.3
8pe	96.9	68.2	115.6
16pe	40.2	48.2	89.5

表 9 処理時間一覧（ケース B）

PE 数	10eigs	20eigs	100eigs	1000eigs
1pe	1314.6	1321.3	1808.9	24879.9
2pe	674.0	721.6	1011.4	10118.0
4pe	520.2	555.1	699.6	7479.5
8pe	366.8	347.4	510.5	3696.2
16pe	143.2	178.1	301.6	2427.2

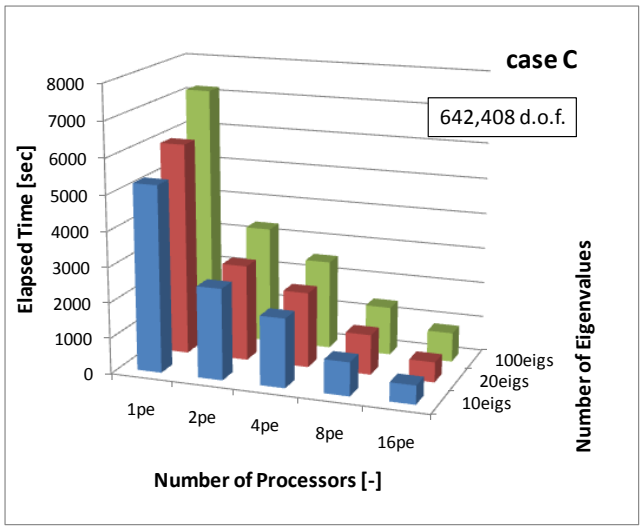

図 13　処理時間（ケース C）

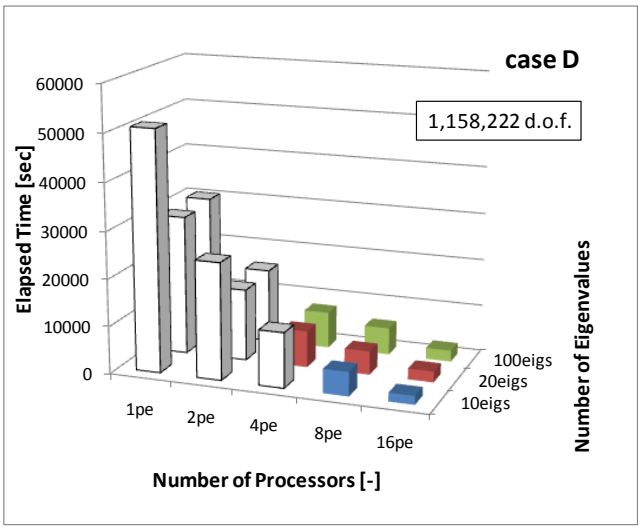

図 15　処理時間（ケース D）

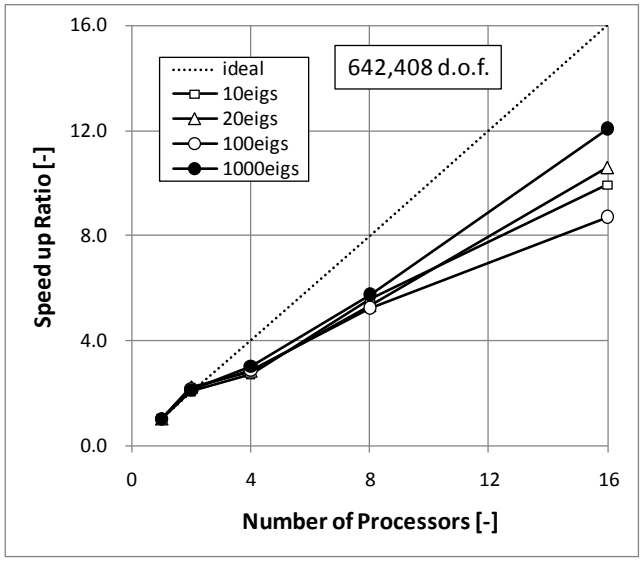

図 14　並列化による速度向上（ケース C）

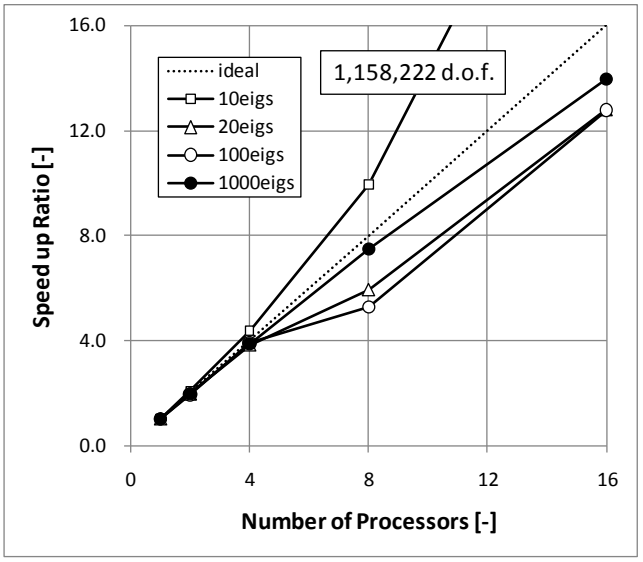

図 16　並列化による速度向上（ケース D）

表 10 処理時間一覧（ケース C）

PE 数	10eigs	20eigs	100eigs	1000eig
1pe	5258.3	5996.4	7182.7	57608.5
2pe	2574.7	2709.2	3314.7	27166.4
4pe	1945.7	2124.1	2515.9	19060.5
8pe	944.8	1127.2	1368.6	10010.8
16pe	529.8	566.8	824.4	4770.7

表 11 処理時間一覧（ケース D）

PE 数	10eigs	20eigs	100eigs	1000eigs
1pe	*50898.0*	*29555.8*	*30346.3*	*193038.0*
2pe	*24736.7*	*15190.5*	*15680.9*	*97448.8*
4pe	*11656.0*	7734.7	7706.3	*49654.3*
8pe	5115.7	4984.9	5739.7	25757.0
16pe	1845.5	2309.0	2374.9	13808.4

ここで、表中の斜体の部分および棒グラフの白抜き部分については、メモリ使用量の関係で、実行ができない部分である（1node あたりの使用記憶容量が十分にある計算機環境では実行可能である）。したがって、その実行できなかった部分については、アムダールの法則から、仮に 1CPU の場合の処理時間を求め、各並列数の処理時間を求めた。具体的には、次のような手順にしたがった。アムダールの式

$$\frac{T_n}{T_1} = 1 - \left(1 - \frac{\alpha}{n}\right) \qquad (1)$$

におけるT_1およびαが未知数であるため、計測済みのT_nとnを代入し、平均 2 乗誤差が最小になるように、T_1およびαを決定した。また、ここで、斜体以外の部分については、すべて実測である。

最後にケース E について示す。ここでは、計算機資源（使用記憶容量）の制約条件から、16PE でのみ実行した。100 固有値を 4 時間以内で求めることができていることが分かる。

表 12 処理時間一覧（ケース E）

case E	10eigs	20eigs	100eigs
16pe	11658.2	12194.8	13732.3

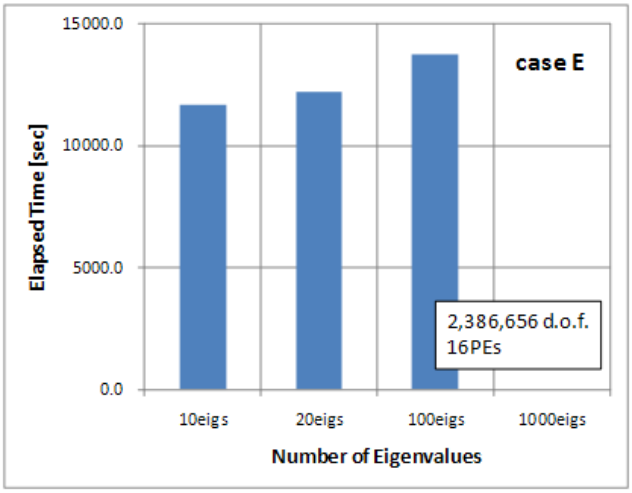

図 17 処理時間（ケース C）

3.5. 処理時間の問題サイズへの依存性

今後大規模化する場合に、もっとも興味のある問題が大規模化した場合の処理時間の予測である。まず、16CPU を利用した代表的なケースに関し、自由度に対する処理時間を示す。通常の場合と同じく、処理時間は、自由度に対して線型以上の割合で増加している。

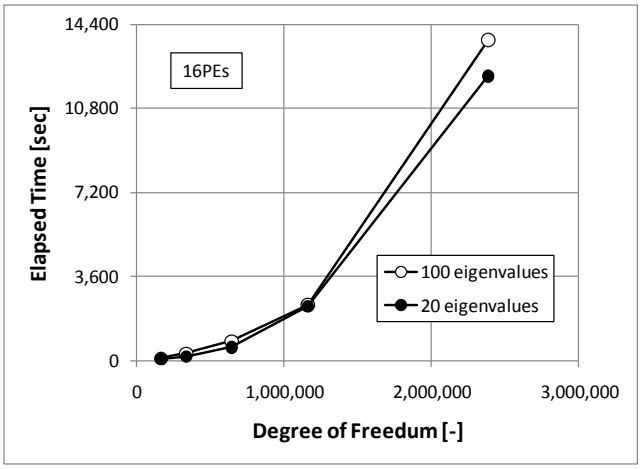

図 18 処理時間の問題サイズ依存性

ここでは、本ベンチマークで掲載したすべてのケースの処理時間について、同じグラフに対数軸で示した。この傾向から、全体の近似曲線を考えることで、処理時間は問題サイズの 1.6～2.0 乗に比例して増加することが分かった。

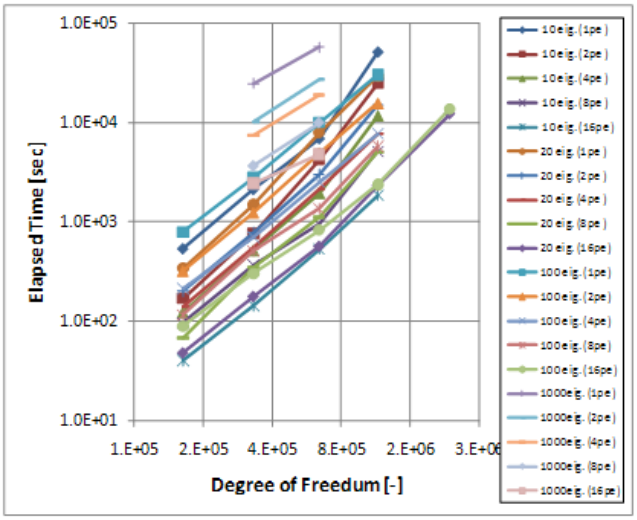

図 19 処理時間の問題サイズ依存性（全ケース）

3.6. 処理時間の求める固有値数への依存性

設計問題では、数百を超す固有値および固有ベクトルを求める必要がある。その場合には、通常、固有値数に比例して処理時間が増加することを覚悟しなくてはならない。逆に、比例程度の時間であれば、ユーザは許容できると考えられる。

図 20はケースCに関する処理時間の固有値の数への依存性である。

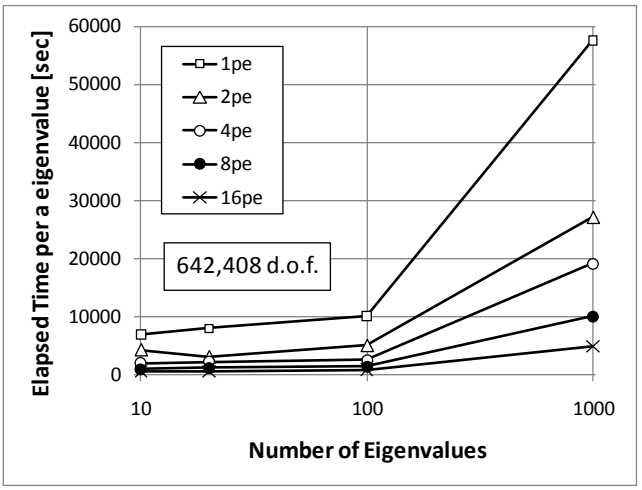

図 20　処理時間の固有値数依存性

次にこの処理時間を固有値当たりの処理時間に換算したグラフが図 21 である。この結果から、処理時間は、固有値数に比例するよりは少ない時間で処理できていることが分かる。

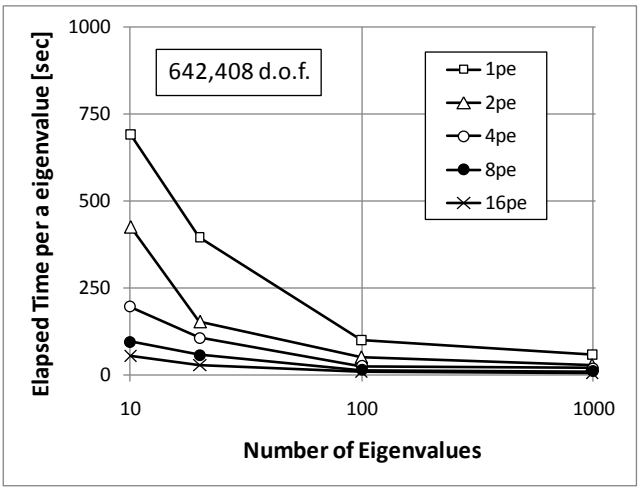

図 21　固有値１個あたりの処理時間

また、多数の CPU が利用可能な場合には、CPU 当たりの処理時間も重要になる。ここでは、さらにさきほどのデータに追加して、CPU ひとつあたり、固有値ひとつあたりの処理時間をまとめた。このグラフから、多くの CPU で必要なだけの固有値を求めるという解析の方法で問題のないことが分かる。

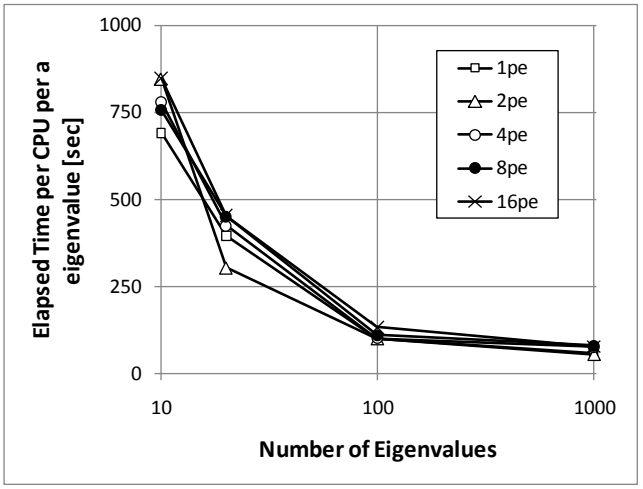

図 22　固有値１個 1CPU 当りの処理時間

4. まとめ

Advance/FrontSTR を利用して、各種サイズのベンチマークを行った。ここでは、並列による速度は十分に得られている結果が得られた。また、それらは精度の面でも、市販のプログラムと比較しても妥当なものであり、また、メッシュ等の解析パラメータを変更することでも、合理的な結果を得ることができた。この結果については、今後の固有値解析の処理時間や並列化効率および使用記憶容量の目安になるものと考えられる。

このベンチマークから、処理速度は十分に高速であり、並列計算機の性能を十分に生かす程度の並列化効率が得られていると考えられる。一方で、近年の計算機の性能向上により、本解析で利用した計算機の使用記憶容量は非常に大きいものがある。計算機が安くなったとはいえ、まだ十分に高価である。したがって、その使用記憶容量を減らす工夫が今後必要と考えられる。多階層モード合成を利用して使用記憶容量を減少させる工夫も考えることができる。今後、現在の手法およびまだ十分には取り入れることができていない多階層モード合成の手法を今後取り入れていく予定である。