サイエンティフィック・システム研究会
科学技術計算分科会第1回会合資料　　2000年9月8日

VPP5000/56 の性能測定と運用経験について

名古屋大学大型計算機センター
永井　亨
nagai@cc.nagoya-u.ac.jp

1 はじめに

名古屋大学大型計算機センターは大学・高等専門学校等の教員、大学院生、その他の研究者が学術研究等のために利用する全国共同利用施設で、センターの利用者数は1400人を越える。センターでは1999年12月にスーパーコンピュータシステムをFujitsu VPP500/42(以下、VPP500)からFujitsu VPP5000/56(以下、VPP5000)に更新した。本稿ではいくつかのベンチマークジョブを用いてVPP5000 の性能を測定した結果と、VPP5000の運用経験、特に、並列ジョブの多重実行を運用にのせるまでの経過を報告する。

2 システムの概要

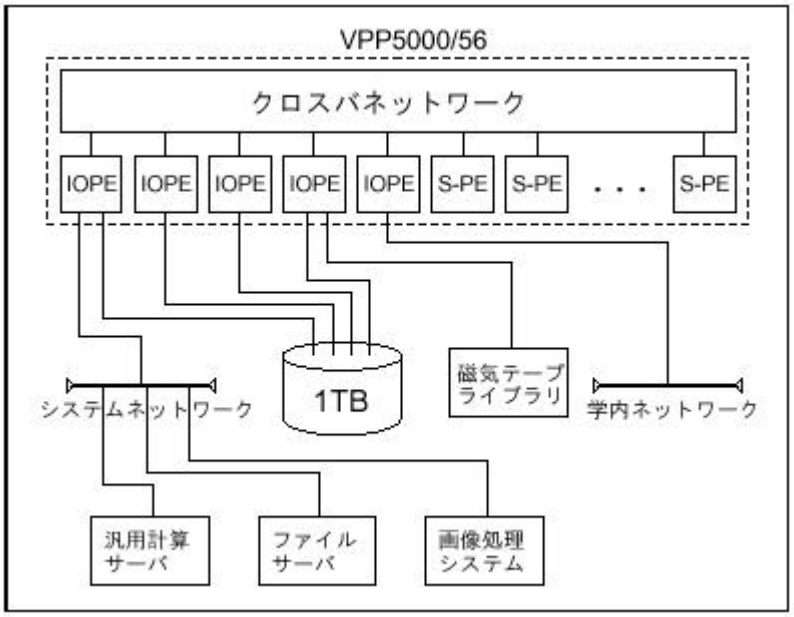

図1: VPP5000 のシステム構成

VPP5000 のシステム構成を図1 に示す。VPP5000 は56台のPEとこれらを結合するクロスバネットワークからなる。各PEの理論最大性能は9.6G Flops、PE 当たり8GBの主記憶を備え、一部のPEでは16GBである。入出力機能をもつIOPE は5台で、4台は磁気ディスク、システムネットワークやバックアップ用磁気テープライブラリに接続され、残りの1台は会話型処理のために学内ネットワークに接続されている。システム全体を統括するIOPEの中の1台はprimary PEと呼ばれる。入出力機能をもたない演算処理専用のPEはsecondary PE（図1ではS-PE）と呼ばれる。磁気ディスクの容量は1TB、バックアップ用磁気テープライブラリの容量は最大5.8TB

である。図1中の汎用計算サーバはFujitsu GP7000F model 900 である。ファイルサーバ上のファイル（容量1.8TB)は汎用計算サーバと共有している。

VPP5000で利用できるソフトウェアを表1に示す。デバッグやチューニングのための支援ツール（アナライザ、VPP Workbench、TotalView など）およびアプリケーションプログラムのプリ・ポストプロセッサは汎用計算サーバや画像処理システム上に用意されている。

表1: 利用できるソフトウェア

プログラミング言語	Fortran C C++
並列処理言語	VPP Fortran HPF (High Performance Fortran) DPCE (Data Parallel C Extension)
数値計算ライブラリ	SL II C-SSL II NUMPAC BLAS, LAPACK, ScaLAPACK
メッセージパッシング ライブラリ	MPI PVM
アプリケーション プログラム	α-FLOW （汎用3 次元流体解析システム） STAR-CD （汎用3 次元流体解析システム） POPLAS/FEM5 （有限要素法による構造解析プログラム） LS-DYNA3D （非線形動的構造解析プログラム） MASPHYC, MASPHYC/SP （材料設計プログラム） Gaussian98 （分子軌道計算プログラム） AMBER （分子構造計算プログラム） fastDNAml （最尤法による進化系統樹推定プログラム） VisLink （可視化支援システム）

表2にVPP5000 のジョブ種別を示す。センターでは使用したCPU時間に対して従量制課金を行っているが、並列ジョブの課金については、システム全体に与える負荷の大きさを考慮して2～8台のPE を使用するジョブには1 台分、9～16台のPEを使用するジョブには1台分の1.5倍、また17台以上では1台分の3倍が課金される。リージョンサイズが15.5GB程度まで利用できる1PE用ジョブクラスも準備が整い次第公開する予定である。

表2: VPP5000 のジョブ種別

	キュー名	使用可能PE数	CPU使用時間	メモリサイズ	
				標準値	制限値
バッチジョブ	C	1	60 分	500MB	2GB
	x	1	1200 分	2GB	7.5GB
	z	2〜16	600 分	1PEあたり 2GB	7.5GB
	ze	17〜32	600 分	1PEあたり 2GB	7.5GB
TSS	−	1	60 分	500MB	2GB
	−	1〜 8	300 分	1PEあたり 2GB	7.5GB

表3: VPP5000 と VPP500の性能測定のためのプログラムの概要

	概要	実行に必要な領域(MB)	使用PE数
ジョブ1	地球磁気圏の3 次元MHD シミュレーション	500	1
ジョブ2	ビル周りの流れのシミュレーション(α-FLOW)	200	1
ジョブ3	8CB 液晶のシミュレーション(MASPHYC)	200	1
ジョブ4	非等方乱流の直接数値シミュレーション	4300	16
ジョブ5	木星磁気圏の3 次元MHD シミュレーション	2800	16
ジョブ6	ビル周りの流れのシミュレーション(α-FLOW)	500	4

3 性能測定

3.1 VPP500との比較

VPP500との比較のために用いたプログラムの概要を表3に示す。これらはVPP5000の導入にあたって使用した性能評価試験プログラムで、センター利用者から提供されたものである。ジョブ2 とジョブ6 ではα-FLOWを使用する。またジョブ3ではMASPHYC を使用する。それ以外のジョブはFortranプログラムである。ジョブ4〜6は複数PEを使用する。ジョブ4とジョブ5は VPP Fortranで記述されている。

ジョブ1〜6を VPP500 および VPP5000で実行させたときの処理時間の測定結果を表4に示す。ジョブ1〜3ではCPU時間が、ジョブ4〜6では経過時間が示されている。VPP500の1PEの理論最大性能は1.6G Flopsであるからその6倍である。多くのジョブでVPP500 とVPP5000との比率は6前後の値が得られた。ジョブ3 (MASPHYC)では比率が10倍を越えるが、

表4: VPP500 とVPP5000 での測定結果

	VPP500 (秒)	VPP5000 (秒)	比率	備考
ジョブ1	5,768	978	5.9	
ジョブ2	3,592	693	5.2	
ジョブ3	2,537	237	10.7	
ジョブ4	7,456	2,226 1,125	3.3 6.6	オリジナルプログラム プログラム書き換え後
ジョブ5	6,039	971	6.2	
ジョブ6	1,592	312	5.1	

注) ジョブ1 ～3 ではCPU 時間を、ジョブ4 ～6 では経過時間を示してある。

これは計算の大部分がスカラ実行されており、スカラ性能の差が現れたものと考えられる。ジョブ4のVPP5000での実行についてはオリジナルのプログラムの場合とプログラムを書き換えた場合とが示されている。書き換え後では配列の分割軸が1 次元目から3次元目に変更され、ベクトル性能が向上している。

3.2 台数効果

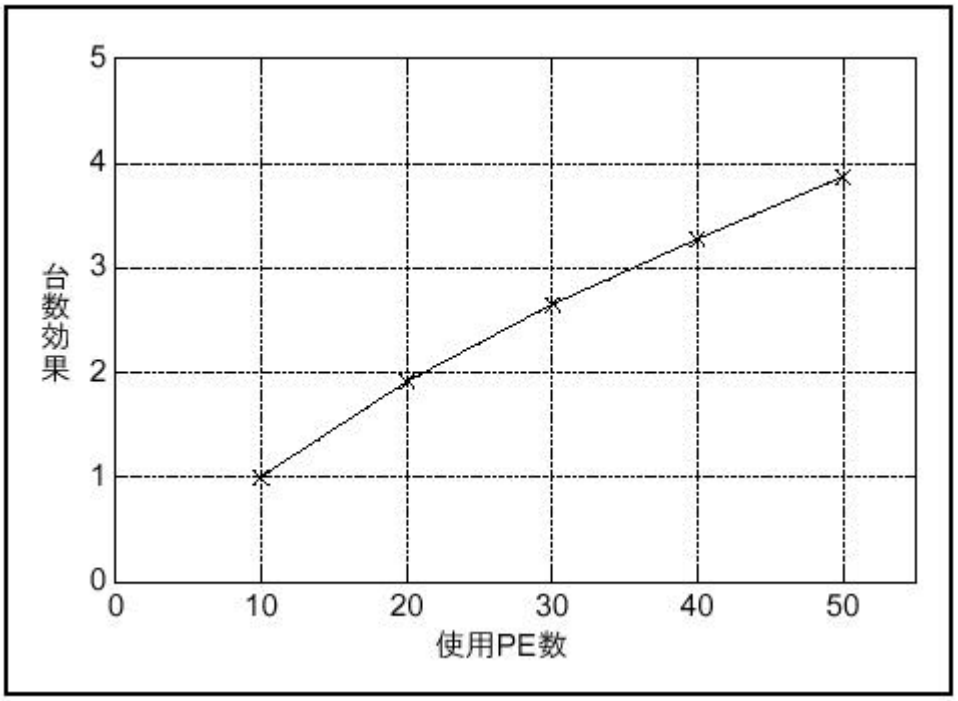

図2: 台数効果の測定例

ジョブ5のプログラムサイズを55GBに拡大したものを、10～50台までのPEを使用して処理時間を測定し、その台数効果を調べたものを図2 に示す。55GBのプログラムを1PEでは実行できないため、図2 では10PEで実行したときの経過時間を1 として示してあるので注意されたい。

表5: HPF とVPP Fortran の比較

使用PE数	HPF(秒)	VPP Fortran(秒)	比率
2	1.390	1.381	1.01
4	0.712	0.715	1.00
8	0.393	0.398	0.99
16	0.202	0.212	0.95
32	0.120	0.131	0.92

注) 時間は経過時間を示してある。

表6: Gaussian98 の測定例

使用PE数	VPP700E版(秒)	VPP5000版(秒)	比率
1	14,944	14,020	1.1
4	5,901	3,819	1.5
台数効果	2.5	3.7	

注) 1PEジョブではCPU 時間を、4PEジョブでは経過時間を示してある。

3.3 HPF とVPP Fortran の比較

現在までのところセンターでは、Fortranプログラムの並列化にはVPP Fortranを使用する利用者がほとんどであるが、HPF(High Performance Fortran)に書き換えた例を1つ紹介しておく。

ジョブ1は元々VPP Fortranで記述されていて並列実行も可能である。これをHPF/JAに書き換えて実行時間を測定したものをVPP Fortran版と比較して表5に示す。一般にHPFはVPP Fortranに比べて性能が劣るといわれるが、表5をみるかぎり、HPF はVPP Fortran と同等あるいはそれ以上の性能が得られている。

3.4 Gaussian98/VPP700E 版とVPP5000 版の比較

分子軌道計算プログラムGaussian98にはシングル版とメッセージパッシングライブラリLindaを使用した並列版があるが、VPP5000の導入当初はいずれもVPP700E互換モジュールがインストールされていた。最近になってVPP5000用にチューニングされたモジュールが利用できるようになったのでVPP700E版とVPP5000版の性能を比較した例を紹介する。

2つのケースについてシングル版で実行したところ、いずれのケースでもVPP700E版に比べてVPP5000版の方がCPU 時間で1.5倍速かった。並列版については、計算する問題によって、また問題のサイズによっても並列効果が現れる場合とそうでない場合があった。Gaussian98で標準に提供されるテストデータtest397.com(SCF)ではかなりよい並列効果が得られたので、その測定結果を表6に示す。

4 並列ジョブの多重実行

VPP500では並列ジョブはsimplexモード(1つのPE上では1つのジョブしか実行できない)でしか実行できなかった。このため空きPE があるにもかかわらず、PE数が不足するために並列ジョブの実行が待たされるという事態が何度も発生した。VPP5000の導入にあたって、PEの効率的利用の観点から並列ジョブの多重実行は是非運用にのせる必要があると考えていた。その際の条件には以下のものがある。

- CPU課金にばらつきがないこと
- PE内で同一ジョブの折り返しができること
- 並列ジョブの多重度は2多重までとする

並列ジョブの多重実行をおこなうにはモードをsharedにする必要があり、sharedモードには次の2種類のスケジューリングクラスがある。

- 同期並列クラス
  並列ジョブに対し各PEが協調して同一時刻にCPU資源を割り当てる。このためジョブの同期ずれが少なくなる。PE内での同一ジョブの折り返しはできない。
- バッチクラス
  実行中のジョブ数に応じてCPU配分制御によりCPU資源を割り当てる。PE内での同じジョブの折り返しが可能(ただし、バリアクラスBのみ) 。

以下で並列ジョブの多重実行の運用経験を説明する。また、この間のVPP5000のCPU稼働率を図3に示す。

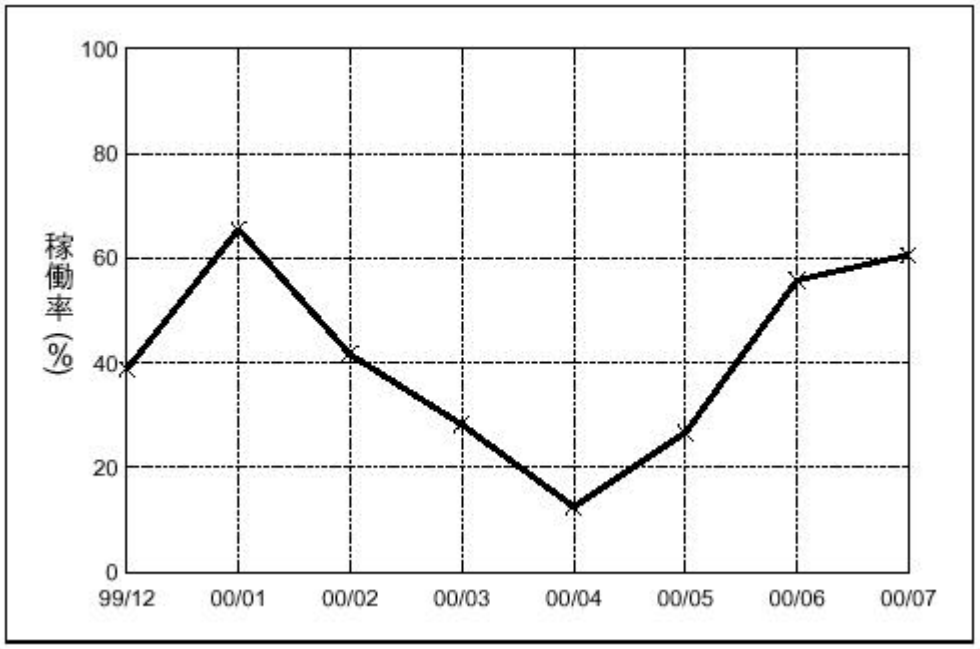

図3: VPP5000のCPU稼働率の推移

1．1999年12月～2000年3月

並列ジョブの多重実行は経験を積んでからと考え、VPP5000導入直後はsimplexモードで運用した。この時期の運用環境は以下の通りである。

モード：simplex バリア同期種別：FLIB SYSPOLL=0 バリアクラス：A

ただし、FLIB SYSPOLLは0のとき同期待ちの際にそのプロセスがCPU資源をつかんだままの状態になり、1のときCPU資源を他のプロセスにわたせる状態になる。また、バリアクラスにはA(高速バリア機構)と B(低速バリア機構)がある。
2月に富士通より並列ジョブの多重実行について説明を受けた。また、3月上旬に数個のジョブを使った並列ジョブの多重実行のテストランを約20パターン行った。バリア同期が頻発する並列ジョブのCPU時間が長くなる現象がみられたが、全体として大きい問題はないと判断し、4月から並列ジョブの多重実行を運用にのせることにした。

2．2000年4月1日～5月9日

この時期の運用環境は以下の通りである。

モード：shared スケジューリングクラス：バッチ バリア同期種別：FLIB SYSPOLL=1 バリアクラス：B

この期間にCPU時間が進まなくなる並列ジョブが4件、実行結果が以前(simplexモードで運用中)のものと異なる並列ジョブが1件、大量のエラーメッセージが出力されるため primary PEに負荷がかかり、システム全体がslow down状態となる並列ジョブが1件発生した。この時点では利用者のプログラムミスなのかシステムの障害なのかは判明していなかったが、図3のCPU稼働率にも現れているようにジョブが処理できない状態に陥った。このため、運用環境を4月以前の状態に戻すことにした。

3．2000年5月10日～6月4日

この時期の運用環境は以下の通りである。(ただし、設定ミスのため5月10日～5月15日まではバリアクラスはBになっていた。)

モード：simplex バリア同期種別：FLIB SYSPOLL=0 バリアクラス：A

CPU 時間が進まなくなる件と実行結果が異なる件についてはシステム障害であることが判明したため修正をあてた後、多重実行を運用することになった。

4．2000年6月5日

運用環境は以下の通りである。

モード：shared スケジューリングクラス：バッチ バリア同期種別：FLIB SYSPOLL=1 バリアクラス：A

この日、CPU時間が進んでいかない並列ジョブが複数でてきた。この現象はバリア同期が頻発するジョブとCPU boundジョブが混在する場合に発生することがわかった。これを図4を使って説明すると、ジョブX(同期頻発ジョブ) が同期待ちになると制御はジョブY(CPUbound ジョブ)にわたりタイムスライス分(100ミリ秒)走る。次にジョブXに制御が渡ると50μ秒走って同期待ちになるので再びジョブYに制御がわたる。これを繰り返すためジョブXは約100ミリ秒のうち50μ秒しか計算が進まない。したがってジョブX の実行が終了するには通常の1000倍以上の時間がかかってしまうことになる。このため、バッチクラスの運用をあきらめ、同期並列クラスでの運用を決定した。

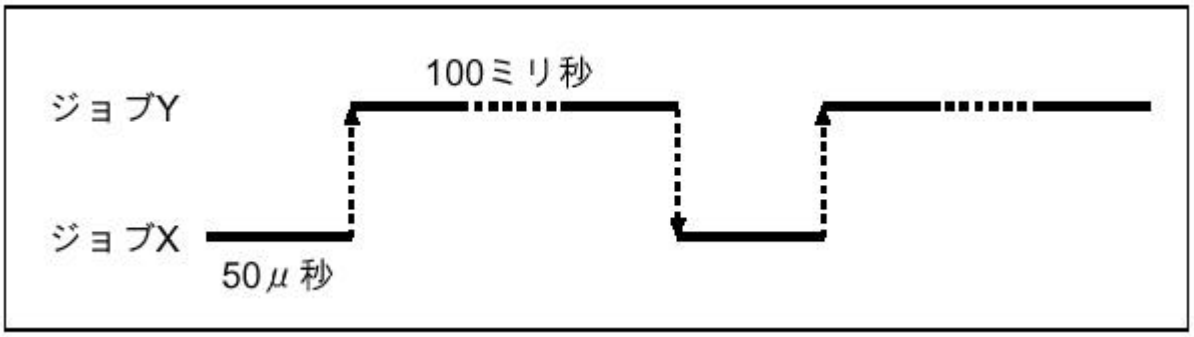

図4: 同期頻発ジョブ(X)とCPU boundジョブ(Y) 混在時の動作

5．2000年6月6日～6月8日

運用環境は以下の通りである。

モード：shared スケジューリングクラス：同期並列 バリア同期種別：FLIB SYSPOLL=1 バリアクラス：A

MPIを使用する並列ジョブでPEごとにCPU時間のばらつきがでていることが観察されたため、バリア同期種別VPP_POLLを0から1に変更することにした。(VPP_POLL は0のとき同期待ちの際にそのプロセスがCPU資源をつかんだままの状態になり、1のときCPU資源を他のプロセスにわたせる状態になる。この時点まではVPP_POLLは 0になっていた。)

6．2000年6月9日～6月13日

運用環境は以下の通りである。

モード：shared スケジューリングクラス：同期並列 バリア同期種別：FLIB SYSPOLL=1 VPP_POLL=1 バリアクラス：A

MPIを使用する並列ジョブでデッドロックが発生したためVPP_POLLを 0に戻すことにした。

7．2000年6月14日～8月16日

運用環境は以下の通りである。

モード：shared スケジューリングクラス：同期並列 バリア同期種別：FLIB SYSPOLL=1 VPP_POLL=0 バリアクラス：A

MPIジョブでのデッドロックの障害は解決したが、VPP_POLLを1としてMPIジョブを試験的に流したところ、CPU時間が進んでいかない障害が再び発生した。

8．2000年8月17日～

運用環境は以下の通りである。

モード：shared スケジューリングクラス：バッチ バリア同期種別：FLIB SYSPOLL=1 VPP_POLL=1 バリアクラス：A

この環境では以下のようなシステムの修正および改良が行われている。

- MPI ジョブでCPU時間が進んでいかない障害を修正した。
- バリア同期頻発ジョブとCPU boundジョブが混在しても問題が生じないように(問題がより小さくなるように)バッチクラスを改良した。
- バリアクラスAを使用してPE内のジョブの折り返しが可能なようにバッチクラスを改良した。

5 おわりに

VPP5000の性能測定結果と、並列ジョブの多重実行の経過について報告した。並列ジョブの多重実行の運用に関しては、ここまで七転八倒した感がある。この問題が終息したどうかは現段階ではまだわからない。

謝辞

表5のデータを提供していただいた名古屋大学太陽地球環境研究所の荻野竜樹教授に謝意を表します。

2000 年 9 月 8 日

補足

富士通(株)ソフトウェア事業本部第二ソフトウェア事業部第五開発部
長屋忠男

名古屋大学大型計算機センター殿のVPP5000の運用方針は、並列ジョブの多重実行、PE内折り返しを利用することにより、

- 多数のPEが遊ばないようにジョブスケジュールする(稼働率を上げる)こと
- 待ち時間をできる限り少なくし、ターンアランド時間を向上させること

であると認識しています。

2000年9月現在、センターで運用変更の効果を評価いただいている状況にありますが、繁忙期でない現状では運用変更の効果が顕在化しにくいため継続評価が必要と考えています。

弊社では、運用変更にあたってセンターより依頼を受け、ジョブスケジューリングについて推奨パターン等の説明を行い、センターでの評価結果の検証に協力しながらシステムの改善を実施してきました。なお、センターの運用変更の効果を評価いただいた際にご指摘された障害や機能不足については、修正を適時実施させていただきながら、センター運用方針の実現へ向けて改善を進めていきます。