ハイパフォーマンス
コンピューティング 54－6
（１９９４．１２．１４）

最近の並列化プログラミング技術について

ースケーラブル並列計算機CONVEX-SPPにおける経験を含めてー

星野　誠三

ＮＫＫ情報システム部

最近の並列計算機システムは、共有メモリ機能とワークステーション並みの使い易さを備えて既に実用段階に入ってきており、先端の研究者のみでなく企業のエンジニアに急速に普及し始めている。本報告では、この様な計算機を利用するため、アプリケーション開発の立場から見た並列化プログラミング技術について、スケーラブル並列計算機システム：CONVEXーSPPを例に、並列化効率の考え方、共有メモリの実現方法、プログラミングモデル、自動並列化コンパイラ、共有メモリプログラム例：FEM解析、メッセージパッシングプログラム例：領域分割法を報告する。

Parallel Programming Technique Using CONVEX SPP：Scalable Parallel Processor

Seizo Hoshino

NKK

Information Processing Sysytem Dept.

1-1-2 Marunouchi Chiyodaku Tokyo 100 JAPAN

Recently, there are several parallel processors that have sheared memory architecture and equipped ease of use as like as workstations. Those machines generate the wave to propagate parallel processor not only among researchers but also industrial engineers for production use.

This paper describes recent parallel programming technique for such systems like CONVEX SPP Scaleable Parallel Processor as example.

In the paper, parallel efficiency , implementation of sheared memory, programming model , automatic parallel compiler , sheared memory example:
FEM analysis and message passing example: DDM were discussed.

1. はじめに

最近の並列計算機システムは、分散共有メモリ機能とワークステーション並みの使い易さを備えて既に実用段階に入ってきており、先端の研究者のみでなく企業のエンジニアに急速に普及し始めている。また、プロダクションに使用される商用の解析コードベンダも、従来のスーパーコンピュータ、ホストコンピュータ、ワークステーションのみでなく、並列計算機が実用段階に入ったとして、並列版のコード開発に取り組んでいる。本報告では、この様な計算機を利用するため、アプリケーション開発の立場から見た並列化プログラミング技術について、スケーラブル並列計算機システム：CONVEX－SPPを例に、システム概要、共有メモリの実現方法、並列化効率の考え方、プログラミングモデル、自動並列化コンパイラ、共有メモリプログラム例：FEM解析、メッセージパッシングプログラム例：領域分割法を報告する。

2. 並列化の効率

2．1　アムダールの法則

複数の CPU を用いて並列計算する場合のスピードアップは、ユーザからみた計算　のエラプス時間の比（1 CPU：m CPU）で評価する。

並列化スピードアップ見積もり ：S

$$S = \frac{1}{(1-r + r/(m*e))}$$

r：並列化対象部の比率
　= モデル規模依存
m：cpuの数
e：局所並列化効率
　並列化対象部のみの
　並列化効率、システム
　、問題の性質依存

ここで cpuの数 m が多くなると、スピードアップ S は、1／（1 － r）に漸近する。逆に、cpuの数を多くしても、高々 1／（1 － r）にしかならない。これが「アムダールの法則」である。（1 － r）は並列化できない部分の比率になり、結果としてスピードアップの最大値を支配する。

局所並列化効率は、並列化によるオーバーヘッド分で、対象の問題にもよるが システムと問題の性質に依存して決まってくる値となる。

全体スピードアップと総合並列化効率例を 図 1、図 2に示す。

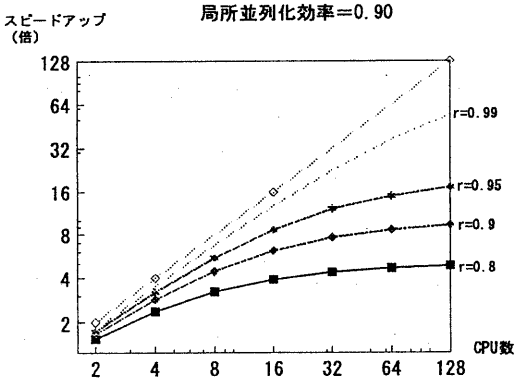


図 1 全体スピードアップ S

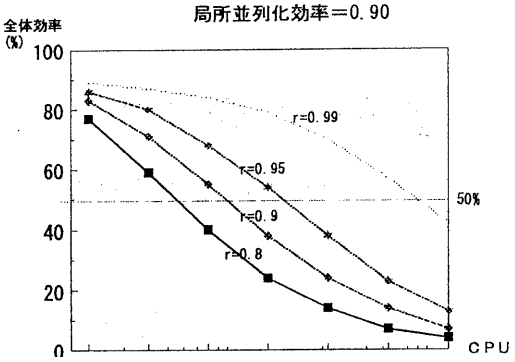


図 2総合並列化効率 ：S/m

2．2　並列化対象部分の比率

並列化の効率をあげるためには、並列化対象部分の比率が重要となる。この比率は、プログラム実行時の各部分の実行時間の測定と、全体に対する割合から算定する。この比率は、一般にモデルの規模に応じて高くなる。

表 1　並列化対象部分の比率例: N 次元の連立一次方程式を解く時。

例題：3次元ソリッドを3方向にk分割
差分法形式。

係数行列の計算他：非並列化部分
行列の三角分解　：並列化部分

分割数 k	10	15	25	40	50	80	100
並列化対象部	18 %	51	86	97.8	99.0	99.8	99.9

2．3　並列化効率はスケーラブル

アムダールの法則は、効率が頭打ちになる形で、よく悲観的に考えられがち であるが、並列化対象部の比率（ r ）は固定ではなく、上記の様にモデルの規模によって十分 1.0 に近ずく。アムダールの法則を積極的に捉え、従来より大きい問題を解くために並列化すると考えると、図 3に示すように問題規模の拡大に応じて、スケーラブルに高速化が可能。

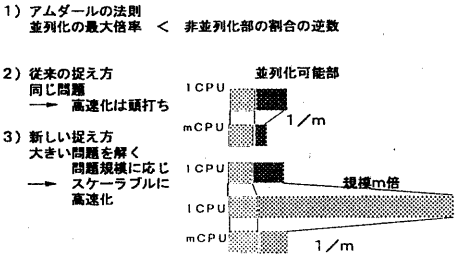


図 3　スケーラブルな並列化効率

3．システム概要

CONVEX社の Exemplar SPP1000(Scalable Parallel Processing)シリーズは、ワークステーションの扱い易さを継承しながら、従来の超並列コンピュータを超えた実行性能を発揮する超並列コンピュータとして開発されている。

3．1　Exemplar SPP1000の特長

Exemplar SPP1000シリーズの主な特長を挙げる。

（1）柔軟なアーキテクチャ
- 柔軟性の高いスケーラブルなシステム構成
- 全システムにわたる共有メモリ方式

（2）豊富なアプリケーション
- HP-UXアプリケーションとバイナリ(ABI)互換
- CONVEX Cシリーズのソースレベル互換
- 自動並列化機能を持つコンパイラ
- 容易な並列化をサポートするグラフィカルな開発環境

3．2 スケーラブルシステム構成

これまで多くのスーパーコンピュータやワークステーションに使われてきたバスベースのＳＭＰ（Symmetric Multi Processor)アーキテクチャでは、システムを増設すると、通信ネックによる性能低下が起こりがちであった。これに対し、システムを増設しても、それに見合った性能を頭打ちなく得られることを「スケーラブル」と呼ぶ。

CONVEX SPP1000は、8プロセッサまでを１個のハイパーノードとし、そのハイパーノードを複数個 接続した、２段の階層構造を採用している。最小２プロセッサから最大128プロセッサまで、スケーラブルな構成である。ハイパーノードの数に対応して、メモリ、I/O、ハイパーノード間通信も同時にスケーラブルに増加可能で、すべての範囲にわたってバランスを考慮している。エントリから、ハイエンドまで、ユーザの必要なだけのアップグレードができる。

(1) ハイパーノードとハイパーノード内の通信

各ハイパーノードは８個までのPA-RISC7100プロセッサとメモリ、I/O、それらを接続する高速クロスバー・スイッチから構成されている（図 4）。メモリは一個のハイパーノードあたり２ギガバイトまで搭載することが可能で、システム全体で32ギガバイトの大容量実メモリが利用できる。一個のハイパーノード内ではメモリは８つのバンクに分割され、インタリーブ方式で記憶を行うため、連続したデータの高速アクセスが可能。また、これらを接続するクロスバー・スイッチはハイパーノードあたり １ギガバイト/秒のメモリバンド幅を持ち、かつ各プロセッサとメモリ・Ｉ／Ｏ間の同時通信を可能とし、従来のバスシステムに発生する通信ネックによるシステムの性能低下を防ぐ。

(2) ハイパーノード間の通信

それぞれのハイパーノードは４本の高速なインターコネクトリングによって接続される。このインターコネクトリングはIEEEのSCI(Scalable Coherency Interface)を超並列システム用に改良したもので、CCTI(Coherent Toroidal Interconnect)と呼んでいる。１つのCTIは64ビット幅の単方向リングで、レイテンシの少ないパケット通信により、ハイパーノード間で600メガバイト/秒の高速通信を行う。

(3) 共有メモリの実現

SPP1000の最大の特長である全システムにわたる共有メモリを高速に実現するのが、ハードウェアによるデータ整合機構を持つメモリコントローラとCTIキャッシュである（図５）。CTIキャッシュはハイパーノード間のデータ利用に対するキャッシュとして、実メモリ上に完全に区分けし、メモリコントローラで管理する。これによりシステム中の全てのプロセッサが他のハイパーノードのデータを効率良く参照することができる。特に、そのハイパーノードで使用されるデータはシステムが集めてくる事になり、同一のデータを連続してアクセスする場合には、データの局所性の考慮が不要で、非常に有効な技術となる。また、ハイパーノード内のプロセッサキャッシュの整合もメモリコントローラで管理される。

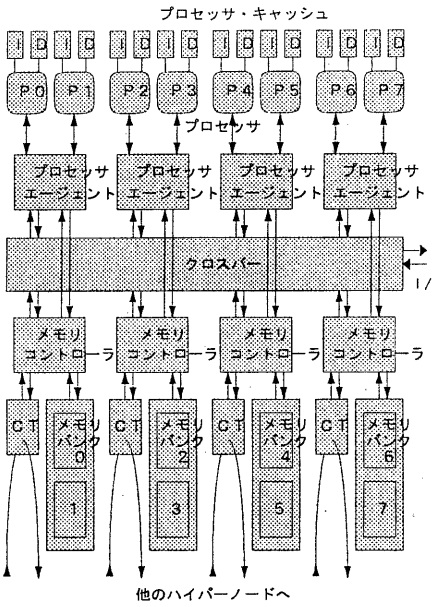


図　4　ハイパーノード内の構成

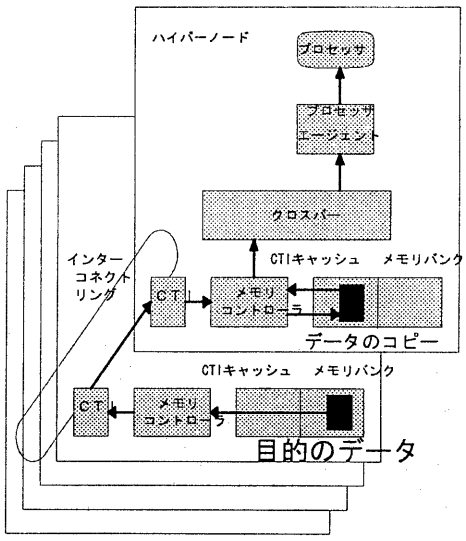


図　5　ハイパーノード間の通信とCTIキャッシュ

(4) Ｉ／Ｏサブシステム

SPP1000のI/Oサブシステムはハイパーノードのクロスバーに接続され、必要に応じて各ハイパーノードに分散できる。　一つのI/Oサブシステムのバンド幅は250メガバイト/秒と非常に高速で、さらにそれらが独立に作動するためシステム全体としては最大４ギガバイト/秒の高速な入出力を可能にする。それぞれのI/Oサブシステムは８つのSBUSポートを持っており、このSBUSには各種の入出力コントローラと入出力装置を接続できる。

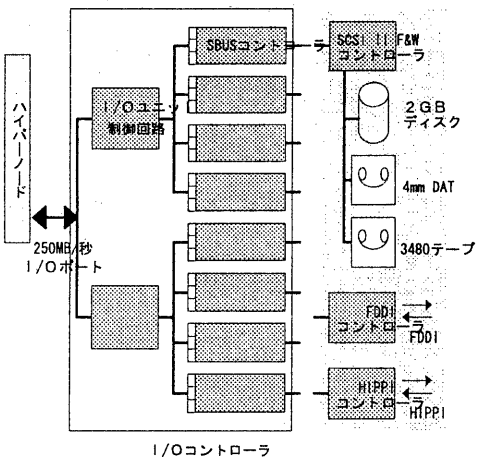


図６　Ｉ／Ｏサブシステム

３．３ オペレーティングシステム

コンベックスはSPP1000のオペレーティングシステムとして、OSF1/ADのMachマイクロカーネルを基本としたSPP-UXを開発した。

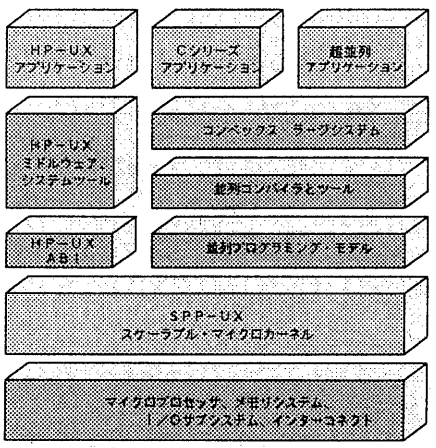


図７　オペレーティングシステム

また、HP社のUNIXであるHP-UXとのABI（アプリケーション・バイナリ・インタフェース）互換機能を備え、HPのワークステーション上で稼動している数多くのアプリケーションソフトやツール群をそのまま利用することが可能。さらにコンベックスのベクトル/パラレル型スーパーコンピュータＣシリーズとのソースレベルでの互換性も持つ。

３．４　自動並列化コンパイラ

(1)　コンパイラ概要

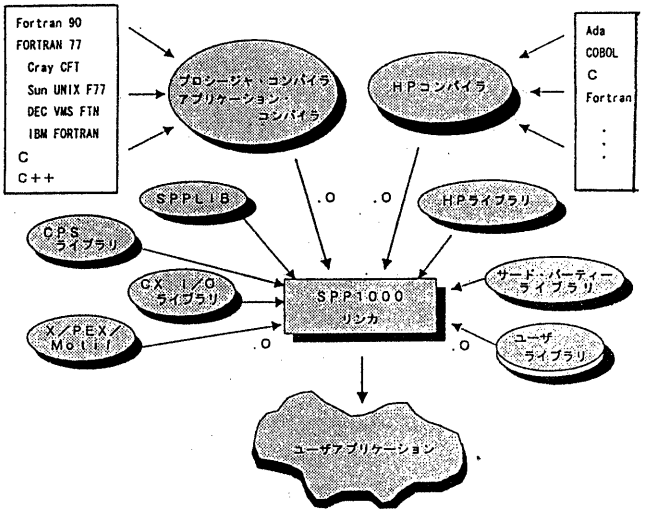


図 ８　自動並列化コンパイラ

このコンパイラは標準のFORTRAN 77、Ｃ、Ｃ++などの言語やそれらが混合したソースコードを自動的に並列化できる。

プロシージャ・コンパイラはコンベックスがＣシリーズで培ってきた最先端の最適化技術をさらに進化させたもの。繰り返し処理のブロッキングやアンロール、ジャムなどの手法を用い、データの局所性を自動的に高め、アクセス速度を向上する。また、生成されたオブジェクトは、HP-UXコンパイラで生成されたオブジェクトや、各種のライブラリと自由にリンクできる。このためユーザは今まで蓄積したオブジェクトコードを利用できる。さらに、アプリケーション全体を見て最適化・並列化するアプリケーションコンパイラもある。

(2)　コンパイラによる並列化

コンベックスの FORTRAN コンパイラは、共有メモリモデルで、ループを主な対象として自動並列化を行なう。その並列化の使い方は次の 3レベルがある。

(A) 自動並列化

基本的には、従来の順次処理型のソースコードのままで、コンパイラが自動的に　　並列化可能部分を抽出し、並列化する。

(B) アシストされた並列化

ユーザが、データ依存性等の追加情報を並列化用コンパイラディレクティブと　　してソースコードに挿入することで、コンパイラの判断を助け、より広い対象範囲の自動並列化を可能とする。

(C) マニュアルによる並列化

ユーザが、並列化実行の指示やメモリ配置まで含むコンパイラディレクティブ　　をソースコードに挿入し、コンパイラに並列化を行なわせる。

4. プログラミングモデル

4．1． 共有メモリアプリケーション

並列処理の際、メモリを複数の各処理単位（スレッド）が共有することが可能な方式。サブコンプレックス内で、１プロッセッサ当たり１スレッドが対応する形の マルチスレッドで構成される単一プロセスとして実行される。

コンパイラは、ループ内の処理をスレッド間に分割します。アプリケーションは、ランタイム時に使用するプロセッサ数に応じてスレッドを生成するように作られ再コンパイルは不要。

プログラムの並列化は、コンパイラが自動的に行う。ハイパーノード間のデータの分配は、CTIキャッシュを通してＳＰＰ-ＵＸが自動的に行う。

コンパイラのディレクティブやプラグマでより深い最適化が行える。

4．2 メッセージパッシングアプリケーション

並列処理の際、明示的なメッセージの交換でプロセス間の通信を行い、並列に動作する独立したプロセスの集合として実行される。サブコンプレックス内では、プロセッサ毎に１プロセスとなり、外部のマシンを使用するのと同様に、１個またはそれ以上のサブコンプレックスを使用できる。

データの通信とプロセス間の同期は、ORNLのＰＶＭをベースにコンベックスで開発したメッセージパッシング機能を用いて行う。実装は共有メモリを用い、プロセス間の直接通信です。共有メモリ内にメッセージはパッキングされます。ＰＶＭデーモンは介在せず効率の良いデータ転送が行われる。

各ＰＶＭのプロセスは独立な４ＧＢの仮想アドレス空間を持つ独立した単一スレッドとして実行される。

4．3 ハイブリッドアプリケーション

ハイブリッドアプリケーションは、共有メモリプログラムの集合で構成されて おり、各プログラムがメッセージパッシングを用いて全体のコーディネートを行うものである。

このモデルでは、メッセージパッシング時のプロセスのプライベートメモリについての利点をフルに使用しながら、プログラムの大部分を共有メモリスタイルで記述できる。

4．4 メモリモデル

SPP1000ではコンパイラによる自動並列化のほかに、５つのメモリクラスを意識して使い分けるさらに進んだ最適化手法が用意されています。この５つのメモリクラスはそれぞれ違ったデータアクセス方式を持ち、

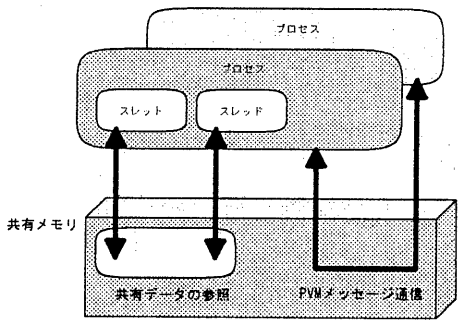


図 9 共有メモリ方式とメッセージパッシング方式

ユーザは目的に合ったクラスを指定する事ができます。

スレッドプライベート
ノードプライベート
ニアシェアード
ファーシェアード
ブロックシェアード

図 １０ メモリモデルの種類

5 並列処理性能

並列処理性能の一例を下図に示す。これは、ハイパフォーマンスコンピューターの性能評価によく使われる，LINPACKのベンチマークテスト結果である。

１つのハイパーノード内の場合ばかりでなく、２つのハイパーノードの場合でも、スケーラブルな性能が得られることが分かる。

なお、この結果は1994年10月現在のもので、最終的な性能ではない。

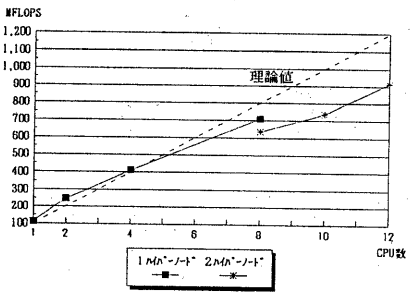


図5 LINPACK 1000x1000 ベンチマーク結果

6．商用アプリケーションの並列化

並列化に取り組んでいる ISV（独立ソフトウエアベンダ）の商用コードの例を以下に示す。

表 2 ISV並列化コード

MCAE	MSC/NASTRAN ANSYS ABAQUS MARC PAM/CRASH LS/DYNA
CFD	FLUENT RAMPANT STAR-CD
CCS	Gaussian Mopac Discover CHARMM Amber GAMESS

主力はシミュレーションで、ISV のコードもユーザのインハウスコードも、そこでの数値解法の並列化が中心課題となり、次の２つの並列化の方向がある。
1) 連立一次方程式解法ルーチン：ソルバーの並列化
2) 解法レベルでの並列化：領域分割法

ISVの商用コードでは両方取り組まれているが、将来的には、機種依存が少なく、 コースグレインとなって並列化効率が上がりやすい領域分割法（ＤＤＭ）が一般化する傾向にある。

7．共有メモリモデルプログラム例

’FEM’プログラムについて、コンベックス SPP1000 による並列化を検討した。

7．1 プログラム構成

(1) プログラム内容解読

FEM による３次元場の問題。係数行列を LDLt に三角分解して、基本解を求める。右辺ベクトルの値が変化した時の目標点への影響を、多数の右辺ベクトルについて求める。

(2) プロファイル

プログラム各部の計算時間の比率を１CPUで求めた。前進消去と後退代入で計 97% を占める。

(3) 並列化対象部と見込み第１

前進消去と後退代入部を並列化する。

期待されるスピードアップ S は、cpu数 m=8 で、局所並列化効率 e=0.90として、

$$S = 1/(1-r + r/(m*e)) = 6.0 \text{（倍）}$$

7．2 最適化／並列化

冗長な演算のスキップ、添字計算の集約、簡略化、小規模なサブルーチンのインライン化、キャッシュ再利用を考慮したブロック化とコンパイラの自動並列化機能と指示文で最適化／並列化。

7．3 並列化効率

下表に、CPU数に応じた計算エラプス時間比（対１cpu）と並列化効率を示す。

表 3 最適化／並列化後の計算結果： SPP1000

cpu 数	1	2	4	6	8
倍率	1.0	1.86	3.92	5.19	6.82
効率 (%)	--	93.1	98.1	86.6	85.3

7．4 考察

並列化対象部の比率 r = 0.97(不変と仮定）、cpu数 m = 8、スピードアップ S = 6.82 から、スピードアップ S = 1 /（1-r + r/(m * e)）の式より局所効率 e を求める。6.82 = 1 /(1-0.97 + 0.97/(8*e)) となり 局所効率 e = 1.06 となる。このケースではCPU数 の増加に伴い、キャッシュサイズが増加して1.0 以上の効率に寄与していると考えられる。

8．メッセージパッシングプログラム例

：領域分割法（ＤＤＭ）FEM静的解析

メッセージパッシングの例として、ＰＶＭを用いた領域分割法のプログラムを作成し、SPP1000で実行した。使用したＤＤＭのプログラムのオリジナルは東京大学矢川教授の開発されたもので、FEM静的構造解析をネットワーク上の複数のマシンで実行する。これを、CONVEXのＰＶＭを利用する形に変更し、8CPUのSPP1000で実行した。

8．1 領域分割法の基本フローと並列化

線形構造解析の基本フローから、並列化した領域分割法のアルゴリズムは、図 １２となる。

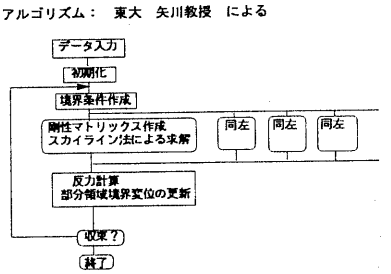
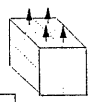


図 １２ 並列化した領域分割法

8．2 SPP1000での実行結果

これをSPP1000で実行した結果を図 １３に示す。使用したテスト計算モデルは下面固定の３次元のソリッドを上方に一様に引っ張ったもの。

・オリジナルコード： 東大 矢川教授 開発
・ＦＥＭ 線形構造解析
・ＰＶＭ 版にＮＫＫで修正
・ＣＯＮＶＥＸ SPP1000 で実行



エラプス時間

CPUs	Elapse(SEC)	Effic.(%)
1	236	100.0
2	120	95.2
3	91	86.4
4	63	93.7

モデル： １領域 4x4x4 8節点ソリッドＦＥＭ、８領域

図 １３ SPPでの領域分割法計算結果

おわりに

本報告では、実用域に入った並列計算機のプログラミングについて、CONVEX SPP-1000 を例として述べた。今後多くの方が並列計算に取り組まれると考えられ、その御参考になれば幸いです。