JAERI-Data/Code
98-015

酸性雨予測計算コードSTEM2の並列化

1998年3月

北端秀行・土浦宏紀・植田洋匡*・相川裕史

日本原子力研究所
Japan Atomic Energy Research Institute

本レポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所研究情報部研究情報課（〒319-1195　茨城県那珂郡東海村）あて，お申し越しください。なお，このほかに財団法人原子力弘済会資料センター〒319-1195　茨城県那珂郡東海村日本原子力研究所内）で複写による実費領布をおこなっております。

This report is issued irregularly.
Inquiries about availability of the reports should be addressed to Research Information Division, Department of Intellectual Resources, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-1195, Japan.

編集兼発行　日本原子力研究所
印　　刷　㈱原子力資料サービス

JAERI－Data/Code 98－015

酸性雨予測計算コードSTEM2の並列化

日本原子力研究所計算科学技術推進センター
北端　秀行・土浦　宏紀・植田　洋匡*・相川　裕史

（1998年2月6日受理）

酸性雨予測計算コードSTEM2（Sulfer Transport Eulerian Model-Ⅱ）に対し、スカラ並列計算機SPとベクトル並列計算機VPP300のタイプの違う2機種の計算機を用いて並列化を行った。その結果、スカラマシンSPの34並列では13.7倍、ベクトルマシンVPP300の12並列で34.6倍の高速化を実現することができた。STEM2は、有害な一次汚染物質であるNO_x、SO_xがラジカル等の高速反応やメタンなどの低速反応を経て、リージョナルスケールの光化学オキシダント、酸性雨にいたる過程を追跡する数値モデルで、大気中に存在するほとんどすべての大気汚染物質を含んでおり、複雑な化学反応過程をモデルの中に組み込んでいるのが最大の特徴である。本報告書ではこの酸性雨予測計算コードSTEM2に対して行ったベクトル化・並列化の方法とそれによる性能向上について報告する。

日本原子力研究所：〒153-0061　東京都目黒区中目黒2-2-54
＊ 京都大学防災研究所

JAERI−Data/Code 98−015

Parallelization for STEM2 Code in Simulating Acid Rain

Hideyuki KITABATA, Hiroki TSUCHIURA, Hiromasa UEDA*
and Hiroshi AIKAWA

Center for Promotion of Computational Science and Engineering
Japan Atomic Energy Research Institute
Nakameguro, Meguro-ku, Tokyo

(Received February 6, 1998)

STEM2 (Sulfer Transport Eulerian Model- Ⅱ) Code in simulating acid rain is parallelized by using two different types of super computer SP which is a scalar machine, and VPP300 a vector one. The result of the parallelization on SP is that it enables to obtain the 13.7 times higher performance with 34CPUs. On VPP300 its performance is enhanced up to 34.6 times by both of the vectorization and parallelization with 12CPUs. STEM2 is a numerical model which is characterized by the detailed chemical processes including the concentration of primary and secondary pollutants and dry and wet depositions during long-range transport. This report shows the parallelization and vectorization method for STEM2 Code and their performance on SP and VPP300.

Keywords: Stem- Ⅱ , Acid Rain, Air Pollutant, Chemical Process, SP, VPP300, Parallelization, Vectorization

* Disaster Prevention Research ins., Kyoto Univ.

目　　次

Contents

1. はじめに

最近、日本では新たにゴミの焼却によるダイオキシン問題が大きな社会問題となっているが、成層圏オゾン層破壊、地球温暖化、酸性雨という広域的でかつ長期的な地球規模での大気環境問題がきわめて深刻さを増している。なかでも、近隣の東アジア地域においては、急速な経済成長や爆発的な人口増加を背景に、化石燃料の大量消費や化学汚染物質の大量排出が問題となっており、大気汚染の拡大が今後さらに懸念される状況である。このような状況から、日本で大気汚染の低減を計画するためには、自国の発生量だけではなく、越境移動を伴う汚染物質の長距離輸送の影響を考慮する必要があり、そのため現在では、リージョナル・スケールでの大気汚染物質の長距離輸送を解析する数値シミュレーションモデルの開発が重要な研究課題となっている。

STEM2(Sulfer Transport Eulerian Model-II)[1)]は一次汚染物質による大気汚染から、ガス状二次汚染物質、粒子状物質による大気汚染を経て、酸性雨、酸性霧に至る一連の汚染形態を的確に予測する目的で作られた3次元オイラー型数値モデルであり、数10種にのぼる化学種について、時間スケールの異なる100以上の素反応を含んだ詳細な化学反応計算モデルに特徴がある。ただ、このSTEM2は物理、化学過程を詳細に扱うことで、他の簡略型モデルと比べ非常に大きな計算時間の実行コストが必要となる。そのため、長時間シミュレーションの実行や計算精度のさらなる向上を図る上でシミュレーションコードの高速化が不可欠である。

そこで我々はこのSTEM2を高速化することを目的とし、分散メモリ型スカラ並列マシンSP(IBM製)と分散メモリ型ベクトル並列マシンVPP300(Fujitsu製)の2機種のスーパーコンピュータを用いて、各マシンでの並列化を行った。この種の詳細型化学反応計算モデルに対して、ベクトル化、並列化等のプログラムの高速化に関する技術的知見を新たに開拓することは、今後の大気環境モデルにおける大規模シミュレーションソフトウェアの開発を行う上でも有効である。なお、本報告書で行った並列化はすべて通信にMPI(業界標準のメッセージライブラリ)を使用しており、MPIをサポートするマシンであれば機種によらず計算が可能である。以下、本報告書の構成について紹介する。

まず、第2章ではSTEM2のモデルの概要を説明し、アルゴリズムを解析する。第3章ではオリジナルソースコードの分析結果について報告する。ここでは実際にスカラマシンSPとベクトルマシンVPP300でオリジナルソースコードの実行時間を計測し、各ルーチンの実行コスト分布やベクトル化率を調査した。そしてその結果をもとに、第4章ではスカラマシンSPで行った並列化について報告する。また、ベクトルマシンVPP300については、ベクトル化とベクトルチューニング後の並列化をそれぞれ第5章と第6章に分けて報告する。そして、最後に第7章でまとめを行う。

2. STEM2 のアルゴリズム

2.1 モデルの概要

酸性雨影響予測計算コード STEM2 は、アイオワ大学の Carmichael[2)]等によって開発された 3 次元オイラー型の数値計算モデルである。このモデルの特徴は大気中の一次汚染物質である窒素酸化物 (NO_x) や硫黄酸化物 (SO_2) から、光化学オキシダントやエアロゾル等の二次生成粒子にいたる大気中に存在するほとんどすべての大気汚染物質を含み、長時間、長距離輸送される間の移流、乱流による拡散、そして化学反応による変質、乾性及び湿性沈着など総合的にシミュレートできるところにある。STEM2 はもともと北米を対象に開発されたコードであるが、その後共同研究等でいろいろな目的に応じて改良されており、今回高速化に取り組むコードは東アジア領域を対象に改良されたコード[4, 5)]である。

2.2 計算領域と座標系

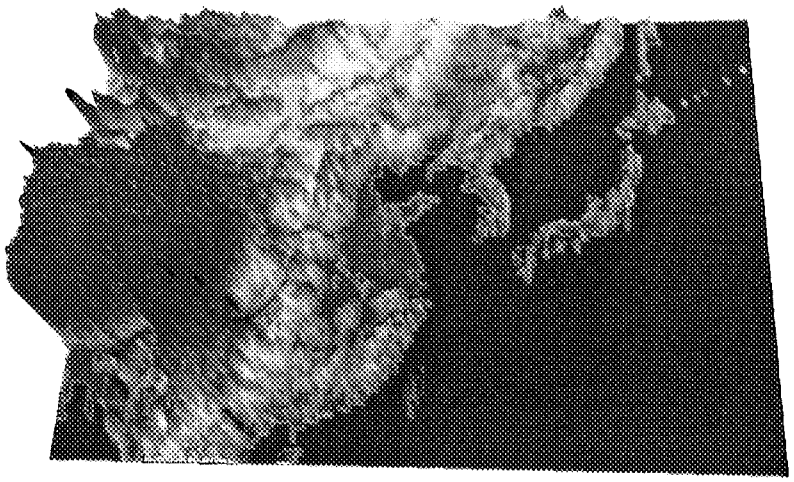


Fig. 1: 東アジアにおける計算領域

Fig.1 に STEM2 の計算領域を示した。経度方向 (x 方向) に 4950km、緯度方向 (y 方向) に 3450km、そして鉛直方向 (z 方向) は地表面から上空 4km までを計算領域としている。また、座標系 (x,y,z= σ) は x-y 面が平面にとった 2 次元直交座標、z 軸方向は地表面の起伏を考慮したσ座標を採用している。z^*は直交座標系の z、h は地上からの高さ、そしてΔHは地表面から上空 4km までの大気層の厚みとすると、σ座標は次式で与えられる。

$$\sigma = \frac{z^* - h(x,y)}{\Delta H} \tag{1}$$

2.3 支配方程式

今回チューニングの対象コードとして用いた東アジア地域版STEM2では、湿性沈着を含まないので気相化学反応のみ扱う。従って、支配方程式は以下に記した気相の物質収支を表す保存式となる。

$$\rho\frac{\partial C_i}{\partial t}+\frac{\partial\left(\rho C_i v_j\right)}{\partial x_j}=\frac{\partial}{\partial x_j}\left[K_{jj}\frac{\partial \rho C_i}{\partial x_j}\right]+R_i+S_i+G_i \qquad (2)$$

i (化学種) ＝ 1、・・・34

j (方 向) ＝ 1、2、3

v_iは風速、C_iは化学種 i の濃度、K_{jj}は乱流拡散係数、R_iは化学反応による生成速度[?]で反応物質の濃度や気象条件等から決まる。S_iは工場等からの排出量から乾性沈着量を差し引いた量(単位時間あたり)、G_iは雲粒･雨滴から気相への蒸散速度、ρは大気の密度である。また、このSTEM2の中で取扱われている化学反応の素反応は112あり、化学物質の総数は54種である。尚、54種のうち輸送追跡される比較的寿命の長い化学種は34種で、残りの20種に関しては、寿命が短いので生成後すぐに反応を繰り返して消滅するようモデル化されている。詳しくは文献参照のこと。[2][6][7]

2.4 計算スキーム

STEM2は入力データとして、化学汚染物質の排出量データや地形データが必要である。また、3次元の時間変化する気象場のデータ、及びそれらから計算される入力データとしては、風速、気温、湿度、大気の密度、乱流拡散係数、乾性沈着速度のデータが必要であり、これらは大気シミュレーションモデルを並立させて計算し、予測することも可能であるが、今回の計算では予め作成したデータを逐次取り込み計算を行った。

次に、上述した3次元微分方程式の解法の手順を解説する。まず、LOD-FEM (Locally One Dimensional Finite Element Method) 法[?] を用いることで、x、y、z 軸の3方向に関する移流拡散方程式と化学反応計算とに分ける。その結果これらの4つの方程式は時間に従属する一次元問題として取り扱うことができる。(2) 式のρ、S_i、G_iは定数として入力データから取り込まれる。さらに移流拡散方程式はPetrov-Galerkin有限要素法[?]を用いて離散化され、生成された線形方程式はLU分解を通して解を求めることができる。一方、化学反応計算は反応の時間スケールが移流拡散の時間スケールと比べ非常に短いことから、輸送の計算部とは完全に独立した時間ステップを持ち、半陰解法による数値積分から解を求めている。

3. ソースコード分析

3.1 全体のルーチン構造

サブルーチン全体の Tree 図を Fig.2 に示す。STEM2 コードは大きくわけて、移流拡散方程式を解く輸送計算部 (HORX：x 方向、HORY：y 方向、VERTCL：z 方向) と化学反応計算部 (RXN)、及び I/O 部によって構成されている。また、I/O 部中の INPUT1 は初期化のルーチンで、最初に 1 回のみ実行される部分であるから、本質的に高速化の評価を議論する際には不要である。

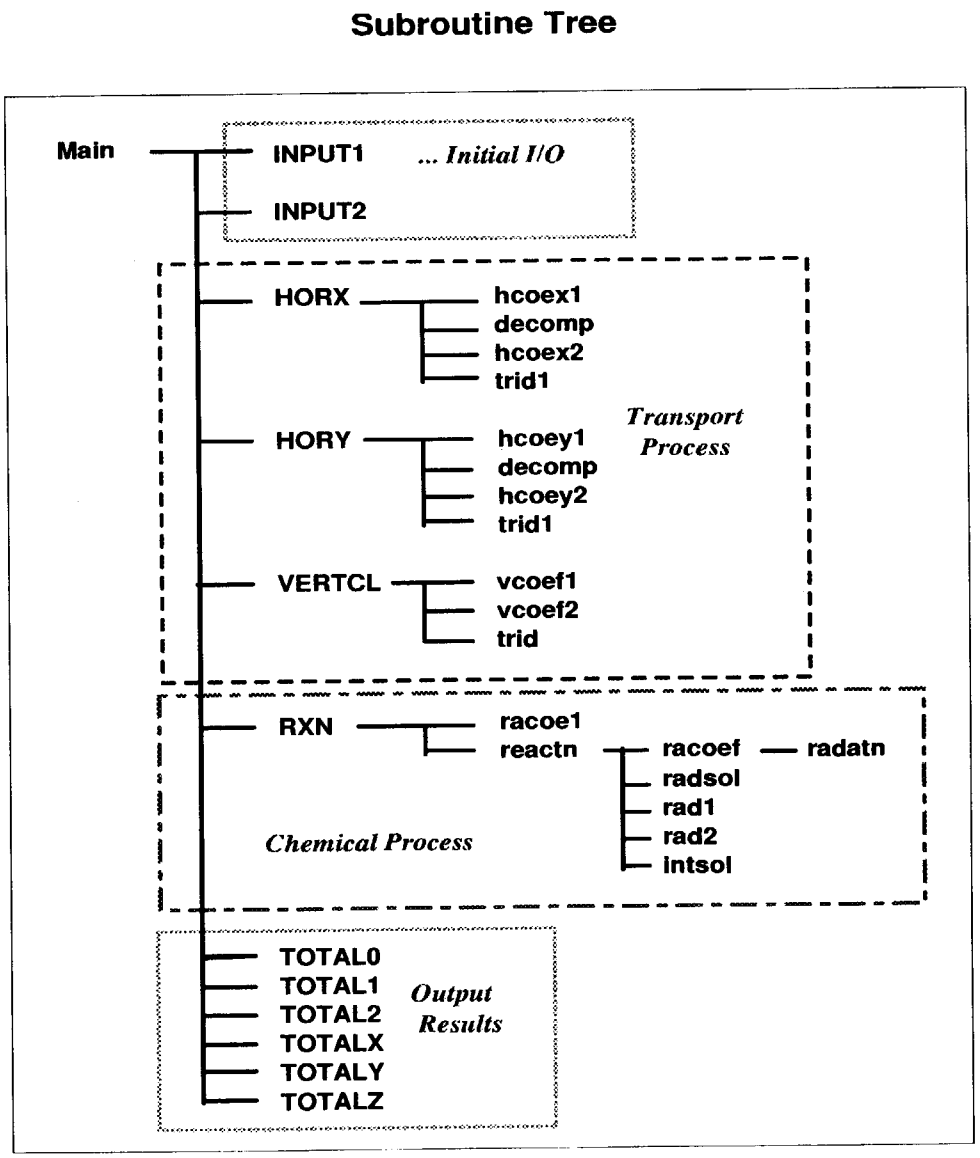


Fig. 2: オリジナルソースコードの Tree 構造

3.2 実行時間コスト

STEM2のオリジナルコードを単一プロセッサにおいて最適化し、ベクトル計算機(VPP300)及びスカラ計算機(SP)を使用して、1CPUで4時間分の実データによる計算を実行した。その結果、各マシン上でのルーチン毎のコスト分布情報を採取した。

(1) コードの規模

本調査に用いたソースコードの問題規模は次の通り。

・コードステップ数 2855 STEP
・メッシュ数 x軸方向 67 y軸方向 47 z軸方向 11

(2) 解析に用いたツール

VPP300 : SAMPLAR, PEPA, gettod *SP* : prof, rtc()

(3) 使用したコンパイルオプション

VPP300 : −Of *SP* : −O3 −qarch=pwr2 −qstrict −qfloat=hssngl & mass library

(4) 解析結果

SP及びVPP300を用いて4時間分データの計算を行った。その解析結果をFig.3, Table1に示す。これによると、オリジナルコードでの計算結果はVPPの方がSPよりも計算時間が約2倍多くかかっていることがわかる。これは後で示すベクトル化率の低さが原因で、ベクトルマシンであるVPPはその性能を十分に活かすことができない。また、両マシン共に計算コストの大半(80% ～)を化学反応計算で費しており、STEM2の高速化を考える場合、まず第一にこの化学反応計算部をいかに高速化するかが課題であると言える。

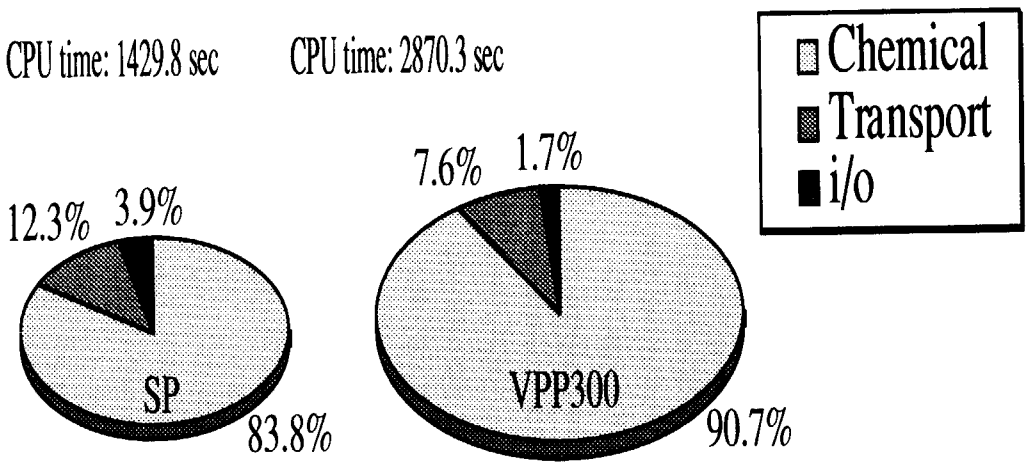


Fig. 3: SP 及び VPP300 における実行コスト分布

Table 1: 各サブルーチンの CPU time (Simulation for 4 hours)

	ルーチン名	CPU(sec)	(%)
SP	RXN	1207.2	83.8
Total CPU time:	HORX	36.8	2.6
1439.8 sec	HORY	44.7	3.1
	VERTCL	95.3	6.6
	I/O	55.8	3.9
	(INPUT1)	(23.1)	(1.6)

	ルーチン名	CPU(sec)	(%)
VPP300	RXN	2602.9	90.7
Total CPU time:	HORX	46.1	1.6
2870.3 sec	HORY	46.7	1.6
	VERTCL	124.8	4.3
	I/O	49.8	1.7
	(INPUT1)	(32.8)	(1.1)

3.3 ベクトル化率

VPP300 でベクトル化率の測定を行ったところ、次のような結果が得られた。

Table 2: 各サブルーチンのベクトル実行コスト (VPP300)

サブルーチン名		COST(%)	VHIT	VL
化学計算部：	RACOEF	35.7	0.0	−
	REACTN	25.4	19.4	16
	RADSOL	13.2	9.0	21
	INTSOL	11.9	30.1	6
	RAD2	2.4	48.0	17
	RACOE1	1.5	0.0	−
	RXN	0.7	14.3	12
輸送計算部：	VERTCL	2.7	16.0	11
	HORX	1.6	57.1	65
	HORY	1.6	81.1	45
	VCOEF2	1.6	29.7	10
	VCOEF1	0.1	75.7	150
	HCOEY1	0.0	12.7	35

VU ビジー率;

- STEM2 全体 (MAIN)
  ... 13.13 %
- 化学計算部 (RXN)
  ... 12.98 %
- 輸送計算部 (HORX,HORY,VERTCL)
  ... 16.48 %

〈定義〉

VU ビジー率: 計測時間のうちベクトル演算器が作動していた時間の割合 (PEPA で測定)

COST: 実計算時間のコスト (SAMPLAR で測定)

VHIT: ベクトル化率の指標 (ベクトル化率の大雑把な目安、 SAMPLAR で測定)

VL: 平均ベクトル長 (SAMPLAR で測定)

4. *SP* によるスカラ並列

4.1 並列化方針

STEM2 の並列化はスカラ並列、ベクトル並列共に標準メッセージライブラリ MPI を用いて行った。並列化の方法としては、計算領域 (xyz の 3 次元グリッド) を分割して各プロセッサに割当てる領域分割法を用いた。領域分割の方法としてはサイクリック分割を採用した。
STEM2 の計算量はイタレーション計算の関係上、化学汚染物質の濃度に比例するので、大規模汚染源と排出源の無い地域とでは自ずと計算量に大きな差が現れる。このためブロック分割ではプロセス間でのロードバランスが著しく不均等になるが、サイクリック分割して合計すればプロセス間のバラつきが相殺され、並列化効率を上げることができる。ただ、通常サイクリック分割は領域間でのデータ参照がある場合には、通信量が多くなるので適さない。しかしながら、その点に関しても STEM2 では、1) ホットスポットである化学反応計算部において差分計算が全くなく、計算領域間でのデータ通信が不要である。2) 輸送計算部においても、x,y,z の各成分に分けることで、1 次元方向の移流・拡散計算となり、他の 2 成分に関して完全に独立性が保たれている。以上の 2 つの理由から通信によるオーバーヘッドの懸念は排除できる。

4.2 化学反応計算部の並列化

化学反応計算部では、内部の演算に使われている配列はメッシュ毎に完全に独立に計算をおこなっている。最外のループがメッシュループで囲まれていることから、下図の様に、サイクリック分割で容易に並列化が可能である。

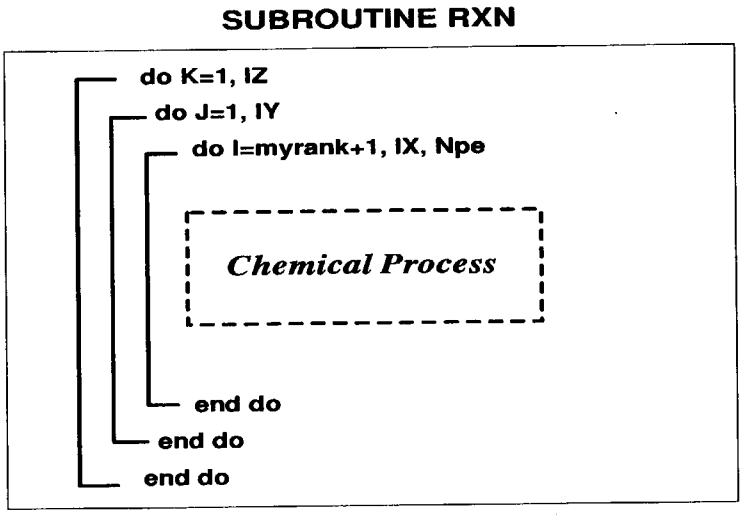


Fig. 4: RXN の並列化 (index I での分割)

尚、分割軸はx軸のインデックスIとした。これはx軸方向のメッシュ数が67と最大であることが理由である。

4.3 輸送計算部の並列化

輸送計算部はx軸方向、y軸方向、z軸方向の3成分に分けて計算を行い、それらは順にサブルーチンHORX、HORY、VERTCLで独立している。各サブルーチンのループ構成は、例えばHORXのようにx軸方向の輸送計算の場合は、y及びz方向の2重メッシュループというように、対象となる方向とは別の2方向のメッシュループで囲まれている。つまりHORXでは、y、zの2重ループの中でx方向の輸送計算(x軸に対して依存関係を持つ)を行っている。従って、対象外のループ(x軸依存であればy or z、y軸依存であればz or x、z軸依存であればx or y)で領域分割することは容易であるが、輸送計算ルーチンの3つすべてを同一のインデックスで分割することは不可能である。

では、いったいどの軸を分割軸にとるのが最適だろうか? 最も効率良く並列化を行うにはすべての領域分割において分割軸を統一することが望ましい。なぜなら、ルーチン間のデータの受渡しに通信が不要となるからである。輸送計算部のルーチンはすべて同様のループ構造を持っていることから、ここでは化学反応計算部と同じく、最もループの回転数が大きいx軸で分割を行うものとする。

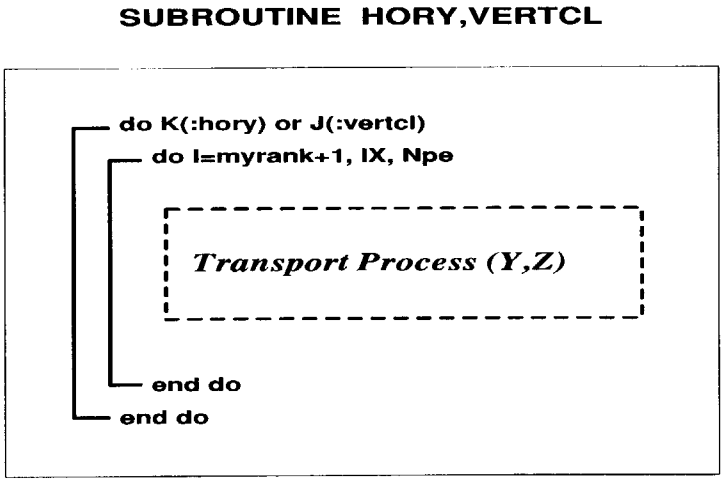


Fig. 5: HORY,VERTCLの並列化 (index Iでの分割)

サブルーチンHORY、VERTCLについてはFig.5に示すように、x軸に対するメッシュループが輸送計算とは独立なループ構造を持つことから、x軸のインデックスIでそのまま容易にサイクリック分割することができる。ところで分割軸をx軸と決めた以上、問題となるのはサブルーチンHORXである。インデックスIで分割不可能なHORXでは、メッシュ数が47とx軸の次にループの回転数が大きい、y軸のインデックスJで分割を行う。しかしそのためにはFig.6 (MPI communication)に示したようにサブルーチンHORXの最初と最後において、通信が必要となら

ざるを得ない。以後具体的にその作業手順を説明する。

サブルーチンHORXがコールされると、最初にMPI通信(***MPI_ALLREDUCE***)により、各プロセッサが分け持つ配列データをルート(ランク0のプロセッサ)に収集して元の配列に再構成する。そして再構成された配列データをすべての各プロセッサに転送する。つまりここでの作業は、各プロセッサがそれぞれ冗長実行できる状態へと復元させることが目的である。その結果HORX内の演算を改めて領域分割することが可能となる。次に、HORX内の計算時間を短縮するために改めてy軸(インデックスJ)でサイクリック分割を行う。サイクリック分割することでHORX内の計算量をプロセッサ数で分けることができ、計算時間を短縮することに繋がる。そして最後に、サブルーチンHORXの前処理として行った作業同様、再びMPI通信を行って各プロセッサが分け持つ配列データを、収集、転送を経て冗長実行可能な状態に復元する。以上これらの作業によって、他のサブルーチンでは矛盾なく配列の受渡しができ、並列計算が実行可能となる。

尚、データの収集方法としてはデータの圧縮や復元が不要である、重ね合わせ[12]を用いる。

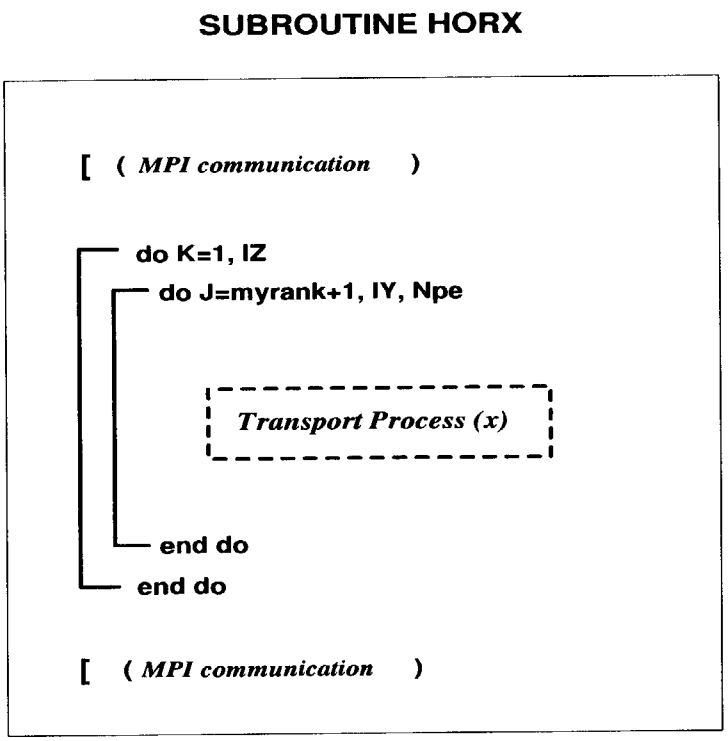


Fig. 6: HORXの並列化 (index J での分割)

以上STEM2のスカラ並列に関して、化学反応計算部と輸送計算部に分けて議論してきたが、全体を通して言うと、出力データの算出部分を除き通信は輸送計算部のHORX内のみであり、それ以外の化学反応計算部RXN、輸送計算部HORY、VERTCLでは全くデータ転送が不要である。従って、これらのサブルーチンでは非常に高い並列化効率が期待できる。また、各ルーチンとも全体が依存関係のないメッシュループにより囲まれているために、プログラムに大幅な修正を加

えることなく容易に並列化処理を行うことが可能で、スカラマシンに関する限り、STEM2は並列化に向いた相性の良いコードであるということができる。

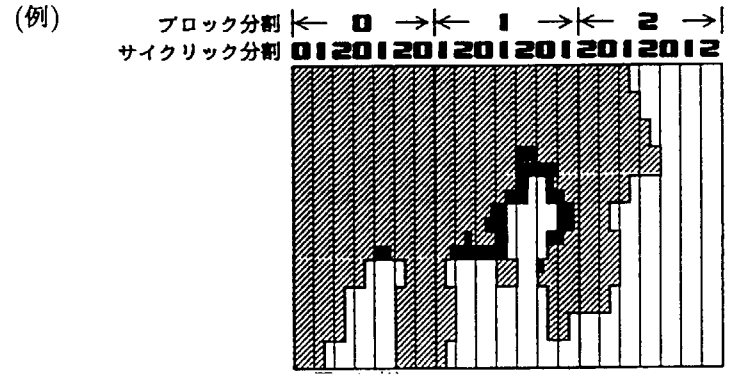
また、これまであまり触れなかったが、このSTEM2では、計算途中の任意のステップ毎に必要な情報を出力するための、出力データを算出するためのルーチン群(TOTAL_)が存在する。このTOTAL_では、鉛直成分のz方向に対して依存関係を持っているので、z軸で分割する場合は領域間のデータ参照が必要となる。しかしながら、分割軸をx軸とすることでここでも通信は不要となる。但し本計算では最初の1ステップ計算した後と最終ステップ後の合わせて2度、MPI標準のグループ通信サブルーチン：*MPI_REDUCE* でルートに集めた後、指標となるサンプリングデータを出力している。

§ 捕捉

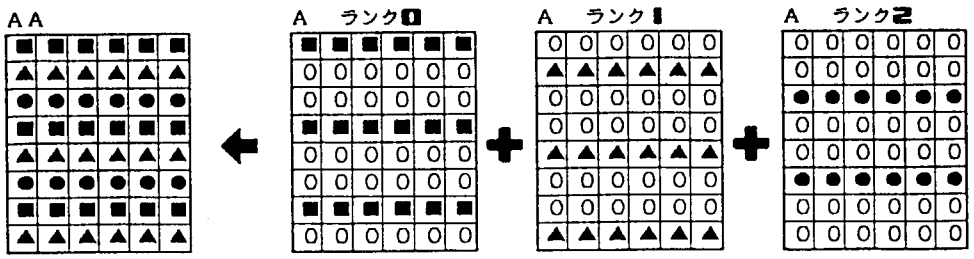
・サイクリック分割

通常、自然現象は滑らかに変化するので、個々の列では計算量にバラつきがあっても、サイクリック分割して合計すれば、プロセス間でのバラつきが相殺されロードバランスがほぼ均等になる。

(例)



・重ね合わせ



4.4 計算量の長時間評価

STEM2は化学反応計算部においてイタレーションループを含んでおり、化学汚染物質の濃度に応じて動的に計算量が変化する非線形性を含んだシミュレーションコードである。従って、我々が任意のシミュレーション時間(計算機のCPU時間ではなく、調べようとする現象の実時間スケール)に対してその計算に要した実行コストを評価する際、コンピュータの計算量がシミュレーション時間に対して、どの程度変化するのか理解することは結果を正しく評価する上で重要である。ここでは化学反応計算部の計算時間量の変化を調べた。結果をFig.7に示す。なお、輸送計算部についてはメッシュ数や物質の数などの初期条件、計算規模に依存するが、これらは固定された一定の条件下で計算を行っているので、線形に変化するものと考え、省略した。

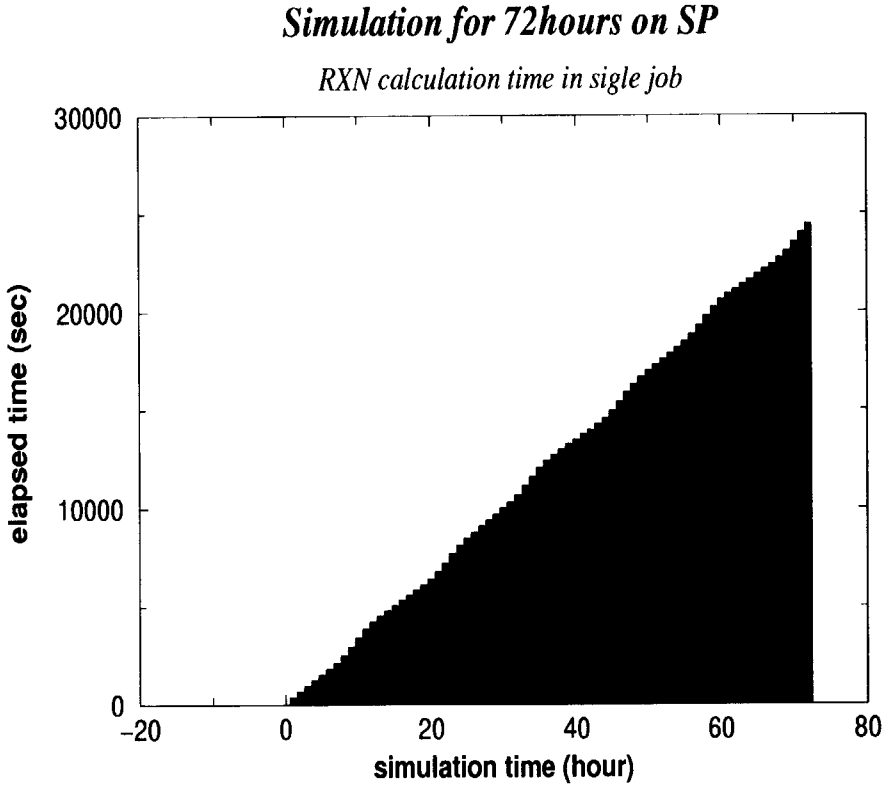


Fig. 7: 化学反応計算部(RXN)の長時間評価

大気中における化学汚染物質の７２時間にわたる輸送・拡散・反応・沈着過程のシミュレーションを行い、その化学反応計算部の計算量の時間発展を調べた。3日分の日変化を含んでいることもあり、周期性を持つような小さな揺らぎは見られるものの、ほぼシミュレーション時間に比例して増加していることがわかる。従って、この結果から今回用いたデータセットに対するSTEM2の計算量は、計算領域全体の平均値としては大きな変動は無いものとみなし、実行コストの評価を行うものとする。

4.5 並列化の効果

前述した方針により並列化を行い、スカラ計算機 SP を用いて並列計算を行った。ここでは最大34個のプロセッサで並列化した STEM2 の 72 時間シミュレーションの結果を紹介する。

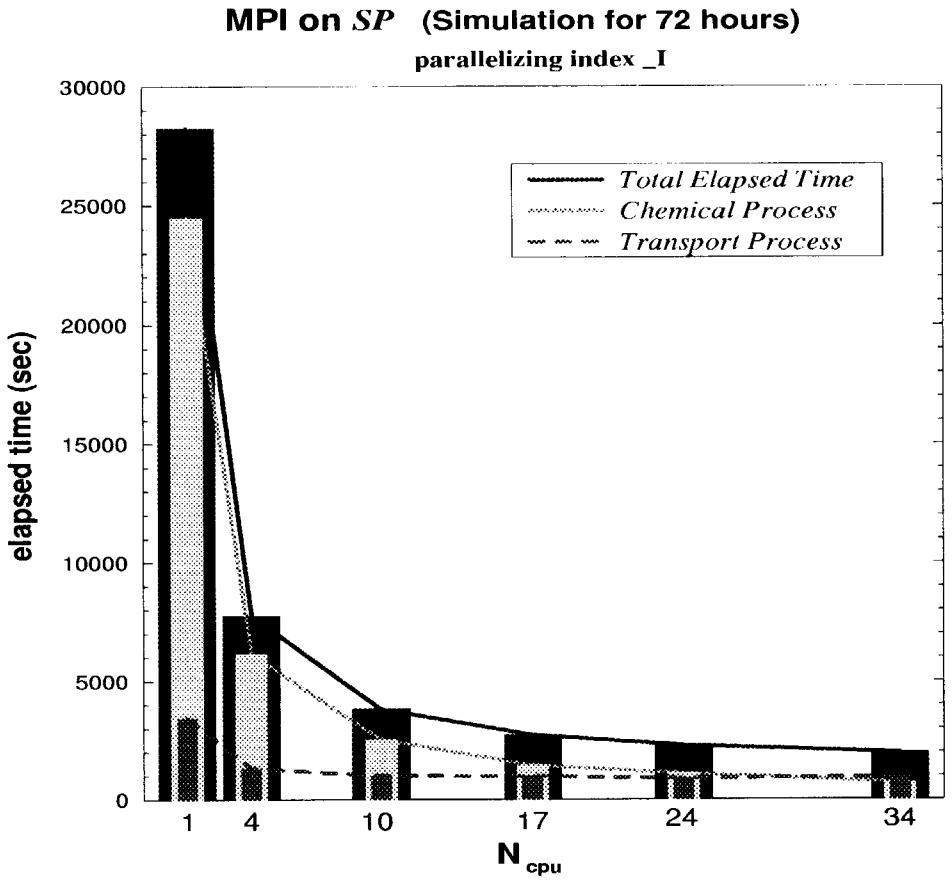


Fig. 8: SP による並列計算結果 (72 時間シミュレーション)

Table 3: 各ルーチンの実行コスト sec (ratio)

	1PE	4PE	10PE	17PE	24PE	34PE
⟨ Total ⟩	28234 (1.0)	7764 (3.6)	3829 (7.4)	2733 (10.3)	2288 (12.3)	2054 (13.7)
RXN	24471 (1.0)	6140 (4.0)	2538 (9.6)	1457 (16.8)	1094 (22.4)	736 (33.2)
HORX†	1107 (1.0)	744 (1.5)	753 (1.5)	835 (1.3)	779 (1.4)	876 (1.3)
HORY	909 (1.0)	227 (4.0)	97 (9.4)	57 (15.9)	44 (20.1)	31 (29.3)
VERTCL	1461 (1.0)	365 (4.0)	154 (9.5)	93 (15.7)	73 (20.0)	52 (28.1)
I/O	286	288	288	290	298	360

† HORX : include MPI communication

PE : Procsssing Element

Fig.8 は実行コストの測定結果を示したものである。最大のホットスポットである化学反応計算部は、通信が不要であることから非常に効率良く並列化されていることがわかる。一方、輸送計算部では 4 プロセッサを超えると飽和状況に陥っており、それ以上プロセッサ数を増やしても効果が見られない。全体の実行時間も輸送計算部の影響を受けて並列化による効果が鈍化している。また、Table3 に各ルーチン毎の実行コストと 1PE の計算時間に対する速度向上率 (Speed Up) を表した。化学反応計算部 RXN は 34PE で速度向上率が 33.2 倍に達し、1PE で全体の 86.7%を占めていた比率が 34PE では 35.8%にまで下がっている。輸送計算部の各ルーチンを比較すると、HORY 及び VERTCL ルーチンでは RXN 同様、通信によるオーバーヘッドがないので極めて高い並列効率が図られている。ところが、唯一通信が生じる HORX ルーチンでは速度向上率が他のルーチンと比べて非常に低く、1PE の結果では実行時間の全体に占める割合が 3.9%であったのに対し、34PE の結果では逆に全体の 42.6%にまで増加している。しかし実際に、HORX でも **4.3** 輸送計算部の並列化のところで説明したように、ルーチン内の計算はインデックスは変わるが並列処理を行っているので、ルーチン内の演算コスト自体は、HORY、VERTCL とほぼ同様にプロセッサ数に比例して短縮されている。34PE で実行時間が 876 秒かかっているものの、1PE での値 1107 秒よりは短縮されている (速度向上率が 1.0 以上である) のはそのためである。但し 4PE 以上になると、並列化効果よりも通信によるオーバーヘッドが上回り、その結果、HORX の計算コストは増加して輸送計算時間全体を飽和状況にしている原因となっている。

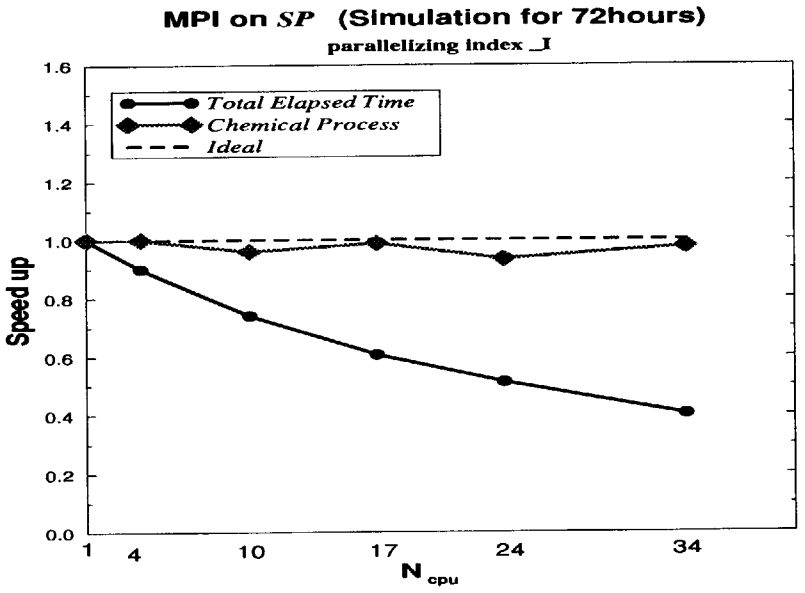


Fig. 9: PE 数に対する並列化効率

また、HORX 以外のオーバーヘッドの原因 (並列化効率が 1 にならない理由) を考えると、計算量のロードバランスの損失が挙げられる。プロセッサ数が多くなると、それだけ 1 プロセッサに

おける計算量は小さくなるため、サイクリック効果が減少しロードバランスが損なわれる。もともと計算粒度の小さい輸送計算部ではその影響が大きいと考えられる。do ループの回転数がプロセッサ数で割り切れないことによるロードバランスの損失も原因の一つである。

Fig.9 に並列計算の効率化という観点から、プロセッサ数に対する並列化効率の関係を表した。今回の結果を全実行時間に着目した場合、計算時間の短縮という視点から見れば、34PE と最も多くのプロセッサを使用した結果が計算時間を最小に抑えている。しかし計算効率の観点から見れば、並列化効率は 0.40 に過ぎない。もし仮に 10 倍程度のスピードアップで十分な評価が行えるのであれば、17PE でも計算は可能であり、この時並列化効率は 0.61 にまで上げることができる。また、化学反応計算部に着目すると、並列化効率が 34PE で 0.98 と極めて理想的な並列性能を示していることがわかる。

以下に、実行効率を評価するために用いた 2 つのパラメータの定義を記す。

P を計算に用いたプロセッサ数、T_p をプロセッサ P 個を用いた計算での実行時間とする。

スピードアップ

$$S_p = \frac{T_1}{T_p} \tag{3}$$

プロセッサ数を増加させた時、何倍速く計算できたかの目安。理想的な場合は P (プロセッサ数に正比例して速度が増加) になる。

並列化効率

$$E_p = \frac{S_p}{P} = \frac{T_1}{PT_p} \tag{4}$$

スピードアップをプロセッサ数で規格化したもの。

理想的な場合は 1 になる。

5. VPP300 によるベクトル化

前述の **3.** ソースコード分析の結果から、VPP の実行コスト分布で化学反応計算部の占める割合が全体の計算時間の 90%以上と大部分を占めている。しかしながら、ベクトル化率をみると化学反応計算部全体で 13%程度とかなり低いことがわかる。これでは最も計算時間が多くかかるホットスポットの部分で、ほとんどの演算がスカラ実行されていることになり、ベクトル計算機の持つ演算能力が十分に引き出されていないと推察できる。従って、この化学反応計算部のベクトル化率を大きく引き上げることにより、飛躍的な計算性能の向上を実現できる可能性がある。また、ベクトル化率を上げるということは演算の並列性を高めることにも繋がり、並列化による並列効率の観点からも重要な点である。

5.1 化学反応計算部のベクトル化

5.1.1 ルーチン構成

化学反応計算部のルーチン構成を Fig.10 に示す。

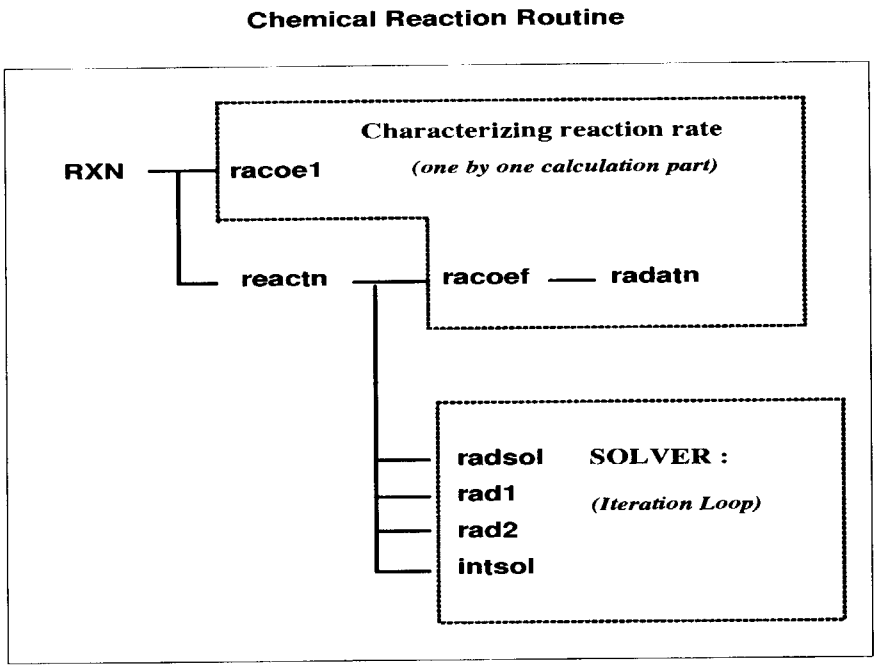


Fig. 10: 化学反応計算部のルーチン構成

この化学反応計算部は、線形微分方程式を計算する上で必要となる各項の係数 (反応定数) を求める部分 (RACOE1、RACOEF、RADATN) と、線形微分方程式の解を求めるソルバー部 (RADSOL、RAD1、RAD2、INTSOL) から成り立っている。反応定数の計算部分は逐次計算で全くループが存在しないのに対し、ソルバー部では方程式の解を求めるためのイタレーションループが含まれている。そしてこれらの各ルーチンは全体が RXN の下で 3 次元メッシュによるループに囲ま

れた構造になっている。

5.1.2 ループ構成

化学反応計算部のループ構成を Fig.11 に示す。

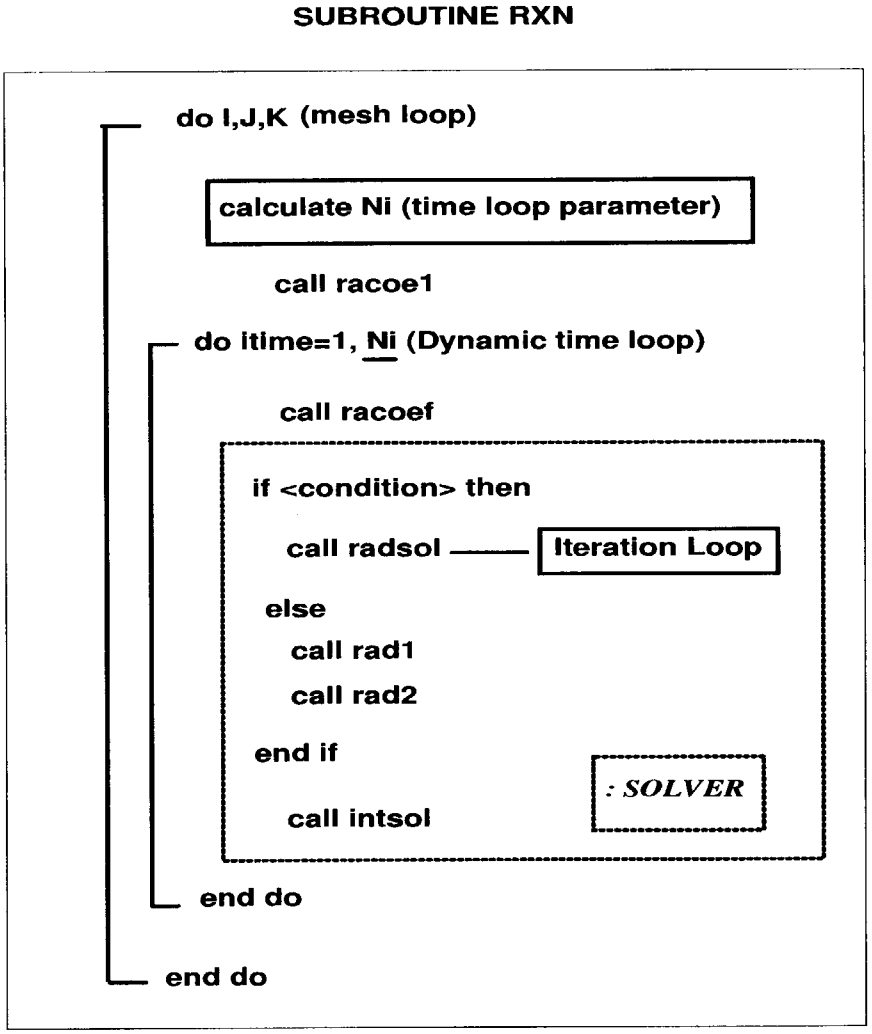


Fig. 11: 化学反応計算部のループ構成

化学反応計算部は、全体が３次元メッシュループから成り、メッシュループ内には、メッシュ毎に回転数 (Ni ; Fig.11) が動的に変化する時間ループが存在する。そして時間ループ内には比較的計算コストが高く、計算の中心であるソルバー部が構築されており、回帰的演算は含まないものの、radsol においては非線形方程式を線形化したために生じるイタレーションループを含んでいる。従って、この化学反応計算部を３次元メッシュループでベクトル実行するためには、動的時間ループ及びイタレーションループの存在が大きな障害であり、何らかの工夫が必要である。

5.1.3 ベクトル化方針

前述のルーチン構造及びループ構造の調査を踏まえて、ベクトル計算機の演算性能をを十分に引き出すために、ベクトル化の方針としては最も長くベクトル長が取れる3次元メッシュループ(i,j,k)により全体のベクトルチューニングを行う。但し、3次元メッシュループの中には、メッシュ毎に動的に決まる回転数を持つ時間ループと、ソルバー部のイタレーションループの2つの大きなループが含まれていることも判っているので、具体的には以下に示す5つの方針を元にベクトルチューニングを行う。

1) 時間ループを最外にしたループ構造に変換する。
2) if文のベクトル処理(真率による分岐を計る。)
3) 変数、配列の次元上げ
4) 細かなループの展開
5) ルーチンの上位展開

(1) 時間ループを最外にしたループ構造への変換

時間ループはメッシュループ毎に回転数が決まる。これらの回転数は主計算部とは別に、全メッシュ分を事前に計算することができるので、次のような手順でループ構造を変更する。
i) 予めメッシュ毎の回転数mend(i,j,k)を計算する。ii) メッシュの内側にある時間ループを最外へ出す。iii) 時間ループは、事前に求めたmend(i,j,k)の中の最大値($max = 45$)を仮の回転数としたダミーループとする。iv) ループの中で、if文による条件分岐により、ダミーループのdo変数($itime$)の値が、mend(i,j,k)に達した場合、即座にループ内の計算を飛ばすようプログラムする。これによってメッシュループ計算のベクトル化が可能となる。

この方法は、ソルバー部のイタレーションループの場合も有効で、ループ内の計算において、反復終了条件が成り立った時点でフラグを立て、時間ループ同様、条件分岐によりループ内の計算を飛ばすようにする。

チューニング後の時間ループ：

```
        do itime = 1, 45 (mend_max) ... dammmy loop
        do k = 1, iz
        do j = 1, iy
        do i = 1, ix
              if ( itime.gt.mend(i,j,k) ) goto 100
                 ♯ mend(i,j,k) : メッシュ毎の回転数
               ⟨calculation⟩
                  .....
                  .....
  100   continue
        end do
```

チューニング後のイタレーションループ：

```
        do ite = 1, 100 (mend_max) ... iteration loop
        do k = 1, iz
        do j = 1, iy
        do i = 1, ix
              if ( if_flag(i,j,k).eq.1 ) goto 200
                 ♯ if_flag(i,j,k) : 収束条件判定のためのフラッグ
               ⟨calculation⟩
                  .....
                  .....
              if ( 収束条件 = yes ) then
                 if_flag(i,j,k) = 1  ; 収束条件が成り立つ場合は 1 を代入
              end if
  200   continue
        end do
```

この指針によりループ構造を変更すると、化学反応計算部のループ構造は次のようなイメージに再編成される。

チューニング後の化学反応計算部：

```
do i,j,k  ( Vectorized )
   前処理部 (時間ループの回転数計算 etc.)
end do
do itime                              ... (時間ループ)
   do ite                             ... (イタレーションループ)
      do i,j,k  ( Vectorized )
         反復計算部
      end do
   end do
   do i,j,k  ( Vectorized )
      濃度係数項の計算部
   end do
end do
do i,j,k  ( Vectorized )
   後処理部
end do
```

(2) *if*文のベクトル処理 (真率による分岐)

前述の (1) において、時間ループ及びソルバー部のイタレーションループをメッシュループの外に出すために、条件分岐により内部の演算を必要な場合のみ行うように改めた。この場合、条件分岐のための *if* 文についてもベクトル処理される。

この *if* 文のベクトル処理には mask 方式 (*if* 文の演算を条件に関係なく一通り行い、条件を満たす部分のみ集める方式。真率が高い場合に有効。) 、list 方式 (条件にあう index の list を作り、条件を満たす部分のみ演算する。真率が低い場合に有効。) の二通りの方法が考えられる。

反復計算の条件分岐の真率を測定してみると、反復を重ねていく (時間が進む) につれ真率が大きく下がってくることがわかった。真率が高いにもかかわらず list 処理を行えば list を作るオーバーヘッドが大きくなり、真率が低いにもかかわらず mask 処理を行えば余計な演算が多くなり、どちらの場合も性能低下に繋がる要因となり得る。ループが進むにつれ真率が変わるこのようなケースでは、その時の真率により分岐の処理方式を変更することは、無駄な演算を省く上で非常に効果的な手段と言える。そこで、このケースでは真率 10%を臨界値として場合分けを行い、10%

以上では mask 処理、10%未満では list 処理というように２通りの処理を併用して用いた。

なお、*if*文の mask 処理、list 処理はプログラム中に次の最適化制御行 (VPP 固有) を挿入することで、セミオートマチックにベクトル化を行う。

```
!ocl vct ( mask )      ... mask 処理
!ocl vct ( list )      ... list 処理
```

(3) 変数、配列の次元上げ

前述の計算を成り立たせるためには、全体を囲んでいるメッシュループを時間ループ、及びソルバー部の手前で切る必要がある。そのためには、変数や配列の次元上げを行い、3 次元メッシュすべての情報を持たせなければならない。また、3 次元メッシュループでベクトル化を行う場合、ループ内の演算の定義部 (左辺) にメッシュループの index(i,j,k) の次元を持つ配列が来る必要がある。この化学反応計算部では主要な演算の多くが変数、もしくは 1 次元配列である場合が多く、それらをすべて次元上げしなければならない。

```
mend = DT/DTC + 0.1
CA(1) = KR(82) * SS1(3)
```

⇓ (変更)

```
mend(i,j,k) = DT/DTC + 0.1   ... メッシュ分の情報を格納
CA(i,j,k,1) = KR(i,j,k,82) * SS1(i,j,k,3)
                  ... ベクトル化のための次元上げ
```

(4) 細かなループの展開

3次元メッシュループでベクトル化する場合、メッシュループを最内ループにする必要がある。そのためメッシュループの内側に複数ある細かなループを次のように展開する。

```
do L = 1, 34
   SS1(L) = ...
end do
```

⇓ (変更)

```
SS1(i,j,k,1) = ...
SS1(i,j,k,2) = ...          \ ループを展開して次元上げ
SS1(i,j,k,3) = ...
```

(5) ルーチンの上位展開

サブルーチンの呼び込み (call) 文はベクトル化の対象とはならない。従って、ループ内に存在する call 文はすべて上位展開する必要がある。この化学反応計算部では、最内となる3次元メッシュループ内に、以下に示す5つのサブルーチンを呼び込んでいる。これらすべてのサブルーチンの上位展開は VPP300 のコンパイルオプション (−Ne) により自動実行することが可能である。

RACOE1, RACOEF, RADSOL, RAD1(RAD2 含), INTSOL ... 計5ルーチン

5.2 輸送計算部のベクトル化

飛躍的に計算性能を向上させるためには、3.2 で示したオリジナルの実行時間コストの解析で、全実行時間の数パーセントに満たないルーチンについても考慮する必要がある。ここでは、ベクトル化の対象として、化学反応計算部に次いでコストの大きい輸送計算部を取り上げる。

5.2.1 ルーチン構成

輸送計算部ではx方向、y方向、z方向のそれぞれの成分に分けられた微分方程式の解を求める。各方向別にHORX、HORY、VERTCL、という名のサブルーチンが用意され、各成分の方程式は同様の型であることから、計算方法としては同じ手法をとっている。以下、輸送計算部のルーチン構成を示す。

⟨ 輸送計算部のルーチン構成 ⟩

サブルーチン：
HORX(x 方向), HORY(y 方向), VERTCL(z 方向)

- 係数行列の生成 (hcoex1, hcoey1, vcoef1)
- LU 分解 (decomp, trid)
- 右辺ベクトルの生成 (hcoex2, hcoey2, vcoef2)
- 線形方程式の解法 (trid1, trid)

この輸送計算部では3重対角行列となる係数行列からなる線形方程式が生成され、LU 分解によりその解を求めている。x方向、y方向、z方向のそれぞれのルーチンでは、化学反応計算部と同様に全体が3次元メッシュループで構成されている。

5.2.2 ループ構成

輸送計算部のループ構成は以下に示すとおりである。

```
do i,j or j,k or k,i
  ・係数行列の生成
  ・LU 分解
    do l = 1, 34
      ・右辺ベクトルの生成
      ・線形方程式の解法
    end do
  ・境界部の処理
end do
```

この輸送計算部のループ構成は全体が、x 方向の場合は y 及び z 方向のメッシュの２重ループとなり、対象となる方向とは別の２方向のメッシュで囲まれている。そして２重ループの最内に HORX では x 方向、HORY では y 方向、VERTCL では z 方向のループが存在し、３４種の反応物質に対する線形方程式を直接解法により解く。また、解法部には依存関係のある回帰演算を含んだループが存在する。従って、このままでは回帰演算が存在するためにベクトル処理ができない。

5.2.3 ベクトル化方針

輸送計算部の全計算は、最外の２重ループでは演算に回帰的な依存関係はなく、独立して計算が行えることから並列処理が可能である。従って、化学反応計算部の場合と同様に依存関係のある最内ループを、最も外側へと持っていくことでベクトル化が可能となる。また、並列分割する際は、最外は回帰的な依存関係があるために分割軸としては２重ループの外側ループで行う。尚、化学計算部同様、ルーチンの上位展開、次元上げ、ループを細かく切る必要がある等、コードに大幅な修正が必要である。

輸送計算部のベクトル化チューニングのイメージを下記に示す。

チューニング後の輸送計算部：

```
do i
    do j, k ( Vectorized )
        前処理部 ( 係数行列の生成、LU 分解 )
    end do
end do
do l = 1, 34
    do i
        do j, k ( Vectorized )
            右辺ベクトルの生成
            線形方程式の解法
        end do
    end do
end do
do j, k ( Vectorized )
            境界部の処理
end do
```

5.3 ベクトル化による性能評価

前述したベクトル化方針に従ってチューニングを行い、VPP300 上で実行時間コスト、及びVU ビジー率を単一プロセッサで実測した。4 時間シミュレーションの結果でオリジナルとの性能比較を行ったところ、実行時間の速度向上率が 9.36 倍となり、13%程度であった VU ビジー率も STEM2 全体で 82%、とりわけ化学計算部では 99%にまで上昇した。また、Table 4 に示したチューニング版各ルーチンの CPU 時間を見ると、1.7%でしかなかったオリジナルの I/O 時間が相対的に 15.4% にまで膨れ上がっている。

1) 実行時間コスト

オリジナル	2870.3 sec
チューニング版	306.6 sec

速度向上率：9.36 倍

2) VU ビジー率

オリジナル

- STEM2 全体 (MAIN) ... 13.13 %
- 化学計算部 (RXN) ... 12.98 %
- 輸送計算部 (HORX,HORY,VERTCL) ... 16.48 %

チューニング版

- STEM2 全体 (MAIN) ... 82.05 %
- 化学計算部 (RXN) ... 99.05 %
- 輸送計算部 (HORX,HORY,VERTCL) ... 80.65 %

Table 4: チューニング版:各ルーチンの CPU time (4 hours calculation)

	ルーチン名	CPU(sec)	(%)
VPP300	RXN	221.4	72.2
Total CPU time:	HORX	4.2	1.4
312.0 sec	HORY	6.2	2.0
	VERTCL	27.8	9.1
	I/O	47.1	15.4
	(INPUT1)	(31.7)	(10.3)

6. VPP300 によるベクトル並列

6.1 並列化方針

ここではベクトルチューニングされたコードに対してさらに並列化を行う。並列化方法は基本的にスカラ並列と同じで、領域分割はサイクリック分割、通信は MPI を用いている。但し、一つだけスカラ並列と違う点は分割軸のインデックスが異なる点である。ベクトルチューニングの効果を最大限に活かすために、分割軸をメッシュ数の少ない z 軸、もしくは y 軸での分割を考える。これはベクトル長を少しでも大きく取るために、最もメッシュ数の多い x 軸方向のインデックス i をベクトル化に充てるためである。そのためここでは、プログラム全体の分割軸として z 軸方向のインデックス k 、及び y 軸方向のインデックス j の 2 つのケースそれぞれについて、並列化を行い比較することにする。

スカラ並列でも述べたように、輸送計算部では 3 つのルーチンすべてを同一のインデックスで分割することはできない。従って、全体を k で分割する場合は輸送計算部のルーチン VERTCL が、全体を j で分割する場合は輸送計算部のルーチン HORY がそれぞれ問題となり、ルーチン内の演算に入る前に通信が必要である。

それでは、両方のケースの違いが輸送計算部 VERTCL、HORY ルーチンの並列処理にあるので、具体的に両ケースの方針について述べる。まず全体を k で分割する場合では、1) 輸送計算部 VERTCL の入口で通信を行う。2)VERTCL 内の演算は k で分割することができないので、改めて VERTCL 内のみインデックスを j に換えて分割し、並列処理を行う。3) 再度、出口で通信を行う。この一連の並列化過程はスカラ並列の手法と全く同じである。ところが、次に全体を j で分割する場合は、HORY 内の演算が j で分割することができないので、最初は k 分割する場合と同様に、1)HORY の入口で通信を行う。しかし次のステップで HORY 内の演算は、2) 並列化せずそのまま冗長実行させる。3) 並列処理を行わないので出口で通信は不要となる。HORY を並列計算させない根拠は、VERTCL に比べ計算量が小さい (計算コストの比率; VERTCL 4.3% に対し HORY 1.6%、 Table 1 参照) ために、ベクトル化を犠牲にしてまで並列化を行っても効果がプラスとして現れないことが理由である。

6.2 並列化による性能評価

6.2.1 index j による分割

index j で分割した場合の並列計算結果を Fig.12 及び Table 5 に示す。
この場合、全体を y 軸のインデックス j で分割し、輸送計算部 HORY では冗長実行を行った。

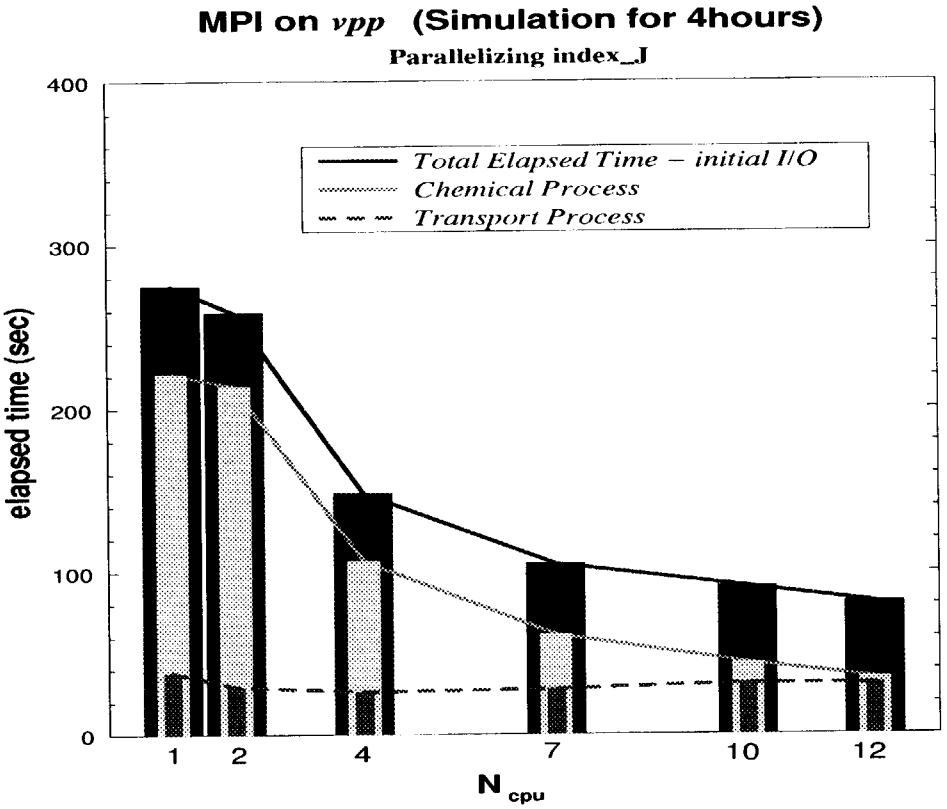


Fig. 12: VPP300 による並列計算結果 (j 分割)

Table 5: 各ルーチンの実行コスト sec (ratio)

	1PE	2PE	4PE	7PE	10PE	12PE
⟨ Total ⟩	306.6 (1.0)	287.8 (1.07)	178.9 (1.71)	137.5 (2.23)	125.5 (2.44)	118.4 (2.59)
RXN	221.4 (1.0)	213.9 (1.04)	106.6 (2.08)	61.7 (3.59)	44.1 (5.02)	34.8 (6.36)
HORX	4.2 (1.0)	3.1 (1.35)	2.3 (1.83)	1.6 (2.63)	1.4 (3.00)	1.3 (3.23)
HORY†	6.2 (1.0)	11.4 (0.54)	16.5 (0.38)	22.1 (0.28)	26.7 (0.23)	27.1 (0.23)
VERTCL	27.8 (1.0)	15.2 (1.83)	7.6 (3.66)	4.5 (6.18)	3.2 (8.69)	2.6 (10.69)
I/O	47.1	44.3	45.9	47.7	50.2	52.6
(input1)	(31.7)	(29.0)	(30.8)	(32.4)	(34.6)	(37.0)

† HORY : include MPI communication

ここで示した 1PE の計算結果はベクトルチューニング版の結果であるが、その 1PE のシングル結果に対し 2PE で並列化した結果を見ると非常に効率が悪い。これは並列化チューニングすることにより、化学反応計算部のベクトル長が短くなったことが原因とみられる。つまり、シングル計算では 3 次元メッシュループ (index_i,j,k) すべてを使ってベクトル実行していたのに対し、並列版では index_j のループを分割軸として使ったために、ベクトル実行ループ (j ループ) が一つ減

り、その分ベクトル長が短くなったことが原因であると考えられる。その証拠に Table 5 の中の速度向上率を、2PE の値を 1.0 と置き換えて 4PE 以上の並列性能を考えると、例えば RXN ではスカラ並列の結果と同様に、非常に高い並列性能を実現している。また輸送計算部 HORY ではプロセッサ数を増すほど実行コストもまた増加している。HORY 内では、そもそも冗長実行しているのでスピードアップは 1 を超えることはない。加えてサブルーチンが呼ばれる度に HORY の入口で、それまで領域分割していたデータを収集するための通信が実行される。HORY の実行コストが増加していくのは、この通信によるオーバーヘッドが原因である。

以上、index_j 分割による並列性能の測定結果は、ベクトルチューニング版 1PE に対して、12 プロセッサ使用で速度向上率が 2.59 倍であり、並列化効率は 0.22 であった。オリジナル比でみると、ベクトルチューニング版をさらに並列化実行した総合的な速度向上率は 12 プロセッサ使用で 24.2 倍となった。

・ベクトル並列計算による性能向上 (オリジナル比)

速度向上率：24.2 倍

オリジナル	2870.3 sec
12 PE (y 軸分割)	118.4 sec

6.2.2 index k による分割

index k で分割した場合の並列計算結果を Table 6 及び Fig.13 に示す。このケースでは、全体を z 軸のインデックス k で分割し、輸送計算部 VERTCL 内のみ特別に y 軸のインデックス j で分割を行った。また、並列化の効率を比較するために、プロセッサ数に対するスピードアップの値を j 及び k 分割の両ケースについて Fig.14 にプロットした。

Table 6: 各ルーチンの実行コスト sec (ratio)

	1PE	2PE	4PE	6PE	11PE
⟨ Total ⟩	306.6 (1.0)	230.5 (1.33)	156.5 (1.96)	142.3 (2.15)	135.0 (2.27)
RXN	221.4 (1.0)	151.7 (1.46)	75.6 (2.93)	50.3 (4.40)	25.4 (8.72)
HORX	4.2 (1.0)	2.8 (1.50)	1.4 (2.80)	1.0 (4.20)	0.5 (8.40)
HORY	6.2 (1.0)	3.2 (1.94)	1.6 (3.88)	1.1 (5.64)	0.6 (10.30)
VERTCL†	27.8 (1.0)	27.3 (1.02)	31.1 (0.89)	39.4 (0.71)	55.0 (0.51)
I/O	47.1	45.6	46.7	50.4	53.4
(input1)	(31.7)	(29.5)	(30.1)	(33.3)	(35.8)

† VERTCL : include MPI communication

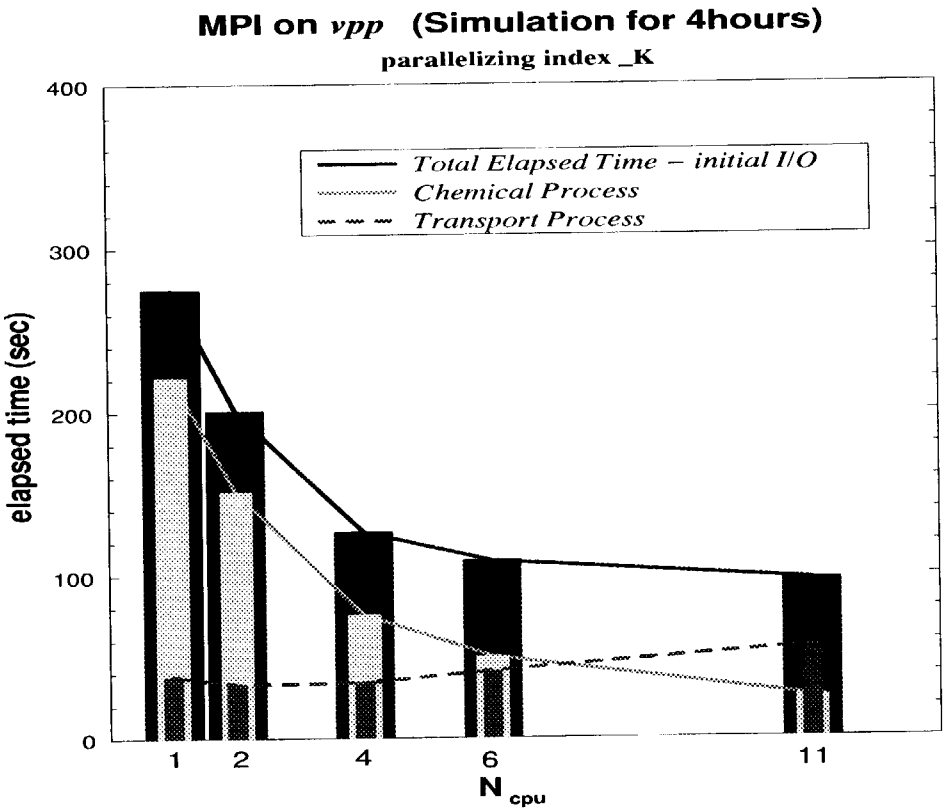


Fig. 13: VPP300 による並列計算結果 (k 分割)

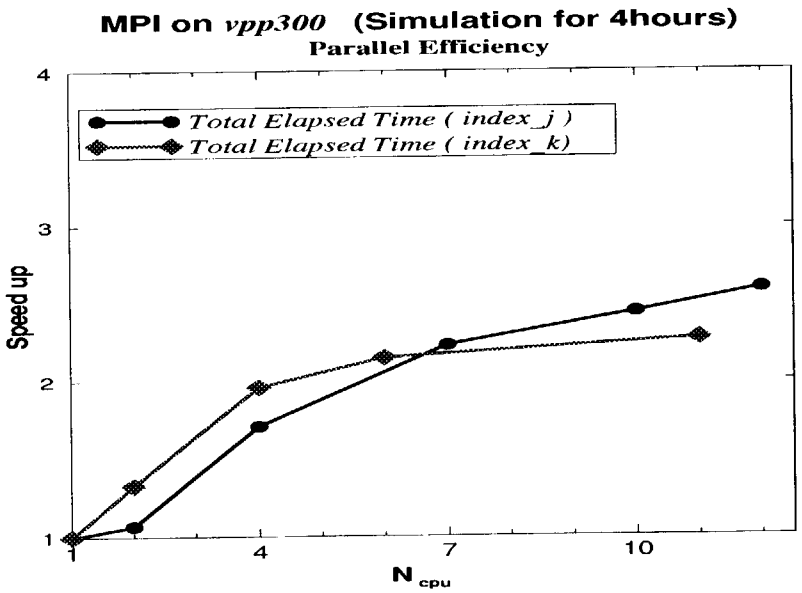


Fig. 14: プロセッサ数に対するスピードアップ (j & k 分割)

インデックス j の計算結果と同様、シングル計算と 2PE の結果を比較するとベクトル長が短くなるために並列化効率が低下してしまう。但し、メッシュの回転数が j の場合 47 なのに比べれば k

は11と小さいことから、幾分かベクトル効率の悪化を緩和することができる。その結果、k分割は6プロセッサあたりまではj分割に比べ効率が良い。ところが、Fig.14を見ればわかるように、11プロセッサにするとむしろj分割より効率が下がっている。これはz軸を11プロセッサで分割すると、メッシュ数が11しかないために実質上、1PEに1メッシュを割り当てたブロック分割(サイクリックバランスが効かない)となり、ロードバランスが低下することに起因するオーバーヘッドの増加を招く。また、特に輸送計算部VERTCLでは内部演算を改めてjで分割を行っており、その分計算量は少なくできるものの、入口と出口で通信が2度必要となり、さらに47メッシュを11分割していることによるロードバランスの低下がオーバーヘッドを大きくしている原因と考えられる。

以上、index_k分割による並列性能の測定結果は、ベクトルチューニング版1PEに対して、11プロセッサ使用で速度向上率が2.27倍であり、並列化効率は0.21であった。オリジナル比でみると、ベクトルチューニング版をさらに並列化実行した総合的な速度向上率は11プロセッサ使用で21.7倍となった。

・ベクトル並列計算による性能向上(オリジナル比)

速度向上率：21.3倍

オリジナル	2870.3 sec
11 PE (z軸分割)	135.0 sec

6.3 スカラ並列との比較

スカラマシンSPとの比較を同じ4時間シミュレーションの結果で行った。結果をFig.15に示す。ここで、VPP300のデータはindex_j分割によるベクトル並列計算結果である。また、Fig.15はトータルの経過時間から初期化のI/O(input1)時間を差し引いた値であり、1PEのシングル計算値はオリジナルの計算結果である。

この結果を見ると、SPの並列性能はプロセッサ数に比例する形で計算コストが短縮されている。一方、SPが比較的滑らかに推移しているのに対しVPPは、オリジナルのシングル実行コストがSPの約2倍であったのに、2PEでは速度向上率が１０倍を超え、逆にSPの実行コストの半分以下にまで劇的に下がっていることがわかる。このVPPの高速化は**5.3**ベクトル化による性能評価で示したようにベクトル化チューニングの効果が極めて大きい。

さらに、Fig.15の実行コストの値に着目すると、12並列の結果は、SPが175.7sec、VPPは81.4secで、VPPはSPの半分以下である。ところが、SPも34並列で計算を行うと103.5secとなり、その差は22.1sec(/simulation for 4hours)にまで短縮される。また、速度向上率は12並列の場合、SPでは8.1倍、VPPではベクトル化、並列化を合わせて34.6倍となる。ここで注意しなければならないのは、VPPの34.6倍という値は初期のI/O時間を除いた実行コストで評価したもので、

6.2.1 で示した速度向上率 24.2 倍とは大きく異なることである。これは VPP の 4 時間シミュレーションの結果では、計算量が小さいこともあって、相対的に初期の I/O 時間 (37.0sec/12PE) が全実行コスト (118.4sec/12PE) に占める割合が 31.3%にもなるためである。従って、実質的な速度向上率を評価する場合、初期の I/O 時間を除いた実行コストで算定すべきであると考える。なお、4.5 で示した SP の並列計算評価は 72 時間シミュレーションの結果で行っているので問題はなく、初期の I/O 時間を考慮したとしても誤差の範囲となり、速度向上率は 34 並列で 13.7 倍と変わらない。

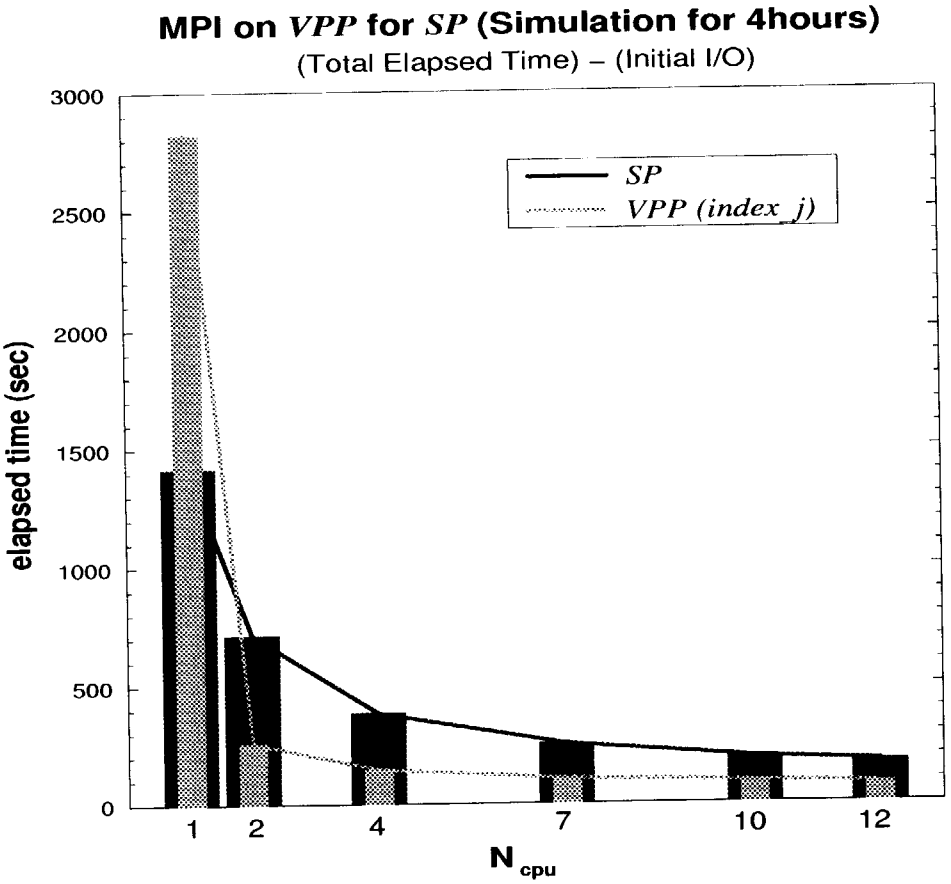


Fig. 15: SP と VPP300 の並列計算結果の比較

・4 時間シミュレーションにおける 12 並列の速度向上率 (1PE 比)

SP	8.1 倍
VPP300 (y 軸分割)	34.6 倍

7. まとめ

STEM2は大気汚染物質をはじめとする化学種について100以上の素反応を含んだ詳細な化学反応計算が特徴である。実際、オリジナルソースコードの実行コストを解析しても、80%を超える圧倒的大部分がこの化学反応計算部に費やされており、いかにしてこの部分の計算コストを下げるかということが、最大の課題であるという認識のもとSTEM2の高速化に取り組んだ。

スカラマシンSP及びベクトルマシンVPP300の2機種を用いて、1)SPによるスカラ並列、2)VPPによるベクトル化チューニング、3)VPPによるベクトル並列という3つの過程に分けて高速化に取り組んだ。その結果、SPでは34プロセッサの使用で13.7倍の性能向上を示し、また、VPPではベクトル化でオリジナルの9.4倍、さらにベクトルチューニング版を並列化することにより、12並列で2.6倍の高速化を実現した。オリジナル版の経過時間と12PEのベクトル並列計算の経過時間を比べると、VPPでは全経過時間で24.1倍、初期化のI/O時間を除いた実質的な経過時間で34.6倍の高速化が図れた。

以上の結果を踏まえて化学反応計算の高速化を考えると、STEM2の化学反応計算は計算領域の各メッシュ毎に独立しており、サイクリック分割等により各プロセッサ間のロードバランスをうまくとることができれば、領域分割による並列処理を非常に効果的に行うことができる。また、ベクトル化についてもメッシュによる独立性を活かしたプログラムの構造を作り上げることが重要で、そのためにはループ構成として最内にメッシュループを置くことが最も大切なポイントである。チューニングの結果、ベクトル化率が99%を達成できることからすると、非常に効果的にベクトル化が可能であると言うことができる。

謝辞

九州大学大学院 総合理工学研究科の大学院生、Seok-Jae Kang氏には、STEM2のソースコード及び入力データ等を提供していただいたことを感謝します。また、(株)富士通の鈴木信太郎、南一生、両氏には、VPPにおけるベクトル化・並列化チューニングに関して適切な助言をいただきましたことを感謝致します。

参考文献

[1] STEM-II User's Guide, Air Pollution Modeling Group, Department of Chemical & Biochemical Engineering, Univ. of Iowa, (1990).

[2] Carmichael G.R., Peters L.K., and Kitada T., Atmos.Envir., 20, p173-188, (1986).

[3] Atkinson R., Atmos.Envir., 24A, p1-41, (1990).

[4] Kang S., Ueda H., 化学工学論文集, 23, **6**, p850-860, (1997).

[5] Kang S., Weiming S. and Ueda H., Atmos.Envir., (submitted).

[6] Carmichael G.R., Peters L.K., Atmos.Envir., 18, p937-951, (1984).

[7] 植田洋匡,"酸性雨のシミュレーション", シミュレーション 8, **3**, (1988).

[8] Brooks A.N. and Hughes T.J.R., Meth.Appl.Mech.Engrg., 32, p199-259, (1982).

[9] Yaneko N.N., "The Method of Fractional Pitches in Solving Many-Dimensional Problems of Mathmetical Physics", NAUKA, The Sibirian Branch Novosibrisk, (1967).

[10] Leonard K. Peters, Carl M.Berkowitz, Gregory R.Carmichel et al., Atmos. Envir., 29, **2**, p189-222, (1995).

[11] 市川陽一, 速水 洋, 電力中央研究所報告, T96044, (1997).

[12] IBM : 並列プログラミング虎の巻 MPI 版 (1997).

[13] 折居茂夫, 太田敏郎, "分子動力学コードの段階的並列化手法", JAERI-DATA/Code, (1996).

国際単位系 (SI) と換算表

表 1　SI基本単位および補助単位

量	名称	記号
長さ	メートル	m
質量	キログラム	kg
時間	秒	s
電流	アンペア	A
熱力学温度	ケルビン	K
物質量	モル	mol
光度	カンデラ	cd
平面角	ラジアン	rad
立体角	ステラジアン	sr

表 2　SIと併用される単位

名称	記号
分，時，日	min，h，d
度，分，秒	°，′，″
リットル	l，L
トン	t
電子ボルト	eV
原子質量単位	u

1 eV=1.60218×10^{-19} J

1 u=1.66054×10^{-27} kg

表 5　SI接頭語

倍数	接頭語	記号
10^{18}	エクサ	E
10^{15}	ペタ	P
10^{12}	テラ	T
10^{9}	ギガ	G
10^{6}	メガ	M
10^{3}	キロ	k
10^{2}	ヘクト	h
10^{1}	デカ	da
10^{-1}	デシ	d
10^{-2}	センチ	c
10^{-3}	ミリ	m
10^{-6}	マイクロ	μ
10^{-9}	ナノ	n
10^{-12}	ピコ	p
10^{-15}	フェムト	f
10^{-18}	アト	a

表 3　固有の名称をもつSI組立単位

量	名称	記号	他のSI単位による表現
周波数	ヘルツ	Hz	s^{-1}
力	ニュートン	N	$m\cdot kg/s^2$
圧力，応力	パスカル	Pa	N/m^2
エネルギー，仕事，熱量	ジュール	J	N·m
工率，放射束	ワット	W	J/s
電気量，電荷	クーロン	C	A·s
電位，電圧，起電力	ボルト	V	W/A
静電容量	ファラド	F	C/V
電気抵抗	オーム	Ω	V/A
コンダクタンス	ジーメンス	S	A/V
磁束	ウェーバ	Wb	V·s
磁束密度	テスラ	T	Wb/m^2
インダクタンス	ヘンリー	H	Wb/A
セルシウス温度	セルシウス度	℃	
光束	ルーメン	lm	cd·sr
照度	ルクス	lx	lm/m^2
放射能	ベクレル	Bq	s^{-1}
吸収線量	グレイ	Gy	J/kg
線量当量	シーベルト	Sv	J/kg

表 4　SIと共に暫定的に維持される単位

名称	記号
オングストローム	Å
バーン	b
バール	bar
ガル	Gal
キュリー	Ci
レントゲン	R
ラド	rad
レム	rem

1 Å= 0.1 nm=10^{-10} m

1 b=100 fm^2=10^{-28} m^2

1 bar=0.1 MPa=10^5 Pa

1 Gal=1 cm/s^2=10^{-2} m/s^2

1 Ci=3.7×10^{10} Bq

1 R=2.58×10^{-4} C/kg

1 rad = 1 cGy=10^{-2} Gy

1 rem = 1 cSv=10^{-2} Sv

（注）

1. 表1－5は「国際単位系」第5版，国際度量衡局 1985年刊行による。ただし，1 eVおよび1 uの値はCODATAの1986年推奨値によった。
2. 表4には海里，ノット，アール，ヘクタールも含まれているが日常の単位なのでここでは省略した。
3. barは，JISでは流体の圧力を表わす場合に限り表2のカテゴリーに分類されている。
4. EC閣僚理事会指令ではbar，barnおよび「血圧の単位」mmHgを表2のカテゴリーに入れている。

換　算　表

力	N(=10^5dyn)	kgf	lbf
	1	0.101972	0.224809
	9.80665	1	2.20462
	4.44822	0.453592	1

粘度　1 Pa·s(N·s/m^2)=10 P(ポアズ)(g/(cm·s))

動粘度　1 m^2/s=10^4St(ストークス)(cm^2/s)

圧力	MPa(=10 bar)	kgf/cm^2	atm	mmHg(Torr)	lbf/in^2(psi)
	1	10.1972	9.86923	7.50062×10^3	145.038
	0.0980665	1	0.967841	735.559	14.2233
	0.101325	1.03323	1	760	14.6959
	1.33322×10^{-4}	1.35951×10^{-3}	1.31579×10^{-3}	1	1.93368×10^{-2}
	6.89476×10^{-3}	7.03070×10^{-2}	6.80460×10^{-2}	51.7149	1

エネルギー・仕事・熱量	J(=10^7erg)	kgf·m	kW·h	cal(計量法)	Btu	ft·lbf	eV
	1	0.101972	2.77778×10^{-7}	0.238889	9.47813×10^{-4}	0.737562	6.24150×10^{18}
	9.80665	1	2.72407×10^{-6}	2.34270	9.29487×10^{-3}	7.23301	6.12082×10^{19}
	3.6×10^6	3.67098×10^5	1	8.59999×10^5	3412.13	2.65522×10^6	2.24694×10^{25}
	4.18605	0.426858	1.16279×10^{-6}	1	3.96759×10^{-3}	3.08747	2.61272×10^{19}
	1055.06	107.586	2.93072×10^{-4}	252.042	1	778.172	6.58515×10^{21}
	1.35582	0.138255	3.76616×10^{-7}	0.323890	1.28506×10^{-3}	1	8.46233×10^{18}
	1.60218×10^{-19}	1.63377×10^{-20}	4.45050×10^{-26}	3.82743×10^{-20}	1.51857×10^{-22}	1.18171×10^{-19}	1

1 cal = 4.18605 J（計量法）

= 4.184 J　（熱化学）

= 4.1855 J　（15 °C）

= 4.1868 J（国際蒸気表）

仕事率　1 PS（仏馬力）

= 75 kgf·m/s

= 735.499 W

放射能	Bq	Ci
	1	2.70270×10^{-11}
	3.7×10^{10}	1

吸収線量	Gy	rad
	1	100
	0.01	1

照射線量	C/kg	R
	1	3876
	2.58×10^{-4}	1

線量当量	Sv	rem
	1	100
	0.01	1

（86年12月26日現在）

酸性雨予測計算コードSTEM2の並列化