

JAERI - M  
86-190

連続エネルギー・モンテカルロ・コード  
VIMのベクトル化

1987年1月

菅沼 正之\*・樋口 健二・浅井 清

JAERI-M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。

入手の間合わせは、日本原子力研究所技術情報部情報資料課（〒319-11 茨城県那珂郡東海村）あて、お申しこしてください。なお、このほかに財団法人原子力弘済会資料センター（〒319-11 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。

JAERI-M reports are issued irregularly.

Inquiries about availability of the reports should be addressed to Information Division Department of Technical Information, Japan Atomic Energy Research Institute, Tokaimura, Naka-gun, Ibaraki-ken 319-11, Japan.

© Japan Atomic Energy Research Institute, 1987

編集兼発行 日本原子力研究所  
印 刷 日青工業株式会社

連続エネルギー・モンテカルロ・コードVIMのベクトル化

日本原子力研究所東海研究所計算センター

菅沼 正之\*・樋口 健二・浅井 清

(1986年12月19日受理)

VIMは臨界計算のための連続エネルギー・モンテカルロ・コードである。組合せ幾何形状システムを使用したランダム・ウォーク制御系のベクトル化をFACOM VP-100で行った。

ベクトル化はイベント・バンク方式と呼ばれる粒子の独立的挙動を利用する手法を用いた。VIMコードのベクトル化にはふたつの問題点がある。ひとつはベクトル化改造時に持ち込まれるオーバーヘッドが大きいことである。もうひとつは中性子の吸収、体系外への漏れ、分散低減のためのカット・オフ等によって同時に処理可能な中性子の数が減少し、並列度が減少するためにベクトル処理効率が低下することである。断面積ライブラリの単一バンド化、イベント・バンクを減少することにより平均ベクトル長が大きくなるように最適化した。

ベクトル版は、オリジナル版と比較して単純幾何形状データの場合1.39倍、複雑幾何形状データの場合1.13倍の速度向上を得ている。

Vectorization of Continuous Energy Monte Carlo Code VIM

Masayuki SUGANUMA\*, Kenji HIGUCHI and Kiyoshi ASAI

Computing Center, Tokai Research Establishment

Japan Atomic Energy Research Institute

Tokai-mura, Naka-gun, Ibaraki-ken

(Received December 19, 1986)

VIM is a continuous energy Monte Carlo code for criticality calculation. The random walk control system which uses combinatorial geometry system has been vectorized on FACOM VP-100.

Vectorization has been done by the event bank method which controls simultaneous multiple particle's random walks, since behavior of neutron is independent. In vectorization of VIM code, we have two problems. One is a large overhead introduced by program modifications for vectorization. Another is a lowering of vector processing efficiency, since the vector length decreases with time according to the absorption and leakage of neutron and cut off of neutron for variance reduction. The average vector length during the random walks has been kept long by utilizing cross section library of single energy band and by reducing the number of the event banks.

The performance ratio of vectorized version to the original one is 1.39 for the simple geometry and 1.13 for the complex geometry.

Keywords: VIM, Continuous Energy, Monte Carlo Codes, VP-100, Event Bank Method, Vector, Supercomputers, Computer Codes

---

\*) On leave from FUJITSU Ltd.

## 目 次

1. はじめに	1
2. 連続エネルギー・モンテカルロ・コードVIMの概要	2
2.1 断面積処理系について	4
2.2 衝突解析系について	6
2.3 取り扱える幾何形状について	7
2.4 評価系(Estimator systems)について	11
3. 動的挙動の解析	12
4. モンテカルロ・コードのベクトル化技法	20
4.1 イベント・バンク方式	20
4.2 ベクトル化の技法	20
4.3 各処理系に対するベクトル化	23
4.3.1 ランダム・ウォーク制御系のベクトル化に対する構造変更	23
4.3.2 断面積処理系のベクトル化	26
4.3.3 衝突解析系のベクトル化	30
4.3.4 幾何形状処理系のベクトル化	32
4.3.5 評価系(Estimator systems)に対する処置	36
5. ベクトル化の結果と問題点	37
6. おわりに	43
謝 辞	43
参考文献	43
APPENDIX A. 浮動小数点データをキーとして使用したエネルギー・グリッドの ハッシュ・サーチについて	45
APPENDIX B. 断面積ライブラリ作成時に必要な入力データとVIM実行時 に必要な入力データ	49

## CONTENTS

1. Introduction .....	1
2. The structure of continuous energy Monte Carlo code VIM .....	2
2.1 Cross section processing system .....	4
2.2 Collision analyzer .....	6
2.3 Geometrical data definition and geometrical data processing .....	7
2.4 Estimator systems .....	11
3. Dynamic behavior analysis for vectorization .....	12
4. Vectorization techniques for Monte Carlo method .....	20
4.1 Event bank method .....	20
4.2 Techniques of vectorization .....	20
4.3 Vectorization and modification .....	23
4.3.1 Structure modification of random walk control system .....	23
4.3.2 Vectorization of cross section processing system .....	26
4.3.3 Vectorization of collision analyzer .....	29
4.3.4 Vectorization of geometrical data processing system .....	31
4.3.5 Modification of estimator systems .....	35
5. Performance analysis of vectorized version and discussions .....	36
6. Concluding remarks .....	41
Acknowledgments .....	41
References .....	41
APPENDIX A. Energy grid hash search method by using floating point data .....	44
APPENDIX B. Input data needed for cross section library processing system and execution for VIM code .....	48

## 1. はじめに

VIM は米アルゴンヌ国立研究所で開発された中性子のエネルギーを連続量として取り扱う臨界計算のためのモンテカルロ・コードである。一般にモンテカルロ法は非決定論的手法であるため、計算結果に対し十分な精度を要求すると、多数の中性子のランダム・ウォークを追跡しなければならない。このため、決定論的な手法である Sn 法、直接積分法などに比べると多量の計算時間を必要とする。しかし、複雑形状の体系を無理なく解析しようとする決定論的方法をもとにするコードでは幾何形状の表現方法の制限から解析がむつかしく必然的にモンテカルロ法に頼らざるを得なくなる。そして、今後ともますますこの傾向は強まる方向にある。この様な理由からモンテカルロ法の計算量が増大することが予想されるため、計算時間を短縮する努力が一層必要となってきている。<sup>1), 2), 3), 4)</sup>

モンテカルロ法には、中性子エネルギーをいくつかの群に分けて近似する多群理論にもとづくモンテカルロ法と、中性子のエネルギーを連続量として取り扱う連続エネルギー・モンテカルロ法とがあり、前者の代表的なコードとして KENO IV, MORSE などがあり、後者の代表的なコードとして MCNP VIM などがある。多群理論にもとづくモンテカルロ・コード KENO IV に関してはすでにベクトル化が行われ 1.4 倍の処理速度の向上が達成されている。<sup>1)</sup> 多群理論コードで使用する群定数は媒質の各エネルギー群間における平均実効断面積である。そのため非均質性の高い系においては群定数作成時の精度の問題から中性子のランダム・ウォークの模擬が十分な精度で遂行されない場合が生ずる。非均質体系の解析が多く要求されるにつれてこの要求を満足する連続エネルギー・モンテカルロ・コード VIM の使用も今後増加することが予想される。このため、VIM のベクトル化を図った。しかし、現在のベクトル版では、熱中性子の散乱解析系、境界反射処理系のベクトル化が未作業のため一般の使用には耐えない。

本報告書では、連続エネルギー・モンテカルロ・コード VIM のベクトル化の方法と結果について記述する。連続エネルギー・モンテカルロ・コードのベクトル化は F.B. BROWN 他により MCV コードにおいて既になされている。<sup>2) 3)</sup> しかし、ベクトル化の手法として用いられているシャフリングと呼ばれる手法は小さなベクトル長で行わなければならない処理が多発する可能性があるため、処理効率が低下するおそれがある。そこで VIM ではイベント・バンク方式を用いてベクトル化を行い、この様な性能低下を極力防いだ結果、単純形状の入力データのもとで約 1.39 倍の性能向上を得た。なお、この時のベクトル化率は 67% であった。連続エネルギー方式のモンテカルロ・コードではエネルギー・グリッドのサーチ処理に多くの時間を消費する。

第 2 章ではベクトル化手法の理解のための予備知識として VIM コードの概略を述べる。第 3 章ではベクトル化のために行った VIM コードの調査結果について述べる。第 4 章ではベクトル化の手法について記述し、第 5 章ではベクトル化の結果及び問題点について論じる。

## 2. 連続エネルギー・モンテカルロ・コードVIMの概要

VIMは中性子の発生から消滅までのランダム・ウォーク中に発生する事象を模擬することによって中性子増倍系の実効増倍系数 ( $k_{eff}$ ) を計算するコードである。中性子のランダム・ウォーク中の事象には、中性子の飛行、媒質を構成する核種との衝突、衝突の結果の散乱、吸収及び体系外への漏れ等がある。コードの主要部は、中性子のランダム・ウォーク中の事象の処理系とそれらを取りまとめるランダム・ウォーク制御系から構成される。

Fig. 2.1で示すようにランダム・ウォーク制御系は逐次的に取り出された中性子の媒質中でのランダム・ウォークを各事象の処理系を用いることによって模擬する。事象の処理系は大別するとつぎのようなものに分類される。

- ① 粒子発生系
  - ・ランダム・ウォークの対象となる中性子を発生させる
- ② 断面積処理系
  - ・中性子のエネルギーに対応する媒質の断面積を計算する
- ③ 衝突解析系
  - ・中性子と媒質を構成する核種の衝突から生じる核反応を模擬する
- ④ 幾何形状処理系
  - ・中性子が体系中のどの区画に移行するかの検索を行う
- ⑤ 評価系 (Estimator-systems)
  - ・中性子束、各種反応率の評価を行う

以下ではVIMのベクトル化の理解に必要な断面積処理系、衝突解析系、幾何形状解析系、評価系の概要を記述する。粒子発生系についてはベクトル化の理解に対してあまり重要でないため省略した。なお、これらの解析は断面積処理系、幾何形状系に関しては参考文献5.をもとにし、衝突解析系、評価系に関してはプログラムを解析した結果から処理概要を記述する。

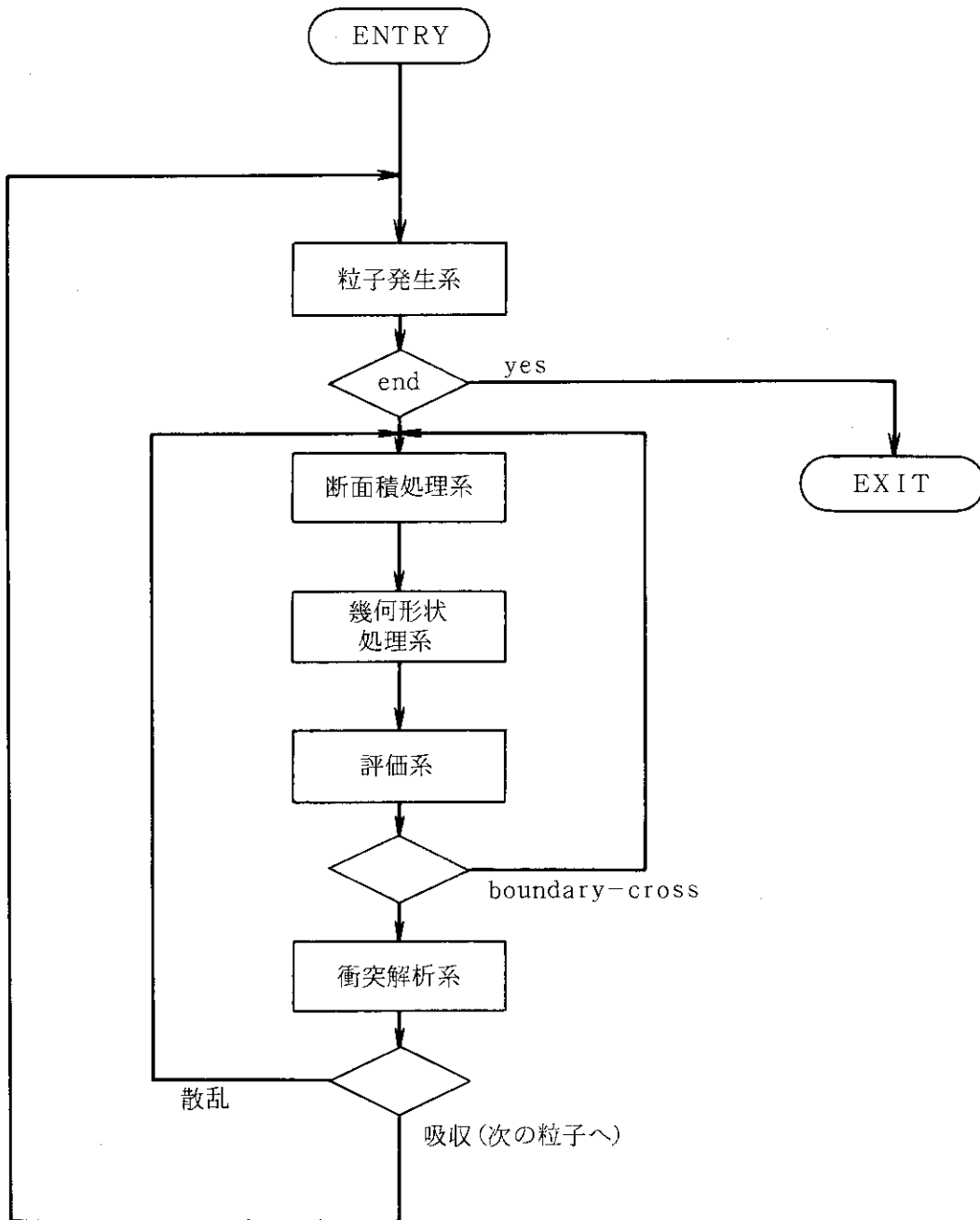


Fig. 2.1 The structure of random walk control routine

## 2.1 断面積処理系について

連続エネルギー方式に対する断面積ライブラリは核種ごとに中性子のエネルギーに対する核反応毎の断面積データのテーブル形式をとる。

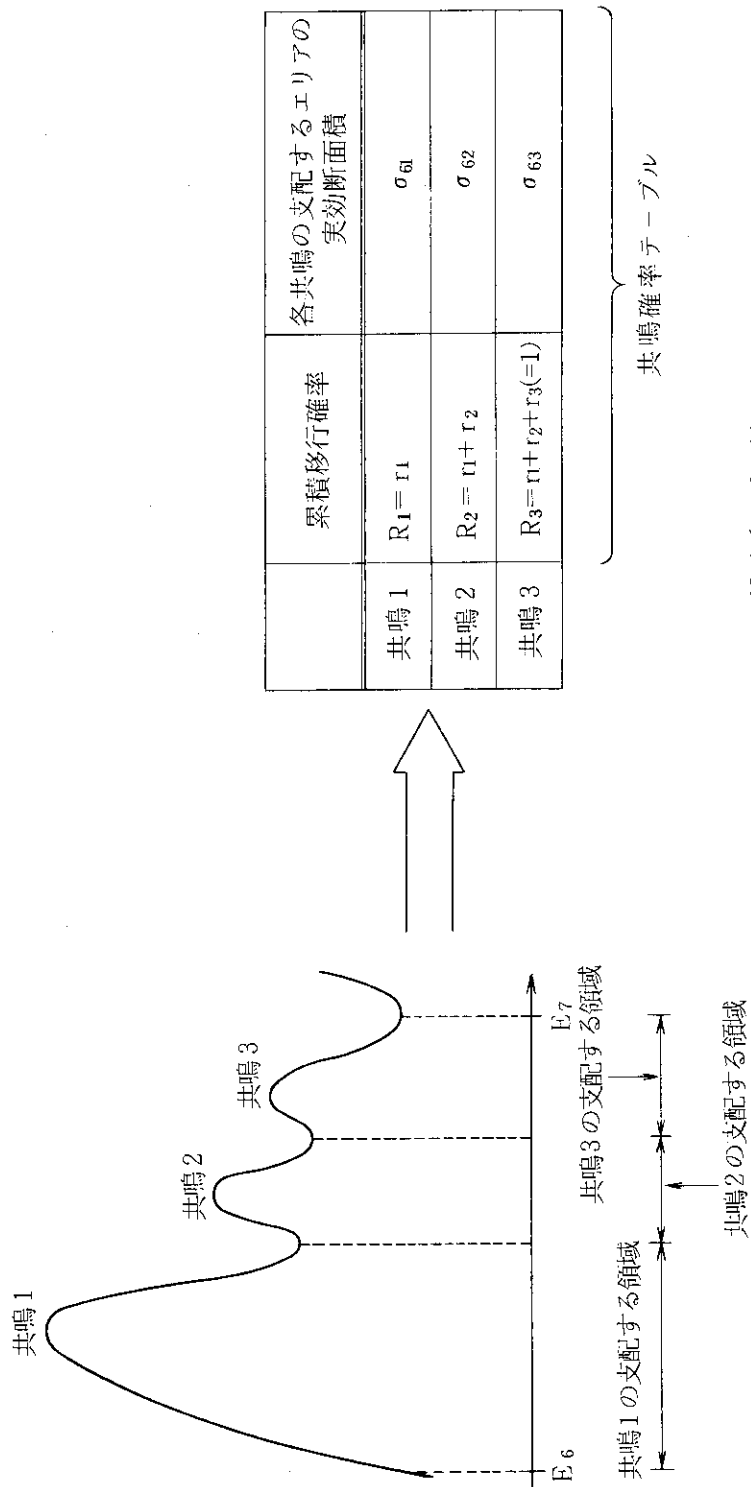
VIMコードの断面積ライブラリは、STAYSと呼ばれ処理中は計算機の主記憶に常駐する制御情報のデータ及びランダム・ウォークの進行に伴い必要部分のみ主記憶に読み込まれるMOVESと呼ばれる断面積データのテーブルから構成される。<sup>5)</sup> MOVESは離散化されたエネルギー・グリッドに対応する断面積の値のテーブルである。核反応断面積はエネルギーに対応して変化するが、比較的なだらかに変化する領域と共鳴領域と呼ばれる断面積ピークの多発する領域がある。このため、共鳴による断面積ピークの立て込んだ非分離共鳴領域では共鳴確率テーブルと呼ばれる断面積の特殊な表現形式が使用される。<sup>5)</sup>

断面積の計算方法はエネルギー領域によって異なる。非分離共鳴領域以外では線形補間が用いられる。一方、非分離共鳴領域ではFig. 2.2に示す様に共鳴の支配する区間での共鳴ピークの実効断面積とその共鳴ピークへの累積移行確率から構成される共鳴確率テーブルを使用し、一様乱数値と累積移行確率の比較から使用すべき共鳴ピークの断面積を選択する。Table 2.1で示すように、VIMでは媒質の微視的断面積八種類と巨視的断面積二種類（全断面積、吸収断面積）が計算される。巨視的断面積は中性子の媒質中の飛程の計算及び衝突時の散乱/吸収の分岐のための判断に使用される。核種毎の微視的断面積は核反応毎の反応率の計算及び散乱の解析に使用される。<sup>5)</sup>

Table 2.1 Cross section data used in the VIM code

	変数名	断面積の種類	VIMにおける定義
微視的	AOSAVE	吸収	POSAVE-SOSAVE-SISAVE
	COSAVE	吸収(核分裂を除く)	AOSAVE-FOSAVE
	FOSAVE	核分裂	補間または共鳴確率テーブルより得る
	POSAVE	全断面積	補間または共鳴確率テーブルより得る
	SISAVE	非弾性散乱	補間による
	SOSAVE	弾性散乱	補間または共鳴確率テーブルより得る
	S2SAVE	(n, 2n)反応	補間による
	XNSAVE		FOSAVE * $\bar{\nu}(E)$
巨視的	SIGTOT	全断面積	$\Sigma$ (原子密度 * POSAVE)
	SIGABS	吸収	$\Sigma$ (原子密度 * AOSAVE)

- $\bar{\nu}(E)$  は一核分裂あたりの中性子の平均発生数
- 巨視的断面積のVIMにおける定義の欄で見られる $\Sigma$ は媒質中の各種の総和を表す



- RN を  $(0, 1]$  区間における一様乱数とし,  
 $R_{i-1} < RN < R_i$  ( $i = 1, 3$ )  
 を満足する  $i$  を持つ共鳴の断面積を選択する。  
 ( $R_0 = 0$  とする。)

- エネルギー・グリッド  $E_6$  と  $E_7$  の間には 3 つの共鳴ピークがあり,  
 共鳴 1 に対する移行確率を  $r_1$   
 共鳴 2 に対する移行確率を  $r_2$   
 共鳴 3 に対する移行確率を  $r_3$   
 とする。

Fig. 2.2 The structure of resonance probability table

2.2 衝突解析系について

衝突解析系は中性子と媒質を構成する核種の衝突によって生じる核反応を解析する。通常、媒質は複数の核種から構成されているため、衝突の target となった核種を選択する必要がある。Fig. 2.3で示すように、target 核種を選択は媒質の巨視的断面積  $\Sigma_t$  に一様乱数值 RANF を乗じた値  $\Sigma_t * RANF$  と核種毎の巨視的断面積から作られる累積確率テーブルの値の比較によって行う。

衝突の結果生じる反応は散乱及び吸収である。衝突の target 核種を選択と同様に散乱及び吸収の選択は target 核種の微視的全断面積に一様乱数值を乗じた値と核種の散乱・吸収の微視的断面積の累積確率テーブルの値の比較によってなされる。

Table 2.2に示すようにVIM では散乱の形式を非熱領域で二種、熱領域で三種に分類して扱う。中性子の持つエネルギー、target 核種の特性に応じて散乱の形式が選択される。

なお、コード内では付加的な処理として衝突の評価、分散低減のためのロシアン・ルーレット、スプリットティングの処理、次世代の fission-site の保存の処理等を行っている。

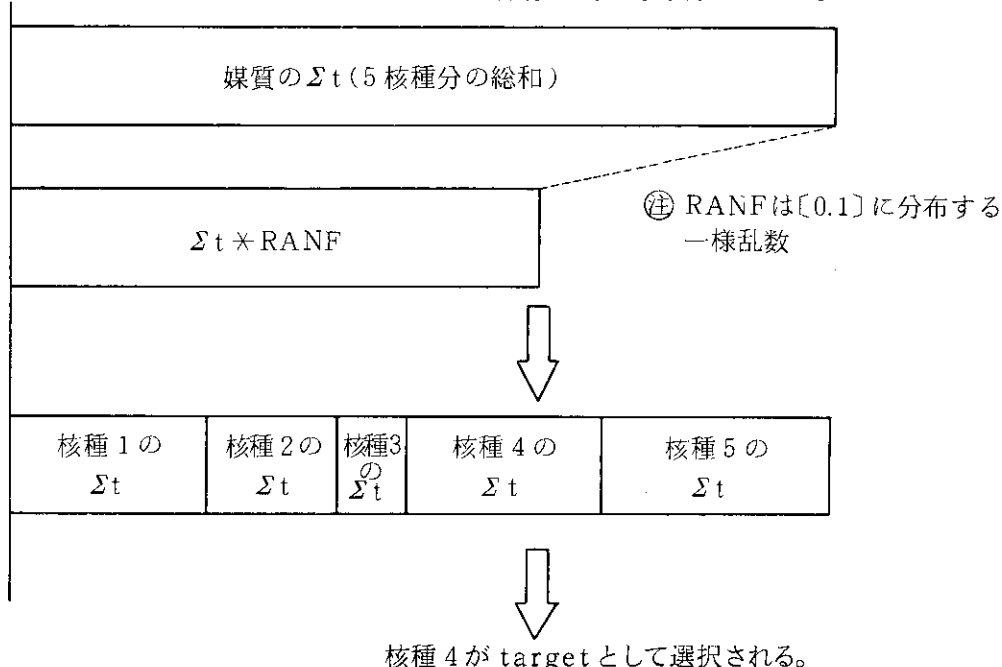


Fig. 2.3 The selection method of target nuclide

Table 2.2 Classification of scattering type in VIM code

反応タイプ名	
非熱領域	非弾性散乱
	弾性散乱
熱領域	Free-gas モデルによる散乱
	非弾性散乱
	弾性散乱

### 2.3 取扱える幾何形状について

モンテカルロ法の特徴は問題としている体系の幾何形状に柔軟に対応できることである。VIMでは幾何形状処理系を三種類用意してより多様な問題に対処している。つぎのような幾何形状処理系が用意されている。<sup>5)</sup>

- (1) 平板格子用幾何形状処理系
  - ・平板を幾何形状の構成要素とする幾何形状処理系
- (2) 組合わせ格子用幾何形状処理系
  - ・六角柱，四角柱等の構成要素を使用して規則的な格子構造を定義
  - ・無限媒質の定義が可能
- (3) 組合わせ幾何形状処理系
  - ・あらかじめ用意されている何種類かの幾何形状構成要素を組合せて，無限媒質を除く任意の幾何形状を定義できる。

これらの幾何形状の処理系毎に専用のランダム・ウォーク制御系が用意されている。第4章で示すように，通常の使用で最も一般的と思われる組合せ幾何形状処理系専用のランダム・ウォーク制御系についてベクトル化作業を行った。ここでは，その組合せ幾何形状処理系の概略について述べる。

組合せ幾何形状方式の場合，あらかじめ何種類か用意されている body と呼ばれる幾何形状の構成要素を，Table 2.3 で示すような集合演算子によって結合して zone と呼ばれる区画を定義する。与えられた問題の体系は zone の和集合によって表現される。<sup>6)</sup> VIM の組み合わせ幾何形状系ではつぎに示す11種の body タイプが用意されている。

Box  
 Right Parallel Piped  
 Sphere  
 Right Circular Cylinder  
 Right Elliptic Cylinder  
 Ellipsoid  
 Truncated Right Cone  
 Right Angle Wedge  
 Arbitrary Polyhedron  
 Right Hexagonal Prizm in Z-direction  
 Right Circular Cylinder in Z-direction

Table 2.3 Logical operator used to define geometrical shape

記号	集合演算的意味	備考
+	積	body の内側であることを示す。
-	積	body の外側であることを示す。
OR	和	

幾何形状処理系内部ではAND演算をもとに処理を進めているため zone の定義にOR演算子が用いられていると処理が不可能となってしまう。これを避けるために幾何形状の前処理の段階で zone の再定義を行ってAND演算だけで処理が行なえるような zone を幾何形状処理系内部で使用する code-zone として再定義する。

Fig. 2.4 の例1は球と円筒の二つの body を用いて zone A の定義を行った例である。この場合 A は集合演算的にはANDの意味合いを持つ $+$ ,  $-$ の演算子によって球の内側かつ円柱の外側の領域として定義される。一方, Fig. 2.4 の例2では例1と同様に球と円柱の二つの body を使って zone B の定義を行っている。この定義ではOR演算子が使用されているため, zone B は球と円柱の結合領域(球の内側又は円柱の内側)として定義されている。B に対し幾何形状処理系は前処理の段階でOR演算子を使用しないで済む形の二つの code-zone を再定義している (B を球の内側かつ円柱の外側で定義される code-zone B1 と円柱の内側で定義される code-zone B2 の二つに分割している)。

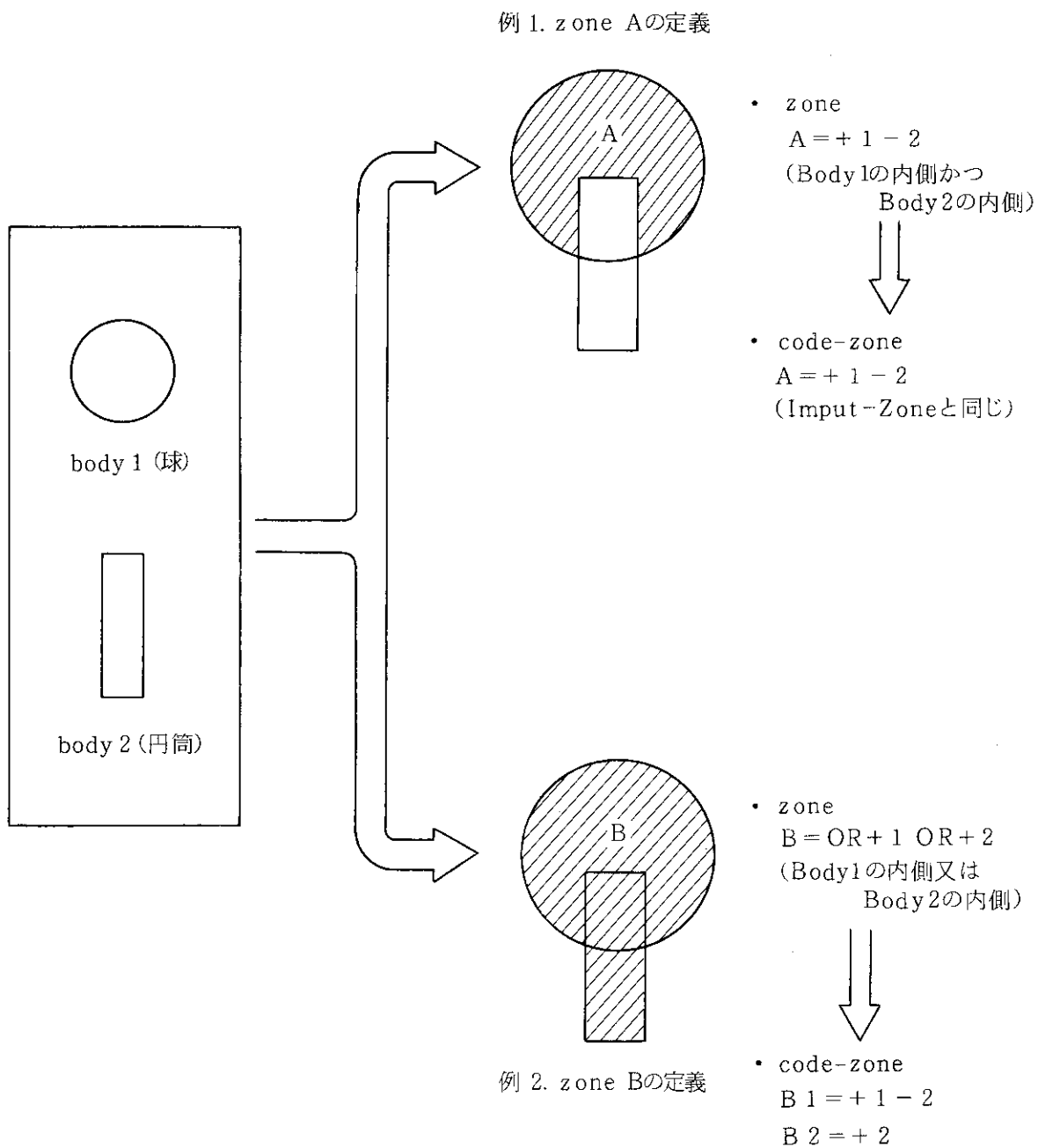


Fig. 2.4 The relation between zone and code-zone

幾何形状処理系の役割は中性子の衝突又は boundary-cross の判定を行うことと、boundary-cross が起こった場合に中性子が次にはいる zone の検索を行うことである。次のような処理が行われる。

- (1) 中性子の属する zone から実際に中性子の属する code-zone を検索する。
- (2) 中性子の飛程の始点の属する code-zone と終点の属する code-zone が同一なら衝突とみなし幾何形状系の処理を終了し、同一でないなら boundary-cross とみなし(3)の処理を行う。
- (3) boundary-cross して次にはいる code-zone の検索を行う。
- (4) 検索された code-zone から対応する zone を得る。検索前の zone と検索後の zone が同一であれば検索された code-zone の境界まで中性子を進め、(2)に戻って処理が続けられる。

boundary-cross して次にはいる code-zone の検索は高速化のために code-zone 間の隣接関係を記述する Fig. 2.5 で示すようなリスト構造のデータを利用して行われる。リスト構造のデータはつぎのようにして作成される。例えば、code-zone 1 に注目する。過去に一度も code-zone 1 から他の code-zone に boundary-cross したことがないならば全 code-zone を検索し、求まった code-zone を隣接 code-zone 番号 1 としてリスト構造のデータとして登録する。次に code-zone 1 から他の code-zone に boundary-cross が起こる場合、まず最初にこのリスト構造上のデータである隣接 code-zone 番号 1 が該当する code-zone かをチェックされ、条件を満たすようであれば(4)で示す処理に移る。また、該当する code-zone でなかった場合、全 code-zone を検索して求まった code-zone を隣接 code-zone 番号 2 としてリストに登録する。幾度か繰り返しを行うことによって隣接 code-zone の番号が全てリスト上に登録されると全 code-zone を対象とする検索は行われなくなる。このようにして code-zone 検索時の時間短縮を行っている。

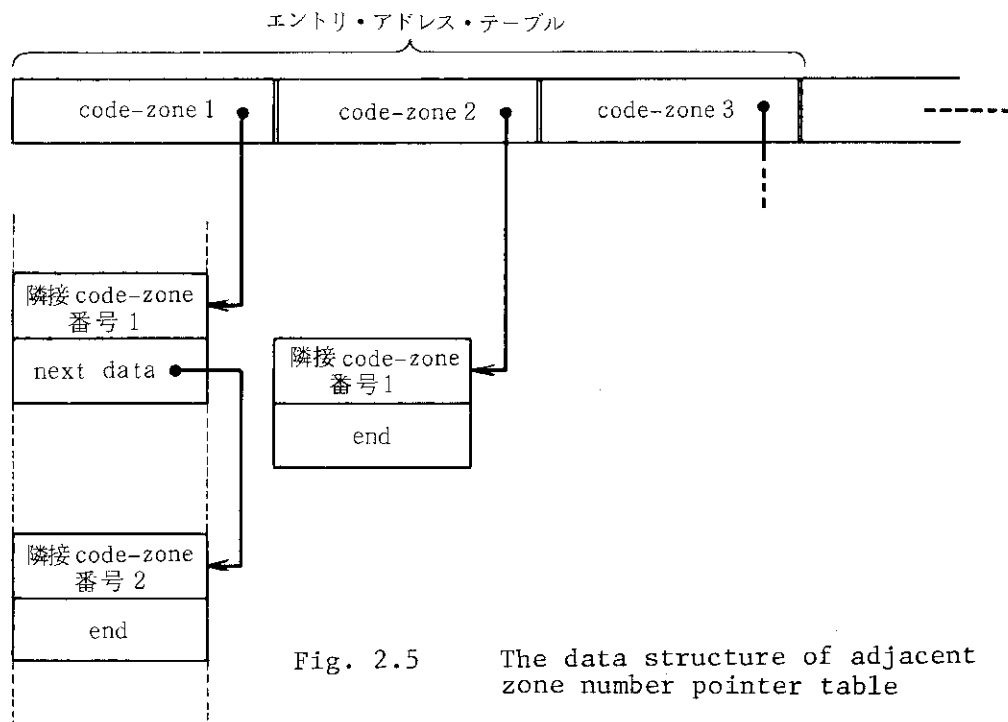


Fig. 2.5 The data structure of adjacent zone number pointer table

## 2.4 評価系(Estimator-systems)について

評価系はランダム・ウォークの中に変化する中性子に関する属性を zone 毎, エネルギー毎に分類して逐次累積してゆくことによって中性子束, 実効増倍係数 ( $k_{eff}$ ), 各種反応率などの巨視的な量を求める。Table 2.4 に VIM で使用可能な評価法を示す。

属性の累積は zone 毎, エネルギー毎に分類して行われる。VIM ではエネルギーを連続量として取り扱っているが, 累積には edit-energy-group と呼ばれるエネルギー群を定義してこれをエネルギー分類用として使用する。

Table 2.4 Estimator systems in VIM code

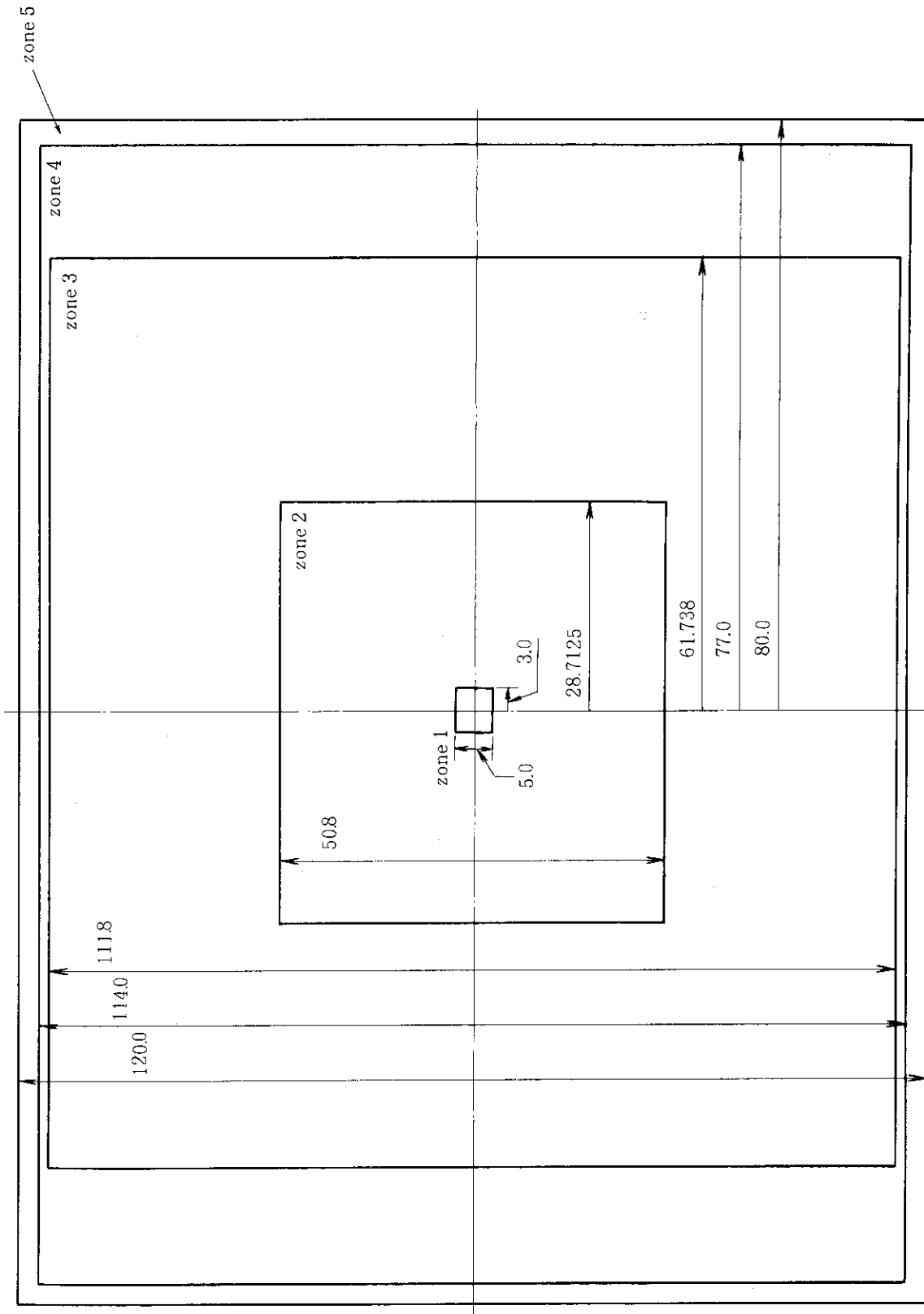
Estimator の種類	評 価 対 象	評価する物理量
Track length	粒子の飛程	粒子束 反応率 実効増倍係数
Collision	衝突	粒子束 実効増倍係数
Analogue	衝突	実効増倍係数

### 3. 動的挙動の解析

動的挙動の解析に使用したサンプル入力データは、高速臨界集合体及び未臨界集合体を模擬するものである。双方のデータとも処理した中性子の粒子数は9000個(900×10バッチ)で、分散低減のためのオプションは使用していない。Fig. 3.1に5個の同心円筒から成る高速臨界集合体の幾何形状の垂直断面図を示す。以下このデータを単純形状の入力データと呼ぶ。ここでは zone 1と zone 2で冷却材を含む核燃料の領域、zone 3でブランケット領域、zone 4で構造材の領域、zone 5で集合体外の領域を示している。また Fig. 3.2-a, 及び Fig. 3.2-bは未臨界集合体の垂直、水平断面図で、zone 1から zone 21の21個の直方体を使用して冷却材を含む核燃料の領域を定義し zone 22から zone 24までの3個の同心円筒でブランケット、構造材集合体外の領域を示す。以下このデータを複雑形状の入力データと呼ぶ。複雑形状の入力データは複雑形状下でのサブルーチン毎のコスト(スカラ計算における実行時間に比例する量)の分布を見るために単純形状の入力データをもとに作成したもので、実際の物理モデルは存在しない。単純形状データと複雑形状データの媒質は冷却材を含む核燃料の領域、ブランケット、構造材の三種類定義され、その構成核種をTable 3.1に示す。

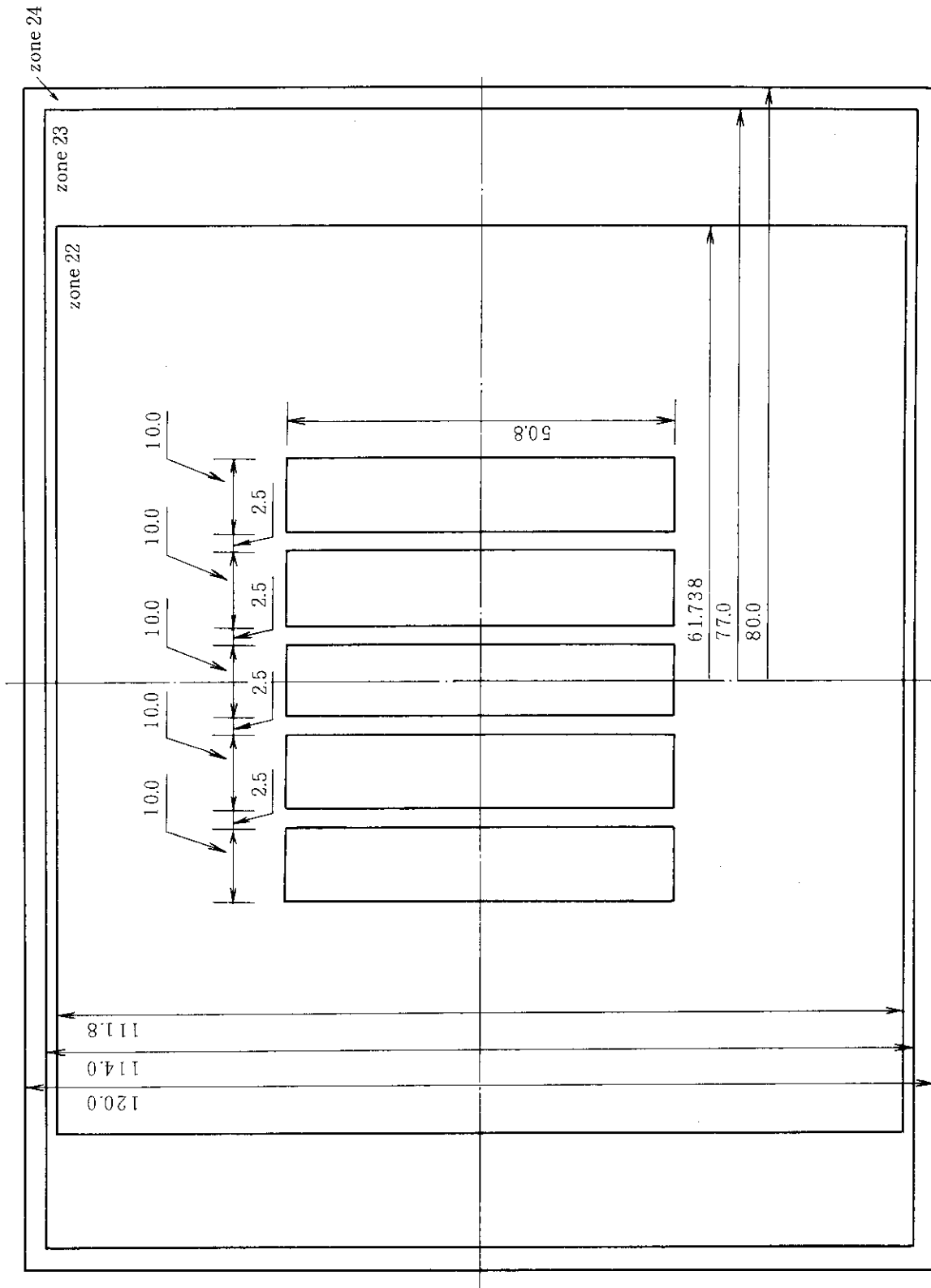
Table 3.2は単純形状の入力データを使用した場合のFORTUNE<sup>7)</sup>によるサブルーチンごとのコストの分布である。最もコストの高いのが断面積処理系の制御ルーチンCROSSで39.2%、次が衝突解析制御ルーチンREACTで13.4%、幾何形状処理ルーチンGG 12.5%と続く。また、Table 3.3は複雑形状の入力データを使用した場合のFORTUNEによるサブルーチン別の実行コストの分布である。この場合、幾何形状が複雑であるため幾何形状処理ルーチンGGが最も高コストで、45.9%、次が断面積処理系の制御ルーチンCROSSで20.7%、幾何形状処理制御ルーチンG1 8.1%と続く。

入力データによって多少のちがいはあるが、ランダム・ウォークの模擬のために必要な幾何形状、断面積処理、衝突解析に必要なルーチン群が大きなコストを占めていることがわかる。動的挙動解析の結果からベクトル化対象となるサブルーチンのコストの総和が95%以上となる様に改造するルーチンを選択する。ベクトル化のために選択されたルーチンは8ルーチンであり、これらをTable 3.4に示す。



单位 : cm

Fig. 3.1 A side view of simple geometrical data



单位 : cm

Fig. 3.2-a A side view of complex geometrical data

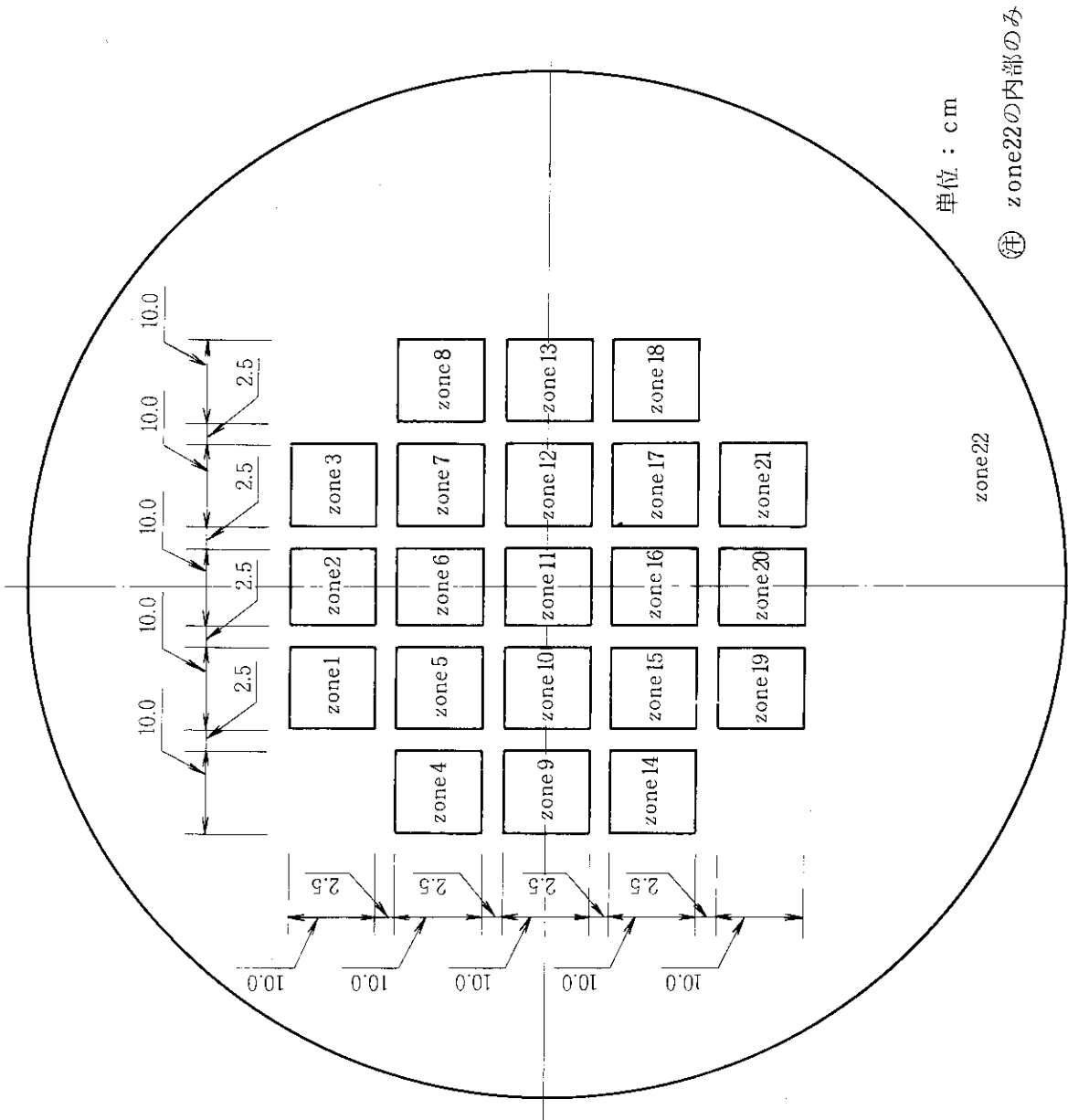


Fig. 3.2-b A top view of complex geometrical data

Table 3.1 Nuclides used in the sample input data

媒 質	同位体数	同 位 体
冷却材を含む 核 燃 料	12	Pu <sup>240</sup> , Pu <sup>241</sup> , U <sup>235</sup> , U <sup>238</sup> , Pu <sup>239</sup> , Am <sup>241</sup> , Cr, Ni, Fe, Al <sup>27</sup> , Na <sup>23</sup> , O <sup>16</sup>
ブランケット	5	U <sup>235</sup> , U <sup>238</sup> , Cr, Ni, Fe
構 造 材	3	Cr, Ni, Fe





Table 3.4 The name and function of vectorized routines

ルーチン名	機能
NUTRNG	組合せ形状処理系専用のランダム・ウォーク制御ルーチン
CROSS1 (CROSS)	断面積処理系の制御ルーチン
UNRES1 (UNRES)	共鳴確率テーブルを検索し、使用すべき共鳴ピークの実効断面積を求める
REACT1 (REACT)	衝突解析系の制御ルーチン
ELAST1 (ELAST)	非熱領域の弾性散乱を解析し散乱後の中性子のエネルギー、飛行方向を決定する。
G1ONE (G1 )	組合せ幾何形状系の制御ルーチン
GGONE (GGONE)	組合せ幾何形状系の work house ルーチン
FLUXX1 (FLUXX)	中性子の飛程による中性子束評価のためのルーチン

・括弧内のルーチン名は実際の処理用サブ・エントリ名であり、この場合メイン・エントリは引数のベース・アドレスの設定の機能しか無い。

## 4. モンテカルロ・コードのベクトル化技法

### 4.1 イベント・バンク方式

オリジナル版では中性子が発生し、体系内部で消滅するか体系外へ漏れ出てしまうまでのランダム・ウォークの追跡を、中性子1個ずつ逐次的に行っている。一方、ベクトル版では中性子に関してベクトル処理するために一度に複数個の中性子のランダム・ウォークを取り扱う様にならなければならない。このためイベント・バンク方式<sup>1)</sup>と呼ばれる複数中性子のランダム・ウォークを管理するための手法を用いた。この方法ではまず、ランダム・ウォーク中の中性子に対するイベントを定義する。イベントとは中性子が媒質中をランダム・ウォークしてゆく過程で必要になる断面積の計算、zoneの検索、中性子束の評価、衝突後の状態の決定及び反射の処理などを指す。同一のイベントが発生した粒子をイベント・バンクと呼ばれる中性子溜りに集めると、同一イベント・バンク中の中性子に対しては同一の処理を施せば良いため、中性子に関してベクトル処理が可能となる。各イベントの処理が終了した後に処理後の中性子の状態に対応したイベント・バンクへ中性子を振り分け、次のイベントの処理に移る。ベクトル演算器を効率良く使用するために、常にその時点で最も多くの中性子が存在するイベントが次の処理として選択される。この選択を行うことでシャフリング方式<sup>2,3)</sup>の問題点である短ベクトル長での処理の多発による性能低下を避けることが可能となる。

### 4.2 ベクトル化の技法

#### (1) ベクトル化DO変数の選択法

各イベントの処理に対し、イベント・バンク方式によって導入された中性子に関するDOループをベクトル化する。新しく追加された中性子のDOループによって多重DOループの構造となる部分は平均ベクトル長の大きい中性子のDOループが最も内側になる様にし、中性子のループに対してベクトル化が行われるようにする。Fig. 4.1に中性子のDOループのベクトル化例を示す。ここでは、N350¥がイベントで処理されるべき中性子の総数を示し、L350¥中にイベントで処理されるべき中性子の識別番号を保存している。SIGTO¥(全断面積)などの個々の中性子に対応する物理量のアクセスはリスト・ベクトルを使用したランダム・アクセス方式をとる。

```

*VOCL LOOP,NOVREC
1-----V-----DO 4120 KI¥ = 1 , N350¥
1          V      KY = L350¥(KI¥)
1          V      IF(NASC¥(KY).EQ.0) DIST¥(KY)=0.0
1          V      DISTO¥(KY)=DIST¥(KY)+STMFP¥(KY)/SIGTO¥(KY)
1          V      NNEXT¥ = NNEXT¥ + 1
1          V      LNXTI¥(NNEXT¥) = KY
+-----V-4120 CONTINUE

```

Fig. 4.1 An example of vectorization for Do loop of neutron

## (2) ベクトル化によるルーチン間の整合

ベクトル化のための改造により COMMON 変数及びルーチンの引数が配列化されたり、配列の次元が上がったりする場合には、ベクトル版ルーチンの配下で使用されるベクトル化の対象となっていないルーチンと整合を取らなければならない。整合を取る方法はインターフェース整合用のルーチンを使用する方法と内部に中性子に関する DO ループを引き込む方法が考えられる。

インターフェース整合用のルーチンは呼出し回数の少ないルーチンに対して使用される。インターフェース整合用のルーチン例を Fig. 4.2 に示す。このサブルーチンの内部では上位ルーチンから渡された処理すべき中性子リスト LIST1 に従ってサブルーチン呼出の前後で一つずつベクトル版 COMMON COMCO2 の配列 D2 と対応する下位ルーチンの COMMON COMCOM の変数 D の間でデータを転送する。

呼出し回数の多いルーチンに対してインターフェース整合用サブルーチンを使用することはサブルーチン呼出時のオーバーヘッドが増加することで好ましくない。このような場合は中性子に関する DO ループを内部に引き込み、リスト・ベクトルを用いたランダム・アクセス方式を用いて直接ベクトル版の COMMON 変数、引数をアクセスすることによってインターフェースの整合を行う。

```

SUBROUTINE      A1*(LIST1,LNG1)
INTEGER*4      LIST1(LNG1)
COMMON        /COMCOM/ D
COMMON        /COMCO2/ D2(1000)
C
DO 4000 KI = 1 , LNG1
  K = LIST1(KI)
  D = D2(K)
  CALL A
  D2(K) = D
4000 CONTINUE
C
RETURN
END

```

Fig. 4.2 An example of subroutine for calling the original subroutine from the vectorized subroutine

## (3) DO ループ中のアセンブラで書かれた外部手続きの取り扱いについて

呼出回数が多く実行ライン数の少ないアセンブラで書かれた外部手続き GTISO, AZIRN 等は Fig. 4.3 で示すように DO ループ中に同一機能の FORTRAN ステートメントとしてインライン展開した。このインライン展開によって呼出し時のオーバーヘッドが低減され DO ループ中の外部手続きの参照がなくなるため、DO ループのベクトル化が促進された。また、インライン展開不可能なアセンブラで書かれた一様乱数ルーチン FLTRNF に関して Fig. 4.4 で示すように、ベクトル版一様乱数発生ルーチン RANU2 を使用するために、あらかじめ作業配列 RR ¥ に書き込まれてある一様乱数値を使用するように変更した。この変更によって一様乱数ルーチン呼出時のオーバーヘッドが減少するとともに、外部手続きの参照が無くなったために DO ループがベクトル化された。

## (4) IF 文のネスト構造の低減

DO ループ中の IF 文のネスト構造が深くなってしまいうような場合、DO ループの実効ベクト

ル長が小さくなることによる性能低下が起こる可能性が大きいためネスト構造が浅くなるよう制御構造を変更する。

```
C..... CALL AZIRN(BSTAR%(K%),GSTAR%(K%))
        BSTAR%(K%) = SIN(PI2%*RR%(NRR%+KI%))
        GSTAR%(K%) = COS(PI2%*RR%(NRR%+KI%))
```

ASIRN インライン展開例

⊕ RR%上には[0,1]の一樣乱数が書き込まれている。

```
C..... CALL GTISO(GAMMA%(K%),ALPHA%(K%),BETAP%(K%))
        WKS1 = SIN(PI2%*RR%(NRR%+KI%**2-1))
        WKC1 = COS(PI2%*RR%(NRR%+KI%**2-1))
        WKS2 = 2.0*RR%(NRR%+2*KI%) - 1.0
        WKC2 = SQRT(1.0-WKS2*WKS2)
        GAMMA%(K%) = WKC2*WKC1
        ALPHA%(K%) = WKC2*WKS1
        BETAP%(K%) = WKS2
```

GTISO インライン展開例

⊕ RR%上には[0,1]の一樣乱数が書き込まれている。

Fig. 4.3 An example of in-line expansion of external procedure

```
DO 4120 KI% = 1 , N350%
K% = L350%(KI%)
RANF=FLTRNF(0)
...
STMFP%(K%) = -ALOG(RANF)
...
4120 CONTINUE
```

: オリジナル版一樣乱数ルーチン FLTRNF。



```
CALL RANU2(IX%, RR%(IREM+1), NRR%, ICON )
...
DO 4120 KI% = 1 , N350%
K% = L350%(KI%)
...
STMFP%(K%) = -ALOG(RR%(NRR%+KI%))
...
4120 CONTINUE
```

RR%上書き込んだ一樣乱数値を使用する。

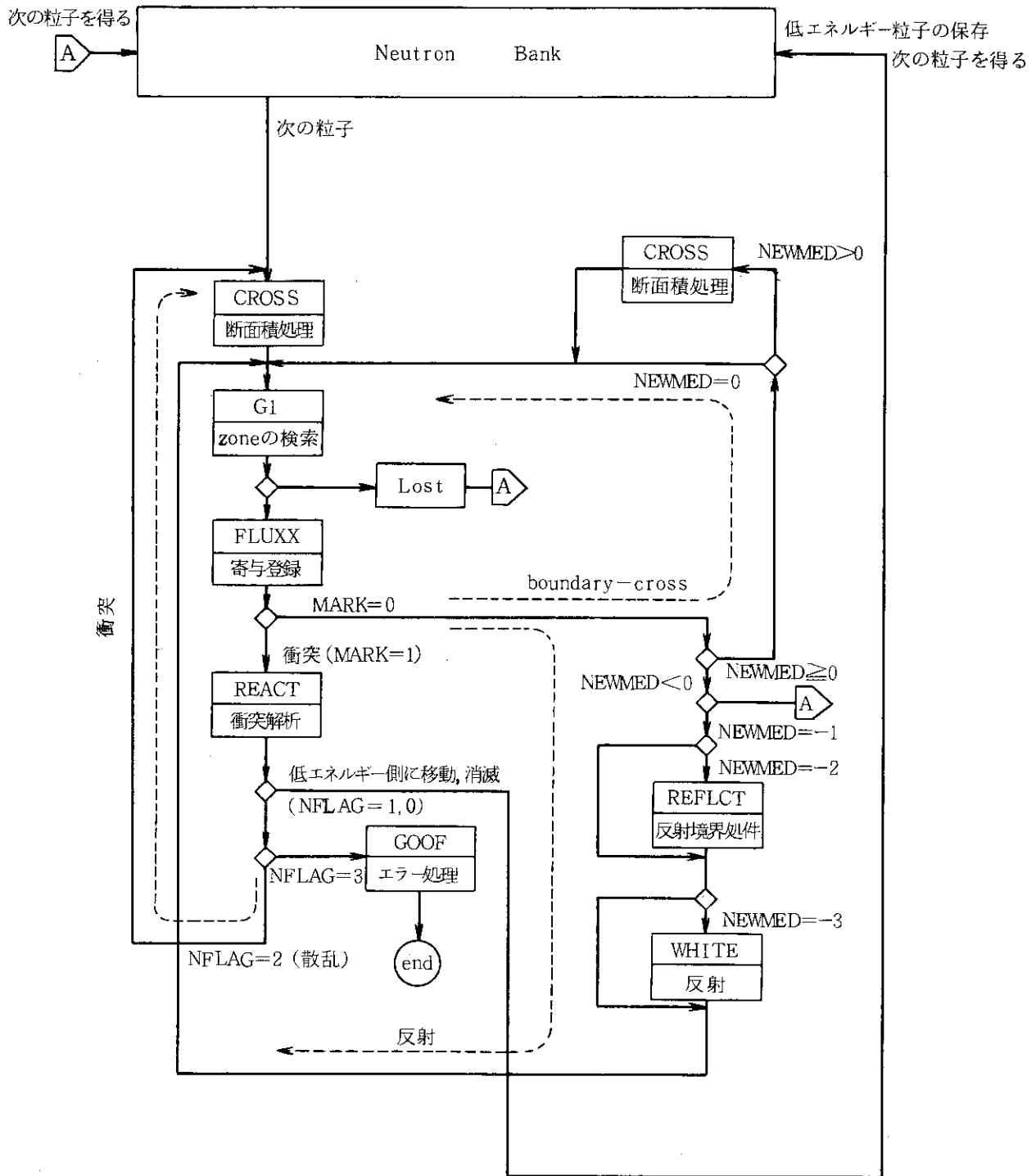
⊕ RANU 2はベクトル版一樣乱数ルーチンである。

Fig. 4.4 An example of removing the random number generator routine

### 4.3 各処理系に対するベクトル化

#### 4.3.1 ランダム・ウォーク制御系のベクトル化に対する構造変更

イベント・バンク方式を用いたベクトル化に際してはランダム・ウォークする中性子に対して生ずるイベントを抽出し、イベント・バンクの定義をしなければならない。そのためFig. 4.5に示すオリジナル版の組合せ幾何形状用ランダム・ウォーク制御ルーチンNUTRNGの制御構造を解析する必要がある。ランダム・ウォーク制御ルーチンNUTRNG内には衝突中性子の流れる経路, boundary-cross 中性子の流れる経路, 及び反射の処理に必要な中性子の流れる経路の3つがある。neutron-bankから取り出された中性子に対して断面積処理ルーチンCROSSを使用することによって中性子のエネルギーに対応した断面積が計算される。断面積から中性子の媒質中での飛程を求め幾何形状処理ルーチンG1を使って中性子の属するzone内で衝突を起こす場合フラグMARKを1に, それ以外ではMARKを0にセットする。このあと, 飛程による中性子束, 各種反応率の評価のためのルーチンFLUXXでzone内の飛程の累積を行う。サブルーチンG1で設定されたフラグMARKによって衝突側に分岐した中性子は衝突解析ルーチンREACTで決定された状態に依って, 散乱の場合はエネルギーが変化するため断面積の再計算を行い, 吸収の場合は粒子が消滅するため新しい粒子のランダム・ウォークの追跡を行う。一方, 衝突を起こさない中性子は次にはいるzoneの媒質番号NEWMEDが負の場合は反射中性子の処理に分岐し, NEWMEDの番号に対応する処理ルーチンで反射の処理が行われる。また, 媒質番号NEWMEDが0以上の場合はboundary-crossの処理となる。この時, NEWMEDが正の場合に限り断面積処理ルーチンCROSSで新しい領域の中性子のエネルギーに対応する断面積が計算される。境界反射, boundary-crossの処理が終った中性子はつぎの状態を決定するために幾何形状処理ルーチンG1に戻る。以上の解析からランダム・ウォーク制御ルーチンNUTRNG内で使用されている下位ルーチン毎にTable 4.1で示すイベント・バンクを定義した。



NEWMED ..... 粒子が次に入る Zone の媒質  
 MARK ..... G1が決定するバウンダリクロス/衝突の状態フラグ  
 NFLAG ..... REACTが決定する粒子の衝突後の状態フラグ

Fig. 4.5 The structure of random walk control routine NUTRNG for combinatorial geometry of the original VIM code

Table 4.1 The name and the physical meaning of event bank

バンク名	実処理ルーチン	物 理 的 意 味
Lower BANK		主記憶上に読み込まれている断面積データでは処理不可能な低エネルギー領域の粒子のためのバンク
Cross BANK	CROSS1 (CROSS)	衝突によってエネルギーが変化したことで断面積の計算が必要になった粒子のためのバンク
Cross2 BANK	CROSS1 (CROSS)	boundary-cross によって媒質が変化したことで断面積の計算が必要になった粒子のためのバンク
Next-zone BANK	G1ONE (G1)	boundary-cross / 衝突の判断が必要となった粒子のためのバンク
Fluxx BANK	ELUXX1 (FLUXX)	zone 内の飛程による中性子束、反応率の評価が必要となった粒子のためのバンク
Reaction BANK	REACT1 (REACT)	媒質を構成する核種と粒子の衝突から生ずる反応を解析する必要の生じた粒子のためのバンク
Reflect BANK	NRML1 (REFLCT)	境界反射条件を適応すべき粒子のためのバンク
White BANK	NRML1 (WHITE)	白色反射条件（エネルギー依存無し）を適応すべき粒子のためのバンク
Error-handling BANK	GOOF	衝突解析時にエラーの生じた粒子のためのバンク

・括弧内のルーチン名は実際の処理用サブ・エントリ名であり、この場合メイン・エントリは引数のベース・アドレスの設定の機能しか無い。

ランダム・ウォーク中の各イベントの平均ベクトル長を可能な限り伸ばすためつぎのような処置を講じた。

(1) イベント・バンクの最適化

ランダム・ウォーク中は各イベント・バンクに中性子が分散するため、イベント・バンクの数はできるかぎり少なくした方が各イベントの平均ベクトル長を大きく出来る。ランダム・ウォーク中のイベント・バンクの中性子数の変化を調べた結果、Cross BANKとCross 2 BANKを合併したほうが平均ベクトル長を大きくできることが判明したため2つのバンクの合併を行なった。

(2) 断面積ライブラリの単一エネルギー・バンド化

オリジナル版では、主記憶の使用量の減少と、断面積計算時のエネルギー・グリッド・サーチ時間を減少させるために断面積ライブラリを複数エネルギー・バンドに分割して使用している。主記憶上に存在するエネルギー・バンドに対応するエネルギーを持つ中性子しかランダム・ウォークが出来ないために、ベクトル版においては断面積ライブラリが複数バンドに分割されている

と、主記憶上に存在するエネルギー・バンドに対応するエネルギーを持つ中性子しかランダム・ウォークが出来ないためにランダム・ウォーク中の平均ベクトル長が低下する。このため、単一エネルギー・バンドの断面積ライブラリを使用した。単一エネルギー・バンドの断面積ライブラリを使用した場合、エネルギーによる中性子の選択の必要性が無くなる。そのためエネルギー・バンドの分割によるベクトル長の低下は無い。しかし、断面積ライブラリのエネルギー・テーブルの検索範囲が増加することによるテーブル・サーチの時間が増加する。その対策を4.3.2節に記述する。

#### 4.3.2 断面積処理系のベクトル化

断面積処理系は Table 4.2 に示す様に制御ルーチンと二つの下位ルーチンによって構成されており、主な処理はエネルギー・グリッドのサーチ、共鳴確率テーブルのサーチ、および断面積の補間である。Fig. 4.6 の(a)のオリジナル版の CROSS の制御構造からわかるように、中性子の属する zone の媒質を構成するすべての核種について核反応別微視的断面積を計算し、さらに媒質中の核種毎の原子密度を使って媒質の巨視的断面積の合成を行っている。このため、制御ルーチン CROSS 内には媒質を構成する核種についての DO ループが存在する。

Fig. 4.6 の(b)にベクトル版の制御構造を示す。ベクトル版では一度に複数個の中性子がランダム・ウォークするため、すべての中性子に対して断面積を計算しなければならない。zone 毎に構成核種が異なるため、予め zone 毎に中性子を振り分けたりストを作成し、zone 内の構成核種の同一性を使って中性子ループのベクトル化したため、当初は zone - 核種 - 中性子の三重 DO ループの制御構造となった。

ベクトル化は最も内側の中性子のループに対してなされる。このため、Fig. 4.6 の(b)のような三重 DO ループの構造の場合、zone 毎に中性子を振りわけてしまうと最も内側の中性子のループのベクトル長が低下してしまう欠点がある。このため、中性子の属する zone の断面積を計算するのに必要な核種毎に中性子のリストを作成し、核種毎に中性子の処理をすることで zone に関する DO ループを取り除き Fig. 4.6 の(c)で示すような核種-中性子の二重 DO ループとした。

Table 4.2 Subroutines in cross section processing system

ルーチン名	措 置	処 理 概 要
CROSS1 (CROSS)	vectorize	断面積処理系の制御ルーチン
UNRES1 (UNRES)	vectorize	非分離共鳴領域での微視的断面積を共鳴確率テーブルから求める(分裂, 散乱, 全断面積)
MORE	use original version	エネルギー・グリッドのサーチのための二分検索ファンクション・ルーチン( assembler )

・括弧内のルーチン名は実際の処理用サブ・エントリ名であり、この場合メイン・エントリは引数のベース・アドレスの設定の機能しか無い。

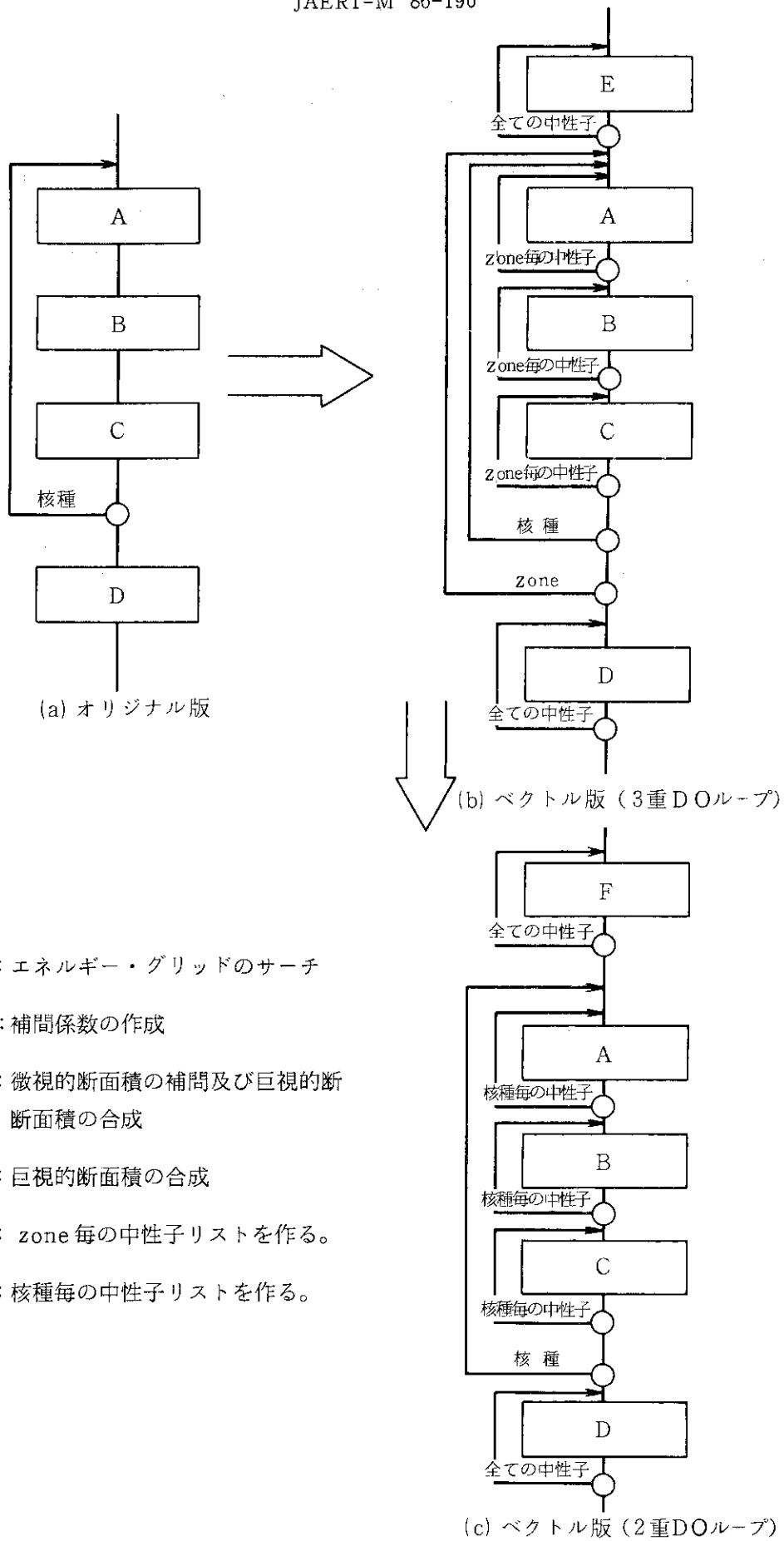


Fig. 4.6 The control structure of cross section processing subroutine CROSS

断面積ライブラリの単一バンド化によりサーチの対象となるエネルギー・グリッドの数が増加する。オリジナル版では、上方散乱のない領域においては Fig.4.7 に示すように、散乱によって中性子のエネルギーが下がるのに応じてグリッド・サーチの検索の上限を下げ検索範囲を狭める方式をとっている。検索の下限は処理中のエネルギー・バンドの下限が使用される。このため多バンド断面積ライブラリの場合は検索の範囲が狭いのに対し、単一バンド断面積ライブラリでは検索の範囲が非常に広がる。このため、検索に要する時間は多バンド断面積ライブラリの使用時に比較して50%ほど増加した。この対策として、Fig. 4.8 で示すように中性子束、反応レート評価用に設定された edit-energy-group のインデックスを使用し、検索の下限を中性子のエネルギーに応じて移動するように改造した。改良された方法を使うことにより、多バンド断面積ライブラリ使用時と同等のエネルギー・グリッド・サーチの所要時間にすることができた。

共鳴確率テーブル・サーチ・ルーチン UNRES は共鳴確率テーブルの選択、及び使用すべき共鳴の実効断面積のサーチの処理によって構成される。ベクトル化は上位ルーチン CROSS より中性子のループを引き込むことによって行われた。しかし、ベクトル処理しないほうが効率が良いループが多いため、大半の DO ループはスカラ・モードによって実行される。

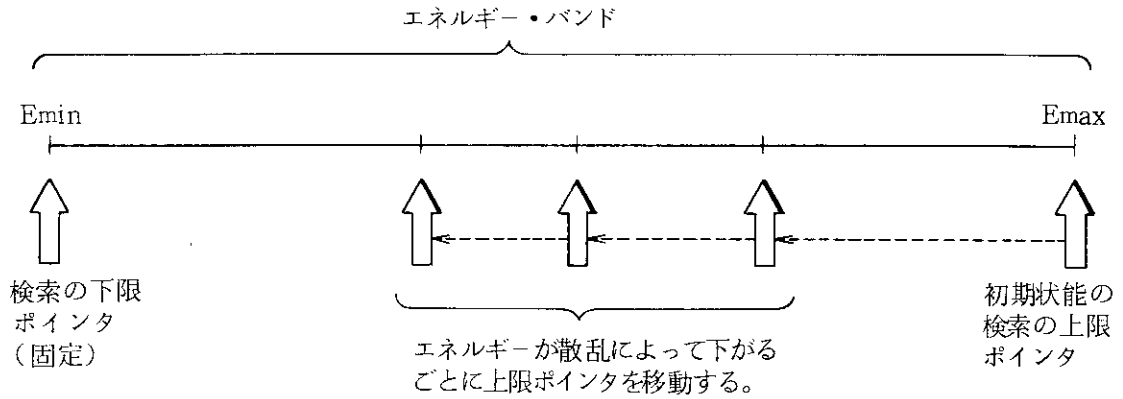


Fig. 4.7 The search range of energy grid of cross section library (original version)

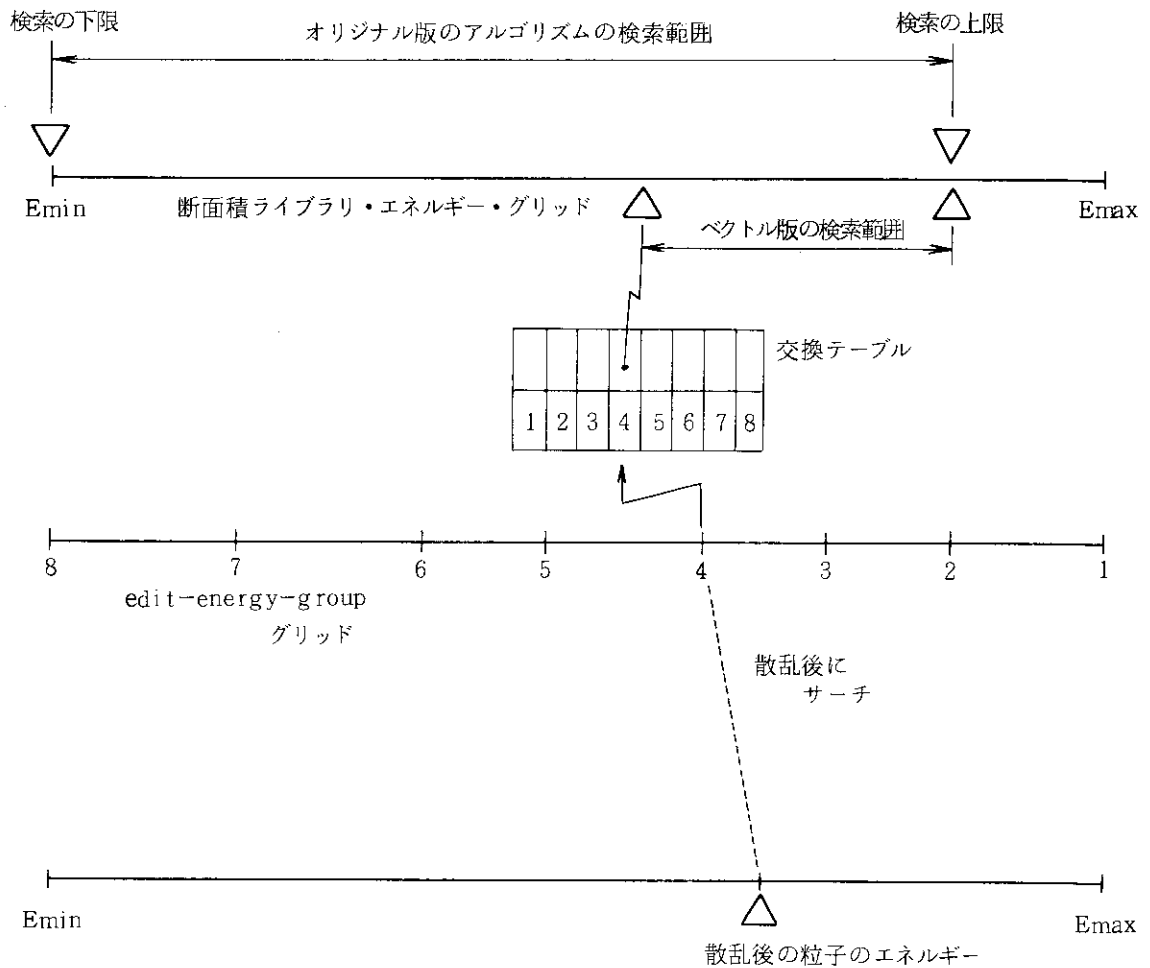


Fig. 4.8 The search range of energy grid of cross section library (vector version)

#### 4.3.3 衝突解析系のベクトル化

衝突解析系は Fig. 4.3 で示すように制御ルーチン REACT, 中性子束, 反応率評価用ルーチン群, 及び散乱解析用ルーチン群から構成される。ここで, ベクトル化されるルーチンは REACT, FLUXX, GG, 及び ELAST である。このうち FLUXX と GG については他の部分で解説する。制御ルーチン REACT は次のような機能がある。

- ・衝突の評価
- ・分散低減のための中性子のスプリッティング, ロシアン・ルーレットの処理
- ・次世代中性子のための fission-site の保存
- ・散乱タイプの選択と処理
- ・吸収の処理

REACT では様々なオプションによる処理の選択から生じる IF 分岐が多い(実行文の 14% が IF 文)。ベクトル化は上位ルーチン(NUTRNG)から中性子の DO ループを引き込み, オリジナル版の IF 分岐は Fig. 4.9 に示すようにリストの作成に変更した。このリストの作成はオリジナル版の IF 分岐と比較すると演算量がふえ処理時間が増加する。ベクトル化された制御ルーチン REACT はリスト・ベクトル作成時の処理時間増加のために, 全体ではベクトル化による処理時間の短縮は無い。しかし, 作成されたリストは下位ルーチン ELAST のベクトル化のために欠くことの出来ないものである。

非熱領域の弾性散乱解析ルーチン ELAST では上位ルーチン REACT で決定された target 核種及び粒子リストに従って散乱の解析が行われる。ELAST は散乱角の分布テーブルの選択, 等方・非等方散乱の選択及び散乱後の状態の決定の 3 つの処理から構成される。上位ルーチンから中性子のループを引き込むことによってベクトル化した。しかし, 散乱角の分布テーブルの選択はサーチであるためベクトル化はしなかった。

Table 4.3 Subroutines in collision analyzer system of neutron

ルーチン名	処 置	処 理 概 要
REACT1 (REACT)	vectorize	反応解析系の制御ルーチン
FLUXX1 (FLUXX)	modified	粒子の飛程による中性子束, 反応率の評価を行う
GGONE (GG )	vectorize	衝突の位置が CENTRAL RATE を登録すべき領域内に有るかどうかを判断する
SPECT1 (SPECTL)	use interfaced original one	CENTRAL CAPTURE RATE, CENTRAL FISSION RATE を登録する
INELS1 (INELAS)	use interfaced original one	非熱領域の非弾性散乱を解析し散乱後の粒子の状態を決定する
FREGAS	use interfaced original one	free-gas モデルの散乱を解析し散乱後の粒子の状態を決定する
XTAL1 (XTAL)	use interfaced original one	熱領域の弾性散乱を解析し散乱後の粒子の状態を決定する
SABSC1 (SABSCT)	use interfaced original one	熱領域の非弾性散乱 (thermal collision) を解析し散乱後の粒子の状態を決定する
GTISO	in-line expansion by FORTRAN	等方的な三次元空間上の単位ベクトルを生成する
ELAST1 (ELAST)	vectorize	非熱領域の弾性散乱を解析し散乱後の粒子の状態を決定する
LABX	use interfaced original one	格子内の局所的な座標を外部の大域的な座標に変換する

・括弧内のルーチン名は実際の処理用サブ・エントリ名であり, この場合メイン・エントリは引数のベース・アドレスの設定の機能しか無い。

```
IF(STOT.LE.0.0) GO TO 800
IF(STOT.GE.SIGMAT) GO TO JBR13,(400,450)
SIGSUR=STOT/SIGMAT
```

(オリジナル版)



```
*VOCL LOOP,NOVREC
1-----DO 4360 KI% = 1 , N2324%
1          K% = L2324%(KI%)
1 2-----IF( STOT%(K%).LE.0.0 )          THEN
1 2          N800% = N800% + 1
1 2          L800%(N800%) = K%
1 +-----ELSE
1 2 3-----IF( STOT%(K%).GE.SIGMA%(K%) )    THEN
1 2 3          N4045% = N4045% + 1
1 2 3          L4045%(N4045%) = K%
1 2 +-----ELSE
1 2 3          SIGSU%(K%)=STOT%(K%)/SIGMA%(K%)
1 2 3          N2027% = N2027% + 1
1 2 3          L2027%(N2027%) = K%
1 2 +-----ENDIF
1 +-----ENDIF
+-----4360 CONTINUE
```

(ベクトル版)

Fig. 4.9 An example of vectorization of IF branch for scattering and absorption of neutrons

#### 4.3.4 幾何形状処理系のベクトル化

幾何形状処理系は制御ルーチンG1と下位の実処理ルーチンGGから構成される。G1はFig. 4.10で示すように(a) code-zoneの検索、(b) 衝突又は boundary-crossの判定、(c) 過去の履歴を使った boundary-cross-zoneの検索、(d) 体系を構成する全ての code-zoneからの boundary-cross-zoneの検索及び、(e) 検索された code-zoneのチェックの五つの処理部分から構成されている。

ベクトル化は基本的には(a)-(e)に関しては中性子のDOループを上位ルーチンから引き込むことによってなされた。最後の(e)に関しては、単に中性子に関するDOループを形成することでベクトル化が可能であるが、(a)-(d)の処理に関しては中性子のDOループの制御構造を加える以外にも構造の変更が必要であった。

(a)、(c)、(d)の処理は条件を満足する zoneの検索であり、(b)の処理は粒子の飛程の始点から zoneの境界を横切る点までの距離の計算である。これらの処理はまず zoneを構成する body毎に計算を行い、それらの結果を総合して zoneに対する判断をすることによって行われる。zone毎に構成 bodyが異なるため、zone単位で中性子のリストを作成し、このリストを使って

body の処理をしなければならない。

(a), (c), (d)の処理は条件を満足する zone の検索であるために, Fig. 4.11 の(a)で示すように最外の zone の DO ループで検索対象の code-zone を与え, 内側の zone-body- 中性子の三重 DO ループの処理で体系内の中性子に対し, この zone が条件を満足する code-zone かどうかをチェックする。このため, zone-zone-body- 中性子の四重 DO ループの構造となっている。

(b)の処理は中性子が属する zone を構成するすべての body に対して飛程が body を横切る点までの距離を計算し, それらの結果から飛程が zone の境界を横切る点までの距離を求めている。このため Fig. 4.11 の(b)のような zone-body- 中性子の三重 DO ループの構造となる。

また, (c)の検索は, オリジナル版では Fig. 2.5 に示すようなリスト構造のデータを使用して行っていたが, これを Fig. 4.12 で示すような隣接 zone の数と隣接 zone 番号のテーブルの形式に改めた。

GG は各 body 要素に対して中性子の存在する位置から body の入口又は出口までの距離を計算する。Fig. 4.13 のオリジナル版の GG の制御構造を示す。VIM で許容されている body タイプに対応する処理部分が並んでいる。このため, ベクトル化は上位の制御ルーチンから中性子に関する DO ループを内部に引き込むことによってなされた。上位の制御ルーチンから渡される処理すべき中性子のリストは zone 毎に分類してあり, zone を構成する body 毎に起動される GG 内では, 渡された body に対するデータを対象としてリスト中の全ての中性子に対し入口点又は出口点までの距離の計算を行う。

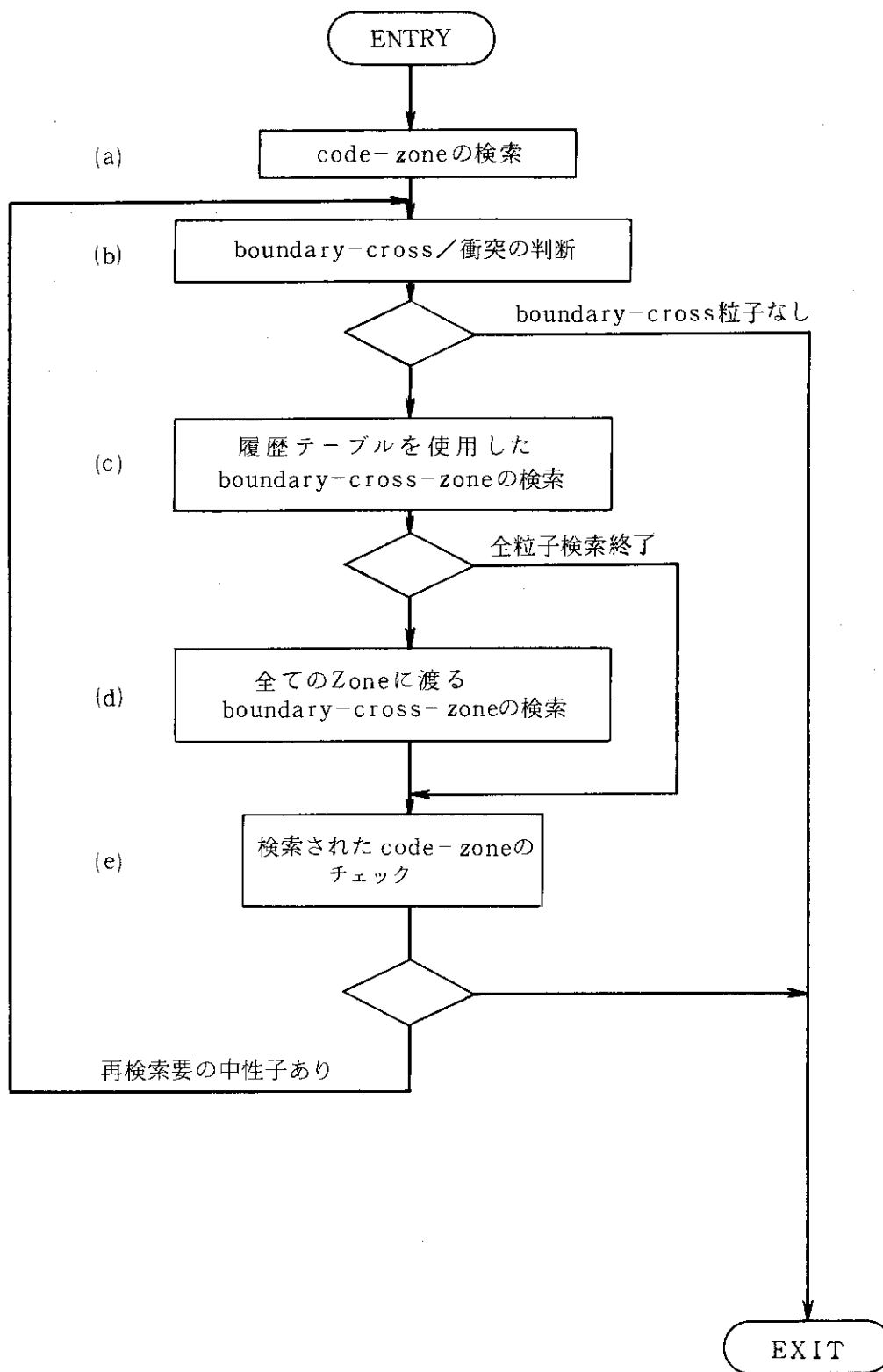


Fig. 4.10 The control structure of geometrical processing routine G1

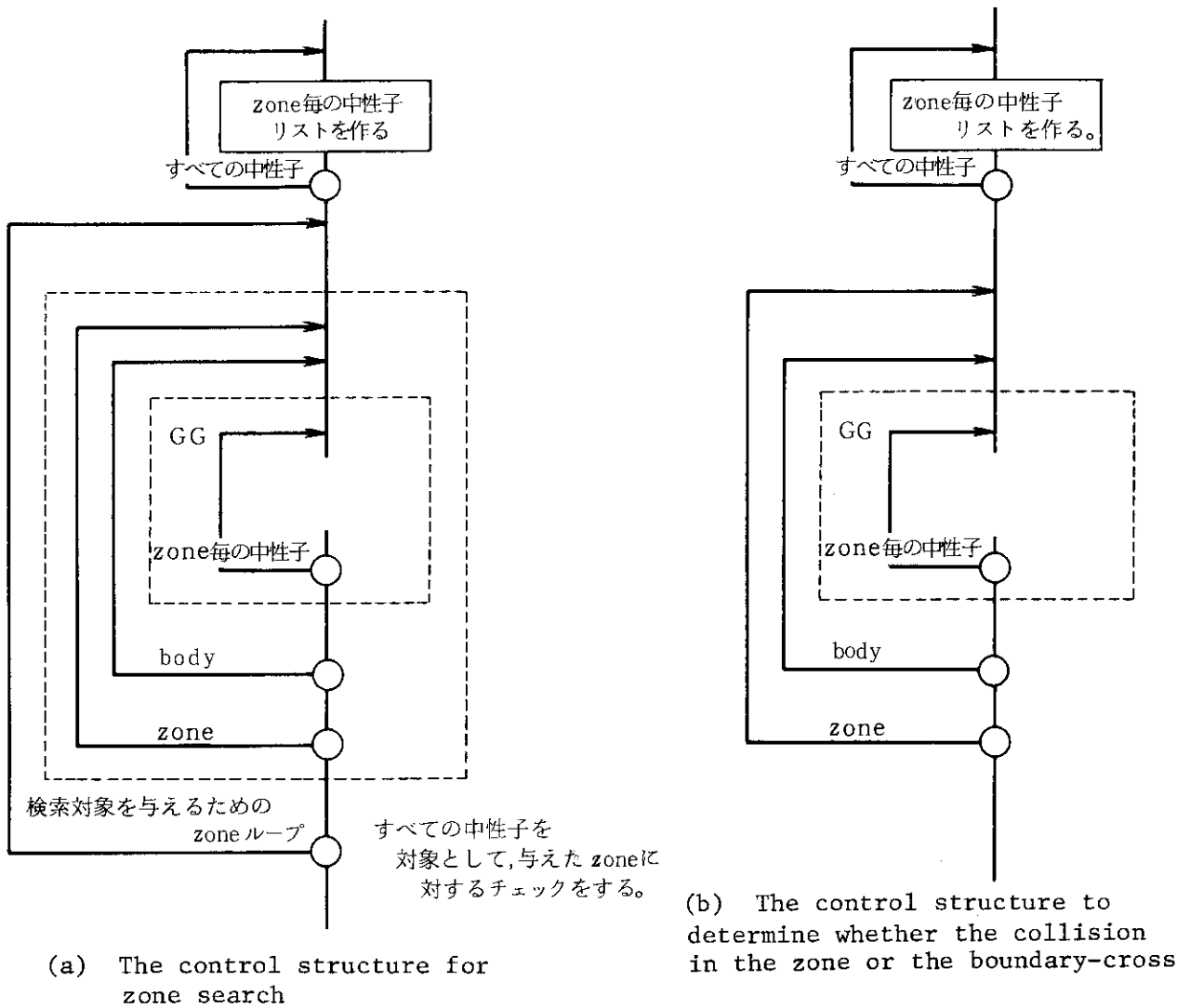


Fig. 4.11 The control structure of vectorized geometrical processing routine G1

zone 1	3	隣接 zone 番号 1	隣接 zone 番号 2	隣接 zone 番号 3
zone 2	2	隣接 zone 番号 1	隣接 zone 番号 2	
zone 3	1	隣接 zone 番号 1		

検索された隣接 zone 数

隣接 zone 番号テーブル

Fig. 4.12 The data structure of adjacent zone number table for vector version

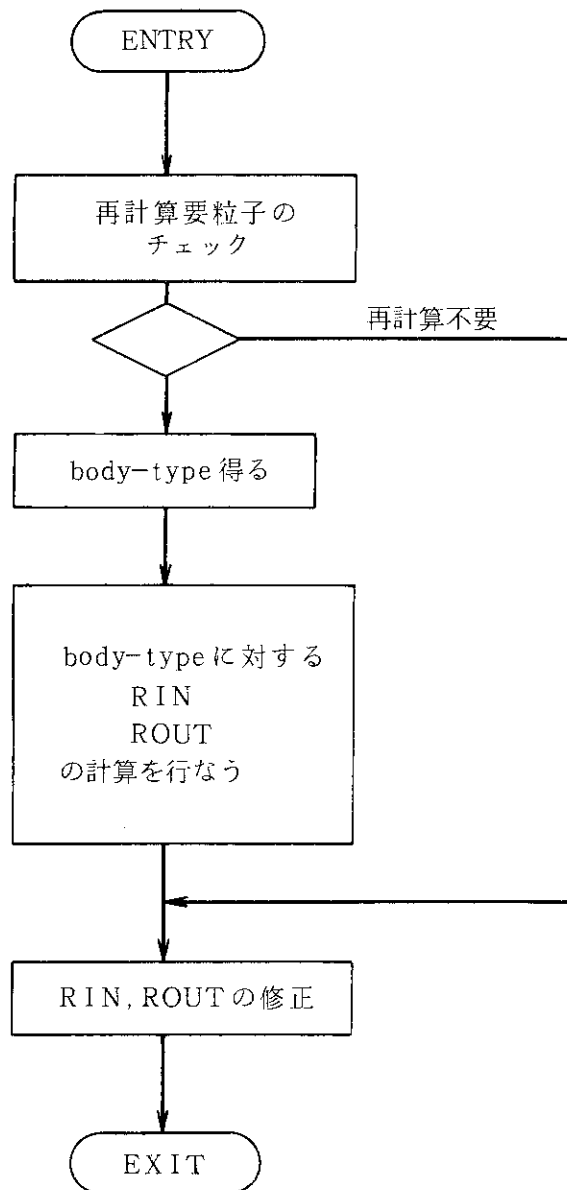


Fig. 4.13 The control structure of geometrical routine GG

#### 4.3.5 評価系(Estimator systems)に対する処置

中性子束、各種反応率の評価は中性子に関連する属性を zone 毎、エネルギー毎に累積することによっておこなわれる。上位ルーチンから中性子のループが引き込まれた。中性子の DO ループを使って累積用の領域に足し込みを行う場合、同じ zone 番号と同じエネルギー群番号を持つ中性子が複数個存在するため、回帰的なデータの参照又は定義をとまなう演算となり、ベクトル演算することができない。このため、中性子束、各種反応率等の評価は全てスカラ演算で処理している。

## 5. ベクトル化の結果と問題点

Table 5.1 にベクトル化版／オリジナル版の処理時間を示す。ベクトル版のスカラ・モードはオリジナル版と比較して単純形状の入力データで1.4倍、複雑形状の入力データで1.5倍処理時間が増加している。これは、ベクトル化のための制御構造のオーバーヘッドのために生じるものである。ベクトル演算を使用し、これらを相殺することによって単純形状の入力データに関してはオリジナル版に対する処理速度比1.39倍、複雑形状の入力データに関して1.13倍の処理速度の向上を得た。尚、ベクトル化率は単純形状の入力データの場合で約67%、複雑形状の入力データの場合で約51%であった。ベクトル版の計算精度の検証のためにTable 5.2に実効増倍係数と標準偏差を示す。オリジナル版とベクトル版の実行結果の違いは乱数列が異なることや演算順序が異なるためによるものである。また、実行時に必要な主記憶容量は単一バンド断面積ライブラリ使用時に、オリジナル版で1.6メガ・バイト又ベクトル版で7メガ・バイトであった。増加量のほとんどの部分はベクトル化のために配列化されたデータ・エリアである。

Fig. 5.1 および Fig. 5.2 に VP-100 上での全体の処理時間に対する各処理系別消費時間と DOループのカテゴリー別消費時間の割合を示す。カテゴリー別消費時間のスカラ演算の中でサーチ及び、body タイプごとの粒子の振り分け処理に対応する geometric branch の処理の二つが連続エネルギー・モンテカルロ・コードにおいて大きなウェイトを占めていることがわかる。

### (1) インデックス・サーチ

インデックス・サーチの割合が高いのは連続エネルギー・モンテカルロ・コード固有のものである。ベクトル版ベクトル・モードでの全体の処理時間に占める割合は単純形状の場合で39.8%、複雑形状の場合で11.4%に及ぶ。インデックス・サーチの消費時間の70%が断面積の作成で、30%が散乱の解析などで使用される。インデックス・サーチ高速化の方法が見つければ、それは連続エネルギー・モンテカルロ・コード高速化に大いに寄与するものと思われる。インデックス・サーチの高速化の一手法として APPENDIX A に示すような浮動小数点データをキーとしたハッシュ・サーチの適用を試みた。断面積ライブラリのエネルギー・グリッドは変更が不可能なためサーチの結果を保証するためには浮動小数点データの仮数部の全ビット(24ビット)を対象としたハッシュ・テーブルを作らなければならない。この場合のハッシュ・テーブルの大きさは16進の1デカードあたり約67メガ・バイト( $2^{24} \times 4$ )となるため、ハッシュ・サーチを使用出来なかった。

Table 5.1 Summary of the execution time of original and vectorized codes

	オリジナル版		ベクトル版 スカラモード		ベクトル版 ベクトルモード	
	単純形状	複雑形状	単純形状	複雑形状	単純形状	複雑形状
処理時間	103.1	159.1	144.8	242.2	73.8	139.8
時間比	1.0	1.0	1.4	1.5	0.72	0.88
速度比	1.0	1.0	0.71	0.67	1.39	1.13

(処理時間の単位は秒)

Table 5.2 Effective multiplication factor ( $k_{eff}$ ) and standard distribution (S.D.) of original and vectorized codes

	単純形状		複雑形状	
	$k_{eff}$	S.D.	$k_{eff}$	S.D.
オリジナル版	0.997089	0.15421E-1	0.711156	0.111738E-1
ベクトル版	1.00729	0.89009E-2	0.719237	0.126967E-1

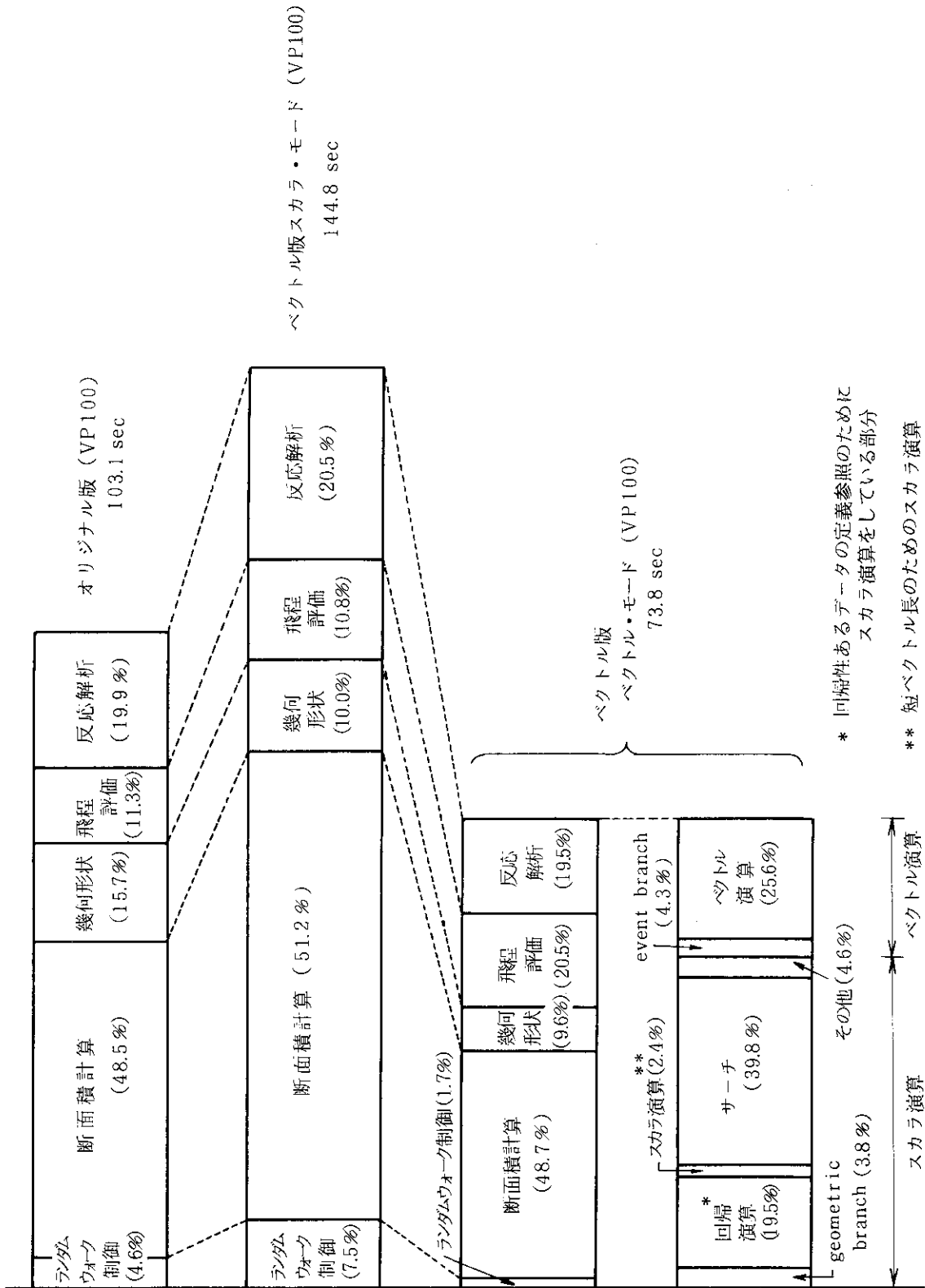


Fig. 5.1 CPU time classified by the calculation type for simple geometric data

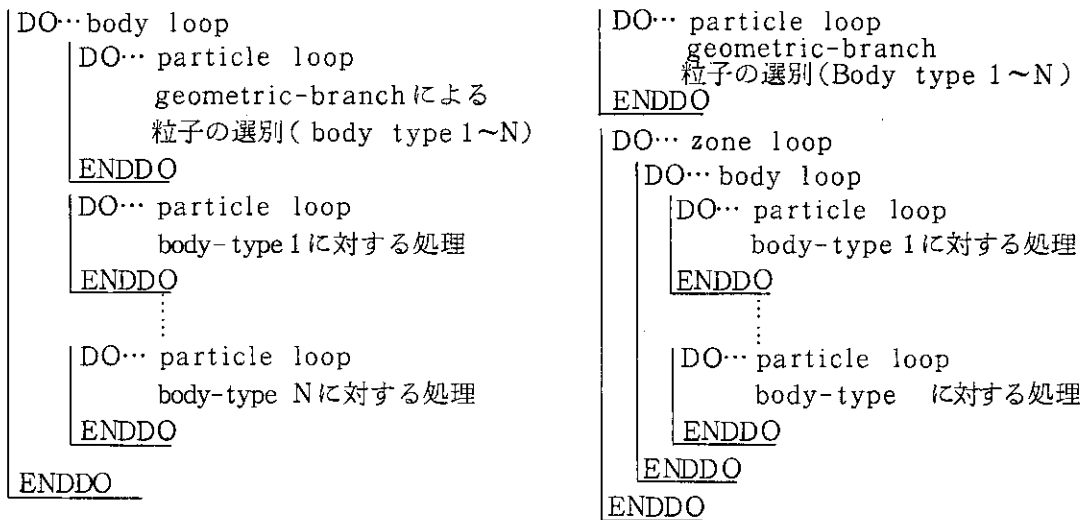


(2) 幾何形状処理系の中性子の分類処理について。

zone 毎に構成 body の数や種類及び、分まれる核種等が異なるために中性子を分類しなければならぬ。この中性子の分類は幾何形状に関する量によって行われる。この分類処理を geometric branch による分類処理と呼ぶ。

幾何形状処理系全体のベクトル化による処理効率を向上させるためには geometric branch による分類処理と body type 毎の処理に対する処理のバランスを考える必要がある。VIMでは geometric branch による分類処理に要する処理時間を減少させる工夫をした。すなわち、Fig.5.3の(b)に示すように zone 番号を使用した geometric branch による分類処理を行い、body type に対する処理の部分では同一 zone 毎に同一 body に対する中性子の処理を行った。この方法の欠点は zone の数と body type 毎の処理のベクトル長が反比例の関係にあるため、zone の数が多い場合は性能向上が望めないことである。

一方、Fig. 5.3 の(a)に KENO IV のベクトル化の際にとった方式を示す。この方式では body type で geometric branch による分類処理を行い、body type による処理では同一 body に対する処理を一度に行う。この方式では geometric branch に要する時間は比較的大きいが、各 body の処理時のベクトル長は zone の個数に依存しないため zone の数が増加した場合の性能低下はない。geometric branch の処理時間を減少させるために幾何形状分類用のハードウェアである geometric pipeline の提案がなされている。<sup>1)</sup>



(a) geometric-branch がループのネストの深い所にある。

(b) geometric-branch が浅い所にある。

↓↓  
分類コストは大

↓↓  
分類コストは(a)に比べて小

Fig. 5.3 Two type of control structure for geometric branch

(3) 減少するベクトル長に対する問題

同じ処理内容のDOループを二つ用意し、一方をスカラ処理のためDOループ、他方をベクトル処理用のDOループとしてベクトル長に応じて使い分けた。スカラ/ベクトルきりわけベクトル長のパラメータ・サーベイを断面積処理ルーチンCROSSで行った。全体の処理時間の1%以下の処理効率の改善しか見られなかったため実際には使用しなかった。

## 6. お わ り に

ベクトル化作業の結果，単純形状の入力データのもとで 1.39倍，複雑形状の入力データのもとで 1.13倍の処理速度の改善が見られた。大幅な性能向上の見込めない原因としては次のようなものが挙げられる。

- ・ VIMはスカラ計算機上で最適化されているため，ベクトル版改造時にプログラム内に持ち込まれるオーバーヘッドが大きい。VIMの場合ベクトル版スカラ・モードの実行時間からわかるように40%～50%の実行時間の増加になる。
- ・ 幾何形状分類，インデックス・サーチ，回帰演算，などのベクトル演算器による高速化の難しい処理が多い。

インデックス・サーチ，寄与評価のための回帰演算はユーザー側でのアルゴリズムの改善等で克服可能な場合もある。しかし，ユーザー側での対処のみでは限界があり，ベクトル・コンパイラ等のシステム・ソフトウェア及びハードウェアの改善が待たれる。モンテカルロ法のベクトル化によって生じた問題は現在のベクトル化コンパイラ，及びハード・ウェア等のベクトル処理システムの改善に対する方針を与えていると考えられる。

### 謝 辞

本プログラム及び単純形状の入力データの提供，並びに断面積ライブラリ処理プログラムのエラー修正に協力して下さった原子炉システム研究室 森 貴正氏に感謝します。

また，ベクトル化に対する貴重な助言をして下さった(財)原子力データセンター 鈴木忠和氏，栗田豊氏に感謝します。

富士通(株)の牧野光弘氏には本報告の作成についての種々のコメントをいただきました，感謝します。

### 参考文献

1. Kiyoshi Asai et al.,  
"Vectorization of the KENO-IV Code", NUCLEAR SCIENCE AND ENGINEERING, 92, 298-307 (1986).
2. F. B. BROWN,  
"VECTORIZED MONTE CARLO METHOD FOR REACTOR ANALYSIS",  
Proc. of a Topical Meetings, Advance in reactor Computations, Vol. 1,  
PP. 108-123, Salt Lake City (1983).
3. Forrest B. Brown and William R. Martin :  
"MONTE CARLO METHODS FOR RADIATION TRANSPORT ANALYSIS  
ON VECTOR COMPUTERS",  
Progress in Nuclear Energy, Vol. 14, No. 13. pp. 269-299, 1984.

## 6. お わ り に

ベクトル化作業の結果、単純形状の入力データのもとで1.39倍、複雑形状の入力データのもとで1.13倍の処理速度の改善が見られた。大幅な性能向上の見込めない原因としては次のようなものが挙げられる。

- ・ VIMはスカラ計算機上で最適化されているため、ベクトル版改造時にプログラム内に持ち込まれるオーバーヘッドが大きい。VIMの場合ベクトル版スカラ・モードの実行時間からわかるように40%～50%の実行時間の増加になる。
- ・ 幾何形状分類、インデックス・サーチ、回帰演算、などのベクトル演算器による高速化の難しい処理が多い。

インデックス・サーチ、寄与評価のための回帰演算はユーザー側でのアルゴリズムの改善等で克服可能な場合もある。しかし、ユーザー側での対処のみでは限界があり、ベクトル・コンパイラ等のシステム・ソフトウェア及びハードウェアの改善が待たれる。モンテカルロ法のベクトル化によって生じた問題は現在のベクトル化コンパイラ、及びハード・ウェア等のベクトル処理システムの改善に対する方針を与えていると考えられる。

### 謝 辞

本プログラム及び単純形状の入力データの提供、並びに断面積ライブラリ処理プログラムのエラー修正に協力して下さった原子炉システム研究室 森 貴正氏に感謝します。

また、ベクトル化に対する貴重な助言をして下さった(財)原子力データセンター 鈴木忠和氏、栗田豊氏に感謝します。

富士通(株)の牧野光弘氏には本報告の作成についての種々のコメントをいただきました、感謝します。

### 参考文献

1. Kiyoshi Asai et al.,  
"Vectorization of the KENO-IV Code", NUCLEAR SCIENCE AND ENGINEERING, 92, 298-307 (1986).
2. F. B. BROWN,  
"VECTORIZED MONTE CARLO METHOD FOR REACTOR ANALYSIS",  
Proc. of a Topical Meetings, Advance in reactor Computations, Vol. 1,  
PP. 108-123, Salt Lake City (1983).
3. Forrest B. Brown and William R. Martin :  
"MONTE CARLO METHODS FOR RADIATION TRANSPORT ANALYSIS  
ON VECTOR COMPUTERS",  
Progress in Nuclear Energy, Vol. 14, No. 13. pp. 269-299, 1984.

## 6. お わ り に

ベクトル化作業の結果，単純形状の入力データのもとで1.39倍，複雑形状の入力データのもとで1.13倍の処理速度の改善が見られた。大幅な性能向上の見込めない原因としては次のようなものが挙げられる。

- ・ VIMはスカラ計算機上で最適化されているため，ベクトル版改造時にプログラム内に持ち込まれるオーバーヘッドが大きい。VIMの場合ベクトル版スカラ・モードの実行時間からわかるように40%～50%の実行時間の増加になる。
- ・ 幾何形状分類，インデックス・サーチ，回帰演算，などのベクトル演算器による高速化の難しい処理が多い。

インデックス・サーチ，寄与評価のための回帰演算はユーザー側でのアルゴリズムの改善等で克服可能な場合もある。しかし，ユーザー側での対処のみでは限界があり，ベクトル・コンパイラ等のシステム・ソフトウェア及びハードウェアの改善が待たれる。モンテカルロ法のベクトル化によって生じた問題は現在のベクトル化コンパイラ，及びハード・ウェア等のベクトル処理システムの改善に対する方針を与えていると考えられる。

### 謝 辞

本プログラム及び単純形状の入力データの提供，並びに断面積ライブラリ処理プログラムのエラー修正に協力して下さった原子炉システム研究室 森 貴正氏に感謝します。

また，ベクトル化に対する貴重な助言をして下さった(財)原子力データセンター 鈴木忠和氏，栗田豊氏に感謝します。

富士通(株)の牧野光弘氏には本報告の作成についての種々のコメントをいただきました，感謝します。

### 参考文献

1. Kiyoshi Asai et al.,  
"Vectorization of the KENO-IV Code", NUCLEAR SCIENCE AND ENGINEERING, 92, 298-307 (1986).
2. F. B. BROWN,  
"VECTORIZED MONTE CARLO METHOD FOR REACTOR ANALYSIS",  
Proc. of a Topical Meetings, Advance in reactor Computations, Vol. 1,  
PP. 108-123, Salt Lake City (1983).
3. Forrest B. Brown and William R. Martin :  
"MONTE CARLO METHODS FOR RADIATION TRANSPORT ANALYSIS  
ON VECTOR COMPUTERS",  
Progress in Nuclear Energy, Vol. 14, No. 13. pp. 269-299, 1984.

4. 鈴木忠和 他,  
“モンテカルロ法による組み合わせ幾何形状用中性子輸送解析コード MORSE-CG のベクトル化”, 1985年9月, 私信
5. L. J. Milton,  
“VIM USER'S GUIDE”,  
Applied Physics Division, Argonne National Laboratory, June 24, 1981,  
private communication.
6. M. B. Emmett,  
“THE MORSE MONTE CARLO RADIATION TRANSPORT CODE  
SYSTEM”. ORNL-4972, FEBRUARY 1975,
7. 富士通株式会社,  
“FACOM OS IV/F4 MSP FORTUNE 使用手引書 V10用”, 78SP-5360.

## APPENDIX A

- ・ 浮動小数点データを利用したエネルギー・グリッドのハッシュ・サーチについて

Fig. A-1に示すようなハッシュ関数を実現することによってサーチの高速化の可能性を調査した。ここではまず、キーとして使用する浮動小数点データの指数部及び仮数部を整数型データとしてそれぞれ取り出す。指数部からハッシュ・テーブル上でのデカードの先頭アドレスを得る。この先頭アドレスに整数型データとして取り出された仮数部を足し合わせることによって予めハッシュ・テーブル上に設定されているエネルギー・グリッドのインデックスを得ている。この手法の欠点は仮数部の使用可能なビット長が使用可能なメモリの量から制限されてしまうことである。このため使用可能なエネルギー・グリッドは切り捨てられた仮数部を持った浮動小数点データで表現可能な値でなければサーチに不都合を生じる点にある。ハッシュ・サーチの性能評価のために edit-energy-group のサーチに適用してテストを行った。この調査に於いて、仮数部のビット長は12ビットで edit-energy-group は仮数部12ビットの浮動小数点データで表現が可能のように再定義した。この再定義は edit-energy-group がユーザー定義であるために可能となっている。Fig. A-3に計測部分のソースを示す。

Fig. A-2に反応解析系の制御ルーチンであるREACT1の散乱後の edit-energy-group の検索に浮動小数点データをキーとしたハッシュ・サーチを適応した際のベクトル長の変化に対するリニア・サーチを基準としたハッシュ・サーチの処理速度の比を示す。ハッシュ・サーチは十分な優位性があることがわかる。

しかし、Table A-1に示すように断面積ライブラリのエネルギー・グリッドの数は非常に多く且つ、ユーザーが変更するのは困難であるためサーチの結果を保証するためには仮数部の全ビット即ち24ビットを使用しなければならない。このため、断面積ライブラリのエネルギー・グリッド・サーチにこの手法を適用することは不可能である。

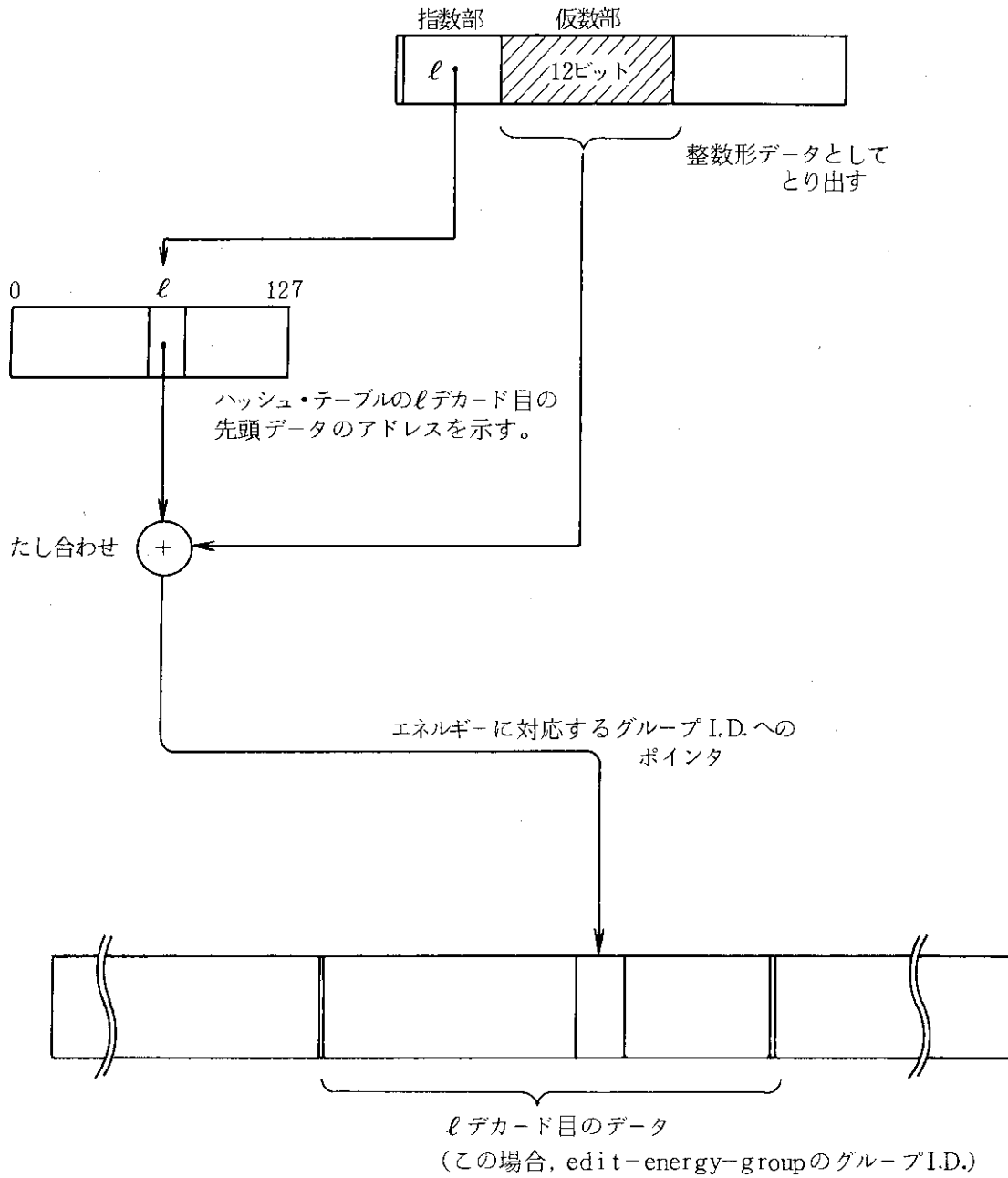


Fig. A-1 Implimentation of hash function needed for cross section energy grid search

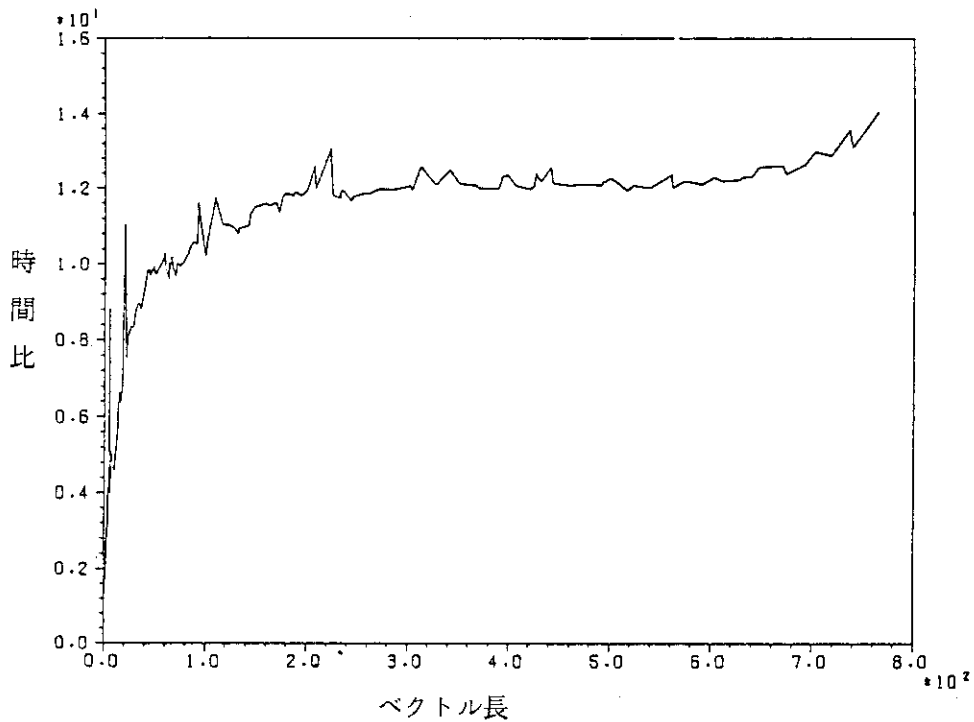


Fig. A-2 The ratio of execution time for hash search to that for linear search

```

C
C   LINEAR SEARCH.
C
CALL CLOCK(T1%,2,2)
*VOCL LOOP,NOVREC
DO 4590 KI% = 1 , N7978%
  KY = L7978%(KI%)
  NFLAG%(KY)=2
C#
DO 980 I=1,NGPX
  IF(ENERG%(KY).GT.EGROUP(I)) GO TO 990
V
V   980 CONTINUE
V       I=NGPX
V       990 CONTINUE
V           IGR%(KY) = 1
4590 CONTINUE
CALL CLOCK(T2%,2,2)
TT% = T2% - T1% - AVE%

```

リニア・サーチ

```

C
C
C
CALL CLOCK(T1%,2,2)
V
V   DO 2010 KI% = 1 , N7978%
V       KY = L7978%(KI%)
V       IWK = IAND(MASKE,INERG%(KY))
V       IEXP = ISHFT(IWK,-24)
V       INC = IAND(MASKM,INERG%(KY))
V       INC2 = ISHFT(INC,-12)
V       IADRS = ISTRT%(IEXP) + INC2
V       IGR2%(KY) = INDEX%(IADRS)
V
V   2010 CONTINUE
CALL CLOCK(T2%,2,2)
TT2% = T2% - T1% - AVE%

```

ハッシュ・サーチ

⊕ DATA MASKE /Z7F000000/  
 DATA MASKM /Z00FFF000/  
 EQUIVALENCE (ENERG%, INERG%)

Fig. A-3 The part of source program for measurement of execution time of both the search by hash and the linear search

Table A-1 Distribution of energy grid point for each nuclides in the VIM cross section library

エネルギー (eV)	グリッド・ポイント数			
	Pu-240	Pu-241	U-235	U-238
1.0E+7 以上	2	2	2	2
1.0E+7 ~5.0E+6	11	20	51	25
5.0E+6 ~1.0E+6	21	28	47	68
1.0E+6 ~5.0E+5	11	11	36	31
5.0E+5 ~1.0E+5	21	19	35	36
1.0E+5 ~5.0E+4	9	9	91	20
5.0E+4 ~1.0E+4	15	15	92	21
1.0E+4 ~5.0E+3	5	6	6	5
5.0E+3 ~1.0E+3	5160	14	15	10254
1.0E+3 ~5.0E+2	1366	5	5	2470
5.0E+2 ~1.0E+2	1341	14	14	2541
1.0E+2 ~5.0E+1	277	6	1258	410
5.0E+1 ~1.0E+1	269	1139	2075	377
1.0E+1 ~5.0E+0	16	155	322	125
5.0E+0 ~1.0E+0	122	252	294	27

## APPENDIX B

結果の検証を可能にするために、断面積ライブラリ作成時の入力データとVIMの実行時に必要な入力データを示す。

## ① 断面積ライブラリ作成時に必要な入力データ

- ・ XSEEDIT 実行時に必要な入力データ

```

      12      2      0      0      1      0      0      0      0
10300 20300 30300 40300 50300130300210300220300230300240300250300260300

```

- ・ FILEONE 実行時に必要な入力データ

```

      1
010300PU240      22      1
020300PU241      23      1
030300U-235      24      1
040300U-238      25      1
050300PU239      26      1
130300AM241      27      1
210300CR          28
220300NI          29
230300FE          30
240300AL-27      31
250300NA-23      32
260300O-16       33

```

- ・ BANDIT 実行時に必要な入力データ

```

      12      2387000      1      0
10300 20300 30300 40300 50300130300210300220300230300240300250300260300

```

但し、単一エネルギーバンドの断面積ライブラリを作成するためには BANDIT のエラーを修正しなければならない。

② VIM 実行のための入力データ (単純形状)

```

1123456789AB      FCA X-1  URANIUM BLANKET * HOMOGENEOUS COMP. * CYLINDE
      10          3          0          0          0          0
      900         900         10         0          2          0
      1           1          0          0          0          0
      12          3          3          70         5          900
      3500.       10.0       275.0       1.0       1. E-5       0
      0.95        0.
1      0          0          0          0          0          0
10300 20300 30300 40300 50300130300210300220300230300240300250300260300
3 30300 40300 50300
3 30300 40300 50300
0
CYL 1 1 0.0 0.0 -2.5 5.00 3.0
CYL 2 0.0 0.0 -25.4 50.8 28.7125
CYL 3 0.0 0.0 -55.9 111.8 61.738
CYL 4 0. 0.0 0.0 -57.0 114.0 77.0
CYL 5 0.0 0.0 -60.0 120.0 80.0
END
1      +1
2      -1      +2
3      +3      -2
4      +4      -3
5      +5      -4
END
1      2      3      4
5
1 1 1 1 2 1 1 1 3 2 0 2
4 3 0 3 5 0 0 -1
10300 20300 30300 40300 50300130300210300220300230300240300250300260300
30300 40300210300220300230300
210300220300230300
1.8432 -4 9.8811 -6 7.39588 -4 5.2067 -3 2.0911 -3 7.4214 -6
3.6152 -3 1.66396 -3 1.33969 -2 5.7219 -3 7.6564 -3 1.2777 -2
8.4000 -5 4.0175 -2 1.827 -3 7.96 -4 6.652 -3
1.2417 -3 5.4128 -4 4.5208 -3
7.788 +6 6.0653+6 4.7237 +6 3.67880+6 2.8650+6 2.23130+6
1.7377 +6 1.3534+6 1.05400+6 8.20850+5 6.3928+5 4.97870+5
3.87740+5 3.0197+5 2.35180+5 1.83160+5 1.4264+5 1.11090+5
8.65170+4 6.7379+4 5.24750+4 4.08680+4 3.1828+4 2.47880+4
1.93050+4 1.5034+4 1.17090+4 9.11880+3 7.1017+3 5.53080+3
4.30740+3 3.3546+3 2.61260+3 2.03470+3 1.5846+3 1.23410+3
9.61120+2 7.4852+2 5.82950+2 4.54000+2 3.5358+2 2.75360+2
2.14450+2 1.6702+2 1.30070+2 1.01300+2 7.8893+1 6.14420+1
4.78510+1 3.7267+1 2.90230+1 2.26030+1 1.7603+1 1.37100+1
1.06770+1 8.3153+0 6.47600+0 5.04350+0 3.9279+0 3.05900+0
2.38240+0 1.8554+0 1.44500+0 1.12540+0 8.7642-1 6.82560-1
5.31580-1 4.1399-1 3.22420-1 1.00000-5

```

終り



③ VIM の実行のための入力データ(複雑形状)

```

1123456789AB      FCA X-1  URANIUM BLANKET * INHOMOGENIOUS COMP.*CYLINDE
      10          3          0          0          0          0
      900         900         10         0          2          0
      1           1          0          0          0          0
      12          3         22         70         24         900
      3500.        10.0       275.0       1.0       1. E-5       0
      0.95         0.
1      0      0      0      0      0      0      0      0      0      0
10300 20300 30300 40300 50300 130300 210300 220300 230300 240300 250300 260300
      3 30300 40300 50300
      3 30300 40300 50300
      0
RPP  1      -17.5      -7.5      25.0      35.0      -25.4      25.4
RPP  2      - 5.0       5.0      25.0      35.0      -25.4      25.4
RPP  3       7.5      17.5      25.0      35.0      -25.4      25.4
RPP  4     -30.0     -20.0       7.5      17.5      -25.4      25.4
RPP  5     -17.5     -7.5       7.5      17.5      -25.4      25.4
RPP  6      - 5.0       5.0       7.5      17.5      -25.4      25.4
RPP  7       7.5      17.5       7.5      17.5      -25.4      25.4
RPP  8      20.0      30.0       7.5      17.5      -25.4      25.4
RPP  9     -30.0     -20.0       5.0      15.0      -25.4      25.4
RPP 10     -17.5     -7.5       5.0      15.0      -25.4      25.4
RPP 11      -5.0       5.0      -5.0       5.0      -25.4      25.4
RPP 12       7.5      17.5     -5.0       5.0      -25.4      25.4
RPP 13      20.0      30.0     -5.0       5.0      -25.4      25.4
RPP 14     -30.0     -20.0    -17.5     -7.5      -25.4      25.4
RPP 15     -17.5     -7.5    -17.5     -7.5      -25.4      25.4
RPP 16      -5.0       5.0    -17.5     -7.5      -25.4      25.4
RPP 17       7.5      17.5    -17.5     -7.5      -25.4      25.4
RPP 18      20.0      30.0    -17.5     -7.5      -25.4      25.4
RPP 19     -17.5     -7.5    -30.0    -20.0      -25.4      25.4
RPP 20      -5.0       5.0   -30.0    -20.0      -25.4      25.4
RPP 21       7.5      17.5   -30.0    -20.0      -25.4      25.4
CYL  22       0.0       0.0   -55.9     111.8     61.738
CYL  23       0.0       0.0   -57.0     114.0      77.0
CYL  24       0.0       0.0   -60.0     120.0      80.0
END
      1          +1
      2          +2
      3          +3
      4          +4
      5          +5
      6          +6
      7          +7
      8          +8
      9          +9
     10         +10
     11         +11
     12         +12
     13         +13

```

続く

14	+14								
15	+15								
16	+16								
17	+17								
18	+18								
19	+19								
20	+20								
21	+21								
22	+22	-1	-2	-3	-4	-5	-6	-7	-8
	-9	-10	-11	-12	-13	-14	-15	-16	-17
	-18	-19	-20	-21					
23	+23	-22							
24	+24	-23							

END

1		2		3		4
5		6		7		8
9		10		11		12
13		14		15		16
17		18		19		20
21		22	1.232062E+6	23		24

1	1	1	1	2	2	1	1	3	3	1	1
4	4	1	1	5	5	1	1	6	6	1	1
7	7	1	1	8	8	1	1	9	9	1	1
10	10	1	1	11	11	1	1	12	12	1	1
13	13	1	1	14	14	1	1	15	15	1	1
16	16	1	1	17	17	1	1	18	18	1	1
19	19	1	1	20	20	1	1	21	21	1	1
22	22	0	2	23	23	0	3	24	0	0	-1

10300 20300 30300 40300 50300130300210300220300230300240300250300260300

30300 40300210300220300230300

210300220300230300

1.8432	-4	9.8811	-6	7.39588	-4	5.2067	-3	2.0911	-3	7.4214	-6
3.6152	-3	1.66396	-3	1.33969	-2	5.7219	-3	7.6564	-3	1.2777	-2
8.4000	-5	4.0175	-2	1.827	-3	7.96	-4	6.652	-3		
1.2417	-3	5.4128	-4	4.5208	-3						
7.788	+6	6.0653+6		4.7237	+6	3.67880+6		2.8650+6		2.23130+6	
1.7377	+6	1.3534+6		1.05400+6		8.20850+5		6.3928+5		4.97870+5	
3.87740+5		3.0197+5		2.35180+5		1.83160+5		1.4264+5		1.11090+5	
8.65170+4		6.7379+4		5.24750+4		4.08680+4		3.1828+4		2.47880+4	
1.93050+4		1.5034+4		1.17090+4		9.11880+3		7.1017+3		5.53080+3	
4.30740+3		3.3546+3		2.61260+3		2.03470+3		1.5846+3		1.23410+3	
9.61120+2		7.4852+2		5.82950+2		4.54000+2		3.5358+2		2.75360+2	
2.14450+2		1.6702+2		1.30070+2		1.01300+2		7.8893+1		6.14420+1	
4.78510+1		3.7267+1		2.90230+1		2.26030+1		1.7603+1		1.37100+1	
1.06770+1		8.3153+0		6.47600+0		5.04350+0		3.9279+0		3.05900+0	
2.38240+0		1.8554+0		1.44500+0		1.12540+0		8.7642-1		6.82560-1	
5.31580-1		4.1399-1		3.22420-1		1.00000-5					

終り

