JAERI-M
85-153

非線型熱伝導計算コードHEATING 6のベクトル化

1985年10月

石黒　美佐子・幾島　毅・篠沢　尚久*
奥田　基**・赤井　礼治郎**

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

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。

入手の問合わせは，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）で複写による実費頒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, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　山田軽印刷所

非線型熱伝導計算コードHEATING 6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

石黒美佐子・幾島　毅[+]・篠沢尚久*

奥田　基**・赤井礼治郎**

（1985年9月17日受理）

非線型熱伝導解析コードHEATING 6がベクトル化され，原研のFACOM VP－100で使用可能になった。6個の核となるサブルーチンをベクトル化のため書換え，95%のベクトル化率を得た。その結果，ベクトル化版はオリジナル版に較べ，核燃料輸送容器の熱解析において3倍，JT－60 NBIの加熱解析で4倍の速度向上を得た。本報告では，コードのベクトル化の概要，使用したベクトル化の手法，得られた速度向上について述べる。

+）燃料安全工学部

*）外来研究員（富士通）

**）原子力データセンター

JAERI-M 85-153

Vectorization of the Nonlinear Heat Conduction Calculation Code HEATING6

Misako ISHIGURO, Takashi IKUSHIMA[+]
Naohisa SHINOZAWA[*], Motoi OKUDA[**] and Reijiro AKAI[**]

Computing Center, Tokai Research Establishment, JAERI

(Received September 17, 1985)

Nonlinear heat conduction analysis code HEATING6 has been vectorized for use on the FACOM VP-100 computer in JAERI. Six kernel subroutines of the code are modified for vectorization, which leads to 95 % vectorized ratio. As the result, the vectorized version of the code runs three times faster than the original version for the problem of cask heating analysis and four times faster for the problem of JT-60 heating analysis. In this report, the vectorization of the code is outlined, and the vectorizing method in use and the obtained speedup gain are shown.

Keywords: Heating, Computer Codes, Heat Conduction, Nonlinear, Vector Computers, Vectorization, HEATING6

+) Department of Fuel Safety Research,
*) On leave from FUJITSU,
**) Nuclear Energy Data Center

目　　次

Contents

1. はじめに

HEATING 6 コード[1] は，使用済核燃料輸送容器の安全性検証のための熱解析用標準プログラムの1つとして米国オークリッジ研究所（ORNL）で開発された。原研では，核燃料施設安全解析の分野で，輸送中に想定される事故に対し輸送容器（キャスク）が規定の条件を満たしているかどうかを解析するために，既に，FACOM M－380版が整備されている。今回，HEATING 6 コードをFACOM VP－100用にベクトル化したので，本報告書ではHEATING 6コードのベクトル化について記述する。本報告書はは，HEATING 6コードベクトル化版の公開に伴って必要となるコードマニュアルとしての役目を果す。

HEATING 6コードは，Heat Engineering And Transfer In Nine Geometries から取ったHEATINGコード・シリーズの6番目のバージョンである。HEATING 6コードは，温度場における熱の流れを記述する一般の非線型放物型偏微分方程式を差分法によって解く計算コードである。

HEATING 6コードは，1次元，2次元，あるいは3次元について定常，非定常いずれの計算にも利用でき，直交，円筒，あるいは極座標の形状に対して温度分布を求めることができる。物性値，境界条件，およびその他の問題に関するパラメータを，空間の位置や時間の関数として与えることもできる。入力データは，自由形式で，できるかぎり簡潔になるように効率的に編成されている。

HEATING 6 コードの直前のバージョンHEATING 5についてもM－380用に整備され，さらに，入力データチェック，図形化，応力解析への結合などが強化されて使い易くなったバージョンHEATING 5－JR[2] も原研で開発されている。

HEATING 6はHEATING 5に較べて次のような点で改良がなされている[1]。

(1) ORNLで熱計算の標準コード・システムSCALEが整備されているが，HEATING6はSCALEを構成するコードの1つに組込まれた。これによって，遮蔽熱計算をSCALEを使用して連続的に計算できるようになった。

(2) SCALEコード・システムで標準的に使用されている自由形式（free format）で入力データを作成するようになった。

(3) 配列変数のとり方が完全に整合配列化された。

(4) 羽根（fin）型表面の熱伝達が取扱えるようになった。

HEATINGコードは，3次元非定常計算では，膨大な計算時間を要するのでベクトル計算機によって高速に計算できることが望まれている。このため，原研では，HEATINGコードの最新版であるHEATING 6を原研の計算機VP－100で高速に計算できるように書換える。すなわち，VP－100向きにベクトル化することになった。

SCALEコード・シリーズの中では，既に応力解析コードSAPV[3] やTRUMP 4[4] のベクトル化が個別にはなされているが，コード・システムとしてベクトル化版をまだ利用していない。HEATING 6コードのベクトル化版も含め，SCALEコード・システムの核となるコードのベ

クトル化が日本で完成したのを機会に，ベクトル計算機の利用技術が米国にフィードバックされ，かつ，国内外でベクトル化版が利用されることを願っている。

HEATING 6 コードのベクトル化にあたっては，原研での代表的な利用分野をなすキャスクの熱分解とJT－60 NBI 加熱の解析の2分野に焦点を絞り，計算実行時のコードの振舞，すなわち，サブルーチン毎のCPU時間の利用分布（以後，タイム・コスト分布と呼ぶ）を分析した。その結果，118サブルーチン（17,500 FORTRANステップ）のうち，タイム・コストの高い6サブルーチン（約4,600ステップ）をベクトル化のための書換えの対象とした。

ベクトル化の実施は3段階に分けて行われた。第1段階では，当初の目的であったキャスクの熱解析の高速化のために，計算コストの約84%を占める過渡温度分布の計算のための熱平衡式の求解部分をベクトル化した。これによって約2倍の速度向上を得た。第2段階ではさらに高速化をはかるべく，熱平衡式の係数となる熱伝導に関わる物性値計算のベクトル化をはかった。第3段階では，JT－60 NBI 加熱計算時に計算コストの高い熱計算の準備或いは計算後の計算結果の編集部分のベクトル化を実施した。

以上，3回のベクトル化作業の結果得られた速度向上の概要をFig. 1.1(a)，Fig. 1.1(b)に示す。(a)，(b)は各々，キャスクおよびJT－60 NBI 加熱の熱解析について示している。図でわかるとおり，キャスクの熱解析で約3倍，JT－60 NBI 加熱解析で約4倍の速度向上を得ている。後者は，前者に較べてタイム・ステップによる繰返しが少ないが，空間ノード点がきわめて多い問題を取扱っており，ベクトル計算はノード点に対してなされるので，速度向上効果が大きくなっている。

本報告では，HEATING 6 コードのベクトル化の概要，採用されたベクトル化方法，ベクトル化による速度向上について記述している。

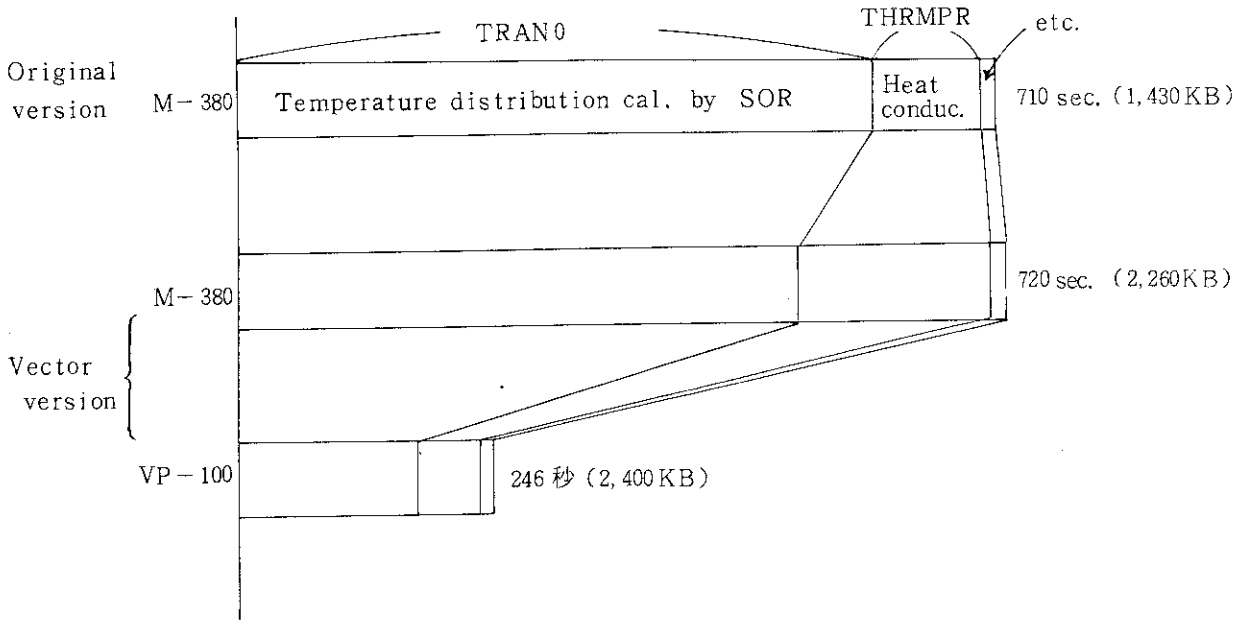


Fig. 1.1(a) Computing time reduction by vectorization for cask heating analysis

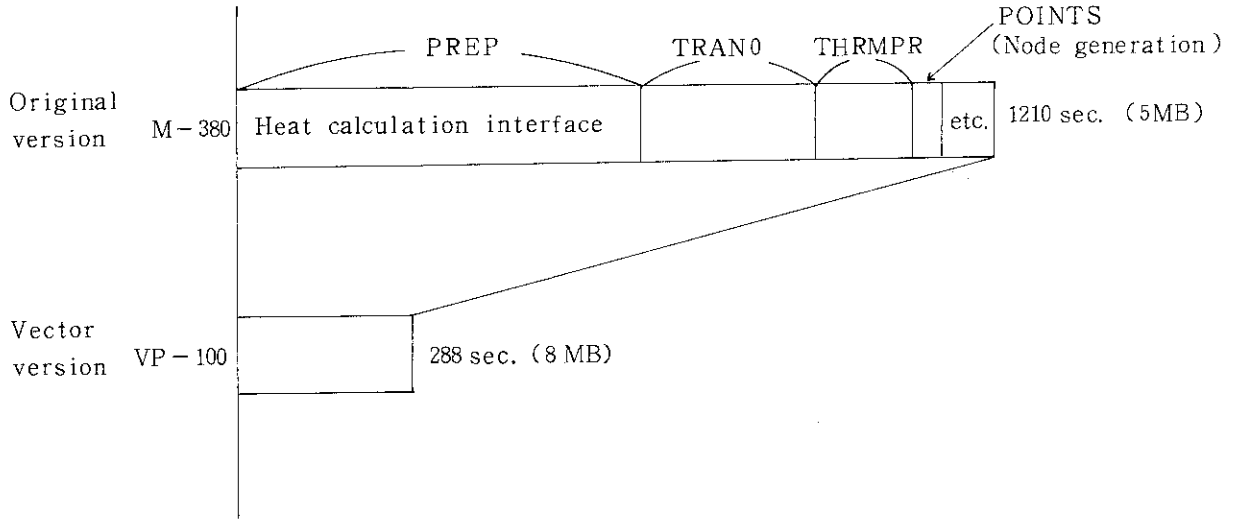


Fig. 1.1(b) Computing time reduction by vectorization for JT-60 NBI heating analysis

2. ベクトル化の概要

2.1 HEATING 6コードの計算式[1],[2]

熱解析コードHEATING 6は，有限差分法による1次元，2次元および3次元の非定常の熱移行問題を取扱う。例えば，想定される輸送容器キャスクの火災現象などが取扱える。また定常時の計算は，臨界，遮蔽計算のための原子密度をきめるための温度計算にも有効である。

後に示すように，HEATING 6コードでは，計算時間のかなりの部分が熱平衡の式による温度分布計算に費される。

Fig. 2.1において，節点Oの熱平衡の式は

$$C_o \frac{T_o^{n+1} - T_o^n}{\Delta t} = P_o^n + \sum_{m=1}^{6} {}_oK_m (T_m^n - T_o^n) \qquad (2.1)$$

ここで，T_m^nは時間t_nにおける節点Oに隣接したm番目の節点の温度，${}_oK_m$は節点Oと節点m間の熱伝導率，C_oは節点の物質の熱容量，P_o^nは時間t_nにおける節点Oの物質の発熱量である。

平面は節点を横切り，軸に沿って2つの連続した平面にはさまれた物質は均一であるので，1つの節点は最大8個の物質にまたがっており，熱は最大4個の平行に位置した異なる物質により構成される隣接した節点間で移動する。

3次元問題では，ある時刻t_nにおいて，それぞれの内存する節点に対し，1つのC，1つのP，6つのKが存在する。それらのパラメータは節点Oに対し次のように算出される。

$$C_o = \sum_{\ell=1}^{8} C_{P\ell}\, \rho_\ell\, V_\ell \qquad (2.2)$$

$$P_o^n = \sum_{\ell=1}^{8} Q_\ell^n V_\ell \qquad (2.3)$$

$${}_oK_m = \frac{1}{L_m} \sum_{r=1}^{4} K_{mr} A_{mr} \qquad (2.4)$$

ここで，

$C_{P\ell}$ ： 領域ℓの比熱
P_ℓ ： 領域ℓの密度
V_ℓ ： 領域ℓの体積
Q_ℓ^n ： 領域ℓの時刻t_nにおける単位体積当りの発熱量
L_m ： 節点Oから隣接する節点mへの距離
K_{mr} ： 節点Oと節点m間の領域rの熱伝導率
A_{mr} ： 節点Oと節点m間の領域rで熱が通過する垂直断面積

Fig. 2.1の下図より，V_ℓとA_{mr}は次のように定義される。

$$V_{\ell} = \frac{(X_{i+1} - X_i)}{2} \cdot \frac{(Y_{j+1} - Y_j)}{2} \cdot \frac{(Z_{k+1} - Z_k)}{2} \qquad (2.5)$$

$$A_{mr} = \frac{(Y_{j+1} - Y_j)}{2} \cdot \frac{(Z_{k+1} - Z_k)}{2} \qquad (2.6)$$

表面または境界上にある節点，または1次元，2次元問題の節点では隣接する節点が6つも必要ないので，M_i 個の隣接した節点を持つi番目の節点の一般的な熱平衡方程式は次の様に書ける。

$$C_i \frac{T_i^{n+1} - T_i^n}{\Delta t} = P_i^n + \sum_{m=1}^{M_i} {}_iK_{\alpha m}(T_{\alpha m}^n - T_i^n) \qquad (2.7)$$

ここでα_mはi番目の節点のm番目の隣接した節点を示す。格子間隔を十分に小さく取ることにより，モデルは適当な微分方程式に置き換えることができる。

境界ではFig. 2.2に示すように，節点1と2，2と4，5と6との間でそれぞれ熱の移動があると考える。

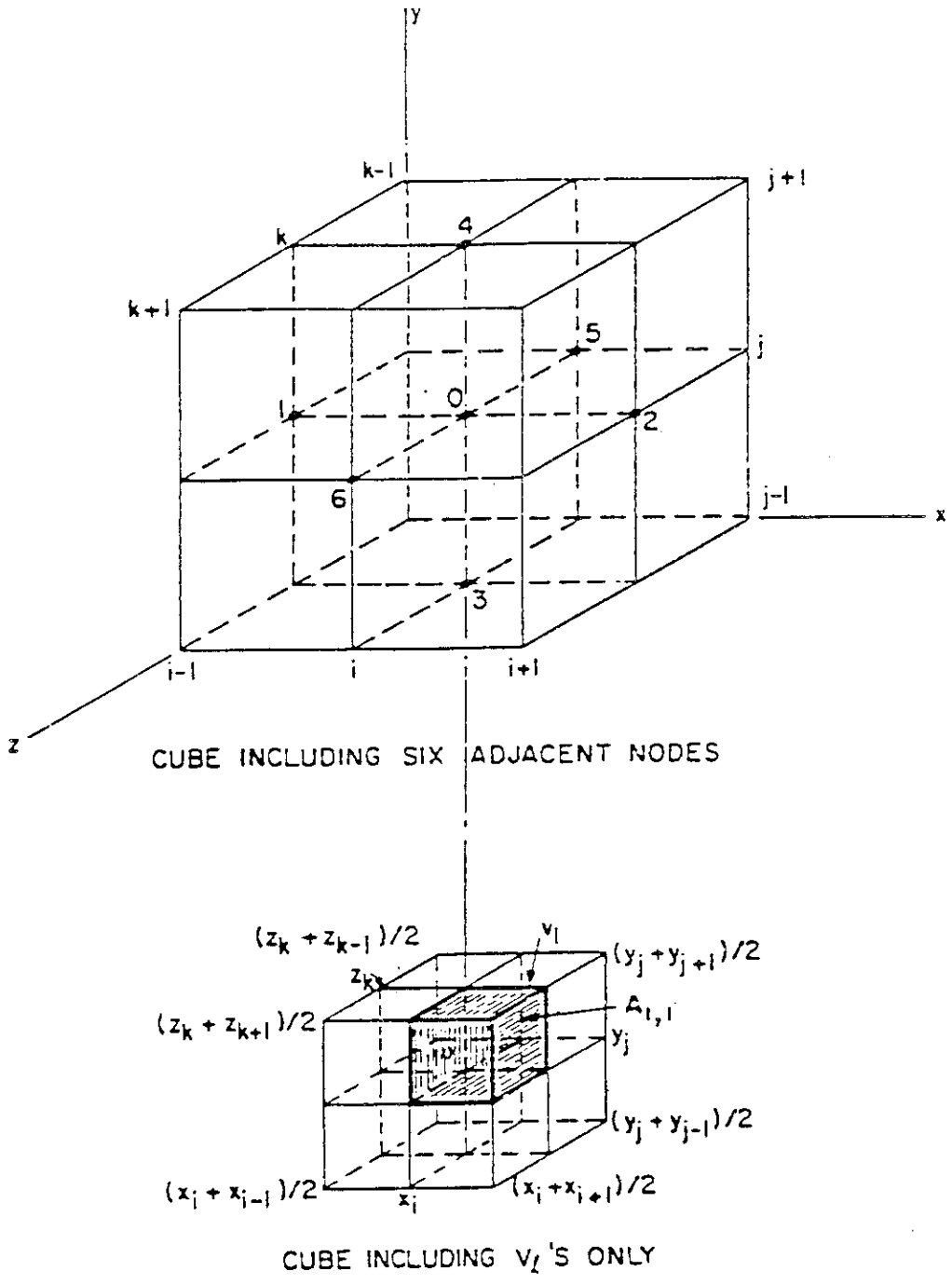


Fig. 2.1 HEATING Nodal Description for Three-Dimensional Problem

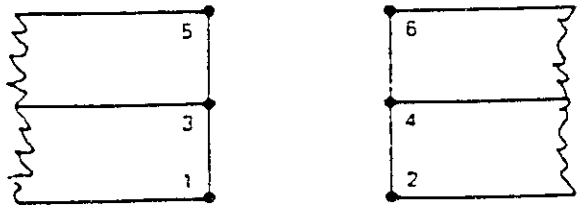


Fig. 2.2 Surface-to-Surface Heat Transfer

2.2 HEATING 6 コードの動的分析

HEATING 6 コードは，118 サブルーチン，約 17,500 行のソース・ステートメントから成る大型コードである。したがって，全ソース・プログラムをベクトル化するのではなく，計算時間のかかるいくつかのサブルーチンをベクトル化の対象とすればよい。

プログラム各部分の計算時間の分布を知るために，ソフトウェア・ツール FORTUNE によって，輸送容器キャスクの熱解析および JT－60 NBI 加熱装置の熱解析の 2 例（Table 2.1）について分析した。各々の分析結果の詳細を Fig. 2.3，Fig. 2.4 に示す。

Table 2.1 Problems for heating analysis

Item	Cask heating problem	NBI heating problem
Geometry	R-Z	X-Y-Z
Type of problem	s-s, trans, s-s	s-s, trans, s-s
Transient time	900 sec.	60 sec.
Number of regions	36	85
Number of materials	3	7
Num. of boundary conds.	5	11
Number of fine grids	12(R) 23(Z)	30(X) 20(Y) 15(Z) ==>6937 (*)
Num. of tabular functions	7	1
Number of time-steps	1492	158
Number of SOR iterations	4041	324

*) Nodes without region are excluded.

キャスクの問題では，上位 3 ルーチン TRAN 0，THRMPR，FUNCTN で全体の 96％のタイム・コストを占めている。これらの 3 ルーチンでのオリジナル版でのタイム・コスト，ベクトル化率，ベクトル長を Table 2.2 の上半分に示す。

JT－60 NBI 加熱の問題では，解析の対象とするタイム・ステップ数が小さい反面，ノード点数が約 7,000 と大きく熱計算そのものよりも，ノード作成とか計算結果の出力準備等の補助作業に時間を費やしている。上位 4 ルーチン PREP，TRAN 0，THRMPR，POINTS で約 94％の時間を占める。それらのタイム・コスト，ベクトル化率，ベクトル長を同じく Table 2.2 下半分に示す。

Table 2.2 Vectorizing condition on the original version

	Subroutine	Contents	Time cost (*1) Ratio	V	S	M	Vector length	Difficulty (*2)
Case	TRANO	Compute transient temperature distribution	.84	.53	.17	.30	276	Modest
	THRMPR	Compute heat conductance, capacity, source,...	.08	.01	.95	.05	2	Difficult
	FUNCTN	Compute values of a parameter dependent on time,..	.04	0	1.0	0	-	Difficult
NBI	PREP	Interface between temp. calcu. and printout	.56	0	1.0	0	-	Easy
	TRANO	Compute transient temperature distribution	.21	.57	.15	.28	6937	Modest
	THRMPR	Compute heat condactance, capacity, source,...	.13	0	1.0	0	-	Difficult
	POINTS	Generate nodal configuration	.04	0	1.0	0	-	Easy

*1) Ratio; subroutine time/total time, V; V-cost, S; S-cost, M; M-cost
*2) Difficulties of rewrites for vectorization

ベクトル化の難易度は，Table 2.2の最後の欄に示すようにサブルーチンによって異なる。TRAN 0ではSOR法による熱平衡方程式の求解部分を，ベクトル計算可能なodd－even SOR法に変更する必要がある。THRMPRは，DOループのボディが数100ステップにおよぶため，これを分割してベクトル計算可能なように再構成する必要がある。FUNCTNは全ノードを同時に（ベクトル）計算できるように下位ルーチンの呼出し方法の変更が必要である。一方，PREPとPOINTSは，数個のDOループにタイム・コストが集中しているけれどもベクトル計算不可能なコーディングとなっており，簡単な書直しが必要である。

ベクトル化に関係する主要ルーチンの木構造は，Fig. 2.5に示される。

NO.	ROUTINE	UNITS	LINES	ERR.	EXECUTIONS	COST	%	0....1.....2.....3.....4.....5.....6.....7.....8...
0001	TRANO	1	918	0	1	7842227095	84.4	**********
	TRANIM				1			
0002	THRMPR	1	1064	0	1	730003198	7.9	****
	THRCOM				1			
	THRM1				4039			
0003	FUNCTN	1	85	0	9218190	331854840	3.6	**
0004	PRETAB	1	142	0	1	223617460	2.4	*
	TABLE				9218190			
0005	SURBC	1	234	0	1244320	121111121	1.3	
	SURBCM				1			
0006	BDCOND	1	207	0	404000	25718641	0.3	
	BDCOCM				1			
0007	ADJUST	1	112	0	1513	9291793	0.1	
0008	PREP	1	574	0	46	2374663	0.0	
	PREPCM				1			
0009	MESH	1	329	0	1	1415168	0.0	
	TEMPNP				2			
0010	POINTS	1	391	0	1	652643	0.0	
0011	ECHO	1	127	0	1	463726	0.0	
0012	DREAD	1	255	0	936	407126	0.0	
0013	TMPOUT	1	339	0	3	283120	0.0	
	TMPCM				1			
0014	YOREAD	1	41	0	140	140703	0.0	
0015	SSNODE	1	112	0	19	117487	0.0	
0016	WRITER	1	612	0	1	85882	0.0	
0017	TMSTEP	1	196	0	1	61991	0.0	
	UDTS				1491			
0018	INCRES	1	38	0	1491	45918	0.0	
0019	AREAD	1	148	0	145	21331	0.0	
0020	FCHECK	1	283	0	1	20612	0.0	
0021	CALQLT	1	892	0	1	19442	0.0	
0022	ICLOCK	1	7	0	5533	16599	0.0	
0023	TPRNGE	1	77	0	2	15831	0.0	
0024	INPUT	1	534	0	2	11405	0.0	
	INPCM1				1			
	INPCM2				1			
0025	OUTPUT	1	427	0	1	11125	0.0	
0026	MAIN01	1	201	0	1	7146	0.0	
0027	TIMES	1	14	0	6	6186	0.0	
0028	DIFFER	1	198	0	1	6139	0.0	
0029	H6	1	44	0	1	6017	0.0	
0030	TPMNTR	1	54	0	1490	5960	0.0	
0031	HEATN6	1	1249	0	1	4930	0.0	
0032	READER	1	237	0	1	3810	0.0	
0033	REGONS	1	99	0	36	3672	0.0	
			17355	0		9290049774		

Fig. 2.3 Time cost of subroutines for cask heating analysis

NO.	ROUTINE	UNITS	LINES	ERR.	EXECUTIONS	COST	%	0.....1.....2.....3.....4.....5...
0001	PREP	1	578	0	9	7449815997	52.9	**********
	PREPCM				1			
0002	TRANO	1	1045	0	1	2991979936	21.3	**********
	TRANOR				1			
	TRANIM				1			
0003	THRMPR	1	2020	0	1	1824817655	13.0	**********
	THRCOM				1			
	THRCM2				1			
	THRM1				324			
0004	POINTS	1	411	0	1	576115923	4.1	***
0005	CHCKBD	1	207	0	324	423105445	3.0	** (part of TRANO)
	CHCKBR				1			
0006	PREFUN	1	267	0	1	270618401	1.9	*
	FUNCT2				2600			
0007	BDCON2	1	275	0	7800	241113997	1.7	*
	BDCOC3				1			
0008	SURBC2	1	309	0	7800	216465601	1.5	*
	SURBC3				1			
0009	TMPOUT	1	339	0	150	35377491	0.3	
	TMPCM				1			
0010	ADJUST	1	112	0	162	24743327	0.2	
0011	FUNCTN	1	85	0	278712	9991520	0.1	
0012	PRETAB	1	251	0	1	7303596	0.1	
	TABLE				236600			
	TABLE2				0			
0013	H6	1	47	0	1	1776018	0.0	
0014	ECHO	1	127	0	1	831916	0.0	
0015	TPRNGE	1	78	0	9	791889	0.0	
0016	DREAD	1	255	0	1783	764525	0.0	
0017	PREANA	1	169	0	1	421121	0.0	
	ANALYT				42112			
	ANALT2				0			
0018	CALQLT	1	924	0	1	326983	0.0	
0019	YOREAD	1	41	0	250	251253	0.0	
0020	TIMES	1	14	0	153	157743	0.0	
0021	WRITER	1	613	0	1	99552	0.0	
0022	FCHECK	1	283	0	1	99394	0.0	
0023	AREAD	1	148	0	271	39930	0.0	
0024	DIFFER	1	200	0	1	24150	0.0	
0025	REGONS	1	99	0	85	17000	0.0	
0026	OUTPUT	1	427	0	1	12124	0.0	
	TRANIM				1			
0118	TYPEBC	1	43	0	255	2853	0.0	
0119	UMONTR	1	21	0	1	159	0.0	
	USRMON				158			
0120	WRITER	1	613	0	1	99552	0.0	
0121	YOPACK	1	11	0	1067	5335	0.0	
0122	YOREAD	1	41	0	250	251253	0.0	
			20129	0		14077145225		

Fig. 2.4 Time cost of subroutines for JT-60 NBI heating analysis

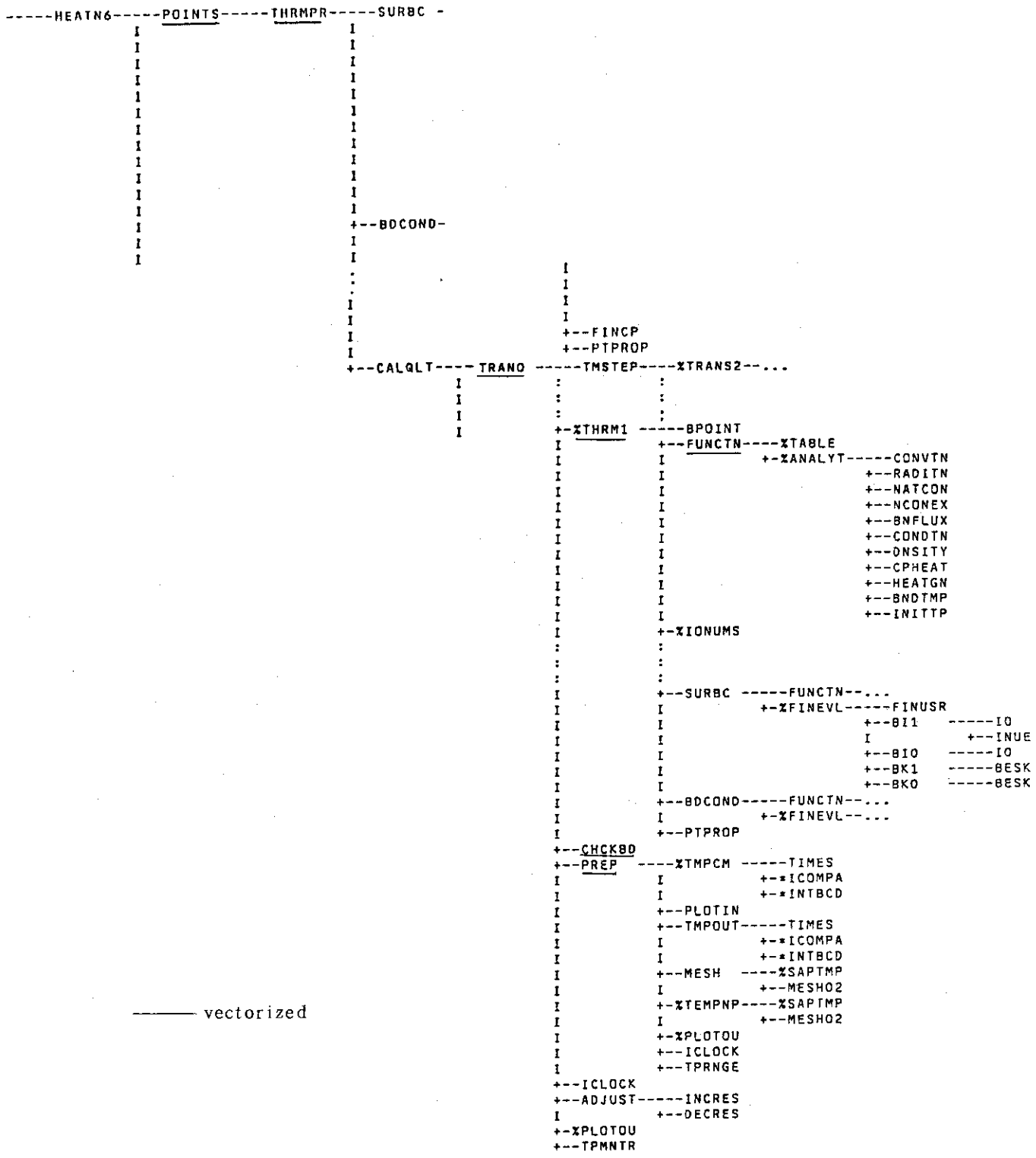


Fig. 2.5 Tree structure of time-consuming routines in HEATING6 code

2.3 ベクトル化の経過

HEATING 6 コードは，本来，安全性研究の一環として，核燃料輸送容器の安全性を確かめるための熱解析に用いられることを意図して整備された。HEATING 6 を使ってのキャスクの熱解析は，原子炉安全解析所でも計画され，原研にFACOM版の提供が要請された。

このような事情から，当初はキャスクの熱解析に適合するベクトル化版の整備が急がれた。この目的で，2回に分けてベクトル化作業がなされた。第1回目は，タイム・コストの84.4%を占める。TRAN 0に関するものである。これは原子力データセンターへ作業を発注する形式で，60年1～3月に奥田，赤井によって実施された[5]。第2回目は，さらにベクトル化範囲を広げるために，THRMPR とTHRMPRから呼ばれるいくつかのルーチンのベクトル化に関するもので主に篠沢が作業を担当した。各々を，ベクトル化第1版，第2版と呼ぶことにする。第1版，第2版に関するベクトル化作業の詳細は第3章および第4章に記述される。

一方，JT－60 NBI加熱に関する熱解析グループは従来から，HEATINGの初期バージョンであるHEATING 3[6]を利用し，大規模な熱計算を行ってきた。HEATING 6 のベクトル化第2版をNBI加熱計算に適用したところ，HEATING 3 に較べて約15倍高速に計算できることがわかった。しかしながら，この速度向上の大部分は，HEATING 3 とHEATING 6 のバージョンの違いによるもので，ベクトル化による速度向上は少ないことがわかった。

JT－60加熱計算でベクトル化効果が上がらない理由は，取扱うノード数が膨大（約7,000点）ではあるが，繰返されるタイム・ステップ数が少ないため，熱計算以外の補助的部分，つまり，モデル体系のノード構成作成や，熱分布計算後の値をグローバル・ノード点に編集するなどの部分にかなりの時間を費しているからである。このため，サブルーチンPOINTSおよびPREPのベクトル化が追加された。これをベクトル化第3版と呼ぶことにする。ベクトル化第3版は主に石黒によってなされ，その詳細は第5章に記述される。

ベクトル化に先立ち，NBI加熱計算のような大規模計算を取扱うために，プログラムの手直しを必要とした。その内容は以下のとおりである。

(1) ベクトル化に要する作業用配列を整合配列化し，汎用性を持たせる。

(2) 従来各サブルーチン単位で，又DOループ毎に独立に取っていた作業用配列を，重複して使用し，プログラム全体として8M（メガ）バイト以下に収まるようにする。

(3) HEATING－JRのPOST処理の一部である図形化のための作業用配列の利用を制限する（暫定的に図形化不可とする。将来的には，同一領域の共用をはかる）。

これらの手直しは，主に篠沢によってなされた。ベクトル化に必要な作業用領域については第6章で示される。

2.4 ベクトル化による速度向上倍率と計算結果

(1) ベクトル化第1版

キャスクの熱解析でタイム・コトの大部分を占めている（84％）サブルーチンTRAN 0をベクトル化した。

TRAN 0は過渡状態の熱伝導方程式をSOR法を用いて計算し，ノード点上の温度分布を計算する。ここを，odd－eveo SOR法[7]に変更しベクトル化した。さらに，odd－even法を適用するためにサブルーチンCHCKBDを新設した。

解法アルゴリズムをodd－even SOR法に変更したことによって，温度分布値が4桁目から異なる場合がある。ベクトル化による速度向上は，オリジナル版との比較で2.2倍である（Table 2.3参照）。

(2) ベクトル化第2版

さらに速度を向上させるために，次にタイム・コストの高い初期温度，熱容量，熱発生量，熱伝導率などを計算するルーチンTHRMPRとそこから呼ばれる関数作成部分FUNCTNとテーブルPRETABをベクトル化した。TRAN 0は，約50行の大きなDOループ・ボディから成っている。このDOループの適当な分割，DOループ構造の内外入れ換え，サブルーチンCALLの一括処理，テーブル構造の変更など大巾なプログラムの変更を行った。しかし，これらの変更は，計算結果に影響を与えない。速度向上は2.9倍と向上した（Table 2.3参照）。

(3) ベクトル化第3版

NBI加熱の熱解析に対するベクトル化に向けてFig. 2.4で示した計算コストの高いサブルーチンPREPとPOINTSのベクトル化を追加した。PREPは，温度分布計算結果を出力するための編集，POINTSは，体系のノード構成を作成する部分である。ベクトル化内容は，DO ループの再構成に留まっているので，計算結果への影響はない。

NBI加熱の熱解析に対する速度向上はTable 2.4に示すように，第2版で約1.2倍，第3版で4.3倍である。HEATING 3を使用していたのと較べると以下に示すとおり，計算時間は約1/50と短縮され，利用者に喜ばれている。

HEATING 3		4時間
HEATING 6	オリジナル版	20.8分
HEATING 6	ベクトル化第2版	16.9分
HEATING 6	ベクトル化第3版	4.8分

Table 2.3 Comparison of CPU times for cask heating analysis (276 nodes, 4041 time-steps)

Version & CPU	CPU-time (*1)	speedup ratio	Vu-time (*2)	Momory
Original M-380	11M50S	1.0	0	1.5M byte
Vector 1st. VP-100	5M31S	2.2	1M54S	1.7M
Vector 2nd. VP-100	4M07S	2.9	2M50S	2.4M

*1) CPU times used on scalar and vector units on VP-100 are included, where the scalar speed is the same with M-380.
*2) Time used on vector unit.

Table 2.4 Comparison of CPU times for JT-60 NBI heating analysis (6937 nodes, 324 time-steps)

Version & CPU	CPU-time	speedup ratio	Vu-time	Momory
Original M-380	20M49S	1.0	0	5.0M byte
Vector 2nd. VP-100	16M52S	1.2	2M22S	8.0M
Vector 3rd. VP-100	4M48S	4.3	2M59s	8.0M

3．過渡温度分布計算ルーチンTRANOのベクトル化

3.1 プログラムの分析

サブルーチンTRAN 0は，過渡状態の温度分布を求めるために，Crank－Nicolson法とFull implicit 法の線型結合によって表わされる差分式をSORを使って求めている。

TRAN 0は，Table 3.1の3つのDOループで全体の84％のタイム・コストを占めている。

Table 3.1 Time cost of DO loops in the original TRANO

DO loop	Time cost	Contents
DO 390	1.3%	Pre-processing for SOR
DO 490	45.0	Solution by point SOR
DO 590	53.2	Convergence test

参考のために，Fig. 3.1に点SOR法による計算のフローを示す。

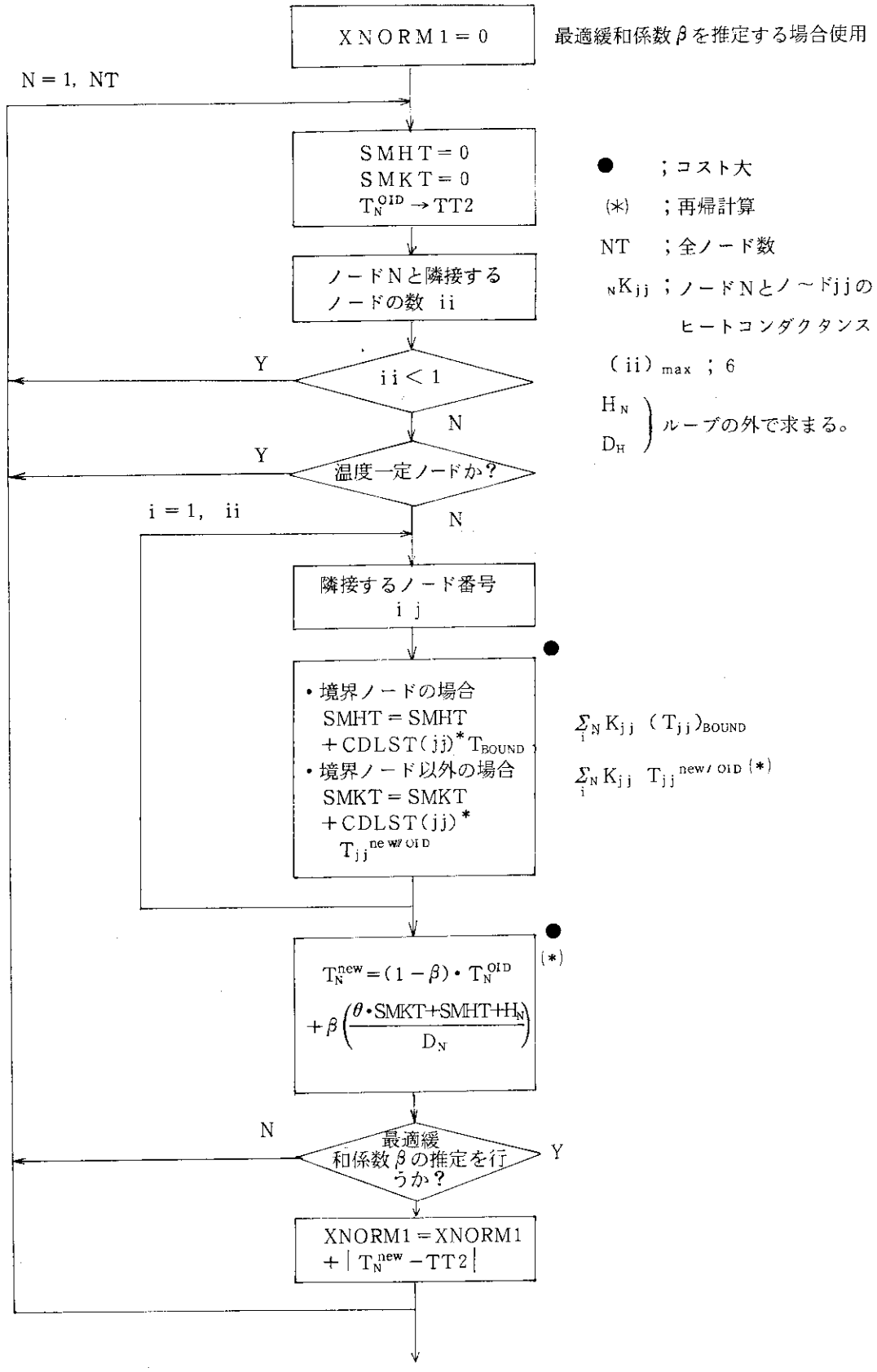


Fig. 3.1 Flow diagram of point SOR method in the original TRAN0

3.2 Odd－even法による温度分布計算のベクトル化

サブルーチンTRAN 0では，特にDO 390，DO 490，DO 590の3つのDOループに対してベクトル化のための修正を行った。

修正を行った3つのDOループの修正後のソース・プログラムをFig. 3.2～Fig. 3.4に示す。

修正の内容を以下に示す。

(1) DO 390，DO 490，DO 590

○ キャスクの熱解析データの場合，外側DOループのDOのくりかえし数は，276であるが，内側DOループのくりかえし数は4と短い。

このため，内側と外側のDOループの入れ換えを行った。

○ INTEGER＊2の配列は，ベクトル化の対象にならないため，INTEGER＊4の配列に置き変えた。

NNB（N） → IIX（N）

NBLIST（M） → JJX（N，I）

○ LOGICAL＊1の配列は，ベクトル化の対象にならないため，INTEGE＊4の配列に置き変えた。

FIXTP（N） → IFIXTP（N）

＂TRUE＂の場合，IFIXTP（N）＝1，＂FALSE＂の場合，IFIXTP(N)＝0

○ リスト・ベクトルを作成し，DOループ内のIFを除いた。

ここで，配列IIX(N)，JJX(N)，IFIXTP(N) およびリスト・ベクトルは新設ルーチンCHCKBDで作成した。

(2) DO 490

点SOR法による計算部分が回帰演算となるため，ベクトル化できない。そこで点SOR法をodd－even SOR法に変更した（Fig. 3.5参照)。odd－even SOR法への変更に際しては，以下の修正を行った。

A. 新規作成ルーチン

CHCKBD……チェッカ・ボードの作成。

配列IIX(N)，JJX(N)，IFIXTP(N)およびリスト・ベクトルを作成した。プログラムの呼び出し関係は以下のとおりである。

HEATN 6 — CALQLT — TRAN 0 — CHCKBD

B. 新設コモン

COMMON / XXX /

COMMON / YYY /

COMMON / ZZZ /

COMMON / XXXX /

COMMON / YYYY /

Fig. 3.2 DO 390 in the vectorized version

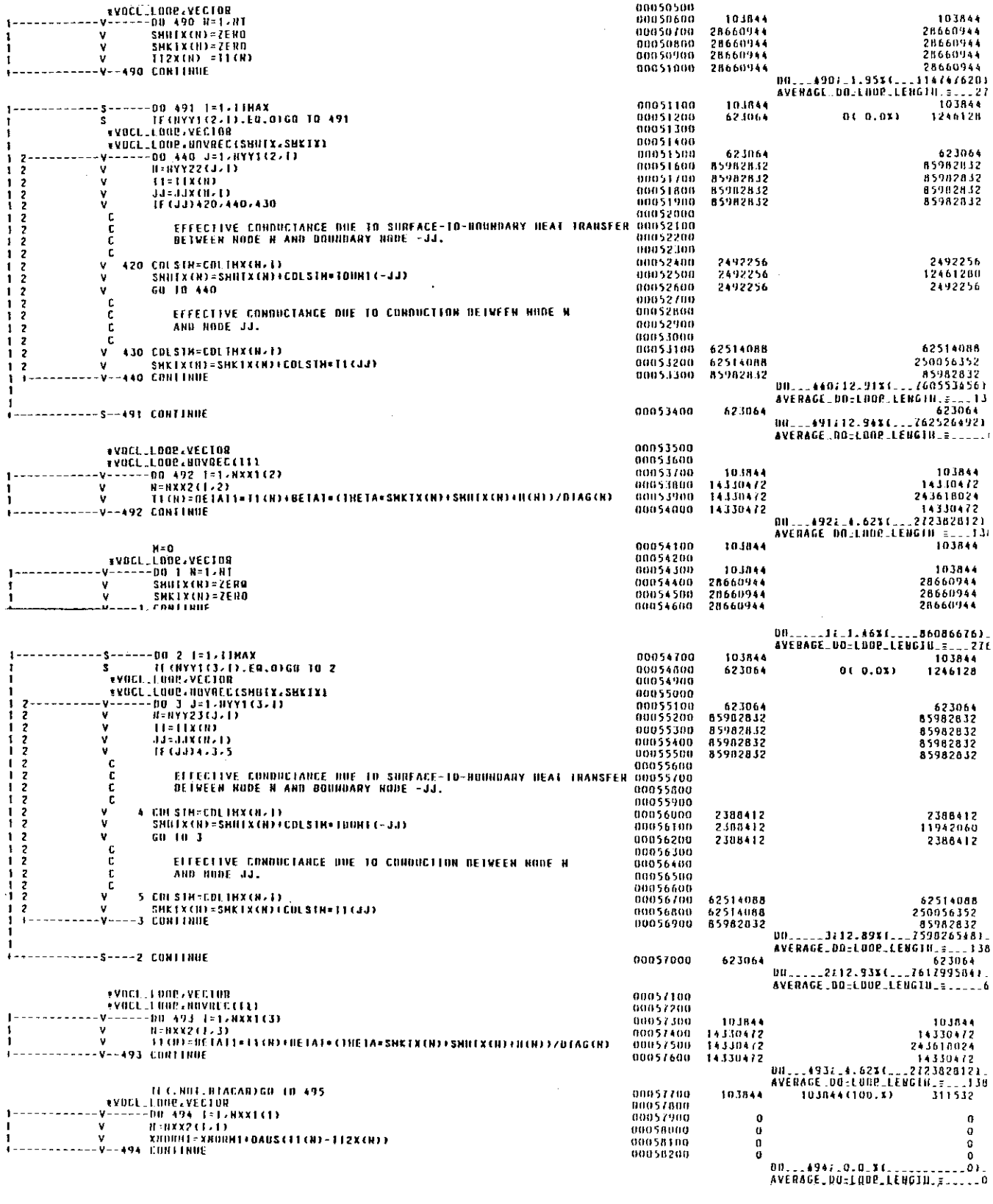


Fig. 3.3 DO 490 in the vectorized version

```
                *VOCL LOOP,VECTOR
            V------DO 7 N=1,NI
            V       SHNIX(N)=ZERO
            V       SHKIX(N)=ZERO
            V       NSHKIX(N)=0
            V----7  CONTINUE


            S------DO 590 I=1,IMAX
                *VOCL LOOP,VECTOR
            V------DO 540 N=1,NI
            V       II=IIX(N)
             *VOCL SCHI,IF(10)
            V       IF(II.LT.1)GO TO 540
             *VOCL SINI,IF(10)
            V       IF(IFIXIP(N).EQ.1)GO TO 540
            V       IF(I.GT.II)GO TO 540
            V       JJ=JJX(N,I)
            V       IF(JJ)520,540,530
              C
              C        EFFECTIVE CONDUCTANCE DUE TO SURFACE-TO-BOUNDARY HEAT TRANSFER
              C        BETWEEN NODE N AND BOUNDARY NODE -JJ.
              C
            V  520 COLSIH=COLIHX(N,I)
            V       SHNIX(N)=SHNIX(N)+COLSIH*TBDN1(-JJ)
            V       NSHKIX(N)=NSHKIX(N)+1
            V       GO TO 540
              C
              C        EFFECTIVE CONDUCTANCE DUE TO CONDUCTION BETWEEN NODE N
              C        AND NODE JJ.
              C
            V  530 COLSIH=COLIHX(N,I)
            V       SHKIX(N)=SHKIX(N)+COLSIH*T1(JJ)
            V       NSHKIX(N)=NSHKIX(N)+1
            V----540 CONTINUE


            S--590 CONTINUE


                *VOCL LOOP,VECTOR
            V------DO 8 I=1,NXX1(1)
            V       N=NXX2(I,1)
            V       RNORM=H(N)
            V       IF(RNORM.EQ.0.0D0)RNORM=(THETA*SHKIX(N)+SHNIX(N)+DIAG(N)*T1(N))/
            V      . DELTA(2*NSHKIX(N)+1)
            V       IF(RNORM.NE.0.0D0)GO TO 9
                    RZORM=DMAX1(1.0D0,RESORM)
            V       GO TO 8
            V    9  RZOR=DABS((THETA*SHKIX(N)+SHNIX(N)+H(N)-DIAG(N)*T1(N))/RNORM)
            V       REZORM=DMAX1(REZORM,RZOR)
            V----8  CONTINUE
```

```
00059200
00059300    103844                      103844
00059400  28660944                    28660944
00059500  28660944                    28660944
00059600  28660944                    28660944
00059700  28660944                    28660944
                    DO_____7:_1.95%(___11878762
                    AVERAGE_DO=LOOP_LENGTH_=___2
00059800    103844                      103844
00059900
00060000    623064                      623064
00060100 171965664                    171965664
00060200
00060300 171965664          0( 0.0%)  343931328
00060400
00060500 171965664          0( 0.0%)  343931328
00060600 171965664   42056820(24.4%)  385988148
00060700 129908844                    129908844
00060800 129908844                    129908844
00060900
00061000
00061100
00061200
00061300   4880668                     4880668
00061400   4880668                    24403340
00061500   4880668                     9761336
00061600   4880668                     4880668
00061700
00061800
00061900
00062000
00062100 125028176                    125028176
00062200 125028176                    500112704
00062300 125028176                    250056352
00062400 171965664                    171965664
                    DO___540:44.08%(__2592346128
                    AVERAGE_DO=LOOP_LENGTH_=___2
00062500    623064                      623064
                    DO___590:44.09%(__2598023036
                    AVERAGE_DO=LOOP_LENGTH_=____
00062600
00062700    103844                      103844
00062060  28660944                    28660944
00062900  28660944                    28660944
00063000  28660944          0( 0.0%)  57321888
00063100
00063200  28660944   28660944(100.%)  85982832
00063300         0                           0
00063400         0                           0
00063500  28660944                    458575104
00063600  28660944                     85982832
00063700  28660944                     28660944
                    DO_____8:13.13%(___2739493328
                    AVERAGE_DO=LOOP_LENGTH_=___27
```

Fig. 3.4 DO 590 in the vectorized version

偶数メッシュと奇数メッシュを分けて計算する

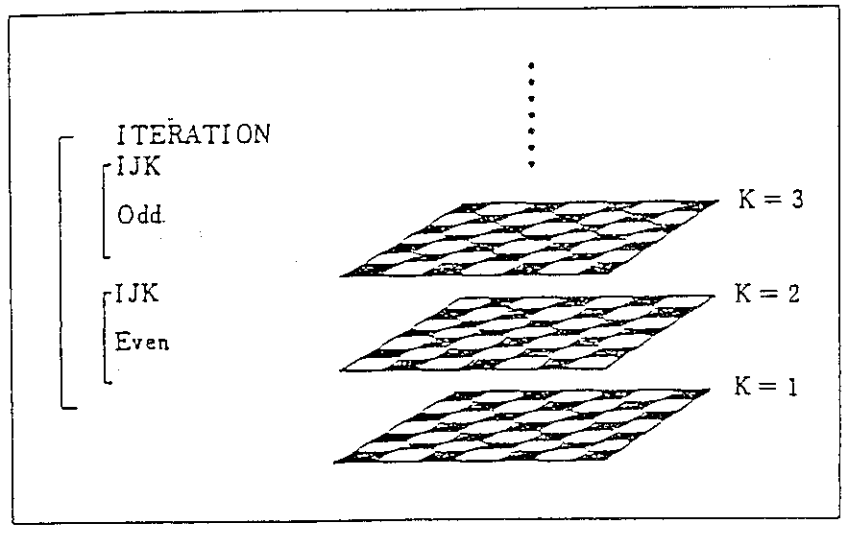


Fig. 3.5 Odd-even SOR method

3.3 その他のベクトル化

本作業では，ベクトル計算によるオーバヘッドをなくすため，VPコンパイル時にSCALARオプションを使用している。このため，最適化制御行を挿入しない限りDOループは，ベクトル化されない。そこで，サブルーチンTRAN 0以外のベクトル化として，Table 3.2に示すDOループに最適化制御行＊VOCL　LOOP, VECTORを挿入しベクトル化をはかった。

Table 3.2 DO loops to which *VOCL LOOP,VECTOR is inserted

Subroutine	DO loop
DAXPY	DO 30
	DO 50
DIFFER	DO 226
	DO 228
MESH	DO 700
	DO 710
	DO 220
TPRNGE	DO 5000
CALQLT	DO 2520

3.4 キャスク熱解析の計算結果

キャスク熱解析データを使用して，オリジナル版をM－380で，ベクトル化版をVP－100で各々実行したところ，有効桁4桁目で一部数値が異なる場合がある。また，全計算に要したタイム・ステップ数の方もオリジナル版で1490回，ベクトル化版で1492回と多くなっている。これらの違いは，点SOR反復をodd－even SOR反復に変更したことから生じている。ここで用いられたSOR法の収束判定基準は 10^{-5} である。SOR法に要した反復回数は，点SOR法，odd－even法いずれの場合も1タイム・ステップあたり平均2.7と変わらない。

Table 2.3の1，2行にオリジナル版とベクトル化版の計算時間，主記憶量などの比較を示した。Table 3.3には，各DOループの速度向上率を示す。また，Table 3.4には，オリジナル版とベクトル化版のベクトル化率の比較を示す。

Table 3.3 Comparison of CPU times of DO loops in TRANO

Version & CPU	CPU time (sec.) (*)				
	CHCKBD	TRANO	DO 390	DO 490	DO 590
Original M-380 (a)	-	769.0	9.1	254.5	277.9
Vector M-380 (b)	26.36	848.6	11.1	266.6	314.0
Vector VP-100 (c)	15.36	347.3	1.8	43.2	78.4
Speedup ratio (a/c)	2.21	2.2	5.0	5.9	3.6

*) CPU time includes those of slave routines, and it also includes some overhead time for CLOCKM routine.

Table 3.4 Comparison of vectorizing ratios of TRANO

Version	V-cost	M-cost	S-cost
Original	52.8%	17.5	29.8
Vector	99.6	0.0	0.4

4．熱伝導計算THRMPR関連サブルーチンのベクトル化

4.1 ベクトル化のための領域合わせ

本コードは，もともと可変長配列方式を用いており，使用するデータ領域をすべて1つの配列に割り当てている。オリジナル版の配列の宣言は，REAL＊8，INTEGER＊2，LOGICAL＊1であり，この2バイト整数，1バイト論理変数はFACOM FORTRAN 77/VPではベクトル化対象外であり，ベクトル化の障害となっていた。そこでベクトル化第2版ではプログラム内で使用している1バイト，2バイト変数を，文字操作に使用している変数を除いてすべて4バイトの整数型，論理型に変更した。

この変更に伴って，領域の割り当て計算を変更する必要が生じたが，この割り当てを行うサブルーチンHEATN 6には，PDP－10用の4バイト割り当てオプションがあり，オプションを切り換えるだけでほとんど完了した。ただし，HEATING－JRのために図形処理用データ出力ルーチンが追加されており，そのための作業用配列がこの中で割り当てられているが，その計算がオリジナルのHEATING 6版の計算方式とは別であるため，このままでは割り当て値が異常な値を示すようになる。そこで，ここをオリジナル方式に合わせるように変更した。

この変更による必要メモリーの増加は，キャスクの熱計算の場合変更前の必要量が95,157語であるのに対して変更後は104,940語であり，問題にならない程度である。また，ベクトル化第1版においてサブルーチンTRAN 0に対して行われた。1バイト変数，2バイト変数の4バイト作業用配列への転送も必要がなくなり，もとの変数を使用するように再変更した。

このソース・プログラムによる実行結果は以前のベクトル化第1版と完全に一致し，またCPU時間もほとんど変わりがなかった。以後，これを基にしてベクトル化を行った。

4.2 THRMPRのプログラム構造

以下では，第3章で扱ったTRAN 0の次に計算コストの高いTHRM 1（THRMPR のエントリー），THRMPRから呼ばれるFUNCTN，TABLE（PRETABのエントリー）のベクトル化について述べる。

THRMPRはノード間の熱伝導率，ノードの物質の熱容量，発熱率を計算するルーチンであり，その下に，各パラメータ値を計算するFUNCTN，表面ノードと境界ノード間の熱伝導率を計算するBDCOND，表面ノード間の熱伝導率を計算するSURBCがついている。簡単な流れはFig. 4.1のとおりである。

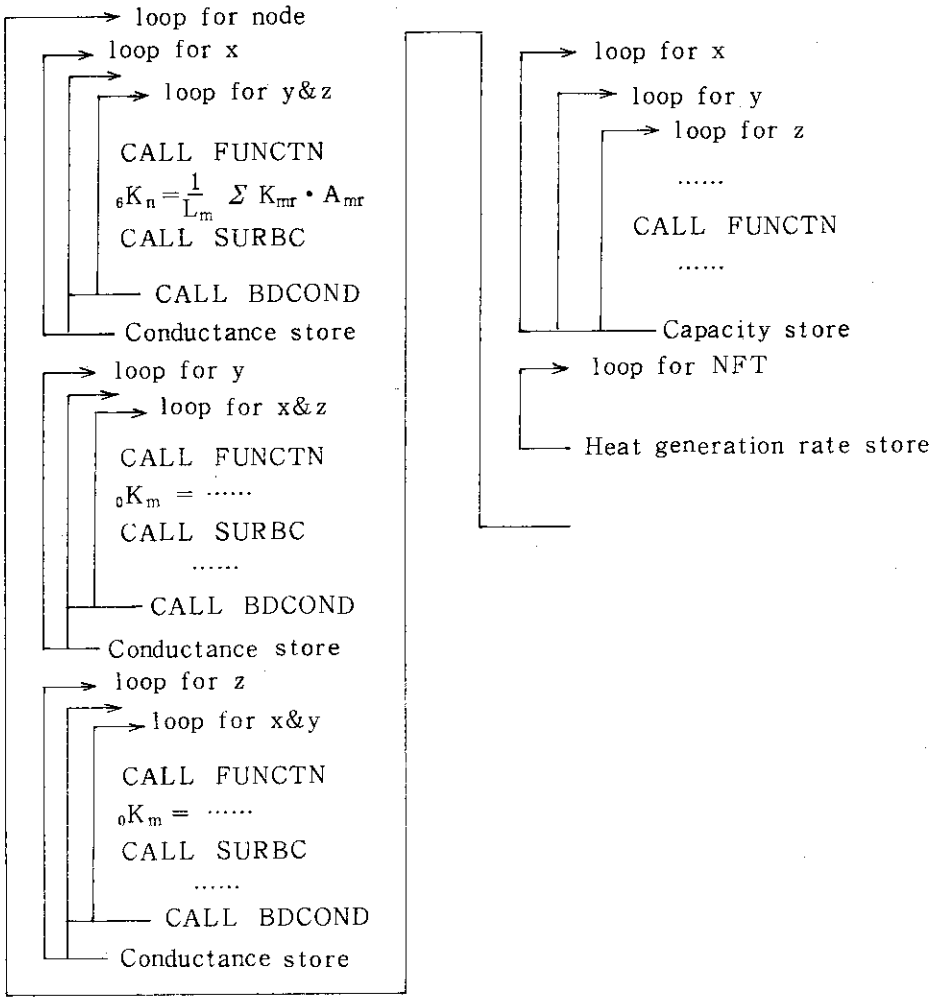


Fig. 4.1 Program structure of the original THRMPR

4.3 ベクトル化のためのプログラムの変更

(1) データ構造

オリジナル版においては，隣接ノード番号及びその熱伝導率は，以下のような形式で一次元配列に保存されている。

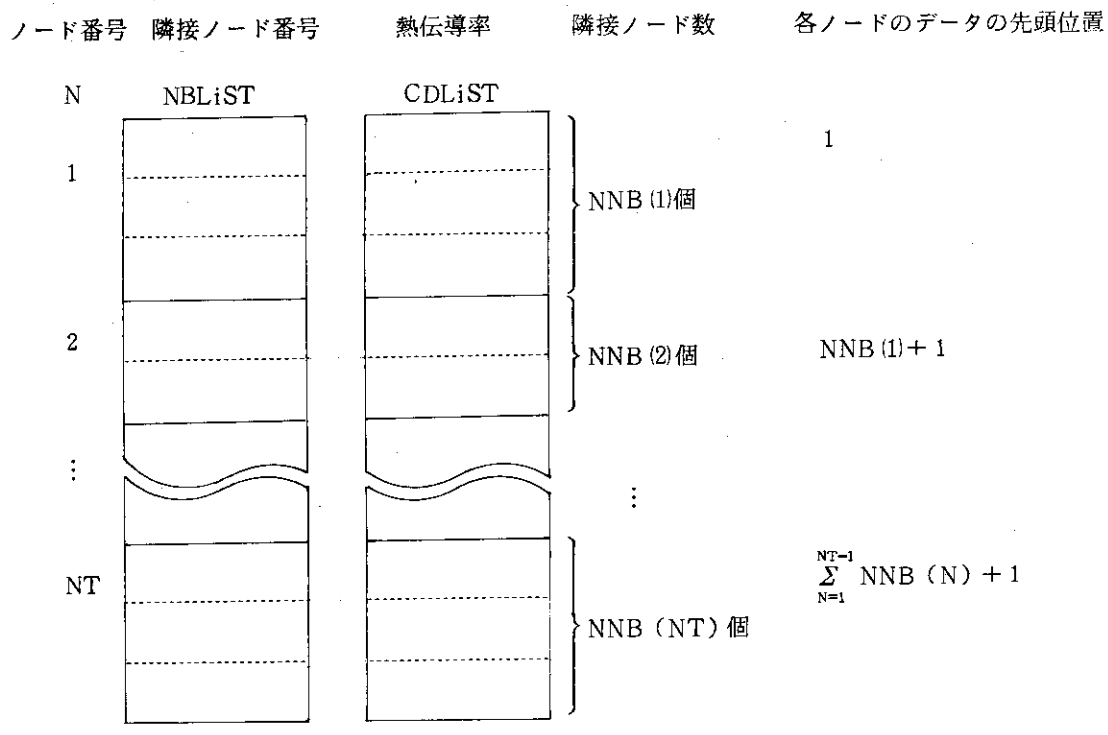


Fig. 4.2 Original tabular form for heat conductions

この方法は領域の大きさが最小で済むうえに，オリジナルDOループの順序ならば1データずつ昇順にアクセスできるため効率が良い。しかし今回このDOループの順序をFig.4.1で示したように入れ換えるため，アクセス順序がでたらめになり，間接アドレス参照となるために効率が良くない。そこで領域は多く必要となるがこれらを2次元配列に変更ノード番号方向に連続となるようにかえた。

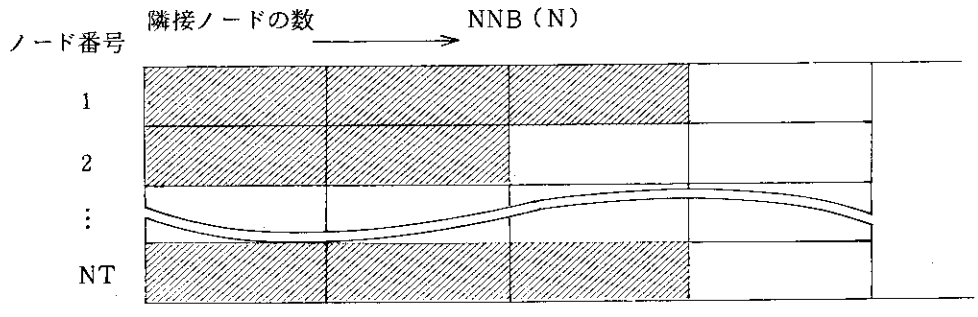


Fig. 4.3 Revised tabular form for heat conductions

この変更によって，必要な領域が増加するためサブルーチンHEATN 6における領域に関する計算式を変えた（Fig. 4.4）。また変更した配列CDLIST，NBLIST(熱伝導率とその隣接ノード番号)，VGLIST，

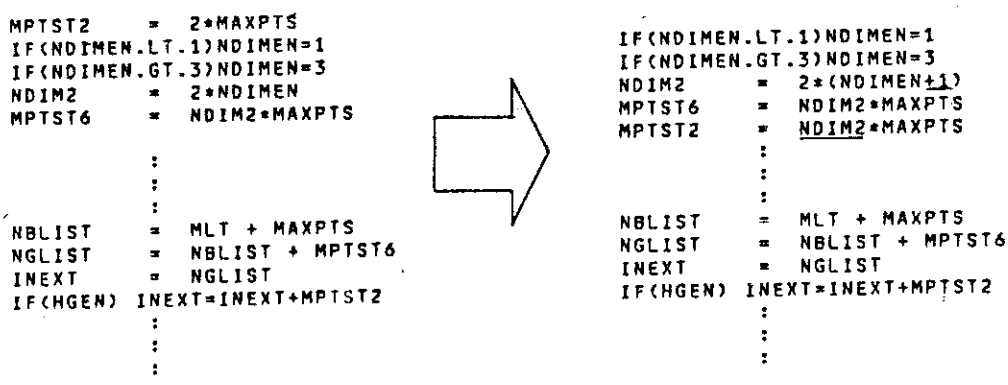


Fig. 4.4 Rewrites of parameters defining array sizes

NGLIST（発熱率とその隣接ノード番号）を使用しているルーチンをそれに合わせて変更した。変更したルーチンは，CALQLT，DIRECT，INFACE，POINTS，PTPROP，THRMPR，TRAN0であり，Fig. 4.5のパターンで変更した。

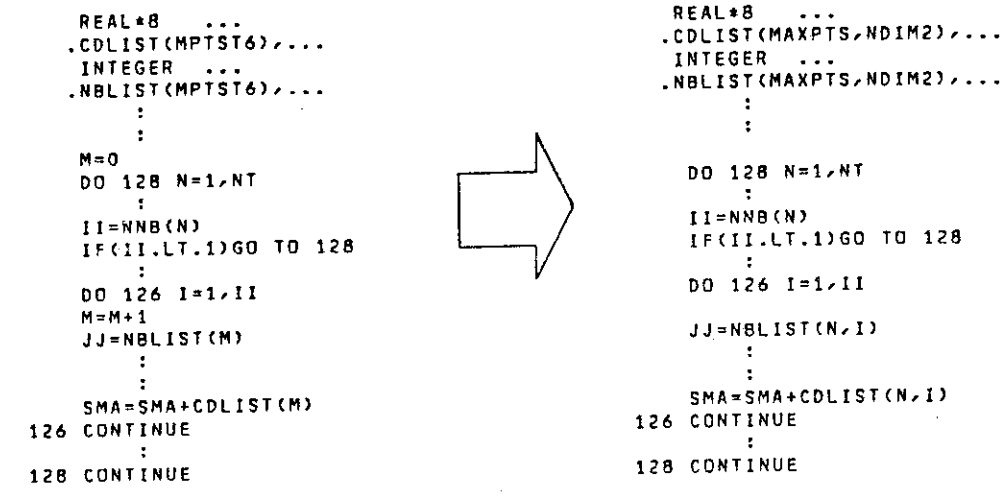


Fig. 4.5 Typical coding rewrite for array data references

(2) THRMPR

サブルーチンTHRMPRの主なループの一部を取り出してみるとFig. 4.6の左半分のようになっている。最外ループにノードがあり，各ノードごとにパラメータを別の小さな配列に転送し，内側のループでそれを使用している。ノード数でベクトル化するにあたってそれをそのまま次元数を増やした配列としても良いのだが，サブルーチンHEATN 6での領域計算においてGIMとGIP，G2MとG2P，……，G7MとG7P，及びNILBR，NILFR，NIRBR，……，NORFRはそれぞれ，連続領域にとられており宣言時にそれぞれ，

G1M (MAXRFG, 2), G2M (MAXRFG, 2)

MREGT (MAXPTS, 2, 2, 2)

と定義することによって，そのままでベクトル化のための配列が準備できることになる（Table 4.1参照)。

ただし，MREGについてはx，y，zの順序が逆になる。

Table 4.1 Correspondence of the vector and scalar variables in THRMPR

Original version	Vectorized version
G1(1)	G1M(IX,1)
G1(2)	G2M(IX,2)
.	.
.	.
G5(1)	G5M(IX,1)
G5(2)	G5M(IX,2)
.	.
.	.
MREG(I,J,K)	MREGT(N,K,J,I)

さらに，IF文によって定義しているNBC，NBC 2においても上と同様に

NBDIN, NIN, NBDOT, NOT

NBDLT, NLT, NBDRT, NRT

NBDBK, NBK, NBDFT, NFR

などの配列をそれぞれひとまとめにして

NBDIN (MAXREG, 2, 2)

NBDLT (MAXREG, 2, 2)

NBDBK (MAXREG, 2, 2)

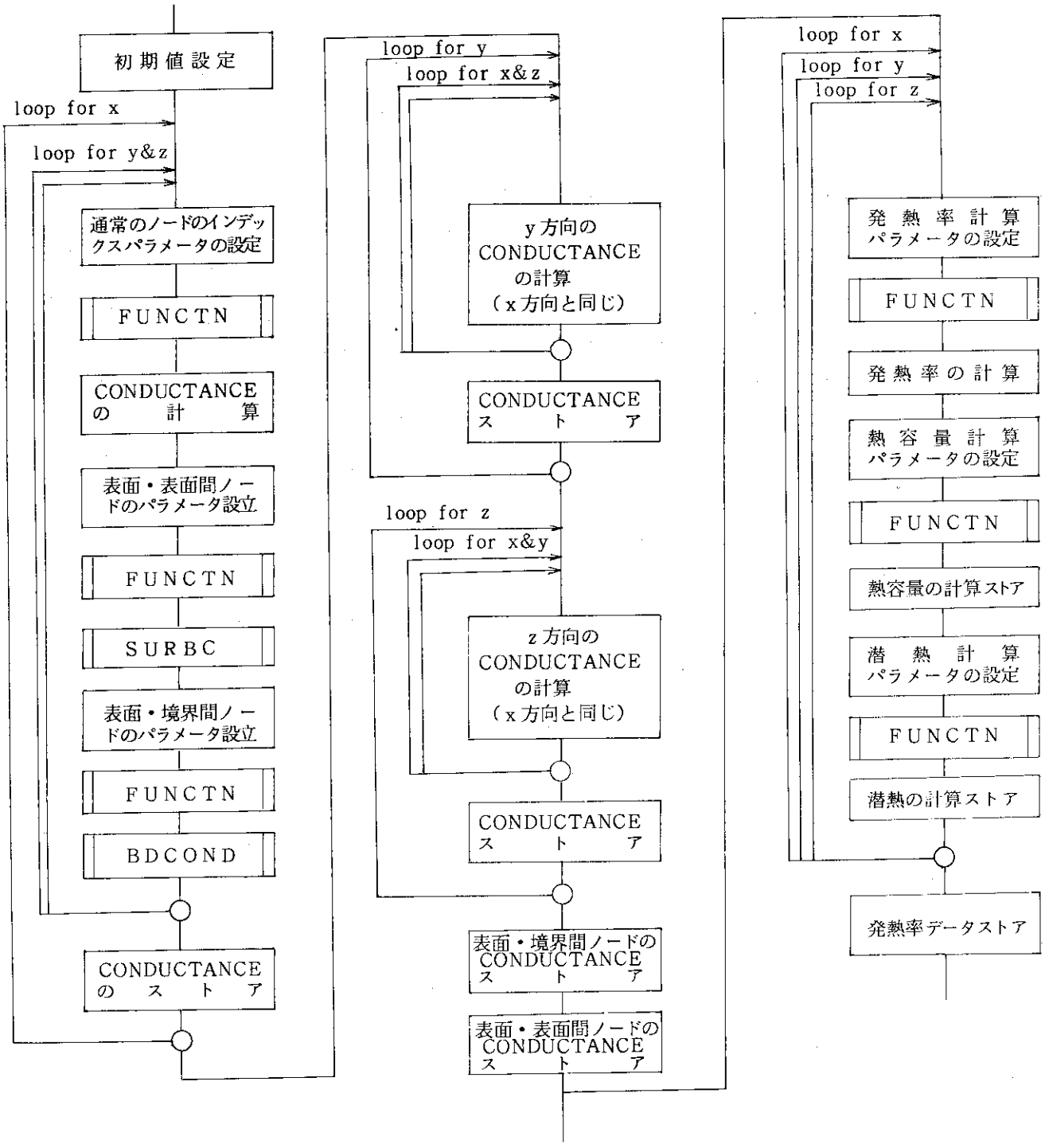
とすることによってIF文が省略できる。ここで，G1，G2，……，G7はコモン配列でありサブルーチンSURBCでそれらを使用しているが，その代入文を省略したために値が送られなくなる。そこで，G1M，G2Mなどの配列要素を引数に追加することによってデータのやりとりを行うようにした。これらの変数は，サブルーチンTHRMPR内における配列の宣言方法，引用方法が変わっただけであり，実際のデータのアドレスなどが変わるわけではないので，他のサブルーチンには一切影響がない。

以上の変更を行った後，最外ループであるノードループを分割し最内に持ち込むことによるベクトル化を行った。なお，ベクトル化の方法はFig. 4.6の右半分に示したとおりである。VHRMPRのベクトル化後の流れ図をFig. 4.7に示す。

```
      REAL*8
     .  G1M(MAXRFG)   , G1P(MAXRFG)   ,
     .  G2M(MAXRFG)   , G2P(MAXRFG)   ,
           :                :
     .  G4M(MAXTFG)   , G4P(MAXTFG)   ,
     .  G5M(MAXZFG)   , G5P(MAXZFG)   ,
           :                :
      INTEGER
     .  NBDIN(MAXREG), NIN(MAXREG)    ,
     .  NBDOT(MAXREG), NOT(MAXREG)    ,
           :                :
     .  NILBR(MAXPTS), NILFR(MAXPTS),
     .  NIRBR(MAXPTS), NIRFR(MAXPTS),
     .  NOLBR(MAXPTS), NOLFR(MAXPTS),
     .  NORBR(MAXPTS), NORFR(MAXPTS),
           :                :
      COMMON /GFACTS/ G1(2) , G2(2) , ...
           :                :
      DIMENSION    ... , MREG(2,2,2) , ...
           :
           :
      DO 1000 N=1,NT
           :
           :
      G1(1)=G1M(IX)
      G1(2)=G1P(IX)
      G2(1)=G2M(IX)
      G2(2)=G2P(IX)
      G4(1)=G4M(JY)
      G4(2)=G4P(JY)
      G5(1)=G5M(KZ)
           :
      MREG(1,1,2)=NILFR(N)
      MREG(1,1,1)=NILBR(N)
      MREG(1,2,2)=NIRFR(N)
      MREG(1,2,1)=NIRBR(N)
      MREG(2,1,2)=NOLFR(N)
      MREG(2,1,1)=NOLBR(N)
      MREG(2,2,2)=NORFR(N)
      MREG(2,2,1)=NORBR(N)
           :
           :
      DO 690 I=1,2
           :
  610 DO 651 J=1,NGYP1
           :
      DO 651 K=1,NGZP1
           :
      NOREG=MREG(I,J,K)
      IF(NOREG.LT.1)GO TO 630
           :
           :
      MATBC      =  MATL(MREG(3-I,J,K))
      MATBC2     =  MATS(MREG(3-I,J,K))
      NQ         =  NQUAD(3-I,J,K)
      IF(I.EQ.1)NBC=NBDOT(NOREG)
      IF(I.EQ.1)NBC2=NOT(NOREG)
      IF(I.EQ.2)NBC=NBDIN(NOREG)
      IF(I.EQ.2)NBC2=NIN(NOREG)
           :
   651 CONTINUE
           :
   690 CONTINUE
           :
           :
  1000 CONTINUE
           :
```

```
      REAL*8
     .  G1M(MAXRFG,2)  , G1P(1)        ,
     .  G2M(MAXRFG,2)  , G2P(1)        ,
           :                :
     .  G4M(MAXTFG,2)  , G4P(1)        ,
     .  G5M(MAXZFG,2)  , G5P(1)        ,
           :                :
      INTEGER
     .  NBDIN(MAXREG,2,2), NIN(1)      ,
     .  NBDOT(1)       , NOT(1)        ,
           :                :
     .  MREGT(MAXPTS,2,2,2), NILFR(1)  ,
     .  NIRBR(1)       , NIRFR(1)      ,
     .  NOLBR(1)       , NOLFR(1)      ,
     .  NORBR(1)       , NORFR(1)      ,
           :                :
      COMMON /GFACTS/ G1(2) , G2(2) , ...
           :                :
      DIMENSION    ... , MREG(2,2,2) , ...
           :
           :
      DO 1000 N=1,NT
           :
           :
           :
           :
      DO 690 I=1,2
           :
  610 DO 651 J=1,NGYP1
           :
      DO 651 K=1,NGZP1
           :
      NOREG=MREGT(N,K,J,I)
      IF(NOREG.LT.1)GO TO 630
           :
           :
      MATBC      =  MATL(MREGT(N,K,J,3-I))
      MATBC2     =  MATS(MREGT(N,K,J,3-I))
      NQ         =  NQUAD(3-I,J,K)
      NBC        =  NBDIN(NOREG,1,3-I)
      NBC2       =  NBDIN(NOREG,2,3-I)
           :
   651 CONTINUE
           :
   690 CONTINUE
           :
           :
  1000 CONTINUE
           :
```

Fig. 4.6 Rewrite for a DO loop in THRMPR



1) 各□ごとにノード数　NTでベクトル化している。

2) 各サブルーチンCALLはNT回行うのではなく，1回だけ呼びサブルーチン内でDOループによる繰り返しを行っている　そのため実際にはサブルーチン名をそれぞれFUNCT2，SURBC2，BDCON2に変更している。

Fig. 4.7　Program structure of the vectorized THRMPR

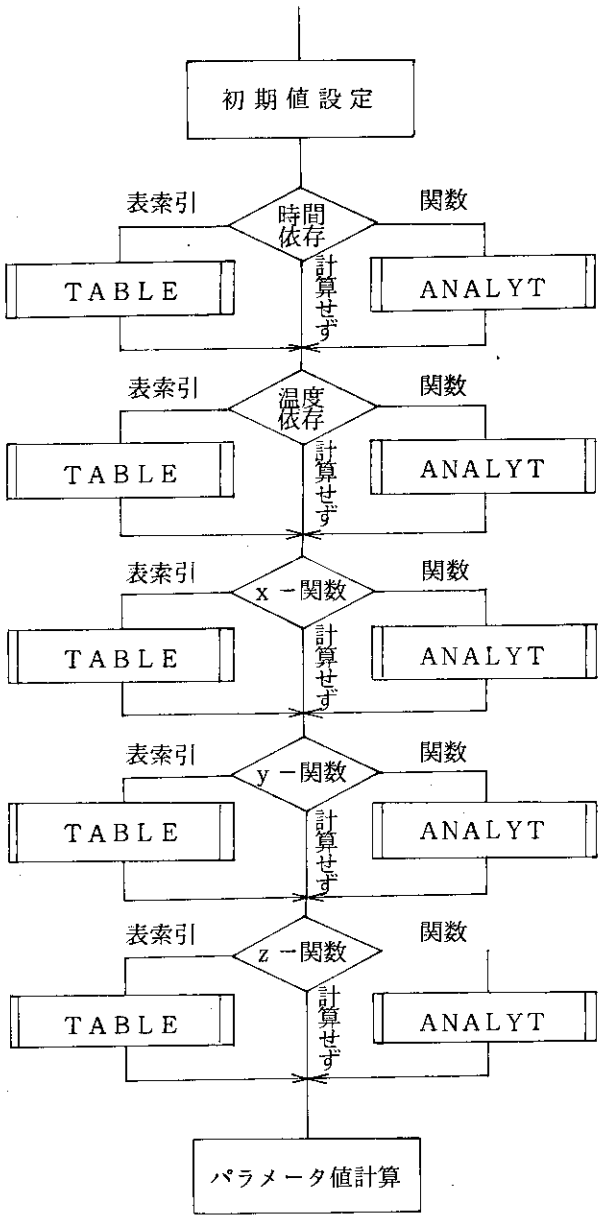
(2) FUNCTN，その他

サブルーチンFUNCTNは，時間，温度などに依存する各パラメータ値を計算するサブルーチンで，ここからテーブルデータの線型内捜サブルーチン又は，関数計算サブルーチンを呼ぶようになっている。関数式は次のようになっている。

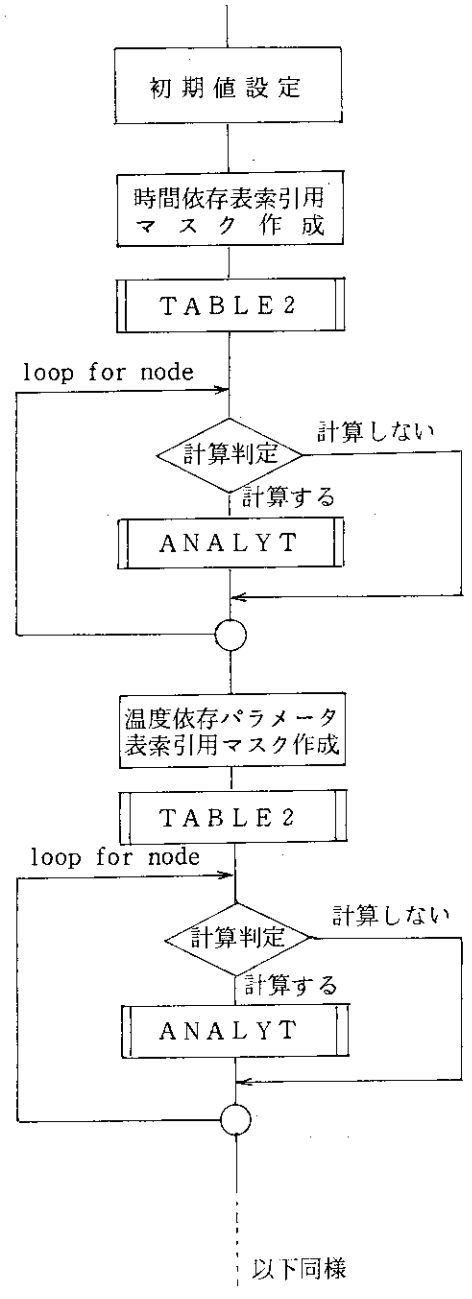
$$F(v)=A_1+A_2 v+A_3 v^2+A_4 \cos(A_5 v)+A_6 \exp(A_7 v)$$
$$+A_8 \sin(A_9 v)+A_{10} \log_e(A_{11} v)$$

さらに，関数計算サブルーチンからはユーザー定義関数サブルーチンも呼ぶことができるようになっている。THRMPRでのノードループをFUNCTNに持ち込むことによって，初期設定の部分と最後のパラメータ値計算部分がベクトル化可能となった。また同様にサブルーチンTABLEにもDOループを送り込むことによって内捜計算部分がベクトル化可能となった。また同様にサブルーチンTABLEにもDOループを送り込むことによって内捜計算部分がベクトル化できるようになった。ただし，サブルーチンFUNCTN及びTABLE(PRETABのエントリ）は他のサブルーチンからもCALLしているためそのまま書き換えるわけにもいかないので，新たにFUNCT 2というサブルーチンを作成，TABLE 2というエントリールーチンをPRETABに追加しこれらをベクトル化した。FUNCTNのベクトル化前の流れ図と，ベクトル化後の流れ図をFig. 4.8に示す。

サブルーチンSURBC及びBDCONDについては，ベクトル長その他を調査した結果，ベクトル化せずそのままとした。



(a) 変 更 前



(b) 変 更 後

(b)において □ ごとにノード数：NTでベクトル化している。

Fig. 4.8 Restructure method of FUNCTN

5. 熱計算とのインターフェイスPREPおよびPOINTSサブルーチンのベクトル化

5.1 プログラムの分析

JT－60 NBI加熱装置の熱解析データ（Table 2.1右欄）を用いて計算実行時のタイム・コストを分析したところ，Fig. 2.4に示すとおりとなった。NBI加熱計算の高速化をはかるには，ベクトル化第2版に対し，新たにタイム・コストの高いPREPとPOINTSをベクトル化する必要がある。

サブルーチンPREPでは，精密ノード（fine node）毎に得た温度値を，入力データで与えたノード（gross node）に編集するのに計算時間がかかる。これに関連するのは3つのDOループで，これらで約99％のタイム・コストを持つ。また，サブルーチンPOINTSでは，各ノード点の隣接ノードとの対応表作成に大部分の時間が費される。これらのループ(Table 5.1参照)が効率良いベクトル計算となればよい。

Table 5.1 DO loops in PREP and POINTS

Subroutine	DO loop	Time cost(*)	Contents
PREP	DO 400	38.2%	Locate the nodes at intersections of gross grid lines
	DO 530	6.2	Determine the temperature of the nodes at intersections of fine grid lines
	DO 330	55.6	
POINTS	DO 550	99.6	Determine the neighbours of nodes

*) Time cost within a subroutine

5.2 ベクトル化のためのプログラム変更

サブルーチンPREPでは，Table 5.1で示した3つのDOループのベクトル化ができれば良い。Fig. 5.1で示す最内のDOループであるDO 400はステートメント420への飛び出しが2個以上あるという理由で，FACOM FORTRAN 77/VPではベクトル化できない。しかし，このDOループから，"2個以上の飛び出しがある"という形式的なベクトル化の障害を取り除いただけでは，入り組んだIF文の重複のため計算速度は上がらない。

DOループ内のIF文は，形状に関する判定であり，これは計算途中で不変であるから，前もって場合分けが可能である。これを利用して，Fig. 5.2で示す方法でベクトル化した。まず,はじめに，形状が，(1) R－θ or X－Y形状，(2) R－Z or X－Z形状，(3) Y－Zまたはθ－Z形状かを判断し，各々IGFLG＝1, 2, or 3とおく。次に，IGFLG毎にDOループを作成する。

上記の変更は，IGFLG毎にDOループ内の演算量の大きな削減となり，スカラー計算でも高速化されているはずである。

サブルーチンPOINTSではFig. 5.3で示すDO 560に大部分の計算時間を要する。これも形式的にはDO 560への飛び出しが2回以上あるという理由でベクトル化できない。図で見るとおり，NB(NTP, M)が0かどうか，つまり，NTP番目の隣（1～6方向）が存在するかどうかの判定を，DO 550の外に出す（Fig. 5.3下段　参照）と万事うまく行く。

```
                ----*----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8  =EXECUTIONS=          =TRUE=       =COST=
1 2              C       Y-Z OR TH-Z GEOMETRY                                                  00037400
1 2             S  304 IF (DABS(Z(K)-ZG(KK)) .GT. 1.0D-6)  GO TO 200                          ISN=0267          0          0( 0.0%)          0
1 2             S  301 II=1                                                                   ISN=0268       2550                            2550
1 2 3-----------------DO 199 IK=2,IMAX                                                        ISN=0269       2550                            2550
1 2 3                  I=IK                                                                   ISN=0270      73950                           73950
1 2 3                  IF(IMAX.GT.1)GO TO 405                                                 ISN=0271      73950      73950(100.%)         221850
1 2 3            C+++++++++++++++++++++++++THE FOLLOWING STATMENT CAUSES A COMPILER ERROR     00038000
1 2 3            C++++++++++++++++++++++++ ON THE PDP10.                                      00038100
1 2 3                  I=1                                                                    ISN=0272          0                               0
1 2 3                  GO TO 407                                                              ISN=0273          0                               0
1 2 3                405 IF ((.NOT.LRT) .AND. (.NOT.LRZ))  GO TO 406                          ISN=0274      73950          0( 0.0%)         295800
1 2 3                  IF(DABS(R(I)-RG(II+1)) .GT. 1.0D-6) GO TO 199                          ISN=0275      73950      17850(24.1%)         387600
1 2 3                  GO TO 407                                                              ISN=0276      56100                           56100
1 2 3                406 IF (DABS(TH(I)-THG(II+1)) .GT. 1.0D-6)  GO TO 199                    ISN=0277          0          0( 0.0%)          0
1 2 3 4----------407 DO 400 N=1,NT                                                             ISN=0278      56100                           56100
1 2 3 4                IF(LRT)GO TO 410                                                       ISN=0279  236806510  236806510(100.%)  473613020
1 2 3 4                IF (LRZ)  GO TO 409                                                    ISN=0280          0          0( 0.0%)          0
1 2 3 4          C       Y-Z OR TH-Z GEOMETRY.                                               00039100
1 2 3 4                IF (MIL.EQ.NTPI(N).AND.K.EQ.NTPK(N).AND.I.EQ.NTPJ(N)) GO TO 420        ISN=0281          0          0( 0.0%)          0
1 2 3 4                GO TO 400                                                              ISN=0282          0                               0
1 2 3 4          C        R-Z OR X-Z GEOMETRY.                                               00039400
1 2 3 4              409 IF (MIL.EQ.NTPJ(N) .AND. K.EQ.NTPK(N) .AND. I.EQ.NTPI(N))GO TO 420ISN=0283          0          0( 0.0%)          0
1 2 3 4                GO TO 400                                                              ISN=0284          0                               0
1 2 3 4          C        R-THETA OR X-Y GEOMETRY.                                           00039700
1 2 3 4              410 IF(.NOT.L1D)GO TO 415                                              ISN=0285  236806510  236806510(100.%)  710419530
1 2 3 4                IF(K+1 .EQ. NTPI(N))GO TO 420                                          ISN=0286          0          0( 0.0%)          0
1 2 3 4                GO TO 400                                                              ISN=0287          0                               0
1 2 3 4              415 IF (MIL.EQ.NTPK(N) .AND. K.EQ.NTPJ(N) .AND. I.EQ.NTPI(N))GO TO 420ISN=0288  236806510      43680( 0.0%) 1420882740
1 2 3 +----------400 CONTINUE                                                                 ISN=0289  236762830                       236762830
1 2 3                                                                                                                 DO   400;38.15%(  2841734220)
1 2 3                                                                                                                 AVERAGE DO-LOOP LENGTH =  4221
1 2 3                  LSTNOD(KK,II)=0                                                        ISN=0290      12420                           12420
1 2 3                  GO TO 430                                                              ISN=0291      12420                           12420
1 2 3                420 LSTNOD(KK,II)=N                                                      ISN=0292      43680                           43680
1 2 3                430 CONTINUE                                                           ISN=0293      56100                           56100
1 2 3                  II=II+1                                                              ISN=0294      56100                          112200
1 2 +------------199 CONTINUE                                                             ISN=0295      73950                           73950
1 2                                                                                                                   DO   199;38.16%(  2843082840)
1 2                                                                                                                   AVERAGE DO-LOOP LENGTH =    29
1 2              S     KK=KK+1                                                             ISN=0296       2550                            5100
1 +-----------S--200 CONTINUE                                                             ISN=0297       2850                            2850
1                                                                                                                     DO   200;38.16%(  2843121990)
1                                                                                                                     AVERAGE DO-LOOP LENGTH =    19
```

Fig. 5.1 Typical DO loop to be vectorized in PREP (Original version)

```
      C  REWRITTEN FOR VECTORIZATION                                    03540000
000249        IF (LRT) THEN                                             03550000
000250        IF(.NOT.L1D) THEN                                         03560000
000251        IGFLG=1                                                   03570000
000252        ELSE                                                      03580000
000253        IGFLG=2                                                   03590000
000254        ENDIF                                                     03600000
000255        ELSE                                                      03610000
000256        IF(LRZ) THEN                                              03620000
000257        IGFLG=3                                                   03630000
000258        ELSE                                                      03640000
000259        IGFLG=4                                                   03650000
000260        ENDIF                                                     03660003
000261        ENDIF                                                     03670000
```

Classify the geometries beforehand

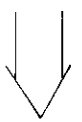
```
000291     407 GO TO (401,402,403,404),IGFLG                            04220004
      C        R-THETA OR X-Y GEOMETRY.                                04230004
000292 V   401 DO 4011 N=1,NT                                           04240001
000293 V   415 IF (MIL.EQ.NTPK(N) .AND. K.EQ.NTPJ(N) .AND. I.EQ.NTPI(N))GO TO 42004250001
000294 V  4011 CONTINUE                                                 04260001
000295        GO TO 4020                                                04270002
000296 V   402 DO 4021 N=1,NT                                           04280001
000297 V       IF(K+1 .EQ. NTPI(N))GO TO 420                            04290001
000298 V  4021 CONTINUE                                                 04300001
000299        GO TO 4020                                                04310002
```

Devide in DO loop into the gcometries

```
      C        R-Z OR X-Z GEOMETRY.                                    04320004
000300 V   403 DO 4031 N=1,NT                                           04330001
000301 V   409 IF (MIL.EQ.NTPJ(N) .AND. K.EQ.NTPK(N) .AND. I.EQ.NTPI(N))GO TO 42004340001
000302 V  4031 CONTINUE                                                 04350001
000303        GO TO 4020                                                04360002
      C       Y-Z OR TH-Z GEOMETRY.                                     04370004
000304 V   404 DO 4041 N=1,NT                                           04380001
000305 V       IF (MIL.EQ.NTPI(N).AND.K.EQ.NTPK(N).AND.I.EQ.NTPJ(N)) GO TO 420   04390001
000306 V  4041 CONTINUE                                                 04400001
000307    4020 LSTNOD(KK,II)=0                                          04410002
000308        GO TO 430                                                 04420000
000309     420 LSTNOD(KK,II)=N                                          04430000
000310     430 CONTINUE                                                 04440000
000311        II=II+1                                                   04450000
000312     199 CONTINUE                                                 04460000
000313        KK=KK+1                                                   04470000
000314     200 CONTINUE                                                 04480000
```

Fig. 5.2 Vectorized DO loop in PREP

```
1 2-----------0520 DO 560 M=1,6                                    ISN=0252       6937                    6937
1 2 3---------------DO 550 N=1,NT                                   ISN=0253      41622                   41622
1 2 3          C.NS  IF(NB(M,NTP))560,560,530                      00034700
1 2 3                IF(NB(NTP,M))560,560,530                      ISN=0254  143499545                143499545
1 2 3          C.NSO IF(NB(M,NTP)-NPT(N))550,540,550               00034900
1 2 3           0530 IF(NB(NTP,M)-NPT(N))550,540,550               ISN=0255  143497686                286995372
1 2 3          C.NSO NB(M,NTP)=N                                   00035100
1 2 3           0540 NB(NTP,M)=N                                   ISN=0256      38186                   38186
1 2 3                GO TO 560                                     ISN=0257      38186                   38186
1 2 +----------0550 CONTINUE                                       ISN=0258  143459500                143459500
1 2                                                                          DO   550;99.65%(   574072411)
1 2                                                                          AVERAGE DO-LOOP LENGTH =   3447
1 2            C.NS  NB(M,NTP)=0                                   00035500
1 2                  NB(NTP,M)=0                                   ISN=0259      1577                    1577
1 +------------0560 CONTINUE                                       ISN=0260     41622                   41622
1                                                                            DO   560;99.65%(   574122547)
1                                                                            AVERAGE DO-LOOP LENGTH =      6
+------------S-0570 CONTINUE                                       ISN=0261      6937                    6937
                                                                             DO   570;99.79%(   574928962)
                                                                             AVERAGE DO-LOOP LENGTH =   6937
```



```
000252     0520 DO 560 M=1,6                                       03450000
000253          IF(NB(NTP,M))560,560,530                           03460001
000254 V    530 DO 550 N=1,NT                                      03470001
          C.NS  IF(NB(M,NTP))560,560,530                           03480000
          C.NSO IF(NB(M,NTP)-NPT(N))550,540,550                    03490000
000255 V        IF(NB(NTP,M)-NPT(N))550,540,550                    03500001
          C.NSO NB(M,NTP)=N                                        03510000
000256     0540 NB(NTP,M)=N                                        03520000
000257          GO TO 560                                          03530000
000258 V   0550 CONTINUE                                           03540000
          C.NS  NB(M,NTP)=0                                        03550000
000259          NB(NTP,M)=0                                        03560000
000260     0560 CONTINUE                                           03570000
000261 S   0570 CONTINUE                                           03580000
```

Fig. 5.3 Rewrite of DO 550 in POINTS

6．ベクトル化に必要な主記憶量

6.1 ベクトル化に必要な作業用領域

HEATING 6のベクトル化版第1，第2版によって新たに作られたベクトル化用作業用配列は，サブルーチンごとに別々に取られており，またサブルーチン内においても，1つのDOループ内で局所的に使用される変数であっても，名前が異なっていれば別の配列を宣言していた。このため，ノード数の多い問題を扱う場合は，これらの配列の合計は巨大なものとなった。

これらの問題を解決するために以下の変更を行った。

(1) オリジナル版は整合配列を使用しており，1つの大きな配列をサブルーチンHEATN6において分割して各サブルーチンに送り使用しているが，ベクトル化用の作業用配列もこの中に組み込み，1か所宣言を変えるだけで通るようにした（サブルーチンMAIN01）。

Table 6.1 Working storage required for vectorization

SUBROUTINE	TYPE	AREA NAME	SUBROUTINE	TYPE	AREA NAME
THRMPR	I*4	ICOUN¥(MAXPTS)			
BDCON2		IKOUN¥(MAXPTS)			
SURBC2		MATS¥ (MAXPTS)			
		NQ¥ (MAXPTS)			
		MATB¥ (MAXPTS)			
		NBC¥ (MAXPTS)			
		NBC2¥ (MAXPTS)			
		NCONT¥(MAXPTS)			
		NGN¥ (MAXPTS)			
		NGNR¥ (MAXPTS)			
		NGNZ¥ (MAXPTS)			
		NGNTH¥(MAXPTS)			
		NGNTM¥(MAXPTS,8)			
		NFT¥ (MAXPTS)			
		I2O¥ (MAXPTS)			
		I¥LST1(MAXPTS)			
		I¥LST2(MAXPTS)			
		NDUM¥ (MAXPTS)			
		NDMNO¥(MAXPTS,NDIM2)			
		NSCON¥(MAXPTS)			
		NSNOD¥(MAXPTS,NDIM2)			
	R*8	ZERO¥ (MAXPTS)			
		SMKK¥ (MAXPTS)	TRANO	I*4	NXX1 (3)
		TEMP¥ (MAXPTS)	CHCKBD		NXX2 (MAXPTS,3)
		RHO¥ (MAXPTS)			NYY1 (3,NDIM2)
		COND¥ (MAXPTS)			NYY21 (MAXPTS,NDIM2)
		CONDU¥(MAXPTS)			NYY22 (MAXPTS,NDIM2)
		CONDB¥(MAXPTS)			NYY23 (MAXPTS,NDIM2)
		CPR¥ (MAXPTS)			NSMKTX(MAXPTS)
		R1¥ (MAXPTS)		R*8	SMHTX (MAXPTS)
		TH1¥ (MAXPTS)			SMKTX (MAXPTS)
		Z1¥ (MAXPTS)			TT2X (MAXPTS)
		V¥ (MAXPTS)			SMHX (MAXPTS)
		GEN¥ (MAXPTS)			SMKK (MAXPTS)
		CSUBP¥(MAXPTS)			SMKT3X(MAXPTS)
		VG¥ (MAXPTS,8)			SMTVGX(MAXPTS)
		CDTY¥ (MAXPTS,NDIM2)			
		CONDT¥(MAXPTS,NDIM2)	CHCKBD	I*4	NCKBD (MAXPTS)
	L*4	UDEC¥ (MAXPTS)			NINDEX(MAXPTS)
		FINN¥ (MAXPTS)			NADD (MAXPTS)
		CCOND¥(MAXPTS)			
		CCNDB¥(MAXPTS)			
		¥EXFL1(MAXPTS)			
		¥EXFL2(MAXPTS)			
PREFUN	I*4	I¥TAB1(MAXPTS)			
(TABLE2)		I¥TAB2(MAXPTS)			
PREFUN		I¥END (MAXPTS)			
(FUNCT2)		I¥ (MAXPTS)			
		J¥ (MAXPTS)			
		K¥1 (MAXPTS)			
		K¥2 (MAXPTS)			
	R*8	F1¥ (MAXPTS)			
		F2¥ (MAXPTS)			
		F3¥ (MAXPTS)			
		F4¥ (MAXPTS)			
		F5¥ (MAXPTS)			
		TIM¥ (MAXPTS)			

これによって実行時に領域が足りない場合，出力されるメッセージに従って配列Dを取り直すだけで実行可能となる。

(2) 各DOループごとに独立した作業用配列を使用している場合，同じ配列を使用するようにした。

```
      DO   100   I＝1，N
       W1(I)=……
         ……
       A(I)=…＊W1(2)
         ……
100   CONTIUNE
         ⋮
      DO   200   I＝1，N
         ……
       W2(I)=……
       B(I)=W2(I)……
         ……
200   CONTINUE
```

```
      DO   100   I＝1，N
       W1(3)=……
         ……
       A(I)=…＊W1(I)
         ……
100   CONTINUE
         ⋮
      DO   200   I＝1，N
         ……
       W1(I)=……
       B(I)=W1(I)……
         ……
200   CONTINUE
```

(3) 各サブルーチン毎の作業用配列が独立している場合，配列定義を重ねることによって同じ配列を別名で使うようにした。

ベクトル化第3版において，ベクトル化のために使用している作業用配列はTable 6.1で示すとおりである。ここで，点線NXX1(1)以下は，SMKK(1)以下と重なって使用されていることを示す。

結局，ベクトル化に必要な作業用配列の数はTable 6.1で示すように約80個であるから，バイト数にすると以下のようになる。

$8 \times 80 \times$（ノード数）

ただし，ノード数は3次元精密ノード数を指す。

JT－60 NBI加熱計算ではノード数が約7,000だから約4.5MB余分に必要である。

ベクトル化第3版で主記憶量を8MB以下で実行するには，7,000ものノード数に対しては後に示すHEATING－JRの新機能である図形化部分を抑止する必要がある。

6.2 HEATING 6 ベクトル化版に要する全作業用領域

HEATING 6 ベクトル化版に必要な作業用領域は，Fig. 6.1で示すように2倍精度配列D上に一括して取られている。それらは，次の3項目から成っている。

① オリジナルHEATING 6コードの作業領域[1]

② HEATING－JRコード（原研改良版）のために付加された作業用領域[2]

③ ベクトル化のために付加された作業用領域

上記3項目は，詳しくは，次のような順番に置れている。

オリジナル・コードの作業領域

HEATING 6－JRのための作業領域

オリジナル・コードの作業領域（続き）

ベクトル化のための作業領域

(1) オリジナル・コードの作業領域[1]

必要な作業領域バイト数は，Fig. 6.2のREGIONで示される。これを計算するのに必要なパラメータの定義は，Fig. 6.2およびFig. 6.3で示される。また，Fig. 6.3のディフォルト値は，NAMELIST形式で変更できる（HEATING 6マニュアル[1]の入力作成部分を参照)。

(2) HEATING－JRに必要な作業領域[2]

入力データのチェックと計算結果の図形化処理，計算結果のSAP 5コードへの結合などのために，原研側で改良がなされた。これらの改良は，はじめ，HEATING 5コードでなされ，HEATING 5－JRコードとして整備され公開された。HEATING 6についても，これに準じた改良がなされており，HEATING 6－JRコードとして今後公開されるであろう。さてこれに必要な領域は，次のとおりである。

8｛6×NREG＋14＊IJLMAX＋4＊ID 23＊IJLMAX｝バイト

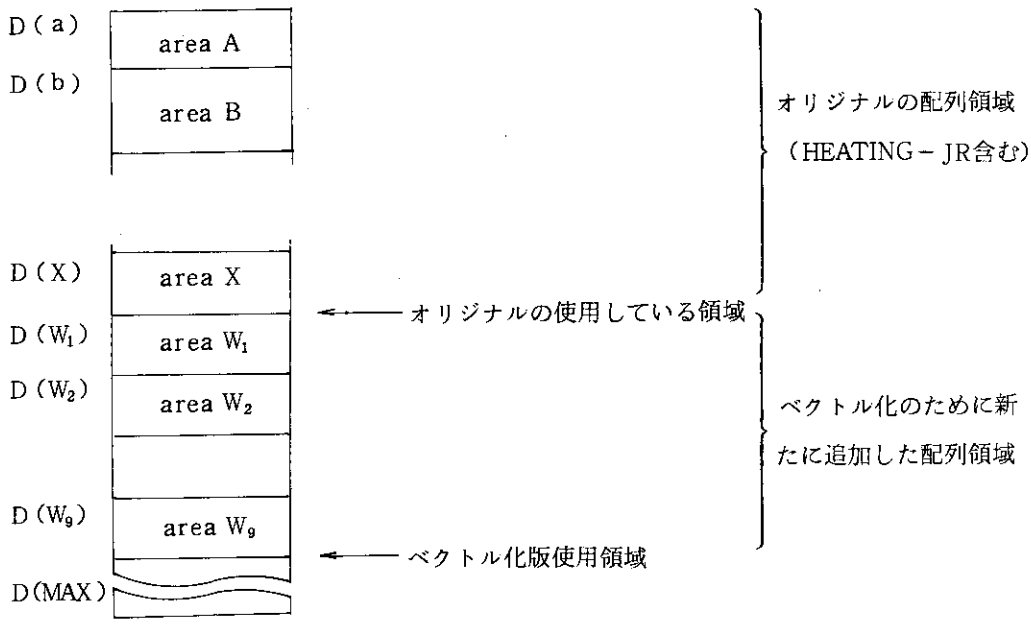
ここで，NREG：物質数，IJLMAX：精密ノード数，ID 23＝4（2次元），8（3次元）

(3) ベクトル化に必要な作業用領域

前節で示されたとおり

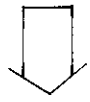
8×80×（ノード数）バイト

付記；(2)，(3)部分の作業用領域の重複（オーバレイ）が可能かどうかを今後検討する。




```
SUBROUTINE HEATN6 (D)
DIMENSION D (MAX)
      ⋮
  a = 1
  b = a + leng a
    ⋮
  w1 = x + leng x
  w2 = w1 + leng w1
    ⋮
  last = wn + leng wn


CALL SUB (D(a), D(b), ···, D(w1), ··· D(wn))
```



```
SUBROUTINE SUB (A, B, ···, W1, ···, Wn)
DIMENSION A (leng a), B (leng b), ·····, Wn (leng Wn)
```

Fig. 6.1 Working storage allocation for the 3rd. vector version

NTPNW = Maximum of (MAXRFG*MAXTFG, MAXRFG*MAXZFG, MAXTFG*MAXZFG)

MXNDGL = Maximum of (MAXPTS, NTPNW)

MAXFGL = Maximum of (MAXRFG, MAXTFG, MAXZFG)

MAXGGL = Minimum of (MAXGGL, MAXFGL)

MG2TPN = Minimum of (NTPNW, MAXGGL*MAXGGL)

MAXLDA = [(MWIDTH - 1)/2] + 1 where [a/b] represents the largest integer less than or equal to a/b

MAXCP = 5

MAXPAR = 11

A = { 0 if IMPFLG = .FALSE.
{ 16*(MAXBDC + MAXPBT) if IMPFLG = .TRUE.

B = { 0 if FIN = .FALSE.
{ 48*MAXBDC if FIN = .TRUE.

C = { 0 if HGEN = .FALSE.
{ 20*MAXPTS if HGEN = .TRUE.

D = { 0 if CEP = LEVI = IMPFLG = .FALSE
{ 8*MAXPTS if CEP = .TRUE. or LEVI = .TRUE. or IMPFLG = .TRUE.

E = { 0 if LBOUND = .FALSE.
{ 2*MAXPTS if LBOUND = .TRUE.

F = { 0 if MELTFG = .FALSE.
{ (21+MAXCP)*MAXPTS if MELTFG = .TRUE.

G = { 0 if LEVI = IMPFLG = .FALSE.
{ MAXPTS if LEVI = .TRUE. or IMPFLG = .TRUE.

H = { 0 if MELTFG = IMPFLG = .FALSE.
{ 3*MAXPTS if MELTFG = .TRUE. or IMPFLG = .TRUE.

I = MXNDGL + G + H

J = { 0 if DIRECT = .FALSE.
{ 4*MAXPTS if DIRECT = .TRUE. and MWIDTH≤7
{ MAXLDA*MAXPTS if DIRECT = .TRUE. and MWDITH>7

L = Maximum of (F, J)

SIZE = 114*MAXREG + 90*MAXMAT + 22*MAXINT
+ 46*MAXHGN + 121*MAXBDC + 14*MAXPBT + A + B
+ 88*MAXRFG + 24*MAXTFG + 24*MAXZFG
+ 28*MAXTGL + (30 + 2*MAXGGL)*MAXGGL
+ 2*MG2TPN + (4 + 8*MAXPAR)*MAXANA
+ (6+16*MAXPRS)*MAXTBL + 8*MAXPRT
+ 2*MAXNSN + 8*MAXSPC + 2*MAXSPL
+4*MAXSUR + (91+20*NDIMEN)*MAXPTS
+ C + D + E + F + 8*L

M = Change in region size in bytes due to modifying BLKSIZE for any of the logical units

REGION = [(SIZE+M)/1024+366]K bytes

Fig. 6.2 Working storage required for the original code[1]

Variable	Default Value	Definition of Variable
MAXANA	5	Analytical Functions
MAXBDC	5	Boundary Condition Functions
MAXCP	*	Materials with Phase Change (Set to 5 by Code)
MAXFGL	*	Fine Grid Lines Along Any Axis (Calculated by Code)
MAXGGL	10	Gross Grid Lines Along Any Axis (May be Decreased by Code if MAXFGL is Less)
MAXHGN	5	Heat Generation Functions
MAXINT	5	Initial Temperature Functions
MAXMAT	5	Materials
MAXNSN	5	Nodes for Special Monitoring of Temperatures
MAXPAR	*	Parameters in an Analytical Function (Set to 11 by Code)
MAXPBT	1	Position-Dependent Boundary Temperature Nodes
MAXPRS	10	Pairs in Each Tabular Function
MAXPRT	10	Printout Times
MAXPTS	50	Lattice Points (Nodes)
MAXREG	5	Regions
MAXRFG	50	Radial (or X) Fine Grid Lines
MAXSPC	10	Printout Times for Selected Planes
MAXSPL	20	Planes for Selected Output
MAXSUR	50	Surface-to-Surface Connectors
MAXTBL	5	Tabular Functions
MAXTFG	50	Theta (or Y) Fine Grid Lines
MAXZFG	50	Z Fine Grid Lines

*Indicates value will be calculated by HEATING6.

Fig. 6.3 parameters Defining Array Sizes for Variably Dimensioned Arrays[1]

7. 実行時の制御文とファイル

7.1 ＪＣＬ事例

Fig. 7.1で示すとおりである。

7.2 ファイル

(1) ノード数が少ない場合（5MB版）

ソース・ファイル（ベクトル化版）・J1431. HEAT6W. FORT77

ロード・モジュール・ファイル（ベクトル化版）・J1431. HEAT6W. LOAD

5MB（バイト）版では1,000ノード程度取扱うことができる。

(2) ノード数が多い場合（8MB版）

ソース・ファイル（ベクトル化版）・J1431. HEAT6V. FORT77

ロード・モジュール・ファイル（ベクトル化版）・J1431. HEAT6V. LOAD

8MB版では7,000ノード程度取扱える。ただし，図形出力処理は不可能である。

現在，5MB版と8MB版を別々に作成しているが，主プログラムMAIN01のDの配列長の変更と，配列長を定義するパラメータを変更すれば，各々の問題に適合した利用方法が可能である。

```
EDIT --- J1431.HEAT6.CNTL(VPGOB) - 01.27 ------------------- COLUM
COMMAND ===>                                                 SCRO
****** ***************************** TOP OF DATA ***************V
000100 //JCLG  JOB
000200 //      EXEC  JCLG
000300 //SYSIN  DD   DATA,DLM='++'
000400 // JUSER 00031431,MI.ISHIGURO,0341.01
000500   T.5 C.5 W.2 I.3   SRP
000600   OPTP  PASSWORD= ,NOTIFY=J1431,CLASS=1
000610 /* EXEC FORT77,SO=J1431.HEAT6W,A='ELM(MAIN01)',B=SOURCE
000620 /* EXEC  LKEDUP,LM=J1431.HEAT6W,DISP=SHR
000700 //  EXEC VP
000800 //  EXEC LMGO,LM=J1431.HEAT6W
000900 //FT05F001 DD DSN=J8770.HEAT85.DATA(TEST20),DISP=SHR
001200 //FT10F001 DD DSN=&&FX10,DISP=(NEW,DELETE),
001300 //             SPACE=(TRK,(20,5),RLSE),UNIT=WK10
001400 //FT20F001 DD DSN=&&FX20,DISP=(NEW,DELETE),
001500 //             SPACE=(TRK,(20,5),RLSE),UNIT=WK10
001600 //FT30F001 DD DSN=&&FX30,DISP=(NEW,DELETE),
001700 //             SPACE=(TRK,(20,5),RLSE),UNIT=WK10
001800 //FT40F001 DD DSN=&&FX40,DISP=(NEW,DELETE),
001900 //             SPACE=(TRK,(20,5),RLSE),UNIT=WK10
002000 ++
002100 //
```

Fig. 7.1 JCL of HEATING6 running on VP-100

8. おわりに

熱解析の代表的なコードのベクトル化が完成し，その結果，3～4倍高速化できた。ベクトル化版コードは，現在JT－60中性粒子入射加熱装置に対する実験解析に多く用いられている他，安全解析研究所で核燃料輸送容器の安全性に関する計算に利用が開始された。

計算センターでは，原研でよく使用されている大型コードのベクトル化をはかるために，ベクトル化作業を支援してきた。HEATING 6コードのベクトル化も計算センターにおけるベクトル化支援の一環をなす。原子力コードのベクトル化版利用では，安全性分野よりも核融合分野の方が積極的である。その理由は，海外から入手したコードの利用が安全性分野に多いことによっている。今回のHEATING 6コードのベクトル化版は，当初は安全性分野での活発な利用を期待し作業を開始したが，意図とは別に核融合研究の方での大型計算に供することになった。SAP－V，TRUMPコードのベクトル化版と合わせ，熱解析の代表的コードとして今後の有効利用を期待する。

現在のところ，HEATING 5－JR，HEATING 6　M 380版，およびHEATING 6ベクトル化版が公開されており，全2者は幾島，後者は石黒の管理となっている。

今後，原研での改良版HEATING 6－JRベクトル化版として統一バージョンを公開，管理していきたい。

付　　記

JT－60 NBI加熱計算に関しては，定常計算部分のベクトル化が充分なされていない。定常計算の比率の高い問題を取扱う場合は，この部分のベクトル化を強化することによって，現在得ている4倍の速度向上が8倍程度になり得る。しかし，そのためにはさらに2Mバイト程度の作業領域が必要となる。ベクトル化版の主記憶容量を8Mバイト以下にすることが今回の目標の1つのため（原研の運用では8Mバイト以上のジョブは特別ジョブとなり許可が必要である），現状ではあきらめざるを得ない。

同じく，JT－60 NBI熱解析で定常計算を実行する場合，SOR法で解かれている温度分布計算が収束しないことがある。数値実験の結果，SORの加速係数を計算の進行に従ってコードで自動的に減じているため，小さくなり過ぎることがわかった。このような場合には，加速係数の減算をやめた方が良い（サブルーチンCALQLTのBETAの計算の改良）。また，SORの加速とは別に，ある条件下でAitken加速を行っているが，Aitken加速が充分効力を発揮しないうちに反復計算を打切ると計算結果がとんでもない値となることがある。

8. おわりに

熱解析の代表的なコードのベクトル化が完成し，その結果，3～4倍高速化できた。ベクトル化版コードは，現在JT－60中性粒子入射加熱装置に対する実験解析に多く用いられている他，安全解析研究所で核燃料輸送容器の安全性に関する計算に利用が開始された。

計算センターでは，原研でよく使用されている大型コードのベクトル化をはかるために，ベクトル化作業を支援してきた。HEATING 6コードのベクトル化も計算センターにおけるベクトル化支援の一環をなす。原子力コードのベクトル化版利用では，安全性分野よりも核融合分野の方が積極的である。その理由は，海外から入手したコードの利用が安全性分野に多いことによっている。今回のHEATING 6 コードのベクトル化版は，当初は安全性分野での活発な利用を期待し作業を開始したが，意図とは別に核融合研究の方での大型計算に供することになった。SAP－V，TRUMPコードのベクトル化版と合わせ，熱解析の代表的コードとして今後の有効利用を期待する。

現在のところ，HEATING 5－JR，HEATING 6　M 380版，およびHEATING 6ベクトル化版が公開されており，全2者は幾島，後者は石黒の管理となっている。

今後，原研での改良版HEATING 6－JRベクトル化版として統一バージョンを公開，管理していきたい。

付　　記

JT－60 NBI加熱計算に関しては，定常計算部分のベクトル化が充分なされていない。定常計算の比率の高い問題を取扱う場合は，この部分のベクトル化を強化することによって，現在得ている4倍の速度向上が8倍程度になり得る。しかし，そのためにはさらに2Mバイト程度の作業領域が必要となる。ベクトル化版の主記憶容量を8Mバイト以下にすることが今回の目標の1つのため（原研の運用では8Mバイト以上のジョブは特別ジョブとなり許可が必要である），現状ではあきらめざるを得ない。

同じく，JT－60 NBI熱解析で定常計算を実行する場合，SOR法で解かれている温度分布計算が収束しないことがある。数値実験の結果，SORの加速係数を計算の進行に従ってコードで自動的に減じているため，小さくなり過ぎることがわかった。このような場合には，加速係数の減算をやめた方が良い（サブルーチンCALQLTのBETAの計算の改良)。また，SORの加速とは別に，ある条件下でAitken加速を行っているが，Aitken加速が充分効力を発揮しないうちに反復計算を打切ると計算結果がとんでもない値となることがある。

謝　　辞

HEATING 6 ベクトル化版テストのために，入力データを提供頂きました臨界プラズマ加熱開発室　荒木政則氏および安全解析所　成田　猛氏に感謝します。また，本リポート投稿にあたり，記述の誤り等チェックいただきました計算センター藤井　実氏に感謝します。

参．考　文　献

(1) D.C. Elrod et al. ; HEATING 6 : A Multidimensional Heat Conduction Analysis with the Finite Difference Formation, NUREG/CR－0200, Volume 2, Section F10, ORNL/NUREG/CSD－2/V2 (1982).

(2) 幾島　毅，中里　力；HEATING 5－JR：有限差分法による非線型熱伝導計算プログラム，JAERI－M (1983).

(3) L. Edwords ; TRUMP : A Computer Program for Transient and Steady－State Temperature Distributions in Multidimensional Systems, LLL W－7405－Eng－45 (1972).

(4) K. J. Bathe et al. : SAP Ⅳ, A Structual Analysis Program for Static and Dynamic Response of Linear Systems, EERC 73－11 (1973).

(5) 差分法による非線型熱伝導計算プログラムHEATING 6のベクトル化，原子力データセンタ（private communication）(1985).

(6) 核融合研究開発の現状，P－43，日本原子力研究所（1984).

(7) 石黒美佐子，難波克光；差分法のベクトル計算，Vol. 24，No. 1 (1983).

謝　　辞

HEATING 6 ベクトル化版テストのために，入力データを提供頂きました臨界プラズマ加熱開発室　荒木政則氏および安全解析所　成田　猛氏に感謝します。また，本リポート投稿にあたり，記述の誤り等チェックいただきました計算センター藤井　実氏に感謝します。

参　考　文　献

(1) D.C. Elrod et al. ; HEATING 6 : A Multidimensional Heat Conduction Analysis with the Finite Difference Formation, NUREG/CR－0200, Volume 2, Section F10, ORNL/NUREG/CSD－2/V2 (1982).

(2) 幾島　毅，中里　力；HEATING 5－JR：有限差分法による非線型熱伝導計算プログラム，JAERI－M (1983).

(3) L. Edwords ; TRUMP : A Computer Program for Transient and Steady－State Temperature Distributions in Multidimensional Systems, LLL W－7405－Eng－45 (1972).

(4) K. J. Bathe et al. : SAP Ⅳ, A Structual Analysis Program for Static and Dynamic Response of Linear Systems, EERC 73－11 (1973).

(5) 差分法による非線型熱伝導計算プログラムHEATING 6のベクトル化，原子力データセンタ（private communication）(1985).

(6) 核融合研究開発の現状，P－43，日本原子力研究所 (1984).

(7) 石黒美佐子，難波克光；差分法のベクトル計算，Vol. 24，No.1 (1983).