Vol. 26 No. 1 情報処理学会論文誌 Jan. 1985

数値シミュレーション用プログラミング言語 DEQSOL†

梅谷征雄†† 辻 みちる†† 岩沢京子††

DEQSOL (Differential EQuation SOlver Language) は数値シミュレーション用に設計された問題向きプログラミング言語であり，偏微分方程式の解法を数値アルゴリズムのレベルで記述することを目的とする．言語設計の狙いは二つあり，第1はこの分野におけるプログラム生産性を飛躍的に向上させることであり，第2は数値シミュレーションに通常用いられるベクトル計算機や並列計算機に向けたプログラムに対しシステム側の最適化の余地を保持することである．このために言語の設計にあわせて，DEQSOL で書かれたプログラムをベクトル/並列計算機向き FORTRAN コードに自動変換するトランスレータを検討・開発した．言語の設計にあたっては，領域形状，メッシュ分割，微分演算子，ベクトル，境界条件など偏微分方程式系の数値シミュレーションに固有の概念を導入して，差分法に基づく計算手順を簡潔・柔軟に記述できるように工夫した結果，典型的な3次元流体シミュレーションプログラムを FORTRAN の10分の1以下の行数で作成する実績を得た．また差分法や線形解法に内在する並列性を生かした FORTRAN コードの生成に努めた結果，M 200 H IAP にて90%以下のベクトル化を達成することができた．DEQSOL は解法記述言語である点で PDEL[4] や ELLPACK[5] などのソルバとは異なり，むしろ PDELAN[7] のアプローチを拡張・発展させたものである．

1. まえがき

スーパコンピュータの出現[1]に象徴されるように，半導体技術と論理方式技術の進展をうけた最近の電子計算機の性能向上には著しいものがあり，その高速性を生かして，物理現象の数値シミュレーションが普及しつつある．しかし，ハードウェア技術の進展とは対照的に数値シミュレーションを取り巻くソフトウェア環境は旧態依然たるものがあり，それが今後の進展を阻む主要な隘路となるおそれが出てきた．とくにシミュレーション用ソフトウェアの作成については応用別パッケージプログラムと個別ユーザプログラムを問わず，それが次のような特徴を有することから問題が深刻である．

第1に，シミュレーションソフトウェアにおける数学的モデルと計算スキームは，計算値と理論値，実験値あるいは観測値の比較を通して改良・更新してゆくものであり，モデルもしだいに詳細化されるのがつねであることから，プログラムの修正と拡張が不可欠である．第2に，数値シミュレーションにおける微分式の差分展開，境界条件式の取扱い，線形計算における数値アルゴリズムなどは混み入った複雑なプログラミングを必要とする．第3に，とくに最近の特徴として，スーパコンピュータ向け並列化プログラミング技法の必要性が挙げられる．

† Numerical Simulation Language DEQSOL by YUKIO UMETANI, MICHIRU TSUJI and KYOUKO IWASAWA (Hitachi Central Research Laboratory).

†† (株)日立製作所中央研究所

このような特徴を有する数値シミュレーションソフトウェアの作成に関して，従来から用いられている FORTRAN 言語は必ずしも適しているとはいえない．モデル拡張のためのプログラムの可読性 (readability) の向上はもとより，差分展開の効率化や並列化プログラミングのためにもより高水準のプログラミング言語が適しているものと思われる．この種のプログラミング言語の備えるべき要件としては，(1)アルゴリズムと計算スキームならびに領域形状の簡潔な記述能力，(2)各種の解法への柔軟な適応性，(3)並列化アルゴリズムの自然な記述能力などが挙げられる．

さて，数値シミュレーション用の在来記述言語[2]としては，1960年代の SALEM[3]，70年代の PDEL[4]，最近80年代の ELLPACK[5]，TWODFPEP[6] などが挙げられるが，いずれも線形計算ライブラリを使用するための制御言語的な性格を有し，記述の柔軟性に欠ける欠点が指摘される．ところで，これらとは異なるアプローチをとる在来言語として NCAR の PDELAN[7] がある．これは通常の FORTRAN 言語に差分演算用の演算子を追加したもので柔軟性に富む利点を評価できるものの機能面では形状や境界条件記述機能をもたぬ点で不十分である．筆者らがここで提案する高水準言語 DEQSOL (Differential EQuation SOlver Language) は PDELAN のアプローチをさらに徹底して推し進めたものであり，偏微分方程式を対象とする種々の解法の記述性を保ちながら高水準化を進めさらに簡潔性をも狙う．また，定型的ではあるが解法の混み入る線型計算アルゴリズムに関しては PDEL 流

のライブラリ取込みも行うこととする．さらに従来は考慮されることのなかった，スーパコンピュータ向け並列化アルゴリズムの自然な記述を目的とする．

本論文では，まず第2章にて DEQSOL 言語の現状における機能と構成，記述例を述べ，続く第3章にて言語仕様の評価を行う．

なお当言語には FORTRAN 言語へのトランスレータを中心とする処理系が付随しており，翻訳の結果として得られる FORTRAN プログラムはベクトルプロセッサに適合するように生成されるが，それらに関連する処理系の翻訳方式や生成プログラムの性能については後報にて述べることとし本論文では結果の一部を紹介するにとどめる．

2. DEQSOL 言語の仕様

2.1 言語の設計方針

（1） 数理モデルと計算スキームレベルの記述

数値シミュレーションの過程は，**図1**に示すように問題の定義にはじまり，数理モデルの決定，計算スキームとアルゴリズムの決定，FORTRAN などによるプログラミング，計算機を用いたランと数理モデル，計算スキーム，プログラムへのフィードバックの経過をとるが，DEQSOL の狙いを計算スキームレベルの記述に置き，それ以降の過程を DEQSOL 処理系により自動化することとする．

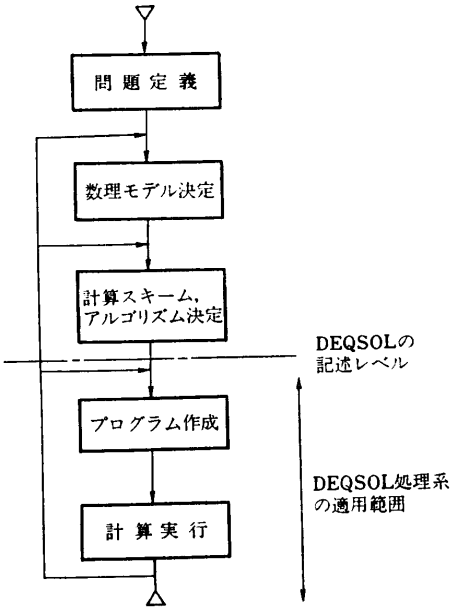


図1 数値シミュレーション過程と DEQSOL の役割
Fig. 1 Numerical simulation process and the role of DEQSOL system.

（2） 記法の簡潔性とモデル・解法の分離記述

図1におけるフィードバックサイクルを短縮するために，可読性向上・記述行数の削減を狙って数値解析書で常用される簡潔な記法を用いることとする．また，数理モデルと計算スキーム・アルゴリズムの記述を分離し，計算スキームのみの変更に即応できるようにする．

（3） 分布定数系（連続系）シミュレーションへの適用

物理現象の数値シミュレーションは，モデル化のタイプにより，時空分布量を基本とする分布定数系モデル，電子回路や構造体などを対象とする離散定数系モデル，質点の運動方程式を基本とする粒子系モデルに分類されるが，当初の適用範囲を基本的でかつ演算量の多い分布定数系モデルに置き，漸次拡張を図ることとした．さらに離散化の方式に関しては差分法から始めることとする．またエレクトロニクス系の応用分野を対象として仕様の検討を行った．

2.2 記 述 例

簡単な問題の記述例を紹介し言語の概要を示す．

図2は簡単な二次元の熱拡散問題と DEQSOL によるその記述例を示す．初期温度100℃が与えられている矩形領域の時間的な温度変化を求める問題である．境界条件として上壁Uと下壁Dにて 200℃，左右の壁 L, RI にて断熱条件が与えられている．

DEQSOL では，キーワードに始まりセミコロンに終わる文を並べて数理モデルと計算スキームを記述する．文の種類とおのおのの記述方法を**表1**にまとめて示す．文はモデルの記述に関するもの（項番1～14, 25）とスキームの記述に関するもの（項番15～24）に分類される．

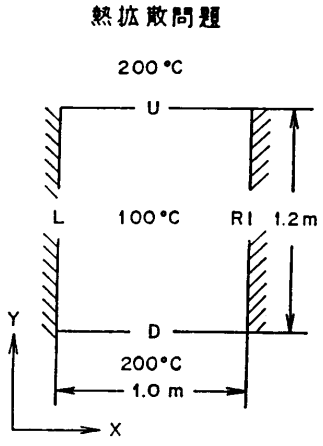
記述例におけるおのおのの文の意味は次のとおりである．先頭の VAR 文は求めるべき温度分布変数 TT を宣言する．続く DOM 文，TDOM 文はシミュレーションを行うべき空間範囲と時間範囲を指定する．MESH 文は連続量の離散化のためのメッシュ系を定義する．CONST 文は既知量を宣言する．このケースでは拡散定数 A を定義している．REGION 文は矩形の空間部分領域を宣言する．＊は対応する座標軸方向の全域指定である．したがって，たとえばLは左側の壁を宣言している．続く EQU 文は系を支配する拡散方程式を宣言している．ここで dt は時間に関する1階微分を，lapl はラプラス演算子を意味する．INIT 文，BOUND 文は変数に対する初期条件と境

表 1 DEQSL における文の種類と指定方法
Table 1 List of DEQSL statements.

項番	指定項目	指定方法
1	プログラム名称 サブルーチン名称 引数名称	・メインプログラムにて名称指定の場合. PROG 〈プログラム名称〉; ・サブルーチン化を行い，名称と引数を指定する場合. PROC 〈プログラム名〉; または PROC 〈プログラム名〉(〈引数名1〉,…,〈引数名N〉);
2	問題定義域(空間域)	DOM $\left\{\begin{matrix}X\\Y\\Z\end{matrix}\right\}$=[〈下限座標値〉:〈上限座標値〉],…, $\left\{\begin{matrix}X\\Y\\Z\end{matrix}\right\}$=[〈下限座標値〉:〈上限座標値〉];
3	問題定義域(時間域)	TDOM T=[〈時間下限値〉:〈時間上限値〉];
4	領域	REGION 〈領域名〉=(〈座標範囲〉,…,〈座標範囲〉),… 〈領域名〉=(〈座標範囲〉,…,〈座標範囲〉); 〈座標範囲〉は ＊(全域)，〈座標値〉， (〈下限座標値〉:〈上限座標値〉)，[〈下限座標値〉:〈上限座標値〉)， (〈下限座標値〉:〈上限座標値〉]，[〈下限座標値〉:〈上限座標値〉] のいずれか.
5	変数	VAR 〈変数名〉,…,〈変数名〉;
6	ベクトル	VECT 〈ベクトル名〉=(〈変数名〉,…,〈変数名〉),… 〈ベクトル名〉=(〈変数名〉,…,〈変数名〉);
7	定数	CONST {〈定数名〉=〈式〉[AT 〈領域名〉+[〈領域名〉+…]], 〈定数名〉=FILE 〈データセット番号〉,}…;
8	定数ベクトル	CVECT 〈定数ベクトル名〉=(〈定数名〉,…,〈定数名〉),…;
9	外部関数	EFUNC 〈関数名〉,…,〈関数名〉;
10	外部サブルーチン	EPROC 〈サブルーチン名〉,…,〈サブルーチン名〉;
11	カウンタ	CTR 〈カウンタ名〉,…,〈カウンタ名〉;
12	等式	EQU 〈式〉=〈式〉,…,〈式〉=〈式〉;
13	境界条件	BOUND {{〈変数名〉/〈ベクトル名〉}=〈式〉[AT 〈領域名〉[+…]] {DX/DY/DZ}({〈変数名〉/〈ベクトル名〉})=〈式〉[AT 〈領域名〉[+…]] {〈変数名〉/〈ベクトル名〉}:={〈変数名〉/〈ベクトル名〉}},…;
14	初期条件	INIT {{〈変数名〉/〈ベクトル名〉}=〈式〉[AT 〈領域名〉[+…]] {〈変数名〉/〈ベクトル名〉}:={〈変数名〉/〈ベクトル名〉}},…;

項番	指定項目	指定方法
15	メッシュ	MESH $\left\{\begin{matrix}X\\Y\\Z\\T\end{matrix}\right\}$=[〈下限座標〉:〈上限座標〉:〈メッシュ間隔〉],…,
16	差分定義	DIFFS 〈微分演算子名〉({@ / [@X, @Y, @Z]}) =〈定義式〉[AT 〈領域名〉[+…]],…; ＊定義式は微分演算子の引用を含んではならない.
17	解法	SCHEME, 〈解法手続き〉 END SCHEME; 〈解法手続き〉は，以下に述べる〈代入文〉，〈ソルブ文〉，〈コール文〉，〈プリント文〉，〈ディスプレイ文〉，〈ライト文〉，〈繰り返しブロック〉の列からなる.
18	代入文	{変数名 / ベクトル名}[(〈添字〉)]=〈離散式〉[WHERE 〈離散条件式〉] [AT 〈領域名〉[+〈領域名〉+…]]; あるいは {〈変数名〉 / 〈ベクトル名〉}=FILE 〈データセット番号〉; 〈離散条件式〉は，〈離散式〉を比較演算子 (LT, GT, LE, GE) と論理比較演算子 (AND, OR) で結んだものである. 〈離散式〉とは，メッシュ位置を示す添字付き変数名ベクトル名を含む式である.
19	ソルブ文 (陰解法指定)	SOLVE 〈変数名〉[(〈添字〉)] OF 〈離散式〉=〈散離式〉 BY 〈解法名〉[WITH (〈解法パラメータ〉[,…])]; 許容する解法は次のものである. JACOBI, SOR, ILUCG, GAUSS, ILUCR, CHOLES
20	コール文 (外部サブルーチンコール)	CALL 〈サブルーチン名〉(〈引数〉,…,〈引数〉); 〈引数〉は添字付き変数名，ベクトル名である.
21	プリント文 (数値出力)	PRINT 〈名称〉[,…,〈名称〉][AT 〈領域名〉] [EVERY 〈整定数〉 TIMES],…; プリント対象となる名称は〈変数名〉，〈ベクトル名〉，〈定数名〉，〈定数ベクトル名〉，〈カウンタ名〉のいずれかである.
22	ディスプレイ文 (図形出力)	DISP 〈名称〉[,…,〈名称〉][AT 〈領域名〉] [EVERY 〈整定数〉 TIMES],…;
23	ライト文 (ファイル出力)	WRITE 〈名称〉 TO FILE 〈ファイル番号〉 [EVERY 〈整定数〉 TIMES],…;
24	繰り返しブロック	ITER 〈カウンタ名〉 UNTIL 〈終了条件式〉; 〈繰り返し対象の解法手続き〉 END ITER; ・カウンタ値はゼロに初期化され，1回ずつ更新される. ・繰り返し対象部で再び繰り返しブロックを呼ぶことができる（2回まで）.
25	プログラムの末尾	END;

・下線を引いた予約語 (KEY WORD) はすべて小文字であってもかまわない.




```
VAR      TT;
DOM      x=[0:1],
         y=[0:1.2] ;
TDOM     t=[0:1];
MESH     x=[0.0:1.0:0.2],
         y=[0.0:1.2:0.2],
         t=[0.0:1.0:0.02] ;
CONST    A=0.62 ;
REGION   R=[*,*],
         L=[0,*],
         RI=[1,*],
         D=[*,0],
         U=[*,1.2] ;
EQU        dt[TT]=A*[lapl[TT]] ;
INIT       TT=100 AT R ;
BOUND      TT=200 AT D,
           TT=200 AT U,
           dX[TT]=0 AT L+RI;
SCHEME ;
   ITER      NT UNTIL    NT GT 4 ;
             TT(+1)=TT+DLT*A*lapl[TT];
   PRINT   TT    AT R;
   DISP    TT    AT R;
   END ITER ;
END SCHEME ;
END ;
```

図 2 二次元熱拡散問題記述例
Fig. 2 Programming sample of 2-dimensional thermal diffusion problem.

界条件を指定する．最後の SCHEME 文から END SCHEME 文までの範囲で計算スキームを記述する．このケースでは時間メッシュのカウンタ NT を変えながら ITER 文から END ITER 文までを繰り返すことを意味する．TT〈+1〉に始まる代入文は拡散方程式の陽解法表現であり，等号右辺の式の値をメッシュ点ごとの離散値を用いて計算し左辺の変数に代入することを意味する．TT〈+1〉は次時間メッシュにおける TT の空間分布を，DLT は単位時間メッシュ幅 0.02 を意味する．続く PRINT 文，DISP 文は結果表示用の文である．最後の END 文は記述の終りを示す．

2.3 言語の構成と機能

（1） 言語の構成

DEQSOL 言語は FORTRAN などと同様に字母の列により構成される．字母は英字，数字，区切り記号より成る．ギリシャ文字や微分演算記号はこれらを用いて代記する．字母列の最小単位は語 (Word) であり，これには予約語 (Reserved Word)，名称 (Name)，数値 (Number)，演算子 (Operator)，区切り子 (Delimitor) の種類がある．名称，数値，演算子の組合せによって式が定義され，これは FORTRAN と同一の構文を有する．数値の表現形式も FORTRAN と同一である．

予約語，式，区切り子の列により表 1 に示すような種々の文が定義される．代入文を除いて，文は必ず予約語から始まるが，予約語は大文字，小文字の 2 通りを用意している．

（2） 言語の機能

名称や式に関する基本機能と文に関する機能に分類して述べる．文に関する機能は，数理モデルを定義するための問題定義機能，計算スキームを記述するための解法記述機能，計算結果の表示機能，他プログラムとの連係をとるためのシステム化機能に分類される．

（a） 基本機能

（i） 名称と演算単位：名称は算術演算の対象となる演算単位を代表するほか，領域名，微分演算子名，関数名，サブルーチン名，解法名，座標値名，メッシュインデクス名，カウンタ名などを示す．演算単位は，未知量と既知量，スカラ量とベクトル量の組合せにより，変数，定数，ベクトル，定数ベクトルの 4 通りに分類される．いずれも対象領域内の分布を意味し，解法記述部ではメッシュ点における離散値の集合を意味する．名称は一般に使用に先立って宣言されねばならないが，いくつかの名称は予約名称として特別な意味をもつように定められており，宣言なしに使用できる．それらは座標値，メッシュ間隔，メッシュインデクスなどである．予約名称の一覧を表 2 に示す．

（ii） 演算子：算術演算子と微分演算子に大別される．算術演算子は，四則・ベキ乗に内積を加えた 6 種

表 2 予約名称一覧
Table 2 List of reserved names.

WORD	意味	種別	WORD	意味	種別
x/X	X座標変数	COORD	dlt/DLT	時間メッシュ間隔	COORD
y/Y	Y座標変数	COORD	i/I	X座標インデクス	IDX
z/Z	Z座標変数	COORD	j/J	Y座標インデクス	IDX
t/T	T座標変数	COORD	k/K	Z座標インデクス	IDX
dlx/DLX	X軸メッシュ間隔	COORD	@, @X, @Y, @Z (@x, @y, @z)	標準仮引数	VAR
dly/DLY	Y軸メッシュ間隔	COORD			
dlz/DLZ	Z軸メッシュ間隔	COORD			

表 3 標準微分演算子一覧
Table 3 List of standard differential operators.

微分演算子	標準離散形式(内部)	標準離散形式(境界)	ARG-TYPE	VAL-TYPE
dx/DX	$dx(u)=(u\langle i+1\rangle-u\langle i-1\rangle)/(2*\mathrm{dlx})$	$dx(u)=(4*u\langle i+1\rangle-u\langle i+2\rangle-3*u\langle i\rangle)/(2*\mathrm{dlx})$ $dx(u)=(-4*u\langle i-1\rangle+u\langle i-2\rangle+3*u\langle i\rangle)/(2*\mathrm{dlx})$	$s(v)$	$s(v)$
dy/DY	$dy(u)=(u\langle j+1\rangle-u\langle j-1\rangle)/(2*\mathrm{dly})$	$dy(u)=(4*u\langle j+1\rangle-u\langle j+2\rangle-3*u\langle j\rangle)/(2*\mathrm{dly})$ $dy(u)=(-4*u\langle j-1\rangle+u\langle j-2\rangle+3*u\langle j\rangle)/(2*\mathrm{dly})$	$s(v)$	$s(v)$
dz/DZ	$dz(u)=(u\langle k+1\rangle-u\langle k-1\rangle)/(2*\mathrm{dlz})$	$dz(u)=(4*u\langle k+1\rangle-u\langle k+2\rangle-3*u\langle k\rangle)/(2*\mathrm{dlz})$ $dz(u)=(-4*u\langle k-1\rangle+u\langle k-2\rangle+3*u\langle k\rangle)/(2*\mathrm{dlz})$	$s(v)$	$s(v)$
dxx/DXX	$dxx(u)=(u\langle i+1\rangle-2*u\langle i\rangle+u\langle i-1\rangle)/\mathrm{dlx}**2$	$dxx(u)=(u\langle i+2\rangle-2*u\langle i+1\rangle+u(i))/\mathrm{dlx}**2$ $dxx(u)=(u\langle i-2\rangle-2*u\langle i-1\rangle+u(i))/\mathrm{dlx}**2$	$s(v)$	$s(v)$
dyy/DYY	$dyy(u)=(u\langle j+1\rangle-2*u\langle j\rangle+u\langle j-1\rangle)/\mathrm{dly}**2$	$dyy(u)=u(\langle j+2\rangle-2*u\langle j+1\rangle+u\langle j\rangle)/\mathrm{dly}**2$ $dyy(u)=u(\langle j-2\rangle-2*u\langle j-1\rangle+u\langle j\rangle)/\mathrm{dly}**2$	$s(v)$	$s(v)$
dzz/DZZ	$dzz(u)=(u\langle k+1\rangle-2*u\langle k\rangle+u\langle k-1\rangle)/\mathrm{dlz}**2$	$dzz(u)=(u\langle k+2\rangle-2*u\langle k+1\rangle+u\langle k\rangle)/\mathrm{dlz}**2$ $dzz(u)=(u\langle k-2\rangle-2*u\langle k-1\rangle+u\langle k\rangle)/\mathrm{dlz}**2$	$s(v)$	$s(v)$
grad/GRAD	grad $(u)=(dx(u), dy(u), dz(u))$		$s(v)$	$v(t)$
lapl/LAPL	lapl $(u)=dxx(u)+dyy(u)+dzz(u)$		$s(v)$	$s(v)$
div/DIV	div $(u, v, w)=dx(u)+dy(v)+dz(w)$		v	s
rot/ROT	rot $(u, v, w)=(dy(w)-dz(v), dz(u)-dx(w), dx(v)-dy(u))$		v	v
dt/DT	$dt(u)=(u\langle+1\rangle-u)/\mathrm{dlt}$		s	s

類，微分演算子は表3に示す1階，2階の空間微分に，ラプラシアン，grad, div, rot, 時間微分を加えた11種類を標準にて用意するほか，微分式と差分式の定義機能を置く．四則・ベキ乗はFORTRANと同じ表記法をとり，内積は..で表す．微分演算子の引用は形式上関数コール形をとる．演算子はスカラとベクトルに対して作用する．

(iii) 関数引用：FORTRAN 77の提供する全組込み関数を許容するほか，ユーザ定義関数を引用することができる．引数にベクトルを許容する．

(iv) 式：名称，数値，演算子，関数引用などで構成されFORTRANと同様であるが，ベクトル値を許容する点が拡張されている．

(b) 問題定義機能

DEQSOLにおける問題記述は，領域定義文群(DOM, TDOM, REGION)，変数・定数・ベクトル定義文群(VAR, VECT, CONST, CVECT)，等式文(EQU)，境界条件文(BOUND)，初期条件文(INIT)を用いて行う．

(i) 領域定義文群

これらによりシミュレーションの対象となる空間時間領域，部分空間領域を定義する．空間領域は，1～3次元の矩形領域を指定する．

DOM文は全空間領域を宣言する．各座標軸方向に上限値と下限値を指定する．

(例) DOM $X=[0:1]$, $Y=[0:1]$, $Z=[0:1]$;

TDOM文は全時間領域を宣言する．シミュレーション範囲の下限値と上限値を指定する．

(例) TDOM $T=[0:1]$;

REGION文は空間部分領域に名前を付けて宣言する．各座標軸方向に，全範囲(*)，部分範囲，特定値を指定できるので，それらを組み合せることにより，点，座標軸に平行な直線，平面などの部分領域に名前をつけることができる．これにより，定数定義，境界条件や初期条件定義，代入文などで名前による部分範囲の引用が可能となる．

(例) REGION $IN=(*,*,0)$, $ZC=(0,0,1)$,
$R=([0:1],[0:1],[0:1])$;

(ii) 変数，定数，ベクトル定義文群

これらは，上記の空間・時間領域中に分布をもつ未知量ならびに既知量の名前を定義する．既知量についてはその分布を合わせて定義する．VAR文，VECT文，CONST文，CVECT文の4種類をおく．VAR文はスカラ未知量の名前を宣言する．これらの空間的，時間的な分布を求めることがシミュレーションの目的である．

(例) VAR Alpha, Beta, C;

VECT文は，流れのようなベクトル未知量の名前をその要素の名前とともに宣言する．ベクトルの要素数はDOM文で指定される空間の次元数に等しくなければならない．

(例) VECT $V=(u,v,w)$;

CONST文はスカラ既知量の名前をその定義式とともに宣言する．

(例) CONST $Q=\exp(-X**2-Y**2$
$-(1.0-Z)**2)$,
rho=5, $C=5$;

ここで，X,Y,Z は座標値の x,y,z 成分であり，上記定義式は座標値の関数として分布を定義する．時刻変数Tを用いることにより，時間とともに変動する既知量を定義することもできる．

CVECT文は，ベクトル既知量の名前とその要素名とともに宣言する．各要素はCONST文によりあらかじめ宣言されていなければならない．

(例) CVECT $CV=(u_0,v_0,w_0)$;

(iii) 等式文

上記時空域において，変数名，定数名で示される物理量の間に成り立つ関係を規定する．すなわち物理法則を記述する．形式は，微分演算子を含む数式を等号で結んだものである．ベクトル式を許容する．

(例) EQU rho$*C*$(dt$(C)+V.$.grad(C))
$=$kk$*$lapl$(C)+Q$;

(iv) 境界条件文

未知量（スカラ，ベクトル）に対する境界条件をこの文を用いて指定する．境界条件としては，境界における値を指定する第1種境界条件と，法線ベクトル方向の微分値を指定する第2種境界条件の両者を許容する．原則として，すべての未知量のすべての境界に対していずれかの条件が指定されなければならないが，流れの下流境界のように指定の不可能な部分に対してはFREEをおくことによりそれを不要としている．

(例) BOUND $C=0$ at $IN+X_1+Y$,
$dz(C)=0$ at O,
$C=$FREE at Z_1;

(v) 初期条件文

非定常問題に対しては原則として全未知量について初期値が必要であるので，この文を用いて指定する．

(例) INIT $C=0$ at R;

(c) 解法記述機能

解法記述は，メッシュ文，離散値/離散式，差分定義文，解法ブロックを用いて行う．

(i) メッシュ文

メッシュ文は差分定義計算のためのメッシュ（格子点)を定義する．指定形式は DOM 文，TDOM 文に類似しているが，メッシュ間隔指定が追加されている．

(例) MESH $X=[0:1:0.1]$,
$Y=[0:1:0.1]$,
$Z=[0:1:0.1]$,
$T=[0:5:0.001]$;

(ii) 離散値/離散式

本来，空間域と時間域内で連続的な分布を示す未知量，既知量ならびにそれらを用いた式のメッシュ点における値，式値を離散値/離散式と呼ぶ．差分に基づく解法は微分演算子も含めてすべて離散値間の操作で表せる．離散値は原則的に連続値と表記上区別をしないが，とくに必要な場合には下記のように名前に添字を付加することにより区別する．

(例) ○ $C\langle ; I, J, K\rangle$, C(同じもの)
○ $V\langle +1 ; I+1, J\rangle$,
$V\langle +1 ; I+1\rangle$(同じもの)

+1 は後に述べる繰り返しブロック内の1反復ステップ後の値を，$I+1$ は X 軸方向に1メッシュ分先の値を意味する．変分のないメッシュの添字は省略を許している．

(iii) 差分定義文

差分法では微分演算は差分式に置き換えられるので，微分演算の差分定義をこの文を用いて行う．差分式は上記の離散式の記法により定義する．一般に領域の内部と境界で定義式が異なるので領域指定付の記述を行う．

(例) DIFFS $dx(ⓐ)=(ⓐ\langle ; I+1\rangle$
$-ⓐ\langle ; I-1\rangle)/(2*\mathrm{DLX})$
at $I_n+Y_0+Y_1+Z_0+Z_1$,
$dx(ⓐ)=(ⓐ\langle ; I+1\rangle$
$-ⓐ\langle ; I\rangle)/\mathrm{DLX}$ at X_0,
$dx(ⓐ)=(ⓐ\langle ; I\rangle$
$-ⓐ\langle ; I-1\rangle)/\mathrm{DLX}$
at X_1;

ただし表3に示す標準差分式を用いる場合には定義は不要である．

(iv) 解法ブロック

具体的な計算手順の記述は解法ブロックにて行う．

解法ブロックは，SCHEME と END SCHEME に囲まれたプログラム部分であり，内部に種々の実行文を書き並べることにより計算手順の指定を行う．実行文中，解法記述に関係するものは，代入文，ソルブ文，繰り返しブロックの3種類である．

(イ) 代入文：これは等号左辺の未知量の離散値を右辺の離散式の値で置き換えることを意味する．FORTRAN の代入文に類似しているが，空間メッシュ点分の並列性を内包すること，ベクトル式を許すこと，代入にあたって左辺未知量の境界条件を考慮すること（すなわち第1種境界に対しては代入を行わない）が FORTRAN とは異なる．さらに at 指定による領域限定と where 指定による代入条件の付加を行うことができる．右辺に離散式を置く代りにファイル名を指定することによりファイルデータを代入することもできる．

(例) $C\langle +1\rangle=C+\mathrm{DLT}*((\mathrm{kk}*\mathrm{lapl}(C)+Q)/$
$(\mathrm{rho}*\mathrm{cc})-\mathrm{VV}..\mathrm{grad}(C)$;
$\mathrm{DXV}=(V\langle ; I\rangle - V\langle ; I-1\rangle)/\mathrm{DLX}$
where $(V\langle ; I\rangle \mathrm{lt}\ V\langle ; I-1\rangle)$ at Inn;
$C=\mathrm{FILE}\,01$;

(ロ) ソルブ文：この文は特定の未知量の離散値をある等式が成り立つように決定すること，すなわち方程式を解くことを指示する．未知量の決定にあたっては境界条件をあわせて考慮する．また線形計算の解法の名称とパラメータを指定する．この文は，定常問題や悪条件の非定常問題に対して陰解法を用いる際に必要となる．方程式は対象とする未知量の一次式でなければならない．

(例) SOLVE PSY of
$\mathrm{SGM}*\mathrm{lapl}(\mathrm{PSY})$
$+\mathrm{grad}(\mathrm{SGM})..\mathrm{grad}(\mathrm{PSC})=0$
by 'GAUSS' with EPS(0.0001);

線形計算の解法としては，反復系のものとしてヤコビ法，加速緩和法 (SOR 法)，不完全コレスキ分解＋共役傾斜法 (ICCG 法)，不完全 LU 分解＋共役残差法 (ILUCR 法) の4種類を，直接法の系統としてガウス消去法とコレスキ分解法を用意する．

(ハ) 繰り返しブロック：ITER と END ITER に囲まれたプログラム部分を繰り返しブロックと呼び，この部分をカウンタ値をゼロから一つずつ増やしながら，終了条件が得られるまで繰り返すことを意味する．繰り返しブロックには実行文を任意個数，任意の順序で入れることができ，繰り返しブロック自身も

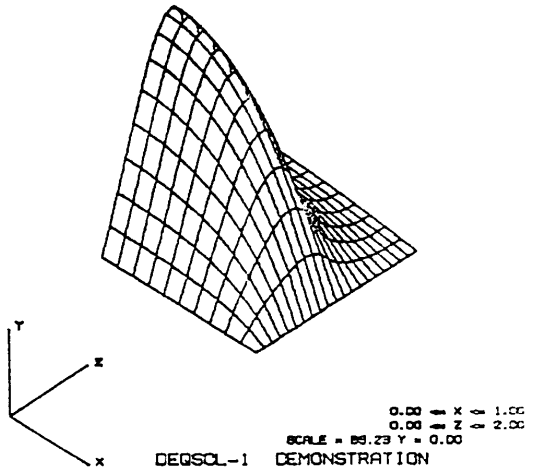


図 3 図形出力例（立体図）
Fig. 3 Output sample (mesh graph).

二重までのネストが許される．各繰り返しの最後にて，添字〈+1〉のついた離散値（ディレイ値）は添字なしの離散変数に無条件に代入され旧値が更新される．繰り返しブロックはおもに非線形計算の線形化収束反復と時間ステップの反復の記述に用いる．カウンタ NT は時間反復に用いられ，このブロックの最後で時刻変数Tの更新と，時刻変数を含む既知量，第1種境界値の更新があわせて行われる．

(例) ITER NT until NT gt 200;
$C\langle+1\rangle = C + \mathrm{DLT} * ((\mathrm{kk} * \mathrm{lapl}(C) + Q) /$
$(\mathrm{rho} * \mathrm{cc}) - \mathrm{VV}..\mathrm{grad}(C))$;
END ITER;

(d) 結果表示機能

表示用にプリント文とディスプレイ文を用意する．いずれも実行文であり解法ブロック内で用いることができる．プリント文は離散値を行列形式にて表示するものであり，at 指定による領域の限定ならびに every times 指定による繰り返し間引きを行うことができる．この機能はデータの正確な把握のために必須である．

(例) print T, C at Y_0 every 50 times;

ディスプレイ文は離散値群を図形端末に特定の形式にて表示する．一般に数値シミュレーションの出力データ量は膨大であり，図形化/画像化表示は結果の直感的理解のための有効な手段を提供する．ディスプレイ文による表示の形式は対象がスカラかベクトルかにより異なり，スカラ量に対しては**図3**に示すような縦軸に値を置く立体図を，ベクトルに対しては**図4**のような矢印図を表示する．ディスプレイ文を繰り返しブロック中で発行すれば動画表示も可能である．

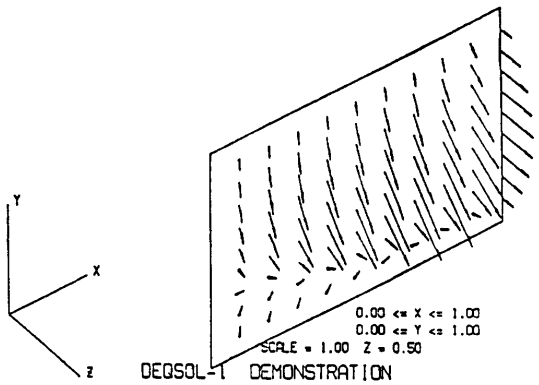


図 4 図形出力例（矢印図）
Fig. 4 Output sample (arrow graph).

(e) システム化機能

DEQSOL 言語を既存のプログラミング言語やオペレーティングシステムと融和させるためにいくつかの機能をおく．それらはプログラム結合機能とファイルアクセス機能に分けられる．

(i) プログラム結合機能

(イ) プログラムヘッダ：DEQSOL の先頭にオプショナルにこの文を置くことによるメインプログラムとするかサブルーチンとするかを指定する．

(例) PROG MAIN;
PROC SUBI(A, B);

PROC を使用することにより全体の計算コードの一部分を DEQSOL を用いて作成することが可能となる．

(ロ) 外部サブルーチン宣言文 (EPROC) とコール文：外部サブルーチンとの結合に用いる．

(ハ) 外部関数宣言文 (EFUNC)：同様にこれを用いて宣言することにより式中で外部関数の引用を行うことができる．

(ii) ファイルアクセス機能

外部プログラムとのデータ受け渡しのためにこの機能をおく．

(イ) ファイルデータの引用：定数定義文，代入文にて等号の右辺に FILE〈番号〉を指定することにより可能とする．番号とファイルの対応は実行時にジョブ制御文によってとる．

(ロ) ファイルへの格納：ライト文を解法ブロック中で発行することにより可能とする．

2.4 言語設計の狙いと言語構成

本言語にて狙った，(1)アルゴリズムと計算スキー

```
方 程 式
dx(ET)**2+dy(ET)**2=R**(-2)
    DEQSOL (ニュートン・ラフソン反復)
iter N until DET(: 25, 50) lt 0.0001;
solve DET of
  2*(dx(ET)*dx(DET)+dy(ET)*dy(DET))
    =R**(-2)-dx(ET)**2-dy(ET)**2
by 'GAUSS';
  ET(+1)=ET+DET;
end iter;
```

図 5 非線形収束反復計算の記述例
Fig. 5 Programming sample describing convergence scheme of nonlinear equation.

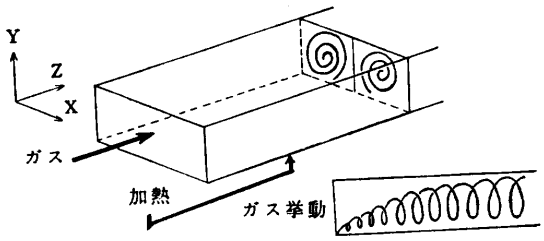
ムの簡潔な記述，(2)各種解法への柔軟な適応，(3)並列化アルゴリズムの自然な記述の3点につき，言語構成との関連を論じる．

(1) アルゴリズムと計算スキームの簡潔な記述

これは，分布量を基本とし境界条件や初期条件あるいはメッシュ分割などの属性を有する“変数”，“ベクトル”などの操作単位を導入し，また微分演算子などのそれらに対する高位演算子と FORTRAN と同一の簡潔な式構文を導入することにより実現した．離散量における添字のオプショナルな指定機能も簡潔度を増すうえで効果的である．

たとえば数値解析書における一般の記法である

$$\frac{\partial C}{\partial t}=k\,\Delta C-\boldsymbol{V}\cdot\mathrm{grad}\,C+Q$$




```
V = ( u , v , w )     流速ベクトル
Q = r o t   V         渦度ベクトル
T                     温度
C                     濃度
G r , P r , S c       係数

dQ/dt=-V・gradQ+Q・gradV+laplQ
      +Gr*rot(0, T, 0)      ・・・(運動方程式)

dT/dt=1/Pr*laplT-V・gradT     ・・・(エネルギー保存式)

dC/dt=1/Sc*laplC-V・gladC     ・・・(質量保存式)
```

図 6 CVD シミュレーション
Fig. 6 CVD simulation.

と DEQSOL における等価な記法である

```
dt(C)=k*lapl(C)-V..grad(C)+Q
```

は簡潔度においてほとんど差がない．

(2) 各種解法への柔軟な適応

DEQSOL においては，解法ルーチンの名称を指定する PDEL[4] や ELLPACK[5] とは異なり，解法ブロックにて実行文を任意に組み合わせて解法を記述する方式をとることにより，各種の問題への柔軟な適応を図った．データ操作のためには代入文とソルブ文の2種類を置き，陽解法，陰解法，準陰解法の種々のスキームに対応できるように配慮した．計算制御のためには繰り返しブロックを置き，おもに時間進行に関する反復と非線形収束反復の目的で使用できるように図った．**図5**にニュートン・ラフソンスキームによる非線形収束反復計算の記述例を示す．

解法ブロック以外では，問題に応じて各種の差分スキームが定義できるように差分定義文を用意した．

(3) 並列化アルゴリズムの自然な記述

ベクトルプロセッサやパラレルプロセッサに向けた並列化アルゴリズムを機種に依存することなく表現するために，代入文とソルブ文を用いて並列演算の自然な記述ができるように工夫した．すなわち，代入文においては右辺の式値の各メッシュ点における並列な計算と左辺への代入を意味しており，またソルブ文においては，連立一次方程式に帰着させるための係数行列要素と定数ベクトル要素の並列計算と，それに引き続くアルゴリズムに応じた並列性を有する線形求解計算を意味する．このような並列性のアーキテクチャへのあてはめは処理系の役割とした．

3. 言語仕様の評価

本言語で狙った，(1)記述の簡潔性，(2)各種解法への柔軟な適用，(3)並列化アルゴリズムの自然な記述の3点につき評価を行う．

3.1 簡潔化の評価

実際の問題への適用例として半導体プロセスの一例である CVD (Chemical Vapour Deposition) における熱流体の挙動シミュレーションを取り上げ，DEQSOL で記述したプログラムと FORTRAN を用いて記述したプログラムを比較する．

この計算は**図6**に示すように，下から加熱される矩形のダクト内を流れる渦流の挙動をシミュレートするものであり，用いる方程式は同じく図6に示す気流の速度（渦度）ベクトルに関する運動方程式，

温度変化に関するエネルギー保存式，ガス濃度に関する質量保存式の3本である．これらを用いて図7に示す手順にて時間に関する陽解法により定常解に収束させる．各段の陽計算では，改良オイラー法（プレディクタ・コレクタ法）を用いることにより解の安定化に努める．本シミュレーションのDEQSOLを用いたプログラム中，計算の手順を示す解法ブロック部分のコーディングを図8に示す．図7の各計算式がプレディクタ・コレクタ法を用いることにより3本に分かれるだけで，ほとんど両者は同一のレベルである．図8中，枠で囲った渦度式の一部分の陽計算に対応するFORTRANコードを図9に示す．DEQSOLの一つの文の記述に対してFORTRANでは26行を要している．FORTRANにおける付加的な情報は，メッシュ点をスキャンするためのDOループ，微分演算の差分展開，ベクトルのスカラ展開，境界部における特殊処理などであり，DEQSOLにおいてこれは計算式に共通の別指定ないし標準指定となっており一部は構文上の取り決めとして組み込まれていることから，簡素化が実現されている．

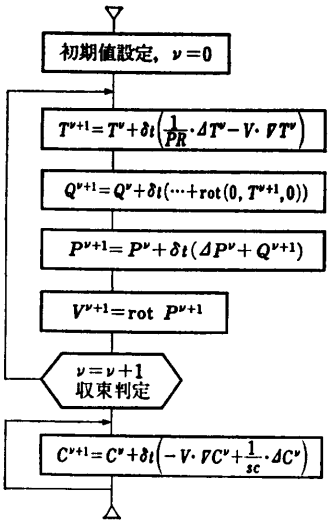


図7 CVDシミュレーション手順
Fig. 7 Process of CVD simulation.

プログラム全体としてのFORTRANとDEQSOLの記述行数の比較を表4に示す．宣言部においてDEQSOLでは，領域宣言や境界条件宣言などの共通指定が付加され，FORTRANよりも行数が多いが，その他の部分では大幅な行数短縮がなされ，全体ではFORTRANに対して約10分の1の記述行数簡約化が達成されている．

3.2 適用性の評価

分布定数系の数値シミュレーション問題数題への

```
SCHEME ;
     PRINT   TT,Q,V,C ;
     ITER NT  UNTIL 100 LT NT AND NT LT 800
                        AND ABS(v<+1;11,6,11>) GT 0.001
                        AND ABS(v<;11,6,11>-v<+1;11,6,11>) LT 0.0005
                        OR NT GT 800 ;
              TN=TT+DLT*(lapl(TT)/Pr-V..grad(TT)) ;
              T1=(TN+TT)/2 ;
              TT<+1>=TT+DLT*((lapl(T1)/Pr)-V..grad(T1)) ;

              QN=Q+DLT*(-V..grad(Q)+Q..grad(V)+lapl(Q)
                              +Gr1*rot(0,TT<+1>,0)) ;
              Q1=(QN+Q)/2 ;
              Q<+1>=Q+DLT*(-V..grad(Q1)+Q1..grad(V)+lapl(Q1)
                              +Gr1*rot(0,TT<+1>,0)) ;
              PN=P+DLT*(Q<+1>+lapl(P)) ;
              P1=(PN+P)/2 ;
              P<+1>=P+DLT*(Q<+1>+lapl(P1)) ;

              V<+1>=rot(P<+1>) ;

              Q<+1>=rot(V<+1>) ;

     END ITER ;

     ITER NC  UNTIL NC GT 100
                    AND ABS(C<;8,8,6>-C<+1;8,8,6>) LT 0.0001 ;
              CN=C+DLT*((lapl(C)/Sc)-V..grad(C)) ;
              C1=(CN+C)/2 ;
              C<+1>=C+DLT*((lapl(C1)/Sc)-V..grad(C1)) ;

     END ITER ;
END SCHEME;

END ;
```

図8 CVDのDEQSOLレコーディング（解法ブロック）
Fig. 8 DEQSOL coding of CVD simulation (scheme block).

指定が付加され，FORTRANよりも行数が多いが，その他の部分では大幅な行数短縮がなされ，全体ではFORTRANに対して約10分の1の記述行数簡約化が達成されている．

```
C     ** VORTICITY
      DO 520 M=2,IZ
      DO 519 K=2,IX
      DO 518 L=2,MY
      IF(K-IX) 473,470,470
  470 QN(K,L,M)=0.
      GOTO 518
  473 QDUX=(U(K+1,L,M)-U(K-1,L,M))*Q(K,L,M)/(2.*DX)
      QDVY=(V(K,L+1,M)-V(K,L-1,M))*Q(K,L,M)/(2.*DY)
      DWX=(W(K+1,L,M)-W(K-1,L,M))/(2.*DX)
      DVZ=(V(K,L,M+1)-V(K,L,M-1))/(2.*DZ)
      DWY=(W(K,L+1,M)-W(K,L-1,M))/(2.*DY)
      DUZ=(U(K,L,M+1)-U(K,L,M-1))/(2.*DZ)
      QDUVW=-QDUX-QDVY-(DWX*DVZ)+(DWY*DUZ)
      UDQX=U(K,L,M)*(Q(K+1,L,M)-Q(K-1,L,M))/(2.*DX)
      VDQY=V(K,L,M)*(Q(K,L+1,M)-Q(K,L-1,M))/(2.*DY)
      WDQZ=W(K,L,M)*(Q(K,L,M+1)-Q(K,L,M-1))/(2.*DZ)
      DQXYZ=-UDQX-VDQY-WDQZ
      GRDTX=GR(JGR)*(T(K+1,L,M)-T(K-1,L,M))/(2.*DX)
      D2QX=(Q(K+1,L,M)-(2.*Q(K,L,M))+Q(K-1,L,M))/(DX**2)
      D2QY=(Q(K,L+1,M)-(2.*Q(K,L,M))+Q(K,L-1,M))/(DY**2)
      D2QZ=(Q(K,L,M+1)-(2.*Q(K,L,M))+Q(K,L,M-1))/(DZ**2)
      A=QDUVW+DQXYZ+GRDTX+(D2QX+D2QY+D2QZ)
      QN(K,L,M)=Q(K,L,M)+(DJ*A)
  518 CONTINUE
  519 CONTINUE
  520 CONTINUE
```

図 9 CVD の FORTRAN コーディング（渦度式の一部）
Fig. 9 FORTRAN coding of CVD simulation (part of vorticity equation).

表 4 FORTRAN, DEQSOL 記述行数比較
—半導体プロセスシミュレーション (CVD)—
Table 4 Lines of code required to describe CVD simulation in FORTRAN and DEQSOL.

内　訳	FORTRAN（行）	DEQSOL（行）
宣言部	13	44
計算部	951	68
出力部	353	3
その他	44	12
合　計	1,361	127
(比)	(10.7)	(1)

表 5 DEQSOL 自動生成プログラムの性能とベクトル化率
Table 5 Processing time and vectorization ratio of the FORTRAN code automatically generated by DEQSOL translator.

プログラム	処理時間 (M 200 H)		ベクトル化率 (%)
	NOIAP(秒)	IAP(秒)	
A	0.45	0.35	91.7
B	0.12	0.09	95.6
C	7.01	1.98	90.8

DEQSOL の適用性をパイロットコーディングにより評価し基本的なフィジビリティを確認したが，機能面で不足している点も明らかになった．それらは不均一メッシュ系のサポート機能，矩形にとどまらぬ一般の幾何学的境界形状の取扱い機能などである．さらに高度な機能としては，波動方程式を取り扱うためのフーリエ展開と固有値解法を用いたモード展開法や，境界要素法などの高能率解法のサポートが求められている．

3.3 並列度の評価

並列化の程度を DEQSOL 処理系が自動生成する FORTRAN プログラムのベクトル化率*で測ることとし，その値を**表 5** に示す．これは M 200 H IAP 用 FORTRAN 77 コンパイラの自動ベクトル化率により評価したものであるが，単純な並列構造を有するプログラムを対象とするのでコンパイラに固有の機能を反映するものではない．NOIAP 指定により並列演算を殺した場合と，IAP 指定により生かした場合の M-200 H による処理時間をあわせて示す．この表から明らかなように，ベクトル化率はいずれの問題でも 90% を越えており高い並列度を示している．この並列性は基本的に DEQSOL で書かれた原プログラムに内包されているものであり DEQSOL 処理系がそれを生かした FORTRAN コードを生成することにより高い並列度を実現したわけである．

なお表 5 において，A, B はそれぞれ 2 次元，3 次元の非定常拡散問題，C は時間項の入らない定常ポテンシャル問題である．

4. 結　び

以上，数値シミュレーションにおけるプログラム生産性を向上し，また並列プロセッサの性能を発揮するための問題向き言語 DEQSOL の仕様を提案し，簡単な評価を行った．三次元の典型的な数値シミュレーション問題に適用した結果，FORTRAN の 10 分の 1 程度の行数で記述が可能であり，また演算の並列度を示すベクトル化率において 90% 以上の評価を得たが，機能面では不十分な点もあり今後の拡充が必要である．

今後は諸機能の拡充によって分布定数系問題への適用性を高める一方，ターゲット言語である FORTRAN との仕様面の融和を図ってさらに柔軟性のある記述を可能としてゆきたい．また分布定数系以外の分野である粒子系や離散定数系の問題に対しても定式化を試み，数値シミュレーションの全分野に適用可能な記述言語に発展させる予定である．

* ベクトル化率：全演算中，ベクトル処理される演算の比率．値が大きいほど並列化の程度が高くベクトルプロセッサによる高速処理が期待できる．

参考文献

1) 成田正弘：スーパーコンピュータによる科学技術計算の行方，日経コンピュータ，1983年4月18日号，pp. 57-73.
2) Machura, M. and Sweet, R. A.: A Survey of Software for Partial Differential Equations, *ACM Trans. Math. Softw.*, Vol. 6, No. 4, pp. 461-488 (1980).
3) Morris, S. M. and Schiesser, W. E.: SALEM —A Programming System for the Simulation of Systems Described by Partial Differential Equations, Proc. Fall Joint Computer Conf., Vol. 33, pp. 353-357 (1968).
4) Cárdenas, A. F. and Karplus, W. J.: PDEL —A Language for Partial Differential Equations, *Comm. ACM*, Vol. 13, No. 3, pp. 184-191 (March 1970).
5) Rice, J. R. and Birkhoff, G. (ed.): ELLPACK Progress and Plans, in *Elliptic Problem Solvers*, pp. 135-162, Academic Press (1981).
6) IMSL Inc.: *TWODEPEP: A Finite Element Program*, 3rd ed. (1982).
7) Gary, J. and Helgason, R.: An Extension of FORTRAN Containing Finite Difference Operators, *Softw. Pract. Exper.*, Vol. 2, pp. 321-336 (1972).

(昭和58年11月7日受付)
(昭和59年9月20日採録)