数値シミュレーション言語 DEQSOL†

佐川暢俊†† 金野千里†† 梅谷征雄††

スーパコンピュータによる大規模な数値シミュレーションの要求が高まっているが，そのソフトウェア環境は必ずしも満足すべきものではない．著者らはプログラム生産性の大幅な向上とスーパコンピュータの有効利用とをねらいとして DEQSOL (Differential EQuation SOlver Language) の開発を進めている．DEQSOL は，問題仕様書レベルの記述から FORTRAN による数値シミュレーションプログラムの自動生成を行う問題向き高水準言語である．入力となる DEQSOL 言語はシミュレーション対象を簡潔に表現するとともに解法アルゴリズムを柔軟に表現できるように設計されている．すなわち，空間メッシュの生成，境界条件の取り込みを自動的に行うのみならず，繰り返し，条件分岐と微分演算子を含む計算式を組み合わせて広範囲の解法スキームを構築することができる．連立偏微分方程式の一括離散化，Body Fit 用プリプロセッサの装備などの拡張を加え，さらなる適用範囲の拡大を図りつつある．一方生成される FORTRAN コードはスーパコンピュータを意識したものであり，ループ長の拡大，コンフリクトの少ないデータ構造の採用などの配慮がなされている．その結果，各種の問題に対して記述効率（記述行数比）で FORTRAN の 10 倍以上，ベクトル化率で 90% 以上のコードの生成を可能としている．

1. 緒　言

スーパコンピュータに代表される高速大容量の計算機の出現は，各種工業製品の開発，設計に大きなインパクトを与えつつある．すなわち，従来の経験と実験を主体とした設計手法に代わり，計算機内で物理現象を再現する数値シミュレーションに基づいた設計パラメータの決定が行われるようになりつつある．数値シミュレーションによれば少ない費用と時間で数多くのケーススタディを行うことができるのみでなく，従来は観測不可能だった現象を追跡することも可能となるからである．

しかし，数値シミュレーションの現状を概観してみると，ハードウェア技術の急速な進歩とは対照的にそのソフトウェア環境は旧態依然としたものがある．シミュレーションを行おうとした場合，多くの利用者は次のような問題に遭遇するであろう．

第1に，物理現象を支配する偏微分方程式の離散化や境界条件の取り込み，連立1次方程式の求解には専門的なアルゴリズムの知識が要求される．第2に，シミュレーションにおいては計算モデルの修正や精密化によってプログラムの改良や更新を行っていくことが多いが，これに必要な工数は一般に膨大となる．第3に，特にスーパコンピュータの有効利用に当たっては問題の並列性を引き出すプログラミング技法が不可欠であるが，このような特殊技法は一般の研究者，技術者には負担が大きい．

これらは，既存の汎用パッケージを使用することができる場合には回避できる問題であるが，利用者のニーズに適合したパッケージが存在しない分野も多く，また利用者専用にアルゴリズムを変更することも容易ではない．

DEQSOL (Differential EQuation SOlver Language) は数値シミュレーションにおいて顕在化してきたこれらの問題点を解決するために開発されたシステムである．その狙いとするところは，主に次の2点である．

（1） 偏微分方程式，領域形状などの物理現象を特徴づける情報を簡潔に記述できるようにし，プログラムの生産性，保守性，および拡張性を大幅に向上させる．

（2） 問題に内包される並列性を利用して，スーパコンピュータ向けに演算並列化率の高いシミュレーションコードを自動生成する．

これまでにも(1)を主目的としたシミュレーション言語はいくつか提案されており，SALEM[1]，PDEL[2] ELLPACK[3] などを挙げることができる．中でも ELLPACK は豊富なライブラリ群を保有する強力なシステムで，現在も対話型や分散型への拡張が継続されている．このシステムはライブラリ群への制御言語的な性格を持ち，その充実によって適用対象の拡大を図る方針をとっている．これに対して DEQSOL は各種の問題，解法を柔軟に記述できる言語であり，各プ

† Numerical Simulation Language DEQSOL by NOBUTOSHI SAGAWA, CHISATO KON'NO and YUKIO UMETANI (Hitachi Central Research Laboratory).

†† (株)日立製作所中央研究所

ログラムに即したコードをその都度生成することを特徴とする．

本稿においては，2章で DEQSOL のシステム構成を示し全体を概観する．3章では DEQSOL 言語の仕様と記述例を示す．4章では DEQSOL プログラムから FORTRAN コードを生成する DEQSOL トランスレータの実現方式について述べ，5章で生成されたコードの評価を行う．

2. システム構成

DEQSOL のシステム構成を**図1**に示す．利用者はまず，対象とする物理現象を特徴づける偏微分方程式，境界条件，領域形状などのグローバルなレベルの情報を記述した DEQSOL プログラムを作成する．この DEQSOL プログラムは DEQSOL トランスレータによって FORTRAN によるシミュレーションコードに自動変換される．生成された FORTRAN コードは通常の手順でコンパイルされた後，必要に応じて線形計算ライブラリ，ユーザライブラリとリンクされ，実行に移される．結果として得られる大量の数値データは，ポストプロセッサ S-GRAF によって等高線図，ベクトル図，濃淡図などによりグラフィック表示することができる．

DEQSOL トランスレータは，連続系を離散化する手法として FDM (差分法)，FEM (有限要素法) の両者を取り込んでいる．FDM は矩形およびその組合せとして表現される領域における問題に対して効率のよい計算コードを生成する．一方 FEM は FDM に比べて処理の負荷は重いが，任意の領域形状に対して適用が可能である．利用者は自分の持つ問題のタイプに合わせて簡単に両者を使い分けられる．

さらに，FDM の領域形状に対する制限を緩和すべく，システム中に BFM (Body Fitted Coordinate Transformation Method : 領域写像法) プリプロセッサが用意されている．曲線 (曲面) に囲まれた領域における問題であっても，これに対する DEQSOL プログラム中の偏微分方程式に写像変換を施し，等価な FDM プログラムを生成することができる．BFM を用いたシミュレーションは，領域形状がやや複雑でかつ高速な計算の要求される半導体 CAD，流体解析などに対して有効と考えられる．

3. 言語仕様

3.1 言語の設計方針

DEQSOL 開発の狙いの1つである数値シミュレーションプログラムの生産性，保守性，および拡張性の向上を実現すべく，システムの入力となる DEQSOL の言語仕様は以下の方針に従って決定した．

(1) 数理モデルと計算スキームとの分離記述

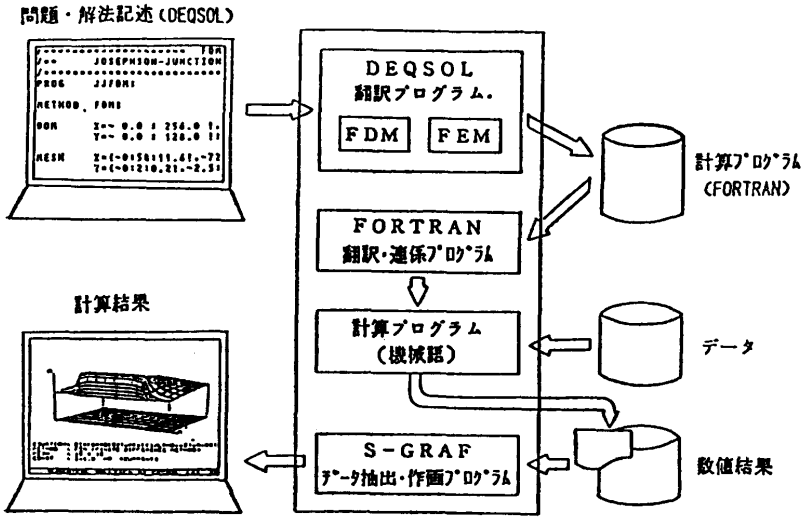
領域形状，物理定数，境界条件などの数理モデル記述部と解法スキーム記述部の明確な分離方式をとる．これによって，シミュレーションの過程で頻発するモデルあるいはアルゴリズムの修正に即応することを可能とし，スループットの向上を図る．

(2) 簡潔な数式記法の採用

記述量の大幅な削減とプログラムの可読性の向上を狙って，数式の数理解析書レベルでの記述を採用する．具体的には，

i) 多様な物理量を表すために，変数としてスカラ量，ベクタ量，テンソル量を導入する．

ii) 偏微分方程式や境界条件などを自然に記述するために，微分演算子として dx, dxx, div, $grad$, $lapl$, rot を導入する．



DEQSOL: Differential EQuation SOLver FDM: Finite Difference Method
S-GRAF: Scientific GRAphing Facilities FEM: Finite Element Method

図1 DEQSOL のシステム構成
Fig. 1 Processing flow of DEQSOL.

（3） 柔軟なスキーム記述能力

アルゴリズムの基本となる逐次実行，反復，分岐の記述を導入し，各種の数値計算スキームを柔軟に表現できるようにする．

（4） 領域形状の言語定義

領域形状の言語による定義を行い，シミュレーションに対して閉じた言語体系を提供する．ブロック積み上げ方式により複雑な形状に対応すると共に，領域の自動分割機能により必要最小限の入力で形状記述を可能とする．

（5） 並列性の自動抽出

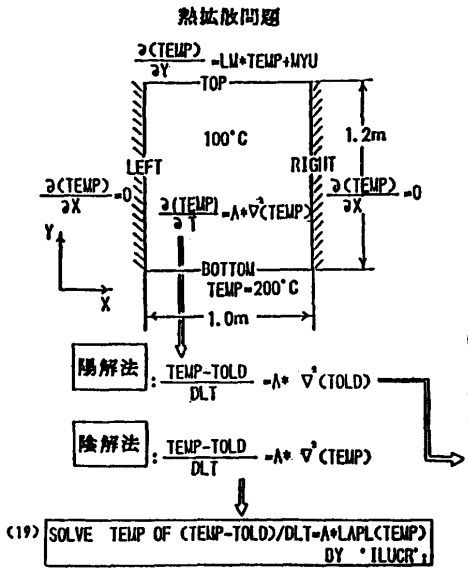
アルゴリズムに内在する並列性は特に利用者が意識しなくても言語中で自然に表現されるものとする．並列性の抽出は DEQSOL トランスレータが行い，スーパコンピュータ向けのコードを自動生成する．

3.2 記 述 例

DEQSOL による簡単な記述例を用いて，言語の概要とプログラムの構成を示す．

初期温度 100℃ のボックスについて，左右の壁を断熱とし，下部を 200℃ に固定，上部から熱の出入りを許した際の内部の温度変化を求める簡単なモデルと，それに対する DEQSOL の記述例を**図 2** に示す．前述のように，DEQSOL プログラムは数理モデルの構造を記述する部分と，計算スキームを記述する部分の 2 つに分けられる．

プログラムの記述はフリーフォーマットで，文種を表すキーワードで始まりセミコロン（;）で終わる文の列により構成される．図中の文(1)はプログラムの始まりとその名称を，文(2)は離散化手法（本例では FEM）の選択を指定している．文(3)から文(10)までは数理モデルの構造を記述している部分である．問題の定義されている空間領域(3)，非定常問題に対する時間領域(4)，差分法による計算格子(5)，未知の物理変数(6)，熱拡散定数などの物理定数(7)，各種の条件や局所的な値の設定時に引用する副領域(8)，初期条件(9)，境界条件(10)などを指定する．続く SCHEME 文(11)から END SCHEME 文(17)の間（スキームブロック）では計算スキームを記述している．非定常熱伝導問題を解くには，例えば空間微分を陽的あるいは陰的に評価することができる．図 2 の記述例はオイラー型の陽解法スキームの例で，ITERATION 文(12)から END ITERATION 文(16)の間（イテレーションブロック）の実行文は UNTIL 以下の条件が成立するまで繰り返し実行される．この間に代入文(14)により各時間ステップの温度が順次求められる．陽解法にかえて陰解法スキームを構成するのであれば，代入文(14)を SOLVE 文(19)で置き換えればよい．SOLVE 文は DEQSOL の基本機能の 1 つで，任意の線形偏微分方程式中の指定した変数の数値解を，BY 以下で指定した行列解法を用いて求めることを指定する．最後の END 文はプログラムの終わりを示す．




```
(1)  PROG      THERM;
(2)  METHOD    FDM;
(3)  DOMAIN    X=[0:1], Y=[0:1.2];
(4)  TIME      T=[0:1];
(5)  MESH      X=[0:1:5],
               Y=[0:1.2:6];
     TSTEP     DLT=[0(0.02)1];
(6)  VAR       TEMP, TOLD;
(7)  CONST     A=0.62, LM=0.8 MYU=50;
(8)  REGION    RIGHT=[1,*],
               LEFT=[0,*],
               TOP=[*,1.2],
               BOTTOM=[*,1],
               TOTAL=[*,*];
(9)  ICOND     TEMP=100 AT TOTAL;
(10) BCOND     DX(TEMP)=0 AT RIGHT+LEFT,
               TEMP=200 AT BOTTOM,
               DY(TEMP)=LM*TEMP+MYU AT TOP;
(11) SCHEME;
(12)      ITER NT UNTIL NT GT 50;
(13)           TOLD=TEMP;
(14)           TEMP=TOLD+DLT*A*LAPL(TOLD);
(15)           PRINT TEMP EVERY 10 TIMES;
(16)      END ITER;
(17) END SCHEME;
(18) END;
```

図 2 DEQSOL の記述例
Fig. 2 An example of description by DEQSOL.

には図2の例で示したのと同様

$D=DI*(1.0+N/NI)+DI**2*(N/NI)**2$

$NNEW=N+DLT*DIV\ (D*GRAD\ (N))$

の代入を繰り返すだけでよい．一方陰解法で解く場合，更新する変数に対して非線形となるので，逐次近似法を用いるかニュートン・ラフソン法を適用することになる．

図3のスキームは，クランク・ニコルソン陰解法に基づく時間進行に対して，ニュートン・ラフソン法を構成したものである．このように DEQSOL では，逐次代入法はもちろんニュートン・ラフソン法もスキーム中で記述できる．その手順は，D を N の関数とみなしてテイラー展開し，

$D(N)=DI\ (1.0+N/NI)+(DI**2)*(N/NI)**2$

$dD/dN=DI/NI+2*(DI**2)*N/(NI**2)=DP$

により，

$D(N+DN)=D+DN*DP$

$DIV\ (D(N+DN)*GRAD\ (N+DN))=DIV\ (D*GRAD(N))+DIV\ (DP*DN*GRAD(N))+DIV\ (D*GRAD(DN))+DIV(DP*DN*GRAD(DN))$

の最後の項を落とした式をもとにして図のようなスキームを構成すればよい．このスキームは2つのイテレーションブロックがネストしており，内側の繰り返しはニュートン・ラフソン法に対するもので，外側の繰り返しは時間進行をつかさどる．このように DEQSOL においてはイテレーションブロックの任意階のネストを許容している．

（2） 連立偏微分方程式系への対応

物理系のなかには，磁場解析におけるベクトルポテンシャルや，複数の不純物の混在する拡散過程のように，複数の変数がカプリングする連立偏微分方程式によって支配される問題がある．このような場合，連立している偏微分方程式を1本ずつ1変数について解く逐次解法と，連立させて解く一括解法，および関係の強い変数だけを一括して他は逐次的に解く中間的な解法などが考えられる．

図4は定常電流による3次元静磁界に対するモデルで，マクスウェルの電磁方程式から導いた拡散方程式を解いてベクタポテンシャル (AX, AY, AZ) を求めるものである．本図のスキーム例は逐次解法に対するもので，AX, AY, AZ に初期値を与えておき，第1式は AX, 第2式は AY, 第3式は AZ について解く．例えば第2式の SOLVE 文に対しては，AX, AZ は既定値を参照し，AY のみについて解く．この計算を AX, AY, AZ の値の変動が無くなるまで繰り返すことにより収束解が得られる．

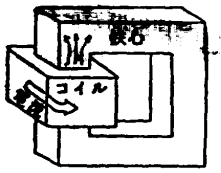
変数間のカプリングの強い連立系や，式のなかでの支配的変数が不明なモデルに対しては，上述の逐次解法は収束に長時間を要したり未収束となったりする．そうしたモデルに対しては，式中に含まれる全変数を直接連立させて大次元の連立1次方程式を構成して解く一括解法が必要となる．これに対応するため，複数の変数と偏微分方程式系を対象とするように SOLVE 文の仕様と機能が拡張されている．

（3） 大規模問題への対応

3次元領域における問題などでは，格子点数の増加にともない計算の負荷が増大する．とくに大規模問題

$ROT(\frac{1}{MYU}*ROT(A))=I$

A =(AX, AY, AZ): 磁場ポテンシャル
I =(IX, IY, IZ): 電流
MYU: 透磁率 { =1.0E-6=MYU0 on 真空
=2500* MYU0 on 鉄心




```
SCHEME;
 MYUX=DX(1/MYU); MYUY=DY(1/MYU); MYUZ=DZ(1/MYU);
 /*INITIAL SET*/
 AX=AX0;  AY=AY0;  AZ=AZ0;
 ITER  I1  UNTIL  NORM  LT  EPS;
   AXOLD=AX;
   AYOLD=AY;
   AZOLD=AZ;
   SOLVE  AX  OF  -DIV(1/MYU*GRAD(AX)) + MYUX*DX(AX)
                  + MYUY*DX(AY) + MYUZ*DX(AZ)  =  IX
                                  BY 'ILUCR';
   SOLVE  AY  OF  -DIV(1/MYU*GRAD(AY)) + MYUY*DY(AY)
                  + MYUZ*DY(AZ) + MYUX*DY(AX)  =  IY
                                  BY 'ILUCR';
   SOLVE  AZ  OF  -DIV(1/MYU*GRAD(AZ)) + MYUZ*DZ(AZ)
                  + MYUX*DZ(AX) + MYUY*DZ(AY)  =  IZ
                                  BY 'ILUCR';
   CALL  DIST2(AX, AY, AZ, AXOLD, AYOLD, AZOLD, NORM);
 END  ITER;
END  SCHEME;
```

図4 連立偏微分方程式の解法スキーム
Fig. 4 Nemerical scheme for simultaneous problem.

表 1 言語仕様
Table 1 Supported statements.

項番	文	指定項目	項番	文	指定項目
1	PROG 文	メインプログラム名称	16	EFUNC 文	外部関数名称
2	PROC 文	サブルーチン名称と引数	17	BCOND 文	境界条件
3	METHOD 文	離散化手法	18	ICOND 文	初期条件
4	DOMAIN 文	空間領域	19	COUNT 文	カウンタ名称
5	TIME 文	時間領域	20	EQU 文	対象式
6	MESH 文	空間領域のメッシュ点	21	解法ブロック	
7	TSTEP 文	時間間隔	22	代入文	変数への代入
8	REGION 文	領域	23	SOLVE 文	行列解法指定
9	CONST 文	定数	24	条件ブロック	条件別の実行文
10	CVEC 文	定数ベクトル	25	CALL 文	サブルーチンの呼び出し
11	VAR 文	変数	26	繰り返しブロック	解法手順の繰り返し
12	SVAR 文	配列を持たない変数	27	PRINT 文	数値出力
13	VEC 文	ベクトル	28	WRITE 文	ファイル出力
14	INTEGER 文	整数型	29	SAVE 文	数値マスタファイル出力
15	EPROC 文	外部サブルーチン名称	30	END 文	プログラム終了

表 1 に離散化手法として FDM を用いる場合の文種の一覧を示す[4].

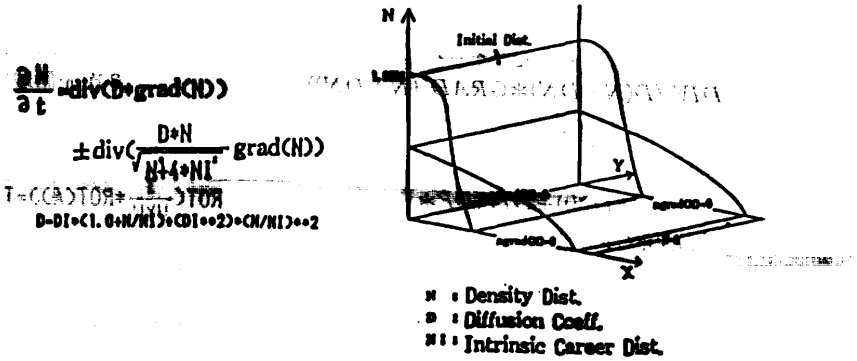
3.3 各種解法スキームの構成例

各種の問題に対する DEQSOL プログラムの構成法を示し，本言語仕様の記述能力を例証する．

（1） 非線形偏微分方程式への対応

通常，非線形方程式を解く場合，非線形項を既知量によって評価する逐次近似法を用いるか，ニュートン・ラフソン法を適用する．

図 3 は半導体製造プロセスに現れる不純物拡散問題を示している．初期注入された不純物は，拡散項とドリフト項からなる非線形の拡散方程式によりその濃度分布が支配される．ここで D は濃度 N に依存する拡散係数であり，拡散項，ドリフト項はともに非線形となっている（図中のスキーム例および以下の説明は簡単のためドリフト項を省略している）．これを解く場合，時間依存性を陽解法で解くのであれば，非線形性は全く意識する必要はない．この場合




```
SCHEME;
 /* Time iteration loop */
 ITER II0 UNTIL II0 GT 20;
   DO=DI*(1.0+N/NI)+DI**2*(N/NI)**2;
   /* Newton-Raphson loop */
   ITER II1 UNTIL VNORM LT eps1;
     D=DI*(1.0+NNEW/NI)+DI**2*(NNEW/NI)**2;
     DP=DI/NI+2*DI**2*NNEW/NI**2;
     SOLVE WITH DN  OF
       (NNEW+DN-N)/DLT =1/2*
           ( DIV(D*GRAD(NNEW))+DIV(DP*DN*GRAD(NNEW))+
             DIV(D*GRAD(DN))+DIV(DO*GRAD(N)) )
     BY 'ILUCR'  WITH EPS(1.0E-5);
     NNEW=NNEW+DN;
     /* Calculate Norm of Residual DN */
     CALL DIST(DN, VNORM);
   END ITER;
   PRINT NNEW AT WHOLE EVERY 5 TIMES;
   N=NNEW;
 END ITER;
END SCHEME;
```

Time Iteration

Newton Iteration

図 3 非線形偏微分方程式の解法スキーム
Fig. 3 Numerical scheme for non-linear problem.

に対して陰解法を適用する場合には，生成される連立1次方程式の元数が膨大となりメモリネックを生じやすい．このような場合にも計算の実行を可能とする手法として，部分構造反復法が知られている．

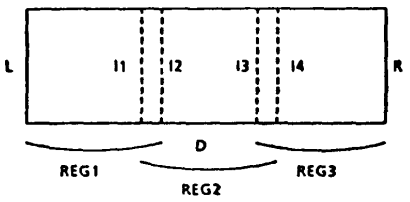
図 5 は2次元場においてラプラス方程式を解く問題に対して，部分構造反復法を適用する場合のDEQSOL スキーム構成例である．対象領域を互いにオーバラップを持ついくつかの部分領域に分割し，領域名称をつける（本例では REG 1, REG 2, REG 3）．このとき部分領域間に生ずる境界（本例では I1, I2, I3, I4）を仮想境界と呼ぶ．SCHEME ブロック中では各部分領域 REG 1, REG 2, REG 3 に順次陰解法を適用する SOLVE 文を列記する．陰解法の適用領域の指定は SOLVE 文の末尾の AT 以下に領域名称を指定することにより行う．これをイテレーションブロック中に記述することにより，オーバラップ領域を介して仮想境界上の境界値を逐次更新する FORTRAN コードが生成され，部分構造反復法による収束解を得ることができる．

部分構造反復法では，SOLVE 文の対象となる部分領域のうち最大のものに対するメモリエリアを確保すれば良い．したがって，問題の規模とメモリ環境に合わせて部分領域への分割を行えば，大規模な問題に対しても柔軟に対応することができる．

（4） FEM, BFM における形状記述法

FEM, BFM 機能を用いる場合には，多様な領域形状を記述するために拡張された領域定義（REGION 文），メッシュ定義（MESH 文）を利用することができる．

図 6 は電子銃の断面を表しており，1点鎖線を中心軸とする回転対称形状である．DEQSOL では，2次元は3辺形か4辺形（3次元は4面体，3角柱か6面体）からなるブロックを積みあげていくことにより，このような自由曲線で囲まれた複雑な形状の定義を可能としている．各ブロックは，まず端点や特徴点などの点を，次いで点を用いて辺を，辺を用いてブロックをというように下位構造より上位構造に向かって定義していく．図の記述例では，POINT 文で定義点や特徴点を，REGION 文で線分，円弧，スプライン曲線を用いて周囲辺とブロックを定義している．このようにして対象領域全体をブロックの集合として表現する．メッシュの疎密制御は，MESH 文でブロックの各辺の分割を指定することにより行う．内部のメッシュは自動的に生成される．




```
VAR     TT;

REGION  TT = 0  AT L,
        TT = 100  AT R,
        DY(TT) = 0  AT U + D,
        TT = GIVEN  AT I1 + I2 + I3 + I4;

SCHEME;
    ITER NN UNTIL NN EQ 10;
      SOLVE TT OF  LAPL(TT) = 0  BY 'GAUSS' AT REG1;
      SOLVE TT OF  LAPL(TT) = 0  BY 'GAUSS' AT REG2;
      SOLVE TT OF  LAPL(TT) = 0  BY 'GAUSS' AT REG3;
    END ITER;
    PRINT PHI;
END SCHEME;
END;
```

図 5 部分構造反復法の解法スキーム
Fig. 5 Numerical scheme for block-iteration.

```
point   P1 = (0.0,5.0),
        P2 = (0.0,1.0),
        P3 = (4.0,0.0),
        ・・・ ;
region  L12 = ln(P1,P2),            ・・・線分
        L23 = arc(P2,PA,P3),        ・・・円弧
        L34 = ln(P3,P4),            ・・・線分
        L41 = spl(P1,PS1,PS2,P4),   ・・・スプライン曲線
        AREA1 = quad(L12,L23,L34,L41),
                                    ・・・部分領域(ブロック)
        ・・・ ;
mesh    L12 = d(P1,3),              ・・・等分割
        L23 = r(P2,5,0.8),          ・・・等比分割
        L34 = d(P3,6),              ・・・等分割
        L41 = r(P1,5,0.8),          ・・・等比分割
        AREA1 = a(P2,P3),           ・・・ブロック自動分割
        ・・・ ;
```

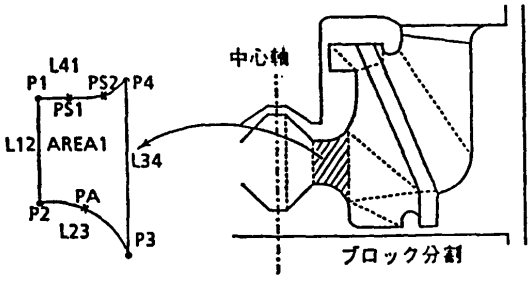


図 6 FEM の形状記述方法
Fig. 6 Shape description using FEM facility.

4. FORTRAN コード生成方式

4.1 生成コードの構成

DEQSOL トランスレータは，構文解析部とコード生成部と呼ぶ2つの処理よりなる．構文解析部は，データ型や文型などの文法チェック，領域の無矛盾性チェック，初期条件や境界条件の適合性と過不足性チェックなどを行い，コード生成に適した中間語を生成する．コード生成部は，その中間語よりプログラムに必要な変数や作業領域の宣言，各実行文対応のサブルーチンや制御ループを FORTRAN によって生成する．特に代入文や SOLVE 文に対しては，偏微分方程式を離散化し，代入式や係数行列を計算するプログラムを生成する．

図7に生成コードの構成を示す．コードは1つのメインプログラムと多数のサブルーチン群よりなる．メインプログラムの冒頭では，VAR 文や CONST 文などで指定された変数や行列解法に必要な作業領域などの宣言が行われる．メインプログラムの実行部は3つの処理からなる．第1の処理は，メッシュ情報の生成やプログラム内パラメータなどの設定を行う前処理部である．第2の処理は，初期条件や既知定数などの初期設定部である．第3の処理は，基本的には DEQSOL のスキームブロックと同等の構造をもち，スキームブロック中の各実行文に対応するサブルーチンを呼び出す CALL 文，入出力文，制御文などからなる．このメインプログラムに引き続いて，各 CALL 文対応のサブルーチンが生成される．

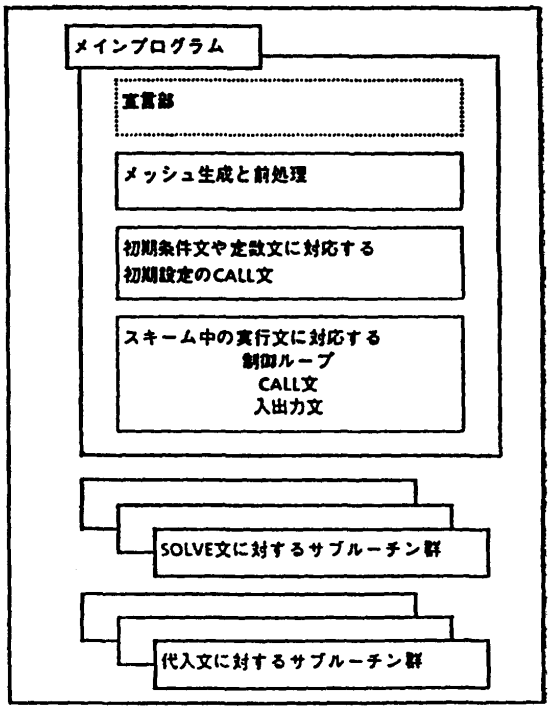


図 7 生成 FORTRAN の構成
Fig. 7 Structure of generated FORTRAN code.

4.2 自動離散化と並列化コード生成方式

DEQSOL トランスレータは，与えられた偏微分方

表 2 差分展開規則
Table 2 Expansion rules of derivative operators (FDM).

微分演算子	数学的表現	内点での差分形	境界での差分形
DX	$\partial/\partial X$	$\{U(I+0.5,J,K)-U(I-0.5,J,K)\}/DLX(I+0.5)$	$\{U(I+I,J,K)-U(I,J,K)\}/DLX(I+1)$ または $\{U(I,J,K)-U(I-1,J,K)\}/DLX(I)$
DY	$\partial/\partial Y$	$\{U(I,J+0.5,K)-U(I,J-0.5,K)\}/DLY(J+0.5)$	$\{U(I,J+1,K)-U(I,J,K)\}/DLY(J+1)$ または $\{U(I,J,K)-U(I,J-1,K)\}/DLY(J)$
DZ	$\partial/\partial Z$	$\{U(I,J,K+0.5)-U(I,J,K-0.5)\}/DLZ(K+0.5)$	$\{U(I,J,K+1)-U(I,J,K)\}/DLZ(K+1)$ または $\{U(I,J,K)-U(I,J,K-1)\}/DLZ(K)$
DXX	$\partial^2/\partial X^2$	$DX(DX(U))$	$DX(DX(U))$
DYY	$\partial^2/\partial Y^2$	$DY(DY(U))$	$DY(DY(U))$
DZZ	$\partial^2/\partial Z^2$	$DZ(DZ(U))$	$DZ(DZ(U))$
GRAD	*grad*	$(DX(U), DY(U), DZ(U))$	$(DX(U), DY(U), DZ(U))$
LAPL	∇^2 または Δ	$DXX(U)+DYY(U)+DZZ(U)$	$DXX(U)+DYY(U)+DZZ(U)$
DIV	*div*	$DX(U)+DY(V)+DZ(W)$	$DX(U)+DY(V)+DZ(W)$
ROT	*rot*	$(DY(W)-DZ(V), DZ(U)-DX(W), DX(V)-DY(U))$	$(DY(W)-DZ(V), DZ(U)-DX(W), DX(V)-DY(U))$
DT	$\partial/\partial t$	——	——

程式を数式的に変形，離散化しコードを生成する．以下では，FDM, FEM の各機能についてコード生成の手順を示す．

（1） FDM

FDM 機能では，与えられた偏微分方程式，あるいは境界条件に差分展開規則を繰り返し適用して離散化を行う．**表 2** にシステムが保持する展開規則の一覧を示す．差分展開は基本的には1階微分 *DX*, *DY*, *DZ* を対象として適用する．対象領域の内部では偏微分方程式に対して両側差分

$$DX(@)=(@\langle i+1/2,j,k\rangle-@\langle i-1/2,j,k\rangle)/DLX$$

を適用し，境界（例えば左側境界）では境界条件に対して片側差分

$$DX(@)=(@\langle i+1,j,k\rangle-@\langle i,j,k\rangle)/DLX$$

を適用する．一般の微分演算子 *DIV*, *GRAD* などは，表2の微分演算子の展開規則を再帰的に適用して1階微分の階層的な表現に変形した後，上記の差分展開を施す．*DIV* を例にとれば，

$$\begin{aligned}DIV(@)&=DXX(@)+DYY(@)+DZZ(@)\\&=DX(DX(@))+DY(DY(@))+DZ(DZ(@))\\&=(@\langle i+1,j,k\rangle-2*@\langle i,j,k\rangle+@\langle i-1,j,k\rangle)/DLX^2\\&\quad+\cdots\end{aligned}$$

のように展開される．

この差分展開は対象領域中の全格子点について行う必要があるが，各格子点ごとに独立に離散化，コード生成を行うことは処理負荷と生成コード量との著しい増加をまねく．これを避けるためにトランスレータはあらかじめ，偏微分方程式，境界条件と離散化規則に従って領域を同一の式と離散化規則に支配される部分領域（基本領域）に分解する．これにより，各基本領域ごとに離散化ができるようになり，トランスレータの処理負荷が減少する．また基本領域が単一処理を行う単位となっているため，これと1対1対応に DO ループを生成することが可能となり，生成コードのコンパクト化と実行効率の向上を図ることができる．

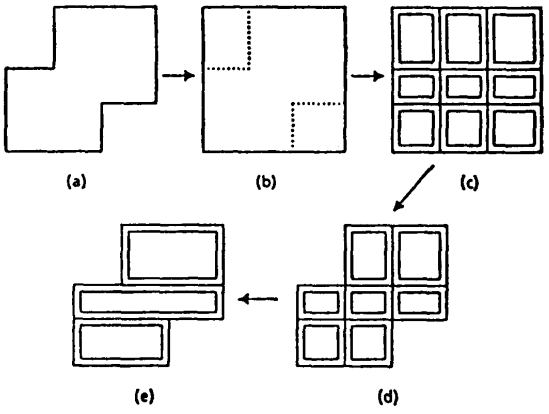


図 8 領域分解方式
Fig. 8 An example of domain decomposition.

図8に領域分解の例を示す．まず，領域を包絡する矩形を生成し(b)，これを各方向に分割して最小の基本領域を生成する(c)．つぎに，各基本領域がもとの領域(a)の内部か外部かを判定し，内部に含まれるもののみを抽出する(d)．この時点で，外部と内部の基本領域に挟まれた基本領域が境界となることを利用して，境界領域の抽出が可能となる．さらに，最小基本領域が統合できるかどうかを調べ，隣接する最小基本領域のうち同一の離散式で処理可能なものを統合する(e)．このようにして単一の DO ループで処理可能な範囲を最適化することにより，ループ長を拡大し，スーパコンピュータによる加速率の向上を図っている．

すべての基本領域についての係数計算コードの生成後，あらかじめシステムが用意している線形計算ライブラリをリンクする CALL 文を生成する．線形計算ライブラリ自身も高度にベクトルチューニングがなされており，全体として高い加速率を得ることができる．

（2） FEM

FEM 機能では，離散化手法として汎用性の高いガレルキン式有限要素法を採用している．以下，数式レベルでの離散化の手順の概略を示す．

偏微分方程式（例えば $DIV\ (GRAD\ (@))=0$）が与えられた場合，これに重み関数 N_i を乗じて領域全体で積分する．

$$\int DIV\ (GRAD(@))*N_i dv=0$$

次にこれに部分積分を施し

$$-\int GRAD(@)*GRAD(N_i)dv+\int NGRAD(@)*N_i ds=0$$

のように変形する．ここで，第2項(境界積分項)に第2種境界条件（例えば $NGRAD(@)=q$）を導入し，

$$-\int GRAD(@)*GRAD(N_i)dv+\int q*N_i ds=0$$

とする．さらに，変数値を補間関数 N_j を用いて

$$@=\Sigma @_j N_j$$

と離散化し，求めるべき変数について $\int$ と Σ の順序を交換して，

$$-\Sigma\left(\int GRAD(N_j)*GRAD(N_i)dv\right)*@_j + \int q*N_i ds=0$$

と変形する．ここで，変数@に対する係数行列と定数ベクトルの成分を

$$M_{ij}=\int GRAD(N_j)*GRAD(N_i)dv$$

$$C_i=\int q*N_i ds$$

のように切り出すことができる．

このように偏微分方程式をガレルキン法に基づいて数式的に変形し，係数行列，定数ベクトルの成分を切り出した後，これを実際に評価する FORTRAN コードを生成する．コード生成に際しては，システムに用意された評価対象式と離散式の間の対応テーブルを参照し，離散式に含まれる積分を4則演算による数値積分式に変換する．

ここで，インデクス i, j は全節点をスキャンする必要がある．DEQSOL の生成する FORTRAN コードのループ構成を**図9**（b）に，通常の有限要素法プログラムに見られるループ構成を図9（a）に示す．DEQSOL は要素に関するループを最内側に入れ，要素ごとの i, j のループを外側に出すようにコードを生成する．要素ごとの i, j に関するループ長はたかだか3ないし4であるのに対し，要素数は通常数百から数千となる．これを最内側のループとすることにより，並列計算可能なループ長を拡大しスーパコンピュータの性能を引き出すことを狙っている．なお，実際に図9（b）に従ってコードを生成すると，係数行列への値の格納時にデータストアのコンフリクトが発生しループ全体がベクトル化

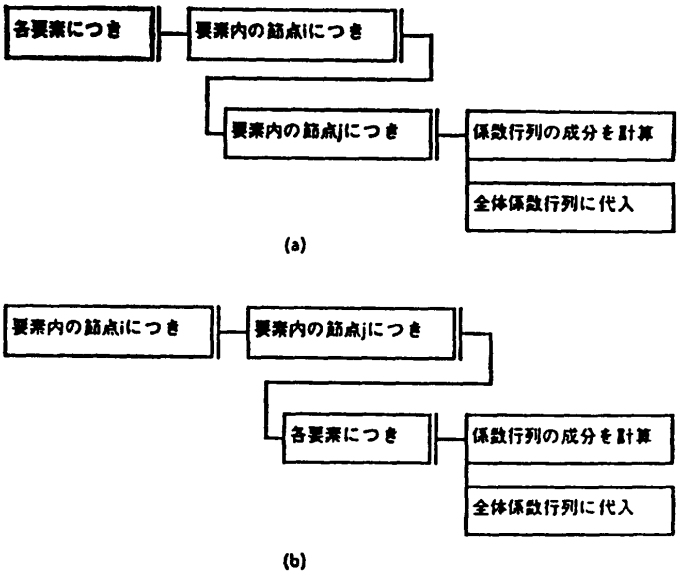


図 9 FEM による生成コードのループ構成
Fig. 9 Loop structure in generated FORTRAN code (FEM).

表 3 生成 FORTRAN の性能評価
Table 3 Evaluation of generated FORTRAN code.

解析手法		FDM			FEM		
適用例		不純物拡散シミュレーション	放電管シミュレーション	LSI チップ冷却シミュレーション	ジョセフソン素子冷却シミュレーション	ディスクヘッド磁場解析	光電管内電位分布解析
次元		2	2	2	2	3	軸対象3
節点数		676	2601	2009	437	715	679
記述行数	DEQSOL	79	35	34	96	904	147
	FORTRAN	1312	370	347	1078	11558	1198
	比	16.6	10.6	10.2	11.2	12.8	8.2
実行性能	ベクトル化率	96	94	98	93	94	96
	スカラ/ベクタ	8.2	5.7	6.1	5.1	7.7	7.6
翻訳性能	翻訳時間 (sec)	30	5	7	18	50	25
	メモリサイズ (MB)	2	2	2	3	5	3

不能になってしまう．これを避けるために，事前に全要素を互いに節点を共有しないいくつかのグループに分類し，このグループごとにループを生成するよう配慮している．

行列生成部分が完了した後，行列解法ライブラリに対する CALL 文を生成するのは FDM の場合と同様である．

5. 評　価

DEQSOL 開発の狙いである，(1)プログラム生産性の向上と，(2)スーパコンピュータの有効活用について評価を行う．

表3は設計，研究の現場の問題に DEQSOL を適用した際の評価結果の一部である．本表は各適用例題に対して，用いた数値計算（離散化）手法，問題規模，DEQSOL の記述行数と生成 FORTRAN 行数との比(生成効率)，HITAC S-810/20 を用いた場合の演算並列化率（ベクトル化率），ベクトル計算時のスカラ計算に対する性能向上比（加速率），および DEQSOL トランスレート時に必要なリージョンサイズと翻訳時間を示している．なお，翻訳処理性能は M-680 H によって計測している．

これらの評価から次のことがいえる．

(1) プログラムの生産性はコード行数を尺度として，FORTRAN に対して1桁程度高い．表3中の FORTRAN 行数のうち，ジョセフソン素子解析は既存の人手によるプログラムの行数を示しているが，これに対しても1桁程度高い記述効率を有している．

(2) 自動生成 FORTRAN コードは HITAC S-810/20 において 93.0 から 98.0% の高い演算並列化率を達成しており，問題規模にもよるが 5.1 から 8.2 倍の性能向上を得ている．内訳で見ると，行列生成部は 6.2 から 8.5 倍，行列解法部は 8.4 から 13.9 倍の性能向上を達成し，ループ長の長い後者の方がより加速率が高い．非ベクトル化部は主に入出力部と個別の代入式である．

6. 結　言

以上，数値シミュレーションシステム DEQSOL の言語仕様，機能と処理方式の概要を述べた．また，いくつかの問題への適用例を通じて，FORTRAN によるコーディングの 1/10 以下の記述行数で 90% 以上のベクトル化率を有するコードの自動生成が可能であることを示した．

今後はさらに機能の拡充を行うとともに，マンマシンインタフェースの面からも使いやすいソフトウェア環境を提供していきたい．

参 考 文 献

1) Morris, S. M. and Sciesser, W. E.: SALEM—A Programming System for the Simulations of Systems Described by Partial Differential Equations, *Proc. Fall Joint Computer Conference*, Vol. 33, pp. 353-357 (1968).
2) Cardenas, A. F. and Karplus, H. J.: PDEL—A Language for Partial Differential Equations, *Com. ACM*, pp. 184-191 (March 1970).
3) Rice, J. R. and Boisvert, R. F.: Solving Elliptic Problem Using ELLPACK, CSD-TR 414, Computer Science Department, Purdue University, September 1982 (Revised May 1983).
4) マニュアル：HITAC プログラムプロダクト VOS 3 偏微分方程式向き数値シミュレーション言語 DEQSOL-BS, 8090-7-059/060 (1986).

(昭和63年 4月 8日受付)
(昭和63年11月14日採録)

佐川　暢俊（正会員）

昭和35年生．昭和58年東京大学工学部精密機械工学科卒業．昭和60年同大学院修士課程修了．同年より(株)日立製作所中央研究所に勤務し，以来数値シミュレーションシステムの研究開発に従事する．機械学会会員．

金野　千里（正会員）

昭和27年生．昭和50年東京工業大学理学部情報科学科卒業．昭和52年同大学院修士課程修了．同年(株)日立製作所入社．以来，中央研究所にて，図形処理システム，数値計算技術の研究・開発に従事．

梅谷　征雄（正会員）

昭和19年生．昭和43年東京大学理学部数学科卒業．同年(株)日立製作所入社．以来，同社中央研究所にて，デジタルシステムの故障診断，ベクトル化コンパイラ，高水準言語 DEQSsL，並列記述言語などの研究に従事して現在に至る．ACM 会員．