ハイパフォーマンス
コンピューティング 59－1
（1995．12．11）

可読性を重視したプログラムの生成に関する研究
－　プログラムの生成支援システム(NCAS-FEM)における一考察　－

渋井俊昭，Choompol Boonmee，川田重夫
長岡技術科学大学

近年，コンピュータは人間の社会活動、日常生活に深く浸透し現代の人間社会を支える上で，なくてはならない存在になっている．このように人間とより密接になりつつあるコンピュータだが，それを動作させるためのプログラム（ソフトウェア）作成は，年々巨大化，複雑化の傾向を呈しており，プログラム開発には非常に多くの労力と費用が必要となってきた．このような中，コンピュータにシステム設計の一部を肩代わりさせ，人間との協調によってシステムを開発して行こうという動きがある．そうしたとき，我々は“コンピュータに対してどのようなスタンスをとれば良いのか”，“人間のパートナーとしてコンピュータを位置付けるには何が必要なのか”を考えていかなくてはならない．そこで本研究では，プログラムを人間とコンピュータとの間のインターフェイス（双方向の意志伝達手段）と位置付け，その際に重要となってくることの一つとして，“可読性”という概念を取り上げた．そして，物理分野の数値シミュレーションコード（有限要素法解析）生成支援システムを作成した．プログラム生成支援システムが可読性を重視したプログラムを生成することによって，人間とコンピュータとのより良い協調関係が実現でき，信頼性の高いプログラムを構築することが可能となる．

Readability of Simulation Code in FEM-Code Generator NCAS-FEM.

Toshiaki Shibui, Coompol Boonmee and Shigeo Kawata
Nagaoka University of Technology,
1603-1 Kamitomioka, Nagaoka, Niigata.940-21, Japan.
e-mail: kawata@voscc.nagaokaut.ac.jp

Nowadays, computer is an important factor for our life and society. Computer programs may be large and complex, so we need more human efforts and much time in order to create them. In our work we recognize that a program(sorce code) is an interface between a human and a computer. In this paper, we discuss on readability of a simulation code generated by our sysytem NCAS-FEM. The NCAS-FEM system is a scientific-problem-solving environment (PSE) based on FEM, and generates a source code with a high-readability. We think that the readability is very important for code generation support systems and for PSEs.

1．導入

我々は，プログラム生成支援システムの出力情報（とりわけ，生成されるプログラムコード）に注目し，出力コードはどうあるべきかを考えた．そして，プログラム生成支援システムの出力コードには，可読性（読みやすさ，解りやすさ）が重要であるとの認識を得た．本研究は，可読性を重視したプログラムを生成するプログラム生成支援システムを構築しその有効性を実証しようというものである．プログラム生成支援システムを構築するにあたり，我々は次のような条件を設定し生成支援システムの設計・制作をおこなった．

1. 物理分野の数値シミュレーションコードを生成する支援システム．
2. 離散化の手法として有限要素法を用いる．
3. 生成されるソースコードに可読性という概念を取り入れる．
4. プログラムコード（キャラクタ・ベース表現）でいかに可読性を高めるかを考慮する．

この分野には物理シュミュレーション支援システムとして，DEQSOL[1-3]，ELLPACK[4]，ALPAL[5,6]，EVE[7]などが存在する．これらのシステムは，プログラム生成型と，ライブラリ駆動型の２つの種類におおまかに分けられる．プログラム生成型は，DEQSOL，ALPAL などで，その他は後者に分類される．本システムは前者に分類される．従来，この分野のシステムには次のような問題点があった．

1. ライブラリ駆動型システムは，計算の内容がブラックボックスのため，ユーザによって計算の詳細を知る術がなく，幾つかの問題による検証を行った後でのみ，信頼性が得られる．（統計的信頼性しか得られない．）
2. ソースコード生成型のシステムは，スーパーコンピュータや並列コンピュータなど多くの処理系でコンパイルできるようにすることが目的となっているため，一般に生成されたコードは人間に読まれることを考慮していない．（したがって，内容の検証は不可能で，統計的信頼性しか得られない．）
3. パラメータスタディ（最適な係数を見つけ出す作業）のようにコードのごく一部を変更すれば良いような作業でも，すべての解析処理をやり直さなくてはならないシステムが多い．とりわけ，この作業は何度となく繰り返されるため，解析時間が無駄となる．

このように，従来のシステムは信頼性の点からも，本来得たいはずの“論理的信頼性”を得ることができない．すなわち，これらのシステムには，ユーザに計算の詳細を検証させるための手段が提供されていないということである．このことは，数値シミュレーションのユーザよりも数値シミュレーションプログラム開発者には問題になってくる．数値シミュレーションプログラムを提供する側にとってみれば，彼らをサポートする生成支援システムが正しいコードを出力しているかどうかは，ソースコードレベルで検証しなくてはならないからである．

2．NCAS-FEMでの可読性について

そこで，プログラム生成支援システム[8-12]に可読性という概念を取り入れることによってどういったメリットが生まれてくるかについて述べる．可読性（読みやすさ，解りやすさ）という概念は，いわば“人間というインタープリタ”に通すための文法（書式）だと考えることができる．可読性を高めることによって，ユーザは生成されたプログラムの動作・構造などを知ることができる．また，可読性の向上は，前述した問題点を解決する．その上，可読性はソフトウェアの機能・自由度を妨げるものではない．すなわち，可読性を向上させることによって，支援システムが生成するプログラムの検証・修正・追加を，ユーザに容易に行わせることができる．このような検証のための手段を提供することが，これからのプログラム開発支援システムに重要になってくる．

我々はシステムを構築するにあたり，可読性についての定義付けをおこなった．（設計・記述等に関する約７０項目の定義付け）

そして，この定義にそった形のプログラムを可読性の高いプログラムとし，これらを生成するシステムを構築した．

3．システムの概要

本システムは，大まかに次のような動作を行う．

1. 入力仕様（問題記述）ファイルを読み込む．
2. メッシュ情報ファイルを読み込む．これらの情報から，プログラム作成に必要な諸情報を抽出する．
3. 方程式情報を元に，有限要素法の離散化手法を用いて離散化処理を行う．（処理の内容は記号処理的手法を用いる．）
4. 計算プログラム（中間言語形式）を生成する．
5. 中間言語プログラムをトランスレータにかけ，実際の計算言語（Ｃ，Ｆｏｒｔｒａｎに変換する．）

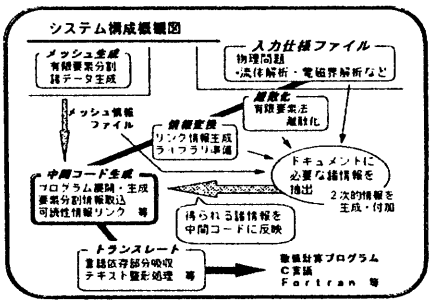

図１．システム構成概観図

４．プログラム生成例

本システムによって生成されたＣ言語の２次元拡散方程式（複数方程式）の数値シミュレーションプログラム例を示す．

$$\frac{\partial}{\partial t}T + L\frac{\partial^2}{\partial x^2}T + L\frac{\partial^2}{\partial y^2}T = f_1(x,y) \quad \cdots (1)$$

$$K = T \wedge A \quad \cdots (2)$$

$$\frac{\partial}{\partial t}N + K\frac{\partial^2}{\partial x^2}N + K\frac{\partial^2}{\partial y^2}N = f_3(x,y) \quad \cdots (3)$$

図２．計算方程式

これら３本の方程式をオイラーの陽解法で計算するプログラムを構築する．例としては本システムの特徴的な部分をあげる．

```
[EQUATION]
        Equation1=(+ (+ (Dt T) (+ (- (Dx (* L (Dx T)))) (- (Dy (* L (Dy T)))))) (+ F1))
        Boundary1=(- q1)
        Equation2=(= K (^ T A))
        Equation3=(+ (+ (Dt N) (+ (- (Dx (* K (Dx N)))) (- (Dy (* K (Dy N)))))) (+ F3))
        Boundary3=(- q3)
```

図３．入力仕様ファイル（一部）

```
/*
 ************************************************************************
 メインプログラム：システムフローチャート

 +------+
 | 開始 |
 +------+
 |
 +--------------------------------+
 | メインルーチン内の変数の初期化 |
 +--------------------------------+
 |
 +------------------------+
 | 出力ファイルをオープン |
 +------------------------+
 |
 +----------------------------+
 | 各方程式の初期化処理呼び出し |
 +----------------------------+
 |
 +--------------------------+
 | メッシュデータの読み込み |
 +--------------------------+
 |
 +----------------+
 | 初期条件の保存 |
 +----------------+
 |
 +------------------+
 | 解の繰り返し計算 |
 +------------------+
 |                  |
 |                  +-----------------+
 |                  | 方程式 1 の計算 |
 |                  +-----------------+
 |                  |
 |                  +-----------------+
 |                  | 方程式 2 の計算 |
 |                  +-----------------+
 |                  |
 |                  +-----------------+
 |                  | 方程式 3 の計算 |
 |                  +-----------------+
 |                  |
 |                  +----------+
 |                  | 解の出力 |
 |                  +----------+
 |
 +------+
 | 終了 |
 +------+
 ************************************************************************
 メインルーチン
 ************************************************************************
*/
void main(void)
{
        long    End_Counter = 0;
        long    Output_Counter = 0;
        long    End_Limit    = (long)(END_TIME / TIME_STEP);
        long    Output_Limit = (long)(END_TIME / OUTPUT_TIME);

        OpenDataFile();

        eqn1_Initialize();
        eqn2_Initialize();
        eqn3_Initialize();

        LoadMeshData();

        eqn1_Store_Initial_Condition();
        eqn2_Store_Initial_Condition();
        eqn3_Store_Initial_Condition();
```

図４．Ｃプログラムメインルーチン

```
        for (End_Counter = 0
             ; End_Counter <= End_Limit
             ; End_Counter = End_Counter + 1) {

                eqn1_Calculate_Equation();
                eqn2_Calculate_Equation();
                eqn3_Calculate_Equation();

                if (End_Counter % (End_Limit / Output_Limit) == 0) {
                        WriteData();
                }
        }

        CloseDataFile();
}
```

図４．Ｃプログラムメインルーチン（つづき）

```
/*
; ============================================================
; この関数は、方程式１のマトリクスの個々の要素の値を計算します。
; ============================================================
*/
void eqn1_Calculate_Local_Matrix_K(long Local_Node_1
                                   , long Local_Node_2
                                   , double Weight)
{
        eqn1_Matrix_Local_K[Local_Node_1][Local_Node_2]
                = eqn1_Matrix_Local_K[Local_Node_1][Local_Node_2]
                + (-1)
                  * Weight
                  * eqn1_Determinant_Jacobian
                  * eqn1_DNDX[Local_Node_1]
                  * ((-1) * L * eqn1_DNDX[Local_Node_2]);
        eqn1_Matrix_Local_K[Local_Node_1][Local_Node_2]
                = eqn1_Matrix_Local_K[Local_Node_1][Local_Node_2]
                + (-1)
                  * Weight
                  * eqn1_Determinant_Jacobian
                  * eqn1_DNDY[Local_Node_1]
                  * ((-1) * L * eqn1_DNDY[Local_Node_2]);
}
```

図５．離散化の結果生成された関数（一部）

```
/*
; ================================================================
; １次元配列の宣言
; ----------------------------------------------------------------
; 各方程式間で利用される１次元配列.
; g_SOLVED_N[TOTAL_NODE]
; g_SOLVED_T[TOTAL_NODE]
; g_SOLVED_K[TOTAL_NODE]
; ----------------------------------------------------------------
; 方程式 １で利用する１次元配列。
; eqn1_Vector_Total_F[]       : 全体右辺ベクトルを格納する配列。
; eqn1_Vector_Local_F[]       : 部分右辺ベクトルを格納する配列。

（途中略）

; eqn3_DNdeta[]  : 形状関数（正規領域）を eta で微分した際の値を格納する配列。
; eqn3_N[] : 形状関数（正規領域）を格納する配列。
; ================================================================
*/
/*
; ----------------------------------------------------------------
; 各方程式間で利用される１次元配列の宣言.
*/
double     g_SOLVED_N[TOTAL_NODE + 1];
double     g_SOLVED_T[TOTAL_NODE + 1];
double     g_SOLVED_K[TOTAL_NODE + 1];
/*
; ----------------------------------------------------------------
; 方程式 １で利用する１次元配列の宣言
*/
double     eqn1_Vector_Total_F[TOTAL_NODE + 1];
double     eqn1_Vector_Local_F[MAX_LOCAL_NODE + 1];
（途中略）
double     eqn3_dNdeta[MAX_LOCAL_NODE + 1];
double     eqn3_N[MAX_LOCAL_NODE + 1];
/*
```

図６．生成された一次元配列の宣言

5．まとめ

可読性の高いプログラムの生成によって，ユーザはプログラムに対する検証が可能になり，論理的信頼性を得ることができるとともに，各自による生成されたプログラムの拡張・変更，計算ロジックのチューニングが容易に行える．このように，可読性の高いプログラムを生成することによって，単なるソースジェネレータとしてのコンピュータではなく，人間に理解されるプログラムを生成するプログラム開発支援システムを構築することによって，コンピュータはより人間のパートナーとしての位置付けを確立してゆくことができると我々は考えている．

本システムは，今回の例で示したように複数本の方程式に対しても適用可能である．数値シミュレーションプログラムは，各方程式毎に計算を行えばよいため，各方程式間で計算プロセスの独立性が保てることに特徴がある．したがって，本システムのように，問題を解析し，プログラムを生成する柔軟な構造を持つシステムでも一般性を持つことができ，多くの方程式に対してプログラムコードを構築するシステムを作成することができる．

参考文献

[1]佐川暢俊，金野千里，梅谷征雄：数値シミュレーション言語 DEQSOL，情報処理学会論文誌，第30巻第1号別刷，Vol.30，No.1，pp.36-45(1989)．

[2]大河内俊夫，金野千里，猪貝光祥：数値シミュレーション向き高水準言語 DEQSOL の分散メモリ型並列計算機向けトランスレーションに関する一考察，並列処理シンポジウム JSPP'93，pp.39-46(1993)．

[3]金野千里，梅谷征雄，大田忠，深田肇，山賀晋，池田美以子：対話型数値シミュレーションシステム：ビジュアル DEQSOL，情報処理学会誌第３３巻，第７号別刷，pp.929-942(1992)

[4]J.Rice:Solving Elliptic Probrem Using ELLPACK，Springer-Verlag，p.497(1984)

[5] G.O.Cook:ALPAL:A Tool for the Development for Large-Scale Simulation Codes，UCID-21482，Lawrence Livermore National Laboratory.(1998)

[6]G.O.Cook,JR:ALPAL，A PROGRAM TO GENERATOR PHYSICS SIMULATION CODES FROM NATURAL DESCRIPTIONS，International Journal of Modern Physics C Vol.1，No.1，pp.1-51(1990)

[7]P.Baras,J.Blum,J.C.Paumier,P.Witomski: EVE:An Object-Centered Knowledge-Based PDE Solver，IMACS(1992)

[8]S.Kawata，K.Iijima，C.Boonmee，Y.Manabe: Computer-assisted scientific computation/simulation-software-development system
-including a visualization system -
IFIP Transactions Vol.A-48，pp.145-153(1994)

[9]川田重夫，飯島邦彦，Choompol Boonmee，真鍋保彦：記号処理手法による数値シミュレーションコード開発支援システム，情報処理学会第３４回プログラミングシンポジウム，pp.61-71(1993.1)．

[10] 渋井俊昭 他： 有限要素法による物理シミュレーションプログラム生成支援に関する研究 ープログラムの可読性についての考察ー，第５０回情報処理学会全国大会（平成７年前期），pp.5-173 - 5-174(1994)．

[11] 渋井俊昭 他：数値ｼﾐｭﾚｰｼｮﾝﾌﾟﾛｸﾞﾗﾑ生成支援に関する研究（有限要素法ﾌﾟﾛｸﾞﾗﾑの生成支援），平成６年度電子情報通信学会信越支部大会，pp.345-346(1994)．

[12] 渋井俊昭 他：可読性を重視した有限要素法数値シミュレーションプログラムの生成支援に関する研究 ，第５１回情報処理学会全国大会(平成７年後期)，pp.5-159 - 5-160(1995)．