I -224

オブジェクト指向プログラミングによる有限要素解析及びプリプロセッサの応用開発

川田テクノシステム　正員　田中 成紀
法政大学工学部　正員　前田 重行
法政大学計算センター　正員　武田 洋

[1.はじめに] オブジェクト指向プログラミングは1970年に Smalltalk 言語として生まれ，1985年頃から C++ 言語の誕生に伴い今日に広く用いられるようになりつつある．最近，オブジェクト指向プログラミングは数値解析の分野においても適用されるようになってきている．オブジェクト指向の概念を取り入れた C++ 言語は，数値解析プログラミングに古くから用いられている，手続き型言語である FORTRAN 言語とは違い，「クラス」や「オブジェクト」に代表される新しい概念を取り入れている．このような概念は，プログラミング時において起こりうる，コードの煩雑化や様々なエラーの回避を容易にするという特徴を有する．本研究では有限要素解析に対し，上記の２つの言語を用いて，コーディング及び実行時間の比較を行う．また有限要素解析におけるプリプロセッサである，メッシュ・ジェネレイターとその可視化ツールを Windows 上で実行可能な Visual C++ 言語によって開発し，それらの有効性について検討する．

[2.クラスの設計] オブジェクト指向プログラミングは最初にクラスを作る事から始まる．例えば有限要素解析において，荷重，変位，ひずみと応力はクラスを用いて定義される[1]．有限要素解析プログラミングにおいて重要となるのは空間の参照点，すなわち節点である．それゆえ，有限要素を記述する最も基本的なクラスの１つは節点クラスとなる．この他にも要素，材料，変位境界条件，力学的境界条件，形状関数と数値積分クラスなどの基本クラスに加え，外部手続きとして，基本的な代数マトリックスを操作するベクトル，マトリックスとバンドシステムクラスがあるが，ここでは節点及び要素クラスについて説明する．

表１ 節点クラス

節点　(オブジェクトへ)	
番号 num　自由度 dof	
座標　x　変位　u	
入力	出力
変位の出力	変位の割当
自由度の割当	

[節点クラス] 節点クラスは節点番号，自由度，座標と変位を参照し，データファイルへの読み書き，番号付けされた節点に対し，計算された変位ベクトルや自由度の割当てを行うメソッドを持つ．節点クラスの構成は表１に示す．

表２ 要素クラス

要素　(オブジェクトへ)	
番号　num　形状関数　shapeFcns	
節点数　nodes　接線剛性マトリックス $\mathbf{K}_T$	
材料　material　応力マトリックス　$\mathbf{S}$	
数値積分 gauss　構成節点番号　numNodes	
入力	要素節点座標値の設定
出力	バンド幅の計算
応力の計算	剛性マトリックスの計算

[要素クラス] 要素クラスは要素節点数，材料，数値積分，形状関数，接線剛性マトリックス($\mathbf{K}_T$)，応力マトリックス($\mathbf{S}$)と要素構成節点番号を参照し，要素剛性を加えるための指標の組立，節点座標と変位の呼び出しとバンド幅の計算を行うメソッドを持つ．剛性マトリックスと応力の計算は形状関数と数値積分のクラスのサブメソッドを用いて実行する事が出来る．要素クラスの構成は表２に示す．

[3.オブジェクト指向有限要素解析] ここでは有限要素解析プログラミングに対し，オブジェクト指向プログラミングと手続き型プログラミングを比較することによってオブジェクト指向プログラミングの利点を説明する．またオブジェクト指向言語には C++ 言語を用い，手続き型言語にはC言語と FORTRAN 言語を用いる．まずコーディングの違いについて例を用いて説明する．接線剛性マトリックスの計算 $\mathbf{K}_T = \int_V \mathbf{B}^T\mathbf{D}_T\mathbf{B}dV$ は通常の手続き型プログラミングを用いると，リスト１のように記述できる．この場合 DO ループをいくつも用いて配列を操作しているのが読みとれるが，それ以上は前後関係を見なければわかりにくい．しかしこれをオブジェクト指向プログラミングを用いて記述するとリスト２のようになる．このようにオブジェクト指向プログラミングを用いると，その計算内容を理解するのにそれほどの労力を必要としない．これは C++ 言語が長い変数名を使えると言うことと，他のクラス（マトリックスクラス）が細かい作業をするため，表面が非常に簡潔なものになるためである．

リスト１ FORTRAN 言語による記述

```
      DO 140 I=1,3
      DO 140 J=1,8
      EB(I,J)=0.0D0
      DO 140 K=1,3
      EB(I,J)=EB(I,J)+EE(I,K)*BB(K,J)
  140 CONTINUE
      DO 180 I=1,8
      DO 180 J=1,8
      SKL(I,J)=0.0D0
      DO 160 K=1,3
      SKL(I,J)=SKL(I,J)+BB(K,I)*EB(K,J)
  160 CONTINUE
      SKL(I,J)=SKL(I,J)*WGHT
  180 CONTINUE
      DO 220 I=1,8
      DO 220 J=1,8
      SK(I,J)=SK(I,J)+SKL(I,J)
  220 CONTINUE
```

リスト２ C++ 言語による記述

```
stiffnessMatrix += trans(bb)
    * constitutiveMatrix * bb * wght;
```

次に計算速度の比較を行う．使用したモデルは図１に示す．また条件は表３に示し，結果は図２，３に示す．この結果から，計算速度はコンパイルオプションレベルにより異なり，３つの言語を用いた速度比較においては，今後異なったモデルやさらに大きな要素などでの検討を必要とする．

表３ システム構成

computer	SUN-4/2
processor	SPARC
operating system	UNIX4.1.1
central memory	32MB
Fortran compiler	SUN Fortran
C compiler	SUN C
C++ compiler	SUN C++

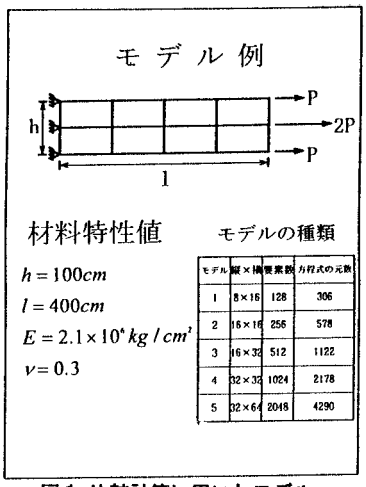

図１ 比較計算に用いたモデル

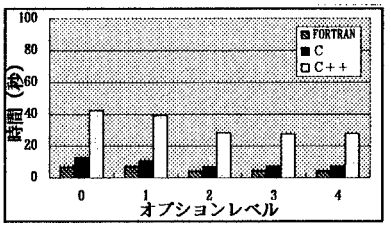

図２ 剛性マトリックス作成時間の比較

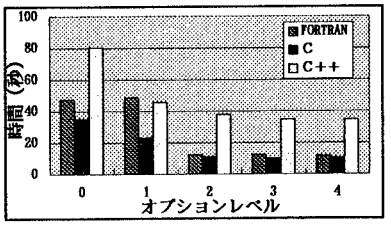

図３ 連立方程式の解法時間の比較

[4. プリプロセッサの可視化プログラミング] 次に有限要素解析において有益と言える，メッシュ・ジェネレイター及びその可視化ツールの開発を行う．特に本開発では可視化をする事を考慮し，オブジェクト指向言語に Visual C++ を用いる．Visual C++ は GUI を持ち Windows 上で実行可能なアプリケーションを作ることのできる強力な言語である．一方 Visual C++ におけるプログラミングは，基本的なクラスが前もって用意されており，開発者はユーザーインターフェイスの部分を特に気にすることなく，本来の計算部分のコーディングに専念できるという特徴がある．ここで開発したメッシュ・ジェネレイターは Martins と Marques [2] のメッシュ・ジェネレイターに基づき，オブジェクト指向プログラミングの概念を取り入れたものである．このメッシュ・ジェネレイターは図４に示すように，構造物をいくつかのサブブロックに分割することを必要とする．また図４で示した構造物をメッシュ・ジェネレイターで再分割し，可視化したものを図５に示す．データを与え図５に至るまでは，数回アイコンをクリックするだけである．このように Visual C++ を用いてコーディングを行うと，プログラミングの面からも，アプリケーションの操作性の面からも有益であると言える．

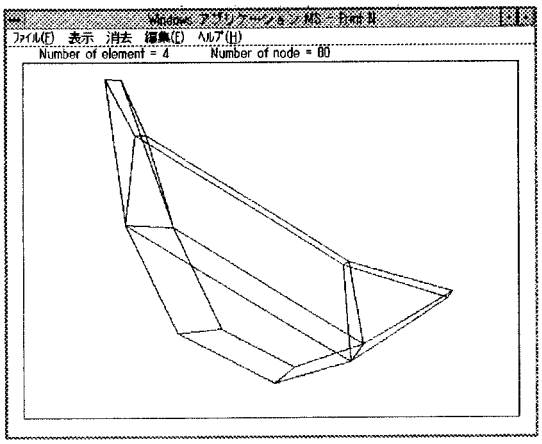

図４ 再分割前のダムのフレームモデル

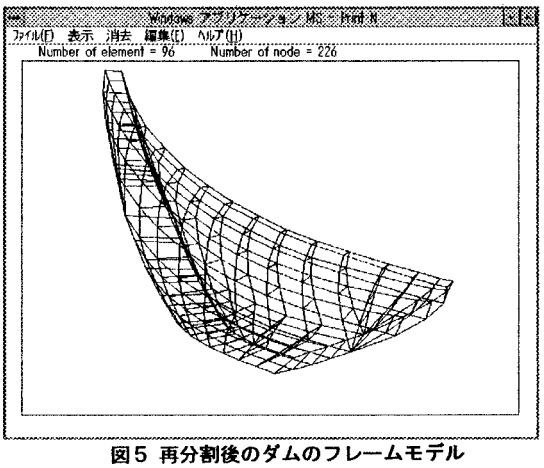

図５ 再分割後のダムのフレームモデル

[5. おわりに] 本研究では有限要素解析，メッシュ・ジェネレイター及びその可視化プログラムのコーディングに対しオブジェクト指向プログラミングを用い，その有効性に付いて検討した．その結果，有限要素プログラムの実行時間の比較においては，さらなる検討を必要とするが，コーディングに関しては有限要素解析などの煩雑になりやすいプログラムにおいても手続き言語に比べより簡単に記述でき，後のプログラムの修正や変更においてとても有効であると言う事が確認できた．

参考文献 1) Bruce W.R.Forde, Ricardo O.Foschi, Siegfried F.Stiemer, Object-oriented finite element analysis, Comput.Struct.Vol.34,No.3,pp.355-374, (1990). 2) P.A.F.Martins, M.J.M.Barata Marques, Model3-A three-dimensional mesh generator, Comput.Struct.Vol.42, No.4, pp.511-529, (1992).