計算機言語の文法に対する構造モデリングと複雑度評価†

遠藤聡志†† 大内東††

ソフトウェアの開発に携わる人が学び利用する最も重要なソフトウェアは計算機言語である．しかし，計算機言語に対しての分析・評価の一般的な方法は確立されていない．計算機言語を利用する最初の局面は言語の文法を習得することである．この局面における文法習得の難易度は，言語の学びやすさ (learnability) として，計算機言語評価の一側面を形成する．しかし，この基準は言語の表現力，拡張性などの能力的な基準に対する認識に比べて軽視されがちである．よって，計算機言語の文法に対して，その学習難易度の観点からの評価法を検討する必要がある．本論文では，以上の点をふまえて計算機言語の文法構造を示すモデル化の方法を提案する．モデル化には視覚的な利点から有向グラフを用いる．また，グラフモデル化された計算機言語の文法構造を，学びやすさの観点から評価する尺度の計算方法を提案する．さらに，提案された方法を用いて，ALGOL 60, C, PASCAL, FORTRAN の4つの計算機言語に対してグラフモデル化および評価尺度の適用を試みる．

1. はじめに

ソフトウェアの分野において，生産物や生産プロセスに対する分析法あるいは評価・比較の定量的な尺度が重要であるという認識がある[1),2)]．しかし，これらに対して基準となる分析・評価の尺度は確立されていない．さらに，ソフトウェアを利用している人々や，生産に携わる人々の認知的負担もほとんど考慮されていない．ソフトウェアの開発に携わる人々が最初に学び利用するソフトウェアは計算機言語である．計算機言語を利用する最初の局面は言語の文法を習得することである．この局面における文法習得の難易度は，言語の学びやすさ (learnability) として，計算機言語評価の一側面を形成する．しかし，この基準は言語の表現力，拡張性などの能力的な基準に対する認識に比べて軽視されがちである．よって，計算機言語の文法に対して，その学習難易度の観点からの評価法を検討する必要がある．

計算機言語の文法を記述する方法として広く知られているものに Backus-Naur-Form (以後 BNF とする) がある．BNF は，左辺項の構文単位を右辺項の構文単位群とメタ記号 '|', '::=' で定義する形式をとる．この記法は，文法を学ぶという局面では，左辺の構文単位を理解するためには右辺の構文単位を理解することが必要であるという内容を示している．したがって，BNF を構成する非終端記号の集合は，その隣接構造を調べることによって言語の習得課程を得ることができると考えられる．

本論文では，以上の点をふまえて計算機言語の文法構造を示すモデル化の方法を提案する．モデル化には視覚的な利点から有向グラフを用いる．また，グラフモデル化された計算機言語の文法構造を，学びやすさの観点から評価する尺度の計算方法を提案する．さらに，提案された方法を用いて，ALGOL 60, C, PASCAL, FORTRAN の4つの計算機言語に対してグラフモデル化および評価尺度の適用を試みる[3)~5)]．

本論文で用いるノード，アーク，経路，経路長等の用語はグラフ理論における一般的な定義，例えば文献6) 等に従う．

2. 構造モデリング

ALGOL 60 の文法記述のために考案された BNF は，計算機言語のシンタクスを明確に記述するために一般的に用いられる．特に BNF は，その記述上の性質から，構文単位間の定義と再帰的定義を明確に表現する点に優れている．しかしながら，シンタクス規則が多数になるため，BNF の非終端記号に対応する各項目の全体構造を把握することに適さない．計算機言語の文法の学びやすさを検討する上では，部分的なシンタクス規則を数多く並べる BNF はむしろ不便であり，構文単位の全体的な構造を知ることができるモデルが必要となる．

一般に多数の構成要素からなる複雑なシステムの構造を有向グラフモデルを用いて明らかにする手法に構造モデリングがある．この章では，BNF から得られる構文単位の構造を示すために，構造モデリングによる以下のグラフモデルを提案する．

† Structural Modeling and Complexity Evaluation for Syntax of High-Level Programming Languages by SATOSHI ENDOH and AZUMA OUCHI (Department of Information Engineering, Faculty of Engineering, Hokkaido University).

†† 北海道大学工学部情報工学科

2.1 シンタクス構造グラフ

任意の計算機言語に対し，そのシンタクスの BNF が与えられた場合を考える．BNF は左辺項の構文単位を右辺項の構文単位とメタ記号の組み合わせによって書き表す．BNF を有向グラフモデルに変換するために以下を定義する．

定義1：以下のノード集合Vとアーク集合Aによって定義される有向グラフ $G(V, A)$ を，計算機言語のシンタクス構造グラフと呼ぶ．

（1） ノード $v \in V$ は BNF の左辺項で定義される構文単位とする．

（2） 左辺項で構文単位＜a＞を定義する BNF の右辺項に構文単位＜b＞が出現するとき，グラフモデルではノードbからノードaへのアーク $\alpha_{ab} \in A$ が存在する．この関係を，bRa と書く．

定義1において，グラフモデルのノードは計算機言語の文法の主たる構成要素に対応する．また，アークは「構文単位aを理解するために構文単位bが必要である」という関係Rを表している．ここで，構文単位の現れ方，出現回数は考慮しない．前述のアークαの示す関係では，必要かそうでないかが本質的であり，どのような形で必要かはそれほど本質的でない．例えば，＜a＞::=＜b＞＜c＞|＜b＞＜d＞，＜a＞::=＜b＞|＜c＞|＜d＞，＜a＞::=＜b＞＜c＞＜d＞などは有向グラフでは同じ表現になる．

2.2 リダクション構造グラフ

シンタクス構造グラフG上で，2個以上の互いに到達可能 ($uRv \cap vRu$, $u, v \in V$) なノードから成る部分グラフは，サイクルを構成する．また，シンタクス構造グラフは，推移的に冗長なアークを持つ可能性がある．ここで推移的に冗長なアークとは，サイクルのない有向グラフからそのアークを除いた部分グラフが元のグラフの持つ可到達性を保つようなアークのことである．一般に，有向グラフの各ノード間の関係を簡潔に表現するために，サイクル部分を1つのノードで置き換え，すべての冗長アークを取り除く操作を行う．この操作は推移的リダクションと呼ばれる[7]（図1）．この操作によってできた有向グラフは，各ノード間の可到達関係を最も少ないアークで表現し，階層的である．

この操作をグラフモデルに適用するためには，グラフに設定した関係が推移性を有することが必要である．シンタクス構造グラフに設定したRは推移性を満たす．また，グラフモデルを文法の学習課程としてみた場合，推移的に冗長なアークは一度理解済みのノードを再び用いることを示しており，これらのアークを取り除いたモデルが本質的である．以下に，シンタクス構造グラフに推移的リダクションの操作を用いたグラフモデルを定義する．

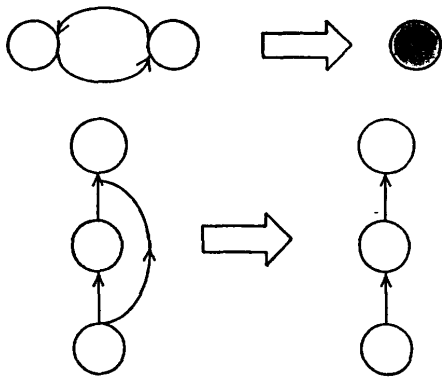

図1 推移的リダクション
Fig. 1 Transitive reduction.

定義2：次の操作によってできる有向グラフをシンタクス構造グラフのリダクション構造グラフと呼ぶ．

操作1 シンタクス構造グラフのサイクル部分を1つのノードで置き換える．このノードを縮約ノードと呼ぶ．

操作2 シンタクス構造グラフの推移的に冗長なアークを取り除く．

2.3 サイクル構造グラフ

シンタクス構造グラフやリダクション構造グラフでは，始点のノードが終点のノードを説明するという関係Rを設定した．サイクルを構成する要素は互いに他を説明するわけであり，この関係は“鶏と卵”で言われるように，順序だてて考えることのできない部分である．よって，サイクル部分は，理解，把握のむずかしい部分と考えることができる．これは，シンタクス構造グラフおよびリダクション構造グラフにおいてサイクル構造が特に注目すべき部分であることを示す．よって，前に定義した2つのグラフのサイクル部分を抽出したグラフモデルを定義する．

定義3：以下のノードとアークによって定義される有向グラフを，計算機言語のサイクル構造グラフと呼ぶ．

（1） ノードはシンタクス構造グラフの縮約ノードとする．

(2) アークはシンタクス構造グラフから得られる各縮約ノードの接続関係に従う.

(3) グラフのノードには,サイクル構造を構成する要素の数を与える.

サイクル構造の理解,把握の難しさは,その構成要素の数に負うところが大きい.このことから,サイクル構成要素数をノードの重みとして与える.

3. 複雑度

BNF から得られた有向グラフモデルの構造上の特徴から判断される計算機言語の文法習得の難易度を,計算機言語の複雑度と呼ぶことにする.グラフのすべての構成要素,すなわち BNF に現れるすべての構文単位を考慮した複雑度をリダクション構造グラフに対して提案する.また,とくに複雑度に影響すると思われるサイクル構造だけに着目した複雑度をサイクル構造グラフに対して提案する.

3.1 リダクション構造グラフを利用した複雑度

リダクション構造グラフに対して複雑度を計算するときに考慮すべき構造には,以下の三点が挙げられる.

(a) 入り次数

グラフのノードに着目した場合,入り次数はそのノードを定義するために必要な項目数を表す.よって,入り次数が高いノードに対応する項目は理解のむずかしい項目と言うことができる.

(b) サイクル

サイクル構造がグラフの上で理解のむずかしい構造であることは2.3節で述べたとおりである.

(c) 経路

グラフの経路は項目を理解していく順序を示しており,経路に対する考慮も必要である.

以下に,これらの点に着目した複雑度の計算法を定義する.

3.1.1 ノードに対する複雑度

リダクション構造グラフのすべてのノードに着目した複雑度を定義する.評価はノードの入り次数によって行う.ただし,リダクション構造グラフには縮約ノードと縮約ノードでないノードの2種類があり,縮約ノードに対しては,入り次数による評価に加えてサイクル部分としても評価する.

以下に複雑度を定義するため必要な記号を定める.

リダクション構造グラフのノードを有限集合 V' で,アークを有限集合 A' で定義する.

k_v をノード $v\in V'$ の入り次数とする.

c_v をノード $v\in V'$ のサイクル構成要素数とする.ただしサイクルを作らないノードの c_v は1とする.

定義4:(1)式で計算される値 CND を言語のノード複雑度と呼ぶ.

$$\mathrm{CND}=\sum_{v\in V'}\{k_v+c_v(c_v-1)\}, \qquad (1)$$

(1)式において第1項はノードの入り次数を示している.第2項はサイクルの構成要素が互いに可到達であり,サイクル構成要素1つを理解するためには他の構成要素すべてを理解しなければならないという性質を評価したもので,入り次数との整合をとるために c_v 個の要素からできる完全グラフのアーク数を計算している.また,サイクルを作らないノードの c_v を1としたことから,サイクルを作らないノードは入り次数のみによって評価される.

3.1.2 経路に対する複雑度

ノードに対する複雑度では,経路に関して考慮されていない.そこで経路に着目した複雑度を定義する.複雑度を計算するために扱う経路は,入り次数0で表される最下位レベルのノードから,出次数0で表される最上位レベルのノードまでのものとする.

経路に対する複雑度を定義するために必要な記号を定める.

言語のリダクション構造グラフの経路集合を P とする.また,ある経路 $p\in P$ 上のノードの集合を V_p とする.

定義5:(2)式で計算される値を,経路 $p\in P$ に対する複雑度とする.

$$\mathrm{CP}_p=\mathrm{Log}\left[\prod_{v\in V_p}(c_v!)\right], \qquad (2)$$

(2)式において c_v は(1)式と同様に経路 p 上のノードのサイクル構成要素数である.$c!$ は,縮約ノード内部の要素の並べ方を計算したものである.1つの経路上で c_v の積をとるのは,縮約ノード内の要素の並べ方の違いによって生じるその経路上の構文単位を理解する手順の場合の数である.

また(2)式では,c_v の増加に伴い値が指数的に増加することを抑えるため,常用対数をとる.

一般に,n 本の経路を持つリダクション構造グラフからは,n 通りの CP_p の値が得られる.この値から,言語の経路に着目した複雑度を1つに決める方法として,本論文では以下の2通りを定義する.

定義6:経路集合 P の中から最長の経路を持つ p を選びその CP_p の値を言語の経路の複雑度とす

る．CP_{long} と書く．また，グラフ中に最長経路が2本以上あるときは，それらの経路について(2)式を計算し，その中から最大値を選ぶことにする．

定義7：すべての経路 $p \in P$ に対して CP_p を計算し，その最大値を言語の経路の複雑度とする．CP_{max} と書く．

3.2 サイクル構造グラフの複雑度

リダクション構造グラフの経路に関する複雑度を計算するときに，サイクルを作らないノードの c_v を1とした．(2)式ではサイクルを作らないノードは $1!=1$ と計算されるため，このようなノードは複雑度に寄与しない．したがって，計算機言語の経路に関する複雑度は，サイクル構造グラフを対象として計算することができる．定義5～定義7によって定めた方法をサイクル構造グラフに適用した場合，CP_{max} はリダクション構造グラフに適用したときと同じ値として求められる．CP_{long} は，リダクション構造グラフの経路長がサイクル構造グラフにおいて保存されないため，必ずしも一致しない．しかし，(2)式ではサイクル構造が複雑度を決める最も重要な要因であり，CP_{long} の計算をサイクル構造グラフに対して行うことで，縮約ノードを1つも含まない経路が選択されることがなくなる．以上の点から，計算機言語の経路に関する複雑度はサイクル構造グラフに対して計算することにする．

4. 構造モデリングと複雑度計算のアルゴリズム

構造グラフ作成と複雑度計算のアルゴリズムを図2(i)に示す．各ステップは有向グラフに対応する2値行列の変換が主となる．行列変換の流れを図2(ii)に示す．

5. PASCAL の各構造グラフと複雑度計算

先のアルゴリズムにより得られる各構造グラフと複雑度を，PASCAL を例にとって示す．

PASCAL の BNF は，文献10) 等で与え

```
S1:BNFから隣接行列Aを求める.
{定義1(2)の関係をR, Aのij要素をa_ij とする. }
if iRj then a_ij = 1
        else a_ij = 0;
{Aの有向グラフ表現がｼﾝﾀｸｽ構造グラフとなる. }
S2:Aの可到達行列Mを求める.
{Iを単位行列, Aのサイズをnとする. }
M=(A+I)^n ;☆
☆隣接行列から可到達行列を計算する効率的な方法
は例えばWarshallのアルゴリズム8)がある.
S3:Mから縮約行列Cを作る.
for i=1 to n do
  for j=1 to n do
    if m_ij∧m_ji=1 then <Mのj行及びj列を削除>;
S4:Cからサイクル縮約行列C_sを作る.
{n'をCのサイズとする. }
for i=1 to n' do
  if <ﾉｰﾄﾞiが縮約ノードでない> then <Cのi行i列を削除>;
S5:Cからスケルトン行列Sを作る9).
for i=1 to n' do
  c_ii=0;
  for j=1 to n' do
    for k=1 to n' do
      if c_ik∧c_kj=1 then c_ij=0;
{Sのグラフ表現がリダクション構造グラフとなる.}
S6:C_sからサイクルスケルトン行列S_sを求める.
{n"をC_sのサイズとする. }
<C_sに対してS5と同様の計算を行なう>;
{S_sグラフ表現がサイクル構造グラフとなる. }
S7:CNDの計算
<行列S及び(1)式よりCNDを計算する>;
S8:CP_1,...,CP_m,...の計算
{CP_1,...}
<S_sから最長経路pを求める>;
<(2)式からCP_pを計算する>;
{CP_m,...}
<S_sの全てのpのCP_pを(2)から計算する>;
<CP_pの最大値を選ぶ>;
```

図2(i) 構造モデリングと複雑度計算のアルゴリズム
Fig. 2(i) Algorithm of structural modeling and complexity calculation.

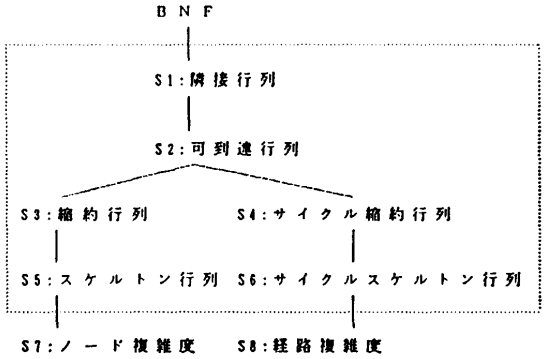

図2(ii) 各ステップに対応した行列変換の流れ
Fig. 2(ii) Flow of matrix transformation.

られる．ステップ1より得られるシンタクス構造グラフは**図3**となる．ステップ5により得られるリダクション構造グラフを**図4**に，ステップ6により得られるサイクル構造グラフを**図5**に示す．ステップ7，8に従い，複雑度を計算する．PASCAL のノード複雑度 CND は(1)式より，

$$CND=726$$

と計算される．

PASCAL のサイクル構造グラフには，2本の経路 P_1 と P_2 が存在する．経路長に着目した方法では，最初に P_1 が選択されて(2)式が計算される．したがって，

$$CP_{long}=CP_1\fallingdotseq 29$$

となる．

また最大値に着目した方法では経路 P_1 と P_2 の複雑度 CP_1 と CP_2 が計算され $CP_1>CP_2$ より

$$CP_{max}=CP_1\fallingdotseq 29$$

となる．

6. 4つの言語に対する構造解析と複雑度の結果

PASCAL のほかに，ALGOL 60, C, FORTRAN

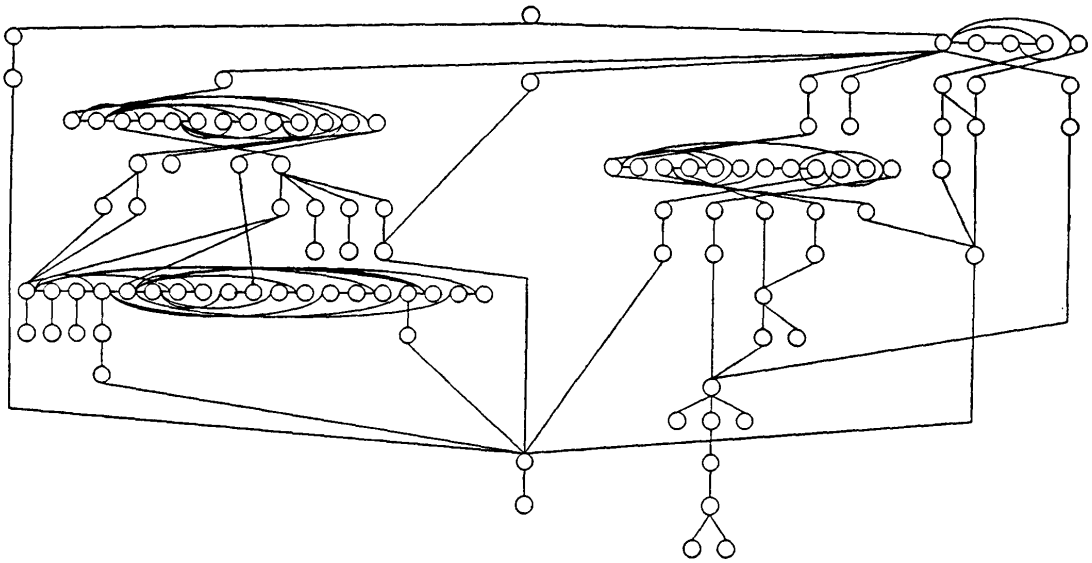

図3 PASCAL のシンタクス構造グラフ
Fig. 3 Syntax structural graph of PASCAL.

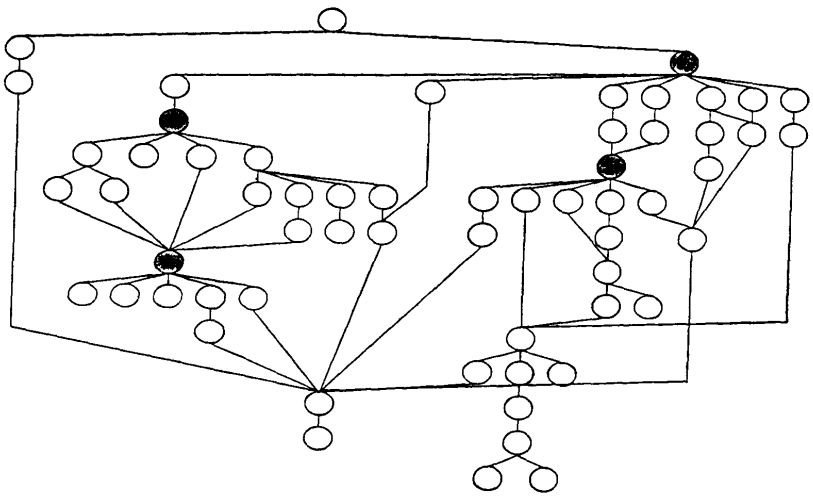

図4 PASCAL のリダクション構造グラフ
Fig. 4 Reduction structural graph of PASCAL.

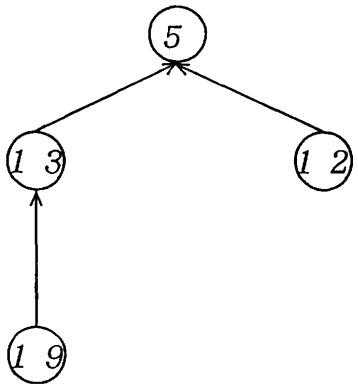

図 5 PASCAL のサイクル構造グラフ
Fig. 5 Cycle structural graph of PASCAL.

の3つの計算機言語に対して仕様書，参考書等[11)~14)]の BNF やそれに準ずる記述を利用してリダクション構造グラフおよびサイクル構造グラフを作成し，それらを利用して複雑度を計算した．

6.1 構造モデリングによる各グラフモデル

構造モデリングの結果として，ALGOL 60, FORTRAN, C のリダクション構造グラフを**図 6**（i）～（iii）に，サイクル構造グラフを**図 7**（i）～（iii）に示す．縮約ノードは，網掛けによって区別して表す．

6.2 複雑度の計算結果

リダクション構造グラフから求められる複雑度 CND とサイクル構造グラフから求められる複雑度 CP_{long}, CP_{max} を**表 1** に示す．表1では入り次数に関して，しきい値5を設けた値も示した．これは，人間が同時に扱える事物の数は5～9程度であるという意見[1)]に基づいて設定した値である．つまり入り次数4以下のノードは構造上理解しやすい形で表現されているとみなし，複雑度として計算しないことを表す．

まず CND に関しては，4つの言語の大小関係は

FORTRAN＜PASCAL＜C＜ALGOL 60

となった．CND の入り次数の制約を $k \geqq 5$ とした場合も大小関係に変化はない．

次に，CP_{long} および CP_{max} については，図6に示すとおりサイクル構造グラフが非常に簡単な構造になるため，それぞれの言語で選択される経路が CP_{long}, CP_{max} の場合で変わらない．したがって，いずれも同じ値となっている．また大小関係は，CP_{long}, CP_{max} とも，

FORTRAN＜C≒PASCAL＜ALGOL 60,

となって一致している．ここで，C と PASCAL に関しては，小数第1位までとって比べると，PASCAL＝29.0, C＝28.9 とごくわずか PASCAL が大きく，CND の結果とは逆転している．これは，双方のリダクション構造グラフにおいて，縮約ノード以外のノードに関してCの方がより入り次数の高いノードが多いことが影響している．したがって文法の学習を考えるときに，特に問題となるサイクル部分のCとPASCAL

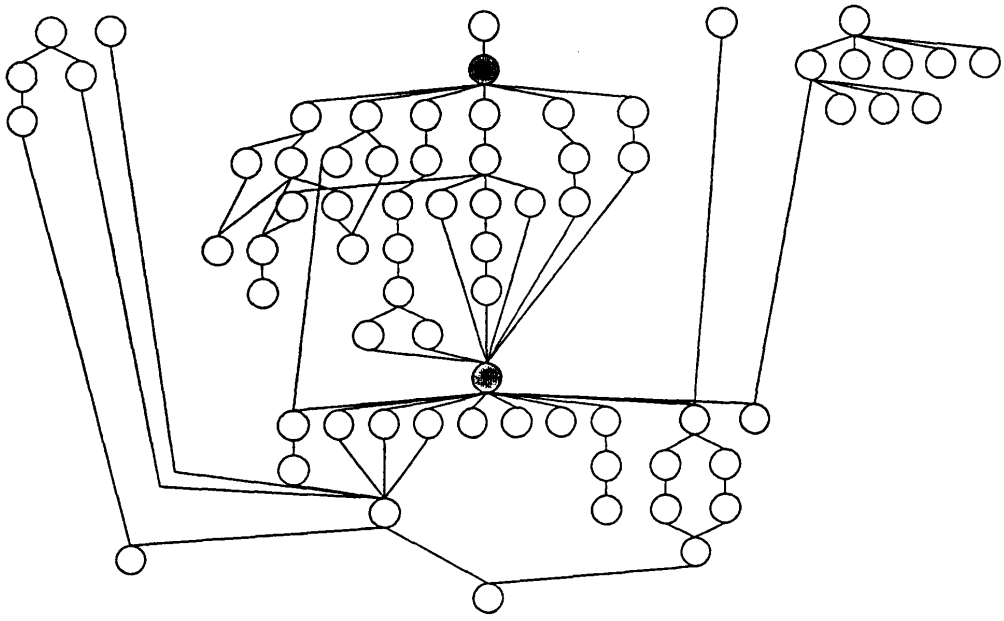

図 6(i) ALGOL 60 のリダクション構造グラフ
Fig. 6(i) Reduction structural graph of ALGOL 60.

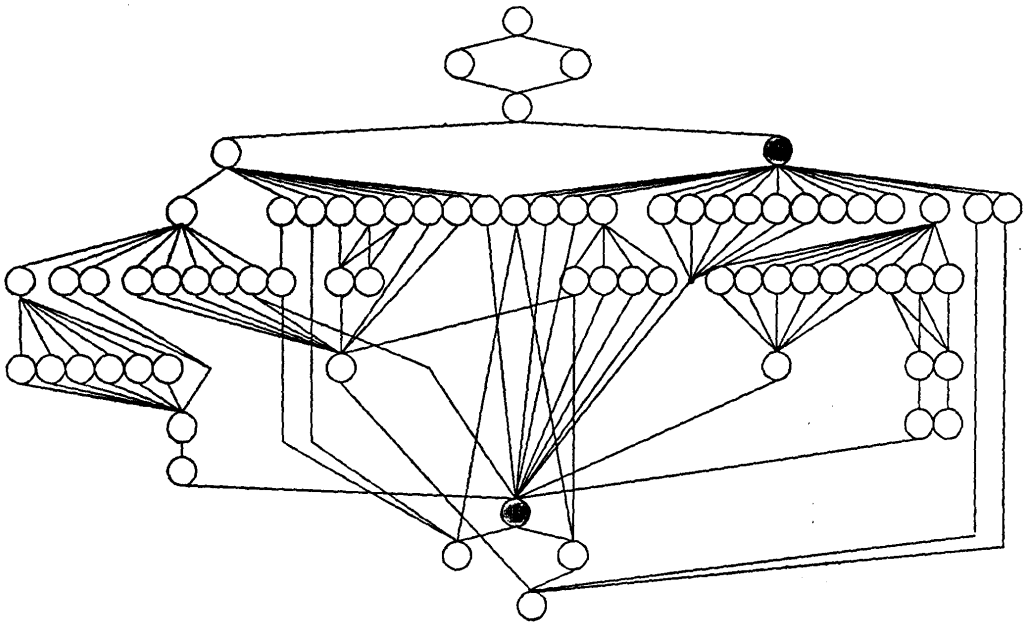

図 6(ⅱ) FORTRAN のリダクション構造グラフ
Fig. 6(ⅱ) Reduction structural graph of FORTRAN.

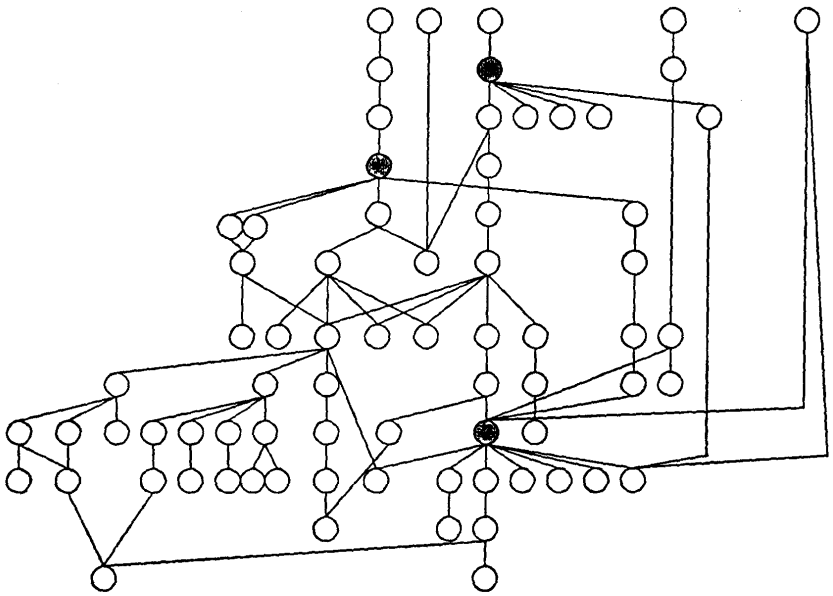

図 6(ⅲ) Cのリダクション構造グラフ
Fig. 6(ⅲ) Reduction structural graph of C.

での負担はほぼ等しいが，それ以外の部分でCの方の負担がより大きいと考えられる．

また，いずれの尺度でも複雑度最大となる ALGOL 60 のサイクル構成要素を見てみると，構文構成要素の中で比較的重要であると考えられる，<文>および<式>，<変数>が含まれ，それらの要素に関連性の高い要素が数多く存在する．このことが，ALGOL 60 の複雑度を高めている要因と考えられる．

図 7(i) ALGOL 60 のサイクル構造グラフ
Fig. 7(i) Cycle structural graph of ALGOL 60.

図 7(ii) FORTRAN のサイクル構造グラフ
Fig. 7(ii) Cycle structural graph of FORTRAN.

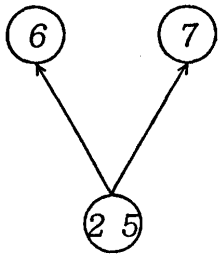

図 7(iii) Cのサイクル構造グラフ
Fig. 7(iii) Cycle structural graph of C.

表 1 FORTRAN, PASCAL, C, ALGOL 60 の各複雑度
Table 1 Complexities of computer languages (FORTRAN PASCAL C ALGOL 60).

	CND	CND ($k \geq 5$)	CP_{long}	CP_{max}
ALGOL 60	867	803	36	36
C	757	689	29	29
PASCAL	726	667	29	29
FORTRAN	338	264	13	13

7. おわりに

計算機言語のシンタクスに関して，有向グラフモデルを用いた構造モデリング法と，そのモデルを用いた複雑度評価法を提案した．また，それを適用して4つの言語の比較を行った．グラフモデルを用いた評価法は，言語のシンタクス全体を視覚的に捉えることに優れた方法である．計算機言語には様々な側面があり，この複雑度を持って計算機言語を是非するものではないが，言語の機能だけでなく利用する側の使いやすさは大いに考慮される部分であると考える．

また，構造モデリングによって得られる各有向グラフモデルは，複雑度評価の利用のほかに，計算機言語を順序だてて学習するための手順を示すチャート図としての利用が考えられる．チャート図上の各要素は明確な順序があることが望ましい．したがって，構造グラフをチャート図として利用する場合，サイクル部分の要素の順序づけが必要となる．有向グラフにおけるサイクル部分の解析は言語のグラフモデルに限らず重要な問題であり，言語のグラフモデルにおいて，サイクル内部の要素をどのように順序づけるかという問題も課題の1つである．

参考文献

1) Warfield, J. N.: Structural Analysis of a Computer Language, *The Seventeenth Southeastern Symposium of System Theory, IEEE*, pp. 229-234 (Mar. 1985).
2) Warfield, J. N.: A Complexity Metric for High-Level Software Languages, *International Conference on Systems, Man, and, Cybernetics, IEEE*, pp. 1-5 (Oct. 1987).
3) 大内：計算機言語の構造解析と複雑度評価，第37回情報処理学会全国大会論文集，4 Y-9 (1988).
4) 遠藤，大内：プログラミング言語の複雑度計算システムの試作，昭和63年電気関係学会北海道支部連合大会論文集，p. 306 (1988).
5) 遠藤，大内：計算機言語の構造解析と複雑度評価，第21回計測自動制御学会北海道支部学術講演会論文集，pp. 99-100 (1989).
6) 榎本：離散構造入門，p. 66，日本コンピュータ協会 (1980).
7) Aho, A. V., Garey, M. R. and Ullman, J. D.: The Transitive Reduction of a Directed Graph, *SIAM J. Comput.*, Vol. 1, No. 2, pp. 131-137 (1972).
8) Warshall, S.: A Theorem on Boolean Matrices, *J. ACM*, Vol. 9, No. 1, pp. 11-12 (1962).
9) Yelowwitz, L.: An Efficient Algorithm for Constructing Hierarchical Graphs, *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 6, No. 4, pp. 327-329 (1976).
10) 原田：PASCAL，培風館，東京 (1981).
11) Naur, P.: Revised Report on the Algorithmic Language ALGOL 60, *Comm. ACM*, Vol. 6, No. 8, pp. 1-17 (1963).
12) 大駒：FORTRAN 77，サイエンス社，東京 (1983).

13) 米田：C一言語とプログラミング一，産業図書，東京 (1982).
14) Borland International : Turbo C 2.0 リファレンスガイド，マイクロソフトウェアアソシエイツ，東京 (1988).

(平成2年5月21日受付)
(平成3年2月12日採録)

遠藤 聡志（正会員）

1964年生．1988年北海道大学工学部電気工学科卒業．1990年同大学院工学研究科電気工学専攻修士課程修了．北海道大学工学部情報工学科助手．システムの分析，設計，評価に興味を持つ．

大内 東（正会員）

昭和20年生．昭和49年北海道大学工学部大学院工学研究科博士課程修了．工学博士．北海道大学工学部情報工学科教授．システム情報工学，応用人工知能システム，医療システムの研究に従事．人工知能学会，電気学会，電子情報通信学会，計測自動制御学会，日本OR学会，医療情報学会，病院管理学会，IEEE-SMC各会員．