ソフトウェア工学　56－5
（１９８７．９．９）

プログラムの制御流れの複雑度と構造度の精度改善について

梁　海述　荒木　俊郎
大阪大学　基礎工学部

本論文では，プログラムの制御流れの複雑度（Control flow complexity)尺度と構造度(Structuredness)尺度の重要な特性と問題点について考察した．そして，従来のプログラムの複雑度尺度の欠点を補完し，それらを改善した２元複合尺度（C,K）（Hybrid measure）を提案した．

また，プログラムの構造率を測定する従来の構造度より，構造率をより正確に測定できる構造度（S≦C／(C+K)）を提案した．

さらに，従来の尺度と本論文の提案尺度との比較を行い，提案尺度の妥当性を評価した．

Refinement on Complexity and Structuredness of Program's Control Flow

Haesool YANG　Toshiro ARAKI

Department of Information and Computer Sciences,
Faculty of Engineering Science,Osaka University,
1-1, Machikaneyama, Toyonaka, Osaka, 560 JAPAN

In this paper, we describe some characteristics and issues in evaluating the complexity and structuredness of program's control flow. A new measure, called Hybrid measure (C,K), evaluates more accurately the complexity of a program than the previous one.

Also, we propose a new Structuredness measure (S≦C／(C+K)), which evaluate more precisely the rate of structuredness than the previous one.

Finally, we estimate the validity of the accuracy about the proposed measures by means of the sample programs.

１．まえがき

近年，ソフトウェアの多様な特性を測定，評価するための方法として，プログラムの複雑度（Program complexity）がソフトウェア工学分野で重要な問題として扱われてきた．

プログラムの複雑度は，プログラムのテスティング能力のスケジュール[3]，プログラミング時間の測定，プログラマのデバッギング能力，プログラムの非構造度（Unstructuredness）の測定及びソフトウェアの生産性評価，信頼性（Reliability），維持補修性（Maintainability）[7]，ソフトウェアの開発要求に対する予測，自動管理制御（Automatic management control）などの分野で適用されている．

また，プログラムの複雑度は，プログラムのサイズ，データの構造[4]，データの流れ，制御の流れ[1,2,6,10]などによって，評価されている．

本論文では，プログラムの複雑度を制御の流れによって評価する．プログラムの制御流れの複雑度は，ソフトウェアの維持補修性，理解性，信頼性，生産性を測定するのに効果的であり，プログラムの構造度とも密接な関係がある．

制御流れを用いてプログラムの複雑度を測定する従来の尺度として，McCabeの Cyclomatic数尺度[1]と，Woodwardのノット数（Knot count）尺度[2]がある．これらは，それぞれ分岐点（原始プログラム中において，IF文，CASE文などの制御の流れを分岐させる文をいう）とジャンプ（分岐点や無条件 GO TO文などによって生じる原始プログラム上での次の文以外への制御の流れ）によってプログラムの複雑度に直接的に影響を受ける．

本論文では，従来の制御流れの複雑度尺度の重要な特性と問題点を指摘し，それらの問題点を補完して，ソフトウェアの特性を客観的に，量的に表せる２元複合尺度(C,K) を提案する．また，プログラムに対する構造率の精度をより正確に示すために，２元複合尺度による構造度尺度S を提案する．

最後に，提案尺度の妥当性を立証するために，実際に使われているサンプルプログラムに対して適用し，従来の尺度との相関関係を分析する．

２．プログラムの複雑度と構造度尺度

２．１　従来のプログラムの複雑度尺度

(1)　Cyclomatic数尺度

制御流れの複雑度は，プログラム内の制御流れを基にして，プログラムが定量的にどれほど複雑であるかを測定することである．今までプログラムの複雑度を評価するために提案された尺度の中で，プログラムの論理的な複雑度を表す尺度は McCabeの Cyclomatic数だけである[9,10]．

一般的に，プログラムから直接そのプログラム中の制御の流れの全ての経路を効率的に数えるのは難しい．McCabeが提案した Cyclomatic数尺度は，与えられたプログラムに対する制御流れグラフから，そのグラフ上の全ての経路の数を線形組合せで測定する方法である．このとき，プログラムから以下のような制御流れグラフを作成し，そのグラフのもとで複雑度を測定する．

制御流れグラフ G=(V,E)は，次のようにして原始プログラムの文と制御の流れから得られる有向グラフである．各頂点 u∈V は，原始プログラムにおいて，シーケンシャル・コード部分を１つにまとめてブロック化したものである．また，各有向辺 (u,v)∈E は頂点u から頂点v への制御の流れを表す．但し，副プログラムの呼び出し文は１つの頂点とし，その呼び出し関係には有向辺を引かない．なお，頂点u において，辺(u,v) を出辺といい，v において，(u,v)を入辺という．

プログラムの制御流れの複雑度で使われている記号を次のように定義する．

i（∈V）：制御流れグラフの開始点．主プログラムや副プログラムの入口を表す．

t（∈V）：制御流れグラフの終了点．主プログラムや副プログラムの出口を表す．tにおいては出辺は存在しない．

e＝|E|：制御流れグラフの有向辺の数

n＝|V|：制御流れグラフの頂点の数

分岐点：出辺の数が２以上の頂点をいう．

経路(v0,v1,･･･,vm)：頂点v0 から頂点vm までの２つ以上の相異なる頂点vi（0≦i≦m)の系列である．但し，以下では v0＝vmとなるような経路，またはv0 が開始点，vm が終了点である経路だけを扱う．

連結成分（Connected component）：１個の開始点と１個の終了点を持ち，開始点から到達可能な頂点であり，かつその頂点から終了点へ到達可能であるような全ての頂点と，それらの頂点間の辺からなるグラフである．すなわち，連結成分には，開始点から到達可能でない頂点や，終了点へ到達可能でない頂点は含まれてない．主プログラムと副プログラムとの間には，辺がないことより，それらは異なる連結成分となる．

以下では，p 個の連結成分のみからなる制御流れグラフを考える．このとき，Cyclomatic数は，

C(G)＝e－n＋2p

で計算される．

また，制御流れグラフの連結成分数が１の場合，分

岐点の集合を Bとすると，

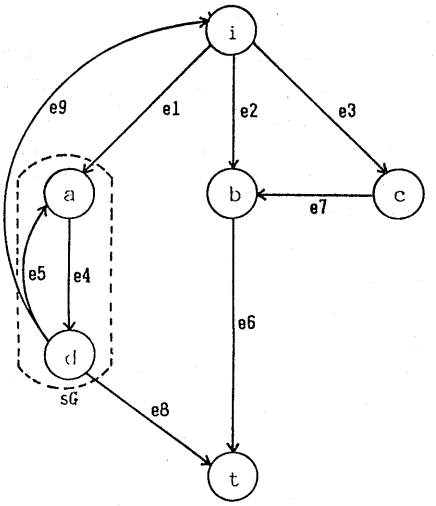
$$C(G)=\sum_{v\in B}(\text{vの出辺の数}-1)+1$$

が成立し，さらに，各頂点の出辺の数が高々２の場合，

C(G)＝分岐点の数＋１

の式が成立し，プログラムの複雑度を容易に計算できる[1,10]．例えば，シーケンシャル・コードのみの場合には，C(G)＝1となる．

プログラムの複雑度C(G)はプログラム中の文などの順序（Ordering）とは関係がない．図１は，制御流れグラフから McCabeの Cyclomatic数尺度の計算例である．また，図１のグラフに対する経路を表１に示す．



G＝({i,a,b,c,d,t},{e1,e2,e3,e4,e5,e6,e7,e8,e9})

i：開始点　　t：終了点

C(G)＝9－6＋2＝5　sG：structured subGraph

図１．Cyclomatic尺度での C(G)＝5の制御流れグラフ

表１．プログラムの制御流れグラフに対する経路

経路	e1	e2	e3	e4	e5	e6	e7	e8	e9
(iadt)	1	0	0	1	0	0	0	1	0
(ada)	0	0	0	1	1	0	0	0	0
(iadi)	1	0	0	1	0	0	0	0	1
(ibt)	0	1	0	0	0	1	0	0	0
(icbt)	0	0	1	0	0	1	1	0	0

(2)　ノット数尺度

Woodwardはプログラムのテキストにおいて，単純にジャンプの数を数えることよりは，ジャンプの実際的な位置を中心とする尺度を提案した[2]．

Woodwardの方法は，原始プログラムをブロック化し，そのブロック化されたプログラムから次のようにして制御流れグラフを得る．原始プログラムのブロックに相当する頂点を上から下の方に配置し，２頂点u,v 間に制御の流れが存在するとき，その２頂点間に有向辺(u,v)を頂点の列の左側に引く(例えば，図２から図３への変換)．このとき，ノット数尺度は，このようにして得られた制御流れグラフにおいて，辺の交差する数 K(すなわち，ノット数)を測定することで，プログラムの構造的な複雑度を測定する尺度である．例えば，図３はノット数４のプログラムであることが分かる．

このように，ノット数はプログラム内の制御流れに左右されるので，制御流れの構造的な複雑度とプログラムの非構造度の測定尺度として使用される．ノット数の測定には２つの方法がある．第一の方法は，プログラムに対する制御流れグラフの辺の重みが一定であるグラフに対する測定方法である．また，第二の方法は，プログラムに対する制御流れグラフの辺の重みが一定でないグラフに対する測定方法である．本論文でのノット数の測定方法は第一の方法を用いる．そして，この尺度はプログラム内の文の順序に依存するので，使用言語の影響を受けることがありうる[5]．

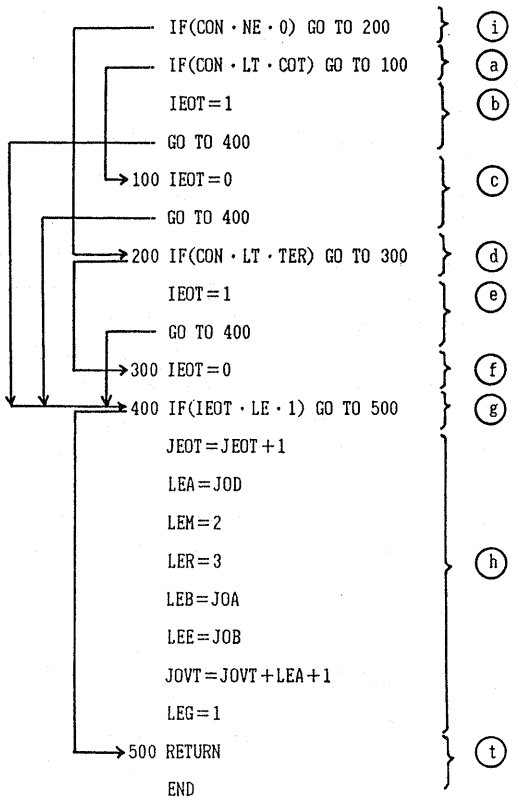


図２．原始プログラムのブロック化

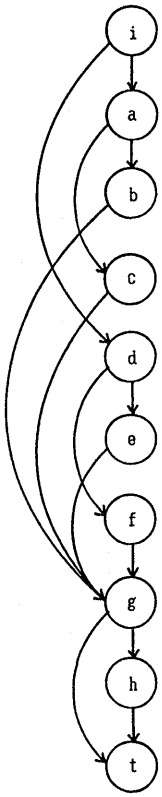


図3．図2の制御流れグラフ（ノット数=4）

2.2 従来のプログラムの構造度尺度

McCabeは，構造化部分グラフの数を用いてプログラム内の構造化精度を測定する構造度尺度eC(essential Complexity）を提案した．構造化部分グラフとは，構造化成分(Structured component;一般的に，SEQUENCE，IF THEN ELSE，DO UNTIL（または DO WHILE)等のような文）に相当する頂点とそれらの頂点間の辺からなるグラフである．このような構造化部分グラフの数をsGとする．

この尺度eC は，プログラムの制御流れグラフで求めた C(G)の値から，sG の数を引くことで求められ，次のように定義される．

eC＝C(G)－sG

また，eCの値は以下の操作を繰り返して得られた新しいグラフG 上での Cyclomatic数C(G)の 値と等しい．構造化された部分グラフ（例えば，IF文に相当する頂点から ELSE文に相当する頂点までの全ての頂点と，それらの頂点間の辺によるグラフ）を１つの頂点に縮小する．このとき，部分グラフの縮小は，最も内側にある部分グラフ（MIS；Most Inner Subgraph）から最も外側にある部分グラフ(MOS；Most Outer Subgraph)の方に行われる．

eC によって，プログラムの構造度を判断することができ，完全に構造化されたプログラムでは eC＝1になる．例えば，図１の制御流れグラフでは，C(G)の値が5，構造化された部分グラフ(点線部分が DO UNTIL 構造で，一つの頂点に縮小可能)の数sG が 1であるので，eC＝4になり，図１の制御流れグラフに対するプログラムは構造率の精度が低いことが分かる．

2.3 従来の尺度の問題点

McCabeの Cyclomatic数尺度と Woodwardのノット数尺度が持つ共通的な特徴は，制御流れグラフ内の制御流れの密度または相互関係から測定される制御流れの複雑度である．従って，このような特性のために発生するいくつかの問題点を含んでいる．これらはプログラムの制御構成から現れる問題点であって，制御流れによって多くの影響を受け，プログラムの複雑度と構造度に深刻な影響を与える．

(1) 複雑度尺度の問題点

a. McCabeの Cyclomatic数尺度

この尺度は一般のグラフに対する複雑度尺度として使われ，簡単に計算が可能であり，プログラムからすぐ求めることができるという特徴から受け入れられた尺度である．

しかし，この尺度には，次のような２つの問題点がある．第一の問題点は，同じ数の分岐点を持つプログラムは，分岐点の位置に関わらず常に同じ複雑度を持つことである（図４は全て C(G)＝6である）．次の問題点は，この尺度は制御流れを表す分岐点の数によって，複雑度が決定されることである（図４は全て分岐点の数が 5である）． 従って，このような問題点は，この尺度がプログラムの構造を充分に評価できないことを意味する．また，プログラムの構造を反映できない複雑度尺度はプログラムの理解度と信頼度に直接的な関係があるので，理解度の観点からも問題点が出てくる．

b. Woodwardのノット数尺度

この尺度は，プログラム内にある制御流れの位置に左右される尺度である．従って，制御流れが多くなるほど，プログラムの複雑度と構造度が増加されるので，プログラム内の構造度精度とプログラムの読み易さをよく表す複雑度尺度である．

この尺度には次のような問題点がある．

①図４の(1)から(6)までのグラフのように，IF文の

ような選択文による制御流れの複雑度はよく反映されるが，DO文のような反復文の場合，制御流れグラフでは反復回数を表現していないので，それによる複雑度はほどんど反映できない．

②図４の(1)のように，構造化プログラムに対しては，常にシーケンシャル・コード（Straight line program)と同一な複雑度を持つ．

③図４の(1)から(6)のように，プログラムのシーケンシャル・コードとか分岐点での制御流れの上下移動の方向によって現れる複雑度，すなわち論理的な複雑度を充分に表せない．

このような問題点は，この尺度がプログラム構造度をよく反映するが，一般的な複雑度尺度としては不適当であることを表している．

しかし，このような問題点は Cyclomatic尺度の長所と対応されるので，Cyclomatic尺度の問題点はこの尺度によって補完できる．

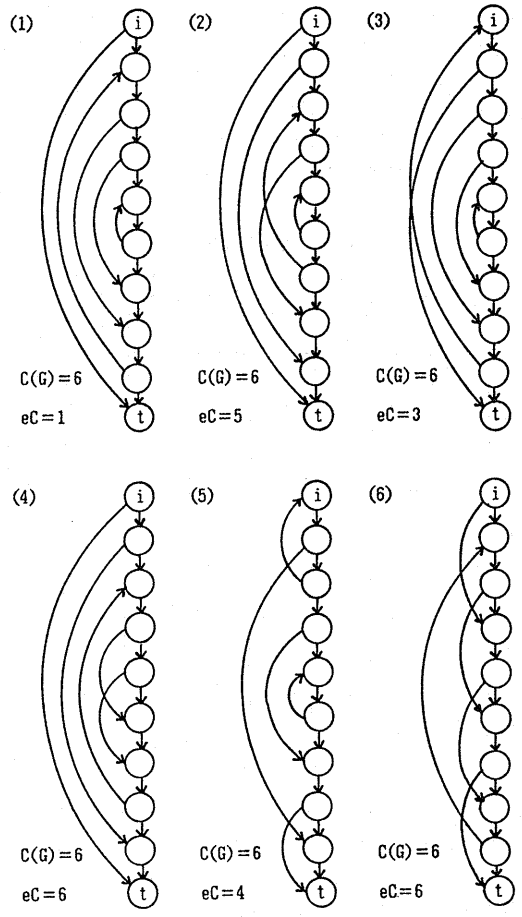


図４．C(G)＝6の制御流れグラフ

(2) 構造度尺度の問題点

McCabeの提案した構造度尺度eC は，制御流れグラフ内の構造化部分グラフを１つの頂点に縮小させることによって構成された新しいグラフ上で評価されるので，同じ個数の分岐点を持つグラフに対しても相異なる複雑度を表すことができる．従って，分岐点の配列による問題点が解決されると同時にプログラムの構造度精度も把握できる．

尺度eC には，次のような問題点がある．sG を測定する時，MISから MOSの方に進行するので，非構造化成分（Nonstructured component)の位置によって eC の値が異なる．

①非構造化成分が MOSにある場合

図４の(3)のように，MOSの内部の部分グラフは１つの頂点に縮小できるので，プログラムの構造度をよく表している．

②非構造化成分が MISにある場合

図４の(4)のような非構造化成分が存在するために，グラフはそれ以上縮小されず，C(G)＝eCになる．従って，このプログラムはほぼ構造化された形にもかかわらず，非構造化されたグラフとして評価される．

また，図４の(4)はほぼ構造化されたグラフで，(6)は非構造化されたグラフであるが，同じeC の値を持つ．しかし，この２つのグラフはプログラムの理解度と構造度の立場から見たとき，大きな差があるので，同じ構造率を持つグラフとして評価されない．

③非構造化成分が MISと MOSの間にある場合

図４の(2)で，非構造化成分の内部にある部分グラフは縮小されることによって，ある程度の構造度は表せるが，外側の部分は縮小できないので，②の場合に指摘された問題点を持つ．

従って，以上のようなeC の問題点を補完する必要がある．

３．２元複合尺度と構造度Sの提案

３．１　提案尺度

2.3節で指摘したような問題点，すなわち McCabeと Woodwardが提案した尺度で現れる問題点を解決できる新しい尺度が必要である．従って，これらの問題点を解決するために，Cyclomatic数とノット数とを両方持つ新しい２元複合尺度（C,K)を提案する．

２元複合尺度（C,K）において，C は，McCabe の Cyclomatic数で論理的な複雑度を表し，Kは Woodward のノット数で構造的な複雑度を表す．

そして，２元複合尺度による評価値の分布状況の概

略図を図５で示し，図中の矢印の方向にあるプログラムほど複雑度が低いと評価され，評価値の分布によってプログラム間の構造度と複雑度の比較ができる．すなわち，提案尺度によるプログラムの複雑度が　aの部分にあるときは，論理的な複雑度と構造的な複雑度が低く，bの部分にあるときは，論理的な複雑度が高く構造的な複雑度が低くなる．また，cの部分にあるときは，論理的な複雑度が低く構造的な複雑度が高くなり，dの部分にあるときは，論理的な複雑度と構造的な複雑度が高くなる．従って，aの方に近い分布を持つプログラムはよいプログラムであることが分かる．また，図４のグラフに対する２元複合尺度による評価値の分布を図６に示す．

さらに，提案した２元複合尺度の定義より，プログラムの構造率をより正確に表せる新しい構造度S を次のように提案する．

S≙C／(C+K)

ここで，構造度S （0<S≦1）は，プログラムが構造化されるほど（C の値は一定であるが，K の値が減少するので） 1に接近し，非構造化されるほど（C の値は一定であるが，K の値が増加するので） 0に接近することを表す．完全に構造化されたプログラムは１になる特性がある．構造度S は，プログラムの構造化された程度を表し，このS によって構造度を分かりやすく理解できる．

表２は，図４の(1)から(6)までのプログラムの制御流れグラフを用いて，従来の尺度と提案尺度との比較をまとめたものである． 表２と図５，図６を見ると，

区分＼部分	a	b	c	d
論理的な複雑度	低	高	低	高
構造的な複雑度	低	低	高	高

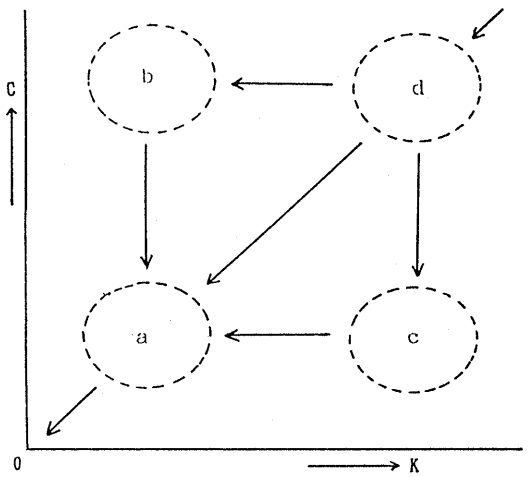


図５．２元複合尺度によるプログラムの複雑度

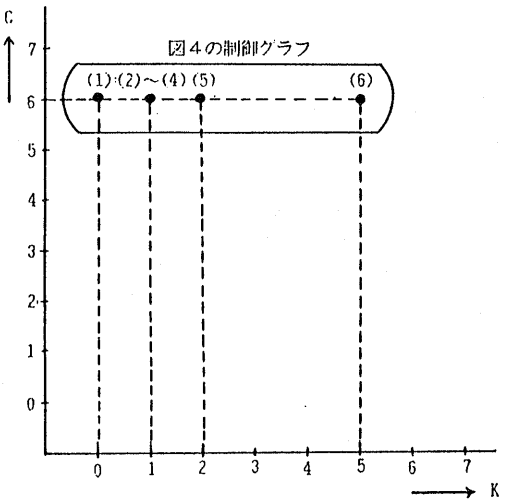


図６．２元複合尺度による分布図

提案尺度は，Cyclomatic数尺度とノット数尺度の特性が全て考慮されている．これは提案尺度がプログラムの構造的な複雑度と論理的な複雑度の表現だけでなく，従来の尺度で現れた問題点を補完しているためである．

そして，提案した構造度尺度S は，McCabeが提案した尺度eC の問題点も解決している．

表２．従来の尺度と提案尺度との比較

尺度＼適用例		図４の制御グラフ (1)	(2)	(3)	(4)	(5)	(6)
従来の尺度	C(G)	6	6	6	6	6	6
	Knot	0	1	1	1	2	5
	eC	1	5	3	6	4	6
提案尺度	２元複合尺度（C，K）	(6,0)	(6,1)	(6,1)	(6,1)	(6,2)	(6,5)
	S (≙C/(C+K))	1	0.86	0.86	0.86	0.75	0.55

また，構造度S の提案で，プログラムの構造度を定量的に評価することが可能になった．そして，提案した２元複合尺度を用いることで，図４の(4)と(6)の制御流れグラフから現れる問題点が解決でき，今まで２つの複雑度の比較が難しかった点を，提案した２元複合尺度として，解決することができた．

3.2 提案尺度の評価と考察

提案した尺度の妥当性を評価するために，最近数学及び統計分野で主に使用される IMSL(International Mathematical and Statistical Library)[8]からランダムに選んだ 110個のサンプルプログラムから提案尺度S，C(G)，eC，ノット数の値と，IF文，DO文，GO TO文，ジャンプなどの数を測定し，提案尺度S の値との相関関係の程度を分析し，相関係数として求めた（表3）．相関係数 r（$-1 \leqq r \leqq 1$）が，正（$r>0$）の場合は大きいほど，負（$r<0$）の場合は小さいほど，提案尺度S が従来の尺度および複雑度に影響を与える要素とほぼ比例し，相関関係が高くなる．このとき，相関関係が低いほど，0に接近し，$r=0$のとき，相関関係はないことを表す．　表3の①は全ての例が C(G)＝7のサンプルプログラムに対する相関関係を表し，②から⑥までは同一のC(G)の値を持つサンプルプログラムを求めることは難しいので，一定の範囲内の C(G)の値に対する相関関係を表す．このときの相関関係は，C(G)の値が一定のときよりは落ちるが，その変化の幅は大きくない．

表3．C(G)の範囲による Sとの相関関係

C(G)		①C(G)＝7の場合			②C(G)＝5～9の場合		
区分		eC	knot	S	eC	knot	S
尺度	eC	1.00000	0.97484	-0.99093	1.00000	0.91010	-0.92307
	knot	0.97484	1.00000	-0.97174	0.91010	1.00000	-0.95788
	S	-0.99093	-0.97174	1.00000	-0.92307	-0.95788	1.00000
要素	IF	0.69858	0.73776	-0.72295	0.69947	0.62116	-0.60074
	DO	-0.83279	-0.88993	0.84292	-0.70590	-0.72859	0.78594
	GOTO	0.59154	0.56144	-0.57706	0.73834	0.73607	-0.75619
	Jump	0.77388	0.77144	-0.77543	0.84256	0.79914	-0.80093

C(G)		③C(G)＝10～14の場合			④C(G)＝15～19の場合		
区分		eC	knot	S	eC	knot	S
尺度	eC	1.00000	0.70405	-0.84153	1.00000	0.73759	-0.83839
	knot	0.70405	1.00000	-0.90918	0.73759	1.00000	-0.94267
	S	-0.84153	-0.90918	1.00000	-0.83839	-0.94267	1.00000
要素	IF	0.49806	0.76233	-0.65652	0.76022	0.65349	-0.69941
	DO	-0.72833	-0.74225	0.80370	-0.77559	-0.71366	0.78370
	GOTO	0.26791	0.33275	-0.24962	0.49165	0.47864	-0.41825
	Jump	0.41541	0.59410	-0.49171	0.74277	0.66465	-0.66744

C(G)		⑤C(G)＝20～29の場合			⑥C(G)＝30～70の場合		
区分		eC	knot	S	eC	knot	S
尺度	eC	1.00000	0.82061	-0.86485	1.00000	0.89600	-0.87546
	knot	0.82061	1.00000	-0.92912	0.89600	1.00000	-0.95053
	S	-0.86485	-0.92912	1.00000	-0.87546	-0.95053	1.00000
要素	IF	0.71076	0.69026	-0.75304	0.88973	0.84719	-0.83751
	DO	-0.47582	-0.56270	0.65180	-0.33068	-0.38736	0.42428
	GOTO	0.26620	0.43106	-0.47000	0.27907	0.26442	-0.34930
	Jump	0.57085	0.65156	-0.70951	0.66242	0.62958	-0.67128

表4．全体サンプルに対する SとeC，ノットとの相関関係

C(G)	7	5～9	10～14	15～19	20～29	30～70
サンプル数	10	30	20	20	20	20
尺度＼行数	15～30	10～45	46～55	56～80	81～115	116～350
eC	-0.9909	-0.9231	-0.8415	-0.8384	-0.8648	-0.8755
knot	-0.9717	-0.9579	-0.9092	-0.9427	-0.9291	-0.9505

そして，表4は全ての適用サンプルプログラムの一定の範囲内のC(G)の値に対して，提案構造度S の値とeC，ノット数の値との相関関係をまとめたものである．

従って，表3と表4のように，C(G)が一定のとき，構造度S はノット数，eCと0.95以上の相関関係が現れる（表3の1～3行と表4の1行参照）．これは，S が eC の属性を充分に持つことを意味する．

また，S はプログラムの複雑度に直接的な影響を与える要素の中で，制御の分岐が生じる IF文や　DO文，GO TO 文などのジャンプの数とも相関関係があることを示している（表3の4～7行参照）．例えば，表3の②において，　GO TO文とS は高い相関関係にあることを表しているが，③においては，低い相関関係にあることを表している．

一方，表3の2行3列を見ると，　Sはノット数とほぼ逆比例関係が成り立っていることが分かる．このような逆比例関係より，ノット数を減らすことによって，構造度S を改善できることが分かる．　また，図7は，従来の尺度と要素が，構造度S に影響を与える順序を表している．　すなわち，その順序は，ノット数，eC，IF文，Jump，DO文，GO TO文の順である．　よって，ノット数を減らすことは重要であり，ノット数はプログラム中の文の順序によるので，その順序を再構成することで，ノット数を減らすことができる．

その方法としては，プログラムテキストからノット数を増加させる要因になる頂点と非構造化成分を MOS に移すこと，あるいは論理の流れを変えることがあり，それによって，ノット数が少ないグラフに変更できる．例えば，図8の(1)を(2)に再構成することで，ノット数が少ないグラフに変更できる．

以上に示したように，2元複合尺度（C,K)と構造度S は従来の尺度の属性を充分に持ちながら，それらが持つ問題点を解決できるので，複雑度と構造度の精度が向上されたことが分かる．従って，プログラムの構

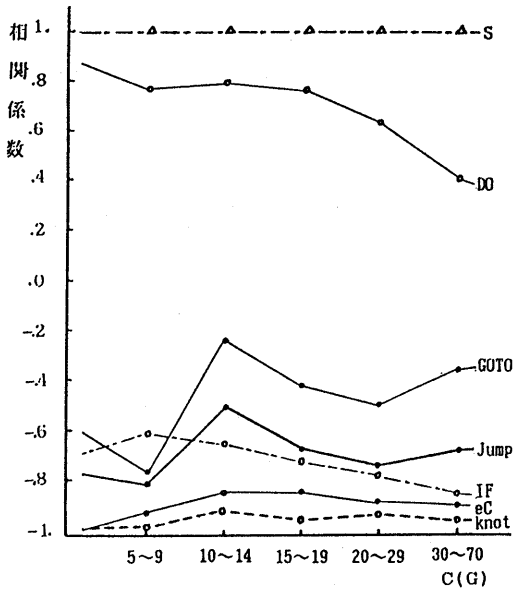


図７．Sと従来尺度および要素との関係

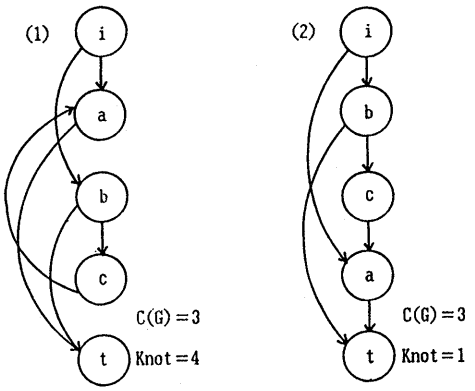


図８．制御流れグラフにおいて，ノット数を減らす例

造的な複雑度と論理的な複雑度がよく反映されていることを示している．

４．むすび

本論文の研究結果として，

①プログラムの複雑性を評価する２元複合尺度（C，K）の提案と，

②２元複合尺度が，従来の尺度より妥当な方法であることを示し，

③従来の構造度の essential Complexity尺度eC より，提案した構造度S （≒C／(C+K)）がプログラムの構造化をより正確に評価することを示し，

④構造度S の妥当性を示すために，従来の構造度尺度とプログラムの構造に影響を与える要素との相関関係を分析した．

また，今後の課題としては，データ流れ情報によるソフトウェアの複雑度問題とプログラムに影響を与える要素（プログラムのサイズ，データ構造，データの流れ，制御の流れ等）を考慮した複雑度の測定方法に関する研究などがある．

［謝辞］

本研究において，適切な御指導と御助言を賜りました都倉信樹教授に深謝いたします．

特に，第一著者は，韓国の江原大学校から研究のために来日した研究員で，いろいろな御指導御討論を頂きました萩原兼一助教授，辻野嘉宏助手，増澤利光助手をはじめ都倉研究室の皆様に深く感謝いたします．

参考文献

[1]T.J.McCabe :" A Complexity Measure", IEEE Trans. Software Eng., Vol.SE-2,No.4, pp.308-320 (December 1976).

[2]M.R.Woodward, M.A.Hennell and D.Hedley :"A Measure of Control Flow Complexity in Program Text",IEEE Trans. Software Eng.,Vol. SE-4, No.1, pp.45-50 (January 1979).

[3]M.R.Paige :" A Metric for Software Test Planning", in Proc. COMPSAC 80, pp.499-504 (1980).

[4]V.R.Basili :" Product Metrics", Tutorial on Models and Metrics for Software Management and Engeering, IEEE Computer Society Press, pp.214-217 (1980).

[5]A.L.Baker and S.N.Zweben :" A Comparison of Measures of Control Flow Complexity", IEEE Trans. Software Eng., Vol.SE-6, No.6,pp.506-512 (November 1980).

[6]W.Harrison and K. Magel :" A Complexity Measure Based on Nesting Level", ACM SIGPLAN Notices,Vol.25,No.3, pp.63-74 (March 1981).

[7]W.Harrison, K.Magel, R.Kluczny and A.Delock: " Applying Software Complexity Metrics to Program Maintenance", IEEE Computer Society, Vol.15,No.9, pp.65-79 (September 1982).

[8]IMSL : Problem-solving Software System, MATH/STAT/SFUN Library Reference Manual,IMSL Inc. (1985).

[9]C.Jones :" Programming Productivity", McGraw-Hill Series in Software Engineering and Technology, New York, pp.70-71 (1986).

[10]S.Yau and J.P.Tsai :" A Survey of Software Design Techniques",IEEE Trans. Software Eng., Vol.SE-2,No.6, pp.718-719 (June 1986).