3N-3

プログラム構造複雑度ＰＳＣの検討

新名　秀章†　　岸本　頼紀‡　　浜名　隆広†　　佐藤　匡正*

島根大学総合理工学研究科†　　名古屋工業大学大学院電気情報工学専攻‡　　島根大学総合理工学部*

１．序論

ソフトウェア品質の重要な項目のひとつにプログラムの複雑さがある．これを示す目的で，プログラムの一部の特性に着目した構造尺度がいくつか提案されている．この代表的な評価尺度として，McCabeはCyclomatic数によるMcCabe複雑度(MC値)[1]を提案した．しかし，この評価尺度は制御構造の分岐のみに着目するため，繰返しで生じる複雑度については考慮されていない．プログラムの複雑さとしては繰返しと分岐の違いを反映し，規模によって生じる複雑さも考慮することが求められる．この目的として幾つかの複雑度を組合せて判断するハイブリッド型手法も提案されている．しかし，明確な複雑度を示すためにはひとつの値で全体の複雑度を示す手法が求められる．

プログラムは関数など共通化の目的で複数の処理経路を内包する可能性がある．したがって，プログラムには複数の処理経路が絡みあうことによる複雑さも発生する．そこでこれを分離し，それぞれの論理構造の複雑度の合計をプログラム全体の複雑度とすれば，規模と論理構造の両方の複雑度を示すことができる．

変数に着目すれば，プログラムの複雑さ要因として変数は命名，型，変数を決定する処理の集合（論理決定構造），変数が影響を与えている処理の集合（影響範囲構造）の４つの複雑さ要因をもつ．命名は官能的なため定量を計測することは困難だから無視する．型は単一型と複合型があり，複合型を単一に分離すれば差異を吸収できる．論理決定構造と影響範囲構造はそれぞれ論理構造としての複雑さをもつ．繰返しに重みを与えれば，繰返しと選択の違いを反映した論理構造の複雑度が得られる．これらの複雑度を計測し，その合計が変数によって分離できる論理構造の複雑度として考えることができる．プログラム中の全ての変数におけるこの総和を得れば，論理構造の複雑度と変数の規模を反映したプログラム全体の複雑度を得ることができる．これをプログラム構造複雑度（PSC:Program Structure Complexity）と呼ぶ．

本論文では，プログラムの複雑度として PSC とその計測方法を提案し，実際のＣ言語の例における PSC と MC 値の比較により本提案の有効性を確認する．

２．ＰＳＣ

２．1考え方

プログラムの複雑さとして，内包されている複数の論理構造を分離し，それぞれの複雑度の総計を考える．この分離方法として変数に着目する．変数はこれに関わる論理構造として論理決定構造と影響範囲構造をもつ．この複雑度の合計をその変数に関わる部分論理構造の複雑度とする．論理構造の複雑度として処理経路に着目する．MC 値に代表される論理構造の複雑さとしてはその経路数が指標とされる．そこで，変数におけるこの 2 面の論理構造の処理経路数を計測し，合計値をその変数における論理構造の複雑さとする．また繰返しが存在すると経路数を得ることが困難となる．プログラム中の繰返しは繰返し回数が明確ならば連接で置換えることができる．そこで，繰返しを連接に置換えることで重み付けした処理経路数を得る．これにより，それぞれの変数について，それぞれの論理構造がもつ複雑度を，繰返しを考慮した処理経路数として得ることができる．この合計値をプログラム全体の複雑度とする．

２．2複合型変数の分離

プログラム変数には，単一のデータのみを扱う単一型と単一のデータの連続とそのアドレスを指定する複合型がある．例えば，C 言語の int 型は単一型であり，配列やポインタ，構造体は複合型である．この型の違いもプログラムの複雑さに影響を与える．この要因を吸収するために複合型を複数の単一型に分離する．複合型を単一型複合型の場合はデータ列とそのアドレスを別の変数として考えれば，全て単一型データとして考えることができる．例えば C 言語におけるポインタ変数*strData については&strData と*strData をそれぞれ単一型データとして考える．

２．3論理決定構造の複雑度

変数を決定する論理決定構造の複雑度として，その変数に対して設定を行う処理の集合に着目する．この集合はその変数を決定する論理の複雑さを反映し，プログラムとしての論理を構成する．そこで，論理決定構造の複雑度として，プログラムの処理経路に着目する．プログラムの論理を分析する手法としてプログラム構造形式化手法が提案[2]されている．これは，プログラムの論理構造を非決定性の正規表現で記述し，正規表現式上でプログラムの論理構成について議論できる．プログラム構造形式化手法では，正規表現中で議論に関係のない変数を空要素（ε）で置換えても論理分析ができる．したがって，変数について論理決定構造と関係のない処理をεに置換えれば，その変数を決定する処理の集合の論理構成が正規表現で得られる．これをプログラム構造スライシングと呼ぶ．そこで，処理経路として変数を設定している処理の集合に関係のない処理を，εで置換え，この正規表現において式を展開する．プログラムの論理構造の経路数は連接構造の項の数として現れるから，この項の数を複雑度とする．

また，プログラムは繰返しの打切りや関数の戻りといった強制制御や継続繰返しをもつ場合がある．プログラム構造形式化手法では，これらをもつ構造をもたない構造に変換する変換公式が提案されているから，これを適用すれば容易に正規表現式の項数を求めることができる．

２．4影響範囲構造の複雑度

影響範囲構造の複雑度として，その変数を参照している処理の集合を考える．この集合は論理構造をもつから，処理経路と同様に参照されている処理集合でプログラム構造スライシングをした結果の式に対して，その項数を計測し複雑度とする．

２．5繰返しの考え方

論理構造の複雑さとして繰返しに重みを与える．プログラム

Considerations of Program Structure Complexity PSC

†SHINMYOU Hideaki

†HAMANA Takahiro

†Faculty of Science and Engineering Graduate School

‡KISHIMOTO Yorinori

‡Electrical and Computer Engineering Nagoya Institute of Technology

*SATOU Tadamasa

*Faculty of Science and Engineering Shimane University

表1　PSC値とMC値計測結果

プログラム名	プログラム構造式	PSC値	MC値
Read_Command_line	$ijka^*(a(b+\varepsilon)(cde+\varepsilon)(fg^*))^*(e+\varepsilon)h$	87	10
Scan	$a(b\#+\varepsilon)((c(\varepsilon)+\varepsilon))^{\infty}(\varepsilon)+\varepsilon)d(c(\varepsilon)+\varepsilon)d)^*ef)^{\infty}g$	19	7
Sample1	$ab(cdb)^*(cded+\varepsilon)(ab(cdb)^*(cded+\varepsilon))^*$	10	7
Sample2	$a(bc)^*a(d((efg+\varepsilon)h)^*c)^*a(c)^*$	20	7
Sample3	$a(b+c)((d+e)+(f+g)+\varepsilon)hi(j+c)k$	8	7

構造では繰返し回数や命題は機構として取り除かれているため，連接回数を特定することができない．そこで，繰返しは最低二回繰返すと仮定し，この繰返し構造を連接構造として展開することでプログラム構造の項数を計測する．例えば$(a+b)^*$は$(a+b)(a+b)$と展開する.これにより，繰返しに重みをつけた論理構造の複雑度を得ることができる.

２．６関数の考え方

関数は引数を参照として扱う．見かけ上関数において設定と参照が分かり難い場合がある．パラメータでは明に指定していないが，大域変数に対して設定や参照を行う場合は，その対象の変数を設定もしくは参照しているものとして扱う.

２．７事物基軸型(Object-Oriented)言語への適用

プログラミング言語によっては事物基軸型言語もある．プログラム構造形式化手法では，この概念の言語にも適用できることが提案されている[3]．そこで，この考えに基づき，クラス・メソッド内部にのみ着目すれば，事物基軸型言語についてもPSCを計測できる.

また，メッセージの授受によって発生するオブジェクト同士の関係は，２．６関数の考え方に基づけば，メッセージを関数のパラメータと考えればオブジェクトの振舞いに対応した変数をそれぞれ設定，参照と考えることができる.

３．手順

本計測方法は次に示す手順から成る.

手順１：プログラム構造形式化

プログラム・ソースにプログラム構造形式化手法を適用し，プログラム構造式を得る．強制制御をもつものはこれをもたないものに変換しておく.

手順２：プログラム構造スライシング

全てのプログラム変数について，その変数に対して設定と参照それぞれを行う処理のみに着目し，それ以外のプログラム構造の変数をεで置換えた式を得る.

手順３：PSC値計測

手順２で得られた２つの部分構造式から変数に関わる部分構造の複雑度を計測する．このとき，εも１つの経路として計測する.

PSCの計測方法は次の通りである.

ー選択構造

選択構造は可能な限り展開し，連接によって構成される項の数を計測する.

ー繰返し構造

繰返し構造は全て２．５に基づき展開する.

ー入れ子構造

入れ子構造はこれを展開しておく.

４．適用結果

本手法を実際のＣ言語で記述されたプログラム例５題に適用した．それぞれのプログラムにおける基となるプログラム構造式とPSC値，MC値を表１に示す.

本例では４つのプログラムについて，それぞれのMC値が７で同じだったがPSC値は異なっていた.

５．考察

５．１規模要因の反映

Scanより Sample1 の方がPSC値は低かった．これは，Scanの方が用いられている変数が少なく，変数の規模によって発生する処理経路が少ないことに起因している．MC値では反映されていないがPSC値はこの複雑さを反映していることが確認できた.

５．２論理構造要因の反映

Sample1 と Sample2 に着目する．Sample1 より Sample2 の方が繰返しの入れ子構造になっているため，官能的には複雑である．MC値では反映されていないがPSC値は繰返しによる複雑さを反映していることが確認できた.

５．３繰返しの有無の反映

Sample2 と Sample3 に着目する．両者はMC値は同一だが，前者は繰返しを用い，後者は繰返しを用いていない．選択と繰返しでは繰返し構造の方が官能的に複雑である．PSC値も論理構造による複雑さを反映していることが確認できた.

６．結論

プログラムの複雑度として，プログラム変数に関わる論理構造として論理決定構造と影響範囲構造を定義し，プログラム構造スライシングによって部分構造式を求め，繰返しを連接で展開することで繰返しに重みを付けた部分構造式の項数の総和を複雑度とする，プログラム構造複雑度(PSC)を提案し，この計測方法について論じた.

本手法を実際のプログラムに適用した結果，本例においてMC値では示すことのできない複雑さとして，変数の規模や論理構造，繰返しの有無による複雑さの違いを，PSC値として示すことができた.

参考文献

[1]McCabe,T."A software complexity measure" IEEE Transactions on software engineering, SE2 (4):308-320

[2]Yorinori Kishimoto, Takako Itou, Tadamasa Satou:"Program Structure Formlizing Technology for Static Analysis", WSEAS TRANSACTIONS on SYSTMS, Issue 4,volume 3,pp1691-1698

[3]佐藤匡正,岸本頼紀:" 事物基軸言語へのプログラム構造形式化の適用法",電子情報通信学会技術研究報告．SS，ソフトウェアサイエンス，Vol．102 Num．617 pp.37-42