ソフトウェア工学　46－15
（１９８６．２．７）

変数の設定参照関係に基づくモジュール間インタフェース決定法

佐藤　匡正（ＮＴＴ　情報通信処理研究所）

1.序

ソースコードからモジュール間の連絡手段（インタフェース）を簡便に解析する手法を考える。連絡手段には値と制御（タイミング）によるものとがある。ここでは、値による連絡手段に注目する。この方法の応用としてプログラム資産の活用に繋がる。例えば，この解析方法によってそれぞれのモジュール毎の連絡手段を把握する．把握できた連絡手段の整合によってモジュールを組み合わせたり，あるいはプログラムの構成要素であるモジュールを別のモジュールに置換するなどして別のプログラムを作成することができる。連絡手段を整合させるには、例えば、他へ値を授ける必要のないものはダミーとしたり、他から値を受け取る必要のないものは疑似的に値が与えられるようなアダプタ付きのソケットを考え、モジュール同士はこれを介して接続するような仕掛けを拵えればよい。

これまで、モジュール間の連絡手段については古くから関心を集めている。これまでの研究を大別すると、連絡手段に概念を与えている設計寄りのものと、実現した結果を分析分類するための管理寄りのものがある。前者には言語処理系における手続き呼び出しの引数機構の［名称、値、参照］による概念整理[1]、抽象データ型[2]や情報隠蔽[3]の概念の提案である。後者には評価基準としての「モジュール間連絡強度」の提案[4]がある。連絡手段がどのように実現されているかを解析する手法を確立させることは　後者の重要なテーマのひとつである。既存のモジュール数は膨大なものであり人手に依らずに解析できれば再利用も促進される。しかし、現状ではこのような解析手法は提案されていない。

連絡手段の解析は、原理上からは動的にデータ流を把握することによって実現できる。ところが、この動的データ流をソースコードの静的解析のみから得るのは困難である。近似的な結果しか得られない。しかもこのデータ流解析のためには繁雑な処理系を必要とする。固より近似的な結果しか得られないのであればこのような処理系を必要としない簡潔な解析手法があれば都合よい。この一手法としてデータ流解析の替わりに変数の設定，参照関係に着目するという着想を得た．そもそも値の連絡手段は変数を介して実現されているので、連絡手段を把握するには、連絡に用いられている変数名とモジュールからみての授受関係が識別できればよい。この授受関係が変数の設定参照に基づいて決定できれば，データ流解析は不要になる筈である．変数の設定，参照関係はクロスリファレンスリスト（以下，X表）に出力されている。よく知られているように、X表にはソースプログラムの全ての変数についてそれぞれごとに定義箇所、値参照箇所、および値設定箇所の対応付けが示されている。X表は誤りや機能変更に伴う修正作業に活用されており，作成機能は大抵の言語処理系が持っている。

本論文では、モジュール間の連絡手段を解析する手法としてX表を利用する方法を考案し、この方法に基づいて試作したツールの概要について述べる。あわせて提案する方法の確実性について論ずる．

2．着想

複数のモジュールをもつプログラムにおいて，モジュール間を連絡している変数（以下，連絡変数）を介した値の授受状況を把握する．あるモジュールでの連絡変数の操作に着目すると，モジュール実行列に沿って一つの操作列が浮かび上がる．常識から，この列を構成している操作の内容は，値の設定，参照，無関係に分類できる．設定，参照などがソースコード上で何に対応するかは後で述べる（表1）．ここでは漠然としたイメージの儘にしておく．変数を，データの流れを仲介するという観点から捉えると，

「最初に，何らかの値が設定され，その値が参照される」という操作列上の規約に気付く．変数がいくつかのモジュール間の連絡用として使用されている場合は，各々のモジュールで区切られた操作列それぞれが，この規約を満足しているという訳ではないが，これらの断片を繋ぎ合わせたものは満足している．

この考えに依ればモジュール毎の断片的な操作列から連絡変数を介した値の授受種別が把握できる．授受種別とは図1のように上下位と連絡しているモジュールで，図の①,③のように他から値を受け取る場合を「受」とし，②,④のように他へ値を授けるのを「授」とする．また，受け取りかつ授ける，あるいは授けかつ受け取る場合を「授受」とし，無関係を「無」とする．さて，あるモジュールにおいて操作列断片が参照のみからなるものとする．このモジュールは連絡変数を介して値を受け取っている，つまり受と判断できる．何故なら，操作列規約に従えば値を設定している操作列断片をもつ他のモジュールがある筈である．同様に，設定のみからなる操作列断片をもつモジュールは変数を介して何れかのモジュールに値を授けている，つまり授と判断できる．設定及び参照の共に存在する操作列断片をもつモジュールでは操作の順序によって二つの場合がある．設定後に参照されている操作列断片をもつモジュールでは何れか別のモジュールに値を授ける授であるし，参照後に設定，もしくはさらに参照がある操作列断片をもつモジュールでは値を受け取りかつ授けている授受である．

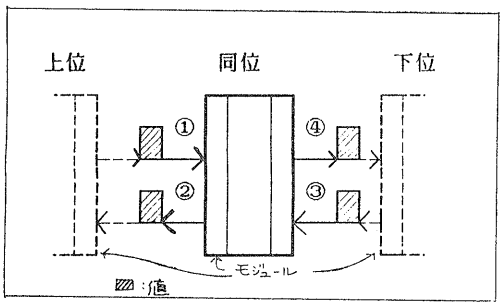

図1　連絡手段の概念図

3.ツールの試作

上の着想に基づいてツールを試作し手近な例に適用してみる．上の着想を実現するには操作列が必要である．しかし，これを正確に把握するにはデータ流を動的に解析せねばならない．このためにはプログラムを動作させるなど大がかりな仕掛が必要となる．簡便に済ませる方法としてx表の利用を思い付いた．固よりX表だけは実行列に関する情報が欠けるために正確さは期待できない．人間への補助としてどの程度利用できる実用性があるかを試してみることとした．

3.1ツールの原理

実用性を狙っているので資産の多いSYSL言語[4]を対象とする．SYSLには一つのコンパイル単位の内部にいくつかの手続きが包含できる．ここではこの内部手続きをモジュールとして授受種別を解析する．

着想の具体化に当たって①連絡変数の識別，②操作の定義が必要である．連絡変数には大局変数と引数があるが後者は前者の議論に含まれるのでここでは主として大局変数について述べる．

(1)連絡変数の識別

SYSLでは，最内側モジュールでの変数名定義が優先されるという有効範囲規則をもっている．モジュールの範囲が不明であると連絡変数が確定できないことがある．X表だけではこの範囲が得られないためにC表[CLD]を補助的に利用してX表から連絡変数を確定する．なお，C表はこの他にも補助的にモジュール授受種別の確定に使用する．

[CLD]C表：コンパイルド・リスト，コンパイラの出力した文番号付ソース・リスト

(2)操作の定義

X表にはデータの流れを仲介しない分岐先きなどの変数名が含まれている．これらを除く．設定・参照の定義の大枠はX表の表示に従うが，変数宣言の定数付きおよび再定義付きの場合と仮引数について用法に沿うように補正する．操作の定義を表1にまとめる．

(3)モジュール授受種別の判断

着想に示した通り，モジュールでの操作が参照，設定，無関係のみの場合は一意に判断できる．設定および参照が同時に存在するのは二つの場合がある．他モジュールの授受種別との関係で判断する．いったんモジュール単位に授受種別を決めたのち，次のように補正する．

1:授受が唯一で他が授であれば授受を受とする．

2:授受が唯一で他が受であれば授受を授とする．

3:他の場合は授受の儘とする．

3.2ツールの概要

ツールの主な構成要素は，X表加工部，解析・表示部である．X表加工部は連絡変数を識別するためにX表及びC表をホストで処理している．加工されたX表の情報は回線を通じてパソコンに送出され，パソコン側で解析表示する．

加工は次の手順により行われる．

S1:値に関する変数の抽出．

C表を参照しながらX表にある全ての変数から値に関する変数のみを取り出し名前と文番号と対応させる(X'表)．

表1　設定・参照操作と文の対応

操作	該変数の文での操作内容
設定	初期値付き定義 代入文の左辺 入力文
参照	条件節（if文,繰り返し文,etc） 代入文の右辺 呼び出し元で設定操作のみの実引数 出力文
無関係	仮引数 該変数に無関係な文

註]再定義付き変数については対応する操作を併合する

S2:モジュール有効範囲の決定

C表からモジュール毎にその定義範囲を示す文番号の上下限値を得る(MA表)．

S3:変数有効範囲の決定

X表の定義文番号とMA表を用いて変数毎に有効範囲を決定する(VA表)．

S4:仮の連絡変数の決定

MA表,VA表を用いてその有効範囲に幾つかのモジュールを含む変数を見つける(TG表)．

S5:連絡変数の抽出

仮の連絡変数の中，複数のモジュールで設定，参照がされている変数をMA表及びTG表を用いて抽出する．

S6:モジュール授受種別の決定

X表における連絡変数についての設定，参照に基づいてモジュール授受種別を確定し，序いで3.1節で述べたモジュール間の関係に従って補正する．

3.3　試用

試作したツールを用いて内部手続き間の連絡手段の解析に試用してみた．適用したプログラムは735文をもちモジュール数8，モジュール呼び出し文数は68の規模で，文字列を処理するツールとして実際に使用されているものである．

解析の結果，本プログラムには12の連絡変数があった．これらの連絡変数を，手作業で分析した結果と本ツールの結果とを対比させて表2に示す．この表に示すように本例では手作業の内容とツールの結果とはほぼ一致している．一方，短所は複数の授および授受がある場合に，これらに対応する受が類別できない．

以上は一例の結果ではあるが種々の連絡手段を含んでいる実用例であることから実際のモジュール連絡手段はそれほど複雑ではなく，ここで考えたX表での設定，参照関係によってもある程度は実用的に解析できることを示唆している．

4.考察

試作したツールはある程度の効果があるとの感触は得られた．しかし，この結果からだけからでは精度の不安が残る．この点について考察する．

4.1用語

■操作　[R操作/S操作/ε操作]

値の設定をS操作、値の参照をR操作、無関係をε操作という。

■操作列

集合{S,R,ε}について、次に定義する代数による代数式を操作列という。

[代数] εを単位元、φを零元とし、・を積、＋を和とする環に於いて、積の可換則の成り立たない代数である。慣例に従って拡張演算子⁺を次のように定義する。

$x^+ = x + x \cdot x + x \cdot x \cdot x + \cdots$、

なお、積の演算子は記述を省略し、X・X=XX と書く。

この代数を用いると変数の操作列を表現するのが簡潔になる．例えば，モジュール内で少なくとも一回以上参照されている変数の操作列はR^+と表現でき，設定後に参照は$S \cdot R^+$と表現できる．

表2 分析結果

手作業 連絡変数	内容[12]	用途	ツールによるモジュール授受種別[11] M1	M2	M3	M4	M5	M6	M7	M8
01	※1	定数	←	→					→	
02	※1	〃	←	→					→	
03	※2	〃	←	→						
04	※3	一般	→	<>	<>	<=		<>	→	→
05	※4	〃	→	→	→	<=		→		→
06	※5	〃	→		→	<=	=>			
07	※6	フラグ	←					<>		
08	※7	〃	←					<>	<>	
09	※8	〃	←					→	←	
10	※6	カウンタ	←					<>		
11	※9	〃	←					→		
12	※10	〃	<>	<>	<>			<>	→	→

[11]… →:授，←:受，<>:授受，<=:引数からの受

[12]…※1　M1での値がM2,M7に授けられる

※2　M1での値がM2に授けられる

※3　M4での値がM2,M3,M6に授けられ更新される．また，M1,M7,M8に授けられる

※4　M4での値がM1,M2,M3,M6,M8に授けられる

※5　M4での値がM1,M3,M5に授けられる

※6　M1での値がM6に授けられ，M6内で再設定・参照される

※7　M1での値がM6,M7に授けられ，M6,M7内で再設定・参照される

※8　M1での値がM6に授けられ，更に,M7で再設定されM6に授けられる

※9　M1での値がM6に授けられる

※10　M1での値がM1,M2,M3,M6に授けられ更新される．また，M7,M8に授けられる

■分割操作列　m_i

連絡変数の操作列について，モジュール呼び出しによる変化を単位として区切られた部分操作列．$m_i \in \{\varepsilon, R^+, S, R^+S, SR^+, R^+SR^+\}$．

図2に例示する．図では操作列が三つのモジュールM_1, M_2, M_3でのモジュール呼び出しによって細分化され，分割操作列$m_1 \sim m_{11}$に区切られている様子を表している．

■モジュール操作列集合　K

分割操作列の集合$\{m_i\}$においてモジュールに関しての部分集合．Kで表す．図2において，モジュールM_1ではK=$\{m_1, m_5, m_7, m_9, m_{11}\}$，モジュール$M_2$ではK=$\{m_2, m_4, m_6, m_{10}\}$，モジュール$M_3$ではK=$\{m_3, m_8\}$である．

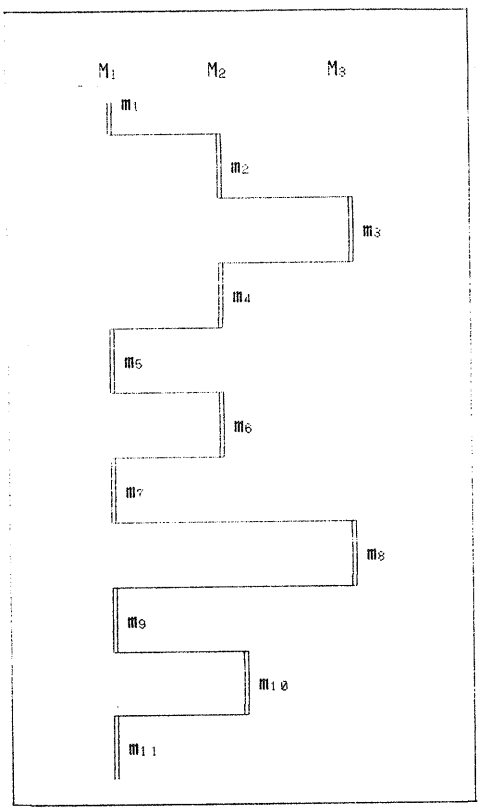

図2　分割操作列の概念図

■#モジュール授受種別　r

#モジュール授受種別rはKの要素k_iに対して演算子#によって求めた値．

$r = k_1 \# k_2 \# \cdots \# k_n$，　ここで，$k_i \in \{0,1,2,3\}$．半群$(\#, \{0,1,2,3\})$の演算子#は表3の演算規則を持つ．

表3　#の演算規則

	K_{i+1}			
#	0	1	2	3
0	0	1	2	3
1	1	1	3	3
2	2	3	2	3
3	3	3	3	3

K_{i+1}

4.2　議論

モジュールに唯一の操作列しかない場合にはデータの授受種別は一意に決まる．$\{\varepsilon, R^+, S, SR^+, R^+S, R^+SR^+\}$に対応して{無関係，受，授，授，授受，授受}である．これは常識と一致する．無関係を0に，受を1に，授を2に，授受を3に対応させると#演算によって#モジュール授受種別が求められる．一つのモジュールに複数の操作列がある場合は，基本的には分割操作列の論理積をとると授受の意味は常識と一致しそうである．演算子#はこの目的で考案されたものである．しかし，モジュール授受種別と実際のモジュールにおけるデータの授受とが一致しない場合がある．例えば，図3の例で$m_1=SR, m_2=RS, m_3=R$と$m_1=SR, m_2=Rm, m_3=R$として#モジュール授受種別を求める．どちらの場合もモジュールM_1の#授受種別は，$m_1 \# m_2 = 3$である．本来は，前者は「授受(つまり，3)」であるが，後者は「授(つまり，2)」であり誤差を生ずる．このような誤差は，①#授受種別と本来のものとの誤差，②X表の設定参照関係に基づくX授受種別と#授受種別との誤差，に分けられる．この誤差について述べる．

分割操作列集合の部分集合$\{R^+, S, R^+S, SR^+, R^+SR^+\}$の外積について#モジュール授受種別と操作列の観察から得られる本来のそれとを比較する．外積の部分集合$D_1=\{(S, R$

$^{+}S),(S,R^{+}SR^{+}),(SR^{+},R^{+}S),(SR^{+},R^{+}SR^{+}),(SR^{+},R^{+})\}$に関しては#演算によると授受となるが，本来は呼び出し先の操作列に依って授受もしくは授であり，呼び出し元の操作列からだけでは判断できない．

次ぎに，#授受種別とX授受種別とでは，$D_2=\{(S,SR^{+})(SR^{+},S)(SR^{+},SR^{+})\}$に関して授が授受と看なされる．

従って，$D_1 \cup D_2$の場合に誤差があるといえる．

6.むすび

コンパイラから出力される相互参照表（クロスリファレンス・リスト）に示されている設定，参照関係に基づいてモジュール間のデータ授受手段を解析する手法を考案し，ツールとして実現した．試用の結果では，手作業による解析結果に則しており，実用性についての見通しが得られた．今後は，本ツールの適用によって連絡手段の実現方法について把握すると伴に現状では制約されているコンパイル間に跨るモジュール間のデータ授受種別を解析できるようにし，最終的な目標であるモジュール結合を可能とする計画である．

以上

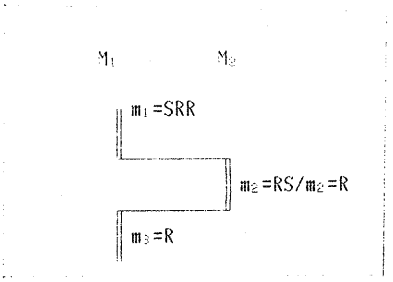

図3　#モジュール授受種別が曖昧になる場合

参考文献

1）島内剛一，"プログラミング言語論"，電子計算機基礎講座，共立出版(1972)

2）B.H.Liskov and S.N.Zilles, "Programming with Abstraction Data Types", SIGPLAN Notices 9,4(1974) pp.50-59

3）D.L.Parnas, "A Technique for Software Module Specification with Examples", CACM 15,2(1972) pp.330-336

4）G.J.Myers, Reliable Software Through Composite Design, Petrocelli/Charter Publishers, Inc., 1975（邦訳「高信頼性ソフトウェア--複合設計，近代科学社　1976)

5）寺島他，"システム製造用言語SYSL"，電気通信研究所実用化報告　第24巻第1号(1975) PP.95-108