Vol. 27 No. 6 情報処理学会論文誌 June 1986

HCP 図法で記述されたプログラム解法の S 代数による定式化†

佐藤 匡正††

一つの問題に対して複数のプログラムができる．このプログラムの差異は解法にあると考えられる．この解法の等価性を数式上で判断することができれば便利である．このためには解法を定式化する必要がある．ここではこの考えから解法の文書化技法の一つである HCP 図法で表現された解法について，その一側面である処理の連なりに関する定式化を試みる．基本的には，処理の連なりは不要な飛び越しのない，いわゆる GOTO-less であれば和と積の演算子をもつ代数で定式化できる．しかし，この形式化では連なりの意味が保存されないことがある．連なりには繰り返しや選択の処理要素における条件記述とこれに条件を与えている処理要素とが動作決定のために結合していることがある．この場合には分配律が成り立たなくなる．代数の適用にはこの動作決定のための結合を切る必要がある．ここでは，この措置として処理連なりの分解策をとる．処理の連なりを，結合に関与している連なりである「機構」部分と解法の入出力に係わり結合には関与していない連なりである「構造」部分とに分解する．この分解により得られる「構造」は解法を反映しており，適当な代数が定義できれば，「構造」の代数式変形によって解法の変形が推定できることになる．本論文では，S 代数の定義，構造と機構の分解，構造の図上での変形と S 代数式の変形との対応の保証などについて述べる．

1. 序　論

これまでのプログラム記述経験によれば，一つの問題に対して複数の解法ができることがある．この解法のバリエーションの差異分析にはいろいろな観点が考えられる．その一つとして動作の一面を表している処理の連なりに注目して分析する方法を試みる．

従来，解法はフローチャートを始めとする種々のプログラム文書化技法[2]によって表現されてきた．その技法の一つである HCP 図法[1),3)]で記述されている解法を図形上でいろいろと変形してみると見掛けは様々な解法であっても処理の連なりが同じであるような図形に帰着させられることがあった．この図形での変形の過程は発見的な面も含まれてはいるが大部分は機械的に行えそうであるとの感触であった．このような経験から解法での処理の連なりを定式化できれば解法バリエーションの差異分析に数式が適用できるために便利になる．

これまでプログラムの定式化といえばその動作の意味に着目して数学的な正当性を与えるのが主目的であった．プログラムの解法に関して停止性や条件の妥当性などのプログラム動作の正当性を保証することを目的として述語論理を始めとする様々な試みが行われてきた．ここでは見方を変え，動作の意味ではなく，動作を表している処理の連なりに注目する．処理の連なりは解法の等価性を判断するための一要因である．

そこで処理の連なりに関して定式化を試みることにする．処理の連なりは不要な飛び越しのない，いわゆる GOTO-less であれば連接，選択，繰り返しで表現できるので基本的には，環のような，和と積をもつ代数で定式化できる．しかし，連なりには繰り返しや選択の処理における条件記述とこれに条件を与えている処理とが動作決定のために意味上で結合することがある．この場合に分配律を適用すると条件の付与と選択における条件判断とが逆転した連なりとなることがあり，式変形によって得られたプログラムには元のプログラムの意味が保存できていない．連なりの定式化にあたっては，この動作決定における意味上の結合を切るための何らかの技術的な措置が必要となる．

ここでは，この措置として処理の連なりを，「解法としての意味は保存したままで，しかも動作決定条件に関する結合はない」ように分解する．処理の連なりには構造と機構が規定されているとみることができる．構造とは処理の連なりとして取り得る範囲の規定であり，機構はこの範囲を一意に決定するための規定である．構造は解法の入出力に連なり，機構は解法内部に閉じる．したがって，構造は解法の骨子をなし，しかも動作決定上の結合はないと考えられる．そこで，この構造に着目して定式化を図るという着想が得られる．

本論文では，① S 代数の定義，② 解法の構造と機構の分解，③ S 代数式の変形と構造の変形との対応保証について述べ，HCP 図法における構造の定式化の確立を目指す．

† A Formalization by S-Algebra for Program Algorithm Described in the HCP-Charting by TADAMASA SATOH (NTT Yokosuka Electrical Communication Laboratories).

†† 日本電信電話(株) 情報通信処理研究所

2. S代数の定義

後述する解法の構造を表現するための代数Sを定義する.

[定義1] 代数 S: 集合 A の元について二つの演算子・と+をもつ代数を定義する. この代数は次の条件を満たす.

<条件1> 分配律, 交換律, および結合律. $a, b, c \in A$ に対して次が成り立つ.

(1) $a\cdot(b+c)=(a\cdot b+a\cdot c)$,
$(b+c)\cdot a=(b\cdot a+c\cdot a)$

(2) $a+(b+c)=(a+b)+c$

(3) $a+b=b+a$

<条件2> 同一元の演算.

(4) $a\cdot a=a^2$, $a\cdot a\cdot a=a^3, \cdots$

(5) $a+a=a$

<条件3> 定数の演算. ϕ を零元, ε を単位元とすると,

(6) $a\cdot\phi=a$, $\phi\cdot a=\phi$, $a+\phi=a$

(7) $a\cdot\varepsilon=\varepsilon\cdot a=a$

ここで, 演算子・, +をそれぞれ, 積, 和と言う. 積は省略することがある.

[定義2] 式: 次の BNF によって生成される<式>を式と言う.

<式>::=<積形式>|<式>+<積形式>
<積形式>::=<単項式>
|<積形式>・<単項式>
<単項式>::=<英文字>|<定数>
|(<式>)
<英文字>::=$a|b|\cdots|A|B|\cdots$
<定数>::=$\varepsilon|\phi$

なお, 式の一部分を指すときには便宜的に「部分式」と言う.

[定義3] 繰り返し; 繰り返し子*: 式 a の繰り返し $(a)^*$ を次のように定義する.

$(a)^*=(\varepsilon+a+aa+aaa+\cdots)$

なお, 繰り返し式のない式を基本式と言う.

以上で核となる代数が定義された. この核に次の定義を加えて拡張する.

[定義4] 出口; 出口子_: 出口をもたない基本式 x の部分式 x_i に記号 _ を付した部分式を出口と言う. 出口子は括弧の展開において「)」とのリンクを示す演算子である. 出口子は入れ子で使用される. いま, 出口子の i 水準での入れ子を $\underline{a}_i$ と略記する. 例えば, $\underline{a}=\underline{a}_2$, $a=\underline{a}_0$ である. ある入れ子の水準における出口子は次の演算規則をもつ.

規則1 括弧の展開: $(\underline{a}_i)=\underline{a}_{i-1}$,
$((a+\underline{b}_i))=(a+\underline{b}_{i-1})$

規則2 積の展開: $(\underline{a}_i)\cdot b=(\underline{a}_i)$,
$a\cdot(\underline{b}_i)=(\underline{a\cdot b_i})$

規則3 和の展開: $(\underline{a}_i+b)=(b+\underline{a}_i)$,
$(\underline{a}_i+\underline{b}_i)=(\underline{a+b_i})$

規則4 繰り返しの展開: 出口をもつ基本式 y の繰り返し式 y^* は規則1, 2により $(\underline{a})^*$, $((a+\underline{b}))^*$ に帰着される. これらの展開形を次式で与える.

$(\underline{a})^*=(a+\varepsilon)$
$((a+\underline{b}))^*=(a^*b+\varepsilon)$

[定義5] 入口; 入口子 $\bar{}$: x の部分式 x^i に記号 $\bar{}$ を付した部分式 $\bar{x}^i$ を入口と言う. 入口子は出口子と双対の演算子で次の演算規則をもつ.

規則1 括弧の展開: $(\bar{a}^i)=\bar{a}^{i-1}$
$((a+\bar{b}^i))=(a+\bar{b}^{i-1})$

規則2 積の展開: $(\bar{a}^i)b=(\overline{ab^i})$
$a(\bar{b}^i)=(\bar{b}^i)$

規則3 和の展開: $(\bar{a}^i+b)=(b+\bar{a}^i)$
$(\bar{a}^i+\bar{b}^i)=(\overline{a+b^i})$

規則4 繰り返しの展開: $(\bar{a})^*=(a+\varepsilon)$
$((a+\bar{b}))^*=(ba^*+\varepsilon)$

3. 解法構造の定式化

3.1 HCP 図法の用語

プログラム解法を表現する技法として HCP 図法を用いる. HCP 図では, プログラムの処理階層, 制御構造, データ流が規定される[1] が, これらの規定の前二つを中心とした処理の連なりに関する規定について形式化する. ここでは, この準備として図法での用語を整備する. 特に階層についての概念を整理するとともに階層開始を表すための記号を新たに規定する.

■データ操作: 図法での処理記述単位である. HCP 図法での記号○, およびこの記号に副記号 ＊, ○, =, −, ◯, ▷, × を組み合わせた複合化記号で表された記述を指す. 本節まで曖昧なままで処理という用語を用いてきたが, 処理はこのデータ操作を指す時とこの組み合わせを指す時がある. 今後は区別する.

■変数: データ操作は情報設定と情報判断を行う選択とで大部分を占めると考えられる (全記述に対して 91% を占める[4]). このようなデータ操作間のやりと

りとしての情報の入れ物が想定される．このような情報の入れ物を変数と言う．

■連なり：連接によって与えられるデータ操作の列を言う．

■解法：HCP 図法の規定によって生成し得る連なりの集合を言う．

■階層：

＊階層水準…主階層開始［記号⊤］で始まり，主階層終了［記号⊥］で閉じられている一本の制御線に連なる範囲を言う．

＊階層開始…主開始［記号⊤］と副開始［記号▽］とがある．主開始は HCP 原図法に従う．副開始は HCP 原図法にはない記法である．階層の開始時点が該水準より上位にあることを表す．▽の数で示された相対水準数だけ上位階層に短絡することを意味する．繰り返しでの副開始は該階層水準における実質的な入口を表す．繰り返しの最初のサイクルでは▽で示されたデータ操作で始まるがその後のサイクルでは無視される．次の副階層終了とは双対の規定である．詳細な規約は表3に示す．

＊階層終了…主終了［記号⊥］と副終了［記号立］とがあり，いずれも HCP 原図法の記号の規定に従う．

3.2 S代数の機構と構造

解法について S 代数の適用を試みる．S 代数の基本項目である演算子および定数を HCP 図法と対応づける．単項式はデータ操作とする．基本演算子については，積を連接，和を選択，繰り返しを繰り返しに対応させる．（ ）を階層水準に，出口 _ は副終了，入口 ¯ は副開始に対応させる．定数については，実行されないデータ操作を ϕ，無効なデータ操作を ε とする．これらの対応の下では等号は代数式によって生成し得る連なりの集合が同じであることを意味する．このほかに，置換，定義，参照があるが通常の代数式のそれと同様に考える．**表1**にこの対応を整理する．

さて，HCP 図法の解法についてこの基本項目の対応を前提として定義 1〜5 の成否を調べると定義1，条件1（1）の分配律が不自然になることがある．この理由は，繰り返しおよび選択のデータ操作とこれに条件を与えているデータ操作とが動的に結合して連なりを決定しているために分配律を適用するとプログラム上の意味を失うことがあるためである．

二次方程式の根を求める解法［**図 1**］を例として上の事情を示す．図1で，データ操作は「P；2次方程式の根を求める，l；パラメータを得る，m；根を求め表示する，m_1；実根を求め表示する，m_2；重根を求め表示する，m_3；虚根を求め表示する，n；$D=b^2-4ac$ を計算する，α；D によって分岐する」である．本例で解法の代数式は，$P=l\cdot n\cdot\alpha\cdot(m_1+m_2+m_3)$ である．ここで，P に分配律が成り立つとすると，式 $P'=(l\cdot n\cdot\alpha\cdot m_1+l\cdot n\cdot\alpha\cdot m_2+l\cdot n\cdot\alpha\cdot m_3)$ が得られる．式 P' のプログラム上での意味を考える．α は分岐条件を与えているのであるから（ ）と対をなす $\alpha\cdot(\cdots)$ の形式で意味をもつ．式 P においては，この形式であるが，式 P' では，$(\cdots\alpha\cdots)$ の形式であり，分岐した後にその分岐条件を与えると解釈せざるを得ない．この解釈は上のプログラムの意味からは成り立たない．

S 代数を適用するためには解法に対してこの結合を切らねばならない．原理的には解法から上の結合をもっていない部分を取り出せばよいが，実用上からは式変形にプログラム上の意味を持たせるような措置が都合よい．このために解法を機構と構造とに分解し，構造に対して S 代数を適用する．基本的には，構造は解法の出力を得るのに係わる連なりであり，機構は解法を一意に決定するのに係わる連なりである．

解法と構造，機構の関係を例によって述べる．図1の二次方程式の解法は，選択を展開した連なりを元とする集合 $\{l\cdot n\cdot\alpha\cdot m_1, l\cdot n\cdot\alpha\cdot m_2, l\cdot n\cdot\alpha\cdot m_3\}$ である．この集合表

表 1 S 代数と HCP 図法の対応
Table 1 Correspondence between S-algebra and HCP charting.

S 代数		HCP図法 用語	HCP図法 記法
単項式		データ操作	○◎✳⊖⊖⊖▷⊗
基本演算子	積 $a\cdot b$	連接	a b
	和 $(a+b)$	選択	a b
	繰り返し $(a)^*$	繰り返し	a
	括弧 (a)	階層水準	a
拡張演算子	出口子	—	立
	入口子	—	▽ 注)
定数	ϕ	実行されないデータ操作	〈なし〉
	ε	無効	

注) HCP原図法にはない記法である．

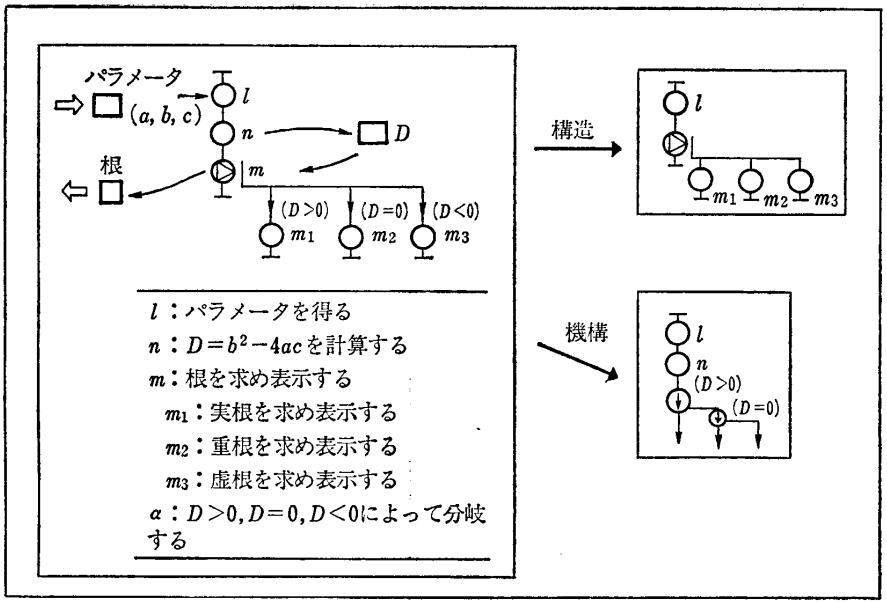


図 1 2次方程式の解法1
Fig. 1 A quadratic equation solution algorithm 1.

現の解法において，どの元が選択されるかは α での D の値によって決まる．D の値は，l と n によって入力から求められる．つまり，$l\cdot n\cdot\alpha$ は解法を一意に決めるための機構である．そこで，これを「機構」と名付けている．この解法の出力は $l\cdot m_i$ $(i=1,2,3)$ で求められる．つまり，集合 $\{l\cdot m_1, l\cdot m_2, l\cdot m_3\}$ は，解法の出力を得るための連なりの構造を保存している．そこで，これを「構造」と名付けている．

構造は，解法から機構で与えられる制約のすべてを取り除いたものである．逆に構造が与えられているとすると，適当な機構を与えることによって解法が確定する．一般に，解法の出力に係わる連なりに関して，構造は解法と等しいか，包含するかである．解法に含まれている選択が相互に独立であれば解法と構造は同じであり，関連があれば包含される．例えば，$a+\sqrt{b}$ の値を求める解法について考える．図2の解法では，α の条件の選択が二つある．解法は，$\{h\cdot\alpha\cdot l_1\cdot m\cdot\alpha\cdot n_1\cdot k,\ h\cdot\alpha\cdot l_2\cdot m\cdot\alpha\cdot n_2\}$ である．一方，構造は，$\{h\cdot l_1\cdot m\cdot n_1\cdot k,\ h\cdot l_1\cdot m\cdot n_2\cdot k,\ h\cdot l_2\cdot m\cdot n_1\cdot k,\ h\cdot l_2\cdot m\cdot n_2\cdot k\}$ である．α は出力には影響を与えないので無視して考えると，構造には解法にはない二つの連なり $h\cdot l_1\cdot m\cdot n_2\cdot k$，$h\cdot l_2\cdot m\cdot n_1\cdot k$ がある．つまり，出力に係わる連なりについて，解法は構造に含まれている．

解法において機構を見つけるにはその構成に着目すれば容易である．機構は条件決定用の情報参照とその情報設定の連なりから構成されているので繰り返しや

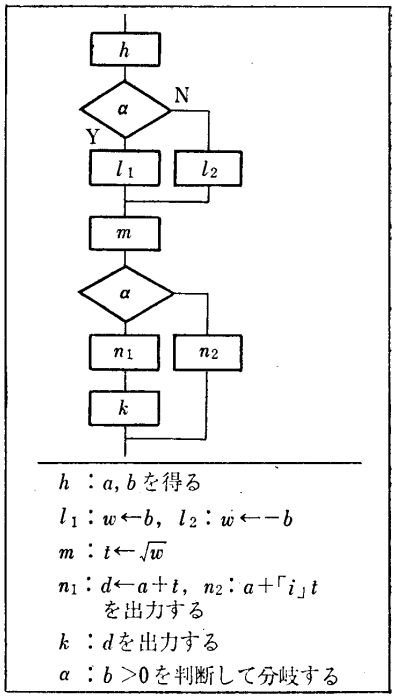


図 2 $a+\sqrt{b}$ を求める解法
Fig. 2 A '$a+\sqrt{b}$' solution.

選択で用いられている変数を鍵としてその変数に情報を設定しているデータ操作を見つければよい．構造を得るには解法からこの機構を取り除けばよい（措置1）．ただし，機構と構造がデータ操作を共用することがある．この場合は単純に機構を取り除くと構造の一部が欠けてしまうので共用部分が構造から除外され

ないような配慮が必要となる（措置2）.

構造は次の措置によって与えられる.

措置1…解法の繰り返しおよび選択のデータ操作における条件記述を削除.

措置2…上の条件記述における変数のうちで条件決定用の変数を識別しこの変数に関する条件設定のデータ操作を削除. 条件決定用の変数とは，解法のすべてのデータ操作において，その変数について情報は設定されるが繰り返しや選択を除いては参照されていないものである.

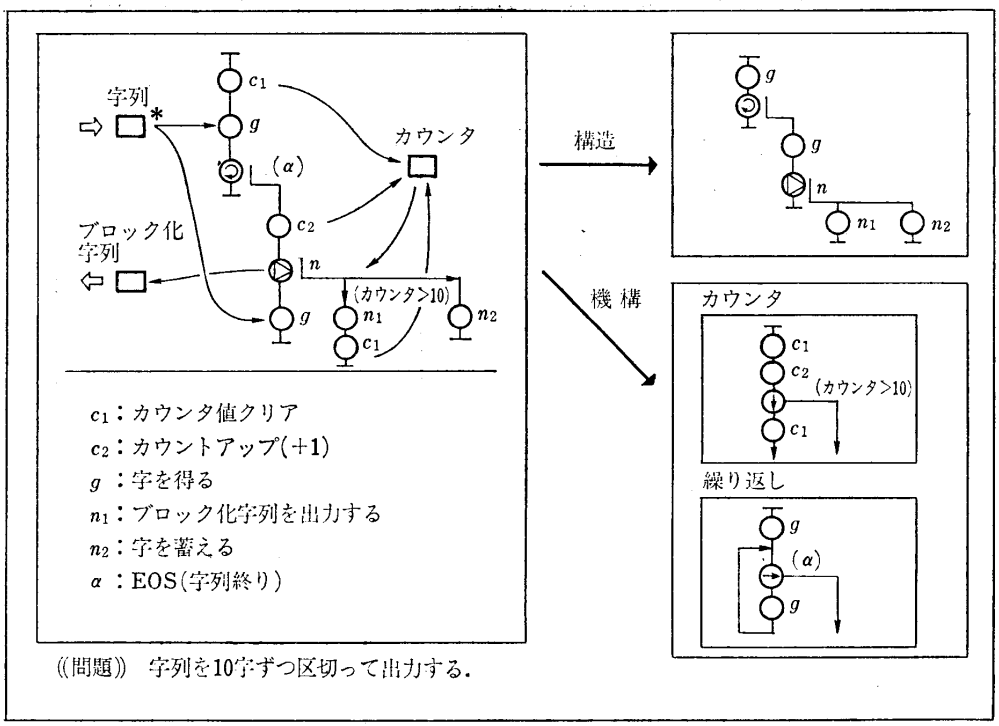


図 3 字列の10文字ブロッキング解法
Fig. 3 A character sting blocking algorithm.

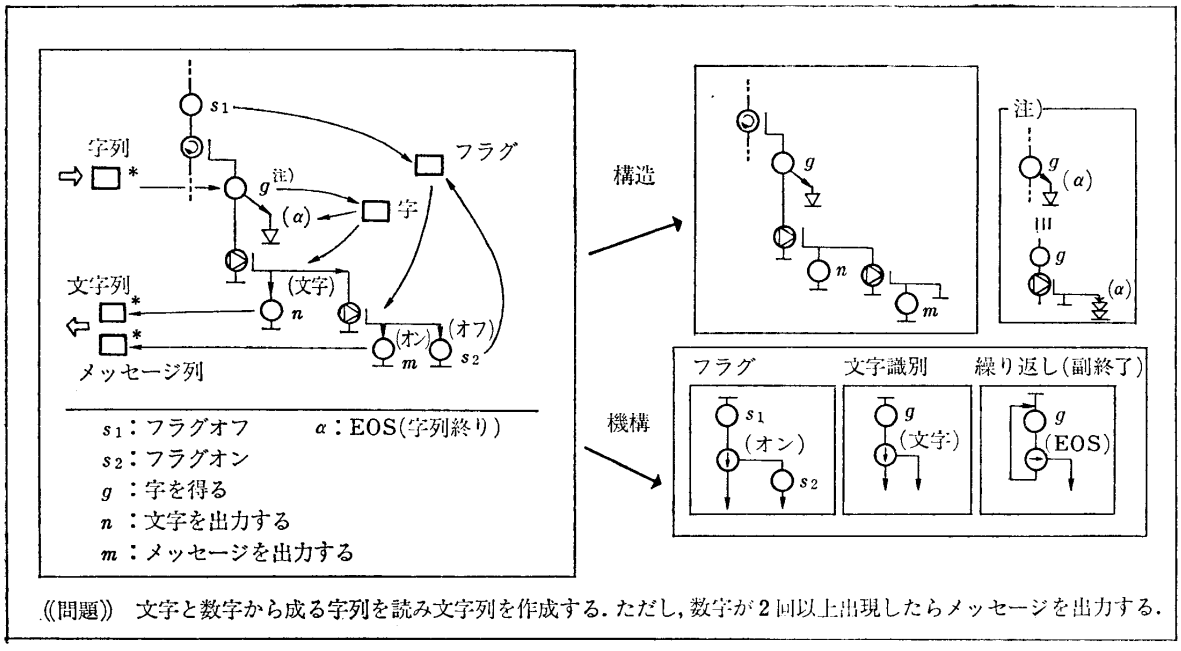


図 4 重複出現をチェックする解法
Fig. 4 A duplication checking algorithm.

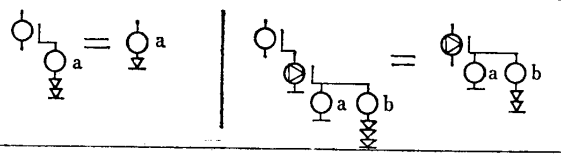
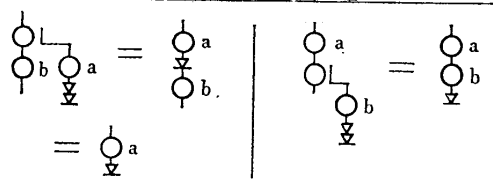
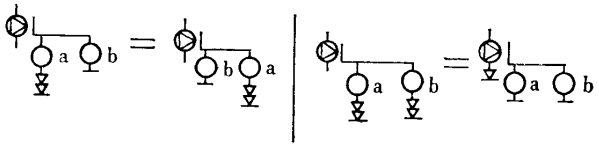
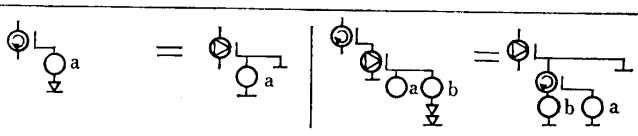
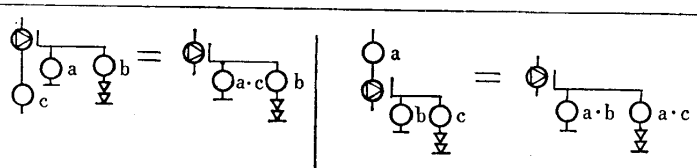
措置3…データ管理用のポインタや配列の制御変数に関する初期値設定，演算などの取り扱い，I/O でのオープン・クローズなど．ただし，データ管理用変数が条件決定用の場合に限る．

図1の二次方程式の解法を例として上の措置1，2の適用を示す．本例での機構は選択の条件決定で使用されている．この条件記述の変数 D に着目する．D に情報設定している n, n の a, b, c に値を与えている l が確定する．つまり，機構はデータ操作である l, n と選択データ操作 α から成る．構造は措置 1，2 から得られる．措置1から選択データ操作 α が除かれ選択での変数 D が定まる．措置2により D への情報設定である n が除かれ構造が確定する．図1の構造図からは解法には3通りの場合が存在することが，機構からは構造における3通りの場合に分ける仕掛けが分かる．なお，本例の機構は計算式の値を用いている選択機構であるが，このほかにもフラグやカウンタなどが用いられる．

カウンタの例として文字列を一定の長さに切り出す（ブロッキング）解法を，フラグの例として文字列中に同一の文字が重複して出現したか否かを調べる解法を，それぞれ**図 3，4** に示す．図3の解法では繰り返しと選択の二つの機構がある．選択 n の変数はカウンタであり，これについての c_1, c_2 および条件（カウンタ＞10）；（それ以外）である．繰り返しの変数は「字」で，これに係わるデータ操作は g と条件 EOS（字列終わり）である．g は n_2 に対する措置2により構造として残る．図4の解法では機構は副終了，n, m に対して三つある．繰り返し1，選択2である．繰り返しの変数は字であり，選択はフラグと字である．これらの変数に係わるデータ操作から n_1 に対する措置2により g が対象外となり構造が得られる．

上述した構造にS代数を適用すれば具合がよい．構造の定義によりデータ操作の連なりの可能性を表して

表 2　出口子と HCP 図法の対応
Table 2　Correspondence between exit operator and HCP charting.

出口子規則		HCP図　法
id	式	
1. 括弧の展開	$(\underline{a})=\underline{a}$ $((a+\underline{b}))=(a+\underline{b})$	
2. 積の展開	$(\underline{a})\cdot b=\underline{a\cdot b}=\underline{a}$ $a\cdot(\underline{b})=\underline{a\cdot b}=\underline{a\cdot b}$	
3. 和の展開	$(\underline{a}+b)=(b+\underline{a})$ $(\underline{a}+\underline{b})=(\underline{a+b})$	
4. 繰り返しの展開	$(\underline{a})^*=(a+\varepsilon)$ $((a+\underline{b}))^*=(a^*b+\varepsilon)$	
5. 複合例	$(a+\underline{b})\cdot c=(a\cdot c+\underline{b})$ $a\cdot(b+\underline{c})=(a\cdot b+\underline{ac})$	

おり，かつすべてのデータ操作は選択条件の決定には関係していない．したがって，構造では動作決定に関する結合は切れ，解法の意味は保存されている．つまり，構造については選択データ操作に関する分配，結合，交換の各律が成り立つ．

3.3 S 代数の出入口子と階層

（1） 階層終了と出口

S代数の出口子の演算規則は HCP 図法の記号の規約に対応している．この記号は階層水準での副終了であり，この意味は階層終了時点で▽の数だけ階層水準を遡り，短絡させる．演算規則ごとに図法上との対応を示す．

出口子の規則1は括弧を階層水準に対応させた副終了の規約と解釈される．▽の数だけ階層水準を遡り，水準を遡るごとに▽の数を減じ，水準を深めるごとに数を増やす．規則2の前半では，データ操作 a が副終了である時はデータ操作 b は実行されないと解釈される．後半はデータ操作 a が b と連接し b が副終了である時は ab が副終了であると解釈される．規則3の前半は選択データ操作の下位階層の a と副終了付き b を入れ替えたものと解釈される．構造の定義により選択データ操作は条件決定に係わらないので図法上では成り立つ．規則3の後半は選択データ操作の下位階層の a, b 共に副終了のときは上位階層の副終了であると解釈される．規則4は副終了付きの繰り返しデータ操作を副終了のないものに展開していると解釈される．図法の規約では立に繋がるデータ操作が繰り返し最後のデータ操作の連なりであるので式展開の意味と一致している．出口規則と階層副出口との対応を**表2**に整理する．

（2） 階層開始と入口

(1)項に準じて入口規則と副入口との対応が成り立つ．詳細は省略する．**表3**にこの対応を整理する．

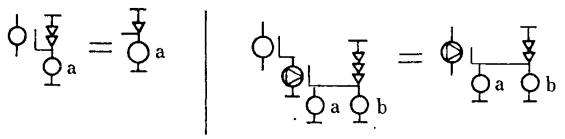
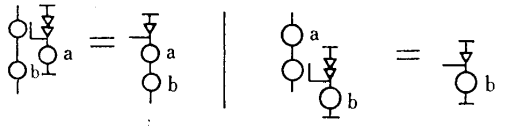
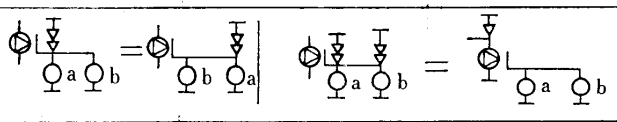
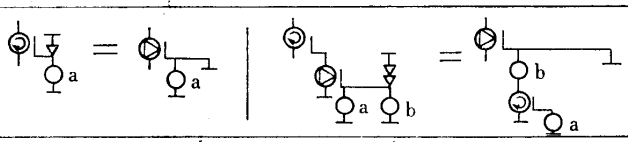
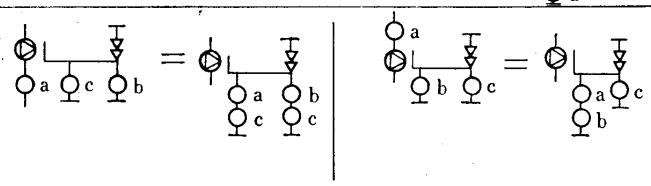
3.4 等号の意味と式変形の応用

以上の 3.2, 3.3 節における対応から解法の構造に対してはS代数の定義 1〜5 が成り立つ．したがって構造はS代数式として表現でき，構造に対する図形上の変形は式変形として扱える．

（1） 構造のS代数による変形と解法の変形の対応

S 代数式の等号はその式で規定し得るすべての連な

表 3 入口子と HCP 図法の対応
Table 3 Correspondence between entrance operator and HCP charting.

入口子規則		HCP図 法
id	式	
1. 括弧の展開	$(\bar{\bar{a}})=\bar{a}$ $((a+\bar{\bar{b}}))$ $=(a+\bar{\bar{b}})$	
2. 積の展開	$(\bar{\bar{a}})\cdot b=\overline{a\cdot b}$ $a\cdot(\bar{\bar{b}})=\bar{b}$	
3. 和の展開	$(\bar{\bar{a}}+b)$ $=(b+\bar{\bar{a}})$ $(\bar{\bar{a}}+\bar{\bar{b}})$ $=\overline{(a+b)}$	
4. 繰り返しの展開	$(\bar{\bar{a}})^*=(a+\varepsilon)$ $((a+\bar{\bar{b}}))^*$ $=(b\cdot a^*+\varepsilon)$	
5. 複合例	$(a+\bar{\bar{b}})\cdot c$ $=(a\cdot c+\overline{\bar{b}\cdot c})$ $a\cdot(b+\bar{\bar{c}})$ $=(a\cdot c+\bar{\bar{c}})$	

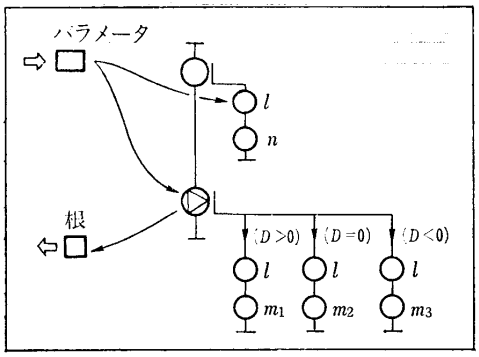


図 5 2次方程式の解法 2
Fig. 5 Another quadratic equation solution algorithm 2.

りが等しいことを意味している．同一計算機に対して同一の初期条件を前提としたときに解法が等しいということはそれぞれの解法が可能なすべての連なりが等しい．等しい解法と等しい構造との関係について考察する．3.2 節で述べたように解法の出力に関して，構造は解法を含んでいる．したがって，構造が等しくとも解法が等しいとは言えない．しかし，機構は定義により，解法の出力には影響を与えない．与えるのは構造である．等構造であれば解法の入出力に対しては同一の連なりを生成し得る．等構造をもつ解法の関係を等価と言うことにする．したがって，構造のS代数による式変形は等価の関係で解法を変形していることに対応する．等価な解法は解法のバリエーションである．

（2） 式変形による解法のバリエーションの例

(2.1) 分 配 律

分配律に伴う式変形の例として図1の二次方程式の解法を考える．

$$R_1=l(m_1+m_2+m_3)=(lm_1+lm_2+lm_3)$$

分配律の適用で得られた右辺で示される構造に対する解法の例を**図 5** に示す．図1と図5は等価である．図1では混在している機構と構造が選択の階層範囲の変化によって図5では機構を分離させている．両者の機構の構成方法が異なる．

(2.2) 繰り返し，選択，出入口子

図4の重複出現チェックの構造を P_3 とする．

$$P_3=(g(\varepsilon+\underline{\varepsilon})(n+(m+\varepsilon)))^* \quad (1)$$

和の積および出口を展開する（出口子規則4）．

$$P_3=(g(n+m+\varepsilon))^*g \quad (2)$$

さて，$(xy)^*x=x(yx)^*$ であるので，

$$P_3=g((n+m+\varepsilon)g)^* \quad (3)$$

式(3)は入口子を用いて一つの繰り返し式にまとめられる（入口子規則4）．

$$P_3=((\varepsilon+\bar{g})(n+m+\varepsilon)g)^*$$
$$=((\varepsilon+\bar{g})(n+(m+\varepsilon))g)^* \quad (4)$$

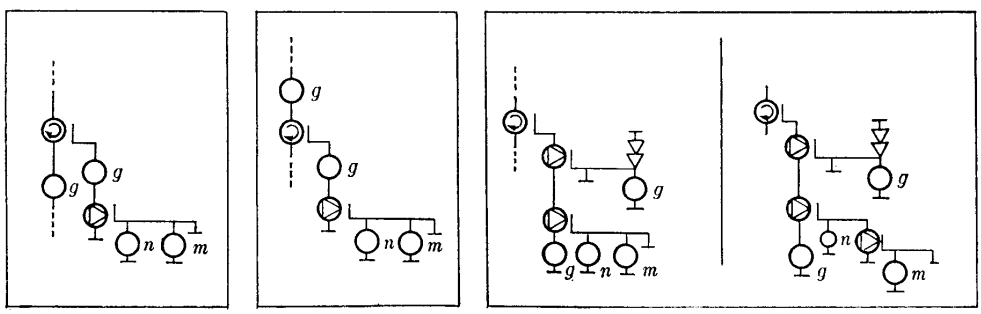
式(4)の図法を**図 6** に示す．図4の構造と上の式変形より得られた図6は等価である．

4. 結 論

HCP 図法で表現された解法の一つの側面であるデータ操作の連なりには構造と機構が規定されている．構造はデータ操作が連なる可能性の範囲を規定したものであり，機構はこの範囲内の連なりを動的に一意に決定するための規定である．これらの規定のうち，構造に対応する代数 S が定義できた．これによって，構造における図上での変形と代数式の変形との対応が保証される．

参 考 文 献

1) Satoh, T. et al.: Documentation Technology for Packing Hierarchical Function, Data, and Control Structures, *COMPCON '81 Fall (IEEE) Proceedings*, pp. 284-290 (1981).
2) 佐藤匡正：プログラミング用ドキュメンテーション，情報処理，Vol. 22, No. 5, pp. 383-389



(a) 式(2)に対する (b) 式(3)に対する (c) 式(4)に対する

図 6 式変形から得られた解法のバリエーション
Fig. 6 Algorithm variation by expression transformation.

(1981).
3) 佐藤匡正，浅見秀雄：フローチャート階層的表現のための一提案，情報処理学会第 20 回全国大会，pp. 285-286 (1979).
4) 佐藤匡正：プログラム処理記述文に関する分析，情報処理学会第 30 回全国大会，pp. 735-736 (1985).

(昭和 60 年 9 月 10 日受付)
(昭和 61 年 3 月 20 日採録)

佐藤　匡正（正会員）

昭和 42 年横浜国立大学工学部電気工学科卒業．同年日本電信電話公社に入社．電気通信研究所，データ通信本部などにおいて銀行業務処理システム，言語処理プログラムの開発に従事．現在，日本電信電話(株)情報通信処理研究所に所属．プログラム設計方法とプログラム構造との関係に興味をもっている．情報処理学会論文賞受賞(昭和 56 年度)．電子通信学会会員．