ソフトウェア基礎論19－2
プログラミング言語 9－2
（1986. 12. 12）

並行システムの代数的仕様記述法

Algebraic Specification Method for Concurrent Systems

高木雄二＊　坂部俊樹＊＊　稲垣康善＊
Yuji TAKAGI　Toshiki SAKABE　Yasuyoshi INAGAKI

＊ 名古屋大学 工学部　＊＊ 三重大学 工学部
Nagoya University　Mie University

あらまし　並行システムを代数として定式化し記述するものにMilnerのCCSがある．CCSは互いに値の受け渡しを行う同期通信を基礎としているが，受け渡しされる値に関しては何の形式的な記述法も与えられていない．そこで，我々は値の空間を抽象データ型としてとらえ代数的に記述する並行システムの代数的仕様記述法CCS/ADTと，その意味論としてCCSのモデルである通信木を修正した状態付通信木を提案する．この方法は，プロセス内部の逐次動作を抽象データ型として記述し，他のプロセスとの通信動作をCCSの枠組で記述する直観に合った仕様記述法である．

Abstract　Milner proposed CCS (Calculus of Communicating Systems) as an algebraic specification method for concurrent systems. CCS is based upon the synchronized communications, but has no specification method for value passing. So we propose an algebraic specification method for concurrent systems, named CCS/ADT, in which we grasp the domains of values as abstract data types and describe them algebraically. The semantics of the specification in CCS/ADT is given by the communication tree with states, the modified communication tree. This allows us to obtain a natural algebraic specification method, fitting our intuition, for concurrent systems.

1．はじめに

近年のハードウェア技術，ソフトウェア技術の発達によって，多彩な並行システムの開発が可能になり，並行システムの重要性が増大しつつある．しかしながら，並行システムは従来の逐次的なシステムに比べて，その本質として内在する非決定性のために非常に複雑なものとなっている．したがって，並行システムの検証や構成を目的として並行システムの形式的仕様記述法やその意味論の研究が盛んに行われている．

並行システムに対するアプローチには種々のものがある．例えば，ペトリネットのような抽象機械による方法，表示的意味論，公理的意味論，代数的な方法がある．中でも代数的な方法は他の方法に比べて，形式性が高い割に記述性，読解性に優れていると思われる．

並行システムを代数的に記述する方法としては，文献(4)のAstesianoらのアプローチとMilnerのCCSを源とする流れのアプローチがある．Astesianoらの方法では，並行システムを3つの推移システムの合成の推移システムとして捉え，それらを代数的に記述するというアプローチを採用しているが，定式化がこみいっており，記述性，理解性の観点から満足のいくものにはなっていない．

一方，MilnerのCCSは，互いに通信を行う独立の動作体の概念を基礎としており，比較的見

通しの良いモデルである．CCSをもとにして，Bergstraらは，ACP[5]，ACP_τ[6]を，また，MilnerはSCCS[9]，ASCCS[9]を提案しているが，基本的にはCCSと同様の考えにもとづくものである．しかし，いずれにせよ，相互通信を基礎にするシステムであるが，通信によって受け渡しされる値に関しては何の形式的な記述法も与えられておらず，暗黙のうちに仮定されているとしている．また，逐次動作も動作体間の通信を介して記述しており，本来の通信動作と区別できない．そこで本報告では，受け渡しされる値の空間を抽象データ型としてとらえ，それをCCSの中に組み込んで，それを代数的に記述することによって並行システムを代数的に記述する方法CCS/ADTを提案する．さらに，CCS/ADTの仕様に対する意味として，CCSプログラムのモデルとして提案されている通信木を修正した状態付通信木を提案する．最後に，CCS/ADTによる記述例として，バッファー付データ転送システムを記述し，その状態付通信木を与える．

2．並行システムの代数的仕様記述法(CCS/ADT)

Milnerの提案したCCS と抽象データ型の代数的仕様記述法を合わせた並行システムの代数的仕様記述法CCS/ADT について述べる．CCSに関する詳しい解説は文献(1)を，抽象データ型の代数的仕様記述法に関する詳しい解説は文献(2)を参照されたい．CCS/ADT の記述法は 2.1節で与え，その意味論は 2.2節で述べる状態付通信木で与える．仕様に対する意味として通信木の各ノードに動作体の状態を与えた状態付通信木を与えたのは，システムの内部通信動作に対する意味も陽に与えられること，および，抽象データ型の項として記述される通信動作以外のシステム内部の逐次動作が動作体の状態として重要な意味を持つからである．以下では，語句"ソート"の意味は，CCSで使われている意味とは違って，抽象データ型の代数的仕様記述法[2]で使われている意味とする．

2．1 CCS/ADT

並行システムの代数的仕様記述法CCS/ADT の定義は以下の通りである．

［定義 1］ 並行システムの仕様SPECは，3項組 $\langle$ ASPEC, BSPEC, SYS $\rangle$ である．ここで，ASPEC は受け渡しされる値の抽象データ型の仕様，BSPEC はシステムを構成する動作体の動作を定義する仕様，SYS は仕様SPECが表している並行システムの初期システムを表している動作体名である．また，ASPEC中のソートの集合SとBSPEC中のソートの集合S'に関して $S' \subseteq S$ の関係がある． □

文献(2) に従って，抽象データ型の仕様の定義，および，仕様が表している抽象データ型は以下のように定義する．ここで，$\Sigma(X)$項は変数集合XとシグニチャΣから生成される項で，$T[\Sigma(X)]$は$\Sigma(X)$項の集合である．

［定義 2］ 抽象データ型の仕様ASPEC は3項組 $\langle \Sigma, X, \Gamma \rangle$ である．ここで，Σはシグニチャ $\langle S, F \rangle$ で，Sはソートの集合，Fは関数記号の集合族 $\langle Fw,s \rangle_{w \in S^*, s \in S}$である．Xは変数集合族 $\langle Xs \rangle_{s \in S}$，$\Gamma$は等式 $\xi == \eta$ の集合で，ξ，ηは$\Sigma(X)$項である． □

［定義 3］ 仕様ASPEC から定まる抽象データ型 A(ASPEC) は商代数$T[\Sigma]/\doteqdot$である．ここで，$\doteqdot$は項代数$T[\Sigma]$上の合同関係 $\langle \equiv_s \rangle_{s \in S}$で，次のように定義される．

ξ，$\eta \in T[\Sigma]s$ について

$$\xi \equiv_s \eta \quad \text{iff} \quad \Gamma \vdash \xi == \eta$$

ここで，$T[\Sigma]s$はソートsのΣ項の集合である． □

この意味論は，$T[\Sigma]/\doteqdot$がΓの全てのモデルの始代数となっていることから，始代数意味論と呼ばれる．次に，動作体の動作に関する仕様について述べる．

［定義 4］ 動作体の動作を定義する仕様 BSPEC は6項組 $\langle$ S', BE, Y, AN, E, PC $\rangle$ である．ここで，S'はソートの集合，BEは動作体名の集合族 $\langle BEw \rangle_{w \in S'^*}$，Yは変数集合族

< Ys >s∈S',ANは動作名の集合族< ANs >s∈S', Eは動作体方程式の集合,PCは動作体方程式の左辺の動作体名と右辺に現れる動作体名の引数の対応関係式の集合である. □

また，動作の集合ACはANより以下のように定義される.

(1) τ∈AC

(2) a?x∈AC, a!e∈AC

ただし，a∈ANs, x∈Ys, e∈T[Σ(Y)]s

ここで，τを内部動作，a?xの形の動作を入力動作，a!eの形の動作を出力動作という．内部動作τは，動作体の内部で1つの入力動作とそれと対応する出力動作の間で通信が起こり，値の受け渡しが行われたことを示す動作である.

また，eは出力動作a!で出力する値を，xは入力動作a?で入力された値を入れる変数をいう.

次に，動作体の動作を表す基礎となっている動作式の定義を行う.

【定義 5】 動作式は以下のものから作られる式である.

入力動作，出力動作，内部動作，逐次合成(・)，選択(+)，並列合成(|)，制限(\a ，ただし，aは動作名)，if文，ラベル変換([S])，引数付動作体名 □

ここで，何の動作も行わない動作体を表している特別な動作体名NIL は，CCS では引数を持たないが，ここでは任意のソートの任意の個数の引数を持つことにする．また，動作式B の自由変数xに値vを割当てることを表すのに B{v/x} の形の記法を用いる.

動作体方程式は動作式を使って以下のように定義される.

【定義 6】 動作体方程式は以下の形の方程式である.

b[ξ1,…,ξn] = Bb

ここで，bは定義される動作体名で，b∈BEs1…snの時，ξi∈T[Σ(Y)]si (1≤i≤n)である．Bbは動作式でBの中の自由変数の集合をFV(B)で表すと，FV(Bb)⊆FV(b[ξ1,…,ξn])である． □

動作式Bの自由変数の集合FV(B)は，直観的には，Bの中にある任意の入力動作a?xに対して，その後に実行される部分の中の変数x以外の変数をすべて集めたものである．また，CCS と違って，左辺の動作体名の引数の所に変数だけでなく抽象データ型の項が書ける.

【定義 7】 動作体方程式の左辺の動作体名と右辺に現れる動作体名の引数の対応関係式は以下の式である.

(b1, i)→(b2, j)

ここで，b1, b2は動作体名，i, jは非負整数であり,動作体名bの第k引数のソートをsort(b,k)で表すと，sort(b1,i)=sort(b2,j) である．また，b1=b2 の時には，i=j である． □

この対応関係式は，左辺の動作体名b1の第i引数は右辺の動作体名b2の第j引数と対応していることを表している．また一般に，自己対応関係式，すなわち，(b, i)→(b, i)の形の式はPCの記述において省略される.

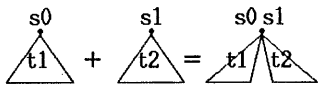
2．2 状態付通信木 (CTS)

2.1 節で定義した並行システムの仕様SPECに対して，その意味を状態付通信木(CTS)で定義する．この節では，この状態付通信木の構成方法について述べる．状態付通信木は，文献(1)で述べられている通信木の各ノードに動作体の状態を割当てた木である．動作体の状態は，対応番号列w と値v との組 < w,v > の集合である．ここで，対応番号列w は動作体内部に記憶されている値v が記憶されている場所を示す名前である．状態付通信木上の動作体の状態は，動作体方程式の集合E と動作体名の引数の対応関係式の集合PCから定義される．ここでは，まず文献(1)の通信木上の演算と同様な演算を状態付通信木上に定義する．この演算は動作式と同様な演算である.

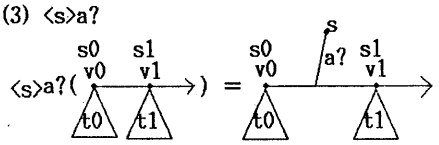
【定義 8】 状態付通信木(CTS)上の演算は以下の通りである．ここで，s,s0,s1は各ノードにおける動作体の状態であり，v,v0,v1は受け渡しされる値である.

(1) <s>NIL　CTS s・ を表す.

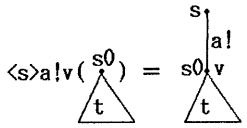
(2) ＋



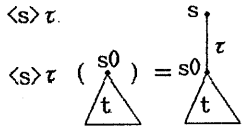
(3) <s>a?



(4) <s>a!

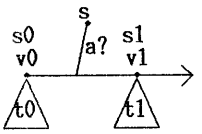


(5) <s>τ

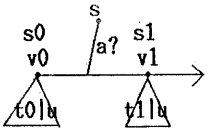


(6) | t,uを根の状態がsであるCTSとするとt|uは次の枝を持っている.

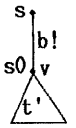
(a) tのおのおのの枝



に対して，枝

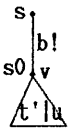


(b) tのおのおのの枝

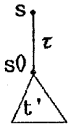


に対して，

枝

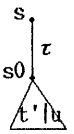


(c) tのおのおのの枝

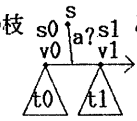


に対して，

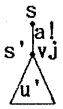
枝



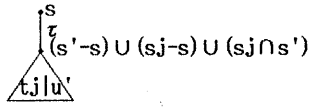
(d) tの枝



とuの枝



のおのおののペアに対して，枝



(e) (a)－(d)のtとuをすべて入れ換えて得られるもの

(7) \a　t\aはtのすべてのa?,a!の枝を取り去った木である.

(8) [S]　t[S]はtのすべての枝に付けられている動作名λをSλに変更する．ただし，Sはラベルの付け替え方を示している関数であるとする．　□

今，動作式B とB の実行前の状態sが与えられた時，sとBから定義される状態付通信木を〖 s,B 〗$_{CTS}$と書く．また，CTS上の演算<s>a?に対する引数CTSの集合のための表記法として，v↦t(v)を用いる．これは，<s>a?で入力される可能性のあるそれぞれの値vに対して t(v)というCTSが対応することを意味している．以上の表記法を用いて，状態と動作式の対に対するCTSが次のように定義される．

以下では，引数の対応関係式の集合PCに，動作体方程式に応じた引数の自己対応関係式も加えた対応関係式の集合をPC' とする．また，対応関係式pの左辺を対応番号列iに変えた対応関係式を lc(p,i)，pの左辺をld(p)と書く.動作式Bを構成している動作体名の集合をCB(B)と書く.

【定義 9】 動作体名，および，状態と動作式の対に対する状態付通信木は以下のように構成される.

(1) 〖 s,NIL[v1,…,vn] 〗$_{CTS}$= <s>NIL

(2) 〖 s,B+B' 〗$_{CTS}$
＝〖 s,B 〗$_{CTS}$+〖 s,B' 〗$_{CTS}$

(3) 〖 s,B|B' 〗$_{CTS}$
＝〖 s,B 〗$_{CTS}$|〖 s,B' 〗$_{CTS}$

(4) $[\![s,B\backslash a]\!]_{CTS}=[\![s,B]\!]_{CTS}\backslash a$

(5) $[\![s,B[S]]\!]_{CTS}=[\![s,B]\!]_{CTS}[S]$

(6) $[\![s,$ if TRUE() then B else B' $]\!]_{CTS}$
$=[\![s,B]\!]_{CTS}$

(7) $[\![s,$ if FALSE() then B else B' $]\!]_{CTS}$
$=[\![s,B']\!]_{CTS}$

(8) 動作体方程式 $b[\xi 1,\cdots,\xi n]=Bb$ が存在し，$\Sigma(Y)$項$\xi 1,\cdots,\xi n$の中の自由変数を$x1,\cdots,xm$とする．この時，
$[\![s,b[\xi 1,\cdots,\xi n]\{v1/x1,\cdots,vm/xm\}]\!]_{CTS}$
$=[\![s,Bb\{v1/x1,\cdots,vm/xm\}]\!]_{CTS}$

(9) 動作体の初期状態を表している動作体名を$b[v1,\cdots,vn]$とすると，(8)におけるsは次のように定義される．
$s=\{ \langle i,vi\rangle \mid i\in\{1,\cdots,n\} \}$
また，$PC''=\{ lc(p,i) \mid p\in PC'$かつ $ld(p)=(b,i) \}$
とする．

(10) $[\![s,a?x\cdot B]\!]_{CTS}$
$=\langle s\rangle a?(v\mapsto [\![t\{v/x\},B\{v/x\}]\!]_{CTS})$
$[\![s,a!v\cdot B]\!]_{CTS}=a!v[\![t,B]\!]_{CTS}$
$[\![s,\tau\cdot B]\!]_{CTS}=\tau\cdot[\![t,B]\!]_{CTS}$
ここで，動作体の状態tは以下のように定義される．以下では，αでa?x,a!v,τのどれかを表すものとし，PC''はsを定義する時に作成されたものとする．

(a) $B=\alpha\cdot B'$ の時，$t=s$ である．PC''は変化しない．

(b) $B=b[v1,\cdots,vn]$の時，
$s=\{ \langle i,ui\rangle \mid i\in T \}$ とする．ここで，$T=T1\cup T2\cup T3$であり，PC''の中のbの引数と対応する式に，左辺がiであるものが1つしかない対応番号列iの集合をT1，2つ以上ある対応番号列の集合をT2，何の対応式もないものの集合をT3とする．
この時，
$t=\{ \langle i,vj\rangle \mid i\in T1, i\rightarrow(b,j)\in PC'' \}$
$\cup\{ \langle i\cdot j,vj\rangle \mid i\in T2,\ i\rightarrow(b,j)\in PC''\}$
$\cup\{ \langle i,ui\rangle \mid i\in T3 \}$
$\cup\{ \langle k,vj\rangle \mid k$は今までに使われていない自然数，vjはs内の対応番号列から対応していない．$\}$
また，
$PC''=\{ lc(p,i) \mid \langle i,vj\rangle\in t,\ ld(p)=(b,j),\ p\in PC' \}$
とする．

(c) $B=(B1|\cdots|Bn)\backslash a1\backslash\cdots\backslash am$ $(m\geq 0,n\geq 1)$の時，
$si'=\{ \langle g,vg\rangle \mid \langle g,vg\rangle\in s,$あるb$\in CB(Bi)$に対して，$g\rightarrow(b,h)\in PC'' \}$
$s0=s-(s1'\cup\cdots\cup sn')$
$si=(si'-\cup ssij)$
$\cup\{ \langle g\cdot i,vg\rangle \mid \langle g,vg\rangle\in\cup ssij \}$ とする．
ここで，$si'\cap sj'=ssij$であり，$\cup ssij$はすべてのjに対するssijの和集合である．
また，
$PCi''=\{g\rightarrow(b,h)\in PC'' \mid \langle g,vg\rangle\in si'$かつ $\langle g,vg\rangle\notin ssij \}$
$\cup\{g\cdot i\rightarrow(b,h) \mid g\rightarrow(b,h)\in PC'',\ \langle g,vg\rangle\in\cup ssij \}$
とする．

各si,Bi,PCi''に対して
$[\![si,\alpha\cdot Bi]\!]_{CTS}=\langle si\rangle\alpha[\![ti,B]\!]_{CTS}$
におけるtiと新しく作られたPCi''を得る．
この時，$t=t1\cup\cdots\cup tn\cup s0$
$PC''=PC1''\cup\cdots\cup PCn''$

(d) $B=B1+\cdots+Bn$ の時，
$[\![s,\alpha\cdot B]\!]_{CTS}$
$=\langle s\rangle\alpha([\![t1,B1]\!]_{CTS}+\cdots+[\![tn,Bn]\!]_{CTS})$
各tiは
$[\![s,\alpha\cdot Bi]\!]_{CTS}=\langle s\rangle\alpha[\![ti,Bi]\!]_{CTS}$
のtiである．

(e) $B=$ if E then B1 else B2 の時，Eの評価値が TRUE() の場合には B=B1，Eの評価値が FALSE() の場合には B=B2 の時と同じである．

(f) $B=B'[S]$ (Sはラベル付け替え演算)

の時，B=B'の時と同じである．□

直観的には，動作体の状態は動作体名の引数の値であり，動作体の内部に記憶されている資源であると解釈される．そして，その値は，動作体方程式の右辺に書かれている動作体名の対応する引数の値が更新される時のみ変化する．状態付通信木の各ノードに与えられている動作体の状態において，その中の対応番号列w は前の状態との資源の対応関係を表したもので，列の先頭から見て前の状態での対応番号列がw の部分列になっているものに対応し，資源の値がそのように変化したことを意味する．

以上のように，並行システムの仕様SPECに対して，その意味を表す状態付通信木が構成できる．

3．仕様記述例

CCS/ADTでの仕様記述例として，簡単なバッファー付データ転送システムの仕様記述を示す．このシステムは，1つの送出プロセス内に記憶されているデータ系列を全くそのまま，バッファーを通して別の 2つの受領プロセスに送るシステムとする．バッファーは送出プロセスと受領プロセスの共有資源として考えられ，バッファーに対して，送出プロセスは書き込み動作を，受領プロセスは読み出し動作を行う．バッファー内に送出プロセスによって書き込まれたデータは，2つの受領プロセスによって読みだされた後にバッファー内から消去される．

このシステムは次のような考え方で記述できる．まず自然に，送出プロセス，バッファー，2つの受領プロセスをそれぞれ動作体と考える．そして，各動作体で別の動作体との必要な通信動作を考える．次に，その通信動作が起きた後に，動作体内部に記憶されている資源がどのように変化するかを考え，変化が起きる時には必要な動作体名を考える．以上をもとに，動作体方程式を作り出す．

ここでは，送出プロセスを表す動作体を表現している動作体名を W，受領プロセスを表す動作体を表現している動作体名をR1，R2，その間のバッファーを持った制御プロセスを表す動作体を表現している動作体名を C，C1，C2とする．すると，CCS/ADT における仕様は以下の通りとなる．

```
SPEC = < ASPEC, BSEC , SYS >
ASEC = <Σ,X,Γ> , Σ= <S,F>
S = { data, queue }
F = { NEWQ: -> queue ;
      ADDQ: queue,data -> queue ;
      DELETEQ: queue -> queue ;
      FRONTQ: queue -> data ;
      UNDEF: -> data                }
X = { queue0: queue ;
      data0,data1: data    }

Γ = { DELETEQ( NEWQ() ) == NEWQ() ;
  DELETEQ(ADDQ(NEWQ(),data0)) == NEWQ() ;

  DELETEQ( ADDQ(ADDQ(queue0,data0),data1))
== ADDQ(DELETEQ(ADDQ(queue0,data0)),data1)
 ;
  FRONTQ( NEWQ() ) == UNDEF() ;
  FRONTQ(ADDQ(NEWQ(),data0)) == data0 ;

  FRONTQ(ADDQ(ADDQ(queue0,data0),data1))
== FRONTQ(ADDQ(queue0,data0))              }

BSPEC = <S',BE,Y,AN,E,PC>

S' = { data,queue }
BE = { R1,R2,C,C1,C2,W,NIL:queue ;
       SYSTEM:queue,queue,queue,queue }
Y = { q1,q2,q3,q: queue ;
      k,y,z: data          }
AN = { w,r1,r2: data }

E = { R1[q1]=r1?y·R1[ ADDQ(q1,y) ] ;
      R2[q2]=r2?z·R2[ ADDQ(q2,z) ] ;

  C[NEWQ()]=w?k·C[ADDQ(NEWQ(),k)] ;
```

```
  C[ADDQ(q3,z)]=w?k・C[ADDQ(ADDQ(q3,z),k)]
    +r1!FRONTQ(ADDQ(q3,z))・C1[ADDQ(q3,z)]
    +r2!FRONTQ(ADDQ(q3,z))・C2[ADDQ(q3,z)] ;
  C1[q3]=w?k・C1[ADDQ(q3,k)]
        +r2!FRONTQ(q3)・C[DELETEQ(q3)] ;
  C2[q3]=w?k・C2[ADDQ(q3,k)]
        +r1!FRONTQ(q3)・C[DELETEQ(q3)] ;

  W[ADDQ(q,z)]
=w!FRONTQ(ADDQ(q,z))・W[DELETEQ(ADDQ(q,z))];
  W[ADDQ(NEWQ(),z)]
=w!FRONTQ(ADDQ(q,z))・NIL[NEWQ()] ;

  SYSTEM[q1,q2,q3,q]
=(R1[q1]|R2[q2]|C[q3]|W[q])\r1\r2\w      }

PC = { (C,1)->(C1,1),(C,1)->(C2,1),
       (C1,1)->(C,1),(C2,1)->(C,1),
       (W,1)->(NIL,1)                    }
SYS = SYSTEM[NEWQ(),NEWQ(),NEWQ(),
             ADDQ(ADDQ(NEWQ(),10),20)]
```

ここでは，SYS は初期状態として送出プロセスにデータ系列10,20が記憶されているとする．SYS の内容を変えることにより，送出プロセスが任意のデータ系列を持っているデータ転送システムが記述できる．また，この仕様ではデータ系列をqueueと考えて記述している．

この仕様に対する意味を与えている状態付通信木は次の図1のようになる．

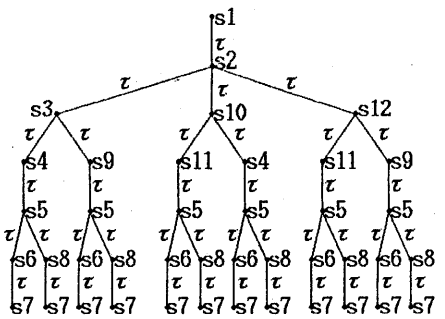


図1　状態付通信木

上図中のs1からs12はそのノードでの動作体の状態を表している．各状態は，対応番号列と値との組の集合で表されるが，この場合は対応番号列は同じものしか現れないので，次の記法を便宜上用いる．すなわち，

{ <1,v1>,<2,v2>,<3,v3>,<4,v4> }　の代りに，[v1,v2,v3,v4]　を用いる．さらに，ADDQ(…(ADDQ(NEWQ(),i)…),j) の省略記法として　(i,…,j)　を，NEWQ() の省略記法として　NQ　を用いる．すると，si (i=1,…,12) は次の通りとなる．

```
s1=[ NQ,NQ,NQ,(10,20) ]
s2=[ NQ,NQ,(10),(20) ]
s3=[ NQ,NQ,(10,20),NQ ]
s4=[ (10),NQ,(10,20),NQ ]
s5=[ (10),(10),(20),NQ ]
s6=[ (10,20),(10),(20),NQ ]
s7=[ (10,20),(10,20),NQ,NQ ]
s8=[ (10),(10,20),(20),NQ ]
s9=[ NQ,(10),(10,20),NQ ]
s10=[ (10),NQ,(10),(20) ]
s11=[ (10),(10),NQ,(20) ]
s12=[ NQ,(10),(10),(20) ]
```

以上のように簡単に，仕様は自然でかつわかりやすく記述できた．

4．CCS/ADTに対する評価

CCS/ADT では，プロセス間の通信動作のみをCCS の記述法で記述し，プロセス内部の逐次動作は抽象データ型の項として記述すればよく，我々の直観によく合った記述法になっていると思われる．また，意味として状態付通信木を与えたために，システムの内部で起こる通信動作にも陽に意味を与えることができる．

一方，状態付通信木の構成は，通信木に比べて複雑である点が欠点である．

5．まとめ

並行システムを代数的に仕様記述する方法として，Milnerの提案したCCS と抽象データ型の代数的仕様記述法を合わせた仕様記述法CCS/ADT を提案した．また，その意味論として，CCS のモデルとして提案された通信木を修正した状態付通信木を提案した．

今後の課題としては，状態付通信木間の関係より，仕様記述された並行システムの等価性を判定すること，この枠組を使っての並行システムの性質の検証方法の開発がある．

謝辞　日頃御指導賜わる豊橋技術科学大学本多波雄学長，東北大学阿曽弘具助教授，名古屋大学福村晃夫教授，平田富夫講師，並びに研究室の諸氏に感謝する．

文献

1) R.Milner, A Calculus of Communicating Systems , Springer LNCS Vol.92 (1980)
2) 稲垣，坂部，抽象データタイプの代数的仕様記述法の基礎 (1,2)，情報処理，Vol.25, No.1 , 5 (1984)
3) R.Milner, Synthesis of Communicating behaviour, Springer LNCS Vol.64 (1978)
4) E.Astesiano, G.F.Mascari, G.Reggio, M.Wirsing, On the parameterized algebraic specification of concurrent systems, Springer LNCS Vol.185 (1985)
5) J.A.Bergstra, J.W.Klop, Process algebra for synchronous communication, Information and Control 60 (1-3) (1984)
6) J.A.Bergstra, J.W.Klop, Algebra of communicating processes with abstraction, TCS Vol.37, North-Holland (1985)
7) 北,坂部,稲垣，プログラミング言語PL/0の代数的仕様記述，信学技報，AL84-56 (1985)
8) 高木,北,坂部,稲垣，入出力機能およびパラメタ付手続きを付加したプログラミング言語PL/0の代数的仕様記述，信学技報,AL85-44(1985)
9) R.Milner, Calculi for synchrony and asynchrony,TCS Vol.25,North-Holland (1983)