プロセス合成のための支援環境とその試作*

3G－3

臼井 伸幸　吉田 仙　木村 成伴　富樫 敦　白鳥 則郎†

東北大学電気通信研究所‡

1　はじめに

プロセスの代数的計算システムとしてMilnerのCCSなどが代表的であり，通信プロセスや並行プログラムなどを形式的に記述するために利用される他，検証問題などにも応用されている．その反面，プロセスの読解性や記述のしやすさなどについては難点がある．具体的な性質からこれらを満たすプロセスを獲得する手法は，これらの問題点を解決するだけではなく，曖昧性がなく厳密な通信プロトコルを得る基礎にもつながる．

そこで本稿では，μ-calculusによる論理式を性質とみなし，その性質を満たす再帰プロセスを合成するための支援環境を示し，その環境を実現したシステムを試作する．

2　準備

2.1　プロセスの代数的形式化

本システムで取り扱う再帰プロセスを以下のBNF表現によって定義する．

$$
\begin{array}{lll}
p ::= & 0 & (inaction) \\
| & a.p & (prefix) \\
| & p_1 + p_2 & (summation) \\
| & \mu x_j.\{x_i = P_i | 1 \leq i \leq n\} & (recursion)
\end{array}
$$

これらを用いた例として，$p = a.\mu x.(a.0 + b.x)$ の導出木を以下に示す．

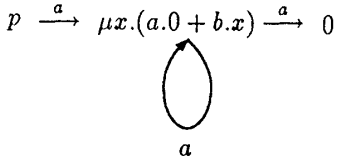

図 1: $p = a.\mu x.(a.0 + b.x)$ の導出木

2.2　μ-calclus

論理式Aは次のBNFで再帰的に与えられる．

$$
\begin{array}{lll}
A ::= & T & (\forall p \models T) \\
| & F & (\exists p \not\models F) \\
| & \langle a \rangle A & (p \models \langle a \rangle A \Rightarrow \exists p'.p \xrightarrow{a} p' \models A) \\
| & [a]A & (p \models [a]A \Rightarrow \forall p'.p \xrightarrow{a} p' \models A) \\
| & \nu x.f(x) & (p \models \nu x.f(x) \Rightarrow \forall g.p \models f(g), p \models g) \\
| & A_1 \wedge A_2 & (p \models A_1 \text{かつ} p \models A_2)
\end{array}
$$

これらのプロセスの意味は，厳密には「ラベル付き遷移システム」[3]によって定められるが，詳細は省略する．

これらの論理式を用いたプロセスの性質の記述例を紹介する．

(1) $p \models \langle a \rangle T$: p において a が動作可能

(2) $p \models [a]F$: p は a- デッドロックする．

(3) $p \models \nu x.\langle a \rangle x : p \models \langle a \rangle A \wedge \langle a \rangle \langle a \rangle T \wedge \langle a \rangle \langle a \rangle \langle a \rangle T \wedge \cdots$

2.3　事実の枚挙

プロセスの具体的な性質として考えた論理式から意図したプロセスと強等価なプロセスを合成するアルゴリズムとして，事実の枚挙[2]がある．このアルゴリズムは，意図するプロセスが満たすべき事実を入力すると，入力された事実の情報を使い，プロセスの骨格を徐々に構築していくというものである．アルゴリズムによって推測された現在のプロセスを p_c，この時点で入力された事実をAとすると，アルゴリズムは p_c がAを満たすように p_c を組織的に変形する．もちろん，変形され，新たに得られたプロセスは，A以前に入力された事実も満たさなくてはならない．そして，更に新たな事実を読み込み，以上の操作を繰り返していく．

3　プロセス合成支援システム

3.1　Prolog

このアルゴリズムを用いると，新たな事実が入力されたときにプロセスの作り直しを行う場合がある．このような処理を行うのに適したプログラミング言語として，Prologが挙げられる．Prologは記号処理，すなわち非数値演算用の論理型プログラミング言語であり，バック

*A Process Synthesis System and its Prototype

†Nobuyuki USUI,Sen YOSHIDA,Shigetomo KIMURA, Atsushi TOGASHI and Norio SHIRATORI

‡Research Institute of Electrical Communication, Tohoku University

トラックを行うことができる．この言語は対象と対象間の関係(ここでは論理式とプロセス)に関した問題解決に特に適している．また，Prologは木構造データを構築するのが容易であり，かつデータベースの操作を行うことができる．

これらの利点から本システムをProlog上に実装した．

3.2 システム構成

システム構成の概略を図2に示す．

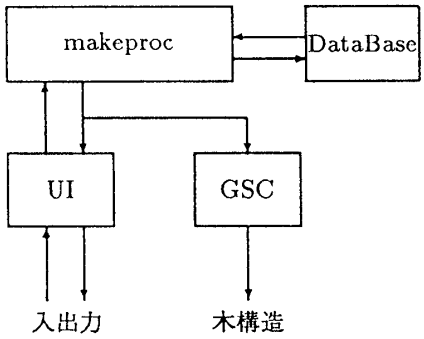

図 2: システム構成図

3.3 合成アルゴリズム

本システムでは，文献[1]で提案されたアルゴリズムを用いている．

3.4 プロセスの内部表現

プロセスを木構造を用いたリストとして扱うと，再帰を行う部分が不自然になる．そのため，プログラム内部では，プロセスを節の集合として扱っている．節は3項組 (c_i,Q,T) である．

c_i : プロセス定数

Q : 節が満たすべき条件

$T = \{(Act, c_j) | \forall c_i \xrightarrow{Act} c_j\}$

これによって木構造だけではなく，網構造の遷移関係を持つプロセスを扱うことができ，再帰プロセスを自然に扱うことが可能である．

3.5 インタフェース

入力は，論理式をキーボードもしくはファイルから逐次的に読み込む．論理式を1つ読み込む度にそれまでに入力された全ての論理式を満たすプロセスを出力する．

出力はプロセスを式として表示するだけではなく，GSCに出力を渡すことで遷移関係をグラフィカルに表示することができる．

4 実行例

実際に論理式をシステムに入力してみる．まず，$A_0 = \langle a \rangle T$ を入力すると p_0 が出力される．さらに $A_1 = \nu x.(\langle a \rangle \langle b \rangle x)$ を入力すると，再帰の部分が作られ，p_1 へと変形する．続いて，$A_2 = \langle b \rangle T$ を入力すると $b.0$ の枝ができるが，その後，$A_3 = [a][b][b]F$ という論理式を入力すると，それを満たすために再帰を展開して，p_3 に変形する．

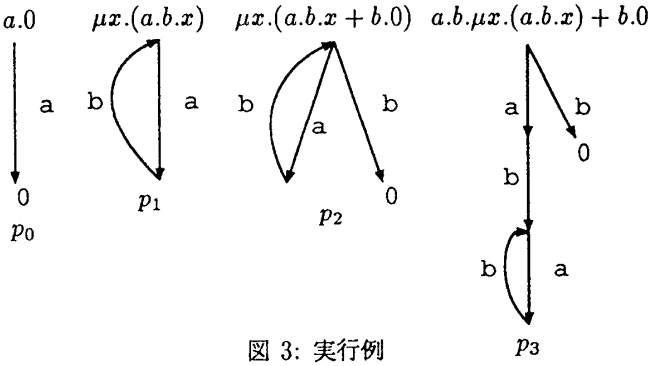

図 3: 実行例

5 まとめ

プロセスを合成するための支援環境を示し，実際に試作を作った．

今後の課題として，効率化や並行プロセス等への応用が挙げられる．

謝辞

本研究は一部、旭硝子財団、文部省科学研究補助金による援助を受けている。

参考文献

[1] 木村成伴, 富樫敦, 白鳥則郎: “Synthesis Algorithm for Recursive Processes by μ calculus”, 電子情報通信学会情報システムグループ *1993* 年度若手研究者のためのセミナー「計算理論・情報認識理解の研究動向と将来展望」資料 (1994).

[2] 木村成伴, 富樫敦, 野口正一 : “様相論理式による基本プロセスの合成アルゴリズム” , 信学論, **J75-D-I**, pp.1048–1061(1992).

[3] 富樫敦: “代数的プロセスの計算モデル”, 日本ソフトウェア科学会チュートリアルテキスト (1992).

[4] Ivan Bratko : “Prologへの入門”, 近代科学社 (1992).