TM-1031

ハイレベルネットによる
並列プログラムのモデル化

福澤　俊幸、長谷川　晴朗、
土田　佳寛（沖）

February, 1991

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191 ~ 5
Telex ICOT J32964

Institute for New Generation Computer Technology

ハイレベルネットによる並列プログラムのモデル化

A Modeling of Parallel Program by using High-Level Nets

福澤 俊幸　上田 佳寛　長谷川 晴朗
Toshiyuki FUKUZAWA　Yoshihiro UEDA　Haruo HASEGAWA

沖電気工業 (株)
Oki Electric Industry Co., Ltd.

1 はじめに

近年、コンピュータの技術の進歩に伴い多数のプロセッサから構成される並列処理計算機の開発が盛んに行われている。それに伴い、並列ソフトウェアを効率良く設計するための開発支援環境の構築が必要とされて来ている。筆者らはこのニーズに従い、並列ソフトウェア開発支援システムの開発を行ってきた [1]。

このシステムの1つのモジュールに、並列プログラムの静的解析を行うプログラム検証系がある。このモジュールでは、並列プログラムを純粋ペトリネットへモデル化することにより、ペトリネットが持つ構造解析手法を利用して、並列プログラムの動作等の解析を行っている [2],[3]。しかし、並列プログラムを純粋ペトリネットにモデル化した場合、変数やプロセス間通信等の表現を充分に行うことが出来ない。そこで、モデル化するペトリネットを純粋ペトリネットからハイレベルネットへ拡張することを試みた。

本報告では、ハイレベルネットを利用した、並列プログラムのモデル化、及び解析方法を論じる。

2 FGHC

本研究で解析の対象としている言語は、FGHC(Flat Guarded Horn Clauses)[4] である。FGHC は第一階述語論理に基づく論理型言語である。その最大の特徴は、並列実行を基本とするところである。FGHC は、言語に並列性を取り入れるため、ガード (Guard) という構文要素を論理プログラミングに導入している。このガードの導入により、プログラム中のプロセス間の同期、及び実行する (リダクション) 節の選択に関するプログラムの制御を実現している。

FGHC プログラムは、一般に、以下に示すガード付き節の有限集合として定義される。

$$H \ :- \ I \mid B.$$

ここで H は原始式, I, B は、原始式の連言である。ただし、I に含まれる原始式はシステム組み込み述語に限られる。演算子 "|" はコミット演算子と呼ばれている。FGHC の各節には、このコミット演算子が必ず含まれる。コミット演算子の左側をガード、右側をボディと呼ぶ。

プログラム実行時、節が実行 (リダクション) されるためには、総てのガードの条件が満たされることが、必要条件となる。同一の述語をヘッド部に持つ節が複数ある場合、どれか一つが選ばれる。複数の節のガード部の条件が、同時に満たされる時、どの節が選ばれるかは、非決定的である。また、各ゴールは、節中に記述する順序に全く関係なく独立したプロセスとして、並列に実行される。

ところで、FGHC のプログラムは、正規形 [5] と呼ばれる以下の形式に変換することが出来る。

$$H \ :- \ I \mid O \cup B.$$

O は、FGHC プログラムのボディ部の述語の中で、H の変数を単一化するような、システム組み込みの述語である。正規形に変換した場合、それぞれ H(ヘッド), I(インプットガード), O(アウトプットガード), B(ボディ) と呼ぶ。

3 Coloured Petri Nets

ハイレベルネットとは、個々のトークンの識別を行えるようなペトリネットである。トークンの色を利用して、発火シークンスを制御できるので、純粋ペトリネットに比べ表現力が、豊かになっている。

今回、FGHC のモデル化に使用するのは、ハイレベルネットの一種である Coloured Petri Nets(以下、CP-Nets)[6] である。

CP-Nets の特徴は、トークンに色を付けることにより遷移するトークンの関係を関数として表現している点、及びトークンの色に対する発火条件を評価式としてトランジションに定義している点である。

4 モデル化の規則

まず、FGHC プログラムを正規形に変換する。正規形は、FGHC のセマンティクスを研究するために考案された概念であるが、インプットプレースとアウトプットプレースの役割に着目すると、プロセス間通信の送受信の関係を明示的に表している。従って、FGHC プログラムを正規形に変換することにより、プロセス間通信のモデル化をネットの構造に取り入れることが容易になる。

FGHC を CP-Nets でモデル化するに当たり、以下の項目を前提に置いた。

- 純粋ペトリネットでモデル化を行った時の枠組みは残す
- プレース・トランジションの増加は最小限にする
- プログラム以外の要素をネットの構造に極力取り入れない

これらの項目を基に表 1に示す対応を定めた。

表 1: FGHC → CP-Nets 対応規則

FGHC プログラム		ペトリネット
節	⟹	トランジション
ヘッド	⟹	インプットプレース
ボディ	⟹	アウトプットプレース
通信用チャネル	⟹	プレース
組み込み述語	⟹	評価式
変数	⟹	トークン

上記の対応関係の中で、前3者は純粋ペトリネットでモデル化した時と同じ規則である。CP-Nets を利用することにより生じた対応関係を以下に説明する。

- プロセス間通信用チャネル
  プロセス間通信を表現するために、プロセス間通信の媒介となる変数にプレースを一つ対応させている。このプレースは、プロセス間通信時の送信プロセス (と成る節) に対してはアウトプットプレースに、受信プロセス (と成る節) に対してはインプットプレースとなる。この対応付けは、FGHC のプロセス間通信は共有変数の単一化を仲立ちに行われる点と、メッセージ受信プロセスは変数の単一化が行われるまでサスペンドしている点で、プログラムの動作と一致している。以上の点からこの対応は、FGHC のプロセス間の同期制御機構と一致している。

- 組み込み述語
  組み込み述語は評価式と対応させている。従って、モデル化された CP-Nets の評価式は、FGHC の組み込み述語の仕様と同じ振る舞いをしている。プロセスの動作的立場からみると、組み込み述語は、変数から真偽の判定を行い、それによりプロセスの動作制御の一端を担っているとみなせる。このことは、トークンが評価式に合わない (組み込み述語の結果は偽) 場合は、トランジションは発火しない (節が選択されない) ことと一致する。以上の理由によりユーザ述語

とモデル化時に区別をしている。インプットガード,アウトプットガードは、組み込み述語から成り立つので、評価式として表現される。

- 変数
  FGHC の変数の設定・参照は、第一階述語論理の単一化の概念に基づいている。従って、トークンに対しても単一化に基づいた扱いをしている。

図1に変換イメージを示す。

```
p(X):-true|q(X),r(X).      %1
q(X):-true|X=a.            %2
q(X):-true|X=b.            %3
r(X):-X=a  |true.          %4
```

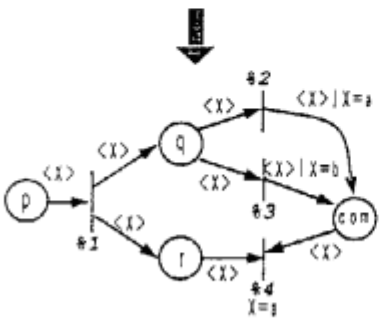

図1: FGHC-ペトリネット変換の例

このプログラムの意味を以下に記す。

プロセスpは、2個のプロセスq,rを生成する。プロセスqは、節選択の非決定性から節%2と節%3のどちらが選ばれるかは、特定できない。この場合、変数Xにどの定数が単一化されるかは、選択される節に依存する。プロセスqとプロセスrは変数Xを媒介としてプロセス間通信を行っている。プロセスqはメッセージの送信を行い、プロセスrはメッセージの受信を行う。プロセスrは、変数Xに定数aが単一化されている時に成功するので、プロセスqとして節%2を選択された場合プログラムは、成功する。

CP-Netsではプロセスp,qのプロセス間通信を、通信用のプレースcomを介してトークンXを渡すことで、表現している。節%2,%3のアウトプットガードは、トランジション%2, %3のトークンの評価式・$<X=a>$と$<X=b>$として表現されている。これらの評価式は、トークンの色をa,bに代入する(単一化)ことを示している。また、プロセスrのインプットガードは、トランジション%4の評価式・$<X=a>$として表されている。これはトークンXの色がaの時、トランジションが発火することを示している。

5 FGHCプログラムの解析

モデル化された、CP-Netsの発火系列を求めることにより、プログラムの解析を行う。純粋ペトリネットでは、接続行列A,初期マーキングM,目的マーキングM'を与えることにより、トランジションの発火回数xを行列方程式、

$$M' = M + xA \qquad (1)$$

を利用して求める。

CP-Netsの場合もアークの多重度を要素とする接続行列(行列の要素は整数)を定義できる。ただし、ここから得られる発火系列は、トークンの色を考慮されていない。得られた発火系列が評価式を満たすことを確認することにより、プログラムとして成功する発火系列を求める。

FGHCプログラムの節のリダクションとトランジションの発火系列は、対応付けが出来るので発火系列の情報からプログラムの動作に関する情報を得ることが出来る。

図1の例を利用してプログラムの解析例を記す。

接続行列は、次のようになる。

$$A = \begin{pmatrix} -1 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & -1 \end{pmatrix}$$

ただし、行は%1,%2,%3,%4のトランジションを順に、列はp,q,r,comのプレースを順に意味している。

$p(X)$をゴール節に与えた場合、初期マーキングは、$M = (1,0,0,0)$で与えられる。プログラムが成功して終了する場合を解析する場合、目的マーキングは、$M' = (0,0,0,0)$となる。これを(1)の行列方程式に代入すると解として、$x_1 = (1,1,0,1)$と$x_2 = (1,0,1,1)$が得られる。

2個の解x_1,x_2は、CP-Netsが、初期マーキングから目的マーキングへ遷移する時のトランジションの発火回数である。発火系列が存在すれば、これらの解を利用して発火系列を見つけることが出来る。

まず、トークンの色を考慮しない発火系列を求める。解として、$\mu_1 = \{\%1, \%2, \%4\}$, $\mu_2 = \{\%1, \%3, \%4\}$の二つの系列が得られる。これらの系列は、成功するとは限らないが、プログラムを実行した時に通る可能性が有る系列である。FGHCは、Prologが有するバックトラックの機能はないので、実行する可能性が有る系列を求める事は充分に意味がある。

次に、トークンの色を考慮して発火系列を見つける。例の場合トークンは、Xのみなのでこのトークンの色に着目する。μ_2の系列は、節%3が発火する時に評価式から、トークンXに色bを単一化している。これは、節%4の評価式の条件と一致しないので、μ_2の系列は失敗することが分かる。逆に、μ_1の系列は、節%2が発火する時に評価式から、トークンXに色aを単一化している。これは、節%4の評価式の条件と一致している。従って、μ_1の系列は、成功することが分かる。

以上の事から、次の様に考察が出来る。図1のプログラムは2通りの実行系列$\mu_1 = \{\%1, \%2, \%4\}$と$\mu_2 = \{\%1, \%3, \%4\}$を取る可能性がある。ただし、プログラムとしては、$\mu_1$の系列の場合は成功し、$\mu_2$の系列の場合は失敗する。

6 まとめと課題

本報告では、FGHCプログラムをCP-Netsにモデル化する方法と、モデル化されたCP-Netsの構造を解析することによりプログラムの動作に関する解析が行えることを示した。モデル化に関しては、CP-Netsを利用することにより、プログラムの動作関係を自然に表現できるようになった。解析に関しては、トランジションの発火系列を求めることにより、プログラムの動作情報を得ることを示した。

今後の課題としては、適用できるプログラムの範囲を拡張することが挙げられる。現方式では、同一変数を介してのプロセスの相互通信・再起呼び出しに対するモデル化が充分でない。また、規模の大きなプログラムに対しては、ネットが大きくなり接続行列を利用する解析を行うことが出来ない。これらの問題に対応するには、トークンに構造を持たせる,ネットを分割して解析を行う方法等が考えられる。以上の問題を解決するために今後も研究を続けていく。

謝辞 本研究は第五世代コンピュータ・プロジェクトの一環として行っている。日頃御指導を頂くICOT研究部長代理長谷川隆三氏に感謝します。

参考文献

[1] 本城 他: GHCプログラムの設計支援に関する一考察, 情処研報 90-ARC-83-8, 1990.

[2] 長谷川 他: ペトリネットを利用した並列プログラムの解析, IEICE 第3回 CAS 軽井沢 WS, pp.199-206, 1990.

[3] 前田 他: ペトリネットによる並列プログラムの動作解析に関する一考察, 第40回情処全大, 2R-3, 1990.

[4] K.Ueda.: *"Guarded Horn Clauses"*: Tech. Report TR-103, ICOT, 1985.

[5] K.Ueda,K.Furukawa.: *"Transformation Rules for GHC programs"*, Proc. of FGCS'88, ICOT, 1988.

[6] K.Jensen: *"Advances in Petri Nets, Part1"* in Lecture Notes in Computer Science, pp.248-299, Springer-Verlag, 1986.