ペトリネットのＡｄａプログラム開発への適用

6D－1

河合一夫
ＭＨＩエアロスペースシステムズ（株）

１．背景

Ａｄａの言語仕様には、Ａｄａタスクの動作を処理系に依存している部分が存在する［１］。そのため、Ａｄａを使用したマルチタスクプログラムを作成した場合、そのプログラムが動作する環境（言語処理系及びＯＳ）によって、タスクの動作結果が違ってくる可能性がある。

２．導入

本稿では、Ａｄａタスクの動作をシミュレーションする方法としてペトリネットによるものを提案する。そして、プログラムの動作環境の違いをそのシミュレーションの中で容易に取り扱うことが出来ることを示す。

３．モデル化

３．１　モデルの概要

Ａｄａタスクの状態に着目し、Ａｄａタスクの実行順序を、その状態遷移の移り変わりを捉えることと考えペトリネットを作成する。また、ペトリネットのサイズを抑える為に色付きペトリネットを用いる。

３．２　モデルの詳細

Ａｄａタスクの状態遷移には、処理系に依存している部分が存在していることは既に述べた。そこで、２種類のペトリネットを使用し、それらの協調動作によりシミュレーションを行う。一つは、対象となるプログラムのタスク構造に基づき、タスクの動作をシミュレーションする為のもの（以下、構造ネットと呼ぶ）であり、もう一つは、処理系に依存した状態遷移を行う為のもの（以下、遷移ネットと呼ぶ）である。この項では、まず本稿で使用するペトリネットの定義を与え、次に、構造ネットと遷移ネットの説明を行い、最後に２つのペトリネットの協調動作について説明する。

３．２．１　ペトリネットの定義

本稿で使用するペトリネットの定義を以下に示す。色付きペトリネット（以下ＣＰＮ）の定義は、

$$CPN=(P, T, C, I-, I+, M_0) \quad (1)$$

である［２］。

(1)Ｐ及びＴはそれぞれプレースの有限集合、トランジションの有限集合を表す。

(2)$P \cap T = \phi$ 及び $P \cup T \neq \phi$

(3)Ｃは、各プレースに対しては可能なトークンの集合を対応づけ、各トランジションに対しては可能な発火カラーを対応づける関数である。トークンは＜属性$_i$，・・，属性$_j$＞と表す。

(4)Ｉ－とＩ＋は、入出力関数を表す。

(5)M_0は、初期マーキングを表す。

３．２．２　構造ネット

構造ネットは、タスクの構造により構築されるもので、そのペトリネットの定義は（１）式に従う。また、その解釈を以下に示す。

(1)プレースは、タスクの状態遷移に影響を及ぼすステートメントやブロックを示す。例えば、ａｃｃｅｐｔ文やエントリコールなどである。

(2)トランジションは、状態遷移及びそれに伴う処理を表す。

(3)トークンを、＜タスク$_i$，状態$_j$＞と表す。これは、一つのトークンが一つのタスクを表すことを意味する。これによって、タスクタイプなど同一のコードによって複数のタスクが生成さ

Application of Petri net to Program Development with Ada
Kazuo Kawai
MHI AERO SPACE SYSTEMS CORP.

れる場合にネットをタスク分持つ必要がなくなる。ここでの状態は、各処理系に依存しない言語仕様から規定される状態値を定義する。

(4)入出力関数は、トークンの属性であるタスクの状態を変化させる関数などを定義する。

(5)初期マーキングは、シミュレーション開始時の各タスクの実行位置と状態を表す。

３．２．３　遷移ネット

遷移ネットは、ターゲットである処理系のスケジューリング方式に従って構築される。遷移ネットの定義は（１）式に従う。また、その解釈を以下に示す。

(1)プレースは、タスクの状態を表す。例えば、実行状態や資源待ち状態などである。但し、遷移ネットに於ける状態は、各処理系で規定される状態値を表す。遷移ネットでの状態は構造ネットの状態にマッピングされる。

(2)トランジションは、状態の遷移を表す。

(3)トークンは、＜タスクｉ，プライオリティｊ＞と表す。必要ならば処理系に依存する属性を追加する。

(4)入出力関数は、処理系に依存した状態遷移を行う為の関数などを定義する。例えば、プライオリティに従ったタスクスケジュールに対応する関数などである。

(5)初期マーキングは、シミュレーション開始時のタスクの状態とプライオリティの値を表す。

従って、動作環境の違いは、遷移ネットを組み替え、構造ネットの状態とのマッピングを取ることで対処可能となる。

３．２．４　ネット間の関係とネットの実行

構造ネットと遷移ネットは、それぞれのトークンの共通属性である＜タスクｉ＞により結び付けられる。従って、それぞれのネットの実行結果は、＜タスクｉ＞を通じて反映される。その際、構造ネットの属性である＜状態ｊ＞と、遷移ネットのプレースが表している状態は、ある規則により対応関係が取られており、その関係に従って反映する。ネット間の関係を図３．２．４－１に示す。

まず遷移ネットの発火可能なトランジションを全て発火させ、各タスクの状態を決める。その結果を、構造ネットの各トークンの状態に反映する。次に構造ネットの発火可能なトランジションを全て発火させる。そして、構造ネットの各トークンの状態に従って遷移ネットの各トークンの位置を変化させる。この２つのネットの発火により単位時間を”１”進める。これにより、疑似的な並行動作をシミュレートすることが出来る。以降、この動作を繰り返し、構造ネットのトークンの発火列を調べることにより、対象とするタスクの動作を解析する。

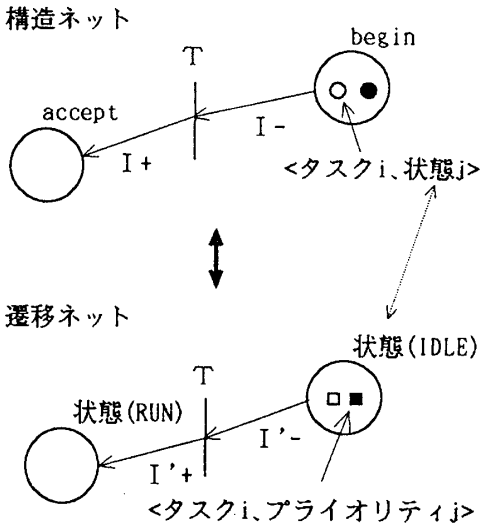

図３．２．４－１　ネット間の関係

４．おわりに

本稿では、タスクの動作シミュレーションの方法としてペトリネットを使用する方法を示した。現在は、プロトタイプツールを使用し、本稿で示したモデル作成方法に従って種々のＡｄａプログラムの動作の確認をしている所である。今後、有効性や課題／問題点を明確にして行きたい。

【参考文献】

[1]United States Department of Defense :Reference Manual for the Ada Programming Language(ANSI/MIL-STD-1815A), Jan. 1983

[2]椎塚:実例ペトリネット, コロナ社, 1992