計測自動制御学会　東北支部　第245回研究集会（2008.10.24）
資料番号　245-11

# フラクタル・ペトリネット・システムのアーキテクチャに関する研究

## Architecture of Fractal Petri Net System

○ 大橋　直子*、渡部　慶二*、村松　鋭一*、有我　祐一*、遠藤　茂*
○ Naoko Ohashi*, Keiji Watanabe*, Eiichi Muramatu*,
Yuichi Ariga*, Shigeru Endo*

*山形大学大学院理工学研究科

*Yamagata University
キーワード：ペトリネット（Petri Net）、フラクタル性（Fractal）
連絡先：山形県米沢市城南４－３－１６

## 1. はじめに

今日、離散事象システムの大規模化・複雑化に伴う問題として以下の７つが挙げられる。

・効率が悪い
・脆弱である
・開発期間の長期化
・開発コストの上昇
・動作把握が悪い
・保守性が悪い
・拡張、変更が困難

本研究の目的は上記の問題を解決する一方法を提案する。具体的には以下の方法で進める。

・解決原理の確立
・問題と解決原理の記述
・解決原理を実現する基本構造の構築
・実現可能性の検証

## 2. 問題の解決原理

前述した７つの問題の解決原理として、以下の４つを挙げる。

・視覚性
・最適性
・ロバスト性
・フラクタル性

視覚性は『動作把握が悪い』『保守性が悪い』という問題の解決原理、最適性は『効率が悪い』という問題、ロバスト性は『脆弱である』という問題の解決原理である。そして、フラクタル性は７つ全ての問題の解決原理である。

フラクタル性とは自己相似性とも呼ばれ、ひとつの図形をそれ自身の中に含ん

でいる事、逆に言うとひとつの図形を組み合わせた全体の図形が元の図形との相似である。離散事象に対しては基本要素の組み合わせがひとつの基本要素になること、あるいは基本要素の中に基本要素の中に入れることができることを言う。

これにより、大規模システムを部分的にまとめ、把握しやすくできる。変更、追加が容易にできる。

## 3. 解決原理を実現する基本構造の記述

離散事象システムと、問題の記述および解決原理をひとつのかたちで記述する必要がある。その方法がペトリネットである[(1)]。

ペトリネットとは図１に示すようにプレース・トークン・アーク・トランジションの４つから構成されるネットである。例を図２に示す

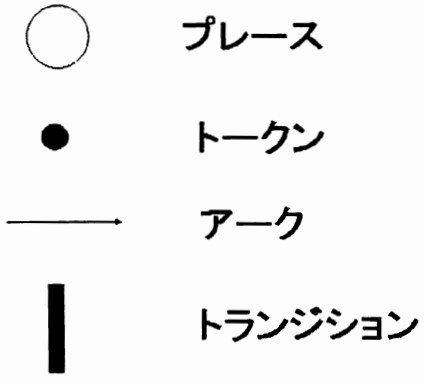


図１　ペトリネットの要素

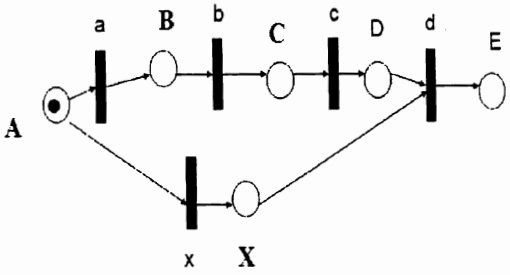


図２　ペトリネットの例

ペトリネットには図３に示す発火規則がある。入力プレースにトークンがあるとき、そのトークンを取り去りトランジションが発火させる。発火とは指定された作業を行うことである。発火終了後、出力プレースにトークンを入れる

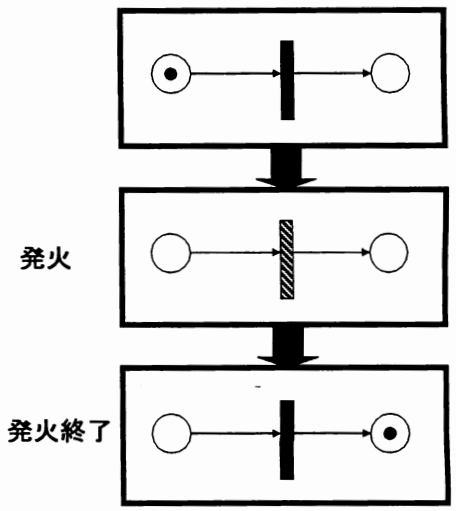


図３　ペトリネットの発火規則

ペトリネットにはプレースに複数のトークンを許す場合やアークに重みをつける場合があるがここでは離散事象のみを扱うのでひとつのプレースに重みは１のセーフティネットとする。

ペトリネットは上記の事象の推移の他に

図４のAND、図５のORの機能もある。

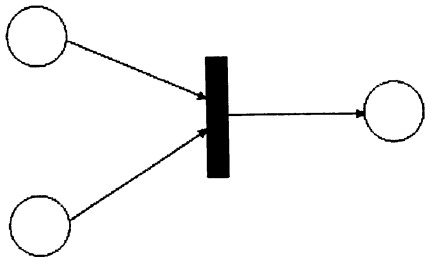

図４　AND

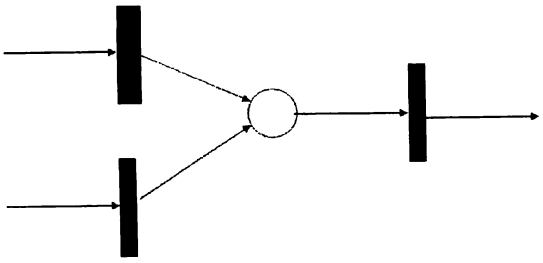

図５　OR

注意しなければならないのは図６のようにひとつのトークンに対して二つのトランジションが関わる場合である。一方が発火するともう一方が発火できない。これを競合という。競合がある場合、発火の優先順位や条件を考慮する必要があり、事象の分岐に使用することもできる。

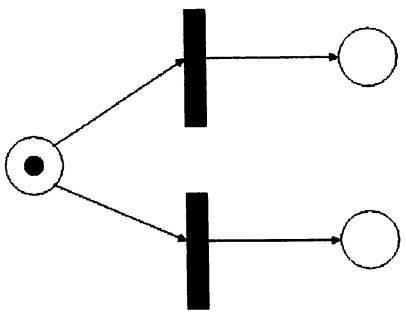

図６　競合

## 4. 実現可能性の検証

（１）離散事象の記述

たとえば、AからB、BからC、CからDに推移する離散事象を図２のように示すことができる。

また、仕事ａの後で仕事ｂを行い、仕事ｃ、仕事ｄと行くときも図２のように示すことができる。

（２）視覚性

トークンの位置によってシステムの状態がわかる。

また、図７のように発火したトランジションをカラー表示することによってどこの仕事が行われているかが確認できる。

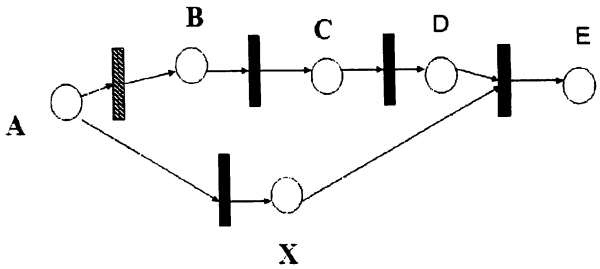


図７　視覚性

（３）最適性

図２のようにAからEへ行くときに複数の経路がある場合、トランジションの少ない経路を選ぶのが最適性である。これは分岐を使ってできる。

（４）ロバスト性

スタートからゴールへの道が複数あり、最適状態で動かしているときに図８のように故障が生じた場合、別の経路があるため、AからEの作業を継続することができる。

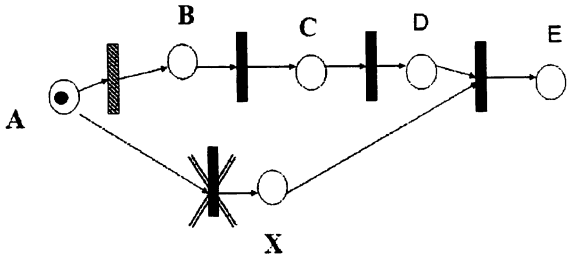


図８　ロバスト性

（５）フラクタル性

プレース、トランジションの組み合わせを水平・垂直につなぐことができる。たとえば、図９のようにトランジションの中にネットを入れることができる。

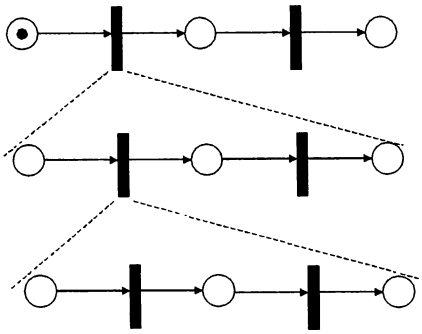

図９　フラクタル性

## 5. フラクタル・ペトリネットシステム

フラクタル・ペトリネット・システムとは、ペトリネットのフラクタル性を用いることによって水平・垂直な構造を持つシステムである。たとえば、図１０のように置き換えや保守、拡張が容易であり、安全性が確認されたパーツを組み込んで効率の向上やトラブルの低減を図ることができる。

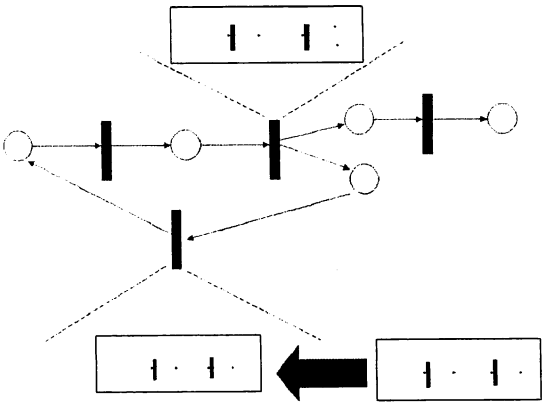

図 10　置き換え・保守・拡張

## 6. フラクタル・ペトリネット・システムの基本構造

フラクタル･ペトリネット･システムの構築のために次の４つの構造を考える。

- 木構造
  システム全体の構造を示す
- データ構造
  プレース・トランジション・アークの情報とそれぞれの関係を示す
- 制御構造
  発火の流れを制御する
- 通信構造
  ＰＣ間のデータの流れを制御する

以下では木構造とデータ構造について詳細を示す。

（１）木構造

図１１のように全体の構成を表すものである。

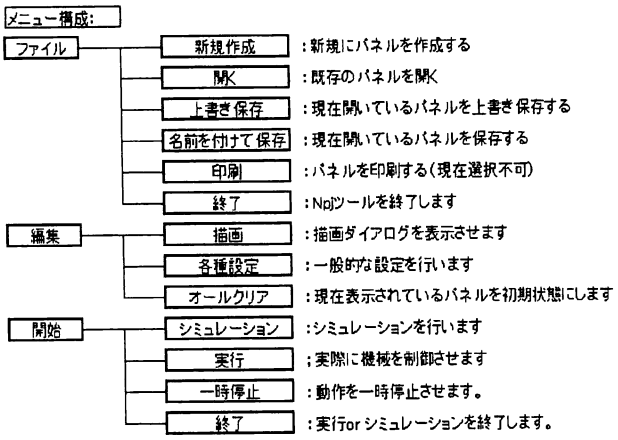


図１１　木構造

（２） データ構造

表示画面と仮想画面

表示画面を 24 ピクセルごとに分割したものを仮想画面とする。たとえば、水平方向をX、垂直方向をYとしてX=２０、Y=１０の点はx=０、ｙ＝０として登録する。

0≦X≦23、0≦Y≦23 の点（X,Y）は仮想画面で（x, y）=（0, 0）となることを意味する。

プレースやトランジションの表示は表示画面の 24*24 ピクセルの枠の中で描き、その位置（x, y）をプレースやトランジションの登録に使う。

プレース

プレースは図１２に示すように一つの構造体として登録し、その中に座標(x,y)とトークンの有無のデータを入れる。構造体同士は fwrd, next でつなぐ。

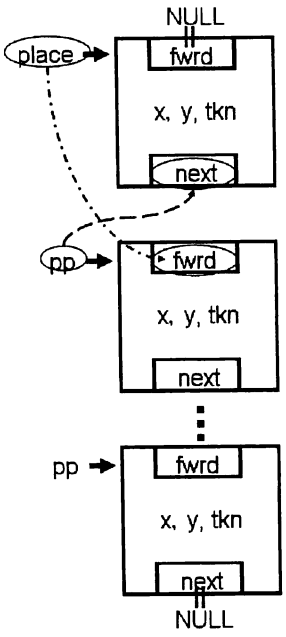


図１２　プレースのデータ構造

トランジション

トランジションも構造体として登録し、入力プレースの情報を持つ input と出力プレースの情報を持つ output と座標（x,y）のデータを入れる。構造体同士は fwrd, next でつなぐ

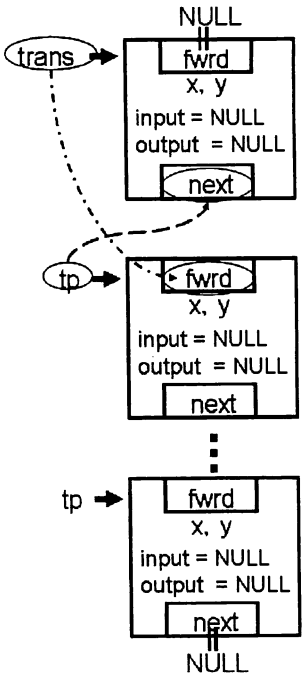

図１３　トランジションのデータ構造

アーク

アークは図 １４に示すように二種類の構造体を持ち、一つはアークの始点・中点・終点の座標（x, y）を記憶する構造体 ap で始点から終点に向けて構造体同士は next でつなぐ。もう一つの構造体 al は始点の座標（x, y）とポインタ ap を記憶する構造体であり、構造体同士は fwrd, next でつなぐ。

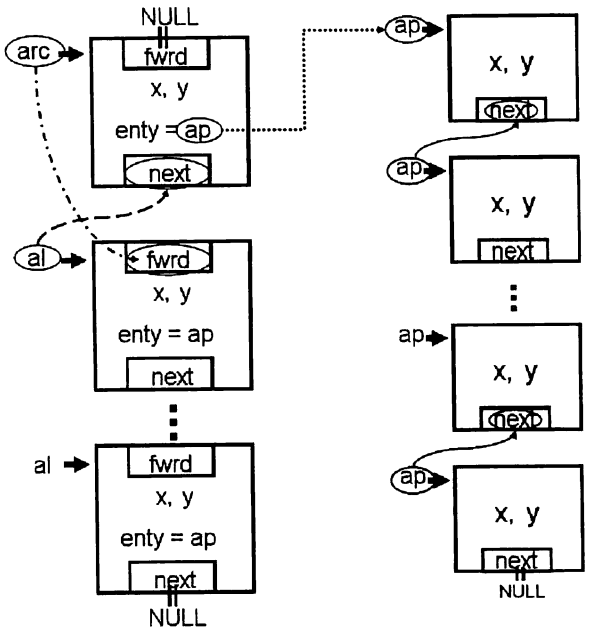


図１４　アークのデータ構造

入出力関数

入出力関数はトランジションの入力プレースと出力プレースのつながりをあらわす関数（構造体）である。

まず、アークの構造体 al 中の始点のポインタ ap から始点の座標（x, y）を呼び出す。たとえば始点がプレースとしたとき、終点の座標（x, y）がトランジションの座標であるか確認する。プレースのポインタ pp を記憶する構造体 fp をつくり、トランジションの構造体 tp の input に fp のポインタを記憶する。これで入力プレースとトランジションをつなぐことができる。（図１５）

始点がトランジションで終点がプレースの場合にもプレースのポインタ pp を記憶する構造体 fp をつくり、トランジションの構造体 tp の output に fp のポインタを記憶する。これでトランジションと出力プレースがつながる。

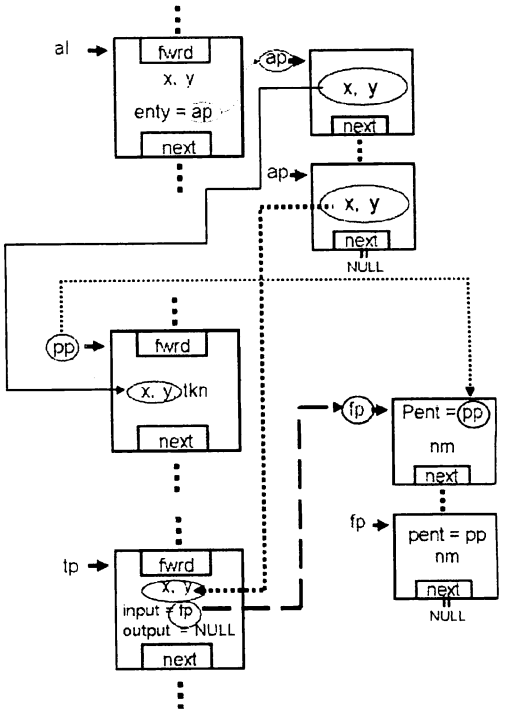


図１５　入出力関数のデータ構造

この構造を元にフラクタル構造、制御構造、通信構造を考えていく。

## ７．スーパーバイザー

ネットの動作部の解析・監視をするためにスーパーバイザー機能を置く。

スーパーバイザーには以下の機能を持たせる。

・　可到達性

図１６に示すようにスタートからゴールまで発火継続できるかを調べる。

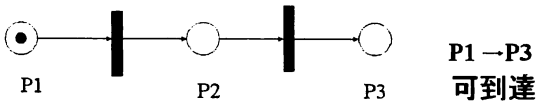


図１６　可到達性

・ 最適性

図１７のように複数の経路がある場合に最短経路を選ぶことである。

・ ロバスト性

複数の経路がある場合に一つの経路にトラブルがあった場合に別の経路を選び、全体の動作を止めないことである。たとえば、図 17 で最短・最適の経路 AXB にトラブルが生じたとき経路 ACB を選ぶことである。

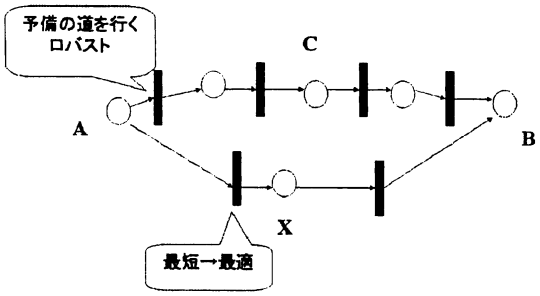


図１７　最適性・ロバスト性

・ デッドロックでトークンが動かなくならないように調べる

たとえば、図 18 のように左下のトランジションが発火するとネットからトークンが無くなり、動作がとまってしまう。

このような構造になっていないかをスーパーバイザーで調べる。

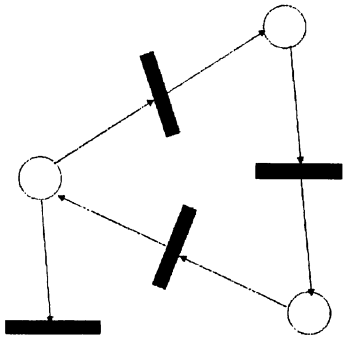

図 18　デットロック

・ 実行時の故障検出

トランジションが発火終了しない場合に故障とみなし、警告を鳴らす。

## ８．おわりに

大規模システムの問題の解決策の一方法としてフラクタル・ペトリネット・システムを提案した。

今後の課題は基本構造（制御構造・通信構造）の詳細の構築と実際のソフトウエアの作成である。

## ９参考文献

（１）椎塚久雄　実例ペトリネット　コロナ社