修 士 論 文

エージェント指向ペトリネットの解析について

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

岡橋 孝治

2003 年 3 月

修 士 論 文

エージェント指向ペトリネットの解析について

指導教官　平石邦彦 助教授

審査委員主査　平石邦彦 助教授
審査委員　金子峰雄 教授
審査委員　浅野哲夫 教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

110026 岡橋 孝治

提出年月: 2003 年 2 月

目 次

第1章　はじめに

マルチエージェントシステムとは，ある環境の下で複数のエージェントが自律的かつ協調的な行動によって分散的に問題を解決するシステムである [1]．このシステムは，ロボット，自動搬送車，加工・組立て機械などをエージェントとしたフレキシブル生産システムや，プロトコルを介してデータを交換する通信システムなど，多くの分野での応用が期待できる．

このようなシステムを，ペトリネットを用いてモデル化することを試みると，カラーペトリネット [2] のような高水準ペトリネットを用いれば，マルチエージェントシステムの動作を表現することは可能である．しかし，カラーペトリネットではデータ構造を持つトークンは扱えるが，データ構造とそれを操作するための方法をカプセル化したオブジェクトの概念は取り入れられていない．よって，カラーペトリネットでは各エージェントを独立したものとして記述することが難しく，あるエージェントの記述を変更しようとすると，それと同時に環境も変化するので他のエージェントの記述にも影響を与え，記述の変更すべき範囲が広くなるので，システムの変更を容易に行うことができない．

このようなシステムのモデル化への問題点を解決するために，ペトリネットの拡張モデルとしてオブジェクト指向ペトリネットやエージェント指向ペトリネットが提案されている [3, 4, 5, 6, 7, 8]．これらはトークンを単なるデータとしてではなく，データ構造とそれを操作するための方法をカプセル化したオブジェクトとして考えている．このトークンをエージェントに対応させることにより，環境ネットの構造と各エージェントの動作をそれぞれ独立して記述できるので，エージェントの記述の変更が環境や他のエージェントに影響を与えるという問題を解消できる．また，エージェント指向ペトリネットでは，実際にエージェントシステムで行われているエージェントの動的な通信リンクの変更や，エージェントの複製や消滅も容易に表現できる．

このような特徴をもつエージェント指向ペトリネットには初等的なモデルとしてPN^2(*Petri Nets in a Petri Net*)[9, 10] がある．PN^2 は，エージェントの動作をモデル化した下位階層のエージェントネットと，エージェントの動作可能な環境をモデル化した上位階層に当たる環境ネットの２階層のペトリネットで構成されている．PN^2 では，環境ネットのトークン自体がペトリネットで記述されており，そのトークンがエージェントネットである．PN^2 でのトランジションの発火は，環境ネットとエージェントネットに共通するトランジションが同期をとって行われる．PN^2 では，このような環境とエージェントの動作の相互作用を表すトランジションの発火を繰り返すことによってマルチエージェントシステムの動作を実現している．

エージェント指向ペトリネットの既存研究の多くは，マルチエージェントシステムをいかにモデル化するかという対象の記述に関するものであり，モデルの解析の面についてはほとんど研究が行われていなかった．そこで，本研究では PN^2 の解析の面に焦点を置き，PN^2 の効率的な解析手法について述べていく．

まず最初に，PN^2 の環境ネットは変えずに，与えられているエージェントネットの動作の記述を縮約しても，そのエージェントに与えられている動作と同じ動作を行えるようなエージェントネットの縮約構成法について述べる．なお本研究では，有限オートマトンで表現できるようなエージェントネットを対象とする．エージェントネットの縮約によりエージェントネットの状態数×遷移の数で決定される接続行列のサイズを小さくできるので，接続行列を用いた既存の PN^2 の解析手法をより効率的に適用することができる．

有限オートマトンで表現できるようなエージェントネットについては本論文で述べる縮約法によってまず縮約を行っておき，次にそのようなエージェントネットで構成される PN^2 に対して，PN^2 の動作を保存する１階層のペトリネット PN への変換を行う．PN^2 よりも PN の方が解析の面においては進んでいるので，PN に対する既存の解析手法を用いて PN^2 の性質を解析することが可能になる．

本論文の構成について説明する．２章ではまず PN^2 の動作を例を用いて説明し，つぎに PN^2 で表現できるエージェントシステムの性質について述べる．３章では PN^2 上のエージェントネットの動作の記述を縮約する方法について例を用いながら説明する．そして４章では，与えられた PN^2 から１階層のペトリネットへの変換方法について述べる．５章では，まとめと今後の課題について述べる．

第2章　PN^2について

2.1　エージェント指向ペトリネットPN^2

PN^2は，エージェントの動作をモデル化した複数のエージェントネット (下位ネット) と，そのエージェントの動作可能な環境をモデル化した１つの環境ネット (上位ネット) の２階層のペトリネットで構成されており，環境ネットのトークンがエージェントネットに相当する．図 2.1 のPN^2の例では，μ_1とμ_2がエージェントネットに対応している．

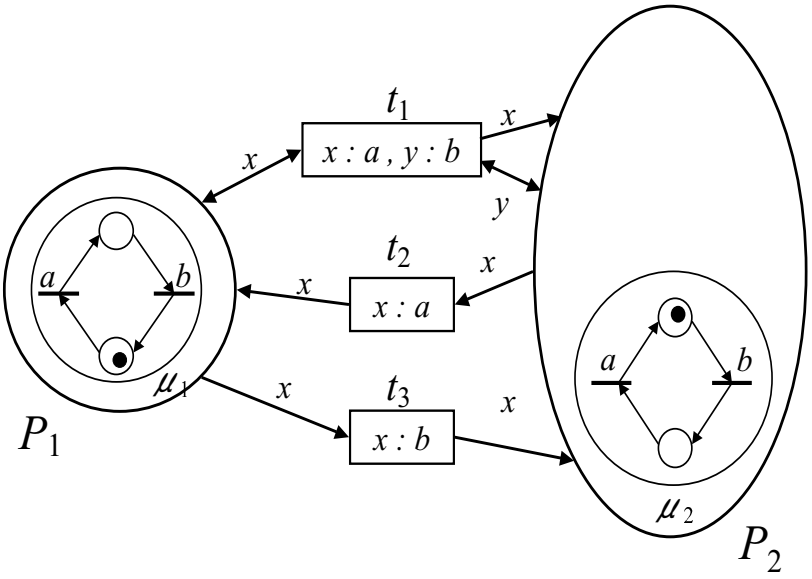

図 2.1: PN^2の例

PN^2では環境ネットと，エージェントネットのトランジションが同期して発火する．環境ネットのトランジションは１つ以上のトランジション成分で構成されている．例えば，図 2.1 のt_1でのトランジション成分はxとyの２つである．トランジション内に書かれた$x : a, y : b$はt_1と同期して発火すべきエージェントネットのトランジションを表す．環

境ネットのトランジションが発火可能となるためには，同期して発火すべきエージェントネットのトランジション成分がすべて発火可能である必要がある．例えば図 2.1 の PN^2 のマーキングでは，トランジション t_1 について，t_1 のトランジション成分 x に対する μ_1 の a，y に対する μ_2 の b が共に発火可能であるので，t_1 は発火可能である．t_1 が発火すると，それと同時に μ_1 がトランジション a を発火し，また μ_2 がトランジション b を発火してそれぞれのエージェントネットが移動し，図 2.1 から図 2.2 のようにマーキングが変化する．ここで，t_1 のトランジション成分 x について，入力数が 1 に対して出力数は 2 であるので，エージェントの数が 1 つ増えている．

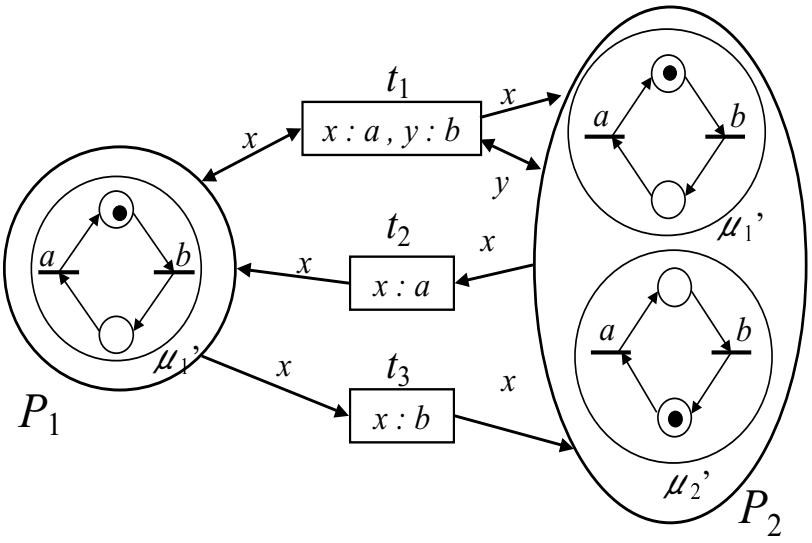

図 2.2: 図 2.1 の t_1 の発火後のマーキング

2.2 PN^2 が表現できるエージェントシステムの性質

PN^2 では，マルチエージェントシステムのもつ以下の性質を表現できる．

- **静的／動的な環境**
  各エージェントが可能な動作は，環境 (物理的制約や他のエージェントの存在) により制限される．また，エージェントは環境に作用するので，エージェントの動作によって環境も部分的に変化を繰り返していく．PN^2 での環境とは，環境ネットの構

造と環境ネット上のエージェントネットの配置およびエージェントネットの動作の可能性 (エージェントに与えらている動作命令) により表現されている．

- **エージェントの移動**
  環境ネットのプレースを，エージェントが存在する物理的な場所に対応させることにより表現できる．

- **エージェントの複製と消滅**
  入力数よりも出力数の方が多いトランジション成分で構成される環境ネットのトランジションの生起により，エージェントの複製を表現できる．また，入力数よりも出力数の方が少ないトランジション成分で構成される環境ネットのトランジションの生起により，エージェントの消滅を表現できる．

- **エージェント間の動的な通信リンクの変更**
  環境ネットのトランジションに対し，それと同期して発火する複数のエージェントネットのトランジションが動的に決定される．これにより，エージェント間の通信リンクの動的な変更が表現できる．

第3章　エージェントネットの縮約

PN^2 では，環境がエージェントに動作可能な範囲を与え，各エージェントはその環境のもとで，与えられた動作を行っていく．このときに，環境がエージェントに与える動作の制約により，環境ネットは変えずにエージェントネットの動作の記述を縮約しても，そのエージェントに設定していた動作と同じ動作を実現できる場合がある [10]．エージェントネットの動作の記述を縮約できれば，そのエージェントネットの状態数×遷移の数で決定される接続行列のサイズを小さくできる．この接続行列は，

- 状態方程式による到達可能性の判定
- システムの不変量を表すインバリアント解析

などに用いられる [11]．接続行列のサイズを縮小することにより，このような性質の解析をより効率的に行うことができる．なお本研究では，エージェントネットが有限オートマトン (トークンを１つもつ状態機械) で表現できるようなエージェントネットの縮約構成法について述べる．

3.1　１つのエージェントからなるシステムの縮約法

本研究では，エージェントネットを縮約する際に， 文献 [12] の手法を基に縮約を行っている．これは，２つのオートマトン G と S があり，G と S の協調動作を表す $G \parallel S$（G と S に共通する入力記号が同期をとって遷移するように振る舞うオートマトン）によって，G の状態遷移を S が制御するという考えにもとづく S の構成法に関する研究である．[12] の縮約法では，G および S が与えらたとき，

$$L(G \parallel S) = L(G \parallel S') \text{ でかつ，} \mid S \mid \geq \mid S' \mid$$

であるような S' を求める．ここで $\mid S \mid$ はオートマトン S の状態数を表す．この縮約法は，エージェントネットが１つかないエージェントシステムの場合に適用できる．

3.2 複数のエージェントからなるシステムの縮約法

PN^2 の協調動作としては，エージェントネットが「環境ネット＋他のエージェントネット」の動作を制御していると考え，[12] での S をエージェントネット，G を「環境ネット＋他のエージェントネット」として見る．ここで，[12] の手法と違うところは，PN^2 では複数のエージェントを取り扱うので，G には環境ネット以外に他のエージェントネットの動作も含まれるということである．そして，１つの環境ネット G に対して，複数のエージェントネット $S_1, .., S_n$ が存在し，

$$L(G \parallel S_1 \parallel ... \parallel S_n) = L(G \parallel S_1' \parallel ... \parallel S_n') \text{ でかつ,} \quad \mid S_i \mid \geq \mid S_i' \mid \ (i = 1, ..n)$$

であるような $S_i'(i = 1, .., n)$ を求めることが目標となる．

縮約の１つの方法として，各エージェントネット S_i に対して，[12] における G を，

$$G_i' = G \parallel S_1 \parallel ... \parallel S_{i-1} \parallel S_{i+1} \parallel ... \parallel S_n$$

に対応させ，G_i' と S_i から [12] の手法により $S_i'(i = 1, .., n)$ を求めることが考えられる．しかし，この方法では G_i' の状態遷移図の状態数がエージェントの数 n の指数関数として増加してしまう．

この問題を避けるために本研究では以下のようにして $L(G_i')$ の上界を計算する．

(i)　$G \parallel S_i(i = 1, ..., n)$ の状態遷移図を作成する．

ここで，環境ネット G のトランジションは，トランジションに属するすべてのトランジション成分が発火可能でないとトランジションを発火することはできない．しかし，$G \parallel S_i$ の状態遷移図では，G の環境ネットのトランジションの中のどれか１つのトランジション成分と S_i のトランジションが一致してさえいれば，そのトランジション成分を含む環境ネットのトランジションは発火できるものとして，G と S_i の状態遷移を同期させている．つまり，S_i は他のエージェントとの協調動作を無視して状態遷移図を作成しているので，

$$L(G_i' \parallel S_i) \ \subseteq \ L(G \parallel S_i) \tag{3.1}$$

となり，$L(G \parallel S_i)$ は実際の動作 $L(G_i' \parallel S_i)$ の上界になる．

(ii)　$G \parallel S_i(i = 1, ..., n)$ の状態遷移図より，S_i に与えられている動作の中で複数のエージェントネットと関与している動作を求める．

しかし，(3.1) 式が ” = ” でないため，実際には不可能である協調動作を可能であると判定される場合がある．しかし，可能なものが不可能と判定されることはない．ここで，判定後の動作を $L'(G \parallel S_i)$ とする．

(iii)　(ii) の結果から $L(G_i')$ の上界，

$$L(G_i") = \Big(L'(G \parallel S_1) \cup .. \cup L'(G \parallel S_{i-1}) \cup L'(G \parallel S_{i+1}) \cup .. \cup L'(G \parallel S_n)\Big) \tag{3.2}$$

を求める．

(i)〜(iii) の手法の利点としては，$G \parallel S_i$ のサイズは $O(\mid G \mid \cdot \mid S_i \mid)$ で，$G'_i \parallel S_i$ のサイズは $O(\mid G \mid \cdot \mid S_1 \mid \cdot_{..} \cdot \mid S_{i-1} \mid \cdot \mid S_{i+1} \mid \cdot_{..} \cdot \mid S_n \mid)$ であるので，$G \parallel S_i$ のサイズは n の大きさに無関係であるということである．

一方，欠点としては，(3.1) 式が ” = ” でないため，エージェントネットを最小化できる保証がないことである．しかし，[12] の手法自体が NP-困難のため，実際には最小化をあきらめざるを得ないので，問題にはならないと考えられる．

そして，

$$L(G'_i) \subseteq L(G_i\text{”}) \quad \text{で} \quad L(G_i\text{”} \parallel S) = L(G_i\text{”} \parallel S')$$

ならば

$$L(G'_i \parallel S) = L(G'_i \parallel S')$$

が成り立つため，本研究の手法で縮約したエージェントネット S' が縮約前に与えられている動作と同じ動作を行うことができるのは保証できる．

3.2.1 エージェントネット縮約の手順

本研究では，エージェントネットの縮約を [12] の手法を基に，以下の手順で行っていく．

手順 1．環境がエージェントの動作に与える制約を求める

PN^2 では，環境ネットとエージェントネットが同期して発火するので，環境によってはエージェントに与えられている動作を許さない場合がある．エージェントに与えられている各動作に対して，環境がその動作を許すかどうかを表す制御則を求める．制御則は $\psi(t,s)$ と表現され，t はエージェントネットの動作を表す入力記号，s は状態である．$\psi(t,s)$ には 0，1，$dc(don't\,care)$ のいずれかの値が与えられ，それぞれ以下のような意味をもつ．

$\psi(t,s) = 0$ は s から t を遷移させてはいけない．
$\psi(t,s) = 1$ は s から t を遷移させても良い．
$\psi(t,s) = dc$ は s から t を遷移できない．

ここで，$\psi(t,s) = dc$ の場合は，環境がエージェントネットに与える制約により，エージェントネットの状態 s から t の遷移を起こさせないようにしている．

PN^2 にとっての環境とは，

- エージェントの動作に関わらず変化しない環境，つまりエージェントの動作可能な範囲をモデル化した環境ネットのような静的な環境

- 環境ネット上のエージェントの動作によって変化していく動的な環境

の２つからなる．そして，エージェントに与えられている動作の中には，

- 静的な環境の制約だけを受ける動作
- (静的＋動的) な環境の制約を受ける動作

の２つがある．

静的な環境による制約のみを受けるエージェントネットの動作の制御則については，環境ネット G による制約のみを受けるので， $G'_i = G$ となり，[12] の手法により求めることができる．

(静的＋動的) な環境による制約のみを受けるエージェントネットの動作の制御則については，環境ネット G 以外に他のエージェントの動作の制約も受ける．このような場合には 3.2 で述べたように，$G \parallel S_i(i = 1, .., n)$ の状態遷移図を生成し，(3.2) 式のように $L(G_i")$ を求め，制御則を求める．このようにして，環境がエージェントの各動作に与える制約を求める．

手順 2．マージできる状態対を調べる

エージェントネットの２つの状態対 (x, y) が，縮約する際に１つの状態にマージできるかどうかを調べる．このために，図 3.1 のようなマージ表を作成する．図 3.1 は３状態からなるエージェントネットのマージ表である．各状態対 (x, y) が１つにマージできる状態対と判定されたときは t，マージできないと判定されたときは f，未定の場合は以下に定義する $TP(x, y)$ をマージ表のセル (x, y) に埋める．

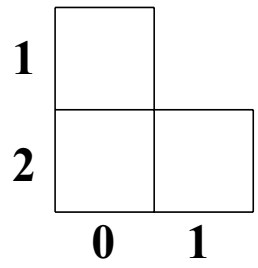

図 3.1: ３状態のエージェントネットのマージ表

マージ表を埋めていく最初の手順としては，まず以下の (3.3) 式が”偽”となる状態対は f となる．ここで，T は，縮約前のエージェントネットの入力記号の集合を表している．

$$(\forall t \in T)(\psi(t, x) \neq dc \neq \psi(t, y) \Rightarrow \psi(t, x) = \psi(t, y)) \tag{3.3}$$

手順１で求められた制御則と (3.3) 式により f となる状態対を求める．次に，まだマージ表の中で空欄となっている状態対 (x, y) に対して，以下の (3.4) 式で示される状態対 (x, y) から共通の入力記号による遷移先の状態対の集合を表す $TP(x, y)$ を求める．

$$TP(x,y) := \{(x',y') \mid x', y' \in S \wedge x' \neq y' \wedge (\exists t \in T : \delta(t,x) = x' \ \wedge \ \delta(t,y) = y')\} \quad (3.4)$$

(3.4) 式の S は縮約前のエージェントネットの状態の集合，δ は状態遷移関数を表している．$TP(x, y) = \phi$ のときは，セル (x, y) は t となり，$TP(x, y) \neq \phi$ のときは，セル (x, y) には $TP(x, y)$ が入る．

$TP(x, y)$ が埋められているセル (x, y) に対しては，$TP(x, y)$ の各要素の状態対 (x', y') のセル (x', y') を参照する．１つでも f と埋められているセル (x', y') が存在すれば，セル (x, y) に f を埋める．以上の処理を値が変化するセルが存在しなくなるまで繰り返す．

最後に，以上の処理を行っても，f と決まらないセルについては t とする．

手順 3. 縮約エージェントネットの状態の生成

状態数が最小の縮約エージェントネットを生成する問題は，NP-困難であるため [12]，以下で示す greedy な方法で縮約エージェントネットの状態に相当するブロック（マージする状態集合）を生成する．縮約前のエージェントネットの各状態が１つのブロックにしか含まれないようにブロックの生成を行う．ここで，ブロックによってカバーされていない状態の集合 U を与える．最初 U は，ブロックが生成されていない状態なので，縮約前のエージェントネットの状態の集合となっている．

step1: U の中で状態番号が最も小さい状態を含むブロック B を生成し，以下の条件を満たす状態が U の中にあれば，その状態をブロック B に加える．

条件:
ブロック B に入れようとする状態を状態 y とすると，ブロック B の中に既に入っている各状態 $x_i (i = 1, .., n)$ において状態 y との状態対 (x_i, y) に対応するマージ表でのセル (x_i, y) がすべて t である．

ただし，セル (x_i, y) に $TP(x_i, y)$ が埋められている場合については，y を x_i が属するブロック B に入れようとしたとき，$TP(x_i, y)$ の状態対 (x'_i, y') を部分集合にもつブロック B' も同時に生成する必要がある．B' を生成できるのであれば，y をブロック $B(\ni x_i)$ に入れることができるが，生成できなければ y をブロック B に入れることはできない．

step2: step1 により，B に入れることのできる状態が決定すれば，U の状態の集合の中から B の状態の集合を取り除く．

step3: step2 により，$U = \phi$ でなければ新しいブロックを生成して，step1 に戻り，$U = \phi$ であればブロックの生成を終了する．

手順 4. 初期状態および，状態遷移関数を定義する

縮約エージェントネットの初期状態は，縮約前のエージェントネットの初期状態の状態を含むブロックとする．

次に，状態遷移関数は以下のように定義する．ブロック B_i に含まれている各状態から，エージェントネットの各入力記号 a について，その状態からの遷移先の状態を含むブロック B_j を求め，$\delta(B_i, a) = B_j$ とする．

3.2.2 エージェントネットの縮約の例

3 つの自動搬送車 AGV1，AGV2，AGV3 をエージェントとして考え，それらのエージェントの動作可能な環境として，図 3.2 のような環境ネットのもとで動くエージェントシステムについて考えていく．ここで，この環境ネットのプレース E とプレース F では，AGV が 2 つ存在するとき AGV 間同士で通信でき，プレース D では AGV が 3 つ存在するとき，その 3 つの AGV 間同士で通信できる環境を与えている．また，この例の最初の各 AGV の存在場所としては，AGV1 は図 3.2 のプレース A，AGV2 はプレース C，AGV3 はプレース F に存在するものとして与える．

AGV1，AGV2，AGV3 には以下のような動作命令を与える．

$$AGV1 \ : \ A\Big(\rightarrow E(\rightarrow talk)^* \rightarrow F \rightarrow D((\rightarrow talk)\ or\ (\rightarrow A \rightarrow D)\ or\ (\rightarrow E(\rightarrow talk)^* \rightarrow F \rightarrow D))^* \rightarrow B \rightarrow A\Big)^*$$

$$AGV2 \ : \ C(\rightarrow D(\rightarrow talk)^* \rightarrow B \rightarrow C)^*$$

$$AGV3 \ : \ F((\rightarrow talk)^* \rightarrow D(\rightarrow talk)^* \rightarrow E(\rightarrow talk)^* \rightarrow F)^*$$

ここで，$(\ )^*$ は繰り返しを表す．以上のような各 AGV の動作命令をそのままモデル化したエージェントネットは，それぞれ図 3.3，図 3.4，図 3.5 のようになる．なお本研究では，有限オートマトンで表現できるようなエージェントネットの縮約であるので，エージェントネットを有限オートマトンで表現している．

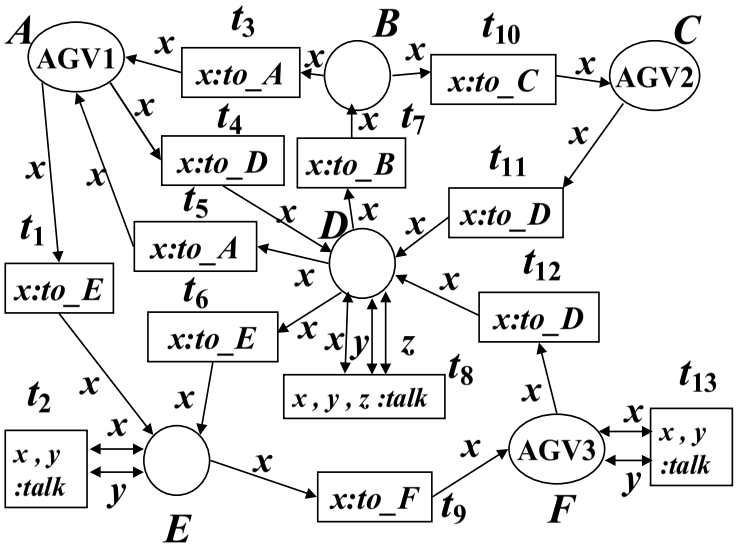

図 3.2: エージェントの環境ネット

本論文では，このような PN^2 を例に挙げ，縮約の対象の例として，図 3.3 の AGV1 のエージェントネットの縮約を 3.2.1 で述べた手順に従って行っていく．

手順１．制御則を求める

静的な環境 (環境ネット) の制約のみを受ける動作の制御則

$talk$以外のすべての動作は，他の AGV とは依存しない動作であるので，静的な環境 (環境ネット) が与える制約だけで制御則を求めることができる．例として，$talk$以外の図 3.3 の AGV1 の状態 0 からの制御則について求める．状態 0 の時に AGV1 は図 3.2 の環境ネットのプレース A に存在する．図 3.2 よりプレース A からは，to_D あるいは to_E の遷移が許されている．そして，図 3.3 の AGV1 の状態 0 からは to_E による遷移のみが与えられている．よって，to_Eについては，環境もエージェントも共に遷移を許しているので，$\psi(to\_E, 0) = 1$ となる．

また，to_D については環境では遷移が許されているが，図 3.3 の状態 0 からは与えられていない動作命令であるので起こしてはいけない．よって，$\psi(to\_D, 0) = 0$ となる．to_D, to_E以外の遷移については，環境が許していない遷移なので，制御則が dc となる．このように，図 3.3 のすべての状態について求めていくと，AGV1 の $talk$ 以外の動作の dc を除いた制御則は以下のようになる．

$$\psi(to\_E, 0) = 1 \quad \psi(to\_D, 0) = 0$$

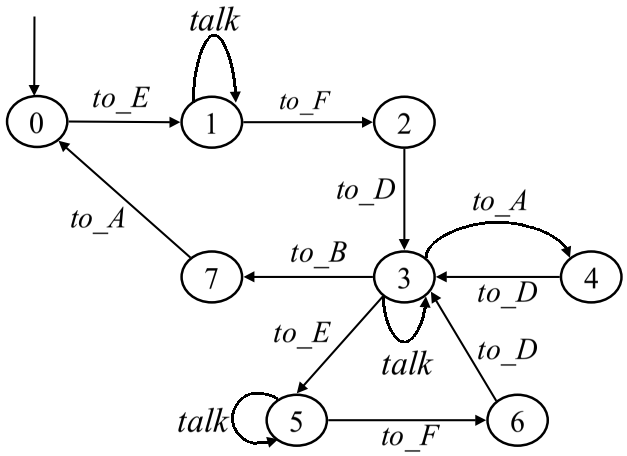

図 3.3: AGV1 のエージェントネット

$\psi(to\_F, 1) = 1 \quad \psi(to\_D, 2) = 1$
$\psi(to\_A, 3) = \psi(to\_B, 3) = \psi(to\_E, 3) = 1$
$\psi(to\_D, 4) = 1 \quad \psi(to\_E, 4) = 0$
$\psi(to\_F, 5) = 1 \quad \psi(to\_D, 6) = 1$
$\psi(to\_A, 7) = 1 \quad \psi(to\_C, 7) = 0$

(静的＋動的) な環境の制約を受ける動作の制御則

talk は他の AGV と協調して起きる動作なので，他の AGV の動作によって変化する動的な環境の制約も受ける．よって，*talk* の制御則を求めるには，他の AGV の動作を調べることが必要となる．これらの動作の求め方は，3.2 で述べたように (環境ネット || 各 AGV) の状態遷移図を生成して，各 AGV の動作の上界を求める．(環境ネット || 各 AGV) の状態遷移図はそれぞれ図 3.6，図 3.7，図 3.8 のようになる．

ここで，状態遷移図の状態は $(0, A)$ で表され，0 はエージェントネットの状態，A はその 0 の状態のときにエージェントが存在する環境ネットのプレースを表している．この状態遷移図により，各 AGV の

- 到達する可能性のあるプレース
- *talk* を起こす命令が与えられているプレース

を求めることができる．このように各 AGV の動作可能性を求め，環境 (環境ネット G || AGV2,AGV3) によって AGV1 の *talk* の遷移が許されるかどうかを調べると，AGV1 の *talk*

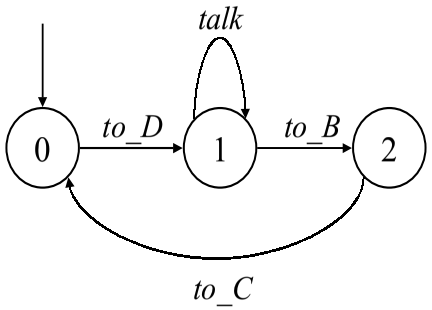

図 3.4: AGV2 のエージェントネット

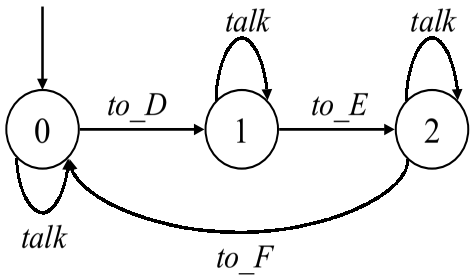

図 3.5: AGV3 のエージェントネット

に関する制御則は以下のように求めることができる.

$\psi(talk, 0) = dc \quad \psi(talk, 1) = 1$
$\psi(talk, 2) = 0 \quad \psi(talk, 3) = 1$
$\psi(talk, 4) = dc \quad \psi(talk, 5) = 1$
$\psi(talk, 6) = 0 \quad \psi(talk, 7) = dc$

手順 2. マージ表を埋める

図 3.3 のエージェントネットに与えられているすべての状態対についてマージできるかどうかを調べるために図 3.9 のようなマージ表を生成する.

マージ表を埋める最初の処理として，手順 1 で求めた制御則により，(3.3) 式で偽となる状態対，つまり f となる状態対を求めていく．例えば，手順 1 で求められた入力記号が $talk$ の 2 つの制御則 $\psi(talk, 2) = 0,\ \psi(talk, 3) = 1$ については，両方とも値が dc でなく，

$$(\psi(talk, 2) = 0) \neq (\psi(talk, 3) = 1)$$

となり，(3.3) 式が偽となるので，セル (3,2) は f となる.

次に，まだ f と求められていない状態対に対して，(3.4) 式により $TP(x, y)$ を求める．$TP(x, y) = \phi$ のときは，セル (x, y) は t となり，$TP(x, y) \neq \phi$ のときは，セル (x, y) には $TP(x, y)$ が入る．(3.4) 式の処理の後の AGV1 のマージ表は，図 3.9 のようになる．$TP(x, y)$ が埋められているセル (x, y) に対しては，図 3.9 のように $TP(x, y)$ の要素であるセル (x', y') を参照して，セル (x, y) の f を求める．セル (x', y') が 1 つでも f であれば，セル (x, y) には f が埋められる．そうすると図 3.9 から,

セル $(7, 3)$ は，f であるセル $(4, 0)$ への参照により f

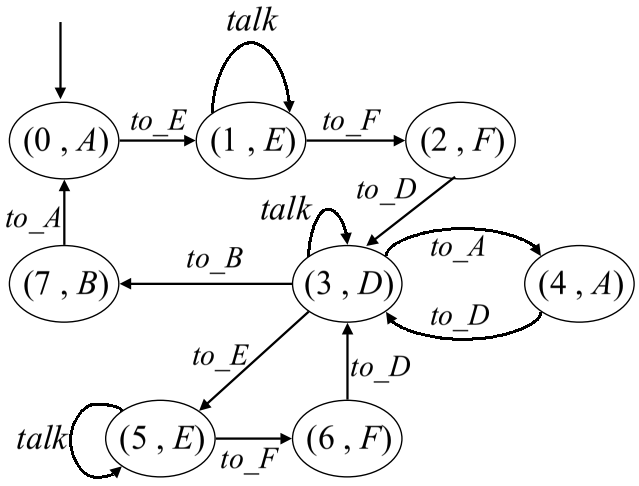

図 3.6: 環境ネット ||AGV1 の状態遷移図

セル $(5,1)$ は，t であるセル $(6,2)$ への参照により t
セル $(3,0)$ は，t となったセル $(5,1)$ への参照により t

となる．ここでセル (5,1) とセル (3,0) については，t かつ $TP(x,y)$ があるので，セル (5,1) は $TP(5,1)=\{(6,2)\}$ を，セル (3,0) は $TP(3,0)=\{(5,1)\}$ を残すというように，$TP(x,y)$ をセル (x,y) に残しておく．以上でマージ表は完成される．

手順 3. 縮約エージェントネットの状態の生成

まず，U の中で状態番号が最も小さい 0 を含むブロック B_0 を生成し，U の中からその B_0 に入れることができる状態を B_0 に入れていく．図 3.9 から状態 1 は，セル (1,0) が t なので $B_0=\{0\}$ に 1 を入れる．状態 2 は，2 と 0 が f なので B_0 に入れることはできない．状態 3 は，0 とも 1 とも t なので $B_0=\{0,1\}$ に入れることができるが，その時に図 3.9 より，セル (3,0) には遷移先の状態対として $TP(3,1)$=$\{(5,1)\}$ があり，またセル (5,1) には遷移先の状態対として $TP(5,1)$=$\{(6,2)\}$ が存在する．よって，{3,0} を部分集合にもつブロックを生成するとき，{5,1} と {6,2} を部分集合にもつブロックも考慮する必要がある．ここで図 3.9 より，

状態 5 は，$B_0=\{0,1,3\}$ に入れることができ，$\{5,1\}\subset B_0$ となる．
しかし，{6,2} は $B_0=\{0,1,3,5\}$ に入れることができない．

よって，B_0 とは別に {6,2} を部分集合にもつブロックを新たに生成する必要がある．次に U の残りの 4 と 7 については，4 と $0(\in B_0)$ が f，7 と $3(\in B_0)$ が f なので共に B_0 に入れることができず，結局

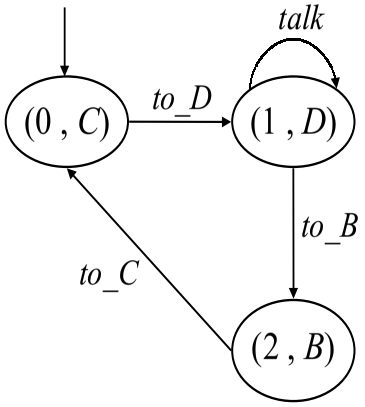

図 3.7: 環境ネット ||AGV2 の状態遷移図

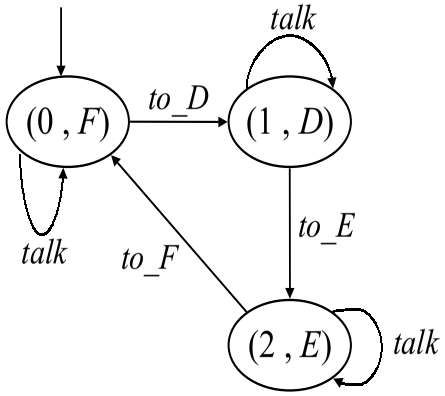

図 3.8: 環境ネット ||AGV3 の状態遷移図

$$B_0 = \{0, 1, 3, 5\} \text{ で決定し，} \quad U \leftarrow U - B_0$$

により，$U = \{2, 4, 6, 7\}$ となる．

次に，新しいブロックとして U の中で状態番号が最も小さい 2 を含むブロック B_1 の生成を行う．ここで，生成した B_0 より，{6,2} を部分集合にもつブロックを生成する必要があるので，まず 6 を $B_1 = \{2\}$ に入れる．残りの 4 と 7 においては図 3.9 より，4 は $B_1 = \{2, 6\}$ に入れることができ，7 も $B_1 = \{2, 4, 6\}$ に入れることができる．結局

$$B_1 = \{2, 4, 6, 7\} \text{ で決定し，} U \leftarrow U - B_1$$

により $U = \phi$ となるので，ブロックの生成を終了する．

以上のようにブロックを生成していくと，最終的に図 3.3 の AGV1 のエージェントネットは 8 状態から,

$$B_0 = \{0,1,3,5\} \qquad B_1 = \{2,4,6,7\}$$

の 2 状態へ縮約することができる．

手順 4．初期状態および状態遷移関数の定義

AGV1 のエージェントネットは，図 3.3 より 0 が初期状態であるので，0 を含むブロック B_0 が縮約エージェントネットの初期状態となる．

手順 3 で生成した B_0={0,1,3,5} からの状態遷移関数は，図 3.3 より以下のように求められる．

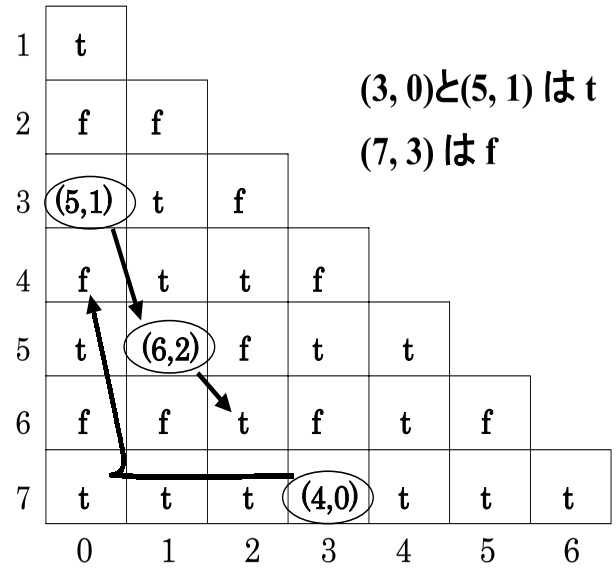

図 3.9: AGV1 のエージェントネットのマージ表

$to\_A : \{3\} \to \{4\} \subset B_1$, $to\_B : \{3\} \to \{7\} \subset B_1$
$to_D : B_0$の中のどの状態からも与えられていない入力記号
$to\_E : \{0, 3\} \to \{1, 5\} \subset B_0$, $to\_F : \{1, 5\} \to \{2, 6\} \subset B_1$
$talk : \{1, 3, 5\} \to \{1, 3, 5\} \subset B_0$

ここで，to_D の遷移については B_0 から起きることはない.

以上により，

$\delta(B_0, to\_A) = B_1$, $\delta(B_0, to\_B) = B_1$
$\delta(B_0, to\_E) = B_0$, $\delta(B_0, to\_F) = B_1$
$\delta(B_0, talk) = B_0$

となる．同様に，B_1={2,4,6,7} からの状態遷移関数は以下のように求められる.

$to\_A : \{7\} \to \{0\} \subset B_0$
$to\_D : \{2, 4, 6\} \to \{3\} \subset B_0$
$to\_B, to\_E, to\_F, talk$: B_1の中のどの状態からも与えられていない入力記号

ここで，$to\_B, to\_E, to\_F, talk$ の遷移については B_1 から起きることはない.

以上により，

$\delta(B_1, to\_A) = B_0$

$\delta(B_1, to\_D) = B_0$

となる．以上の手順で行うと，AGV1 のエージェントネットは図 3.3 から図 3.10 のように縮約できる．図 3.4 の AGV2 と図 3.5 の AGV3 のエージェントネットについても以上のような方法で縮約を行うと，図 3.11, 図 3.12 のような縮約エージェントネットを生成できる．

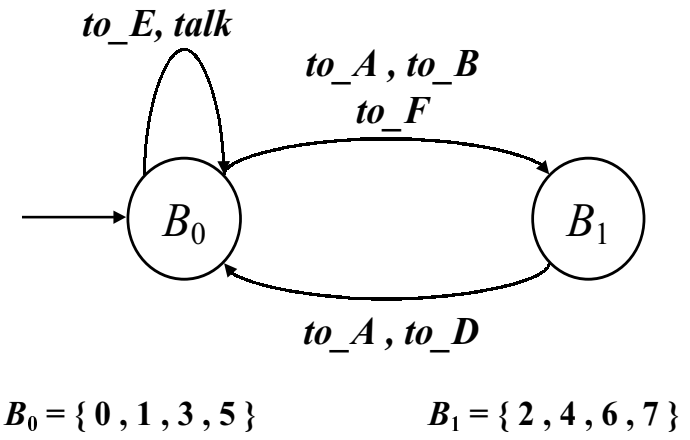

図 3.10: 図 3.3 の縮約エージェントネット

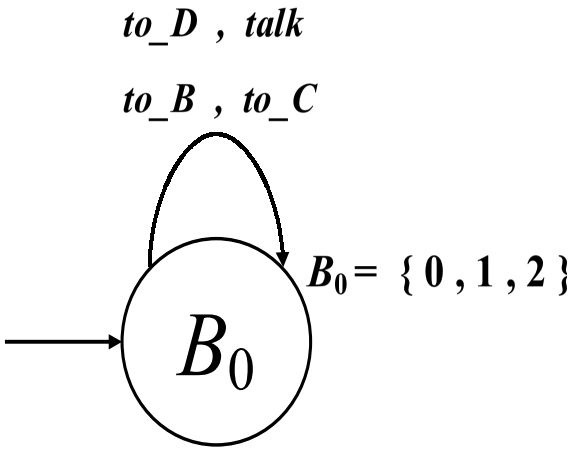

図 3.11: 図3.4の縮約エージェントネット

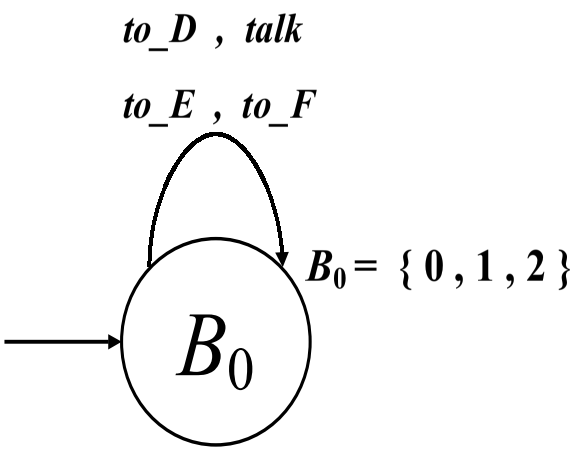

図 3.12: 図3.5の縮約エージェントネット

第4章　PN^2から１階層のペトリネットへの変換について

本章では，PN^2 のマーキング及び PN^2 のトランジションの発火によるマーキングの変化を表現できるような１階層のペトリネット (プレース/トランジションネット) の変換を行う．以降，プレース/トランジションネットを PN と表記する．

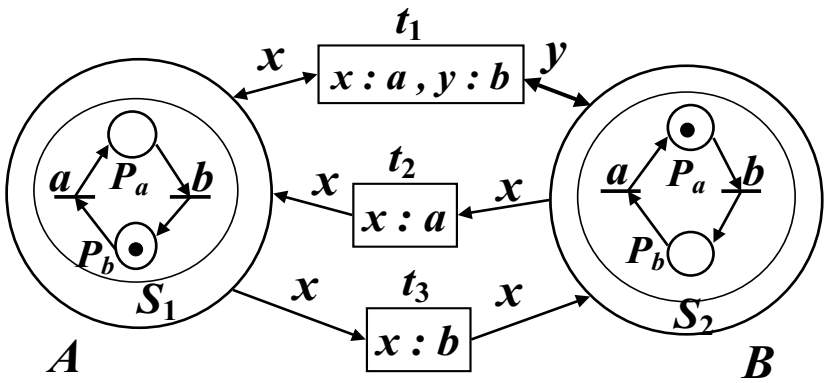

図 4.1: PN^2 の例

例えば，図 4.1 に示すエージェントネット S_1 と S_2 からなる PN^2 に対して，本研究で述べる手法で PN^2 から PN へ変換すると図 4.2 の PN となる．環境ネット上に存在するそれぞれのエージェントネットに対して，そのエージェントネットのマーキング及びエージェントネットの環境ネット上の存在場所を PN で表現することによって，PN^2 のマーキングを表現している．

- 図 4.2 のプレース P_{1A}, P_{1B} は，環境ネット上の S_1 の存在場所を表すためのプレースとして与えている，最初 S_1 は，環境ネットのプレース A に存在しているのでプレース P_{1A} にトークンを与えることによって S_1 の存在場所を表現している．また，図

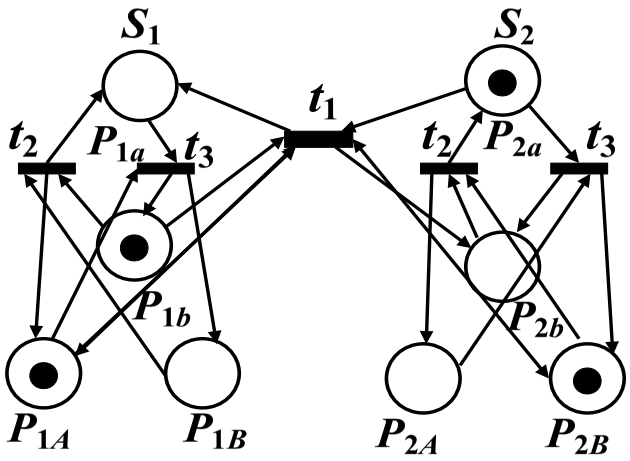

図 4.2: 図 4.1 の PN^2 を変換した PN

4.1 の S_1 のプレース P_a, P_b に対して，図 4.2 の PN では S_1 のプレースの集合 P_{1a}, P_{1b} を与えることにより，S_1 のマーキングを表現している．他のエージェントネットについてもこのように PN で表現することにより，PN^2 のマーキングを PN で表現している．

- 図 4.1 の t_1 のトランジションは，PN では図 4.2 の t_1 に対応しており，図 4.2 の PN の t_1 の発火後のマーキングは，図 4.1 の PN^2 の t_1 の発火後のマーキングに対応するようにトランジションの発火を表現している．

本研究での PN^2 に対しての PN の変換法とは，PN^2 の初期マーキングを M_0 とし，その M_0 を表現している変換した PN のマーキングを m_0 とすると，

$$L(PN^2, M_0) = L(PN, m_0)$$

が成り立ち，さらに PN^2 のトランジションに対応する PN のトランジションのそれぞれの発火後の PN^2 と PN のマーキングを対応させる写像が定義できるような PN への変換を行う．ここで，$L(PN^2, M_0)$ とは，PN^2 の初期マーキング M_0 から始まる発火系列の集合を表しており，$L(PN, m_0)$ とは M_0 を表現している PN のマーキング m_0 から始まる発火系列の集合を表している．このような変換を行うことにより，PN の解析手法として既に存在している

- 到達可能性
- 活性 (デッドロックが起こらない性質)
- システムのインバリアント (不変量)

の解析手法を変換した PN に適用することにより，変換前の PN^2 の解析を行うことができる．なお，PN へ変換する前に，有限オートマトンで表現できるようなエージェントネットについては，3 章で述べた縮約法を用いて縮約しておくことができる．

なお，図 4.2 で示した PN の構成法は，後の 4.2 で述べる方法 1 に基づいた PN の変換であり，図 4.1 では環境ネット上のエージェントネットの数が変化しなかったため変換できたが，エージェントネットが無限に増えてしまうような場合は，構成上 PN に変換できない．よって，PN^2 を PN へ変換する前に，エージェントネットの数が有限になるのかどうかを調べる必要がある．

4.1 PN^2 の有界性とその判定方法

PN の各プレースに入るトークンの数に上限が存在する PN の性質のことを有界性と呼んでおり，上限が存在する PN は有界，逆に上限が存在しない PN は非有界であると呼ばれる．PN^2 では環境ネットとエージェントネットの 2 階層のペトリネットで構成されているので，環境ネットの有界性とエージェントネットの有界性の 2 種類の有界性が PN^2 には存在する．

環境ネットが有界であるときは有界な PN^2，逆に非有界であるときは非有界な PN^2 と呼び，環境ネット上のすべてのエージェントネットが有界であるときは有限ソートの PN^2，そうでないときは無限ソートの PN^2 と呼ぶ．

4.1.1 有界性の判定方法

有界性の判定法として，PN の解析手法として既に存在している被覆木を用いる [2]．被覆木はエージェントネットの各マーキングをノードとし，発火可能なトランジションを有向枝に対応させた有向木である．PN が有界でなければ，とりうるマーキングの数が無限になるので生成される木の大きさも無限に大きくなる．これを避けるために，被覆木では，プレースに入りうるトークンの数が無限に大きくなることを表す記号 ω を導入することにより，木の大きさを有限に抑えている．よって，被覆木のどのノードの中にも ω が存在しなければ，その PN は有界と判定される．このように被覆木ではノードの中の ω の存在の有無で有界性の判定を行う．

PN^2 の有限ソート性の判定

まず有限ソートかどうかの判定については，各エージェントネット $S_a(a = 1, .., n)$ を環境ネット上に 1 つだけ入れ，その S_a の初期マーキングから環境ネットのトランジションの中に発火可能なトランジション成分があれば，そのトランジション成分を発火させて新しいマーキングを得ることにより，各 S_a の被覆木を生成する．他のエージェントと S_a が共に発火可能かどうかについては考えずに，各 S_a を動作させているので，

本研究の S_a 上のトークンの総数　≧　実際の S_a 上のトークンの総数

である．したがって，S_a の被覆木で S_a が有界と判定されれば，実際の S_a のエージェントネットも有界であると判定できる．しかし，本研究の手法で S_a が非有界と判定されたときは，上のような大小関係になるので，実際の S_a が非有界であるとは限らない．このように，各 S_a の有界性の判定を行うことにより，PN^2 の有限ソート性の判定を行う．

PN^2 の有界性の判定

環境ネット上に存在するエージェントネットをすべて黒丸のトークンと置き，また環境ネットにあるトランジションをすべて黒四角のトランジションと置いたペトリネット PN を考え，被覆木を用いてこの PN の有界性を判定する．

PN では，環境ネットのトランジションのトランジション成分の制約は受けずにエージェントネットつまり PN でのトークンが存在するだけで発火できる．よって，環境ネットのトランジションの発火条件が，すべてのトランジション成分が発火可能でないと発火できない実際のトランジションの発火条件よりも緩くなっていることから

本研究の環境ネット上での エージェントネットの総数　≧　実際の環境ネット上での エージェントネットの総数

になるので，PN が有界と判定されれば，置き換える前の PN^2 も有界と判定される．しかし，本研究の手法で PN が非有界と判定されたときは，上のような大小関係になるので，置き換える前の PN^2 が非有界であるとは限らない．

4.1.2　有界性判定の例

例として，図 4.3 の環境ネットと図 4.4 のエージェントネットで構成される PN^2 について，4.1 で述べた方法で 2 種類の有界性の判定を行う．

PN^2 の有限ソート性の判定

まず，PN^2 の有限ソート性の判定を行うために，図 4.4 のエージェントネットの有界性の判定を行う．図 4.4 のエージェントネットの被覆木は図 4.5 のように生成することができる．

図 4.5 の被覆木について説明すると，被覆木の各ノードはエージェントネットのマーキングとそのマーキングの時にエージェントネットが存在する環境ネットのプレースの 2 つで構成されている．例えば，最初 AGV のエージェントネットは (1 0 1 0) というマーキン

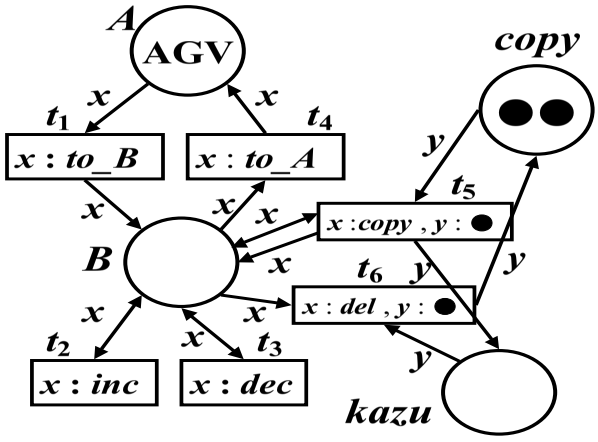

図 4.3: 環境ネット

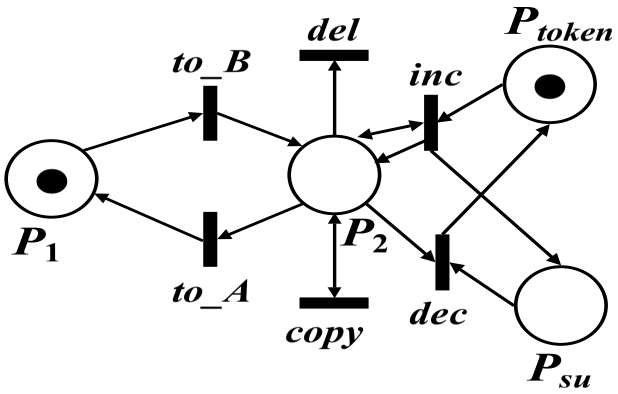

図 4.4: 図 4.3 の AGV のエージェントネット

グで環境ネットのプレース A に存在しているので，$\big((1\,0\,1\,0),A\big)$ が被覆木の初期ノードとなっている．そして，初期ノードから発火可能なトランジション成分が存在すれば，発火させて新しいノードを生成し，さらにそのノードからも発火可能なトランジション成分があれば新しいノードを生成することによって木を生成していく．

なお，図 4.5 の消滅と書かれたノードについては，環境ネットのトランジション t_6 のトランジション成分 $x : del$ は発火後の出力エージェントネット数が 0 で，このトランジション成分を発火できるエージェントネットは，$x : del$ の発火により環境ネット上から消えてしまうので，消滅と書いている．

この図 4.5 の AGV のエージェントネットの被覆木には，どのノードにも ω が存在していないので有界と判定される．環境ネット上の全てのエージェントネットが有界であれば，PN^2 は有限ソートであるので，この PN^2 の例は有限ソートであると判定できる．

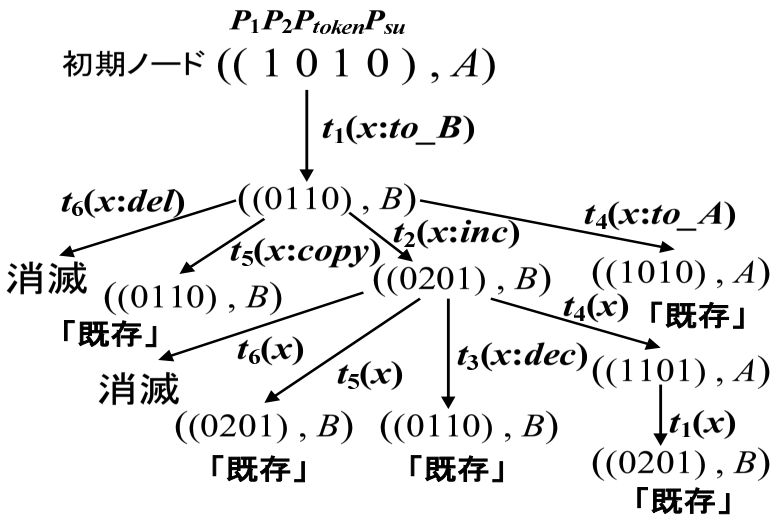

図 4.5: 図 4.4 のエージェントネットの被覆木

PN^2 の有界性の判定

図 4.6 のように図 4.3 の環境ネットの中のエージェントネットを黒丸トークン，トランジションを黒四角のトランジションに置き換えたペトリネット PN を考える．図 4.6 の PN に対する被覆木は，図 4.7 のようになる．

図 4.6 の PN に対する被覆木は図 4.7 になる．図 4.7 では，結局 ω はどのノードにも存在しなかったので，この例の PN^2 は有界であると判定できる．

4.2 PN^2 の変換方法について

4.1 で有界で有限ソートと判定された PN^2 においては，被覆木によって以下の値を求めることができる．

S_{max} ：環境ネット上に存在できるエージェントネットの上界の数．PN^2 の有界を判定する被覆木により求めることができる．被覆木のノードの中で各プレースに入るトークンの数の総和が最も大きいときの値が S_{max} となる．

$B(P_{ac})$ ：各エージェントネット $S_a(a = 1, .., n)$ の中の各プレース $P_c(c = 1, .., |P(S_a)|)$ に入りうるトークンの上界の数．各 S_a の有界性を判定する被覆木により求めることができる．ここで，$P(S_a)$ は S_a のプレースの集合を表す．

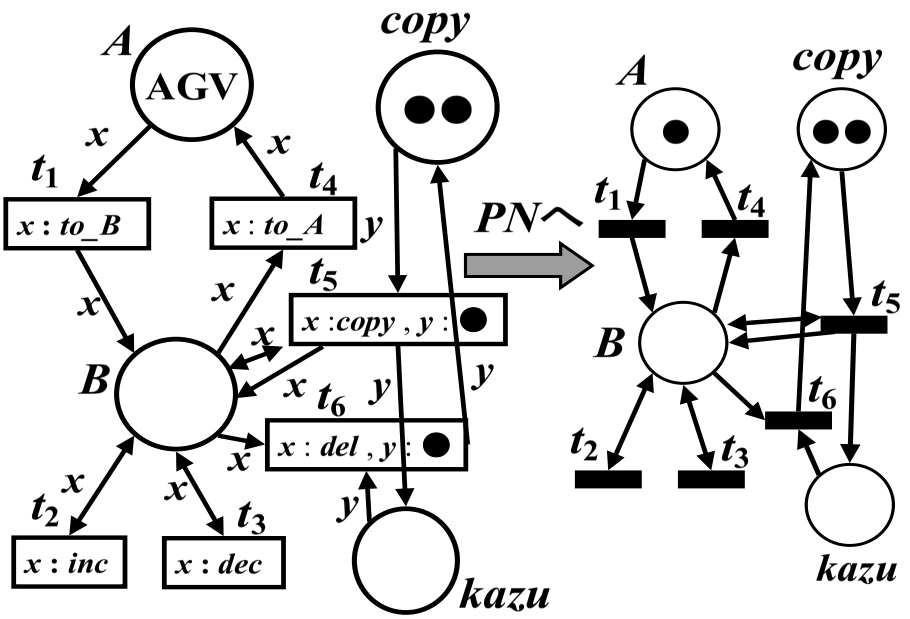

図 4.6: PN^2 の有界を判定するための PN への置き換え

$M(S_a)$: 各 $S_a(a = 1, ..n)$ のとりうるマーキングの有限集合．各 S_a の有界性を判定する被覆木により求めることができる．

$N(t_{l,m})$: 環境ネットの各トランジション $t_l(l = 1, ..L)$ の各トランジション成分 $m(m = 1, .., |E_l|)$ を発火できるエージェントネットのマーキングの有限集合．ここで，E_l は t_l のトランジション成分の集合を表す．

これらの値を基に，まず有界で有限ソートと判定される PN^2 についてのペトリネットへの変換方法 (方法１，方法２) について述べていく．ここで，結果から先に示すと，方法１と方法２でペトリネットに変換できる PN^2 のクラスは表 4.1 のようになる．○は変換可能，×は変換不可能を表しており，△は PN^2 の動作を近似的にシュミレートするようなペトリネットの変換であれば，方法１に基づいた PN の構成法で可能であることを示している．この方法については，後の 4.3.2 のところで述べる．

また，この表の ※ の箇所は，トランジション成分の入力が高々１つしかない PN^2 のクラスであれば，変換可能であるということを表している．これについても，4.3.2 で述べる．なお，図 4.1 で示した PN^2 のクラスは，表 4.1 でいう有限ソートで有界 (コピーなし) の PN^2 に相当する．

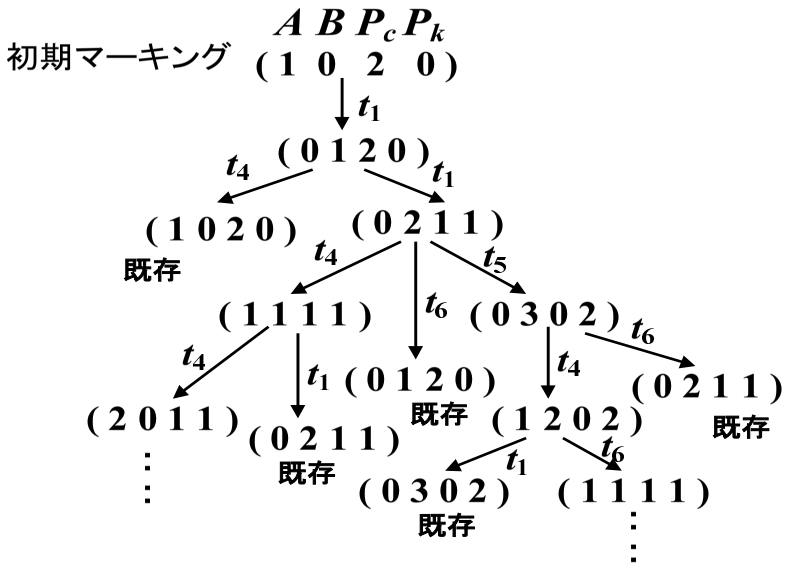

図 4.7: 置き換えた PN の被覆木

4.3 方法１

環境ネット上の各エージェントネット $S_a(a = 1, .., n)$ に対して，S_{max} 個のエージェントネットを与え，各 $S_{ab}(b = 1, .., S_{max})$ のすべてが S_a と同じ動作を実現できるように，それぞれの S_{ab} を PN で表現していく．

方法１では，PN^2 の１つのマーキングに対して複数のマーキングが対応する．例えば，$S_{max} = x$ で S_1 というエージェントネットがプレース A に１つ存在しているということを PN で表現しようとすると，S_1 を x 個与えているので，S_{11}〜S_{1x} の中のどれか１つを環境ネット上に存在させれば表現できる．これを PN では，S_{11}〜S_{1x} の中からのエージェントネットの１つの選び方が，１つのマーキングに対応するため，PN^2 の１つのマーキングに対して PN では x 個のマーキングが対応する．このことから，この PN では，PN^2 の１つのマーキング M に M のマーキングの集合 $\alpha(M)$ が対応することになる．

また，PN^2 での１つのトランジション t の発火を PN で表現するために，PN^2 での t に対し，PN では t のトランジション束縛ごとに１つのトランジションを生成していき，それらの集合 $\beta(t)$ を与える．ここで t のトランジション束縛とは，t を発火できる１つの方法 (t の各トランジション成分を発火できるマーキングの組合せ) を表している．

よって，PN^2 でのトランジション t に対応する PN のトランジションはトランジション集合 $\beta(t)$ となる．この時，

$f(m_0)$ ： $\alpha(M)$ の任意の要素のマーキング m_0 に対応する PN^2 のマーキング

$g(t)$ ： $\beta(t)$ の各トランジション t に対応する PN^2 のトランジション

表 4.1: 方法１および方法２で変換できる PN^2 のクラス

		有界 (コピーなし)	有界 (延べコピー回数が有限)	有界 (延べコピー回数が無限)	非有界
有限ソート	方法１	○	○	○	×
	方法２	○	○	○	○
無限ソート	方法１	※○	※△	×	×
	方法２	×	×	×	×

とすると，方法１での PN^2 の動作を保存するペトリネットの変換とは,

$$PN \text{ で } m_0 \xrightarrow{t} m_1 \text{ ならば } PN^2 \text{ で } f(m_0) \xrightarrow{g(t)} f(m_1) \quad \text{かつ}$$

$$PN^2 \text{ で } M_0 \xrightarrow{T} M_1 \text{ ならば}$$
$$PN \text{ で } \Big(\exists m_0 \in f^{-1}(M_0), \exists m_1 \in f^{-1}(M_1), \exists t \in g^{-1}(T)\Big)(m_0 \xrightarrow{t} m_1)$$

が成り立つような変換である．ここで，$m_0 \xrightarrow{t} m_1$ の意味は m_0 から t の発火によって m_1 に到達できるということを表している．

PN のプレース

- S_a の各 $S_{ab}(b = 1, .., S_{max})$ ごとに，S_a のプレースの集合 $P(S_a)$ を与え，その集合の要素の各プレース $P_{abc}(c = 1, .., |P(S_a)|)$ に対して，P_{abc} に入るトークンの数を表す番号として d を与え，$P_{abc(d)}(d = 0, .., B(P_{ac}))$ のプレースを与える．これにより，S_a のとりうる全てのマーキングを，与えたプレースによって表現できる．

- S_a の各 $S_{ab}(b = 1, .., S_{max})$ が環境ネットのどのプレースに存在しているかを表現するために，環境ネットのプレースの数が K 個あるとすると各 S_{ab} に対して，$P_{ab1}, P_{ab2}, .., P_{abK}$ のプレースを与える．なお，エージェントネットが内部状態をもたない黒丸のトークン●に対しても，$P_{\bullet\,1}, P_{\bullet\,2}, .., P_{\bullet\,K}$ のプレースを与える．

- 各 S_{ab} が環境ネット上に存在しているかを表現するためのプレースとして，各 S_{ab} に対して，P_{abnot} を与える．P_{abnot} にトークンが入っているときは，S_{ab} が環境ネット上に存在していないことを表しており，環境ネット上に存在している S_{ab} 以外の他

の $S_a(S_{a1},..,S_{a(b-1)},S_{a(b+1)},..,S_{amax})$ のマーキングを S_{ab} にコピーできる状態を表している．

PN^2 から方法１で PN へ変換した時のプレースの数は以下の値となる．

$$K+\sum_{a=1}^{n}\Big(S_{max}\times(K+1+\sum_{c=1}^{|P(S_a)|}(B(P_{ac})+1))\Big) \tag{4.1}$$

ここで，(4.1) 式の最初の K は，エージェントネットが●のときのために与える環境ネットのプレースの数を表しており，$(K+1+\sum_{c=1}^{|P(S_a)|}(B(P_{ac})+1))$ の $K+1$ の K は各 S_{ab} に与える環境ネットのプレースの数，1 はプレース P_{abnot} を表している．そして，$(B(P_{ac})+1)$ の 1 は S_{ab} のプレース P_c に入るトークンの数が 0 の時のプレース $P_{abc(0)}$ を表している．

PN のトランジション

環境ネットの各トランジション $t_l(l=1,..,L)$ に対し，t_l のトランジション束縛の集合を与える．t_l に対してのトランジション束縛の集合の要素数は，t_l の各トランジション成分 $m(m=1,..,|E_l|)$ を発火できるマーキングの数の総積となる．ただし，各 S_a ごとに S_{max} 個のエージェントネットが与えられているので，m を発火できる１つのマーキングに対して，S_{max} 個のエージェントネットを考える必要がある．

例えば，図 4.8 で S_1 のあるマーキングが t_1 の $x:a$ を発火できるとき，x へのエージェントネットの入力数が３，$S_{max}=5$ であるとすると，x が発火するには S_1 が３つ必要となるので，５個の S_1 の中から３つを取り出し，またその選んだ３つの S_1 の入力プレースへの割り当て方の１つ１つが x を発火できる１つの方法となるので，発火できる１つのマーキングに対して ${}_5P_3$ というように順列の数を考える必要がある．ただし，入力プレース B から x への入力数は 2 であるので，プレース B に入る２つの S_1 については B に入るエージェントネットの組合せだけを考えればよい．

よって，図 4.8 の例で，トランジション成分 x を発火できる１つのマーキングに対して必要なトランジションの数は，$\frac{{}_5P_3}{1!2!}=30$ となる．分母の 1!の 1 はプレース A から m への入力数，2!の 2 はプレース B から m への入力数を表している．これを一般的に書くと，t_l のトランジション成分 m を発火できる１つのマーキングに対して与えるトランジションの数は (4.2) 式となる．

$$\frac{{}_{S_{max}}P_{IN(t_{l,m})}}{\prod_{q=1}^{Q(t_{l,m})}(IN(t_{l,m}[q])!)} \tag{4.2}$$

ここで，(4.2) 式のそれぞれの値は以下のことを表している．

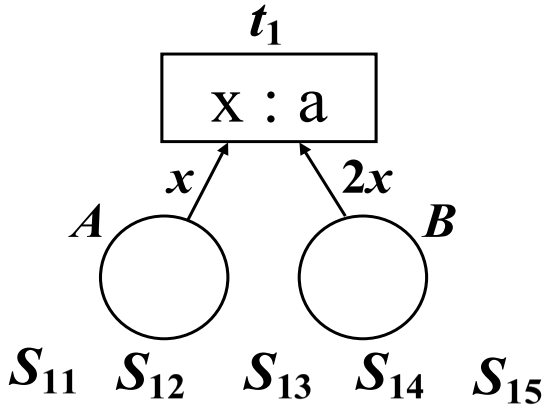

図 4.8: 複数の入力をもつトランジション成分の例

$IN(t_{l,m})$: t_l の m へのエージェントネットの入力数

$Q(t_{l,m})$: t_l の m への入力プレースの数

$IN(t_{l,m}[q])(q = 1, .., Q(t_{l,m}))$: 入力プレース $t_{l,m}[q]$ から t_l の m への入力数

$$IN(t_{l,m}) = \sum_{q=1}^{Q(t_{l,m})} IN(t_{l,m}[q])$$

ただし，トランジション成分 m からのエージェントネットの出力数を示す $OUT(m)$ がエージェントネットの入力数を表す $IN(m)$ よりも大きくなるトランジション成分 m については m の発火により，$OUT(m) - IN(m)$ のエージェントネットが増えるので，$S_{max} - IN(m)$ のエージェントネットの中から，発火により増える $OUT(m) - IN(m)$ 個のエージェントネットを選ぶ組合せを考える．したがって，$IN(t_{l,m}) < OUT(t_{l,m})$ となるトランジション成分 $t_{l,m}$ を発火できる 1 つのマーキングに対して与えるトランジションの数は (4.3) 式となる．

$$\frac{{}_{S_{max}}P_{IN(t_{l,m})}}{\prod_{q=1}^{Q(t_{l,m})} (IN(t_{l,m}[q])!)} \times {}_{(S_{max}-IN(t_{l,m}))}C_{(OUT(t_{l,m})-IN(t_{l,m}))} \tag{4.3}$$

以上より，トランジション成分 $t_{l,m}$ に対して与えるトランジションの数は，その $t_{l,m}$ の $IN(t_{l,m})$ と $OUT(t_{l,m})$ の大小によって，(4.5) 式，(4.6) 式のように場合分けして求める．その後，t_l に対して与えるトランジションの数は，各トランジション成分 $t_{l,m}(m = 1, .., |E_l|)$ に与えたトランジションどうしの組合せの数となるので，各 $t_{l,m}$ に与えたトランジションの数の総積となる．よって，PN^2 から方法 1 で PN へ変換する際に与えるトランジショ

ンの数は (4.4) 式の値となる．ここで，(4.5)，(4.6) 式の中の $|N(t_{l,m})|$ はトランジション成分 $t_{l,m}$ を発火できるマーキングの数を表している．

$$\sum_{l=1}^{L} \prod_{m=1}^{|E_l|} Z \tag{4.4}$$

($IN(t_{l,m}) \geq OUT(t_{l,m})$ である $t_{l,m}$の時)

$$Z = \frac{{}_{S_{max}}P_{IN(t_{l,m})}}{\prod_{q=1}^{Q(t_{l,m})} (IN(t_{l,m}[q])!)} \times |N(t_{l,m})| \tag{4.5}$$

($IN(t_{l,m}) < OUT(t_{l,m})$ である $t_{l,m}$の時)

$$Z = \frac{{}_{S_{max}}P_{IN(t_{l,m})})}{\prod_{q=1}^{Q(t_{l,m})} (IN(t_{l,m}[q])!)} \times {}_{(S_{max}-IN(t_{l,m}))}C_{(OUT(t_{l,m})-IN(t_{l,m}))} \times |N(t_{l,m})| \tag{4.6}$$

PN のアーク

トランジション t_l のトランジション成分 m をマーキング $S_a^{(h)}$ が発火できるとき，$S_a^{(h)}$ が m を発火するときのために展開されたすべての t_l のトランジションに対して，$S_a^{(h)}$ のマーキングに対応する各プレース及び，m を発火できる環境ネットのプレースに対応する S_a のプレースからアークを結ぶ．

4.3.1 変換の例

方法 1 による PN への変換法を用いて，有界で有限ソートと判定されている 4.1.2 で挙げた PN^2 を PN に変換すると，図 4.9 のように構成することができる．

図 4.7 の被覆木により $S_{max} = 5$ であることが求められているので，AGV のエージェントネットを S_1 とおくと，$S_{11}, S_{12}, S_{13}, S_{14}, S_{15}$ のエージェントネットを用意しておく．なお図 4.9 では，S_{11} と S_{12} のエージェントネットだけを表記している．S_{11}〜S_{15} はすべて同じエージェントネットなので，S_{11} のエージェントネットだけについて説明する．

- PN のプレース
  S_1 の各プレース $P_1, P_2, P_{token}, P_{su}$ に入るトークンの上界の数は，

$$B(P_{11}) = 1 \qquad B(P_{12}) = 2 \qquad B(P_{1token}) = 1 \qquad B(P_{1su}) = 1$$

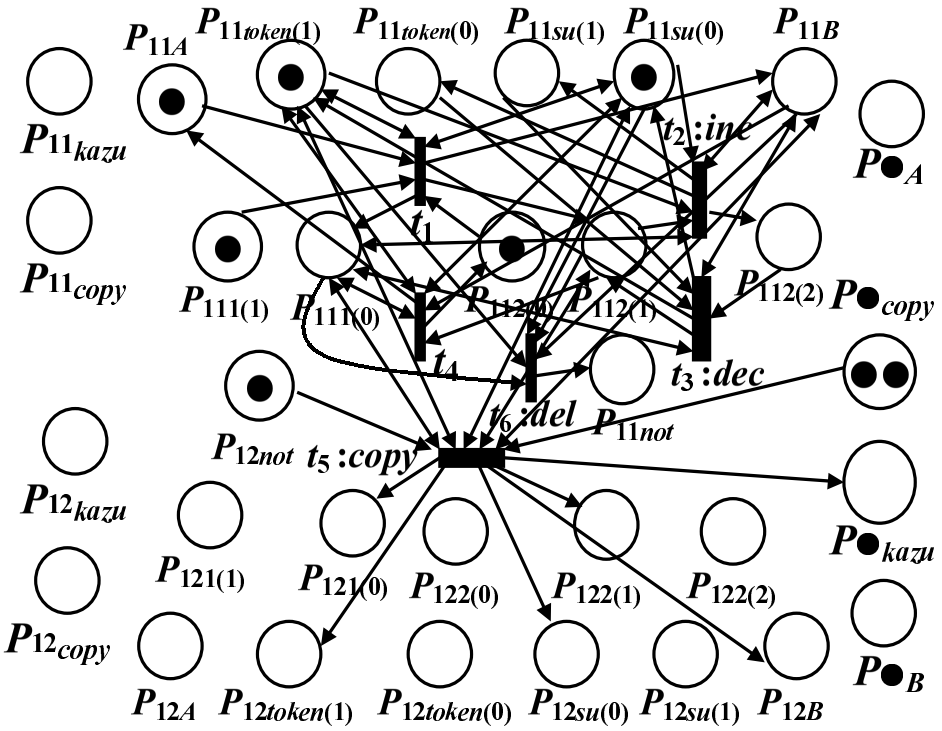

図 4.9: 方法 1 による PN の変換 (1 部分)

であることが図 4.5 の被覆木により求められている．このことから，例えば S_{11} のプレース P_{111} については，$B(P_{11}) = 1$ であることから，$P_{111(0)}$ と $P_{111(1)}$ のプレースを与える．$P_{112}, P_{11token}, P_{11su}$ についても同様にプレースを与える．また，S_1(AGV) の環境ネットでは図 4.3 より $A, B, copy, kazu$ の 4 つのプレースが与えられているので，$P_{11A}, P_{11B}, P_{11copy}, P_{11kazu}$ のプレースを与える．そして，S_{11} が環境ネット上に存在しているかどうかを表すためのプレースとして，P_{11not} を与える．このようにして，S_{12}〜S_{15} についても同様にプレースを与える．

また，エージェントネットが●のときに対しては，●の存在場所を表現するためのプレースとして $P_{● 1}, P_{● 2}, P_{● copy}, P_{kazu}$ を与える．

- PN のトランジション
  例えば，t_1 のトランジション成分 $x : to\_B$ を発火できるマーキングは，S_1 の (1010) と (1101) の 2 つであることが図 4.5 で求められており，t_1 への x の入力エージェントネット数 $IN(t_{1,x})$ 及び出力エージェントネット数 $OUT(t_{1,x})$ は共に 1 である．よって，t_1 のトランジション成分 x を発火できる 1 つのマーキングに対して与えるトランジションの数は，(4.5) 式により $\frac{5P_1}{1!} \times 2 = 10$ となり，t_1 はトランジション成分が x のみであるので，t_1 に対して展開されるトランジションの数は 10 となる．

- PN のアーク
  被覆木によりトランジション成分を発火できるときの環境ネットのプレース及びマーキングは求められている．例えば，t_1 の $x : to\_B$ を発火できる S_1 は (1010) のマーキングでプレース A に存在するときであることが図 4.5 の被覆木により分かる．よって，マーキング (1010) に対応する S_1 の各プレース及び存在場所がプレース A に対応する S_1 のプレースから，x を発火できるマーキングが (1010) であるときのために展開されたすべての t_1 のトランジションに対してアークを結ぶ．

- PN の初期マーキング
  PN^2 の初期マーキングとして，$S_1(\mathrm{AGV})$ が (1010) というマーキングでプレース A に１つ存在しているので，S_{11}〜S_{15} の中から１つ，番号が最も小さい S_{11} を選び，その S_{11} に対して，(1010) のマーキングに対応するようにトークンを与える．よって，変換した PN の $P_{111(1)}, P_{112(0)}, P_{11token(1)}, P_{11su(0)}$ のそれぞれのプレースにトークンを与え，S_{11} はプレース A に存在しているので P_{11A} にもトークンを与える．S_{12}〜S_{15} については環境ネット上に存在していないという意味で，P_{12not}〜P_{15not} にトークンを与える．また PN^2 では●がプレース $copy$ に２つ存在しているので，$P_{\bullet\ copy}$ にトークンを２つ与える．

4.3.2 方法１で PN に変換できる PN^2 のクラス

有界で有限ソートと判定される PN^2 のクラスについては，上で述べたような方法で PN へ変換できる．非有界な PN^2 のクラスは S_{max} が無限になるので，PN に変換することができない．また，無限ソートの PN^2 のクラスについてもエージェントネットのプレースに入るトークンの数が無限となり，プレースが無限に増えてしまうので変換することができない．

方法１で変換できる無限ソートの PN^2 の条件

ただし，次のような２つの条件を満たす無限ソートの PN^2 であれば PN へ変換できる．

- すべてのトランジション成分への入力が高々１つしかない
- エージェントの複製を表す $copy$ のようなトランジション成分をもたない

PN に変換できる理由としては，エージェントネットの各プレースに入るトークンの数ごとにプレースを与える必要がないからである．方法１でエージェントネットの各プレースに入るトークンの数ごとにプレースを与えてきたのには以下の理由があるからである．

- PN^2 でのエージェントネットのコピーとは，コピー元のエージェントネットと同じマーキングをもつエージェントネットを生成することであるので，コピー元のエージェントネットの各プレースに入っているトークンの数と等しくなるようなエージェントネットの生成を PN で表現する必要がある．

- また図 4.8 のような複数の入力をもつトランジション成分については，それらの入力のエージェントネットは全て同じマーキングでないと発火できないので，複数のエージェントネットの各プレースに入るトークンの数がすべて等しければ，それらのエージェントネットがトランジションを発火できるということを PN で表現する必要がある．

以上のことから，このようなトランジション成分をもたない PN^2 であれば無限ソートであっても方法１で PN に変換できる．

PN に変換できる無限ソートの PN^2 の例

図 4.10 のような上で述べたトランジション成分を持たない環境ネットと，その環境ネットの下で動く図 4.11 のような非有界なエージェントネット S_1(AGV) で構成される PN^2 は図 4.12 のように PN に変換できる．

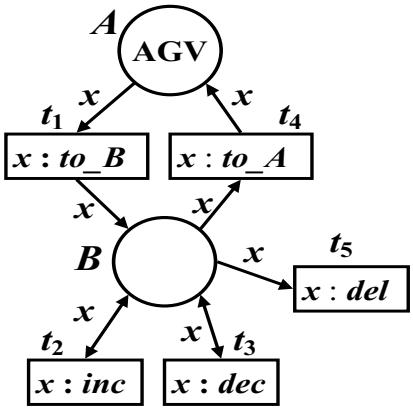

図 4.10: 環境ネット

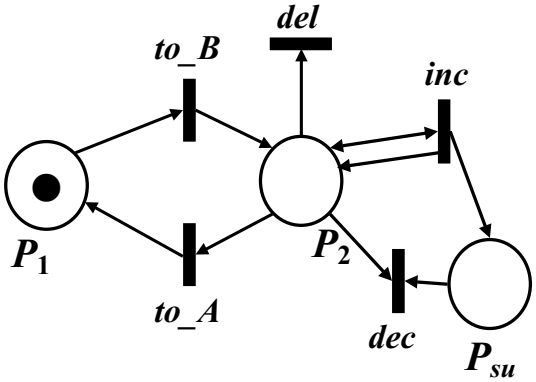

図 4.11: 図 4.10 の S_1(AGV) のエージェントネット

方法１では，エージェントネットのプレースに入るトークンの数ごとにプレースを用意していたが，このような PN^2 の場合は，図 4.12 のように S_1 のプレースの集合を与えるだけで非有界である S_1 のマーキングも表現できる．S_{11} の存在場所を示すための環境ネットのプレースの集合とプレース P_{11not} は方法１と同様に与える．図 4.12 のように S_1 のプレースに入るトークンの数ごとにプレースを与えなくてよいので，非有界なエージェントネットで構成される無限ソートの PN^2 でも PN へ変換できる．

PN による非有界なエージェントネットのコピーの表現法

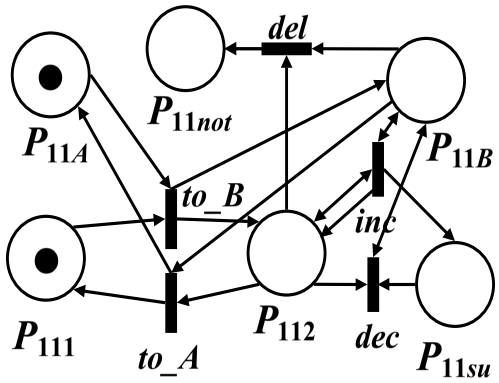

図 4.12: 図 4.10 と図 4.11 からなる PN^2 からの PN への変換

上で述べた方法のように各エージェントネットごとにプレースの集合を与え，非有界なエージェントネットのマーキングのコピーを PN で近似的に表現するものとして，PN^2 の 1 つのコピーを表すトランジションに対して，PN では複数のトランジションとプレースによって表現される一連のコピーの動作を対応させる．

図 4.13 のように，S_{11} のマーキングを S_{12} にコピーするために破線で示したトランジションとプレースを用いる．図 4.13 のプレース P_{stop} とは，PN ではコピーの動作を複数のトランジションとプレースによって表現しているので，コピーを実行している間は，他のすべてのトランジションは発火できないようにするために与えているプレースである．P_{stop} にトークンが入っているときは，コピーが実行中ではない状態で，P_{stop} にトークンが入っていないときはコピーの実行中で，他のトランジションは発火できない．図 4.13 ではトランジション *copy* の発火によってコピーの動作が始まり，t_{c6} の発火によってコピーの動作が終了する．

コピーの動作について述べると，*copy* を発火させた後，S_{11} のプレース P_{111} にはω 1 個のトークンが入っているので，破線のトランジション t_{c1} を最大ω 1 回発火できる．t_{c1} の発火回数の分，トークンが P_{c1} に入り，t_{c2} では最大 P_{c1} に入っているトークンの数の分発火でき，その発火回数の分のトークンが S_{12} のプレース P_{121} に入る．S_{11} のプレース P_{112} についても同様にして，S_{12} の P_{121} のなかにトークンを入れる．

このようなコピーの表現法では，

プレース P_{121} には最大ω 1(P_{111} のトークン数) まで
プレース P_{122} には最大ω 2(P_{112} のトークン数) まで

トークンを入れることができる．実際に，PN^2 ではコピーの発火により，プレース P_{121} にω 1 個，プレース P_{122} にはω 2 個のトークンが入る．このことから，この PN のコピーの表現法では，PN^2 のコピーの発火により得られる各プレースに入るトークンの数以下

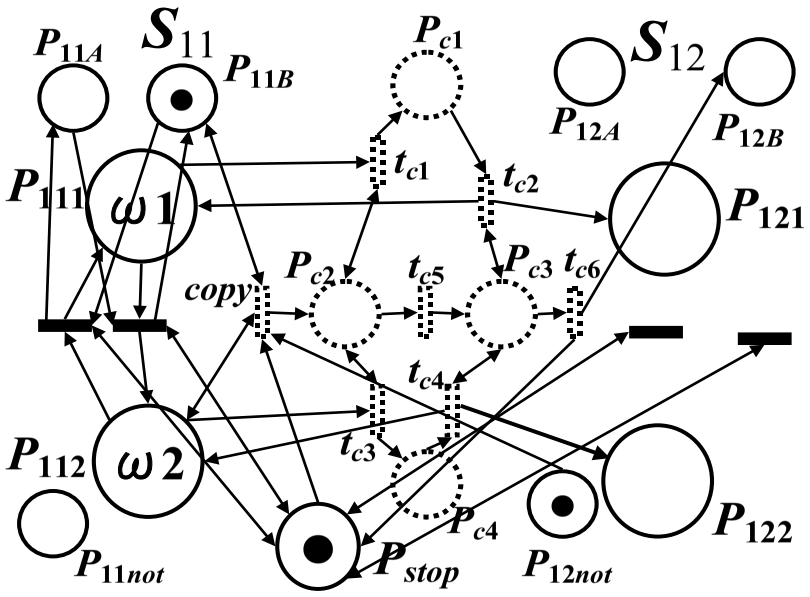

図 4.13: PN による非有界のエージェントネットのコピーの表現法

のトークンが各プレースに入ることを表している．今後，PN のコピーの動作を *copy* の発火とみなして述べていく．

この時，PN^2 と変換した PN の発火系列だけに注目すると，以上で述べたように PN の各プレースに入るトークンの数は PN^2 以下であるので，PN で起きるすべての発火系列は PN^2 にも存在し，また PN の *copy* の発火後の各プレースに入るトークンの数がすべて最大のとき，図 4.13 では *copy* の発火後，P_{121} に ω 1 個，P_{122} には ω 2 個のトークンが入るとき，実際の PN^2 のコピーの発火を実現することもできる．

以上のことから，この方法での PN^2 の動作を保存する PN の変換とは，

$L(PN^2, M_0)$ ： PN^2 の初期マーキング M_0 から始まる発火系列の集合

$L(PN, m_0)$ ： PN^2 の M_0 に対応する PN のマーキングの集合 $\alpha(M_0)$ の中の任意の要素のマーキング m_0 から始まる発火系列の集合

とすると，

$$L(PN, m_0) = L(PN^2, M_0) \tag{4.7}$$

の式が成り立つような変換である．

しかしこの PN の構成法では，次のような制約を考える必要がある．

1. *copy* のトランジション，図 4.13 でいう *copy* のトランジションは 1 回しか発火させてはいけない．なぜならば，1 度 *copy* を発火した後，P_{c1} や P_{c4} にはトークンが残って

しまう場合があり，再び S_{11} から S_{12} に $copy$ の発火が起こったとき，S_{12} の各プレースに入るトークンの数が，S_{11} の各プレースに入るトークンの数を超えてしまう場合が起きる．この時，PN^2 では起こりえない発火系列が，変換した PN では存在してしまうことになるので，(4.7) 式が成り立たなくなる．

2. エージェントネット S_{1a} の del(エージェントネットの消滅を表すトランジション) の発火をこの PN で表現しようとするとき，S_{1a} の各プレースに入っているすべてのトークンの消去を表現することはできない．
つまり，S_{1a} が del を発火した後，S_{1a} の各プレースにはトークンが残っている場合がある．del を発火した後の S_{1a} に対して，S_{1a} と同種類の他のエージェントネット S_{1b} が $copy$ を発火させたときも同様に，S_{1a} の各プレースに入るトークンの数が S_{1b} よりも多くなる場合が起きうるので，(4.7) 式が成り立たなくなる．

この 1 と 2 の制約から，この方法で PN に変換できる PN^2 のクラスは，延べコピー回数が有限の PN^2 の場合である．

また，この方法でもトランジション成分の入力が高々 1 つしかない PN^2 でないと PN に変換できない．なぜならば，この方法でも無限ソートでコピーなしのときの PN への変換法と同様に，各エージェントネットにはエージェントネットのプレースの集合だけでマーキングを表現しているからである．複数のエージェントネットの各プレースに入っているトークンの数がそれぞれ等しいときに発火できるという表現をこの PN では表現できない．

4.4 方法２

環境ネット上の各エージェントネット $S_a(a = 1, .., n)$ のとりうる各マーキング S_a^e(e=1,.., $|M(S_a)|$) に対して，それぞれのマーキングが環境ネットのどのプレースに存在しているかを表現するために，エージェントネットのマーキングと環境ネットの存在場所とを組み合わせて 1 つのプレースを生成する．

この PN の構成法では，PN のトークンはエージェントネットのマーキングとそのマーキングが環境ネットのどのプレースに存在しているかという存在場所の２つの情報をもっている．よって，ペトリネットのマーキングと PN^2 のマーキングとの対応は１対１である．

PN^2 でのトランジション t の発火を表現するために，PN では t のトランジション束縛ごとに１つのトランジションを生成していき，それらの集合 $\beta(t)$ を与える．この時，

$f(m_0)$ ： PN のマーキング m_0 に対応する PN^2 のマーキング，f は全単射

$g(t)$ ： $\beta(t)$ の各トランジション t に対応する PN^2 のトランジション

とすると，方法２で PN^2 の動作を保存する PN の変換とは，

$$PN \text{ で } m_0 \stackrel{t}{\rightarrow} m_1 \text{ ならば } PN^2 \text{ で } f(m_0) \stackrel{g(t)}{\rightarrow} f(m_1) \quad \text{かつ}$$

$$PN^2 \text{ で } M_0 \stackrel{T}{\rightarrow} M_1 \text{ ならば}$$
$$PN \text{ で } \Big(\exists t \in g^{-1}(T)\Big)\Big((f^{(-1)}(M_0) \stackrel{t}{\rightarrow} f^{(-1)}(M_1)\Big)$$

が成り立つような変換である．

PN のプレース

- 各 $S_a(a = 1, .., n)$ のとりうるマーキングの有限集合 $M(S_a)$ の要素である各マーキング $S_a^e((e = 1, .., |M(S_a)|)$ に対して，環境ネットのプレースの数を示す K 個のプレース $P_1S_a^e, P_2S_a^e, .., P_kS_a^e$ を与える．これにより，S_a のとりうる各マーキングが環境ネットのどのプレースに存在しているかを表現できる．例えば，プレース $P_1S_a^1$ にトークンが１つ入っていることは，PN^2 では S_a^1 というマーキングをもつエージェントネットが環境ネットのプレース１に１つ存在していることを表している．このような事から，変換した PN のトークンは，PN^2 でのあるマーキングをもつエージェントネットに対応している．
- また，エージェントネットが●のときに対しても，●の存在場所を表現するために，環境ネットのプレースの数を示す K 個のプレース P_1●, P_2●, .., P_K● を与える．

したがって，PN^2 を方法２で PN に変換する際に必要なプレースの数は以下になる．

$$K \times \sum_{a=1}^{n} |M(S_a)| \tag{4.8}$$

なお，エージェントネットが●の時はマーキングが１つしかないので，S_a が●の時であっても (4.8) 式となる．

PN のトランジション

PN^2 の各トランジション $t_l(l = 1, .., L)$ に対して必要なトランジションについて考えると PN では，各 t_l に対して，t_l のトランジション束縛の集合の要素数のトランジションを与える．この t_l のトランジション束縛の集合の要素数は，t_l の各トランジション成分 $t_{l,m}$ を発火できるマーキングの数の総積となる．以上のことから，PN^2 から方法２で PN に変換したときのトランジションの数は (4.9) 式になる．

$$\sum_{l=1}^{L}\prod_{m=1}^{|E_l|}|N(t_{l,m})| \tag{4.9}$$

PN のアークの生成

各 S_a の被覆木を生成したときに，環境ネットの各トランジション t_l のトランジション成分 m を発火できるエージェントネットのマーキングとそのときのエージェントネットの存在場所は求められている．

よって，もし環境ネットのプレース P_x に存在する S_a^h というマーキングが t_l のトランジション成分 m を発火可能であるならば，$t_{l,m}$ のトランジション成分を発火できるエージェントネットのマーキングが S_a^h であるときに展開されているすべてのトランジション t_l に対して，そのトランジションの入力プレースとしてプレース $P_xS_a^h$ からアークを結ぶ．

また，その時のトランジション t_l の出力プレースとして，プレース P_x に存在している S_a^h が t_l のトランジション m の発火により $S_a^{h'}$ にマーキングが変化し，m の出力プレースが P_y であるならば，そのトランジション t_l の出力プレースとして，プレース $P_yS_a^{h'}$ へアークを結ぶ．

4.4.1 変換の例

方法 2 による PN の変換法を用いて，4.1.2 で挙げた PN^2 を PN に変換すると図 4.14 のように構成することができる．

- PN のプレース
  AGV のとりうるマーキングは図 4.5 の被覆木により求められており，図 4.14 では AGV のエージェントネットを S_1 とおいて，S_1 のとりうる各マーキングを以下のように置く．

  $$(1010)\Rightarrow S_1^{(1)}\quad (0110)\Rightarrow S_1^{(2)}\quad (0201)\Rightarrow S_1^{(3)},\ (1101)\Rightarrow S_1^{(4)}$$

  環境ネットのプレースは図 4.3 より $A, B, copy, kazu$ であるので，S_1 のとりうる各マーキング $S_1^{(1)}, S_1^{(2)}, S_1^{(3)}, S_1^{(4)}$ と●のエージェントネットに対して，環境ネットの各プレース $A, B, copy, kazu$ を組み合わせて図 4.14 のようにプレースを生成する．

- PN のトランジション
  図 4.5 の被覆木により，環境ネットのトランジション成分を発火できるマーキングを求められる．例えば，図 4.5 により環境ネットの t_1 のトランジション成分 $x : to\_B$ を発火できるマーキングは $S_1^{(1)}$ と $S_1^{(4)}$ の 2 つであることが分かる．よって，t_1 のト

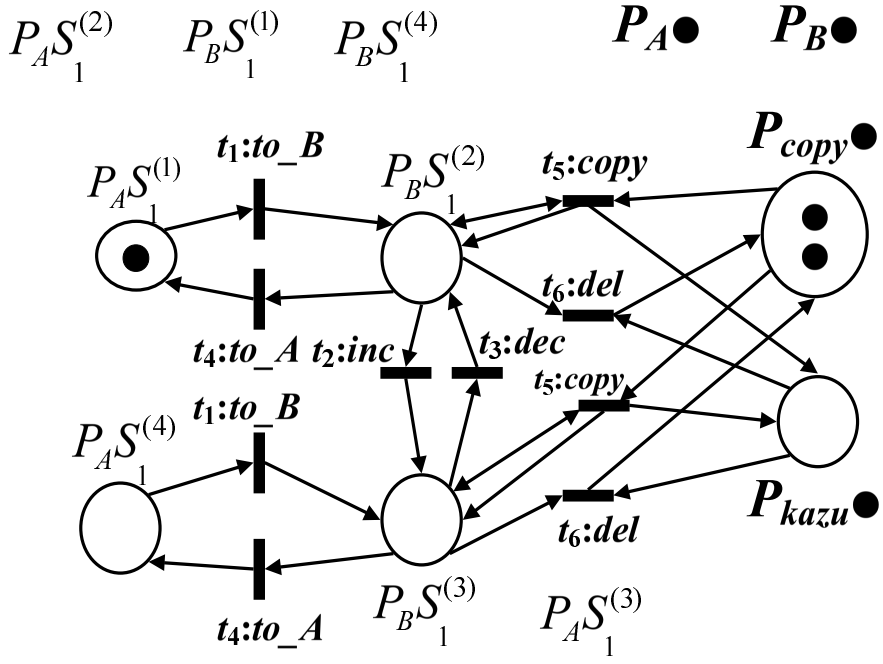

図 4.14: 方法 2 による PN への変換

ランジションは，x を発火できるマーキングが $S_1^{(1)}$ のときのトランジションと $S_1^{(4)}$ のときのトランジションの 2 つに展開される．

- PN のアーク
  例えば t_1 のトランジションは $S_1^{(1)}$ がプレース A に存在するときに発火でき，また $S_1^{(4)}$ がプレース A に存在するときに発火できるということが，図 4.5 によって求められる．このように，トランジション成分 m を発火できるマーキングが環境ネットのどのプレースに存在するときに m を発火できるかも被覆木によって求められる．この例では，$S_1^{(1)}$ がプレース A に存在するとき，$x : to\_B$ が発火できることを図 4.5 から求めらるので，t_1 の $x : to\_B$ を発火できるマーキングが $S_1^{(1)}$ のときに展開された t_1 のすべてのトランジションに対してプレース $P_A S_1^{(1)}$ からアークを結ぶ．

- PN の初期マーキング
  PN^2 の初期マーキングにおいて，$S_1(\mathrm{AGV})$ のエージェントネットが $S_1^{(1)}$ というマーキングでプレース A に 1 つ，また●がプレース $copy$ に 2 つ与えられているので，PN では $P_A S_1^{(1)}$ にトークンを 1 つ，P_{copy}● にトークンを 2 つ与える．

4.4.2 方法２で PN に変換できる PN^2 のクラス

方法２で構成される PN のトークンは，あるマーキングを持つエージェントネットに対応している．よって，環境ネットのプレースの中にエージェントネットの数が無限に増えていくような非有界と判定される PN^2 のクラスでも PN に変換することができる．ただし，エージェントネットのプレースにトークンが無限に入るような無限ソートの PN^2 においては，とりうるマーキングが無限になってしまい，PN に変換したときのプレースの数が無限になってしまうので表現できない．以上により，方法２の変換法では，無限ソートと判定されない PN^2 のクラスであれば，有界，非有界のどちらであっても PN に変換することができる．

4.5 方法１と方法２の比較

有界で有限ソートと判定される PN^2 は方法１でも方法２でも PN に変換できるので，どちらの方法で構成される PN のほうが PN^2 をより小さいサイズで表現できるかについて述べる．

- プレースの数の比較
  方法１で PN に変換したときのプレースの数=(4.1) 式で，方法２ で PN に変換したときのプレースの数=(4.8) 式で求められる．環境ネット上のエージェントネットの数について考えてみると，方法１では環境ネット上のすべてのエージェントネットをそれぞれ PN で表現しているので，エージェントネットの数，つまり S_{max} が多くなるほど，方法１で変換した時の PN のプレースの数は大きくなる．これに対して，方法２では (4.8) 式より S_{max} が増えてもプレースの数は増えない．よって，環境ネット上のエージェントネット数が多くなるような PN^2 の場合は，方法２のほうが少ないプレースの数で PN^2 を PN に変換できると考えられる．

- トランジションの数の比較
  方法１で PN に変換したときのトランジション数は=(4.4),(4,5),(4,6) 式で求められ，方法２ で PN に変換したときのトランジション数は=(4.9) 式で求められる．あるトランジション成分 $t_{l,m}$ に必要なトランジションの数は，方法１では (4,5) 式，もしくは (4.6) 式によって求まるのに対して，方法２ではトランジション成分 $t_{l,m}$ を発火できるマーキングの数 $|N(t_{l,m})|$ だけで求まる．よって，トランジションの数は方法２のほうが方法１よりも少ない数で PN^2 を表現できる．

第5章　おわりに

本研究では，PN^2 の効率的な解析手法として，エージェントネットの縮約構成法と，ペトリネットの既存の解析手法を PN^2 にも適用できるような PN^2 から１階層のペトリネットへの変換手法について述べた．

上記の手法において環境ネット上の各エージェントネット $S_a(a = 1, ..n)$ の動作可能性を調べるときに，他のエージェントとの協調動作については考えずに，状態遷移図および被覆木を作成する．この各 S_a の動作の求め方は，S_a の実際に起こる動作の上界を用いるため，実際には S_a が他のエージェントネットの動作によって起こり得ない動作も許してしまう場合があるが，S_a で実際に起きるすべての動作は含んでいる．したがって，次のことが言える．

- 縮約したエージェントネットにおいて，エージェントネットの最小化は保証できないが，エージェントネットに与えられているすべての動作は縮約後のエージェントネットで行うことができるのは保証できる．

- また，変換した PN の中には，実際には発火することのない無駄なトランジションやトークンが入ることのない無駄なプレースが含まれている場合があるが，実際の PN^2 で起こり得るすべての動作を実現するためのトランジションやプレースは含まれている．

他のエージェントネットの動作も考えた実際の環境下で，S_a の状態遷移図および被覆木を作ろうとすると，環境ネット上のすべてのエージェントネットのマーキングも同時に考える必要があるので，状態遷移図の状態数及び被覆木のノードの数が，環境ネット上に存在するエージェントネットの数の指数関数として増加してしまう．よって，最小化まではできる保証がないが，本研究での各 S_a の起こり得る動作を調べる方法のほうが，厳密に各 S_a の起こり得る動作を調べる方法よりも効率的である．このような動作の上界を見積もる手法は，モデルの階層性を有効に利用した１例である．

最後に，PN^2 の解析についての今後の課題として，以下のことが挙げられる．

- PN への変換について，表 4.1 のように，無限ソートで非有界の PN^2 のクラスに対しては，方法１でも方法２でもペトリネットに変換することはできない．よって，この PN^2 のクラスの動作をシュミレートできるような PN への変換法を検討する．

- 既存の PN^2 の接続行列の表現法ではエージェントネットのとりうるマーキングの数が増えるごとに行列サイズも大きくなるという問題がある．よって，このような問題を解決し，よりコンパクトに表現できるような PN^2 の接続行列の構成法を検討する．

謝辞

最後に，本研究を進めるにあたり，指導教官であり，北陸先端科学技術大学院大学情報科学研究科の平石 邦彦助教授には，終始様々な助言，ご指導を頂きました．また，同研究室の宋 少秋助手，高島 康裕助手にも終始様々な助言，ご指導を頂きました．

ここに深く感謝の意を表します．

発表論文

岡橋，平石，”エージェント指向ペトリネット PN^2 の効率的解析について”，信学技報，CAS2002-101,No.427,pp.23-28,2002-11.

参考文献

[1] 木下哲男，”エージェントシステムの作り方”，(社) 電子情報通信学会，(2001)

[2] 村田忠夫，”ペトリネットの解析と応用”，近代科学社，(1992)

[3] S.Philippi，”System Modeling Using Object-Oriented Pr/T-Nets”，Research Report No.25/97，Institute for Computer Science，University Koblenz - Landau，(1997)

[4] E.Battiston，F.De Cindio, G.Mauri，”OBJSA Nets : A Class of High-Level Nets Having Objects as Domains”，Lecture Notes in Computer Science，Vol.340，(1988)

[5] R.Valk，”Petri nets as token objects - an introduction to elementary object nets”，Lecture Notes in Computer Science，Vol.1420，(1998)

[6] M.Baldassari,”An Environment for Object-Oriented Conceptual Programming Based on PROT Nets”,Lecture Notes in Computer Science, Vol.340(1988)

[7] J.Engelfriet,G.Leih,G.Rozenberg,”Net-based Description of Parallel Object-based Systems”,Lecture Notes in Computer Science, Vol.489(1990)

[8] M.Ceska,V,Janousek,and T.Vojnar,”PNTalk-A Computerized Tool for Object Oriented Petri Nets Modelling”,Lecture Notes in Computer Science, Vol.1333(1994)

[9] K.Hiraishi，”PN^2 : An Elementary Model for Design and Analysis of Multi-Agent System”，(2002)

[10] 平石邦彦，”ペトリネットによるマルチエージェントシステムのモデル化”，システム/制御/情報，Vol.45，No.8，(2001)

[11] 青山，内平，平石，”ペトリネットの理論と実践”，システム制御情報ライブラリー13，朝倉書店，(1995)

[12] A.F.VAZ and W.M.WONHAM，”On supervisor reduction in discrete-event systems”，INT.J.CONTROL，Vol.44，No.2，(1986)