分散協調ＰＩを用いたマルチエージェントシステム
- 実装モデルとシステム構成 -

阿部 倫之　　服部 進実
金沢工業大学情報工学科

ネットワーク上に配置された複数の処理ノードで構成される分散システム環境では，動的に変化する負荷等に対し，状況に応じた最適な戦略を自律的かつ実時間で獲得する能力が要求される．このような能力を持つシステムとして，新しいマルチエージェントシステムの考え方が必要となる．本論文では，ルールベースシステムに，ＰＩ (process of induction) のコンセプト構造および分類子システムの報酬と罰則に基づく強化学習メカニズムの考え方を融合した実装モデルを示し，これを分散環境に拡張した分散協調ＰＩを提案する．また，この分散協調ＰＩを用いたマルチエージェントプラットフォームについて述べる．

1　はじめに

ネットワーク上に配置された複数の処理ノードで構成される分散システム環境では，動的に変化する負荷等に対し，状況に応じた最適な戦略を自律的かつ実時間で獲得する能力が要求される．例えば，ネットワーク負荷制御系ではマルチメディアトラヒック制御やルーティング制御などがこれに相当する．これら広域的かつ動的に変化する世界にスムーズに対応するためには，あらかじめ与えられた明示的シナリオに基づいた大域的な同期制御の考え方からさらに進んで，

1. 局所的な同期による大域的波及メカニズム，
2. 環境変動に対して処理メソッドを適応的に修正することによる予測能力，
3. 予測が外れた場合に対処するための学習能力，

を分散システムに与えることが重要である．すなわち，自律性を持った主体によって環境の局所的および大域的秩序を維持していくシステムの構築が望まれる．このような能力を持つシステムとして，新しいマルチエージェントシステムの考え方が必要となる．

本論文では，ルールベースシステムに，ＰＩ (process of induction)[4] のコンセプト構造および分類子システム [4] の報酬と罰則に基づく強化学習メカニズムの考え方を融合した実装モデルを示し，これを分散環境に拡張した分散協調ＰＩを提案する．また，この分散協調ＰＩを用いて試作したマルチエージェントシステムの構成を示す．これは，記号処理と非記号処理を融合したモデルであり，分類子システムのようなフラットなビット列ルール空間ではなく，理解性の高い記号ルールをコンセプトと呼ぶフレームで構造化したコンセプトネットワークを持つ．

Multi-agent System Based on the Distributed and Cooperative PI, Noriyuki ABE, Shimmi HATTORI, Kanazawa Institut of technology

2　分散協調ＰＩ

2.1　コンセプトネットワークとルール

Thagard らが提案したＰＩ (process of induction) では，ルールベースシステムにおける「条件ー行為ルール」とクラスに相当するフレーム型のデータ構造「コンセプト (concept)」を用いて環境内の各種情報および関係のモデル化を行う．全てのルールはコンセプトに所属しており，例えば 'パケット処理系 (PacketProcessor)' というコンセプトには，'パケット処理系' に関する事を条件部に持つルールが所属する．ここで，「Ｘが 'パケット処理系' でかつＹが 'パケット (Packet)' であるならばＸにＹを '接続 (Connect)' 」のように，ルールの行為部にコンセプト '接続' に関する事が記述されている場合，コンセプト 'パケット処理系' とコンセプト '接続' はルールによって関係付けられているという．この様子を図１に示す．

このとき，ルールの行為部のコンセプトは，ルールの条件部のコンセプトから支持 (support) されているという．例えば，コンセプト '接続は二つのコンセプトから支持されている．ここで，

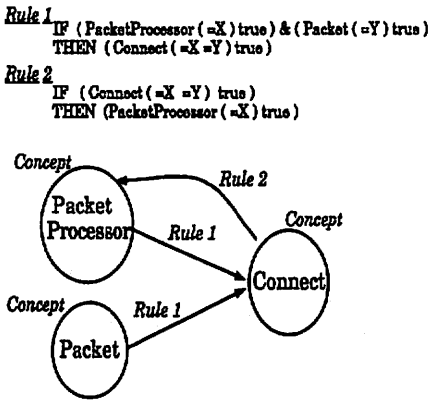


図 1: コンセプトとルール

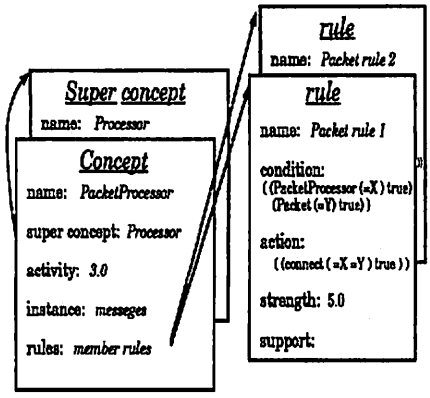


図 2: コンセプトのデータ構造

ルールの条件部と行為部は,

([コンセプト名] ([引数]) [真理値])

を基本要素とした記号列で表現する. これをメッセージと呼ぶ.

ＰＩにおけるコンセプトのデータ構造を図２に示す. コンセプトは, コンセプト名, 活性度 (activity), ルールリストなどの属性を持ち, ルールは, 条件部と行為部, および付け値 (bid) を計算するための強度 (strength), 支持度 (support) などの属性を持つ. ルールは, メッセージリストと呼ぶワーキングメモリ (WM) 内の環境情報 (メッセージ) と条件部のマッチングを試み, 成功すると発火ルールとなる. この

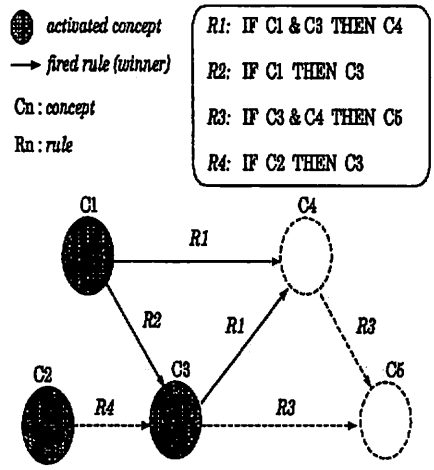


図 3: コンセプトネットワーク

点ではプロダクションシステムと同様であるが, 次の点で大きく異なる.

1. 活性化コンセプト (活性度がしきい値を越えているコンセプト) に所属しているルールをマッチング処理の対象とする.
2. 競合によって並列に実行できない発火ルールは付け値を用いて解消する.
3. 競合解消を含む実行の結果は, 報酬や罰則を用いた学習によって活性度と付け値 (特に強度) に反映される.

この結果, コンセプトで構造化されたルール群は, 「認識ー行動ー学習サイクル」を繰り返すに伴い, コンセプトの活性度とルールの付け値の強弱によって, 図３に示すようなコンセプトネットワークを適応的に自己形成する.

2.2 基本処理サイクル

強化学習機能を有するプロダクションシステムでは, 環境変動に応じた学習によってルールの評価値 (ここでは付け値) を適応的に変えれる自由度があるため, 時系列的なルール実行結果とそれに対する環境変動パターンをルールの評価値 (連続量) として記憶できる可能性をもつ. 本論文で提案するＰＩでは, Thagard らが提案したＰＩと比較して, 予測

のための強化学習および分散環境まで拡張可能な競合ルールの解消および並列処理メカニズムを導入している点が大きく異なる．基本処理サイクルを次に示す．

1. コンセプトの活性度を調べ，しきい値以上のコンセプトを活性化コンセプトとする．
2. 活性化コンセプトに所属するルールを取り出し活性化ルールとする．
3. 活性化ルールの条件部とワーキングメモリ内のメッセージをマッチングし，成功したルールを発火ルールとする．
4. 付け値を用いてルールの競合状態を解消し，並列実行可能な全ての発火ルールの行為部を実行する．これにより行為部のメッセージに従ってワークングメモリの内容を更新し，さらにメッセージに対応する手続きを効果器を用いて実行する．
5. 実行したルールの行為部が目標とマッチした場合目標が達成される．
6. コンセプトの活性度は目標達成後に強化学習を用いて更新する．ルールの強度は，目標達成までは backet brigade 法 [4]，目標達成後は profit sharing 法 [4] をベースにして強化学習を行う．
7. 目標達成によって得られた結果を評価し，予測の成功および失敗に対して報酬と罰則による強化学習を行う．この予測評価のための経験的知識を教師情報として設定する．

2.3 強化学習

強化学習では，行動に対して報酬または罰則を与えることにより，時系列的な行動パターン（リズム）を学習する [6]．この報酬と罰則は強化信号（reinforcement）によって実現し，報酬には正の強化信号（positive reinforcement），罰則には，負の強化信号（negative reinforcement）を用いる．報酬が無い場合は負の強化信号としてあつかう．

2.3.1 コンセプトの強度調整法

コンセプトの強化学習は1推論サイクル毎に実施する．推論に貢献したコンセプトの活性度は，正の強化信号（定数（＋1）またはメッセージ加重）を加算することによって強化する．貢献の基準としては，

- 発火ルールの条件部で使用されたコンセプト
- 発火ルールの行為部で使用されたコンセプト

がある．これ以外のコンセプトは，当該推論サイクルにおいて貢献しなかったものと判断し，負の強化信号（例えば－1）を加算して活性度を減少する．活性度がしきい値以下になるとコンセプトは非活性化状態となり，所属するルールは次の推論サイクルにおいてマッチングの対象から除外される．逆に，活性度がしきい値以上になると活性化状態となって，所属ルールが発火する可能性を持つ．活性化の状態は，環境変動によるルールの発火パターンに影響を受けるため，目標に至る過程において環境変動に対応したコンセプト空間が形成される．

2.3.2 ルールの強度調節法

（1）付け値

ルールの付け値は，強度（strength），支持度（support），特殊性（specificity）から算出する．提案するＰＩにおいて，この属性は次のように機能する．

（a）強度 過去にどの程度ルールが有効であったかを示す，目標達成に貢献するルールに連なっている場合，backet brigade 法と profit sharing 法により強度調整される．また，目標達成によって得られた環境変動の予測結果を評価に基づいてさらに強度調整される．

（b）支持度 ルールの所属しているコンセプトの活性度がルールの支持度となる．ルールの条件部が複数のメッセージから成る場合，ルールは複数のコンセプトに所属しているため，支持度は所属しているコンセプトの活性度の和で表現する．支持度の算出式を下記に示す．

$$V(C,t) := a \sum_{M \in \{M^*\}} A(C,M,t)$$

a:定数 ($\ll 1$)

$\{M^*\}$: ルールCが所属するコンセプトの集合
A(C,M,t): 時刻tでルールCが
所属するコンセプトMの活性度

(c) **特殊性** ルールの条件部が詳細に記述され，制約が強くなるほど特殊性は高くなる．これは汎用的なルールの動作を，より具体的なルールによって抑制する効果をもつ．ここでは，条件部の長さ（メッセージの個数）を特殊性の値とする．

付け値の算出式を下記に示す．

$$B(C,t) := bR(C)S(C,t)V(C,t)$$

b:定数 ($\ll 1$)
R(C):ルールCの特殊性，条件部の長さ
S(C,t):時刻tにおけるルールCの強度
V(C,t):時刻tにおけるルールCの支持度

この付け値を，ワーキングメモリに書き込むメッセージに荷重として与える．

（2）強化学習法

今回提案する強度調整法において，backet brigade法とprofit sharing法は次のように動作する．

(a) **backet brigade法** 発火ルールの行為部の実行によって，メッセージをワーキングメモリに書き込むとき，発火ルールの強度をその付け値の量だけ減少させる．この発火ルールを生産者ルールと呼ぶ．次の推論サイクルにおいて，このメッセージによって発火したルールを消費者ルールと呼び，生産者ルールに対して自分の付け値の量を報酬として送る．このとき消費者ルールの強度は送った報酬の量だけ減じる．算出式を次に示す．

$$S(C,t+1) := S(C,t) - B(C,t)$$
$$S(C',t+1) := S(C',t) + aB(C,t)$$

a:定数 (≤ 1)
S(C,t):時刻tにおける消費者ルールの強度
B(C,t):時刻tにおける消費者ルールの付け値
S(C',t):時刻tにおける生産者ルールの強度

(b) **profit sharing法** 目標に到達した時点において，ワーキングメモリに存在しているメッセージを送った生産者ルールに対して報酬を均等に分配する．

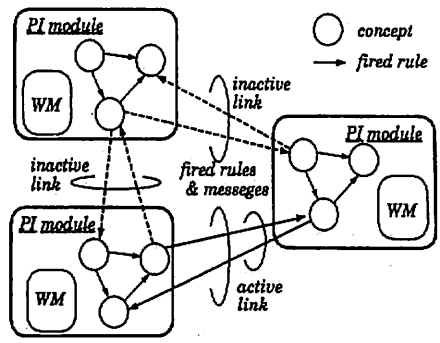


図4: 分散ＰＩモジュール間の活性度

backet brigade法では，ワーキングメモリに送ったメッセージが多数のルールを発火させた場合に大きな報酬が得られる．逆に他のルールを発火させる事ができなければ強度は減少したままとなる．強度が減少したルールを含む系列が再試行されていくと，強度の減少が後ろ向きに伝搬して，ルール系列全体が抑制されていく．逆にルールの強度が増加した場合は，そのルール系列が強化されていく．ルール系列の強化学習は緩やかに進行するが，profit sharing法を併用することでルール系列の途中に点在するルールを直接強化し，学習の早期収束を図ることが可能である．

2.4 分散協調メカニズム

ＰＩは，コンセプト，ルール，ワーキングメモリから成る．これをＰＩモジュールとしてカプセル化し，複数のＰＩモジュールを分散環境上に配置するモデルが分散協調ＰＩである．このとき，コンセプトはモジュール間にまたがったルールからも支持されることになり，この支持によって分散ＰＩ間の協調を実現する．前節で述べた強化学習の枠組みを，ＰＩモジュール間にも適用し，モジュール間にまたがった強度の伝搬を行う．このイメージを図4に示す．

ここで，協調を実現するためには，推論1サイクル毎にワーキングメモリの内容や発火ルールをブロードキャストする必要があり直接適用するのは通信量の点より困難といえる．そこで，発火ルールの

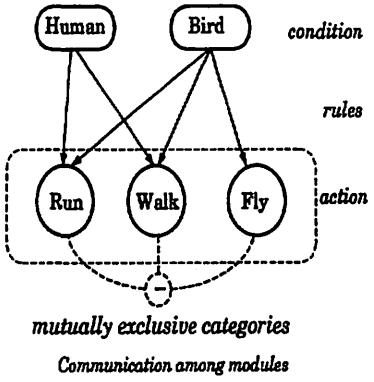


図 5: ＰＩモジュール間通信の条件

競合判定に使用する相互排他的カテゴリ（mutually exclusive categories）を用いた間引き協調メカニズムを導入する．

相互排他的カテゴリは，同時に存在できない事象（この場合はメッセージ）を定義したものである．この例を図５に示す．提案するＰＩでは，相互排他的カテゴリを，発火ルールの競合集合を生成する のに用いており，分散協調ＰＩでは，さらに協調のタイミングとして使用する．すなわち，相互排他的カテゴリに属する行為部を持つルールが発火した場合に，他のＰＩモジュールに対してメッセージと発火ルールを要求する（オンデマンド）．要求した発火ルールは自モジュールの競合発火ルール集合に含めて競合解消を図り，メッセージはワーキングメモリに格納して強度調整に用いる．

各ＰＩモジュールの自律性は，相互排他的カテゴリに登録するメッセージの種類に影響を受けるため，協調の効率を考慮してシステム設計する場合には，相互排他的カテゴリを意識したルール設計およびモジュール化が必要となる．また，強化学習の枠組みの中で協調の効率化を実現するには，「モジュール間を結合するルールの付け値の総量からモジュール間の結合強度（活性度）を算出し，あるしきい値以上の結合強度をもつモジュールに限定して協調する」方法が考えられる．

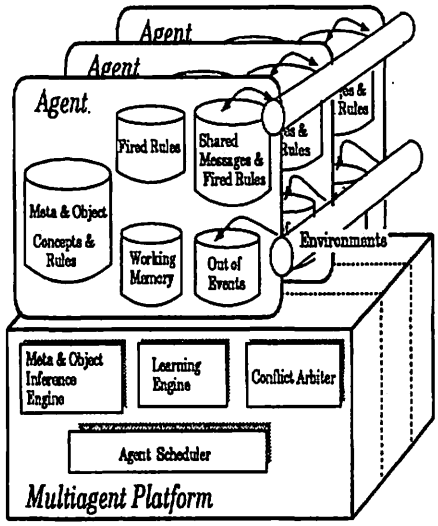


図 6: 分散ＰＩモジュールによるマルチエージェントシステムの構成

3 マルチエージェントシステム

分散協調ＰＩのＰＩモジュールをエージェントとして実装し，相互排他的カテゴリによる協調メカニズムを有するマルチエージェントシステムを構成する．システムは，Lisp系オブジェクト指向言語CLOS（Common Lisp Object System）を用いてSUN/4上に実装し，Lispオブジェクトで記述したエージェントをコンカレントに実行できる汎用プラットフォームとして構築する．

3.1 システム構成

プラットフォームは，対象推論エンジン（object inference engine），メタ推論エンジン（meta inference engine），競合解消部（conflict arbiter），学習エンジン（learning engine），エージェントスケジューラ（agent scheduler）から構成される．このシステム構成を図６に示す．

プラットフォーム上のエージェントの処理フェーズを次に示す．

（Ｆ１）profit sharing法によってコンセプトの活性度を更新する．このとき，コンセプトが他エー

ジェントの場合，協調領域（shared message & fired rules）に登録する．

（F 2）活性化コンセプトを選別する．

（F 3）ワーキングメモリ内のメッセージと活性化ルールをマッチングし，発火ルールを生成する．発火ルールは協調領域に登録する．

（F 4）発火ルールが競合しているかまたは発火ルールがない場合，協調領域から自分宛の発火ルールとメッセージを取込み競合を解消する．メッセージはワーキングメモリに格納する．荷重付きメッセージの場合，対象ルールの強度やコンセプトの活性度を調整する．

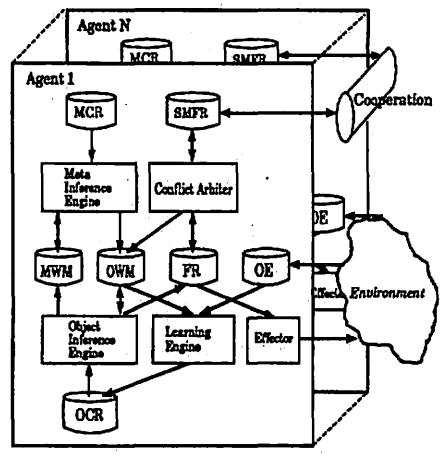
（F 5）発火ルールの行為部を実行する．このとき backet brigade 法で強度を調整する．処理対象のルールが他エージェントの場合，荷重付きメッセージを協調領域に送る．

（F 6）目標に到達した場合，profit sharing 法によってルールの強度を更新する．強度更新ルールが他エージェントの場合メッセージを協調領域に送る．また，環境情報（out of events）を取込み，教師ルール（learning rule）を用いた強度調整を実施する．

スケジューラは，処理するエージェントを（F1）から（F6）の各フェーズごとに切り換えることでコンカレント処理を実現している．また，この切り換えのタイミングを利用して対象推論とメタ推論を切り換えている．エージェントを実行する際のデータフロー図を図 7 に示し，各モジュールの機能を次に示す．ここで，図中の効果器（effector）は対象推論エンジン内に含まれている．

（1）メタ推論エンジン メタワーキングメモリ（MWM）内のメッセージに基づいてエージェントの次の処理プロセスを決定する．メッセージ内容は，現在のエージェントの状態（AgentStatus）とユーザコマンド（UserCommand）である．発火ルール獲得のためのメタルールの例を次に示す．

```
IF ((AgentStatus (fired-mode) true)
    (AgentStatus (conflict) true)))
THEN ( (AgentExec (get-fired-rule)
true))
```



MCR: Meta Concepts and Rules
OCR: Object Concepts and Rules
MWM: Meta Working Memory
OWM: Object Working Memory
FR: Fired Rules
SMFR: Shared Messages and Fired Rules
OE: Out of Events

図 7: エージェントエンジン間のデータフロー図

実行結果である AgentExec メッセージは，対象推論エンジンのワーキングメモリ（WM）に送る．

（2）対象推論エンジン ワーキングメモリ内の AgentExec メッセージに従い処理フェーズを決定する．推論エンジンの処理フェーズは（F3）と（F5）である．フェーズの終了時に，現在のエージェントの状態を AgentStatus メッセージにまとめてメタワーキングメモリに送る．

（3）競合解消部 処理フェーズは（F4）である．エージェント内に定義している相互排他的カテゴリリスト（メッセージのリスト）に基づいて処理する．例えば，Action 1 と Action2 が排他的行為の場合，次のようなリスト構造で定義されている．

```
((Action1(=ID)true)
 (Action2(=ID)true))
```

この構造は，ルールの条件部と同じであるため，メッセージのパターンマッチには，推論エンジンと同様のユニフィケーション機構を用いる．フェーズ終了時に現在のエージェントの状態を AgentStatus メッセージにまとめてメタワーキングメモリに送る．

（4）学習エンジン ワーキングメモリ内の AgentExec メッセージに従い処理フェーズを決定する．処理フェーズは（F1）と（F2）および（F6）である．（F6）において負の強化信号を与える教師ルールの例を次に示す．

```
IF ((Event(=ID check)true)true)
   (Event(=ID overflow)true))
THEN
((Reward((Action1(=ID)true)-1)
  true))
```

フェーズの終了時に，現在のエージェントの状態を AgentStatus メッセージにまとめてメタワーキングメモリに送る．

（5）エージェントスケジューラ プラットフォーム上のエージェントは，このスケジューラの ready queue に格納される．スケジューラは，queue からエージェント取り出し，最初にメタ推論エンジンへ送る．次にエージェントのワーキングメモリ内にある AgentExec メッセージに基づいて対象推論エンジン，競合解消部，学習エンジンのいずれかにエージェントを送る．目標に到達して学習を終了したエージェントは wait queue に移動し，他は ready queue に戻す．この基本サイクルの他に，協調領域をエージェント間の分散共有メモリとして扱うための一貫性制御（coherence control）および外部環境インタフェース（環境情報の獲得，行為部メッセージの環境への適用）の機能がある．

3.2 エージェントのデータ構造

エージェントは，CLOS 上のオブジェクトとして宣言している．オブジェクトを構成する属性を図9に示す．エージェント内には，対象推論およびメタ推論用のコンセプトとルールの実体集合があり，ワーキングメモリとゴールも同様である．その他，相互排他的カテゴリリスト（conflict），協調領域（SR，SM），教師ルールリスト（LR）の集合がある．

3.2.1 コンセプトとルールの構造

コンセプトとルールは，CLOS 上のオブジェクトとして宣言され，エージェント内に存在する．オブジェクトを構成する属性は図2に示したとおりである．コンセプト内には，活性度，ルール（名前のみ）の集合，および事例（インスタンス）がある．ルール内には，リスト表現された条件部と行為部，および強度と支持度がある．付け値計算に用いる特殊性は動的に求めており，属性としては定義していない．

attribute name	*semantics*
name:	*agent name*
concepts:	*a list of concept objects*
rules:	*a list of rule objects*
conflict:	*a list of interexclusive categories*
goal:	*a list of goals*
subgoal:	*a list of subgoals*
WM:	*a list of messages(working memory)*
SR:	*a shared list of rules*
SM:	*a shared list of messages*
events:	*out of events*
LR:	*a list of learning rules object*
mconcepts:	*a list of concept object for mata inference*
mrules:	*a list of rule objects for meta inference*
mgoal:	*a list of goals for meta inference*
MWM:	*a list of messages(meta working memory)*

図 8: エージェントのデータ構造

3.2.2 ワーキングメモリと推論

ワーキングメモリにメッセージを格納する場合，次の規則によってメッセージが書き換えられる．

（1）同じ内容のメッセージを書き込む場合，古いメッセージは消去する．

（2）相互排他的カテゴリに属するメッセージを書き込む場合，相互排他的カテゴリに属する古いメッセージを消去する．

（3）上記以外はメッセージの追加のみを実施する．

ワーキングメモリ内のメッセージは，相互排他的カテゴリに属するもの以外は長期間留まる可能性を持っている．従って，一度発火したルールは何度も連続して発火する可能性がある．この多数の選択肢の中で環境変動による強化学習が実施され，ルールが淘汰される．また多数の選択肢は，強化学習（予測）の失敗経験を直ちに反映させる機会も与えており，これによって微妙な均衡状態を適応的に維持する方向にシステムを導いている．選択肢の多さは処理効率の低下につながるが，コンセプトの自己組織化によ

るルール群の適応的な選別によってこの問題を解決している.

4 おわりに

本論文では，ルールベースシステムにコンセプト構造と強化学習メカニズムの考え方を融合した実装モデルを示し，これを分散環境に拡張した分散協調ＰＩを提案した．また，この分散協調ＰＩを用いた，具体的なマルチエージェントプラットフォームの構成を示した．今後，このプラットフォームを用いて予測学習能力の評価を行なう予定である．

謝辞
本研究は（財）テレコム先端技術研究支援センター殿の助成研究の一環として行なわれたものである．

参考文献

[1] 菅原俊治，Victor Lesser: “協調のためのルールの学習について”, マルチメエージェントと協調計算III, 日本ソフトウェア科学会 MACC '93 ,pp.121-136(1994)

[2] 畝見達夫：“強化学習エージェントの集団行動”, マルチメエージェントと協調計算III, 日本ソフトウェア科学会 MACC '93 ,pp.137-150(1994)

[3] 石田亨: “マルチエージェント X モデルに基づく分散資源割当—ATM 網における帯域割当—”, 情処学マルチメディア通信と分散処理研報 DPS62-4 (1993)

[4] J.H.Holland, K.J.Holyoak, R.E.Nisbett,P.R.Thagard : “Induction:Process of Inference, Learning, and Discovery”,the MIT press(1986)

[5] 目黒雄峰 米沢隆利 阿部倫之 服部進実: “分散協調型帰納推論エージェントシステムによるサーバ負荷分散処理” 情処学マルチメディア通信と分散処理研報, 95-DPS-68/95-GW-9 (1995)

[6] 畝見達夫 :“強化学習”, 人工知能誌, vol.9,no.6,pp830-836(1994)

[7] 山田誠二 :“リアクティブプラニング”, 人工知能誌, vol.8,no.6,pp35-41(1993)

[8] 良永和幸, 寺野隆雄 :“分類子システムを用いた環境からの知識獲得”, 人工知能学会全国大会（第8回）, pp.231-234(1994)