修 士 論 文

マルチエージェント系における組織学習を用いた動的環境への適応に関する研究

北陸先端科学技術大学院大学
知識科学研究科知識社会システム専攻

篠田 孝祐

2001 年 2 月

修 士 論 文

マルチエージェント系における組織学習を用いた動的環境への適応に関する研究

指導教官　國藤 進 教授

北陸先端科学技術大学院大学
知識科学研究科知識社会システム専攻

950045　篠田 孝祐

審査委員:　國藤 進 教授 (主査)
藤波 努 助教授
西本 一志 助教授

2001 年 2 月

要 旨

従来のエージェントの学習は，主に個体の行動決定能力の向上を目標として行なわれてきた．マルチエージェントシステムとなってもその目標に大きな変化はなく，集団組織を対象とした学習モデルは情報の共有化，獲得報酬の分配などでいくつか設計されたが，それらの多くの最終的な目標はあくまでも個体の能力設定よって協調行動を獲得していくボトムアップな協調システムである．

また，従来の学習では一度学習したことを忘却することで環境の変化に対応してきた．それは，すべての主体が学習データを記憶することは非常に無駄が多いためである．また，不完全知覚たるエージェントには環境の変化を認識することは困難であるためのである．しかしながら，この忘却が通常の環境 (Online) でのエージェント学習を利用することを難しくしている．それは，学習した知識が環境で適応できない場合でも，古い知識を忘却するまではエージェントの行動に影響を与える．また，再び同じような環境になったとき一から学習しなくてはならない．

本研究では，これらを踏まえ従来の完全な集中型もしくは分散型システムでなく集中・分散を組み合わせた組織モデルを採用しその学習モデルとして組織論などで論じられている組織学習をマルチエージェントシステムの学習モデルとして採用した．この組織学習を用いたマルチエージェントシステムを RoboCup で使われるサッカーシミュレーションゲームのサッカーチームとして実装しその効果を検証した．

目 次

図 目 次

表 目 次

第1章
はじめに

自律した行動主体としてのエージェントは複数集まることで行なう集合行為は，ここのエージェントの個別的な行為を蓄積したのものと全く異なった性質を示すといわれる．その集合体としての性質は，構成要素であるどのエージェントにも見いだすことが出来なく，全く異なるタイプのエージェントの行為にさえ類似していることがある．このような集合体としての性質は例えば各人の利益だけを考えた利己的な経済活動が許される自由な競争原理の働く市場のような場合においては，多様な性質を引き出す事が可能である．しかしながら，サッカーゲームのような特定の目標が設定され迅速な対応を求められる環境やITS[1]などで研究されている交通誘導システム安全性や確実性を求められる環境においては，単なるエージェントの集合体ではなく確実性のもつシステムとして性質を求められる．つまり，環境を支配的な立場から操作するのではなく，環境と調和し常に適応できるシステム設計が社会基盤システムには求められる．

本研究では，マルチエージェントシステムの将来的な社会基盤システムへの適応を目指して常に変化する可能性をもつ環境への適応システムの構築をすることが主たる目的である．そして，この目的を従来のエージェント学習に組織学習の概念を導入することで実現することを課題とした．本研究で提案する組織学習を基底としたマルチエージェントシステムにおける学習モデルが目指すところは以下の3点である．

1. 学習体験の共有化による，短時間での学習の収束
2. 状況の変化に応じた動的組織形成
3. 環境の変化に応じた的確な組織の相転移の操作

[1]Intelligent Transport System(高度道路交通システム)

この課題の実現にあたり具体的には，不完全知覚たるエージェントの部分観測環境下での強化学習と集団組織内での学習の共有化を行なうための学習モデルを提案する．また，本研究の実験環境としては，RoboCup で知られているサッカーシミュレーションゲームを利用し，そのシミュレーション上で動作するソフトウェアエージェントを実装することで学習モデルの有効性を検証したい．

以降，2 章ではエージェント，マルチエージェントの基礎的な事柄について述べ，3 章ではエージェントの学習行為について述べた後本研究で用いる部分観測環境かでのエージェント学習に説明する．そして，それらをふまえてマルチエージェントにおける組織学習モデルを提案したのち，4 章で具体的にサッカーチームを例として説明し，5 章で実験評価を行なう．最後に，6 章で社会基盤システムへの適応モデルについて述べ，7 章において論文の結びとしてまとめと今後の課題についての考察を行なう．

第2章
マルチエージェントシステム

自律した行動主体としてのエージェントは複数集まることで行なう集合行為は，ここのエージェントの個別的な行為を蓄積したのものと全く異なった性質を示すといわれる．その集合体としての性質は，構成要素であるどのエージェントにも見いだすことが出来なく，全く異なるタイプのエージェントの行為にさえ類似していることがある．このような集合体としての性質は例えば各人の利益だけを考えた利己的な経済活動が許される自由な競争原理の働く市場のような場合においては，多様な性質を引き出す事が可能である．しかしながら，サッカーゲームのような特定の目標が設定され迅速な対応を求められる環境やITS[1]などで研究されている交通誘導システム安全性や確実性を求められる環境においては，単なるエージェントの集合体ではなく確実性のもつシステムとして性質を求められる．前者のような環境においては，長期的視点から集合体としてのエージェントシステム (以下，マルチエージェントシステムと呼ぶ) に環境を支配するための発展性が求められる．それに対して，後者では環境を支配するというよりは，相手より優位に立つことや確実性を保つことが主となるため効率を重視した，より短期的な視点から目標をたて堅実にこなす事が求められる．

2.1　エージェントとエージェンシー

マルチエージェントシステムを設計するにあたり，それを構成する要素たるエージェント (agent) とエージェンシー (agency) は重要な概念である．エージェントという用語

[1]Intelligence Transport System

は，いわゆる人工知能といわれる分野のみではなく知能ロボット，CSCW[2]やグループウェア，通信工学に至る広い分野で用いられている．その解釈は非常に困難とされエージェントを明確に位置付けるような定義は今のところ存在しない．そのためエージェントに関する解釈は非常に多岐にわたっているのが現状と言える．例えば，自律分散システムの領域ではそれらシステムを構成する一要素をエージェントと呼ぶ一方で，擬人化されたキャラクタをエージェントと呼ぶ場合もある．このようにエージェントはそれぞれの研究分野において様々な解釈がなされている．

またエージェントの粒度も様々で，小さいものでは神経細胞の集合体やモジュール大きいものでは一人の人間，あるいは人間の集団組織などをあらわす．この粒度もエージェントの意味を考えるにあたって重要な要素となる．

本論文におけるエージェントは，自らの規範をもとに行動する主体とする．その意味を最も適切に表しているのは，行為者または代理人であろう．これらは独自の目的をもち，それを実現するための主体ということができる．具体的には，単独で自律的な動作をする計算上のプロセス，または自己充足的な動作をする計算プログラムいう人工知能分野で基本的とされる立場をとる．

一方，エージェンシーという用語については，ミンスキー (M. Minsky) がエージェントの集合体をエージェンシーと呼んでいる [Misky86]．一方で，単なるエージェントの集合体ではなく，その集合体が集合的な性質によって特徴づけられた場合にエージェンシーと呼ぶ [生天目 98] する意見もある．これは多くのエージェントが単に集まっただけでは全体のタスクを効率的に実行することは困難であるため，集団内部に組織的構造を作り出すことが求められる．これにより複数のエージェントが組織化されることで，一つの高次エージェントを見いだすことが出来る．この集合体としてのエージェントたる高次エージェントをエージェンシーを呼ぶというものである．また，エージェントの集合体がエージェンシーとして機能する場合には，その動作を管理する役割をもつエージェントを必要としない点がエージェンシーのもつ特徴だとされている．本研究でも，後者の立場をエージェンシーとしてとらえることする．

以下，エージェントの様々な特性やその分類について述べる．

[2]Computer Supported Cooperative Work

2.1.1 エージェントの基本特性

エージェントの基本的な特性としては次の４つがあげられる (pp.7, 図 2.1 参照).

自律性 (Autonomy)

エージェントは自己の行動や内部状態を制御する仕組み (問題解決機構) を持ち，他のシステムから干渉を受けることなく自律的に動作する．つまり，エージェントは自らの知識や種々の情報を利用することで問題の判断や解決をすることができる．この性質はエージェントを特徴付ける主たる特性とされている．

社会性 (Social Ability)

エージェントは，他のエージェントや人間との相互作用が不可欠である．そこでは，双方の対象が相互に理解可能な言語 (エージェントコミュニケーション言語) やプロトコルが導入され，円滑な情報交換やコミュニケーションが実現される．これによって，たとえば複数のエージェントが一つの作業グループ (組織) を構成して互いに協力しながら (協調的に) 問題解決を行ったりする．

反応性 (Reactivity)

エージェントは自分がおかれた環境 (外部環境) を認識し，そこで生起する様々な変化に対して適切に応答する．すなわち，エージェントの枠組みではエージェントや人間を含む外部環境との相互作用が各エージェントの振舞いに影響を与える重要な要素になっている．

自発性 (Pro-activeness)

エージェントは，単に外部環境に応じて反射的に動作するだけでなく，ある目標を目指して自発的に行動できる．つまり，エージェントは何らかの目標の達成に必要となる処理や作業に対して能動的に参加したりする．これは，前述した自律性に関連した性質であり，外部から与えるデータやイベントによって受動的に動作するソフトウェアとの違いといえる．

2.1.2 エージェントの分類

エージェントに要求される諸特性を背景として，エージェントという言葉も様々な意味合いで用いられているのが現状である．ここではいくつかの例を挙げ，それぞれの持つ意味合いの違いについてみていく．(pp.7, 図 2.1 参照)

利用者エージェント (User Agent)

コンピュータシステムとそれを利用する人間とのインタラクションの仲介役となるエージェントで，インタフェースエージェント (Interface Agent) と呼ばれることもある．

ソフトウェアエージェント (Software Agent)

コンピュータシステムのソフトウェア環境で稼働するエージェントの総称であり，利用者の代理人として働いたり，種々の作業を支援したりする．インターネットのようなコンピュータネットワーク環境を対象とする場合には，特にネットワークエージェント (Network Agent) とか, ウェブロボット (Web Robot) などと呼ばれることもある．

モバイルエージェント (Mobile Agent)
移動するという概念を持ち合わせたエージェント．計算機から計算機へと物理的な空間を移動することができる．ウェブロボット (Web Robot) などもモバイルエージェントの一種であると考えることもできる．

知的エージェント (Intelligence Agent)
エージェント内部に問題解決や学習をおこなう仕組みやそのための知識を持つことで，知的に振る舞うエージェントの総称である．その実体がコンピュータ環境で動作する場合には，知的ソフトウェアエージェントと呼ばれる．

自律エージェント (Autonomous Agent)
エージェントの基本特性である自律性に着目した場合の呼び方である．自律的推論システム構成要素としての意味合いで使われる場合もある．

協調エージェント (Cooperative Agent)
相互協力しながら動作するエージェントのことで，分散人工知能の分野ではマルチエージェント (Multi-Agent) ということもある．また協調分散システムの構成要素という意味合いで使われる場合もある．

ビリーバブルエージェント (Believable Agent)
エージェントシステムによるアプリケーションの一種．迫真性を備えたエージェント．情感を持ち (もしくは持っているように感じられ)，通常のソフトウェアをこえた存在感を有する．A-Life プログラムなどによってつくられた疑似生物はこれに近い立場をとっている．これは，エモーショナルエージェントとも呼ばれる．

2.2 エージェントとマルチエージェント

マルチエージェントとは，2.1 節で述べたエージェントが多数集まり形成された集団組織のことをさす．それらエージェント同士は，原則的に協調や競争行為が可能であり，同一の目的をもった組織構造を形成する場合もあるが．特別に協調や競争関係を強いられ必要はない．マルチエージェントがもつ意味は「単なる同一環境上に存在するエージェントの集団」でありその意味ではミンスキーの言うところのエージェンシーと同じと言える．以下，エージェントモデルとマルチエージェントモデルについて述べる．

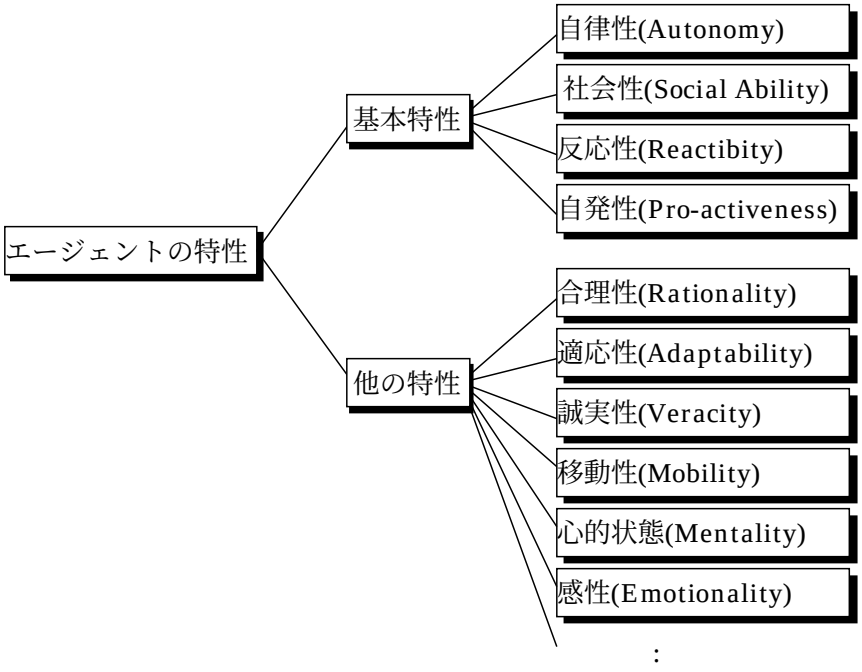

図 2.1: エージェントの諸特性

2.2.1 マルチエージェント環境でのエージェントの振る舞い

エージェントは，自己の価値基準に基づき自らの行動を最適にすることを主眼において，合理的戦略を選択する．この自己の価値基準は，自身が環境のなかで唯一の存在として行動している場合には，自己の選択に影響を与えるのは環境のであるために環境に対して合理性を充たせばよい．個体の最適行動をエージェントの**個人合理性**の追求と呼ぶ．

しかしながら，マルチーエージェント環境では，エージェント間に協調・競争関係が発生する場合がある．このような場合では，集団としての目的適合性を追求することも重要である．この時，エージェントが集団の一員として行動する上で重要とされるのは**集団合理性**である．この２つの合理性は次のように定義されている [生天目 98]．

個人合理性

個人のエージェントの利益や目的を最適にすること

集団合理性

集団の利益や目的を最適にすること，あるいは他のエージェントの効用の犠牲にすることなく特定のエージェントの効用を高める余地のない状態 (パレート最適) にすること

エージェントに与えられる条件が上記の二つの合理性を満たすのであれば，ここのエージェントは行為の選択で迷いが生じる (行動選択の競合が発生する) ことはなくなる．しかしながら，二つの合理性の条件が一致することは非常に稀であるといえる．そのため，集団組織に属するエージェントは主体は万全な存在ではなく，様々な面において限界がある存在であるという限定合理性に基づいて行動するとされている．つまり限定合理性的な主体は，自分ですべてを行なうのではなくその判断を含めて他の主体に委託，あるいは主体同士で協力をする．

限定合理性に基づいてエージェントが行動する場合，取り巻く環境が複雑になるにつれて集団組織における意思決定が意思決定や問題解決に要求される能力と主体単独の能力の間の格差の拡大という背景を受けて複数の個体間により行なわれる．これは，組織的な活動を行なうことでより有効な意思決定が導かれることが期待されるためであると言える．しかしながら，実際にはその逆で単独な意思決定者よりも劣悪な決定を行なうこともある．それは集団としての質の高い問題解決能力を発揮するには限定された自分のもつ情報処理能力を相互補完し合うことでここのメンバの働きを有機的に結合させる工夫が必要とされる．

2.2.2 エージェントモデル

エージェントは，自身を取り巻く世界 (以下，環境という) について知覚やその知覚を基礎とする認識イメージをもって環境に対して様々な働き掛けを行なう．個々のエージェントがもつ認識イメージや環境に対する働き掛けは，そのエージェントの内部属性依存する．エージェントは，ありのままの環境を認識することはなく，自分にとって望ましい観点から認識をしてその内容を評価するための内部規範を求められる．また，その規範を環境へと適応するための学習も必要とされる．

図 2.2 で示しているのは，一般的な問題解決の流れを示したものである．問題解決においては，ある問題を種々の知識を利用する問題解決器 (推論エンジン) で問題をとき目的の解を導出するという流れをもつ．これは知識システムにおける推論処理の基本プロセスであり，エージェントも自身の知識をもちいて問題を解決することから知識システムのひとつと言える．つまり，エージェントも自身のもつ知識により解を導く．この時用いる知識は，主観的な認識イメージにそって把握し情報であり，その主観によって認

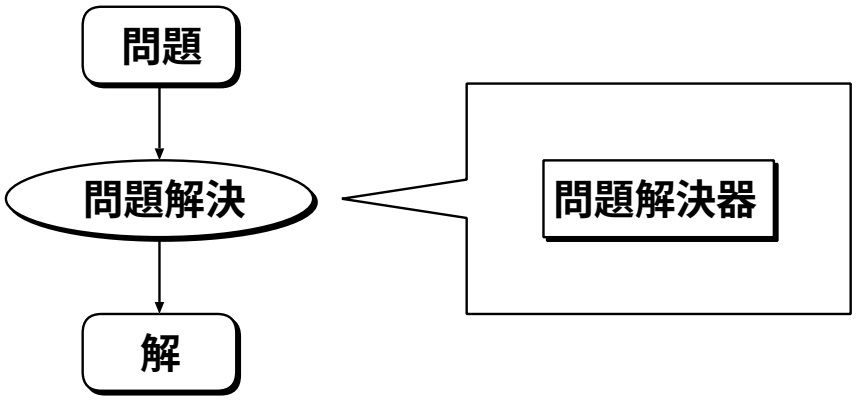

図 2.2: 単一のエージェントによる問題解決

知した問題を問題解決器にて内省することで解を導くためにエージェントは客観的な合理性をもつ主体と言うよりは，主観的合理性をもつ主体であると言える．

次に，pp.10 の図 2.3 ではエージェントの構成を示している．この図のエージェントがもつ知識には以下の三種類の知識が存在する．

環境知識 外部環境に関する知識

自己知識 自分に関する知識

他者知識 他のエージェントに関する知識

エージェントは，この知識をもとに自らの行動を決定する推論をもつことで内部モデルを構成し，いわゆる自律性を確立する．エージェントが主体的存在として自律的かつ自己充足的であるには次の機能が要求される．

- 固有の目的とそのための評価基準
- 目的を実現するために必要な知識
- 環境の変化に適応していくための学習機能

エージェントは，自己の行為の決定メカニズム (意思決定機構) をもつことで合理的な意思決定を可能としている．一方で，自身の価値や目的を表す内部規範を学習によって環境への適応を試みることで，自らの行為を進化させる．

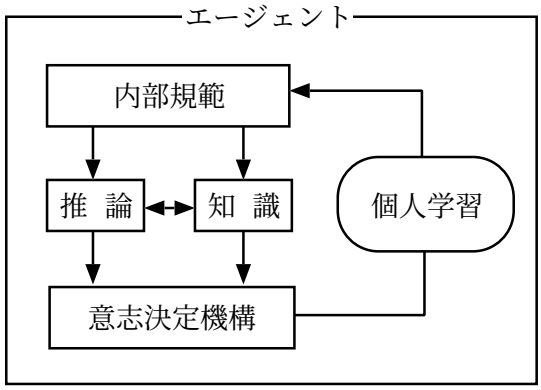

図 2.3: エージェントの機構

2.2.3 マルチエージェントモデル

2.2.2 節に示したエージェントに比べマルチエージェントはもう少し複雑である．マルチエージェントには以下の２つのタイプに分類して考えることが出来る．

1. 協調問題解決
   複数のエージェントが共通の目的をもち大域的な効用を考慮しながら互いに協力して集団組織を構成し問題解決を行なうシステム．このシステムにおいては，トップダウン的に問題を分割しエージェント間で分担する問題分割や，複数の問題解決を寄せ集めて目標とする解を導出するプロセスが必要とされる．場合によっては，そのプロセスを担う専用の問題分割・統合処理器を用意する必要がある．
2. マルチエージェントシステム
   合理的かつ自律的に動作する複数のエージェントが，船体として好ましい均衡を維持しながら各々の目標を達成するシステム．このシステムにおいては，各エージェントの目標が必ずしも共通であるとは限らない．

前者は，集団組織を構成する主体であるエージェントは，自律的な存在ではなく，集団目的を実現するために全体に従属的な存在として扱われる．一方，後者は集団の目標を達成するという前提は前者と変らないが，その達成にあたって自己の目標を自律的に設定し解決する．つまり，前者は集権的な意思決定機構の一部としてエージェントが動作するのに対して，後者は分権的な意思決定機構としてエージェントは動作する．

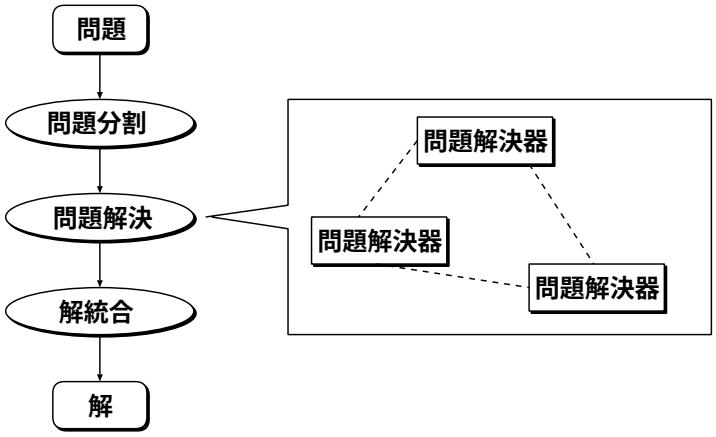

図 2.4: 複数のエージェントによる問題解決

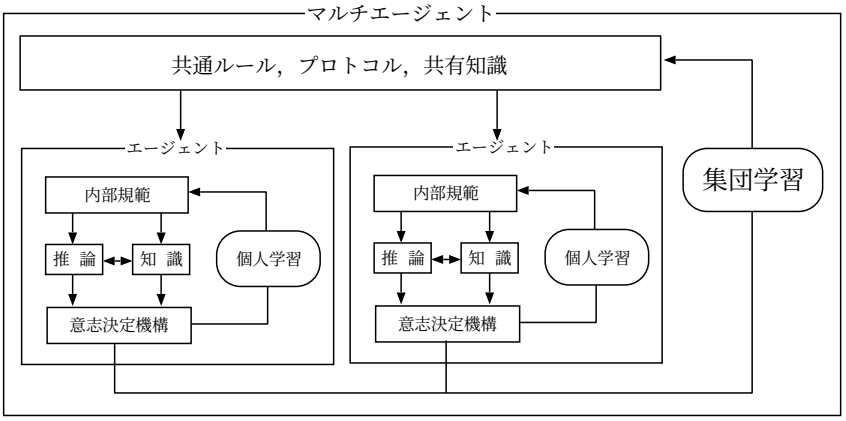

図 2.5: マルチエージェントの機構

図 2.5 では，全体のシステムとエージェントの関係を中心にマルチエージェントの機構を示している．マルチエージェントにおいて各エージェントは自律的な存在であると同時に全体の目標への寄与も求められることからマルチエージェントの全体の目標との調和を図るための機能が必要となる．図 2.5 にある共通ルール，プロトコル (規範) などは，個体と全体との目標を円滑に達成するために必要な個々のエージェントの役割や責務を規定するものである．それらは，集団学習を通じてより進化的にまとめあげられる．

2.3 階層的マルチエージェントシステム

従来のマルチエージェントシステムでは，基本的に集団組織に属するエージェント同士は対等な立場であり，利己の利益を最適化を目指すものとして設計されてきた．しかしながら，システムの大規模化や環境が複雑化に伴い分散によって利点とされた知識共有などでは通信コストなどが大きな問題とされている．そのため，現在ではマルチエージェントシステムなど様々なシステム設計で集中制御と分散制御の在り方が見直され，システム設計容易さや可用性の面から集中と分散制御を組み合わせた緩やかな結合を持った制御システムが注目されている．

本節では，他のエージェントとは異なる立場にありその集団の中でより上位のノードに位置するエージェントが存在する集団を形成するマルチエージェントシステムについて述べる．本論文では，そのような立場にあるエージェントのことを**上位エージェント**と呼ぶことにする．また，上位エージェントによって行動選択に影響を受けるエージェントを**下位エージェント**と呼ぶことにする．このようにマルチエージェントシステムに階層を導入することで次のような利点が生じると考える．

- 上位と下位のエージェント間でのプロトコルを設定しておけば，それぞれの構成の変更は自由である．
- 下位のエージェント間だけでは調整のつけにくい組織全体のバランス調整などを上位のエージェントに委任することが可能となる (タスクの分散).

以下，上位エージェントによる集団行為の操作性について述べる．

2.3.1 上位エージェントによる集合行為の操作性

上位エージェントは，共通ルールなど集団組織が持つ性質を決定する権限を持つエージェントである．上位エージェントは，自分の支援環境下に 0 体以上の下位エージェントが存在するとき機能する．下位エージェントは，上位エージェントが想定する環境下へ直接的関与可能な存在である．そして上位エージェントは，環境への直接的関与はしないもとし，下位エージェントを通じて間接的に関与する．そのため，上位エージェントと下位エージェントの関係は，下位エージェントが行なう意思決定や情報獲得，学習行為などを円滑に進めるための支援するという関係を持つものとする．上位エージェン

トは，環境の変化に応じて共通ルールを設定することで，下位エージェント郡の均衡状態を質的に異なるものへと変更する**相転移**を生じさせることが可能となる [生天目 98].

2.3.2 組織構造の操作性

上位エージェントが存在することで，自律的な存在であるエージェントは，自己の自由な存在が多かれ少なかれ拘束される組織を持つことになる．エージェントが組織に属する利点は次のような点である．

- 個人では充たし得ないニーズや共通利益
- マルチエージェントの集合行為の自己組織化

エージェント個体のみで解決するより，複数の個体の協調によって解決することで問題の解決に必要なコストを減らすことや発生する可能性のあるリスクの回避が出来る場合がある．サッカーゲームを例にとるならば，センタ付近から得点を狙うときドリブルでゴールまで運ぶより，パスをつないでゴールに近い味方プレイヤがシュートを打つほうが成功する割合が高いのは用意に想像がつく．また，図 2.6 のような組織内のエージェント同士で協調関係を持つ場合．環境の変化や状況に応じて組織内の協調関係を変化させる必要がある．

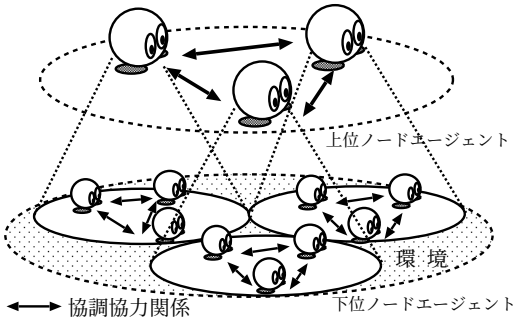

図 2.6: 組織構造をもつマルチエージェント

一般にフラットな組織構造をもつ集団組織では，エージェント間の情報共有などネゴシエーションにより自己組織化が期待されるが，自己組織化を安定状態にするには多くの試行と時間が必要とされる．そのため，実システムへの運用には場合には危険が伴う．しかしながら，階層的組織を持つことで組織的な学習行為を行なうことにより知識の蓄

積と運用が期待できる．そのため，上位エージェントは下位エージェントが行なう集合行為の自己組織化を組織構造的な面からの支援が可能となる．

次章では，本章で述べたエージェント・マルチエージェントを踏まえ，それらの学習行為を中心に述べる．

第3章
マルチエージェントシステムにおける組織学習モデル

本章では，マルチエージェントにおいての学習行為について述べる．前章でも述べたようにマルチエージェントシステムにおいて学習は自身のもつ規範などを変化させることで動的に変化する環境に適応するために必要な行為であるといえる．しかしながら，個々のエージェントが学習によって環境に最適な状態になることは，自身が存在する環境が複雑になればなるほど困難となる．その理由としては，以下の点が考えられる．

1. 学習が環境からのフィードバックに基づいて行なわれる
2. エージェントは，環境をありのまま認識するのではなくその主観によって選別された情報をもとに意思決定を行なう主体である
3. 環境全体を把握することは困難 (不完全情報下での意思決定)

以下，これらの点を考慮にいれてマルチエージェントシステムにとって適切な組織学習を提案するにあたり，従来のエージェント学習の概略及び問題点を述べる．

3.1　マルチエージェントシステムにおける学習

実ロボットやエージェントが様々な環境のもとでも自律的動作を獲得するため手法として強化学習が研究されている．この強化学習とは，主体が試行錯誤を繰り返し最終的に目標を達したときに得られる報酬のみから観測状態に対する正しい行動出力を学習する手段である [Kaelbling96]．設計者が目標を設定しておけば，目標への到達の仕方は自動的に獲得されるため，未知環境下での学習に適しており，不確実性や未知のパラメー

タが多い環境では人がプログラムした行動より優れた解を導くことがある [木村 99b].

強化学習はこれまで離散マルコフ決定過程 (以下 MDPs[1] と呼ぶ) 環境下で研究されてきた．MDPs 環境下では，主体は環境の状態一つ一つを区別することができるために，環境の状態遷移を確率的に求めることが可能である．しかしながら，マルチエージェント環境でのエージェントは環境から得られる情報が不完全であり不確実な場合が多い．そのためすべての状態を区別することは不可能であると言える．また，同一の環境に学習主体が自身以外にも存在する場合，状態遷移を確率的に求めることは困難である．すなわち，実問題では環境が MDPs である可能性は非常に低い．

そのため，現在では部分観測マルコフ決定過程 (POMDPs[2]) を対象とした強化学習が広く行なわれている．以下，一般的な強化学習と部分観測環境かにおける強化学習の違いにつて述べる．

3.1.1 強化学習

強化学習とは，試行錯誤を繰り返すなかで，最終的にその行動の系列の良し悪しを示す情報のみから行動決定戦略を追求する教師無し機械学習の総称である．強化学習についての理論的成果が最も多く蓄積されている問題クラスはマルコフ決定過程である．その理由としては，環境を数学的に扱いやすくするためである．以下，マルコフ決定過程について説明し，また同様に古典的な手法である Q-learning と ProfitSharing について述べる．

3.1.1.1 マルコフ決定過程モデル

マルコフ決定過程によってモデル化された環境は，次のような要素からなる．

- 状態の集合 $\mathcal{S}$
- 行動の集合 $\mathcal{A}$
- 報酬関数 $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{N}$ (ここで $\mathcal{N}$ は実数の集合を表す)
- 状態遷移確率関数 $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{N}_P$ $(\mathcal{N}_P = \{n \mid 0 \leq n \leq 1\})$

[1] Markov Deceision Process
[2] Partially Observable Markov Deceision Process

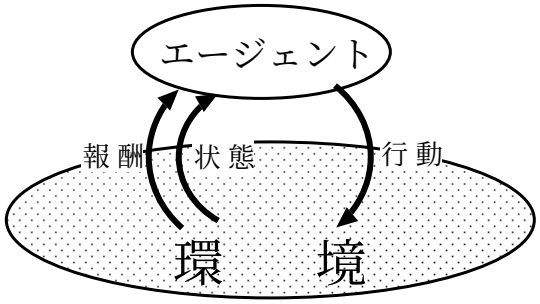

図 3.1: MDPs 環境下の学習モデル

状態遷移確率関数 $P(s,a,s')$ はエージェントが環境がある状態 $s \in \mathcal{S}$ において行動 $a \in \mathcal{A}$ を実行したとき，状態 $s' \in \mathcal{S}$ に遷移する確率を表すものである．また，報酬関数 $R(s,a,s')$ はそのときにエージェントに確率的に与えられる報酬の期待値を表す.

エージェントはこのようにモデル化された環境の中で，環境から状態 $s \in \mathcal{S}$ を入力として与えられ，その入力をもとに行動 $a \in \mathcal{A}$ を出力する．その結果，遷移した状態 $s' \in \mathcal{S}$ と報酬 $R(s,a,s')$ が次の入力として与えられる (図 3.1)．環境での時間はこの入力-出力のサイクルを 1 単位として離散化され，本論文ではこれを**クロック**と呼ぶ.

エージェントは，次のように表される政策を用いてその行動を決定する.

$$\text{政策 } \pi : \mathcal{S} \to \mathcal{A}$$

政策 $\pi(s) = a$ は，エージェントが状態 $s \in \mathcal{S}$ で行動 $a \in \mathcal{A}$ を実行することを表す. マルコフ決定過程では，他のどんな政策よりも優れた，あるいは同等な政策が少なくとも一つ存在する．これを最適政策 π^* という．また，本論文では，それを用いることによって必ず報酬を得ることの出来る政策を合理的政策と呼ぶ.

3.1.1.2 Q-Learning

Q-Learning はマルコフ決定過程を対象とした代表的学習手法である．この手法ではある状態においてある行動を選択するという対を考え，その評価を割引報酬の累計として計算する．割引報酬とは，状態行動対で得られた報酬からそれ以前に訪れた状態行動対に割り引かれた報酬のことをさす．この状態行動対の評価は次の式で表され Q 値と呼ぶ.

$$Q^*(s,a) = \sum_{s' \in S} P(s,a,s')(R(s,a,s') + \gamma \max_{a'} Q^*(s',a)) \tag{3.1}$$

Q-Learning の目標はこの Q 値を推定するところにある．エージェントは状態遷移確率関数 $P(s,a,s')$ や報酬関数 $R(s,a,s')$ についての予備知識を持たない．つまり，Q 値の推定は $P(s,a,s')$，$R(s,a,s')$ を推定する意味を持つ．そのため，Q-learning は環境同定型の学習方法に分類される．

図 3.2 は，Q-learning の学習アルゴリズムを示す．学習率 α は学習速度を表し，一般に学習が進むに従って値が小さくなるように設定される．また，割引率 γ はある時点からどれだけ先の状態で得られる報酬まで考慮するかを表す．

1. エージェントは環境の状態 s を知覚する．
2. エージェントは任意の行動選択方法に従い行動 a を実行する．
3. 環境から報酬 r を受け取る．
4. 状態遷移後の状態 s' を観測する．
5. 次の更新式によって Q 値を更新する．

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha\left\{r + \gamma \max_{a'} Q(s',a')\right\}$$

   ここで，α $(0 < \alpha \leq 1)$ は学習率，γ $(0 \leq \gamma < 1)$ は割引率である．
6. 1. に戻る．

図 3.2: Q-learning のアルゴリズム

Q-learning ではすべての状態行動対を十分な回数選択すれば，必ず最適な Q 値が求まることが証明されている [宮崎 97]．ただし，学習環境がエルゴート性を有する離散有限マルコフ決定過程である必要があるが，この条件下での最適政策の獲得が保証されていることは Q-learning の利点である．

しかしながら，Q-learning では最終的に得られる報酬が，割引報酬として徐々にすべての状態行動対に対する Q 値に伝搬されるのを待つアルゴリズムであるため，学習が収束するまでに膨大な行動回数を必要とする場合がある．また，学習が収束するまである程度の解でさえ得られない場合がある．さらに，学習率や割引率などのパラメータ変化に対して Q 値の変動が大きく変るため，実際に適用するにはパラメータチューニングが必要となる．

また，学習過程における行動選択には，Boltzmann 選択，max 選択とランダム選択の組み合わせ，ルーレット選択などがあるが，Boltzmann 選択によって良い結果を得られ

ることが経験的に知られているが，学習後の行動選択には，マルコフ決定過程下であれば max 選択を用いることにより最適選択を得ることができる．しかし，非マルコフ決定過程下では確率的政策を取ったほうが良い解をえることが経験的に知られている．

3.1.1.3 Profit Sharing

Profit Sharing は Q-learning のような環境同定型の手法ではなく，経験強化型と呼ばれる学習手法の一つである．Profit Sharing を説明するに辺り以下の用語を定義しておく．

ルール ある状態 s で選択可能な行動 a の対．この時ルールは sa で表わされる

エピソード 初期状態から報酬を得るまで，あるいは報酬を得た直後から次の報酬を得るまでのルール系列

無効ルール，有効ルール エピソードで，同一知覚状態にたいして異なるルールが選択されているとき，そのあいだのルール系列を迂回系列と呼び，すべてのエピソードで常に迂回系列上にあるルールを無用ルール，それ以外を有効ルールと呼ぶ．

Profit Sharing ではそれぞれのルールには評価値が付加されており，エピソードに含まれるルールの評価値を強化することによって学習が行われる．Profit Sharing の学習アルゴリズムを pp.19 図 3.3 に示す．

1. エージェントは環境の状態 s を知覚する．
2. エージェントは任意の行動選択方法に従い行動 a を実行する．
3. 環境から報酬 r を受け取る．$r > 0$ ならば，次の更新式を用いてエピソードに含まれるルールを強化する．

$$w(s_i, a_i) \leftarrow w(s_i, a_i) + f(r, i)$$

ここで，$w(s_i, a_i)$ はエピソードを構成するルール系列の報酬から i クロック前のルールの重みを表す．また，$f(r, i)$ は強化関数といい，あるルールが報酬を得られた時点からどれだけ遡って用いられたのかを引数とし，それに対する強化値を返す．

4. 手順 1 に戻る．

図 3.3: Profit Sharing のアルゴリズム

Profit Sharing は無効ルールの強化を抑制することにより，次のような条件下で合理的な政策を獲得することが出来ることが証明されている．

$$\forall i = 1, 2, ..., W. \quad L \sum_{j=i}^{W} f_j < f_{i-1} \tag{3.2}$$

ここで，W はエピソードの最大長，L は同一感覚入力下に存在する有効ルールの最大個数を表す．

Profit Sharing では，エピソード単位でルールの強化を行うため，一回の報酬で多くのルールが強化され，学習の効率が良い．また，学習環境がマルコフ決定過程であることを仮定していないため，さまざまな環境での応用が期待できる．実際，マルチエージェント環境では Q-learning よりも Profit Sharing の方が良い結果を得ることができるという報告もある [宮崎 99b, 荒井 98]．しかし，Profit Sharing では環境がマルコフ決定過程であっても，学習の結果として得られる解が最適であるとは保証していない．

Profit Sharing の学習過程における行動選択の方法としては，ルーレット選択が良い性能を示すことが経験的に知られている．また，ルーレット選択は，非マルコフ決定過程環境下での学習後の行動選択方法としても有効である．

3.1.1.4 強化学習の問題点

強化学習における問題点としては，次のようなものが考えられる．

1. 遅延報酬　エージェントの行動は，即時報酬と次状態に基づいて決定される．よって，次状態以降に得られる報酬 (**遅延報酬**) を考慮に入れた学習が必要である．
2. 最適行動選択と行動最適化のトレードオフ　強化学習では，正しい行動を直接的に教示されるのではなく，環境からのフィードバックにより行動に対する評価を決定する．そのため，学習が収束するまでの行動に対する評価はある程度の信頼性はあるが，必ずしも最適なひょうかであるとは言えない．そのため，行動選択においてその段階の最適な行動を選択することと，最適ではない行動と選択することで行動最適化を図ることは，トレードオフの関係にある．

3.1.2 部分観測状態における強化学習

部分観測マルコフ決定過程 (以下，POMDPs と呼ぶ) とは，不完全知覚のために実際には異なる環境の状態を同一の状態として置換される可能性を有する問題クラスである．このような問題では従来の学習は困難とされている

本節では，部分観測状態における強化学習の問題点を考えエージェントに必要な強化学習について述べる [木村 97].

3.1.2.1 POMDPs 環境下での強化学習の問題点

POMDPs 環境下では，不完全知覚により異なる状態を同一の状態と知覚してるとき混同が生じているという．この混同が生じることにより，従来のアルゴリズムによる学習は妨げられている．POMDPs 環境下での強化学習の問題点としては以下のような点があげられる．

1. 状態の価値の混同
   価値の高い状態と価値の低い状態が同一の状態として知覚されること．とくに Q-learning では，状態の価値の混同により，合理的政策を学習することが出来ない．
2. ルールの有効性の混同
   有効ルールと無効ルールが同一のルールとして知覚されること．このルールの有効性の混同により，Q-learning では報酬から近いルールほど強化されることから 合理的政策を得ることが難しく，Profit Sharing ではルールを強化する強化関数は一般的に報酬から遠いルールほど小さく出力されることから Q-learing と同じく合理的政策を得ることが困難である．

3.1.2.2 POMDPs を対象とした学習手法

前節で述べた問題点などを考慮した POMDPs 環境下での学習手法としては次のようなものがある．

- [Util Suffix Memory(USM)][McCallum95]
  過去の履歴を木構造で表現し，それぞれの葉ノードを内部状態とすることで混同している

状態の分離を行なう手法．それぞれの葉ノードに対して Q 値を Q-learning によって学習し，その結果を統計的に検定することで非マフコフ性を排除するのに十分な履歴の長さを得る．

この USM では十分な履歴を用いれば POMDPs に属する問題を MDPs に属する問題として扱うことが可能であるという利点をもつ．しかしながら，最悪の場合行動を無視したとしても $O(n^L)$ という膨大な記憶容量が必要となる．また，状態分離に統計的な手法を用いるためにかなりの試行回数を必要とする上，木構造の履歴であるために，確率的な状態遷移を扱えないという問題点をもつ．

- [確率的傾斜法][木村 96, 木村 99a]

  それぞれの観測において報酬を最大化するように行動を選択する確率分布を形成することを目的とした学習手法．確率傾斜法による強化学習アルゴリズムの一般形を pp.55 の図 A.1 に示す．

  確率傾斜法では，ある観測においてある行動を選択する確率を政策と呼び，主体のもつ内部パラメータを変化させることで確率的政策を変化させる．また，報酬獲得に関係ない行動を打ち消すことで関係した行動だけが強化され，行動の履歴を強化するために報酬の獲得の遅れもある行動強化される．この手法は報酬を受け取った時点で今までの経験を強化することから経験強化型の学習アルゴリズムとして分類される．

  この手法の利点としては，確率的政策を用いることにより，混同が起きている状態から確率的に脱出することが可能であるという点である．しかしながら，確率関数による出力のため解の均質化がおこり一定以上にはならないという理論的な限界をもつ．また，ルールの混同が多く含まれる環境では効率的ではない．

- [合理的政策形成アルゴリズム][宮崎 99a]

  最適政策を求めるのではなく，合理的政策の獲得を目的とした学習アルゴリズム．合理的政策形成アルゴリズムの学習手順は pp.56 の図 A.2 に示す．

  このアルゴリズムでは，1 次記憶と 2 次記憶の 2 種類の記憶領域を用意し，学習主体は行動を出力する毎に 1 次記憶に行動を上書きし，報酬を得た時点で 1 次記憶の情報を 2 次記憶に複写する．これにより 2 次記憶には合理的ルールのみが記録され，合理的ルールが判明している感覚入力を近くしたときにはそのルールを用いて行動し，そうでない場合には環境を探査するための行動を出力する．この時の探査戦略としては POMDPs 環境下ではランダム探査が有効とされている．

  このアルゴリズムの利点は，学習の収束が早く，学習に要する行動数が少ないことである．また，非常に少ないメモリで学習可能である．しかしながら，合理的政策が獲得できなかった場合には，現在の政策を放棄し学習をやり直すために，実ロボットなどロバスト性を求

められる問題への応用にとっては大きな欠点と言える．また，合理的政策が存在しない環境や確率的な状態変化も全く扱えない．

3.1.3 POMDPs環境下のエージェントの強化学習

これまで述べたことをふまえ，本論文では図 3.4 を学習アルゴリズムとして利用する．このアルゴリズムを説明するにあたり以下の用語を定義する．

履歴

POMDPs 環境下では不完全知覚であるために状態の混同などを考慮にいれた学習が必要である．そのため同一知覚状態を区別する手法として，過去の知覚を蓄積した**履歴**を用いる．このアルゴリズムでは状態行動対ではなく，履歴行動対で学習が行なわれる．

履歴は，タイムステップ t のときの履歴を H_t，知覚状態 $s \in \mathcal{S}_O = \{s^1, s^2, ..., s^n\}$ に対応する要素を $h_{(t,s)}$ としたとき次のような式で表される．

$$H_t =< h_{(t,s^1)}, h_{(t,s^2)}, ..., h_{(t,s^n)} > \tag{3.3}$$

$$h_{(t,s)} = \begin{cases} \delta^{t-laststep(s)} - \delta_{threshold} & (\text{過去に状態 } s \text{ を経験しているとき}) \\ 0 & (\text{過去に状態 } s \text{ を経験していないとき}) \end{cases} \tag{3.4}$$

このとき，δ は記憶の減衰率 $(0 < \delta < 1)$，$laststep(s)$ は，ある知覚状態 s を最後に経験したタイムステップを示す．つまり，履歴は過去の知覚最も新しい状態を $\delta^{t-laststep(s)} - \delta_{threshold}$ とした数値ベクトルとして表される．なお，知覚状態 s はエージェントの主観によって構成された内部情報のベクトルである．

シナリオ

Profit Sharing では，選択したルールの系列記録していただけであったが，今回は過去の状態を蓄積した履歴と共にそれぞれのタイムステップで選択したルールを共に記録する．本論文では，これを**シナリオ**と呼ぶ．このシナリオをもとにエージェントは学習を行なう．

図 3.4 の t はタイムステップを示す変数．N は学習したエピソードの数である．また，$\mathcal{A}$ は行動の集合を表す．

エージェントは，t における環境状態 x_t を観測し知覚状態 s_t を生成する．次に，式 3.4 を用いて履歴の生成を行なう．そして，学習初期には学習が進んでないために，初期状態の確率選択にしたがって行動を選択し，ある程度の学習が進んだ後には以下のルールを用いて現在の状態から選択可能な行動をすべて評価する．

procedure　学習アルゴリズム
begin
　$t = 1$; $N = 1$
　do
　　環境の状態 $x_t \in \mathcal{S}_O$ を観測する.
　　foreach $s \in \mathcal{S}_O$　　//履歴を生成
　　$h_{(t,s)} = \delta^{t-laststep(s)}$
　　if $N > enoughNum$ **then**
　　foreach $a \in \mathcal{A}$　　//適用可能なルールを評価
　　$V(t, x_t a) = rw(x_t a) \times HS(t, x_t a)^e$
　　任意の行動選択法を用いてルールを選択，行動実行.
　　else
　　初期設定に従いルールを選択，行動実行
　　選択したルール $x_t a_t$ と履歴の対を履歴リストに追加
　　if 報酬を得た **then**
　　for 1 **to** t
　　$rw(x_t a_t) = rw(x_t a_t) + f(r, l_N - t + 1)$
　　if $l_N < \frac{\sum_{i=1}^{N-1} l_i}{N-1} \times \eta$ **then**
　　for $i = 1$ **to** t
　　foreach $s \in \mathcal{S}_O$
　　$rh_{(x_i a_i, s)} = (1 - \alpha) rh_{(x_i a_i, s)} + \alpha m_{(t,s)}$
　　$t = 0$
　　$N = N + 1$
　　履歴リストを空にする.
　　問題を初期状態へ戻す.
　　$t = t + 1$
　while　学習が未収束
end.

図 3.4: POMDPs 環境下での学習アルゴリズム

$$HS(t, rule) = \sum_{s \in \mathcal{S}_O} \left(h_{(t,s)} \times rh(rule, s) \right) \tag{3.5}$$

$$V(t, rule) = rw(rule) \times HS(t, rule)^e \tag{3.6}$$

$V(t, rule)$ はタイムステップ t における $rule$ の総合評価，$rw(rule)$ は $rule$ の重み $HS(t, rule)$ タイムステップ t における $rule$ の履歴スコア，そして e $(e >= 1)$ は履歴による評価が総合評価にどれだけ影響するかを表わす.

また，学習の初期段階の $enoghNum$ は環境で想定される状態により大きく個なるが，あらかじめ設計者が設定しておくものとする．

行動を実行した後，そのとき採用したルールと履歴の対をシナリオに追加しておく．この行動の結果報酬が得られなかったときには，タイムステップを一つ進めて，状態観測からの流れを繰り返す．報酬が得られた場合には強化関数を用いてシナリオに含まれるルールの重みを強化する．さらに，次の式によってエピソードの有効性を判定する．

$$l_N < \frac{\sum_{i=1}^{N-1} l_i}{N-1} \times \eta \tag{3.7}$$

この式により，有効であると判定された場合には次の式により，重みつきシナリオテーブルの学習を行なう．

$$rh_{(rule_t,s)} = (1-\alpha) rh_{(rule_t,s)} + \alpha h_{(t,s)} \tag{3.8}$$

学習後，シナリオを空にし問題を初期状態に戻して次のエピソードを開始する．

3.2 組織学習

前節では，エージェント個体が自身または集団の合理性を求めて行なう学習について説明した．本節では，複数の個体が知識の共有化などを行なうことで集団の合理性を求める学習である組織学習について説明する．その組織学習を説明するにあたりまず，組織とは何であるかについて言及し，集団が行なう学習行為について述べる．そして，マルチエージェントシステムにおける組織学習モデルについて述べる．

3.2.1 組織とは

“組織” とは明示的な目標を達成するために合理的に分配され整合化された人間諸力ないし活動であると規定される [佐藤 72]．このように規定することで “組織” を実体概念でなく，あくまでも目的意識的な機能関係を持った機能概念であると捉えることが出来る．こうすることで，それぞれの個体同士は直接的，全体的そして感性的な関係ではなく，目標を媒介として組織形成をする．逆にいえば，一定の行動目標を持った個体同士が集まった集団では “組織” として振舞ったほうがよい結果が期待できるという暗黙的仮定があるといえる．つまり，1) サッカーでのチームはもちろん，2) 偶然事故に遭遇し

た集団や 3) レスキュー活動で同じ災害現場に居合わせた複数の組織 (消防，警察，自衛隊) に属する個体により構成されている集団でも，組織として行動したほうがよりよい結果が期待できるといえる．また，4) 複数の組織が関係する環境問題においても，同一の目標を持つ組織は協力し一つの “組織” として行動したほうがよいといえる．

上であげた 4 つの組織の例をそれぞれ抽象的に表現すると，1) は狭い範囲で単一組織に属する個体が形成されている場合，2) は狭い範囲で一時的な組織が複数組織に属する個体によって偶然に形成された場合，3) は広い範囲で一時的な組織が複数組織に属する個体によって形成された場合，4) は広い範囲で長期的な組織が組織間で形成された場合を示している．これら 4 つの “組織” はそれぞれ Table 3.1 のような特徴を持っている．

表 3.1: 組織のもつ特徴比較

	構成単位	拘束性	所属組織の影響	集団の大きさ
1)	個体	強い	多少あり	全体
2)	個体	弱い	希薄	一部
3)	個体, 集団	やや強い	あり	一部
4)	組織	やや強い	—	全体

3.2.2 集団における学習行為

人間によって形成される組織についての議論は，社会科学における組織論の中で行われている．学習行為や協調システムとしての組織については，その組織論の組織学習 [Schon78] や組織間関係 [山倉 93] において様々な研究がなされている．組織学習とは個人では目標の達成が困難な問題を組織全体としての問題解決能力を向上させながら解決の糸口を創出するための組織的活動である．この組織学習の中では組織には次の 4 種類の学習が存在すると示唆している [Schon78]. ただし，これらはあくまでも仮定であり現象について述べているが，学習過程やそのメカニズムに関しての議論がなされているわけではないので必ずしも厳密な規定はされていない．

- **個体のシングルループ学習** 個体の持つ規範の中で，個体の問題解決能力を向上させる
- **個体のダブルループ学習** 個体の持つ規範を変えながら，個体の問題解決能力を向上させる

- **組織のシングルループ学習**　組織のもつ規範の中で，組織の問題解決能力を向上させる
- **組織のダブルループ学習**　組織のもつ規範を変えながら，組織としてのパフォーマンスを向上させる．

ここでいうところの規範とは，学習の主体である個体もしくは組織がもつ役割 (能力として持てる範囲) だと考えればよい．サッカーを例とするならば，規範が固定ということは，プレイヤはボールを蹴ることしか出来ないとすることであり，シングルループによる問題解決能力の向上とは正確にボールを蹴る能力，蹴ることに関する判断能力を向上させる学習のみであるためにシングルループ学習となる．また，ダブルループ学習のときには規範を帰ることが出来るために規範そのものの学習行動が必要となるためにダブルループ学習となる． 本研究では，これら４つの学習行為が同一の集団に複合的に現れる場合を対象とする．

3.2.3　エージェント組織の知識とは

個体がもつ知識と，組織の持つ知識との違いはどこにあるのであろうか．一般的に，個体が独自に持つ知識を**個体知識**，個体間で共有可能であり個体知識の和として実現される知識を組織知識としている．しかしながら，複数の個体が存在する環境では，成立する組織のタイプや数は計り知れない．また，組織知識を個体知識の和と見なすと組織知識に組織にとって必要ではない個体知識まで含まれることになる．そこで，本論文では組織知識と個体知識を以下のように定義する．

- **組織知識：**組織に属する個体が利用できるルールの集合．
- **個体知識：**個体自身のルールの集合と組織知識にあるルールに必要なパラメータの集合

このように定義することで，組織はそれに属する個体に関わりなく必要な知識を蓄え，個体に提供することが可能となり，また個体も所属組織を変えたり複数の組織を属した上での学習が可能となる．ただし，この定義の問題点は組織知識としての共通ルールをどこで蓄えるかといことだが，システム全体の情報を管理可能なポジションに知識ベースを設計するのが適当であると考える．そのため，個体だけでなくシステムの中で組織または集団を主体とした学習を行なうシステムが必要であると言える．そこで本研究ではマルチエージェントシステムに適した組織学習モデルを導入する．

3.2.4 組織学習とは

組織は個体と個体間関係，２つの要素から成り立つ．そして，２つの間に何かしらの個体間関係が存在すれば，すでにそれは組織といえる．ただし，組織には 3.2.1 節で述べたように 2 つのタイプが存在する．組織学習において個体は，“個体のダブルループ学習” にて目的志向型 (戦術的) 組織での行動学習を行う一方，“組織のダブルループ学習” における機能志向型 (戦略的) 組織での行動学習を行なうを求められる．“組織学習” では，対象とする組織に属する個体は，個体のダブルループ学習を行うと同時に，組織形成と組織知識蓄積のためのダブルループ学習を行う．

言い換えるなら，個体は機能志向型組織のための組織学習と状況の変化に応じた目的指向型組織形成，つまり動的組織形成を行なうことが組織学習だと言える．本研究では，以下のような目的を持って行われる学習行動を “組織学習” と呼ぶことにする．

1. 個体の行動の最適化 (最適化学習)
2. 組織内での学習経験の共有化 (少ない経験での最適化)
3. 状況の変化に応じた目的型組織の形成 (動的組織形成学習)
4. 環境の変化への短時間での適応 (動的環境への適応学習)

つまり，個体が行うべき学習プロセスは以下のとおりである．

- 問題解決に必要な行動ルール及び知識の獲得 (個人のシングルループ学習)
- 協調行動獲得のための学習 (個人のダブルループ学習)
- 状況の変化に適応可能な組織形成モデルの獲得 (組織のシングルループ学習)
- 個体が獲得した経験や知識の組織で利用可能な知識への変換 (組織のダブルループ学習)

これらをもとに，マルチエージェントシステムでの組織学習モデルについてのべる．

3.2.5 組織学習モデル

2.3 節で述べたように，階層的なマルチエージェントシステムでは，上位ノードのエージェントが集団がもつ共通ルールを操作することで，全体の均衡状態が質的に変化する相転移が生じる．また，下位ノードエージェントが行なった学習を，上位ノードエージェントを経て学習 d データを共有することで少ない時間で多数の学習をこなすことが可能となる．本来の組織学習では個々のエージェントが行なった学習行為を抽象化したデータに変換することで組織内での知識の共有化を行なう．しかしながら，エージェントシス

テムにおいてデータの抽象化は難しいことから学習に用いた学習データを上位エージェントを通して共有化することで知識の共有化を行なう．

組織学習とは，最上位である設計者の意図を上位ノードのエージェントを通して下位ノードのエージェントのもつ均衡状態を調整するトップダウン的な調整機構と下位ノードからのレポートをもととするボトムアップ的な調整機構をもつ学習モデルである．図 3.5 は組織学習を概念的に表現したものである．システムの設計者は上位エージェントを通じてシステムの学習などの調整を行なう．そして，上位エージェントは設計者の意図を下位エージェントへ伝え自身がカバーするシステム全体の調整を行なう．この時，上位エージェントは環境への直接的なインタラクションを行なわない．上位エージェントが行なうのはあくまでも，設計者と下位エージェントの仲介と下位エージェントの支援を目的とした行動のみである．つまり，実際の環境もしくはモデル化された環境とのインタラクションを行なうのは下位エージェントのみである．下位エージェントとは，常に環境に対して最適な行動を選択するための学習する主体である．

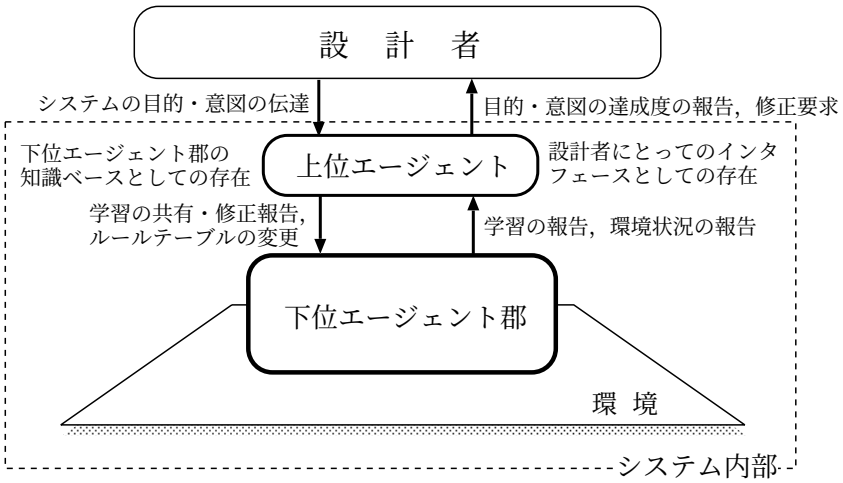

図 3.5: 組織学習の概念図

このような，組織学習の枠組みの中でエージェントは自身の行動を最適化するための学習を行なう．

3.3　動的環境への組織学習の導入

組織学習とは，3.2 節にて述べたように本来は人間によって構成された組織における知識創造のためのメカニズムとして研究されている分野である．つまり，現実社会のダイナミズムに対応可能な組織を作り出すために組織学習はモデル化されてきたと言える．そのため，一つの組織としての行動を求められるマルチエージェントシステムにとって組織学習という学習メカニズムは適しているだけでなく次のような利点があると考えられる．

- 柔軟な組織構造
- 個体の学習体験の共有化 (知識共有ではない)
- 組織知識の蓄積による，環境の変化への適応性の向上

この組織学習で重要なことは，組織構造において上位ノードと下位ノードに位置する個体間の関係が互いを拘束しない緩やかな関係にあるということである．組織は，特に目的型組織は，目的や状況の変化に適応するためには構造は柔軟である必要がある．

これらを踏まえて，次節ではマルチエージェントシステムに組織学習を組み込んだ学習モデルについて述べる．

3.4　本研究における学習モデル

3.1 節，3.2 節では個体もしくは組織を主体とした学習について述べた．本節では，3.1.3 節で述べたシナリオ行動対による個体の強化学習をもとに，マルチエージェントシステムの学習モデルについて提案を行なう．

3.4.1　提案モデルの目標

本研究で提案するモデルの目標は以下の点である．

1. 学習体験の共有化による，短時間での収束
2. 状況の変化に応じた動的組織形成
3. 環境の変化に応じた的確な組織の相転移

ここで述べる状況の変化とは，サッカーゲームを例にとるならば攻守や試合状況のような，ルール選択に影響はあってもルールのテーブル自体には影響を与えない外部環境のことである．一方，環境とは同じくサッカーゲームを例とするならば，対戦相手が変るような質的変化をもたらす外部環境のことで，環境が変化するということはルールテーブルそのものにも影響を与える可能性がある．

また，本研究における学習モデルでは学習後の知識の共有ではなくその学習の要素となる環境状態の履歴と行動の履歴を示すシナリオを共有し，擬似的体験として学習することで異なる性質を持つ個体間での学習を行なうことを可能としている．これは，組織内での能力の均質化を防ぐと共に，複数の組織が存在する環境下での適応を考慮したものである．

3.4.2 シナリオによる学習体験の共有

下位エージェントは，それぞれの学習をシナリオと行動の対をもとに行動決定選択を学習する．この時用いられるシナリオとは，抽象化された状況変化のことである．そのため，このシナリオは他のエージェントと共有することが可能でありこのシナリオと行動対を他のエージェントと共有することで，他のエージェントは仮想的な学習行為を行なう．つまり，人間で言うところのイメージトレーニングや訓練にあたる行為である．エージェントは，シナリオを通じて体験学習を行なう．シナリオを利用する利点は次のような点である．

- ある意図を持って動作する主体の行動は，状況は無限であってもその行動ルールは無限ではない．
- 状況の履歴をシナリオに圧縮することで，必要な記憶容量が少なくなる．

このシナリオは，上位エージェントを通じて下位エージェントに提供されるが，シナリオ自身は下位エージェントの不完全知覚によって情報が欠落している場合があるので上位エージェントはそれを補完して提供する．

3.4.3 学習の流れ

図 3.6 は，組織学習モデルおける学習の流れを示している．また，図 3.7 は下位エージェントに属する個体が行なう学習のアルゴリズム，そして図 3.8 では上位エージェントが行なう学習のアルゴリズムを示している．個体は基本的に自身に必要な個体学習を行なっている．学習を行ない報酬を得たときはそのとき用いた学習データを上位エージェントへ報告する．そして，上位エージェントは自身の環境状態の分析と合わせて学習データを分析し，必要があれば下位エージェントへ学習データの提供を行なう．

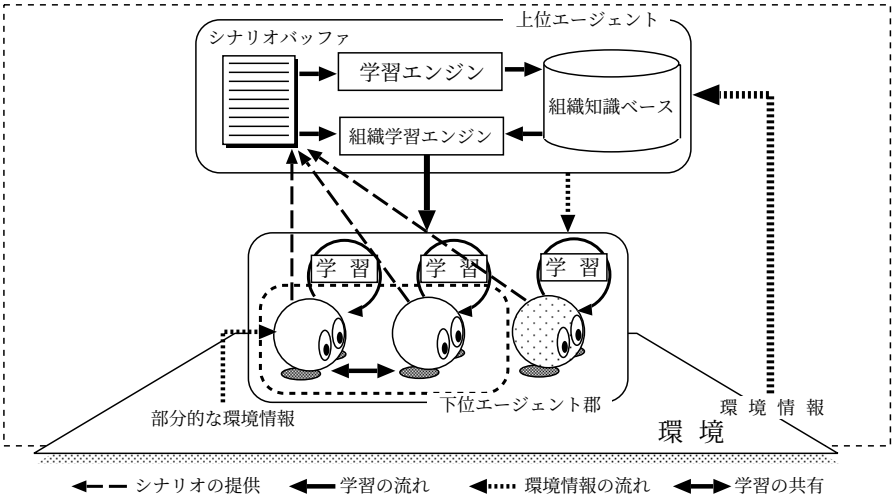

図 3.6: 組織学習モデル

下位エージェントは，上位エージェントから提供されたデータをもとに学習を行なう．この時，上位エージェントからの情報に信頼度を付加することで，現在持っている知識への影響度を調整することが可能である．これは，システム的に現実に自分で行なった学習行為より上位エージェントを通して提供される情報の方が明らかに多いためである．組織全体の学習は，上位エージェントから与えられた学習データを消化すると，再び近くに戻りこの流れを繰り返す．

1. 下位エージェントは環境の状態 s を知覚する.
2. 知覚した状態 s をもとに図 3.4 にしたがって行動の選択と学習を行なう.
3. 行動実行後，報酬を獲得したときはその学習で利用した学習データを上位エージェントへ自分のエージェントタイプと合わせて報告する．このとき，初めて上位エージェントと通信するとき自身の登録も行なう.
4. 上位エージェントから学習データを受けたとき，そのデータをもとに学習を行なう.
5. 手順 1 に戻る

図 3.7: 下位エージェントの学習の流れ

1. 上位エージェントは環境の状態 s を知覚する.
2. 知覚した情報 s をもとに環境の状態，状況の変化などを分析する
3. 個体からの学習の報告がある場合には，シナリオバッファに記録.
4. 現在の分析をもとにエージェントタイプの標準モデルに対して学習を実行.
5. 下位エージェントへ報告可能であれば，状況に応じてシナリオバッファに記録されている学習データもしくは各エージェントタイプの標準モデルを通信する
6. 手順１に戻る

図 3.8: 上位エージェントの学習の流れ

1. 下位エージェントは環境の状態 s を知覚する.
2. 下位エージェントの内部処理 (図 3.7 参照)
3. その個体が学習を行なったとき，そのときの学習に用いたシナリオ $scinario$ と選択した行動 a の対を上位エージェントに報告.
4. 受け取った報告はいったんシナリオバッファに蓄積.
5. 上位エージェントの内部処理 (図 3.8 参照)
6. 環境の相転移の確認

   if　環境の相転移　**then**

   　下位エージェントにタイプ毎の行動選択テーブルを与える

   else

   　学習データの配布.
7. 下位エージェントは上位エージェントからのデータをもとに学習または，行動選択テーブルの修正を行なう.
8. 手順１に戻る

図 3.9: 組織学習の流れ

第4章
サッカーシミュレーションゲームへの適応

本章では，3.4 節にて述べたマルチエージェントの組織学習モデルをもとにサッカーシミュレーションゲームへの適応について述べる．

4.1 RoboCup:マルチエージェントサッカーゲーム

4.1.1 標準問題としてマルチエージェントサッカーゲーム

現在，マルチエージェント研究のためのランドマークには次の要素が必要であると考えられている．

- 抽象化の程度が低い．全く抽象化しない問題を設定するのは非現実的であるが，実世界により近い問題であることが望ましい．
- 対象とする問題領域が広い．問題を解決のためには，様々な手法を統合する必要のある問題でなければならない．
- これから必要とされる技術的ニーズを内包できる問題でなければならない．
- 多くの研究者やほかの人にも理解しやすい問題が良い．また，その評価も容易でなければならない．

そこで，これらの条件を満たすことのできる人工知能と知能ロボット分野での研究のためのランドマークとして提案されたのが RoboCup で用いられているマルチエージェントサッカーゲーム [Noda98] である．

4.1.2 Soccer Game Server

サッカーサーバは，基本的にはフィールド上での選手の動き，ボールの動きなどなどのシミュレートをおこなう．また，審判として試合の進行を制御する．プレイヤはこのサッカーサーバのクライアントプログラムとして実装され，1 プレイヤにつき 1 クライアントが必要とされ各クライアントはサッカーサーバを通しての通信のみ許可されている．プレイヤの視覚情報，聴覚情報は一定時間おきにサッカーサーバからクライアントプログラムに送られ，クライアントプログラムはその情報を利用しサーバ上でのシミュレーションに必要な行動を決定する．また，クライアントプレイヤは，通常のフィールドプレイヤとボールのキャッチが許可されるゴールキーパの二種類設定可能である．ゴールキーパのボールをキャッチ出来る能力以外クライアントとなるプレイヤの能力は同一に設定されておりヘテロジーニアスなプレイヤを設定することは許可されていない．現在[1]の設定ではフィールドプレイヤのほかにコーチクライアントが用意されている．以下，フィールドプレイヤとコーチクライアントの特徴を示す．

フィールドプレイヤ

- 環境情報
  - [視覚情報] 視界方向に存在するオブジェクトの距離，方向情報などがあたられる．フィールド全体は把握不可能となっておりクライアントプレイヤは不完全な視覚情報をもとにフィールドの状況を解析する．
  - [聴覚情報] フィールド上の音声情報．審判からの情報なども音声情報として伝えられる．
  - [身体情報] プレイヤ自身のスタミナなど，身体に関わる情報を獲得できる情報
- 行動選択
  - [フィールドプレイヤ共通] フィールド上のアクションとして dash, turn, kick, move[2], 他のプレイヤなどとの通信手段として say が用意されている．
  - [ゴールキーパ専用] catch

コーチクライアント

- 環境情報
  - [視覚情報] Goal, Ball, Player オブジェクトにより構成されたフィールド全体の情報．
  - [聴覚情報] フィールドプレイヤと同じ

[1] soccer server 6.07 現在

[2] フィールド上の特定の位置に 1 サイクルで移動可能．ただし，他の行動とは異なり試合中には許可されない

- 行動選択 say を用いた情報伝達のみ．ただし，試合進行中には許可されない．

フィールドプレイヤは，上記にあるような１シミュレーションサイクルに１つの発火のみ許可された dash, turn, kick などのコマンドを組み合わせることにより様々な振る舞いを実現する．マルチエージェントサッカーゲームについての詳しいことは，[Noda98] などを参考にして欲しい．

4.2 SoccerTeam:Japanner

本研究で作成した Soccer Team： Japanner はオブジェクト指向言語である C++ を用いて開発した．Team:Japanner のチーム設計コンセプトを以下に示す．

1. 各フィールドプレイヤは自身の行動最適化のための学習を行なう
2. どんな対戦相手でも互角に戦える用にする．
3. 試合の状況によってフォーメイションの変更を可能とする

1) はこれまでの RoboCup 参加チーム同様にプレイヤの学習行動を追加することである．この理由としては，固定的な戦術のみのクライアントでは対戦チームによっては，手詰まりの状況が生まれやすく新しい局面を作り出すなどクライアントに多様性を持たせるには，経験的に学習が必要であるとされているためである．また，2) はチームの戦略的相性などから勝ち負けがある程度決ってしまったりする場合がある．そのために本研究では対戦相手に対してチームが持つ力を最大限引き出すためのシステムを設計することを目標とした．そして，3) は 2) を実現するには必要不可欠なものである．

SoccerTeam： Japanner のプログラムは次のような構成となっている．

フィールドプレイヤ

- [libsoccer] フィールドプレイヤの基本部分．入出力センサ機構，センサ情報解析機構，環境情報記憶機構，基本動作決定機構 により構成．
- [SpatialTimer] 時空間情報記憶機構：基本部分で一時記憶された情報を時系列にそって蓄積・解析することで一時記憶では不完全な情報を補完する機能と時空間情報から計算される状態空間を動作決定機構に提供するための機構
- [Japanner] クライアントプレイヤの行動最適化のための学習機構．

コーチクライアント

- [libcoach] コーチクライアントとしての基本部分．入出力センサ機構，環境情報記憶機構，フィールド解析機構 により構成.
- [Japanner] クライアントが行なった学習結果の記憶・解析・分類などを行ないクライアントプレイヤの学習補助を行なう機構

各クライアントは，基本構成部分の上にそれぞれ，SpatialTimer[Shinoda00] としての時空間分析補助を行なう機能と，Japanner としての組織学習機能を機能を追加してある．本論文では，主に Japanner としての組織学習機能を主に取り扱うことにする．

SoccerTeam:Japanner ではフィールドプレイヤとコーチクライアントの関係は次のような構成になっている．

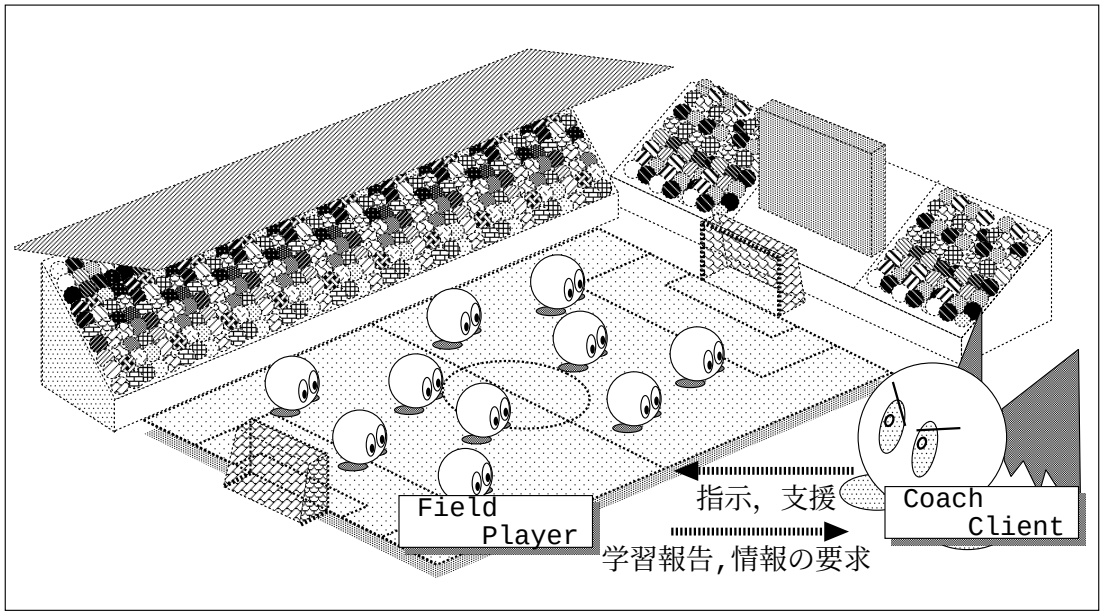

図 4.1: Team:Japanner の FieldPlayer と CoachClient の関係の概念図

図 4.1 で示すように，サッカーフィールド上でゲームに直接的関与するのはフィールドプレイヤである．しかしながら，フィールドプレイヤは不完全知覚であり不完全情報環境下での意思決定，学習を行なわなければならない．一方，コーチクライアントはゲームに直接的関与はできない．しかしながら，フィールド全体の情報を獲得可能であり，その情報を利用して試合の状況分析や対戦相手の分析などを行ないチームに対して情報を

提供することが可能[3]である．Team : Japanner では，FieldPlayer から見て CoachClient は支配的な立場にあるエージェントと言うよりは，環境の分析や行動最適化，学習行動などの支援を行なってくれる助言者，もしくはチームの自律的知識ベースと言う立場にある．

以下，Player Client と Coach Client の構成とそれぞれの状態決定，行動決定の仕様などについて述べる．

4.2.1 Player Client

PlayerClient は フィールド上で振る舞う FieldPlayer のことである．PlayerClient の行なう振る舞いは主に次の３つである．

- Positionning (フィールドの位置どり)
- Kick Ball (Pass, Shoot など)
- Say (情報伝達)

この３つの中で，最後の情報伝達は他の二つとは質の異なる振る舞いである．それは，Say は他の二つの振る舞いと同時に成立するためである．また，Say は試合に直接的関与は出来ない．しかしながら，Say は場合によっては他の主体の意思決定に関与することも可能であるために PlayerClient にとっては重要な振る舞いの一つである．

Positionning の行動選択肢としては以下が用意されている．

Basic Position 任意の方法に従い位置どりをきめる．
Chase Ball ボールを追い掛ける
Mark Player 特定のプレイヤを監視可能な位置どりをする

図 4.2: FieldPlayer の Positionning 時の行動選択肢

Basic Position をきめる方法としては，ポジション固定方式と，ポジション不定方式がまず考えられる．固定方式では，オフサイドラインによる影響のみを計算しながら位置を決定する方式で，各プレイヤの位置どりはあまり代ることなくチームの位置どり

[3]soccer-server 6.06 現在では試合が中断してるときのみ可能

の重心はあまり変ることがない．それに対して，ポジション不定方式では，チームの重心は試合の状況により大きく変化する．それは位置どりの決定にオフサイドライン以外に，ボールの位置や近隣のプレイヤの位置なども考慮に入れるためである．よって Basic Position の式は 4.1 式のような形で一般化可能である．

$$BasicPosition = Formation_{player} + Offsideline * a + Ball_Pos * b + TargetPlayer * c \quad (4.1)$$

それぞれの要素は xy 座標を表わすベクトルである．上の式は基本位置はフォーメイションできめられた座標とオフサイド，ボールと近隣のプレイヤの影響をどのように与えるかで決まる．

次に KickBall の行動選択肢は以下のようになっている．

Shoot Goal に向かってシュートを打つ (比較的ゴールから近距離)
PowerShoot Shoot を基本としたゴール前の動作 (比較的ゴールから遠距離)
Pass 味方が現時点でいる位置への Kick
ThroughPass 敵方のオフサイドラインの後方への Pass
Centering ゴールエリア内へのパス
Dribble ドリブル
ClearBall ボールを前方もしくはサイドへフィードする．

図 4.3: FieldPlayer の KickBall 時の行動選択肢

上の図から，FieldPlayer の KickBall の行動選択は 4 種類分けることが出来る．それは，タスクの完了を意図した Shoot とタスクの委任を意図した Pass そしてタスクの実行を意図した Dribble，そしてタスクの放棄を意図した ClearBall である．サッカーゲームではこのタスクの実行権はすべてボールの所有権と同一である．つまり，ボールの所有権を確保した状態で如何にタスクを完了することが可能であるかを FieldPlayer は学習していくことになる．学習に関しては後ほど述べることにする．

また，FieldPlayer の行動決定のアルゴリズムは図 4.4 に示す通りである．

```
procedure      FieldPlayer の 行動決定アルゴリズム
  time = 0
begin
   do
      環境の状態 s をもとに行動決定に必要な情報処理
      time = s.time
      if  ボール情報を把握していない  then
          look_for_ball
      else
         if  ボールを蹴ることが可能である  then
             set_kick_target(time)                  // 図 A.3 参照
         else
             set_move_target(time)                  // 図 A.4 参照
   while  ゲームが継続中である
end.
```

図 4.4: FieldPlayer の KickBall 時の行動選択肢

4.2.2 Coach Client

CoachClient は，ゲームに直接的関与は出来ない．そのため，FieldPlayer の補助的役割が主な役割である[4]．その CoachClient の役割には以下に示す通りである．

1. ゲームの環境 (対戦チーム) の分析
2. 味方チームの環境情報獲得の補助やフォーメイションの修正など
3. 味方チームの学習の補助

本節では，1,2 について説明する．3 については別の節にて触れることにする．

1. の対戦相手の分析は主に次の点で分析する．

- ボールの支配率
- パスの数，成功率
- シュートの数，成功率
- スルーパスの数，成功率
- オフサイドの数

[4]ただし，soccer-server7.00 以降では FieldPlayer のヘテロジーニアス化にともない CoachClient の機能も強化されたことで活用の幅が広がった．soccer-server7.00 については後ほど触れることにする

- インターセプト (敵チームのパスのカット) の数

ボールの支配率は，単純に試合進行中にボールに近いポジションにプレイヤがいるチームが支配していることにしている．またパスはボールの移動先の近くに同一チームプレイヤがいる場合，スルーパスは敵方のオフサイドラインを越える位置へのパスと単純化してある．これらを分析することでチームの特徴が明らかになるとされている [高橋 99].

Team:Japanner ではこれらをもとにして，敵チームの攻撃パターンなどの分析を行なった.

また，CoachClient は FieldPlayer が Field 全体を近くできないことを受けて FieldPlayer の情報補完の補助を目的として Field 情報を FieldPlayer に渡している．これに関して詳しくは，[Shinoda00] に記述してある.

4.3 Team : Japanner の学習システム

Team : Japanner では次の 2 種類のタイプの学習を行なう複合学習である.

1. 行動の最適化のための学習
2. 行動選択の最適化のための学習

この二つの学習の違いは，前者が自己最適化のための学習であるのに対して後者はチームとしての利益を最適化するための学習である．そのため，前者は従来から行なわれてきた強化学習を 3.1.3 節で述べたアルゴリズムを用いて行なうが，後者は本論文で導入したマルチエージェントの組織学習を利用する．本節では，Team : Japanner で行なわれるチーム全体での学習の枠組みを述べたあと，FieldPlayer，CoachClient それぞれの行なっている学習について述べる.

4.3.1 Team 学習

Team : Japanner では，チーム全体の学習としてフォーメーションの学習と個体が行なう強化学習の補助を行なっている．フォーメーションの学習は主に，試合を行なっていくなかで学習する Online 学習を行なっている．これらは，FieldPlayer では環境全体

を見渡すことが難しために，CoachClient が行なっている．また，現時点では得に細かい分析の上で学習をしていない．フォーメーションの学習では次のことを行なっているだけである．

1. 敵チームのフォーメーション分析
2. 得失点によるフォーメイションの相性.
3. 味方プレイヤの学習の習熟度によるポジションバランス

相手のポジションをあらかじめ用意したフォーメーションのタイプに分けて，それに対しての相性などの情報で味方チームのポジションを決定する．また，味方プレイヤの学習の習熟度を調べるのは，学習が進むポジションは重要なポジションんであると言えるのでフォーメーションを決定するのに重要な要素と言える．また，このフォーメーションの学習では，フィールドをメッシュ状に区切った環境情報を用いて分析を簡略化している．

4.3.2 PlayerClient の学習

PlayerClient が行なう学習は次の 2 つである．

- Basic Position の学習
- KickBall の選択

BasicPosition の学習では，主に味方からパスを受けたときと，敵のパスをインターセプトしたときとした．それぞれの学習に使った情報は以下の通りである．

Pass Receive

パスを出した味方の距離，一番近い味方からの距離，一番近い敵からの距離，x 座標の絶対値[5]

Pass Intercept

パスを出した敵の相対距離，一番近い味方の距離，ボールの受け手の距離，x 座標の絶対値

[5]ゴール方向を x 軸，サイド方向を y 軸とみなしたとき

BallKick に関しては，4.2.1 節の図 4.4 にある選択肢に関して，シナリオ行動対を利用した強化学習を行なう．このとき，シナリオはフィールドをメッシュ状に区切ることで表現したベクトルの集合で示される．

4.3.3 CoachClient の学習

CoachClient では，おもに FieldPlayer の学習の補助を行なっている．CoachClient は FieldPlayer から学習の報告をうけ，それの学習データを各 FieldPlayer に提供する．しかしながら，その学習データに使われたシナリオは不完全なものである場合がほとんどであるから，そのシナリオの欠落した情報を補完して他のプレイヤに提供する．

また，フォーメーションごとに様々なタイプのエージェントが存在する (通常でも，FW, MF, DF, GK) が，それぞれのエージェントが環境が変る毎に再学習しなくても良いように，エージェントタイプ毎にエージェントからある報告をもとに標準モデルを作成する．そして，新しい環境に直面したとき各プレイヤにその標準モデルを提供することで，同じ学習をすることを減らす試みをしている．

第5章
実験・評価：サッカーゲームシミュレーション

5.1 実験環境

前節で述べたサッカープレイヤクライアントをオブジェクト指向言語である C++ を用いて実装した．また，サッカーゲームシミュレータとしては，RoboCup のシミュレーションリーグで採用されている SoccerServer[1] を利用した．なお，シミュレータの実験環境は下記の機種を複数台使用して行なった．

Machine SUN Ultra5 270
CPU UltraSPARC-IIi 270MHz
Memory 128MB

以下，実験に利用したチームとそれぞれのチームの相性を調べ，それらを用いて行なった 2 種類の実験について述べる．

5.1.1 実験対象チーム

まず，実験の対象として従来から行なわれる学習を行なわないチームと，個々のエージェントのみが学習するチーム，そして本研究の組織学習を行なうチームの 3 タイプのチームを用意した．また，学習を行なわないチームいくつかの戦略パターンが考えられるので複数のチームを用意した．

[1] soccer-server-6.07 を利用

5.1.1.1 戦略固定チーム

学習を行なわないチームは以下の 5 種類のチームを用意する．なお，このチームの特徴はあくまでも主観的なものである．また，表 5.1 では，それぞれのチーム同士の対戦を各カード 20 試合ずつ行ないそれぞれのチームの相性を示す.

- 基準チーム (Team A)

  フォーメーション：4(DF)-4(MF)-2(FW)

  基本となるチーム．各プレイヤは [Shinoda00] の RoboCup2000 で使用したチームのプレイヤをもとにプログラムした．サッカーシミュレータでは多くのチームがこのフォーメーションを採用していた.

- 攻撃的チーム A (Team B)

  フォーメーション：3-5-2

  バックラインを 3 人にすることで，中盤を重視したチーム構成．攻撃と守備の切り替えが比較的スムーズであるが中盤のプレイヤの攻撃への参加

- 攻撃的チーム B (Team C)

  フォーメーション：3-4-3

  FW を 3 人にしたことで，攻めるポイントを増やしたチーム構成．敵チームの両サイドを攻撃ポイントとして使える.

- 守備的チーム A (Team D)

  フォーメーション：4-5-1

  中盤でも，特に守備に重心をおいたチーム．通常のサッカーではカウンター攻撃などを中心に利用されることがあるようだが，フィールドが空間ではなく面であるサッカーサーバではあまり使われない.

- 守備的チーム B (Team E)

  フォーメーション：5-3-2

  完全に守備に重心をおいたチーム，広い領域で守備が出来るがその反面攻撃につなげるための中盤が弱くなる．これもサッカーシミュレータでは見ないフォーメーションである.

表 5.1 から分かることは，基本チームである Team A は Team B, C と結果がにており比較的攻撃的なチームであることが分かる．また，Team A はゲームのなかで中盤を支配的な立場に立てる状態になるゲーム (Team C, Team E の対戦など) では強いが，その状態にならないときには攻撃にうつる機会が乏しく勝てないなど対戦相手によって特徴がでた．また，そのほかのチームにおいてもそれぞれ対戦相手による相性がでている.

表 5.1: Team A - E のチーム比較 (各組み合わせ 20 試合)

	Team A	Team B	Team C	Team D	Team E
Team A		35% 22 45%	50% 24 50%	25% 9 48%	35% 9 58%
Team B	60% 31 52%		45% 31 48%	55% 26 52%	15% 11 56%
Team C	40% 24 44%	40% 31 47%		40% 16 43%	20% 13 48%
Team D	20% 8 49%	20% 13 45%	35% 7 52%		20% 8 47%
Team E	20% 6 40%	20% 7 40%	35% 12 46%	20% 8 48%	

上段:勝率 中段:得点 下段:ボール支配率

5.1.1.2 戦略学習チーム：個体学習のみ (Team L_a)

この個体学習のみでの戦略学習チーム (Team L_a) では，個体のみで学習を行ない，情報の共有及び知識の共有などは個体間のネゴシエーションによってのみ行なう．なお, このチームは Team A のフォーメーションを持ったチームを初期状態として学習を行なった．Team B - E に対しての学習の連続した学習の結果は，図 5.1 に示す通りである．

この図は，Team L_a が Team B から Team E までの 4 チームを同一のチームとのゲームを連続 20 試合の繰り返し学習を全部で 20 回行ない，それぞれの試合経過での平均得失点誤差をグラフにしたものである．

図 5.1 のグラフからから，個体学習でも複数の試合を重ねることで徐々に対戦相手に適応したチームになっていくことが分かる．しかしながら，およそ 10 試合ほどの事前学習が必要であり，トーナメントのような 1 試合しか行なえないような状況では不十分であると言える．

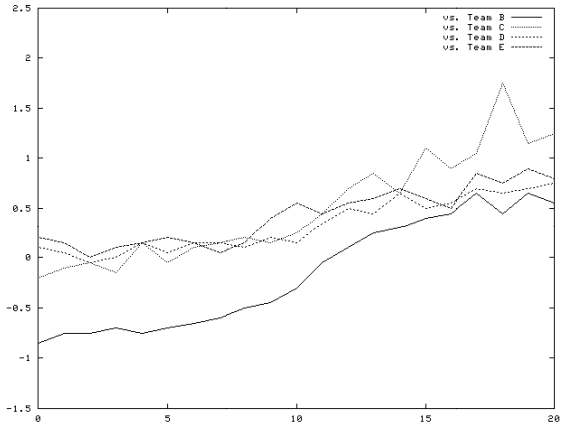

図 5.1: Team L_a と同一チームとの連続試合の得失点の変化

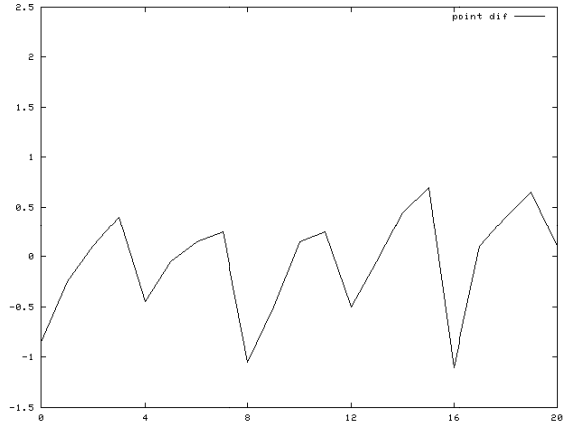

図 5.2: Team L_a と異なるチームとの連続試合の得失点の変化

また，図 5.2 では，Team B-E の４つのチームを順番に対戦させたゲーム連続 20 試合を 20 回行ないそこでのゲームの得失点の平均をグラフで示したものである．このグラフから，得失点差が，図 5.1 での最低点差よりも開く場合があり，学習として効果があるとは言えない．これは，前のチームでの学習が次の試合に影響を残すために，学習の忘却と再学習に時間が必要であるためだといえる．

5.1.1.3 戦略学習チーム：コーチクライアントの併用 (Team L_b)

本論文で実験の対象となるチーム．このチームは，Team L_a での個体学習に合わせ，コーチクライアントを併用することで組織学習を導入している．このチームによって次の 2 種類の実験を行なう．

1. 同一チームとの繰り返し学習
2. 複数チームとの繰り返し学習

実験１，２は前節の個体学習のみの Team L_a でも同様の実験を行なっている．それぞれの実験の持つ意味とは，実験 1 では単一の環境への適応速度を調べる．そして，実験 2 では変化する環境への適応速度を調べる．これらを Team L_a の実験例と比べることで本研究の効果があった部分を明らかにする．

なお，このチームでは得点をいれることを目標として学習を行なうものとする．また，CoachClient の学習に関しては実験１，２共に初期状態からの学習を行なうものとする．ただし，フォーメーションの基礎知識は Team A - Team E の比較試合のデータを参考に初期値を決定した．

5.2 同一チームとの繰り返し学習

同一チームの繰り返し学習では，図 5.3 のような結果となった．この図と図 5.1 と比べると，明らかに Team Lb のほうが良い結果をだしているのが分かる．このような結果が生じる理由としては次のような要素が関係していると考えられる．

1. コーチエージェントによる学習の共有化
2. コーチエージェントによりポジションの修正

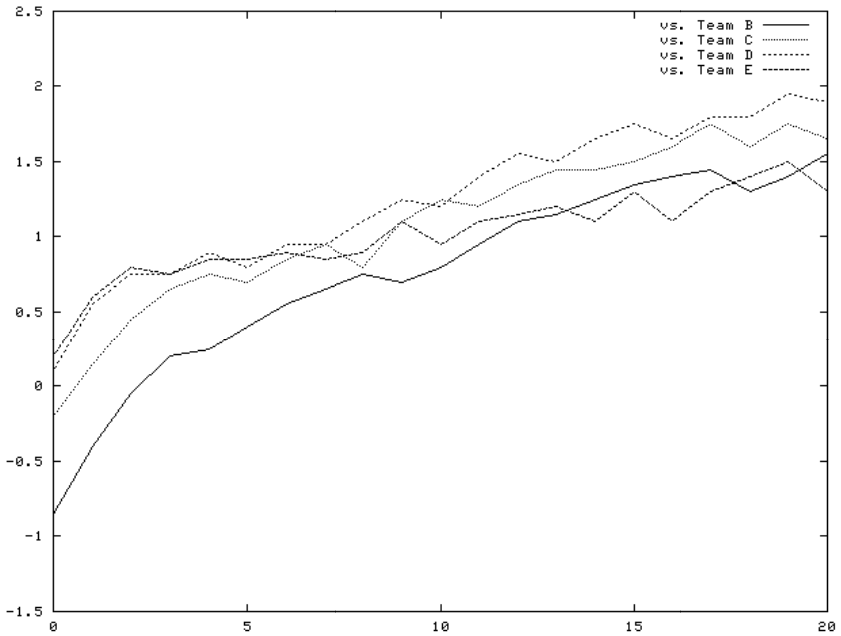

図 5.3: Team L_b と同一チームとの連続試合の得失点の変化

つまり，フィールドプレイヤは単独で学習を行なうよりも多くの学習の機会をコーチエージェントによって得られるために，学習の収束が早まったと思われる．

5.3 複数チームとの対戦学習

次に，複数チームによる繰り返し学習 (動的な環境変化への適応学習) では，図 5.4 になった．これと図 5.2 とを比較すると，Team L_b の方が Team L_a に比べ得失点差にばらつきが少なく，徐々に得失点差が良くなっている点から良い結果を出していると言える．この結果が導かれた理由としては以下の点が考えられる．

1. Team L_a で発生していた，過去の学習による影響と再学習がコーチクライアントの組織知識ベースとしての役割が有効であった
2. Team のタイプを分けることで環境の変化を認識しやすかった

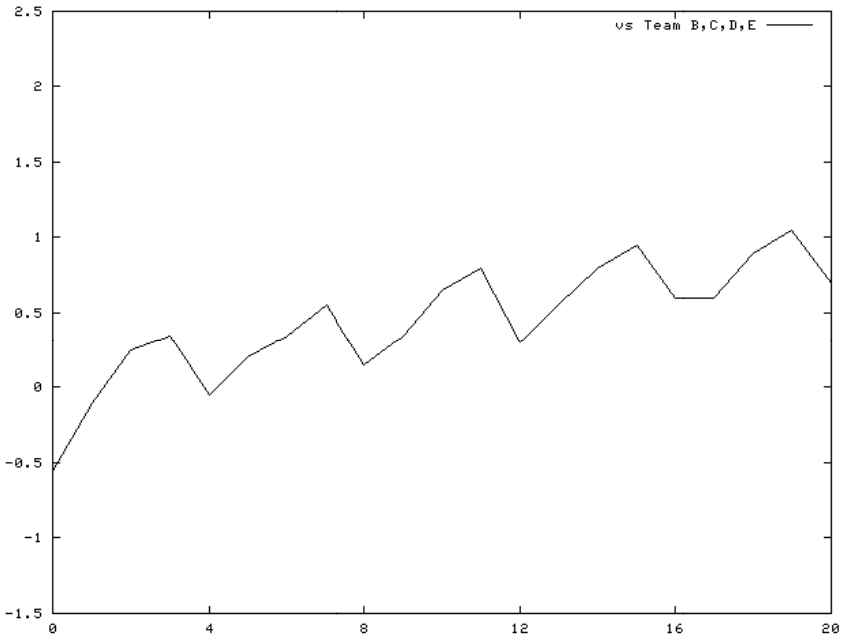

図 5.4: Team L_b と異なるチームとの連続試合の得失点の変化

以上から，組織学習を用いて環境の状態を様々なシチュエーションに分けることで動的な環境に適応しやすくなるといえる．

第6章
社会基盤システムへの適応

これまで，マルチエージェントにおける組織学習を適応することによる環境への適応について述べてきたが，本章ではそれらの現実的なシステムへの適応について述べる．その対象としては，RoboCup Rescue[Rescue00] での大規模災害シミュレーションと ITS [ITS01] での自動車運行補助システムについて述べる．

6.1 RoboCup Rescue

RoboCup-Rescue シミュレーションは，計算機上での災害救助シミュレーションを通じて新しい防災・救助システムの積極的な発見，または人間の活動をサポートするシステム構築するための試みである [Rescue99]．このシミュレーション上では，Ambulance[1], Police, FireBrigade[2], Civilian 等様々なタイプのエージェントが存在する．これら複数のエージェントの分散協調により新しい防災モデルを模索するのが RoboCup Rescue の大きな目標である．

このなかで，Ambulance, Police, FireBrigade がメインとなって防災活動を行ないこれらはそれぞれの持つ司令部の指揮下のもとで活動する形態を持っている．そのため，本研究で提案した学習モデルは適応しやすい環境にある．しかしながら，以下の点が問題である．

- Ambulance, Police, FireBrigade などそれぞれのエージェントの役割が強く限定されているため，組織の構成に柔軟さを欠く．

[1]救助隊
[2]消防隊

- 各司令部が現実の縦割り行政を反映しているために組織同士の連携行動は困難な状態になっている．
- 震災でも消火・救命活動という側面に限定されているためにエージェントが対応すべき環境が限定されている．

そのため，現時点では組織での学習まで必要とされていないのが現実である．しかしながら，今後発展が見込まれている分野であり，多くのタイプのエージェントが行動することが要求されるシミュレーションであるために，組織が注目されるものと思われる．現在は，災害活動から除外されている Civilian(一般市民) が，活動に加わることが予定されているためにこれらを対象とした組織行動に注目している．

6.2 ITS - 走行支援システム

現在，Intelligence Transport System(ITS) の一貫として自動車に自律的な判断システムを搭載することで運転者の支援活動を行なうためのシステムとして走行支援システム [ITS01] が研究されている．これは，基地局と複数の自動車を連携して安全を高めるための運用されるシステムであるため本研究で対象としたエージェントの組織モデルと類似している．また，自動車の運転も天候や道路状況などにより大きく変化することから適応できるのではないかと考えている．

現時点では，実際のシステムの適応は難しいく今後の課題と言える状況であるが，シミュレーションをつくることで実験は可能である．

6.3 その他のシミュレーション

それ以外には，今回実験に用いたサッカーサーバの新しいバージョンが提示され，Field-Player が多様化され，複数のタイプの能力を持ったエージェントが用意されることになった．これによりフォーメーションや CoachClient の重要度が上がったと言える．そのため，新しいサッカーサーバでの評価も必要だといえる．

第7章
まとめと今後の課題

7.1　まとめ

本論文では，動的環境へのマルチエージェントシステムの適応を実現するための組織学習モデルを提案し，サッカーエージェントチームを実装した．そして，サッカーシミュレーション上での実験により，本研究で提案した学習モデルの有効性について検証を行なった．

本論文では，従来から行なわれてきたエージェント学習が学習主体である個体の行動選択の向上が主たる目的であることの問題点として，学習の忘却による環境の変化に対する遅延と同一環境の再学習などがあった．これらの問題点を解決し変化する環境に対して安定しかつ確実に適応するための学習モデルとして組織学習をマルチエージェントシステムに適応する手法について述べた．この提案した手法が目標としたことは次の3点である．

1. 学習体験の共有化による，短時間での収束
2. 状況の変化に応じた動的組織形成
3. 環境の変化に応じた的確な組織の相転移

この目標を達成するために，本研究では強化学習の過程にてシナリオと行動の対を学習データとして利用する部分観測環境下での強化学習の手法を提案し，組織学習にてこの学習データをエージェント間で共有するための上位エージェントを持つ学習モデルを提案した．また，上位エージェントでは環境を分析することでそれぞれの環境に合わせたエージェントの標準モデルをつくることにより環境の変化への適応を目指した．これ

らの評価はサッカーチームとして実装することで試合を行ない学習モデルの有効性を確認した.

7.2 今後の課題

今後の課題としては，次のような点がある.

1. エージェントの部分観測環境下での強化学習の手法の有効性の検証が不十分である.
2. 本研究で提案した学習モデルを具体的に適応するためには環境の分析手法を確立する必要がある.
3. 他の具体的なシステムへの適応を示す必要がある.

これらを解決することが今後の課題である.

また，サッカーシミュレーションに関してもつい先ほど[1]新しいサッカーサーバがリリースされた．この新しいサッカーサーバの特徴は次の点にある.

- FieldPlayer に制限はあるがヘテロジーニアスプレイヤが採用された
- FieldPlayer の変更に伴い，選手交替など CoachClient の機能が強化された.
- Stamin, Kick_Power_Rate などの変更により FieldPlayer のプレイスタイルが変更する可能性が示された

これらの変更は，本研究で作成したチームにも大きな影響を与える事柄である．そのため大会への参加には新たなチーム作成を必要とする.

[1] 2001 年 1 月 28 日

謝辞

本研究を進めるにあたり，多くのご指導，ご鞭撻を賜りました國藤進教授，ならびに藤波努助教授に深く感謝致します.

また，様々なご協力や日頃有意義な討論をさせて頂いた創造性開発システム論講座のみなさんに深く感謝致します.

付 録A
アルゴリズム

1. 環境の観測 X_t を受け取る.
2. $\pi(a_t, W, X_t)$ の確率で行動 a_t を実行する.
3. 環境から報酬 r_t を受け取る.
4. 内部変数 W の全ての要素 w_i について以下の $e_i(t)$ と $D_i(t)$ を求める.

$$e_i(t) = \frac{\partial}{\partial w_i} \ln\{\pi(a_t, W, X_t)\}$$

$$D_i(t) = e_i(t) + \gamma D_i(t-1)$$

   ここで，γ は割引率 $(0 \leq \gamma < 1)$ である

5. 以下の式を用いて $\Delta w_i(t)$ を求める.

$$\Delta w_i(t) = (r_t - b) D_i(t)$$

   ただし，b は報酬基底と呼ばれる定数である.

6. 以下の式で W を更新することにより，政策の改善を行う.

$$\Delta W(t) = (\Delta w_1(t), \Delta w_2(t), ..., \Delta w_i(t), ...)$$

$$\Delta W \leftarrow W + \alpha(1-\gamma)\Delta W(t)$$

   ここで，α は学習定数 $(0 < \alpha)$ を表す.

7. クロック t を $t+1$ に進めて，手順 1 へ.

図 A.1: 確率的傾斜法の一般形

```
procedure　合理的政策形成 (Rational Policy Making)
begin
  do
    1 次および 2 次記憶領域の内容を初期化する.

    do
      if　現在の感覚入力の 2 次記憶上に行動が記憶されている　then
        その行動を出力する.
      else
        その行動を 1 次記憶上に上書きする.

      if　報酬を得た　then
        1 次記憶領域の内容を 2 次記憶領域に複写する.
    while　2 次記憶領域が未収束

    if　合理的政策が得られている　then
      2 次記憶領域の内容を保存する.
  while
end.
```

図 A.2: 合理的政策形成アルゴリズム

```
procedure      set_kick_target
begin
   KickBall の行動リスト内の評価式を環境状態に照らし合わせて評価 (シミュレート)
   if   評価のなかで 1 次水準の閾値を越えた    then
        行動を実行
   else
        評価の中で 2 次水準の閾値を越えたものを選択
      if   選択したリストが空でないとき    then
           任意の選択ルールで行動選択, 実行
      else
           ClearBall を実行
end.
```

図 A.3: set_kick_target

```
procedure      set_move_target
begin
   if   ボールに一番近いところにいる    then
      chase_ball
   elsif   マークする選手がいる    then
        mark_player
   else
      if   BasicPosition の付近にいない then
           BasicPosiiotn へ移動
      else
           look_at_ball
end.
```

図 A.4: set_move_target

付 録B
試合結果

表 B.1: Team A vs. Team B

Team A kicks off.			Team B kicks off.		
A	B	win	A	B	win
1	0	A	0	1	B
2	1	A	2	1	A
0	2	B	1	3	B
1	2	B	2	3	B
2	3	B	2	3	B
0	2	B	0	1	B
1	0	A	2	0	A
0	3	B	1	2	B
2	1	A	0	1	B
1	1	-	2	1	A

Winning ratio of A : 0.35
Winning ratio of B : 0.60

表 B.2: Team A vs. Team C

Team A kicks off.			Team C kicks off.		
A	C	win	A	C	win
2	1	A	0	1	C
1	0	A	1	0	A
1	2	C	1	3	C
2	3	C	2	2	-
3	2	A	0	1	C
0	1	C	1	0	A
0	2	C	2	1	A
1	0	A	1	2	C
2	2	-	2	1	A
1	0	A	1	0	A

Winning ratio of A : 0.50
Winning ratio of C : 0.40

表 B.3: Team A vs. Team D

Team A kicks off.			Team D kicks off.		
A	D	win	A	D	win
1	1	-	0	0	-
1	0	A	0	1	D
0	0	-	2	1	A
0	0	-	0	0	-
0	1	D	0	1	D
0	0	-	1	1	-
1	0	A	1	0	A
0	0	-	0	1	D
1	1	-	0	0	-
0	0	-	1	0	A

Winning ratio of A : 0.25

Winning ratio of D : 0.20

表 B.4: Team A vs. Team E

Team A kicks off.			Team E kicks off.		
A	E	win	A	E	win
1	0	A	0	0	-
1	0	A	0	1	E
0	0	-	1	0	A
0	0	-	0	0	-
1	0	A	1	1	-
1	0	A	0	0	-
0	0	-	0	1	E
1	1	-	1	0	A
0	1	E	1	0	A
0	0	-	0	1	E

Winning ratio of A : 0.35

Winning ratio of E : 0.20

表 B.5: Team B vs. Team C

Team B kicks off.			Team C kicks off.		
B	C	win	B	C	win
1	2	C	2	1	B
0	0	-	0	3	C
1	0	B	1	2	C
3	2	B	3	2	B
2	1	B	2	3	C
1	1	-	1	2	C
2	3	C	3	2	B
1	1	-	2	1	B
1	2	C	3	1	B
2	0	B	0	2	C

Winning ratio of B : 0.45

Winning ratio of C : 0.40

表 B.6: Team B vs. Team D

Team B kicks off.			Team D kicks off.		
B	D	win	B	D	win
1	0	B	3	1	B
2	0	B	0	1	D
1	1	-	3	1	B
2	0	B	1	1	-
0	1	D	2	1	B
2	1	B	1	1	-
2	1	B	2	0	B
1	2	D	0	0	-
0	0	-	2	0	B
1	0	B	0	1	D

Winning ratio of B : 0.55

Winning ratio of D : 0.20

表 B.7: Team B vs. Team E

Team B kicks off.			Team E kicks off.		
B	E	win	B	E	win
1	0	B	0	0	-
1	1	-	0	0	-
2	0	B	0	1	E
0	0	-	2	1	B
0	1	E	0	0	-
1	1	-	0	0	-
3	0	B	0	0	-
0	0	-	0	1	E
0	0	-	0	1	E
1	0	B	0	0	-

Winning ratio of B : 0.25

Winning ratio of E : 0.20

表 B.8: Team C vs. Team D

Team C kicks off.			Team D kicks off.		
C	D	win	C	D	win
1	0	C	1	2	D
2	0	C	0	0	D
0	0	-	1	1	-
1	0	C	3	1	C
0	0	-	1	0	C
2	0	C	0	0	-
1	1	-	1	0	C
0	1	D	0	0	-
0	0	-	1	1	-
1	0	C	0	0	-

Winning ratio of C : 0.40

Winning ratio of D : 0.15

表 B.9: Team C vs. Team E

Team C kicks off.			Team E kicks off.		
C	E	win	C	E	win
0	1	E	0	0	-
1	1	-	0	1	E
3	0	C	0	1	E
0	0	-	3	1	C
0	0	-	0	0	-
0	1	E	1	1	-
0	0	-	0	1	E
0	1	E	2	1	C
1	1	-	0	1	E
0	0	-	2	0	C

Winning ratio of C : 0.20

Winning ratio of E : 0.35

表 B.10: Team D vs. Team E

Team D kicks off.			Team E kicks off.		
D	E	win	D	E	win
0	0	-	0	0	-
0	0	-	0	2	E
1	1	-	0	0	-
0	0	-	0	1	E
1	0	D	1	1	-
2	0	D	0	0	-
0	0	-	0	1	E
0	1	E	1	1	-
0	0	-	0	0	-
1	0	D	1	0	D

Winning ratio of D : 0.20

Winning ratio of E : 0.20

参考文献

[Schon78] C. Argyris and D. A. Schon. *Organization Larning*. Addison-Wesley, 1978.

[ITS01] ITS. *http:://www.its.go.jp/ITS/j-html/whatITS/*. http://www.mlit.go.jp/, 2001.

[Noda98] Ituki Noda, David Andre, Emiel Corten, Klaus Dorer, Pascal Gugenberger, Marius Joldos, Johan Kummeneje, Ituki Noda Paul Arthur Navratail, Patrick Reley, Peter Stone, Tomoichi Takahashi, Tralvex Teap. Soccerserver manual ver.4 rev00. Technical report, tmp, November 1998.

[Kaelbling96] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement Learning: A Survey. *journal of Artificial Intelligence Research*, Vol. 4, pp. 237–277, 1996.

[Shinoda00] Kosuke Shinoda, Susumu KuniFuji. Team description of spatialtimer. *RoboCup2000 Book(To be appearance))*, 2000.

[McCallum95] R. Andrew McCallum. Instance-Based Utile Distinctions for Reinforcement Learning with Hidden State. In *Proceedings of the 12th International Conference on Machine Learning*, pp. 387–395, 1995.

[Misky86] M.Minsky. *The Society of Mind*. Simon & Schuster, 1986.

[Rescue99] rescue. *http://robomec.cs.kobe-u.ac.jp/robocup-rescue/*. http://www.robocup.org/, 1999.

[Rescue00] RoboCup-Rescue 技術委員会, 田所諭. ロボカップレスキュー 大規模災害救助への挑戦. 共立出版, 2000.

[ITS01] スマートシステム. *http:://www.its.go.jp/ITS/j-html/*. http://www.mlit.go.jp/, 2001.

[宮崎 97] 宮崎和光, 小林重信. 離散マルコフ決定過程下での学習. 人工知能学会誌, Vol. 12, No. 6, pp. 811–821, 1997.

[宮崎 99a] 宮崎和光, 荒井幸代, 小林重信. POMDPs 環境下での決定的政策の学習. 人工知能学会誌, Vol. 14, No. 1, pp. 148–155, 1999.

[宮崎 99b] 宮崎和光, 荒井幸代, 小林重信. Profit Sharing を用いたマルチエージェント強化学習における報酬配分の理論的考察. 人工知能学会誌, Vol. 14, No. 6, pp. 1156–1164, 1999.

[荒井 98] 荒井幸代, 宮崎和光, 小林重信. マルチエージェントと強化学習の方法論 –Q-Learning と Profit Sharing による接近–. 人工知能学会誌, Vol. 13, No. 4, pp. 609–618, 1998.

[高橋 99] 高橋友一, 成瀬正, 二間瀬真司. サッカーシミュレーションゲームにおけるチームの評価. 第 4 回 ＡＩチャレンジ研究会, pp. 21–26, 1999.

[佐藤 72] 佐藤慶平. 現代組織の論理と行動. 御茶の水書房, 1972.

[山倉 93] 山倉健嗣. 組織間関係. 有斐閣, 1993.

[生天目 98] 生天目章. エージェントと複雑系. 森北出版, 11 1998.

[木村 96] 木村元, 山村雅幸, 小林重信. 部分観測マルコフ決定過程下での強化学習:確率的傾斜法による接近. 人工知能学会誌, Vol. 11, No. 5, pp. 761–768, 1996.

[木村 97] 木村元, Leslie Pack Kaelbling. 部分観測マルコフ決定過程下での強化学習. 人工知能学会誌, Vol. 12, No. 6, pp. 822–830, 1997.

[木村 99a] 木村元, 小林重信. ロボットアームのほふく行動の強化学習:確率的傾斜法による接近. 人工知能学会, Vol. 14, No. 1, pp. 122–130, 1999.

[木村 99b] 木村元, 宮崎和光, 小林重信. 強化学習システムの設計指針. 計測と制御, Vol. 38, No. 10, pp. 618–623, 1999.

本研究に関する発表論文

[1] SMC'99 IEEE Vol.VI pp.722–727 1999 "Action Decision with Incompleate Information : A Temporal Database Approach" N. Itom, K. Shinoda, K. Nakagawa, N. Ishii.

[2] RoboCup Conference IV 2000 Team Description "Team Description of Spatial-Timer" K. Shinoda, S. Kunifuji.

[3] 第 9 回 インテリジェント・システム・シンポジウム 進化的計算論 (ECOmp) 研究会 日本ファジー学会「階層的組織構造における戦略決定と組織学習に関する研究」篠田 孝祐，國藤 進 1999 年 10 月

[4] 第 40 回 基礎論研究会 人工知能学会 「不完全情報環境下における時系列情報の生成」 篠田孝祐，伊藤 暢浩，國藤 進 1999 年 3 月

[5] SIG-Challenge 研究会 人工知能学会 「マルチエージェント環境下における組織的学習効果に関する一考察」篠田 孝祐，國藤 進 2000 年 8 月

[6] SI2000 計測自動制御学会 「RoboCup-Rescue における Civilian Agent モデルの一考察」篠田 孝祐，國藤 進 2000 年 12 月