静的・動的環境における
通信なしマルチエージェント強化学習の
協調行動を導く合理的目的設定

上野　史

電気通信大学 大学院情報理工学研究科
博士 (工学) の学位申請論文

2020 年 3 月

静的・動的環境における
通信なしマルチエージェント強化学習の
協調行動を導く合理的目的設定

論文審査委員会

主査　高玉　圭樹　　教授
委員　西野　哲朗　　教授
委員　大須賀　昭彦　教授
委員　庄野　逸　　教授
委員　佐藤　寛之　准教授

Abstract

This thesis proposes new multi-agent reinforcement learning (MARL) that can derive cooperative behaviors without communication among agents and extends it from the static environment version to the dynamic environment version to tackle a communication delay and an uncertain information of surrounding environments which often occur in real world problems, and investigates the effectiveness of the proposed MARL through the maze problem as the first step towards real world problems. For this purpose, this thesis explores the mechanism that provides the internal rewards to the agents instead of the ordinary rewards to promote them to select the rational goals for cooperation without communication, and integrates it to propose the following MARL: (1) Profit minimizing reinforcement learning (PMRL), which is based on non-communication in the static environment; (2) Profit minimizing reinforcement learning with oblivion of memory (PMRL-OM) and (3) Profit minimizing reinforcement learning for dynamic rewards and environmental states (PMRL-DRES), both of which are based on non-communication in the dynamic environment. In detail, PMRL-OM and PMRL-DRES are designed for the spatial environment change where the maze size/shape change or the start/goal location change and the agent-goal change where the number of the agents/goals changes, respectively. Through the intensive experiments of the proposed MARL, Profit Sharing and Q-learning as the major conventional methods, the following implications have been revealed: (1) agents with the conventional methods cannot acquire the cooperative behaviors while PMRL agents can reach their own goals with the mostly minimum steps by solving a conflict with agents in the goal of the static environment; (2) Profit Sharing and PMRL agents require many trials to acquire the cooperative behaviors, while PMRL-OM agents can quickly acquire them by appropriately changing their own goals after the spatial environment change, and (3) PMRL-OM and Profit Sharing agents are hard to adapt to the environmental change in some cases while the PMRL-DRES agents succeed to select their appropriate goal by limiting the space after the agent-goal change.

概要

本論文では，現実問題に起こる通信遅延や情報の不確かさに対処するために，複数のエージェント間の協調行動を通信なしに導く強化学習手法を提案するとともに，変化のない静的環境に加えて不測の事態などで変化する動的環境に対応できるように拡張し，それらの有効性の検証と合理性を示す．その実現に向け，本論文では，(1) 各エージェントが自身の目的達成のために行動すると競合が起こる可能性があるため，全てのエージェントが他のエージェントに対する譲歩行動の学習を通して，他のエージェントと競合しない目的を見出し，その目的を達成することで情報通信なしに協調行動獲得を実現する手法 (Profit minimizing reinforcement learning: PMRL) を提案する．このとき，各エージェントは協調に必要な目的達成に向け，通常の報酬ではなく新たに内部報酬を設定し，逐次的にエージェントの行動を決定するアプローチをとる．次に，(2) 環境変化に伴って達成すべき目的が変化する可能性があるため，常に最新の情報を維持することで適切な目的に切り替え，その目的変化にあわせて内部報酬を設定する手法 (Profit minimizing reinforcement learning with oblivion of memory: PMRL-OM) を提案する．また，環境が変化しても，全てのエージェントが達成すべき目的を変更する必要はなく，必要なエージェントのみ変更すればよいため，環境変化に応じて学習する範囲を適切に制限することで，その範囲内で目的を変更する提案手法 (Profit minimizing reinforcement learning for dynamic rewards and environmental states: PMRL-DRES) を提案する．実験の結果次の知見が得られた．まず，(1) 静的環境において，Q 学習や Profit Sharing では協調行動を獲得できないのに対し，PMRL ではエージェントは適切な目的決定で競合を解消でき，ほぼ最短ステップでゴールに到達することに成功した．次に，(2-1) 環境形状の動的変化において，Profit　Sharing は学習できず，PMRL は環境変化前に決定した目的変更のために環境変化前以上の学習回数が必要だが，PMRL-OM は環境変化後の最新の情報を活用することで即座に目的を変化させ，ほぼ最短ステップのゴールに成功した．最後に，(2-2) エージェント・ゴール数の動的変化において，Profit Sharing，PMRL-OM と PMRL-DRES を比較したところ，Profit Sharing や PMRL-OM では環境変化への適応に限界があるが，PMRL-DRES では環境変化後に学習領域を分割することで，エージェントは適切な目的を選択でき，ほぼ最短ステップでのゴール到達に成功した．

目次

図目次

表目次

第 1 章

序論

1.1　マルチエージェントシステムとその制御

マルチエージェントシステム（Multi-Agent System: MAS）は内部状態，意思決定，通信機能を備えた複数の主体（エージェント）によるシステムであり，社会現象を複数のエージェントと環境でシミュレートすることでそこに潜む問題の解決を目指す．図 1.1 はマルチエージェントシステムにより応用問題をモデル化して解決する例を示している．図のように都市交通の渋滞解消や倉庫ロボットの最適制御を実現するために，行動主体である自動車やロボットをエージェントに置き換え，お互いに協調して衝突回避するように，または効率的に物資を運ぶようにスケジューリングすることでそれぞれの問題解決や最適制御を実現する．しかしながら，マルチエージェントシステムにおいて，各エージェントの振舞いや協調行動はシステムの設計者が決めるため，より複雑なモデルを考えたときに適切な解決策を導けないという問題がある．

マルチエージェントシステムの従来モデルとして大きく 2 種類が存在し，図 1.2 はその 2 種類を示している．1 つは問題を解く上で情報を 1 つの PC に集約してその中ですべてをシミュレートし，その情報を個々のエージェントに与えて制御することでマルチエージェントシステムを機能させる集中型マルチエージェントシステムであり，もう 1 つは個々のエージェントが各自目的を持ち，各々が観測する情報に基づきその目的達成を目指す分散型マルチエージェントシステムである．集中型マルチエージェントシステムは強力であるが，情報を集約する PC の目指す目的が達成不可能であることがあるため目的設定が難しく，現実の問題を取り扱う際のより複雑な状況を想定する際には利用することが難しい．そのため本研究では分散型マルチエージェントシステムに取り組む．

しかしながら分散型マルチエージェントシステムにも問題点は存在する．それは各エージェントが個々に目的を持つことで競合が起こり，エージェント全体の目的達成に障害が起こることである．なおこの問題はそもそも集中型マルチエージェントシステムには起こり得ない．なぜならエージェント全体の目的に従って各エージェントを動かすからであ

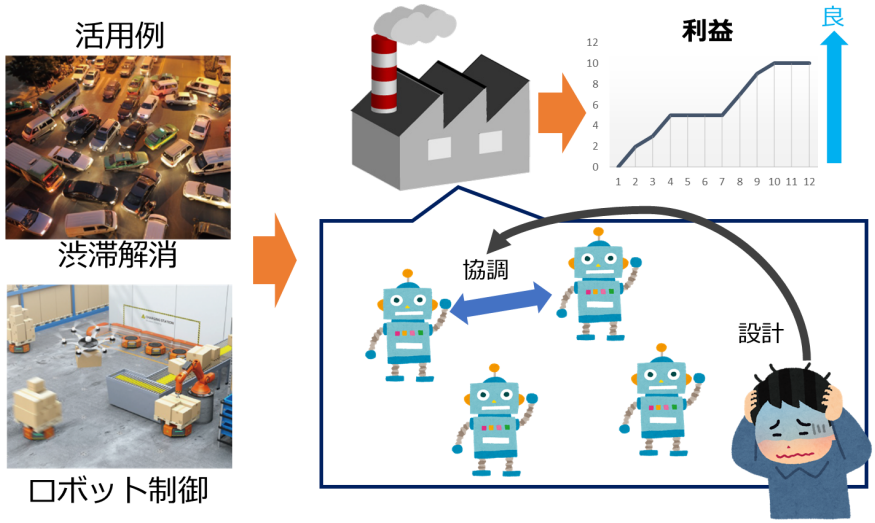


図 1.1　マルチエージェントシステム

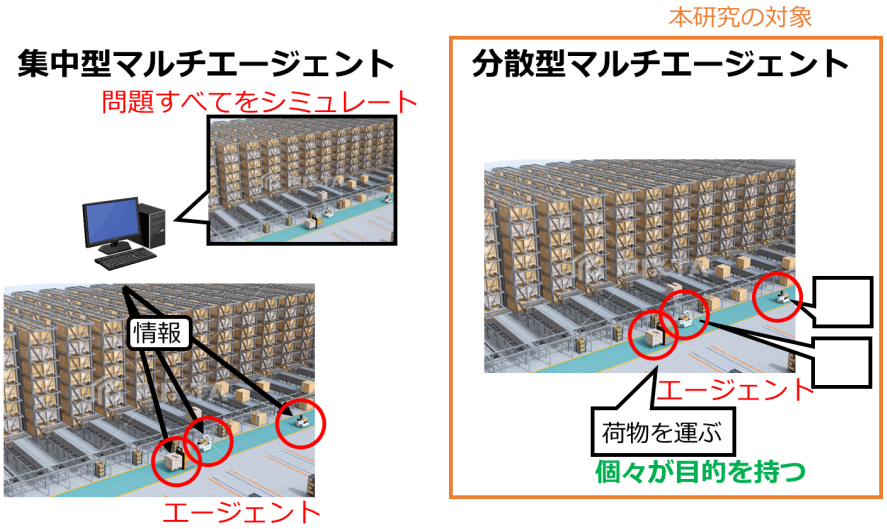


図 1.2　目的とするマルチエージェントシステム

る．図 1.3 は迷路問題におけるエージェントの個々の目的と全体の目的の違いを示している．迷路問題においては各エージェントは (特に最短で) ゴールに到達することが目的となるが，2 体のエージェントが存在する迷路問題においてはエージェント 2 体がそれぞれゴールへ到達し，そのステップ数を最短にすることが目的となり，その達成のために一部のエージェントは最短ではなくゴールへ到達することが求められる．以上が分散型マルチエージェントシステムにおける問題である．実際の問題では，例えば自動車をエージェントとした交通渋滞の解消問題を考えれば，エージェント自身は目的地へ到達することが目的であり，交通渋滞を解消することは目的とならないため，全体の目的とは異なるものとなる．

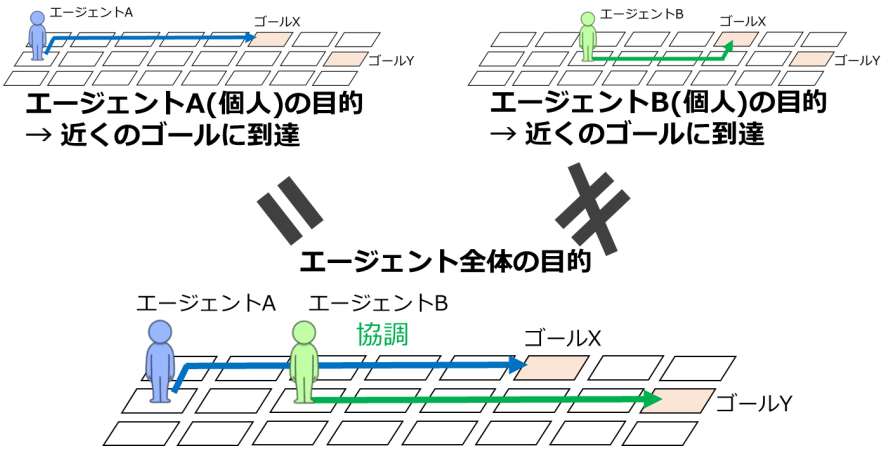


図 1.3　マルチエージェントシステムにおけるエージェント個々と全体の目的の違い

マルチエージェント強化学習（Multi-Agent Reinforcement Learning: MARL）は，その問題解決のため，各エージェントが強化学習により自身の意思決定則を制御し，環境や他のエージェントの振舞いに適応する．強化学習は，他の機械学習法と異なり，少ない手がかりで適切な意思決定を行え，さらに目的の結果（物流システムでは満たした需要量など）さえ分かれば，後はその結果に至るまでの各エージェントの振舞いの評価は各々の学習に任せ自動的に決定されるため，複雑なシステムを制御する上で都合がよい．

例えば図 1.4 は強化学習と教師あり学習を比較したときの違いを示している．図の下段矢印は学習の頻度を示している．左端によるほど 1 回でより多くのことが学習でき，右端によるほど各エージェントの行動を逐一評価する必要がある．つまり左側が強化学習を示し，右側が教師あり学習を示している．強化学習は例えばある時点 t における行動 a_t が評価されたとすると，その行動の評価を利用してそれ以前の行動 $a_{t-4}, a_{t-3}, a_{t-2}, a_{t-1}$ をも評価する．もちろんこれは a_t 以外の行動を直接評価したわけではないため正確ではないが，ある程度適切な評価を行える．その一方で，教師あり学習では行動 a_t の評価はその行動のみのものであるため，それ以前の行動 $a_{t-4}, a_{t-3}, a_{t-2}, a_{t-1}$ が評価されることはない．そのため，無数に存在するエージェント同士の相互作用を逐一評価する術がないマルチエージェントシステムにおいて，強化学習を導入することは必要不可欠であるといえる．近年のマルチエージェント強化学習では，ゲーム AI をはじめ，航路や宝石採掘のための経路プランニング，風力発電機やロボット制御からデータマイニングに到るまで幅広く活用されている [1, 2, 3]．そのとき，エージェント間の相互作用を制御して各エージェントが協調的振舞いを学習により獲得することが重要となる．

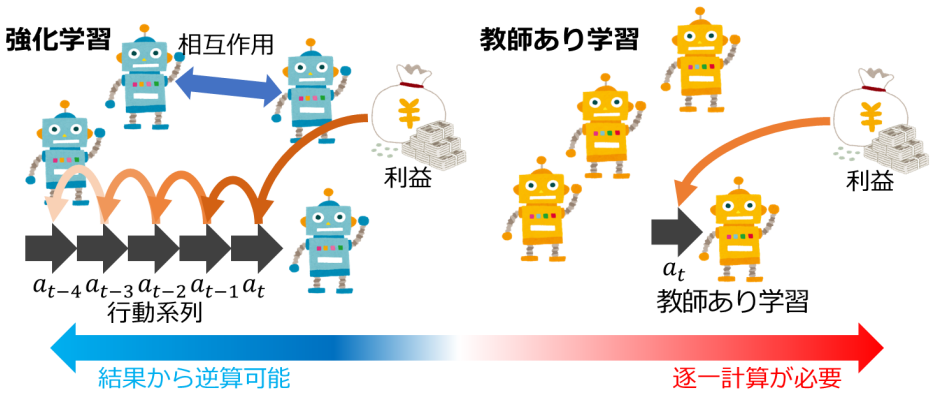


図 1.4　強化学習の特徴

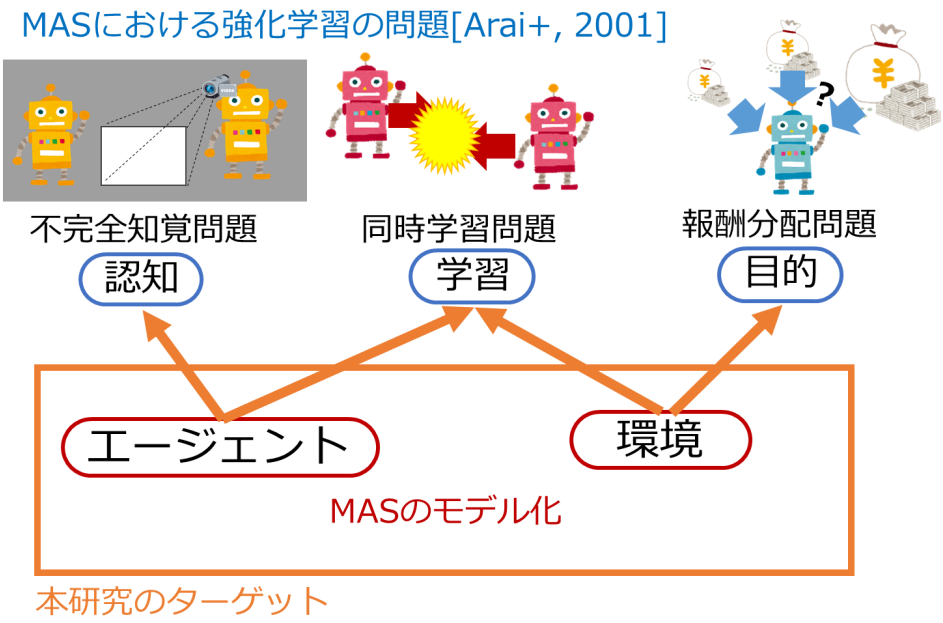


図 1.5　本研究の着目する点

1.1.1　本研究の着眼点

図 1.5 は，従来のマルチエージェント強化学習が取り組む問題と本研究が着目する点の違いである．荒井らによればマルチエージェント強化学習における従来の問題点は不完全知覚問題，同時学習問題，報酬分配問題の 3 つが存在する [4]．不完全知覚問題とは，他のエージェントの位置関係が異なるなど，本当は異なる状況であってもエージェントの知覚では判別がつかずに，それぞれの状況における適切な協調が特定できないという問題である．同時学習問題とは，同じ環境に複数エージェントが存在することにより，協調行動の学習が困難であるという問題であり，報酬分配問題は，協調行動を学習させるための報

酬設定が困難であるという問題である．以上のようにマルチエージェント強化学習では，図のようにエージェントの認知，学習，目的それぞれに対して問題が存在し，その解決を目指している．しかしながら，それを構成するモデル (環境とエージェント) は研究者が独自に設定していることが多く，本研究はその点に関して問題があると考えている．その理由はマルチエージェントにおける問題 3 点が，知覚する情報や環境の設定次第で難しさが大きく変わってくるのにもかかわらず，従来研究としては特に議論していないためである．特に，従来のマルチエージェント強化学習では動的環境における相互作用を学習することが非常に難しく，環境の動的変化に対して適切な行動を学習する方法が存在しない．しかし現実問題を想定すれば環境が静的であることの方が少なく，動的環境におけるマルチエージェント強化学習法を確立することには大きな意義がある．更に，従来手法は通信などにより他エージェントの位置や行動が獲得できなければ協調行動を学習することはできない．その上，ここで想定する情報は遅延なく常に正確な情報が得られるという前提の上で成り立っており，実際には利用することが難しい．特に動的環境を想定した場合，他エージェントの情報を多大に利用することは不可能であり，部分的に利用することを考えても，通信遅延や情報の不正確性を考えなくてはならない．また従来法でそのような想定をすると学習空間が膨大になり，学習が上手くできない可能性が大いにある．以上を鑑みても他エージェントの情報を利用しないマルチエージェント強化学習法を確立することは大きな意義がある．

それに加えて，従来のマルチエージェントシステムは数多くのエージェントがそれぞれの相互作用によりシステム全体の動きを形成するため，システムを適切に動作させるための各エージェントの動作を明らかにすることが難しく，各エージェントの動作の最適性を理論的に示すことは非常に難しいという問題点がある．特に近年はグラフ理論やゲーム理論を用いて，エージェント全体が学習した結果から最適な解を理論的に導くことが主に行われている．例えば，Honig らは物流システムにおける経路プランニング問題に取り組み，従来すべての経路を探索した上でその中から適切なものを選び出すことに対し，要求に合わせてその場で経路生成を行い，それに合わせてタスク割り当てを行うように CBS を拡張した CBS-TA を提案している [5]．ここで利用しているものはグラフ理論であり，全エージェントの学習した経路に基づき，その中から最適なものを選び出し，最適なものを必ず選び出せることを理論的に示している．以上のように，従来のマルチエージェント強化学習では，各エージェントの学習した行動に対してその合理性を理論的に示している例はない．しかしながら，より一般的なマルチエージェント強化学習法に適用する上で，各エージェントの学習した行動の合理性は必要不可欠なものであり，エージェント全体の振舞いではなく各エージェントの行動そのものに合理性を付加することは非常に画期的であり，探究する価値は大いにある．

1.2　研究目的

本研究は，現実問題に起こる通信遅延や情報の不確かさに対処するために，複数のエージェント間の協調行動を通信なしで導く強化学習手法を提案するとともに，変化のない静的環境に加えて不測の事態などで変化する動的環境に対応できるように拡張し，それらの有効性の検証を目的とする．その実現に向け，各エージェントの目的を合理的に設定し，逐次的にエージェントの行動を決定させることで協調行動を導く．さらに，環境変化にあわせて学習範囲を制限することで，動的環境時でも適切に目的を切り替えるように改良する．

本研究における目的達成のため，まず静的環境において通信を行わずに協調行動を学習する理論的マルチエージェント強化学習法を提案し，それを動的環境へ展開する．ここでいう協調行動は全てのエージェントが単位時間で獲得報酬を最大化することを指し，また合理的手法は学習による行動の価値が収束すること，そして収束値により協調行動を必ず取ることを示すことを指す．

1.3　本研究の意義

本研究は他エージェント情報を利用することなく動的環境に適応した協調行動を学習する手法を提案し，その合理性を示すことに意義がある．以下に各項目に対する他の従来法を述べ，動的環境を想定した，他エージェント情報を前提としない協調行動学習法が従来提案されておらず，そしてそれを合理的に示した手法も存在しないことの問題点を俯瞰的に示すとともに，本研究の意義を述べる．

1.3.1　他エージェント情報を利用しない協調行動学習法の意義

従来のマルチエージェント強化学習は通信や観測により獲得した他エージェント情報から，エージェント間の相互作用を設計し，協調行動を学習させる．Tan[6] はエージェントの通信を用いて協調する際にどのような情報が適切であるか明らかにしている．具体的に他エージェントの情報を知る方法は，情報通信と観測の 2 つが存在し，それぞれに対して制約を課すことで情報共有のモデルとする．以下それぞれの方法について参考文献を引用しながら説明する．

観測は主に複数ロボットを想定したマルチエージェント強化学習で利用される．通常マルチエージェント強化学習では状態というものが決まったうえで学習するが，ロボットの分野ではまずセンサ情報からどのように状態を分けるか (例えばカメラ画像の処理によりランドマークを発見する等 [7]) というところから始まる．そのため，ロボット分野にお

ける学習は通常シングルエージェントが多く，マルチロボットでは群ロボットかロボカップサッカーの分野がかねてから研究されている．特に Stone らはロボカップシミュレーションサッカーがマルチエージェント強化学習にとって多くの解決すべき課題を提示すると述べている [8]．東らの研究では群ロボットの上空をカメラで観察して状態観測し，シグナルで各ロボットにそれを知らせることで操作する [9]．これらの手法はロボットとして現実世界に適用しているが，これらの手法はエージェントにあまり複雑な動作や戦略を学習させるに至っていない．

また，ロボットに限らずに言えば，従来のゲーム理論 (確率ゲーム) の世界では全エージェントが共通の状態を持っており，そのときの状態共有を観測と呼んでいる [10, 11]．近年では深層強化学習の 1 手法である A3C[12] が登場し，観測情報の複雑さは増している [13]．例えば Raileanu らは各エージェントが他のエージェントの状態と行動から目的を推定し，それに基づき推定した自身の適切な目的を目指し，行動を学習するという手法（Self-other modeling: SOM）を提案しているが，このとき状態は環境のすべての情報を入力している [14]．これらの手法はより困難な協調 (例えば時系列に関係する協調行動) の学習を可能としているが，その適用範囲は大きくなく，ロボットや現実世界の媒体に導入するに至っていない．

以上のように，従来のマルチエージェント強化学習は実用方向とシミュレーション方向双方からの研究が進められおり，また近年は高機能なプロセッサの小型化や深層強化学習の登場によって，シミュレータ上でより複雑な情報に基づくマルチエージェント強化学習という手法が増えてきているのが実情であるが，現実世界に適用している例を見てみると，Web 上のシステムなどを除けば，その前提は超理想的なものであると分かる．以上を鑑みて今後の発展を考えれば，他エージェント情報を前提としない手法は今後重要な研究となると考えられる．

1.3.2 動的環境を想定した協調行動学習法の意義

マルチエージェント強化学習はエージェント間の相互作用の問題は取り上げるが，それ以外の動的要因を考慮していない．例えば，Tuyls らは他エージェントの振舞いに影響して自身の学習すべき協調行動が変化することを “Unstable Environment (非定常環境)” と呼び，マルチエージェント強化学習による解決を目指している [15]．しかしこの非定常環境は他エージェントとの協調行動のみを想定した環境であるため，本研究の位置づけでいえば静的環境ということになる．また実用研究では Marinescu らはスマートシティにおける電力供給問題へ適用し [16]，MacAlpine らはロボットによるサッカーに対して適用している [17]．そして Fang らは野生生物や漁業保護に活かすためにマルチエージェントを利用している [18]．しかしながら，これらはエージェントの行動以外の環境の変化やエージェント自身の故障といった動的変化を想定していない．これはつまりマルチエー

ジェントにおいて学習する環境が変動し，エージェントの初期状態や目的，そこに至るまでの状態が別の状態へずれること，そしてそもそも環境が変化して，エージェントや目的そのものが増えることの 2 つを示している．そのため，より現実的な問題へマルチエージェント強化学習を適用するときにうまく機能しない．これらの問題に対して，荒井らは動的変化する環境としてマルチエージェントの追跡問題を取り上げ，その上で機能する強化学習を示したが，それは各エージェントの学習する動作が最適であるかについては言及していない [19]．また，Zemzem らは環境変化に適応して協調行動を学習可能な手法を提案したが，それは環境変化に対する追従性に着目しており，動的変化に対して適切な行動を学習する方法に関して言及していない [20]．以上の通り，従来のマルチエージェント強化学習では動的環境における相互作用を学習することが非常に難しく，環境の動的変化に対して適切な行動を学習する方法が存在しない．しかし現実問題を想定すれば環境が静的であることの方が少なく，動的環境におけるマルチエージェント強化学習法を確立することには大きな意義がある．

1.3.3 合理性を持つマルチエージェント強化学習法の意義

従来，マルチエージェントシステムは数多くのエージェントがそれぞれの相互作用によりシステム全体の動きを形成するため，システムを適切に動作させるための各エージェントの動作を明らかにすることは難しい．そのため，各エージェントの動作の合理性を示すことは非常に難しい．

従来，特に理論的手法を目指すマルチエージェント強化学習は確率ゲームというゲーム理論の枠組みで語られることが多い [11]．それはいわば数体のエージェントの情報が完全に知れ渡っている前提で，各エージェントが合理的選択を行うためにはどうすればよいかということに着目していることを意味する．例えば Hu らは Nash Q-learning という手法を提案し，Nash 均衡 (どのエージェントも自身の戦略を変えることで高い利得が得られない状態) 戦略を学習する手法を提案している [21]．またその一方で，強化学習とは少々異なるが，グラフ理論によるマルチエージェントの制御は今もなお活発に研究されている分野である．Iwashita らの研究ではマルチエージェントシステムの都市セキュリティシステムへの適用のためにグラフ理論を応用している [22]．また Honig らは物流システムへの適用に向けてグラフ理論を用いた最適経路探索手法を提案している [5]．以上のように，マルチエージェントシステムの振舞いを理論的に示す上で，単体のエージェントにのみ適用する強化学習等ではエージェント全体の振舞いを制御できないので，グラフ理論やゲーム理論のように全体を制御できるものでなければ理論的手法を提案することは非常に難しいと分かる．

特に，近年深層強化学習に基づくマルチエージェント強化学習が多々研究されており，各エージェントが全ての可能性をいかに早く探索できるかということに力を入れ，その中

から協調行動を選択している．つまりマルチエージェント強化学習の分野として，今は強力な計算機資源で全ての可能性を計算するという方向に進んでいる．例えば Florensa らは Generative Adversarial Nets (GAN)[23] を利用し，エージェントがそれぞれ探索範囲を広めるように自身で報酬を設定する Goal GAN という手法を提案し，それをマルチエージェントシステムへ展開しようとしている [24]．これらの手法は強力だが，いまだ 1 体や 2 体のエージェントのみを扱っており，今後多数エージェントに適用していくことを考えれば，グラフ理論などのように簡単化した環境でも合理性を示すことは非常に重要であるといえる．また，従来のマルチエージェント強化学習では，学習の収束性を示したものは存在するが，各エージェントの学習した行動に対してその合理性を理論的に示している例はない．その意味において，エージェント全体の振舞いではなく各エージェントの行動そのものに合理性を付加することは非常に画期的であり，探究する価値は大いにある．

1.4 本論文の構成

本論文の構成は以下の通りである．

第 1 章では，本研究における背景・問題・目的・手段を簡潔に述べている．具体的には，マルチエージェント強化学習が社会にある様々な現象の中の活動主体をエージェントに置き換えてシミュレートし，各エージェントが協調して解決方策を学習させることで，その社会現象に含まれた問題を解決する手法であることを述べ，そこには従来手法がマルチエージェント強化学習を考える上でのモデル化をどのようにするかということを特段議論しておらず，問題解決のためにエージェント間の通信を自由に設定し，環境も静的であるという前提を置くことに問題点が存在することを述べた．そして本研究ではそれを解決し，現実問題に起こる通信遅延や情報の不確かさに対処するために，複数のエージェント間の協調行動を通信なしで導く強化学習手法を提案するとともに，変化のない静的環境に加えて不測の事態などで変化する動的環境に対応できるように拡張し，それらの有効性の検証を目的とすることを述べた．そして目的達成のため，各エージェントの目的を合理的に設定し，逐次的にエージェントの行動を決定させることで協調行動を導き，環境変化にあわせて学習範囲を制限することで，動的環境時でも適切にエージェントのゴールを切り替える手法を提案することを述べた．

第 2 章では，本研究のベースとなるマルチエージェント強化学習における 2 つの要素であるマルチエージェントシステムと強化学習について詳しく説明し，エージェントの構成から学習エージェントの構成，エージェントの相互作用とその効果についてそれぞれ説明する．そして強化学習として本研究で用いる Q 学習を紹介し，その性質である学習結果の収束についても説明する．その後マルチエージェント強化学習について概説して，他のエージェントの振舞いを知ることなく協調行動を学習することの難しさを具体的に説明する．そして，その解決策として Q 学習における学習結果の収束性に着目し，その収束値

を適切な目的に向けるための報酬設計が重要であることを述べる．

第 3 章では，マルチエージェント強化学習における基本的な問題設定とその性質について概説し，本研究における目的を達成するための適切な問題設定方法と実際の問題設定について述べる．その後実際に扱う迷路問題とその動的変化について説明し，本研究における到達目標を明らかにする．

第 4 章では，1 つ目の提案手法である静的環境における通信なしマルチエージェント強化学習法 (Profit minimizing reinforcement learning: PMRL) について説明する．具体的には，エージェントのメモリ内部の設計，メカニズムとアルゴリズムを説明し，2 体エージェントにおいて，内部報酬値が一定の値に収束し，その内部報酬値に基づいた学習結果が適切な振る舞いを呼び起こすように収束することを理論的に示す．

第 5 章では，動的変化に適応するため，まず PMRL の内部報酬の設計関数に知識の忘却関数を新たに組み込み，最新の情報を重視した内部報酬設計を行う提案手法 (Profit minimizing reinforcement learning with oblivion of memory: PMRL-OM) の具体的なメカニズムとアルゴリズムを説明する．そして，内部報酬設計関数を理論的に分析し，PMRL と内部報酬の収束値が等しいことを理論的に証明する．

第 6 章では，PMRL-OM の学習領域を分割して個別に協調行動を学習する提案手法 (Profit minimizing reinforcement learning for dynamic rewards and environmental states: PMRL-DRES) の具体的なメカニズムとアルゴリズム説明し，本論文で想定する静的・動的環境において有効性を検証する．

第 7 章では，本論文にて提案した手法 PMRL，PMRL-OM，PMRL-DRES の結果を統合し，本論文において主張しているアプローチの適用範囲とその効果について議論する．

最後に，第 8 章で本論文のまとめを述べ，今後の課題と展望を示す．

第 2 章

マルチエージェント強化学習

マルチエージェント強化学習は，各エージェントが強化学習により自身の行動を制御し，その集合体であるマルチエージェントシステム全体を機能させる手法である．本章では，まず強化学習をその代表的な手法と共に紹介し，その後マルチエージェント強化学習について一般的な説明をする．

2.1 基本的設定

2.1.1 環境の定式化

環境をモデル化する上で必要な要素として環境状態，状態遷移関数，報酬関数が存在する．以下の式はその要素を示す式である．式 (2.1) は環境状態を示す式であり，S は環境状態すべてを要素として持つ集合であり，$state_0 \cdots state_D$ はそれぞれの環境状態を示す (D は環境状態の要素数である)．式 (2.2) は各エージェントの行動 a_i の直積 A を示し，式 (2.3) の状態遷移関数に S, A を適用することにより，ある一定の確率で状態遷移する．式 (2.4) は報酬関数であり，エージェントの状態 s が目的の状態集合 S_{goal} の要素であった場合に報酬 $reward$ を獲得する．以上が環境の基本的設定であり，マルチエージェント強化学習では問題の持つ各要素を以下の変数で置き，解決を目指す．なおこのとき以下の式の内容は問題に合わせて変更することも可能である．

$$S := \{state_0, state_1, ..., state_D\} \tag{2.1}$$

$$A = a_1 \times a_2 \times ... \times a_n \tag{2.2}$$

$$\tau : S \times A \times S \to [0, 1] \tag{2.3}$$

$$r = \{reward | s \in S_{goal}\} \tag{2.4}$$

2.1.2　マルコフ決定過程

強化学習の前提として，現在の状態における最適な行動は前状態にのみ依存するというものがあり，この性質をマルコフ性 (Markov property) と呼ぶ．そして各状態において最適な行動を取り続ければ目的を達成できる，つまり状態が確率的に遷移してその状態遷移がマルコフ性を満たすものをマルコフ決定過程 (Markov decision process: MDP) という．ここで数式を用いてマルコフ性を説明する．式 (2.5) は通常の状態遷移確率を表している．$state(t), a(t)$ は時間ステップ t のときの状態と行動であり，$P(state(t+1))$ は時間ステップ $t+1$ の時の状態 $state(t+1)$ に遷移する確率を示している．そして一般的に式 (2.5) のように，確率 $P(state(t+1))$ は今までに観測した状態 $(state(0) \cdots state(t))$ とそのとき取った行動 $(a(0) \cdots a(t))$ 全てによって決まる．つまり，今までの状態と行動すべてが影響して今の状態と行動が決まるということである．

$$P(state(t+1)) = P(state(t+1)|state(t), a(t), \cdots, state(0), a(0)) \tag{2.5}$$

一方で式 (2.6) はマルコフ決定過程における確率 $P(state(t+1))$ を示している．この式から，$t+1$ の時の状態 $state(t+1)$ に遷移する確率は 1 ステップ前の状態 $state(t)$ とそのとき取った行動 $a(t)$ のみから決定されることは明らかである．以上がマルコフ性である．

$$P(state(t+1)) = P(state(t+1)|state(t), a(t)) \tag{2.6}$$

2.2　強化学習

強化学習は試行錯誤により環境に適応する学習法である [25]．それは心理学による強化に基づいており，例えば動物であれば餌を使って目的の行動をとるように躾けるように，エージェント上では餌として報酬という値を利用して，報酬をより多く獲得できる行動を取るように学習する．そして各強化学習法はその報酬値をエージェントのエフェクタの動作に割り振るかを定めた手法である．図 2.1 は強化学習を行うエージェント (以降強化学習エージェントと呼ぶ) の知的活動を示している．強化学習エージェントは変数として状態 s，報酬 r，行動 a を持つ．状態 s とはセンサによって観測した情報を基に場合分けした際の 1 状態を示す (例えば GPS を基にエリア分けした場合はそのエリアが状態を示す)．報酬 r はその状態において獲得した報酬を示し，行動 a はエフェクタの動作を示す．なお，行動 a の種類は予め決まっており，例えば移動ロボットであれば前後左右の 4 種類に分ける．エージェントはまず観測データに基づいて状態を知覚する (図の状態観測に対応)．そしてその状態が目的の状態であれば信号として報酬を獲得する (図の報酬獲得

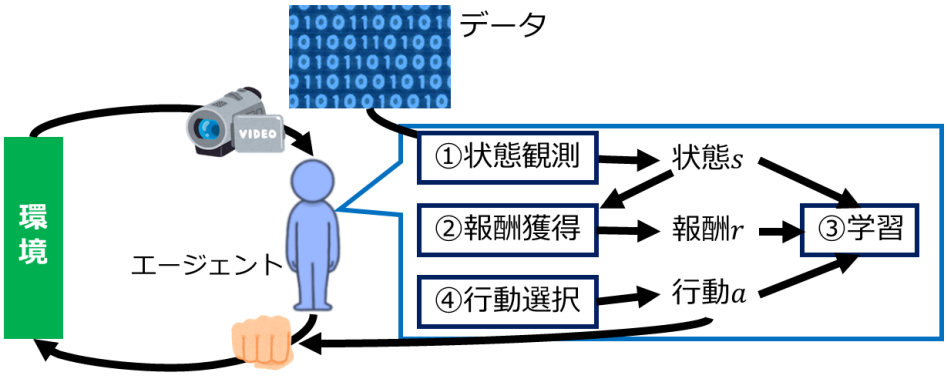


図 2.1　強化学習エージェントの知的活動

に対応)．その後状態 s と報酬 r と行動 a を基に強化学習により行動選択規則を学習する (図の学習に対応)．最後に学習した行動選択規則により新たな行動 a を選択し，その行動 a に従ってエフェクタを動作させる (図の行動選択に対応)．その後新たにセンサで環境情報を観測し，状態観測からプロセスを繰り返す．以上が強化学習エージェントの振舞いである．なお以降では強化学習の中でも本研究で用いる Q 学習について説明する．

2.2.1　行動選択方法

強化学習では価値の最も大きな行動をとるのみの戦略ではうまく学習できない．それは，例えばある状態においてとる行動が 4 つあり，それぞれに報酬値が存在したとき，最初に取った行動以外を取らなくなることである．そのため，強化学習では探索と利用を考えた行動選択をする必要がある．以下にその代表的行動選択法の説明をする．なお，行動選択法について明確な優劣を言及できないことに注意されたい．なお以降では学習した行動の価値を Q 値と呼ぶ (詳しくは Q 学習にて説明．)

- グリーディ選択法
  グリーディ選択法は必ず Q 値が最大となる行動を選択する方法であり，式 (2.7) に従って行動を選択する手法である．上記の表現でいえば探索と利用の利用のみを行う行動選択法である．

$$a = \arg\max_{b} Q(s, b) \tag{2.7}$$

- ϵ-グリーディ選択法
  ϵ-グリーディ選択法は，グリーディ選択法に探索を取り入れた行動選択法である．具体的には式 (2.8) に従って行動選択する．この式は行動選択則が 2 つ存在し，基本的にグリーディ選択法で行動選択するが，ある一定確率 ϵ でランダムに行動選択することを示す．この選択法はエージェントの行動の中で最適なものが一意に定

まっている (最大ではないが報酬が獲得可能な行動を探索する必要が大きくない) ときに有効であると考えられる．本研究ではその観点から ϵ-グリーディ選択法を採用する．

$$a = \begin{cases} \arg\max_b Q(s,b) & if \quad some\ probability\ 1-\epsilon \\ random\ action & otherwize \end{cases} \tag{2.8}$$

- ボルツマン行動選択法
  ボルツマン行動選択法はそれぞれの行動に対する Q 値に従って確率的に行動を選択する方法である．具体的には式 (2.9) により設定した値に従って確率的に行動を選択する．式 (2.9) において，$p(a|s)$ は状態 s において行動 a を選択する確率を表し，β は温度パラメータというハイパーパラメータである．また A は実行可能な全ての行動を持つ集合であり，ここでは β を掛けた Q 値の指数を取り，それをすべての行動に対して平均した値を確率としている．なお，ネイピア数の指数を取ることで，Q 値が 0 の時でも確率計算できるようにしており，β によって探索と利用のバランスが調節できるようになっている．具体的に β は大きな値であればあるほど，Q 値の大小関係が大きくなるため，Q 値が最大の行動を選択する確率が高まり，Q 値を利用した行動選択が進む．逆に β が小さい値であればあるほど，Q 値の大小関係が小さくなるため，ランダムな行動選択を行う探索が進む ($\beta = 0$ の時完全にランダム探索を行う)．この選択法は Q 値の大きさに従って確率的に行動選択を行うため，エージェントの取るべき行動が複数あり，それが一意に定まらない (報酬が獲得可能な行動全てを探索する必要がある) ときに有効であると考えられる．

$$p(a|s) = \frac{exp(\beta Q(s,a))}{\sum_{b \in A} exp(\beta Q(s,b))} \tag{2.9}$$

2.2.2 Q 値と方策

強化学習の学習結果として大きく分けて Q 値と方策 (政策) の 2 個が存在する．Q 値は行動の価値であるが方策とはその Q 値と行動選択法などから作られるそれぞれの状態に対する行動選択確率をマッピングしたものである．図 2.2 は 3 × 3 のグリッドワールドにおける状態と行動を示している．図のマスがそれぞれの状態を示しており，矢印が行動を示している．Q 値であればそれぞれの矢印の価値として存在しているが，方策とはそれぞれの矢印を選択する確率をまとめたものである．

強化学習における出力は学習結果である．それは，学習によって導かれた最適な方策 (それぞれの状態における最適な行動) である．つまり，マルチエージェント強化学習における出力はシミュレートする環境において各エージェントが強化学習をした時に導かれる最適方策である．そしてそのために各エージェントはシミュレートした結果得られる利得

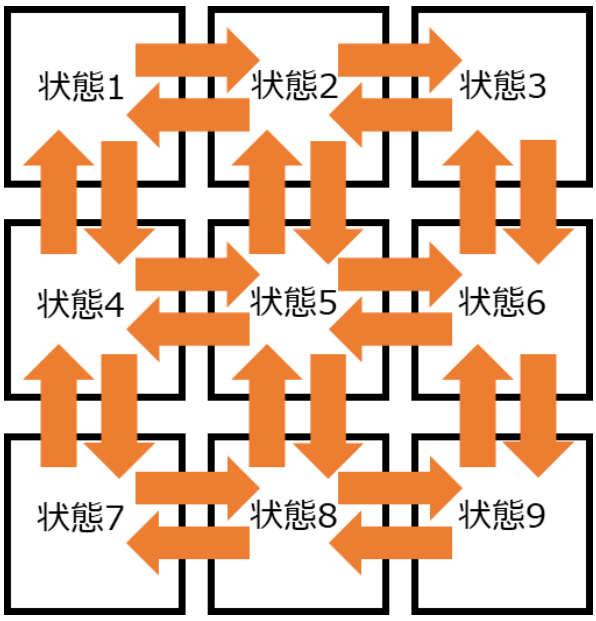


図 2.2　3 × 3 のグリッドワールドにおける状態行動

を手掛かりに学習を繰り返し，利得が最大となる行動を学習する．なお本研究において学習結果は Q 値である．そして例えば現実に倉庫ロボットにおける物流システムを想定すると，各ロボットに方策を導入して協調行動による物資運搬を実現することを目指す．

2.2.3　Q 学習

Q 学習 [26] は強化学習における代表的手法である．Q 学習は行動選択規則を制御するものとして状態行動価値 (Q 値) を持ち，報酬を基に Q 値を更新することで学習する．図 2.3 は，例として迷路問題を使用して，Q 学習のエージェントを示している．図において，右側がエージェント及び学習する迷路を示しており，吹き出しがエージェント内部を示している．迷路において，エージェントは自身の座標を状態として把握し，上下左右の行動をとる．そして Q 学習ではその状態と行動を紐づけて行動に対して Q 値を設定している (図においては状態 (2,1) の時の下の行動は $Q((2,1),下)$ という風に価値を設定する)．

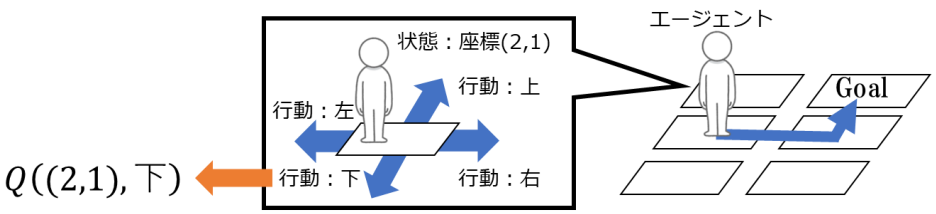


図 2.3　Q 学習エージェント

図 2.4 は Q 学習のアルゴリズムフローを示している．Q 学習は状態 s を観測し，行動 a をとり, 次状態 s' へ遷移するとともに報酬を獲得し，Q 値を更新するという流れを繰り返して学習する．このときの流れを本論文ではステップと呼ぶ．そして，特に今回取り上げる迷路問題のように明確な目的が 1 つに定まるのであれば，目的達成後に初期状態に戻って学習をやり直すというプロセスを実行する．つまり，目的達成時にまた学習をやり直すというサイクルを繰り返す．このサイクルを本論文ではエピソードと呼ぶ．

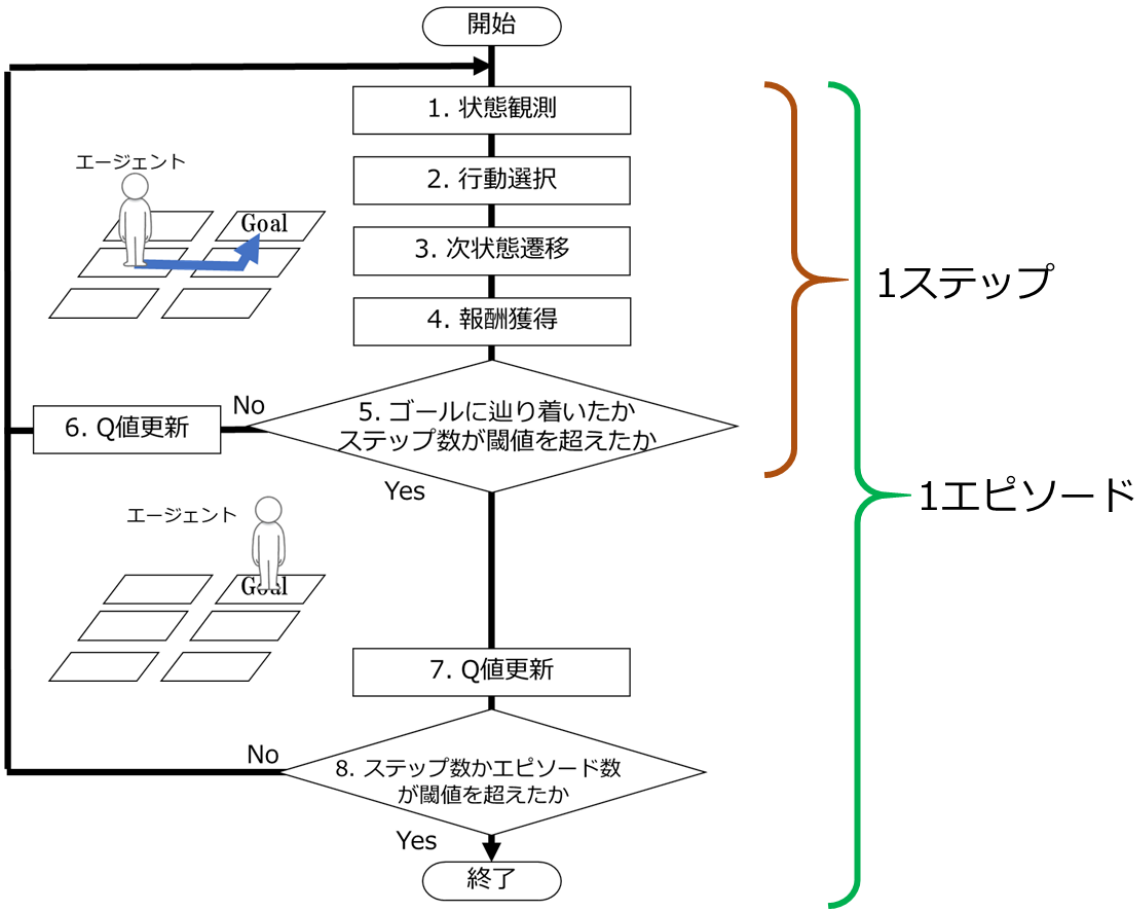


図 2.4　Q 学習のアルゴリズムフロー

- 状態行動価値 (Q 値)
  Q 値は状態行動価値と呼び，状態に紐づいた行動の価値を示している．Q 値は全ての状態と全ての行動の組み合わせの数だけ存在し，各ステップの学習の際に式 (2.10) で Q 値を更新する．式 (2.10) は強化学習エージェントにおける Q 学習の更新式である．

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)], \tag{2.10}$$

  s，a は現在観測した状態と実行した行動を示し，その時の Q 値を $Q(s,a)$ で表現

する．r は獲得報酬，$\max_{a'} Q(s', a')$ はその次に観測した状態における最大の Q 値を表す．このとき s' と a' は次に観測した状態とその時の任意の行動を示している．また，γ は割引率と呼ばれる次状態の Q 値に掛ける重みを表した 0 から 1 までの実数値である．割引率により，未来に得られる報酬をどれだけ期待値として組み込むかが決まり，大きな割引率とすることで，より未来に得られる報酬を Q 値本来の割合を高めるように計算できる．また，学習率 α は，1 回の更新で得られた価値を何割利用するかを決める．学習率を大きく設定することで，新たに得られた情報 (報酬値や未来の Q 値の最大値) を Q 値に大きく反映させることができ，これにより新しく得た情報を優先的に学習する．式 (2.10) から Q 値は割引期待報酬和に収束する．割引期待報酬和とはその行動をとった時に得られる報酬の重み付き和の期待値である．

- Q 値の収束性

  Q 学習における大きな特徴として Q 値の収束性が挙げられる．以下にそれを証明する．

  $q(s_t, a_t)$ を状態 s_t 行動 a_t の時の Q 値，s_t, a_t をそれぞれステップ数 t の時の状態と行動，r_{t+1} を $t+1$ の時の報酬値そして α, γ をそれぞれ学習率と割引率とすると，Q 学習における Q 値の更新は以下の式 (2.11) で行う．

$$q(s_t, a_t) \leftarrow q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a_{t+1}} q(s_{t+1}, a_{t+1}) - q(s_t, a_t) \right] \tag{2.11}$$

  式 (2.11) を単純に考えるため $r_{t+1} + \gamma \max_{a_{t+1}} q(s_{t+1}, a_{t+1})$ を Q^{t+1} とおく．すると式 (2.11) は以下の式 (2.12) で表せる．

$$q(s_t, a_t) \leftarrow q(s_t, a_t) + \alpha \left[Q^{t+1} - q(s_t, a_t) \right] \tag{2.12}$$

  式 (2.12) の更新回数を i として $q(s_t, a_t) = q_i^t$ と置き換え，Q_i^{t+1} を状態 s_{t+1} 行動 a_{t+1}，式 (2.12) の更新回数 i のときの Q^{t+1} とすると，式 (2.12) は以下のように書き換えられる．

$$q_i^t = q_{i-1}^t + \alpha \left[Q_i^{t+1} - q_{i-1}^t \right] \tag{2.13}$$

  式 (2.13) は i を変数とする漸化式であるので，以下のように展開する．

$$\begin{aligned}
q_i^t &= (1-\alpha) q_{i-1}^t + \alpha Q_i^{t+1} = q_{i-1}^t + \alpha \left[Q_i^{t+1} - q_{i-1}^t \right] && (2.14)\\
&= (1-\alpha)^2 q_{i-2}^t + \alpha(1-\alpha) Q_{i-1}^{t+1} + \alpha Q_i^{t+1} && (2.15)\\
&= \cdots \\
&= (1-\alpha)^i q_0^t + \alpha(1-\alpha)^{i-1} Q_1^{t+1} + \cdots + \alpha(1-\alpha) Q_{i-1}^{t+1} + \alpha Q_i^{t+1} && (2.16)\\
&= (1-\alpha)^i q_0^t + \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} Q_j^{t+1} && (2.17)
\end{aligned}$$

q_0 は Q 値の初期値であり，ここでは 0 と仮定すると学習途中の Q 値は以下のように表すことができる．

$$q_i^t = \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} Q_j^{t+1} \tag{2.18}$$

ここで，Q 値の初期値を 0 と仮定し，上記の式 (2.18) である $q_i^t = \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} Q_j^{t+1}$ を数学的帰納法により証明する．

- $i = 1$ のとき

$$q_1^t = \alpha \sum_{j=1}^{1} (1-\alpha)^{1-j} Q_j^{t+1} \tag{2.19}$$

$$= \alpha Q_1^{t+1} = q_0^t + \alpha \left[Q_1^{t+1} - q_0^t \right] \tag{2.20}$$

よって $i = 1$ のとき成り立つ

- $i = k$ のとき成り立つと仮定する

$$q_k^t = \alpha \sum_{j=1}^{k} (1-\alpha)^{k-j} Q_j^{t+1} \tag{2.21}$$

$$\tag{2.22}$$

- $i = k+1$ のとき

$$q_{k+1}^{t+1} = q_k^{t+1} + \alpha \left[Q_{k+1}^{t+1} - q_k^{t+1} \right] \tag{2.23}$$

$$= (1-\alpha) q_k^{t+1} + \alpha Q_{k+1}^{t+1} \tag{2.24}$$

$$= (1-\alpha) \alpha \sum_{j=1}^{k} (1-\alpha)^{k-j} Q_j^{t+1} + \alpha Q_{k+1}^{t+1} \tag{2.25}$$

$$= \alpha \sum_{j=1}^{k} (1-\alpha)^{(k+1)-j} Q_j^{t+1} + \alpha (1-\alpha)^{(k+1)-(k+1)} Q_{k+1}^{t+1} \tag{2.26}$$

$$= \alpha \sum_{j=1}^{k+1} (1-\alpha)^{(k+1)-j} Q_j^{t+1} \tag{2.27}$$

よって $i = k+1$ のときも成り立つ．

以上から自然数 i のとき常に $q_i^t = \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} Q_j^{t+1}$ が成り立つといえる．

次に今証明した式を用いてゴールへたどり着く行動の Q 値について考える．更新回数 i，報酬値 R，時間 t のとき，ゴールにたどり着く行動の Q 値は

$$q_i^t = \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} R \tag{2.28}$$

時間が $t-1$ のときの Q 値，つまりゴール前の状態へ向かう行動の Q 値は，i_j を Q_j^t のときの更新回数とすると

$$q_i^{t-1} = \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} Q_j^t \tag{2.29}$$

$$= \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} \gamma q_{i_j}^t \tag{2.30}$$

$$= \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} \gamma\alpha \sum_{k=1}^{i_j} (1-\alpha)^{i_j-k} R \tag{2.31}$$

これを繰り返していくと，ゴールの n ステップ前の状態へ向かう行動の Q 値は以下のように推測される．

$$q_i^{t-n} = \gamma^n \alpha^{n+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \sum_{j_2=1}^{i_{j_1}} (1-\alpha)^{i_{j_1}-j_2} \cdots \sum_{j_{n+1}=1}^{i_{j_{n-1}}} (1-\alpha)^{i_{j_{n-1}}-j_n} R \tag{2.32}$$

ここで簡単のため，更新回数 $i, i_{j_1}, \cdots, i_{j_{n-1}}$ をすべて一定として，数学的帰納法により証明を行う．

全ての正の整数 n において $q_i^{t-n} = \gamma^n \alpha^{n+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \sum_{j_2=1}^{i} (1-\alpha)^{i-j_2} \cdots \sum_{j_{n+1}=1}^{i} (1-\alpha)^{i-j_n} R$ が成り立つことを証明する．

- $n=0$ のとき

$$q_i^t = \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} R \tag{2.33}$$

  よって $n=0$ のとき成り立つ

- $n=k$ のとき成り立つと仮定する

$$q_i^{t-k} = \gamma^k \alpha^{k+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \cdots \sum_{j_{k+1}=1}^{i} (1-\alpha)^{i-j_{k+1}} R \tag{2.34}$$

- $n=k+1$ のとき

$$q_i^{t-(k+1)} = q_{i+1}^{t-(k+1)} + \alpha \left\{ Q_j^{t-k} - q_{i-1}^{t-(k+1)} \right\} \tag{2.35}$$

$$= (1-\alpha) q_{i-1}^{t-(k+1)} + \alpha\gamma q_i^{t-k} \tag{2.36}$$

$$= \cdots \tag{2.37}$$

$$= (1-\alpha)^i q_0^{t-(k+1)} + (1-\alpha)^{i-1} \alpha\gamma q_i^{t-k} + \cdots + \alpha\gamma q_i^{t-k} \tag{2.38}$$

$q_0^{t-(k+1)}$ は更新数が 0 で Q 値の初期値となる．仮定より Q 値の初期値は 0 であるから $q_0^{t-(k+1)} = 0$．よって

$$q_i^{t-(k+1)} = (1-\alpha)^{i-1}\alpha\gamma q_i^{t-k} + (1-\alpha)^{i-2}\alpha\gamma q_i^{t-k} + \cdots + \alpha\gamma q_i^{t-k} \tag{2.39}$$

$$= \alpha\gamma \sum_{m=1}^{i} (1-\alpha)^{i-m} q_i^{t-k} \tag{2.40}$$

$$= \alpha\gamma \sum_{m=1}^{i} (1-\alpha)^{i-m} \gamma^k \alpha^{k+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \cdots \sum_{j_{k+1}=1}^{i} (1-\alpha)^{i-j_{k+1}} R$$

$$= \gamma^{(k+1)} \alpha^{(k+1)+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \cdots \sum_{j_{(k+1)+1}=1}^{i} (1-\alpha)^{i-j_{(k+1)+1}} R \tag{2.41}$$

よって $i = k+1$ のときも成り立つ．

以上から正の整数 n のとき常に $q_i^{t-n} = \gamma^n \alpha^{n+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \sum_{j_2=1}^{i} (1-\alpha)^{i-j_2} \cdots \sum_{j_{n+1}=1}^{i} (1-\alpha)^{i-j_n} R$ が成り立つといえる．

この式において学習回数が限りなく大きくなるとき，つまり $i \to \infty$ のとき

$$\lim_{i\to\infty} q_i^{t-n} = \lim_{i\to\infty} \gamma^n \alpha^{n+1} \sum_{j_1=1}^{i} (1-\alpha)^{i-j_1} \cdots \sum_{j_{n+1}=1}^{i} (1-\alpha)^{i-j_n} R \tag{2.42}$$

ここで，等比数列の和の公式から

$$\sum_{j=1}^{i} (1-\alpha)^{i-j} = -(1-\alpha)^i \frac{1 - (\frac{1}{1-\alpha})^i}{\alpha} \tag{2.43}$$

$$= \frac{1}{\alpha}\left\{1 - (1-\alpha)^i\right\} \tag{2.44}$$

よって式 (2.42) は学習率 α が 1 よりも小さいため

$$\lim_{i\to\infty} q_i^{t-n} = \lim_{i\to\infty} \gamma^n \alpha^{n+1} \left[\frac{1}{\alpha}\left\{1 - (1-\alpha)^i\right\}\right]^{n+1} R \tag{2.45}$$

$$= \gamma^n \alpha^{n+1} \left(\frac{1}{\alpha}\right)^{n+1} R \tag{2.46}$$

$$= \gamma^n R \tag{2.47}$$

以上からゴールから n ステップ手前の状態へ向かう行動の Q 値は $\gamma^n R$ に収束する，つまり Q 値が割引期待報酬和に収束することが示された．

- アルゴリズム

Q 学習のアルゴリズムを下記に示す．Q 学習ではまず Q 値を初期化し，初期状態 s_0 及び最終状態 s_n を設定する (1,2 行目)．その後エージェントの状態を s_0 に設定

し，学習を開始する (3,4 行目)．学習中，エージェントは Q 値に基づき行動選択をし，選択した行動 a を実行，その結果として報酬 r を獲得し，次の状態 s' を観測する (6,7 行目)．このとき $ActionSelect(Q, s)$ は行動選択関数であり，2.2.1 節にて説明した行動選択方法に従い行動を決定する関数である．そして獲得した報酬を基に Q 値を更新する (8 行目)．以上のステップを最終状態 s_n となるまで繰り返す．

Algorithm 1 Q-Learning

1: $Qvalues : \forall s \in S, \forall a \in A | Q(s, a) = 0$
2: 初期状態 s_0, 最終状態 s_nの設定
3: **for** $cycle = 1$ to MAX_CYCLE **do**
4: $s = s_0$
5: **while** $s \neq s_n$ **do**
6: 行動 $a = ActionSelect(Q, s)$
7: 行動 a を実行，報酬 r を獲得後，次状態 s'を観測
8: $Q(s, a) = Q(s, a) + \alpha \left[r + \gamma \max_{a' \in A(s')} Q(s', a') - Q(s, a) \right]$
9: **end while**
10: **end for**

2.2.4 Profit Sharing (PS)

Profit Sharing (PS) は Grefenstette により提案されたエピソードを基本に学習を行う強化学習である [27]．つまり PS はエージェントが目的を達成した際に今までの行動すべてを一括で評価するブートストラップ型強化学習の 1 つである．下記の式 (2.48) は状態行動価値 (Q 値) の更新式である．式 (2.48) において，$Q(s(t), a(t))$ がエピソード中のステップ t のときの状態 $s(t)$ とそのとき取る行動 $a(t)$ の Q 値であり，学習率 α によりそのときの利得関数である $PS(s(t), a(t))$ の値から Q 値を更新する割合を決める．利得関数は下記の式 (2.49) に従って計算され，式中の R は報酬値，S は割引率という 0 より大きく 1 よりも小さい実数定数であり，N はエピソード数全体のステップ数を示している．

$$Q(s(t), a(t)) \leftarrow (1 - \alpha) Q(s(t), a(t)) + \alpha PS(s(t), a(t)), \tag{2.48}$$

$$PS(s(t), a(t)) = S^{N-t} R \tag{2.49}$$

図 2.5 は PS の概略図である．PS はエージェントがゴールへ到達した (1 エピソードが終了した) 時点でその経路に存在する状態と行動を逐次的に更新する．例えば，エージェント B は図のケースにおいて 8 ステップでゴールに到達しているため，N は 8 となり，スタートからとる行動は下記の式にて計算される．そして次状態のときは，$S^8 R$ を $S^7 R$

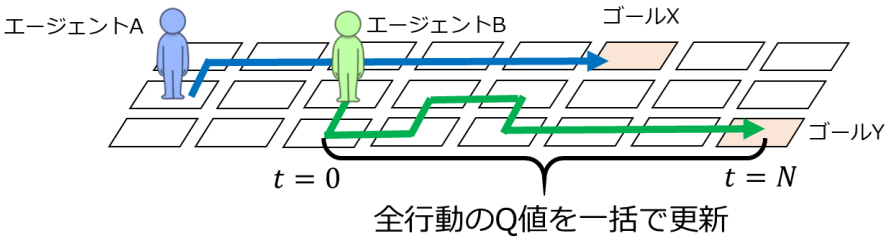


図 2.5　PS の概略図

へ変更した上で同様に計算される．

$$Q(s(0), a(0)) \leftarrow (1-\alpha)Q(s(0), a(0)) + \alpha S^8 R \tag{2.50}$$

また下記の式は報酬関数である．本論文において報酬は下記の式に従い与えられる．PS においては協調の成功/失敗を報酬で示すため，下記の式 (2.51) に示す報酬関数を採用している．下記の式では他エージェントが同じゴールへ到達していれば報酬値が得られず，そうでない場合に外部報酬値 r に全エージェントがゴールに到達するまでのステップ数 N を割った数値を報酬とする．このとき各エージェントが別々のゴールへ到達すれば獲得報酬 (分子) は最大となり，その到達ステップ数 (分母) が最小であればエージェントの獲得する報酬値が最大となるため，PS はこの報酬設定により協調行動を学習することができる．ただし，これはエージェント全体の到達ステップ数や獲得報酬など通信しなければ得られない情報であるため，実質 PS では通信なしの学習は難しい．

$$R = \begin{cases} \frac{allreward}{\max_i step_i} & if\ \text{エージェントが別々にゴール} \\ 0 & otherwise \end{cases} \tag{2.51}$$

また，式 (2.51) の設定は最小限の設定であり，エージェントが学習する上で最適であるといえるものではない．強化学習における報酬設計は状態行動空間全体を全て網羅した探索を可能にするように設定することが望ましい [24]．その点を加味すると，式 (2.51) は最適なゴールへエージェントが到達すれば最大の報酬を獲得可能であるという点では適切であるが，そこへ至るまでに獲得する報酬値は最適なゴールへ導くようにはできていないため，その点は適切ではない．例えば，同じゴールへ到達するエージェントが多ければ多いほど獲得報酬値はその分だけ割られていくため，ゴールへ到達する行動はある程度適切な行動であると評価すべきところを評価できない．それを打破するためには分母のステップ数を対数値に置き換えるなどしてステップ数による報酬値の影響を小さくすることが考えられる．しかしながら，最適な報酬関数設計は決まった理論があるわけではないため，その方法により学習が促進する保証はない．その意味で本研究の提案手法は報酬設計を考える必要がないため，PS よりも手法の適用が容易であるといえる．

2.3　マルチエージェントシステム

マルチエージェントシステムはエージェントと呼ばれる内部状態，意思決定，通信機能を備えた主体が複数集まってできたシステムである．その大きな特徴はミクロマクロループと呼ばれ，それは社会経済現象のように，社会経済（マルチエージェントシステム）のマクロな動向がそれを構成する個々の人（エージェント）のミクロな働きの集積によって構成され，各エージェントは社会経済の変化に影響してその振舞いを変化させる循環のことである．つまりマルチエージェントシステムは，人間，ロボット，自動車，その他様々な主体が複数存在する大きなシステムの振舞いや，それを機能させたときの問題点をシミュレーションにより解明できるシステムのことを指す．

2.3.1　エージェント

エージェントとは代理人という意味の英語であり，世の中の様々な主体を代替するものである．具体的には，複数種類のエージェントが存在するが，ここでは学習エージェントについて説明する．図 2.6 はエージェントの構造を示している．エージェントは外部環境から情報を得るセンサ，知的活動を行うプロセッサ，そして環境に対して行動して作用するためのエフェクタの 3 つの機構を持ち，センサで知覚した情報を基に知的活動を行い，エフェクタにより行動を取る．このときセンサ，エフェクタ，知的活動の設計によりエージェントの振舞いが変わり，様々な行動主体を表現することができる．例えば，ロボットであればカメラ等のセンサとアームと移動用の足や車輪のエフェクタを持ち，画像処理により障害物を検知して避けるようにエフェクタを動かすプログラムを知的活動として設計する．特に本研究におけるマルチエージェント強化学習では，この知的活動を学習により獲得することを目指す．つまり，センサ情報に従いどのエフェクタをどのように動かすかを強化学習により制御する．

2.3.2　エージェントのモデリング

エージェントはセンサ，エフェクタ，知的活動を持つものであるが，エージェント自体に明確な定義があるわけではなく，適用する問題に応じてその能力が変わる [28]．その結果として 1 つの問題に対してエージェントの捉え方が多数存在する．図 2.7 は複数ロボットにおける物資運搬問題を想定した時のエージェントのモデリング方法の違いを示している．上段は複数ロボットの物資運搬問題を示し，下段の左側と右側がそれぞれのモデリングの違いを示している．下段左側は各ロボットをエージェントにした時のモデルであり，下段右側はロボットと荷物，届け先すべてを見渡し，各ロボットに命令を送れるエージェ

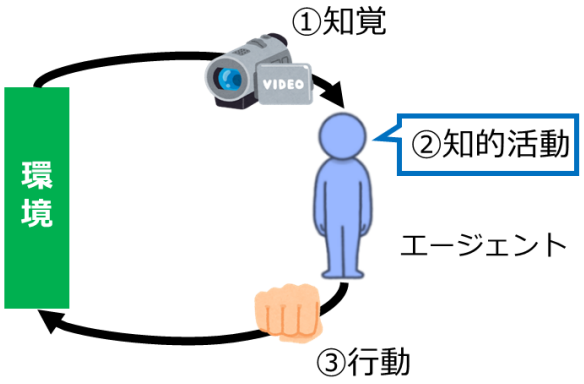


図 2.6　エージェント構造

ントを想定した時のモデルである．これはどちらが優れているということはないが，それぞれモデル化できる条件が異なり，その性質も異なる．まずロボットをエージェントにする方法であるが，これは一般的なマルチエージェントシステムの考え方に基づくモデル化であるといえる．これはロボットの機能を代替したエージェントを用意するため，ロボットの動作を学習する上で適したモデル化である．またエージェントの振舞いを考える上でロボットと同様の入出力を持つため，例えばロボットが増えるなどしても入力内容がさほど変わらなければ，柔軟に学習内容を変更することができる．一方で下段右側のモデルでは将棋の盤面を見るように，ロボットと荷物と届け先すべてを見渡せるエージェントが 2 体のロボットの移動方針を立てる．これはある意味で神の視点を持っており，全体を俯瞰した時のロボットの動きを決定できるという点で優れている．しかしながらこのモデルはロボットをエージェントとしたときと比較して必要な情報が多く，フィールドロボティクスのようにその情報を獲得できる保証がない場合に適用できないなどの制約がある．またエージェント 1 体に求められるリソースが大きく，より高度な問題をモデル化するとき適切な答えを導くことができない．また仮にコンピュータ技術が発展し，その高度な問題を解き切る計算能力が存在したとしても，エージェントの入力情報が複雑化し，その入力に対する適切な出力を返すための知的活動も複雑化していくため，人間の手でそれを設計することは現実的に不可能であるといえる．もっとも，従来法の中では図の下段左側と右側双方の要素を部分的に併せ持つモデルも存在しており，これらのエージェントの設計方法は適用する問題次第で変わってくる．以上を鑑みて本研究では，ロボット同士通信を行うことなく協調行動を獲得することを目指しており，下段左側のモデル以外は不適格といえる．また，下段左側のモデルはマルチエージェントシステムにおける基本的なモデルであり，このモデルにおいて性能を検証すればおよそほとんどの制約を考慮することなく実問題に適用することが可能となる．

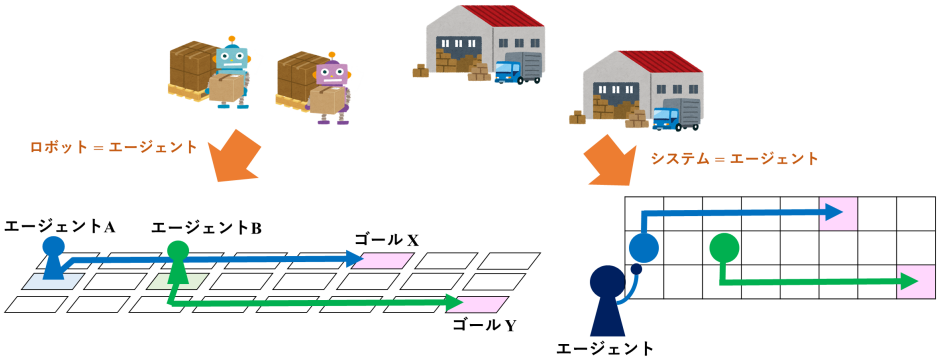


図 2.7　同じ問題におけるエージェントの設計方法の違い

2.3.3　相互作用

マルチエージェントシステムにおける特徴として相互作用というものが存在する．これは複数エージェントが存在した時に他エージェントの動作に影響されて自身の取るべき行動や目的が変化することを指す．図 2.8 はフォーメーション問題におけるロボット（エージェント）の相互作用を示している．フォーメーション問題とは決められた数のロボットがフォーメーションを作ることを目的とした問題であり，図においては 4 箇所ある指定された地点（橙のマス）に協力して到達することが目的となる．このとき矢印は各ロボットの経路を示しており，赤矢印のロボットが相互作用により影響を受けたロボットである．このとき赤矢印のロボットに注目すると，上段の状況では他のロボットが矢印に従ってフォーメーションを作るため，注目するロボットは一番近くの地点に到達する．一方で下段の状況では，1 体のエージェントが到達する地点を変化させたため，注目するロボットは到達する地点を変更せざるを得ない．このように，他のエージェントの振舞いにより影響されて自身の振舞いを変えることを相互作用と呼ぶ．マルチエージェントシステムにおいてはこのような相互作用が様々な場面で起こるため，それをうまく機能させるようにエージェントの知的活動を設計することは非常に難しい．例えば図 2.8 でいえば各エージェントに「最も近い地点に到達する」ように知的活動を設計しては遠くの地点に到達するロボットがいなくなり，フォーメーションを形成できない．また「最も近い地点に到達するが，他にエージェントがいれば別の近い地点に到達する」ように設計すればフォーメーションは形成できるが，それは効率的であるとはいえない．つまりマルチエージェントシステムに強化学習を導入する理由は人間の集団のように各々が相互作用により知的活動を適応的に変更し，集団を適切に機能できるようにするためである．

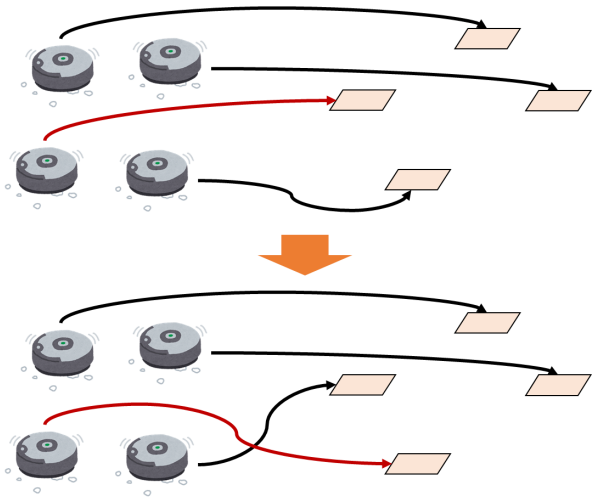


図 2.8　フォーメーション問題におけるエージェントの相互作用

2.4　マルチエージェント強化学習と協調行動

図 2.9 は，n 体のエージェントにおける学習構造を示している．各エージェントは環境を観測し，その状態の一部もしくは全部を状態 s_i として入力し，報酬もその一部または全部 r_i を獲得する (i は任意のエージェントを識別する番号である)．また各エージェントが行動 a_i を実行し，その直積により環境が変わり，次の状態に遷移する．このとき，環境は全エージェントの行動の直積で決まるため，1 体のエージェントが同じ行動を選択し続けても他のエージェントの行動が変化すれば次の状態や報酬値が変化し，適切な行動が学習できない．そのため，従来マルチエージェント強化学習においては通信などを用いて他のエージェントの動作を知ることで対処する．つまり，現在各エージェントは状態 s_i を観測しているが，全エージェントがそれぞれの状態を共有する (つまり全エージェントの状態を環境状態の s と等しくする) ことで，自身の状態が変化しなくとも他エージェントの状態の違いから状況を分けて学習することができるため，適切な行動を学習することができる．しかしながらこの情報共有は現実問題に適用する上での障害となりうる．

2.4.1　エージェント単位の学習

本研究では各エージェントが独立に学習して方策を獲得するが，それを統合して 1 つの解を求めるというものではない．つまり各エージェントが学習した方策がそのまま出力され，それが対応するロボットや自動車などの主体に導入される．つまり，図 2.10 に示す通り，各エージェントがそれぞれ独自の Q 値を持ち，お互いにそれを共有することなく独立に学習する．そのため，マルチエージェントにおける強化学習は報酬に基づく自身の

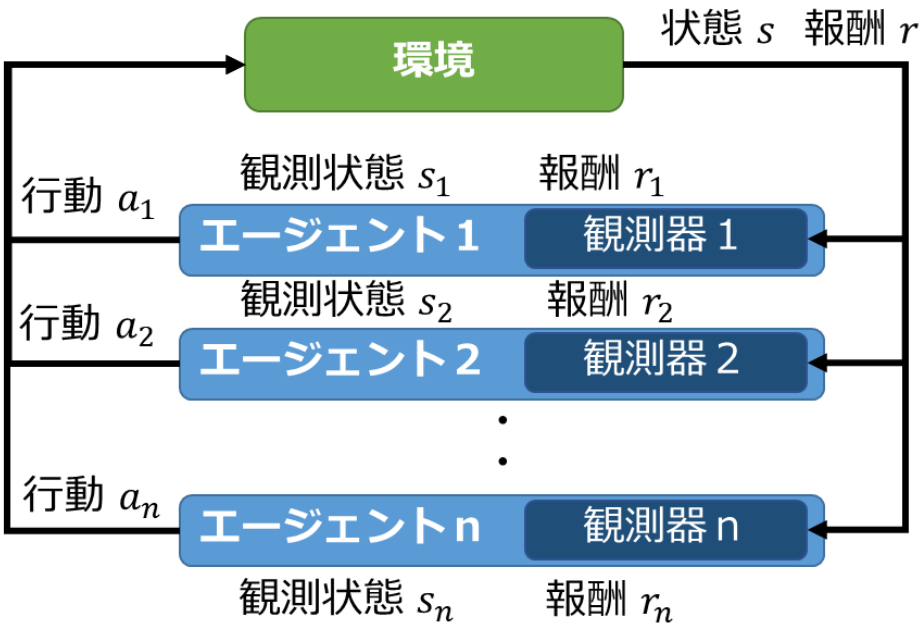


図 2.9　マルチエージェントシステムの学習構造

学習の他に，他エージェントに対する相互作用も考慮する必要がある．

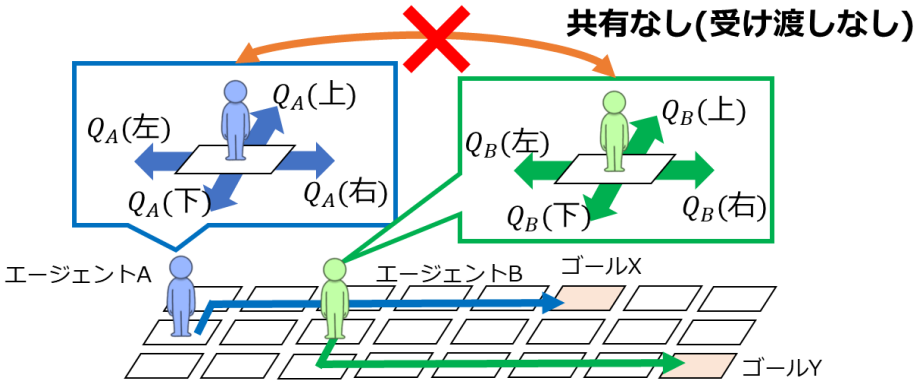


図 2.10　マルチエージェントシステム環境における強化学習

2.4.2　協調行動

上記マルチエージェント強化学習の目的達成に向けて，協調は重要な要素となる．それは分散人工知能として機能するマルチエージェントシステムにとって，エージェント全体の目的と各エージェントがそれぞれ持つ目的が異なるため，各エージェントがお互いに協力することは必要不可欠であることが起因している．一般的な定義として協調は「共同調和」の略語であり，「利害の対立する者同士がおだやかに相互間の問題を解決しようとす

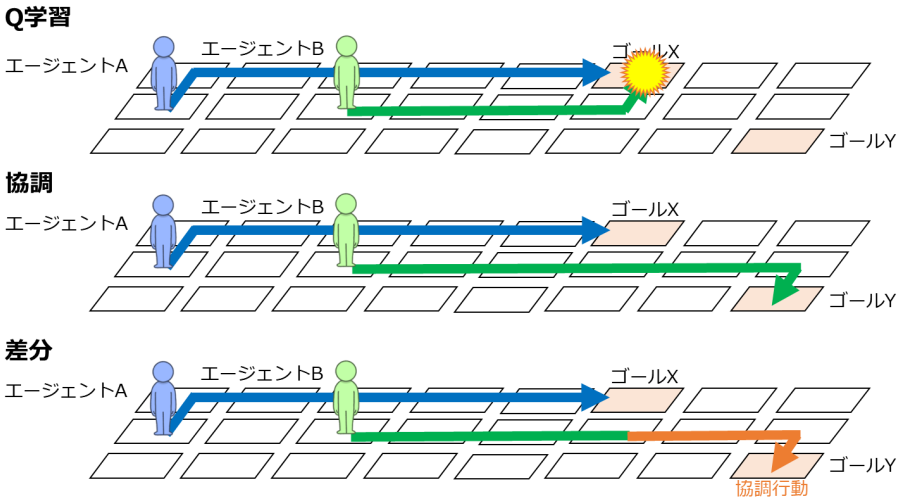


図 2.11　協調行動

ること」という意味である．しかし，人間社会でもそうであるように，マルチエージェント強化学習においても協調の意味合いは複数存在する．具体的には下記 2 点の協調が従来行われている．それはすなわちエージェントの知識をどのように利用するかということに尽きる問題であり，石田らの報告によればエージェントのコミュニケーションには知識共有と調整の 2 種類が存在し，それは下記の 2 点に対応する．[28]

1. 最も性能の良いエージェントの学習結果を全体で共有 (他エージェントの教育)
2. 全体の目的のために相手エージェントの動作を補助 (他エージェントと協働)

協調行動とは協調によって変化した行動のことを示す．図 2.11 は協調行動を示している．この図は 2 体エージェントによる迷路問題を示しており，各エージェントが協調して最短で全てのゴールに到達することを目指す．図において上段が Q 学習の結果，中段が協調した際の結果，下段がその差分を示している．通常 Q 学習を行えば，各エージェントが単位時間当たり最大の報酬を獲得するように学習するため，2 体のエージェントは共にゴール X に到達する行動を学習する．しかしそれは適切な学習をしたとは言えず，マルチエージェントとしての最適方策はエージェント A，B がそれぞれゴール X，Y に到達することであり，つまり図の中段のようにエージェント B があえて自身の利益とならない行動 (ゴール Y へ到達する行動) をとることで協調を実現する．つまり，協調行動とは，それぞれのエージェントが自身の利益最大化を目指した際に取るべき行動とは異なる行動であり，図の下段の橙線の行動が協調行動ということになる．

2.4 節にて説明した 2 種類の協調の中で，1 つ目の協調は他のエージェントの学習を補助するという意味では協調行動であるが，エージェント全体の目的と個々のエージェントの目的はほぼ等しい．その意味でこの協調はエージェント 1 体の能力を極限まで伸ばすた

めに利用しており，分業が必要となる困難な問題を目指していく上では利用することが難しい．そのため本研究では，2 つ目の協調である各エージェントの適切な行動が異なる問題に取り組む．更に本研究では，エージェント間の情報通信や観測による他エージェントの情報利用を制限しているため，通常の協調方法とは異なるアプローチをとる．従来マルチエージェント強化学習において，協調行動を学習するためにはそれぞれのエージェントが学習した知識をどのように他のエージェントに伝えて利用するかが重要となる．Silva らによればエージェントの知識利用戦略は 6 個存在し，それは転移学習，デモンストレーション学習，模倣学習，アドバイス，カリキュラム学習，その他である [29]．しかしながら本研究において他エージェントと通信や観測による情報利用はできないため，これらの手法は利用できない．

2.5　研究の位置づけ

2.5.1　着眼点

エージェントに与える情報

本論文において，エージェントの情報獲得方法は 3 種類存在する．図 2.12 は情報利用方法の違いを示しており，それぞれ左が全情報利用，中央が近傍情報のみ利用，右が情報利用無しの前提を示している．具体的にはエージェントが協調に必要な情報をすべて利用可能であるという全情報，自身の近くにいるエージェントの情報のみ利用可能な近傍情報のみ，そして近傍であれエージェントの情報を利用することができない情報なしの 3 種類である．

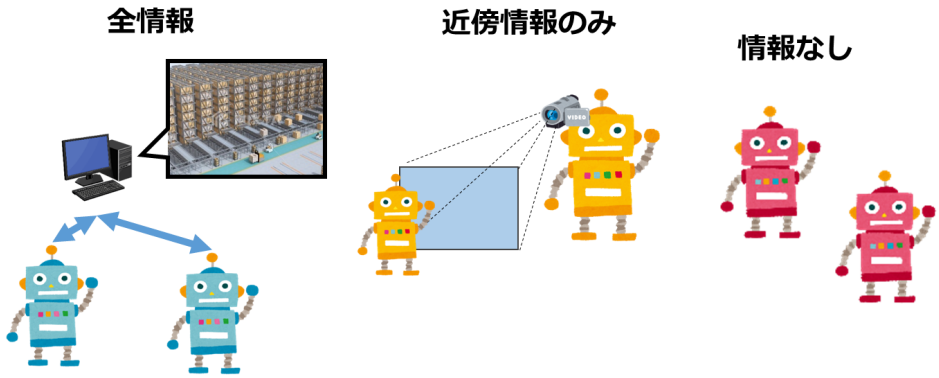


図 2.12　エージェントに与える情報の違い

マルチエージェントシステムにおける動的環境

本論文では，静的環境と動的環境の 2 種類の環境を想定する．そして，動的環境の中でも 2 種類のものを想定する．1 つはエージェントの振舞い以外環境の変化がない静的環境，もう 1 つは環境の形状やスタート・ゴール位置などが変化する空間的環境変化，最後はエージェント数やゴール数，報酬値が変化するエージェント・ゴール数変化である．図 2.13 は迷路問題における，上段が静的環境，中段が 1 つ目の動的環境である空間的環境変化，下段がもう 1 つの動的環境であるエージェント・ゴール数変化を示しており，特に動的変化が起こる場合にはその前後の環境を橙の矢印によって示している．静的環境においては，環境もゴール (報酬とその位置) も変化がなく，その上で 2 体のエージェントが動くのみである．空間的環境変化においては，エージェント・ゴール位置が変化し，更に行き止まりや新たなマスが出現する等環境の形状も変化する．エージェント・ゴール数変化においては，エージェントやゴールの数が増減し，報酬値も変化する．

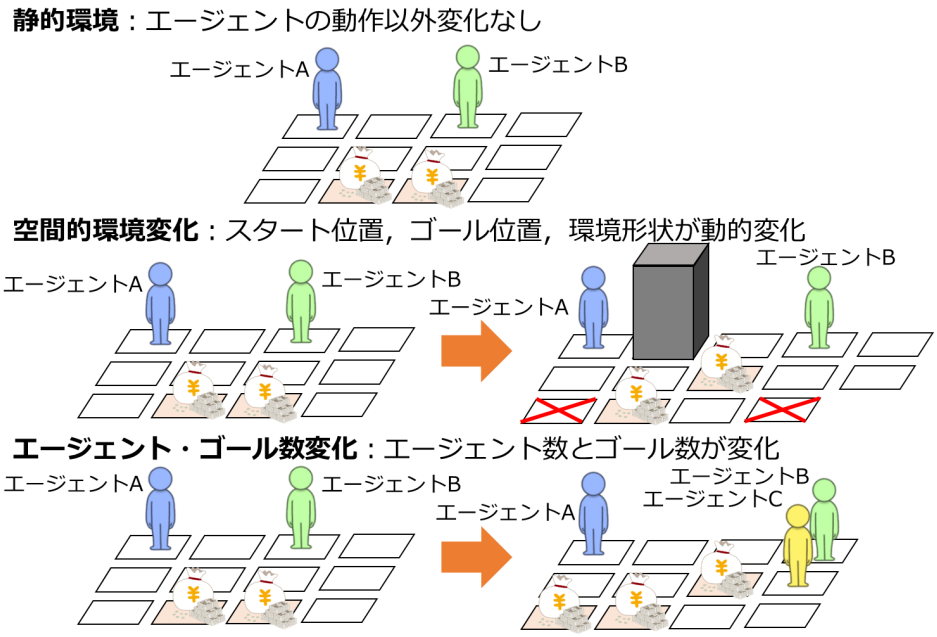


図 2.13　エージェントの学習環境の違い

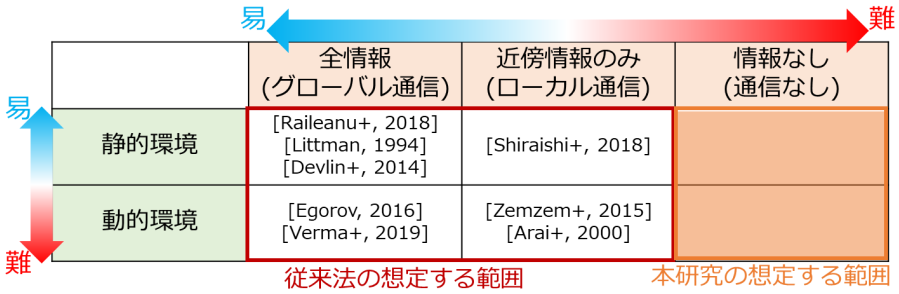
2.5.2　従来研究との関連性

以上の観点を踏まえると，本研究の位置づけは表 2.1 にて纏められる．表 2.1 の各行は想定する学習環境を示し，各列は協調行動学習のために利用する他エージェント情報を示している．なお動的環境においては前節の動的環境の内どれか 1 種類でも満たしている場合は動的環境としている．また利用する他エージェント情報に関しては，前節に

て説明した全情報を想定するか，ロボット制御のように自身の近傍のみの情報だけがセンサにより得られる近傍情報のみを想定するか，そのどれをも想定しない情報なしを想定するかの 3 種類が存在する．そして各項目にはその代表的な従来研究を配置している [10, 14, 30, 31, 32, 33, 19]．表 2.1 において，従来研究が想定する範囲が赤い枠で囲った部分であり，本研究は橙色の枠内に対する協調行動学習法を提案することを目的とする．以上の通り，本研究は従来のマルチエージェント強化学習とは全く異なる観点の協調行動学習法を確立する．そして本研究では，提案する協調行動学習法の合理性を理論的に示すことも行う．表 2.2 は各条件に属する手法が合理性を持つか否かを示している．表において合理性を持つ手法は全情報を利用可能でありかつ静的環境へ適用可能な手法に限られることがわかる．

本研究においては，エージェントに導入する「より困難な目的達成を目指す」という規則が，協調を達成するものであるという仮説を立て，その正当性を検証することで他エージェント情報を利用することなく協調行動を学習する．そのため，従来のマルチエージェント強化学習とは最終的に達成する協調は等しく，そのために学習する協調行動も同様であるが，それをエージェント同士の通信を介することなく達成するという意味で目指す協調行動が異なる．そして，エージェント同士は他のエージェントの情報はおろか存在すら認知することなく学習することになるため，各エージェントはシングルエージェントとしての学習をし，そのエージェントが複数集まることによりマルチエージェントにおける協調を実現することが本研究における新規的部分である．

表 2.1　本研究の位置づけ



以下では表 2.1 における各手法について紹介し，本研究の具体的な達成目標を述べる．

2.5.3　全情報＋静的環境の従来手法：Minimax-Q 学習と CBS-TA

・Minimax-Q 学習

Littman はゲーム理論におけるミニマックス戦略に基づき学習する Minimax-Q 学習という手法を提案している [10]．この手法では主に 2 体エージェントを想定しており，下記の式 (2.52) で Q 値を更新する．それぞれの添え字はエージェントの番号を示しており，

表 2.2　合理性証明の有無

例えば Q_1 はエージェント 1 の Q 値であり，a_1, a_2 はそれぞれエージェント 1 と 2 の行動である．式 (2.52) は Q 値に他エージェントの行動が含まれている点，そして次状態の最大の Q 値ではなく V_1 という値を割り引いて計算している点が通常の Q 学習と異なる．また式 (2.53) はその値 V_1 の計算方法を示している．式の π_1 は方策といい，エージェントが各状態において取る行動の確率を示しており，この式では方策による利得のミニマックス値を V_1 として計算していることが分かる．また方策もミニマックス値戦略に従ったものを選ぶ．

$$Q_1(s, a_1, a_2) \leftarrow Q_1(s, a_1, a_2) + \alpha \left[r + \gamma V_1(s') - Q_1(s, a_1, a_2) \right] \tag{2.52}$$

$$V_1(s) = \max_{\pi_1} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} \pi_1(s, a_1) Q_1(s, a_1, a_2) \tag{2.53}$$

Minimax-Q 学習ではゼロ和ゲームにおいて最適の Q 値に収束することが証明されている．ゼロ和ゲームとは，ゲーム理論におけるゲームの 1 つであり，2 体エージェントにとって片方の利益が片方の不利益となりるような，エージェントの利得の合計値が必ずゼロになるゲームをいう．表 2.3 はゼロ和ゲームの利得表を表している．各セルの数値は利得を示しており，左からエージェント A と B の利得を示している．ここはゼロ和ゲームであるので，各セルの利得の合計は 0 となる．ここでいえばエージェント A，B はそれぞれ政策 2，1 を選ぶように学習する．以上のように，強化学習をマルチエージェント環境に適用するためには，他エージェントの情報を Q 値の更新に取り入れることが一般的であり，そのためには観測や通信を行うことが必要となる．そのため，本研究により一般的 Q 学習の枠組みで協調行動を学習することには大きな意味がある．

・CBS-TA

Honig らは物流システムにおける経路プランニング問題に取り組み，従来すべての経路を探索した上でその中から適切なものを選び出すことに対し，要求に合わせてその場で経路生成を行い，それに合わせてタスク割り当てを行うように CBS を拡張した CBS-TA を提案している．更に，Honig らは CBS-TA がタスク割り当てと経路探索を両立して最適

表 2.3　ゼロ和ゲームの例

		エージェント B	
		政策 1	政策 2
エージェント A	政策 1	5,-5	-5,5
	政策 2	-3,3	6,-6

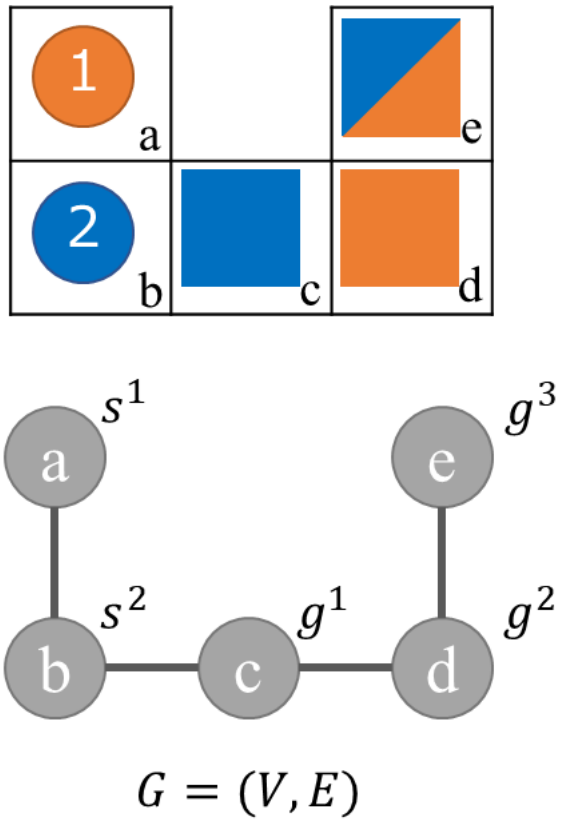


図 2.14　CBS-TA の環境とそのグラフ

解を求め，エージェントの合計のコスト最小化を可能にすることを証明している．

ここでは論文にある図 2.14 に示す迷路問題を使用して説明する．図の上部が迷路であり，右下のアルファベットは各状態の識別子，丸印が 2 体エージェントのスタート位置であり，四角印がゴールを示している．スタートとゴールは色に対応しており，エージェント 1 のゴールは d と e，エージェント 2 のゴールは c と e となる．また，下段はそれをグラフ化したものである．

図 2.14 においてタスク割り当て行列は式 (2.54) に従い，それぞれのコスト (到達までの最短ステップ数) 行列は式 (2.55) に従う．そしてこれらの行列から図 2.15 と図 2.16 に示す割り当てに関する探索木と探索森を CBS に従い構築する (詳しい説明は [34] を

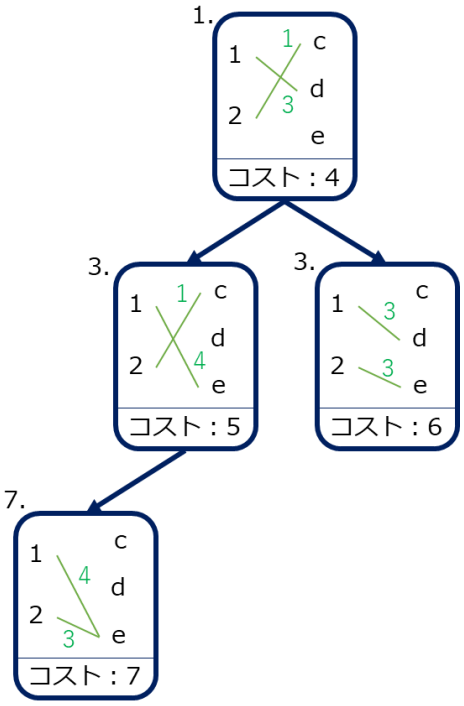


図 2.15　割り当ての組み合わせ探索木

参照のこと)．図 2.15 において，各ノードの数値とアルファベットはエージェントのスタートとゴールを示しており，緑線と数値は割り当てと行列 (2.55) に示したそのときのコストであり，その合計値をノードの下段に示している．そして図 2.15 の探索木に従って構築したものが図 2.16 である．ここではそれぞれのコストに含めて競合も計算している．CBS-TA ではこれらの候補を左上の番号の順番に探索していき，競合の無くなった図 2.16 の右上の経路を選び出す．以上のように CBS-TA はグラフ理論における探索法 CBS を利用し，最適経路を発見することを示すが，それはエージェント個々の学習を理論的に示すことはなく，エージェントの学習の結果得られた様々な経路の中から最適なものを選び出すことを理論的に示しており，個々のエージェントの振舞いを制御する強化学習によりエージェント全体の振舞いを理論的に示すことは非常に難しいと分かる．

$$A = \begin{matrix} \\ 1 \\ 2 \end{matrix} \begin{matrix} \begin{matrix} c & d & e \end{matrix} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} \end{matrix} \tag{2.54}$$

$$C = \begin{matrix} \\ 1 \\ 2 \end{matrix} \begin{matrix} \begin{matrix} c & d & e \end{matrix} \\ \begin{pmatrix} \infty & 3 & 4 \\ 1 & \infty & 3 \end{pmatrix} \end{matrix} \tag{2.55}$$

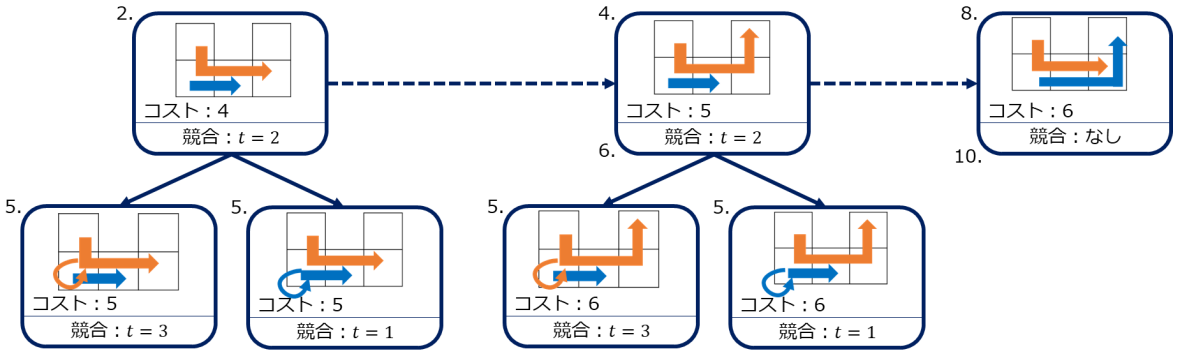


図 2.16　CBS-TA の探索森

2.5.4　近傍情報＋静的環境の従来手法：DPSM

白石らの提案する DPSM[35] は図 2.17 に示す迷路問題に取り組み，各エージェントがそれぞれに与えられたゴールの到達を目指す．図において，S1，S2，S3 がエージェントのスタート位置であり，G1，G2，G3 が各エージェントのゴールである．エージェントは各マスを移動できるが太線を乗り越えることはできない．またエージェントは図の左に示す周囲 4 近傍のみ知覚することができ，自身の位置を知ることはできないため，周囲の情報から場合分けしてその時に必要な行動を学習する．このとき，黄色いマスはエージェントが同じ状態として知覚する場所を示しており，例えば S2 のエージェントであれば G2 の下のマスを知覚したときに上に行く必要があるが，そもそもそのマスに行くためには同じ知覚のときに右の行動をとらなければならない．DPSM ではこれを解決するために他の S1，S3 のエージェントが G1 下のマスに来た S2 のエージェントの知覚情報にあえて入り込み，G2 下の状態と別の状態であるという判別をさせることによって解決を目指す．また，DPSM では PS に基づき協調行動を学習し，そのエピソードの内冗長な状態行動の

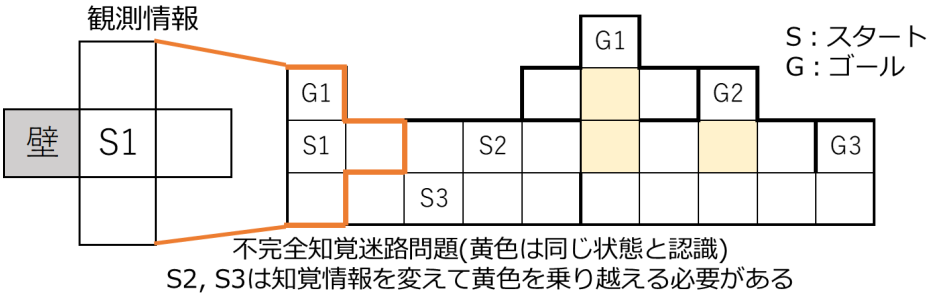


図 2.17　DPSM において使用する問題

割引率を操作することで，最短経路による協調行動の学習を実現する．具体的には，エピ

ソード中に同じ状態を観測した時や，目的状態へ到達しなかった経路の学習には，余分に割引率をかけて，獲得報酬から得られる利得の配分を小さくしている．

2.5.5　全情報＋動的環境の従来手法：MADRL

Egorov はマルチエージェントにおける深層強化学習のアプローチを試み，Multi-Agent Deep Reinforcement Learning (MADRL) を提案した [31]．ここでは追跡回避問題 (Pursuit-evasion problem) を取り上げている．この問題は追跡者と逃亡者の 2 種類のエージェントがフィールド上に存在しており，追跡者が逃亡者をとらえることが目的の問題である．図 2.18 は MADRL の学習メカニズムを示している．図の左側は問題環境を示しており，青と赤のロボットがそれぞれ逃亡者と追跡者となっており，黒いマスは通行できない壁になっている．MADRL はまず問題環境を Background Channel, Opponent Channel, Ally Channel, Self Channel の 4 個のチャンネルに分割し，4 チャンネルの画像データとして畳み込みニューラルネットワーク (Convolutional Neural Network: CNN) に入力する．そして各行動の Q 値を出力としてネットワークの重みを報酬に基づき更新し，各エージェントは自身にとってより良い行動を学習する．なおこのとき全エージェントを捕獲した (逃亡者と同じマスに追跡者が到達する) 時に報酬を獲得する．

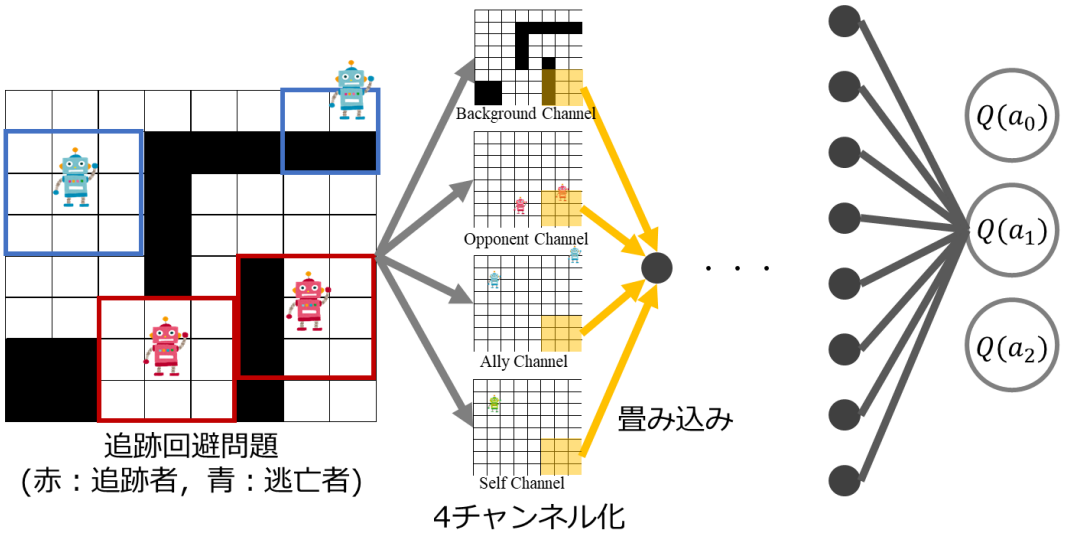


図 2.18　MADRL の学習メカニズム

またこれはエージェント 1 体の場合の話であり，MADRL ではこの学習の後に各エージェントの方策 (Q 値) を共有して，各エージェントの持つシミュレータを更新する．つまり図 2.18 はエージェント 1 体の中にある実験環境，自エージェント，味方の他エージェントと逃亡者のエージェントがシミュレータであり，味方の動きは各エージェントが持つ味方の方策によって決まる．そしてこの方策の共有によってシミュレータ上の味方の動き

が更新され，それに合わせて自身の動きも新たに学習されることで協調行動を学習する仕組みとなっている．

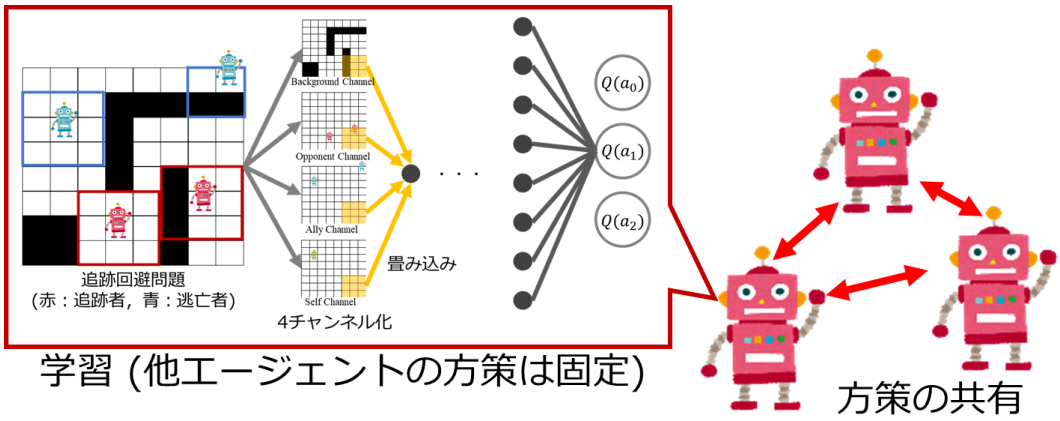


図 2.19　MADRL のエージェント間相互作用

2.5.6　近傍情報＋動的環境の従来手法：CMRL-MRMT

Zemzem らは大きく未知な動的環境上のエージェントの協調行動学習法を提案するために，従来法である D-DCM-MultiQ Method[36] を拡張した CMRL-MRMT という手法を提案している．式 (2.56) は D-DCM-MultiQ Method における Q 値の更新式である．この式において i はエージェントの識別番号であり，k はステップ数である．また $R_{k,i}(s_i, a_i)$ はステップ k でエージェント i が状態 s_i から行動 a_i を取った時の報酬値であり，$f(i,j)$ は重みパラメータであり，$\sum_{j=1}^{N} f(i,j) = 1$ となるように値を設定する．ちなみに j は任意の近隣エージェントの番号であり，その合計は N となる．つまりエージェントは自身の Q 値を更新する際に，次状態の最大の Q 値ではなく，近隣エージェントの Q 値の最大値の重み付き和を利用する．そうすることで局所的であるがエージェント全体の利益を想定した Q 値の更新が可能となる．

$$Q_{k+1,i}(s_i, a_i) = (1 - a_k)Q_{k,i}(s_i, a_i) + a_k * \left(R_{k,i}(s_i, a_i) + \gamma \sum_{j=1}^{N} f(i,j) \max_{a_j} Q_{k,j}(s'_i, a_j) \right) \tag{2.56}$$

CMRL-MRMT は上記 D-DCM-MultiQ Method に行動選択戦略と学習戦略の 2 つを導入して動的環境への追従を目指す．直接的貢献ではないためここでは行動選択戦略の詳しい説明は省くが，従来の行動選択戦略 EEP(exploration/exploitation policy) を Q 値が最大であり最近に更新された行動を優先的に選ぶように改良した CG-MPA を提案している．また下記の式 (2.57) は CMRL-MRMT における Q 値の更新式である．ここで

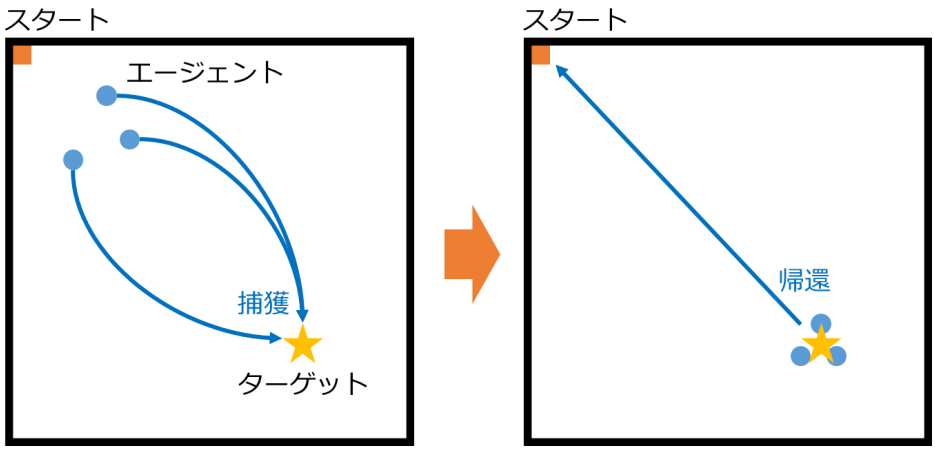


図 2.20　CMRL-MRMT において使用する問題

は t がステップ数であり，k は状態 s' において行動 b を取り Q 値を更新したエージェントの中で，最近更新したエージェントを示している．この機構により，直近で更新された Q 値を優先的に使用して Q 値を更新するため，環境変化に追従することができる．以上が CMRL-MRMT である．これを見ると分かる通り，この手法ではエージェントが Q 値の更新における環境変化前の情報への重みを小さくして，環境変化後に観測した情報に基づき行動を学習することで，動的変化に追従する手法であるため，観測情報や通信情報が正確でない場合において適切に動的変化に追従できないという問題点が存在することになる．また，CMRL-MRMT の機構は，環境変動には対応できても環境変化までには対処できないことを示唆している．

$$Q_{t,i}(s,a) = (1-\alpha)Q_{t-1,i}(s,a) + \alpha\left(R_{t,i}(s,a) + \gamma \max_{b} Q_{t,k}(s',b)\right) \tag{2.57}$$

以上に加え，Zemzem らの提案する CMRL-MRMT[20] は前節にて紹介した協調の内教育と呼ばれる 1 つ目の協調の例といえる．CMRL-MRMT は図 2.20 に示す問題に取り組んでいる．この問題は巣と呼ばれるスタート地点からターゲットに向かってエージェントが進み，それを確保する．その後エージェント全員でそのターゲットをスタート地点に戻すことを目指している．これを見ると分かる通り，全体のエージェントがほぼ同じ行動をとるため，より学習の進んだ (最も性能の良い) エージェントの情報は他のエージェントにとっても重要であるといえる．CMRL-MRMT は各エージェントが学習する際に，近傍にいる他のエージェントの学習結果を利用して性能の良いエージェントの動きを取れるように学習している．

2.6　本研究の解決方策

本研究における達成目的は下記の 2 点である．つまり，エージェントがそれぞれ別々のゴールへ到達し，エージェント全体の到達ステップ数を最小にすることである．これ以降は，その達成に向けたアプローチを紹介する．

- 全エージェントがそれぞれ異なるゴールに到達する
- エージェント全体のゴール到達ステップ数を最短にする

2.6.1　アプローチ

本研究は全エージェントが遠くのゴール到達を目指し学習し，遠くのゴールから最短ステップで到達可能なエージェントを設定するというアプローチをとる．そしてそのために合理的な目的設定法を提案し，その目的達成のために内部報酬を設定する．なお図 2.21 は 3 体エージェントの迷路問題における本研究のアプローチを示す．この図は迷路問題であり，ここでは，各エージェントがそれぞれ別々のゴールへ到達 (報酬を獲得) し，そしてその達成までのステップ数を最小にすることを目指す．そしてそのために，全エージェントをまず最も遠くのゴールへ向かわせ，そのゴールに最初に到達するエージェントを 1 体決定する (図の左側上段)．その後残るゴールからエージェントの到達するゴールがなくなるまでこれを繰り返す (図の左側中下段)．本アプローチはこのようにすることで，各エージェントが到達するゴールが存在しないということを防いでいる．図の右側は各エージェントの振舞いによる他エージェントへの影響を示している．例えば図の左側のようにエージェント A，B，C がそれぞれゴール X，Y，Z へ到達すると想定すると，エージェント C が到達するゴールを変更すれば，その変更したゴールに対応して，今までそのゴールへ到達していたエージェント (図の右側ではエージェント A) が到達できず他のゴール到達をめざす．そしてその影響により 3 体目のエージェントも学習をやり直す必要が出てくる．以上のようにマルチエージェントシステムにおいて各エージェントの動きはお互いに影響するものであるため，本アプローチにより確実に競合なくエージェントの到達ゴールを決めることには意味がある．

図 2.22 は本研究において達成する協調行動を示している．エージェント A，B，C はそれぞれゴール X，Y，Z に到達する行動を学習する．またエージェントの学習により，図の上段から下段にかけて学習方策が遷移する．本研究においては全エージェントが達成可能な最もステップ数のかかる目的達成を目指す．図でいえばまず全エージェントが最も遠くのゴール到達を目指して学習し，そのステップ数が最小のエージェントがそのゴールに達成する行動を学習する (図の上段)．その後残ったエージェント A，B が残りのゴールの

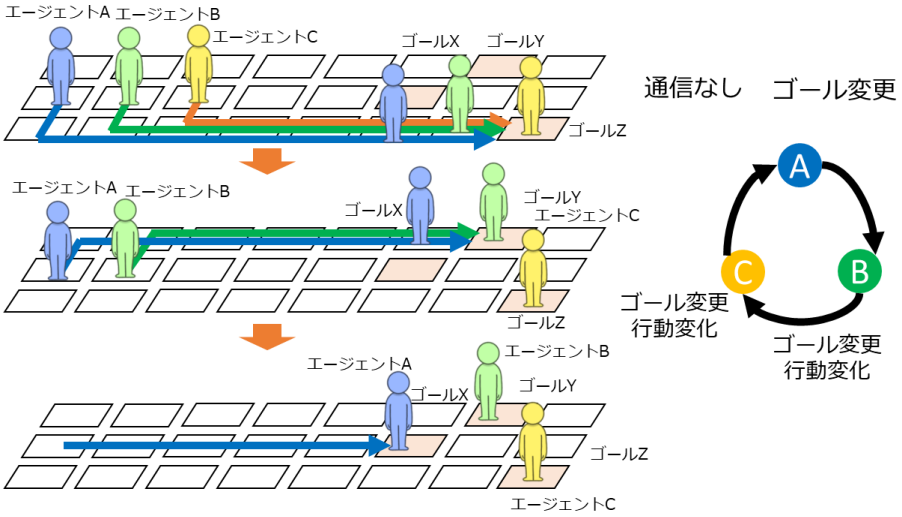


図 2.21　本研究のアプローチ

中で最も遠くを目指し，そのステップ数が最小のエージェントがそのゴールに到達するよう学習し (図の中段)，最後に残ったエージェント A が残るゴール X へ到達するように学習する (図の下段)．これにより全エージェントが別々のゴールに到達し，更にエージェント全体の到達ステップ数を最小にすることができている．以上の通り，本研究においては迷路問題における「エージェント全体が単位時間当たり最大の報酬を獲得する (エージェント全体の到達ステップ数を最小化)」という目的に向けて，各エージェントが最も遠くのゴールを目指していき，それぞれのゴールに対して最短で到達できるエージェントをそこに到達させるアプローチをとることで，エージェントに協調行動を学習させる．

2.6.2　合理的目的設定

本研究において達成する合理的目的設定とは全エージェントがそれぞれ異なるゴールに到達することが理論的に示されている目的の設定である．そのために今回は目的価値というものを設定し，エージェントはその目的価値の最も大きなゴールに到達するように学習することでそれを達成する．また本研究はそれに加えて「最も達成に時間のかかる目的を優先して学習する」提案手法の学習戦略によりエージェント全体のゴール到達ステップ数が最小となることを示すことで，提案手法の合理性を示している．なお，本アプローチはカリキュラム学習と似通っているが，目指すゴール (カリキュラム) を合理的に自動で決めるという点に新規性がある．図 2.23 は本研究における目的選択法とカリキュラム学習の関係性を示している．図においては 3 体エージェントにおける目的選択を示している．提案手法は図の上段から中段，中段から下段へと進んでいく．図のように，まず最初はエージェント 3 体で最も遠くのゴールを目指す (上段)．その後そのゴールに最も早く到達できるエージェントを残し，他のエージェントで次に遠くのゴールを目指す (中段)．こ

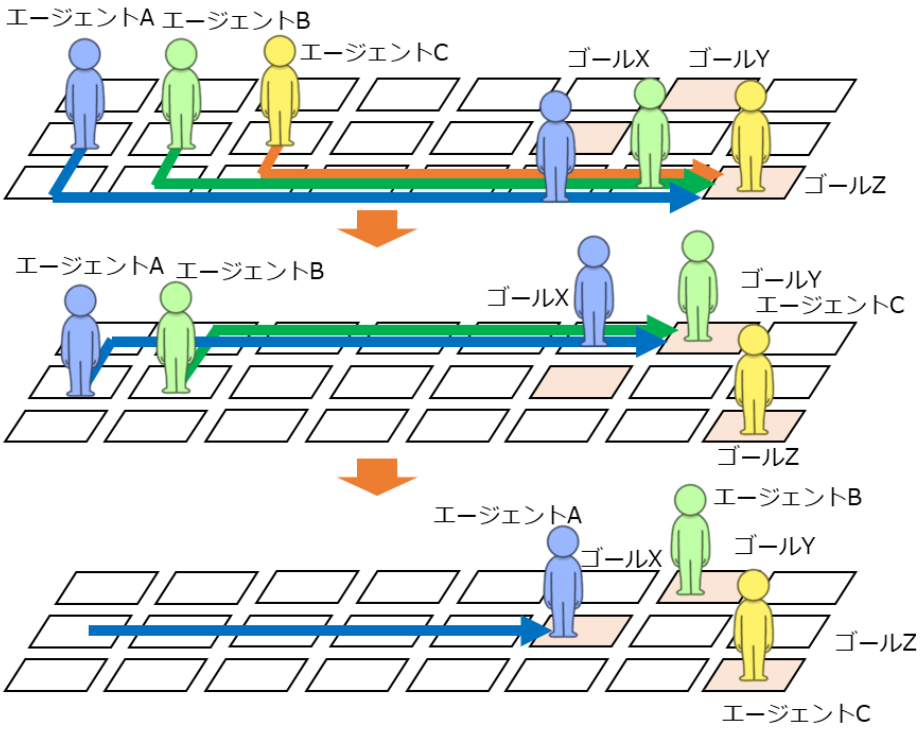


図 2.22　本研究にて達成する協調

れをエージェントがいなくなるまで繰り返す (下段)．以上が本研究における目的設定であり，カリキュラム学習と似通った点である．ただし，カリキュラム学習と異なる点はカリキュラム学習が簡単なタスクから取り組むことで困難なタスクをこなすことに対し，本研究では一番困難なタスクから取り組む点である．また現在のカリキュラム学習における挑戦的課題はどのようにタスクを生み出してエージェントに与えるかという部分であり，近年の研究ではタスクを与えるエージェントを設定して報酬を適切に設定することでカリキュラムの自動生成を実現する手法が報告されている [37]．その点において本研究はエージェントにある特定の規則を持たせることで自動的にカリキュラムを分ける方法であり，カリキュラム学習分野においても新規的手法であるといえる．

2.6.3　内部報酬設計

マルチエージェント強化学習において, 報酬以外の要因から学習する手法は内発的動機づけという．本研究において重要な点は協調行動を自ら探索して，エージェントにその行動を学習させるような動機付けをすることである．内発的動機づけはロボット工学などでも使用されるものであり，エージェントに与えられたタスクを抽象的空間に写像したタスク空間の上でその中を探索するもの [38]，そして各エージェントの報酬を設計するものの 2 つがある．本研究では目的の状態が 1 つ存在しており，通常の学習では局所的

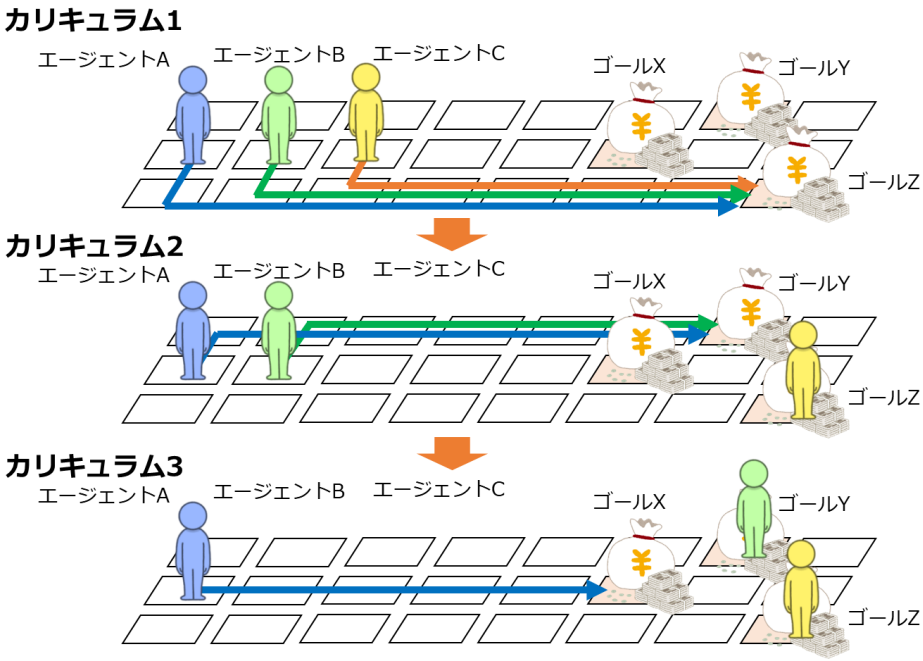


図 2.23　目的設定とカリキュラム学習

な戦略に陥ることを避けるということを目的とするため，報酬設計が適切である．特に報酬設計に関しては近年，Ng らにより提案されている Potential-based reward shaping (PDRS))[39] や逆強化学習 [40] などがよく研究され，その合理性証明も活発に行われている．また昨今の報酬設計は事前知識をエージェントに取り入れ，その知識を活用する．つまり，目的とする協調行動の学習はあるが，そこに至るまでのエージェントの探索と学習を制御できないという立場である．また，GAN[23] を利用して学習空間を探索して，有用な情報を得て，それをエージェントに組み入れるという手法が提案される [41]．しかしながら，学習を続けていても目的とする協調行動が得られないことも考えられる．つまり，内発的動機づけを学習の促進ではなく，直接協調行動の学習を想起させることが今必要となっている．Florensa らは [23] の中でも直接同期付けの重要性を述べている．上記を踏まえ，本研究では内発的動機付けとして，内部報酬という要素を利用して，各エージェントに外部報酬以外の内発的動機づけを起こす手法を提案する．特に本研究では，強化学習が本来持つ要素 (目的達成により報酬を獲得する) を崩すこと無くマルチエージェントシステムを制御しうる報酬設計の方法論を確立することで，従来法では実現することが難しい動的環境などの非定常状態に対して学習の安定性を維持しつつ，適切なシステムの制御を目指す．

第 3 章

モデル化と問題設定

マルチエージェント強化学習を適用して問題解決する上では，対象とする問題を環境としてモデル化することが必要である．ここでは 2 章にて説明した環境の設計について，本論文における設定と使用する迷路問題について説明する．

3.1　迷路問題

本研究では迷路問題を用いて提案手法の有効性を検証する．図 3.1 は 2 体のエージェント，2 個のゴールがある迷路問題の例である．迷路問題において，各マスの座標が環境状態であり，各エージェントの初期状態は “Start A” と “Start B” で示される座標である（以降 “Start A” と “Start B” から行動開始するエージェントをエージェント A，B と呼ぶ）．そして，目的状態は “Goal X” と “Goal Y” の座標（以降目的状態をゴールと呼び，図 3.1 においてはゴール X，ゴール Y と呼ぶ）で示される．エージェントの行動は上下左右 4 種類存在し，それぞれのエージェントは衝突することなく同じ環境状態を観測できるものとする．このとき，全てのマスが状態 S の要素として存在し，任意のエージェント i における上下左右の行動が行動 a_i となる．そしてゴールの状態が S_{goal} の集合に格納されている．ここでは報酬関数に従い，各エージェントが同じゴールへ到達することなくゴールに到達すれば報酬 r を獲得する．本研究では各エージェントは互いに通信することなくエージェント全体として単位時間当たり最大の報酬を獲得することを目指す．ここではエージェント A，B がそれぞれゴール X，Y へ到達することを目指す．なお，本研究では特筆しない限り全ゴールで同値の報酬値を与える．

3.1.1　エージェントの知覚

各エージェントの知覚は自身の居る場所のみである．従来は自身の位置とその周囲のマス 8 個を知覚することができる前提などがあるが，本研究における知覚は自身の位置のみ

					Goal X		
Agent A		Agent B					
							Goal Y

図 3.1　2 体エージェントによる迷路問題

であり，エージェント自身は観測する情報が自分の位置であることはわからない設定を置く．また報酬はエージェントの目的とする状況を観測した時に，エージェント内の信号として持つ．また壁は認識できず，壁の影響で行動による状態遷移ができない．

3.1.2　エージェントの衝突

迷路問題においてエージェントの衝突を考慮するか否かは重要な点である．これはつまり 2 体以上のエージェントが同じマスに到達することができるか否かを設定することである．なお本研究では図 3.2 のようにエージェント同士の衝突を考慮しない．その理由は本研究における目的が衝突回避ではなく，協調的戦略を獲得できるかという部分にあるからである．つまり，ある状況におけるエージェントの局所的な協調行動の獲得を目指すのではなく，迷路問題におけるエージェントの大局的な協調方策の獲得を目指し，各エージェントがどのゴールに到達する行動を獲得するかを見るために衝突は考慮しない．なお，衝突を考慮しないためエージェントが同一のゴールに到達することができる．これらの設定は複数ロボットにマルチエージェント強化学習を適用している例などにおいても，状態をエリア分けして与えていることが一般的で，同一の状態 (マス) であっても衝突なく到達できることは一般的な設定である．

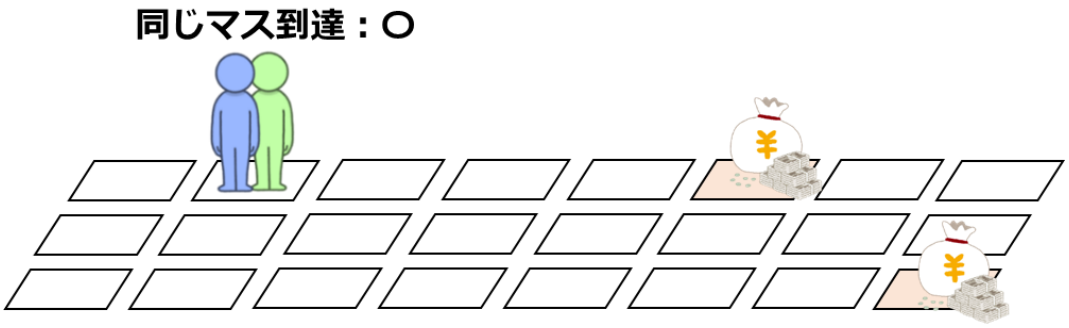


図 3.2　エージェントの衝突

3.1.3　吸収状態

吸収状態とは，図 3.3 のようにゴールへ到達したときに次の学習が始まるまで行動をとることができず留まってしまう状態を指す．迷路問題においてはゴールとなる状態が吸収状態となる．つまり，各エージェントがゴールに到達した際は次の学習が始まるまでそのゴールから移動することはできず，各エージェント 1 つのゴール以外に到達することはできない．なお，前節においてエージェントが同一のゴールに到達することができるとしているが，その際はゴールに到達したエージェントすべてが吸収状態となり，どのエージェントもゴールに到達した時点でそのゴールから動くことはできない．この設定も迷路問題としては一般的であり，物資運搬問題などにおける最短経路を求める際はシミュレーション環境上でこの設定を用いる．

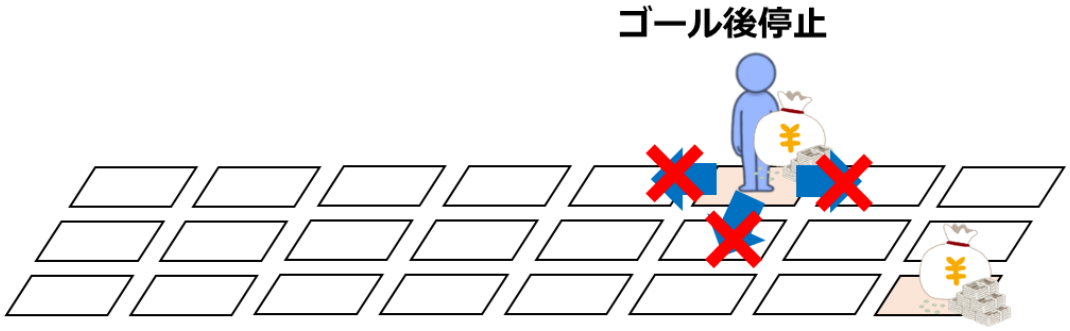


図 3.3　吸収状態

3.1.4　終了条件

迷路問題の学習は全エージェントがゴールに到達するか，ある一定回数ステップが過ぎればリセットされる．このときリセットは，各エージェントがスタート地点に戻されるだけであり，学習結果が失われるわけではない．そして全エージェントがゴールに到達するということは全てのゴールに到達するということではなく，同一のゴールに到達することがあっても，全エージェントがゴールに到達していることが条件となる．またある一定回数のステップは設計者自らが設定することが一般的である．なお，本研究におけるゴールとは式 (2.4) におけるゴール状態集合 S_{goal} を観測すること (迷路問題でいえばゴール位置に到達すること) であり，報酬はその上で式 (2.4) を満たした際に獲得することができる．つまり，各エージェントが同一のゴールに到達したとしても，本研究ではそれらエージェントをゴールしたものとみなすが，報酬の獲得はその中で最初に到達したエージェントのみである．図 3.4 は同一のゴールへ到達したエージェントを示しているが，このように最初に到達したエージェントはゴールへ到達した上で報酬を獲得し，それ以後に到

達したエージェントはゴールへ到達するものの報酬を得ることはない (報酬値 0 の報酬を得る．)

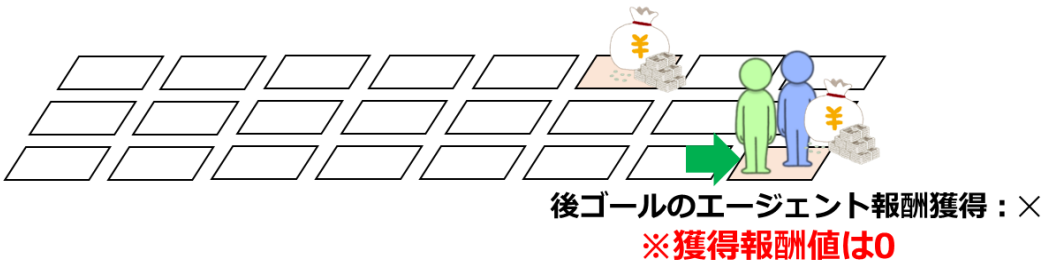


図 3.4　同一ゴール到達時のエージェント

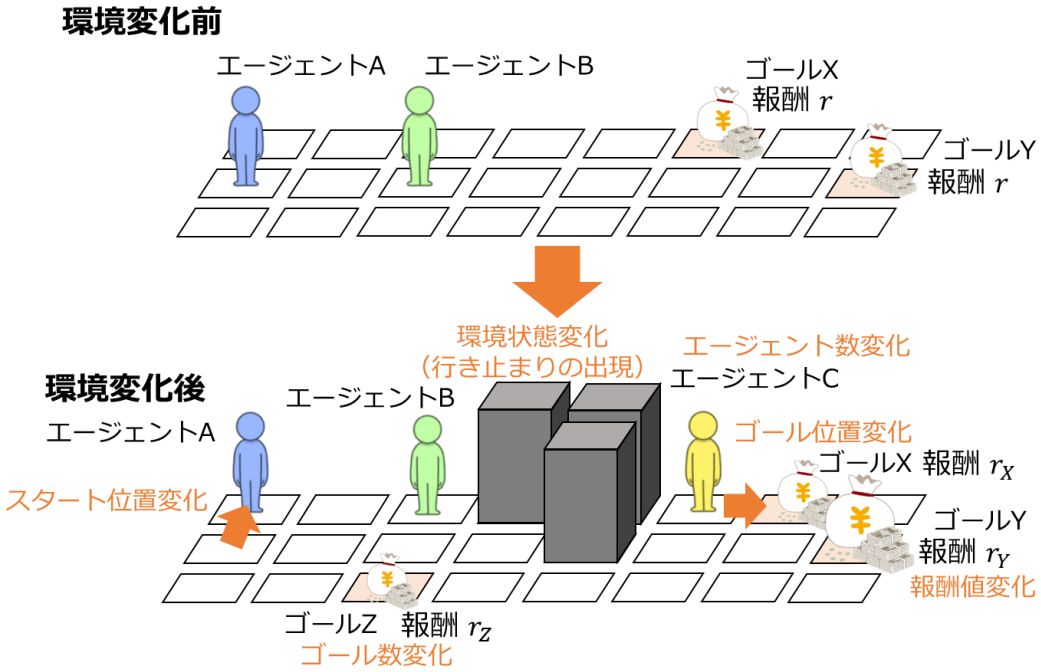


図 3.5　迷路問題における動的変化

3.1.5　動的環境

ここで，迷路問題における動的変化について説明する．図 3.5 はその例である．図の下段にある矢印は時間の流れを表し，そこに書いてある環境変化点において環境が変化することを表す．本研究は動的環境として環境状態，初期状態，目的，エージェント数と目的数の比率，報酬値の 6 個を想定しており，それは次のように変数が変化することを言う．

- 環境状態

環境状態の動的変化とは上記状態 S 及びその遷移関数 τ の規則が変化することをいう．

- 初期状態
  エージェントの初期状態の動的変化は強化学習が始まる際の初期状態 s_{start} が別の値になることをいう．
- 目的
  目的の動的変化は目的状態 $goalstate_g \in S_{goal}$ が別の状態となることをいう．
- 目的数
  目的数の動的変化はゴール数 $|S_{goal}|$ が変化することをいう．
- エージェント数
  エージェント数の動的変化はエージェント数 n が変化することをいう．
- 報酬値
  報酬値の動的変化とは報酬値 $reward$ が変化することをいう．

また本研究では，下記にまとめる通り，上記の動的変化を空間的環境変化，そしてエージェント・ゴール数変化に分類し，2 つの動的変化に対して手法の提案を目指す．2 つに分類している意味は難しさが違うためであり，空間的環境変化が変化後の環境情報を手に入れて学習結果を修正することが必要であることに対し，エージェント・ゴール数変化は環境変化に適応して目的を変更して学習していく必要がある．

1. 空間的環境変化：環境状態，初期状態，目的
2. エージェント・ゴール数変化：目的数，エージェント数，報酬値

図 3.6, 3.7 はそれぞれ空間的環境変化とエージェント・ゴール数変化の例を示している．ここでは矢印を境に環境が変化し，空間的環境変化では，通行できるマスに通行できなくなることや，エージェントのスタート位置やゴールの位置が変化することが起こる．ここでは各エージェントが環境変化前に学習したことから環境変化に合わせてその内容を修正することが求められる．その一方で図 3.7 のようなエージェント・ゴール数変化では文字通りエージェントやゴールの数，そして報酬値が変化するため，空間的環境変化の時と異なり，環境変化にあわせてエージェントの学習方針を変える必要がある．そのためエージェント・ゴール数変化は空間的環境変化よりも難しくなる．

3.2 本研究の問題設定

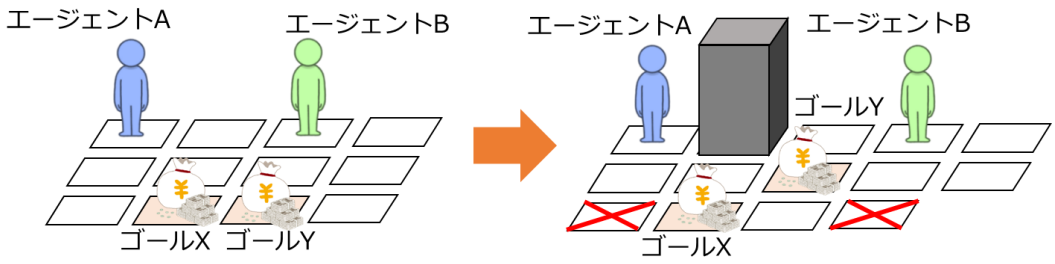


図 3.6　空間的環境変化

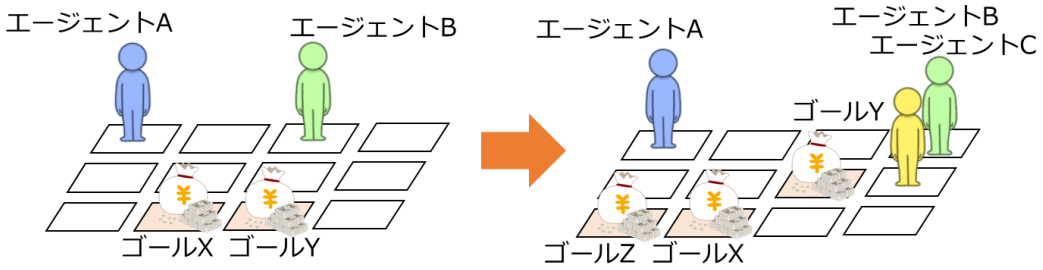


図 3.7　エージェント・ゴール数変化

3.2.1　測りたい協調行動と報酬設定

従来手法は設計者が望む協調行動を獲得するエージェントを設計する．そのため，想定する協調行動を起こして初めて獲得報酬値が最大となる設定を置く．しかし従来，2 体以上のエージェントにおけるゴールを設定することは可能だが，どのゴールの組み合わせが最適であるか判別することはできない．そのため，本研究では下記の点を測ることのできる設定を置くことにする．

- エージェント全体がそれぞれ別々のゴールに到達するときに獲得報酬が最大となる
- エージェント全体が最適なゴールに到達するとき単位時間当たり最大の報酬を獲得する

前節の測りたい協調行動に基づき，本研究における報酬設定は下記の通りとなっている．

- 各ゴールに対してそれぞれ報酬が設定される
- 各ゴールの報酬は 1 個のみ存在し，最初に到達したエージェントのみに与えられる

まず各ゴールに報酬を必ず設定することを行い，そしてそのゴール到達による報酬値は 1 回のみ与えることにする．つまり同じゴールに到達した際の報酬値は最初に到達したエージェントのみに与えられ，それ以外のエージェントは報酬を獲得することはできないとい

う設定である．この設定により，エージェントが同一のゴールに到達した場合は全エージェントの獲得した合計の報酬値は必然的に小さくなり，更に各エージェントが到達したゴールによって一定額の報酬値が決まるため，到達すべきゴールが決まった後はそのゴールに到達する最大のステップ数を小さくするインセンティブが働く．これは上記の測りたい条件を満たす設定であり，本研究ではこの設定を採用することにする．

3.2.2 取り上げる環境の定式化

以下の式は本研究で扱う環境の状態遷移を示す式である．S, A, τ はそれぞれ環境の状態，状態遷移のきっかけとなるエージェントの行動，そして状態遷移関数を示す．環境状態 S はそれぞれエージェントに観測される状態 $state_0, \cdots, state_D$ から成り立ち，行動 A は n 体のエージェントの行動 $a_1, \cdots, a_n$ の積で計算され，環境は状態遷移関数 τ_1 と τ_2 に従って，次の状態へ遷移する．τ_1 において右辺は 1 であり，次状態へ確率 1 で遷移することを示す．また τ_2 において，$goalstate_g$ は任意の目的状態であり，S_{goal} はその集合である（なお g は目的状態の任意の識別番号を示す）．つまり，τ_2 は目的状態からは状態遷移ができないこと (吸収状態) を示す．そして，式 (3.5) は報酬関数であり，各エージェントがこの式に従い報酬を獲得する．式 (3.5) は任意のエージェント i の報酬関数を示す．s_i は任意のエージェント i の現在状態，j は i 以外の任意のエージェントの識別番号，I は全エージェントの識別番号を含む集合である．つまりこの式はエージェント i が目的状態へ遷移し，そのとき同じ状態にいる他エージェントを観測しなかった場合に，報酬値 $reward$ を獲得するという意味である．この式は全エージェント共通で所持している．ここでの目的は，全エージェントがそれぞれ異なる目的状態に到達し，合計の獲得報酬値を単位ステップあたり最大化することである．

$$S := \{state_0, state_1, ..., state_D\} \tag{3.1}$$
$$A = a_1 \times a_2 \times ... \times a_n \tag{3.2}$$
$$\tau_1 : S \times A \times S \to 1 \tag{3.3}$$
$$\tau_2 : goalstate_g \times A \times goalstate_g \to 1,\ where\ goalstate_g \in S_{goal} \tag{3.4}$$
$$r = \{reward | s_i \in S_{goal} \wedge s^j \notin S_{goal} \wedge \forall j \in I \wedge i \neq j\} \tag{3.5}$$

第 4 章

Profit minimizing reinforcement learning (PMRL)

Profit minimizing reinforcement learning (PMRL) は，他エージェント情報の利用無しに協調行動を学習するため，すべてのエージェントが譲歩行動をとることによって，システム全体として協調的に振舞えるように学習する手法である．PMRL は Q 学習に基づき，内部報酬と目的価値の 2 個のメカニズムを導入している．学習する環境の報酬値が直接的に協調行動を学習させるものではないため，目的価値の設定によりエージェント自らが適切な目的を選択し，内部報酬の設定によりその目的達成のために学習する．

4.1　アプローチ

本研究のアプローチは，Q 学習が最短ステップ数で目的達成を目指すこととは逆に，エージェント全体が最短ステップ数が最大となる目的の達成を目指す．そして各目的に対してステップ数を最も短く達成できるエージェントがその目的達成の行動を学習する．そのように各エージェントがお互いに譲りあう行動を学習することで，エージェント全体の全目的達成の時間を最短にする．また迷路問題における本研究のアプローチを図 4.1 に示す．図は 3 段の構成になっており，それぞれ上段中段下段の順番で PMRL の手法が進んでいくことを示している．全てのエージェントが譲歩行動を取ることはつまりすべてのエージェントが自身の最も遠くのゴールを目指して学習する (図の上段)．そしてそのとき，ゴールに最初に到達するエージェントは報酬を獲得するため，そのゴールに到達する行動を学習するが，その他のエージェントはそのゴールに到達する行動を学習できないため，他のゴールの中から最も遠くのゴールを目指す (図の中段)．そしてこれを繰り返すことで，最終的にそれぞれのゴールに到達し，その時間ステップが最短となる (図の下段)．特にここで重要な点はエージェントの学習則を変化させているだけであるため，他エージェントの情報を必要としていないという点と，エージェント C の 1 体の振舞いだけで

エージェント全体の合理性が示せる (この問題における最悪のケースはエージェント A がゴール Z に到達することであるため，エージェント C がゴール Z に到達するということだけでこのケースにおけるエージェント全体が合理的に学習していることを示している) という点である．なお，学習初期はまだ到達していないゴールも存在するため，PMRL では到達しているゴールの中から最大ステップ数となる (最も遠くの) ものを目指して学習する．

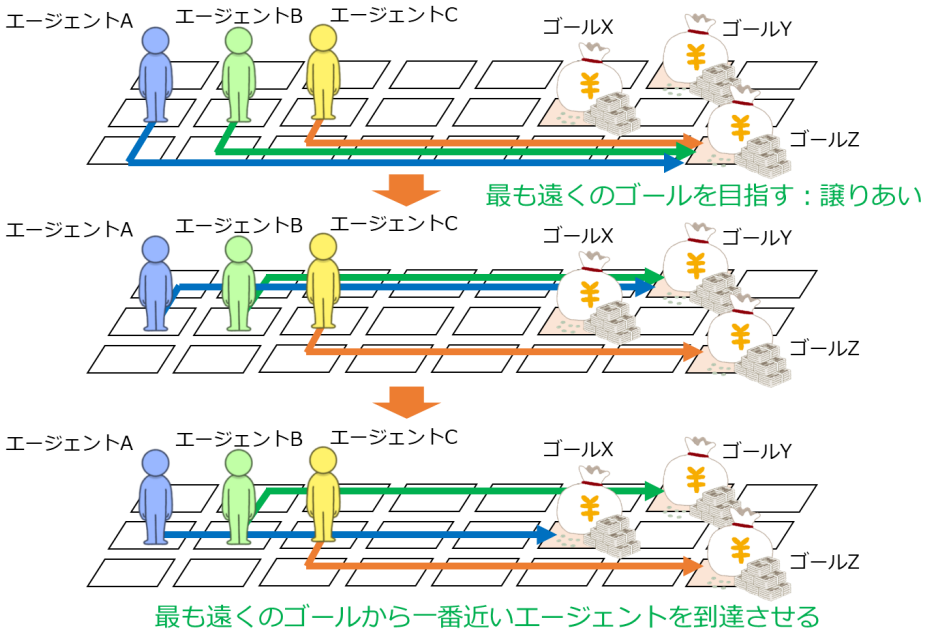


図 4.1　PMRL のアプローチ

4.2　エージェントのアーキテクチャ

図 4.2 は単体のエージェントにおける PMRL の概略図である．図は強化学習における環境とエージェントの相互作用を示しており，網掛け部分はエージェントの内部処理を示している．エージェントは環境から状態 $state(t)$ を観測し，報酬 $reward(t)$ を得て，行動 $a(t)$ を実行する．そして，環境はエージェントに次の状態 $state(t+1)$ と，そのときの報酬 $reward(t+1)$ を返す．このとき $state(t)$ が目的状態集合 S_{goal} の要素であれば，PMRL は報酬 $reward(t)$ をそのまま学習に使用するのではなく，内部報酬 $ir(t)$ へ変換したのちその内部報酬を基に学習する．そしてその内部報酬を設定するために目的価値を設定する．このとき，内部報酬設定は各ステップ毎に行うが，目的価値設定は各学習毎に行う．

PMRL において，エージェントは Q 学習を行う通常の強化学習エージェントとアーキ

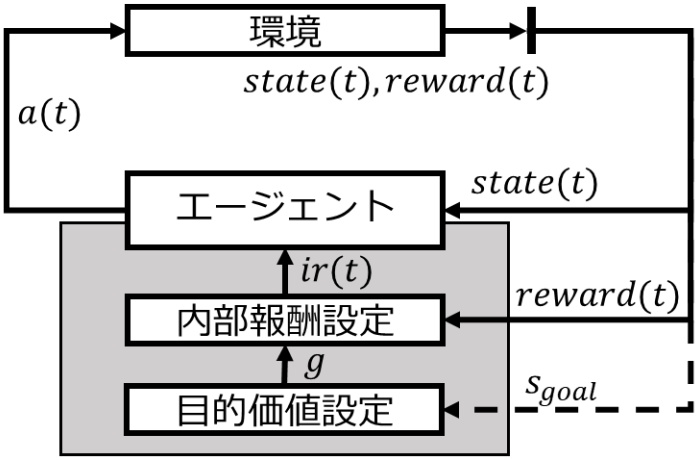


図 4.2　PMRL の概略図

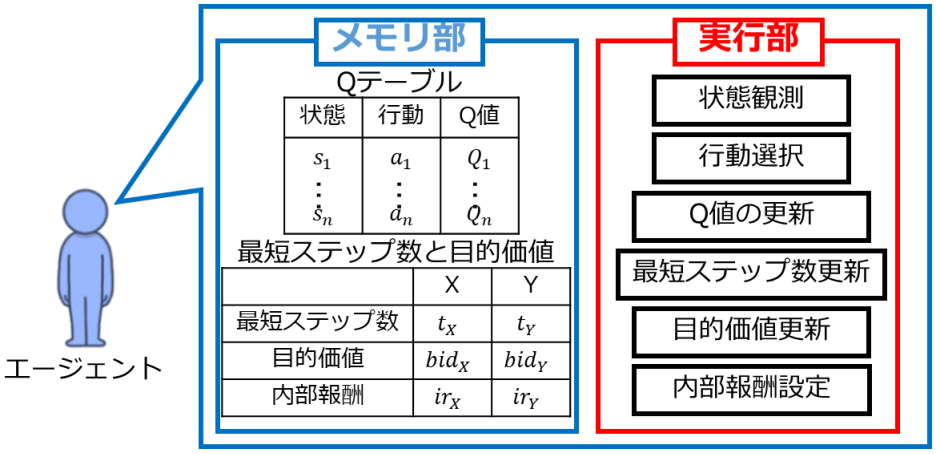


図 4.3　PMRL エージェントのアーキテクチャ

テクチャが少々異なる．図 4.3 は PMRL エージェントのアーキテクチャを示している．エージェント内部は大きくメモリ部と実行部に分かれており，メモリ部には Q 値など学習に必要な変数値が格納されており，実行部は PMRL のアルゴリズムを実行するプロセスが格納されている．具体的にメモリ部は環境の全状態と全行動に対する Q 値を示す Q テーブルと，各目的達成までの最短のステップ数と，協調のために達成すべき目的を示す価値である目的価値，そして内部報酬を持つ (目的価値と内部報酬は以降の節で詳しく説明する)．実行部は Q 学習において実行される状態観測，行動選択，Q 値の更新の他に，最短ステップ数更新，目的価値更新と内部報酬設定のプロセスを持つ．

最短ステップ数更新に関し，エージェントは学習を繰り返す度に，到達ゴールに対するステップ数が今までより最も小さいときにアーキテクチャ内の最短ステップ数の表を更新する．図 4.4 はあるエージェントの最短ステップ数の更新方法を示す．図の下段にある軸はエージェントの学習回数 (以降エピソードと呼ぶ) を示し，エージェントの左上の表は

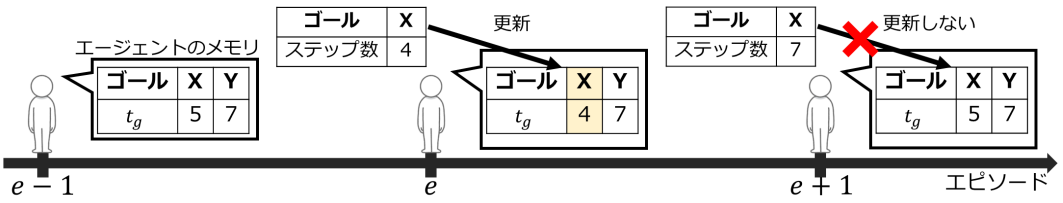


図 4.4　PMRL の最短ステップ数更新

そのときのエピソードにおける到達ゴールと到達ステップ数を示し，吹き出し内はアーキテクチャにおける最短ステップ数を示している．ここでは，エピソード $e-1, e, e+1$ における最短ステップ数の遷移を示している．エピソード e のとき，図ではエージェントはゴール X に 4 ステップで到達する．このときエージェントのメモリ内の最短ステップ数よりも小さくなるため，最短ステップ数を 4 に更新する．更にエピソード $e+1$ のとき，エージェントはゴール X に 7 ステップで到達する．このときエージェントのメモリ内の最短ステップ数よりも大きいため，最短ステップ数は更新しない．以上のようにエピソードが終わるたびにステップ数を比較し，最短ステップ数を更新する．以上が最短ステップ数更新のメカニズムである．

4.3　メカニズム：目的価値と内部報酬

4.3.1　目的価値設定法

目的価値は協調行動を学習するためにそれぞれのゴールがどれだけふさわしいかを示す価値である．目的価値はその値自体に大きな意味はなくその大小関係に意味がある．そして，目的価値の最も大きいゴールが最適なゴールとなる．式 (4.1) は目的価値の更新式を示している．式 (4.1) において，bid_g^i が任意のエージェント i の任意のゴール gs_g における目的価値であり，t_g^i はエージェント i がゴール gs_g へ到達するまでの最短ステップ数を示す．そして n_g^i はエージェント i がゴール gs_g の目的価値を更新した回数である．$\phi(g)$ は式 (4.2) により表現される関数であり，ゴール gs_g へ最初に到達したか否かを判別し，エージェント i が最初に到達 (報酬獲得) していた場合は 1，そうでない場合は 0 を返す．各エージェントは学習毎に目的価値を式 (4.1) に従って更新し，次の学習から目的価値の最も大きなゴール gs_g に到達するように内部報酬を設定する．式 (4.1) は更新を繰り返すと最短ステップ数 t_g^i に収束するため，自身から遠いゴールほど評価される．また関数 $\phi(g)$ の効果により，自身が他のどのエージェントよりも近いゴールの中から，最も遠くのゴールを選択することになる．

$$bid_g^i \leftarrow \frac{n_g^i - 1}{n_g^i} bid_g^i + \frac{t_g^i \phi^i(g)}{n_g^i} \tag{4.1}$$

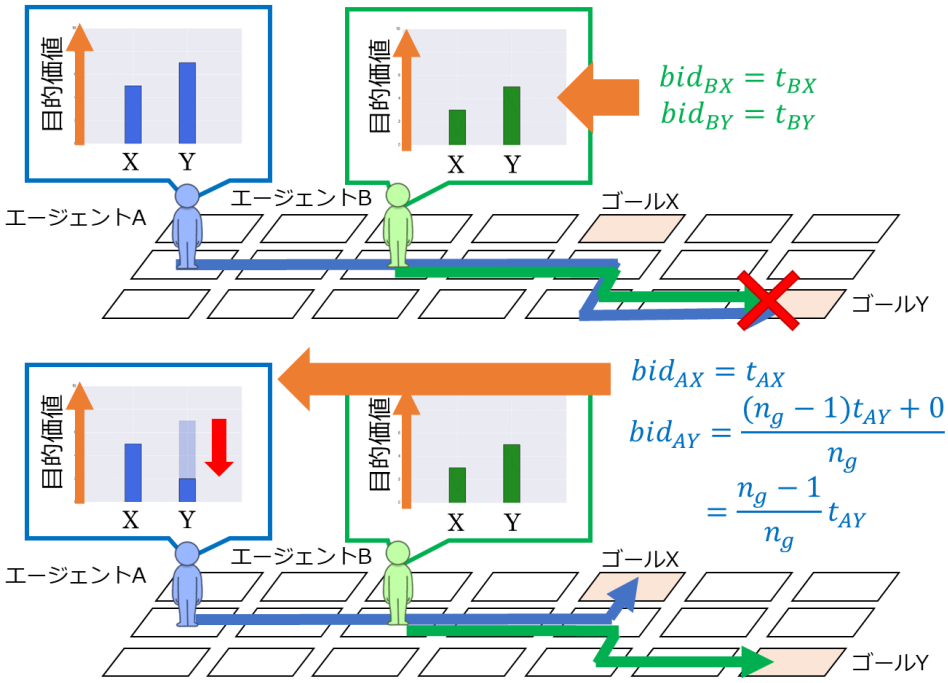


図 4.5　エージェント A と B の目的価値

$$\phi^i(g) = \begin{cases} 1 & if \quad \text{報酬獲得} \\ 0 & otherwize \end{cases} \tag{4.2}$$

ここで迷路問題における目的価値の推移を説明する．図 4.5 はエージェント B と A の目的価値の推移を示している．図のように，エージェント B はゴール X，Y どちらに対してもエージェント A よりも先に到達することができるため，目的価値は最短ステップ数の値に収束する．一方で，エージェント A においては，ゴール X，Y どちらに対してもエージェント B の方が先に到達する．しかしながら，エージェント B は目的価値からゴール Y に到達する行動を学習するため，ゴール X の目的価値は推定可能となる．つまり，エージェント A にとってゴール X の目的価値は最短ステップ数に収束し，ゴール Y の目的価値はエージェント B の影響で 0 に近づいていく．結果としてエージェント A はゴール X に到達する行動を学習し，エージェント B はゴール Y に到達する行動を学習することにより，適切な目的達成の行動が学習可能となる．

4.3.2　内部報酬設定法

PMRL では報酬を獲得した際に，その報酬で学習するのではなく内部報酬から Q 値を学習する．内部報酬とは，エージェントが内部的に持つ報酬のことであり，環境から得られる報酬が適切でない場合に，その適切な学習のためにエージェントが設計する報酬である．図 4.6 は，本研究における内部報酬を示している．エージェント B の吹き出しはエー

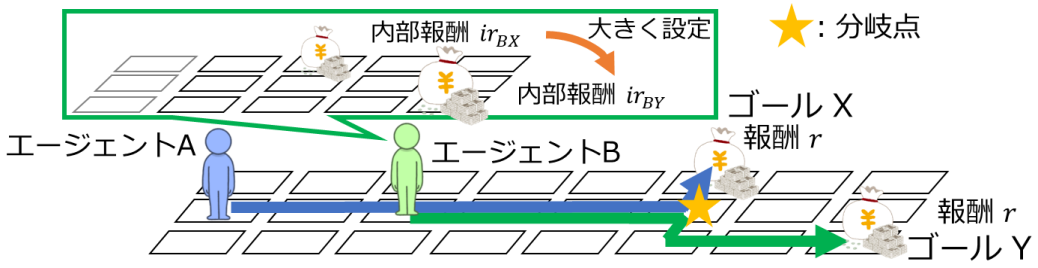


図 4.6　内部報酬

ジェント B のメモリ内部を示しており，メモリ内の内部報酬を変更させ，ゴール Y に到達可能な Q 値を獲得するために，ゴール Y に対する内部報酬値をゴール X のものよりも大きく設定している．以上のように，本研究では，エージェントはそれぞれのゴールに対して内部報酬を持ち，ゴールに到達した際に獲得した報酬値に基づき内部報酬値を設定する．

内部報酬は，獲得した報酬に基づき式 (4.3) に従い計算する．式 (4.3) において与える報酬は 3 種類存在する．まず目的価値により設定したゴール g_{sel} と今現在到達したゴール g が等しいとき，そのゴールに到達する行動を学習するために上段の通り内部報酬を更新する．次に到達したゴールが向かうべきゴール g_{sel} と異なる場合，獲得した報酬値を変更することなく内部報酬として学習する．最後にゴールに到達していない場合，内部報酬値は 0 を設定します．そしてエージェントは各状態 s に合わせてこの内部報酬値を使って学習する．

式 (4.3) において，$ir(g)$ はゴール g へ到達する行動を学習するための内部報酬値を示している．また，γ は Q 学習における割引率を示し，g_o は g 以外の任意のゴールを示す．そして r_{g_o} はゴール g_o における報酬を示し，t_g はゴール g に到達するまでの最短のステップ数を示している．この式のように内部報酬を設定することで，エージェントはゴール g へ到達できるように学習できる．その理由は，Q 学習が距離に応じて Q 値を γ だけ等しく割り引くため，報酬値の大きさとゴールまでの距離によってエージェントがどのゴールへ到達するかが分かるためである．つまり，初期状態での目的のゴールへ向かう行動の Q 値が最大となるように報酬を設定すればその状態へ到達することが可能となる．

$$ir(g) = \begin{cases} \max\limits_{g_o \in G, g_o \neq g} r_{g_o} \gamma^{t_{g_o} - t_g} + delta & if\ getting\ the\ reward \wedge g_{sel} = g \\ r_g & if\ getting\ the\ reward \wedge g_{sel} \neq g \\ 0 & otherwise \end{cases} \tag{4.3}$$

4.4 PMRL の合理性

ここでは PMRL エージェントが学習により，協調行動を獲得し，全エージェントの単位時間当たりの獲得報酬値を最大化させることを示す．そのために，内部報酬設定による Q 値，そして目的価値の収束性を証明し，そのときの内部報酬値により推定される Q 値の収束値によりエージェントが適切なゴールに到達すること (合理性) を示す．なお，証明において想定する条件は下記の通りである．

- 各エージェントのゴール到達までのステップ数はそれぞれ異なる値を示す
- 学習回数は無限大
- 各エージェントの衝突はなし

4.4.1 内部報酬の収束性証明

ここでは，報酬設定により目的のゴールに近づく行動の Q 値がその他の行動のどの Q 値よりも大きくなることを示す．任意の状態からゴール g へ向かう行動の Q 値は式 (4.4) に従い収束する．このとき，t_{any} はスタート地点から任意の状態へ到達するまでの最短ステップ数を示す．

$$Q_g^*(s,a) = ir_{ig} + \gamma \max_{a'} Q(s',a') = ir_{ig}\gamma^{t_{any}-t_g} \tag{4.4}$$

エージェントをゴール g に到達させることを考えると，式 (4.4) に示す Q 値の収束値はゴール g に向かう行動のものが最大であればよい．つまり，式 (4.5) の条件に従えばよい．また式 (4.5) を式 (4.4) に従って展開すると，式 (4.6) が成り立つ．そしてこの式を移項すると，式 (4.8) が成り立つ．この式に従って内部報酬を設定することで，エージェントをゴール g に到達させることができる．なお，この条件ではステップ数 t_{ang} が消えるため，どの状態においてもゴール g に到達する内部報酬の設定が可能となる．このとき式 (4.3) は δ によってこの条件を満たす．

$$Q_g^*(s,a) > \max_{og} Q_{og}^*(s,a) \tag{4.5}$$

$$ir_{ig}\gamma^{t_{any}-t_g} > \max_{og} ir_{iog}\gamma^{t_{any}-t_{og}} \tag{4.6}$$

$$ir_{ig}\gamma^{-t_g} > \max_{og} ir_{iog}\gamma^{-t_{og}} \tag{4.7}$$

$$ir_{ig} > \max_{og} ir_{iog}\gamma^{t_g-t_{og}} \tag{4.8}$$

4.4.2　目的価値の収束性証明

ここでは，各ゴールに対する目的価値が最短ステップ数に収束することを示す．まず目的価値の更新式を展開すると，下記のようになる．

$$bid_g^i \leftarrow \frac{(n_g-1)bid_g^i + ft(n_g)}{n_g} \tag{4.9}$$

$$\leftarrow \frac{(n_g-1)}{n_g}\left(\frac{n_g-2}{n_g-1}bid_g^i + \frac{ft(n_g-1)}{n_g-1}\right) + \frac{ft(n_g)}{n_g} \tag{4.10}$$

$$= \frac{\sum_{m=1}^{n_g} ft(m)}{n_g} \tag{4.11}$$

$ft(m)$ が一定の値 t_g を返すとき (エージェントがそのゴールに最も早く到達する確率が高いとき)，上記の式 (4.11) は式 (4.12) に従い，定数値 t_g を示す．t_g はゴール g へ到達するための最短ステップ数であるため，目的価値推定の式は最短ステップ数に収束する．

$$\frac{\sum_{m=1}^{n_g} ft(m)}{n_g} = \frac{n_g t_g}{n_g} = t_g \tag{4.12}$$

4.4.3　2 体のエージェントの到達ゴールの合理性証明

式 (4.11) から関数 $ft(m)$ が必ず最短ステップ数 t_g を返すことが無いとすれば，下記の通り目的価値 bid_g^i が展開される．このとき $p_{i,g}$ はエージェント i がゴール g へ到達して報酬を獲得する ($ft(m)$ が最短ステップ数を返す) 割合を示している．

$$bid_g^i \leftarrow \frac{(n_g-1)bid_g^i + ft(n_g)}{n_g} = \frac{\sum_{m=1}^{n_g} ft(m)}{n_g} = p_{i,g}t_{i,g} \tag{4.13}$$

このとき PMRL では，すべてのエージェントにとって最も遠くのゴールが g であれば，割合 $p_{i,g}$ は下記の性質を持つ．$p_{i,g}$ はゴール g にエージェントが到達したケースの中で最も早くに到達した割合である．裏を返せば最も早くに到達できなかった場合は他のエージェントが最も早くに到達していることになるため，全エージェントが同じゴールを目指しているのであればその割合の合計は近似的に 1 になる．

$$\sum_i p_{i,g} \approx 1 \tag{4.14}$$

ここで，もしゴール g に対して最も近いエージェント i がゴール g への最短経路を学習しているとすると，式 (4.14) 下記の式が成り立つ．これはエージェント i がゴール g を目指していれば，ゴール g に対する割合 $p_{i,g}$ がほぼ 1 になるため，それ以外のエージェン

ト $-i$ の割合 $p_{-i,g}$ はほぼ 0 になる．結果としてエージェント i のゴール g に対する目的価値は最短ステップ数となり，その他のエージェントの目的価値は 0 となる．

$$bid_g^i = p_{i,g} t_{i,g} \approx t_{i,g} \tag{4.15}$$

$$bid_g^{-i} = p_{-i,g} t_{-i,g} \approx 0 \tag{4.16}$$

また，もしエージェント i がゴール g に到達したエピソード以外の時他エージェントがそのゴールへ到達するとき，式 (4.14) の性質が成り立つことはないが，そのとき割合 $p_{i,g}$ に限らず他エージェントの持つ割合 $p_{-i,g}$ も 1 となる．すると結果としてどのエージェントもゴール g を目指し，上記の示す状況と等しくなるため，エージェント i のゴール g に対する目的価値は最短ステップ数となり，その他のエージェントの目的価値は 0 となる．

そしてもしエージェント i とそれ以外のエージェント $-i$ の最も遠くのゴールが g であり，その到達ステップ数が等しい場合，上記の記述より両エージェントはゴール g を目指し，結果として式 (4.14) を満たす．このとき割合 $p_{i,g}, p_{-i,g}$ と最短ステップ数割合 $t_{i,g}, t_{-i,g}$ がそれぞれ同値になるため，エージェントは別々のゴールに到達することができない．しかしながら，本提案手法においてエージェントは完全に同時に行動することはなく，予め決められたエージェントから動き出す (エージェント番号があればその数値の小さい順など)．そのため，このときステップ数が等しいといえども割合は異なる．結果としてそれぞれの目的価値は式 (4.15) および (4.16) と同等の値を示し，エージェント i のゴール g に対する目的価値は最短ステップ数となり，その他のエージェントの目的価値は 0 となる．

以上から，エージェントは PMRL の目的価値更新を行うことで，どの状況においてもそれぞれのエージェントが別々のゴールに到達することができる．

4.4.4 2 体エージェントの到達ゴールの最適性証明

2 個のゴールと 2 体エージェントの迷路問題において，エージェントのゴール到達は下記 2 通りのみ存在する．迷路問題ではスタート地点，ゴール地点，報酬値等様々な要素があり，その違いにより迷路形状が決定するが，その全ては下記 2 ケースに帰着できる．

ケース 1　強化学習により競合が起こらない (協調する必要がない) 場合

このケースでは，全てのエージェントが強化学習を行うことで，自身に最も近くのゴールに競合なく到達可能であるため，特に新たな手法を必要としないケースである．

ケース 2　強化学習により競合が起こる (協調する必要がある) 場合

このケースでは，全てのエージェントが強化学習を行うことで，エージェントが同じゴールへ到達する．

ケース 1 はそのまま強化学習で解くことができるため，ケース 2 を提案法で解決可能であることを示し，2 体エージェントの協調を必要とする問題全てを提案手法で解決可能であることを示す．下記の表 4.1 は各エージェントのそれぞれのゴール到達までの最短ステップ数を示している．

表 4.1　各エージェントがゴールに到達するまでの最短ステップ数

	Goal X	Goal Y
Agent A	t_{AX}	t_{AY}
Agent B	t_{BX}	t_{BY}

ケース 2 ではつまり，全エージェントがゴール X またはゴール Y に到達する．ここでは全エージェントがゴール X に到達することを想定する．このとき，Q 学習はより近い (到達までのステップ数がより小さい) ゴールに到達する性質があるため，下記の不等式が成り立つ．

$$t_{AX} < t_{AY} \tag{4.17}$$
$$t_{BX} < t_{BY} \tag{4.18}$$

また，$t_{AY} < t_{BY}$ であれば，PMRL で遠くのゴールへ到達する内部報酬を設定した時，エージェント A が最初にそのゴールへ到達する可能性が非常に高くなるため，エージェント B はゴール Y に到達しても報酬獲得ができなくなる．このとき，目的価値の収束値の証明により，目的価値は下記の値に収束する．そして式 (4.17) から，$t_{AX} < t_{AY}$ であるため，エージェント A はゴール Y を目指し，エージェント B はゴール X を目指す．加えて，$t_{AY} < t_{BY}$ であるため，別のゴール到達の組み合わせ (エージェント A がゴール X を目指し，エージェント B がゴール Y を目指すとき) よりも到達ステップ数が小さい．そしてこのときゴール到達の組み合わせは 2 通りしかないため，このとき PMRL により全エージェントが最短ステップ数でゴールに到達することが分かる．

一方で $t_{AY} > t_{BY}$ であっても，エージェント A と B のラベルを入れ替えれば上記と同様の状況と分かる．また，全エージェントがゴール Y に到達することを想定してもゴール Y と X のラベルを入れ替えれば上記と同様の状況となる．以上からケース 2 のすべての環境において，PMRL により全エージェントが最短ステップ数でゴールに到達することが分かる．

$$bid_{AX} = t_{AX} \tag{4.19}$$
$$bid_{AY} = t_{AY} \tag{4.20}$$
$$bid_{BX} = t_{BX} \tag{4.21}$$
$$bid_{BY} = 0 \tag{4.22}$$

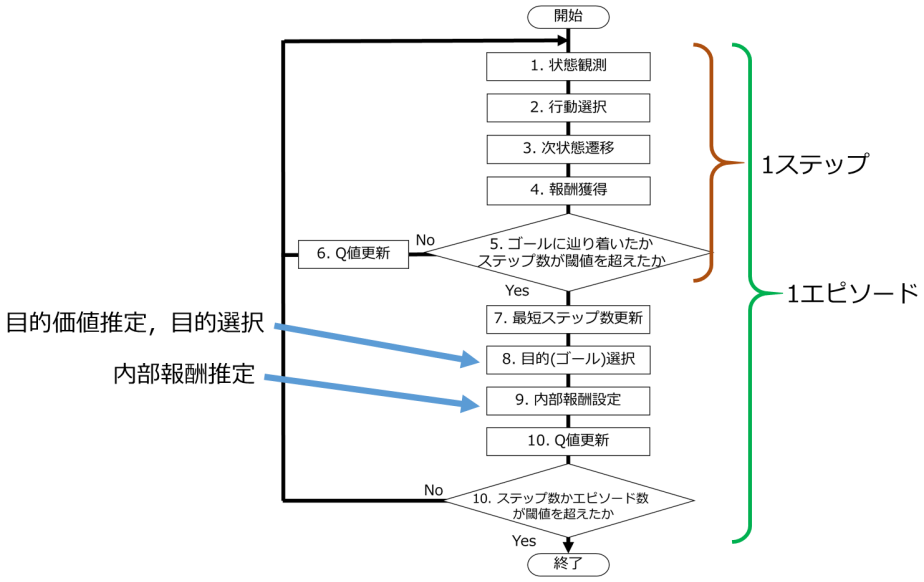


図 4.7　PMRL のアルゴリズムフロー

4.5　アルゴリズム

PMRL のフローチャートを図 4.7 に示す．PMRL では，エージェントがまず Q 学習と同様に状態を観測し，行動を選択することで次状態に遷移して，報酬を獲得する (プロセス 1-4)．その後の処理が PMRL 独自のものとなっている．PMRL はエージェントはゴールに到達したか否かによって処理が異なり，ゴールに到達していないのであれば獲得した報酬に基づき Q 値を更新してまた次のステップを繰り返す (プロセス 9)．一方でゴールに到達しているのであれば，そのゴール到達までのステップ数が今までそのゴールへ到達したステップ数の中で最小であれば更新し，その後更新された最短ステップ数から目的を選択し，目的価値を更新する (プロセス 6, 7)．そして選択した目的からその選択したゴールに到達できるように更新された最短ステップ数から内部報酬を設定する (プロセス 8)．以上をステップ数やエピソード数が閾値を超えるまで繰り返して学習する (プロセス 10)．以上が PMRL のアルゴリズムフローの全貌である．

PMRL のアルゴリズムを Algorithm2 に示す．Algorithm2 において，$\boldsymbol{t}$ は各エージェントがそれぞれのゴール到達のために必要な最短ステップ数を格納する配列である．また，$\boldsymbol{bid}$ は目的価値を格納する配列で，$\boldsymbol{ir}$ は内部報酬を格納する配列である．初期状態は各学習毎同じ状態 s_{start} を設定し，ゴールは集合 S_{goal} に格納する．まずはじめにエージェントは初期状態 s_{start} を観測し（5 行目），行動を選択する（7 行目）．その後行動 a を実行し，報酬 r を獲得して，次状態 ns へ遷移する（8 行目）．そして報酬から内部報酬 ir を計算し，内部報酬に基づき Q 値を更新する（10 行目）．このとき次状態 ns がゴー

ルであった場合学習を終了し（12，13，14 行目），そうでなかった場合はステップを繰り返す．最終的に一定のステップ $MaxStep$ 経った後にゴールの状態を観測できなかった時は学習を終了する．学習終了後，もしゴールの状態で，そのゴールへの到達ステップ数が配列 $\boldsymbol{t}$ へ保存された値よりも小さかった場合，最短ステップ数の配列 $\boldsymbol{t}$ を更新する（16，17，18 行目）．その後は，目的価値を設定し，内部報酬へ設定するために最も適したゴールを決定する（19 行目）．以上が PMRL のアルゴリズムである．

続いて Algorithm2 の関数 BID のアルゴリズムを Algorithm3 に示す．関数 BID はエージェントの目的価値配列 $\boldsymbol{bid}$，エージェントを識別する番号 i を入力として，マルチエージェントシステムの協調行動学習のためにエージェント i が到達すべきゴールを $selection$ として返す．関数はまず獲得報酬に従って関数 $\phi^i(selection)$ の値を更新する．具体的には報酬獲得が可能なゴールであれば 1 を格納し，そうでない場合は 0 を格納する（1-5 行目）．そして関数 $\phi^i(selection)$ を使い，目的価値を更新する（6 行目）．その後，配列 $\boldsymbol{bid}$ の値が最大となる引数 g を $selection$ に格納し（7，8，9 行目），一定の確率で $selection$ をランダムなゴールに置き換える（10 行目）．最後に選択 $selection$ を返す（12 行目）．以上が関数 BID のアルゴリズムである．

Algorithm 2 PMRL

1. $Q(s,a)$ を初期化，$\forall s \in S, \forall a \in A$
2. 最短ステップ数配列 $\boldsymbol{t}$ を $MadxStep$ で，目的価値 $\boldsymbol{bid}$ と内部報酬 $\boldsymbol{ir}$ を 0 で初期化
3. 初期状態 s_{start} とゴール集合 S_{goal} の設定
4. **for** $iteration = 1$ to $MaxIteration$ **do**
5. $\quad s = s_{start}$
6. $\quad$ **for** $step = 1$ to $MaxStep$ **do**
7. $\quad\quad$ 行動 $a = ActionSelect(Q, s)$
8. $\quad\quad$ 行動 a を実行，報酬 r を獲得，次状態 ns へ遷移
9. $\quad\quad$ 報酬から $selection$ へ到達するように内部報酬 ir を計算
10. $\quad\quad$ $Q(s,a) = Q(s,a) + \alpha\left[ir + \gamma \max_{na} Q(ns,na) - Q(s,a)\right]$
11. $\quad\quad$ $step = step + 1$
12. $\quad\quad$ **if** $ns \in S_{goal}$ **then**
13. $\quad\quad\quad$ break
14. $\quad\quad$ **end if**
15. $\quad$ **end for**
16. $\quad$ **if** $ns \in S_{goal} \wedge step < \boldsymbol{t}[g]$ **then**
17. $\quad\quad$ $\boldsymbol{t}[g] = step$
18. $\quad$ **end if**
19. $\quad$ $selection = BID(\boldsymbol{bid}, i)$
20. $\quad$ $iteration = iteration + 1$
21. **end for**

Algorithm 3 PMRL の目的価値設定関数 BID

Input: 自身の目的価値 $\boldsymbol{bid}$, エージェント識別番号 i

Output: 適切なゴール $selection$

if 報酬獲得可能 **then**
  $\phi^i(selection) = 1$
else
  $\phi^i(selection) = 0$
end if
$\boldsymbol{bid}[selection] = \frac{\boldsymbol{n}[selection]-1}{\boldsymbol{n}[selection]}\boldsymbol{bid}[selection] + \frac{\boldsymbol{t}[selection]\phi^i(selection)}{\boldsymbol{n}[selection]}$
if $\boldsymbol{bid}[g]$ が最大 **then**
  $selection = g$
end if
一定確率で $selection$ をランダム選択
$\boldsymbol{n}[selection]++$
return $selection$

4.6　実験

4.6.1　実験内容

PMRL の有効性を検証するため，本研究では 2 体・5 体エージェントにおける迷路問題の 2 つのケースにおいて PMRL と Q 学習および PS の性能を比較する．具体的にはケース 1(2 体エージェント) では，図 4.8-4.12 の 5 種類の迷路問題（以降それぞれの迷路を上から迷路 1, 2, 3, 4, 5 と呼ぶ）を採用する．これらの迷路は，スタートとゴールの距離と各ゴールに向かう方向の違いにより難易度が異なる．例えば，図 4.12 はスタート地点とゴール地点が近くにあるため簡単であるが，図 4.8 はスタート地点とゴール地点が遠くにあるため難しい．また，図 4.8 の迷路はゴールへ向かう方向が等しいため簡単であるが，図 4.11 の迷路は各ゴールに向かう行動の方向が異なるため難しい．各図の各マスが状態を表し，A，B および X，Y と表示されたマスがそれぞれのエージェントの初期状態とそれぞれのゴールを表す．ケース 2 では，スタート，ゴールの位置関係がランダムに決定された 10 種類の迷路問題を採用し，任意の迷路に対する性能を評価する．

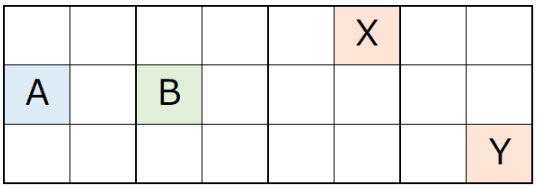


図 4.8　迷路 1

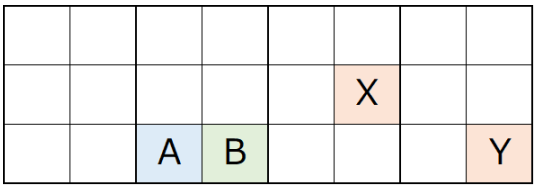


図 4.9　迷路 2

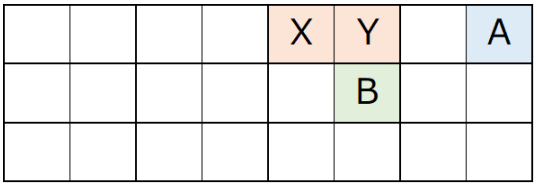


図 4.10　迷路 3

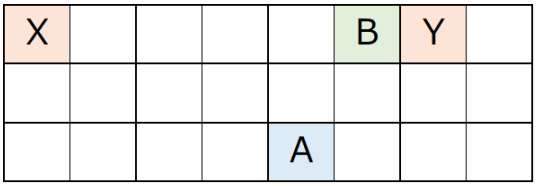


図 4.11　迷路 4

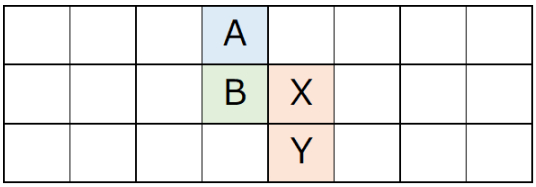


図 4.12　迷路 5

4.6.2　評価基準とパラメータ設定

本実験は各迷路で乱数のシードを変更した 30 試行 (ケース 1 : 2 体エージェント) および 10 試行 (ケース 2 : 2 体，5 体エージェント) で実験を行い，すべてのエージェントが最短ステップ数でゴールに到達した割合とすべてのエージェントがゴールに到達したステップ数を評価する．なお学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じように学習反復を行い，その時の到達に要したステップ数を評価する．なお，ケース 2 においては，Q 学習と比較していないが，これは迷路問題で得られる外部報酬のみでは絶対に協調行動を学習できないことがケース 1 により判明しているためである．

表 4.2 に実験パラメータを示す．このとき学習エピソード数が 2 体エージェントの時 50000 回であり 5 体の時 500000 である (1 行目)，1 学習における最大のステップ数を 100 に設定する (2 行目)．また学習において初期 Q 値は 0(3 行目)，学習のパラメータをは学習率 α が 0.1(4 行目)，割引率 γ は 0.9 に設定する (PS の割引率 S もこれに相当)(5 行目)．報酬値は 10 に設定し (6 行目)，PMRL の内部報酬設定のための定数 δ は 10 とする (7 行目)．PS は本論文にて紹介している報酬関数を用いる．

表 4.2　PMRL の実験パラメータ

	PMRL	Q 学習	PS
エピソード数	50000(2 体)，500000(5 体)		
最大ステップ数	100		
初期 Q 値	0		
学習率 α	0.1		
割引率 γ	0.9		
報酬値	10		
定数 δ	10	なし	

4.6.3　実験結果

・ケース 1

実験結果を図 4.13 および 4.14 に示す．縦軸が最短ステップ数でゴールに到達した割合，横軸がエピソード数である．また青，橙，緑，赤，紫の線がそれぞれの迷路における結果を示している．これを見ると分かる通り，従来の Q 学習では最短ステップ数で全ゴールへ到達する試行が全く存在しないことに対し，PMRL ではすべての迷路において

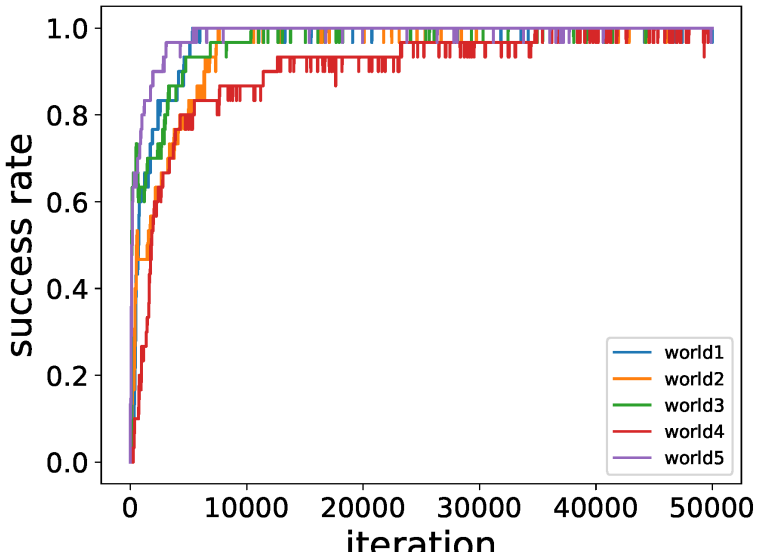


図 4.13　PMRL の結果

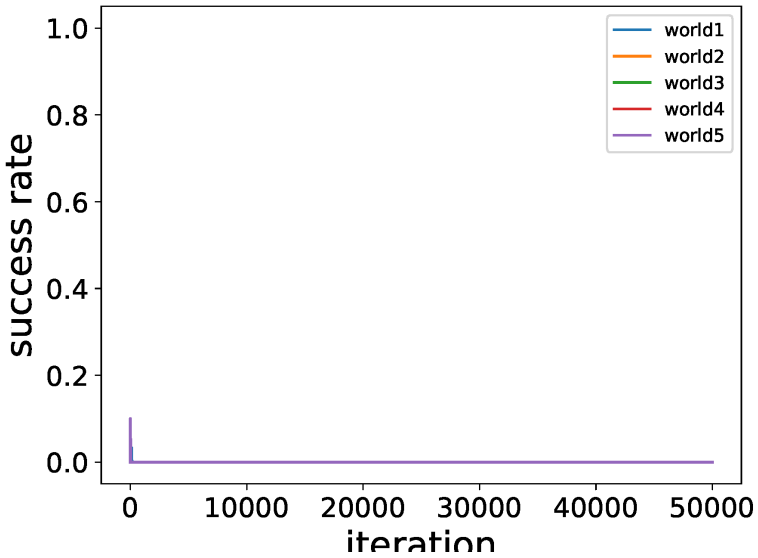


図 4.14　Q 学習の結果

確率 1 で最短ステップ数でゴールに到達することができている．

また，図 4.15-4.19 はそれぞれの迷路における PMRL と Q 学習と PS での到達ステップ数の違いを示している．縦軸が最短ステップ数でゴールに到達した割合，横軸がエピソード数である．また青，橙，緑，赤の線がそれぞれの迷路における最短のステップ数，Q 学習の結果，PS の結果，そして PMRL の結果を示している．このとき仮にエージェントが同じゴールに到達した場合，最大ステップ数を示す (ここでは 100)．この図を見る

と分かる通り，Q 学習は同じゴールに到達する行動を学習した結果ステップ数が 100 であるのに対して，PMRL は全ての迷路において最短ステップ数に収束しており，PMRL によって適切な行動が学習できていることがわかる．また PS もある程度ステップ数を小さくできているが値のぶれが大きく，特に迷路 4 では学習できていないことからも，PMRL よりも性能が高いとは言えない．

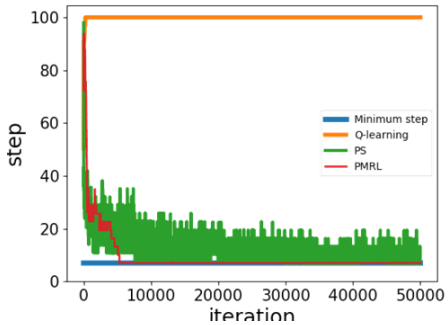


図 4.15　迷路 1 のステップ数の比較結果

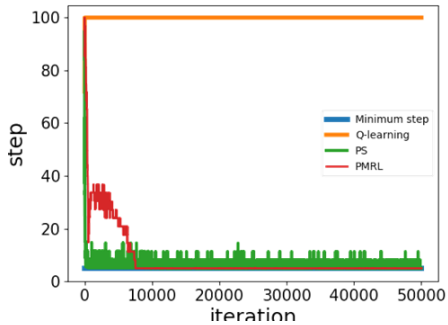


図 4.16　迷路 2 のステップ数の比較結果

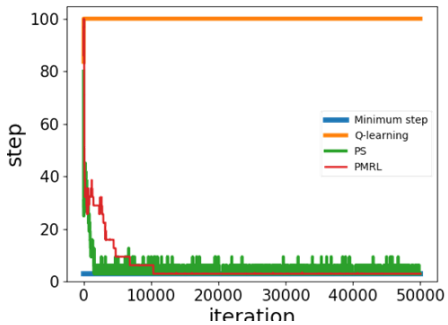


図 4.17　迷路 3 のステップ数の比較結果

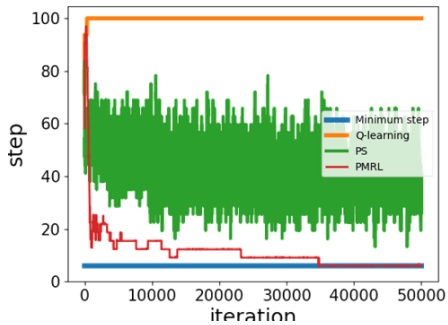


図 4.18　迷路 4 のステップ数の比較結果

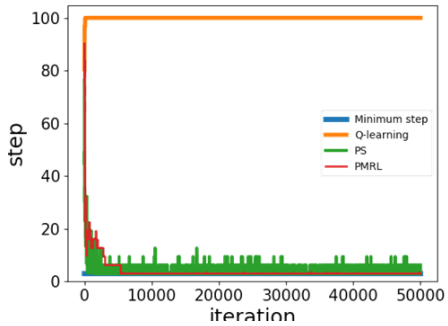


図 4.19　迷路 5 のステップ数の比較結果

・ケース 2

表 4.3 は各 10 種類の迷路における PMRL と PS の結果の比較である．表の上段は迷路 1-5, 下段は 6-10 までの結果であり，各段の上側が 2 体エージェントの結果であり，下側

表 4.3　各迷路における到達ステップ数の中央値

		maze 1	maze 2	maze 3	maze 4	maze 5
2 体	PMRL	**7**	**5**	**3**	**7**	7
	PS	**7**	**5**	**3**	53.5	**5**
5 体	PMRL	**100(9)**	**100(9)**	**100(15)**	**100(7)**	**100(13)**
	PS	100	100	100	100	100

		maze 6	maze 7	maze 8	maze 9	maze 10
2 体	PMRL	**3**	**4**	**8**	**6**	**7**
	PS	**3**	**4**	100	8	**7**
5 体	PMRL	**100(15)**	**100(9)**	**100(7)**	**100(10)**	**100(16)**
	PS	100	100	100	100	100

が 5 体エージェントの結果である．そして，各マスには最終エピソードのステップ数の全シードを通した中央値が格納されている．括弧付きの数値は最小値であり，太字は結果が上回っているものを示している．これを見ると，2 体エージェントの迷路 5 を除いて従来の PS よりも PMRL が勝っていることがわかる．特に PS では各エージェントが最適なゴールに到達した場合に報酬値が最大となる設定を置いており，PMRL よりも有利な条件で実験を行っている．その中で PMRL の精度が PS を上回っていることから，PMRL の結果の方がより良いことがわかる．

4.6.4　考察

・2 体エージェント

PMRL により，従来の Q 学習や PS ではできていなかった，全エージェントを最短ステップ数で全ゴールに到達させることが可能となった．以降は，目的価値と Q 値を実際に比較して，協調行動を学習できているか，そしてそのための内部報酬及び目的価値が設定できているかを考察する．

図 4.20 は各迷路におけるエージェント A，B の Q テーブルを示す．図の左側がエージェント A，右側がエージェント B を示し，上段からそれぞれ迷路 1, 2, 3, 4, 5 の Q テーブルを示す．各 Q テーブルにおいて，白マス，紫マスがそれぞれスタート地点，ゴール地点を表し，各マスからのびる黒い矢印は行動であり，その脇にある赤い数値はその行動の Q 値を示している．また橙の矢印はスタートから Q 値が最大の行動を選択していったときに辿る経路である．図を見ると分かる通り，全エージェントが自身よりも遠くのゴール到達を目指し学習して，結果として最短ステップ数でゴールに到達できていることが分か

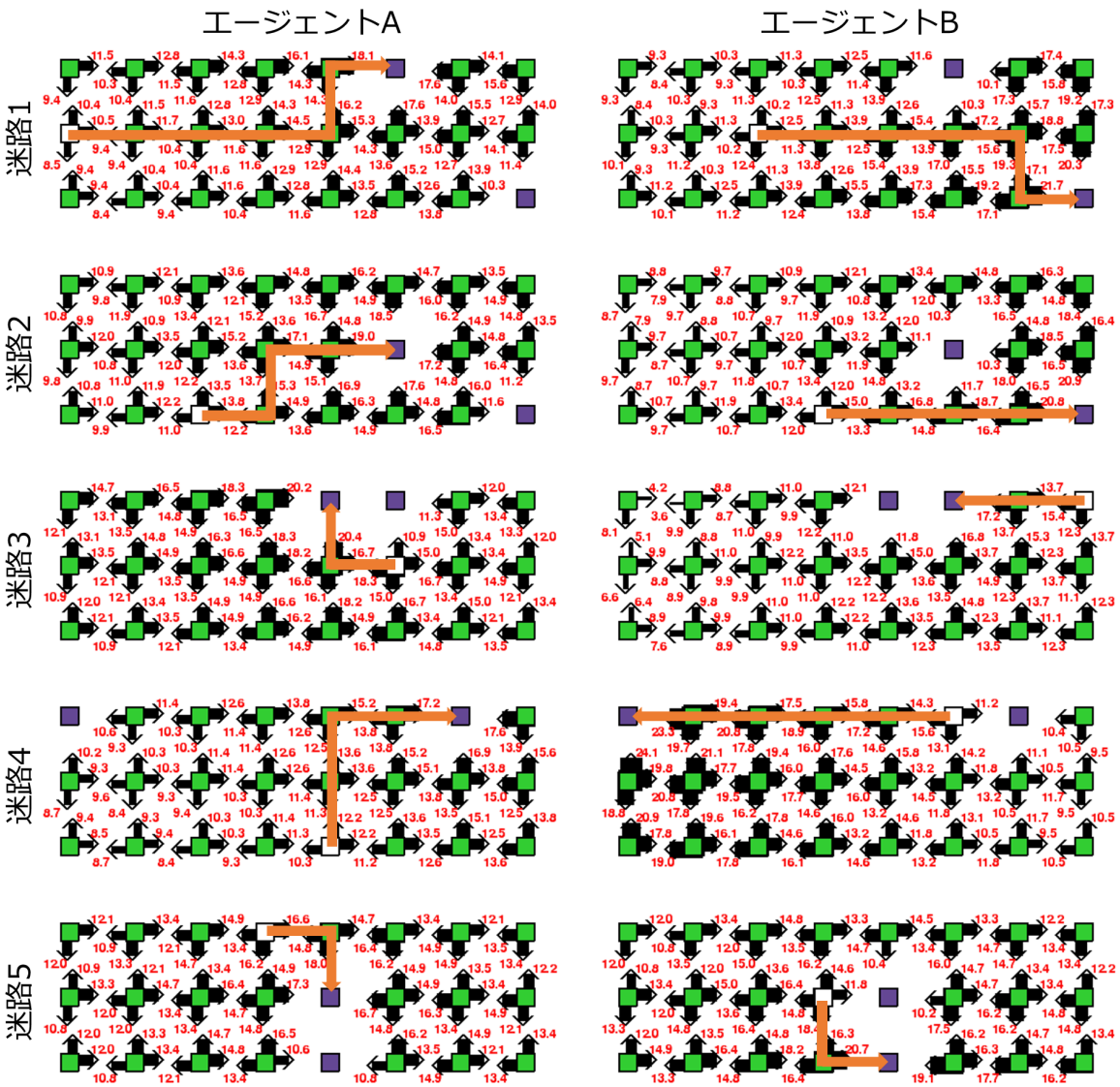


図 4.20　各迷路各エージェントの Q テーブル

る．また，図 4.21，4.22 は迷路 1 において目的価値の実験値と理論値を示している．なお η はアルゴリズムにある一定確率であり，内部報酬の設定を目的価値に従わない確率である．これを見ると分かる通り，PMRL の目的価値は理論値に近く，大小関係は等しく推定できていることが分かる．以上から，PMRL は目的価値を適切に設定し，それに合わせて Q 値を適切に推定できていることが分かる．

また，図 4.23 は，変数 δ を 0.1, 0.5, 1.0, 10 と設定した時の結果を示したものである．縦軸が最短ステップ数でゴールに到達した割合，横軸がエピソード数である．青，橙，灰，黄の線はそれぞれ変数 δ が 0.1, 0.5, 1.0, 10 の時の結果である．結果から δ が 0.1 と小さいときは Q 値の差がほぼないため，結果が安定していない．また δ が 10 の時は最も早く確率 1 に近づくが，1 シード結果が悪いときがある．これは逆に内部報酬の変化に過敏に

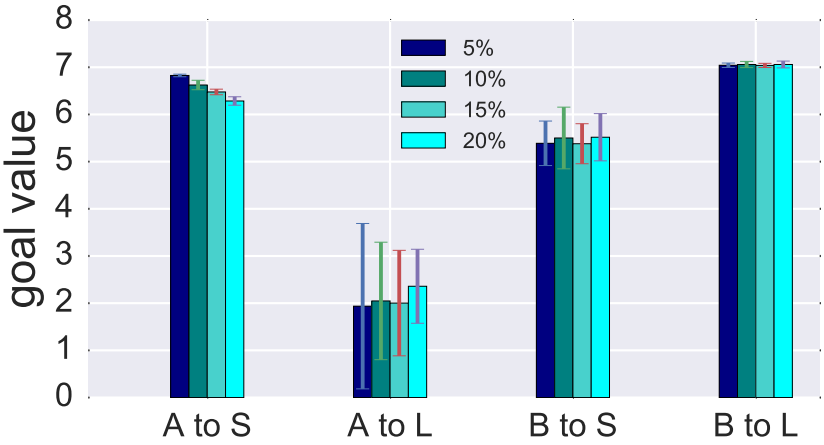


図 4.21　迷路 1 における実験で得られた目的価値

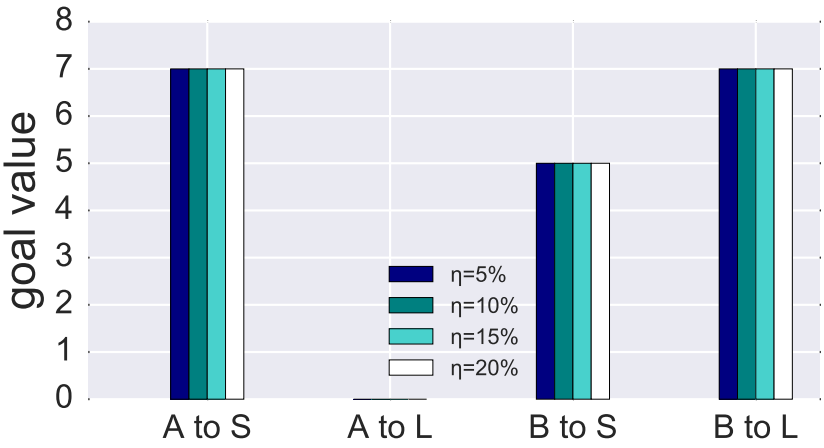


図 4.22　迷路 1 における理想的目的価値

なっているためであると考えられる．PMRL は η の確率で，目的価値が最大ではない別のゴールに到達するように内部報酬を設定する．つまり，図 4.8 のようにゴール付近に分岐点が存在するとき，内部報酬の差を大きく設定してしまうと η に従ってランダムにゴールを選択して目的価値を設定した結果，1 エピソードのみ別のゴールに到達するように学習してしまう．そのため多少ゴールの位置関係が関わってくるものの，δ は 0.1 より大きい値にすれば協調行動を学習する．

また，ここでは，ケース 2 の結果において，PS が PMRL より勝っている 2 体エージェントの迷路 5 を取り上げる．図 4.24 は各エージェントの持つ Q 値を示している．図の左側と右側は PMRL と PS の Q 値を分けており，上段下段はそれぞれエージェント A と B を分けている．またそれぞれの図において，白および紫のマスはエージェントのスタートおよびゴールを示している．また各マスから伸びる黒い矢印は行動を示し，その太さおよび傍らの赤い数値がその行動の Q 値を示している．最後に橙の矢印は各エージェント

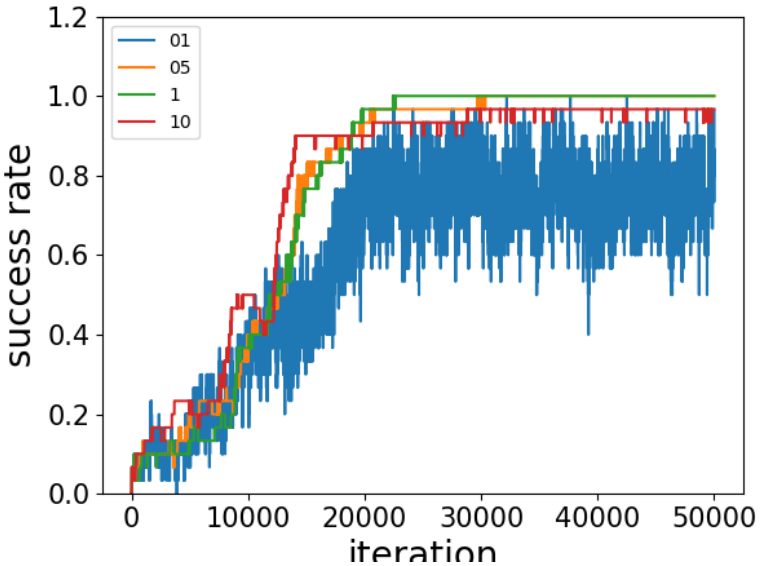


図 4.23　定数 δ の違いによる結果の変化

がスタートから Q 値の最も大きな行動をとった時に通る経路である．これを見ると，迷路 5 においてはエージェント A，B の到達するゴールが PMRL と PS で異なり，そして PS の方がよりステップ数少なく目的を達成している．これは PMRL のエージェント A においてどちらのゴールも到達ステップ数が等しく，目的価値としても同等の値になるため，エージェント B が遠くのゴールへ到達する行動を学習することを阻止できない．結果としてどちらのゴールも等しい目的価値であったエージェント A も上のゴールのほうが最初に到達できる (報酬を獲得可能である) 可能性が高いため，図のようにステップ数が多くかかると考えられる．ただしあくまで可能性 (確率) であるため PMRL はシードにより図の右側となるケースもある．図 4.25 は 2 体エージェントの迷路 5 における各エージェントの最終的な目的価値である．ここでゴール x, y とはそれぞれ上下のゴールを指し示す．これを見れば先程の推測の通りエージェント A の各ゴールの目的価値がほぼ等しくなっていることがわかる．

・5 体エージェント

ここでは，ケース 2 の結果において，PS が PMRL より勝っている 5 体エージェントの結果の詳しい分析として迷路 4 を取り上げる．

図 4.26 は 5 体エージェントの迷路 4 における PMRL が成功したケースの Q 値を示している．図の見方は図 4.24 と等しい．この図において PS はエージェント D と E がともに同じゴールへ到達してしまっているため失敗している．その一方で PMRL はすべてのエージェントが別々のゴールへ到達し，かかるステップ数も小さくすることができ

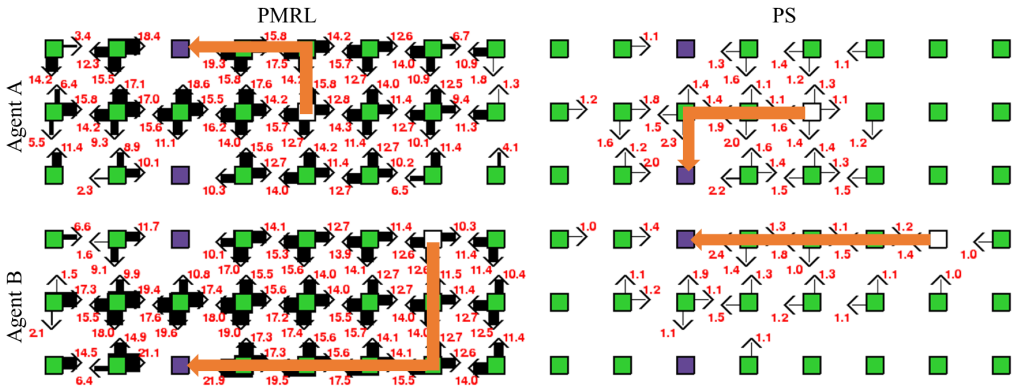


図 4.24　2 体エージェント迷路 5 における Q 値

図 4.25　2 体エージェント迷路 5 における各エージェントの最終的な目的価値

ている．一見 PS はエージェント D か E の学習を修正すれば良いように見えるが，例えばエージェント D の学習を修正した場合次に向かうゴールはエージェント C の到達しているゴールとなり，また別の競合を起こす．そのためエージェント 5 体において，PS の結果は実際に見るよりも不完全なものであるといえる．そして PMRL の結果はより優れていることがわかる．ランダム性により選択する行動が変わることで PMRL は協調行動を学習できていることから，他のケースにおいてもより多くの学習回数を費やすことで PMRL は協調行動を学習できる可能性がある (PS では全ケースにおいて成功していないため，明確に PMRL が優れているといえる．)

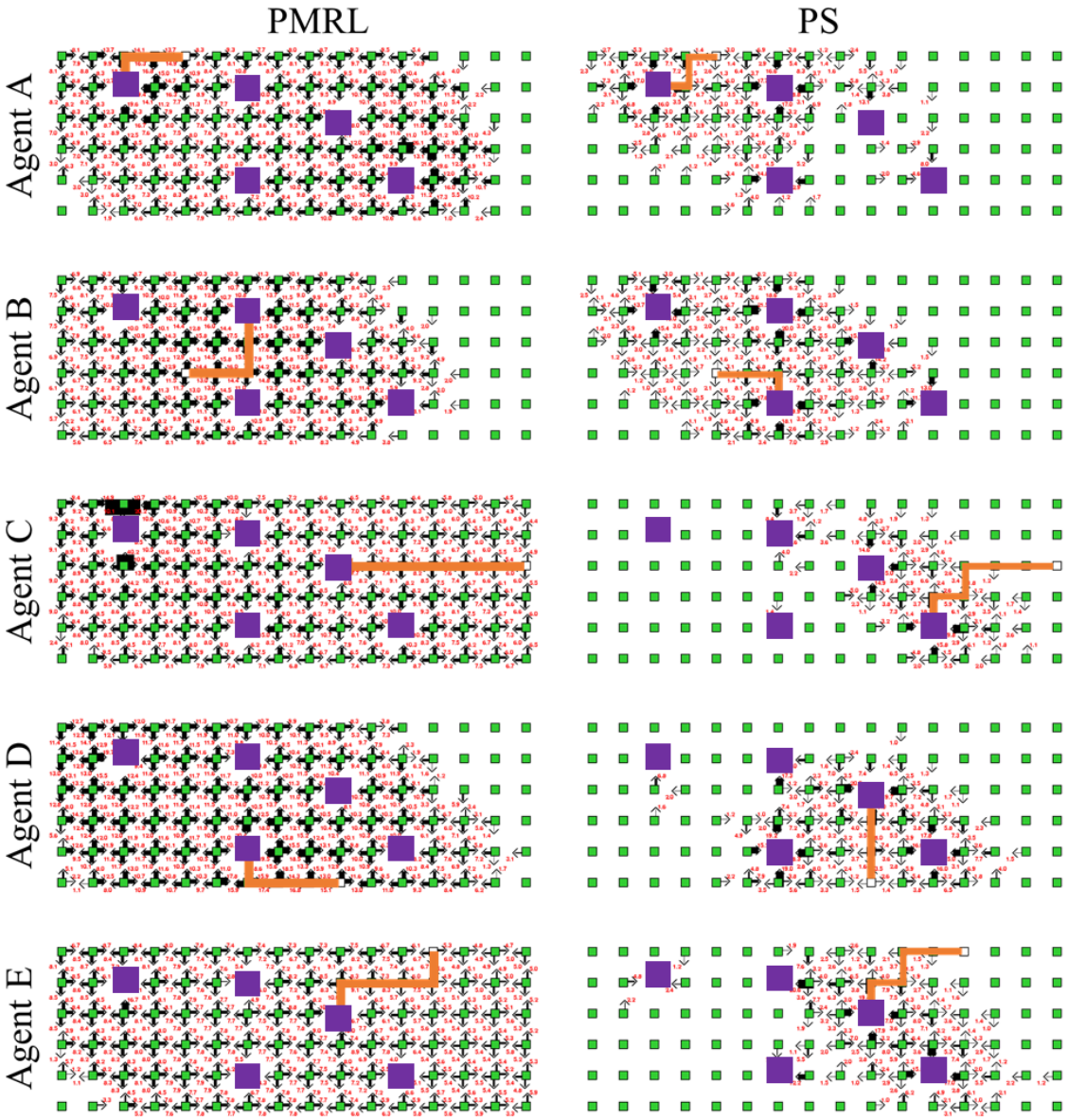


図 4.26　5 体エージェント迷路 4 における Q 値

4.7　4 章のまとめ

4.7.1　理論的証明の限界

ここでは本章の実験結果を踏まえて，PMRL の適用範囲について考察する．PMRL による理論的証明は下記の 2 点であるが，適用範囲としてはその枠を逸脱していないことがわかる．

1. 目的価値によりエージェントはお互いに異なるゴールへ到達する行動を学習
2. 各ゴールの到達ステップ数が互いに異なるとき 2 体エージェントにおいて最短ステップ数でゴールに到達する行動を学習

例えば証明 1 に関して言えば実験 2 においてステップ数が等しい迷路に PMRL を適用したが，その際でもお互いのエージェントが等しいゴールへ到達することが無い．証明 2 に関しては実験 1 から互いに異なる到達ステップ数であれば最短ステップ数でゴールに到達可能であることを示しており，想定する範囲で正しいことを示している．また想定の範囲外の到達ステップ数が等しいゴールが存在していたとしても，適切なゴールへ到達する行動を学習できる可能性がある．その理由は目的価値によるゴール選択のランダム性が関係している．つまり，到達ステップ数が等しいゴールでは目的価値が等しくなり，それぞれのゴールに対してほぼ等しい確率で選択される．その場合，獲得した報酬値の合計がより大きい (最初に到達する割合の大きい) ゴールへ到達するように学習する．結果として，例えば実験 2 の 2 体エージェントにおける迷路 5 の環境であればエージェント B が上側の (最短ステップ数の小さいほうの) ゴールをより高い確率で選択できていれば，PMRL も適切な行動を学習したことになる．そのため，2 体エージェントの環境全体でいえば，各ゴールに対する最短ステップ数が等しい場合はそれぞれ別々のゴールは選択できるが，到達ステップ数を最小にするためには乱数に依存する．一方で最短ステップ数が異なれば必ず最短ステップ数でゴールに到達することが可能である．

4.7.2 適用するエージェント数の限界

実験 2 の結果からエージェント数に関しても上記の証明同様に示せることは明白である．つまり，エージェント数の増加に従いより多くの学習回数を必要とするが各エージェントが別々のゴールに到達することは可能であり，最短ステップ数でゴールに到達するかどうかは上記の証明の条件に従うか否かに依存する．実験 2 の 5 体エージェントにおける迷路 4 の結果を見ればゴールとのステップ数が等しいエージェントも存在しているが，それでも別々のゴールに到達する行動を学習していることからもそれがわかる．なお証明は 2 体エージェントのみを想定しているため，エージェント数が増えた時点で条件に従うケースは存在しないが，文献 [42] にて考察しているとおり各ゴールの位置関係が全エージェントで等しい場合 (つまり各エージェントで全ゴールを近い順に並べたときその全てが等しくなる場合) には理論的な証明の範囲であるとはいえ，その際は最短ステップ数でゴールに到達することができると考えられる．

4.7.3 報酬設計の限界

PMRL は内部報酬を設定し，各エージェントが目的価値により選択したゴールに到達する行動を学習させる．内部報酬は正の数であれば理論上どのような値でも PMRL は適切に機能する．しかしながら行動選択法次第では適切な値が変わってくる．つまり，ϵ-グリーディ選択であればどれほど小さな Q 値の差であれ Q 値が最大の行動をとるためどのような内部報酬値でも良いが，ボルツマン選択のように Q 値に従って確率的に行動を選択するような場合，それぞれの行動の Q 値の差が重要となってくるため，学習により Q 値の差を生む内部報酬値の差が重要となる．具体的に適切な内部報酬値の差は迷路の広さ，もといゴールに到達するまでのステップ数に依存する．つまり最も遠くのゴールを目指す際にその確率が大きくなるような差を付けることが重要である (ただしそうしなくてもいずれは適切なゴールに到達することができるが，学習回数がより多く必要となる．)．一方で実験 2 の PS の Q 値を見るとあまり大きく出ておらず，ゴール周辺の Q 値だけが大きくなっている．これは報酬値が安定しておらず学習が上手くできていないという証拠である．これからわかることは報酬の設計は難しく，より多くの情報を使う設定 (PS の設定) であれ適切に学習できるような最適な設定は発見困難であるといえる．今回の設定は文献 [19, 35] などでも使われる一般的なものであり，その難しさがうかがえる．以上の点においても，適切な報酬値の設定法が確立されている PMRL の内部報酬設計は大きな価値があるといえる．

第 5 章

Profit minimizing reinforcement learning with oblivion of memory (PMRL-OM)

5.1　アプローチ

PMRL-OM は空間的環境変化に対する提案手法である．以下にその詳しい説明と実験結果を載せる．ここでは空間的環境変化に対するアプローチを説明する．空間的環境変化に追従する上で重要な点は変化する環境の状態行動空間に適応することである．特に PMRL では目的価値を環境変化にあわせて変化させることが求められるが，PMRL は目的価値を収束させるため，それができない．そのために PMRL-OM では，PMRL の目的価値推定において初期の結果を減衰させる関数を導入して，学習初期の情報を利用しないように変更を加えた．

図 5.1 は PMRL-OM の想定する空間的環境変化である．具体的には，3 章の中の，環境状態変化およびエージェントのスタート位置，ゴール位置の変化である．このとき，PMRL が利用する情報の中で最短ステップ数のみが動的変化するため，環境変化にあわせて最短ステップ数を更新し，その更新した最短ステップ数から目的価値を推定しなければ適切に学習することはできない．しかしながら，PMRL は記憶する最短ステップ数や目的価値を収束させてしまうため環境変動に対して適切な学習ができない．そこで，PMRL-OM では，最短ステップ数の更新方法及び目的価値の推定において初期に学習した情報を忘却する機構を導入し，収束性を取り払うことで，空間的環境変化に追従して協調行動を学習可能とする．なお PMRL-OM も PMRL と同様にして，学習初期はまだ到達していないゴールも存在するため，到達しているゴールの中から最大ステップ数となる (最も遠くの) ものを目指して学習する．

図 5.2 は空間的環境変化により起こる PMRL の問題点と PMRL-OM により解決する

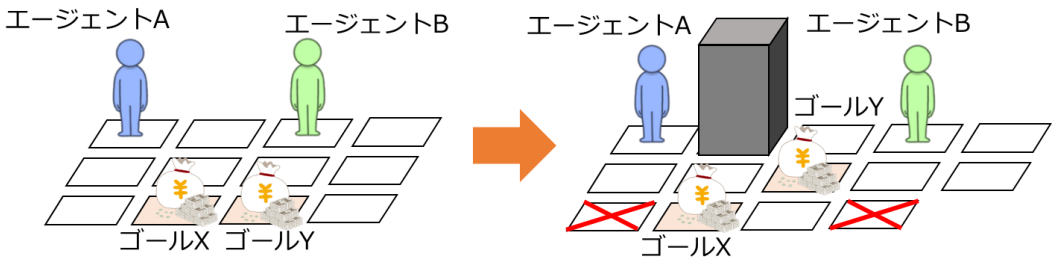


図 5.1　PMRL-OM の想定する環境変化

様子を示している．図の上段は空間的環境変化における PMRL を示し，下段はそれが解決した時のエージェントの経路と目的価値である．まず PMRL は環境変化を知らずに最短ステップ数を更新するため，環境変化後に最短ステップ数が変化したとしてもそれを反映できない．更に PMRL は目的価値を収束させ，環境が変化しても環境変化前の目的価値を保持し続けてしまうため，例えば環境変化前はエージェント A，B の目的価値がゴール X，Y が最大である図のような状況では環境変化後にはエージェント A，B がそれぞれゴール Y，X の目的価値を上昇させなければならないが，既に目的価値が収束しているため変更できない．そのため，PMRL-OM ではまず最短ステップ数の更新を環境変化後には変化後の最短ステップ数に更新できるようにし，目的価値の更新は図の下段のように収束性をなくし，環境変化後にはそれに合わせて目的価値を変化させられるように改良する．

5.2　アーキテクチャ

PMRL-OM において，エージェントのアーキテクチャは PMRL と等しく，実行部の中身が異なる．図 5.3 は PMRL-OM エージェントのアーキテクチャを示している．エージェント内部は大きくメモリ部と実行部に分かれており，メモリ部には Q 値など学習に必要な変数値を持ち，実行部には PMRL-OM のアルゴリズムを実行するプロセスを持つ．具体的にメモリ部は環境の全状態と全行動に対する Q 値を示す Q テーブルと，各目的達成までの最短のステップ数と，協調のために達成すべき目的を示す価値である目的価値，そして内部報酬を持つ．実行部は Q 学習において実行される状態観測，行動選択，Q 値の更新の他に，最短ステップ数更新，目的価値更新と内部報酬設定のプロセスを持つ．以上は PMRL と同様の機構である．

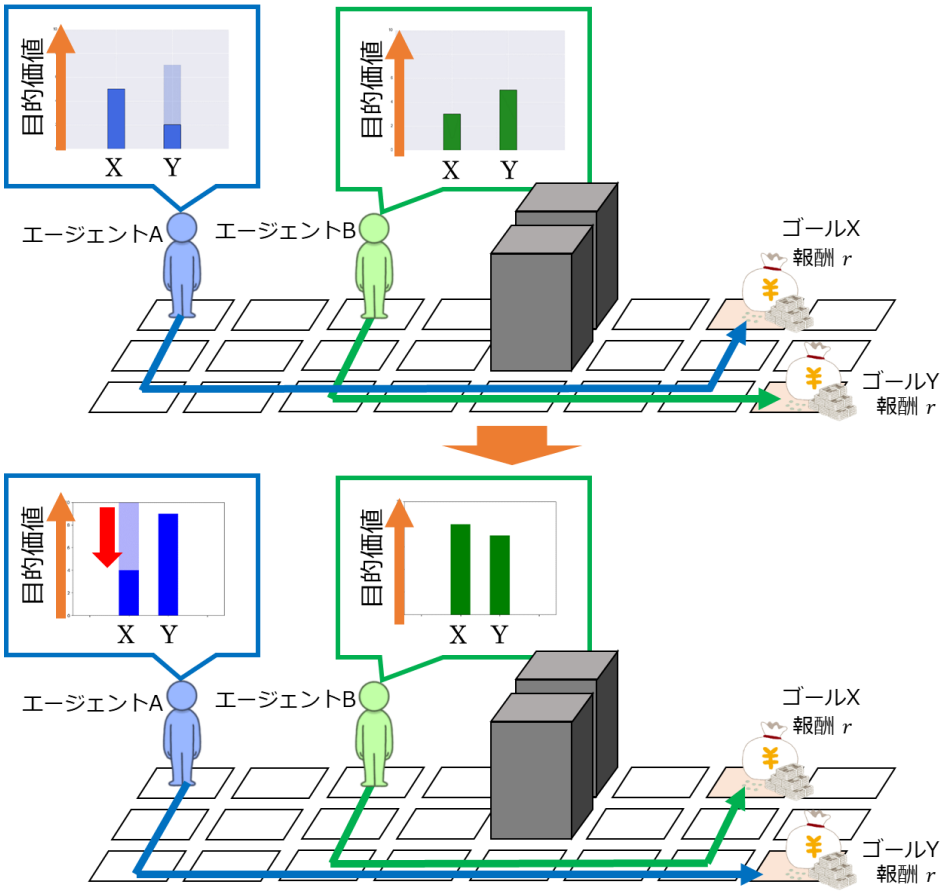


図 5.2　PMRL により起こる問題と PMRL-OM による解決

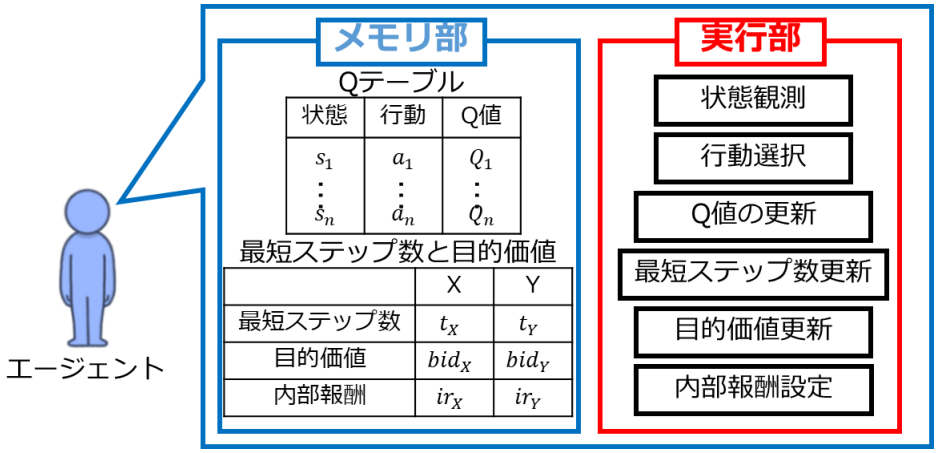


図 5.3　PMRL-OM エージェントのアーキテクチャ

5.3 メカニズム

5.3.1 最短ステップ数の更新

図 5.4 に PMRL-OM におけるエージェントの最短ステップ数の保存法を示す．エージェントは学習毎に到達したゴールに対するステップ数を更新しており，図はエピソード e と $e+1$ のときの最短ステップ数の保存を表している．エージェントの吹出しはエージェントのメモリを示し，その中の表は各学習回数（Time）とエージェントが到達したゴールの種類（Goal），そしてそこへ到達するまでのステップ数（Step）を示している．そして吹き出しの外にある表はメモリ内のデータから作成した最短ステップ数を保存した表である．エージェントは各エピソードでゴールに到達した場合，到達までのステップ数をメモリ内の表に挿入していく．図では，新たに (Time, Goal, Step) = (51, X, 7) が挿入されている．PMRL-OM は窓長 e をメモリ長の閾値として設定しており，例えば窓長 e であれば，図のようにメモリ長が e になるようにメモリ内の最も古い（Time の最も小さい）データから消去される．PMRL-OM はこのように古くから記憶されているステップ数から削除することで，仮に環境状態が変化し，ゴールまでのステップ数が変化した際にも環境変化後の最短ステップ数を推定することができる．

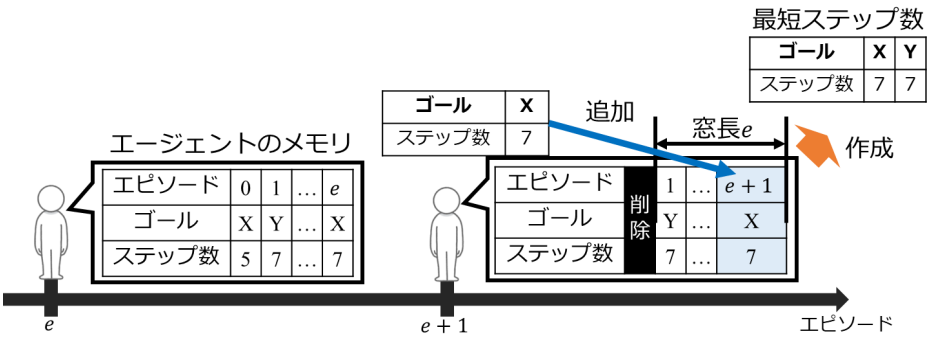


図 5.4　PMRL-OM の最短ステップ数保存法

5.3.2 目的価値の更新

目的価値は協調行動を学習するためにそれぞれのゴールがどれだけふさわしいかを示す指標である．目的価値はその値自体に大きな意味はなく，その大小関係に意味がある．そして，目的価値の最も大きいゴールが最適なゴールとなる．式 (5.1) は目的価値の更新式を示している．式 (5.1) において，bid_g^i が任意のエージェント i の任意のゴール g における目的価値であり，t_g^i はエージェント i がゴール g へ到達するまでの最短ステップ数を示す．ξ は定数を表し，本論文では 500 を設定する．なお ξ は目的価値を計算する際に前

の学習をどれほど重要視するかを表す変数であり，この値が大きいほど初期の学習により求めた目的価値が含まれることになる．各エージェントは学習毎に目的価値を式 (5.1) に従って更新し，次の学習から目的価値の最も大きなゴール g に到達するように内部報酬を設定する．式 (5.1) は更新を繰り返すと最短ステップ数 t_g^i に収束するため，自身から遠いゴールほど評価される．また各ゴール到達までの最短ステップ数の大きさを評価しているため，自身が他のどのエージェントよりも近いゴールの中から，最も遠くのゴールを選択することになる．

$$bid_g^i \leftarrow \begin{cases} \frac{\xi-1}{\xi} bid_g^i + \frac{t_g^i}{\xi} & if\ r^i = r_g \\ \frac{\xi-1}{\xi} bid_g^i + \frac{0}{\xi} & otherwize \end{cases} \tag{5.1}$$

5.4 PMRL-OM の合理性

5.4.1 目的価値の収束性証明

ここでは PMRL-OM エージェントが学習により，協調行動を獲得し，全エージェントの単位時間当たりの獲得報酬値を最大化させることを示す．PMRL-OM が PMRL と異なる部分は目的価値設定のみであるため，ここでは，目的価値の収束性を証明することで，そのときの内部報酬値により推定される Q 値の収束値によりエージェントが最適ゴールに到達可能となることを示すこととする．

ここでは，各ゴールに対する目的価値が最短ステップ数に収束することを示す．まず目的価値の更新式を展開すると，下記のようになる．

$$bid_g^i \leftarrow \frac{(\xi-1)bid_g^i + ft(n_g)}{\xi} \tag{5.2}$$

$$\leftarrow \frac{(\xi-1)}{\xi}\left(\frac{\xi-1}{\xi} bid_g^i + \frac{ft(n_g-1)}{\xi}\right) + \frac{ft(n_g)}{\xi} \tag{5.3}$$

$$= \sum_{m=1}^{n_g} \left(\frac{\xi-1}{\xi}\right)^{n_g-m} \frac{ft(m)}{\xi} \tag{5.4}$$

$ft(m)$ が一定の値 t_g を返すとき (エージェントがそのゴールに最も早く到達する確率が高いとき)，ξ は正の定数であり，$0 < \frac{\xi-1}{\xi} < 1$ となるため，上記の式 (5.4) は式 (5.5) に従い，定数値 t_g を示す．t_g はゴール g へ到達するための最短ステップ数であるため，n_g が無限大であれば PMRL-OM の目的価値推定の式は PMRL と同様に最短ステップ数に収束する．

$$\sum_{m=1}^{n_g} \left(\frac{\xi-1}{\xi}\right)^{n_g-m} \frac{ft(m)}{\xi} = \frac{t_g}{\xi} \frac{1}{1-\frac{\xi-1}{\xi}} = t_g \tag{5.5}$$

5.4.2 最短ステップ数の必要性

エージェントは学習が進めばそのゴールに最短ステップ数で到達可能であるため，上記の予め求めた最短ステップ数 t_g^i が誤差を含むときは，最小値を計算するのではなく到達ステップ数の平均をとることが適切である．また式 (5.4) は近似的に平均を求める式 (以降で説明) であるため，最短ステップ数を計算せずにステップ数をそのまま導入しても PMRL-OM は適切に機能する．ただし最短ステップ数を使うことで，PMRL-OM は PMRL の保証する性能が出せるため，最短ステップ数に基づく PMRL-OM も必要な技術であると付け加える．

ここで各ステップ数と最短ステップ数との差分をゴール到達数 n の関数 $d_g^i(n)$ とすると，改良した目的価値を $mbid_g^i$ として式 (5.4) は下記の式 (5.6) の通りに書き換えられる．またこの式を展開すると式 (5.7) の通り，従来の目的価値関数に誤差関数の合計値を加えた形になる．前節において $bid_g^i \approx t_g^i$ としており，また式 (5.7) の誤差関数は式 (5.8) の通り展開できる．n_g が左程大きくなく ξ が十分に大きな値であれば $\xi \approx \xi - 1$ とすることができるため，目的価値は近似式 (5.9) の通り計算できる．この式は $n_g = \xi$ であれば今までのエピソード内でのステップ数誤差の平均値を計算していることとなる．つまりパラメータ ξ は学習回数 (環境変化のタイミング) と等しくすることが PMRL-OM を適切に機能させる上で重要である．ただし，ξ が大きく異ならない限り $n_g = \xi$ の時と同様の性能を示す．

$$mbid_g^i = \sum_{m=1}^{n_g} \left(\frac{\xi - 1}{\xi}\right)^{n_g - m} \frac{t_g^i + d_g^i(m)}{\xi} \tag{5.6}$$

$$= bid_g^i + \sum_{m=1}^{n_g} \left(\frac{\xi - 1}{\xi}\right)^{n_g - m} \frac{d_g^i(m)}{\xi} \tag{5.7}$$

$$\approx t_g^i + \frac{(\xi - 1)^{n_g - 1} \cdot d_g^i(1)}{\xi^{n_g}} + ... + \frac{\xi^{n_g - 1} \cdot d_g^i(n_g)}{\xi^{n_g}} \tag{5.8}$$

$$\approx t_g^i + \frac{\sum_{m=1}^{n_g} d_g^i(m)}{\xi} \tag{5.9}$$

またエピソードが進んで n_g が大きな値になるとき $\xi \approx \xi - 1$ とすることはできず，重み付き平均の値となる．学習初期におけるその重みの上限を $weight$ と設定すれば，各パラメータ ξ, n_g は下記の式の通り計算される．この重みは 0 になることはないが，例えば $weight = 0.001$ を想定して $\xi = 500$ を設定すれば，$n_g \leq 3451$ となり，3451 エピソードの時点で学習初期の誤差はほぼ 0 になっていることがわかる．そして 3451 エピソード以降も重みは徐々に小さくなっていくため，$\xi = 500$ での目的価値は直近約 3500 エピソードの誤差平均を最短ステップ数に加えた値になる．学習が進めばエージェントの通る経路も確定されて誤差も小さくなるため，より最短ステップ数に近づくことがわかる．以上か

ら本改良は断続的環境変化によるステップ数の不安定性に対して頑健であることがわかる．つまりパラメータ ξ の条件は多少厳しくなるものの，PMRL-OM は近似的に最短ステップ数なしでも機能することができる (以降は PMRL-OM+ と呼ぶ．)．そして，環境が複数回変化し，そのタイミングが自明でないときは最短ステップ数をそもそも適切に求めることが困難であり，PMRL-OM+ の効果は大きいと思われる．

$$\left(\frac{\xi-1}{\xi}\right)^{n_g-1} \leq weight \tag{5.10}$$

$$n_g \leq \frac{\ln weight}{\ln(\xi-1)-\ln\xi}+1 \tag{5.11}$$

5.5 アルゴリズム

PMRL-OM のフローチャートを図 5.5 に示す．PMRL-OM のアルゴリズムは PMRL のものと同じものであり，違いは最短ステップ数更新と目的選択の中身であり，アルゴリズムフローとしてはどういうつのものとなる．詳しく説明すると，PMRL-OM ではエージェントがまず Q 学習と同様に状態を観測し，行動を選択することで次状態に遷移して，報酬を獲得する (プロセス 1-4)．その後 PMRL-OM も同様に，エージェントはゴールに到達したか否かによって処理が異なり，ゴールに到達していないのであれば獲得した報酬に基づき Q 値を更新してまた次のステップを繰り返す (プロセス 9)．一方でゴールに到達しているのであれば，そのゴール到達までのステップ数が今までそのゴールへ到達したステップ数の中で最小であれば更新し，その後更新された最短ステップ数から目的を選択し，目的価値というものを更新する (プロセス 6, 7)．そして選択した目的からその選択したゴールに到達できるように更新された最短ステップ数から内部報酬を設定する (プロセス 8)．以上をステップ数やエピソード数が閾値を超えるまで繰り返して学習する (プロセス 10)．以上が PMRL-OM のアルゴリズムフローの全貌である．

PMRL-OM のアルゴリズムを Algorithm4 に示す．Algorithm4 において，$\boldsymbol{mt}$，$\boldsymbol{t}$ は各学習ごとのステップ数を保存する配列と，それを基に最短ステップ数を求め格納する配列である．また，$\boldsymbol{bid}$ は目的価値を格納する配列で，ir は内部報酬を格納する配列である．初期状態は各学習毎同じ状態 s_0 を設定し，ゴールは配列 s_{end} に格納する．まずはじめにエージェントは初期状態 s_0 を観測し（6 行目），観測状態に対し行動を選択する（8 行目）．その後行動 a を実行し，報酬 r を獲得して，次状態 s' へ遷移する（9 行目）．そして報酬から内部報酬 ir を計算し，内部報酬に基づき Q 値を更新する（11 行目）．このとき次状態 s' がゴールであった場合学習を終了する（13，14，15 行目），そうでなかった場合はステップを繰り返す．最終的に一定のステップ $MaxStep$ 経った後にゴールの状態を観測できなかった時は学習を終了する．学習終了後，$UpdateStepOM$ 関数を実行する (17 行目)．その後目的価値を設定し，内部報酬へ設定するために最も適したゴールを決定

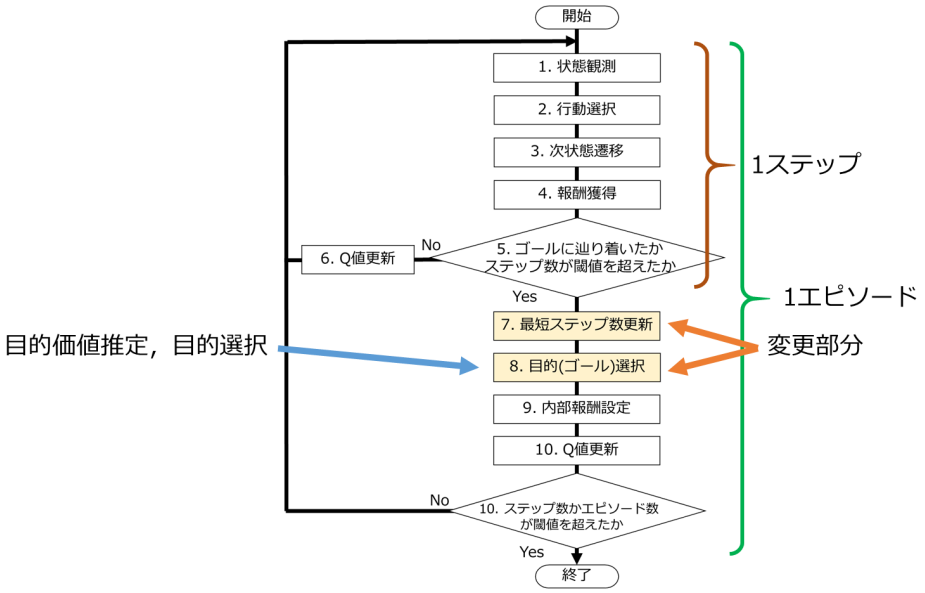


図 5.5　PMRL-OM のアルゴリズムフロー

する（18-25 行目）．具体的には，まず配列 $\boldsymbol{bid}$ の値を更新する（18-22 行目）．そしてその値が最大となる引数 g を $selection$ に格納する（23，24，25 行目）．その後一定の確率で $selection$ をランダムなゴールに置き換える（26 行目）．以上が PMRL-OM のアルゴリズムである．

また，関数 $UpdateStepOM$ のアルゴリズムを Algorithm5 に示す．関数 $UpdateStep$ は配列 $\boldsymbol{mt}$ と最短ステップ数配列 $\boldsymbol{t}$ を入力として，$\boldsymbol{t}$ へ値を格納するため出力はない．まず関数 $UpdateStep$ を呼び出した際にゴールの状態であった場合，その時の学習回数と，ゴールの種類，そしてステップ数を配列 $\boldsymbol{mt}$ へ格納する．格納後，配列の長さが閾値 $MaxLength$ よりも大きくなってしまった場合，配列の先頭要素を削除する（1-6 行目）．その後は配列 $\boldsymbol{mt}$ に基づき，最短ステップ数を求める．具体的には配列のすべての要素を探索し，各ゴールに対して最も小さいステップ数を見つけ，配列 $\boldsymbol{t}$ へ格納する（7-12 行目）．仮に $\boldsymbol{mt}$ に存在しないゴールがあれば，その要素は $MaxStep$ となる．最短ステップ数が求まれば関数 $UpdateStep$ を終了する．以上が $UpdateStep$ 関数のアルゴリズムである．なお PMRL-OM+ のアルゴリズムは PMRL-OM の最短ステップ数を求める記述を除き，目的価値関数にステップ数を直接入力すればよい．

Algorithm 4 PMRL-OM

1. $Q(s,a)$ を 0 で初期化，$\forall s \in S, \forall a \in A$
2. ステップ数保存用配列 $\boldsymbol{mt}$ と最短ステップ数配列 $\boldsymbol{t}$ を $MaxStep$ で初期化
3. 目的価値 $\boldsymbol{bid}$ と内部報酬 $\boldsymbol{ir}$ を 0 で初期化
4. 初期状態 s_{start} とゴール集合 $\boldsymbol{s_{end}}$ の設定
5. **for** $iteration = 1$ to $MaxIteration$ **do**
6. $s = s_{start}$
7. **for** $step = 1$ to $MaxStep$ **do**
8. 行動 $a = ActionSelect(Q, s)$
9. 行動 a を実行，報酬 r を獲得，次状態 s' へ遷移
10. 報酬から $selection$ へ到達するように内部報酬 ir を計算
11. $Q(s,a) = Q(s,a) + \alpha\left[ir + \gamma \max_{a' \in A} Q(s',a') - Q(s,a)\right]$
12. $step = step + 1$
13. **if** $s' \in s_{end}$ **then**
14. break
15. **end if**
16. **end for**
17. $UpdateStepOM(\boldsymbol{mt}, \boldsymbol{t})$
18. **if** 報酬獲得した **then**
19. $\boldsymbol{bid}[selection] = \frac{\xi - 1}{\xi}\boldsymbol{bid}[selection] + \frac{\boldsymbol{t}[selection]}{\xi}$
20. **else**
21. $\boldsymbol{bid}[selection] = \frac{\xi - 1}{\xi}\boldsymbol{bid}[selection] + \frac{0}{\xi}$
22. **end if**
23. **if** $\boldsymbol{bid}[g]$ が最大 **then**
24. $selection = g$
25. **end if**
26. 一定確率で $selection$ をランダム選択
27. **end for**

Algorithm 5 PMRL-OM の目的価値設定関数 $UpdateStepOM$

Input: 自身の到達ゴールとステップ数の配列と最短ステップ数配列 $\boldsymbol{mt}$,$\boldsymbol{t}$

Output: なし

1. **if** $g \in s_{end}$ **then**
2. 配列 $\boldsymbol{mt}$ の末尾に $(iteration, g, step)$ を挿入
3. **if** 配列 $\boldsymbol{mt}$ の長さが閾値 $MaxLength$ 以上 **then**
4. $\boldsymbol{mt}$ の先頭要素を削除
5. **end if**
6. **end if**
7. $\boldsymbol{t}$ の全要素に $MaxStep$ を代入
8. **for** $index = 0$ to $MaxLength$ **do**
9. **if** $\boldsymbol{mt}[index, g] < \boldsymbol{t}[g]$ **then**
10. $\boldsymbol{t}[g] = \boldsymbol{mt}[index, g]$
11. **end if**
12. **end for**

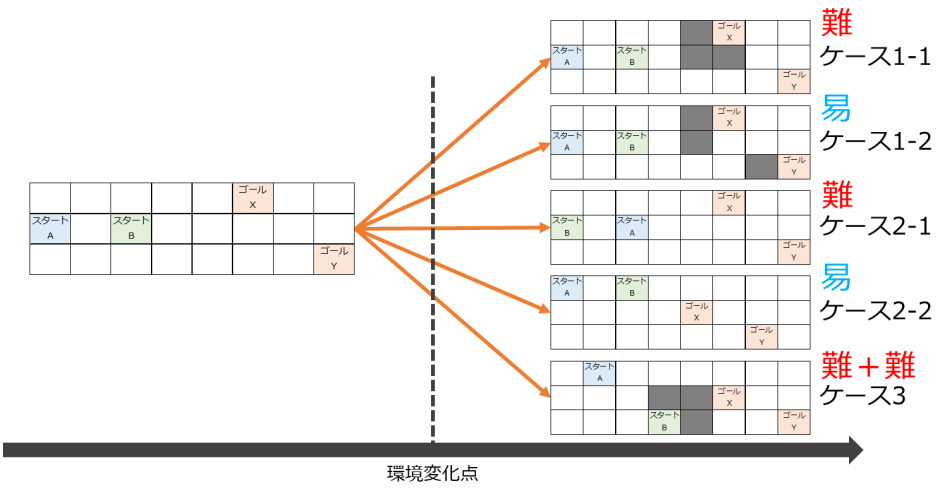


図 5.6　環境変化を伴う迷路

5.6　実験 1: 全ケースにおける結果

5.6.1　実験内容

PMRL-OM の有効性を検証するため，本研究では 2 体エージェントにおける迷路問題の上で PMRL-OM と PMRL および PS の性能を比較する．具体的には図 5.6 の 5 つのケースの動的変化する迷路問題（以降それぞれのケースを上からケース 1-1, 1-2, 2-1, 2-2, 3 と呼ぶ）を採用する．図において，下段はエピソード数であり，ある一定のエピソード数 (環境変化点) を境にして，左側の迷路から右側の迷路へ環境が変化する．また各図の各マスが状態を表し，A，B および X，Y と表示されたマスがそれぞれのエージェントの初期状態とそれぞれのゴールを表す．また図の黒マスは通行止めの状態を示している．ケース 1-1, 1-2, 3 においては壁が発生し，ケース 2-1, 2-2 ではそれがない．また，ケース 1-2, 2-2 では協調行動学習のために各エージェントが環境変化前と別のゴールに到達する必要がなく，ケース 1-1, 2-1, 3 ではそれがないため，この 5 つのケースを検証することで，環境状態，エージェントのスタート位置，ゴール位置の動的変化の要素すべてを検証する事ができる．なお，環境変化においてはその前後でゴールに到達するか変化する方が難しく，ケース 1-1，2-1 の方がケース 1-2，2-2 よりも難しく，その全てが複合したケース 3 が最も難しい．

表 5.1　PMRL-OM の実験 1 のパラメータ

	PMRL-OM	PMRL	PS
エピソード数	50000		
環境変化点	25000		
最大ステップ数	100		
初期 Q 値	0		
学習率 α	0.1		
割引率 γ	0.9		
報酬値	10		
定数 δ	10		なし
定数 ξ	100	なし	

5.6.2　評価基準とパラメータ設定

本実験は各迷路で乱数のシードを変更した 30 試行実験を行い，すべてのエージェントがゴールに到達したステップ数を評価する．なお，学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じようにエピソードを繰り返し，その時の到達ステップ数を評価する．

表 5.1 に実験パラメータを示す．このとき学習エピソード数が 50000 回 (1 行目)，環境変化点は 25000 回 (2 行目) に設定する．また 1 学習における最大のステップ数を 100 に設定する (3 行目)．また学習において初期 Q 値は 0(4 行目)，学習のパラメータをは学習率 α が 0.1(5 行目)，割引率 γ は 0.9 に設定する (PS の割引率 S もこれに相当)(6 行目)．報酬値は 10 に設定する (7 行目)．また，PMRL の内部報酬設定のための定数 δ は 10 とする (8 行目)．PS は本論文にて紹介した報酬関数を利用する．

5.6.3　実験結果

図 5.7-5.11 はそれぞれの迷路における PMRL-OM と PMRL と PS での到達ステップ数の違いを示している．縦軸が最短ステップ数でゴールに到達した割合，横軸がエピソード数である．また青，橙，緑の線がそれぞれ PS の結果，PMRL の結果，PMRL-OM の結果を示している．このとき仮にエージェントが同じゴールに到達してしまった場合，最大ステップ数を示す (ここでは 100)．この図を見ると分かる通り，PMRL はステップ数を小さくすることができているが，ケース 2 番台においては同じゴールに到達する行動を学習してしまってステップ数が 100 になっている．一方で PMRL-OM の方は PMRL よりも

収束ステップ数は小さく，そして PMRL-OM は全てのケースにおいて最短ステップ数でゴールに到達できている．一方で，PS はケース 3 以外はステップ数が 100 になっており，ケース 3 においても適切に学習できているとはいいがたい．この結果から，PMRL-OM は環境状態，スタート位置，ゴール位置の動的変化に追従して協調行動を学習し，全エージェントが最短ステップ数でゴールに到達可能であることがわかる．

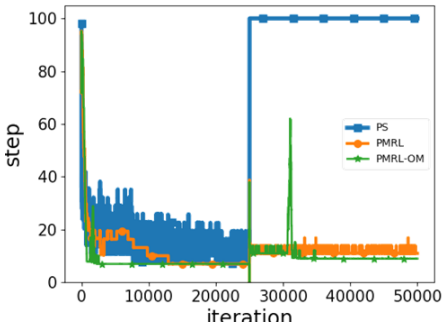


図 5.7　ケース 1-1 のステップ数の比較結果

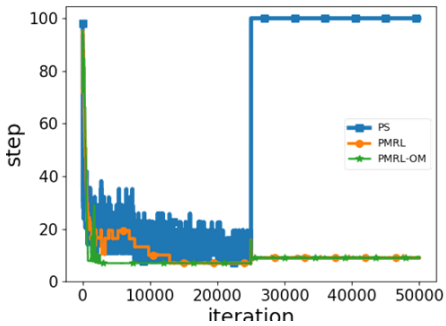


図 5.8　ケース 1-2 のステップ数の比較結果

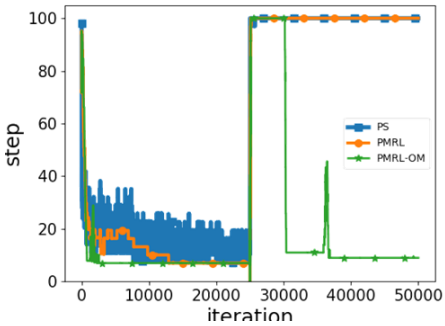


図 5.9　ケース 2-1 のステップ数の比較結果

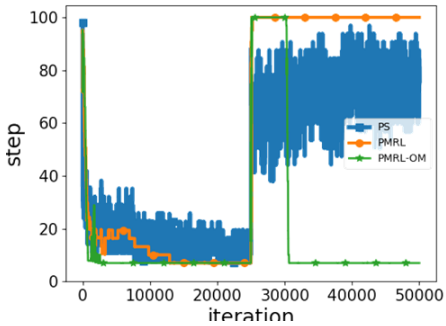


図 5.10　ケース 2-2 のステップ数の比較結果

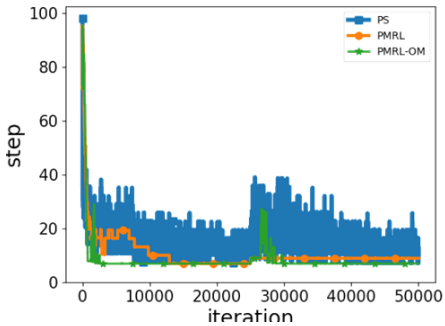


図 5.11　ケース 3 のステップ数の比較結果

図 5.12-5.15 はケース 1-1 から 2-2 までの 4 つのケースにおいて新たに用意した 3 種類の迷路に PMRL-OM を適用した際の主な結果を示す．それぞれの図において縦軸はエー

ジェントがゴールに到達するまでのステップ数，横軸はエピソード数を示す．丸付き線と四角付き線，そして点線はそれぞれ PMRL-OM，PMRL の結果とこのケースにおける最短ステップ数の推移を表している．なおそれぞれの線は乱数シードの異なる 30 試行の平均値を示しており，全エージェントが同じゴールに到達するときはここでは 100 を示すことに注意されたい．これらの結果から，PMRL はケース 1-2 を除いて最短ステップ数を示しておらず，ケース 2-2 においては全くゴールに到達できていないことがわかる．一方で PMRL-OM は環境変化後も良好な性能を示していることがわかる．これは他の 2 つの迷路においても同様である．ここで，それぞれの迷路の結果についてウィルコクソンの順位和検定により検証を行う．またここでは 2 種類の検定を行い，それぞれ検定 1 と 2 と呼ぶことにする．検定 1 ではそれぞれの迷路の結果における最終エピソードの結果 30 試行分を比較する．このときこれらのデータ 30 個は特定の分布に従っているわけではなく (つまり，ノンパラメトリック検定がふさわしい)，これらのデータはお互いに関連している (同一シードでそれぞれの手法を比較する)．つまり検定 1 はウィルコクソンの順位和検定が適している．我々は検定 1 によりそれぞれの迷路における全乱数シードで PMRL よりも PMRL-OM の方が性能が上回ることを確認する．表 5.2 は検定 1 における全ケースの全迷路における p 値を示している．各行はケース番号を示しており，上からケース 1-1, 1-2, 2-1, 2-2 となっている．また各列は左から迷路 1, 迷路 2, 迷路 3 を示している (図 5.6 のそれぞれの迷路が，この表の迷路 1 を示す)．結果として，p 値は限りなく小さい値を示しており，検定の結果は良好である (優位に PMRL-OM が性能が良い) といえる．ちなみに 2 箇所 NaN の値を示しているが，これは PMRL-OM と PMRL で性能に差がない (結果のステップ数が等しい値を示す) ためである．これらの結果から PMRL-OM が PMRL と同等あるいはそれを上回る性能を示していることがわかった．

検定 2 では，各ケースの 3 つの迷路における 30 試行のステップ数平均 12 個をそれぞれ検定する．これらのデータは特定の分布に従っているわけではなく (つまり，ノンパラメトリック検定がふさわしい)，これらのデータはお互いに関連している (同一シードでそれぞれの手法を比較する)．このことから，検定 2 もウィルコクソンの順位和検定が適している．そしてこの検定により，すべての迷路において PMRL-OM の性能が PMRL より同等もしくは上回っていることを確認する．p 値は 5.83e-3 を示している．この値は非常に小さいものであるため，すべての迷路において，PMRL-OM が PMRL よりも同等もしくはそれを上回る性能を示したことが統計的に明らかとなった．

5.6.4 考察：各ケースにおける分析

ここでは，実験 1 の各ケースにおける詳しい振舞いを分析し，考察する．図 5.12-5.15 の結果から，PMRL-OM が動的環境において良好な性能を示している．加えて，エージェントは協調行動を学習し最短ステップ数で全ゴールに到達する．この図において，環

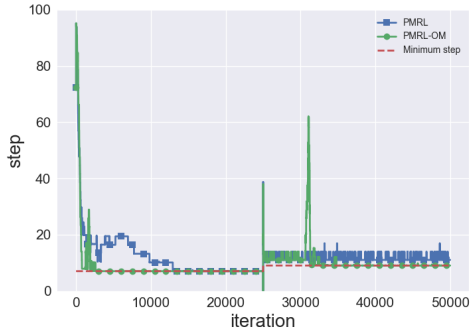


図 5.12　ケース (1-1) における結果

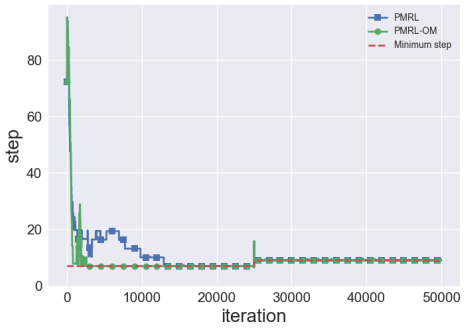


図 5.13　ケース (1-2) における結果

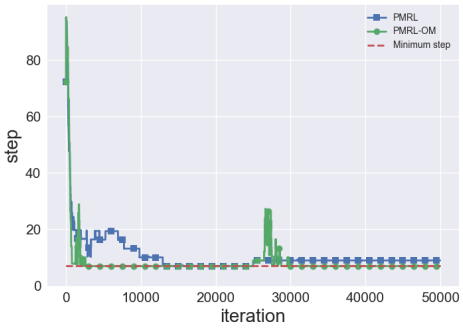


図 5.14　ケース (2-1) における結果

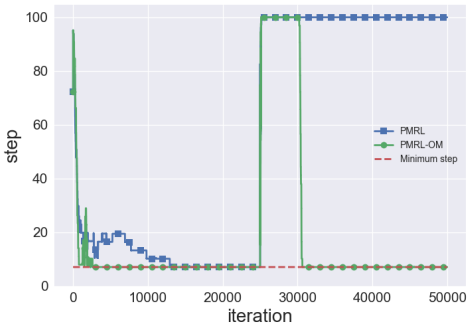


図 5.15　ケース (2-2) における結果

表 5.2　ウィルコクソン検定の結果.

	迷路 1	迷路 2	迷路 3
ケース 1-1	4.62e-8	3.42e-7	4.62e-8
ケース 1-2	NaN	4.62e-8	NaN
ケース 2-1	4.62e-8	4.62e-8	1.51e-6
ケース 2-2	4.62e-8	4.62e-8	4.62e-8

境変化前においても PMRL-OM は PMRL よりも性能が良い．この違いは目的価値の更新式が PMRL-OM と PMRL で異なるためである．つまり PMRL-OM は初期エピソードで獲得した最短ステップ数の影響を受けにくい．エージェントは学習初期に適切な最短ステップ数を保存する可能性は少ないため，PMRL は誤った目的価値を推定する事に対し,PMRL-OM は忘却関数により誤った目的価値を推定することを防止する．これは，PMRL-OM が PMRL よりも早く, 誤った最短ステップ数で計算された目的価値を修復できることを示唆している．したがって，PMRL-OM は，実験結果に基づいて環境が変化する前でも PMRL よりも優れたパフォーマンスを発揮した．また，PMRL-OM のパフォーマンスが偶然ではなく，目的価値の大小関係を適切に変更して学習できていること

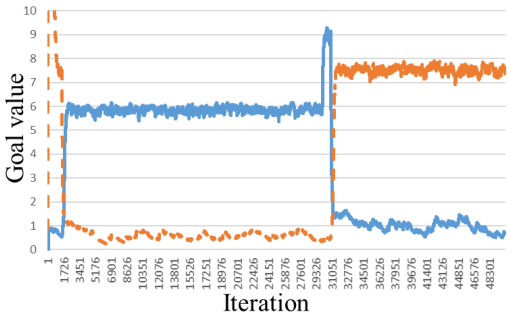


図 5.16　ケース (1-1)，エージェント A の目的価値推移

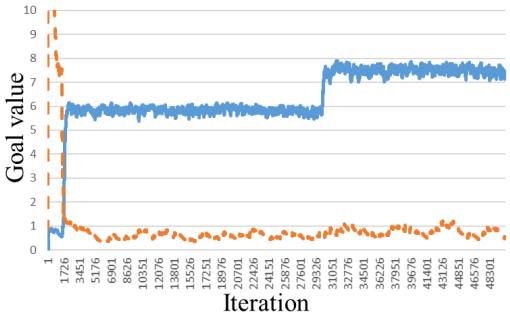


図 5.17　ケース (1-2)，エージェント A の目的価値推移

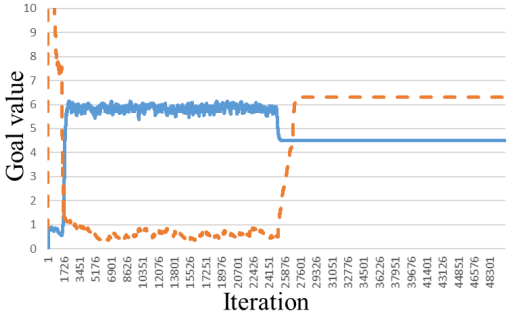


図 5.18　ケース (2-1)，エージェント A の目的価値推移

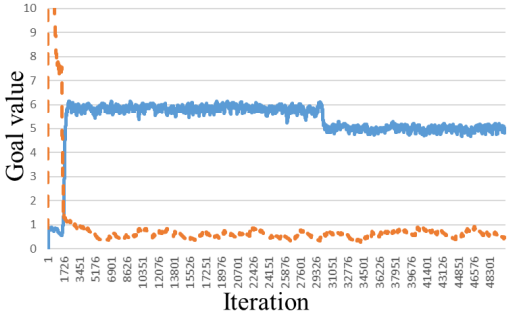


図 5.19　ケース (2-2)，エージェント A の目的価値推移

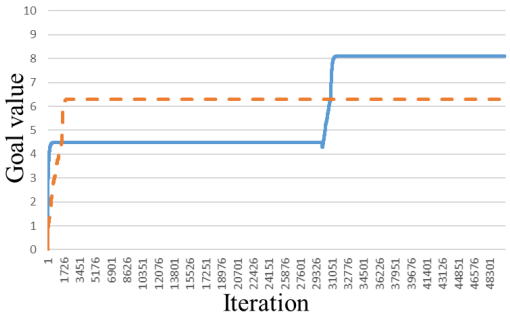


図 5.20　ケース (1-1)，エージェント B の目的価値推移

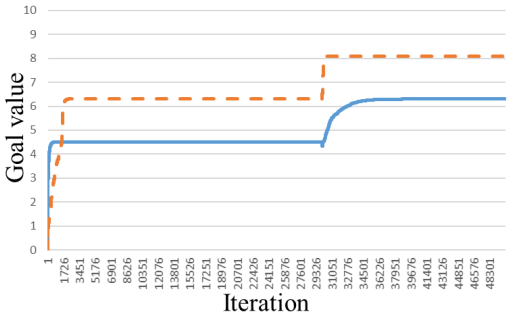


図 5.21　ケース (1-2)，エージェント B の目的価値推移

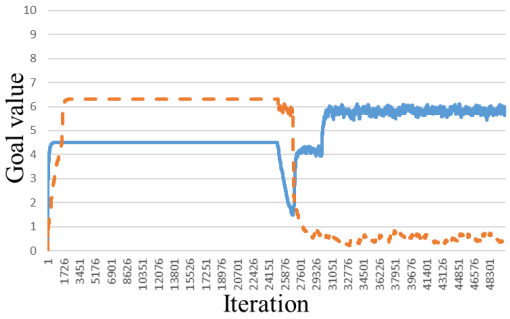


図 5.22　ケース (2-1)，エージェント B の目的価値推移

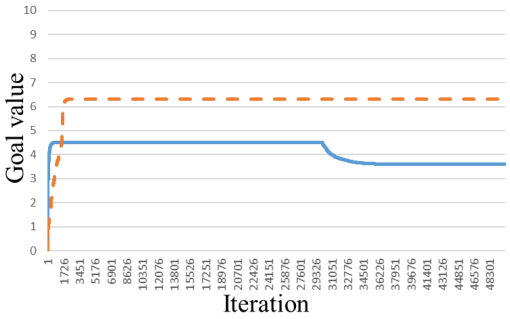


図 5.23　ケース (2-2)，エージェント B の目的価値推移

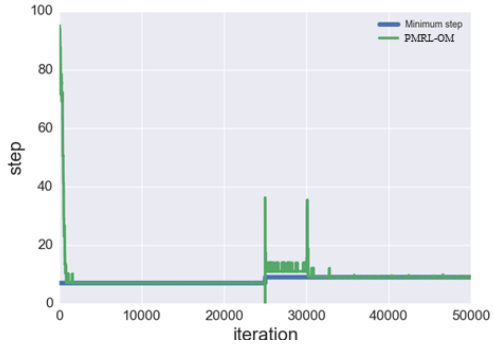


図 5.24　$\xi = 10$ の結果

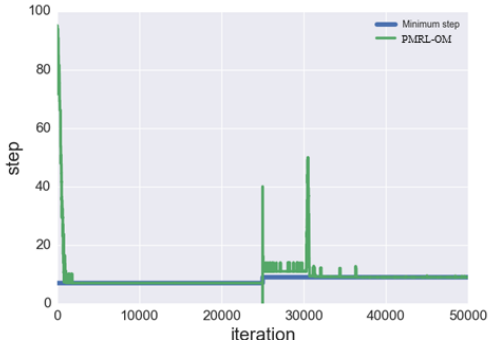


図 5.25　$\xi = 50$ の結果

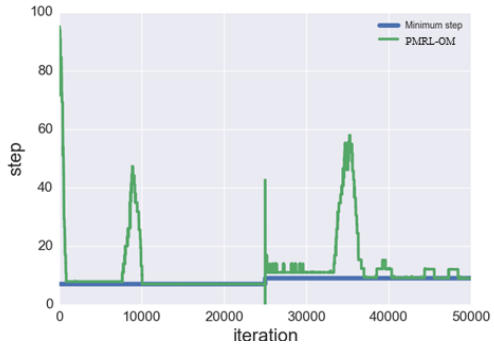


図 5.26　$\xi = 500$ の結果

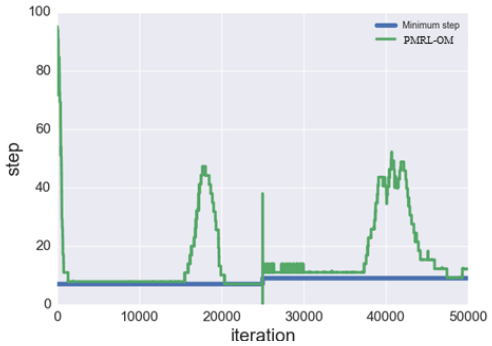


図 5.27　$\xi = 1000$ の結果

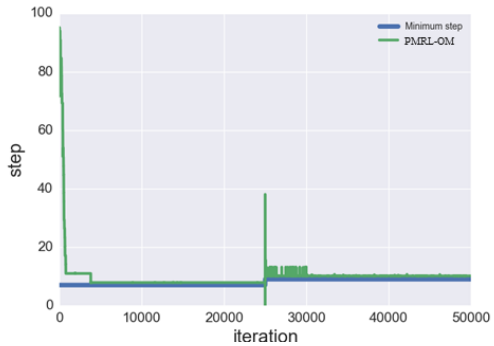


図 5.28　$\xi = 5000$ の結果

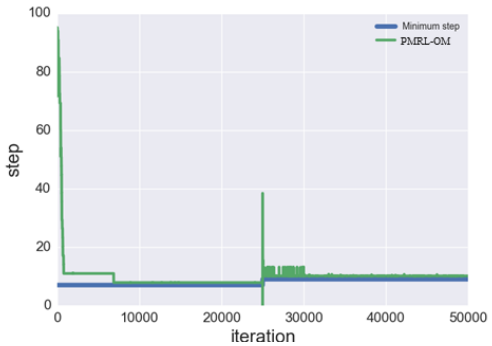


図 5.29　$\xi = 10000$ の結果

を図るため，図 5.16-5.23 は、すべてのケースにおける PMRL-OM のすべてのエージェントの目的価値を示している．この図の上部 4 つはエージェント A の目的価値を示し, 下部 4 つはエージェント B の目的価値を示している．この図では, 縦軸は目的価値を表し、横軸はエピソードを表します. 実線はゴール X の目的価値を表し, 点線はゴール Y の目的価値を表す．エージェントは, 内部報酬を設定して, 目的価値が最大となる目標に到達する．図 5.16-5.23 から，エージェントが目的価値の大小関係を変更でき，動的な環境に適用できることを示している．つまり，エージェントはケース 1-1，2-1 の環境変更後に異なる目標を達成するために学習する必要がある．一方で，エージェントはケース 1-2 およ

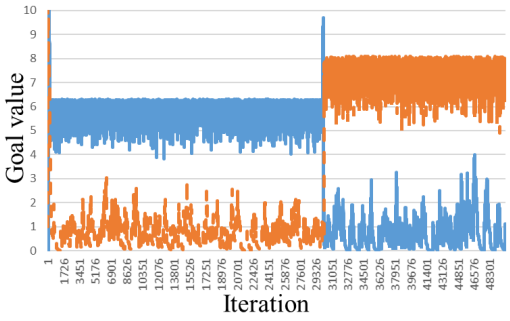


図 5.30　エージェント A の目的価値 ($\xi = 10$)

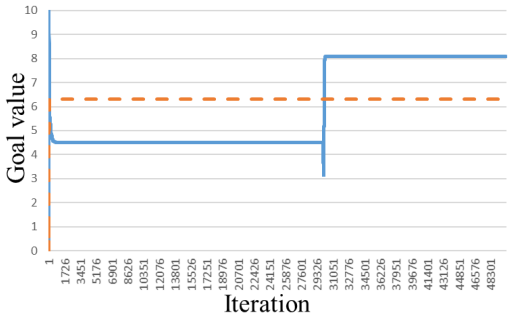


図 5.31　エージェント B の目的価値 ($\xi = 10$)

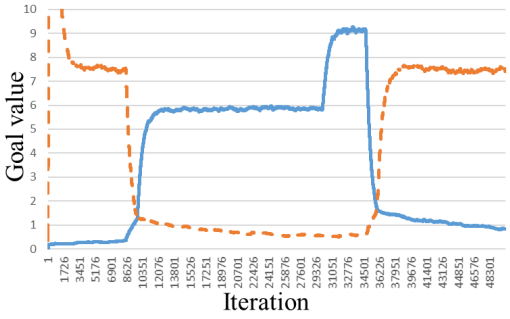


図 5.32　エージェント A の目的価値 ($\xi = 500$)

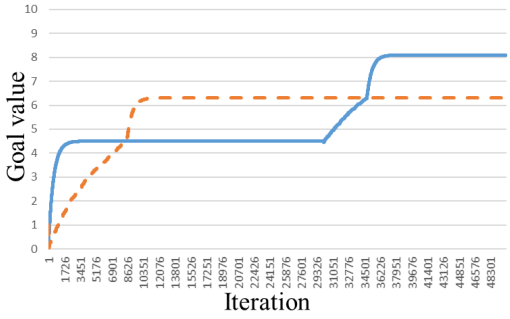


図 5.33　エージェント B の目的価値 ($\xi = 500$)

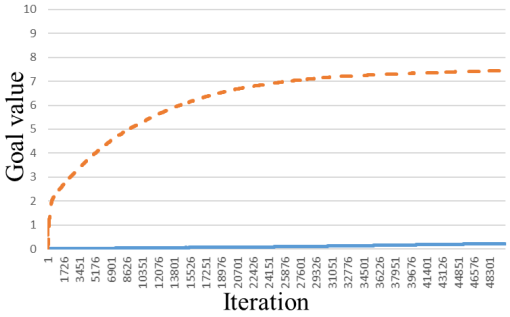


図 5.34　エージェント A の目的価値 ($\xi = 10000$)

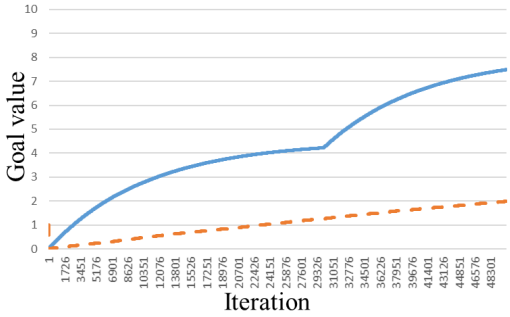


図 5.35　エージェント B の目的価値 ($\xi = 10000$)

び 2-2 では大小関係を維持する必要がある．これらの結果から，エージェントは目的価値を設定し，PMRL-OM を使用して環境変化に適応して学習できる．

- PMRL-OM と PMRL の比較
  全ケースの環境変化前, どちらの手法もエージェントに最短ステップ数でゴールに到達できるような協調行動を学習させる事ができる. そして PMRL-OM は環境変化後も最短ステップ数でゴールに到達するように機能するが，PMRL は変化後の環境に適応できない．特に，ケース 1-1, 2-1 ではステップ数が他の値に収束して

おり, ケース 1-2 では最短ステップ数に収束する．そしてケース 2-2 では結果のステップ数は 100 を示している．つまり PMRL はケース 1-2 にのみ適応可能であるといえる．言い換えれば，それはエージェントは目的価値の大小関係を変える必要がなく，そして環境変化前と同じゴールに到達する行動を学習すればよいためである．しかしながら，PMRL はケース 1-2 で良い性能を発揮しない可能性がある．エージェントは最短ステップ数を保存し，そしてこれらは環境変化前に保存したものと等しくなる．これはエージェントが PMRL により内部報酬をうまく計算できない可能性を示唆している．この実験ではこの問題による影響を小さくするに十分なほど δ が大きいので PMRL は性能が良い．一方で，PMRL-OM は環境変化後の最短ステップ数を保存可能であり，これは常にエージェントに最短ステップ数でゴールに到達するような協調行動を学習させることが可能である．他のケースでは，PMRL エージェントは PMRL-OM と異なり，協調のために適切なゴールに到達できない．上記の議論から，PMRL-OM は動的環境において PMRL よりも良い性能を示すことがわかる．

- 情報をどのように利用するのか

  図 5.24-5.29 はケース 1-1 において定数 ξ が異なる値であったときの結果である. 縦軸はエージェントの費やしたステップ数, 横軸はエピソード数である. 緑の線は PMRL-OM の結果を示し, 青い線は最短ステップ数を示している. この図において, 定数 ξ が大きいとき, ステップ数がある一定の値に収束することが難しくなる, 一方でもし ξ が 5000 を超えた場合, ステップ数は最短ステップ数ではないある一定の値に収束する. すなわち,PMRL-OM は環境変化に適応できない. 図 5.24-5.29 の結果から，ξ は 1000 未満の値でなければならない．一方で，図 5.30-5.35 は，ケース 1-1 で ξ が他の値になるときの目的価値を示している．図の上段はエージェント A の目的価値の振舞いであり，下段はエージェント B の目的価値の振舞いである．左から右の順に $\xi = 10, 500$，および 10000 の 3 種類のグラフがある．縦軸は目的価値を表し，横軸はエピソード数を表す．実線はゴール X の目的価値を表し，点線はゴール Y の目的価値を表す．$\xi = 10$ の場合，エージェントは適切な目的価値を設定できるが，エージェント A の目的価値は収束しない．それは ξ が小さく，目的価値が最新の更新の影響を大いに受けるためである．取得した情報のバランスが崩れている場合，エージェントは最適な目的価値を完璧に推定できない．$\xi = 10000$ の場合，エージェントは変化する前の環境に完全に対応することはできないが，変化した後の環境に対応することができる．$\xi = 500$ の場合，目的価値は特定の値に収束し，環境の変化に適用できる．$\xi = 10$ の場合，目的価値は収束できない．これはエージェント B の影響である．最短ステップ数は変わらないため，目的価値の関係は変わらない．これらの結果から，$\xi = 500$ が最適である．ξ の他の値の結

果は，$\xi = 500$ よりも良くなることはないが，すべてが悪いわけではない．変数 ξ は，目的価値を計算するために使用されるパラメーターであり，環境の変化前のステップ数の影響を受けられない．PMRL を使用した目的価値計算の問題は，学習が進行するにつれて，エージェントが目的価値に大きな値 (ゴール到達回数) で割ったステップ数を追加することである．PMRL-OM の目的価値の計算は，エージェントが学習の各反復で目的価値に常にステップ数を追加することである．これは PMRL-OM の重要な側面であり，ξ が正の値の場合に確立される．このため，ξ は敏感ではないが，ξ が巨大な値である場合，目的価値の計算は，図 5.24-5.29 に示すように，PMRL-OM と PMRL の間でほぼ同じになる．そして，ξ は 500 未満であるべきである．加えて，図 5.30-5.35 の結果から，$\xi = 10000$ の結果から，1 つのゴールの目的価値はすべての反復を通じて常に他のゴールの目的価値よりも大きいため，ξ は 500 未満であるべきである．PMRL-OM は $\xi = 1000$，$\xi = 5000$，および $\xi = 10000$ で性能が低下するが，PMRL-OM はほとんどすべての試行で最適な結果を取得できる．つまり，PMRL-OM は特定のシードではパフォーマンスが低下する．ξ の異なる値の間の性能の違いは，学習反復の回数とともに発生する．これは，反復回数が多い場合，ξ が大きい値になりうることを示唆している．それ以外の場合は，小さい値にする必要がある．ただし，小さな反復の状況で ξ が大きな値であっても，性能は最悪にならない．エージェントは，変化前の環境に完全に対応することはできないが，図 5.30-5.35 の変更後の環境に対応可能であることに注意されたい．また今回は最短ステップ数を計算する上でのメモリ長を 5000 に設定しているが，これは 5000 よりも小さい値でも機能する．ただし，メモリ長が小さくなればなるほど参照するステップ数の量が減るため，完成した最短ステップ数が小さくなる．今回はグリッドワールドであり，最適方策であればほぼ最短ステップ数が得られるため，5000 よりももっと小さくても良いが，ここでは環境変化後のエージェントの Q 値がある程度収束するであろうステップ数 5000 をメモリ長としている．

5.7 実験 2: 断続的環境変化への適用

5.7.1 実験内容

次に環境変化が一度ではなく断続的に発生する際の PMRL-OM の有効性を検証するため，あるエピソード数毎にランダムに変化する環境において PMRL-OM の追従性能を検証する．具体的には 1000 エピソード毎と 10000 エピソード毎にランダムに変化する環境を用意する．なおこの際変化する環境は 1000 エピソード毎も 10000 エピソード毎も

等しいものとする．また，比較する手法は PMRL-OM の他に，Profit Sharing (PS)[27] と Profit sharing with expected failure probability (PS with EFP)[43] を取り上げ，PS と PS with EFP の報酬関数は式 (5.12) に示す通り「エージェント全体の獲得報酬値/全エージェントのゴール到達までのステップ数」とする (このままではエージェント全体の情報が扱える分 PS 系手法が有利であるが，今回はその PS と性能を比較する．).

$$R_i^{ps} = \left\{ \frac{r}{N} | s_i, s_{-i} \in S_{goal} \wedge s_i \neq s_{-i} \right\} \tag{5.12}$$

5.7.2 評価基準とパラメータ設定

本実験は各迷路で乱数のシードを変更した 30 試行実験を行い，すべてのエージェントが最短ステップ数でゴールに到達した割合と獲得報酬値の合計を評価する．なお，学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じように学習反復を行い，その時の到達ステップ数を評価する．

表 5.3 に実験パラメータを示す．このとき学習エピソード数が 1000 毎に変化する場合は 50000 回，10000 毎に変化する場合は 500000 回に設定する (1, 2 行目)．また 1 学習における最大のステップ数を 100 に設定する (3 行目)．また学習において初期 Q 値は 0(4 行目)，学習のパラメータをは学習率 α が 0.1(5 行目)，割引率 γ は 0.9 に設定する (6 行目)．そして報酬値は 10 に設定する (7 行目)．また，PMRL の内部報酬設定のための定数 δ は 10 とし (8 行目)，PMRL-OM のパラメータ ξ は 300 とする (9 行目)．PS は本論文にて紹介した報酬関数を使用する．

表 5.3　断続的環境変化実験におけるパラメータ

	PMRL-OM+	PMRL-OM	PS with EFP	PS
エピソード数	50000, 500000			
環境変化点	1000 毎, 10000 毎			
最大ステップ数	100			
初期 Q 値	0			
学習率 α	0.1			
割引率 γ	0.9			
報酬値	10			
定数 δ	10		なし	なし
定数 ξ	300		なし	なし

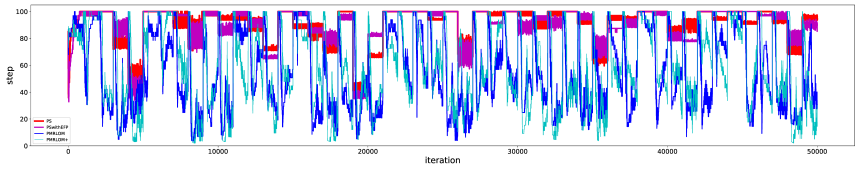


図 5.36　1000 毎に断続変化する環境上の到達ステップ数の比較

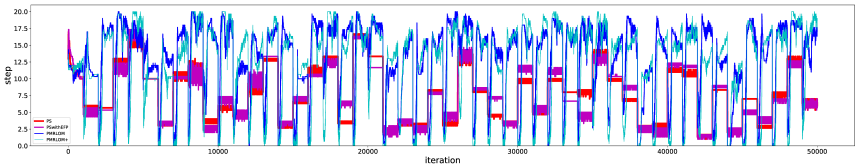


図 5.37　1000 毎に断続変化する環境上の合計獲得報酬値の比較

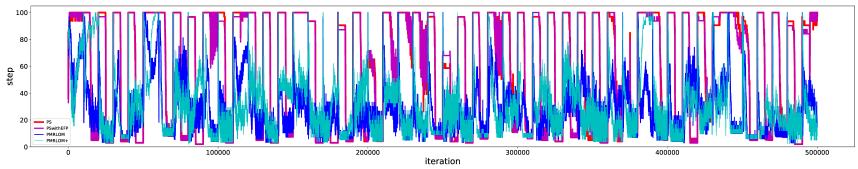


図 5.38　10000 毎に断続変化する環境上の到達ステップ数の比較

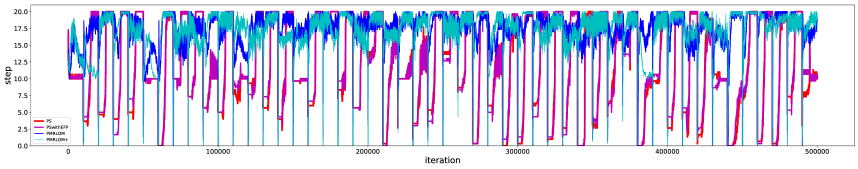


図 5.39　10000 毎に断続変化する環境上の合計獲得報酬値の比較

5.7.3　実験結果

1000 ステップ毎に変化する環境上での実験結果を図 5.36，5.37 に，10000 ステップ毎に変化する環境上での実験結果を図 5.38，5.39 にそれぞれ示す．図 5.36，5.38 では縦軸がエージェント全体がゴールに到達したときのステップ数，横軸がエピソード数である．一方で図 5.37，5.39 では縦軸がエージェント全体が獲得した報酬値の合計，横軸がエピソード数である．また青線，赤線がそれぞれ PMRL-OM と PS の結果を示している．

図 5.36 の結果から，PMRL-OM は環境変化後一瞬でステップ数を小さくすることができているが，PS は一定のステップ数に収束するかステップ数 100 で収束している．図 5.38 の結果では一定数の学習数後にステップ数を縮めることができていることを見ると，1000 回毎の学習変化のような短期間での動的変化に対して PS は弱く，PMRL-OM は強いことが分かる (なお図 5.38 では，一定エピソード後は PS が上回っているように見えるが，実験設定では PS が有利な設定であり，むしろ不利でありながら PMRL-OM の結

果はステップ数が早々に小さくなり，それぞれのステップ数も PS を上回るものも存在する．)．一方で図 5.37 では PMRL-OM の獲得報酬値が PS を上回っている．これは PS が PMRL-OM よりも環境変化に追従することができないことを示している．現に図 5.39 を見ると，PS の獲得報酬値は環境変化直後に落ち，その後多くのエピソード数を掛けて獲得報酬値が上昇することがわかる．一方で PMRL-OM では瞬時に獲得報酬値が上昇しており，動的変化に対する追従性の良さがうかがえる．

5.7.4 考察

以上から，PMRL-OM+ および PMRL-OM は断続的環境変化に対して頑健であることがわかる．また以降では断続的環境変化に対して必要な要素を探究するとともに，各手法の有効性を議論する．

・PMRL-OM 系手法と PS 系手法の比較

図 5.36 の結果から，PMRL-OM 系手法は環境変化後一瞬でステップ数を小さくすることができているが，PS 系手法は一定のステップ数に収束するかステップ数 100 で収束している．図 5.38 の結果では一定数の学習数後にステップ数を縮めることができていることを見ると，1000 回毎の学習変化のような短期間での動的変化に対して PS 系手法は弱く，PMRL-OM 系手法は強いことが分かる (なお図 5.38 では，一定エピソード後は PS 系手法が上回っているように見えるが，実験設定では PS 系手法が有利な設定であり，むしろ不利でありながら PMRL-OM 系手法の結果はステップ数が早々に小さくなり，PS 系手法を上回るものも存在する．)．一方で図 5.37 では PMRL-OM 系手法の獲得報酬値が PS 系手法を上回っている．これは PS 系手法が PMRL-OM 系手法よりも環境変化に追従することができないことを示している．現に図 5.39 を見ると，PS 系手法の獲得報酬値は環境変化直後に落ち，その後多くのエピソード数を掛けて獲得報酬値をあげていることがわかる．一方で PMRL-OM 系手法では瞬時に獲得報酬値を上げており，動的変化に対する追従性の良さがうかがえる．

・PS 系手法内の比較

PS with EFP と PS を比較すると，細かな違いはあるものの結果に大きな差が無いことがわかる．特に結果の収束時点は等しく，追従性に関しては EFP などは寄与の度合いが大きくないことがわかる (ただし，PS with EFP は本来 Robocup Soccer の Keepaway タスクにて有効性を示されているため，迷路問題により元来想定した効果を損なっていることも考えられる)．特に今回同一のゴールに到達することを罰則としているが，結果が 100 を示している (同一のゴールに到達している)．図 5.40 は 2000 エピソード時点の環境変化を示している (1000 毎の環境変化では PS 系手法はどちらもこれに追従できていな

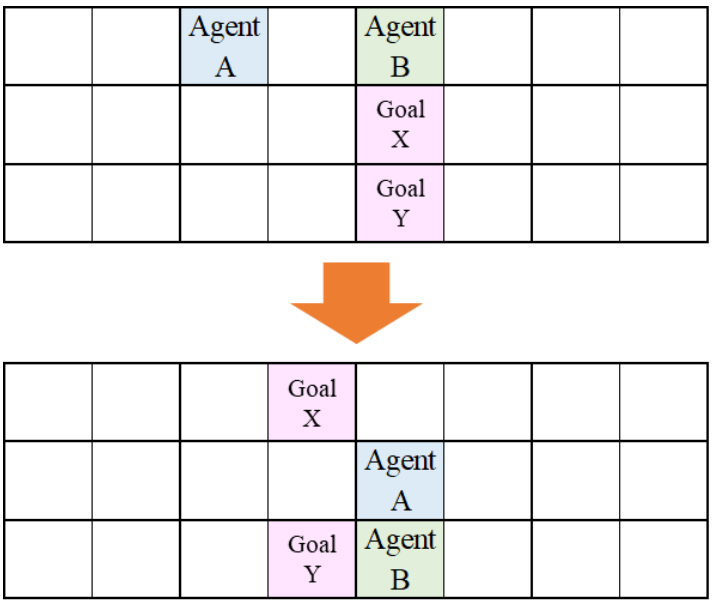


図 5.40　2000 エピソード時点の環境変化

い)．これを見ると環境変化前においてはどちらも下に行く行動が適切であるが，変化後は上に行く行動が適切であると分かる．PS with EFP では主に Q 値を用いたソフトマックス行動選択であり，罰則を得るような行動はその確率を下げるように設定しているものであるため，Q 値の影響が大きくなる．そのためこのケースにおいてはもともと Q 値の大きな下の行動の影響により，環境変化後の罰則が上手く機能していないことが考えられる．以上から EFP のような Advising による断続的環境変化の追従は難しく，ある程度同じ環境であり十分なエピソードが必要となることがわかる．

・PMRL-OM 系手法内の比較

PMRL-OM+ は 10000 エピソード毎の環境変化では PMRL-OM よりも比較的性能がよろしくないが，1000 エピソード毎の環境変化では比較的性能が良いことから，環境変化の追従性が上がっていることがわかる．また，PMRL-OM は窓長 e の設定が必要となり，特に環境変化のタイミングに合わせて変更していくことが必要となるため (例えば 1000 毎の環境変化では少なくとも $e < 1000$ でなければならない)，実際は PMRL-OM+ の方が種々の環境変化に対して追従可能であるといえる．特に実際の問題では環境変化のタイミングを事前に知ることはほぼ不可能であると言えるため，窓長を指定せずに PMRL-OM とほぼ同等の性能を示す PMRL-OM+ の有効性は十分にあるといえる．

5.8　5 章のまとめ

5.8.1　理論的証明の限界

PMRL-OM は PMRL の理論的証明を崩すことなく動的変化へ適用する手法であり，実験 1 において環境変化として想定しうる 5 種類の動的変化に追従して最短ステップ数で全エージェントがゴールへ到達していたことから，PMRL-OM は理論的証明を維持することができていることがわかる．なお環境変化前のステップ数を見ると PMRL と PMRL-OM で振る舞いが異なっているが，PMRL の示した理論的証明は下記の 2 つであり，それが学習中の振舞いも正確に制御するものではない．

1. 目的価値によりエージェントはお互いに異なるゴールへ到達する行動を学習
2. 各ゴールの到達ステップ数が互いに異なるとき 2 体エージェントにおいて最短ステップ数でゴールに到達する行動を学習

それは最終的にエージェントは互いに異なるゴールへ到達し，各ゴールへの到達ステップ数が異なる 2 体のエージェントにおいて最短ステップ数でゴールに到達する行動を獲得可能であるというものである．以上を踏まえると，PMRL-OM は環境変化前も後も PMRL と同様の条件であれば適切に学習することができるといえる．

5.8.2　情報利用法の限界

PMRL-OM には最短ステップ数の利用法 (情報の利用法) が重要であり，具体的には窓長 e とパラメータ ξ の設定が重要な役割を担う．窓長とパラメータの関係については前述の通り $e = \xi$ であれば目的価値は最短ステップ数の窓長 e の範囲の平均値となり，PMRL と同様の結果となる．そして環境変化が起これば窓長 e 内の情報が全て新しい環境のものになるまで環境変化に追従することはできない．またパラメータ ξ は最短ステップ数の平均値をとる際に利用するデータ数を示すものであり，大きくなればなるほどより初期に学習した時の最短ステップ数を利用していることになる．パラメータ ξ は環境変化ということを想定すれば，環境変化後に可能な学習回数よりも小さく設定することが必要となる．そして窓長 e も環境変化後に可能な学習回数よりも小さく設定することが必要となる．一方で今回実験 2 の結果を見ても，PMRL-OM の窓長 e による最短ステップ数推定は実装上必要ないことがわかっている．この理由は目的価値がステップ数の平均値であり，ステップ数の大きなゴールへ到達するように学習するという性質にある．仮に純粋なステップ数を目的価値の推定式に代入すれば，到達したことのないゴールはまだステップ数が大きいため優先して到達するようになり，優先して到達すればそのゴール到達まで

の Q 値が収束し，必然的に最短ステップ数が目的価値の推定式に代入されるようになるということである．そのため，PMRL-OM の最短ステップ数の導出がなくとも協調行動を学習することはでき，窓長 e は必要がない．しかしながら，最短ステップ数を使わなければ PMRL-OM は PMRL の理論的証明を維持することができないという問題点がある．また最短ステップ数の利用を停止するとパラメータ ξ の設定法がより難しくなる．とはいえ基本的に環境変化後に可能な学習回数よりも小さく設定することに変わりはなく，PMRL-OM よりも学習回数を必要とするところが異なる．

5.8.3 適用する環境変化の限界

実験 2 の結果から，PMRL-OM および PMRL-OM+ は断続的に変化する環境であってもその一つ一つに追従することが可能であり，環境変化に対して従来法よりも素早く適応可能であることが判明した．それは他の手法が環境変化により変化する報酬値から再度学習し直すのに対し，PMRL-OM や PMRL-OM+ はより遠くのゴールを目指すという方針が変わらず，変化する環境に対しても今までの学習の中で利用できる部分を利用して学習をやり直すよりも素早く追従できていることがわかる．ただし実験 2 の 10000 エピソード毎に変化する環境では PS と PS with EFP に負けており，完全に適切な行動を学習するためにはある程度の学習回数が必要である (エージェント数が増える場合はその限りではない．)．

第 6 章

Profit minimizing reinforcement learning for dynamic change of rewards and environmental states (PMRL-DRES)

6.1　アプローチ

エージェント数とゴール数の動的環境に対するアプローチを図 6.1 に示す．図の上段から中段に向かって環境変化し，下段はそのときの対応を示している．ここでは環境変化として障害物の発生及びエージェント数とゴール数の変化を示している．この環境変化に注目すると，エージェント A，B とゴール X，Y は位置関係共に変化しておらず，発生したエージェント C とゴール Z がお互いに近くに存在するため，ここではエージェント A，B とエージェント C の学習環境を下段のように切り分けて，エージェント A，B においては環境変化前の学習結果をそのまま継続し，エージェント C には新たな環境を学習することで，全体として動的変化に適応して協調行動を学習する．以上が動的環境に対する本研究のアプローチである．ここで特に重要なポイントは学習環境の分割によりエージェント同士の相互作用により決定することで，新たな目的に向かう必要がある動的変化に対して適応すること (エージェント数やゴール数における動的変化に対する効果) である．

図 6.2 は PMRL-DRES の想定する環境変化である．具体的には，3 章における，報酬値の変化およびエージェント数，ゴール数の変化である．このとき，目的数やエージェント数，報酬値が変化することによって PMRL や PMRL-OM は異なる戦略で学習することが求められる．しかしながら PMRL や PMRL-OM は単一の戦略でのみ学習が可能であるため，適切に学習できない．そこで，PMRL-DRES では学習範囲を制限して，PMRL

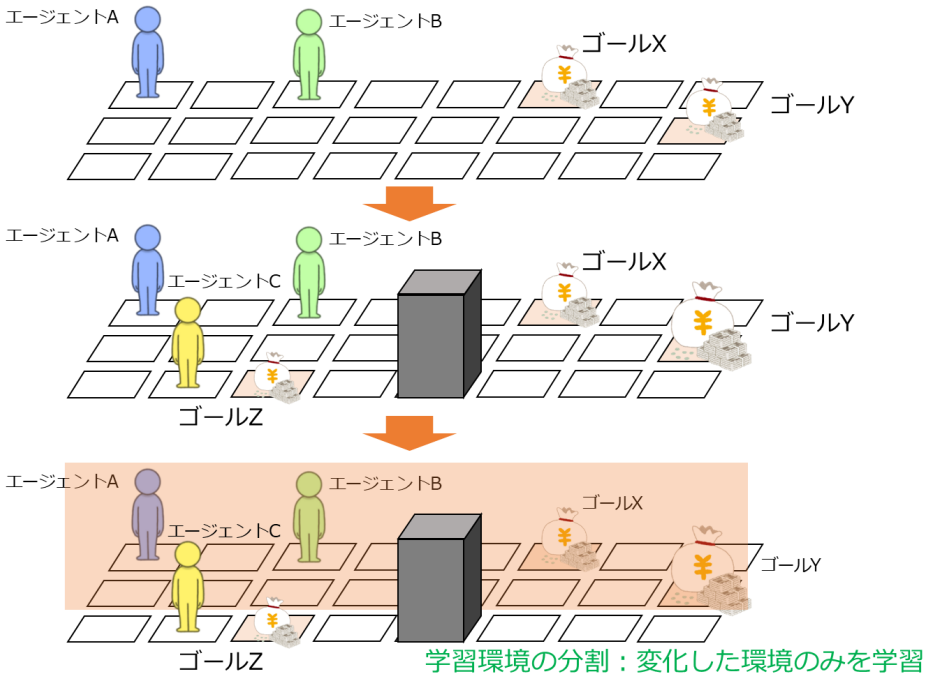


図 6.1　動的環境におけるアプローチ

や PMRL-OM の学習戦略で協調行動を学習可能とする．具体的には，標準ステップ数を用いた学習範囲の制限と，獲得報酬期待値を用いた学習範囲の制限を組み合わせることで，学習環境を PMRL の保証する範囲に分割し，報酬値とエージェント数，目的数の動的変化に追従し，協調行動を学習する．なお PMRL-DRES も PMRL と同様にして，学習初期はまだ到達していないゴールも存在するため，到達しているゴールの中から最大ステップ数となる (最も遠くの) ものを目指して学習する．

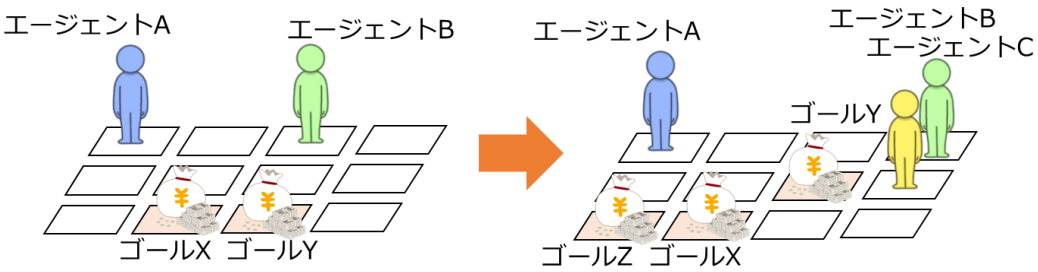


図 6.2　PMRL-DRES の想定する環境変化

6.2　アーキテクチャ

PMRL-DRES において，アーキテクチャとしての PMRL-OM との違いは，獲得報酬期待値 er，標準ステップ数 st，ゴール到達可能範囲 gr を新たにメモリ部に持ち，実行部にゴール到達可能範囲更新というプロセスを持つ点が異なる．図 6.3 は PMRL-DRES エージェントのアーキテクチャを示している．エージェント内部は大きくメモリ部と実行部に分かれており，メモリ部には Q 値など学習に必要な変数値が格納されており，実行部は PMRL のアルゴリズムを実行するプロセスが格納されている．具体的にメモリ部は環境の全状態と全行動に対する Q 値を示す Q テーブルと，各目的達成までの最短のステップ数と，協調のために達成すべき目的を示す価値である目的価値，そして内部報酬と PMRL-DRES の新たなメカニズムを実行する上で必要な変数である獲得報酬期待値と標準ステップ数，およびゴール到達可能範囲を持つ．実行部は Q 学習において実行される状態観測，行動選択，Q 値の更新の他に，最短ステップ数更新，ゴール到達可能範囲更新，目的価値更新と内部報酬設定のプロセスを持つ．

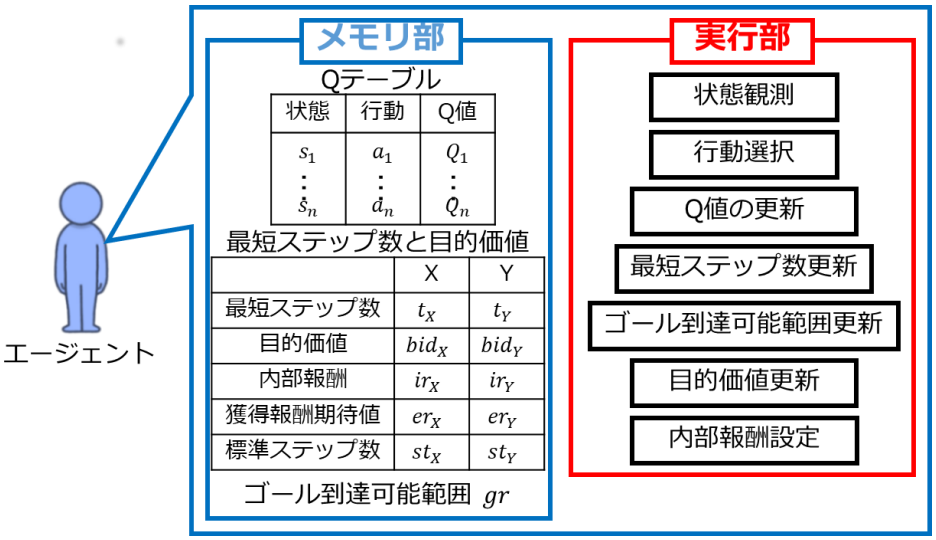


図 6.3　PMRL-DRES エージェントのアーキテクチャ

6.3　メカニズム

6.3.1　報酬値の動的変化対策：標準ステップ数

標準ステップ数とは各ゴールの報酬値から割引期待報酬値を計算した際に，その値が等しい状態へ到達するために必要なステップ数を疑似的に割り出したものである．標準ステップ数は式 (6.1) により計算される．st_g^i は標準ステップ数，t_g^i はエージェント i が

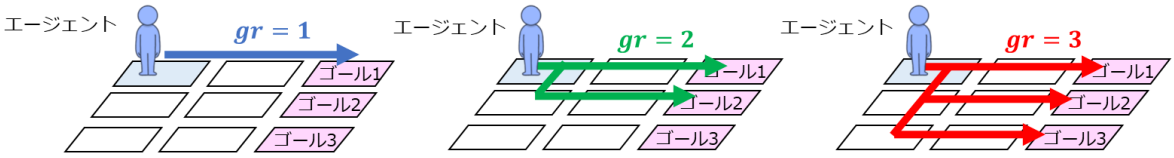


図 6.4　ゴール到達可能範囲

ゴール g へ到達するまでの最短ステップ数，r_g は任意のゴール g へ到達した際得られる報酬値，r_{stan} は基準となる報酬値を表す．エージェントは各ゴールの報酬値が r_{stan} の時，現状のステップ数からどれだけ変化するのかを計算する．そのため，r_{stan} は適当な値でよい（本研究では $r_{stan} = 10$ とする．）PMRL-DRES はこの標準ステップ数に基づき，目的価値設定を行う．

$$st_g^i = t_g - \log_\gamma \frac{r_{stan}}{r_g} \tag{6.1}$$

具体的にエージェントはまず，ステップ数と報酬値から標準ステップ数を求める．そして，標準ステップ数の最も小さいゴールからエージェント数分のゴールのみから適切なゴールを選択し，目的価値を推定する．そうすることで，もし報酬値が等しくゴールがエージェント数以上に存在していた場合，エージェントに最も近いゴールへ到達することができる．加えて，報酬値が異なる場合であっても，報酬値が大きいほど標準ステップ数が小さくなるため，報酬値の大きさとステップ数を考慮したゴールへ到達することができる．

6.3.2　目的数の動的変化対策：ゴール到達可能範囲

標準ステップ数を計算した後，その値に基づきゴール到達可能範囲 gr を設定する．ゴール到達可能範囲は，最短ステップ数に基づき学習エージェントの最も近いゴールから gr 個のみ到達できるようにする．図 6.4 はゴール到達範囲の違いによるエージェントの振舞いの違いを示したものである．このとき 3×3 の迷路にエージェント 1 体と 3 個のゴールが存在し，$gr = 1$ のとき最も近いゴール (ゴール 1 のみ)，$gr = 2$ のとき最も遠いゴール以外のゴール，$gr = 3$ のとき全ゴールに到達可能となる．PMRL-DRES では，ステップ数に代わり標準ステップ数の小さいゴールから到達可能とする．以上のように標準ステップ数に基づいたゴール到達可能範囲の制限により，ゴール数の比率が大きい動的変化に追従可能となる．

6.3.3　エージェント数の動的変化対策：獲得報酬期待値に基づく学習分割

エージェント数，ゴール数の動的変化に追従するために，学習範囲を分割することを考える．図 6.5 は学習分割による効果を示している．図の上段は環境変化前，中段が PMRL-OM の結果で下段は学習範囲の分割後の結果である．また吹き出しはエージェントの目的価値を示している．このとき，環境変化により通常の PMRL-OM ではエージェントを最短ステップ数でゴールに到達させることができない (図の中段)．このとき，エージェント A，B をゴール Y に到達しないように制限し，エージェント C をゴール Z に到達しないように制限する (つまりエージェント A，B がゴール X，Z に対して PMRL-OM で学習させ，エージェント C がゴール X，Y に対して PMRL-OM で学習させる) ことで全エージェント最短ステップ数でゴールに到達する (図の下段)．以上のように，各エージェントが到達ゴールを制限し，各エージェント毎個別に PMRL-OM で学習することで (学習範囲の分割)，協調行動を学習することがここでの狙いである．ここで注目すべきは，制限したゴールが対象のエージェントにとって遠くにあり，他エージェントが近くにいることである．以上を加味するため，PMRL-DRES では下記の獲得報酬の期待値というものを定義し，その大きさにより制限するゴールを選択する．式 (6.2) は獲得報酬の期待値の計算式である．er_g が獲得報酬の期待値であり，n_g, r_g はゴール g への到達数と報酬値である．r_i は任意のエージェント i の獲得した報酬値を示している．$fr(m)$ は m 回目にゴールに到達したとき，報酬を獲得していれば (最初にそのゴールに到達していれば) 報酬値を，そうでなければ 0 を返す関数である．この獲得報酬期待値が大きければ最初に到達することが難しく，遠くに存在するか近くに他のエージェントが存在することが間接的に表現できている．PMRL-DRES はこの期待値に閾値を設け，それ以下であればそのゴールに到達しない (内部報酬を設定しない) ようにすることで，協調行動を学習する．

$$er_g = \frac{1}{n_g} \sum_{m=1}^{n_g} fr(m), \tag{6.2}$$

$$fr(m) = \begin{cases} 0 & if \quad r_i = 0 \\ r_g & otherwize \end{cases} \tag{6.3}$$

6.4　アルゴリズム

PMRL-DRES のフローチャートを図 6.6 に示す．PMRL-DRES でも PMRL と同様に，エージェントがまず Q 学習と同様に状態を観測し，行動を選択することで次状態に遷移して，報酬を獲得する (図 6.6 の 1-4)．その後も同様に，PMRL-DRES のエージェ

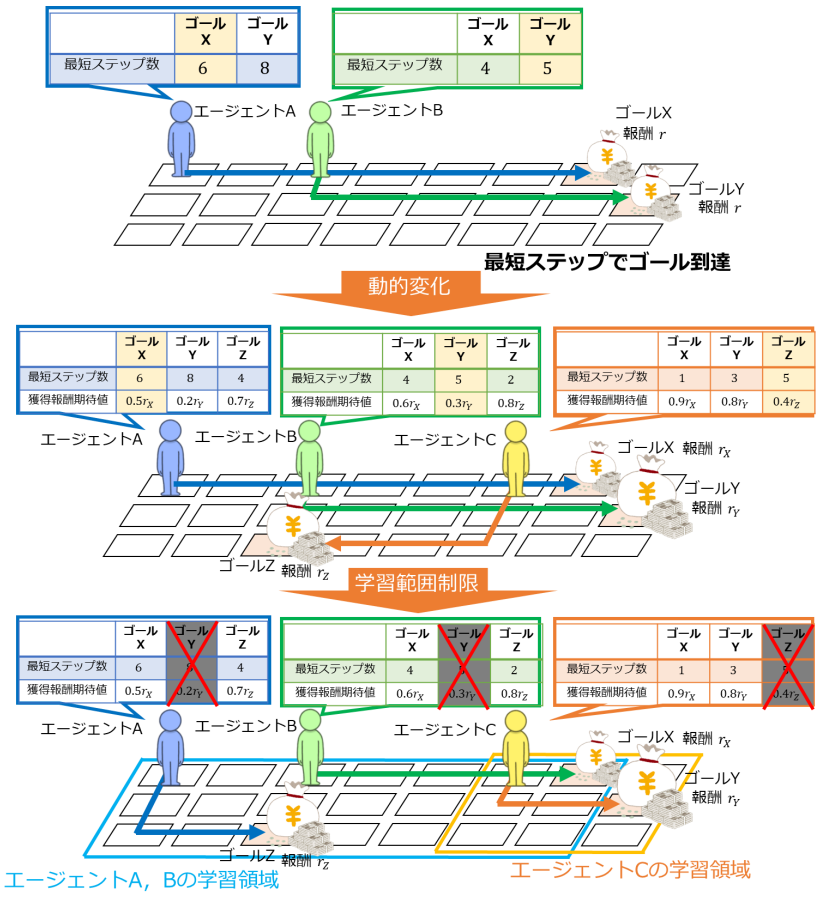


図 6.5　獲得報酬期待値に基づく学習範囲の分割

ントはゴールに到達したか否かによって処理が異なり，ゴールに到達していないのであれば獲得した報酬に基づき Q 値を更新してまた次のステップを繰り返す (図 6.6 の 11)．一方でゴールに到達しているのであれば，そのゴール到達までのステップ数が今までそのゴールへ到達したステップ数の中で最小であれば更新する (図 6.6 の 6)．その後獲得した報酬値から獲得報酬期待値を計算し，その後標準ステップ数からゴール到達可能範囲を設定する (図 6.6 の 7, 8)．ここは PMRL-DRES が PMRL とは異なる部分である．図 6.6 の 8 以降は PMRL と同様にして目的価値というものを更新し，選択した目的からその選択したゴールに到達できるように更新された最短ステップ数から内部報酬を設定する (図 6.6 の 10)．またその際に，各ゴールに到達した際に得られる報酬値から，ゴール到達可能範囲 gr を設定する (図 6.6 の 9)．以上をステップ数やエピソード数が閾値を超えるまで繰り返して学習する (図 6.6 の 12)．以上が PMRL-DRES のアルゴリズムフローの全貌である．

PMRL-DRES のアルゴリズムを Algorithm6 に示す．PMRL-DRES の大まかなアルゴリズムは PMRL-OM と等しく，Algorithm4 と比較して，獲得報酬保存用配列 $\boldsymbol{mr_g}$ と獲得報酬期待値の閾値 $Thresholdmr_g$ が新たに増えて，アルゴリズムは 25 行目が異な

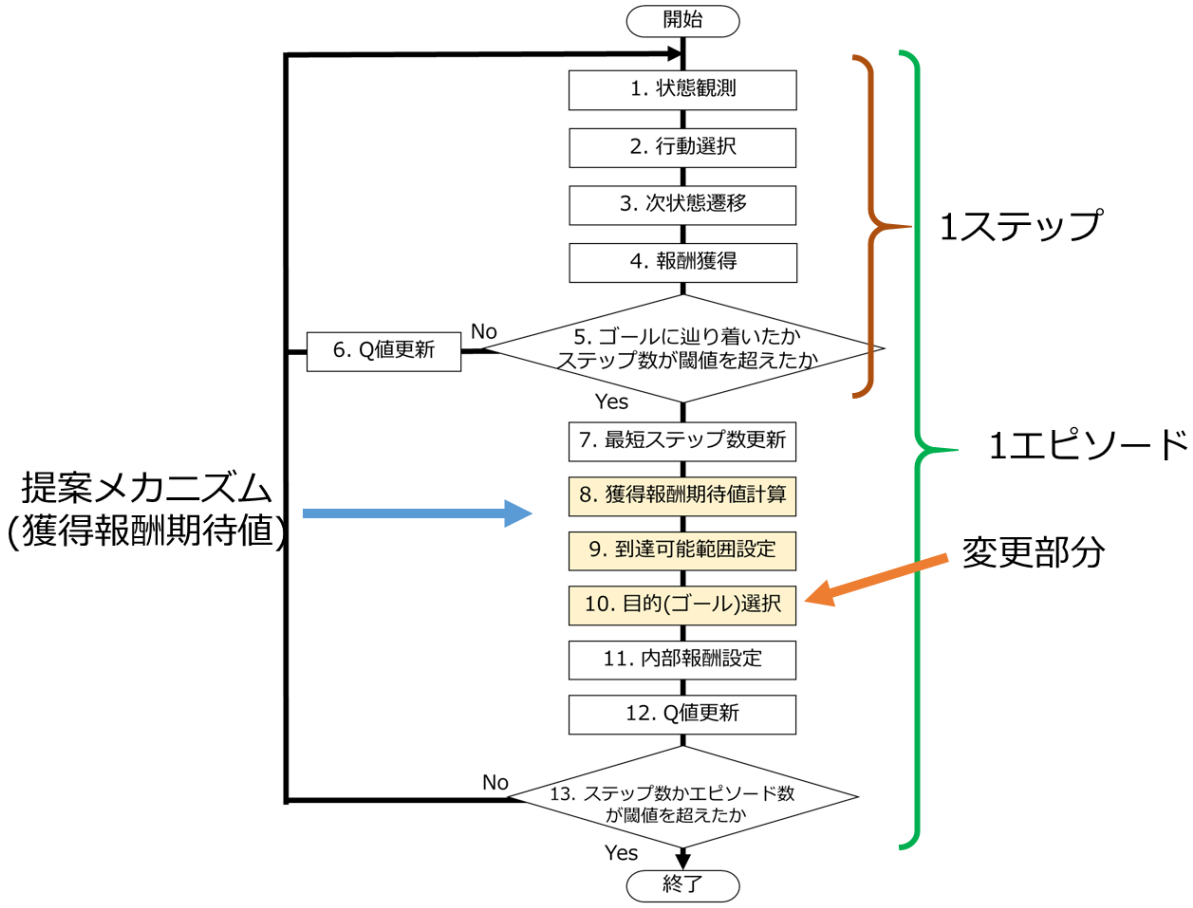


図 6.6　PMRL-DRES のアルゴリズムフロー

る．25 行目は BID 関数であり，Algorithm8 で示される．BID 関数は目的価値を推定する関数である．

BID 関数は選択ゴール変数 $selection$ と目的価値配列 $\boldsymbol{bid}$，獲得報酬保存配列と獲得報酬期待値の閾値 $\boldsymbol{mr_g}$ と $Thresholdmr_g$ を入力として，$selection$ の値を決めるため出力は無い．BID 関数では，まずフラグ配列 $\boldsymbol{flag}$ を設定する．このとき $\boldsymbol{flag}$ の要素はすべて $False$ とする (1 行目)．次にすべてのゴールに対して標準ステップ数を計算し，配列 $\boldsymbol{st}$ に格納する (2 行目)．その後，フラグ配列 $\boldsymbol{flag}$ が False を示しかつ配列 $\boldsymbol{st}$ の要素が最も小さくなるゴールを選び出し，そのゴールを添え字としたフラグ配列 $\boldsymbol{flag}$ を True に置き換える．これをエージェントの数だけくりかえす (3-7 行目)．そして配列 $\boldsymbol{bid}$ の値が最大となり，$\boldsymbol{flag}$ が True である引数 g を $selection$ に格納する (8,9,10 行目)．その後一定の確率で $selection$ を $\boldsymbol{flag}$ が True を示すゴールの中からランダムに置き換える (11-15 行目)．以上が PMRL-DRES のアルゴリズムである．

Algorithm 6 PMRL-DRES

1. $Q(s,a)$ を 0 で初期化，$\forall s \in S, \forall a \in A$
2. ステップ数保存用配列 $\boldsymbol{mt}$ と最短ステップ数配列 $\boldsymbol{t}$ を $MaxStep$ で初期化
3. 獲得報酬保存用配列 $\boldsymbol{mr_g}$ を 0 で初期化，獲得報酬期待値の閾値 $Thresholdmr_g$ の設定
4. 目的価値 $\boldsymbol{bid}$ と内部報酬 $\boldsymbol{ir}$ を 0 で初期化
5. 初期状態 s_{start} とゴール集合 $\boldsymbol{s_{end}}$ の設定
6. **for** $iteration = 1$ to $MaxIteration$ **do**
7. $\quad s = s_{start}$
8. $\quad$ **for** $step = 1$ to $MaxStep$ **do**
9. $\quad\quad$ 行動 $a = ActionSelect(Q, s)$
10. $\quad\quad$ 行動 a を実行，報酬 r を獲得，次状態 s' へ遷移
11. $\quad\quad$ 報酬から $selection$ へ到達するように内部報酬 ir を計算
12. $\quad\quad$ $Q(s,a) = Q(s,a) + \alpha\left[ir + \gamma \max_{a' \in A} Q(s',a') - Q(s,a)\right]$
13. $\quad\quad$ $step = step + 1$
14. $\quad\quad$ **if** $s' \in s_{end}$ **then**
15. $\quad\quad\quad$ 配列 $\boldsymbol{mr}$ の末尾に報酬 r を挿入
16. $\quad\quad\quad$ break
17. $\quad\quad$ **end if**
18. $\quad$ **end for**
19. $\quad$ $UpdateStepDRES(\boldsymbol{mt}, \boldsymbol{t})$
20. $\quad$ **if** 報酬獲得した **then**
21. $\quad\quad$ $\boldsymbol{bid}[selection] \quad = \frac{\xi - 1}{\xi}\boldsymbol{bid}[selection] + \frac{\boldsymbol{t}[selection]}{\xi}$
22. $\quad$ **else**
23. $\quad\quad$ $\boldsymbol{bid}[selection] \quad = \frac{\xi - 1}{\xi}\boldsymbol{bid}[selection] + \frac{0}{\xi}$
24. $\quad$ **end if**
25. $\quad$ $BID(selection, \boldsymbol{bid}, \boldsymbol{mr_g}, Thresholdmr_g)$
26. **end for**

6.5 実験 1: エージェント数＋ゴール数の動的変化

6.5.1 実験内容

ここでは PMRL-OM のできない問題に適用可能であるということを検証するために，2 体エージェントにおける迷路に適用させ，PMRL-DRES と PMRL-OM の性能を比較

Algorithm 7 PMRL-DRES の目的価値設定関数 $UpdateStepDRES$

Input: 自身の到達ゴールとステップ数の配列と最短ステップ数配列 $\boldsymbol{mt}$,$\boldsymbol{t}$

Output: なし

1. **if** $g \in s_{end}$ **then**
2. 配列 $\boldsymbol{mt}$ の末尾に $(iteration, g, step)$ を挿入
3. **if** 配列 $\boldsymbol{mt}$ の長さが閾値 $MaxLength$ 以上 **then**
4. $\boldsymbol{mt}$ の先頭要素を削除
5. **end if**
6. **end if**
7. $\boldsymbol{t}$ の全要素に $MaxStep$ を代入
8. **for** $index = 0$ to $MaxLength$ **do**
9. **if** $\boldsymbol{mt}[index, g] < \boldsymbol{t}[g]$ **then**
10. $\boldsymbol{t}[g] = \boldsymbol{mt}[index, g]$
11. **end if**
12. **end for**

Algorithm 8 PMRL-DRES の目的価値設定関数 BID

Input: 選択したゴールと目的価値配列 $selection, \boldsymbol{bid}$　　獲得報酬保存配列と獲得報酬期待値の閾値 $\boldsymbol{mr_g}$,$Thresholdmr_g$

Output: なし

1. 初期値 $False$ の選択可能なゴールを示すフラグ集合 $\boldsymbol{flag}$ を設定
2. $\boldsymbol{st} = \boldsymbol{t} - \log_\gamma \frac{r_{stan}}{\boldsymbol{r}}$
3. **for** $n = 1$ to $AgentNumber$ **do**
4. **if** $\boldsymbol{flag}[g] = False \wedge \boldsymbol{st}[g]$ が最小であるとき **then**
5. $\boldsymbol{flag}[g] = True$
6. **end if**
7. **end for**
8. **if** $\boldsymbol{flag}[g] = True \wedge \boldsymbol{bid}[g]$ が最大 $\wedge \frac{1}{|\boldsymbol{mr_g}|} \sum \boldsymbol{mr_g} > Thresholdmr_g$ **then**
9. $selection = g$
10. **end if**
11. **if** 一定確率 **then**
12. **while** $\boldsymbol{flag}[selection] \neq True \wedge \frac{1}{|\boldsymbol{mr_{selection}}|} \sum \boldsymbol{mr_{selection}} > Thresholdmr_{selection}$ **do**
13. $selection$ をランダム選択
14. **end while**
15. **end if**

する．また実験に使用する迷路はある学習回数を境に別の環境に変化する動的迷路を採用する．具体的には図 6.7，図 6.8，図 6.9，図 6.10 の 4 種類の迷路 (それぞれケース 0，ケース 1，ケース 2，ケース 3 と呼ぶ) に対し PMRL-DRES と PMRL-OM の性能を比較する．各図において，上段が変化前，下段が変化後の環境を示している．なお本実験において，環境変化は図 6.7，図 6.8，6.9 は学習回数 50000 回，図 6.10 は学習回数 25000 回を境に起こる．また各マスが状態を表し，A，B と表示されたマスがそれぞれのエージェントの初期状態を表し，数字の書いてあるマスがゴールを表す (図 6.7 においては X，Y，Z と表示されたマスがゴールであり，報酬値は一律で 10 が与えられる)．また，ゴールに書かれた数字はそのゴールに到達した際に得られる報酬値を表している．図 6.10 において，黒いマスが存在しているが，これはエージェントが到達できない状態を表している．エージェントが黒いマスへ到達するように行動を選択すれば状態遷移せず，同じマスに止まり続ける．なおケース 0 はエージェントが増えた場合，ケース 1 はゴールが増えた場合，ケース 2 は報酬値が変化した場合，ケース 3 はゴールと報酬値の変化が複合した場合の PMRL-DRES と PMRL-OM の性能の違いを比較するために採用している．

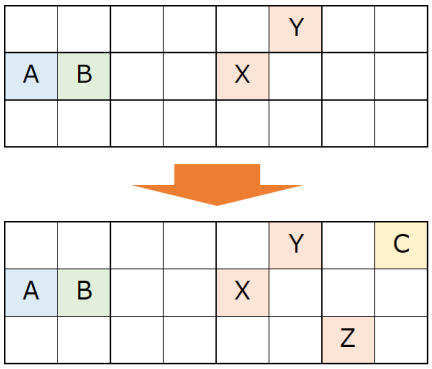


図 6.7　ケース 0 の迷路

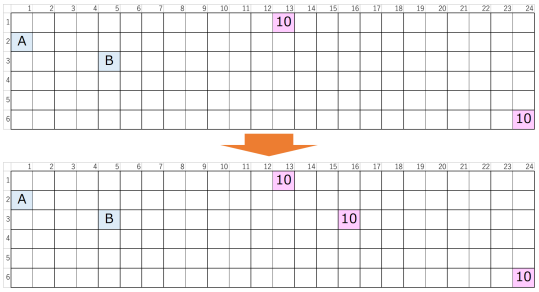


図 6.8　ケース 1 の迷路

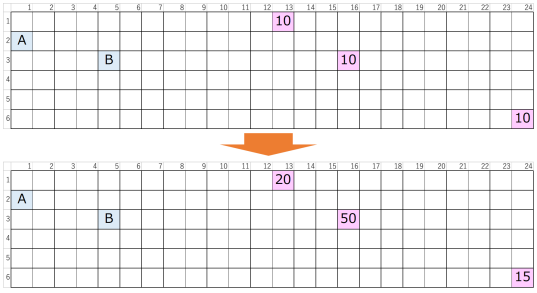


図 6.9　ケース 2 の迷路

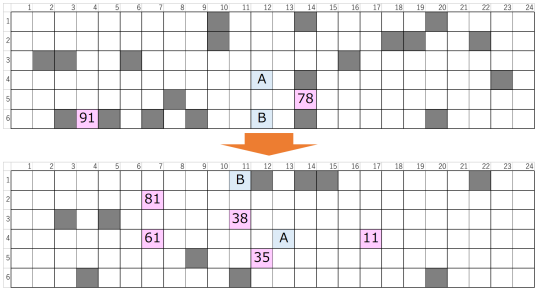


図 6.10　ケース 3 の迷路

6.5.2　評価基準とパラメータ設定

本実験は各迷路で乱数のシードを変更した 30 試行実験を行い，ケース 0 においてはすべてのエージェントが最短ステップ数でゴールに到達した割合を評価し，他のケースに

表 6.1　PMRL-DRES の実験 1 のパラメータ

	PMRL-DRES	PMRL-OM
エピソード数	100000	
環境変化点	50000	
最大ステップ数	100	
初期 Q 値	0	
学習率 α	0.1	
割引率 γ	0.9	
報酬値	10	
定数 δ	10	
定数 ξ	100	
基準報酬値 r_{stan}	10	なし
閾値 $Thresholdmr_g$	5	なし

おいては式 (6.4) にて計算される値で評価する．式 (6.4) において，$evalue$ は評価値を示し，r^i は任意のエージェント i が獲得した報酬値を示しており，t^i はその報酬を獲得するまでに費やしたステップ数を示している．この式により，エージェントはより多くの報酬を獲得し，獲得までに費やしたステップ数の最大値が小さくなる時もっとも良い状況として評価される．なお，学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じように学習反復を行い，その時のステップ数や到達したゴールを評価する．そして，実験はランダムシードの異なる 10 試行を行い，その平均値で評価する．加えて，もし 2 体のエージェントが同じゴールへ到達してしまった場合，良い評価値となることを避けるため分母のステップ数は 100 とする．

$$evalue = \frac{\sum_{\forall} \quad ir^i}{\max_{\forall} it^i} \tag{6.4}$$

表 6.1 に実験パラメータを示す．このとき学習エピソード数が 100000 回 (1 行目)，環境変化点は 50000 回 (2 行目) に設定する．また 1 学習における最大のステップ数を 100 に設定する (3 行目)．また学習において初期 Q 値は 0(4 行目)，学習のパラメータをは学習率 α が 0.1(5 行目)，割引率 γ は 0.9 に設定する (6 行目)．報酬値は 10 に設定する (7 行目)．また，PMRL の内部報酬設定のための定数 δ は 10 として (8 行目)，PMRL-OM の定数 ξ は 100 とする (9 行目)．また PMRL-DRES 固有の変数として基準報酬値 r_{stan} を 10(10 行目)，獲得報酬期待値の閾値 $Thresholdmr_g$ を 5 に設定する (11 行目)．

6.5.3　実験結果

実験結果を図 6.11，6.12，6.13，6.14 に示す．ケース 0 において縦軸が最短ステップ数でゴールに到達した割合，横軸がエピソード数であり，他のケースにおいて縦軸は評価値 *evalue*，横軸は学習回数を示している．また橙，青の線がそれぞれ PMRL-DRES と PMRL-OM における結果を示している．これを見ると分かる通り，ケース 0 において従来の PMRL-OM では環境変化前には確率 1 で最短ステップで全ゴールに到達できていたことに対し，環境変化後には最短ステップ数で全ゴールへ到達する試行が 7 割まで落ち込んでいる．一方で PMRL-DRES では環境変化前も後もすべてのケースにおいて確率 1 で最短ステップ数でゴールに到達することができている．他のケースにおいては，図 6.12 では，環境変化前はほぼ同じ評価値だが，環境変化後は PMRL-DRES が上回っていることが分かる．図 6.13 では，環境変化前では PMRL-DRES が PMRL-OM を上回っており，環境変化後も時折同じになるときはあるが，平均的に PMRL-DRES が PMRL-OM を上回っていることが分かる．図 6.14 では，環境変化前はどちらも同じ評価値を示しており，環境変化後は圧倒的に PMRL-DRES が PMRL-OM を上回っている．

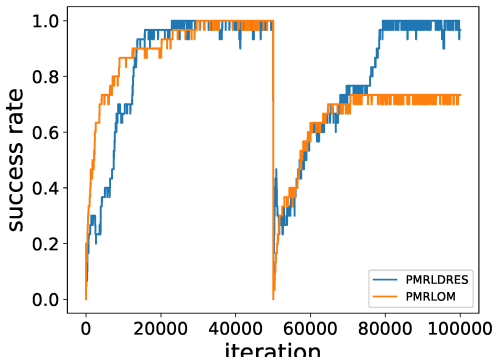


図 6.11　ケース 0 における結果

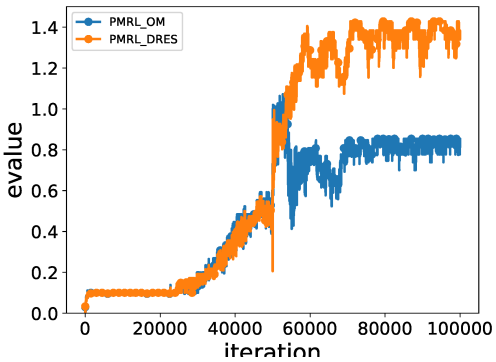


図 6.12　ケース 1 における評価値

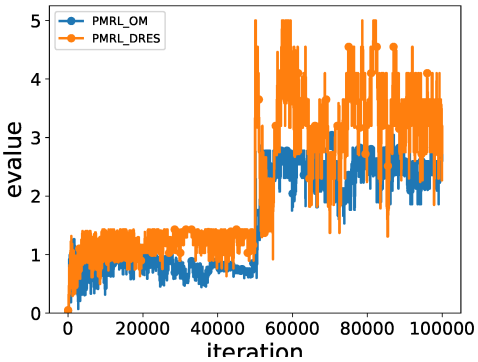


図 6.13　ケース 2 における評価値

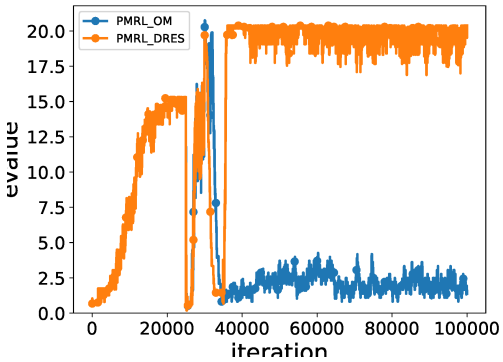


図 6.14　ケース 3 における評価値

6.5.4 考察

以上の結果により，すべての環境変化に対して PMRL-DRES が PMRL-OM を上回っており，PMRL-DRES の改良により動的変化に適応して学習することができていると分かる．加えて，環境の変化とゴール数，報酬値の変化が複合した迷路においても PMRL-DRES は圧倒的に高い評価値を示しており，今回想定した複合した動的変化に対し追従性が高いことが明らかになった．これから各ケースを分析し，PMRL-DRES の有効性を検証する．

・ケース 0 の分析

ケース 0 において，通常の PMRL-OM は環境変化後に 7 割の確率で学習できているという結果になった．これは，エージェント C にとってすべてのゴールの距離が他のエージェントよりも近く，エージェント C の学習した行動の違いによって生まれているものと考えられる．図 6.15 は PMRL-OM のケース 0 における 30 試行の内の成功例と失敗例を示している．図の左が成功例で右が失敗例であり，それぞれ上段からエージェント A，B，C の Q テーブルを示している．各 Q テーブルにおいて，白マス，紫マスがそれぞれスタート地点，ゴール地点を表し，各マスからのびる黒い矢印は行動であり，その脇にある赤い数値はその行動の Q 値を示している．また橙の矢印はスタートから Q 値が最大の行動を選択していったときに辿る経路である．図を見ると分かる通り，成功例においてはエージェント C が最も遠くのゴールを目指しておらず，その影響によりエージェント A，B がゴール X，Y の中で最も遠くのゴール到達を目指して学習し，全体として最短ステップ数で全ゴールに到達することができている．一方で失敗例においては，全エージェントが自身よりも遠くのゴール到達を目指し学習して，結果としてエージェント C がゴール X へ到達する行動を学習してしまい，最短ステップ数でゴールに到達できていないことが分かる．これは PMRL にて説明した，確率でランダム選択したゴールに到達するように内部報酬を設定する機構により起こると考えられる．つまり，通常であれば全ケース失敗するが，エージェント C は付近に別のゴールが多く，別のゴールへ到達するように内部報酬を設計すると，数回のエピソードでそのゴールに到達するようになる．一方でエージェント A，B はゴールが遠いため，別のゴールへ到達するように内部報酬を設計したとしても，多くのエピソード学習しない限りそのゴールに到達しない．そしてその状況になる前に適切なゴールに到達するように内部報酬を設計し直すため，エージェント A，B は他のゴールに到達するように学習することはない．そして，ゴール X に着目すると，エージェント A，C はそこに到達するための最短ステップ数が等しい．つまり，各エージェントがゴール X に到達するように内部報酬を設計していても，エージェント C が別のゴールへ到達するように内部報酬を設計する機構の影響で，そのゴールに最初に到達できる確率は

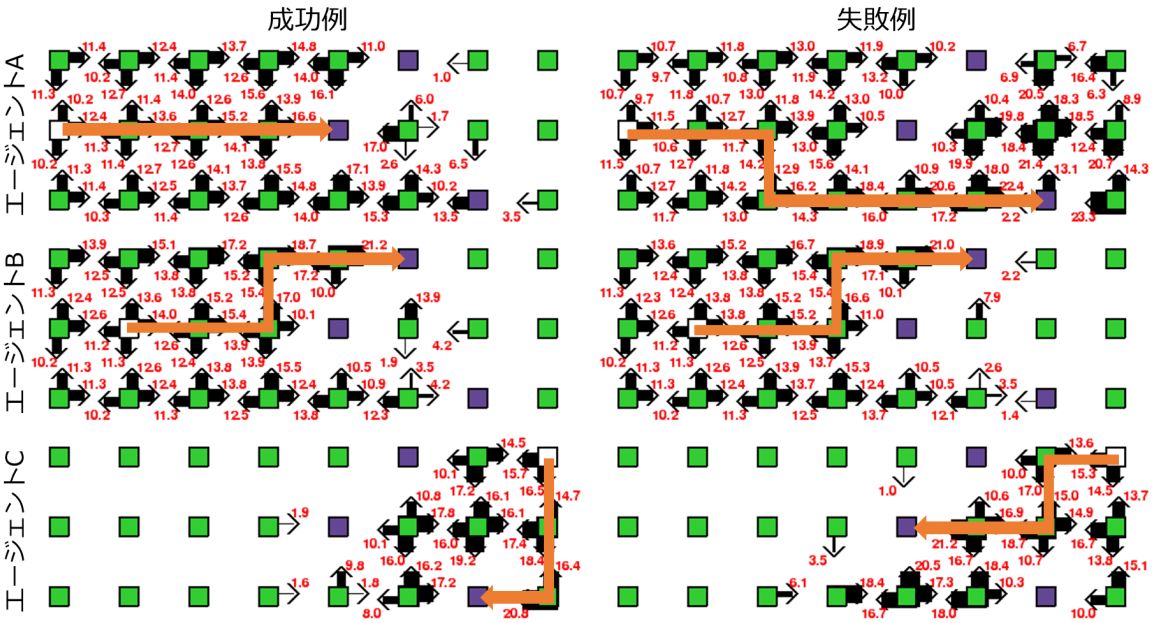


図 6.15　PMRL-OM におけるケース 0 の失敗例と成功例

エージェント A よりも小さくなる．そのため，PMRL-OM では 7 割成功し，3 割失敗するという結果になったものと考えられる．

また，図 6.16 は環境変化点においてエージェントの学習結果などを完全にリセット (ハードリセット) したときとそうでないときでの PMRL-DRES の結果の変化を示している．図の縦軸は最短ステップ数でゴールに到達した割合を示しており，横軸は学習回数を示している．また，青線がハードリセットの結果であり，橙線が通常の PMRL-DRES の結果を示している．結果を見ると分かる通り，ハードリセットしない方が収束が早く，ハードリセットよりも約 3000 学習早く確率 1 に収束している．このことにより，PMRL-DRES は環境変化前の学習結果を適切に利用し，活用していることがわかる．

最後に，図 6.17 と図 6.18 は獲得報酬期待値 er_g の閾値をそれぞれ 1 刻みに 1 から 10 まで変化させたときの結果の違いを示している．図の縦軸は最短ステップ数でゴールに到達した割合を示しており，横軸は学習回数を示している．図 6.17 は 1 から 5 までの結果がそれぞれ青線，橙線，緑線，赤線，紫線で示され，図 6.18 は 6 から 10 までの結果をそれぞれ青線，橙線，緑線，赤線，紫線で示している．これを見るとわかる通り閾値は 5 が最適であり，環境変化に適応するためにはある程度閾値を厳しくする必要がある．ただし閾値 8 や 9 のときのように厳しすぎると制限されすぎてしまい，適切な目的達成が逆に阻害されてしまう恐れがあることがわかる．

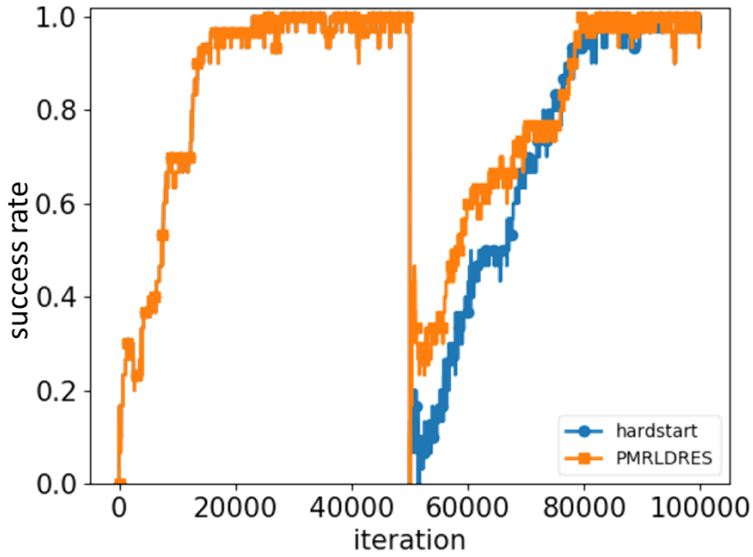


図 6.16　ハードリセットとの比較

・ケース 1 の分析

表 6.2 はケース 1 の環境変化後の各ゴールの報酬値と到達するまでの最短ステップ数を示している．ゴール X，Y，Z は一番左の列からゴールを識別したものである（仮に同じ列であれば上の行から識別することにする）．このとき報酬値は等しいので標準ステップ数は通常の最短ステップ数と変わりはない．PMRL-DRES では標準ステップ数の最も小さいゴールからエージェント数分だけゴールを制限し，そのゴール内で協調行動を学習する．そのため，PMRL-DRES ではゴール X と Y へ到達するようにエージェントは協調行動を学習する．一方で，PMRL-OM ではそのような制限がないため，遠くのゴールであればあるほど到達すべきゴールであると学習を行うため，ここではゴール Y と Z へ到達するように協調行動を学習する．このとき評価値 $evalue$ は PMRL-DRES で $20/13 = 1.5$，PMRL-OM で $20/22 = 0.9$ と推定できる．図 6.12 を見るとほぼその評価値に収束していることが分かる．以上から，ケース 1 のゴール数の動的変化に対し，PMRL-DRES は適切に学習できていることが分かる．

・ケース 2 の分析

次に，表 6.3 はケース 2 の環境変化後の各ゴールの報酬値と到達するまでの最短ステップ数，及びそこから計算される標準ステップ数を示している（このとき基準とする報酬値 r_{stan} は 10 とする．）このとき，PMRL-DRES はゴール X と Y の標準ステップ数が小さいため，各エージェントはゴール X，Y だけで協調行動を学習していることが分か

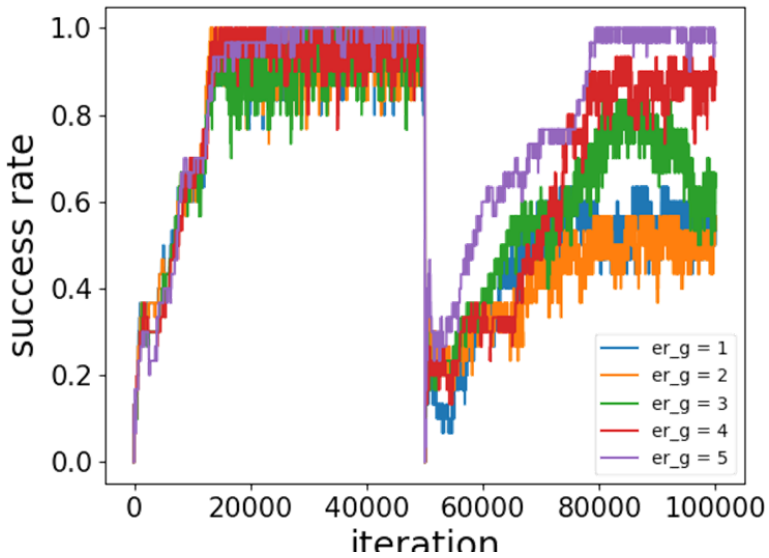


図 6.17　er_g の閾値による結果の変化 (閾値が 1 から 5 まで)

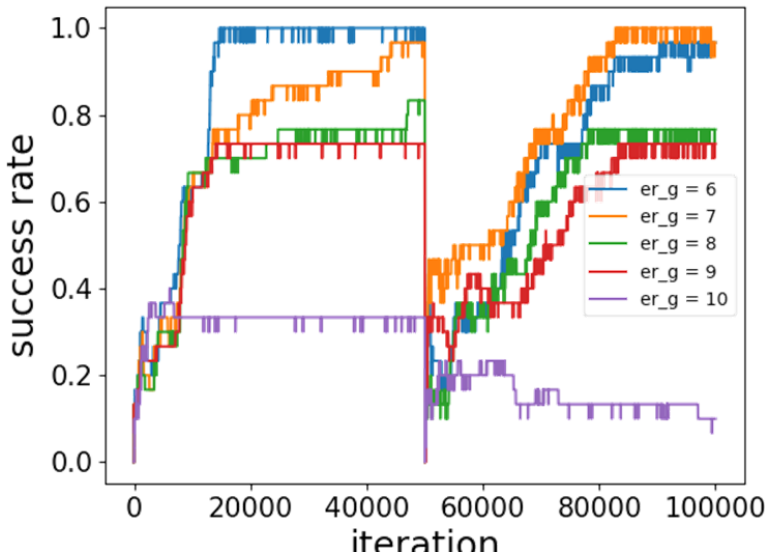


図 6.18　er_g の閾値による結果の変化 (閾値が 6 から 10 まで)

る．一方で PMRL-OM では報酬値を加味していないため，ケース 1 の時と同様にゴール Y，Z で協調行動を学習することが分かる．このとき評価値 *evalue* は PMRL-DRES で $70/13 = 5.4$，PMRL-OM で $65/22 = 3.0$ と推定できる．図 6.13 を確認すると，環境変化後に PMRL-OM は確かに 3.0 付近を推移しているのに対し，PMRL-DRES は 2 から 5 までを推移していることが分かる．この原因として学習がまだ完了していないことが考えられる．PMRL-DRES の 100000 学習における 10 シードの評価値はそれぞれ 0.5, 5,

表 6.2　ケース 1 における最短ステップ数

	Goal X	Goal Y	Goal Z
Agent A	13	16	27
Agent B	10	11	22
Reward	10	10	10

表 6.3　ケース 2 における標準ステップ数

	Goal X	Goal Y	Goal Z
Agent A	13	16	27
Agent B	10	11	22
Reward	20	50	15
Agent A	0.42	0.72	23.2
Agent B	-2.58	-4.28	18.2

5, 0.5, 0.5, 5, 0.2, 5, 5 であり，大体は推定値の 5.4 に近づけていることが分かる．またここにある 0.5 や 0.2 はエージェントがどちらもゴール Y や X に到達してしまったために評価値が 50/100 = 0.5 や 20/100 = 0.2 となってしまったためであると考えられる．この現象は両エージェントゴール X，Y の範囲内のことで，根本的に PMRL-DRES が機能していないわけではないことが分かる．以上から，ケース 2 の報酬値の動的変化に関しても PMRL-DRES は適切に学習できていることが分かる．

表 6.4　ケース 3 における標準ステップ数

	Goal X	Goal Y	Goal Z	Goal V	Goal W
Agent A	8	6	3	2	4
Agent B	5	7	2	5	9
Reward	81	61	38	35	11
Agent A	-200	-174	-130	-123	-5.48
Agent B	-203	-173	-131	-120	-0.48

・ケース 3 の分析

最後にケース 3 の結果について分析する．表 6.4 はケース 3 の環境変化後の各ゴールの報酬値と到達するまでの最短ステップ数，及びそこから計算される標準ステップ数を

表 6.5　ケース 3 の 0 試行における目的価値

	Goal X	Goal Y	Goal Z	Goal V	Goal W
Agent A	0.75	7	0	0	0
Agent B	3.6	3.0	0	0	0

示している（このとき基準とする報酬値 r_{stan} は 10 とする.）このとき，PMRL-DRES はゴール X と Y の標準ステップ数が小さいため，各エージェントはゴール X，Y だけで協調行動を学習していることが分かる．また一方で PMRL-OM ではエージェント A はゴール X を目指し，エージェント Y はゴール W を目指すことが分かる．このとき評価値 $evalue$ は PMRL-DRES で $142/6 = 23.7$，PMRL-OM で $92/9 = 10.2$ と推定できる．図 6.14 を確認すると，環境変化後に PMRL-DRES は確かに 20 付近に収束しているのに対し，PMRL-OM は 2.5 付近を推移していることが分かる．この原因として学習がまだ完了していないことが考えられる．特に 2.5 あたりの数値はどの組み合わせでもないため，PMRL-OM は一定のゴールに到達するように学習できていないことが考えられる．加えて，表 6.5 はケース 3 の 0 試行の時の各エージェントが推定した目的価値である．これを見ると，ゴール Z，V，W に関して目的価値が 0 であり，PMRL-DRES がきちんとゴール X，Y 以外のゴールを制限できていることが分かる．また目的価値もエージェント A はゴール Y，エージェント B はゴール X を最も適したゴールとして評価できている．エージェント A，B がそれぞれゴール Y，X へ到達するとき最もステップ数が小さくなることからもそれはわかる．このことから，ケース 3 の複合した動的変化に関しても PMRL-DRES は適切に学習できていることが分かる．

6.6　実験 2: 全動的変化が複合した環境における検証

6.6.1　実験内容

「空間的環境変化」と「エージェント・ゴール数変化」2 種類の動的変化が複合して起こった際の PMRL-DRES の有効性を検証するため，本研究では図 6.19-6.24 に示す 6 種類の迷路があるエピソードを境に切り替わる動的環境に PMRL-DRES と Profit Sharing を適用させ，性能を比較する．具体的に 0 から 24999 エピソードまで図 6.19 の迷路であり，25000 エピソードからは図 6.20，50000 エピソードからは図 6.21，75000 エピソードからは図 6.22，100000 エピソードからは図 6.23，150000 エピソードからは図 6.24 の迷路になる．各図において，各マスが状態を表し，A，B，C，D，E および X，Y，Z，V，W と表示されたマスがそれぞれのエージェントの初期状態とそれぞれのゴールを表す．また灰色のマスは壁であり，エージェントはこのマスに到達することはできない．エージェン

トがこのマスへ到達するように行動を選択すれば状態遷移せず，同じマスに留まる．

これら 6 種類の変化は図 6.25 に従って変化する．このとき図の橙字はそれぞれの環境変化の種類を示している．これにより，各動的変化はそれぞれ種類が異なり，そしてそれら 6 種類の変化によって全動的変化を検証しているといえる．具体的には，25000 エピソード後に空間的環境変化 (スタート地点とゴール地点の変化 + 壁の出現) が起こり，50000 エピソード後に空間的環境変化に加えて，エージェント・ゴール数の変化が起こる．その後，75000 エピソード後には，スタート地点とゴール地点の変化と壁の出現に加えて，学習空間の拡大が起こる．そして，100000 エピソード後にはゴール数のみを増加させ，最後に 150000 エピソード後にエージェント数を増加させる．

6.6.2 評価基準とパラメータ設定

本実験は各迷路で乱数のシードを変更した 30 試行実験を行い，エージェント全体の到達ステップ数を評価する．なお，学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じように学習反復を行い，その時のステップ数や到達したゴールを評価する．そして，実験はランダムシードの異なる 30 試行を行い，その平均値で評価する．加えて，もし 2 体以上のエージェントが同じゴールへ到達してしまった場合，良い評価値となることを避けるため分母のステップ数は 100 とする．

表 6.6 に実験パラメータを示す．このとき学習エピソード数が 100000 回 (1 行目)，環境変化点は 50000 回 (2 行目) に設定する．また 1 学習における最大のステップ数を 100 に設定する (3 行目)．また学習において初期 Q 値は 0(4 行目)，学習のパラメータをは学習率 α が 0.1(5 行目)，割引率 γ は 0.9 に設定する (6 行目)．報酬値は 10 に設定する (7 行目)．また，PMRL の内部報酬設定のための定数 δ は 10 として (8 行目)，PMRL-OM の定数 ξ は 100 とする (9 行目)．また PMRL-DRES 固有の変数として基準報酬値 r_{stan} を 10(10 行目)，獲得報酬期待値の閾値 $Thresholdmr_g$ を 5 に設定する (11 行目)．また，比較手法である PS の報酬関数は式 (6.5) に示す通り「エージェント全体の獲得報酬値/全エージェントのゴール到達までのステップ数」とする (このままではエージェント全体の情報が扱える分 PS が有利であるが，今回はその PS と性能を比較する．)

$$R_i^{ps} = \left\{ \frac{r}{N} | s_i, s_{-i} \in S_{goal} \wedge s_i \neq s_{-i} \right\} \tag{6.5}$$

6.6.3 実験結果

図 6.26 に実験結果を示す．縦軸はエージェント全体の到達ステップ数であり，横軸は全体のエピソード数である．また青線は PS の結果であり，緑線は PMRL-DRES の結果である．これを見ると全環境変化に対してより適応できているのは PMRL-DRES ということになるが，最後の環境におけるステップ数はわずかに PS が上回っている．特に

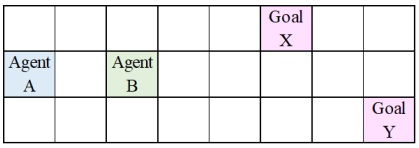


図 6.19　0～24999 エピソードまでの環境

図 6.20　25000～49999 エピソードまでの環境

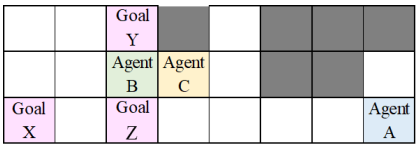


図 6.21　50000～74999 エピソードまでの環境

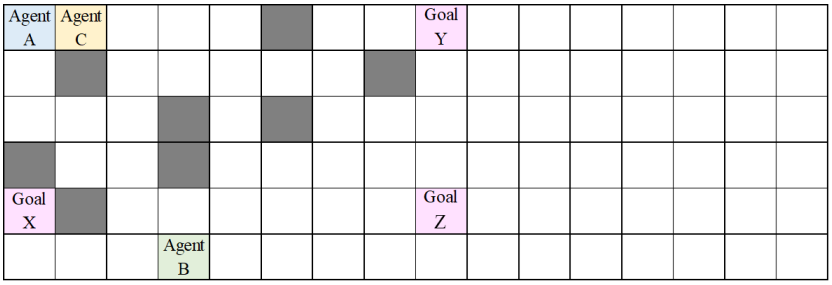


図 6.22　75000～99999 エピソードまでの環境

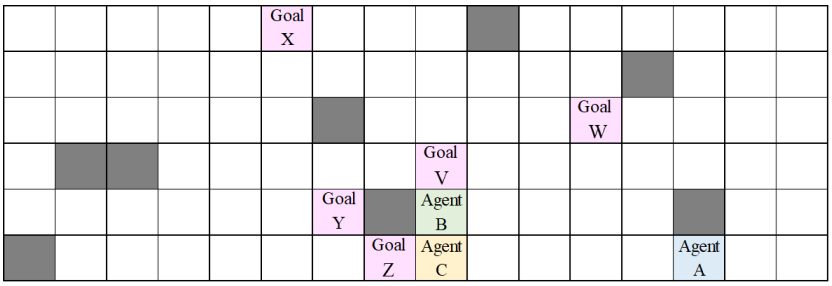


図 6.23　100000～149999 エピソードまでの環境

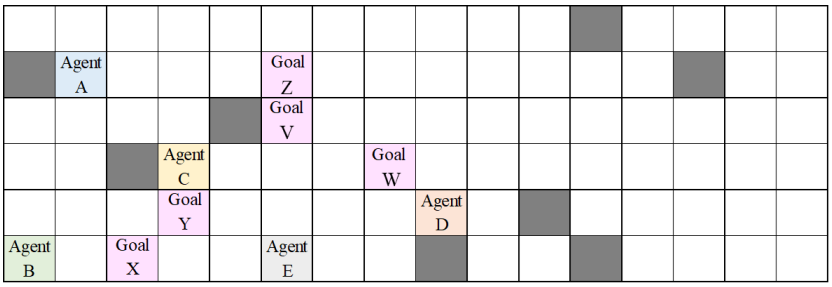


図 6.24　150000～199999 エピソードまでの環境

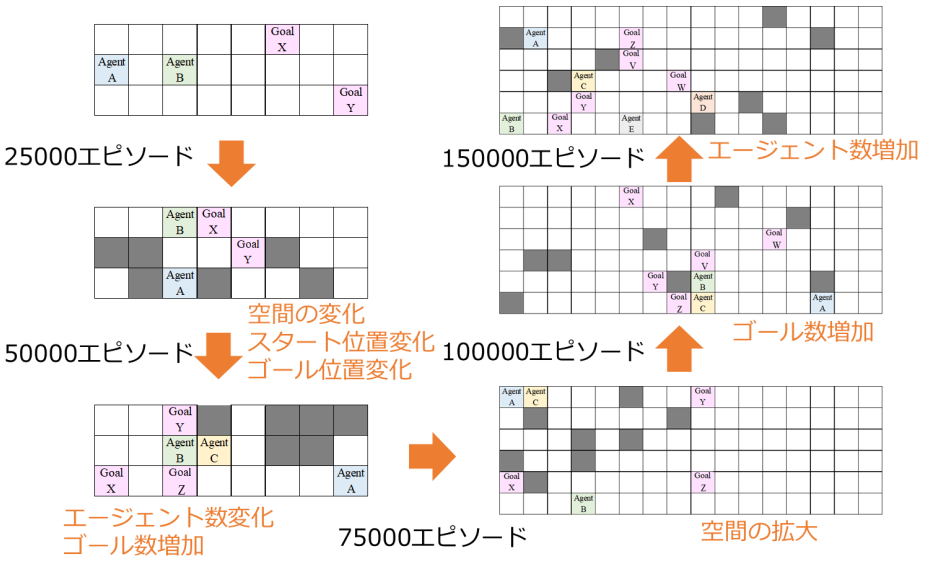


図 6.25　実験 2 の動的変化の推移

表 6.6　全動的変化が複合した環境における実験のパラメータ

	PMRL-DRES	PS
エピソード数	200000	
最大ステップ数	100	
初期 Q 値	0	
学習率 α	0.1	
割引率 γ	0.9	
報酬値	10	
定数 δ	10	
窓長 e	5000	なし
定数 ξ	500	
基準報酬値 r_{stan}	10	なし
閾値 $Thresholdmr_g$	5	なし

75000，100000，150000 エピソードは環境変化点であり，PS は一度学習をやり直している (ステップ数が全シード通して 100 になる) ことに対し，PMRL-DRES においてはそれがない．このことから PMRL-DRES も PMRL-OM 同様環境変化前の情報をうまく利用して環境変化による影響を抑制できていることがうかがえる．

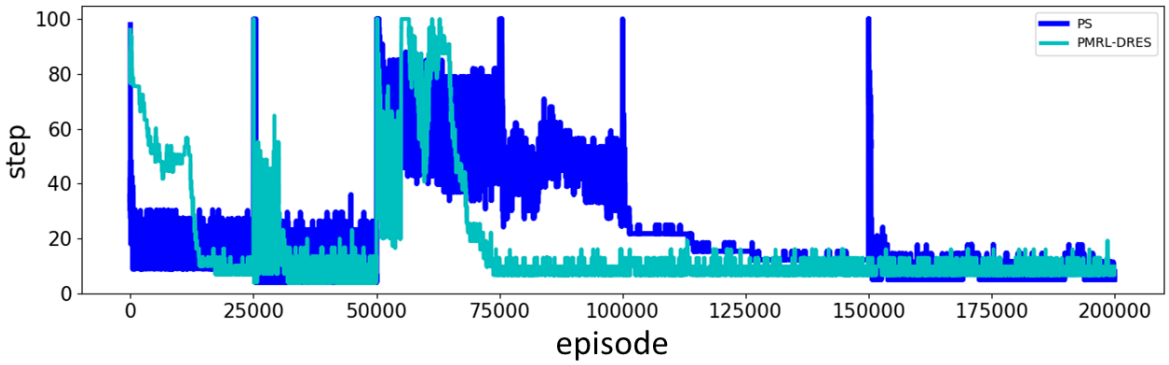


図 6.26　複合した動的環境における実験結果

6.6.4　考察

・複合した環境変化に対する性能差

表 6.7 は環境変化の最後のエージェント全体の到達ステップ数の中央値である．上段下段でそれぞれ PMRL-DRES と PS の結果を示し，マス内の太字はもう一方の結果と比較して上回っていることを示している．これを見ると分かる通り最後の環境変化を除いて PMRL-DRES が上回っている．特に PMRL-DRES は一度も結果が 100 になることはなく，環境変化前の情報を環境変化後にうまく利用して学習していることがわかる．また，環境変化によって環境変化前の情報に引っ張られるということもうまく解消できていることがわかる．特に図 6.20 から図 6.21 への環境変化は右へ移動する行動を学習したエージェント B が逆方向の左へ移動する行動を獲得しなければならないため難しい．実際 PS はほとんどのケースで協調行動を学習できていない．しかし PMRL-DRES はその中で適切な行動を学習して到達ステップ数を小さくすることに成功している．これは環境変化前と後に関わらず遠くのゴールを目指すという方針を維持しているためであり，エージェント B が即座に遠くのゴール X へ向かう行動を学習しているためである．

表 6.7　各迷路における到達ステップ数の中央値

	maze 1	maze 2	maze 3	maze 4	maze 5	maze 6
PMRL-DRES	**7**	**4**	**6**	**6**	**6**	6
PS	9	**4**	100	13	**6**	**5**

・最後の環境における性能の違い

図 6.27 は学習終了時の PMRL-DRES と PS の Q 値の違いである．図の左側と右側は PMRL-DRES と PS の Q 値を分けており，上段下段はそれぞれエージェント A と B を分けている．またそれぞれの図において，白および紫のマスはエージェントのスタートおよびゴールを示している．また各マスから伸びる黒い矢印は行動を示し，その太さおよび傍らの赤い数値がその行動の Q 値を示している．最後に橙の矢印は各エージェントがスタートから Q 値の最も大きな行動をとった時に通る経路である．これを見ると，エージェント B を除いて各エージェントの到達するゴールが PMRL と PS で異なり，そして PS の方がよりステップ数少なく目的を達成している．これは PMRL-DRES が環境変化前の学習結果に引きずられてエージェント A，B，C の到達するゴールがずれてしまったために全体として異なるゴールに到達しているものと考えられる．その一方で PS は以前の章より見られた特徴であるが環境変化毎に学習をやり直している性質を持っており，この環境変化においてはうまく機能しているものと考えられる．具体的に見てみると図 6.23 においてはエージェント A，B，C は迷路の中心，特に衷心より下側に近づくように行動を学習する．そして図 6.24 においては全体的に左側を目指すように学習する必要があり，その差が影響を及ぼす．図 6.28 は環境変化前と後におけるエージェントの学習した最適行動を示している．各迷路における橙の矢印が最適行動であり，図の迷路 6 にのみ存在する紫の矢印が迷路 5 で学習した結果を用いた際に各エージェントがとる行動の方向を示している．これを見ると分かる通りエージェント B 以外のエージェント A，C は向かうべき方向とは異なる方向へ行動するように学習してしまっている．実際に図 6.27 を見るとエージェントは取るべきではない下方向の行動を取り，エージェント C は真下へ行くべき所を右側の行動をとる．

なお，図 6.27 はエージェント全体の到達ステップ数が 8 であり，表の結果とは異なる．表の通りの 6 ステップでゴールに到達するケースの Q 値は図 6.29 に示す．図の内容は等しいが，各エージェントの到達するゴールが異なっている．これは 4 章で説明したが，ランダムシードにより到達するゴールの割合が変わることで起こる．しかしながらエージェント A，C は同じように取るべきではない下と右の行動をとっている．以上を見ても PMRL-DRES は環境変化前の学習結果を利用することにより，最適なゴールの到達が難しくなるケースが存在する．より具体的にいえば，PMRL-DRES は各ゴールに最初に到達するか否かの確率をとっており，エージェント数が増えたときに環境変化前に居たエージェントの到達するゴールがもうある程度決まってしまっており，その増えたエージェントが確率を適切に計算できなくなるために起こると考えられる．

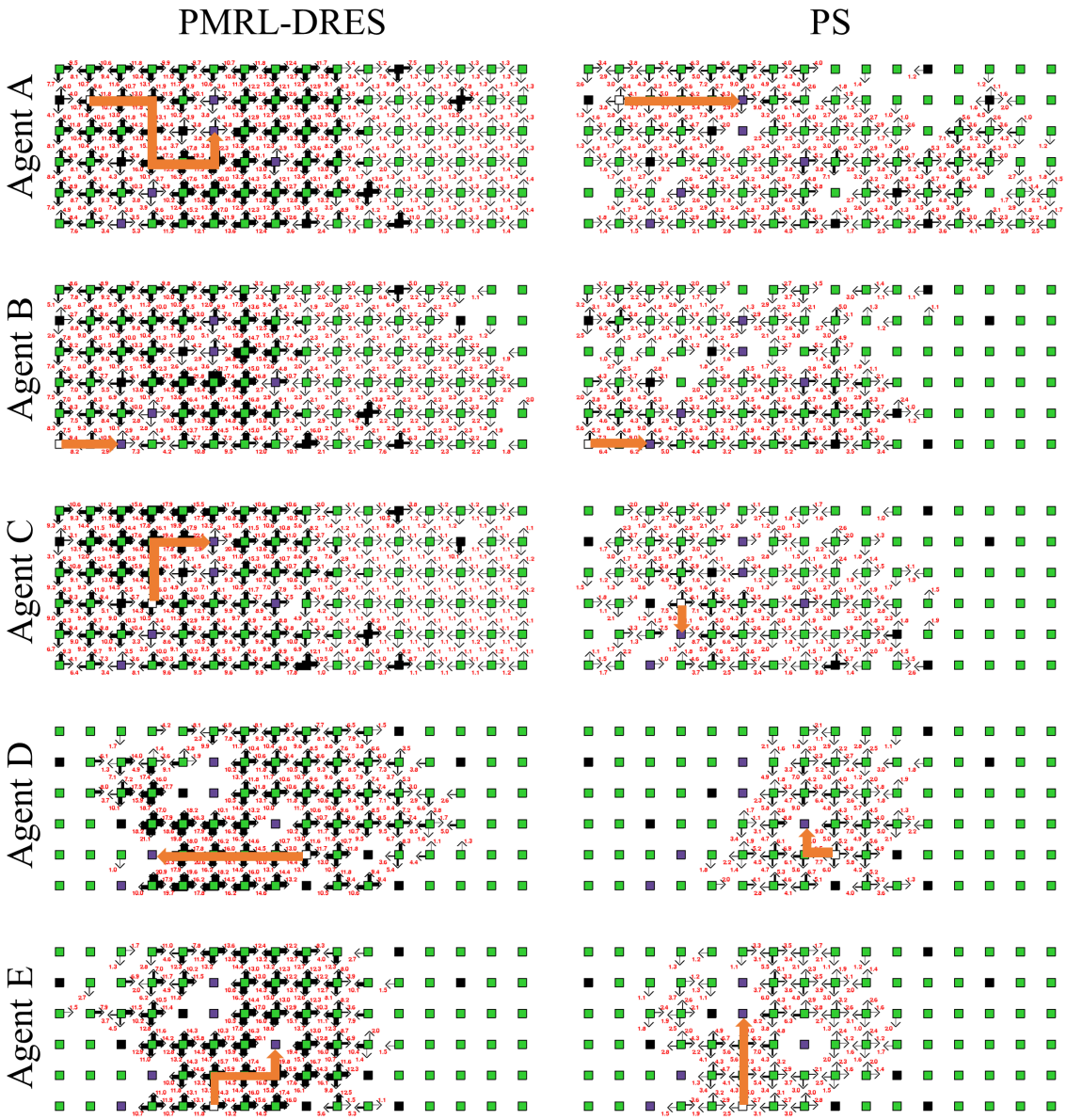


図 6.27　学習終了後の各エージェントの Q 値

6.7　6 章のまとめ

6.7.1　報酬値の動的変化に対する限界

報酬値が動的変化する際の PMRL-DRES の振舞いは，自身にとって期待報酬値が高くなるゴールをエージェント数だけ選び出すというものであるため，すべてのエージェントとゴールの組み合わせを鑑みて，評価値が最も小さくなるようなゴール選択をすることはない．実際，実験 1 においてエージェントは最良の組み合わせのゴールに到達しているわ

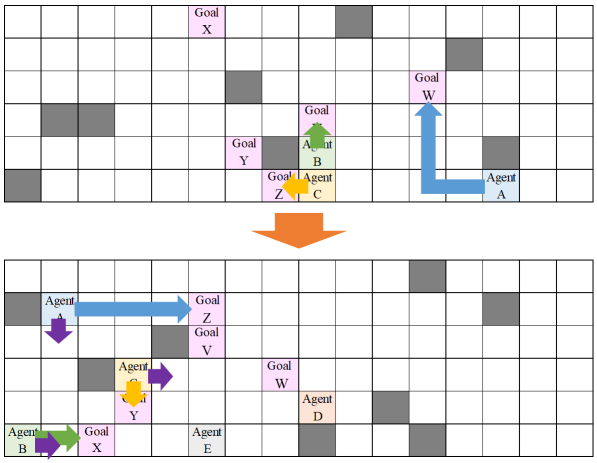


図 6.28　迷路 5 から 6 への環境変化と最適行動

けではないが，2 番目に良いゴールへ到達することができている．またこのメカニズムは到達するゴール数をエージェント数と等しくする手法であるため，各エージェントが同じゴールへ到達する状況は起こり得ない．それは実際に実験 1 のケース 3 を見ればわかる通りである．

また報酬値が変化せず，ゴール数だけが変化する場合においても PMRL-DRES は期待報酬値からゴールを選別するため，各エージェント遠くのゴールを適切に弾いて協調行動を学習できる．そのため PMRL-DRES は最悪のケースを排除することができるが，最適なゴールに到達することはない．しかし各エージェントは期待報酬値の高いゴールへ競合することなく到達できる．

6.7.2　閾値 er_g の限界

PMRL-DRES の重要な機構として獲得報酬値の平均から到達ゴールを制限するものがある．ここで重要な点はその平均値を見て到達すべきゴールか否かを決める閾値 er_g である．実験 1 において閾値 er_g は報酬値の半分が適正であると述べた．そして実験 2 においても同様の閾値で臨んだ結果，PMRL-DRES は全ての環境に適応して適切な行動を学習可能であることを示した．このことから閾値 er_g は報酬値の半分とすることで，どのような環境であれ適正であることが言える．

6.7.3　複合した環境変化に対する頑健性の限界

PMRL-DRES は複合した環境においても各エージェント異なるゴールへ到達し，そのステップ数も従来法より小さくすることに成功した．このことから各動的変化が複合した際にも PMRL の理論的証明が機能していることがわかる．以上を踏まえると，各動的変

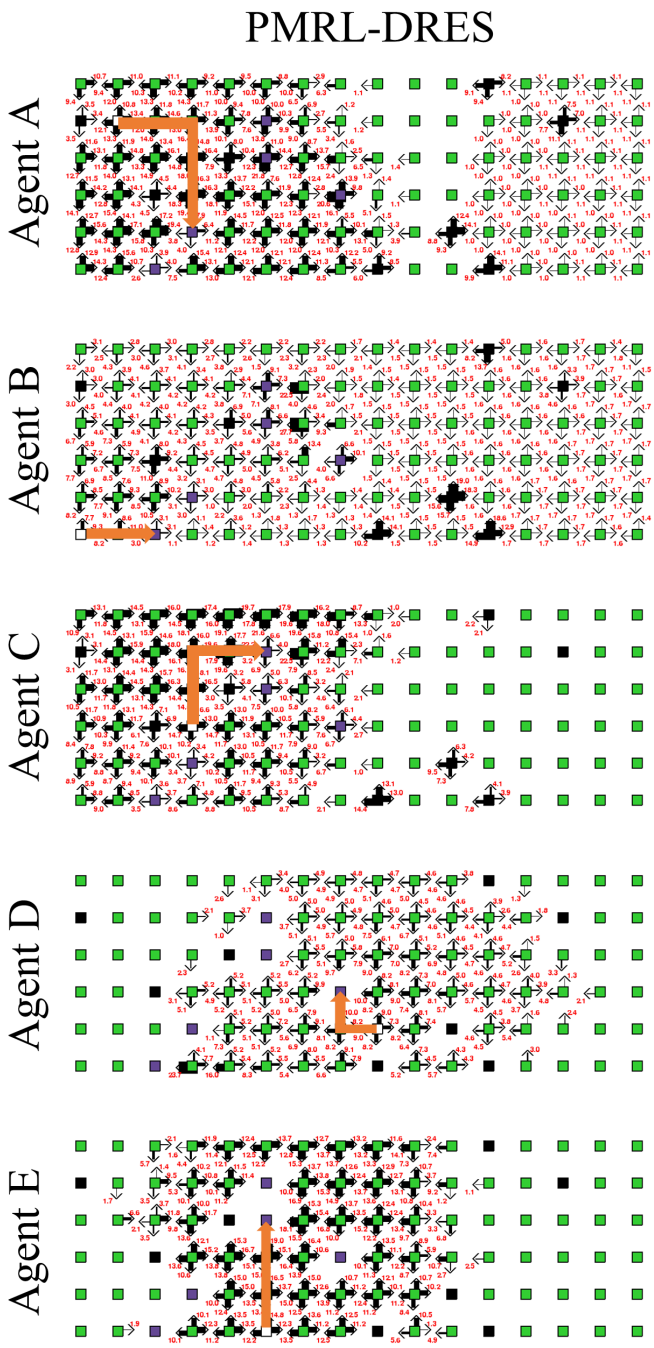


図 6.29　異なるシードにおける学習終了後の PMRL-DRES エージェントの Q 値

化であってもPMRL-DRESにより各エージェントがそれぞれ異なるゴールへ到達することができる．また，PMRL-DRESは環境変化において環境変化前のQ値を利用して，環境変化による影響を抑制して学習のやり直しを防止している．これはどのように環境が変化してもPMRL-DRESの最も遠くのゴール到達を目指して学習するという方針が変化しないため，使えるQ値はそのままに学習のやり直しが必要な行動は学習をやり直すということをしてうまく機能していると考えられる．しかしながら，実験2を踏まえるとその機能が悪影響を及ぼすこともある．それはPMRL-DRESが学習の他に目的価値や獲得報酬期待値のような内部パラメータをも学習しており，それはある程度エージェント同士同期的であることが求められるからである．特に獲得報酬期待値は，各ゴールに対して他のどのエージェントよりも早く到達できた割合を計算しているため，環境が変わった時に目的価値を学習済みのエージェントとそうでないエージェントが発生した際に，学習済みのエージェントは環境変化前に計算した目的価値に引きずられてゴールしてしまい，新しく学習するエージェントの獲得報酬期待値が適切に計算できないということが起こりうる．そして実験2の結果はまさにそのケースが起こった例である．以上を踏まえ，最低でもどの環境変化が起こったとしてもPMRL-DRESは個々のエージェントを別々のゴールへ到達させることができるが，それが最適であるかは分からない．そして環境変化前の学習結果をうまく使い，学習のやり直しを防ぐことができている．

第 7 章

内部報酬設計指針とその活用

本章では第 4 章，第 5 章，第 6 章において提案した 3 つの手法 (PMRL, PMRL-OM, PMRL-DRES) を受けて，まとめとして静的・動的環境における通信なしマルチエージェント強化学習における本提案手法の効果と限界を示した後，設計方法について議論する．

7.1　内部報酬設計法の適用方針

PMRL および PMRL-DRES は MDP 環境の中でエージェントの初期状態及び目的状態があり，そして目的状態で他のエージェントの存在を検知することができる際に機能する．それは例えば迷路問題などの経路プランニングの問題全般に適用できる．詳しい設定は 3 章を参照のこと．つまり，PMRL および PMRL-DRES はシミュレータの中でのエージェントの動作を学習する手法である．ここで具体例として複数ロボットによる物流システムを用いて本研究の提案する報酬設計法を適用する方法を下記にまとめる．

7.1.1　環境とエージェントの整備

マルチエージェント強化学習では，まず環境とエージェントを設計する必要がある．つまり，実際の問題を 3.2 節に示した MDP に落とし込むことが必要となる．図 7.1 は実際のロボットにおける状態観測方法を示している．これは東らの研究などで使用されている方法だが，エージェント集団の頭上にカメラを設置し，カメラ画像を処理してエージェントの状態を分割し，通信によりエージェントにそれを知らせる [9]．このほかにも，エージェントが何かしらのランドマークを検知することで状態を観測する方法などが存在するが，一般的に設計者が状態を分けてそれをエージェントに予め与えることが多い．また，複数のエージェントはお互いに機構が異なっていてはならない．それは，エージェントの観測できる状態，とれる行動，そしてゴールである．その理由は，PMRL-DRES の，獲得報酬期待値によるゴール制限は，他エージェントの動作が等しいという前提のもと，自

身の適切なゴールを通信なしに決定するためである．

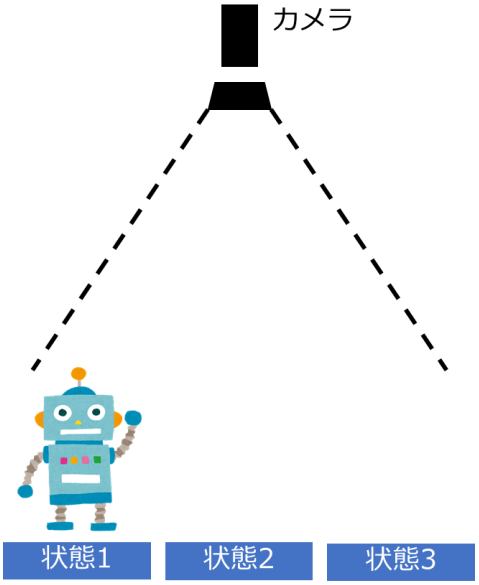


図 7.1　ロボットの状態観測の例

・シミュレータの整備

ハードウェアの設定が終わったら今度はエージェント内のシミュレータを整備する．とはいえ環境とエージェント，報酬を作るだけである．図 7.2 はシミュレータの例である．このようにエージェントとする個々の主体 (ロボット) にシミュレーション環境を導入する．その際に各エージェントのスタート，ゴール，環境の情報が必要となる．各エージェントが何をするかは，各エージェント同じアーキテクチャを持ち同じ手法で学習するため，その情報は必要ない．以上のようにエージェント内にシミュレータを構築して学習を行い，実際の行動をとる．

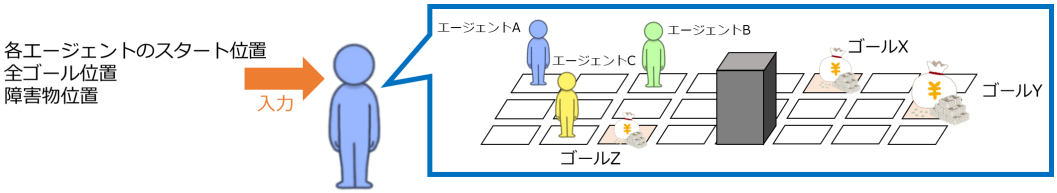


図 7.2　シミュレータ

・環境の整備

まず環境を設定する．環境は状態，報酬，状態遷移則の設定が必要である．つまり 3 章にて説明しているが，実環境情報を下記の変数に落とし込む作業である．

$$S := \{state_0, state_1, ..., state_D\} \tag{7.1}$$
$$\tau_1 : S \times A \times S \to 1 \tag{7.2}$$
$$\tau_2 : goalstate_g \times A \times goalstate_g \to 1,\ where\ goalstate_g \in S_{goal} \tag{7.3}$$
$$r = \{reward | s_i \in S_{goal} \wedge s^j \notin S_{goal} \wedge \forall j \in I \wedge i \neq j\} \tag{7.4}$$

図 7.3 は変数に落とし込む上でのモデル化の例である．図では，ロボット 2 体がある地点からある地点まで物資を運ぶ．ここでまず必要な事は状態を定義することであり，MDP とするためにここでは格子状にエリアを分割し，それぞれを状態として定義する．そして分割した状態の中から初期状態と目的状態を定義する (図では丸印が初期状態，星印が目的状態を示している．)．

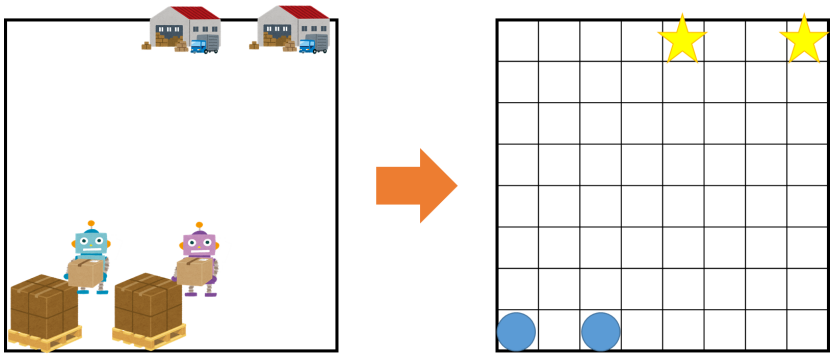


図 7.3　モデル化

・エージェントの整備

エージェントにおいては，下記の図のようにメモリ部と実行部を用意すればよい．その際，Q 値は全て 0 を代入し，最短ステップ数は最大ステップ数，目的価値および内部報酬には 0 を代入する．ゴール到達可能範囲 gr に関しては目的数を初期値とする．

全てのゴールを知る必要性

各エージェントは環境に存在するゴールをすべて知っている必要はない．各エージェントはゴールに到達した際にそのゴールへの最短ステップ数を計算する．そして，最短ステップ数から計算した目的価値の大きいゴールへ到達した際に，そのゴールへ到達するように内部報酬値を設定する．以上からエージェントはゴールに対し，学習前に知ることはない．

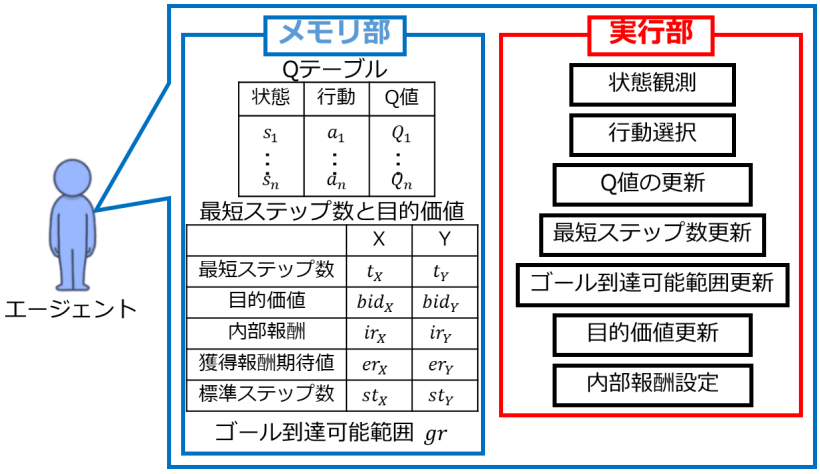


図 7.4　エージェントのアーキテクチャ

7.1.2　手法の適用

環境とエージェントの設定が終了すれば，内部報酬設計法を適用する．具体的には標準ステップ数を計算し，それに基づきゴール到達可能範囲を決定する．そしてそれらの値から目的価値を推定するゴールを選別する．学習中には最短ステップ数の更新と目的価値の更新を繰り返し，その値の最も大きなゴールに到達するように内部報酬値を設定する．また，その際ゴール到達可能範囲 gr は下記の条件に従うことで任意数エージェントにおける協調行動の獲得を可能にする．仮に事前知識などにより下記の式を計算できるのであれば，適切なゴール到達可能範囲で内部報酬設定が可能となる．

$$gr^i = \sum_g GoalNumber(t^i, g, fg) \tag{7.5}$$

$$GoalNumber(t^i, g, fg) = \begin{cases} 1 & if \quad t^i_g \leq t^i_{fg} \wedge g \leq fg \\ 0 & otherwize \end{cases} \tag{7.6}$$

$$fg = \arg\min_g \sum_i FarthestGoal(i, g) \neq 0 \tag{7.7}$$

$$FarthestGoal(i, g) = \begin{cases} 1 & if \quad t^i_g > t^i_{og} (\forall og \in G) \\ 0 & otherwize \end{cases} \tag{7.8}$$

なお，式 (7.9) は上記のゴール到達可能範囲設定法における条件となる．

$$gr^i \geq AgentNum - \sum_j^I FarGr(i, j) \tag{7.9}$$

$$FarGr(i, j) = \begin{cases} 1 & if \quad gr^i < gr^j \\ 0 & otherwize \end{cases} \tag{7.10}$$

7.2　実応用問題への適用に向けて

7.2.1　物資輸送問題

実応用問題として，本研究では物流システムへの適用を目標としている．図 7.5 は物資輸送問題を示している．物資輸送においては，図の左側の通り，ある地点から物資を受け取り，ある地点に届ける．そのため，輸送トラックをエージェントとすれば，スタート地点から行動を開始したエージェントがゴールへ到達する問題となる．そして物資の輸送はある地点だけ早く輸送すればよいというわけではなく，全体に行き渡らせることが重要となるため，本研究の実現する学習により適切な行動を学習可能であることがわかる．この問題は迷路問題に帰着できるため，前節の 3 つの条件を満たし，提案手法は適切に学習できると分かる．

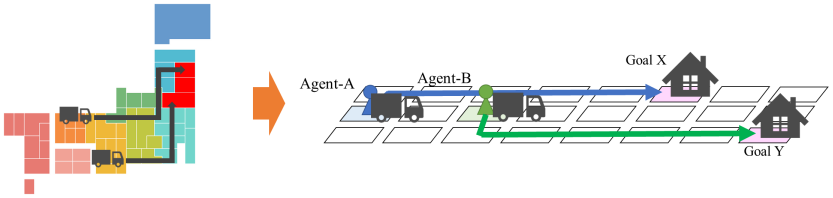


図 7.5　物資輸送問題

7.2.2　倉庫ロボットシステム

ここでは，swisslog 社の autostore[44] のようなスタックタイプの倉庫ロボットのシステムを想定する．図 7.6 はスタックタイプの倉庫システムの概略図である．図においては，9 個の筒状の格納庫が並んでおり，その上にある緑色のロボットが荷物を決められた地点から受け取り，格子の上を車輪で移動して，決められた格納庫に降ろすことで荷物を保存する (なお図においては手前側に腕を持つロボットしかいないため一番奥の格納庫へ物資を置くことができないが，実際は奥側に腕を持つロボットも存在しているため奥にも物資を格納することができるようになっている．)．このとき状態を格納庫の場所であるとすれば，受取地点がスタート地点，物資を届ける格納庫がゴール地点となる．

以上の問題を変数化すれば，状態 $state_0, ..., state_D$ はそれぞれの格納庫の場所となり (D は格納庫の総数)，各エージェントの行動 a は格子を動くため上下左右となる．そして物資を目的の格納庫に届ければ報酬を与え，またゴールへ到達すれば物資を格納するため，それ以上別の物資を格納する (移動して報酬を獲得する) ことがないこととなる．これは前節にて説明した式 (7.1)-(7.4) を満たすものであり，この問題は本研究における提案手法で解くことができるといえる．なおこのときロボット同士の衝突が考えられるが，

その際はある一方のロボットを優先する等のルールを導入しておけばよい．その理由として前節の 3 つの条件を満たしていることが挙げられる．つまり衝突が存在するとしても状態行動空間を網羅することは可能であり，それぞれの経路も場合によるものの到達不可能ではない．そして報酬値やステップ数はどちらかのエージェントの動きが決まれば安定するため，十分適用できると考えられる．

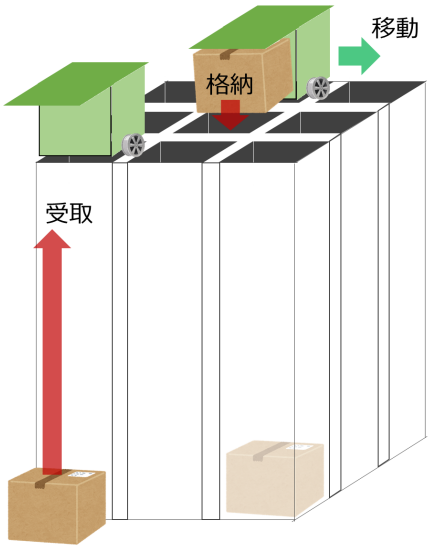


図 7.6　物流システムの概略図

7.2.3　工場ロボットシステム

もう 1 つの実応用例として，工場における産業ロボットシステムへの適用も考えられる．図 7.7 は産業ロボットの例であり，各ロボットは流れてくる物資にある処理を施し，限られた回数ですべての物資に対して効率よく処理することを目的とする．このときロボットはエージェントとなり，流れてくる物資が状態を示す，順番に流れて来る物資に対して各ロボットはどの物資に対して処理を行えばよいかを学習する問題となる．このとき各エージェント 2 回まで処理できるとし，4 個の物資があると仮定して，状態遷移図を描くと図 7.8 の通りとなる．各マスは物資を示し，各行動は処理を行うか否かを示すものとなる．このときそれぞれのエージェントは同等の状態遷移図を持ち，そして図 7.8 を見ればわかる通り全てのゴールへ到達する経路が存在し，最短ステップ数や報酬値にノイズが乗ることはないため，提案手法で解くことができる．そして提案手法により，手前のエージェントは奥のエージェントに物資を譲ることで，より短期間ですべての処理を終えることができると分かる．

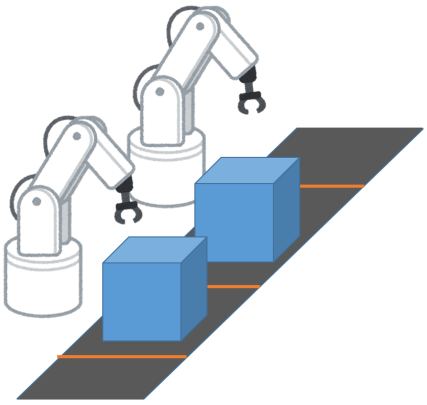


図 7.7　工場ロボットシステムの概略図

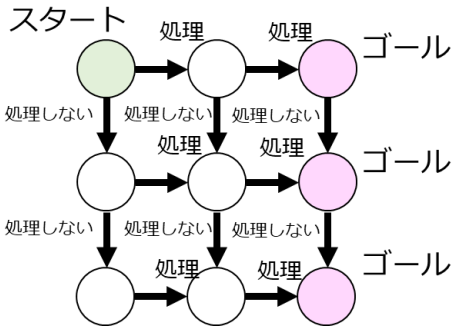


図 7.8　工場ロボットシステムの状態遷移図

7.3　実験：エージェント同士の衝突を考慮した協調行動学習

7.3.1　実験内容

上記にて言及した，エージェント同士衝突を考慮したとしても提案手法が適用可能であるということを検証する．実験は第 4 章の実験にて使用した迷路問題，第 6 章にて利用した動的変化が複合した場合の迷路問題の 2 個を用意し，エージェント同士衝突することを設定した上で，PMRL-DRES を適用する．図 7.9 は第 4 章の実験にて使用した迷路を示し，図 7.10 は，第 6 章にて利用した動的変化が複合した場合の迷路の変化を示している．このとき図の橙字はそれぞれの環境変化の種類を示している．これにより，各動的変化はそれぞれ種類が異なり，そしてそれら 6 種類の変化によって全動的変化を検証しているといえる．また，エージェントの衝突においては，エージェントが同じ状態になるような行動をとった場合に発生し，エージェントの識別番号の若い順 (つまり A，B，C，D，E の順) に優先される．

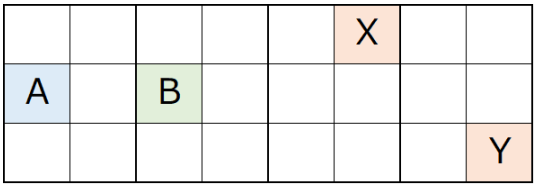


図 7.9　第 4 章における実験の迷路 1

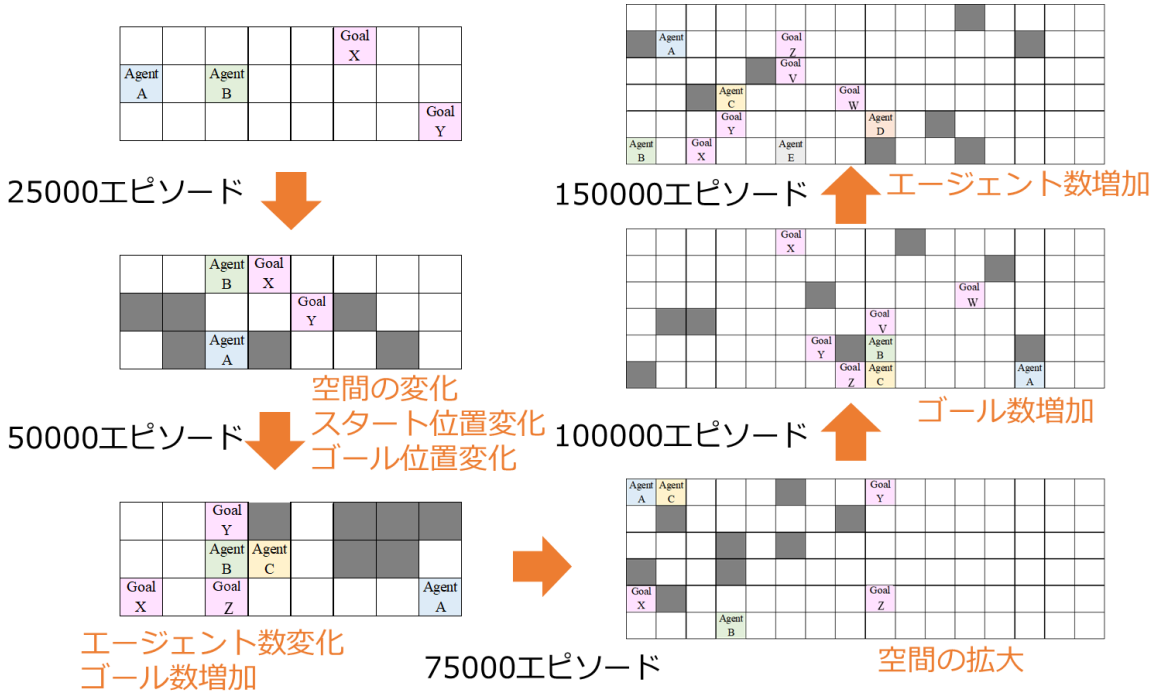


図 7.10　第 6 章における実験 2 の動的変化の推移

7.3.2　評価基準とパラメータ設定

本実験は各迷路で乱数のシードを変更した 30 試行実験を行い，エージェント全体の到達ステップ数を評価する．なお，学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じように学習反復を行い，その時のステップ数や到達したゴールを評価する．そして，実験はランダムシードの異なる 30 試行を行い，その平均値で評価する．加えて，もし 2 体以上のエージェントが同じゴールへ到達してしまった場合，良い評価値となることを避けるため分母のステップ数は 100 とする．

表 7.1 に実験パラメータを示す．このとき学習エピソード数が 200000 回 (1 行目)，1 エピソードにおける最大のステップ数を 100 に設定する (3 行目)．また学習において初期 Q 値は 0(3 行目)，学習のパラメータをは学習率 α が 0.1(4 行目)，割引率 γ は 0.9 に設定する (5 行目)．報酬値は 10 に設定する (6 行目)．また，PMRL の内部報酬設定のための定数 δ は 10 として (7 行目)，PMRL-OM の窓長 e は 5000(8 行目)，定数 ξ は 500 とする (9 行目)．また PMRL-DRES 固有の変数として基準報酬値 r_{stan} を 10(10 行目)，獲得報酬期待値の閾値 $Thresholdmr_g$ を 5 に設定する (11 行目)．

表 7.1　衝突を考慮した環境における実験のパラメータ

エピソード数	200000
最大ステップ数	100
初期 Q 値	0
学習率 α	0.1
割引率 γ	0.9
報酬値	10
定数 δ	10
窓長 e	5000
定数 ξ	500
基準報酬値 r_{stan}	10
閾値 $Thresholdmr_g$	5

7.3.3　実験結果

それぞれの迷路における結果を図 7.11 および図 7.12 に示す．図の赤線，橙線，緑線，青線はそれぞれ最短ステップ数，ステップ数の 30 試行の平均値，中央値，最小値を示している．第 7.11 章の迷路においては，全ての値においてステップ数を小さくできているが，平均値は値のぶれが大きく，中央値と最小値においては最短ステップ数に収束している．また第 6 章の迷路においては，ステップ数を小さくすることができており，最小値においては最短ステップ数に収束している．

7.3.4　考察

図 7.13 および図 7.14 は，第 7.11 章の迷路における，エージェント A と B の学習後の Q 値を示している．図において，白および紫のマスはエージェントのスタートおよびゴールを示している．また各マスから伸びる黒い矢印は行動を示し，その太さおよび傍らの赤い数値がその行動の Q 値を示している．これを見ると分かる通り，最大の Q 値である行動をとると，エージェント A と B はそれぞれゴール X と Y へ最短ステップで到達することができる．以上からエージェント同士が衝突する環境においても適切に学習することが可能であることがわかる．しかしながら，衝突を想定しない状態よりもステップ数が安定していないこともわかる．その理由は前章にて述べたとおりであるが，PMRL-DRES の最短ステップ数の獲得がうまくいかないことが挙げられる．特に，学習が進むにつれ，一方のエージェントが 1 つのゴールへ到達する行動をとり続けてしまうとき，他のエー

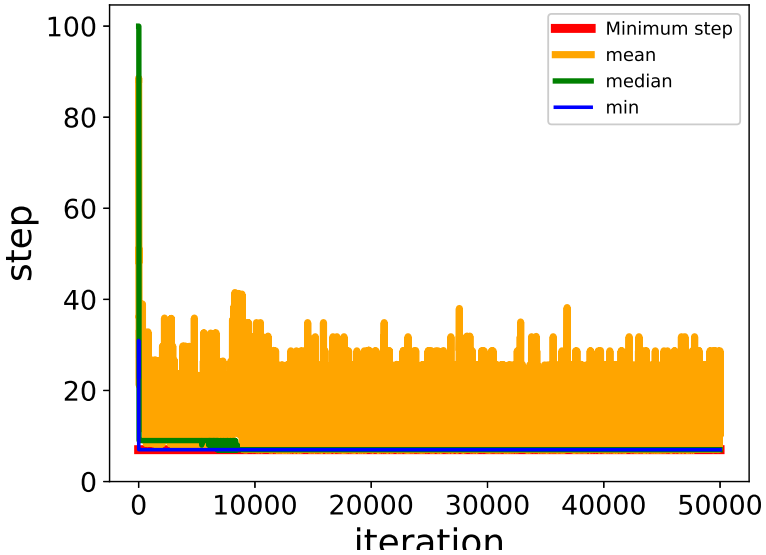


図 7.11　第 4 章の迷路における結果

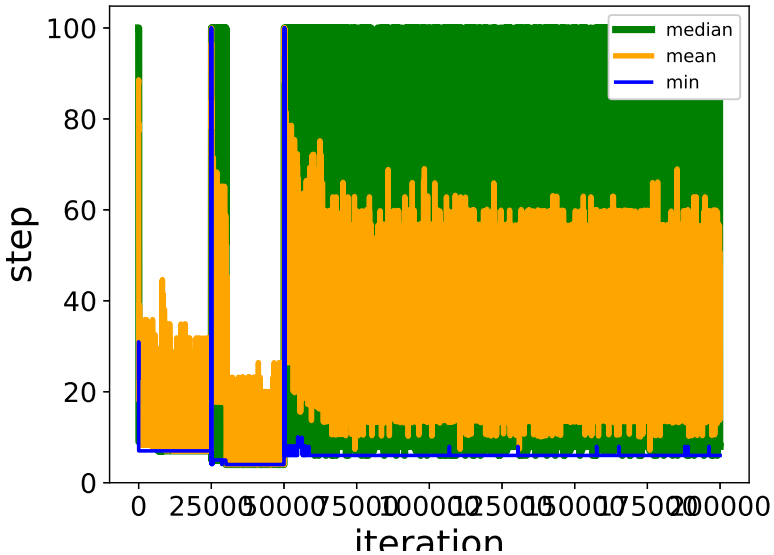


図 7.12　第 6 章の迷路における結果

ジェントはそのゴールまでの最短ステップ数の情報が古くなり，更新しなければならないが，ゴールへ到達できないがためにそれができない．その結果として学習が多少不安定になっていると考えられる．

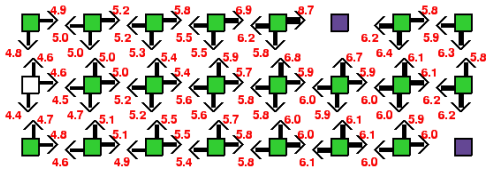


図 7.13　第 4 章の迷路におけるエージェント A の Q 値

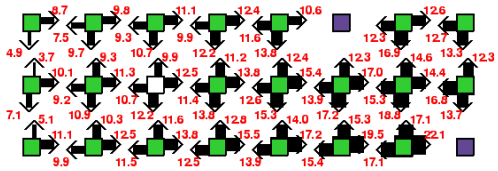


図 7.14　第 6 章の迷路におけるエージェント B の Q 値

7.4　提案手法の効果と限界

本論文は通信なしマルチエージェント強化学習における協調行動学習のための内部報酬の設計法を述べたが，これは「エージェントがゴールに到達するまでのステップ数の最大値を短くする」ことを良しとしたものである．これによる効果は下記の 2 点が挙げられる．

7.4.1　通信なしの協調

全エージェントが到達ステップ数をメモリに保存し，そのステップ数の最も大きな目的に達成することで，学習開始時の有利不利を利用して協調を実現する (多数の目的達成が容易なエージェントは困難な目的達成を目指す) ことに成功した．ただし，これは今回想定する問題においての話であり，物流システムの効率制御 (経路最適化) など今回の設計の有効性を示しうる問題も多々あるが，それが全てではなく，例えば CartPole 問題のように同じ状態をいかに長く維持できるかという問題であれば今回の内部報酬設計は適さないといえる．その理由は，明確なゴールが存在せず，ある状態を長時間維持することで獲得報酬が大きくなる問題であり，経路最適化とは性質の全く異なる問題であるからである．

7.4.2　ステップ数による制御

また，本研究の提案手法では，エージェント単体が持つ目的価値および内部報酬の合理性を証明することで，マルチエージェント全体の振舞いが，全エージェント到達可能なゴールの中では適切なものを選択可能であり，エージェントの行動により他エージェントが確実に到達不能となるゴールがある場合においては学習不可能となるという限界を示した．これは，通常のマルチエージェント強化学習においては利己的エージェントの一部を協調的に振る舞わせることに対し，すべてのエージェントが協調的に動き，一部を利己的に動かすことを目指す学習を行ったため，最もステップ数のかかる目的に到達するエー

ジェント 1 体の振舞いを制御することによってシステム全体の振舞いをある程度制御することができた．しかしながら，最もステップ数のかかるエージェントは問題に依存して変わるものであり，その決定法に関して今後必要となると思われる．

7.4.3　適用可能範囲

・状態行動空間

本論文では，迷路問題において手法の性能を検証したが，他の問題への適用性能を議論するためには，状態行動空間を考慮することが重要である．図 7.15 は 1 体のエージェントにおける状態行動空間の例である．図の円は状態であり，緑と赤はそれぞれスタートとゴールである．また黒矢印はそれぞれの状態において選択可能な行動を示しており，行動選択によってそれぞれ次状態に遷移する．この状態行動空間を全エージェントが持つのであれば (他エージェントはスタート位置が違う)，それぞれのゴールへ到達するための経路が存在し，ここでは環境が変化していないため，提案手法により協調行動を学習することができるといえる．

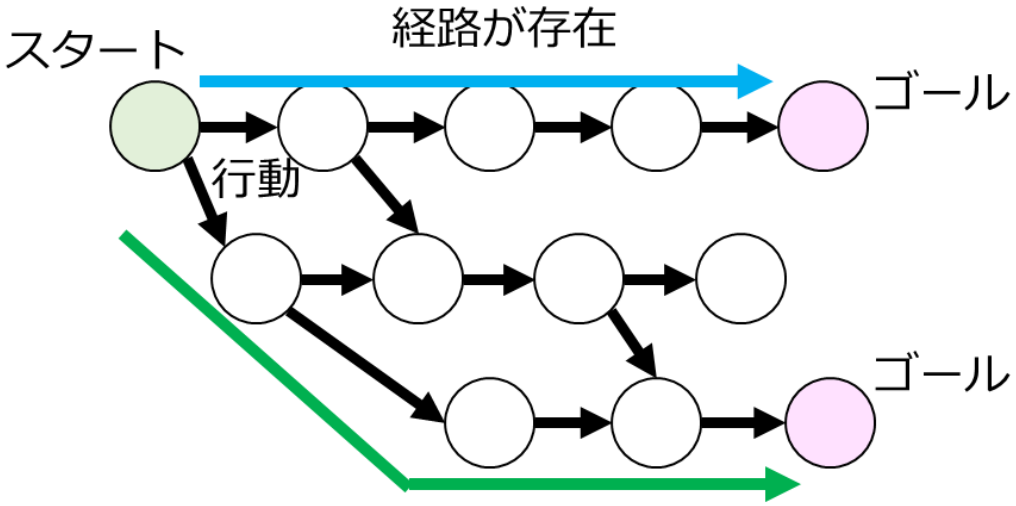


図 7.15　本提案手法の適用可能例

・静的環境において

表 7.2 は本論文において想定した環境の種類とそれぞれにおいて PMRL-DRES の合理性をまとめたものである．各行は各ゴールの報酬値が同一であるかそうでないかの違いであり，各列はエージェント数とゴール数が等しいかゴール数が多いかの違いを示している．なおエージェント数よりもゴール数が小さいケースはここでは想定していない．その理由は余ったエージェントがどうであれば良いか想定できないためである (到達すべき

エージェントの選別は PMRL-DRES でも可能であると考えられる．)．本研究における提案手法である PMRL-DRES はエージェント数とゴール数が等しい場合，ゴール選択法の合理性証明から本論文において定義する合理的選択ができていると言える．一方でゴール数が大きい場合，報酬値が同一であれば合理的に目的選択ができているが，理論的に示しておらず，報酬値が異なる場合においては優良なゴールを選択できているが合理的であることまでは言えていない．

表 7.2　本提案手法の適用範囲

	エージェント数=ゴール数	エージェント数<ゴール数
報酬値が同一	合理性：〇	合理性：△
報酬値が異なる	合理性：〇	合理性：×

・動的環境において

本研究における協調行動学習法はエージェントが各ゴールへ到達する経路の内どれを選択するかを学習するということに集約されている．加えて通信なしを実現するために，各エージェントのアーキテクチャを同一にして他エージェントの行動をある程度推定した上で協調行動を学習している．また利用する情報としては各ゴールへ到達するまでのステップ数と報酬値であるため，経路とならなくなる動的変化やステップ数と報酬値が変化するような動的変化に対しては弱いという特性を持つ．つまり，動的変化へ追従可能である条件として下記の 3 点を満たす必要がある．

1. 全エージェントの状態行動空間とゴール状態が等しいこと
2. 各ゴール到達までの経路が必ず存在すること
3. 最短ステップ数と報酬値が適切に獲得可能であること

以上を踏まえると，動的変化に関しては，その変化によりエージェント毎の状態行動空間およびゴール状態が異なるとき，状態行動空間の変化によりゴールへ到達することができなくなるとき，そして最短ステップ数と報酬値が適切に得られなくなるときに適切に協調行動を学習できなくなることがわかる．つまりそれはセンサの故障などによりエージェント毎に状態やゴールが変わってしまったとき，空間的環境変化などによりあるゴールへ到達することができなくなってしまったとき，そして断続的な環境変化により最短ステップ数と報酬値が真値ではなくなったときが考えられる．ここでは特に断続的環境変化に対する影響について考察する．断続的動的変化に対しては，より短いタイミングで動的変化を繰り返す場合には提案手法は適用することができない．より具体的に言えば問題のス

ケールに依存し，各ゴール到達までの経路長に依存する．それはつまりゴール到達までに学習が必要とするステップ数がその経路により変わるためであり，例えば図 7.15 では各ゴール到達までのステップ数が 4 ステップかかるため，少なくとも $4! \times 2 = 48$ ステップは必要であるという計算になる (4! はゴール報酬をスタート地点まで伝播させるために必要なステップ数を示しており，2 はゴール数を示している．)．しかしながら実際はゴールへ何度も到達して報酬を獲得できる割合を計算し，そしてゴールまで最短ステップ数で到達することで最短ステップ数を記録することが必要となるため，実際はより多くのステップ数が必要となる．またエピソードで考えると，今回は最短ステップ数や報酬値に雑音が存在しない環境を想定しているため，エージェントの初期の行動選択におけるランダム性が合えば，ゴール数分の断続変化のみで追従することが可能である．しかし実際はそのようなことが起こることはないため，環境の状態全てを網羅した学習をするだけのステップ数とエピソード数が必要となる．実験的結果を述べると，3×8 の迷路においては 3000 から 5000 エピソードで環境全体を網羅することができる．

7.4.4　エージェントの故障

エージェントの故障により 1 体のエージェントの到着するタイミングが大きく異なる場合，故障如何に関わらずゴールへ到達可能である場合，故障を考慮した最短ステップ数に従って学習するため，故障していないエージェントが遠くのゴールへ到達し，故障したエージェントは近くのゴールへ到達し，エージェント全体として適切な行動を学習できる．一方で，エージェントの故障により，到達できないゴールが存在する場合，ゴールへの最短ステップ数が更新され，到達できないゴールに対する最短ステップ数が存在しなくなるため，そのゴールへ向かう行動を学習することがなくなるため，エージェント同士別々のゴールへ到達することができる．しかしながら，最短ステップ数でゴールに到達するためには，各エージェントが目的価値により到達するゴールが等しくなる必要がある．最後に，エージェントの故障により到達できるゴールが存在しない場合，残るエージェントはステップ数の大きなゴールへ到達するように学習する．このときステップ数が最短であるためには到達可能なゴールが自身の最も近くのゴールであるか，自身の最も近くのゴール以外のゴールが到達できないくらいに遠くにある場合である必要がある．

7.4.5　環境が非常に小さい (大きい) 場合

・環境が非常に小さい場合

市川らの取り上げた，狭路すれ違い問題 [45] における PMRL-DRES の性能について議論する．図 7.16 は実際の狭路すれ違い問題を示している．図の “Start A”，“Start B” はそれぞれエージェント A，B のスタート位置を示し，“Goal A”，“Goal B” はそれぞれ

エージェント A と B のゴールである．そして，各エージェントは指定のゴールに到達することを目指す．この問題はエージェント同士衝突が起こるため，エージェント B が狭路の上にある窪みに移動して，エージェント A を通過できるようにさせる必要がある．この問題は全エージェントの位置を状態としているため，通常の強化学習でも解くことができ，PMRL-DRES でも解くことができる．なぜなら，このときの状態行動空間は，スタート地点とゴール地点がそれぞれ 1 個の空間となり，ゴールの選択により起こる問題が存在しないためである．しかしながら，本研究ではエージェント A と B の位置を知るような想定ではない，通信なしを想定すると，MDP ではなくなる (つまり，エージェントの動作により同じ状態でも適切な行動が異なる) ため，適用することはできない．ただし，1 つ分過去の状態をと現在の状態をセットにして 1 つの状態とし，エージェントの動作として上下左右とその場にとどまるというものを追加することで MDP となり，全エージェント共通の状態行動空間を持つため，PMRL-DRES が適用可能であるといえる．

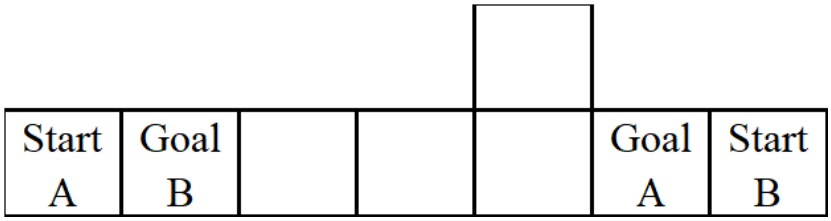


図 7.16　狭路すれ違い問題

・環境が非常に大きい場合

次に，環境が非常に大きく，エージェントが到達できないゴールが存在するとき，PMRL-DRES はその到達できないゴールを除いたゴールの中から適切なものを選択し，そこに到達するように学習する．その理由は，PMRL-DRES の機構が，獲得報酬期待値が閾値を超えるゴールのみを考慮して目的価値を推定し，学習するためである．

第 8 章

結論

8.1　知見のまとめ

本研究では，静的環境における他エージェント情報を利用しない協調行動学習法 PMRL を起点として，空間的環境変化に展開した PMRL-OM を提案，そしてそれを更にエージェント数とゴール数の環境変化に展開した PMRL-DRES を提案するという方法を取っている．そのため，手法による細かな違いはあるものの，PMRL-DRES が PMRL-OM と PMRL の機能を包含しており，PMRL や PMRL-OM にて取り上げた様々な問題は PMRL-DRES で解くことができる．

本論文においては，まず，第 1 章にて，各エージェントが強化学習により，複雑な問題を解決可能なことが期待できることを述べ，従来は，通信を前提とした上で，学習環境が静的であるため，現実問題に適用する上での性能が保証できないという問題を提起した．そしてそれを解決し，静的・動的環境における通信なしマルチエージェント強化学習法の提案とその合理性保証を目的とすることを述べた．また，第 2 章では，マルチエージェント強化学習について詳しい説明を述べ，他のエージェントの振舞いを知ることなく協調行動を学習することの難しさを具体的に説明した．そして，本研究の位置づけとして，Q 学習における学習結果の収束性に着目し，その収束値を適切な目的に向けるための報酬設計が重要であることを述べた．そして，第 3 章では，本研究にて扱う迷路問題とその動的変化について説明し，本研究における到達目標を明らかにした．

また，本研究における成果は下記の通りであり，下記の知見全てを PMRL-DRES が備えている．

・静的環境におけるエージェント間協調

第 4 章の知見により，本研究における提案手法は 2 体エージェントにおける協調が必要な迷路問題において，その合理性を保証している．そして実験結果により 5 種類の迷路問題においてその有効性を検証し，その合理性を確認した．3 体以上のエージェントについ

ては 4 章では明確に合理性を示していないが，付録 A にて任意数エージェントにおける協調行動学習とその合理性について議論しており，現状として多少の他エージェント情報を必要とするものの，合理性を示している．

・空間的環境変化

第 5 章の知見により，最短ステップ数の更新方法及び目的価値の更新式を変更し，より新しい情報に重み付けすることで空間的環境変化に追従可能であることを示した．また合理性保証により，空間的環境変化において前節の PMRL と同等の効果を発揮することを示した．実験では空間的環境変化のすべてのケースにおいて変化する環境に追従し，目的価値の大小関係を変化させることで協調行動を確率 1 で学習することを示した．またパラメータ ξ の設定によりエージェントの持つ学習初期の情報をどれほど利用するかが設定可能であり，環境変化からの学習回数が関わるもののパラメータはある程度の自由度を持って設定することができることを示している．そして断続的に複数回環境変化が起きた際にもそれに素早く適応して，学習のやり直しを防ぐことに成功している．そしてそれは環境変化の回数に依存せず，環境変化の間のエピソード数が少ない場合であっても適応可能である．

・エージェント数とゴール数の環境変化

第 6 章の知見により，学習環境を分割することで，学習結果を利用できる部分は利用し，新たに学習すべき部分を学習することでエージェント数とゴール数の環境変化に追従可能であることを示した．実験ではまず，エージェント数とゴール数が増える迷路問題においてそれに追従し，全エージェントの到達ステップ数を最小にした．また分析として環境変化直後に学習をリセットするハードリセットとの比較を行い，学習環境の分割により適切に学習結果を利用できていることを示した．また学習環境の分割に利用したパラメータもある程度自由度を持って設定可能であることを示している．次にゴール数が増え，報酬値が動的変化する迷路問題において有効性を示し，最後に空間的環境変化とゴール数，報酬値の動的変化を組み合わせた環境にて有効性を示した．その際目的価値設定による目的選択が最適ではないものの，適切 (2 番目に最適) な目的を選択できていることを確認した．また「空間的環境変化」と「エージェント・ゴール数変化」が複合した環境においても素早く適応し，学習のやり直しを防ぎ，全動的変化に適応可能であることを示した．

最後に，第 7 章では，第 4 章，第 5 章，第 6 章において提案した 3 つの手法 (PMRL, PMRL-OM, PMRL-DRES) をまとめ，全体として静的・動的環境における適用性及びその効果について示した．PMRL-DRES は経路探索問題に適用可能であり，適切な経路を探索できることを示した．

8.2 今後の課題

今後の展開として，本研究で採用した例題以外にも提案手法の適用範囲を広げ，そして更なる精度向上のため，以下の課題に取り組む必要がある．

8.2.1 提案手法に関する課題

・PMRL 固有の問題: スケーラビリティ

スケーラビリティとしてはエージェント数とゴール数の関係が考えられる．エージェント数よりもゴール数が大きい場合は，付録に記載のゴール到達可能範囲 gr を導入した PMRL で解くことができる．また，エージェント数の方が大きい場合は，そもそも何をもって良しとするかがわからないため考慮しない．ただ，少なくとも PMRL によって，ゴール数と同等の数のエージェントが適切な Q 値を推定し，それ以外のエージェントはどのゴールにも Q 値を適切に推定できないため，適切なエージェントをゴールに到達させることはできる．

また，本研究では，任意数エージェントにおける PMRL 適用のために，到達ゴールの制限法を提案し，その有効性を示したが，事前知識を必要とするという問題点がある．その一方で，動的変化において提案した PMRL-DRES の学習分割により，任意数エージェントにおける通信なし協調行動学習が成功する例が今回確認された．このことから現状の PMRL には改善の余地があり，そして PMRL-DRES における学習分割を理論的に分析し，任意数エージェントにおける学習の適切な分割方法を探究することにより PMRL の性能向上を目指す．また今回は環境の設定として達成すべき目的が分かっているが，協調を達成する上で各エージェントの達成すべき目的が分からない時に，各目的達成時に報酬の獲得可能回数を 1 回に定め，各エージェントが各エピソードで獲得する報酬値から提案手法により協調行動を学習させる手法であった．そのため，正確な意味で従来手法が存在せず，今回の比較対象である Q 学習は適切であるかは疑問が残る．今後の課題としては環境の設定から等しい従来法を用意するか，または適切ではないが全エージェントの獲得報酬を共有できるという想定を置いたうえで，獲得報酬の合計をステップ数で割った値を報酬値とした Q 学習において比較することも提案手法の性能を比較する上では必要となる．

・PMRL-OM 固有の問題: 忘却関数の設計

PMRL-OM の忘却関数はどの値に設定しても PMRL と同等の性能を示すことを理論的に明らかにしたが，これは Q 学習同様無限回のゴール到達を前提にしており，学習途中の振舞いを示すものではない．そのため，今回の例題における結果でも忘却関数のパラ

メータ設定により性能の低下が確認された．これは PMRL-OM にまだ改善の余地が残されていることを示しているが，今回の分析によりある程度例題の種類に依存することが判明している．以上から，今後は観測情報を利用した忘却関数のパラメータの動的設定により更なる性能向上を目指す．

・PMRL-DRES 固有の問題: 最適な学習領域分割法

PMRL-DRES は従来手法と比較しても短時間でより多くの報酬獲得を可能だが，その一方で例題における最適値よりは性能が落ちるという問題点が存在する．その理由は各エージェントの基準とエージェント全体の基準に未だにずれが存在し，PMRL-DRES の学習領域制限にまだ改良の余地があることを示している．より具体的には，分割基準を数式的に分析した結果，PMRL-DRES は各エージェントに対し報酬値の大きいゴールを優先的に学習させるように学習領域分割をしていることが分かっている．そのため，分割基準における報酬値の優先度が小さくなるように設定して，より全体の基準に近づくような改良を施すことで，更なる性能向上に取り組む．

また，PMRL-DRES は図 8.1 の通り，全エージェントが最も遠くのゴールへ到達することを防止することができる．つまり，各エージェントがそれぞれ異なるゴールへ到達した上で，その到達までのステップ数が最大になることはないといえるはずである．しかし現状としてそれを理論的に示しているわけではない．そのため，今後の課題として PMRL-DRES のメカニズムを定式化し，最悪のケースを回避可能であることを示すことが求められる．

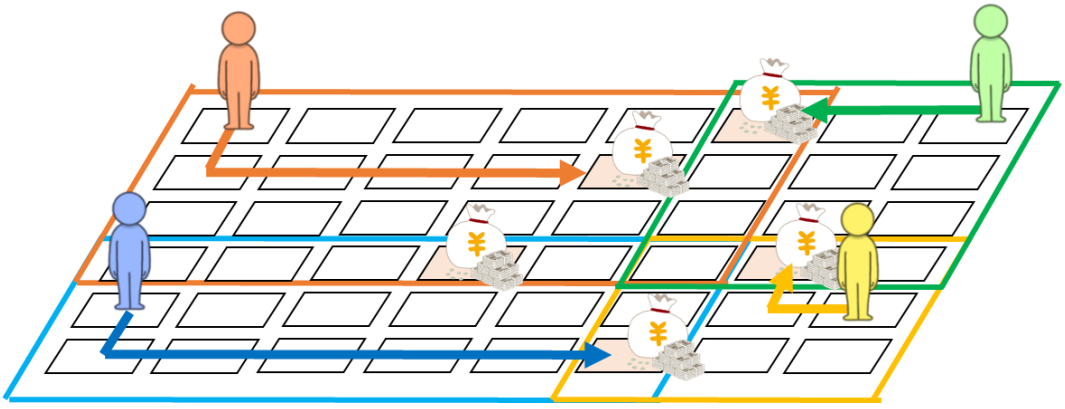


図 8.1　PMRL-DRES の効果

8.2.2　本研究にて扱う動的変化に関して

本研究は動的環境として，ある時点から別の環境に変化するという動的環境を取り上げ，その中で提案手法の有効性を検証した．しかしながら動的環境は本研究にて想定する

もの以外にも存在する．特にそれは環境変化のタイミングにおいて顕著であり，大きく分けると下記の 2 つが考えられる．断続的環境変化とはある時点から別の環境に変化するということが何回か起こる環境変化を表しており，本研究において想定する環境変化は断続的環境変化であるといえる．一方で連続的環境変化とは，環境変化の切れ目なく時間に合わせて徐々に別の環境へ推移するような環境変化を表しており，文献 [19] で取り上げられた追跡問題などは連続的環境変化の一例であるといえる．ここではそれぞれに対する課題を述べる．

1. 断続的環境変化：ある時点から環境が何回か変化する
2. 連続的環境変化：環境が連続的に別の環境へ変化していく

断続的環境変化に関する課題

今回は環境変化のタイミングが 1 回であり，変化前後の環境における学習回数は十分にとれるようになっていた．それは今回合理的保証を併せて行っており，その検証も兼ねていたという点においては有効性を示しているが，限られた学習回数における環境変化への適用そして，何回か環境変化お起こしたときにそれに適応することができるかについて取り組むことは今後重要な課題となるといえる．現状でいえば，環境変化の前後であまり環境が変化していない場合において，今回の提案手法は少ない学習回数においても追従することができると考えられる．また，何回か環境変化が起こった時に，学習回数の問題を考えなければ必ず追従することができると考えられる．また学習回数が少ない場合においても，前述の通り変化量が少ない場合には十分追従することができるといえる．しかしながら，現実問題を考えると，環境が大きく変化することもあり，それに少ない学習回数で追従していくことが求められる．そこで，本研究の PMRL-DRES では学習環境の分割により学習結果の活用を試みたが，それをさらに超えて別の状態で学習した結果をうまく別の状態に活用することを考えることが本研究の次のステップとして重要な課題となるといえる．

連続的環境変化に対する課題

本研究では連続的環境変化を対象としなかったが，それは本研究の主眼が他エージェント情報を利用することなく，環境変化に追従した協調行動学習法を提案し，その合理性を示すというところにあり，特に MDP になり得ない各エージェントの動作を 1 つに収束させることで疑似的に MDP 環境とすることにあったためである．そして変化する環境をうまく切り分けて MDP 環境と等しくすることを目指したためである．そのため，今後の課題として連続的環境変化に取り組むことは意義があるといえる．特に現状の PMRL-DRES においても，連続的環境変化が小さいのであれば，状態観測を疎にとるこ

とによってある程度追従することができる可能性がある．大きな環境変化に対してはより素早く変化する環境に追従する必要があり，観測する状態や報酬の変化も利用して追従する手法の提案により解決を目指す．

8.2.3 取り組む問題と実応用展開

本研究では例題として迷路問題を想定して手法を提案したが，問題変数を設定しているため，他の問題においても適用可能性がある．しかしながらその検証は行っていないため，他のマルチエージェント強化学習の問題において本提案手法を適用し，提案手法の一般性を検証する必要がある．具体的には，Suhkbaatar らの考案している [46]，迷路形状に基づいたエージェントが協力してコインを集める問題や片方のエージェントがスイッチを押し，もう片方のエージェントをゴールに導く問題などに取り組み，提案手法の有効性と限界を検証する必要がある．また，実問題への適用も急務となっており，現実を想定した通信を配した動的環境上での協調行動学習のための内部報酬設計法を提案していることから，このような特性を持つ実問題への適用が求められる．具体的には，今マルチエージェント強化学習においてはアマゾンロボティクスの物流システム [5] などへの展開が考えられ，今後は提案手法の物流システムへの適用を目指す．特に物流システムにおいては，複数種類存在するタスク (スタートからゴールへ到達する) が動的に割り当てられ，更にその時点に位置する倉庫ロボットも異なるため，各タスクに対する各ロボットの適性が動的変化する中で各ロボットにうまくタスクを分配していく必要があり，それが今後取り組む上での大きな課題となると考えられる．

謝辞

本論文をまとめるにあたり始終多大なるご指導と御教示をいただいた主任指導教員である髙玉圭樹教授，指導教員の西野哲朗教授，大須賀昭彦教授，そして博士論文の審査をしていただいた庄野逸教授，佐藤寛之准教授に心より感謝の意を表します．また研究会に積極的に誘っていただき，その中でご助言いただくだけでなく，研究者として活動する上での様々なサポートをして頂いた下原勝憲教授，研究を進める上で学習領域を分割するという案の基となる助言をいただいた Tim Kovacs 博士，学会で重要な助言を多々いただきました宮崎和光准教授，山口智浩教授，卒業後も様々な学会にてご助言いただきました先輩の原田智広助教，中田雅也准教授，並びに日頃より様々な議論をして頂きました研究室の方々に感謝申し上げます．そして，特別研究員制度によって研究費の面で助成いただいた日本学術振興会様，各助成金にて研究生活をサポートしていただいた電気通信大学に感謝申し上げます．最後に金銭面に限らず日ごろより私を支援していただきました，家族の皆様である上野広文，上野佐和子，上野みなみ，皿井聖，皿井登幾子，難波勉，難波加代子，上野眞二，上野典子に対してこの場を借りてあらためて深く感謝申し上げます．

参考文献

[1] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML2018)*, pages 4295–4304, 2018.

[2] Eugenio Bargiacchi, Timothy Verstraeten, Diederik M Roijers, Ann Nowé, and Hado van Hasselt. Learning to Coordinate with Coordination Graphs in Repeated Single-Stage Multi-Agent Decision Problems. In *Proceedings of the 35th International Conference on Machine Learning (ICML2018)*, pages 482–490, 2018.

[3] David L. Leottau, Javier Ruiz-del Solar, and Robert Babuška. Decentralized Reinforcement Learning of Robot Behaviors. *Artificial Intelligence*, 256:130–159, 2018.

[4] 荒井 幸代. マルチエージェント強化学習 : 実用化に向けての課題・理論・諸技術との融合. **人工知能学会誌**, 16(4):476–481, 2001.

[5] Wolfgang Hönig, Scott Kiesel, Andrew Tinka, Joseph W. Durham, and Nora Ayanian. Conflict-based search with optimal task assignment. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS2018)*, pages 757–765, 2018.

[6] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the 10th International Conference on Machine Learning (ICML1993)*, pages 330–337, 1993.

[7] Ming Wu, Feifei Huang, Long Wang, and Jiyin Sun. Cooperative multi-robot monocular-slam using salient landmarks. In *Proceedings of the 2009 International Asia Conference on Informatics in Control, Automation and Robotics*, pages 151–155, 2009.

[8] Peter Stone, Richard S Sutton, and Gregory Kuhlmann. Reinforcement learning for robocup soccer keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.

[9] Shun-Ichi Azuma, Ryota Yoshimura, and Toshiharu Sugie. Broadcast control of multi-agent systems. *Automatica*, 49(8):2307–2316, 2013.

[10] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. *Proceedings of the 11th International Conference on Machine Learning (ICML1994)*, pages 157–163, 1994.

[11] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[12] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML2016)*, pages 1928–1937, 2016.

[13] Mohammad Al-Zinati and Rym Wenkstern. Agent-environment interactions in large-scale multi-agent based simulation systems. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS2019)*, pages 763–771, 2019.

[14] Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus. Modeling others using oneself in multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML2018)*, volume 80, pages 4257–4266, 2018.

[15] Karl Tuyls and Gerhard Weiss. Multiagent learning: Basics, challenges, and prospects. *AI Magazine*, 33(3):41–52, 2012.

[16] Andrei Marinescu, Ivana Dusparic, and Siobhán Clarke. Prediction-based multi-agent reinforcement learning in inherently non-stationary environments. *ACM Transaction on Autonomous Adaptive System (ACM TAAS)*, 12(2):9:1–9:23, 2017.

[17] Patrick MacAlpine, Daniel Urieli, Samuel Barrett, Shivaram Kalyanakrishnan, Francisco Barrera, Adrian Lopez-Mobilia, Nicolae Stiurca, Victor Vu, and Peter Stone. UT Austin Villa 2011: A champion agent in the RoboCup 3D soccer simulation competition. In *Proceedings of the 11th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS2012)*, pages 129–136.

[18] Fei Fang, Peter Stone, and Milind Tambe. Defender strategies in domains involving frequent adversary interaction. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS2015)*, pages 1663–1664, 2015.

[19] Sachiyo Arai and Sycara Katia. Effective learning approach for planning and scheduling in multi-agent domain. In *Proceedings of the 6th International Conference on Simulation of Adaptive Behavior (SAB2000)*, pages 507–516, 2000.

[20] Wiem Zemzem and Moncef Tagina. Cooperative multi-agent learning in a large dynamic environment. In Vicenc Torra and Torra Narukawa, editors, *Modeling Decisions for Artificial Intelligence*, pages 155–166. Springer International Publishing, 2015.

[21] Junling Hu and Michael P. Wellman. Nash q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003.

[22] Hiroaki Iwashita, Kotaro Ohori, Hirokazu Anai, and Atsushi Iwasaki. Simplifying urban network security games with cut-based graph contraction. In *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems (AAMAS2016)*, pages 205–213, 2016.

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[24] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *Proceedings of the 35th International Conference on Machine Learning (ICML2018)*, pages 1515–1528, 2018.

[25] Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1st edition, 1998.

[26] Christopher J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, 1989.

[27] John J. Grefenstette. *Credit Assignment in Rule Discovery Systems Based on Genetic Algorithms*, volume 3. Kluwer Academic Publishers, 1988.

[28] 石田亨. エージェントを考える. **人工知能学会誌**, 10(5):663–667, 1995.

[29] Felipe Leno Da Silva, Matthew E. Taylor, and Anna Helena Reali Costa. Autonomously reusing knowledge in multiagent reinforcement learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI2018*, pages 5487–5493, 2018.

[30] Sam Devlin, Logan Yliniemi, Daniel Kudenko, and Kagan Tumer. Potential-based difference rewards for multiagent reinforcement learning. In *Proceedings*

of the 2014 International Conference on Autonomous Agents and Multi-agent Systems (AAMAS2014), pages 165–172, 2014.

[31] Maxim Egorov. Multi-agent deep reinforcement learning. *CS231n: Convolutional Neural Networks for Visual Recognition*, pages 1–8, 2016.

[32] Sagar Verma, Richa Verma, and PB Sujit. Mapel: Multi-agent pursuer-evader learning using situation report. In *Proceedings of 2019 International Joint Conference on Neural Networks (IJCNN2019)*, pages 1–8, 2019.

[33] Daisuke Shiraishi, Kazuteru Miyazaki, and Hiroaki Kobayashi. Proposal of detour path suppression method in ps reinforcement learning and its application to altruistic multi-agent environment. In *Proceedings of the 21st Principles and Practice of Multi-Agent Systems (PRIMA2018)*, pages 638–645, 2018.

[34] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015.

[35] Daisuke Shiraishi, Kazuteru Miyazaki, and Hiroaki Kobayashi. Proposal and evaluation of detour path suppression method in ps reinforcement learning. *SICE Journal of Control, Measurement, and System Integration*, 12(5):190–198, 2019.

[36] Hongliang Guo and Yan Meng. Distributed reinforcement learning for coordinate multi-robot foraging. *Journal of Intelligent and Robotic Systems*, 60(3):531–551, December 2010.

[37] Sainbayar Sukhbaatar, Ilya Kostrikov, Arthur Szlam, and Rob Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. *Computer Research Repository*, 2017.

[38] Adrien Baranes and Pierre-Yves Oudeyer. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Computer Research Repository*, 2013.

[39] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML1999)*, pages 278–287, 1999.

[40] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML2000)*, 2000.

[41] Xingyu Wang and Diego Klabjan. Competitive multi-agent inverse reinforcement learning with sub-optimal demonstrations. In *Proceedings of the 35th Interna-*

tional Conference on Machine Learning (ICML2018), pages 5143–5151, 2018.

[42] Fumito Uwano and Keiki Takadama. *Communication-Less Cooperative Q-Learning Agents in Maze Problem*, pages 453–467. 2016.

[43] Kazuteru Miyazaki, Koudai Furukawa, and Hiroaki Kobayashi. Proposal of pswithefp and its evaluation in multi-agent reinforcement learning. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 21(5):930–938, 2017.

[44] Autostore: Space saving storage and order picking system for small parts. https://www.swisslog.com/en-au/products-systems-solutions/asrs-automated-storage-retrieval-systems/boxes-cartons-small-parts-items/autostore-integrator.

[45] 市川嘉裕 and 髙玉圭樹. 学習進度に基づくマルチエージェント q 学習における競合回避. **計測自動制御学会論文誌**, 48(11):764–772, 2012.

[46] Sainbayar Sukhbaatar, Arthur Szlam, Gabriel Synnaeve, Soumith Chintala, and Rob Fergus. Mazebase: A sandbox for learning from games. *Computer Research Repository*, 2015.

関連論文の印刷公表の方法および時期

1. 全著者名：上野　史，髙玉 圭樹
   論文題目：目的制限に基づく通信なしマルチエージェント協調行動学習とその効果の証明
   印刷公表の方法および時期：電気学会論文誌 *C*, Vol.140, No.1, 2020
   （第 A 章に関連）
2. 全著者名：Fumito Uwano and Keiki Takadama
   論文題目：Utilizing Observed Information for No-Communication Multi-agent Reinforcement Learning toward Cooperation in Dynamic Environment
   印刷公表の方法および時期：*SICE Journal of Control, Measurement, and System Integration (SICE JCMSI)*, Vol.12, No.5, pp.199-208, 2019.
   （第 5 章に関連）
3. 全著者名：Fumito Uwano, Naoki Tatebe, Yusuke Tajima, Masaya Nakata, Tim Kovacs, and Keiki Takadama
   論文題目：Mulit-Agent Cooperation Based on Reinforcement Learning with Internal Reward in Maze Problem
   印刷公表の方法および時期：*SICE Journal of Control, Measurement, and System Integration (SICE JCMSI)*, Vol.11, No.4, pp.321-330, 2018.
   （第 4 章に関連）
4. 全著者名：Fumito Uwano and Keiki Takadama
   論文題目：Comparison Between Reinforcement Learning Methods with Different Goal Selections in Multi-Agent Cooperation
   印刷公表の方法および時期：*Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII)*, Vol.21, No.5, pp.917-929, 2017.
   （第 4 章に関連）
5. 全著者名：Fumito Uwano and Keiki Takadama

論文題目：Strategy for Learning Cooperative Behavior with Local Information for Multi-agent Systems
印刷公表の方法および時期：*PRIMA 2018: Principles and Practice of Multi-Agent Systems*, pp.663-667, 2018.10.
（第 6 章に関連）

6. 全著者名：Fumito Uwano and Keiki Takadama
論文題目：Communication-Less Cooperative Q-Learning Agents in Maze Problem
印刷公表の方法および時期：*Intelligent and Evolutionary Systems*, pp453-467, 2016.12.
（第 4 章に関連）

7. 全著者名：Fumito Uwano, Naoki Tatebe, Masaya Nakata, Keiki Takadama, and Tim Kovacs
論文題目：Reinforcement Learning with Internal Reward for Multi-Agent Cooperation: A Theoretical Approach
印刷公表の方法および時期：*Proceedings of the 9th EAI International Conference of Bio-inspired Information and Communications Technologies*, pp.332-339, 2015.12.
（第 4 章に関連）

8. 全著者名：上野　史，髙玉 圭樹
論文題目：エージェント間通信を伴わず環境状態および報酬の包括的動的変化に追従する理論的マルチエージェント強化学習
印刷公表の方法および時期：合同エージェントワークショップ&シンポジウム *2019 (JAWS2019)*，大分，2019.9.
（第 6 章に関連）

9. 全著者名：上野　史，髙玉 圭樹
論文題目：非通信マルチエージェント強化学習における獲得報酬値の変動を用いたエージェント数の動的変化への追従
印刷公表の方法および時期：第 *18* 回情報科学技術フォーラム *(FIT 2019)*, 岡山, 2019.9.
（第 6 章に関連）

10. 全著者名：上野　史，髙玉 圭樹
論文題目：報酬の動的変化に適応する通信無しマルチエージェント協調学習のための公平性に基づく内部報酬設定法
印刷公表の方法および時期：計測自動制御学会システム・情報部門学術講演会

2018，富山，2018.11.

（第 6 章に関連）

11. 全著者名：上野　史，髙玉 圭樹

    論文題目：知識の忘却に基づく迷路形状の変化に追従する非通信マルチエージェント強化学習

    印刷公表の方法および時期：計測自動制御学会システム・情報部門学術講演会 *2017*, SS13-4, 静岡, 2017.11.

    （第 5 章に関連）

12. 全著者名：上野　史，建部 尚樹，中田 雅也，髙玉 圭樹

    論文題目：ジレンマ問題におけるマルチエージェント間協調のための内部報酬推算

    印刷公表の方法および時期：第 *42* 回知能システムシンポジウム, F-11, 兵庫, 2015.3.

    （第 4 章に関連）

付録

A　任意数エージェントの協調行動学習の理論的証明

ここでは適切にゴール到達可能範囲 gr というものを定義し，それを適切に設定することで，任意数エージェントがゴールに到達するまでのステップ数が最悪値にならないことを示す．

A.1　目的選択法

ゴール到達可能範囲 gr は下記の式により決定する．式 (1) は gr を求める式，式 (2) は式 (1) を求めるために必要な $GoalNumber$ 関数の定義式，式 (3) は到達可能な最も遠くのゴールの識別番号 fg を計算する式，そして式 (4) は fg を計算するために必要な $FarthestGoal$ 関数の定義式である．なお，ここではエージェント数ゴール数ともに等しいという前提を置いており，各ゴールにどのエージェントが到達すべきか一意に決まるものを考える．$FarthestGoal$ 関数はエージェントの識別番号 i と任意のゴールの識別番号 g を与え，g のゴールがエージェント i にとって達成するのに最もステップ数を必要とするゴールであれば 1，そうでなければ 0 を返す関数である．これを使うことで，式 (3) ではそれぞれの最も遠くのゴールの識別番号 g が等しいエージェントの数が分かる．そして 0 は除外した上でその中で最も小さいゴールの識別番号を fg に代入する．そして $GoalNumber$ 関数は最短ステップ数配列 $\boldsymbol{t^i}$，任意のゴール識別番号 g と式 (3) にて計算した fg を引数として，g のゴールが fg のゴールよりも到達するまでの時間が同等か小さければ 1 を返し，そうでなければ 0 を返す関数である．これを使い，式 (1) では到達までの最短ステップ数がゴール fg 以下のゴール数を計算し，それをゴール到達可能範囲

gr^i として代入する．

$$gr^i = \sum_g GoalNumber(t^i, g, fg) \tag{1}$$

$$GoalNumber(t^i, g, fg) = \begin{cases} 1 & if \quad t^i_g \leq t^i_{fg} \wedge g \leq fg \\ 0 & otherwize \end{cases} \tag{2}$$

$$fg = \arg\min_g \sum_i FarthestGoal(i, g) \neq 0 \tag{3}$$

$$FarthestGoal(i, g) = \begin{cases} 1 & if \quad t^i_g > t^i_{og} (\forall og \in G) \\ 0 & otherwize \end{cases} \tag{4}$$

なお，上記の条件に従い各エージェントの gr^i を設定していくが，このとき式 (5) が条件となる．この条件を満たさなかった時，その時のゴール fg を除外した状態で，式 (1) から (4) の工程を繰り返す．$FarGr$ 関数は 2 体の任意のエージェントの識別番号 i と j を引数として，gr^j の方が gr^i よりも大きかった場合 1 を返し，そうでなければ 0 を返す関数である．また，式 (5) において $AgentNum$ はエージェント数，I はエージェントの識別番号集合を示す．つまり，任意のエージェント i のゴール到達可能範囲 gr^i は，全体のエージェント数から gr^i よりも大きいゴール到達可能範囲を持つエージェント数を引いた数以上でなければならない．そうしなければ，gr^i により 1 個のゴールを 2 体のエージェントで取り合うというような状況を招くためである．なお，本研究ではゴール到達可能範囲をあらかじめ事前知識として計算し，エージェントに適用する．

$$gr^i \geq AgentNum - \sum_j^I FarGr(i, j) \tag{5}$$

$$FarGr(i, j) = \begin{cases} 1 & if \quad gr^i < gr^j \\ 0 & otherwize \end{cases} \tag{6}$$

A.2 効果の理論的保証

ここではゴール到達範囲を上記に従い設定することで，全ゴール到達までのステップ数が最大値を回避することを示す．任意数エージェント用の PMRL では，各エージェントの最も遠くのゴールの識別番号が fg となるようにそれぞれ別々の gr^i を持つ．そして PMRL のアルゴリズムに従って協調行動を学習する．PMRL はすべてのエージェントが最も遠くのゴールへ向かい，遠くのゴールに最も近いエージェントがそのゴールへ到達することを良しとするように学習する．つまり，任意数エージェント用の PMRL により全エージェントが識別番号 fg のゴールへ到達するステップ数の最小値が全エージェントが全ゴールへ到達するまでのステップ数となる．これを数式で表すと以下のようになる．

$$\min_i t^i_{fg} < t^j_{fg} (\forall j \in I) \tag{7}$$

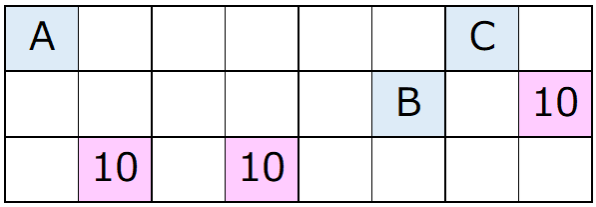


図 A.2　ケース 1 の迷路

ここで任意のエージェント i のゴール fg に到達するステップ数が最も小さいとすると，式 (7) の左辺は t^i_{fg} ということになる．また，識別番号 fg のゴール以外のゴール到達可能範囲 gr^j 内の任意のゴールの識別番号を g とすると，下記の式 (8) が成り立つ．

$$t^j_g < t^j_{fg} \ (g < fg) \tag{8}$$

任意のエージェント j は gr^j の範囲内にある任意の識別番号 g のゴールへ到達するように学習する．よってこのときすべてのエージェントは少なくとも t^j_{fg} よりも小さいステップ数で各ゴールへ到達することが分かる．また，式 (7) から識別番号 fg のゴールに関しては PMRL がステップ数が最小となるように学習する．以上から，任意数エージェント用の PMRL は少なくとも $\max_{i \in I} t^i_{fg}$ よりも小さいステップ数でエージェントを全ゴールへ到達させることができる．つまり，最大のステップ数で全ゴールへ到達するように学習することはないと言える．

A.3　実験: 任意数エージェントにおける協調

実験設定

任意数エージェント用の PMRL の有効性を検証するため，本研究では 3 体，5 体のエージェントにおける迷路に適用させ，PMRL と比較する．具体的には図 A.2，A.3 の 2 種類の迷路（以降それぞれの実験をケース 1，2 と呼ぶ）に対し任意数エージェント用の PMRL と PMRL の性能を比較する．各図において，各マスが状態を表し，A，B，C，D，E と表示されたマスがそれぞれのエージェントの初期状態を表し，数字の書いてあるマスがゴールを表す．また，ゴールに書かれた数字はそのゴールに到達した際に得られる報酬値を表している．　本実験はすべてのエージェントが全ゴールに到達するまでのステップ数を評価する．各エージェントはそれぞれのゴール到達までに費やしたステップ数の最大値が小さくなる時もっとも良い状況として評価される．なお，学習と評価は別々に行い，各アルゴリズムで学習が終了した後，学習を抜いた状態で同じように学習反復を行い，その時のステップ数を評価する．そして，実験はランダムシードの異なる 30 試行を行い，その平均値で評価する．加えて，もし複数のエージェントが同じゴールへ到達してしまった場合，良い評価値となることを避けるためステップ数は次に示す最大のステップ

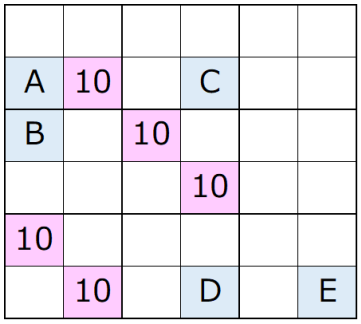


図 A.3　ケース 2 の迷路

表 A.1　各ケースにおけるそれぞれのエージェントのゴール選択可能範囲 gr

	A	B	C	D	E
ケース 1	1	3	3		
ケース 2	5	5	3	2	2

数である 100 とする．

実験パラメータは，学習回数が 50000 回，1 学習における最大のステップ数を 100 に設定する．また学習において初期 Q 値は 0，学習のパラメータをは学習率 α が 0.1，割引率 γ は 0.9 に設定する．内部報酬を設定するためのパラメータ δ は 10，関数 BID 内のランダム選択の確率は 0.15 に設定する．また，各ケースに対するエージェントのゴール到達可能範囲 gr を表 A.1 に従い設定する．

実験結果

各ケースにおける実験結果を図 A.4，A.5 に示す．縦軸はエージェントが全ゴールに到達するまでのステップ数，横軸は学習回数を示し，破線は任意数エージェント用の PMRL の結果，実線は PMRL の結果を表している．これらの結果から，任意数エージェント用の PMRL が PMRL よりもステップを費やさずに全エージェントを全ゴールへ到達させていることが分かる．特に，図 A.5 では PMRL のステップ数は 60 に達しており，協調行動そのものが学習できていないことに対し，任意数エージェント用の PMRL はステップ数が収束傾向にあり学習できていることが分かる．また図 A.4 では任意数エージェント用の PMRL は 4 ステップで全ゴールに到達できるようにエージェントを学習させている．これはケース 1 において全ゴールに到達するまでの最短ステップ数を示しており，任意数エージェント用の PMRL はエージェントに最適方策を学習させていることが分かる．図 A.5 から任意数エージェント用の PMRL は PMRL よりも短いステップ数でエージェントを全ゴールに到達できるように学習できていることが分かる．

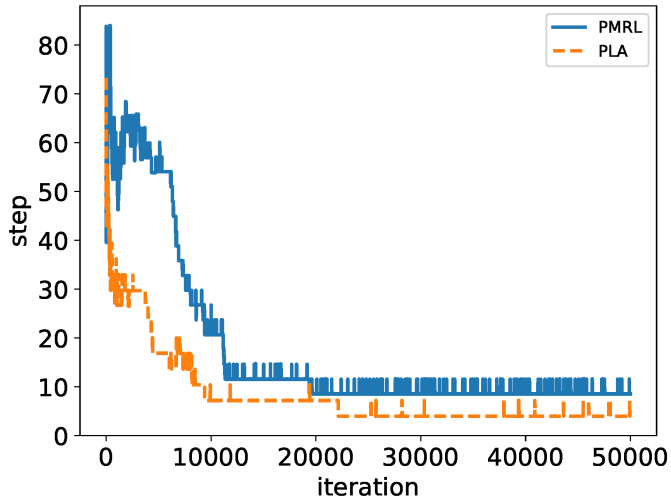


図 A.4　ケース 1 の迷路でエージェントが全ゴールに到達するまでのステップ数

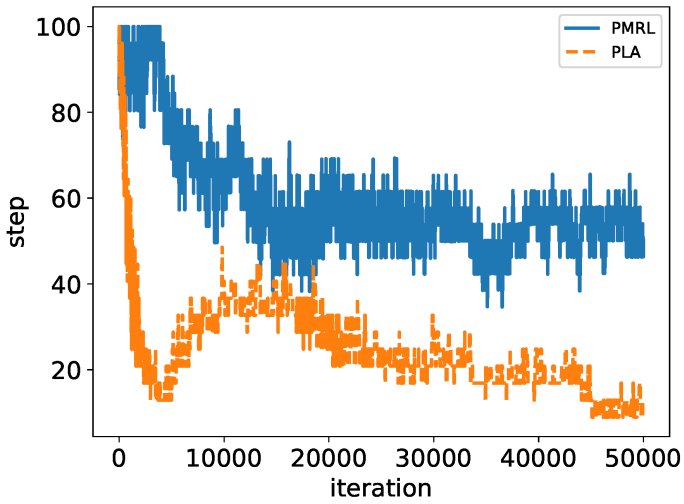


図 A.5　ケース 2 の迷路でエージェントが全ゴールに到達するまでのステップ数

考察

以上の結果により，すべてのケースに対して任意数エージェント用の PMRL が PMRL を上回っており，任意数エージェント用の PMRL の改良によりエージェント数が増えたときであっても適切に学習することができると分かる．これから各ケースを分析し，任意数エージェント用の PMRL の有効性を検証する．なおここでは，ゴールの識別番号を最も左のゴールから順番に数字を与える（もし同じ列のゴールがあれば上から与える）こととする．すなわち図 6.8 であれば上から 3 行目左から 2 列目のゴールが 1，上から 3 行目

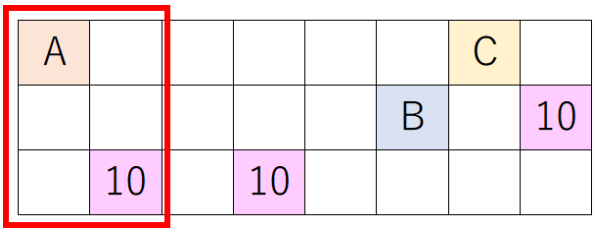


図 A.6　ケース 1 のエージェント A の制限

左から 4 列目のゴールが 2，上から 2 行目左から 8 列目のゴールが 3 となる．

- **ケース 1**

  ケース 1 において，エージェント A はゴール到達可能範囲が 1 であり，他のエージェント B，C は 3 である．図 6.8 から，エージェント A，B，C にとって最も遠くのゴールは 3，1，1 の識別番号のゴールである．それは図 A.6 のようになる．図の四角はエージェント A のゴール到達可能範囲を示したイメージである（エージェント B と C に関してはゴール到達可能範囲が 3 で制限されていないため図として表現していない）．この図のように，任意数エージェント用の PMRL はエージェント A の到達可能なゴールを制限し，識別番号 1 のゴール以外に到達しないようにして，全エージェントの最も遠くのゴールをそろえている．なおこのとき，任意数エージェント用の PMRL のメカニズムでは各エージェントのゴール到達可能範囲は 3，1，1 という可能性もある（つまり識別番号が 3 のゴールが最も遠くのゴールとなるようにエージェント B と C のゴール到達可能範囲を 1 にする）．しかしながら，それではエージェント B と C が 1 つのゴールに対して協調行動を学習することになってしまい，任意数エージェント用の PMRL は式 (5) を満たさなくなってしまうという理由ででその可能性を弾くことができている．

  その上で PMRL を実行すると，エージェント A，B，C は共に識別番号 1 のゴールに到達するように学習を行う．しかしこのとき，エージェント A が最短ステップでそのゴール到達できるため，目的価値の更新式における関数 $\phi^i(1)$ がエージェント A には 1 を示すが，エージェント B と C には 0 を示す．結果としてエージェント A のみが識別番号 1 のゴールに到達する行動を学習できる．その後，同様にしてエージェント B，C は次に遠いゴールである識別番号 2 のゴールへ到達するように学習し，エージェント B が近いので識別番号 2 のゴールに到達するように学習し，エージェント C は識別番号 3 のゴールへ到達する行動を学習する．以上がケース 1 の任意数エージェント用の PMRL による各エージェントの予測された振舞いである．仮にこれが正しければエージェントが全ゴールに到達するのに最短で 4 ステップかかる．図 A.4 を見ると確かに任意数エージェント用の PMRL は 4 ステップでエージェントを全ゴールに到達するように学習させていることが分かる．

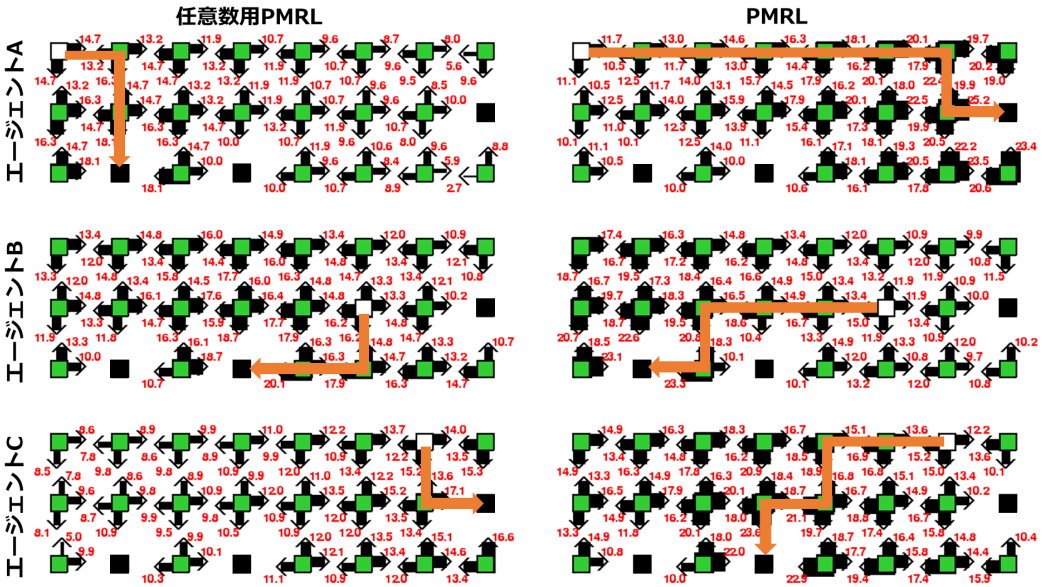


図 A.7　ケース 1 の Q テーブル

つまり，ケース 1 に関して任意数エージェント用の PMRL は最適方策を学習できていることが言える．

一方で PMRL は 8，9 ステップでエージェントは全ゴールに到達している．図 6.8 から，PMRL は全エージェントが自身の最も遠くのゴールへ到達するように学習し続けるため，エージェント A は識別番号 3 のゴール，エージェント B と C がそれぞれ識別番号 1 のゴールに到達するように学習すると推測される．実際このときかかるステップ数は最大で 9 ステップであり，図 A.4 の結果とも一致する．つまり，PMRL はケース 1 の迷路において不要にステップ数をかけて全ゴールに到達するような行動を学習していることが分かる．PMRL はそのように不要な学習を行っているため，収束も遅くなっている．これは図 A.4 を見ても明らかである．

図 A.7 はケース 1 における任意数エージェント用の PMRL と PMRL の全エージェントが持つすべての Q 値を示している．上段，中段，下段はそれぞれエージェント A，B，C を示し，左側と右側がそれぞれ任意数エージェント用の PMRL と PMRL を示している．図中の四角は迷路を示しており，白色，黒色の四角がそれぞれスタートとゴールである．迷路の各マスから伸びる黒い矢印は行動，その付近の数字は Q 値である．スタートからゴールへ伸びる矢印はそのエージェントがスタートから Q 値が最大の行動を選択した時に通る経路を示している．この図が示す通り，任意数エージェント用の PMRL では全エージェントが短い経路を通りすべてのゴールに到達することができているが，PMRL ではより長い経路で全ゴールに到達していることが分かる．

- ケース 2

  次にケース 2 の結果について考察する．ケース 2 においては，任意数エージェント用の PMRL，PMRL どちらの結果もステップ数が収束していない．ケース 2 において，エージェント A，B，C，D，E はゴール到達可能範囲がそれぞれ 5，5，3，2，2 であり，識別番号 5 のゴールを全エージェントの最も遠くのゴールと設定している．表 6.3 はケース 2 の迷路における各エージェントが各ゴールに到達するまでに必要な最短のステップ数である．また，表 A.3 は各エージェントから距離の遠いゴールを並べたものである．このとき，ケース 1 と同様に任意数エージェント用の PMRL においてゴール到達可能範囲の候補は 3 つある．1 つ目は上記のもの，2 つ目は識別番号 3 のゴールを最も遠くのゴールに設定した時のゴール到達可能範囲で，具体的にはエージェント A，B，C，D，E に対して 4，4，5，1，1 を設定する．3 つ目は識別番号 2 のゴールを最も遠くのゴールに設定した時で，エージェント A，B，C，D，E に対してゴール到達可能範囲を 1，2，1，5，5 に設定する．しかしながら，ケース 1 と同様 1 つのゴールに対して協調行動を学習する複数のエージェントが出てきてしまい，任意数エージェント用の PMRL は式 (5) を満たさなくなってしまうという理由でその可能性を弾くことができている．

  また実際の学習では，まず全エージェントが識別番号 5 のゴールへ到達するように学習する．このとき，エージェント C，D がともに最短ステップでそのゴールに到達可能であるため，ここはゴール到達可能範囲の小さいエージェント D がゴール W に到達するように学習し続ける．これを繰り返すと，最終的にエージェント A，B，C，D，E はそれぞれ識別番号が 1，2，4，5，3 のゴールへ到達するように学習し続ける．すると表 A.2 から，任意数エージェント用の PMRL ではこのとき最短でも 5 ステップかかることになる．図 A.5 を見ると任意数エージェント用の PMRL のステップ数は 10 付近の値を示しているため，最適な方策を学習できていない．しかしながらこれは 30 試行の平均であり，実際は 28 個の試行ではエージェントが最短ステップ数の 5 ステップで全ゴールに到達することができており，残り 2 試行も 100 を示しており学習途中に同じゴールへ到達してしまっているだけである．

  また，図 A.5 では任意数エージェント用の PMRL の曲線は 1 度下がってから上がっているが，これは最初はランダムに学習してそれが当たっただけの状態であり，その後色々なゴールへの方策を学習するにつれてステップ数が増え，最終的に最もステップ数の小さい方策を学習できているため，ここでは適切に学習できていることが分かる． つまり，ケース 2 に関しても任意数エージェント用の PMRL は最適方策を学習できていることが言える．一方で PMRL は 60 ステップ程でエージェントは全ゴールに到達している．これは 30 試行の大半が 100 を示しており，

表 A.2　ケース 2 の最短ステップ数

	ゴール 1	ゴール 2	ゴール 3	ゴール 4	ゴール 5
エージェント A	4	2	6	4	6
エージェント B	3	3	5	3	5
エージェント C	9	3	9	3	3
エージェント D	5	7	3	5	3
エージェント E	7	9	5	7	5

表 A.3　エージェントのスタート位置とゴール位置の距離関係

	1 (最近)	2 (近い)	3 (普通)	4 (遠い)	5 (最遠)
エージェント A	2	1	4	3	5
エージェント B	1	2	4	3	5
エージェント C	2	4	5	1	3
エージェント D	3	5	1	4	2
エージェント E	3	5	1	4	2

PMRL はそもそもすべてのゴールへ到達できるような学習をできていないことが分かる．

図 A.8 はケース 2 における任意数エージェント用の PMRL と PMRL の全エージェントが持つすべての Q 値を示している．上段と下段がそれぞれ任意数エージェント用の PMRL と PMRL を示し，左上からエージェント A，B，C，D，E の結果となっている．図中の白色，黒色の四角がそれぞれスタートとゴールである．迷路の各マスから伸びる黒い矢印は行動，その付近の数字は Q 値である．スタートからゴールへ伸びる矢印はそのエージェントがスタートから Q 値が最大の行動を選択した時に通る経路を示している．この図が示す通り，任意数エージェント用の PMRL では全エージェントが短い経路を通りすべてのゴールに到達することができているが，PMRL ではより長い経路で全ゴールに到達していることが分かる．

以上から，実験で取り上げた 2 ケースにおいて，任意数エージェント用の PMRL のゴール到達可能範囲の設定は適切に行われており，その設定に基づいた学習により，迷路において最適方策を獲得可能であることが明らかになった．

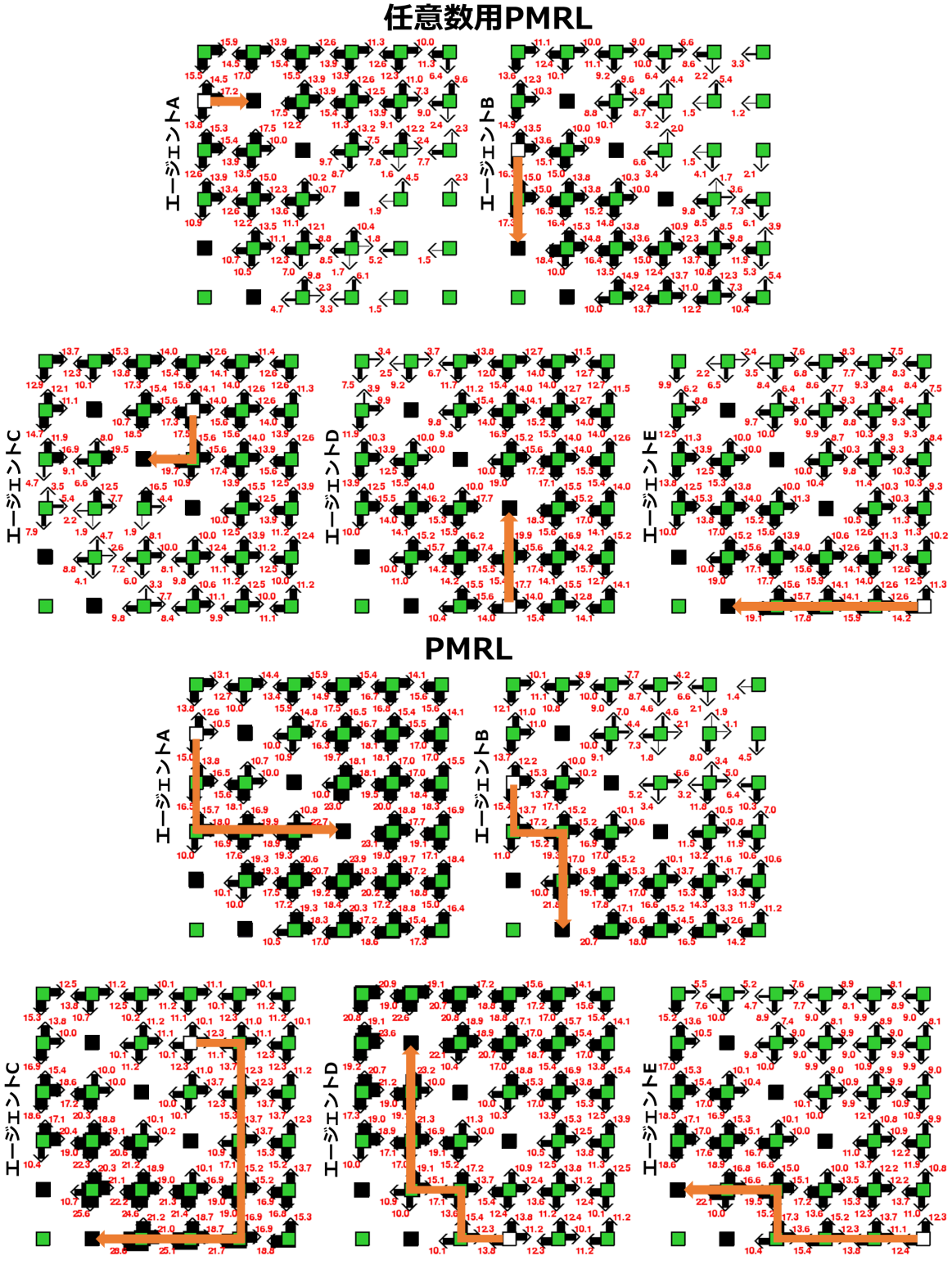


図 A.8　ケース 2 の Q テーブル