修 士 論 文 の 和 文 要 旨

研究科・専攻	大学院　情報理工学研究科　　情報・ネットワーク工学専攻　博士前期課程		
氏　　名	岡野　光稀	学籍番号	2031034
論 文 題 目	ゲーム AI のための動的な目標獲得システム		

要　　旨

本研究の目的は，コンピュータゲームプレイヤ（ゲーム AI）がゲームをプレイする際の目標設定を動的に行うシステムを提案し，その設計方法について議論することである．接待プレイなど，ゲーム本来の目標とは異なる多様な目標をもつゲーム AI の目標設定は，ゲーム本来の目標と比較して複雑になる場合がある．本研究ではゲーム AI の目標設定を動的に行えるようにすることで、多様な目標をもつゲーム AI の開発の効率化に貢献することを目指す。

本研究では目標の動的獲得システムの実現のため思考領域・本能・記憶の 3 つのサブシステム構造を提案する．各サブシステムの機能と相互の通信により目標設定の更新を行う．提案システムは人間のような柔軟かつ動的な目標設定の実現を目的としており，そこには設計における巨大さや複雑さといった課題がある．提案システムの 3 つの実装案を紹介し，提案システムの実装の実現性を検討した．

1 つ目の実装案は，記憶サブシステムのベースとなるネットワーク型記憶システムである．これは情報をノードとエッジで構成したネットワーク形式で表現し，情報の記憶・忘却・想起の 3 つの機能でこれを管理する．評価実験より，情報とその関係性を記憶しつつ，記憶データ全体をコンパクト化する効果があることを明らかにした．

2 つ目は，多様な目標の獲得の例としての最適でない方策獲得のための動的な報酬設定システムである．過去に学習した最適方策に制限を加えるように報酬関数を更新することで，最適でない方策を獲得できるようにした．評価実験より，提案システムの報酬設計の複雑さを緩和した上で，動的な目標設定を実現できた．

3 つ目は，自立的な報酬設定に必要なイベント認知に基づく目標獲得システムである．プレイヤが目標とすることのあるようなゲーム内情報をイベントと定義し，ゲームプレイ中に認識したイベントを用いて目標設定の動的な更新を行うシステムである．評価実験より，提案システムの目標設定における柔軟性を維持しつつ，提案システムよりも簡素な設計で動的な目標設定の更新を行えることを示した．

結論として，3 つの実装案はいずれも実用的な時間での，多様な目標の動的な設定に効果があることを示した．本研究の更なる発展によって，より提案システムに近い動的な目標獲得システムが設計可能となり，ゲーム AI の研究開発の効率化に貢献することが期待される．

電気通信大学情報理工学域
情報・ネットワーク工学専攻修士論文

ゲーム AI のための動的な目標獲得システム

2022 年 3 月 23 日

情報・ネットワーク工学専攻

学籍番号 2031034

岡野 光稀

主任指導教員 村松正和
指導教員 高橋里司

目 次

第1章　序論

1.1　研究の目的

本研究の目的は，コンピュータゲームプレイヤ（ゲーム AI）がゲームをプレイする際の目標設定を動的に行うシステムを提案し，その設計方法について議論することである．近年では接待プレイ [1] など，ゲーム本来の目標とは異なる目標をもつゲーム AI の開発が行われているが，そのような多様な目標をもつゲーム AI の目標設定は，ゲーム本来の目標と比較して複雑になる場合がある．たとえば囲碁において相手の力量に合わせた手を選択する接待プレイ AI[1] では，相手の力量の推定や着手の自然さなど考慮すべき要素が多く,「相手に接待する」という目標設定のためのモデルや評価関数の設計が AI の設計の大部分を占めている．また，接待という目標は曖昧な目標であり，その解釈によって様々な設計方法が考えられる．こういった曖昧な目標の設計および改善は，設計者の価値観や技術力に大きく依存することがある．本研究ではゲーム AI 自身が目標設定を動的に行うことで，目標設定の設計における設計者の負担を軽減し，ゲーム AI の研究開発の効率化に貢献することを目指す．これにより，目標の内容やゲーム AI の利用方法などの本質的な部分の設計に多くのリソースを割けるようになり，多様な目標をもつゲーム AI の研究開発がより活発に行われるようになると期待される．

本稿では第 2 章でゲーム AI のための動的な目標獲得システムの構成について説明し，提案システムの設計における課題と，その実装案である 3 つのシステムを紹介する．第 3 章から第 5 章では各実装案の具体的な設計方法，および評価実験とその結果を説明し，提案システムの実装案としての有効性を議論する．第 6 章では第 5 章までの内容のまとめと今後の展望を述べる．

1.2　背景

近年，コンピュータゲームプレイヤ（ゲーム AI）に関する研究はますます盛んになってきている．その研究・応用範囲は多岐に渡り，たとえば囲碁ゲーム AI の AlphaGo[2] やチェスゲーム AI の StockFish[3] などの強い AI や，感情に基づいた人間のようなプレイをする AI[4]，相手プレイヤーの実力に応じたプレイを行う接待 AI[1] など，多様な目的をもった AI(多様なゲーム AI) の研究が盛んに行われている．

多様なゲーム AI は人間にとっての楽しいゲームプレイの提供など，ゲーム AI の新たな利用方法をもたらす可能性をもつ．挑発行為や特定のプレイ方法への拘りなど，人間プレイヤのとるゲーム内目標に直接関係しない行動 [5] のいくつかは，ゲーム開発者の予期しないバグを発

見する手がかりとなることがある．このような人間プレイヤの行動を再現する多様なゲーム AI を作成できれば，ゲーム開発におけるデバッグ作業の効率化に貢献することが期待される．また，多様なゲーム AI を用いて人間プレイヤから見て自然かつ個性的な振る舞いをする NPC や bot を作成することも考えられる．

さらに，多様なゲーム AI の研究によって人間の知能や心のメカニズムへの理解がより深まることも期待される．Minsky の「心の社会」[6] では，人間の幼児が行う積み木遊びなどを例として，人間の心の働きを専門的な役割を持つ小さなエージェントの集まりとして表現することを試みている．人間が積み木を見て何かを思いつき，それで遊ぼうとするという目標の発生は単純に見えるが，実際には AI での再現が難しい課題の 1 つである．多様なゲーム AI によって人間が自主的に目標設定を行う仕組みを再現出来れば，このような課題も解決出来る可能性がある．

一方で，多様なゲーム AI はゲーム本来の目的に対する最適解とは異なった行動をとることから，勝利や高得点などのゲーム内報酬に基づく目標設定のみでは実現が難しく，設計が複雑になりやすいという問題がある．多様なゲーム AI の研究では，学習の目標設定は人間の経験則に基づくモデルや評価関数の設計によって実現されることが多い．池田らの接待プレイ AI は，囲碁において対戦相手との接戦を演出するために，相手の力量や着手の自然さを評価する関数を用いて次の行動を選択する [1]．Sila らは AI がマリオブラザーズを人間のようにプレイするための,「安心」や「焦り」などの感情に対応したプレイスタイルでゲームプレイを行うモデルを提案している [4]．このような目標設定は既存の研究の手法をそのまま利用することが難しく，しばしば研究者自らの手による設計や調整が必要となる．多様なゲーム AI の目標設定を動的に生成，調整するシステムを作成し，設計に掛かるコストを緩和出来れば，多様なゲーム AI の研究をより効率的に行えるようになると考えられる．

1.2.1 ゲーム AI の目標設定の関連研究

ゲーム環境において特定のタスクをこなすための方策の学習におけるゲーム AI の目標設定に関して，これまで様々な取り組みが行われてきた．たとえば迷路ゲーム環境の場合，強化学習 [7] によってスタート地点からゴール地点まで移動する方策を学習するために，ゴール時の正の報酬と時間経過に応じた負の報酬の 2 つの報酬設定で目標を設定するという古典的な方法が知られている．しかしこの方法では，迷路のスタートからゴールまでが非常に遠い場合，ゴール時の報酬がほとんど得られず，ランダムな行動では学習時の探索が進まないという問題が発生する．このような報酬が疎な環境における探索について，Pathak らは好奇心という内部報酬をゲーム AI エージェント（エージェント）が自らに与えることで探索を進めていく手法を提案した [8]．好奇心とはエージェントが予測と異なる情報を観測する，すなわち未知の環境を探索した際に得られる内部報酬であり，この好奇心によって環境からの外部報酬が得られない場合でも未探索の場所を積極的に探索し，ゴールへと向かう方策を学習することが可能となる．た

だし，初期の好奇心を用いた探索手法では初めて見る情報を重視するあまり，ボタンを押すと切り替わるテレビ画像のようなギミックに釘付けになって探索が進まなくなるという問題（カウチポテト問題）が存在した．この問題を解消するために，Savinov らはただ新しいだけではなく，そこに到達するのがむずかしい，到達に要するステップ数が大きい情報にのみ好奇心報酬を与える手法を提案した [9].

好奇心などの内部報酬は，強化学習における探索と搾取のトレードオフのバランスをとるためにも有効である．探索と搾取のトレードオフとは，エージェントはより良い報酬を見つけるためには探索を進める必要があるが，より多くの報酬を得るためには現在の知識に基づく最適方策の学習を進める必要があるという問題である．この探索と搾取のトレードオフに関わる有名な理論として，自由エネルギー原理が存在する [10]．自由エネルギー原理では生物の感覚入力の将来に渡る予測困難さ・不確実性をサプライズと定義し，全ての生物はこのサプライズを最小化する方向に行動や予測の修正を行うものとする [10]．自由エネルギー原理は強化学習などの機械学習のみならず，人間の脳の統一理論として多くの分野への応用が期待されている．

強化学習では一般に，毎時刻の行動は環境の観測情報と方策から決定される．これに対し，エージェントの連続した行動の系列を「スキル」と定義し，スキルを用いてエージェントの探索と学習を進めていくというアプローチが存在する．この方法のメリットとして，アクションゲームなどの単一の行動が与える変化が小さいゲーム環境や，格闘ゲームなどの行動空間が広いゲーム環境において，探索空間を縮減し学習の収束を早める効果が挙げられる．梅野らは離散空間における教師なしスキルの獲得方法として，行動のエントロピーが大きくなるようなスキルの組み合わせを自動で獲得し，そのスキルを用いて学習することによって探索空間の縮減および学習の効率化を行う手法を提案した [11]．増山らはエージェントが過去の成功体験から得られた行動の系列をスキルとして記憶し，さらにアフィン変換不変量を元にそれらにスキル報酬という内部報酬を与えることにより，さまざまな環境で再現可能なスキルの強化とそれを用いた探索空間の縮減を可能とする手法を提案した [12]．Oh らはリアルタイム格闘ゲームのためのエージェントの学習手法として，受動的な「何もしない」状態のデータのスキップと，ひとつの行動を複数時間連続して入力する手法を用いた自己学習を行うことで，プロゲーマーを打ち破るほどの実力をもつエージェントを開発することに成功した [13].

ゲーム AI エージェントの他の学習方法として，人間プレイヤのエキスパートのプレイデータを教師データとして学習する模倣学習が存在する．Jack らは複雑な 3D ゲーム環境における深層強化学習において，模倣学習によってエキスパートのレベルにまで迅速に学習し，そこから時間差強化学習 (TD 学習) によってエキスパートを超える能力を獲得させることで，単一行動選択の TD 学習と比較して学習時間を 4 倍，性能を 2.5 倍向上させることに成功した [14].

Benjamin らは事例の再帰的分類 (RCE:Recursive Classification of Examples) という手法で，報酬関数が不要な強化学習を実現した [15]．RCE ではタスクの成功事例（たとえば,「壁に釘を打つ」というタスクであれば壁に釘が打たれた状態）からエージェントが直接学習し，複雑な報酬の設定を省略することが可能である．RCE は模倣学習とは異なり，用意する事例はエキスパートのものである必要はなく，ただタスクが完了していればよいという特徴がある [16].

第2章　ゲームAIのための動的な目標獲得システム

2.1　ゲームプレイにおける人間プレイヤの目標設定

AIによる自動目標獲得の設計に先立ち，人間の目標獲得について考える．多くのゲームAIは設計者の手によって設定された目標に基づき，ゲームプレイを行う．人間プレイヤーの場合は誰かに指示されずとも自分の意志で目標を設定し，また柔軟に変更することが可能である．本章ではAIがこのような自動的かつ柔軟な目標の設定を行う方法として，思考領域，本能，記憶の3つのサブシステムの組み合わせによって目標設定を獲得・更新するシステム (以下，本システム) を提案する．

このような自律的なシステムについては様々な取り組みがなされている．Marvin Minskyの「心の社会」ではコネクショニズムにもとづく構成が検討されている [6]．VanRullenらは各分野に特化したディープラーニングをモジュールとし，それぞれのモジュールの入出力を翻訳しつつ配信するためのグローバル潜在作業空間 (GLW) を生成することで，より広範なタスクへの対応を可能とするGlobal Workspace Theory(GWT) の明示的な実装に向けたロードマップを提案した [17].

2.2　目標とは何か

2.2.1　目標とは

ゲームプレイに限らず，人は何らかの行動をする際に，その行動が目指す目的地となる目標をもつことが多い．そして行動によってその目標に到達できたとき，その目標は達成したとみなされる．これらの目標や目標達成とはどのようなものであるか，本研究では次のように考える．人が目標をもって行動する際，その人の脳内には今現在の自分が観測する世界の状態である「現状」と，到達すべき理想の世界の状態の想像である「理想」の2つの状態が存在する．人は行動によって現状を変化させ理想へと近づけようとするが，理想はあくまで想像の状態であるため，現状と理想が100%一致することはない．そこで，人は現状が理想に十分近づいた時点で目標を達成したとみなし，行動を終了する．この目標達成の判断基準となる，現状が理想に十分近づいたとみなされる状態の集合を理想の「近傍」と定義する（図2.1）．目標とは理想とその近傍であり，目標達成とは現状を理想の近傍まで近づける行為である．たとえば「コーラ

が飲みたい」という目標をもって行動している人は,「喉が渇いている」という現状を「コーラを飲んでいる」という理想に近づけようとしていると考えられる．しかしその人が自動販売機にたどり着くとコーラが売り切れており，代わりにサイダーを購入したならば，その人にとって「サイダーを飲んでいる」という現状は「コーラを飲んでいる」という理想の近傍に属しており，これで目標を達成したと考えられる．また，仮にその人がコーラを購入できた場合でも，それを口にしたときの味わいや温度などの情報が理想と完全に一致することはないため，この場合も「コーラを飲んでいる」という理想の近傍に現状を近づけたことで目標を達成したと考えられる．

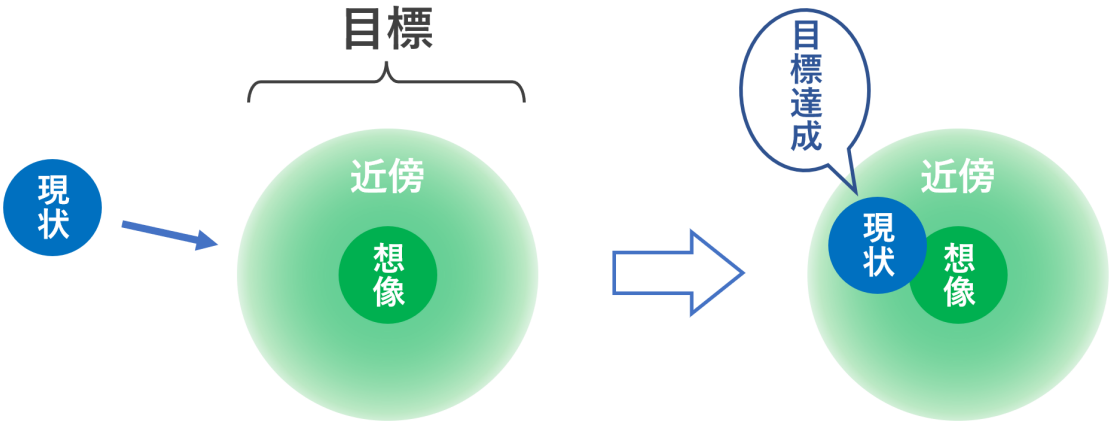


図 2.1: 目標と目標達成のイメージ

2.2.2　3 つの機能による動的な目標設定

ゲーム AI が人間のように自ら目標を設定し，それを達成したかどうか判断するためにはどうすればよいか．本研究では「目標設定に必要な情報の収集と処理」と「目標を設定する動機」と「目標設定に関する過去の経験の蓄積と参照」の 3 つの要素が目標設定に必要であると考える．これら 3 つの要素をもとに，ゲーム AI の目標設定のための「思考領域」「本能」「記憶」の 3 つの機能を定義する．まず，ゲーム AI が目標を設定するためには，観測した世界の情報から現状を認識し，現状や過去の経験から理想を生成する必要がある．このように「思考領域」とは観測した情報を収集・処理し目標設定を行う機能である．次に，ゲーム AI が目標設定を行うにはそのきっかけ・動機も必要である．たとえば人間の「パンを食べる」という目標には「空腹感」という動機が存在し，この空腹感を解消するために「パンを食べる」という目標が設定される．このように「本能」とは目標設定を行うきっかけを思考領域に与える機能である．最後に，現状に対してより良い目標を設定するためには，過去の経験を蓄積し，それを用いて目標設定の方法を改善していく必要がある．このように「記憶」とは目標設定に関係する過去の経験を記憶・管理し，必要に応じて取り出す機能である．以上の 3 つの機能があれば，ゲーム

AI は自ら目標を設定し，その達成を判断することが可能である．2.3 節ではこれらの 3 つの機能を元にした，ゲーム AI が動的に目標設定を行うためのシステムを提案する．

2.3　3 つのサブシステムから構成される動的目標獲得システム

本章で提案する動的目標獲得システムは，思考領域サブシステム，本能サブシステム，記憶サブシステムの 3 つのサブシステムの組み合わせによって，ゲームプレイにおける目標設定を獲得・更新する．

2.3.1　思考領域サブシステム

思考領域とは，システムが集約した情報を保持および処理するときの空間をもつサブシステムである．思考のためのシステム領域という意味で，この空間を思考領域と呼ぶこととする．

ある情報は，個の思考用域の一点として表され，位置座標をもつ．複数の情報の位置座標の関係によって知覚や情報統合を行う．情報に対応する位置座標の決定はその情報の様々な性質を用いて行う．

2.3.2　本能サブシステム

本能サブシステムとは，AI の目標設定のきっかけを生み出すサブシステムである．思考領域を監視し，その状態に応じた定型的な変化をもたらすことで，その変化に対応するための目標設定を行うように思考領域に促す．

2.3.3　記憶サブシステム

記憶サブシステムとは，AI の過去の経験を蓄積するサブシステムである．思考領域の情報の保存・管理・取り出しを行う．

2.3.4　目標設定の動的更新の流れ

以上の 3 つのサブシステムからなる本システムは，以下の 4 つの処理を並行して行い，図 2.2 のように目標設定の更新を行う．

1. 環境 E の観測情報 I_{E} により思考領域 T の情報 I_{T} を更新する
2. I_{T} に応じて, 本能 I が動機 M_{I} で I_{T} を更新する
3. 思考領域 T が記憶 M へ時刻 t の情報 $I_t \subset I_{\mathrm{T}}$ を送り，関連する情報 I_r を受け取る（記憶による補正）
4. 思考領域 T が現在の目標設定 $G_{\mathrm{T}} \subset I_{\mathrm{T}}$ を更新する

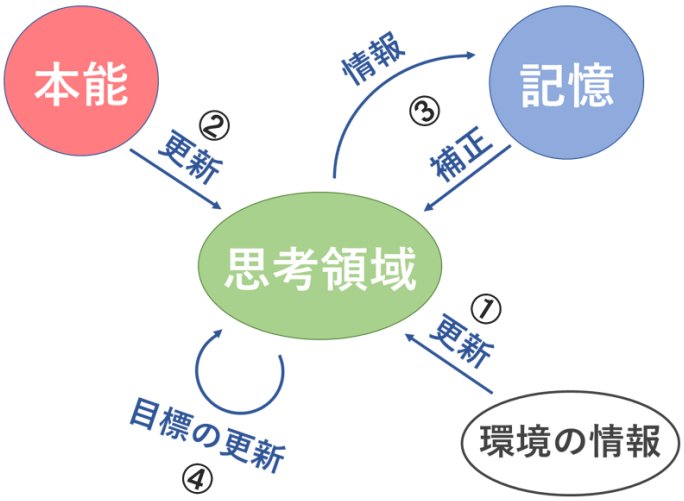


図 2.2: サブシステムの処理と目標設定の更新

2. の本能による変化では，思考領域がその変化に対応するという目標設定のきっかけが生じる．本能は思考領域の状態を監視することで間接的に環境の情報を取得することが出来る．例えば，本能がエネルギーの減少を感知し,「空腹感」を感じるように思考領域を変化させたならば，思考領域は空腹感を解消するための目標設定を行うことになる．3. の記憶による補正では，過去の経験を元に思考領域がどのような目標を設定するかを決定する．本能によって本システムの方向性をある程度決めることで，目標設定は過去どのように目標を設定したか，それらはどの程度上手くいったか，といった記憶の蓄積に基づいて本システム自身が決定することになる．

たとえば，本能が思考領域の状態を「空腹」に変える場合を考えると，初期状態では記憶に何の情報もないため，本システムは空腹を解消するための目標を設定できない．しかし一度でも何かを口にした経験があれば，何かを食べたいという「食欲」を目標に設定する可能性がある．あるいは痛みで空腹が紛らわされたという経験があれば，自分を殴りつけるという目標を設定する可能性もある．このように本システムは記憶サブシステムに経験を蓄積することで，本能によるきっかけに対応する多様な目標設定を自動で獲得できると期待される．

2.4　動的目標獲得システムの実装

2.4.1　設計における課題

2.3 節で提案した動的目標獲得システムを設計する上でいくつかの課題が存在する．特にシステムの巨大化や，それに付随する課題として各サブシステムが扱うデータの形式や，初期状態における本能サブステムの機能はどのようなものが必要であるか等の課題が重要である．提案システムは人間プレイヤの高度な認知機能と膨大な知識に基づいた，柔軟な目標設定を再現することを目標としているため，各サブシステムやシステム全体が必要とする処理が複雑かつ巨大になると予想される．そこで本稿ではサブシステムの部分的な実装や，各サブシステムの機能を制限したシステムの実装を通じて，提案システムの設計における課題を解決または緩和する方法について議論する．

2.4.2　実装案 1：ネットワーク型記憶システム

ネットワーク型記憶システムとは，記憶サブシステムのベースとなる，情報と情報同士の関係性の記憶・忘却・想起のためのシステムである．提案システムでは思考領域サブシステムが現状に適した目標設定を行うために，記憶サブシステムに蓄積された情報同士の関係性を上手く扱う必要がある．本実装案では情報同士の関係性をノードとエッジで構成されるネットワークの形式で表現し，それらを保存・管理・取り出す機能をもつシステムを提案する．

2.4.3　実装案 2：最適でない方策の学習のための動的な報酬設定システム

最適でない方策の学習のための動的な報酬設定システムとは，ゲーム AI が過去の最適方策と異なる方策を獲得するように報酬設定を更新することで，最適でない方策を学習するための目標設定を動的に更新するシステムである．提案システムの目標のひとつとして，多様なゲーム AI の複雑な目標設定を効率的に行えるようにすることがある．本実装では，近いゴールにまっすぐ向かわずに遠回りをしたりより遠いゴールに向かうといった，複雑になりやすい方策の報酬設定を，最適方策の制限によって動的に設定するシステムを提案する．

2.4.4　実装案 3：イベント認知に基づく目標獲得システム

イベント認知に基づく目標獲得システムとは，プレイヤが意味付け可能なゲーム内情報をイベントと定義し，ゲーム AI が経験したイベントに基づく目標設定の獲得と更新を動的に行うシステムである．提案システムでは思考領域サブシステムが目標設定を行うために，目標とは

何か，どのような場合に目標を達成したといえるのかを判断する必要がある．本実装案ではこのような目標設定の対象をイベントと定義し，ゲームプレイ中に経験したイベントを目標の候補として獲得・設定することで，ゲーム AI の動的な目標設定を実現するシステムを提案する．

第 3 章から第 5 章では各実装案における，システムの概要と設計方法，および評価実験の結果を説明し，提案システムの実装案としての有効性を議論する．

第3章　ネットワーク型記憶システム

3.1　システムの概要

本章では，第2章で提案した動的目標獲得システムの3つのサブシステムのうち，記憶サブシステムのベースとなるネットワーク型記憶システムを提案する．まず記憶サブシステムとして必要となる情報処理の3つの機能（記憶・忘却・想起）を説明し，それらの実装方法およびシステムの扱うデータ形式を説明する．次に実装したシステムを用いた評価実験の結果を示し，記憶サブシステムのベースとしてのネットワーク型記憶システムの有効性について考察する．

3.2　記憶サブシステムに必要な機能

動的目標獲得システムにおける記憶サブシステムの役割は，思考領域の情報を経験として保存・管理し，必要に応じてそれを取り出すことである．したがって，記憶サブシステムは以下の3つの機能をもつ必要がある．

1. 情報を保存する機能（記憶）
2. 情報を削除する機能（忘却）
3. 情報を取り出す機能（想起）

これら3つはいずれも記憶サブシステムの利用・維持に必要なものである．記憶と想起の機能は記憶サブシステムの役割を果たすために必要であり，忘却機能は不要な記憶があふれることによって他の機能の性能悪化を防ぐのに必要である．忘却において，記憶サブシステム自身は推論や価値評価を行わないため，どの記憶が重要かまたは不要かという判断を機能に組み込むことができない．以上を踏まえ，本実装案では記憶サブシステムのベースとなるネットワーク型記憶システムを以下のように設計することを考える．

3.3　ネットワーク型記憶システムの設計

本節ではネットワーク型記憶システム（以下，本システム）における 3 つの機能およびデータ形式を説明し，これらの設計方法を提案する．

3.3.1　ネットワーク型のデータ形式

本システムにおける記憶データを，図 3.1 のようなノードとエッジによって表す．このデータ形式はグラフデータベース [18] の手法を参考にしている．グラフデータベースはテーブル形式のリレーショナルデータベースと比較して,「ある人の友人の友人の総数」のようなデータ同士の関係性に基づく検索が非常に高速に行えるという特徴がある．本システムは記憶データの関係性を記憶し，想起する機能をもつため，グラフデータベースのようなネットワーク型のデータ形式が適当であると考え，これを採用した．

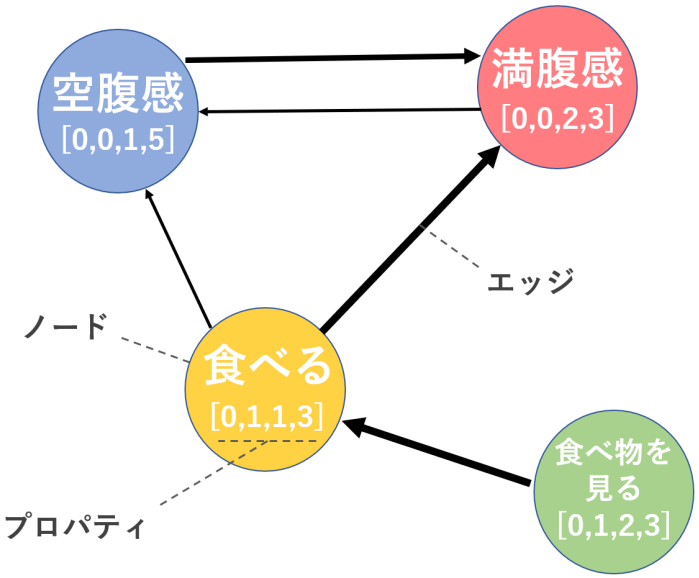


図 3.1: 記憶のネットワークのイメージ

知識をネットワークの形で表現する手法については様々な研究がなされている．砂山は類推による知識獲得と創発に関するアイデアとして，新しい知識をローカルなエッジで接続するだけでなく，少し離れた情報をグローバルなエッジで接続し，関連する複数の知識の検索や共通概念の取り出しを行う手法を提案した [19]．越中らは大学の授業評価アンケートの自由記述のテキストデータについて，KH Coder を用いて頻出する語やその共起関係を共起ネットワークとして可視化し，全体的な傾向の分析を試みている [20]．

本システムのネットワークではノード n が記憶 m の内容を表し，エッジ e は記憶 m 同士の接続とその重みによって m の関係性 (繋がり) を表す．m の内容は情報 i の思考領域における座

標 c_i と値 v_i を整数値のベクトルで表したものであり，これをプロパティ p_n と呼ぶ．たとえば c_i が 3 次元のベクトル $c_i' = (c1, c2, c3)^\mathsf{T}$，$v_i$ が 1 次元のベクトル $v_i' = (vi)^\mathsf{T}$ ならば，p_n は図 3.2 のような 4 次元のベクトル $p_n' = (c1, c2, c3, v1)^\mathsf{T}$ となる．プロパティが等しいノードは同じ記憶として扱う．エッジ e は重み w_e と接続先のノードのアドレス a_n をもち，w_e の大きさで n 同士の関係性の強弱を表現する．本システムはこれらのノードとエッジを用いて，記憶をネットワーク型のデータ形式で保存・管理する．

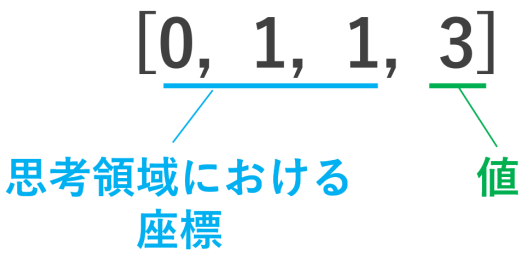


図 3.2: 4 次元ベクトルのプロパティ

3.3.2 記憶機能

本システムにおける記憶機能は受け取った情報 i を記憶 m としてノード n に変換し，記憶のネットワークに追加する処理とする．m は必ず関連性のある 2 つのペア $(m1, m2)$ で受け取り，図 3.3 のようにそれらのノードをエッジで接続することでネットワークに追加するものとする．このとき，ネットワークに m とプロパティが同じ記憶 m' が存在している場合は m を m' に置き換えて重複を回避する．プロパティは記憶ノードのもつ情報を表す値であり，同じ情報であれば同じプロパティとなる．ネットワーク内で $(m1, m2)$ とプロパティが同じ記憶のペア $(m1', m2')$ がエッジ e' で接続されている場合は，接続の代わりに e' の重み e_w' を増やす．ひとつのノード n が接続に使えるエッジの本数や重みの総和には上限があり，上限まで使用している状態でエッジの接続や重みの増加を行う場合は，次の 3.3.3 節で述べるエッジの本数や重みの奪い取りを実行する．

3.3.3 忘却機能

記憶のネットワークはノードとエッジからなる．このため本システムにおける忘却機能には，ノードの忘却とエッジの忘却の 2 種類がある．

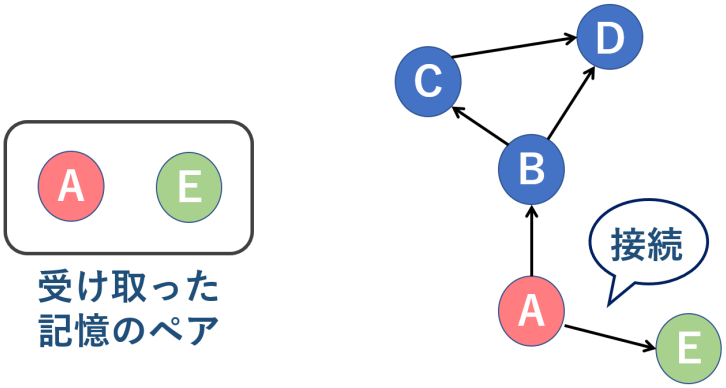


図 3.3: 記憶のネットワークへの追加のイメージ

ノードの忘却

ノードの忘却は，図 3.4 のようにノード n に忘却までの残りカウント数 c_n を紐づけ，カウントが 0 になったものをネットワークから削除する処理である．c_n は記憶システムのクロックに応じて減算し，想起される毎に増加させる．これにより，頻繁に想起される記憶と最近想起された記憶がネットワークに残り，古く想起されない記憶はネットワークから削除される．

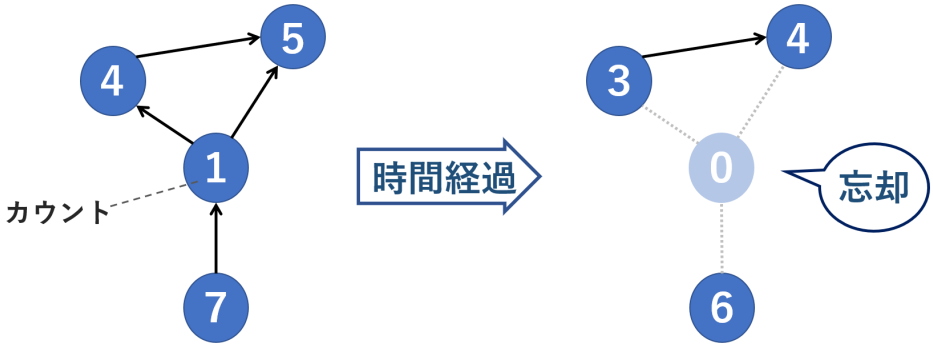


図 3.4: ノードの忘却のイメージ

エッジの忘却

エッジの忘却は，ひとつのノード n で使用できるエッジ e の本数（スペース）s_n と重みの総和（リソース）r_n に上限を設け，エッジ同士で s_n と r_n を奪い合う処理である．ノードの忘却では記憶そのものの忘却を行ったのに対し，エッジの忘却では情報同士の関係性の忘却を行う．これにより，あまり想起されない情報同士の関係性はネットワークから削除され，よく想起に用いられるエッジによって記憶のネットワークが構成されるようにする．エッジ e の重み w_e の

増加 (エッジの強化) に必要なリソース s_n が足りない場合は，図 3.5 のように他のエッジ e' の重み $w_{e'}$ を奪い取る処理を行う．奪い取りによって重みが 0 になったエッジはノードから削除される．エッジの接続に必要なスペース s_n が足りない場合は，他のエッジのスペース（全ての重み）s_e を奪い取って削除する処理を行う．これらの w_e と s_e の奪い取りの成功率は，奪い取る側の w_e を $a\,(\geq 0)$，奪い取られる側の w_e を $b\,(> 0)$ として式 (3.1) と (3.2) で計算する．ただし，R は w_e を 1 奪い取る確率，S は s_e を奪い取る確率であり，$b_{win}\,(\leq 1)$ は基本勝率である．

$$R(a,b) = \min\left(b_{win}, \frac{b}{a+b}\right) \tag{3.1}$$

$$S(a,b) = \prod_{k=0}^{a-1} R(a+k, b-k) \tag{3.2}$$

$R(a,b)$ は a が b に対して小さいほど高くなり ($a=0$ なら必ず成功)，a が b より大きい場合でも b_{win} 以上の確率で成功することを保証する．このようにすることで，想起に使われないエッジは時間が経つほど他のエッジから重みを奪われ，削除される可能性が高くなる．

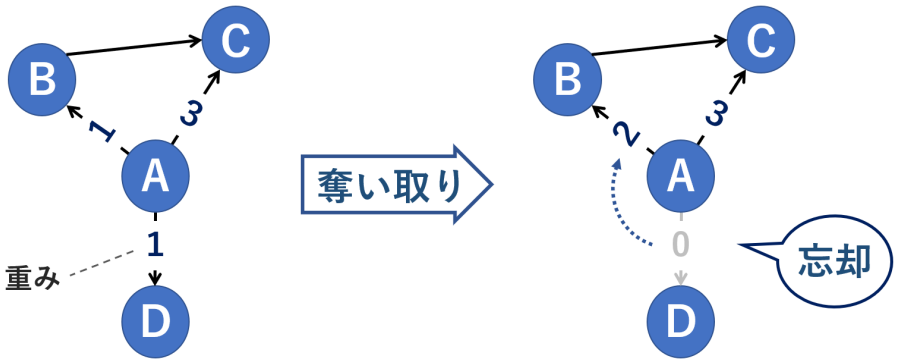


図 3.5: エッジの忘却のイメージ

3.3.4 想起機能

本システムにおける想起機能には，直接想起と連想想起の 2 種類がある．直接想起は新しい記憶をネットワークに追加する際に，その記憶が既にネットワークに存在するか確認して重複を避けるための機能である．連想想起は思考領域サブシステムの要求に応じて，ある記憶と関連する記憶を探すための機能である．

直接想起

直接想起は，図 3.6 のようにあるノード n と同じプロパティを持つ n' がネットワークに存在するか確認する検索である．この直接想起は 3.3.2 節の記憶機能を利用する際に，新しい記憶が既にネットワークに存在するか確かめるために実行される．検索するノード n' が他のノードに接続していない (孤立している) 場合でも想起を行うために，n' のアドレス $a_{n'}$ を保存するハッシュテーブル H を用いた検索を行う．ここで，n のハッシュ値はプロパティ p_n のベクトルから算出する．プロパティが 4 次元のベクトル $p'_n = (c1, c2, c3, v1)^\mathsf{T}$ である場合，ハッシュ関数は $h(c1, c2, c3, v1)$ として定義できる．直接想起の結果，n とプロパティが等しいノード n' が見つかった場合は n' が想起されたとみなし，カウント数 $c_{n'}$ を増やすことで忘却までの時間を延長する．

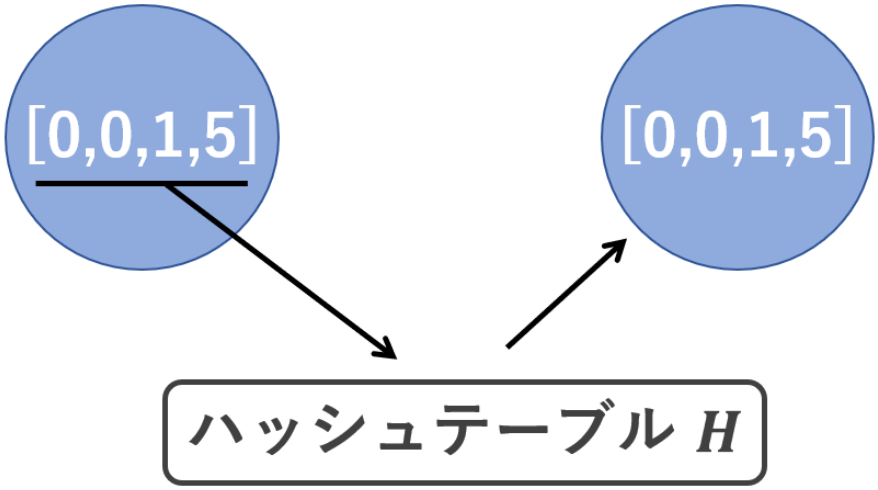


図 3.6: 直接想起のイメージ

連想想起

連想想起とは，図 3.7 のように想起の開始地点となる記憶のノードから，ノードのエッジを順に辿っていく検索である．エッジ e の選択方法は, 重み w_e に比例したルーレット選択か，最も w_e の大きい e を選択する重み優先選択の 2 つである．どちらの方法を用いるかは，記憶サブシステムに想起を要求する思考領域サブシステムが決定する．また，想起中に通過したノード及びエッジは全て想起した記憶として扱い，カウント数の増加やエッジの強化を行うことで，忘却までの時間を延長する．

各想起におけるノード n のカウント数 c_n の増加は式 (3.3) で計算する．ここで，$c'\,(>0)$ はカウントの初期値，$a\,(>0)$ はカウント数の増加率である．

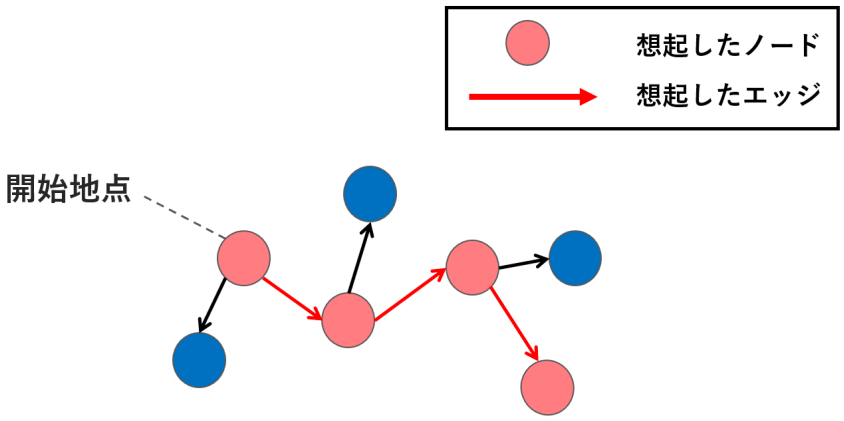


図 3.7: 連想想起のイメージ

$$c_n = \begin{cases} c_n + c' & (a\,c_n < c') \\ a\,c_n & (a\,c_n \geq c') \end{cases} \tag{3.3}$$

連想想起におけるエッジ e の強化は，重み w_e に 1 を加え，ノード n のリソース r_n を 1 減らす処理とする．r_n が 0 の場合は 3.3.3 節で述べた重みの奪い取りを実行する．

3.3.5　その他の機能

思考領域サブステムが記憶サブシステムに情報の記憶や想起を要求する際に必要な機能を説明する．

過去に追加・想起した記憶の一時保存

記憶を追加する際に必ず 2 つ以上のペアで追加するためには，思考領域が複数の記憶 (情報) を扱う必要がある．これを実現するため，前回追加された記憶や想起された記憶 m を一時的に保存しておき，次の記憶 m' が渡されたときに m とのペア (m, m') とみなしてネットワークに追加する処理を行う．

3.4　記憶サブシステムの機能の評価

本節では本章で提案するネットワーク型記憶システムシステムついて，情報とその関係性の記憶・忘却・想起を行うことが出来るか確認した評価実験とその結果を説明する．

3.4.1　目的と概要

本実験の目的は，本システムが逐次的に渡される情報に対して，どのような情報が多く現れるか，ある情報のあとに現れやすい情報は何かといった，情報とその関係性の記憶や想起を行うことが出来るか確認することである．

本実験では図 3.8 のようなある情報の次にある情報が出現する確率を表した行列 (以下，遷移行列) を用意し，遷移行列の確率に従って現れた情報を逐次的に記憶システムに記憶させた後に，特に確率の高い情報同士の関係性を想起出来るかどうか確認した．ただし，遷移行列では i 番目の情報から $i+1$ 番目の情報が出現する確率が他より高くなるように設定し，記憶システムが適切に情報を記憶し想起したならば $1 \to 2 \to 3 \to \ldots$ というような情報の列が得られるようにした．

また，忘却機能の効果を確認するため，3.3.3 節で述べた機能のうち,「ノードとエッジの忘却機能あり」「エッジの忘却機能のみあり」「忘却機能なし」の 3 パターンの記憶サブシステムを用意して実験を行った．

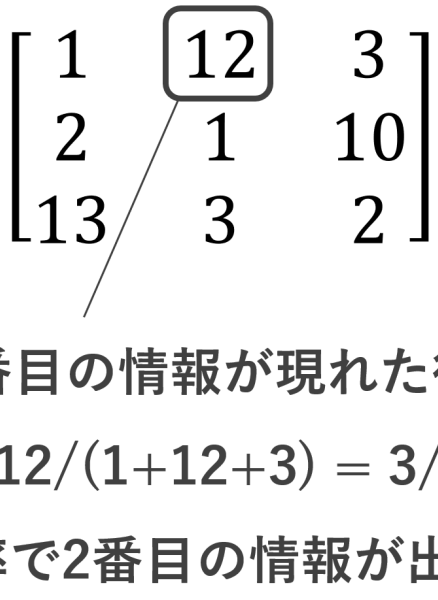


図 3.8: 3 つの情報に関する遷移行列の例

3.4.2 準備

パラメータ

各パラメータは表 3.1 のように設定した．ただし，エッジの忘却なしの条件ではエッジの最大本数およびリソース (エッジの重みの総和) は無制限とした．

表 3.1: パラメータ

エッジの最大本数	5
リソース	25
基本勝率 b_{win}	0.05
カウントの初期値 c'	5
カウントの増加率 a	1.1
カウントの最大値	10,500

3.4.3　手順

手順 1，2 を順に計 1000 回行い，想起列の長さおよび記憶ネットワークのノード数とエッジ数，ノードの平均カウント数の平均値を記録した．

手順 1：情報の記憶

プロパティを適当に設定した n 個の情報とその遷移行列 $(n \times n)$ を用意し，記憶サブシステムに情報を追加する $\rightarrow$ クロックを 1 進める $\rightarrow$ 遷移行列を元に次の情報をルーレット選択するという操作を 10000 回行った．最初に追加される情報は 1 番目の情報とした．

手順 2：想起

記憶サブシステムに対して 1 番目の情報からエッジの重み優先想起を行わせ，$1 \rightarrow 2 \rightarrow 3 \rightarrow \ldots \rightarrow n$ という情報の列 (以下，想起列) がどの程度の長さまで得られるか確認した．この想起列はたとえば，$1 \rightarrow 2 \rightarrow 3$ であれば長さ 3 とし，情報 1 が記憶サブシステムに存在しない場合は長さ 0 とした．

実験は $n = 10, 100$ の 2 パターンの情報および遷移行列に対し,「ノードとエッジの忘却機能あり」「エッジの忘却機能のみあり」「忘却機能なし」の 3 パターンの記憶サブシステムを用いて行った．ただし，$n = 100$ の場合の「ノードとエッジの忘却機能あり」による実験のみ，通常の実験に加えて手順 1 を手順 1 *に置き換えた追加実験を行った．

手順 1*：想起列の教え込みと情報の記憶

最初に $1 \rightarrow 2 \rightarrow 3 \rightarrow \ldots \rightarrow 10 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow \ldots$ という順番での記憶の追加を 1000 回行った後，手順 1 と同様の操作を 9000 回行った．

3.4.4　結果

$n = 10, 100$ における想起列の長さ，ノード数，エッジ数，ノードの平均カウント数の結果を表 3.2，3.3 に示す．ただし「あり」は「ノードとエッジの忘却機能あり」を,「エッジ」は「エッジの忘却機能のみあり」を,「なし」は「忘却機能なし」を表し,「あり*」は手順 1*による追加実験を表す．

表 3.2: $n = 10$ の実験結果 (1000 回の平均)

忘却	想起列	ノード数	エッジ数	平均カウント
あり	10.0	10.0	46.9	10500
エッジ	10.0	10.0	46.0	–
なし	10.0	10.0	90.0	–

表 3.3: $n = 100$ の実験結果 (1000 回の平均)

忘却	想起列	ノード数	エッジ数	平均カウント
あり	0.33	15.0	14.0	7.97
あり*	10.0	24.1	35.9	4240
エッジ	100	100	418	–
なし	100	100	3730	–

3.5 情報の想起の精度と忘却の効果

本システムは記憶ネットワークが情報とその関係性を記憶し，忘却機能によって情報の想起の精度を維持しつつ，ネットワーク全体をコンパクト化することを目的とする．よって理想的な忘却機能の効果は，よく想起される情報の関係性をしっかり記憶しつつ，その他のあまり想起されない情報と関係性の記憶を最小限に留めることであるといえる．

表 3.2 と 3.3 より，$n = 10$ ではどの条件でも長さ 10 の想起列を想起出来たのに対し，$n = 100$ では「あり」の場合では想起列の長さはほぼ 0 で，平均カウント数も非常に少なかった．これは情報の総数が増えた事によって同じ情報が出現する頻度が下がったため，ノードのカウントが増えずに次々に忘却されてしまった為だと思われる．一方で，最初に長さ 10 の想起列を教え込む手順 1*を行った「あり*」の場合では長さ 10 の想起列を想起出来たため,「あり」の条件でも特に記憶したい情報を集中して記憶させればノードの忘却を抑えることができると考えられる．

表 3.3 より,「エッジ」は「なし」と同様の長さの想起列を想起しつつ，エッジの使用数を大幅に削減出来ていた．また,「あり*」の場合では長さ 10 の想起列を想起しつつ，それ以外の記憶を除いたコンパクトなネットワークを形成出来ていた．よって，特定の情報のみ記憶したい場合はノードの忘却は有効であり，そうでない場合はエッジの忘却のみでも十分な効果（ネットワークのコンパクト化）が得られると考えられる．

3.6　動的目標獲得システムの実装案としての有効性

本システムは第 2 章で提案した動的目標獲得システムにおける記憶サブシステムの実装案であり，記憶サブシステムのベースとなるシステムである．提案システムの課題であったシステムの巨大化という問題に対し，本章の評価実験で確認したエッジおよびノードの忘却の機能の効果は，記憶サブシステムとしての性能を保ちつつデータをコンパクト化するための方法として有効であると考えられる．一方で，提案システムが人間のような柔軟な目標設定を可能とするためには，記憶サブシステムは本システムよりも更に複雑な情報やその関係性を扱う必要がある．それを踏まえると，本システムは記憶サブシステムのベースとして，より高度な想起機能の実現や情報の表現能力の向上などいくつかの改善の余地が残されていると考えられる．

第4章　最適でない方策の学習のための動的な報酬設定システム

4.1　システムの概要

本章では，第2章で提案した動的目標獲得システムの実装案である，最適でない方策の学習のための動的な目標獲得システムを提案する．本システムではゲームクリアなどの単純なゲーム内報酬に基づく最適方策を学習したのち，学習した方策と類似の方策に制限を加えるように報酬関数を更新することで，次の学習で最適でない方策を学習する可能性を高める．この制限によって，本システムは迷路ゲームにおいてあえて遠回りな経路でゴールへと向かうような，最適でない方策を学習するための報酬設定を動的に更新する．

4.2　人間プレイヤは常に最適なゲームプレイを目指すとは限らない

人間プレイヤがゲームをプレイする際，常にゲーム本来の目的に対する最適なゲームプレイを目指すとは限らない．人間プレイヤがゲームをプレイする大きな理由の1つに「楽しさ」があるが，ゲーム中のどのような場面に楽しさ（興奮や感動など）を感じるかは，ゲームの性質やプレイヤの性格によって変化することが知られている [21]．また，楽しさ以外の人間の感情もゲームプレイに影響を与える重要な要素であり [4]，感情は行動決定における価値評価の基準としての役割をもつと考えることが出来る [22]．また，栗原はゲーム本来の目的を阻害しない範囲での行動選択の自由さを余剰自由度と定義し，この余剰自由度の有無が人間プレイヤの非ゲーム的目的の達成に影響を与えると考察している [23]．以上のように，ゲームプレイにおける人間プレイヤの目標は様々な要因によって変化し，その内容によっては一見非効率的に見えるような，最適でないゲームプレイを行う場合もある．

ゲームAIが人間プレイヤの最適でないゲームプレイを再現する場合，記憶や感情等の内部状態から全て再現するのは難しい．一方で，最適でないゲームプレイを行うという見かけ上の振る舞いのみに限定すれば，それを再現するための現実的な方法はいくつか考えられる．本章ではそのような方法のひとつとして，ある目標に対する過去の最適方策と同様の方策の学習を制限することで，最適でない方策を学習する確率が高まるように目標設定を更新するシステム

を提案する．

4.3　最適でない方策の学習の目標設定

ゲーム AI の学習における目標設定を，ゲーム AI の行動に対する報酬設定で表現することを考える．このとき，ある目標に対する最適方策を学習する目標設定より，最適でない方策を学習する目標設定の方が複雑になりやすいと考えられる．たとえば迷路ゲームで「ゴールに向かう」という目標に対する最適方策を学習する目標設定は，ゴールへ到達したときの正の報酬 (ゴール報酬) と，時間経過に応じた負の報酬 (移動コスト) という 2 つの報酬設定で簡単に表現できる．一方で,「ゴールに向かう」という目標に対する最適でない方策を学習させる，すなわち「遠回りしてゴールする」や「より遠いゴールへ向かう」などの方策を学習させる報酬設定は，単純なゴール報酬や移動コストのみでは表現が難しく，最適方策の学習と比較してより複雑な報酬設定が必要となる．

そこで本章では，過去に学習した最適方策と同様の行動に制限を加えることによって，最適でない方策の学習のための目標設定を動的に更新するシステムを提案する．本システムでは現在の目標設定における最適方策を学習したあと，その方策と同じ行動をした場合に得られる報酬が減少するように目標設定を更新する (図 4.1)．これにより，AI は過去と同様の最適方策を学習しにくくなり，最適ではない，異なる解に至る方策を学習する可能性が高まる．

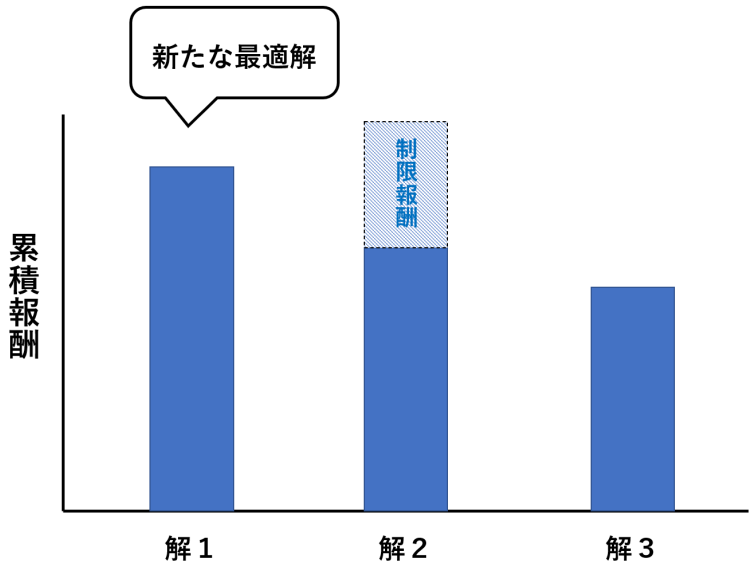


図 4.1: 最適でない方策の学習のための報酬設定の更新のイメージ．初めに解 2 が求まり，その後に制限報酬を加えることで，学習によって解 1 が得られるようにする．

本システムの利点は，学習初期の目標設定は最適方策の学習に必要な目標設定のみでよく，その後の最適でない方策の学習のための目標設定の更新は自動的に行えることである．また，目標設定を報酬設定のみで表現しているため，強化学習などの学習手法と組み合わせることが

容易であることも利点のひとつである．

4.3.1 Q 学習

Q 学習 [24][25] は強化学習 [7] のアルゴリズムのひとつであり，状態 s における行動 a の価値 (行動価値)$Q(s,a)$ の値を探索によって更新していく手法である．関数 $Q(s,a)$ は式 (4.1) で更新する．α は学習率，r は報酬，γ は将来の報酬の割引率であり，s' は次の状態である．

$$\begin{aligned} Q(s,a) = & Q(s,a) \\ & + \alpha(r + \gamma \max_a Q(s',a) - Q(s,a)) \end{aligned} \tag{4.1}$$

関数 $Q(s,a)$ の値は次の状態で行動価値最大の行動を選択するという貪欲法に基づいて更新されるが，実際の探索では探索を上手く進めるために確率でランダム行動をとる ε-greedy 方策や，行動価値の重みに応じた確率で行動を選択する softmax 方策などがよく利用される．このように，Q 学習は更新式の方策と実際の方策が異なることから方策オフ型の学習法と呼ばれる．

4.4 動的な報酬設定の更新

本システムにおける報酬設定を，状態遷移に応じた報酬を返す報酬関数 $R(s,s')$ として定義する．s は現在の状態，s' は次の状態である．$R(s,s')$ の値は，式（4.2）より基本報酬 $r_b(s,s')$ と制限報酬 $r_r(s,s')$ の和として算出する．

$$R(s,s') = r_b(s,s') + r_r(s,s') \tag{4.2}$$

基本報酬は学習の初期目標として設定する報酬であり，全学習を通して同じ値を返す．制限報酬は過去に学習した方策と同じ行動を制限するための報酬であり，学習が終わるごとに値が更新される．制限報酬の更新によって，本システムにおける動的な報酬設定の更新を実現する．

4.4.1 制限報酬の意味と更新方法

本システムにおける制限報酬 r_r は学習した方策と同様の状態に至る行動に与える負の報酬である．r_r によって，本システムは過去と異なる方策を学習する確率が高まるように目標設定を

更新する．例えば，図 4.2 ではスタートからゴールへと向かう経路は青い経路が最適解である．本システムでは一度この最短経路を学習したあと，同様の経路を通過するような行動に対して負の報酬を与えるように制限報酬を更新する．これにより，二度目に学習するときの最適解は一度目の経路を迂回してゴールへ向かう赤い経路となる．このように，r_r には過去に学習した最適方策とは異なる方策を学習する可能性を高める効果がある．

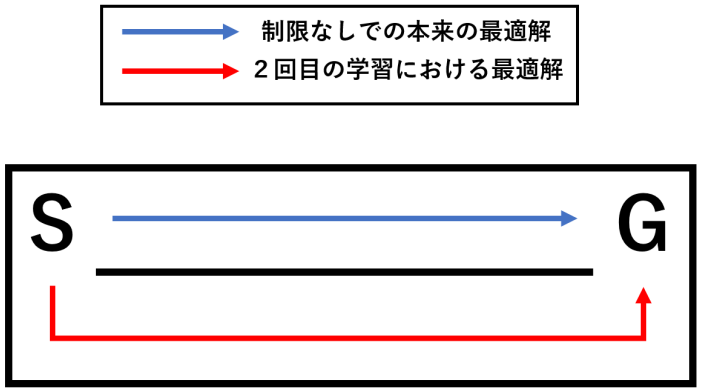


図 4.2: 制限報酬による最適な経路の変化

制限報酬 $r_r(s, s')$ は式 (4.3) で更新する．s_t は学習した方策 $\pi(a|s)$ に基づく行動の軌跡 $\mathbb{T} = \{s_1, s_2, ..., s_n\}$ の要素である．α_r は制限報酬の学習率である．

$$r_r(*, s_t) = (1 - \alpha_r) r_r(*, s_t) + \alpha_r \, r_l \tag{4.3}$$

ここで，r_l は制限報酬の下限値である．r_l は制限報酬によって更新に用いる軌跡上の報酬の累積値が負にならないように設定する．これは制限報酬によって報酬の累積値が負になる状態が存在する場合，そのような状態に阻まれどの初期目標へも向かわなくなってしまう可能性があるためである．このような事態を避けるために，本システムでは制限報酬の値に下限を設ける．AI の学習法に Q 学習を用いる場合は，r_l の値は報酬の累積値 r_s, 軌跡の長さ n と行動価値の更新式における将来の報酬の割引率 γ より，式（4.4）で求まる．δ は更新後の報酬の累積値を正にするための小さな正の実数である．なお，式（4.4）の右辺第 1 項は漸化式 $a_{n+1} = \gamma(a_n - r_l), a_1 = r_s - r_l$ から導出される．

$$r_l = r_s \frac{\gamma^{n-1}(\gamma - 1)}{\gamma^n - 1} + \delta \tag{4.4}$$

4.5　動的目標獲得システムとの対応

本システムは 2.3 節で述べた動的目標獲得システムの機能を制限した実装案である．本システムでは動的目標獲得システムの 3 つのサブシステムを報酬関数 R で代替しており，それぞれ以下のように対応している．

思考領域サブシステム：制限報酬による報酬関数の更新

本システムにおける制限報酬 r_r による R の更新が，動的目標獲得システムの思考領域サブシステムに対応する．本システムでは，学習した方策によって観測した状態 s への状態遷移に対して，r_r を与えるように R を更新することで，最適でない方策の学習のための目標設定を動的に更新する．これが目標設定に必要な情報の収集・処理を行うという思考領域サブシステムの働きに相当する．

本能サブシステム：過去と異なる方策の学習を強制する力

本システムにおける過去と異なる方策を学習を強制する力が，動的目標獲得システムの本能サブシステムに対応する．本システムでは，R は常に過去と異なる方策を学習するように報酬の値を更新する．この報酬設定に対する強制力が，目標設定のきっかけを与えるという本能サブシステムの働きに相当する．

記憶サブステム：報酬関数の値

本システムにおける R の値が，動的目標獲得システムの記憶サブシステムに対応する．本システムでは，R の値は r_r による更新の反復によって，過去と異なる方策を学習する報酬設定となるように少しづつ変化していく．これが，思考領域サブシステムの情報の保存・管理・取り出しを行い，ゲーム AI の過去の経験を蓄積するという記憶サブシステムの働きに相当する．

4.6　評価実験 1：複数のゴールがある迷路環境における経路学習

本節では本章で提案する最適でない方策の学習のための動的な報酬設定システムついて，本システムと Q 学習を用いて 3 つのゴールがある迷路ゲーム環境における経路学習を行うことで，最適でない方策をどの程度学習できるかを確認した評価実験とその結果を説明する．

4.6.1 目的と概要

本実験の目的は，本システムを用いて 3 つのゴールが存在する迷路ゲーム環境での経路学習を行うことで，より遠いゴールへの到達という最適でない方策をどの程度学習できるか確かめることである．本ゲーム環境における本来のゲーム目標はより短い経路でゴールすることであり，3 つのゴールのうちもっともスタート地点に近いゴールへと最短経路で向かうことが最適解となる．

本実験では 3 つゴールがある迷路マップおよび壁を減らした準迷路マップを用意し，提案手法を用いた Q 学習と通常の Q 学習による経路学習を行った．学習後，獲得した方策によってゲーム AI エージェント（以下，エージェント）がどのゴールに到達したか記録し，最適でない (より遠い) ゴールへの到達回数がどの程度変化するか確認した．

4.6.2 準備

実験環境

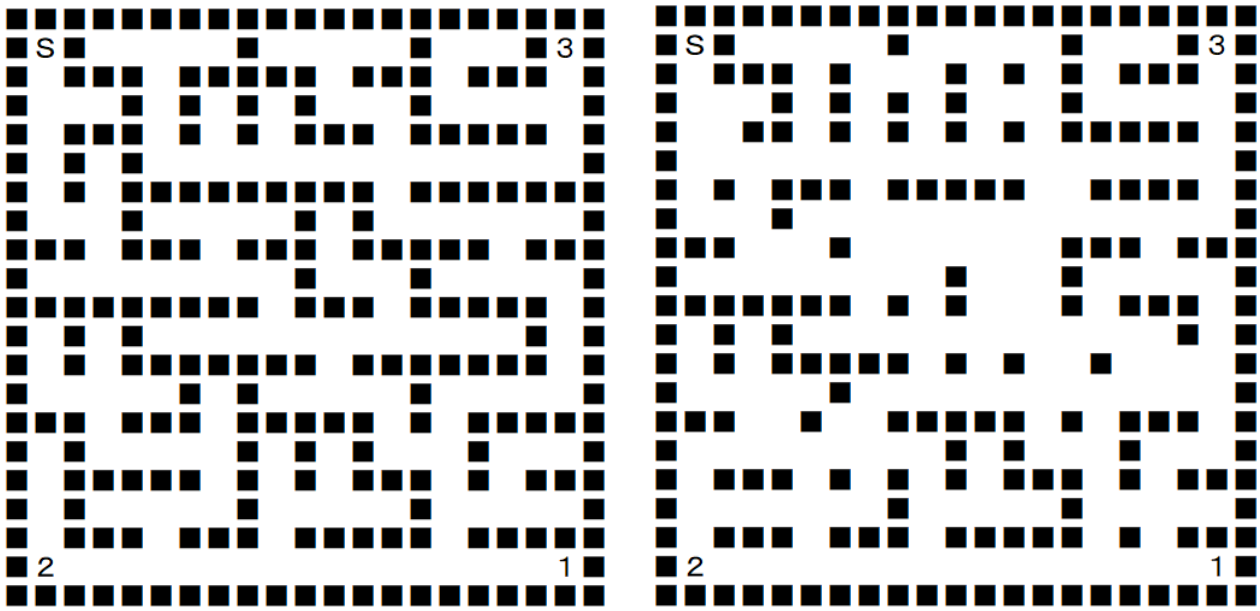
本実験は図 4.3 の 2 つの 2 次元格子状マップで行った．各マップは左上の隅をスタート地点，残りの 3 つの隅をゴールとした．迷路マップは壁延ばし法のアルゴリズム [26] で作成し，準迷路マップは迷路マップから n 個の内壁をランダムに削除することで作成した．今回は $n = 40$ として作成した．壁延ばし法のアルゴリズムを採用した理由は，迷路の自動生成アルゴリズムのなかでも比較的ランダムな形状の迷路を生成しやすいためである．各マップで可能な行動は上下左右の通路への移動とした．環境から得られる観測情報はエージェントの現在の座標とした．

学習法

エージェントの経路学習には強化学習の 1 種である Q 学習を利用した．Q 関数の値はルックアップテーブル形式で用意し，テーブルの値は 0 以上 1 未満のランダムな値で初期化した．

探索手法

エージェントの経路学習時の探索手法には ε-greedy 法を利用した．



迷路マップ　　準迷路マップ

図 4.3: 実験に用いた迷路マップ

パラメータ

ゲームプレイは 1 エピソードあたり最大 200 ステップとし，学習 1 回あたり 1000 エピソードのゲームプレイを行った．基本報酬 r_b はゴール報酬を 100，1 ステップ毎の移動コストを-0.1 とした．制限報酬 r_r の更新における学習率 α_r は 0.001 とし，各学習の最後のエピソードの行動の軌跡と累積報酬値を報酬関数の更新に利用した．Q学習の学習率 α は 0.3，割引率 γ は 0.9995 とした．ε-greedy 法におけるランダム行動確率 ε は 0.3 とし，1 エピソード毎の減衰率は 0.99 とした．

4.6.3　手順

通常の Q 学習による経路学習

ゲームプレイはいずれかのゴールへの到達か 200 ステップの経過で 1 エピソードとし，合計 1000 エピソードの学習を 1000 回行った．1 回の学習が終わるごとに学習した方策によって到達したゴールを記録した．

本システムと Q 学習を用いた経路学習

学習 1 回の手順は通常の Q 学習と同様に行い，合計 1000 回の学習を行った．1 回の学習が終わるごとに学習した方策によって到達したゴールを記録し，制限報酬による報酬関数の更新を

行った上で次の学習を行った．

4.6.4　結果

各マップにおける実験結果を図 4.4 および図 4.5 に示す．GOAL1 から 3 は各ゴールへの到達回数であり，NOGOAL はいずれのゴールへも到達できなかった回数である．青が提案手法を用いた Q 学習の結果，緑が通常の Q 学習の結果である．

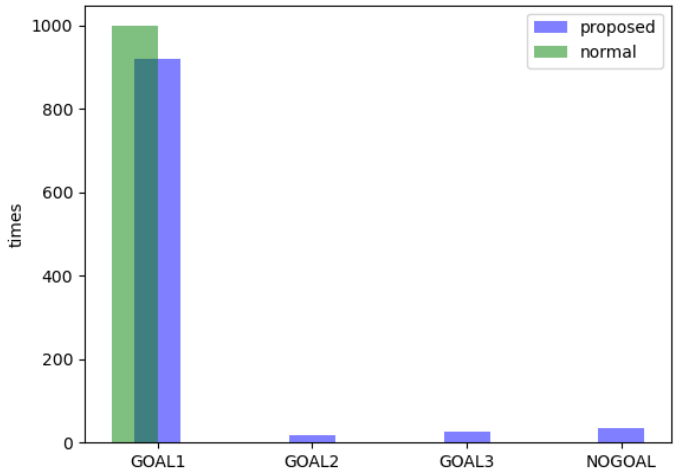


図 4.4: 迷路マップにおける各ゴールへの到達回数の比較

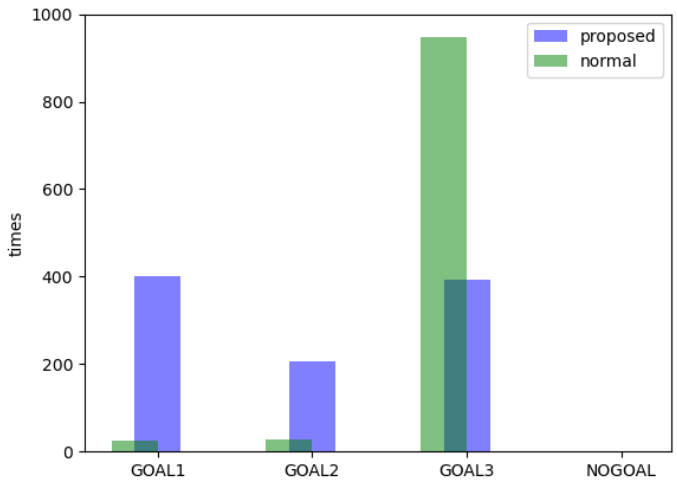


図 4.5: 準迷路マップにおける各ゴールへの到達回数の比較

4.7　本システムによる最適でない方策の学習確率

どちらのマップにおいても，本システムを用いた方が各ゴールへの到達回数の偏りが少なくなり，最適でない方策の学習確率が高まることが確認できた．特に，迷路マップでは通常のQ学習ではほぼ GOAL1 にしか到達できていないことから，より遠いゴールへと向かいやすくするという制限報酬の効果が有効に働いていたといえる．ただし，迷路マップでは本システムを用いた場合でも GOAL1 以外のゴールへの到達回数が少ないという結果になった．特に，学習後期においては GOAL1 以外のゴールに到達することがほぼなかった．これは，GOAL2 と GOAL3 に到達する経路は必ず GOAL1 の近くを通るため，それぞれのゴールに到達した後の制限報酬の更新によって GOAL1 への到達が再度最適解になりやすかったためだと考えられる．また，迷路マップにおける結果ではどのゴールへも到達できないケースが生じていたが，これは制限報酬によって一番近い GOAL1 への到達でさえも最適解でなくなってしまったためだと予想される．

4.8　評価実験2: 学習のパラメータや環境が与える影響の調査

本節では本章で提案する最適でない方策の学習のための動的な報酬設定システムついて，本システムを用いた学習が環境やパラメータにどのような影響を受けるか確認した評価実験とその結果を説明する．

4.8.1　目的と概要

本実験の目的は学習パラメータや迷路マップを変更して経路学習を行い，本システムを用いた学習が上手く行く条件を調査することである．

本実験では 4.6 節の評価実験 1 の環境設定やパラメータを基本とし，迷路マップや各パラメータを変更した経路学習を行うことで，各ゴールへの到達回数がどのように変化するか確認した．

4.8.2　準備

実験環境

本実験では，学習パラメータを変更して行う学習は図 4.3 の迷路マップで行った．迷路マップを変更して行う学習は自動生成した計 100 種類の迷路マップを使用した．迷路マップにおけるゴールの位置は全て同じとし，いずれかのゴールが別のゴールへの経路を塞いでいるマップ

は除外した．

学習法

4.6 節の評価実験 1 と同様にした．

探索手法

4.6 節の評価実験 1 と同様にした．

パラメータ

基本のパラメータは 4.6 節の評価実験 1 と同様にした．パラメータを変更して行う学習は，制限報酬の学習率 α_r，Q学習の行動価値更新式における将来の報酬の割引率 γ，基本報酬 r_b における移動コストのいずれかひとつを変更して行った．

4.8.3 手順

学習パラメータを変更して行う経路学習

学習の手順は評価実験 1 の本システムと Q 学習を用いた経路学習と同様に行った．1000 回の学習が終わる毎に到達したゴールを記録し，パラメータを変更して次の 1000 回の学習を行った．

迷路マップを変更して行う経路学習

学習 1 回の手順は評価実験 1 の本システムと Q 学習を用いた経路学習と同様に行った．各マップにおいて 3 つのゴール全てに到達する方策を学習するか学習回数が 100 回に達した時点で学習を終了した．学習終了後，到達したゴールの数と平均学習回数を記録し，次のマップでの学習を行った．

4.8.4 結果

実験の結果を図 4.6，4.7，4.8 および表 4.1 に示す．図の縦軸は各ゴールへの到達回数，横軸は学習パラメータの値であり，特に遠いゴールへ到達できた回数が多かった範囲を赤丸で囲っている．なお，遠いゴールへの到達回数を比較するため，GOAL1 への到達回数など極端に大

きい値は図から除外した．また，迷路マップを変更して行う経路学習において，3 つのゴール全てに到達出来たときの平均学習回数は 22.09 であった．

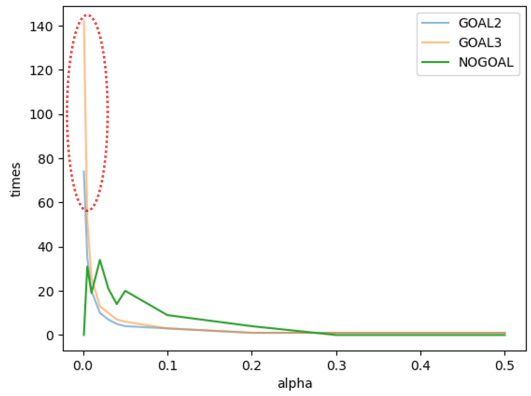


図 4.6: 学習率によるゴールへの到達回数の変化

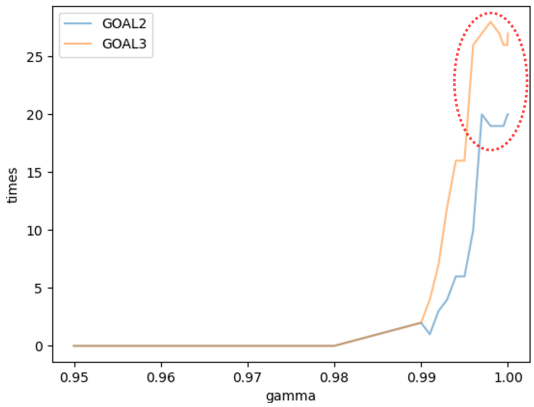


図 4.7: 割引率によるゴールへの到達回数の変化

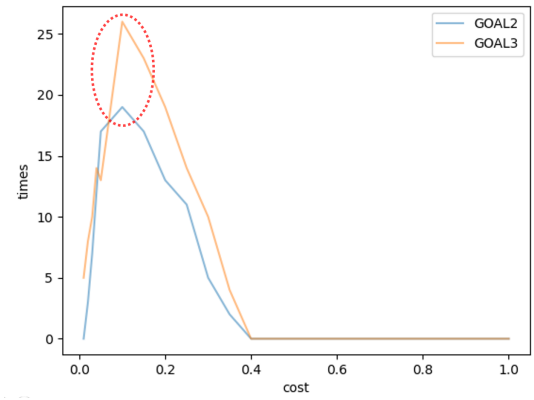
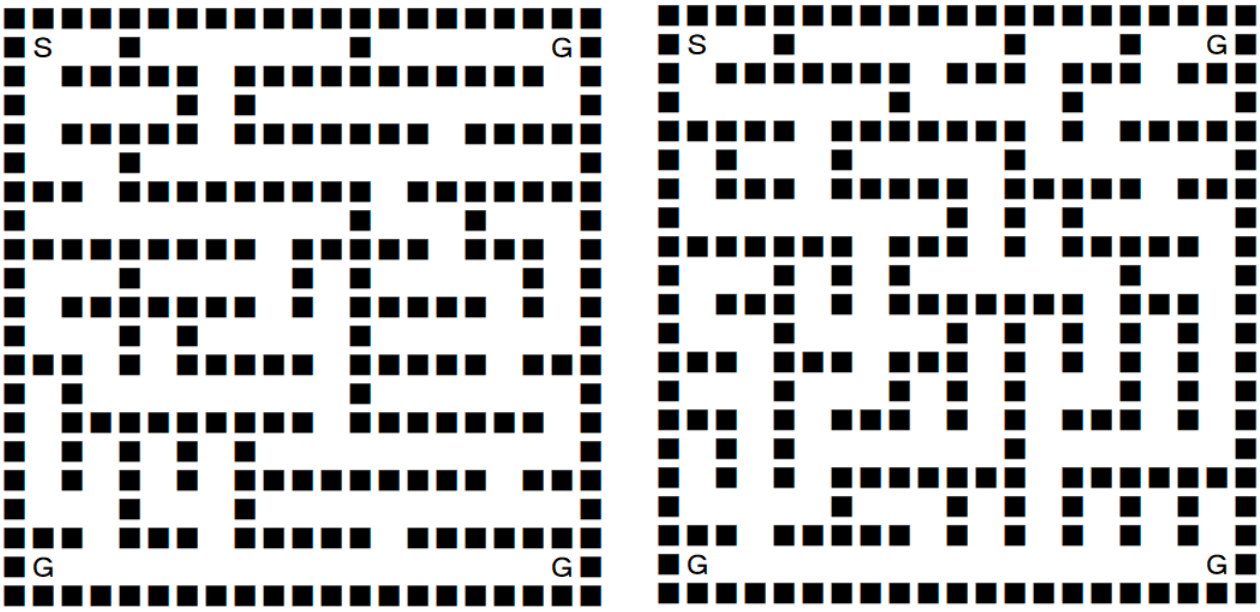


図 4.8: 移動コストによるゴールへの到達回数の変化

表 4.1: 100 種類の迷路マップの学習結果

到達したゴール	3	2	1
マップ数	33	56	11



ゴール到達数 3　　ゴール到達数 1

図 4.9: ゴールの到達数ごとの迷路マップの例

4.9　学習パラメータや環境が学習に与える影響

図 4.6 より，制限報酬の学習率は小さいほど遠いゴールへの到達回数が増えることが確認出来た．これは，制限報酬の学習率が大きいと一度到達した遠いゴールに到達できなくなってしまうまでの時間が短くなったためだと考えられる．図 4.6 では学習率が 0.02 を超えた辺りから遠いゴールへの到達回数が大きく下がっていることから，学習時には制限報酬の学習率を 0.01 以下の値に設定するのがよいと思われる．

図 4.7 より，割引率は 1 に近いほど遠いゴールへの到達回数が増える傾向にあることが分かった．これは，割引率が 1 に近づくことによって遠いゴールの報酬を大きく見積もり，そこに到達するような方策を学習しやすくなったためだと考えられる．図 4.7 では割引率が 0.995 を超えた辺りから遠いゴールへの到達回数が大きく増えるため，学習時には割引率を 0.995 以上の値に設定するのがよいと思われる．

図 4.8 より，移動コストはおよそ-0.1 付近で遠いゴールへの到達回数が最大となることが確認できた．これは移動コストは大きいほど近いゴールに向かいやすくなり，小さいほどどのゴールにも向かいにくくなるというトレードオフの効果をもつためだと考えられる．移動コストは正の報酬であるゴール報酬と対になる基本報酬であり，今回のゴール報酬は 100 であったことから，学習時には移動コストとゴール報酬の大きさの比を 1:1000 程度に設定するのがよいと思われる．

100 種類の迷路マップの学習実験では，表 4.1 よりおよそ 3 割のマップで 3 つのゴールに到達することができた．また，3 つのゴールに到達出来たときの平均学習回数は 22.09 であり，100 回の学習上限と比較して早い段階で達成できていた．このことから，迷路のマップとの相性に

よって提案手法の効果に大きな差が現れたことが分かった．特に，図 4.9 の 2 つのマップを比較して分かるように，近いゴールを無視しないと遠いゴールへ到達できないようなマップでは，遠いゴールへ到達する方策を学習しにくいという傾向が確認できた．これは，提案手法では制限報酬を軌跡上に一様に分布させるため，制限報酬の下限に至っても途中の分岐から近いゴールへ向かう方が報酬の累積値が大きくなってしまったためだと考えられる．

4.10 動的目標獲得システムの実装案としての有効性

本システムは第 2 章で提案した動的目標獲得システムの機能を制限した実装案である．提案システムの課題であったシステムの設計の複雑化という問題に対し，本システムは提案システムの 3 つのサブシステムの機能を制限報酬による報酬関数の更新というシンプルな手法で代替しつつ，最適でない方策の学習のための動的な目標設定を実現した．一方で，本システムは提案システムの機能を大幅に制限した結果，ある目標に対する過去の最適方策と異なる方策を探すことに特化したシステムとなっている．過去の最適方策を単純に制限するだけでは実現が難しい方策（たとえば接待プレイ [1] の方策など）を学習したい場合には，本システムの報酬関数とは異なるアプローチによる提案システムの設計が必要になると考えられる．

第5章　イベント認知に基づく目標獲得システム

5.1　システムの概要

本章では，第2章で提案した動的目標獲得システムの実装案である，イベント認知に基づく目標獲得システムを提案する．本システムではゲームクリアやアイテムの取得など，ゲームプレイの目標として設定可能な情報を「イベント」として定義し，ゲームAIが認識したイベントを用いて新たな目標の獲得や目標設定の動的な更新を実現する．

5.2　ゲームにおけるイベントと目標設定の関係

人間やAIなどのプレイヤがゲームをプレイするとき，そのプレイヤはゲームにおいて達成したい目標をもつ，あるいは与えられることがある．ゲームプレイにおけるプレイヤの目標は様々であり，ハイスコアの獲得や対戦相手への勝利などその基準が明確なものもあれば，楽しみたい，感動したいという曖昧な目標も存在する．中川らはゲームにおける人間プレイヤの行動について，ゲーム本来の目的との関係性やゲーム内での行動であるかどうかという基準から，ゲーム内主目的行動，ゲーム外主目的行動，ゲーム内非主目的行動，ゲーム外非主目的行動の4種類に分類し，これらのうちゲーム内非主目的行動にあたる行動の分類と考察を行った[5]．本章では，中川らの区分のうちゲーム内主目的・非主目的行動に関係する，ゲーム内のみで達成可能なプレイヤの目標設定(以下，ゲーム内目標)について考える．

5.2.1　イベント

あるプレイヤがゲーム内目標を設定し，ゲームプレイによってそれを達成したかどうか判断するためには，プレイヤがゲーム内における「目標を達成した状態」を他の状態と区別して認識する必要がある．たとえばアクションゲームにおいて「現在のステージをクリアする」というゲーム内目標を設定するためには，クリア時の演出や画面遷移等の情報から，そのステージをクリアしたことを認識出来なければならない．逆に，そういった情報に対して「ステージク

リア」という意味付けが出来なければ,「現在のステージをクリアする」というゲーム内目標を達成することは不可能である.

本システムにおける「イベント」とはあるプレイヤがゲーム内で他の情報と区別し，意味付けを行う事の出来る情報である．ここで，意味付けとはゲーム内情報集合 I からプレイヤの持つ概念集合 C への写像 f である（式 5.1）.

$$f : I \longrightarrow C \tag{5.1}$$

概念集合とは，プレイヤがゲーム内情報を解釈するために用いる，情報の見方 (概念) の集合である．写像 $f(i)$ が空でないゲーム内情報 $i \in I$ の集合を，プレイヤのイベント集合 $E \subset I$ と定義する（図 5.1）．写像 f によって概念 $c \in C$ に意味付けされたゲーム内情報 $i \in I$ は，プレイヤにとって c という意味のある「イベント」$e \in E$ となる.

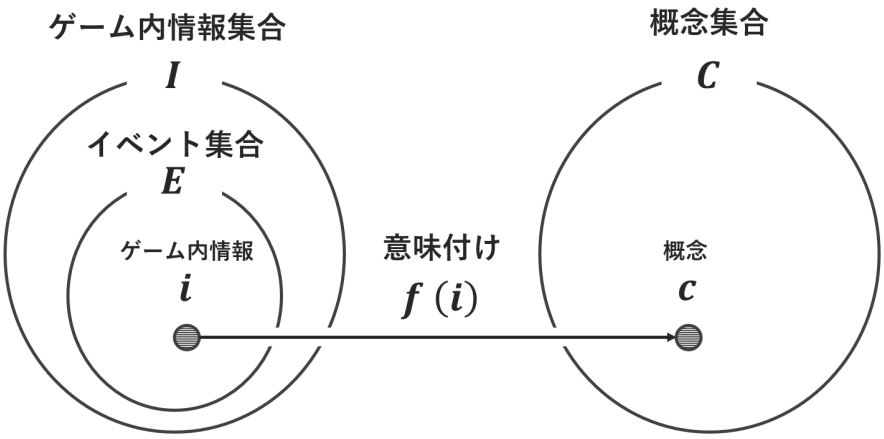


図 5.1: 意味付けとイベント集合のイメージ

一般的な用語としてのイベントは，ゲームにおけるアイテムの取得や敵との戦闘などの，開発者によって設定された出来事（または事件）を意味することが多いが，本システムにおける「イベント」はそれよりもより広い意味を持つ．たとえば，ゲーム画面の背景に雲が流れる様子は一般的にはイベントとみなされない．しかし本研究ではプレイヤがその情報に意味付けを行い，それ（雲の流れる様子）を見るなどのゲーム内目標を設定できるのであれば，そういった情報もプレイヤにとっての「イベント」のひとつであると考えることが可能である.

5.2.2　イベントの認知

あるプレイヤが画面や音などのゲーム内情報によってイベントの発生に気付くことを，イベントを「認知する」と定義する．認知されるイベントは，プレイヤにとって既知のものだけでなく，そのイベントを認知して初めてその存在を知ったもの（たとえば，ゲーム内オブジェクトの意外な挙動など）も含まれる．プレイヤが事前知識として持っていないイベントについても，ゲームプレイを通じて新たな知識として獲得し，それを目標として設定することが考えられる．

5.3　ゲーム AI のイベント認知

人間プレイヤの場合，他のゲームやゲーム以外の日常生活で得た膨大な事前知識をもとに，自分にとって「何がイベントであるか」を判断することが出来る．しかし，多くのゲーム AI は事前知識やその利用方法を開発者が用意しなければならない．本節ではゲーム AI がイベントを認知し，新たな目標設定を獲得するのに必要な識別機について説明する．

5.3.1　識別機

ゲーム AI がイベントを認知するためには，観測したゲーム内情報の中から，イベントを表す情報を識別する必要がある．ゲーム AI がもつ事前知識とゲーム内情報を元に，イベントの認知を判定する仕組みを識別機と定義する．識別機はゲーム内情報を入力，イベント情報を出力とする．識別機が内部処理に利用する事前知識の設定方法は，ゲームプレイの開始前に設計者が与える他に，ゲームプレイ中の経験をもとに新たな知識を更新していく方法なども考えられる．

5.3.2　認知率

ゲーム AI が識別機を用いてイベントの認知を行う場合，観測した情報や事前知識の性質によって，イベントの誤認識が生じる可能性がある．これは例えば，ゲーム画面に偶然生じたノイズによって，識別機が存在しないイベントを認知してしまう場合が考えられる．この場合，イベント認知の再現性がないため，誤認したイベントを目標として設定している場合のゲームプレイが不安定になるという問題が生じる．このような識別機の性能によるイベント認知の精度を認知率と定義する．5.7 節では特定のイベントに対する識別機の認知率が，ゲーム AI のゲーム内目標や行動にどのような影響を与えるかを調査した評価実験について説明する．

5.4　イベント認知に基づく目標獲得システム

本節ではイベント認知に基づく目標獲得システム（以下，本システム）の構成と動的目標獲得システムとの対応，および具体的な設計方法を説明する．

5.4.1　システムの構成

本システムは識別機，イベント記憶機，目標設定機，の 3 つの機能から構成される（図 5.2）．これは，本システムがイベントの認知と目標設定の更新を行うためには,「イベントの認知」と「初めて認知したイベントと既知のイベントの区別」と「認知したイベントによる目標設定の更新」の 3 つの機能が必要なためである．

識別機はゲーム AI が観測したゲーム内情報を入力とし，ゲーム AI が認知したイベント情報を出力する．イベント記憶機はイベント情報を入力とし，ゲーム AI が過去に経験したイベント情報を出力する．目標設定機はイベント情報と過去に経験したイベント情報を入力とし，ゲーム AI の目標設定の更新と，目標達成の判定情報の出力を行う．本システム全体としては，ゲーム AI が観測したゲーム内情報を入力とし，ゲーム AI の目標設定の更新と，ゲーム AI が目標を達成したかどうかの判定情報の出力を行う．目標設定機の出力である目標達成の判定情報は，強化学習を用いたゲーム AI であれば，学習機に渡す報酬の値として表現する方法が考えられる．

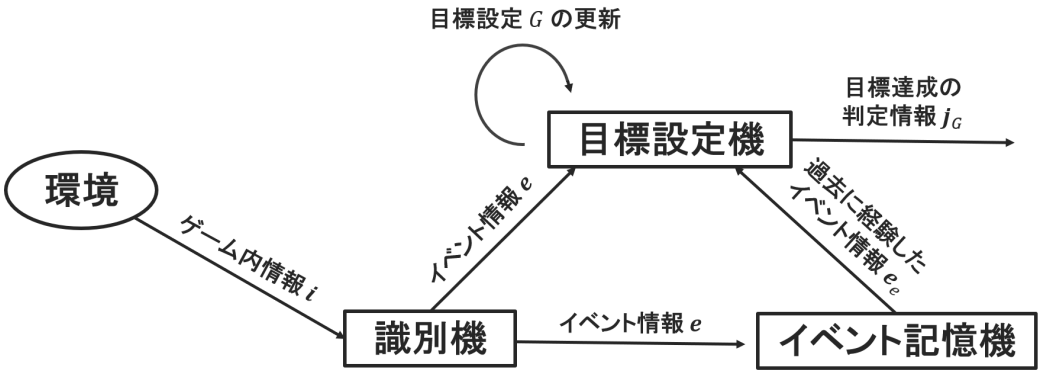


図4：イベント認知に基づく動的目標獲得システムの構成

図 5.2: イベント認知に基づく目標獲得システムの構成

識別機はイベント認知に必要な事前知識 (概念集合) を持ち，それを用いて入力されたゲーム内情報を意味付けることにより，イベント情報への変換・出力を行う．イベントが存在しない場合は「イベントなし」という情報を出力する．イベント記憶機は識別機が出力したイベント情報を過去に経験したイベントの履歴に追加し，それが初めて認知したイベントか，または過去に認知したことのあるイベントであるかを出力する．目標設定機は認知したイベントが初めて

認知したものであれば新しいゲーム内目標として目標候補に追加する．そうでなければ現在のゲーム内目標と認知したイベントを比較し，両者が同じイベントであった場合は現在のゲーム内目標を達成したという判定情報を出力する．また，目標設定器は現在のゲーム内目標が達成された後，また達成していない目標候補のうち最も古いものを新しいゲーム内目標として設定する．全ての目標候補を達成した場合は「ゲーム内目標なし」という目標を設定する．以上より，本システムは事前知識を用いて観測したゲーム内情報からイベントを認知し，ゲーム内目標の設定・更新と，ゲーム内目標達成の判定情報の出力を行う．

5.4.2　動的目標獲得システムとの対応

本システムは 2.3 節で述べた動的目標獲得システムの機能を制限した実装案である．動的目標獲得システムの 3 つのサブシステムについて，本システムの各機能は以下のように対応している．

思考領域サブシステム：識別機，目標設定機

本システムにおける識別機，および目標設定機が動的目標獲得システムの思考領域サブシステムに対応する．本システムでは，識別機が観測したゲーム内情報をイベント情報として処理し，目標設定機が目標設定の更新と目標達成の判定を行う．これらが目標設定に必要な情報の収集・処理を行うという思考領域サブシステムの働きに相当する．

本能サブシステム：目標設定機

本システムにおける目標設定機が動的目標獲得システムの本能サブシステムに対応する．本システムでは，目標設定機は常に何らかのイベントへと向かうゲーム内目標を設定する．この目標設定に対する強制力が，目標設定のきっかけを与えるという本能サブシステムの働きに相当する．

記憶サブステム：イベント記憶機

本システムにおけるイベント記憶機が動的目標獲得システムの記憶サブシステムに対応する．本システムでは，イベント記憶機は識別機が認知した過去のイベントを記憶し，目標設定機の要求に応じてそれを取り出して渡す．これが，思考領域サブシステムの情報の保存・管理・取り出しを行い，ゲーム AI の過去の経験を蓄積するという記憶サブシステムの働きに相当する．

5.4.3 設計方法

本システムは併用するゲーム AI の性質によって，様々な設計方法が考えられる．ここでは，本システムを強化学習 AI と組み合わせて使用し，強化学習における報酬設定の代替として用いる場合の各機能の設計方法を説明する．ただし，報酬機はゲーム内目標の達成の判定情報を報酬の値に変換するための補助的な機能である．

識別機

識別機の入力は観測したゲーム内情報を表すベクトルとし，出力は認知したイベント情報を表すベクトルとする．イベント情報の次元は認知したイベントの数に等しく，ベクトルの各成分はイベントを表す整数値とする．認知したイベントがない場合は空のベクトルを出力する．事前知識（概念集合）はゲーム内情報のベクトルをイベント情報のベクトルへと変換するフィルタとする．ただし，イベントに関するメタ情報なしにフィルタを設計するのは難しい．たとえば CNN[27] をフィルタとして利用する場合，各概念を別のフィルタとして用意する場合はフィルタ全体のサイズが巨大になり，複数の概念をひとつのフィルタで変換する場合はイベント認知の精度（認知率）の維持が困難になると予想される．本章の評価実験ではフィルタの代わりにイベントのメタ情報を直接受け取るように設計した識別機を利用した．このメタ情報とは，プレイヤの行動によって発生したイベントを表す番号である．この場合識別機の認知率は常に 100%であるため，一定の確率で認知したイベント情報を破棄することにより，擬似的な認知率を表現した．

イベント記憶機

イベント記憶機の入力はイベント情報を表すベクトル，出力は過去に認知したイベント情報を表すベクトルとする．出力のベクトルは入力のベクトルの各要素を，イベントの履歴に含まれていれば 1，含まれていなければ 0 に変換したものとする．イベントの履歴は過去に入力として渡されたイベント情報のベクトルの要素の集合とし，入力されたベクトルの要素のうち現在の履歴に含まれないものを追加して更新する．

目標設定機

目標設定機の入力はイベント情報のベクトルと過去に認知したイベント情報のベクトルとし，出力は目標達成の判定情報とする．出力の判定情報は，ゲーム内目標を達成した場合は達成した目標に対応するイベント情報のベクトル，そうでない場合は空のベクトルとする．目標候補はゲーム内目標（イベント）を添え字，達成の有無を表す情報を値とする連想配列とする．連想配列の値は，まだ達成していない場合は 0，現在の目標として設定されている場合は 1，既に達成している場合は-1 とする．過去に認知したイベント情報のベクトルの要素のうち 0 である

もの，すなわち初めて認知したイベントがある場合はそのイベントを目標候補の末尾に追加する．入力のイベント情報のベクトルの要素と目標候補を比較し，現在のゲーム内目標と同じイベントがあれば出力のベクトルに追加し，次のゲーム内目標を未達成の目標候補の先頭から設定する．これを現在のゲーム内目標と同じイベントがなくなるか，未達成の目標候補が無くなるまで繰り返す．達成したゲーム内目標は，外部から初期化の信号を受ける事で再設定可能となる．

報酬機

報酬機の入力は目標達成の判定情報のベクトル，出力は報酬 (実数値) とする．報酬機は内部に報酬リストをもち，報酬リストはゲーム内目標（イベント）を添え字とし，達成時の報酬を値とする連想配列とする．入力のベクトルの各要素 (イベント) と報酬リストを比較し，報酬リストに存在する場合は対応する報酬の値を出力する報酬に加算する．報酬リストに存在しない場合はイベントと対応する報酬値を新たに追加したあと，報酬の値を出力する報酬に加算する．また，時間経過に応じた負の報酬など，ゲーム内目標の達成に依らない報酬も設定可能とする．

5.5 評価実験 1：L 字形マップにおけるイベント認知と経路学習

本節では本章で提案するイベント認知に基づく目標獲得システムついて，本システムを用いたイベントの認知とゲーム内目標の設定を行うことが出来るか確認した評価実験とその結果を説明する．

5.5.1 目的と概要

本実験の目的は本章で提案するイベント認知に基づく目標獲得システムを用いて，ゲーム AI がイベントの認知によって新たなゲーム内目標を獲得し，目標設定を動的に更新出来るか確かめることである．

本実験ではゲーム AI の 2 次元格子状マップにおける経路学習を，本システムおよび Q 学習を用いて行わせた．使用したマップは図 5.3 のような L 字形のマップであり，L 字の通路の角の部分にスタート地点（S）があり，L 字の通路の一方の突き当りにゴール地点（G）を，もう一方の突き当りにアイテム取得のイベント (!) を設置した．ゲーム AI エージェント (以下，エージェント) はゲーム開始時点ではゴール到達やアイテム取得というイベントがこのマップに存在することを知らず，実際の探索を通じてそれらのイベントを発生させることにより，それらのイベントの認知とゲーム内目標への設定が可能となる．エージェントが認知したイベントを目標として設定し，そこへ向かう経路を学習させることにより，どのような順でイベントを認知

し，目標として設定できたかを確認した．実験は 2 種類行い，それぞれ L 字形マップの通路の長さ (スタート地点と各イベントの距離) を変更しながら実験することで，マップの形状によってエージェントの目標設定がどのように変化するか確認した．

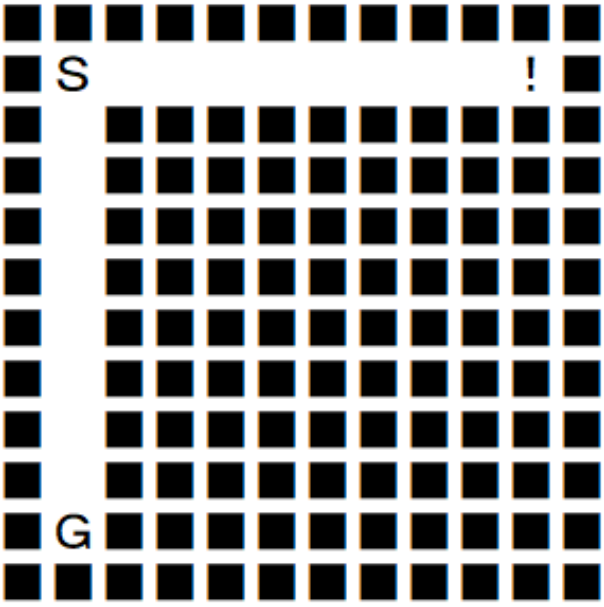


図 5.3: L 字形マップ

5.5.2 準備

実験環境

本実験では図 5.4 と図 5.5 の 2 種類のマップを使用した．図 5.4 の L 字形マップはスタート地点から各イベントまでの距離の比が等しいマップをサイズを変えて複数用意した．最小のマップはスタート地点と各イベントの距離が 1 のものとし，最大のマップはスタート地点と各イベントの距離が 28 のものとした．図 5.5 の L 字形マップはスタート地点からゴール地点までの距離を 10 とし，スタート地点からアイテム取得イベントまでの距離を変化させたマップを複数用意した．最小のマップはスタート地点からアイテム取得イベントまでの距離が 10 のものとし，最大のマップはスタート地点からアイテム取得イベントまでの距離が 28 のものとした．各マップで可能な行動は上下左右の通路への移動とした．環境から得られる観測情報はエージェントの現在の座標と後述するイベントの発生情報とした．

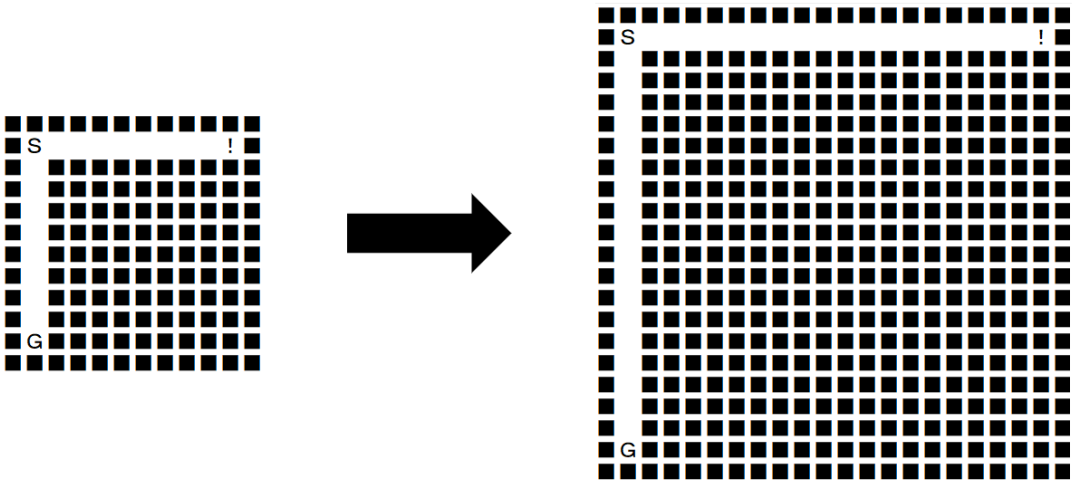


図 5.4: 評価実験 1 の L 字形マップ 1

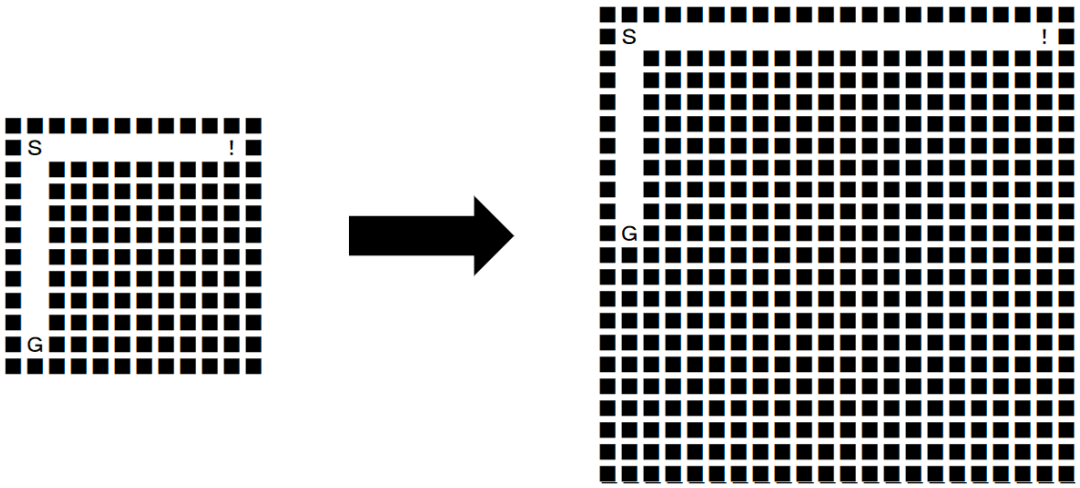


図 5.5: 評価実験 1L 字形マップ 2

イベント

L 字形マップにおけるイベントは「ゴール到達」と「アイテム取得」の 2 種類を用意した．それぞれのイベントはエージェントがマップ上のシンボル（Gと！）に到達することで発生し，観測情報としてイベントを表す整数値を受け取る．ゴール到達イベントが発生した場合はゲームをクリアしたとみなされ，エージェントは初期状態（スタート地点）に戻る．アイテム取得イベントが発生した場合はマップ上からアイテムのシンボル（！）が消失し，初期状態に戻るまで再度のイベント発生が不可能になる．アイテム取得のイベントはゲームクリアに関係しないが，エージェントのゲーム内目標として利用できるものとした．

学習法

エージェントの経路学習には強化学習の 1 種である Q 学習を利用した．Q 関数の値はルックアップテーブル形式で用意し，テーブルの値は 0 以上 1 未満のランダムな値で初期化した．テーブルはゲーム内目標（イベント）毎に別のものを用意し，初期テーブルとして「ゲーム内目標なし」のテーブルを用意した．Q 関数の引数として受け取る観測情報は現在の座標のみとし，イベント情報については目標候補の更新と目標達成の判定のみに利用するものとした．ゲーム内目標が達成されると次の目標のテーブルに切り替え，目標候補を全て達成した場合はゲーム内目標なしのテーブルに切り替えるようにした．

探索手法

エージェントの経路学習時の探索手法には ε-greedy 法を利用した．ランダムな行動の選択率である ε の値はテーブル (ゲーム内目標) 毎に独立した値を使用・更新するようにした．

パラメータ

ゲームプレイは 1 エピソードあたり最大 200 ステップとし，学習 1 回あたり 1000 エピソードのゲームプレイを行った．イベントの認知率は 1（100%）とした．ゲーム内目標達成時の報酬は 100，1 ステップあたりの移動コストは-0.1 とした．Q 関数の学習率 α は 0.3，報酬の割引率 γ は 0.9995 とした．ランダム行動率 ε の初期値は 0.3，1 エピソード辺りの減衰率は 0.99 とした．

5.5.3　手順

L 字形マップ 1 における経路学習

エージェントの初期位置 (初期状態) をスタート地点とし，学習開始時のゲーム内目標は「ゲーム内目標なし」とした．エージェントがいずれかのイベントの発生を初めて認知したとき，そのエピソードの終了時に目標候補に追加し，対応するテーブルを生成した．このとき，目標候補にイベントを追加する順序はゲーム内でイベントを認知した順序と同様にした．目標候補が空でないとき，最も先に追加された未達成のゲーム内目標を次の目標として設定し，対応するテーブルに切り替えて学習を進めた．1 回の行動にかかる時間を 1 ステップとし，200 ステップ経過するか，ゴール到達のイベントを発生させる（ゲームクリアする）までを 1 エピソードとした．各エピソードの開始時にゲームを初期状態に戻し，計 1000 エピソードの経路学習を行った．1000 エピソード終了後，学習したテーブルと目標候補を用いてゲームプレイを行い，認知したイベントとその順序を記録した．以上を 5.5.2 節で述べた最小のマップから最大のマップまで，通路の長さを 1 ずつ増やした全 28 種類のマップを用意し，1 マップあたり 1000 回の学習を行った．

L 字形マップ 2 における経路学習

学習 1 回の手順は L 字形マップ 1 と同様に行った．マップは 5.5.2 節で述べた最小のマップから最大のマップまでスタート地点からアイテム取得イベントまでの距離を 1 ずつ増やした全 19 種類のマップを用意し，1 マップあたり 1000 回の学習を行った．

5.5.4　結果

L 字形マップ 1 および 2 における実験結果を図 5.6 と図 5.7 に示す．図 5.6 の横軸はスタート地点と各イベントとの距離，図 5.7 の横軸はスタート地点とアイテム取得イベントの距離である．各図の縦軸は学習後のゲームプレイでイベントを認知した回数である．GOALED はゴール到達イベントのみを認知した場合のデータ，ITEM GET -> GOALED はアイテム取得イベント → ゴール到達イベントの順に認知した場合のデータである．アイテム取得イベントのみを認知した場合と，いずれのイベントも認知しなかった場合のデータは確認されなかった．

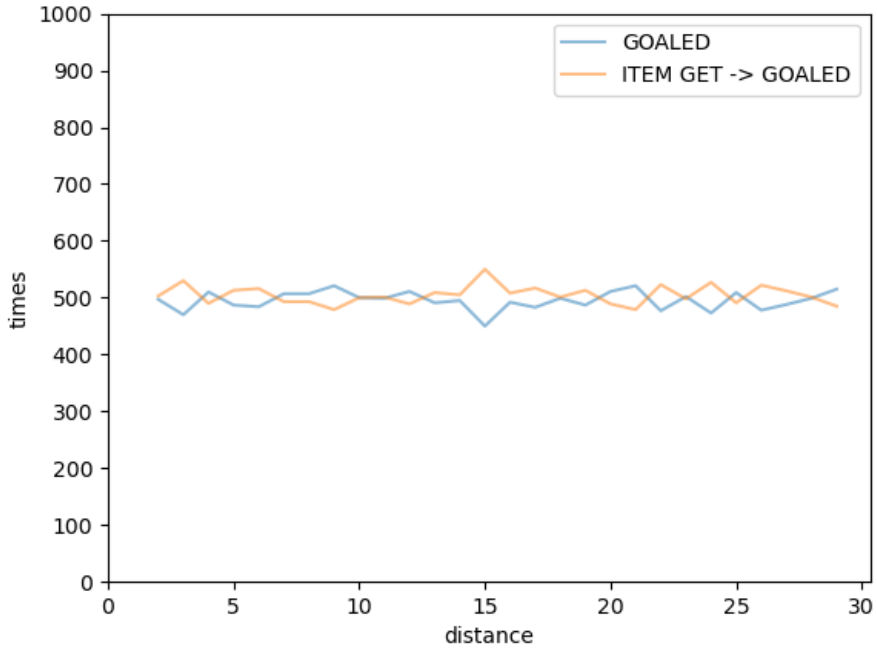


図 5.6: L 字形マップ 1 の実験結果

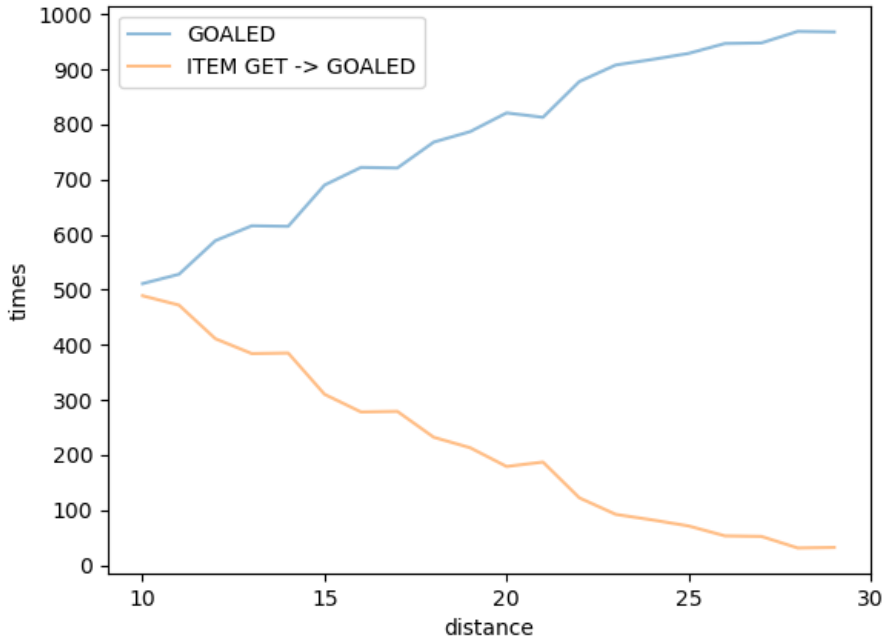


図 5.7: L 字形マップ 2 の実験結果

5.6　認知したイベントを用いたゲーム内目標の設定の効果

図 5.6 と図 5.7 より，エージェントは経路学習を通じてゴール到達イベントか，アイテム取得イベントとゴール到達イベントの両方をゲーム内目標として設定することができた．エージェントの初期目標は「ゲーム内目標なし」であり，ゲーム内で認知したイベントのみをゲーム内目標として設定可能であった．このことから，本実験の目的である，本システムを用いたイベントの認知による動的なゲーム内目標の設定が可能であることを確認できたといえる．

いずれのマップにおいてもアイテム取得イベントのみの認知やいずれのイベントも認知しない場合のデータは確認されなかったが，これはマップの形状が一本道かつそこまで大きくないため，ランダム探索でも容易に全てのイベントを発生させることが出来たためだと考えられる．

図 5.6 ではマップのサイズによらず GOALED と ITEM GET -> GOALED の比率はおおよそ 1: 1 であった．これはスタート地点からの距離の比が同じであればいずれかのイベントに先に到達する確率は同じであったためだと考えられる．プレイヤの行動がランダムである場合，各イベントに到達するまでの平均経過時間は現在地とイベントとの距離から算出される．したがって 2 つのイベントとの距離が等しい場合，ランダム行動で片方のイベントに到達する確率は 1/2 となる．

図 5.7 ではスタート地点とアイテム取得イベントの距離が離れるほどゴール到達イベントのみを経験する回数が増加する傾向が見られた．これはスタート地点に近いイベントの方が先に到達しやすく，マップが大きくなるほどアイテム取得イベントを先に認知する確率が小さくなったためだと考えられる．

5.7　評価実験 2：認知率が目標設定に与える影響の調査

本節では本章で提案するイベント認知に基づく目標獲得システムついて，本システムの実用における識別機の要求精度を確認するた，識別機の認知率がイベントの認知やゲーム内目標の設定にどのような影響を与えるか調査した評価実験とその結果を説明する．

5.7.1　目的と概要

本実験の目的は本章で提案するイベント認知に基づく目標獲得システムについて，5.3.2 節で述べた識別機の認知率が目標設定にどのような影響を与えるか調査し，本システムの実用性を保つために必要な認知率を確認することである．

本実験では評価実験 1 と同様に，ゲーム AI の 2 次元格子状マップにおける経路学習を，本システムおよび Q 学習を用いて行った．イベント認知や学習はマップの形状にも影響を受けると考えられたため，実験では各イベントに到達する経路が 1 本である L 字形マップと図 5.8 のよう

な各イベントに到達する経路が複数存在する正方形マップの 2 種類を用意した．L 字形マップにおけるイベントの設置場所は評価実験 1 と同様であり，正方形マップでは右上の隅にアイテム取得イベント (!) を，左下の隅にゴール地点（G）を設定し，左上の隅をスタート地点とした．学習において，認知率が 0 に近いほど観測したイベント情報を破棄する確率が上がるように設定した．これによって「発生したイベントを見逃す」という識別機の誤認を再現し，認知率によってゲーム内目標の設定と経路学習がどのような影響を受けるか確認した．また，学習後のプレイにおいて認知率をそのままにした場合と 1(100%) に再設定した場合の 2 通りのデータを記録し，イベントを認知できなかった場合に，偶然イベントを見逃したのか，または学習そのものが上手く行かなかったのかを確認した．

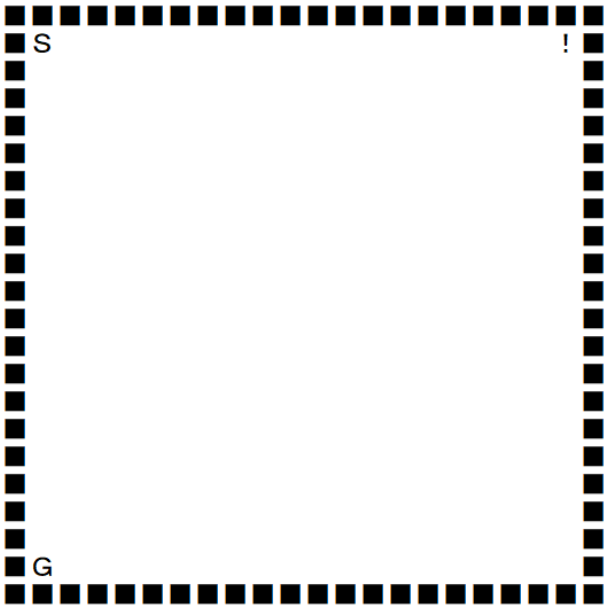


図 5.8: 正方形マップ

5.7.2 認知率の要求精度

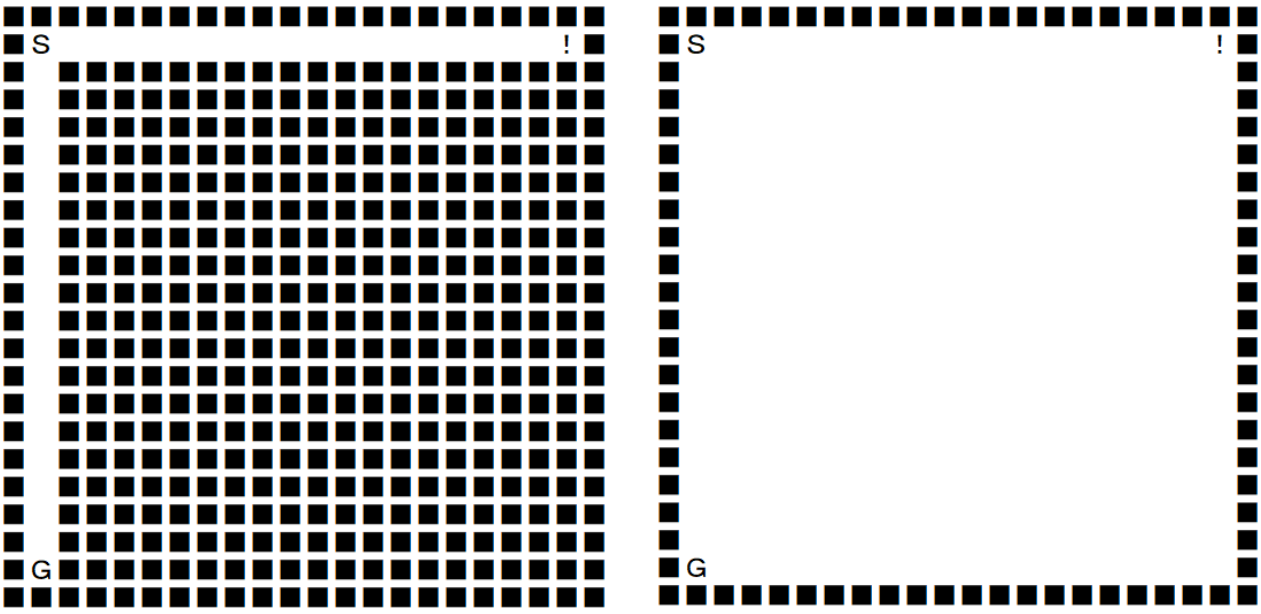
本実験では本システムの実用性を判断する基準として，学習後に認知率を 1 に戻したゲームプレイにおいて NO_EVENT となる確率が 10%未満であれば十分な精度があり，10%以上であれば認知率の精度が足りていないとする基準を設定した．NO_EVENT とは，学習後のゲームプレイにおいていずれのイベントも認知出来なかった場合のデータである．このように認知率の要求精度を決定した理由は，学習失敗を理由とする再学習の回数を 1 回以内に抑えるためである．認知率を 1 に戻したゲームプレイではそのゲームプレイ内でのイベントの見逃しは発生しないため，NO_EVNENT となる要因は主に「学習中に 1 度もイベントが発生しなかった」か「イベント認知の失敗により学習が上手く進まなかった」の 2 つが考えられる．しかし本実験では小さなマップを使用したため，前者が要因となる確率は非常に低いと予想される．よって認知率を 1 に戻したゲームプレイで NO_EVENT となったならば，イベント認知の精度不足に

よって学習が失敗したと考えられるため，再学習の必要が生じる．このとき，NO_EVENT となる確率が 10%未満であったならば，2 度目の学習でも NO_EVENT となる確率は 1%未満であり，3 度目の学習が必要になる確率は十分小さいといえる．一方で NO_EVENT となる確率が 10%以上の場合，2 度目の学習でも NO_EVENT となる確率が 1%以上あり，3 度目の学習が必要になってしまう可能性がある．当然ながら，学習にかかる時間的・計算資源的コストを考えれば再学習は 1 度も行わないことが望ましい．しかしながら，本章では識別機の具体的な設計をしておらず，識別機の精度向上がどの程度困難であるのかを予測することが難しい．そこで本実験では学習失敗による再学習を 1 度までは許容するものとし，その基準にしたがって実験環境における認知率の要求精度を判断した．

5.7.3 準備

実験環境

本実験では図 5.9 の 2 つのマップを使用した．いずれのマップもスタート地点と各イベントとの直線距離は 20 とした．各マップで可能な行動は上下左右の通路への移動とした．環境から得られる観測情報はエージェントの現在の座標とイベントの発生情報とした．



L 字形マップ　　正方形マップ

図 5.9: 評価実験 2 のマップ

イベント

5.5 節の評価実験 1 と同様にした．

学習法

5.5 節の評価実験 1 と同様にした．

探索手法

5.5 節の評価実験 1 と同様にした．

パラメータ

ゲームプレイは 1 エピソードあたり最大 200 ステップとし，学習 1 回あたり 1000 エピソードのゲームプレイを行った．イベントの認知率は最小 0.1，最大 1 とし，0.1 刻みで設定した．ゲーム内目標達成時の報酬は 100，1 ステップ辺りの移動コストは-0.1 とした．Q 関数の学習率は 0.3，報酬の割引率は 0.9995 とした．ランダム行動率 ε の初期値は 0.3，1 エピソード辺りの減衰率は 0.99 とした．

5.7.4 手順

本実験は学習フェーズと評価フェーズの 2 段階の手順で行った．学習フェーズでは 5.5 節の評価実験 1 と同様の手順で計 1000 エピソードの学習を行った．学習フェーズの終了後に評価フェーズへと移行し，認知率をそのままにした場合と 1 に再設定した場合の 2 パターンのゲームプレイを行い，それぞれで認知したイベントとその順序を記録した．以上の学習フェーズおよび評価フェーズを，各マップにおいて認知率を 0.1 から 1 まで 0.1 ずつ変更して各 1000 回ずつ行った．

5.7.5 結果

L 字形マップおよび正方形マップにおける実験結果を図 5.10 と図 5.11 に示す．A のグラフは学習後のゲームプレイにおいて認知率を 1 に戻した場合のデータ，B のグラフは認知率をそのままにした場合のデータである．各図の横軸は認知率であり，縦軸はイベントを認知した回数である．各図における 4 つのデータはそれぞれ，

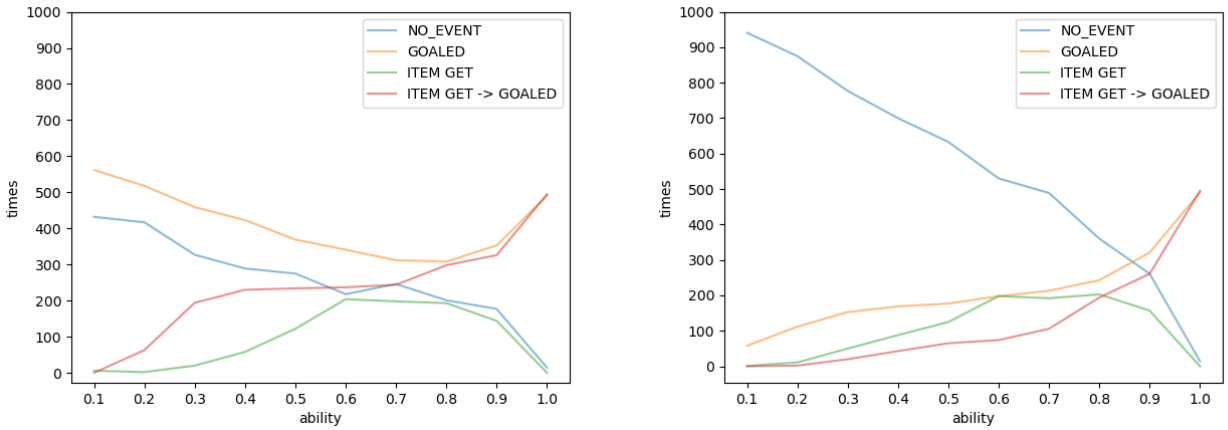
NO_EVENT ： いずれのイベントも認知しなかった場合のデータ

GOALED ： ゴール到達イベントのみ認知した場合のデータ

ITEM GET ： アイテム取得イベントのみを認知した場合のデータ

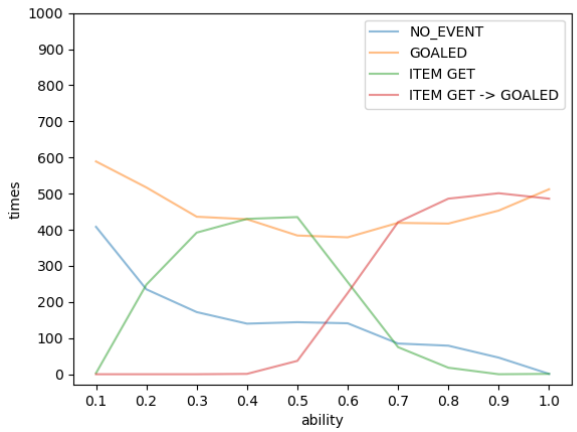
ITEM GET -> GOALED ： アイテム取得イベント → ゴール到達イベントの順に認知した場合のデータ

である．イベントの性質上，ゴール到達イベント → アイテム取得イベントの順で認知した場合のデータは確認されなかった．

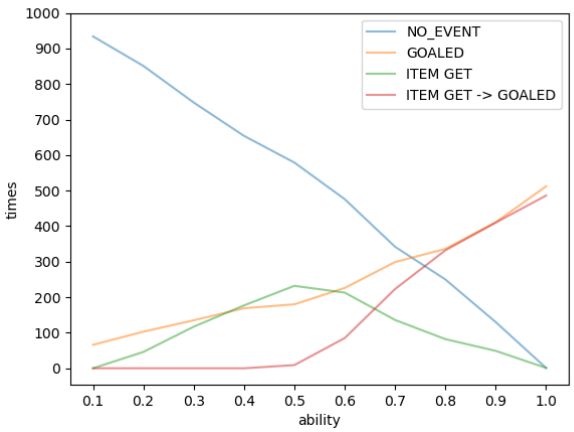


A（認知率を 1 に再設定）　　B（認知率そのまま）

図 5.10: L 字形マップにおける実験結果



A（認知率を 1 に再設定）



B（認知率そのまま）

図 5.11: 正方形マップにおける実験結果

5.8 実験環境における認知率の要求精度

図 5.10 と図 5.11 の A のグラフにおける認知率 0.9 の結果を見ると，L 字形マップでは約 20%の確率で NO_EVENT であったのに対し，正方形マップでは NO_EVENT は 10%未満であった．L 字形マップでは 5.7.2 節の基準を満たしておらず，本システムを L 字形マップで使用する場合は 0.9 を超える認知率が必要であると考えられる．一方で正方形マップでは 5.7.2 節の基準を満たしていることから，本システムを正方形マップで使用する場合は 0.9 程度の認知率で十分であると考えられる．

ただし図 5.11 の B のグラフから，認知率 0.9 の場合，正方形マップにおける実際のゲームプレイでは約 15%の確率で NO_EVENT となる．このため学習に失敗したのか，それとも偶然イベントを認知出来なかったのかを確認するためには数回のテストプレイが必要になると考えられる．このことから，再学習や 1 回のテストプレイにかかるコストが大きい場合には，正方形マップにおいても認知率 0.9 を超える精度が必要になる可能性がある．

図 5.10 と図 5.11 を比較すると，正方形マップでの実験結果の方が NO_EVENT のデータが少ないという傾向が見られた．これは，イベント認知に失敗した場合は経路上のテーブルの値が誤った値で更新されてしまうが，経路が複数存在する場合は経路ひとつ辺りの失敗する確率が小さくなるため，正方形マップのような経路の多いマップではイベント認知の失敗による学習への悪影響を受けにくかったためだと考えられる．また，特に図 5.11 の正方形マップの結果では，ITEM GET のグラフが認知率 0.5 付近で最大となる山なりの形状となった．これは認知率が 0.5 の場合では ITEM GET -> GOALED の確率が $0.5^2 = 0.25$ となるため，アイテム取得イベントを目標とする学習のみ上手く進み，その後のゴール到達イベントへの学習は上手く進まなかったというパターンが最も多くなったためだと考えられる．

5.9　動的目標獲得システムの実装案としての有効性

本システムは第 2 章で提案した動的目標獲得システムの機能を制限した実装案である．提案システムの課題であったシステムの設計の複雑化という問題に対し，本システムではイベントの認知を行う識別機の機能を軸として，目標設定における柔軟性を維持しつつ，よりシンプルな設計でゲーム AI の動的な目標設定を実現した．本システムは識別機以外の機能を可能な限り簡素に設計しており，たとえばイベント記憶機の部分を第 3 章のネットワーク型記憶システムに置き換えるなど，提案システムの実現に向けた改善の余地が多く残されている．一方で，本章では識別機やその事前知識の具体的な設計は行っていないため，これらをどう設計していくかが実用における課題となる．

第6章　結論

6.1　まとめ

本稿ではゲーム AI のゲームプレイにおける目標設定を柔軟かつ動的に行うシステムを提案し，その概要と設計方法について議論した．第 1 章では本研究の目的および背景を説明し，多様なゲーム AI に関する先行研究を紹介した．第 2 章では本研究における目標や目標達成という言葉を定義し，本研究の目的である思考領域・本能・記憶の 3 つのサブシステムからなる動的目標獲得システムを提案した．その後，提案システムの設計におけるシステムの巨大化や複雑化といった課題を説明し，それらを解決するための 3 つの実装案を説明した．第 3 章ではネットワーク記憶型システムの概要と設計方法を説明し，評価実験の結果から提案システムの記憶サブシステムのベースとしての有効性を議論した．結論として，情報とその関係性を記憶しつつ，記憶データ全体をコンパクト化する効果があることが明らかになった．第 4 章では多様な目標の獲得の例としての，最適でない方策のための動的な報酬設定システムの概要と設計方法を説明し，評価実験の結果から本システムが提案システムの課題解決に有効であるかを議論した．

結論として，本システムは提案システムの 3 つのサブシステムの機能を制限報酬による報酬関数の更新というシンプルな手法で代替し，最適でない方策の学習のための動的な目標設定を実現出来た．第 5 章では自立的な報酬設定に必要なイベント認知に基づく動的目標獲得システムの概要と設計方法を説明し，評価実験の結果から本システムが提案システムの課題解決に有効であるかを議論した．結論として，本システムではイベントの認知を行う識別機の機能を軸として，提案システムの目標設定における柔軟性を維持しつつ，よりシンプルな設計でゲーム AI の動的な目標設定を実現出来た．以上より，いずれの実装案も提案システムの課題に対して，データのコンパクト化や設計の複雑さの改善などの有効性があることが明らかになった．

6.2　今後の展望

本研究の今後の展望として，本稿で説明した 3 つの実装案を改善し，提案システムの課題を解決できれば，多様なゲーム AI の研究開発を効率化するより実用性の高いシステムが実装可能になると期待される．特に 3 つ目の実装案であるイベント認知に基づく動的目標獲得システムは，識別機の具体的な設計方法という課題が残っているものの，目標設定の柔軟性や拡張性の高さといった多くの利点が存在する．本システムの課題を改善しつつ提案システムの機能に近づけていくことで，より多様な目標設定に対応可能になると考えられる．また，本研究で提

案するシステムの機能は人間の認知機能をモデルとしているため，提案システムの実現によって，ゲーム環境において人間の目標設定に影響を与えやすい要因の調査など，他の分野への応用も期待される．

参考文献

[1] 池田心. モンテカルロ碁における多様な戦略の演出と形勢の制御: 接待碁 ai に向けて. 2012.

[2] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, Vol. 529, No. 7587, pp. 484–489, 2016.

[3] Stockfish - open source chess engine. `https://stockfishchess.org/`. (Accessed on 01/24/2022).

[4] Sila Temsiririrkkul, Huu Phuc Luong, and Kokolo Ikeda. Production of emotion-based behaviors for a human-like computer player. 2016.

[5] 中川絢太, 佐藤直之, 池田心. ゲームの目的達成のみを追求した ai では生まれにくいゲーム内行動の分類と考察. 2016.

[6] 安西祐一郎 Marvin Minsky. 心の社会. 産業図書株式会社, 1992.

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

[8] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

[9] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *arXiv preprint arXiv:1810.02274*, 2018.

[10] 磯村拓哉. 自由エネルギー原理の解説: 知覚・行動・他者の思考の推論. 日本神経回路学会誌, Vol. 25, No. 3, pp. 71–85, 2018.

[11] 海野良介, 鶴岡慶雅ほか. 離散行動空間における教師なしスキルの獲得手法. ゲームプログラミングワークショップ 2020 論文集, Vol. 2020, pp. 90–97, 2020.

[12] 増山岳人, 山下淳, 淺間一. 変換不変性を用いた経験の抽象化と内発的動機づけに基づく強化学習. 日本機械学会論文集 C 編, Vol. 79, No. 798, pp. 289–303, 2013.

[13] Inseok Oh, Seungeun Rho, Sangbin Moon, Seongho Son, Hyoil Lee, and Jinyun Chung. Creating pro-level ai for a real-time fighting game using deep reinforcement learning. *IEEE Transactions on Games*, 2021.

[14] Jack Harmer, Linus Gisslén, Jorge del Val, Henrik Holst, Joakim Bergdahl, Tom Olsson, Kristoffer Sjöö, and Magnus Nordin. Imitation learning with concurrent actions in 3d games. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. IEEE, 2018.

[15] Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. Replacing rewards with examples: Example-based policy search via recursive classification, 2021.

[16] dahara1. Rce:報酬関数が不要な強化学習(1/2). `https://webbigdata.jp/ai/post-10063`. (Accessed on 01/26/2022).

[17] Rufin VanRullen and Ryota Kanai. Deep learning and the global workspace theory, 2021.

[18] Ralf Hartmut Güting. Graphdb: Modeling and querying graphs in databases. In *VLDB*, Vol. 94, pp. 12–15. Citeseer, 1994.

[19] 砂山渡. Ai による類推思考˜ 知識獲得と知識創発˜. 知能と情報, Vol. 30, No. 4, pp. 212–215, 2018.

[20] 越中康治, 高田淑子, 木下英俊, 安藤明伸, 高橋潔, 田幡憲一, 岡正明, 石澤公明. テキストマイニングによる授業評価アンケートの分析: 共起ネットワークによる自由記述の可視化の試み. 宮城教育大学情報処理センター研究紀要, COMMUE (22), pp. 67–74, 2015.

[21] 山下利之, 清水孝昭, 栗山裕, 橋下友茂. コンピュータゲームの特性と楽しさの分析. 日本教育工学会論文誌, Vol. 28, No. 4, pp. 349–355, 2005.

[22] 大森隆司. 試論: 人はなぜ感情をもつのか—行動決定における感情の計算論的役割—. 人工知能, Vol. 31, No. 5, pp. 710–714, 2016.

[23] 栗原一貴ほか. Toolification of games: 既存ゲームの余剰自由度の中で非ゲーム的目的を達成するゲーミフィケーション周辺概念の提案と検討. 情報処理学会論文誌, Vol. 58, No. 4, pp. 919–931, 2017.

[24] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

[25] 吉本潤一郎, 銅谷賢治, 石井信. 強化学習の基礎理論と応用. 計測と制御, Vol. 44, No. 5, pp. 313–318, 2005.

[26] Ishida So. 迷路自動生成アルゴリズム. `http://www5d.biglobe.ne.jp/stssk/maze/make.html`. (Accessed on 01/24/2022).

[27] Leon O Chua and Tamas Roska. The cnn paradigm. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, Vol. 40, No. 3, pp. 147–156, 1993.

謝辞

本稿の執筆にあたり，指導教員である西野順二助教には研究の方針や論文内容等についての，親身かつ丁寧なご指導を頂きました．深く感謝申し上げます．

学外発表

- 目的外行動を行う AI，第 48 回東海ファジィ研究会，2020
- ゲーム AI のための価値観とプレイサイクルモデル，第 36 回ファジィ システム シンポジウム，2020
- 忘却と想起を行うネットワーク型記憶システムの提案，第 49 回東海ファジィ研究会，2021
- 最適でない方策の学習のための動的な報酬設定の提案，第 37 回ファジィ システム シンポジウム，2021
- イベント認知に基づく動的目標獲得システム，第 51 回東海ファジィ研究会，2022