人工知能学会研究会資料
SIG-AGI-009-05

RGoal Architecture: 再帰的にサブゴールを設定できる階層型強化学習アーキテクチャ

The RGoal Architecture: A Hierarchical Reinforcement Learning Architecture That Can Set Subgoals Recursively

一杉裕志 [1*]　高橋直人 [1]　中田秀基 [1]　佐野崇 [2]
Yuuji Ichisugi[1]　Naoto Takahashi [1]　Hidemoto Nakada[1]　Takashi Sano[2]

[1] 産業技術総合研究所 人工知能研究センター
[1] National Institute of Advanced Industrial Science and Technology (AIST), AIRC
[2] 成蹊大学 理工学部 情報科学科
[2] Department of Computer and Information Science, Faculty of Science and Technology, Seikei University

Abstract: Humans can set suitable subgoals in order to achieve some purposes, and furthermore, can set sub-subgoals recursively if needed. It seems that the depth of the recursion is unlimited. Inspired by this behavior, we have designed a new hierarchical reinforcement learning architecture, the RGoal architecture. The algorithm is designed to solve the MDP on the augmented state-action space. The action-value function becomes shareable among multi-tasks due to the value function decomposition. The sharing accelerates learning in multi-task setting. The mechanism named "think-mode" is a kind of model-based reinforcement learning. It combines learned simple tasks in order to solve inexperienced complicated tasks quickly, or in zero-shot in some cases. The algorithm is realized by a flat table and repetition of simple operations, without a stack. Hereafter, we will extend this architecture, and will build the model of the information processing mechanism of the prefrontal cortex in the brain.

概要

人間は何か目的を達成するために適切なサブゴールを設定できる。さらに必要に応じてそのサブゴールを再帰的に設定することができ、その再帰の深さには制約がないように見える。この振る舞いにヒントを得た階層型強化学習の新しいアーキテクチャとして、RGoal アーキテクチャを提案する。アルゴリズムは、拡張状態行動空間上の MDP を解く形で定式化される。行動価値関数は、価値関数分解により複数のタスク間で共有可能になり、マルチタスク環境での学習を効率化する。「思考モード」における振る舞いは一種のモデルベース強化学習であり、学習済みのタスクを組み合わせることで、一度も経験したことのないタスクを少ない試行錯誤で、場合によってはゼロショットで解くことができる。アルゴリズムはスタックを用いず、フラットなテーブルとシンプルな操作の繰り返しで実現される。今後

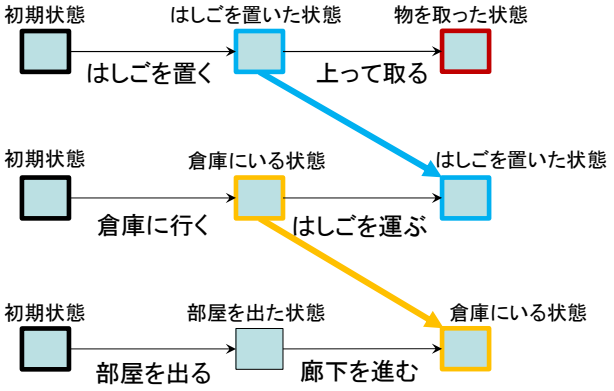

図 1: 人間が再帰的なサブゴールを設定する一例。

このアーキテクチャを拡張し、脳の前頭前野周辺の情報処理機構のモデルを構築する。

*連絡先：産業技術総合研究所
茨城県つくば市梅園 1 − 1 − 1 中央第 1
E-mail: y-ichisugi@aist.go.jp

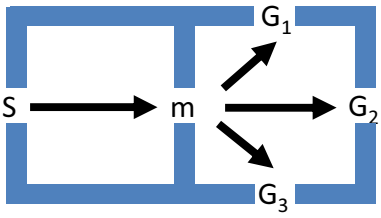

図 2: ゴールが異なっていても、中間地点（サブゴール）までの経路が共通ならば、それを共有することで学習が早く進む。

1 はじめに

人間は何か目的（**ゴール**）を達成しようとする際に、適宜**サブゴール**を設定している [1]。例えば高いところにあるものを取りたいとき、「はしごに上って取る」という方法を思いついたならば、まずはそこにはしごを置く必要がある。つまり、「はしごを置いた状態」がサブゴールになる (図 1)。さらに、サブゴールを達成するために必要であれば、再帰的にサブサブゴールも設定される。例えばはしごが倉庫にあるならば、まず倉庫に行く必要がある。つまり、「自分が倉庫にいる状態」がサブサブゴールになる。人間にとってこのような再帰的なサブゴールの設定の深さには、制限はないように見える。

そもそもサブゴールを設定する利点はなんだろうか。1 つには、マルチタスク環境において、タスクの一部を**サブルーチン**として共有できる点がある。図 2 のようにスタート S からゴール G_i 行くタスクにおいては、ゴールが異なっていても、 S から m への最短経路は共通である。S から m へに向かうというサブタスクの解き方をサブルーチンとして複数のタスク間で共有すれば、学習に必要なパラメタが少なくなり、学習が速くなる。例えば、「はしごを置く（はしごを置いた状態を目指して行動する）」という動作は、高いところのものを取るときだけでなく、電球を取り換えるときなど様々なタスクで共通して利用できる。サブルーチンの共有による性能向上は、**階層型強化学習** [6][7][8][10][11][13][14][15] の目的の 1 つである。

もし再帰的にサブゴールが設定できれば、サブルーチンの共有の機会はより増え、よりメリットが大きいだろう。MAXQ[11] はそれを可能にする多層の階層型強化学習アーキテクチャの 1 つである。MAXQ は、階層的なサブルーチンの学習・実行にスタックを必要とする。しかし人間の脳内に、デジタル計算機のように頑健に動作するスタックがあるとは考えにくい。脳の再帰的なサブゴール設定の機構は、サブゴール達成後、次にすべき動作を必ずしも正確に想起できなくても動作できるように作られているのではないか。

このような背景から、人間の振る舞いにヒントを得て、スタックの機構がなくても動作する、再帰的にサブゴールを設定可能な階層型強化学習のアーキテクチャを設計した。これを *RGoal Architecture* と呼ぶ。以下の章で、このアーキテクチャの詳細を述べていく。

まず 2 章でマルコフ決定過程について簡単に説明した後、3 章で提案アーキテクチャの詳細を説明し、4 章で評価、5 章で議論を行う。6 章では関連研究について述べる。7 章でまとめと今後について述べる。

2 マルコフ決定過程

マルコフ決定過程 *(Markov Decision Process, MDP)* を、状態の集合 $\mathcal{S}$、行動の集合 $\mathcal{A}$、推移確率関数 $P : \mathcal{S} \times \mathcal{A} \to (\mathcal{S} \to [0,1])$、報酬関数 $r : \mathcal{S} \times \mathcal{A} \to \mathbf{R}$ の 4 つ組 $< \mathcal{S}, \mathcal{A}, P, r >$ として定義する。以下に、 2 次元格子上の迷路の場合について具体例を挙げる。状態はエージェントの位置の x 座標と y 座標の組で、行動は上下左右斜めの 8 方向とすると、$\mathcal{S}$ と $\mathcal{A}$ は次のようになる。

$$\begin{aligned} \mathcal{S} &= \{(0,0),(0,1),\ldots\} \\ \mathcal{A} &= \{Up, Down, Right, Left, \\ &\quad UpRight, UpLeft, DownRight, DownLeft\} \end{aligned} \tag{1}$$

ある座標 $s = (10,10)$ において行動 $a = Up$ をとると確率 1 で座標 $s' = (10,11)$ に移動するならば、そのことは推移確率関数 $P(s'|s,a)$ を用いて下記の式で表現される。

$$P(\ (10,11)\ |\ (10,10)\ , Up) = 1 \tag{2}$$

そのとき報酬 -1 が与えられるならば、そのことは報酬関数 $r(s,a)$ を用いて下記の式で表現される。

$$r(\ (10,10)\ , Up) = -1 \tag{3}$$

強化学習の目的は、与えられた MDP のもとで、累積報酬期待値を最大化する方策（行動のルール）を獲得することにある。

3 提案アーキテクチャ

3.1 仮定

我々が提案する階層型強化学習のアーキテクチャは、HDG[8] と MAXQ[11] に強く影響を受けているが、設計にあたっては、脳内の神経回路でも容易に実現でき

るような単純なアーキテクチャになることを目指している。

スタックなしで動作するアーキテクチャを設計するにあたって、いくつかの仮定を置く。

第 1 に、次々に再帰的にサブゴールを設定しても、おおもとのゴール（**グローバルゴール** と呼ぶ）は決して忘れない、もしくはグローバルゴールは外界の状態の一部であり必要に応じていつでも観測することができる、と仮定する。そうすれば、サブゴール達成後、あらためてグローバルゴールを目指して動作を継続することができる。生物におけるグローバルゴールの例は、空腹やのどの渇きのような生理的欲求の解消である。

第 2 に、**ランドマーク**と呼ぶ状態の集合があらかじめ与えられていること、そしてグローバルゴールやサブゴールは必ずそのランドマークのうちのどれかであることを仮定する。ランドマークは、タスクを解く上でのマイルストーン（中間目標）となり得る状態である。ランドマークは生物の場合、模倣学習や何らかのヒューリスティックスで獲得されるものと想定する。ヒューリスティックスとしては、例えば、顕著な刺激や大きな TD 誤差が発生した瞬間の状態を記憶し、ランドマークとすることが考えられる。

提案アーキテクチャでは、タスク間で共有されるサブルーチンを、「任意の状態からある 1 つの状態（サブゴール）に向かう方策」と定義する。Options [10] や MAXQ[11] 等では 1 つのサブルーチン (option) が終了した時の状態が一意に決まらないが、提案アーキテクチャでは、H-DYNA[6][7] や HDG[8] と同様に、1 つに決まる。この性質によりアーキテクチャを大幅に単純化できる上、3.7 章で述べる思考モードが実現可能になる。サブルーチンの終了状態がたった 1 つの状態しかなければタスク間での共有の機会が著しく落ちると思われるかもしれないが、将来は注意の機構などを取り入れることでサブゴールの状態を抽象化し、汎用性を持たせることを計画している。

なお、提案アーキテクチャでは、ランドマークの与え方は性能に大きな影響を与えはするものの、どんな与え方をしてもタスクが解けなくなることはない。ランドマークが 1 つしか与えられない場合は通常のフラットな強化学習と等価である。ランドマークを多く与えすぎた場合は、利用価値のないランドマークは学習が進むにつれ単に使われなくなっていく。

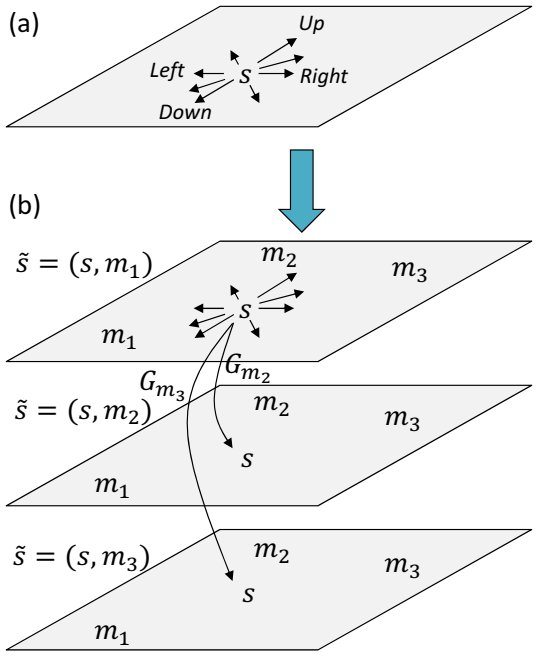

図 3: 拡張状態行動空間。(a) オリジナルの状態空間。ここでは 2 次元平面としている。(b) ランドマークの集合 $\mathcal{M} = \{m_1, m_2, m_3\}$ が与えられたときの、拡張された状態空間。拡張された状態は、オリジナルの状態 s とサブゴール m_i のペア (s, m_i) で表現される。行動は、もともとの 2 次元平面内の移動に加え、サブゴールを切り替える行動 $\mathcal{G}_{\mathcal{M}} = \{G_{m_1}, G_{m_2}, G_{m_3}, \}$ の中の 1 つが選択可能になる。各ランドマークの間に上下関係はなく、相互再帰的にいつでも他のランドマークをサブゴールに設定することができる。

3.2 拡張状態行動空間

ある MDP $< \mathcal{S}, \mathcal{A}, P, r >$ とランドマークの集合 $\mathcal{M} \subseteq \mathcal{S}$ が与えられたとき、**拡張状態行動空間** [14] の上での MDP $< \tilde{\mathcal{S}}, \tilde{\mathcal{A}}, \tilde{P}, \tilde{r} >$ を以下に定義する。まず、拡張された状態 $\tilde{\mathcal{S}}$ と行動 $\tilde{\mathcal{A}}$ の集合を下記のように定

義する。

$$\begin{aligned}\tilde{\mathcal{S}} &= \mathcal{S}\times\mathcal{M}\\ \tilde{\mathcal{A}} &= \mathcal{A}\cup\mathcal{G}_{\mathcal{M}}\\ \mathcal{M} &= \{m_1,m_2,\cdots\}\subseteq\mathcal{S}\\ \mathcal{G}_{\mathcal{M}} &= \{G_{m_1},G_{m_2},\cdots\}\end{aligned}\quad(4)$$

状態 $\tilde{s}=(s,g)\in\tilde{\mathcal{S}}$ は、オリジナルの状態 s とサブゴール $g\in\mathcal{M}$ の組である。$G_m\in\mathcal{G}_{\mathcal{M}}\subset\tilde{\mathcal{A}}$ は、ランドマークの 1 つ m を新たなサブゴールに設定する行動である。この行動をとることで状態 (s,g) は (s,m) に変化する。拡張された推移確率関数 $\tilde{P}(\tilde{s}'|\tilde{s},\tilde{a})$ は、オリジナルの推移確率関数 P を使って以下のように定義される。

$$\begin{aligned}\tilde{P}((s',g)|(s,g),a) &= P(s'|s,a)\\ \tilde{P}((s,m)|(s,g),G_m) &= 1\end{aligned}\quad(5)$$

拡張された報酬関数 $\tilde{r}$ は以下のように定義される。

$$\begin{aligned}\tilde{r}((s,g),a) &= r(s,a)\\ \tilde{r}((s,g),G_m) &= R^{\mathcal{G}}\end{aligned}\quad(6)$$

定数 $R^{\mathcal{G}}$ はサブゴール切り替えのコストを表すハイパパラメタである。

拡張状態行動空間を図 3 に示す。オリジナルの状態空間が 2 次元マップだったとすると、ランドマークを n 個与えた拡張状態空間は n 階建ての 2 次元マップになる。エージェントは 1 回の行動で、フロア内を移動するか、フロア内での座標を変えずに別のフロアに移動するかのいずれかを行う。

提案するアーキテクチャは、この拡張状態行動空間上の MDP を解くアルゴリズムとして設計される。拡張状態行動空間は、本来エージェントの内的状態であるサブゴール g を外界の状態の一部と見なしている。しかし数学的構造は通常と MDP と同じなので、MDP を前提とした様々な理論的帰結（例えば厳密解への収束性）や強化学習の高速化技術（例えば関数近似や適格度トレース）が利用可能である。

拡張状態行動空間上では、サブゴール g に到達していなくても、サブゴールを他のランドマークに切り替えることが許されている。つまり、サブルーチンを実行途中であっても、よりよいサブルーチンがあればそれに実行を切り替えられる。この方が、終了するまでサブルーチンを抜けられないアーキテクチャと比べてより柔軟に行動できる。同様の動作は HDG[8] で実装されており、Options [10] では option の中断として、MAXQ[11] では非階層的実行と呼ぶ形で取り入れられている。

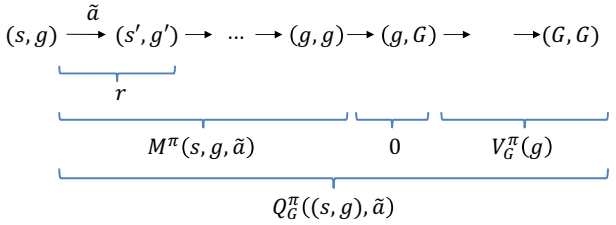

図 4: 提案アーキテクチャにおける価値関数分解。状態 s からサブゴール g を経由してグローバルゴール G に到達した場合における、各区間における報酬の総和の期待値。M^π の値はグローバルゴール G に依存しないため、様々なタスク間で共有することができる。また、V_G^π の値は M^π から効率的に計算できる。（詳細は本文参照。）

3.3 価値関数分解

拡張状態行動空間の上で**価値関数分解** [11] を行うことで、行動価値関数の一部をタスク間で共有し、学習速度を上げることができる。提案アーキテクチャにおける価値関数分解は MAXQ[11] の方法よりも、 H-DYNA[6][7] や HDG[8] で行われている方法に近い。以下に具体的に説明する。

まず方策 $\pi:\tilde{\mathcal{S}}\times\tilde{\mathcal{A}}\to[0,1]$ とグローバルゴール $G\in\mathcal{M}$ が与えられたときの行動価値関数 Q_G^π を以下のように定義する。

$$Q_G^\pi((s,g),\tilde{a})=E_G^\pi[\Sigma_{t=0}^{\infty}r_{t+1}|\tilde{s}_0=(s,g),\tilde{a}_0=\tilde{a}]\quad(7)$$

この式は、初期状態 (s,g) において行動 $\tilde{a}$ を取った後、方策 π に従って行動し続けたときに得られる報酬の列 $r_1=\tilde{r}(\tilde{s}_0,\tilde{a}_0),r_2=\tilde{r}(\tilde{s}_1,\tilde{a}_1),\cdots$ の総和の期待値である。なお、状態 s がグローバルゴール G に到着した時刻以降は報酬 r_t の値は 0 とする。また、本稿では報酬割引は行わないものとする。

方策 π に従って行動することで 状態 (s,g) からサブゴール (g,g) に、さらに (g,G) からグローバルゴール (G,G) に必ず到着できると仮定する。また、 (g,g) に到着した直後の状態は (g,G) に強制的に切り替わり、その際の報酬は 0 と仮定する。そのとき、図 4 から明らかなように、Q_G^π は以下のように、g への到着前と到着後に分解することができる。

$$Q_G^\pi((s,g),\tilde{a})=M^\pi(s,g,\tilde{a})+V_G^\pi(g)\quad(8)$$

ここで、 $M^\pi(s,g,\tilde{a})$ は状態 s において行動 $\tilde{a}$ を取った後、方策 π に従って行動しサブゴール g に到着するまでの報酬の総和の期待値である。また、 $V_G^\pi(g)$ は、サブゴール g に到着した直後の状態 $\tilde{s}=(g,G)$ から方策 π に従って行動しグローバルゴール G に到着する

までの報酬の総和の期待値で、

$$\begin{aligned} V_G^\pi(g) &= \Sigma_{\tilde{a}}\pi((g,G),\tilde{a})Q_G^\pi((g,G),\tilde{a}) \\ &= \Sigma_{\tilde{a}}\pi((g,G),\tilde{a})(M^\pi(g,G,\tilde{a})+V_G^\pi(G)) \\ &= \Sigma_{\tilde{a}}\pi((g,G),\tilde{a})M^\pi(g,G,\tilde{a}) \end{aligned} \quad (9)$$

である。($V_G^\pi(G)=0$ である点に注意。)

このように、$V_G^\pi(g)$ の値は $M^\pi(s,g,\tilde{a})$ から効率的に計算できる。今回の実装では $M^\pi(s,g,\tilde{a})$ はテーブルとして保持し、3.5 章で述べる学習則により学習する。

3.4 行動選択

現在の $Q((s,g),\tilde{a})$ の値を用いてグリーディーに行動を選択する場合は以下のようにする。

$$\begin{aligned} \tilde{a}' &= \operatorname*{argmax}_{\tilde{a}} Q((s,g),\tilde{a}) \\ &= \operatorname*{argmax}_{\tilde{a}} (M(s,g,\tilde{a})+V_G(g)) \\ &= \operatorname*{argmax}_{\tilde{a}} M(s,g,\tilde{a}) \end{aligned} \quad (10)$$

今回の実装では以下の式で定義される softmax での行動選択を行っている。

$$\pi((s,g),\tilde{a}) = \frac{exp(\beta M(s,g,\tilde{a}))}{\Sigma_{\tilde{a}'} exp(\beta M(s,g,\tilde{a}'))} \quad (11)$$

3.5 学習

行動価値テーブル M の値は、通常の強化学習アルゴリズムと同様の方法で学習することが可能である。例えば Sarsa で学習する場合の Q の更新式は以下のようになる。

$$Q(\tilde{s},\tilde{a}) \leftarrow Q(\tilde{s},\tilde{a}) + \alpha(r + Q(\tilde{s}',\tilde{a}') - Q(\tilde{s},\tilde{a})) \quad (12)$$

この更新式と式 (8) から次の更新式を容易に導くことができる[1]。

$$\begin{aligned} &M(s,g,\tilde{a}) \leftarrow M(s,g,\tilde{a}) \\ &+\alpha(r + M(s',g',\tilde{a}') - M(s,g,\tilde{a}) + V_G(g') - V_G(g)) \end{aligned} \quad (13)$$

ただし、$s=g$ の場合、すなわち状態 s がサブゴール g に到達した場合には特別な扱いが必要である。定義により $s=g$ のときは必ず $M(s,g,\tilde{a})=0$ でなければならないため、学習時には $s=g$ の場合だけはその値を更新しないように実装する。

[1]なお、$g'=g$ のときに $V_G(g')-V_G(g)=0$ の計算を省くことで計算時間を少し短くできる。

3.6 行動価値テーブルの初期化

定義により $s=g$ のときは必ず $M(s,g,\tilde{a})=0$ なので、そのように初期化しておく。

それ以外の場合、つまり $s\neq g$ については初期値は理論上は任意の値でよい。しかし 4 章でも示すように、初期値は実際の性能に大きく影響する。一般に、解こうとする問題の特性についての事前知識を行動価値テーブルの初期値として与えることで性能を上げることができる [12]。

提案アーキテクチャにおいては、事前知識を用いた初期値設定の極端な場合として、$-\infty$ を設定することが特に有効である。ランドマークを切り替える行動 G_m が任意の状態 s において選択可能だとすると、学習すべき行動価値テーブルのサイズが大きくなり、性能低下の原因となる。そこで、行動 G_m を選択可能なのは、状態 s がランドマーク上にあるときのみであるように制限したいとする。そのためには、$s\notin\mathcal{M}$ に対して

$$M(s,g,G_m) = -\infty \quad (14)$$

と設定すればよい。初期値に $-\infty$ を設定された行動は決して選ばれることはないため、探索空間が狭くなる。なお、選ばれない行動は 3.5 章で述べた学習にも関わらないため、$-\infty$ の値が他のテーブルの要素に伝搬していくことはない[2]。

必要ならば適切な m_1,m_2,m_3 に対してさらに $M(m_1,m_2,G_{m_3})=-\infty$ を設定することでどのサブゴールがどのサブゴールを呼び出せるかを制限し、探索空間を減らすことができる。制約をしすぎると性能がかえって悪くなることも起こり得るが、理論上はもともとの MDP が解けなくなることはない。この方法を用いて MAXQ[11] のタスクグラフ似た制約を設計者が与えることができる。設計者が与えなくても、そのような制約を模倣や言語活動を通じて獲得するような機構も考え得る。

3.7 思考モード

階層型強化学習には、1 章で述べたサブルーチンの共有という目的とは別の目的もある。それは、学習済みの簡単なタスクを組み合わせて複雑なタスクを近似的に、しかし高速に解くという目的である。しかし、ここまで述べた機構にはその機能はない。そこでアルゴリズムに**思考モード**と呼ぶもの導入する。

[2]実装上の注意点をいくつか述べておく。ε グリーディーで行動選択をする場合は、$-\infty$ の値を必ず避けるような実装になっている必要がある。 softmax の場合は価値が $-\infty$ の行動は自然に排除される。いずれの場合も式 (9) の値を計算する時、$0\times-\infty=NaN$ が発生しないように注意し、$0\times-\infty=0$ であるかのように実装する必要がある。

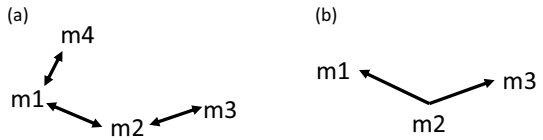

図 5: (a) 隣接するランドマーク間の最適移動経路は学習済みだとする。(b) 離れたランドマーク間の最適移動経路の近似解は、隣接するランドマークをつなぐことで得られる。

思考モードが解こうとする問題を図 5 を使って説明する。今、隣接するランドマーク間の最適移動経路は学習済みだとする。このとき、離れたランドマーク間の最適移動経路の近似解は、隣接するランドマークをつなげば得られるはずである。都合のよいことに、この近似解は実際に行動しなくても「エージェントの脳内シミュレーション」だけで高速に見つけられる [6][7][8]。しかも提案アーキテクチャにおいては、この機能は、アルゴリズムにわずかに修正を加えるだけで実現できる。

思考モードでは、選択された行動 $\tilde{a}$ が G_m でない場合 (サブゴール切り替えでない場合)、s は 1 ステップでサブゴール g まで飛ぶことができる。その場合の報酬は s と g の間の報酬の総和の期待値 $r = M(s, g, \tilde{a})$ とする。このとき $M(s, g, \tilde{a})$ は更新しないようにする。$\tilde{a} = G_m$ の場合は通常通りサブゴールを切り替え、$M(s, g, G_m)$ も通常通り式 (13) を用いて更新する。以上の振る舞いは、学習済みの $M(s, g, \tilde{a})$ を環境のモデルと見なした一種のモデルベース強化学習 [5][9] である。

思考モードでの実行により、未経験の $M(s, g, G_m)$ の値を脳内で学習できる。これにより、離れたランドマーク間を適切につなぐサブゴールはどれなのかを学習できる。例えば図 5 の m_1 から m_3 への最適経路の近似解を求めるには、スタート $S = m_1$、ゴール $G = m_3$ として思考モードによるエピソードを繰り返せばよい。学習の結果、$M(m_1, m_3, G_{m_2})$ が他の $M(m_1, m_3, \tilde{a})$ よりも大きな値であれば、 m_3 に向かうためにまず m_2 がサブゴールとして選択されることになる。

なお、必要ならば近似解ではなく厳密解を獲得することも可能である。思考モードではなく通常モードで、$S = m_1, G = m_3$ として実際の経験を繰り返せば、m_2 を必ずしも経由しない厳密な最適経路がやがて学習される。

3.8 アルゴリズム

以上の結果をまとめた、Sarsa に基づくアルゴリズムの疑似コードを図 6 に示す。

このように、アルゴリズムはスタックを用いず、フラットなテーブル M と効率的でシンプルな操作の繰

```
1: procedure EPISODE(S, G, think-flag)
2:     s ← S; g ← G
3:     Choose ã from s, g using policy derived from M
4:     loop
5:         # Take action.
6:         if ã is G_m then
7:             s' ← s; g' ← m; r ← R^G
8:         else
9:             if think-flag then
10:                s' ← g; g' ← g; r ← M(s, g, ã)
11:            else
12:                Take action ã, observe r, s'
13:                g' ← g
14:        # Choose action.
15:        if s' = g' then
16:            ã' ← G_G
17:        else
18:            Choose ã' from s', g' using policy derived from M
19:        # Learn.
20:        if s = g or (think-flag and ã is not G_m) then
21:            # Do nothing.
22:        else
23:            M(s, g, ã) ← M(s, g, ã) + α(r + M(s', g', ã') − M(s, g, ã) + V_G(g') − V_G(g))
24:        s ← s'; g ← g'; ã ← ã'
25:        if s = G then
26:            return
```

図 6: 1 つのエピソードを実行するアルゴリズムの疑似コード。テーブル M はあらかじめ 3.6 章で述べたように初期化されているものとする。

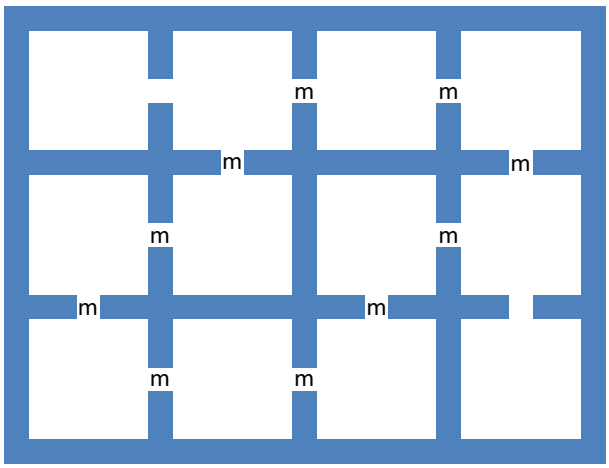

図 7: 評価に用いた 2 次元格子上の迷路のマップ。ランドマーク (m で示した) は部屋をつなぐ通路上の 1 0 か所に設定した。このマップ上で、エピソードごとに異なるスタート S とゴール G が与えられる。ゴールは必ずランドマーク上に設定される。

り返しで実現される。また、思考モードは環境のモデルを別途用意することなしに、アルゴリズムにわずかな修正を加えるだけで実現されている。

4 評価

今回はアルゴリズムの基本動作の確認を行うことが目的のため、実行中の振る舞いの可視化が容易な迷路タスクを題材として性能を評価した。

マップとランドマークの集合は固定である（図 7）。このマップ上で、エピソードごとに異なるスタート S とゴール G が与えられる。エージェントが S から移動して G に到達したときに与えられる報酬は 0 で、その時点でそのエピソードを終了し、スタートとゴールを変えて次のエピソードを始める。上下左右の移動は -1 、斜めの 4 方向いずれかへの移動は $-\sqrt{2}$ 、壁への衝突は -1 、サブゴール切り替え G_m の実行は $R^{\mathcal{G}} = -1$ の報酬が与えられる。（前に述べたように報酬割引はない。）

このタスクを生物の振る舞いのモデルとして解釈するならば、マップは環境のモデル $P(s', r|s, a)$ を表していて、すべてのエピソードを通じて不変である。生物の脳には「空腹が解消された状態」「のどの渇きが解消された状態」というふうに緊急に向かうべきゴールが身体によって次々に提示され、生物は適切に行動することで提示されたゴールを達成しようとする。

テーブルの初期値は、3.6 章で述べたとおりで、サブゴール切り替えはランドマーク上でのみ可能とした。

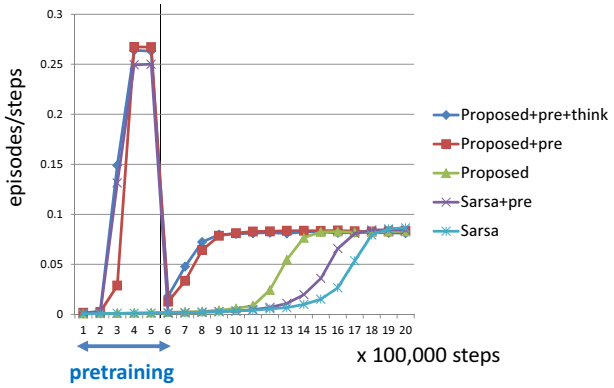

図 8: 実験 1 の結果。実験したいずれの条件においても、提案アーキテクチャの収束速度が Sarsa を上回っている。pretraining フェーズにおけるスコアが高いのは、スタートとゴールの距離が短いためである。

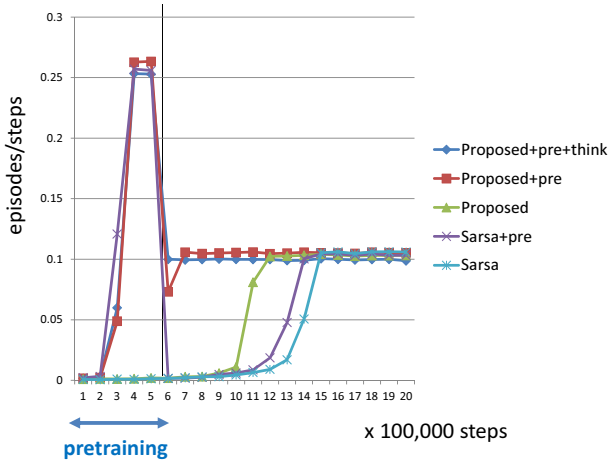

図 9: 実験 2 の結果。実験 1 とほぼ同じだが、スタート位置もランドマーク上から選ぶ。思考フェーズを実行した場合、未経験のタスクに対してもゼロショットでほぼ最適な行動が実行できている。

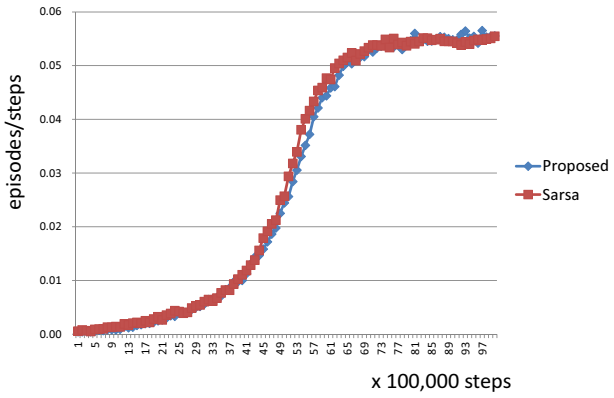

図 10: 実験 3 の結果。テーブル M の要素の初期値を 0 にした場合、 2 つのアルゴリズムのスコアに差はあまりない。

$s = g$ 以外の $M(s, g, \tilde{a})$ に対しては、$-50 - n$ (n は小さなノイズ) に初期化した。

行動選択は softmax を用い、逆温度 $\beta = 1$ とした。学習率は $\alpha = 0.1$ である。

思考モードは 3.7 章で述べたように、現在の状態とグローバルゴールの間の経路の探索に随時用いることを想定しているが、今回の実験では以下に述べるように 1 つのフェーズにまとめて実行することとした。

実験 1 は以下の 3 つのフェーズから構成される。

1. pretraining フェーズ：隣接するランドマークのみを S と G とするエピソードを５０万ステップ分実行。（S と G の組み合わせは 20 通り。）
2. 思考フェーズ：ランドマーク上のランダムな S と G を設定したエピソードを思考モードで 1 万ステップ分実行。（S と G の組み合わせは 100 通り。）
3. 評価フェーズ：任意の場所 S とランドマーク上の G をランダムに設定したエピソードを１５０万ステップ実行。

以上の条件で、提案アルゴリズムの pretraining フェーズと思考フェーズあり、pretraining フェーズのみあり、両方なし、Sarsa の pretraining あり、なしの 5 つのケースについて計測した。

実験 1 の結果のグラフを図 8 に示す。横軸はステップ数であり、縦軸はステップ数あたりのエピソード数である。ここでステップ数とはマップ内の移動もしくは壁への衝突の回数であり、サブゴール切り替え G_m の実行回数は含まれない。実際の行動の経験時間と性能の関係を公平に比較するために、pretraining フェーズがある場合はそのステップ数は横軸に含めた。思考フェーズは実際の行動ではないので横軸に含めていない。

提案アーキテクチャは、 pretraining フェーズと思考フェーズのいずれも行わない場合でも、 Sarsa に比べて早く収束している。pretraining フェーズは提案アルゴリズムの収束をさらに速くしている。この実験では思考フェーズの効果はほとんどない。これは、スタート地点 S がランドマーク上にないため、S から最初のランドマークに到達するまでの経路探索に時間がかかるためと思われる。

実験 2 は実験 1 とほぼ同じだが、評価フェーズのエピソードにおいても、スタート S をランドマーク上のみから選ぶというものである。思考フェーズでシミュレーションしたものと同じタスクを、思考フェーズ終了後に実際に行動して試すことになる。結果のグラフを図 9 に示す。思考フェーズ終了後、ゼロショットで、すなわち未経験のタスクに対して実際の環境での試行錯誤はまったくせずに、ほぼ最適な行動が実行できている。

実験 3 は、実験 1 と同じ条件だがテーブル M の要素の初期値を 0 に設定した。この場合、提案アルゴリズムと Sarsa は性能にほとんど差が出なくなる（図 10）。グラフには示していないが、 pretraining フェーズや思考フェーズを追加しても性能はほとんど変わらなかった。テーブルが楽観的に初期化されている場合、すでに近道を見つけていてもマップの隅々まで他の経路を探索しようとするため、部分的な経路の共有が行われず、提案アーキテクチャと Sarsa であまり違いが出ないことが、理由として考えられる。

5 議論

提案アーキテクチャでは行動価値テーブル $M(s, g, \tilde{a})$ のサイズが従来の行動価値テーブル $Q(s, a)$ よりも大きくなるが、実際にアクセスされる状態行動対はごく一部である。ニューラルネットワークを用いて関数近似したり、テーブルをハッシュ表を用いて実現するなどの工夫により、テーブルサイズの増大は実質的に問題にならないであろうと考えている。生物の脳では、もしテーブルを大脳皮質や海馬の連想記憶機構を使って実現しているならば、同様にテーブルのサイズは問題にならないであろう。

脳との関係で興味深い点として、サブゴールを表す変数 g への参照と書き込みが、人間が思考するときのワーキングメモリへの読み書きに類似しているように見えることを指摘しておく。変数 g を 1 つではなく複数（例えば 1 万個以上）の変数に拡張し、任意の変数への参照と書き込みができる機構を実現すれば、提案アーキテクチャはコンピューターの振る舞いと似てくる上、人間の思考にもより近づくだろう。

今回の提案アーキテクチャでは、エージェントはサブゴール達成後、あらためてグローバルゴールを目指すこととしたが、人間の場合は、グローバルゴールではなく次に達成すべきサブゴールを「思い出す」ことがある。これは心理学で展望記憶 [2] と呼ばれている。展望記憶の機構をアーキテクチャに加え、その有効性について分析することも今後の課題である。

6 関連研究

本研究の最大の貢献は、先行研究ですでに提案されているいくつかの重要なアイデアを単一のシンプルなアーキテクチャに統合し、理論的基盤を整理した上、さらなる拡張を容易にした点にある。

もう 1 つの貢献は、再帰的なサブゴール設定を可能にしつつ、スタックの不要なアーキテクチャが可能であることを示した点である。従来の階層型強化学習では、 n 個の階層があれば実行のコンテキストを記憶す

るための $n-1$ 個のメモリを必要とする。提案アーキテクチャは現在のサブゴールとグローバルゴールのみを記憶し、サブサブゴールが設定されればその前のサブゴールは捨てられる。

我々の提案アーキテクチャは HDG(Hierarchical Distance to Goal)[8] と本質的にかなり似た構造を持っている。HDG は離れたランドマークをつなぐ経路はオフラインで探索するが、そのためには専用のアルゴリズムを用いている。我々のアーキテクチャでは同じことを MDP の枠組み内でモデルベース強化学習の考え方を使った思考モードで実現しており、オンデマンドのプランニングが行いやすくなっている。

MAXQ[11] は 2 層よりも深い層を扱える階層型強化学習のアーキテクチャであり、価値関数分解によりサブルーチンの学習結果の共有を図っている。これは本論文で述べた価値関数分解と考え方は近いが、我々のアーキテクチャでは終了条件が必ず単一の状態（サブゴール）であると仮定することで分解結果は単純になり、価値の計算がしやすくなっている。MAXQ の学習・実行にはスタックが必要だが、我々のアーキテクチャはスタックが不要で、シンプルである。MAXQ アーキテクチャの設計の動機の 1 つに、状態を抽象化する機会を増やすことで探索空間を大幅に狭くすることがある。本論文では状態の抽象化は行っていないが、大脳皮質の機能を模した別の機構 [3][4] で実現することを検討中である。

思考モードのような時間を抽象化した高速なプランニングは H-DYNA[6][7] においてすでに実現されており、その際に本稿で述べた価値関数分解と同じものを用いている。

R-MAXQ[13] は MAXQ アーキテクチャにモデルベース強化学習の機能を加えたものである。しかし、環境のモデルを素直に学習・利用するものであり、時間を抽象化してプランニングを高速化する機構はない。

Option-Critic アーキテクチャ[15] は、2 層の階層構造をしており、経験のみからの end-to-end での option（サブルーチン）の獲得を実現しているが、我々のアーキテクチャでは、ランドマーク獲得は将来の課題である。我々のアーキテクチャの思考モードは 2 層構造と見なすこともでき、Option-Critic アーキテクチャの用語を使うならば、ランドマークは option 、隣接ランドマーク間の移動経路は intra-option-policy 、離れたランドマーク間の移動経路は policy over options にほぼ相当する。しかし、 Option-Critic には思考モードに相当するものはなく、policy over options も実際の経験で学習している。

7 まとめと今後

再帰的にサブゴールを設定可能な RGoal アーキテクチャを提案した。本研究の最大の貢献は、階層型強化学習の先行研究で実現されている重要なアイデアのいくつかを単一のシンプルなアーキテクチャに統合し、理論的基盤を整理した上、さらなる拡張を容易にした点にある。

アルゴリズムはスタックを用いず、フラットなテーブルとシンプルな操作の繰り返しで実現される。思考モードを用いて学習済みのタスクを組み合わせることで、一度も経験したことのないタスクを少ない試行錯誤で、場合によってはゼロショットで解くことができる。

本研究の動機の 1 つに、汎用人工知能の実現に向けた、脳の前頭前野周辺の情報処理機構のモデルの構築がある。本稿でこれまで述べてきたように、アーキテクチャの設計において、人間や動物の振る舞いとの類似性を強く念頭に置いている。また、前頭前野周辺の神経科学的知見を暗に設計の指導原理として用いている。提案アーキテクチャは、再帰的サブゴール設定、時間を抽象化したプランニングという、人間の知能の 2 つの重要な特性を再現できている。今後アーキテクチャを拡張し、単一化を用いた記号推論機構、外界の状態の抽象化機構などを実現し、より人間に近い思考の実現を目指していく。そうしてできたアーキテクチャは、脳を模倣した汎用人工知能を実現するための中核技術になるだろう。

謝辞

ディー・エヌ・エー 甲野佑氏、東京電機大 高橋達二氏との議論から研究の示唆をいただいており、深く感謝いたします。

本研究は JSPS 科研費 JP18K11488 の助成を受けたものです。

参考文献

[1] 高田司郎, 新出尚之, 意図に基づくエージェントアーキテクチャ(<特集>意図研究のスペクトル) 人工知能学会誌/Journal of Japanese Society for Artificial Intelligence,20(4),433-440 (2005-07-01) , KJ00003364545.

[2] 奥田次郎, 意図とその遅延後の実現 : Prospective Memory の脳内過程 (<特集>意図研究のスペクトル), 人工知能学会誌/Journal of Japanese Society for Artificial Intelligence,20(4),418-424 (2005-07-01) , KJ00003364543.

[3] Yuuji ICHISUGI, The Cerebral Cortex Model that Self-Organizes Conditional Probability Tables and Executes Belief Propagation, In Proc. of IJCNN 2007, pp.1065–1070, Aug 2007.

[4] 一杉裕志, 疑似ベイジアンネットを用いた認知モデルのプロトタイピング手法の提案, 第 4 回 人工知能学会 汎用人工知能研究会 (SIG-AGI), 2016.

[5] Sutton, R. S., Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In Proceedings of the Seventh International Conference on Machine Learning, 1990.

[6] Singh, Satinder Pal, Reinforcement learning with a hierarchy of abstract models. In Proceedings of the Tenth National Conference on Artificial Intelligence, San Jose, California. AAAI Press. 202207, 1992.

[7] Singh, Satinder Pal, Scaling reinforcement learning algorithms by learning variable temporal resolution models. In Proceedings of the Ninth International Conference on Machine Learning, Aberdeen, Scotland. Morgan Kaufmann. 406415, 1992.

[8] Kaelbling, L.P.: Hierarchical Learning in Stochastic Domains: Preliminary Results. In: Proceedings of the 10th International Conference on Machine Learning, pp. 167173. Morgan Kaufmann, San Francisco, CA, 1993.

[9] Sutton, Richard S.; Barto, Andrew G., Reinforcement Learning: An Introduction. MIT Press, 1998.

[10] Sutton, R. S.; Precup, D.; and Singh, S. P., Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence 112(1-2):181211, 1999.

[11] Thomas G. Dietterich, Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition, Journal of Artificial Intelligence Research 13, 227-303, 2000.

[12] Wiewiora, E., Potential-based shaping and Q-value initialization are equivalent, Journal of Artificial Intelligence Research 19, 205-208, 2003.

[13] N. Jong and P. Stone. Hierarchical model-based reinforcement learning: R-Max + MAXQ. In Proc. of ICML, 2008

[14] Levy, K. Y., and Shimkin, N., Unified inter and intra options learning using policy gradient methods. In EWRL, 153164, 2011.

[15] Bacon, P.-L., Harb, J., Precup, D. The option-critic architecture. Proceedings of AAAI, 17261734, 2017.