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

第9回 汎用人工知能研究会

2018-08-30

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

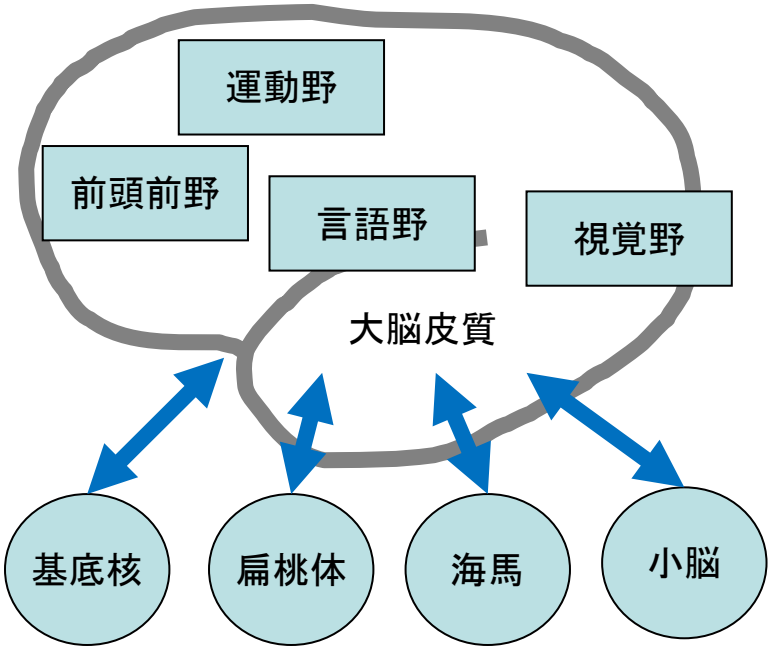
[1] National Institute of Advanced Industrial Science and Technology (AIST), AIRC

[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

私の研究の長期的目標

- 脳を模倣して「人間のような知能を持つ機械」を作る。

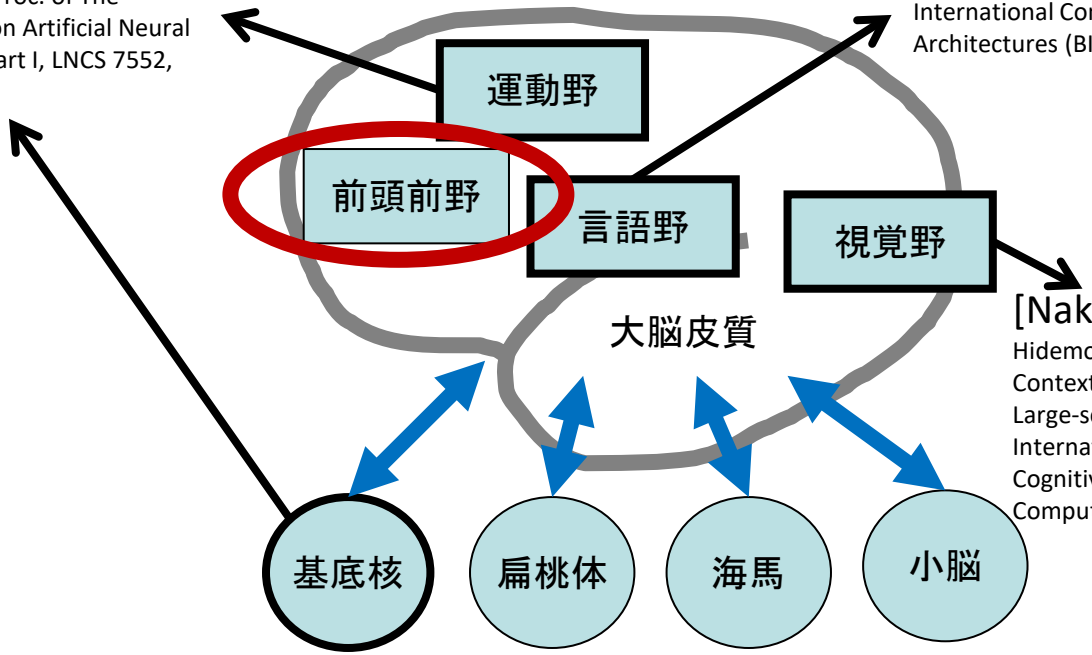


脳のリバースエンジニアリング

現在の短期的目標：前頭前野周辺の計算モデルの構築を目指す

[Ichisugi, ICANN 2012]
Yuuji Ichisugi, A Computational Model of Motor Areas Based on Bayesian Networks and Most Probable Explanations, In Proc. of The International Conference on Artificial Neural Networks (ICANN 2012), Part I, LNCS 7552, pp.726--733, 2012.

[Ichisugi and Takahashi, BICA 2018]
Yuuji Ichisugi and Naoto Takahashi, A Formal Model of the Mechanism of Semantic Analysis in the Brain, In Proc. of International Conference on Biologically Inspired Cognitive Architectures (BICA 2018), (to appear).



[Nakada and Ichisugi, BICA 2017]
Hidemoto Nakada and Yuuji Ichisugi, Context-Dependent Robust Text Recognition using Large-scale Restricted Bayesian Network, In Proc. of International Conference on Biologically Inspired Cognitive Architectures (BICA 2017), Procedia Computer Science, Vol. 123, pp.314--320, 2018.

前頭前野(PFC)の機能

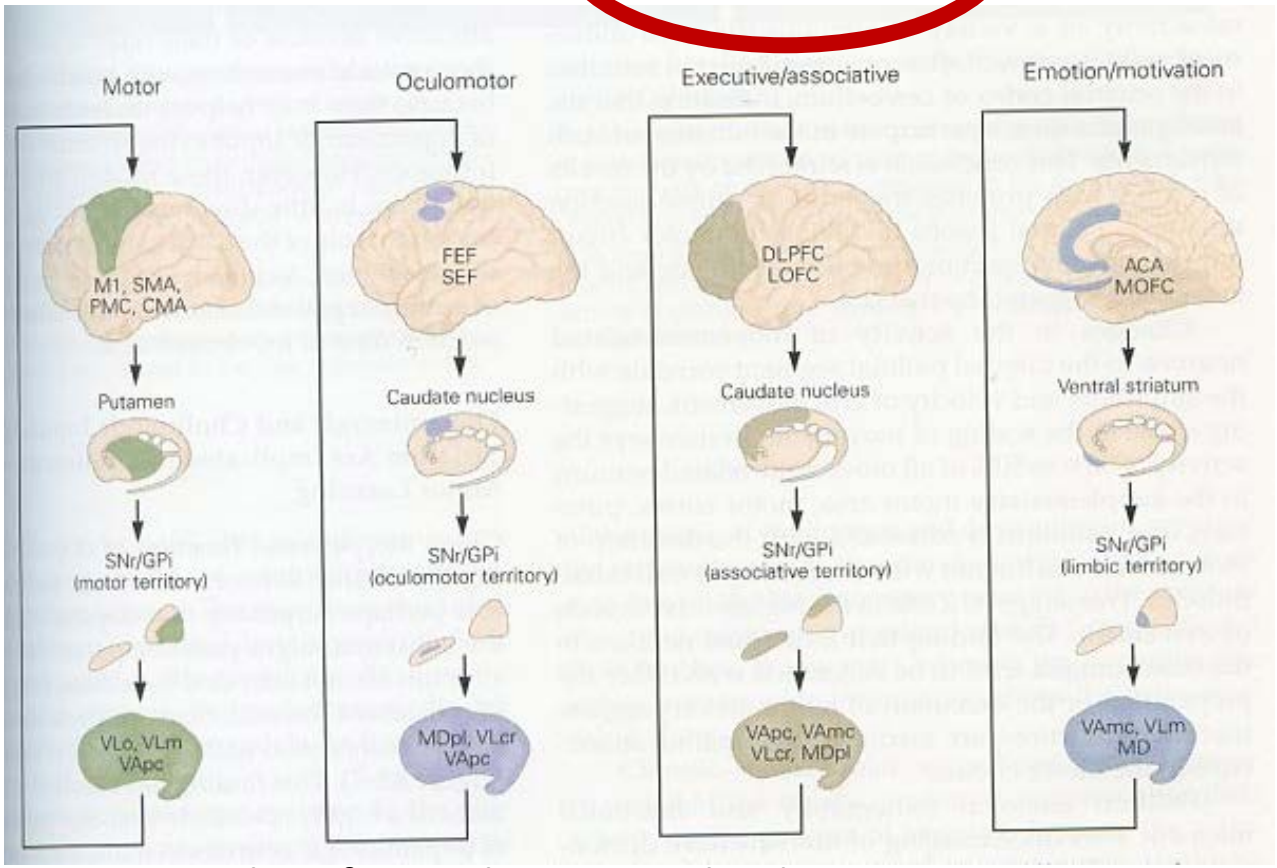
- **「前頭前野はヒトをヒトたらしめ，思考や創造性を担う脳の最高中枢であると考えられている。」**
- 脳科学者ペンフィールドの姉の例：
  「彼の姉は前頭前野に脳腫瘍ができたため，その切除手術を受けた結果、例えば「料理」のような行動が困難になったことが報告されている。」
  「前頭前野に損傷を受けると，このような順序だった行動の組立をする，つまり段取りをうまくとる事ができなくなってしまうのである。」

「前頭前野 – 脳科学辞典」
http://bsd.neuroinf.jp/wiki/%E5%89%8D%E9%A0%AD%E5%89%8D%E9%87%8E

4つの並行した皮質－基底核ループ

関係する主な領野と機能：

M1・PM 運動　　**FEF** 眼球運動　　**前頭前野** 実行機能・連合　　**ACC** 情動・動機



大脳皮質

線条体

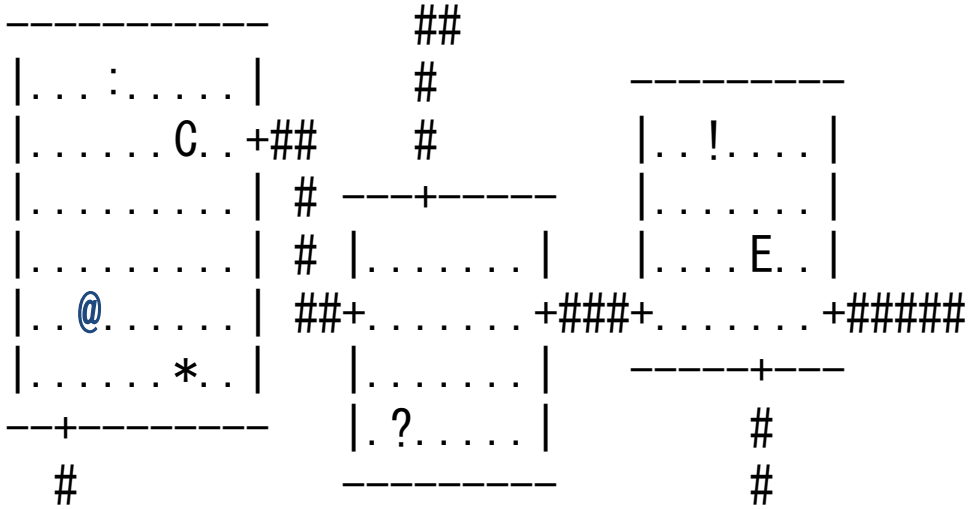
黒質網様体・淡蒼球内節

視床

Principles of Neural Science 5th ed.
Eric R. Kandel et al. , McGraw-Hill
p. 987

前頭前野モデルの目標

- ２次元マップ上で行動計画を立てながら道具を使ってエサを取り、仲間と簡単な言語で情報交換をする知的エージェントを作る。



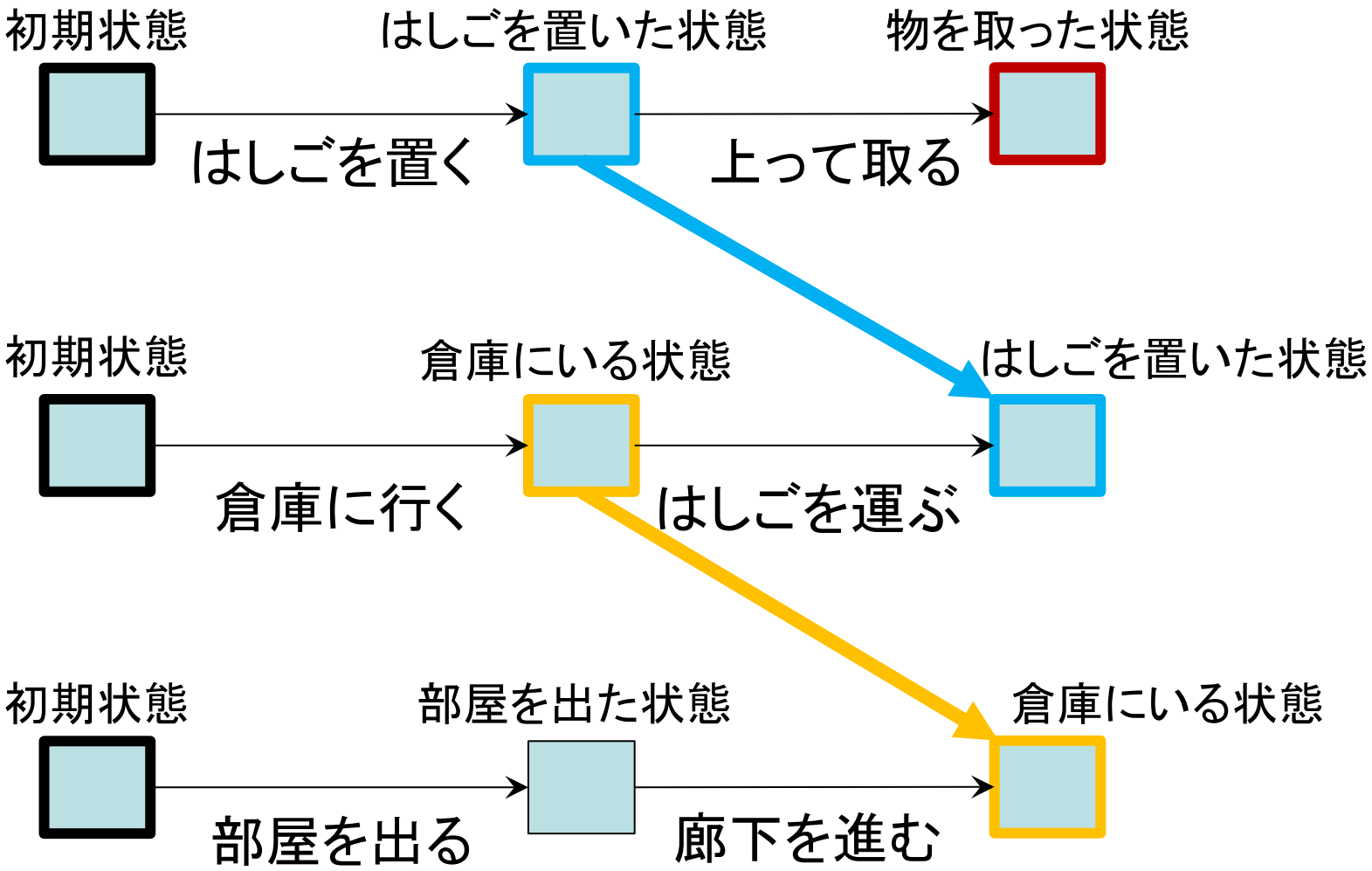
無限に広い空間
数に制限のないアイテム

エージェントが動作する環境のイメージ

前頭前野モデル構築のアイデア

- 再帰的に動作する階層型強化学習　今日の話
- 単一化の機構
- 行動価値テーブルの記憶域管理機構
- 注意とオブジェクトファイルの機構
- ワーキングメモリへの読み書きの機構
- メタ認知、メタ学習の機構

人間が再帰的にサブゴールを設定する例

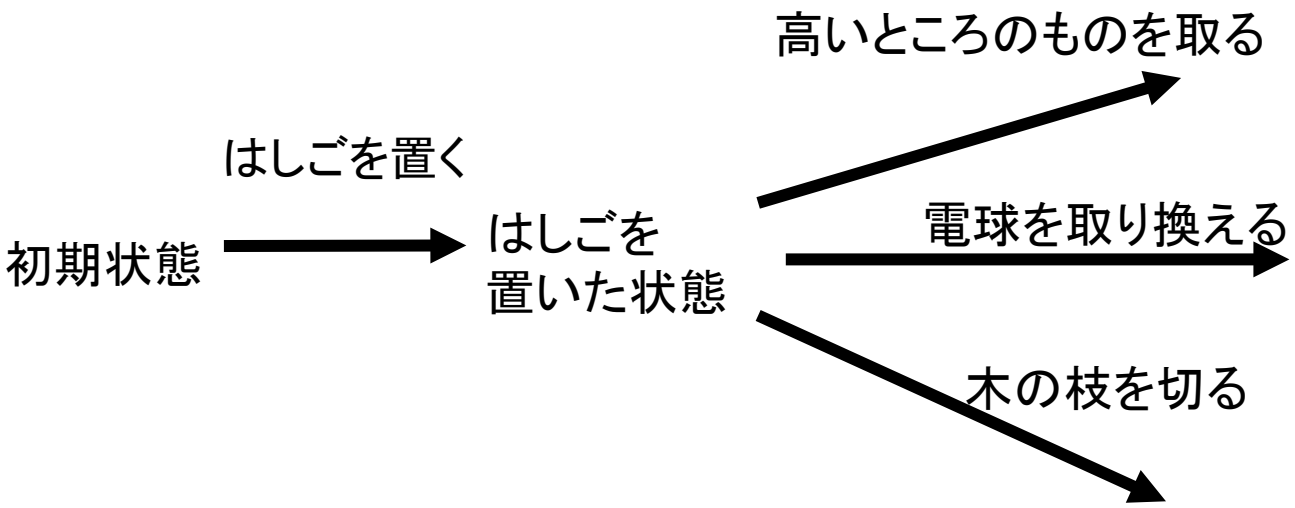


階層型強化学習の利点

1．サブルーチンをタスク間で共有すれば、マルチタスクでの学習が速くなる。

2．学習済みのタスクを組み合わせて、未学習のタスクの近似解が高速に得られる。

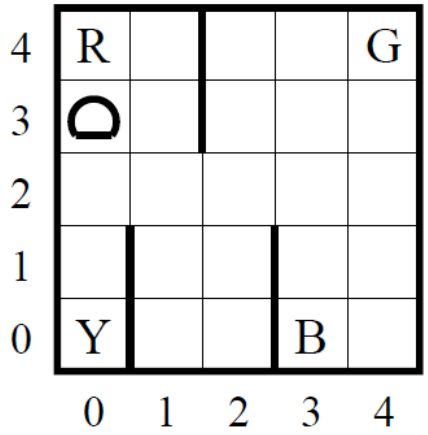
サブルーチン共有の例

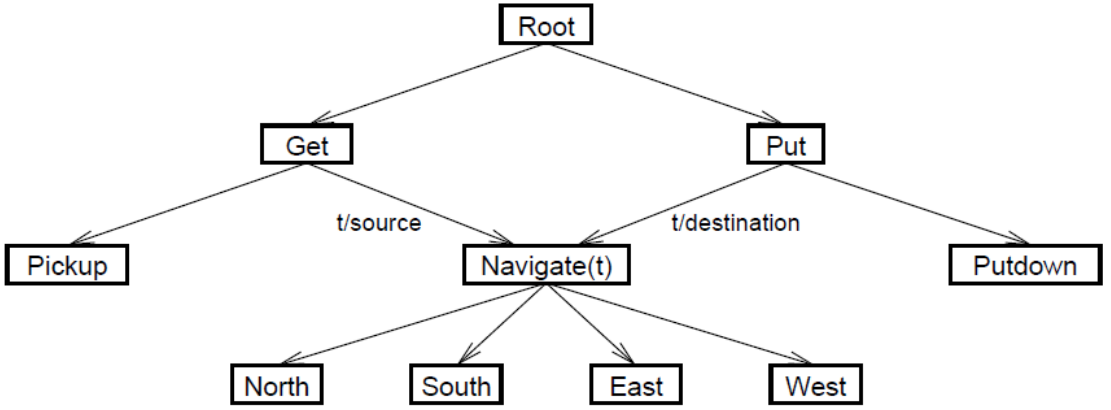


最終目的は違っていても、
「はしごを置く」というサブルーチンは共通

MAXQ [Dietterich 2000]

- 多層の階層型強化学習アーキテクチャ。
- タスクの階層構造を設計者が与える。
- 実行・学習にはスタックが必要。
- 探索空間に制約があり近似解しか得られない。





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

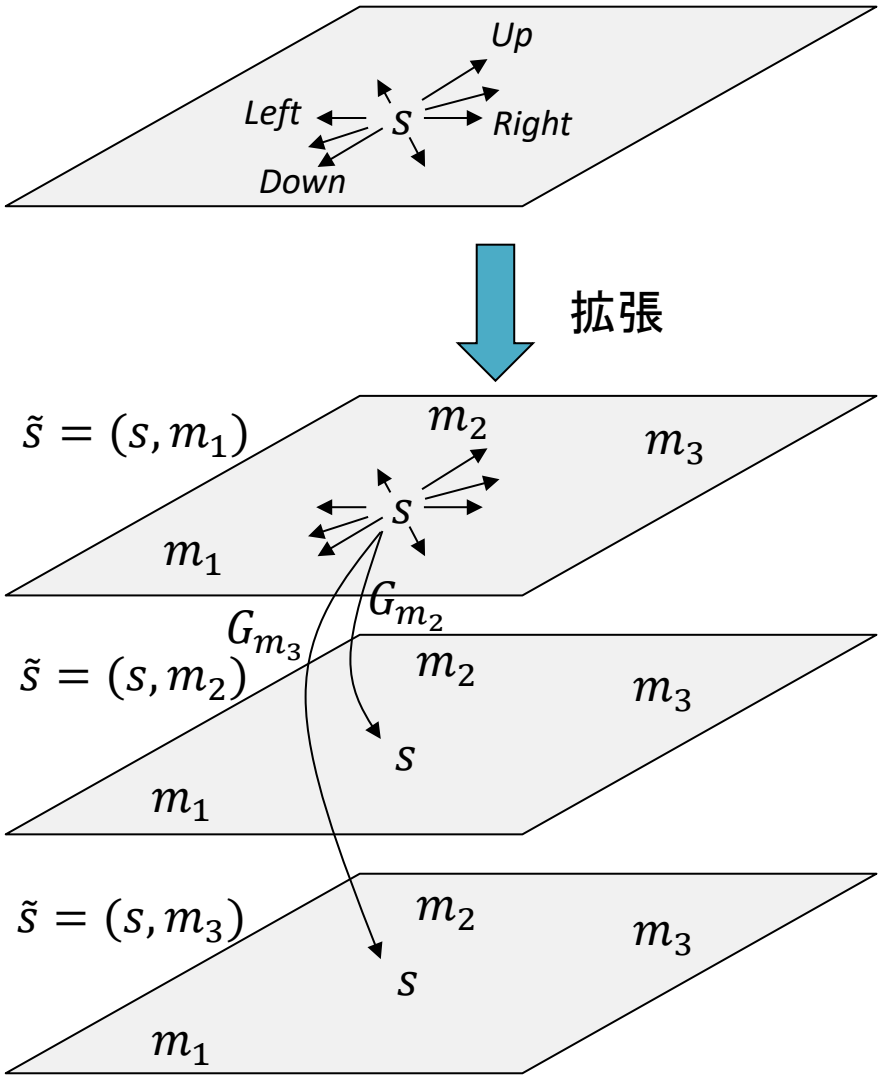
RGoal アーキテクチャ

- 特徴:
  - 再帰的にサブゴールを設定可能
  - スタックなしで、1つのテーブルと1重ループのみで動作
- 2つの仮定:

  1. サブゴールを切り替えると直前のサブゴールは忘れるが、おおもとのグローバルゴールは忘れない。

  2. ランドマークと呼ぶ状態の集合をあらかじめ与える。ゴール、サブゴールは必ずランドマーク上。

拡張状態行動空間 [Levy and Shimkin 2011]

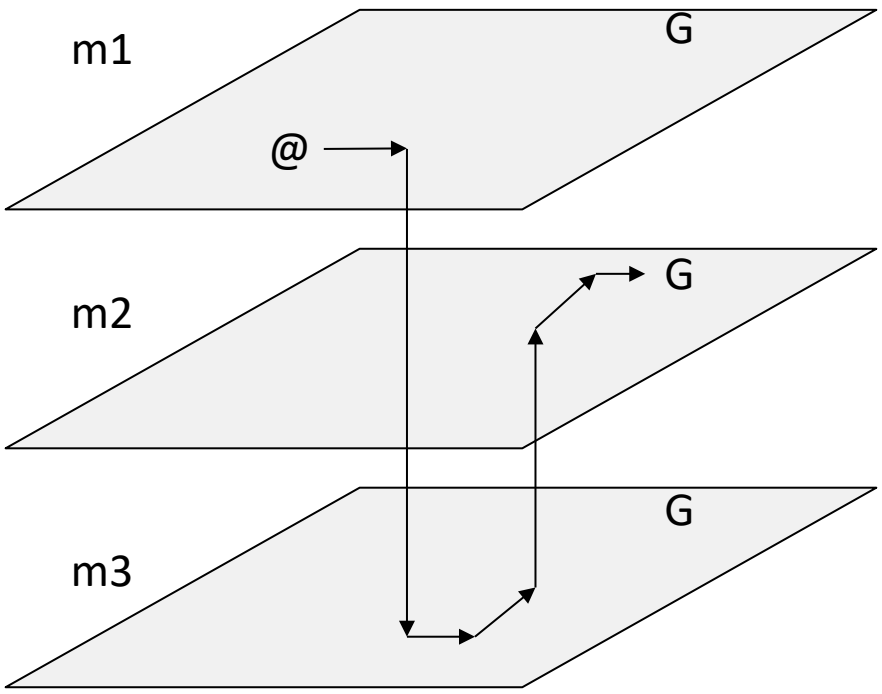


$\mathcal{S} = \{(0,0), (0,1), \dots\}$
$\mathcal{A} = \{Up, Down, Right, Left, \dots\}$

もともとの状態 s とサブゴール g の組 $\tilde{s} = (s, g)$ を拡張された状態と見なす。

$\tilde{\mathcal{S}} = \mathcal{S} \times \mathcal{M}$
$\tilde{\mathcal{A}} = \mathcal{A} \cup \mathcal{G}_{\mathcal{M}}$
$\mathcal{M} = \{m_1, m_2, \dots\} \subseteq \mathcal{S}$
$\mathcal{G}_{\mathcal{M}} = \{G_{m_1}, G_{m_2}, \dots\}$

拡張行動空間上のMDP



・ランドマークが n 個与えられれば、
マップは n 階建てになる。

・エージェントはフロア内を移動するか、
フロアを切り替えるのかを選択する。
（フロア切り替え＝サブゴール切り替え）

数学的構造は通常の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。

価値関数分解[Singh 1992][Dietterich 2000]

- 関数 Q をサブゴール g の到着の前後で分解

s → g → G

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

- **関数 M はグローバルゴール G に依存しないため、異なるタスク間で共有できる。**
  - パラメタが減って、学習が速くなる。
- 例:「はしごを置く」という行動のコスト(労力)は、はしごを使う目的に依存しない。

行動選択と学習

グリーディーな行動選択:

$$a = \underset{a}{\operatorname{argmax}}\, M(s, g, a)$$

Sarsa による学習:

$$M(s, g, a) \leftarrow M(s, g, a)$$
$$+\alpha(r + M(s', g', a') - M(s, g, a) + V_G(g') - V_G(g))$$

$V_G(g)$ は M を使って高速に計算可能

$$V_G(g) = \Sigma_a \pi((g, G), a) M(g, G, a)$$

階層型強化学習の利点

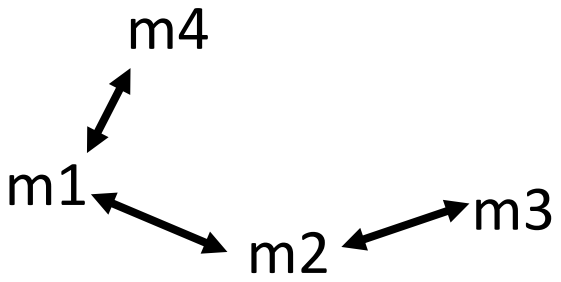
1. サブルーチンをタスク間で共有すれば、マルチタスクでの学習が速くなる。

2. 学習済みのタスクを組み合わせて、未学習のタスクの近似解が高速に得られる。

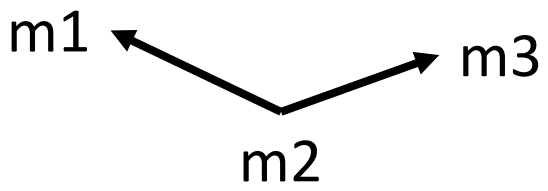
- ここまでの機構は、**1の利点**のみを実現。
- 2を実現するために「思考モード」を導入。（次ページ）

思考モードで解く問題

- 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。



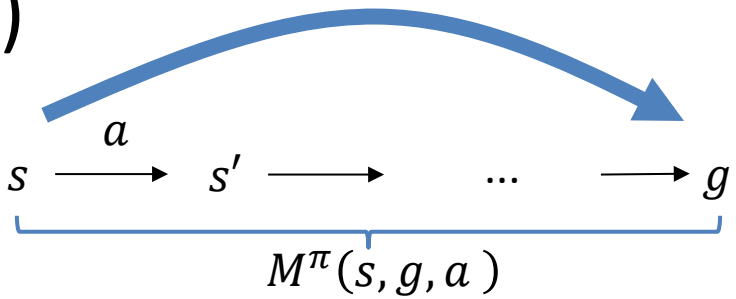
- 離れたランドマーク間の最適移動経路の近似解は、隣接するランドマークをつなげば得られるはず。



- **この近似解は実際に行動しなくても「脳内シミュレーション」だけで見つけられる。**[Singh 1992]
  - 一種のモデルベース強化学習

思考モードにおける移動

- s から g（ランドマーク間）の移動を**1ステップで**脳内シミュレーションする。そのときの報酬は M(s,g,a)



- 未経験のタスクを脳内で解くために、**時間を抽象化して高速にシミュレーション**

アルゴリズム

```
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 de-
   rived 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
```

- 基本構造は Sarsa
- フラットなテーブル M を1つだけ使用
- 単純な操作の1重ループ
- 思考モードもわずかな修正だけで実現

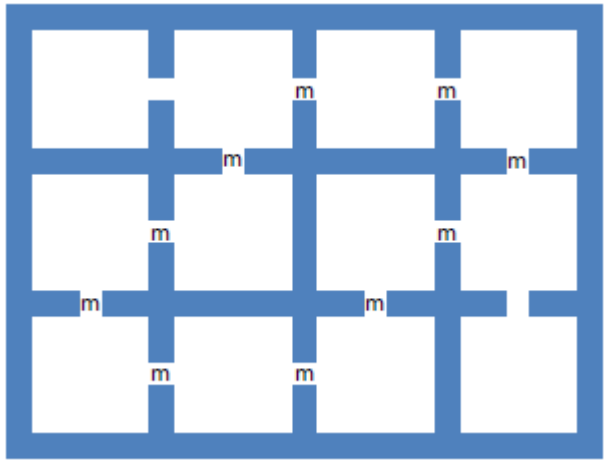
アルゴリズム

```
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 de-
   rived 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
```

- **思考モードもわずかな修正だけで実現**

環境のモデル P(s',r|s,a) を別途学習する必要がない。

実験１



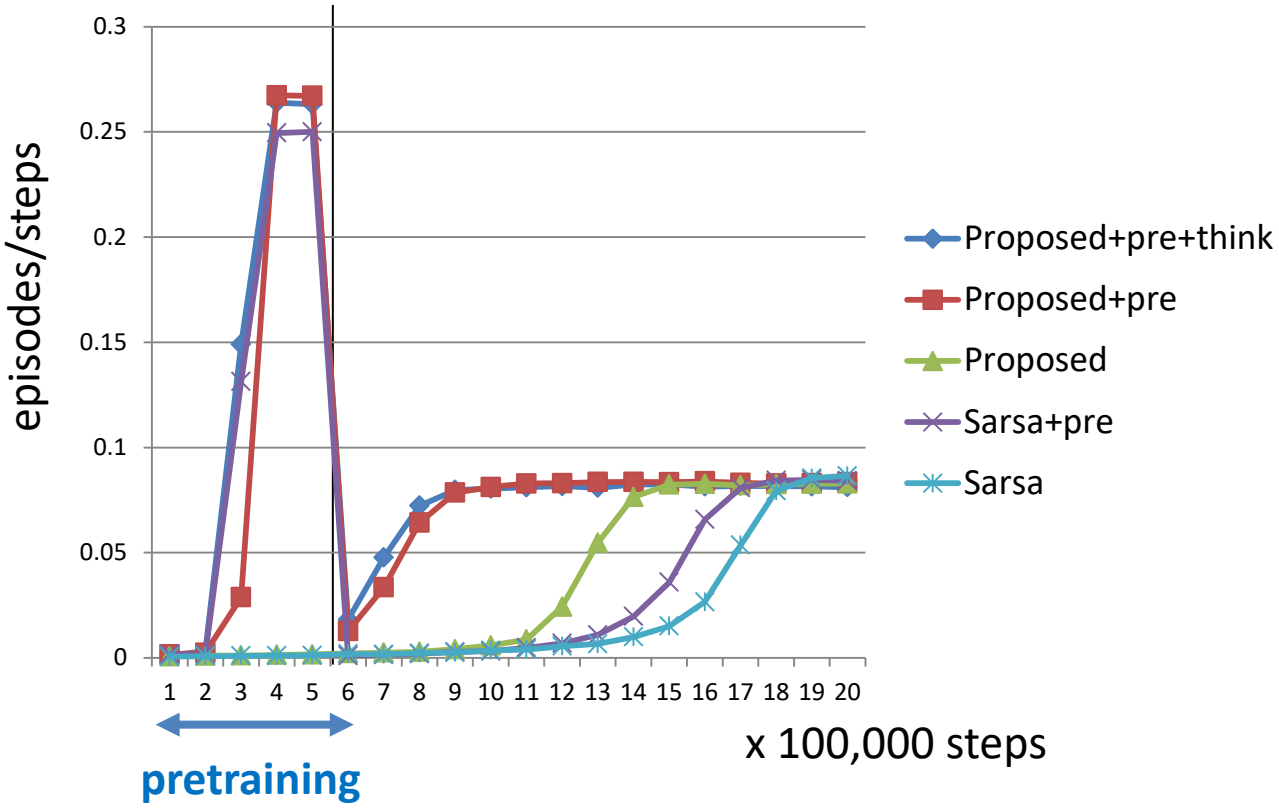
・マップとランドマークの集合は固定。
スタートＳとゴールＧをエピソードごとにランダムに設定する。

・ **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。

・**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。

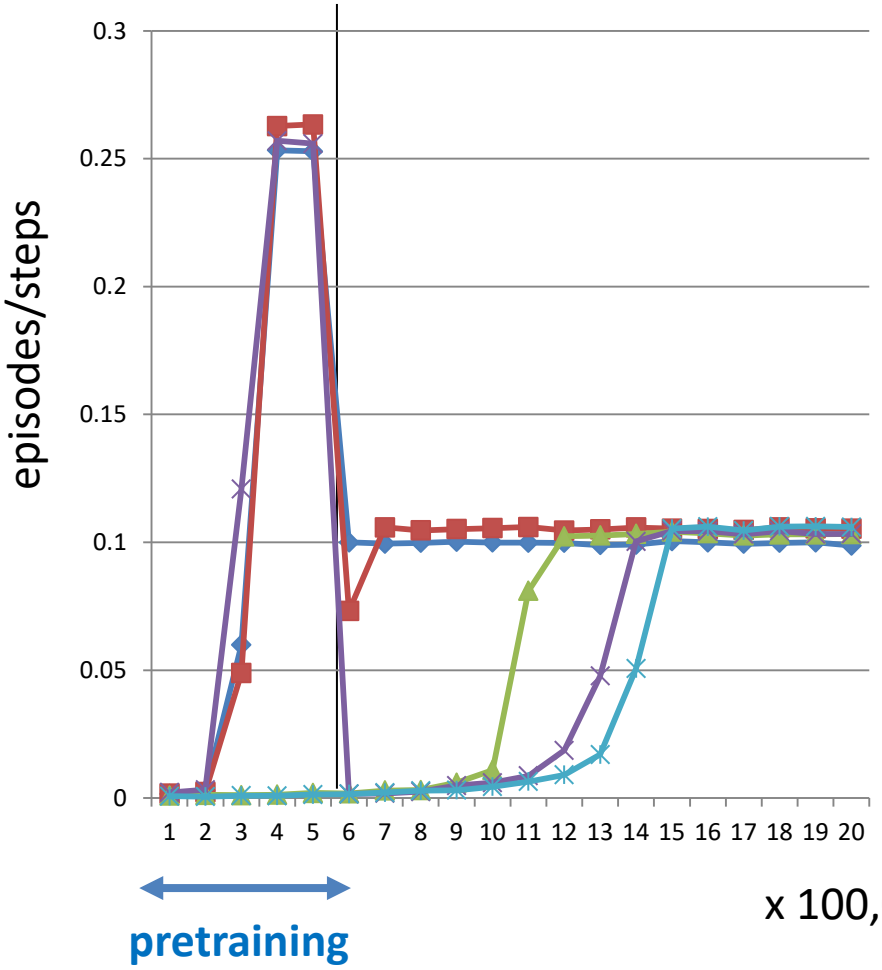
・**評価フェーズ：**
任意の場所Ｓとランドマーク上のＧをランダムに設定したエピソードを１５０万ステップ実行。

実験1の結果



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

実験2の結果



実験1とほぼ同じだが、思考フェーズで学習したものとおなじタスクで評価フェーズを実行

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。

先行研究との関係

- 提案アーキテクチャは、階層型強化学習のいくつかのアイデアをシンプルな1つのアーキテクチャに統合
  - 可変時間分解能モデル [Singh 1992]
  - 再帰的なサブゴール設定 [Kaelbling 1993]
  - サブルーチンの早期終了[Kaelbling 1993]
  - 価値関数分解 [Singh 1992][Dietterich 2000]
  - 拡張状態行動空間 [Levy et al. 2011]
- 状態抽象、自律的もしくは模倣によるランドマーク獲得の機能は今後追加
  - 提案アーキテクチャの拡張はおそらく容易

まとめと今後

- RGoal アーキテクチャを提案
- 人間の知能の2つの重要な特性を再現
  - 再帰的サブゴール設定
  - 時間を抽象化した高速なプランニング

- 今後、提案アーキテクチャを拡張し、前頭前野周辺の情報処理機構のモデルを構築

予備スライド

RGoal アーキテクチャの特徴

- 3つの制限のおかげでアルゴリズムがシンプルに
  - ランドマークのみがゴール・サブゴール
  - サブゴールとグローバルゴール以外の実行コンテキストを忘却
  - サブルーチン終了時の状態が一意

グローバルゴール

- 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)は決して忘れないと仮定
- 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行動を開始
- 生物におけるグローバルゴールの例:
  - 空腹やのどの渇きなどの生理的欲求の解消

ランドマーク

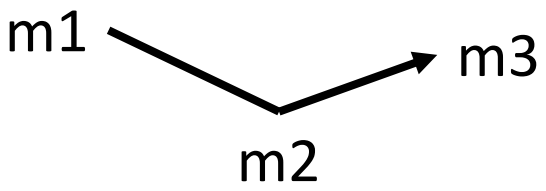
- タスクを解く上でのマイルストーン（中間目標）となり得る状態
- ゴール、サブゴールは必ずランドマーク上にあると仮定
- 今回はランドマークの集合はあらかじめ与えるものと仮定
- 将来は、模倣や何らかのヒューリスティックスで獲得

提案アーキテクチャにおける「サブルーチン」の特徴

- Options [Sutton et al. 1999] より
  HDG [Kaelbling 1993] の考え方に近い。
- 開始状態集合は全状態、終了状態は1つ
  - **アーキテクチャがシンプルに**
- サブゴールに到着しなくても、いつでも別のサブゴールに切り替え可能
  - **行動がより柔軟に**

思考モードの例

- m1 と m3 の間の最適経路の近似解を求めるには S=m1, G=m3 として思考モードによる学習を続ければよい。
- 学習の結果、$M(m1,m3,G_{m2})$ が大きな値を持つことによって、m1 → m2 → m3 という経路が表現される。

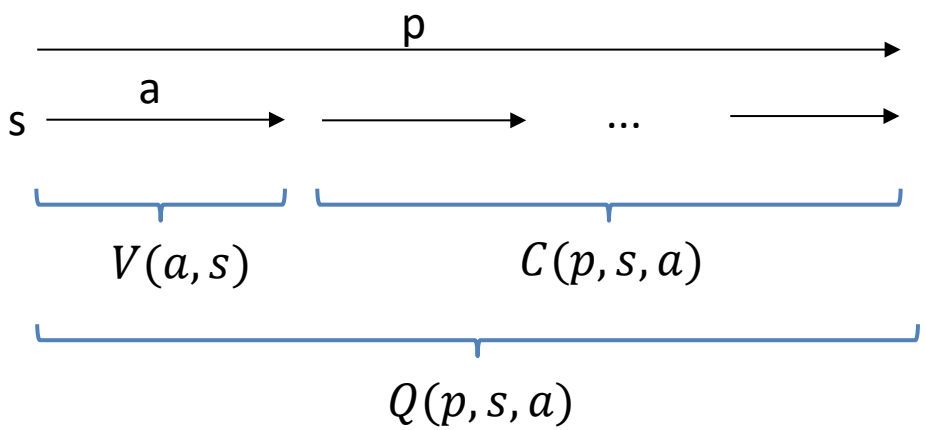


MAXQ 価値関数分解[Dietterich 2000]

$$Q(p,s,a) = V(a,s) + C(p,s,a)$$

p は親タスク、a は子タスクまたはプリミティブな行動。
状態（サブゴール）ではなく行動（サブルーチン）の価値に着目した式。
V(a,s) は親タスク p に依存しないので**タスク間で共有できる**。
ただし V(a,s) は以下の式で再帰的に計算する必要がある。

$$V(p,s) = \begin{cases} \max_a Q(p,s,a) & if\ p\ is\ composite \\ V(p,s) & if\ p\ is\ primitive \end{cases}$$
$$Q(p,s,a) = V(a,s) + C(p,s,a)$$



式 (11) の softmax の式の妥当性

$$\pi\big((s,g)a\big) = \frac{\exp\Big(\beta Q\big((s,g),a\big)\Big)}{\Sigma_{a'}\exp\Big(\beta Q\big((s,g),a'\big)\Big)}$$

$$= \frac{\exp(\beta(M(s,g,a)+V_G(g)))}{\Sigma_{a'}\exp(\beta(M(s,g,a')+V_G(g)))}$$

$$= \frac{\exp(\beta V_G(g))\exp\big(\beta M(s,g,a)\big)}{\exp(\beta V_G(g))\,\Sigma_{a'}\exp\big(\beta M(s,g,a')\big)}$$

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