確率的２分木の行動選択を用いた強化学習による多数の類似行動の扱いについて

○ 木村 元，　小林 重信

東京工業大学 大学院総合理工学研究科

Binary Tree Action Selector: Reinforcement Learning to Cope with Enormous Similar Actions

○ Hajime Kimura,　Shigenobu Kobayashi

Interdisciplinary Graduate School of Science and Eng., Tokyo Institute of Technology

Abstract: In real world applications, learning algorithms often have to handle several dozens of actions, which have some distance metrics. Q-learning or SARSA algorithms with epsilon-greedy exploration strategies are very popular, simple and effective in the problems that have a few actions, however, the efficiency would decrease when the number of actions is increased. We propose a policy function representation that consists of a stochastic binary decision tree, and we apply it to an actor-critic algorithm for the problems that have enormous similar actions. Simulation results show the increase of the actions does not affect learning curves of the proposed method at all.

1　はじめに

強化学習は，ロボットの挙動を自ら改善したり知識の欠落を自ら補うための接近法として有望である．実環境のロボット学習制御において，繊細な制御を行うためには連続で膨大な状態空間の扱いが求められると同時に，多数の行動選択の学習も求められる．連続または膨大な状態空間を扱うための方法として CMAC(Sutton, 1996) などの関数近似や instance-based な方法を用いて状態空間を汎化する接近法が確立されている．しかし多数の行動を扱うためには，action value を汎化する技術だけでなく，行動選択や探査戦略についても特別の工夫が必要である．Q-learning(Watkins,1992) や SARSA アルゴリズムと共によく用いられるε -greedy 戦略やボルツマン選択戦略は仕組みが単純なのでよく用いられ，行動数が少ない場合は強力であることが知られている．しかしこれらは行動の数が増加すると効率が悪くなるため，別の工夫が必要である．一方で，Value 関数とは別に任意のパラメータ表現された政策関数を持つことにより，連続値の行動を扱うことが容易な強化学習アルゴリズムとして actor-critic が知られている．本論文では actor-critic の政策関数表現として確率的な 2 分木を用いることにより，数十以上の多数の類似した行動を効率良く扱う方法について示す．

1.1　マルコフ決定過程 (MDP)

状態空間を $\mathcal{S}$，行動空間を $\mathcal{A}$，実数の集合を $\mathcal{R}$ と表す．各時刻 t でエージェントは状態観測 $s_t \in \mathcal{S}$に基づいて行動 $a_t \in \mathcal{A}$ を実行し，状態遷移に伴う報酬 $r_t \in \mathcal{R}$ を得る．多くの強化学習法が基礎としているマルコフ決定過程 (MDP) では，一般に次の状態や報酬は確率的で，その分布は s_tと a_tにのみ依存する．MDP では次の状態 s_{t+1}は遷移確率 $T(s_t, a, s_{t+1})$ に従って決まり，報酬 r_tも期待値 $r(s_t, a)$ によって与えられる．エージェントは予め $T(s_t, a, s_{t+1})$ や $r(s_t, a)$ についての知識は持っていない．強化学習の目的はエージェントの挙動を最適化する政策を得ることである．無限期間のタスクにおいて自然な評価規範として，以下のような割引報酬の合計がある．

$$V_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \ , \tag{1}$$

割引率 $0 \leq \gamma \leq 1$ は未来に得るであろう報酬の現時点での重要度を表し，V_tは時刻 t の評価値 (value) を表す．MDP では value は以下に定義される．

$$V^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s, \pi \right] \ , \tag{2}$$

ただし $E\{\cdot\}$ は期待値を表す．MDP における学習の目的は，各状態 s において式 2で定義される value を最大化するような最適政策を見つけることである．

1.2 多数の類似行動を有する環境

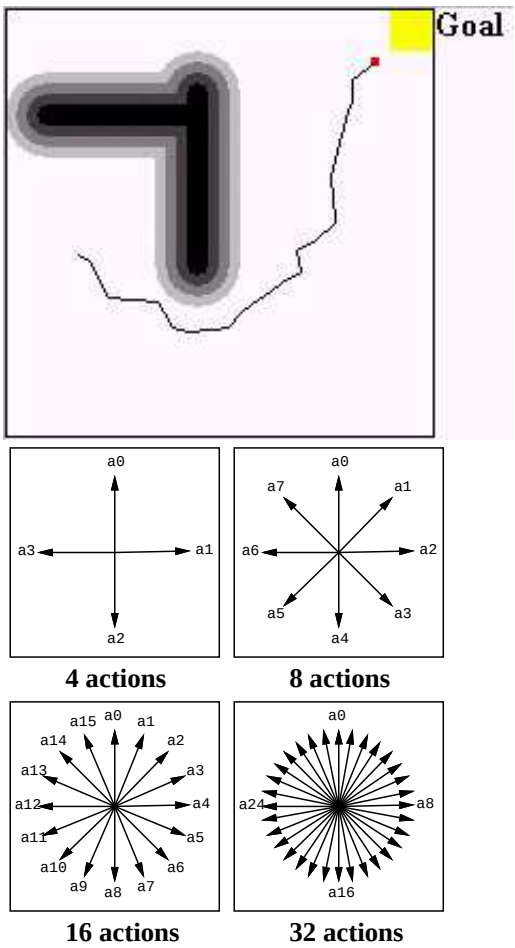


図 1: 水たまり問題．行動の種類は $4,\ 8,\ 16,\ 32$ または 64 の場合がある．行動を選択すると，その矢印の方向へおよそ 0.05 ほど移動する．状態空間は 2 次元で，各軸がそれぞれ $[0,1]$ の範囲に限定される．

多くの実問題では，隣り合う行動の多くは類似した状態遷移となる性質を有している．図 1は (Sutton, 1996) の水たまり問題 (puddle world task) を示す．状態空間は 2 次元の連続値で各軸は $[0,1]$ の範囲である．エージェントは現在位置の座標を状態として観測する．エージェントが行動を選択すると，上記の範囲を超えない限り選択された方向へおよそ 0.05 の距離を移動する．移動の際，標準偏差 0.01 のガウス分布のノイズが各軸方向に付加される．行動の種類が増加すると隣接する行動が類似してくることは図より明らかである．各時間ステップにおいて -1 の報酬に加えて，水たまり (puddle) に入ることでさらに負の報酬が与えられる．この水たまりによる負の報酬は，境界線からの最短距離に -400 を掛けた値である．水たまりは半径 0.1 で中心座標は $(0.1, 0.75)$ から $(0.45, 0.75)$ までおよび $(0.45, 0.4)$ から $(0.45, 0.8)$ までの位置に配置される．各試行 (episode または trial と呼ばれる) における初期状態はゴール状態を除いて一様分布で生成する．ゴールは $[0.9, 0.9]$ から $[1.0, 1.0]$ までの矩形領域である．

2 勾配法による政策改善

2.1 REINFORCE アルゴリズム

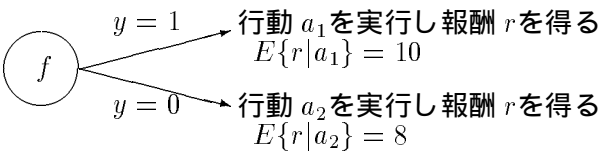


図 2: Bernoulli semilinear ユニットの一例．確率変数 $y \in \{0, 1\}$ は分布 $P(y = 1) = f$に従う．

本章は本手法に関連の深い REINFORCE アルゴリズム (Williams, 1992) の特殊な場合について述べる．図 2のような Bernoulli semilinear ユニットを考える．確率変数 $y \in \{0, \text{or } 1\}$ は分布 fに従い，$P(y = 1) = f$または $P(y = 0) = 1 - f$のように振舞う．エージェントは現在の状態を特徴ベクトル $(x_1, x_2, \cdots, x_i, \cdots, x_n)$ によって観測する．分布関数 fは以下の式で与えられる：

$$f = \frac{1}{1 + \exp\left(-\sum_i \theta_i x_i\right)}\ , \tag{3}$$

ただし x_iは状態入力の特徴ベクトルの要素，θ_iは政策パラメータの要素を表す．エージェントは政策パラメータθの調節により挙動を改善する．エージェントは行動 1 または 0 を選択し，報酬 rを受け取る．そのとき政策パラメータθの適正度 (eligibility) は以下の式で計算される．

$$\begin{aligned} e &= \frac{\partial}{\partial \theta_i} \ln P(\text{実行された行動}) \\ &= \begin{cases} \frac{\partial}{\partial \theta_i} \ln f & \text{if } y = 1 \\ \frac{\partial}{\partial \theta_i} \ln (1 - f) & \text{if } y = 0 \end{cases} \\ &= (y - f)\, x_i \end{aligned} \tag{4}$$

このとき REINFORCE アルゴリズムは

$$\begin{aligned} \Delta\theta_i &= e\,(r - b) \\ \theta_i &\leftarrow \theta_i + \alpha\, \Delta\theta_i\ , \end{aligned}$$

ただしαは学習率，bは報酬基線 (reinforcement baseline) と呼ばれる値で，yとは条件付独立である．このアルゴリズムは以下の勾配に比例して政策パラメータを更新する：$E\{\Delta\theta_i\} = \alpha\frac{\partial}{\partial\theta_i}E\{r\}$. 報酬 rを報酬合計 V_tに置き換えてもこの特徴は失われない。

このアルゴリズムは報酬基線 b を適切に設定することで性能を改善できる．例えば図 2のような簡単な環境を考える．$b = 9$ のとき，エージェントはどちらの行動を選択しても常に正しい行動 a_1を強化できるため，性能が最も良い．例えば行動 a_1を実行した場合，a_1は正の強化値 $r - b = 10 - 9$ により強化される．行動 a_2を実行した場合，a_2の選択確率は負の強化値 $r - b = 8 - 9$ により減少するため a_1が強化される．さらに洗練された方法として b を適応的に生成する方法がある．(Sutton & Barto, 1998) の Reinforcement comparison は，報酬の予測値$\bar{r}$を適応的に更新し，以下のように報酬基線として用いる．

$$\begin{aligned} \Delta\theta_i &= e\,(r - \bar{r}) \\ \theta_i &\leftarrow \theta_i + \alpha\,\Delta\theta_i\ , \end{aligned} \tag{5}$$

ただし$\bar{r}$の計算においては現在の行動 yの情報を用いてはならない．Actor に適正度の履歴を用いた actor-critic アルゴリズム (Kimura & Kobayashi, 1998) は，rを V_tに，$\bar{r}$を$\hat{V}(s)$ にそれぞれ置き換えたものである．

2.2 Actor-Critic アルゴリズム

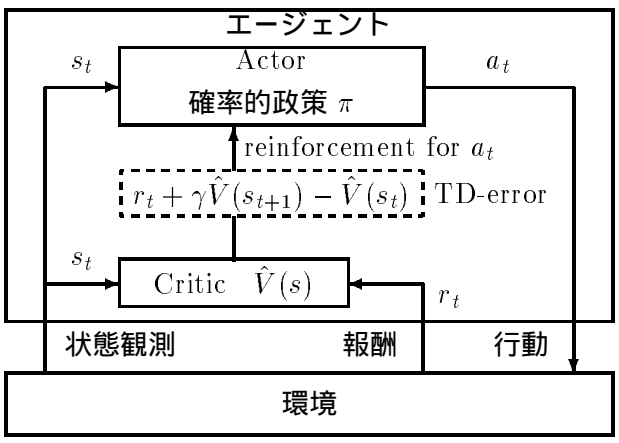


図 3: 一般的な actor-critic アルゴリズムの枠組．

Actor-Critic 学習システムは，政策反復法 (Policy iteration) の一種であると言われている．政策反復法は状態評価と政策改善を交互に繰り返す．Actor は，状態から行動への確率分布である確率的政策 (**stochastic policy**) に従って行動を実行する．Critic は，Actor の政策のもとでのそれぞれの状態の value を推定する．Actor は，環境から直接受け取る報酬と，critic で推定される value より計算される TD エラーを手がかりに政策を改善する．図 4は本論文で用いたアルゴリズムを示す．

Critic における処理手順は図 4のステップ 3 から 5 までに示されている．パラメータλ_vは TD($\lambda = \lambda_v$) の適正度の履歴を表す．政策は actor において関数として明示的に表現され，政策パラメータを value 関数の勾配によって更新する (Kimura and Kobayashi, 1998; Sutton, McAllester, Singh & Mansour,2000)．観測 s においてエージェントが行動 a を選択する確率を政策πと呼び，関数$\pi(a, \theta, s)$ で表す．政策$\pi(a, \theta, s)$ は行動 a の集合が連続値の場合は確率密度関数である．エージェントは内部変数θを調節することにより確率的政策πを変える．

図 4のステップ 4,5 は actor の処理手続きである．適正度 e_πは REINFORCE algorithm で定義されているものと同一である．パラメータλ_πは actor の適正度の履歴の性質を定めるが，その性質は TD(λ) の場合とはやや異なる．λ_πが 0 に近い場合，critic が学習している$\hat{V}$の勾配により政策が改善され，λ_πが 1 に近い場合は式 1で定義される実際の利得の勾配によって政策が改善される．ここで$\lambda_\pi = 1$，かつ $t < 0$ のとき$\overline{e_\pi}(t) = 0$ と仮定する．このとき図 4のアルゴリズムは以下のように政策パラメータを更新する：

$$\begin{aligned} \sum_{t=0}^{\infty}\Delta\theta(t) &= \sum_{t=0}^{\infty}\left(r_t + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t)\right)\overline{e_\pi}(t) \\ &= \sum_{t=0}^{\infty}\left(r_t + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t)\right)\left(\sum_{\tau=0}^{t}\gamma^{t-\tau}e_\pi(t)\right) \\ &= \sum_{t=0}^{\infty}e_\pi(t)\left(\sum_{\tau=t}^{\infty}\gamma^{\tau-t}\left(r_\tau + \gamma\hat{V}(s_{\tau+1}) - \hat{V}(s_\tau)\right)\right) \\ &= \sum_{t=0}^{\infty}e_\pi(t)\left(\left(\sum_{\tau=t}^{\infty}\gamma^{\tau-t}r_\tau\right) - \hat{V}(s_t)\right) \qquad (7) \\ &= \sum_{t=0}^{\infty}e_\pi(t)\left(V_t - \hat{V}(s_t)\right)\ . \qquad (8) \end{aligned}$$

式 8は式 1および式 7より得る．ここで式 8のΣ 中の項について $e_\pi(t)$ は行動 a_tに関連し，V_tは a_tの割引報酬であり，$\hat{V}(s_t)$ は a_tや V_tを計算するよりも前に与えられるため a_tや V_tとは条件付独立である．よって式 8のΣ 中の項は式 5と同じであり，図 4が reinforcement comparison の拡張であることが分かる．

1. 状態 s_tを観測し，確率分布$\pi(a_t, \theta, s_t)$ に従って行動 a_tを選択／実行する．

2. 報酬 r_tを受け取り，遷移後の状態 s_{t+1}を観測して以下の TD-error を計算する．

$$(\text{TD-error}) = r_t + \gamma \hat{V}(s_{t+1}) - \hat{V}(s_t) \ , \quad (6)$$

γ $(0 \le \gamma \le 1)$ は割引率，$\hat{V}(s)$ は critic が推定した割引報酬の期待値を表す．

3. TD 法を用いて critic の$\hat{V}(s)$ を更新する．

$$\begin{aligned} e_v(t) &= \frac{\partial}{\partial w}\hat{V}(s_t) \ , \\ \overline{e_v}(t) &\leftarrow e_v(t) + \overline{e_v}(t) \ , \\ \Delta w(t) &= (\text{TD-error})\,\overline{e_v}(t) \ , \\ w &\leftarrow w + \alpha_v \, \Delta w(t) \ , \end{aligned}$$

ただし e_vは$\hat{V}(s)$ のパラメータ wの適正度，$\overline{e_v}$はその適正度の履歴，α_vは学習率を表す．

4. TD-error で actor の行動選択確率を更新する

$$\begin{aligned} e_\pi(t) &= \frac{\partial}{\partial \theta} \ln\Big(\pi(a_t, \theta, s_t)\Big) \ , \\ \overline{e_\pi}(t) &\leftarrow e_\pi(t) + \overline{e_\pi}(t) \ , \\ \Delta\theta(t) &= (\text{TD-error})\,\overline{e_\pi}(t) \ , \\ \theta &\leftarrow \theta + \alpha_\pi \, \Delta\theta(t) \ , \end{aligned}$$

ただし e_πは政策パラメータπの適正度，$\overline{e_\pi}$はその適正度の履歴，α_πは学習率を表す．

5. 適正度の履歴を以下のように割り引く．

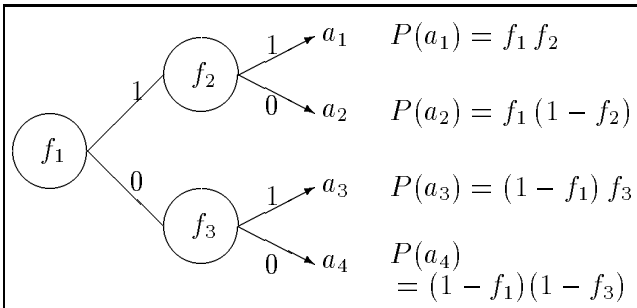
$$\begin{aligned} \overline{e_v}(t+1) &\leftarrow \gamma\,\lambda_v\,\overline{e_v}(t) \ , \\ \overline{e_\pi}(t+1) &\leftarrow \gamma\,\lambda_\pi\,\overline{e_\pi}(t) \ , \end{aligned}$$

ただしλ_vおよびλ_π $(0 \le \lambda_v, \lambda_\pi \le 1)$ はそれぞれ critic と actor の割引率である．

6. 時刻を $t \leftarrow t+1$ に進めてステップ 1 へ戻る．

図 4: Actor と critic の両方に適正度の履歴を用いる actor-critic アルゴリズム．

3　2 分木構造の確率的行動選択



1. 最上位ノードにおいて f_1の確率で 1 を選択 (または $1-f_1$の確率で 0 を選択)．選択した数字のリンクの方向へ進む．

2. 次のノードにおいても同様に f_2または f_3の確率で 1 を選択，さもなければ 0 を選択し，選択した数字のリンクの方向へ進むという処理を葉ノードにたどり着くまで繰り返す．

3. 最終的にたどり付いた葉ノードの行動 a を実行

4. 全てのノードについて以下の適正度を計算：

$$\frac{\partial}{\partial f_j} \ln \pi(a, \theta, s) = \begin{cases} \frac{y_j - f_j}{f_j(1-f_j)} & \text{if } f_j \text{ is active} \\ 0 & \text{otherwise,} \end{cases}$$

ただし $y_j \in \{0, 1\}$ はノード f_jの出力を表す．

図 5: 行動が 4 個の場合における 2 分木行動選択の処理例と確率的政策表現．f_iは確率分布関数を表す．

図 5は 2 分木構造の行動選択方法について，行動が 4 個の場合の例を示す．葉ノードはそれぞれ行動に対応する．それ以外のノードは図 2の Bernoulli semi-linear unit によって構成される．理論上，この確率的政策関数は以下のように記述される．

$$\pi(a, \theta, s) = \begin{cases} f_1\,f_2 & a = a_1 \\ f_1\,(1-f_2) & a = a_2 \\ (1-f_1)\,f_3 & a = a_3 \\ (1-f_1)\,(1-f_3) & a = a_4 \end{cases} \quad (9)$$

このとき f_1の適正度は以下の式で与えられる：

$$\begin{aligned} \frac{\partial}{\partial f_1} \ln \pi(a, \theta, s) &= \begin{cases} 1/f_1 & a = a_1 \\ 1/f_1 & a = a_2 \\ -1/(1-f_1) & a = a_3 \\ -1/(1-f_1) & a = a_4 \end{cases} \\ &= \frac{y_1 - f_1}{f_1(1-f_1)} \ , \end{aligned} \quad (10)$$

ただし $y_1 \in \{0,1\}$ はユニット f_1 が選択した値を表す．ここで分布関数 f_j $(j = 1, 2 \text{ or } 3)$ を式 3のシグモイド関数で与えるもととする．式 4と式 10より，f_1の政策パラメータの適正度は以下のようになる．

$$\frac{\partial}{\partial \theta_i}\pi(a,\theta,s) = \frac{\partial f_1}{\partial \theta_i}\frac{\partial}{\partial f_1}\ln \pi(a,\theta,s) = (y_1 - f_1)x_i \ .$$

同様に f_2や f_3の適正度は以下のようになる．

$$\begin{aligned}\frac{\partial}{\partial f_2}\ln \pi(a,\theta,s) &= \begin{cases} 1/f_2 & a = a_1 \\ -1/(1-f_2) & a = a_2 \\ 0 & a = a_3 \\ 0 & a = a_4 \end{cases} \\ &= \begin{cases} \frac{y_2 - f_2}{f_2(1-f_2)} & \text{if } f_2 \text{ is active} \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

よって f_2の政策パラメータについての適正度は

$$\frac{\partial f_2}{\partial \theta_i}\frac{\partial}{\partial f_2}\ln \pi(a,\theta,s) = \begin{cases} (y_2 - f_2)x_i & \text{if } f_2 \text{ is active} \\ 0 & \text{otherwise.} \end{cases}$$

以上より，各ノードの計算処理はローカルな情報だけで実行できる．2分木のルートノードより葉ノードまでの選択されたパス上の適正度は単純に $(y_j - f_j)x_i$ で与えられ，それ以外のノードでは 0 になる．この性質は木のサイズが変わっても全く同じである．必要なメモリ容量 (ノード数) は 行動数-1 に等しい．

4　計算機シミュレーション結果

行動数の増加がアルゴリズムに与える影響を調べる．図 1の水たまり問題において提案手法と Q-learning および SARSA(λ) を比較する．状態入力より特徴ベクトル $(x_1, x_2, \cdots x_{140})$ を生成するため図 6のタイルコーディング (Sutton & Barto, 1996) を用いる．

actor-critic では状態 value を$\hat{V}(s) = \sum_{i=1}^{140} x_i w_i$ と表し，actor は図 5同様 $f_j = 1/(1+\exp(-\sum_{i=0}^{140}\theta_{ji}x_i))$ と表す．行動の個数 Nのとき critic 中の w_iとその適正度の履歴$\overline{e_v}$について 2×140 個の変数を要し，actor における政策パラメータθ_{ji}とその適正度の履歴$\overline{e_\pi}$について $2\times 140\times(N-1)$ 個の変数を要することから，本手法は $2\times 140\times N$個の変数を必要とする．

Q-learning と SARSA(λ) では，状態-行動 value を $Q(s,a) = \sum_{i=1}^{140} x_i w_{ai}$と表す．行動選択戦略として ϵ-greedy を用いる．つまり確率ϵでランダムに行動を選び，残りの確率で最も大きな Q 値を持つ行動を選ぶ．時刻 t の状態を s_t，その特徴量を $x_i(t)$，そのとき選択した行動を a_t，状態遷移後の状態と特徴量を $st+1$, $x_i(t+1)$，時刻 $t+1$ で選んだ行動を a_{t+1}とすると，Q-learning の学習則は

$$\begin{aligned}\Delta w_{a_t i} &= x_i(t)\left(r_t + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right) \\ w_{a_t i} &\leftarrow w_{a_t i} + \alpha \Delta w_{a_t i} \ ,\end{aligned}$$

これを全ての $i \in \{1,2,\cdots 140\}$ について行う．αは学習率．SARSA(λ) の学習則は以下のとおり

$$\begin{aligned}(\text{TD-error}) &= r_t + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t) \\ \overline{e_{a_t i}}(t) &= x_i(t) + \overline{e_{a_t i}}(t) \\ \Delta w_{ai} &= \overline{e_{ai}}(t)\,(\text{TD-error}) \qquad \text{for all } a \\ w_{ai} &\leftarrow w_{ai} + \alpha \Delta w_{ai} \qquad \text{for all } a \\ \overline{e_{ai}}(t+1) &= \gamma\,\lambda\,\overline{e_{ai}}(t) \qquad \text{for all } a,\end{aligned}$$

これを全ての $i \in \{1,2,\cdots 140\}$ について行う．$\overline{e_{ai}}$は適正度の履歴である．Q-learning では $Q(s,a)$ を近似するため変数 w_iを $140\times N$個要する．SARSA(λ) では変数 w_iとその履歴$\overline{e_{ai}}$のために $2\times 140\times N$個の変数を要する．SARSA が消費するメモリーは提案手法と全く同じである．

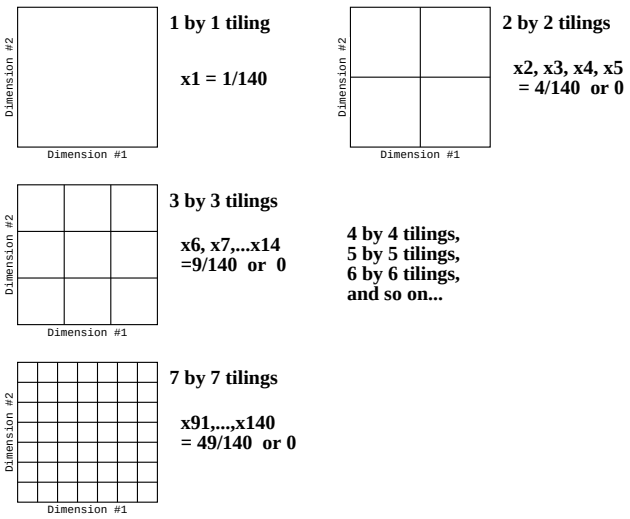


図 6: 2次元状態入力から広域および局所的特徴量を含む特徴ベクトルを生成するタイルコーディング．個々のタイルは特徴ベクトル $(x_1, x_2, \cdots x_{140})$ の各要素に対応．数字 140 は $1^2+2^2+3^2+\cdots+7^2$より．状態入力があるタイル内にある場合，そのタイルは発火，対応する特徴量はある正の値 (例えば 1×1 のタイルは $1/140$ など) をとる．それ以外の要素はゼロ．

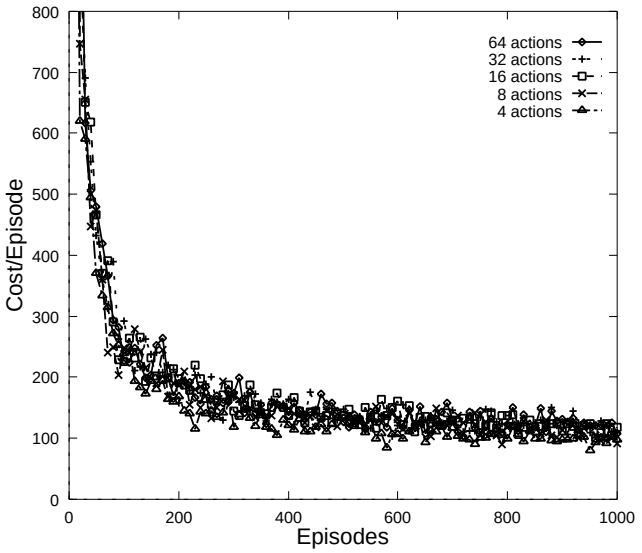


図 7: 2 分木行動選択と actor-critic を用いた手法を 100 試行について平均した学習曲線．$\alpha_v = 0.1$, $\alpha_\pi = 0.0005$, $\lambda_v = 0.9$, $\lambda_p = 0.9$ ．

図 7, 8 は各手法の学習結果を示す (割引率$\gamma = 1$)．提案手法では，行動の個数が増加しても，性能がほとんど変化しない．一方ϵ-greedy 政策を用いた Q-learning や SARSA では，行動の個数によって性能が大きく影響を受ける．

5 まとめ

本稿では 2 分木による確率的政策表現方法を提案した．本手法は，隣接する行動が類似の状態遷移をもたらすことが多い環境にて効果を発揮する．行動選択や適正度の計算は単純かつローカルな情報だけで可能であり，大規模問題への拡張も容易である．提案手法は単に確率的政策表現方法の一つであり，理論的にも単純なため，actor-critic だけでなく REINFORCE アルゴリズム等へも適用できる．

参考文献

Barto, A.G., Sutton, R.S. & Anderson, C.W.: Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC–13, no.5, pp. 834–846 (1983).

Kimura, H. & Kobayashi, S.: An Analysis of Actor/Critic Algorithms using Eligibility Traces: Reinforcement Learning with Imperfect Value Function, *15th International Conference on Machine Learning*, pp.278–286 (1998).

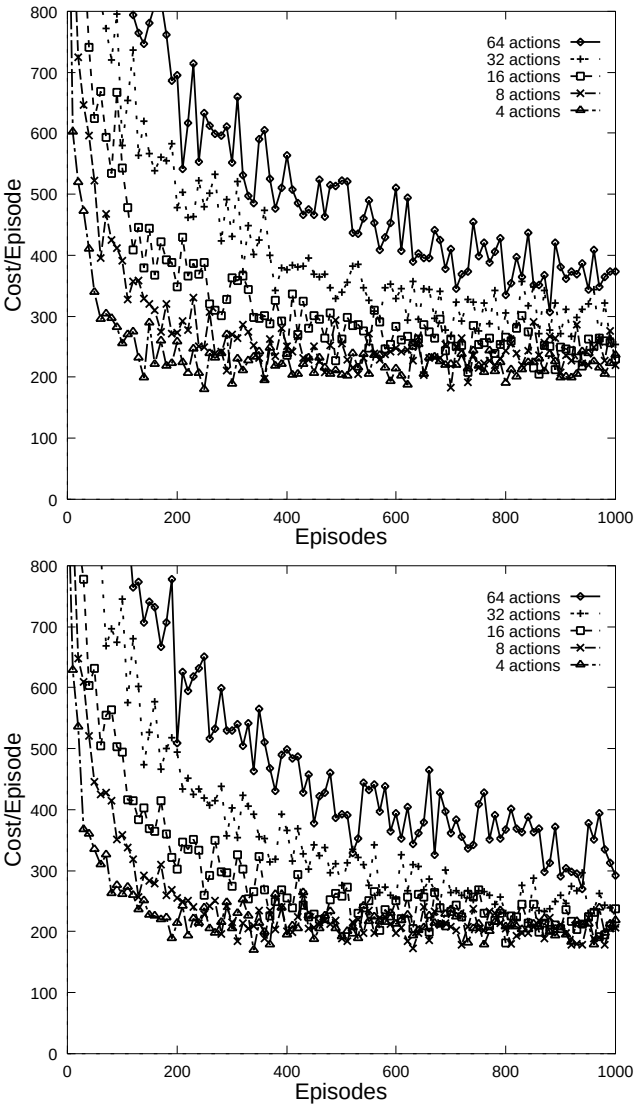


図 8: Q-learning とϵ-greedy 政策を用いた場合 (上) および SARSA(λ) とϵ-greedy 政策を用いた場合 (下) の 100 試行平均の学習曲線．$\epsilon = 0.1$，学習率$\alpha = 0.1$ ．

Sutton, R.S.: Generalization in reinforcement learning: Successful examples using sparse coarse coding, *Advances in Neural Information Processing Systems 8 (NIPS8)*, pp.1038–1044 (1996).

Sutton, R. S. & Barto, A.: Reinforcement Learning: An Introduction, *A Bradford Book*, The MIT Press (1998).

Sutton, R. S., McAllester, D., Singh, S. & Mansour, Y.: Policy Gradient Methods for Reinforcement Learning with Function Approximation, *Submitted to Advances in Neural Information Processing Systems 12 (NIPS12)*, (2000).

Watkins, C. J. C. H. & Dayan, P.: Technical Note: Q-Learning, *Machine Learning 8*, pp.279–292 (1992).

Williams, R. J.: Simple Statistical Gradient Following Algorithms for Connectionist Reinforcement Learning, *Machine Learning 8*, pp. 229–256 (1992).