# 未来状態の予測を利用したマルチエージェント強化学習法

長行 康男　柴田 直樹　伊藤 実

## 1 まえがき

マルチエージェント系における適応行動の実現は，工学及び認知科学の観点から興味深い課題である．その中でも，学習による適応行動の自律的獲得に関する研究が，強化学習（RL）[1] の発展を契機として近年注目を集めている．

マルチエージェント系において，個々のエージェントが実行する行動の善し悪しは，他エージェントが実行する行動に依存する．この点に注目したマルチエージェント強化学習（MARL）法（マルチエージェント系における強化学習法）として，他エージェントの政策（行動決定関数）を推定しながら学習を進行する手法が提案されている [2] [3] [4]．これら 3 つの MARL 法では，推定した他エージェントの政策を基に他エージェントの未来の行動を予測し，その予測行動を考慮に入れながら学習を進行する．これらの MARL 法は，シングルエージェント系を対象として提案された RL 法である Q 学習 [5] を基盤にしている．そして，学習時に他エージェントの政策（を基に実行される行動）を考慮に入れるため，学習関数である Q 関数を，状態 $s$ と行動 $a$ の関数 $Q(s,a)$ から，状態 $s$ とすべてのエージェントの行動 $a^1,\cdots,a^n$ の関数 $Q(s,a^1,\cdots,a^n)$ に拡張している．しかしながら，このように Q 関数を拡張した場合，学習関数はすべてのエージェントの行動に依存するため，学習空間サイズはエージェント数の増加に対して指数関数的に増加する．強化学習において，学習空間サイズの指数関数的増加は『次元の呪い』と呼ばれ，学習を大幅に遅らせる原因となる．

本稿では，他エージェントの政策を推定しながら学習を進行する MARL 法で，エージェントの行動に依存しない学習関数を利用した新たな MARL 法を提案する．提案する MARL 法は，TD 学習 [6] を基盤にしたもので，学習関数として V 関数 $V(s)$ を用いる．提案する MARL 法では，他エージェントの政策推定に加えて，状態遷移確率関数の推定を行う．そして，推定した状態遷移確率関数と推定した他エージェントの政策を基に未来の状態を予測し，その予測状態における V 関数値を利用しながら強化学習を進行する．

本研究では，マルチエージェント系のモデルとして，先行研究 [2] [3] [4] と同様，マルコフゲームの枠組みを採用する．

本稿では，提案する MARL 法と拡張 Q 関数 $Q(s,a^1,\cdots,a^n)$ を利用した MARL 法 [4] を，マルコフゲームの枠組みでモデル化した 2 体エージェント問題と 3 体エージェント問題に適用する．そして，それらの実験結果を比較することにより，提案手法を評価する．

## 2 マルコフゲーム

マルコフゲームは，マルチエージェント環境における行動決定問題のモデルで，組 $\langle n,S,A^1,\cdots,A^n,T,R^1,\cdots,R^n\rangle$ で定義される．ここで，$n$ は環境内に存在するエージェントの数，$S$ は環境状態の有限集合，$A^k$（$k=1,\cdots,n$）はエージェント $k$

A multi-agent reinforcement learning method with the estimation of futer states.

Yasuo NAGAYUKI, Naoki SHIBATA, Minoru ITO, 奈良先端科学技術大学院大学, Nara Institute of Science and Technology

の行動の有限集合，$T$ は環境状態の遷移確率関数，$R^k$ はエージェント $k$ の報酬関数である．

各離散時間ステップ $t = 0, 1, 2, \cdots$ において，エージェント $k$（$k = 1, \cdots, n$）は，現在の状態 $s_t \in S$ を観測し，行動 $a_t^k \in A^k$ を実行する．そして，状態は $s_{t+1} \in S$ に遷移し，エージェント $k$ は環境から直接報酬 $r_{t+1}^k$ を受け取る．状態の遷移は状態遷移確率関数 $T$ に従う．この関数は時不変な状態遷移確率

$$T(s'|s, a^1, \cdots, a^n) = \Pr(s_{t+1} = s'|s_t = s, a_t^1 = a^1, \cdots, a_t^n = a^n) \quad (1)$$

の集合で表される．ここで，$\Pr(s'|s, a^1, \cdots, a^n)$ は，状態 $s$ でそれぞれのエージェントが行動 $a^1, \cdots, a^n$ を実行したときに状態が $s'$ へ遷移する確率を表す．エージェント $k$ が環境から受け取る直接報酬 $r_{t+1}^k$ も確率的で，その期待値は報酬関数

$$R^k(s, a^1, \cdots, a^n) = E\{r_{t+1}^k = r^k|s_t = s, a_t^1 = a^1, \cdots, a_t^n = a^n\} \quad (2)$$

で表される．ここで，$E\{r^k|s, a^1, \cdots, a^n\}$ は，状態 $s$ でそれぞれのエージェントが行動 $a^1, \cdots, a^n$ を実行したときにエージェント $k$ が受け取る直接報酬 $r^k$ の期待値を表す．

マルコフゲームにおいて，エージェント $k$ の目的は，式 (3) で表される関数を最大にするような自分（エージェント $k$）の政策 $\pi^k$ を見つけることである．

$$V^{\pi^k}(s) = E\left\{\sum_{n=0}^{\infty} \gamma^n r_{t+n+1}^k | s_t = s, \pi^1, \cdots, \pi^n\right\} \quad (3)$$

ここで，政策 $\pi^k$ は各状態において各行動を選択する確率への写像（$\pi^k : S \times A^k \to [0, 1]$），$\gamma \in [0, 1]$ は割引率と呼ばれるパラメータである．$\pi^1, \cdots, \pi^n$ は，それぞれエージェント $1, \cdots, n$ の政策を表す．式 (3) の右辺は，それぞれのエージェントが時刻 $t$ 以降の行動を政策 $\pi^1, \cdots, \pi^n$ に従って選択したときに，エージェント $k$ が受け取る割引報酬の和の期待値を表す．$V^{\pi^k}$ は（エージェント $k$ の）V 関数と呼ばれる．以下，$V^{\pi^k}$ を $V^k$ と書く．

## 3　マルチエージェント強化学習

本稿では，マルコフゲームにおける新たなマルチエージェント強化学習（MARL）法を提案する．ここで，本研究で取り扱うマルコフゲームは，不完全情報ゲーム [7] を仮定し，以下の 3 つの条件を満たすものとする．

- 両エージェントが同期して行動を実行する．
- 両エージェントがお互いの行動を観測できる．
- エージェント間のコミュニケーションは存在しない．

マルコフゲームにおいて，エージェント $k$ の V 関数はすべてのエージェントの政策に依存している．これは，エージェント $k$ の目的（V 関数の最大化）が，他エージェント（エージェント $k$ 以外のエージェント）の政策に依存していることを意味している．この点に注目した MARL 法として，他エージェントの政策を推定しながら学習を進行する手法がいくつか提案されている [2] [3] [4]．これら 3 つの MARL 法では，推定した他エージェントの政策を基に他エージェントが未来に実行する行動を予測し，その予測行動を考慮に入れながら学習を進行する．これらの MARL 法は，シングルエージェント環境を対象として提案された RL 法である Q 学習 [5] を基盤にしている．そして，学習時に他エージェントの政策（を基に実行される行動）を考慮に入れるため，学習関数である Q 関数を，状態 $s$ と行動 $a$ の関数 $Q(s, a)$ から，状態 $s$ とすべてのエージェントの行動 $a^1, \cdots, a^n$ の関数 $Q(s, a^1, \cdots, a^n)$ に拡張している．この拡張した Q 関数を用いた場合の学習空間数は，

$$|S| \times |A^1| \times \cdots \times |A^n| \quad (4)$$

である．ここで，$|S|, |A^1|, \cdots, |A^n|$ は，それぞれ集合 $S, A^1, \cdots, A^n$ の要素数を表す．式 (4) は，Q 関数を $Q(s, a^1, \cdots, a^n)$ と拡張することにより，学習空間がエージェント数の増加に対して指数関数的に増加することを示している．強化学習において，学習空間の指数関数的増加は『次元の呪い』と呼ばれ，学習を大幅に遅らせる原因となる．

本稿では，他エージェントの政策推定を利用した MARL 法で，エージェントの行動に依存しない学習関数を利用した新たな MARL 法を提案する．本稿で提案する MARL 法は，TD 学習 [6] を基盤にしたもので，学習関数として V 関数 $V(s)$ を用いる．ここで，学習空間の数は，エージェント数に関わらず，$|S|$ である．提案する MARL 法では，他エージェントの政策推定に加えて，状態遷移確率関数 $T$ の推定を行う．そして，推定した状態遷移確率関数と，推定した他エージェントの政

策を基に，自分（エージェント $k$）がどの行動を実行すれば，未来にどの状態に遷移するかを式 (5) で予測する．

$$\hat{P}^k(s'|s,a^k) = \sum_{a^{o_1}} \cdots \sum_{a^{o_{n-1}}} \hat{\pi}_k^{o_1}(a^{o_1}|s) \cdots \hat{\pi}_k^{o_{n-1}}(a^{o_{n-1}}|s) \hat{T}^k(s'|s,a^1,\cdots,a^n) \quad (5)$$

ここで，$\hat{P}^k(s'|s,a^k)$ は，エージェント $k$ の観点から，自分が状態 $s$ で行動 $a^k$ を実行したときに状態が $s'$ へ遷移すると予想される確率を表す．$\hat{T}^k(s'|s,a^1,\cdots,a^n)$ は，エージェント $k$ が推定した状態遷移確率関数で，状態 $s$ でそれぞれのエージェントが行動 $a^1,\cdots,a^n$ を実行したときに状態が $s'$ へ遷移すると予想される確率を表す．$\hat{\pi}_k^{o_1},\cdots,\hat{\pi}_k^{o_{n-1}}$ は，エージェント $k$ が推定した他エージェントの政策を表す．ここで，$o_1,\cdots,o_{n-1}$ は，$k$ 以外のエージェント（他エージェント）のいずれか 1 体に対応するものとする．$\hat{\pi}^o(a^o|s)$ という表記は，他エージェント $o$ が状態 $s$ で行動 $a^o$ を実行すると予想される確率を表す．本稿で提案する MARL 法では，式 (5) の $\hat{P}^k$ で予測した未来の環境状態 $s'$ に対する V 関数値 $V^k(s')$ を考慮しながら行動選択を行い，強化学習を進行する．

以下で，本研究で採用する他エージェントの政策の推定法，状態遷移確率関数の推定法，そして，提案する MARL 法を示す．

### 3.1 状態遷移確率関数の推定法

状態遷移確率関数 $T$ の推定には，式 (6) を利用する．

$$\hat{T}^k(s^*|s,a^1,\cdots,a^n) = \frac{C(s,a^1,\cdots,a^n,s^*)}{N(s,a^1,\cdots,a^n)} \quad (6)$$

ここで，$N(s,a^1,\cdots,a^n)$ は，状態 $s$ でそれぞれのエージェントが行動 $a^1,\cdots,a^n$ を実行した回数，$C(s,a^1,\cdots,a^n,s^*)$ は，状態 $s$ でそれぞれのエージェントが行動 $a^1,\cdots,a^n$ を実行したときに状態が $s^*$ へ遷移した回数を表す．

### 3.2 他エージェントの政策の推定法

他エージェントの政策の推定には，我々が以前提案した手法 [4] を採用する．以下に，その手法を示す．

時刻 $t$ において，他エージェント $o$ が状態 $s_t$ で行動 $a_t^o$ を実行したとする．そのとき，状態 $s=s_t$ で実行可能な，すべての行動 $a^o \in A^o$ に対して，式 (7) に従って $\hat{\pi}_k^o$ を更新する．

$$\hat{\pi}_k^o(a^o|s) \leftarrow (1-\theta)\hat{\pi}_k^o(a^o|s) + \begin{cases} \theta & (a^o = a_t^o) \\ 0 & (\text{otherwise}) \end{cases} \quad (7)$$

ここで，$\theta \in [0,1]$ は観測した行動を将来の行動予測時にどれくらい考慮するかを決定するパラメータである．式 (7) の更新則によって $\sum_{a^o \in A^o} \hat{\pi}_k^o(a^o|s) = 1$ が保たれることに注意する．本研究では，それぞれのエージェントが他エージェントの行動集合 $A^o$ について予め知っていることを仮定する．式 (7) の推定法が強化学習エージェントの政策の推定に適していることが実験的に示されている [4]．

### 3.3 未来状態の予測を利用したマルチエージェント強化学習法

以下に，本稿で提案する未来状態の予測を利用した MARL 法の学習の流れを示す．

1. 現在（時刻 $t$ とする）の状態 $s_t \in S$ において，エージェント $k$ は，式 (8) の soft-max 関数で与えられる政策 $\pi^k$ に従って行動 $a^k$ を選択する．

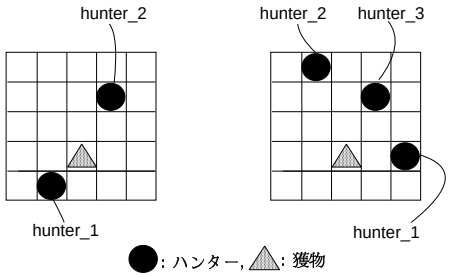
$$\pi^k(a^k|s_t) = \frac{e^{J(s_t,a^k)/\tau}}{\sum_{b \in A^k} e^{J(s_t,b)/\tau}} \quad (8)$$

ここで，政策 $\pi^k(a^k|s)$ は状態 $s$ で行動 $a^k$ を選択する確率を表す．$\tau$ は温度パラメータと呼ばれ，行動選択のランダムさを調整するパラメータである．$J(s,a^k)$ は，V 関数 $V^k$ の $\hat{P}^k$（式 (5)）に関する期待値で，式 (9) で与えられる．

$$J(s,a^k) = \sum_{s^*} \hat{P}^k(s^*|s,a^k)V^k(s^*) \quad (9)$$

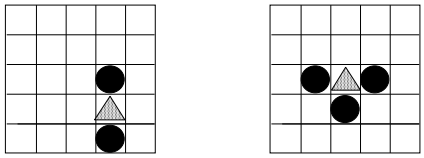
2. エージェント $k$ は，手続き 1 で選択した行動 $a_t^k$ を実行する（ここで，他エージェントも同期して行動 $a_t^{o_1},\cdots,a_t^{o_{n-1}}$ を実行する．環境の状態は，状態遷移確率関数 $T$ に従って $s_t$ から $s_{t+1}$ へ遷移する）．エージェント $k$ は，他エージェントの行動 $a_t^{o_1},\cdots,a_t^{o_{n-1}}$ を観測する．また，環境から直接報酬 $r_{t+1}^k$ を受け取る．
3. エージェント $k$ は，関数 $\hat{\pi}_k^o$（$o=o_1,\cdots,o_{n-1}$）を式 (7) に従って更新し，関数 $N$，関数 $C$ をそれぞれ式 (10)，式 (11) に従って更新する．

$$N(s_t,a_t^1,\cdots,a_t^n) \leftarrow N(s_t,a_t^1,\cdots,a_t^n)+1 \quad (10)$$




（a）ハンターが 2 体の追跡問題　（b）ハンターが 3 体の追跡問題

図 **1**　追跡問題のグリッド空間



（a）ハンターが 2 体の追跡問題　（b）ハンターが 3 体の追跡問題

図 **2**　捕獲状態の例

$$C(s_t, a_t^1, \cdots, a_t^n, s_{t+1}) \leftarrow C(s_t, a_t^1, \cdots, a_t^n, s_{t+1}) + 1 \quad (11)$$

そして，状態 $s_t$ における V 関数値を式 (12) に従って更新する．

$$V^k(s_t) \leftarrow (1-\alpha)V^k(s_t) + \alpha(r_{t+1}^k + \gamma V^k(s_{t+1})) \quad (12)$$

ここで，$\alpha \in (0, 1]$ は学習率と呼ばれるパラメータである（式 (12) の更新式は TD 学習 [6] における更新式と同じものである）．

4. 学習の終了条件を満たしていれば学習終了．そうでなければ $t$ に 1 を加えて，手続き 1 に戻る．

## 4　実験

### 4.1　追跡問題

本研究では，実験に使用するタスクとして追跡問題 [8] を取り上げる．追跡問題は，複数のハンターが獲物を追いかけて捕獲する課題である．以下に，本研究における追跡問題の問題設定を示す．

- 2 次元（5 × 5）のグリッド空間中に，複数のハンターと 1 体の獲物が存在する（図 1）．ここで，グリッド空間の上と下，左と右の境界は繋がっているものとする．
- 本研究では，ハンターを『エージェント』と定義する．本研究では，ハンターが 2 体の場合（図 1（a））と，3 体の場合（図 1（b））の 2 つの実験を行う．
- 各時間ステップ毎に，ハンターと獲物は，それぞれ 1 つの行動を同期して実行する．ここで，ハンターが実行可能な行動は，隣接する上，下，左，右のグリッドへ移動する，現在位置に留まる，の 5 通りとする．また，獲物が実行可能な行動は，隣接する上，右のグリッドへ移動する，現在位置に留まる，の 3 通りとする．
- ハンターの目標は獲物を捕獲することである．ここで，捕獲の定義は，
  - ハンターが 2 体の実験では，『2 体のハンターが獲物を上下，あるいは左右から挟んだ状態』（図 2（a））
  - ハンターが 3 体の実験では，『獲物に隣接する 4 近傍のグリッドうち，いずれか 3 つのグリッドを 3 体のハンターが占有している状態』（図 2（b））

  とする．
- 初期配置から獲物が捕獲されるまでを『1 エピソード』とする．獲物が捕獲されると，ハンターと獲物はグリッド空間中にランダムに初期配置され，新たなエピソードを開始する．
- 環境状態は，それぞれのハンターと獲物の相対位置の組合せ $s = (p^1, \cdots, p^n)$ とする．ここで，$p^i$ $(i = 1, \cdots, n)$ は hunter_$i$ と獲物の相対位置を表す．例えば，図 1（a）では $s = ([1, 1], [-1, -2])$，図 1（b）では $s = ([-2, 0], [1, 2], [-1, -2])$ である．
- 獲物は学習を行なわず，3 通りの行動の中から 1 つの行動を確率的に選択する．実験では，右へ移動する確率と現在位置に留まる確率をそれぞれ $\frac{2}{5}$，上へ移動する確率を $\frac{1}{5}$ としている．この行動選択確率は時不変とする．

以上の問題設定では，報酬関数について言及していないが，報酬関数が式 (2) を満たすように設定された場合，この追跡問題はマルコフゲームの条件を満たす．

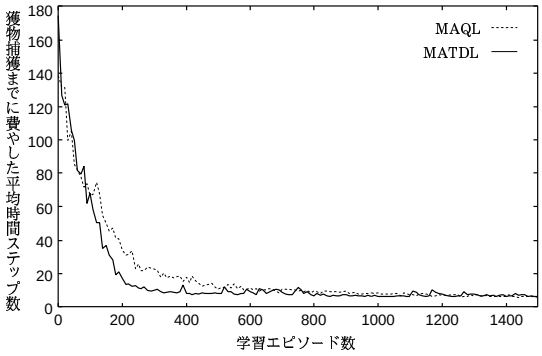



図 3　獲物捕獲までに費やした平均時間ステップ数：ハンターが 2 体の実験.

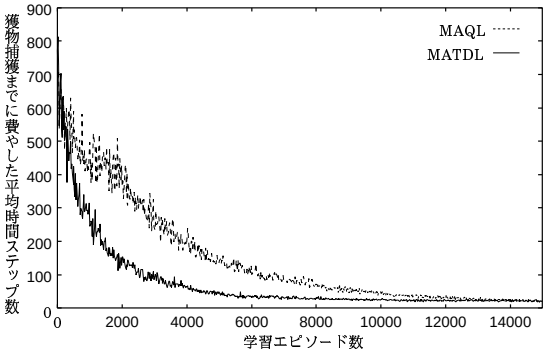



図 4　獲物捕獲までに費やした平均時間ステップ数：ハンターが 3 体の実験.

### 4.2　他エージェントの政策推定を利用したマルチエージェント Q 学習法

提案した MARL 法の性能を評価するための比較対象として，本研究では，他エージェントの政策推定を利用したマルチエージェント Q 学習法 [4]（以下 MAQL と書く）を取り上げる．MAQL では，学習関数に Q 関数 $Q(s, a^1, \cdots, a^n)$ を用いており，3 章で述べたように，学習空間数（$|S| \times |A^1| \times \cdots \times |A^n|$）は，エージェント数の増加に対して指数関数的に増加する．MAQL の学習の流れを以下に示す.

1. 現在（時刻 $t$ とする）の状態 $s_t \in S$ において，エージェント $k$ は，式 (13) で与えられる政策 $\pi^k$ に従って行動 $a^k$ を選択する.
$$\pi^k(s_t, a^k) = \frac{e^{\bar{Q}(s_t, a^k)/\tau}}{\sum_{b \in A^k} e^{\bar{Q}(s_t, b)/\tau}} \qquad (13)$$
ここで，$\bar{Q}(s, a^k)$ は式 (14) で与えられる関数である.
$$\bar{Q}(s, a^k) = \sum_{a^{o_1}} \cdots \sum_{a^{o_{n-1}}} \hat{\pi}_k^{o_1}(a^{o_1}|s) \cdots \hat{\pi}_k^{o_{n-1}}(a^{o_{n-1}}|s) Q^k(s, a^1, \cdots, a^n) \qquad (14)$$
2. エージェント $k$ は，手続き 1 で選択した行動 $a_t^k$ を実行する（ここで，他エージェントも同期して行動 $a_t^{o_1}, \cdots, a_t^{o_{n-1}}$ を実行する．環境の状態は，状態遷移確率関数 $T$ に従って $s_t$ から $s_{t+1}$ へ遷移する）．エージェント $k$ は，他エージェントの行動 $a_t^{o_1}, \cdots, a_t^{o_{n-1}}$ を観測する．また，環境から直接報酬 $r_{t+1}^k$ を受け取る.
3. エージェント $k$ は，他エージェントの政策 $\hat{\pi}_k^o$（$o = o_1, \cdots, o_{n-1}$）を式 (7) に従って推定（更新）する．そして，状態 $s_t$，行動 $a_t^1, \cdots, a_t^n$ における Q 関数値を式 (15) に従って更新する.
$$Q^k(s_t, a_t^1, \cdots, a_t^n) \leftarrow (1-\alpha) Q^k(s_t, a_t^1, \cdots, a_t^n) + \alpha(r_{t+1}^k + \gamma \max_{a^k} \bar{Q}(s_{t+1}, a^k)) \qquad (15)$$
4. 学習の終了条件を満たしていれば学習終了．そうでなければ $t$ に 1 を加えて，手続き 1 に戻る.

### 4.3　実験結果

提案したマルチエージェント強化学習法（以下，MATDL と書く）と MAQL を追跡問題に適用した．ハンターが 2 体の場合の実験結果を図 3 に，3 体の場合の実験結果を図 4 に示す．図の横軸は学習エピソード数，縦軸は 1 エピソード中で獲物捕獲までに費やした平均時間ステップ数を表す．図の結果は，10 学習エピソード毎に，そのときまでの学習性能を評価するため，初期配置を変えた 100 評価エピソード（このエピソードでは学習を行なわない）の実験を行ない，その平均時間ステップ数を示したものである．2 つの学習法（MATDL と MAQL）で使用した学習パラメータは $\alpha = 0.3 \times decay^{num\_ep}$，$\gamma = 0.9$，$\tau = 0.1 \times decay^{num\_ep}$，$\theta = 0.5 \times decay^{num\_ep}$ である．ここで，$decay$ は減衰係数，$num\_ep$ は学習エピソード数を表す．減衰系数 $decay$ は，ハンター数が 2 体の実験では $decay = 0.9977$，3 体の実験では $decay = 0.99977$ としている．減衰係数 0.9977，0.99977 は，それぞれ $0.9977^{1000} \approx 0.1$，$0.99977^{10000} \approx 0.1$ となるように選ばれた値である．エージェントが環境から受け

取る直接報酬 $r^k$ は，獲物捕獲時に $r^k = 1.0$，それ以外の時に $r^k = -0.05$ としている．すべての $s \in S$，$s^* \in S$，$a^1 \in A^1, \cdots, a^n \in A^n$ に対して，V関数，Q関数，関数 $N$，関数 $C$ のそれぞれの初期値は，$V^k(s) = 0.0$，$Q^k(s, a^1, \cdots, a^n) = 0.0$，$N(s, a^1, \cdots, a^n) = 0$，$C(s, a^1, \cdots, a^n, s^*) = 0$ としている．また，すべての $s \in S$，$a^o \in A^o$ $(o = o_1, \cdots, o_{n-1})$ に対して，関数 $\hat{\pi}_k^o$ の初期値は $\pi_k^o(s, a^o) = 0.2$ としている．

図 3，図 4 の結果は，ハンター数が2体の場合，3体の場合の両方で，MATDL が MAQL よりも学習が速いことを示している．また，獲物捕獲までに費やした平均時間ステップ数の差（MATDL−MAQL）と比率（MATDL:MAQL）は，ハンター数が3体の場合の方が，2体の場合より大きいことを示している．

### 4.4 考察と今後の課題

2つの学習法 MATDL と MAQL の主な相違点は以下の3つである．

1. 学習関数として，MATDL では V 関数 $V(s)$，MAQL では Q 関数 $Q(s, a^1, \cdots, a^n)$ を用いている．
2. MAQL の学習空間は，ハンター数が2体の場合，MATDL の 25（$5^2$）倍，3体の場合，MATDL の 125（$5^3$）倍である．
3. MATDL では状態遷移確率関数 $T$ を式 (6) により陽に推定しているのに対して，MAQL では学習（試行錯誤の経験）を通じて，Q 関数中で陰に推定される．

MATDL が MAQL よりも学習が高速である原因は，これら3つの違いが影響していると考えられるが，獲物捕獲までに費やした平均時間ステップ数の比率（MATDL:MAQL）が，ハンター数が2体の場合より3体の場合の方が大きいことより，項目2の学習空間の増加が大きく寄与していることが予想される．より詳細な調査は今後の課題である．

## 5 あとがき

本研究では，未来状態の予測を利用したマルチエージェント強化学習法を提案した．提案したマルチエージェント強化学習法では，他エージェントの政策と，環境の状態遷移確率関数を推定し，推定したそれらの関数を利用して，どの行動を実行すればどの環境状態に遷移するかを予測した．そして，その予測した環境状態における V 関数値を基にどの行動を実行すればよいかを決定し，強化学習を進行した．提案したマルチエージェント強化学習法では学習関数として V 関数を採用した．提案したマルチエージェント強化学習法をマルコフゲームの枠組みでモデル化した追跡問題に適用し，実験を行った結果，ハンターが2体の場合，3体の場合の両方の実験で，他エージェントの政策推定を利用したマルチエージェント Q 学習法 [4] より学習が高速になった．また，その学習の高速化は，2体エージェント問題よりも，3体エージェント問題の方がより顕著に現れた．学習が高速になった原因の詳細な調査は今後の課題である．

### 参考文献


[1] R. S. Sutton, and A. G. Barto, Reinforcement Learning : An Introduction, MIT Press, Cambridge, Massachusetts, 1998.

[2] M. L. Littman, “Markov games as framework for multi-agent reinforcement learning,” Proc. 11th International Conference on Machine Learning, pp.157-163, New Brunswick, New Jersey, USA, July 1994.

[3] J. Hu, and M. P. Wellman, “Multiagent reinforcement learning: theoretical framework and an algorithm,” Proc. 15th International Conference on Machine Learning, pp.242-250, Madison, Wisconsin USA, July 1998.

[4] Y. Nagayuki, S. Ishii, and K. Doya, ”Multi-agent reinforcement learning : An approach based on the other agent's internal model,” Proc. 4th International Conference on Multi-Agent Systems, pp.215-221, Boston, Massachusetts, USA, July 2000.

[5] C. J. C. H. Watkins, and P. Dayan, “Technical Note Q-Learning,” Machine Learning, vol.8, no.3, pp.279-292, 1992.

[6] R. S. Sutton, “Learning to predict by the methods of temporal differences,” Machine Learning, vol.3, pp.9-44, 1988.

[7] G. Owen, Game Theory: Third edition, Academic Press, San Diego, California, 1995.

[8] M. Benda, V. Jagannathan, and R. Dodhiawalla. “On optimal cooperation of knowledge sources”. Technical Report BCS-G2010-28, Boeing AI Center, 1985.