計測自動制御学会東北支部 第209回研究集会 (2003.6.12)
資料番号 209-9

部分観測マルコフ環境における階層型強化学習$SSS(\lambda)$

Hierarchical Reinforcement Learning $SSS(\lambda)$ in Partially Observable Markovian Environments

○釜谷博行*，阿部健一**

○Hiroyuki Kamaya*, Kenichi Abe**

*八戸工業高等専門学校, **東北大学大学院工学研究科

*Dept. Electrical Eng., Hachinohe National College of Technology,
*Dept. Electrical and Communication Eng., Tohoku University

キーワード ： 部分観測マルコフ決定過程(partially observable Markov decision process),
強化学習 (reinforcement learning)

連絡先： 〒039-1192 八戸市田面木字上野平16–1 八戸工業高等専門学校 電気工学科
釜谷博行，Tel.: (0178)27-7283，Fax.: (0178)27-9379，E-mail: kamaya-e@hachinohe-ct.ac.jp

1. はじめに

数理心理学の分野で動物の学習行動を記述する数学モデルが種々提案され，そのモデルが学習機能をもつ工学システムの構築に応用されるようになった。強化学習[1)]もそのモデルのひとつである。すなわち，強化学習は報酬(あるいは罰)という特別な情報を手掛かりに，エージェントが環境との相互作用を通してあらかじめ定められた明確なゴールを達成するための行動決定戦略を自律的に獲得する学習システムと捉えることができる(Fig. 1)。各時間ステップ$t \in 0, 1, 2, \ldots$において，エージェントは環境状態$s_t \in \mathcal{S}$ を取得し，何らかの行動$a_t \in \mathcal{A}$を選択・実行することで環境状態を変化させ，この状態遷移に対する評価値(報酬)をスカラーの強化信号$r_t \in \mathcal{R}$として受け取る。ここで，学習システムの目標は，強化信号として与えられる報酬の長期間にわたる収益をある評価規範にしたがって

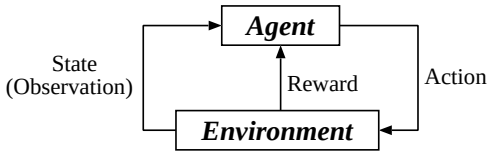

Fig. 1 強化学習モデル.

最大化するような政策π(状態$\mathcal{S}$から行動$\mathcal{A}$への写像) を見出すことといえる。強化学習では，一般的に環境モデルが未知で，しかも正解を教えてくれるような教師は存在しない。また，不確実性のある環境，報酬に遅れが存在する環境にも適用可能であるという特徴をもつ。

多くの強化学習問題において，環境はマルコフ決定過程（Markov decision process；MDP）によりモデル化できるものとして扱っており，エージェントは環境状態を完全に観測できると仮定している。しかし，自律型移動ロボットやマルチエージェントシステムなどの多くの分野では，センサの計

測範囲の制約などにより環境状態を完全に観測することは困難な場合が多い。すなわち，実際に真の環境状態が異なっていたとしてもエージェントはそれらを区別することができない。この問題は，「不完全知覚(incomplete perception)問題」,「知覚エリアジング(perceptual aliasing)問題」,「隠れ状態(hidden state)問題」などとも呼ばれ，これはMDPを拡張した部分観測マルコフ決定過程（partially observable MDP；POMDP）を用いて定式化される[2]。近年，MDP環境下での強化学習アルゴリズムをPOMDPの環境に拡張し適用する試みが多くの研究者によってなされつつある。しかし，時系列データから隠れマルコフモデル（hidden Markov model；HMM）を学習する問題でさえもNP困難であると指摘されていることから，HMMに行動決定が伴うPOMDP学習問題の複雑度は非常に大きなものとなる。このため，POMDPは近似解法によらざるを得ず，対象とする問題のクラスに応じてPOMDPの性質をうまく利用した効率的なアルゴリズムを開発する必要がある。

本研究では，環境モデルが未知で遅れ報酬を伴い，しかも従来の強化学習法では解くことのできない比較的規模の大きなPOMDP学習問題を対象とする階層型強化学習について考える。階層型強化学習の基本的な考えは，多数のQ-モジュールを用意しておき，ある特徴的な観測値（サブゴール）が得られたときにQ-モジュールをうまく切り替えることで，部分観測マルコフ環境を局所的にはマルコフ環境へ近似しながら学習を行うところにある。未知環境を想定しているため，設計者があらかじめ観測空間上でのサブゴールを決定することは難しい。このため，エージェントの行動学習と同時にサブゴールの学習も行う。階層型強化学習の先駆的な研究として，WieringらはQ-モジュールの切り替え機構にQ-学習と類似の方法を用いたHQ-学習[3]を提案している。また，Sunらは行動シーケンスの自動分割(*Self-Segmentation of Sequences*；SSS) アルゴリズム[4]と呼ばれる，より一般的な枠組みの階層型Q-学習を提案している。階層型強化学習は，すべての問題に対して最適性が保証されているわけではない。このため，最適値に近い準最適な政策を学習することを目指す。

本稿では，学習速度の改善およびノイズ環境における頑健性の向上を目指すため，SunらのSSSアルゴリズムにTD(λ)の考えを導入し改良を加えた$SSS(\lambda)$を提案する。比較的規模が大きく複雑なPOMDP環境にある迷路探索問題に適用し，その学習性能について検討する。また，HQ-学習との性能比較を行なう。

2. 強化学習

各時点$t \in \{0, 1, 2, \ldots\}$において，環境状態が$s_t \in S$のとき，その状態観測に基づき，エージェントが行動$a_t \in A$をとったとすると，報酬$r_t$を受け取り，環境状態は未知の遷移確率でつぎの状態$s_{t+1}$に遷移する。エージェントの目標は，報酬の割引期待利得$E\{\sum_{t=0}^{\infty} \gamma^t r_t\}$ を最大にすることである。ここで，$\gamma (0 \leq \gamma \leq 1)$は割引率を表わす。

強化学習では，ある状態 s において行動 a を選択するときの評価値をQ値と呼び，$Q(s, a)$で表わす。Q値の大きさ応じて，エージェントは状態s において実行すべき行動aを決定する。環境と対峙したエージェントは試行錯誤を繰り返しながら，割引期待利得の最大化を目的として，各時点で得られる報酬r_tに基づいてQ値を更新していく。強化学習アルゴリズムとしてTD法[5]，Q-学習[6]，Sarsa[1]などが広く知られている。

3. SSSアルゴリズム

SSSの基本的な考えは，状態-行動系列をうまく分割するための知識を設計者があらかじめ与えなくとも，POMDP問題を非マルコフ性を解消する

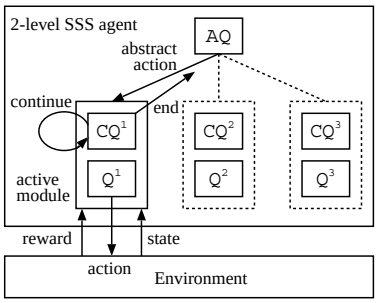

Fig. 2　2階層SSSシステムの概念図.

ようなサブタスクに自動的に分解することで，複数のQエージェントが協力しながら問題を解決するところにある。

SSSアルゴリズムはつぎに示す3種類の学習モジュールから構成される。

- 行動モジュール Q: 各モジュールはサブゴールを目指す局所的な政策を学習する。
- コントローラ CQ: 各Qモジュールに対してそれぞれ一つのコントローラCQが用意される。CQモジュールはある状態においてQモジュールの動作を *"continue"*(継続)すべきか *"end"*(終了)すべきかを学習する。
- 抽象コントローラ AQ: 下位に複数個のQ/CQモジュールをもち，ある状況でアクティブにすべき下位モジュールをどれにすべきか学習する。AQモジュールの行動を抽象行動と呼ぶ。

Fig. 2に上位レベルにある1つの抽象コントローラ AQと下位レベルにある3つのQ/CQモジュールから構成される2階層SSSシステムの概念図を示す。アルゴリズムはつぎの手順で動作する。試行開始時，AQはスタート時の観測状態においてどのQ/CQ対を選択すべきかを決定する(Fig. 2ではQ^1/CQ^1が選択されている)。各時点で1つのQ/CQ対のみがアクティブになり，そのCQにおいて行動選択が行われる。このとき，*"continue"*が選択されると，これと対をなすQモジュールは現在の観測状態にしたがって行動を選択し，外部環境へ出力する。その結果，エージェントは環境から強化信号を受け取り，新たな状態を観測する。つぎに，新たな状態においてアクティブなCQが再び行動選択を行う。このとき*"continue"*が選択されると，同様な手続きが繰り返される。一方，アクティブなCQにおいて*"end"*が選択されると，Qモジュールはその動作を終了し，上位のAQに制御が戻される。AQでは，現在の状態において実行すべきQ/CQ対を新たに1つ決定する。選択されたQ/CQ対がアクティブになり，そのCQにおいて行動選択が行われ，*"continue"*または*"end"*にしたがって，以上述べた一連の手続きが繰り返される。このようにSSSでは，階層構造をオンラインで柔軟に生成できるため，下位のモジュールを再利用できる，永続的なタスクにも適用できるなどの特徴がある。

4.　$SSS(\lambda)$アルゴリズム

学習速度の改善およびノイズ環境における頑健性を向上させるため，SunらのSSSアルゴリズムにTD(λ)の考えを取り入れた新たな学習アルゴリズム$SSS(\lambda)$を提案する。TD(λ)手法を利用することで，現在のTD(temporal-difference)誤差を過去に訪問したすべての状態-行動系列にわたり一度に分配することができ，報酬伝搬を効率的に行なえる。ここでは，政策オン型のアルゴリズムであるSarsa(λ)に基づいて学習システムを構築する。以下に2階層$SSS(\lambda)$アルゴリズムを示す。

1) Initialize Q, CQ, and AQ
2) **Repeat** (for each trial)
3)　η_Q, η_{CQ}, and η_{AQ} are initialized to zero
4)　Observe s
5)　$k \leftarrow AQ$ (AQ selects Q/CQ from s),　$\tilde{s} \leftarrow s$
6)　$a \leftarrow Q^k$ (Q^k selects an action from s)
7)　**Repeat** (for each step of trial)
8)　　Take action a, observe r, s'
9)　　$ca' \leftarrow CQ^k$ (CQ^k selects *continue* or *end* from s')
10)　　Update CQ^k ..... Eq.(10)∼(13)

11) $ca \leftarrow ca'$

12) **if** ca is *continue* **then**
$a' \leftarrow Q^k$ (Q^k selects an action from s')
Update Q^k ..... Eq.(5)∼(8)

13) **else** (that is, ca is *end*)
$k' \leftarrow AQ$ (AQ selects Q/CQ from s')
Update Q^k ..... Eq.(5)(9)(7)(8)
Update CQ^k ..... Eq.(10)(14)(12)(13)
Update AQ ..... Eq.(16)∼(19)
$\tilde{s} \leftarrow s', k \leftarrow k'$
η_Q and η_{CQ} are reset to zero
$a' \leftarrow Q^k$ (Q^k selects an action from s')

14) $s \leftarrow s', a \leftarrow a'$

15) **until** s is the terminal

アルゴリズム中で，kはAQモジュールの抽象行動を表わす。計算量を低く押えるため，*eligibility*の計算はアクティブなQ/CQモジュールに制限する。すなわち，上位レベルのAQモジュールに制御が移った場合には，アクティブなQ/CQモジュールのQ値の更新後に，それらの*eligibility*を0にリセットする。

学習ルールはつぎのようになる。アルゴリズムの表現を簡潔にするため，エージェントの経験組$< s, a, r, s', a' >$に対する通常のSarsa(λ)学習ルールをつぎのように与える。

$$\eta(s, a_i) \leftarrow \begin{cases} 1, & \text{for } a_i = a \\ 0, & \text{for all } a_i \neq a \end{cases} \quad (1)$$

$$\delta = r + \gamma Q(s', a') - Q(s, a), \quad (2)$$

すべてのs_i, a_iに対して，

$$Q(s_i, a_i) \leftarrow Q(s_i, a_i) + \alpha\delta\eta(s_i, a_i), \quad (3)$$

$$\eta(s_i, a_i) \leftarrow \gamma\lambda\eta(s_i, a_i), \quad (4)$$

ここで，α ($0 < \alpha \leq 1$)は学習率，γ ($0 \leq \gamma \leq 1$)は割引率，$\eta(s, a)$は*eligibility trace*関数，λ ($0 \leq \lambda \leq 1$)は*eligibility*係数を表わす。また，sは観測状態である。$SSS(\lambda)$アルゴリズムの各モジュールに対する学習ルールを以下に示す。

Q： アクティブなQ^kモジュールが状態sにおいて行動aを実行したとき，報酬rを取得し，状態がs'に遷移する。このとき，状態s'において，Q^kと対をなすCQ^kによって選択された行動が*"continue"*ならば，Q^kはs'において次の行動a'を選択する。エージェントの経験組$< s, a, r, s', a' >$に基づいて，アクティブなQ^kモジュールがつぎのように更新される。

$$\eta_Q^k(s, a_i) \leftarrow \begin{cases} 1, & \text{for } a_i = a \\ 0, & \text{for all } a_i \neq a \end{cases} \quad (5)$$

$$\delta_Q = r + \gamma Q^k(s', a') - Q^k(s, a), \quad (6)$$

すべてのs_i, a_iに対して，

$$Q^k(s_i, a_i) \leftarrow Q^k(s_i, a_i) + \alpha_Q\delta_Q\eta_Q^k(s_i, a_i), \quad (7)$$

$$\eta_Q^k(s_i, a_i) \leftarrow \gamma\lambda\eta_Q^k(s_i, a_i). \quad (8)$$

一方，状態s'においてCQ^kで選択された行動が*"end"*ならば，制御がAQモジュールへ戻る。AQは状態s'において新たな抽象行動k'を選択する。上位層$AQ(s', k')$のQ値に基づいてアクティブなQ^kモジュールが更新される。これは，(6)式をつぎのように置き換えることで実現される。

$$\delta_Q = r + \gamma AQ(s', k') - Q^k(s, a). \quad (9)$$

CQ： 状態sにおいてアクティブなCQ^kで選択された行動が*"continue"*のとき，それに対応するQ^kモジュールにおいて行動が実行される。その後，報酬rを取得し，状態がs'に遷移する。状態s'においてCQ^kは新たな行動ca'を選択する。このとき，*"continue"*が選択されたならば，エージェントの経験組$< s, ca = continue, r, s', ca' >$に基づいて，アクティブな$CQ^k$モジュールがつぎのように更新される。

$$\eta_{CQ}^k(s, ca_i) \leftarrow \begin{cases} 1, & \text{for } ca_i = ca \\ 0, & \text{for } ca_i \neq ca \end{cases} \quad (10)$$

$$\delta_{CQ} = r + \gamma CQ^k(s', ca') - CQ^k(s, continue), \quad (11)$$

すべてのs_i, ca_iに対して，

$$CQ^k(s_i, ca_i) \leftarrow CQ^k(s_i, ca_i)$$

$$+ \alpha_{CQ}\delta_{CQ}\eta^k_{CQ}(s_i, ca_i), \quad (12)$$

$$\eta^k_{CQ}(s_i, ca_i) \leftarrow \gamma\lambda\eta^k_{CQ}(s_i, ca_i). \quad (13)$$

一方，状態s'においてCQ^kによって選択された行動が"end"ならば，制御がAQモジュールへ戻る。AQは状態s'において新たな抽象行動k'を選択する。上位層$AQ(s', k')$のQ値に基づいてアクティブなCQ^kモジュールが更新される。これは，(11)式をつぎのように置き換えることで実現される。

$$\delta_{CQ} = AQ(s, k') - CQ^k(s, end). \quad (14)$$

AQ： AQがアクティブになった過去の時点$t+1-m$における状態$\tilde{s}$で抽象行動kを実行したとき，現時点$t+1$の状態s'に遷移するまでに得られた(割引)総報酬$\tilde{R}$がつぎの式で計算される。

$$\tilde{R} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ... + \gamma^{t+m-1} r_{t+m-1}, \quad (15)$$

ここで，$m(m > 0)$は状態$\tilde{s}$から状態s'に到達するまでに要したステップ数，$r_t, r_{t+1}, \ldots$は現在までの各時点で得られた報酬を表わす。AQは状態s'において新たな抽象行動k'を選択する。エージェントの経験組$< \tilde{s}, k, \tilde{R}, s', k' >$に基づいて，$AQ$モジュールがつぎのように更新される。

$$\eta_{AQ}(\tilde{s}, k_i) \leftarrow \begin{cases} 1, & \text{for } k_i = k \\ 0, & \text{for all } k_i \neq k \end{cases} \quad (16)$$

$$\delta_{AQ} = \tilde{R} + \gamma^m AQ(s', k') - AQ(\tilde{s}, k), \quad (17)$$

すべてのs_i, k_iに対して，

$$AQ(s_i, k_i) \leftarrow AQ(s_i, k_i) + \alpha_{AQ}\delta_{AQ}\eta_{AQ}(s_i, k_i), \quad (18)$$

$$\eta_{AQ}(s_i, k_i) \leftarrow \gamma^m\lambda\eta_{AQ}(s_i, k_i). \quad (19)$$

エージェントがゴール到達時あるいは制限時間内にゴールに到達できず試行を途中で打ち切る場合には，(6)式，(11)式，(17)式において，つぎの状態s'の価値を0とする。例えば，AQモジュールの更新式はつぎのようになる。

$$\delta_{AQ} = \tilde{R} - AQ(\tilde{s}, k). \quad (20)$$

以上，2階層の$SSS(\lambda)$アルゴリズムについて述べた。このアルゴリズムは3階層以上のシステムへ拡張できる[7)8)]。例えば，3階層システムでは，最下層を$AQ_0(=Q), CQ_0(=CQ)$，中間層を$AQ_1(=AQ)$，CQ_1，最上位層をAQ_2とする。ここで，AQ_2は中間層のAQ_1/CQ_1の中から一つのモジュールを選択するためのモジュール，CQ_1は対応するAQ_1の動作を継続すべきか終了すべきかを決定するためのモジュールとなる。

5. シミュレーション実験

2つのPOMDP迷路探索問題において，提案した$SSS(\lambda)$アルゴリズムの評価実験を行う。すべての問題において，エージェントの目的はスタート地点から出発し，ゴール地点Gへ向かう行動を学習することである。

エージェントの知覚および行動はつぎのようにする。エージェントはすべての時点で自分のグローバルな(x, y)座標を認識できず，現在いるセルに隣接する4方向（西W，南S，東E，北N）のセルに関する情報のみを観測できるものとする。このため，エージェントの観測値は最大でも16種類となり，異なった行動が要求される多くの場所が，エージェントにとって同じものと観測される。各時点でエージェントはW，S，E，Nの4方向の中から選択した1つの方向へ1セルだけ移動できる。壁方向へ移動しようとした場合には移動できず，その場所に留まるものとする。

5.1 行動選択

本実験では，AQモジュールの行動選択にはϵ-greedy法を，Q/CQモジュールの行動選択にはMax-Boltzmann法を用いる[3)]。ϵ-greedy法は，確率 ϵでランダムな行動を，確率 $1-\epsilon$で最大のQ値をもつ*greedy*な行動を選択するものである。また，Max-Boltzmann法は，確率 p_{max} で最大のQ値をもつ

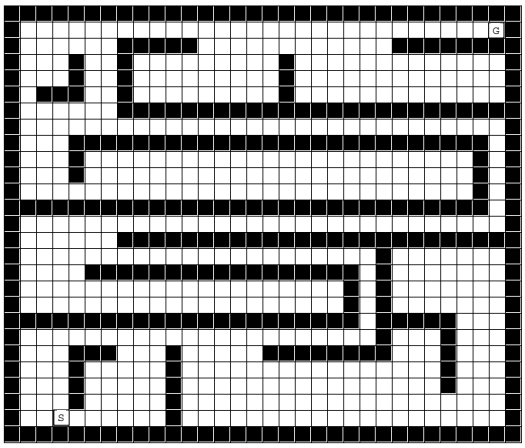

Fig. 3　迷路1.

*greedy*な行動を，確率 $1-p_{max}$ でBoltzmann分布にしたがって行動を選択するものである。Boltzmann分布では，状態s_tにおいて行動a_iを選択する確率は次式で与えられる。

$$\Pr(a_i|s_t) = \frac{e^{\frac{Q(s_t,a_i)}{\tau}}}{\sum_k e^{\frac{Q(s_t,a_k)}{\tau}}}, \qquad (21)$$

ここで，τ は行動選択のランダムさの度合いを決定するパラメータで温度係数と呼ばれる。なお，学習終了時に決定的な方策を得るために，p_{max}をある初期値から1まで徐々に増加させる。

5.2　大きな迷路

ここでは，下位のQ/CQモジュールの再利用性能について検討する。Fig. 3に示す大きさ30×25の迷路[9] について考える。この迷路では，"S"とマークされた左下のスタート地点と"G"とマークされた右上のゴール地点がある。この問題の最適値は131ステップで，かなり大きなものである。

この迷路では，2階層$SSS(\lambda)$アルゴリズムについて評価実験を行う。$\lambda = 0$とし，これ以降$SSS(0)$と記す。報酬はエージェントがゴールに到達できたならば500を，その他の行動実行毎(壁方向の移動で動けない場合も含む)に-0.1を与えた。積極的なサブゴール探索を行うため，AQモジュールの行動選択にはϵ-greedy法を用いた。実験ではϵの初期値を0.1とし，最終試行で0になるように直線的に減少させた。Q/CQモジュールの行動選択にはMax-Boltzmann法を用いた。このとき，p_{max}は初期値0.9から1.0まで試行回数とともに直線的に増加させた。温度係数および学習率は一定とし，それぞれ$\tau_{AQ} = 0.1$，$\tau_{CQ} = \tau_Q = 0.05$，$\alpha_{AQ} = 0.1$，$\alpha_{CQ} = \alpha_Q = 0.05$とした。また，割引率$\gamma$は0.97を用いた。1回のシミュレーション実験は20,000試行からなり，各試行のステップ数の最大値を1,000とした。

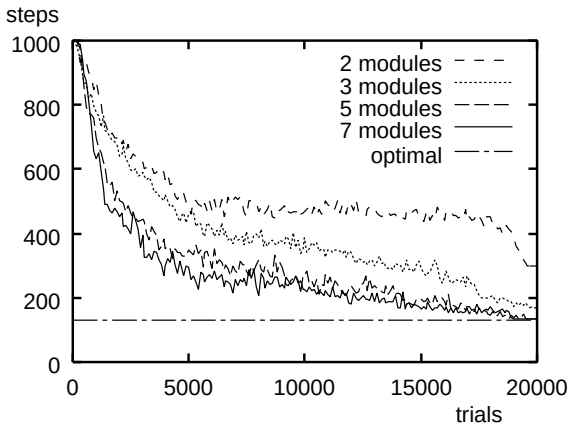

Fig. 4　迷路1における下位のモジュール数に対する平均学習性能.

Fig. 4は，下位の学習器のモジュール数がそれぞれ2，3，5，7の場合の2階層$SSS(0)$アルゴリズムの平均学習性能を示す。このグラフは乱数のシード値を変えた独立な100シミュレーションの平均値をプロットしたものである。横軸は試行回数を，縦軸はゴールまでのステップ数を表わす。グラフを見ると，試行回数とともにステップ数が減少し，学習性能が向上していることがわかる。また，モジュール数の増加につれて学習の収束性能が良くなることがわかる。

Table 1にモジュール数を変えた場合の学習後の性能（100シミュレーション中の最終試行におけるゴールまでの平均ステップ数，最適ステップを学習した割合，タスク成功率，タスク成功時の平均ステップ数）を示す。モジュール数が3以上のときに

Table 1　迷路1における学習後の性能.

モジュール数	平均ステップ数	最適値の割合(%)	成功率(%)	成功時の平均ステップ数
2	298.8	53	81	134.3
3	168.8	62	96	134.2
4	162.1	60	97	136.2
5	135.3	60	100	135.3
6	144.0	63	99	135.4
7	135.7	58	100	135.7

成功率がほぼ100%で約60%の割合で最適値を学習している。ところで，壁方向へ移動する行動は選択しないという制約のもとにHQ-学習の学習性能を調べたところ，成功率がわずか29%であったという報告がある[9]。この結果と比較すると，$SSS(0)$を用いる方法は非常に良好な学習性能を示していると言える。

Fig. 5とFig. 6は下位の学習器のモジュール数が2と7の場合において，最適値を学習したときに発見されたサブゴールと使用モジュールの一例を示している。図中の○印がサブゴールを表わし，直線のそばに記した数字が使用されたモジュールの識別番号を示す。モジュール数が2の場合には，$0 \to 1 \to 0 \to 1 \to 0 \to 1$の順番でモジュールが使われている。このときサブゴール数は5個であるが，2つのモジュールを再利用により交互に使用することで効率良く学習していることが確認できた。また，モジュール数が7の場合には，$2 \to 0 \to 6 \to 3 \to 1 \to 0 \to 6 \to 3 \to 1 \to 0 \to 6$ の順番でモジュールが使われている。このときサブゴール数は10個であるが，4つのモジュールが再利用により順序良く使用され，合計で5つのモジュールを用いることで最適値を実現している。このように，モジュール数を増やしても，最適値の実現には必ずしもすべてのモジュールが使われるわけではないということも確認できた。また，下位の学習器のモジュール数を増やすことで，学習性能が向上することも確認できた。

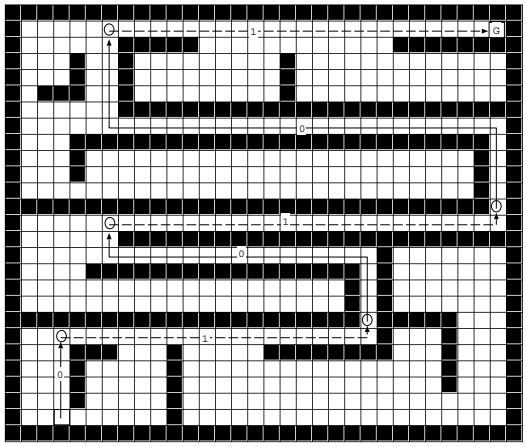

Fig. 5　モジュール数が2の場合に発見されたサブゴールと使用モジュールの例.

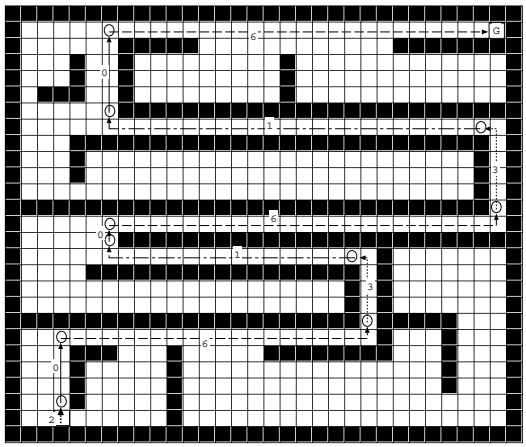

Fig. 6　モジュール数が7の場合に発見されたサブゴールと使用モジュールの例.

5.3　鍵・ドア問題

実験環境として，26×23の部分観測迷路[3]について考える(Fig. 7)。エージェントの目的は，左下のスタート地点Sから出発し，右上のK地点で鍵を取得し，右下のゴール地点Gへ向かう行動を学習

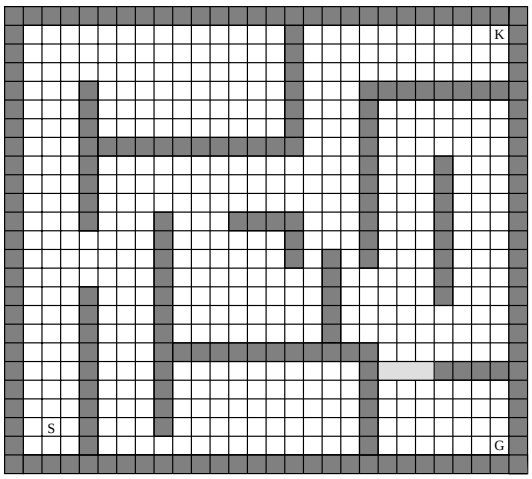

Fig. 7　鍵・ドア問題.

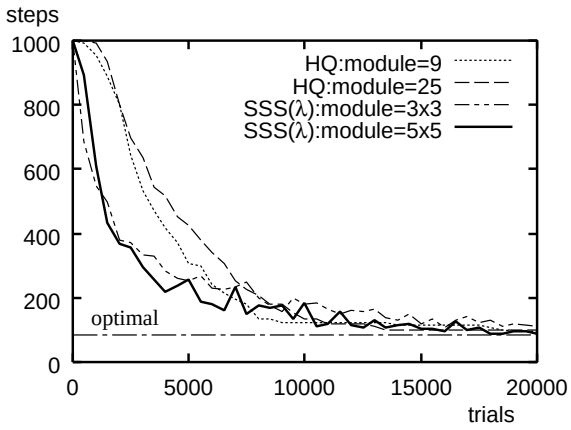

Fig. 8　鍵・ドア問題における平均学習性能(HQ-学習と3階層$SSS(\lambda)$).

することである。迷路中の斜線で示した3つのセルがドアを表わす。エージェントが鍵をもたない場合，ドアは壁として認識される。鍵をもっている場合，自由空間となり通過できる。しかし，エージェントは鍵やドアなどを知覚できず，また，鍵の取得やドアの通過に際しては報酬は与えられない。この問題の最適値は83ステップである。

報酬として，エージェントがゴールに到達できたならば500を，その他の移動毎(壁方向へ移動しようとして動けない場合も含む)に-0.1を与える。

本問題では，3階層$SSS(\lambda)$とHQ-学習について学習実験を行う。3階層$SSS(\lambda)$のパラメータはつぎのように設定した。すべてのモジュールにおいて一律に$\lambda = 0.9$とし，割引率γは0.97を用いた。学習率は$\alpha_{AQ_0} = \alpha_{CQ_0} = 0.05$，$\alpha_{AQ_1} = \alpha_{CQ_1} = \alpha_{AQ_2} = 0.2$とし，学習中は一定とした。積極的な探索を行うため，行動選択においてAQ_2, AQ_1にϵ-greedy法を，CQ_1，AQ_0，CQ_0にMax-Boltzmann法を用いた。ϵ-greedy 法ではϵの初期値を0.1とし，最終試行で0になるように直線的に減少させた。また，Max-Boltzmann法のp_{max}は初期値0.9から1.0まで試行回数とともに直線的に増加させた。温度係数は$\tau_{AQ_0} = \tau_{CQ_0} = 0.1$，$\tau_{CQ_1} = 0.2$とし，学習中は一定とした。

HQ-学習のパラメータは，文献[3]と同一の値に設定した。つまり，サブゴール数を16，$\lambda = 0.9$とし，割引率γは1.0を用いた。学習率は，$\alpha_Q = 0.05$，$\alpha_{HQ} = 0.01$とし，学習中は一定とした。また，Max-Boltzmann法のp_{max}は初期値0.4から0.8まで試行回数とともに直線的に増加させた。温度係数は0.2とし，学習中は一定とした。

1回のシミュレーション実験は20,000試行からなり，各試行のステップ数の最大値を1,000とした。ノイズなしの場合と行動ノイズのある場合について実験を行った。

5.3.1　ノイズなしの場合

Fig. 8は，学習器のモジュール数が9，25のHQ-学習と下位の学習器のモジュール数が3×3，5×5の$SSS(\lambda)$の平均学習性能を示す。これらのグラフは乱数のシード値を変えた独立な100シミュレーションの平均値を500試行毎にプロットしたものである。結果から$SSS(\lambda)$の収束速度が速いことがわかる。

Table 2，Table 3は，2つの学習システムにおいてモジュール数を変えた場合の学習後の性能を示している。どちらの学習システムでも，モジュール数が増えるにしたがい学習性能の向上していることがわかる。HQ-学習のモジュール数が25のところと$SSS(\lambda)$のモジュール数が5×5のところで比較

Table 2　鍵・ドア問題における学習後の性能(HQ-学習).

モジュール数	平均ステップ数	最適値の割合(%)	成功率(%)	成功時の平均ステップ数
3	232.6	15	84	86.4
4	115.5	15	97	88.1
5	124.1	14	96	87.6
9	97.7	13	99	88.6
16	88.9	8	100	88.9
25	99.9	11	99	90.8

Table 3　鍵・ドア問題における学習後の性能(3階層$SSS(\lambda)$).

モジュール数	平均ステップ数	最適値の割合(%)	成功率(%)	成功時の平均ステップ数
2×2	237.6	8	84	101.0
3×3	111.5	19	98	93.4
4×4	99.0	28	99	89.9
5×5	88.3	34	100	88.3
6×6	88.0	21	100	88.0
7×7	88.5	17	100	88.5

すると，どちらも成功率がほぼ100%であるが，最適値の割合は，それぞれ11%，34%，また，平均ステップ数で比較するとそれぞれ99.9，88.3である。このことから，$SSS(\lambda)$の学習性能が高いことが確認できた。

$SSS(\lambda)$において最適行動を学習したときに使用される下位のモジュール数を調べてみたところ，3〜4モジュールが学習結果の大部分を占めた。したがって，学習システムにたくさんのモジュールを用意しても，最終的には少ないモジュールで最適行動を学習していることが確認できた。

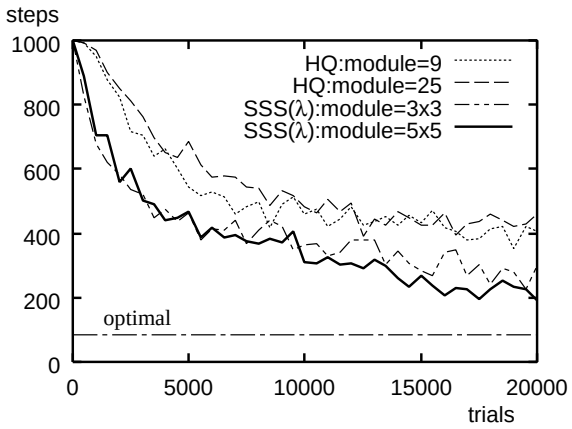

Fig. 9　行動ノイズのある鍵・ドア問題における平均学習性能(HQ-学習と3階層$SSS(\lambda)$).

5.3.2　行動ノイズのある場合

10%の確率でエージェントが選択した行動と無関係にランダムな行動が実行される行動ノイズのある環境で実験を行なう。Fig. 9は，学習器のモジュール数が9，25のHQ-学習と下位の学習器のモジュール数が3×3，5×5の3階層$SSS(\lambda)$の平均学習性能である。試行回数の増加とともにゴールまでのステップ数が減少していることから，ノイズのある環境でもゴールへ到達するための行動を学習している。しかし，行動ノイズの影響で学習性能がかなり悪化していることがわかる。HQ-学習と比較すると，$SSS(\lambda)$の学習性能が良好であることが確認できた。

Table 4，Table 5は，2つの学習システムにおいてモジュール数を変えた場合の学習後の性能を示している。モジュール数の大きいところでタスク成功時の平均ステップ数を比較すると，$SSS(\lambda)$は約151〜160，HQ-学習は約310〜347である。このことから，$SSS(\lambda)$はHQ-学習と比較して，ノイズ

Table 4　行動ノイズのある鍵・ドア問題における学習後の性能(HQ-学習).

モジュール数	平均ステップ数	最適値の割合(%)	成功率(%)	成功時の平均ステップ数
5	406.9	0	86	310.3
9	405.3	0	88	324.2
16	393.0	0	91	333.0
25	458.2	0	83	347.2

Table 5　行動ノイズのある鍵・ドア問題における学習後の性能(3階層$SSS(\lambda)$).

モジュール数	平均ステップ数	最適値の割合(%)	成功率(%)	成功時の平均ステップ数
3×3	299.0	0	86	184.8
4×4	219.2	0	93	160.4
5×5	192.7	0	96	159.0
6×6	252.1	0	90	169.0
7×7	227.9	0	91	151.5

環境における学習性能がかなり高いことがわかる。

6. おわりに

本稿では，SunらのSSSアルゴリズムにTD(λ)の考えを導入することで，部分観測問題における学習性能の向上を目指した新たな階層型強化学習アルゴリズム$SSS(\lambda)$を提案した。$SSS(\lambda)$の学習性能を2つの部分観測迷路探索問題においてシミュレーションにより実験的に評価した。実験結果から，$SSS(\lambda)$はHQ-学習に比べて学習速度の向上が図れることを確認した。また，学習システムのモジュール数を増やすことで，学習性能を改善できることを示した。さらに，$SSS(\lambda)$はノイズなどの非決定性をもつ比較的規模の大きな部分観測問題においても，良好な政策を学習できることを明らかにした。HQ-学習と比較しても，ノイズ環境での学習性能が高いことが確認できた。

参考文献

1) Sutton, R. S. and Barto, A. G.: *Reinforcement Learning : An Introduction.*, MIT Press, 1998.

2) Kaelbling, L. P., Littman, M. L. and Moore, A. W.: "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, **4**, pp.237–285, 1996.

3) Wiering, M. and Schmidhuber, J.: "HQ-learning," *Adaptive Behavior*, **6**-2, pp.219–246, 1997.

4) Sun, R. and Sessions, C.: "Self-Segmentation of Sequences: Automatic Formation of Hierarchies of Sequential Behaviors," *IEEE Trans.*, SMC-B-**30**-3, pp.403–418, 2000.

5) R. S. Sutton: "Learning to predict by the methods of temporal differences," *Machine Learning*, **3**, pp.9–44, 1988.

6) C. J. C. H. Watkins and P. Dayan: "Q-learning," *Machine Learning*, **8**, pp.279–292, 1992.

7) H.Kamaya and K.Abe, "Hierarchical Self-Segmentation Algorithms for Q-learning in Non-Markovian Environments," *Proc. of the ACIS 2nd International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD'01)*, pp. 55–62, 2001.

8) H.Kamaya and K.Abe, "Self-Segmentation of Sequences Algorithm with Eligibility Traces in POMDPs," *Proc. of the 4th Asian Control Conference (ASCC 2002)*, pp. 408–413, Singapore, September, 2002.

9) 山城啓秀，上野敦志，武田英明: "遅れ報酬に基づく遺伝的アルゴリズムによる部分観測マルコフ決定問題の解決手法," 電子情報通信学会論文誌 D-I, **J84-D-I**-12, pp.1635–1647, 2001.