協調的学習オートマトンチームモデル
に関する研究

2017年7月

汪 岑

（千葉大学審査学位論文）

協調的学習オートマトンチームモデルに関する研究

2017年7月

汪 岑

概 要

学習オートマトンとは未知環境上で動作する確率オートマトンのことである。学習オートマトンに関する研究は，その理論の明快さ，実装しやすさ，モデルフリー性，収束性などの利点から注目され，理論面では，自律分散システムとしてのマルチエージェントシステムのモデル化手法，学習セルオートマトンへの展開，応用面では，通信ネットワークの制御問題，群知能ロボット制御問題，超大規模計算システムの負荷分散問題など，様々な方面から研究が行われている。

学習オートマトンチームモデルは N 複数個の確率オートマトンとひとつ，または複数の未知環境からなり，マルチエージェントシステムのモデル化の一手法として提案されている。現在までに提案された学習オートマトンチームモデルはチーム型非協力ゲームに帰着され，混合戦略空間上にナッシュ均衡点，すなわち，最適混合戦略が存在することが証明されている。学習オートマトンチームモデルの簡単な組合せ最適化問題や，ネットワークの分散的制御問題などへの応用では，学習オートマトンチームモデル中の各確率オートマトンの制御機構は完全並列分散的に動作できるので，システム全体の目標が達成されやすい。

一方，マルチエージェントシステムの立場で考える場合，エージェント同士の情報交換や，協調動作の取得，コミュニティの自律的形成などが要求される。これらの一般性のある分野へ学習オートマトン チームモデルを応用するためには，従来のモデルに協調学習機能を導入する必要がある。

本研究では，マルチエージェントシステムの並列分散動作のみならず，協調動作をもモデル化できる協調学習機能を持つ学習オートマトンチームモデルを提案する。まず，協調的学習オートマトンチームモデルの詳細な定義を与え，学習の収束性について議論する。さらに，マルチエージェントシステムの協調動作の例題として，ネットワークにおける分散型最短経路探索問題と HighSpeed TCP 輻輳制御問題を取り上げて，提案モデルを応用する。計算機実験の結果，前者では各ノード上に置かれた確率オートマトン間の協調関係が局所的に確立され，大域的目標を実現でき，また，後者では通信回線容量の増大時における帯域使用率の大幅なが改善を達成でき，提案モデルの有効性が確認できた。

A study on a Collaborative Learning Automata Team Model

Cen Wang

Abstract : The Learning Automaton(LA) is a Stochastic Automaton(SA) operates on a Random Environment(RE). Research on The Learning Automata is drawing attention from the advantages such as clarity of the theory, ease of implementation, model-freeness, and convergence. In theory, modeling method of multi-agent system as autonomous distributed system, development to learning cellular automaton. On the application side, research is being conducted from various aspects such as control problem of communication network, group intelligent robot control problem, and load balancing problem of very large scale calculation system.

The Learning Automaton Team Model consists of multiple Stochastic Automata, one or more Random Environments, and is proposed as a modeling method of multi agent system. The Learning Automaton Team Model proposed to date is reduced to a team type non-cooperative game and Nash equilibrium point on mixed strategy space. In other words, it has been proven that an optimal mixing strategy exists. In application to a simple combination optimization problem of The Learning Automaton Team Model and distributed control problem of network, since the control mechanism of each stochastic automaton in the Learning Automaton team model can operate in a completely parallel distributed manner, the goal of the entire system is easily achieved.

On the other hand, when considering from the standpoint of multi agent system, it is required to exchange information between agents, acquire cooperative actions, and autonomously form communities, In order to apply The Learning Automaton Team Model to a field of generality. it is necessary to introduce cooperative learning function in the collaborative model.

In this research, we propose a Learning Automaton Team Model with collaborative learning function that can model not only parallel distributed operation but also cooperative behavior of multi agent system. First, we give detailed definition of The Collaborative Learning Automaton Team Model and discuss convergence of learning. Furthermore, as an example of cooperative operation of the multi-agent system, the shortest path search problem and High Speed TCP congestion control problem is taken up and the proposed model is applied. In the shortest path search problem, cooperative relationships between probabilistic automatons placed on each node were locally established, and global targets could be realized. In the High Speed TCP congestion control problem, the improvement of the bandwidth utilization ratio at the time of increase of the communication line capacity can be improved, and the effectiveness of the proposed model can be confirmed.

Key words : Learning automaton, Team model, Reinforcemenet learning, Collaboratiive behavior, Multi-agent system

目 次

第1章

序論

1.1 背景及び目的

学習オートマトン（LA : Learning Automaton）とは未知環境（RE : Random Environment）上で動作する確率オートマトン（SA : Stochastic Automaton）のことである [1–8]。学習オートマトンに関する研究は，その理論の明快さ，実装しやすさ，モデルフリー性，収束性などの利点から注目され，理論面では，自律分散システムとしてのマルチエージェントシステムのモデル化手法，学習セルオートマトンへの展開，応用面では，通信ネットワークの制御問題，群知能ロボット制御問題，超大規模計算システムの負荷分散問題など，様々な方面から研究が行われている [9–11]。

学習オートマトンチームモデル（Learning Automata Team Model）[12–15] は複数個の確率オートマトンとひとつ，または複数の未知環境からなり，マルチエージェントシステムのモデル化の一手法として提案されている。現在までに提案された学習オートマトンチームモデルはチーム型非協力ゲームに帰着され，混合戦略空間上にナッシュ均衡点，すなわち最適混合戦略が存在することが証明されている [13]。学習オートマトンチームモデルの簡単な組合せ最適化問題や，ネットワークの分散型制御問題などへの応用では，学習オートマトンチーム中の各確率オートマトンの制御機構は完全並列分散的に動作できるので，システム全体の目標が達成されやすい [16, 17]。

一方，ロボットサッカーなどに代表される群知能ロボット制御問題，インターネットのバックボーンネットワーク上で利用されるパスベクタ型ルーティング問題などを，マルチエージェントシステムの立場で考える場合，エージェント同士の情報交換や，協調動作の取得，コミュニティの自律的形成などが要求される [12, 18, 19]。これらの一般性のある分野へ学習オートマトンチームモデルを応用するためには，従来の学習オートマトンチームモデルに協調学習機能を導入する必要がある。

本研究では，マルチエージェントシステムの並列分散動作のみならず，協調動作もモデル化できる様々な協調学習機能を持つ学習オートマトンチームモデルを提案し，協調的学習オートマトンチームモデルの詳細な定義を与え，学習の収束性について簡単に述べる。また，実問題において，離散型である協調的学習オートマトンチームモデルは最適動作が得られないことがあるため，連続入力，連続出力の両方また一部分を持つモデルが必要

である。本研究では，連続型学習オートマトンを取り入れたハイブリッド型学習オートマトンチームモデルも提案する。最短経路探索問題 [20–23] と HighSpeed TCP 輻輳制御問題 [24, 25] を用いて，二つの提案手法の有効性を検証する。

1.2 論文構成

以下は本論文の構成について説明する。第 2 章では，学習オートマトン，最短経路探索，TCP 輻輳制御などの基本概念について解説する。第 3 章では，提案する二つの手法，協調的学習オートマトンチームモデル，ハイブリッド型協調的学習オートマトンチームモデルについて述べる。第 4 章では，最短経路探索問題と HighSpeed TCP 輻輳制御問題を用いて，提案手法有効性を検証する。また，比較するため，いくつの予備実験も行う。第 5 章では，本論文の結論である。

第2章

基本概念

2.1 学習オートマトン

2.1.1 マルコフ過程 [26–30]

確率変数間の依存関係により特徴つけられるいくつの重要な確率過程を以下に示す。

- 独立過程

  独立過程は確率の中で最も簡単で，確率変数 $X(t)$ の実現値が x_t のとき，すべての実現値 $(x_1, x_2, \ldots, x_t)$ と時間 $(t \geq 2)$ において条件確率が以下を満たす確率過程のことを独立過程という。

$$P(X(t) = x_t \mid X(1) = x_1, \ldots, X(t-1) = x_{t-1}) = P(X(t) = x_t) \tag{2.1}$$

- マルコフ過程

  確率変数 $X(t)$ の実現値が x_t のとき，すべての実現値 $(x_1, x_2, \ldots, x_t)$ と時間 $(t \geq 2)$ において条件確率が以下を満たす確率過程のことをマルコフ過程という。

$$P(X(t) = x_t \mid X(1) = x_1, \ldots, X(t-1) = x_{t-1}) = P(X(t) = x_t \mid X(t-1) = x_{t-1}) \tag{2.2}$$

  ただし，$t_1 < t_2 < \ldots < t_{n-1} < t_n$ である。マルコフ過程における概念は学習オートマトン理論に関する研究の基礎となる。

- 定常過程

  確率変数 $X(t_n)$ の実現値が x_n のとき，すべての実現値 $(x_1, x_2, \ldots, x_n)$ と時間 $(t_1, t_2, \ldots, t_n)$ において条件確率が以下を満たす確率過程のことを定常過程という。

$$P(X(t_1) = x_1, \ldots, X(t_n) = x_n) = P(X(t_1 + \tau) = x_1, \ldots, X(t_n + \tau) = x_n) \tag{2.3}$$

  ただし，τ は任意の定数である。

- 可逆過程

  確率変数 $X(t_n)$ の実現値が x_n のとき，すべての実現値 $(x_1, x_2, \ldots, x_n)$ と時間 $(t_1, t_2, \ldots, t_n)$ において条件確率が以下を満たす確率過程のことを可逆過程という。

$$P(X(t_1) = x_1, \ldots, X(t_n) = x_n) = P(X(\tau - t_1) = x_1, \ldots, X(\tau - t_n) = x_n) \tag{2.4}$$

ここでは，学習オートマトン理論に関する研究の基礎となるマルコフ過程（Markov process）について簡単に述べる。

マルコフ過程では実現値 x は状態と呼ばれ，確率変数 $X(t)$ が取り得る状態集合 S これを状態空間（statespace）という）が離散集合の場合（discrete or countable set）と連続集合（continuous set）の場合とに分類される。ここで，離散時刻 t での状態を表す確率変数を $X(0), X(1), \ldots, X(t)$ とする。過去の履歴が与えられた下での確率変数 $X(t)$ の確率（密度）関数が 1 つ前の時点の状態 x_{t-1} のみにしか依存しない，すなわち，

$$P(X(t) = x_t \mid X(1) = x_1, \ldots, X(t-1) = x_{t-1}) = P(X(t) = x_t \mid X(t-1) = x_{t-1}) \tag{2.5}$$

であるとき，確率過程 $\{X(t)\}$ はマルコフ性を持つという。また，マルコフ過程は時間変数の取り方が離散時間の場合と連続時間の場合に分類される。

離散状態でかつ離散時間であるとき，確率過程 $X(t)$ はマルコフ連鎖（Markov chain）であるいい，マルコフ連鎖では以下が成り立つ。

$$\begin{aligned} P(X(1) = x_1, \ldots, X(t) = X_t) \quad = \quad & P(X(t) = x_t \mid X(t-1) = x_{t-1}) \\ \cdot \quad & P(X(t-1) = x_{t-1} \mid X(t-2) = x_{t-2}) \\ & \vdots \\ \cdot \quad & P(X(2) = x_2 \mid X(1) = x_1) \\ \cdot \quad & P(X(1) = x_1) \end{aligned} \tag{2.6}$$

これは，状態遷移確率（state-transition probabilities）

$$P(X(t) = x_t \mid X(t-1) = x_{t-1}) \triangleq P(x_t \mid x_{t-1}) \qquad t = 2, 3, \ldots \tag{2.7}$$

と初期状態確率分布

$$P(X(1) = x_1) \triangleq P(x_1) \tag{2.8}$$

とによって特徴付けられる。ここで，状態遷移確率が時刻 t に依存しないマルコフ連鎖を時不変（time-homogeneous）あるいは定常（stationary）マルコフ連鎖と呼ぶ。

状態空間が有限集合 $x_1, x_2, \ldots, x_n$ である定常マルコフ連鎖を考える。ここで，要素が P_{ij} である $n \times n$ 行列として状態遷移確率を状態遷移確率行列（state-transition probabilities matrix）$P = [P_{ij}]$ で表現する。状態 x_i から状態 x_j へ r ステップで行くとき，以下が成り立つ。

$$P(X(t) = x_j \mid X(t-r) = x_i) \triangleq P_{ij}^{(r)} = (P^r)_{ij} \tag{2.9}$$

ここで，$P(r)$ は状態遷移行列 P の r 回の積 $P(r) = P^{(r)} = P \times \ldots \times P$ で与える。チャプマン・コルモゴロフ（Chapman Kolmogorov）の関係は以下のようになる。

$$P^{r+s} = P^{(r)} P^{(s)} \tag{2.10}$$

時点 t における確率変数 $X(t)$ の各状態 x_k での存在確率を $P(X(t) = x_k) \triangleq P_k(t), k = 1, 2, \ldots, n$ とおくとき，以下を得る。

$$\pi(t) = [P_1(t), P_2(t), \ldots, p_n(t)]^T \tag{2.11}$$

ここで，$\pi(t)$ を状態確率ベクトルあるいは状態確率分布とよぶ。状態確率分布については，以下の関係がなりたつ。

$$\pi(t+1) = \pi(t)P \tag{2.12}$$

とくに，

$$\pi = \pi P \tag{2.13}$$

を満たすベクトル π を定常分布（stationary distribution）という。

以下のような正の整数 r が存在するとき，状態 x_i から状態 x_j に可到達（reachable）であるといって，$x_i \to x_j$ と略記する。

$$P_{ij}^{(r)} = P(X(t) = x_j \mid X(t-1) = x_i) > 0 \tag{2.14}$$

ただし，$P^{(0)} \triangleq I$（単位行列）である。また，$x_i \to x_j$ かつ $x_j \to x_i$ が成り立つとき，状態 x_i と状態 x_j とは強連結（strongly connected）であるという。以下のような正の整数の最大公約数を状態 x_i の周期（cyclic）という。

$$P_{ij}^{(r)} = P(X(t) = x_t \mid X(t-r) = x_i) > 0 \tag{2.15}$$

定常エルゴード・マルコフ連鎖の状態は共通の周期を持つことが示され，これを連鎖の周期という。状態集合が強連結集合（任意の状態対が強連結である集合）である定常マルコフ連鎖を定常エルゴード・マルコフ連鎖（constant ergodic Narkov chain）と呼ぶ。このとき，定常分布 π は唯一存在する。この定常分布 π が存在するための条件は，マルコフ連鎖が既約（irreducible），非周期的（aperiodic），再帰的（recurrent）であることを満たすことである。既約とは状態空間 S の任意の状態対 (x_i, x_j) に対し，状態 x_i からスタートしいつかは状態 x_j に至る確率が 0 より大きいことをいい，非周期的とはマルコフ連鎖が幾つかの状態を周期的に訪れることがないことをいう。そして，再帰的とはある状態 x_i からスタートし，また状態 x_i に戻ってくる期待時間が有限なことをいう。また，周期 1 を持つ（非周期的）定常エルゴード・マルコフ連鎖を正則マルコフ連鎖（regular Malkov chain）という。このとき，唯一存在する定常分布は極限分布に一致する。しばしば，この正則マルコフ連鎖をエルゴード・マルコフ連鎖と呼ぶこともある。以上より，マルコフ過程においては次の関係が得ることができる。

正則マルコフ　⊂　エルゴード・マルコフ　⊂　一般のマルコフ

最後に，強化学習でよく見られるマルコフ決定過程（Markov decision process）とは，マルコフ過程において各状態に報酬（reward）が与えられるようなモデルで，短期的ではなく長期的観点から報酬の総和が大きくなるような行動を決定する。学習の問題としては，確率モデルの不定性を解決しながら報酬も増やさねばならないというトレードオフの解決が問題となっている。

2.1.2　学習オートマトン [1–8, 31]

学習オートマトンは，優れた学習性能や高い自律性を持ち学習過程の収束性が理論的に保証されているモデルであり，未知環境上で動作する確率オートマトンとして定義され

る。学習オートマトンにおける学習は，本質的には確率的最大傾斜法のクラスに属する問題である。また，オートマトンの学習に対するアプローチは，与えられる入力に対して出力集合から最適出力を決定することである。これらの出力は未知環境を同定する上で形成されると仮定される。一方，環境では入力に対して出力集合から確率的に対応する出力を選択することにより応答する。

環境

一般的に，環境という言葉は，生命に作用するすべての外部条件や影響または生物の発展の集合体を言及するものとして定義される。学習オートマトンにおいては，オートマトンやオートマトンが動作できる集団に対して一般的に知られていない未知の媒体としての大きなクラスが含まれる。環境を以下のように形式的に定義する。

$$\{Y, C, R\} \tag{2.16}$$

ただし，

Y：RE の入力集合　　$Y = \{y_1, y_2, \ldots, y_s\}$
C：RE のペナルティ応答確率分布　　$C = (c_1, c_2, \ldots, c_s)^T$
R：RE の出力集合　　$R = \{r_1, r_2, \ldots, r_u\}$

である。

環境に対する入力 $y(t)$ は集合 Y に属し，時刻 $t(t = 0, 1, 2, \ldots)$ で環境へ作用する。環境の出力 $r(t)$ は集合 R に属し，以下では 2 値（r_1 と r_2）のうち 1 つをとると仮定する。また，数学的に便利なように r_1 と r_2 はそれぞれ 0 と 1 とする。出力 $r(t) = 0$ は成功（success），好ましい（favorable），リワード（reward）応答として定義し，$r(t) = 1$ は環境の失敗（failure），好ましくない（unfavorable），ペナルティ（penalty）応答として定義する。応答確率分布 C は各要素 c_i がそれぞれ入力アクション y_i と対応するペナルティ応答確率の集合であり，要素 c_i は以下のように環境を特徴付ける。

$$P_r(r(t) = 1 \mid y(t) = y_i) = c_i, \qquad i = 1, 2, \ldots, s \tag{2.17}$$

これは，c_i は入力アクション y_i に対して環境がペナルティ応答を返す確率を意味する。また，入力アクションの複雑さ，リワードまたはペナルティ応答の対応は環境の性質を決定する。

オートマトン

オートマトン理論（Automata Theory）で知られているオートマトンの概念は，抽象システムの広い範囲を取り囲む非常に一般的なものである。もし，Φ を内部状態集合，R を入力集合，Y を出力（アクション）集合，$F : \Phi \times r \to \Phi$ を現在の状態と入力から次の状態へ写像する遷移関数，$G : \Phi \times r \to Y$ を現在の状態と入力から現在の出力への写像とすると，オートマトンは 5 つの構成要素 $\{\Phi, Y, R, F, G\}$ により定義される。

このようなオートマトンは，現在の入力と状態により現在の出力と次の状態を決定する。すなわち，ミーリー型順序機械（mealy sequential machine）として考えられる。しか

しながら，現在の出力が現在の入力には依存せず現在の状態にだけ依存するとき，オートマトンは状態-出力オートマトンとして考えられる。すなわち，ムーア型順序機械（moore sequential machine）として考えられる。そのような場合，上で定義される関数 G は出力関数：$\Phi \to Y$ により置き換えられる。

ここで，もし集合 Φ, R, Y が有限ならば，オートマトンの両タイプは有限オートマトンとして知られている。有限である状態-出力オートマトンは確率順序機械（stochastic sequential machine）と考えることができ，オートマトンを以下のように定義できる。

$$\{\Phi, Y, R, F, G\} \tag{2.18}$$

（1）すべての時刻 t において，オートマトンの状態 $\Phi(t)$ は有限集合 $\Phi = \{\phi_1, \phi_2, \ldots, \phi_m\}$ の要素である。

（2）時刻 t において，オートマトンの出力（アクション）$y(t)$ は有限集合 $Y = \{y_1, y_2, \ldots, y_s\}$ の要素である。

（3）時刻 t において，オートマトンの入力 $r(t)$ は集合 R の要素である。この集合は有限集合だけでなく実線上の区間のような無限集合 $R = \{r_1, r_2, \ldots, r_u\}$ ，$r \in (a, b)$ でも存在できる。ただし，a, b は実数である。

（4）遷移関数 F は，次式のように，時刻 t の状態 $\phi(t)$ と入力より時刻 $(t+1)$ の状態 $\phi(t+1)$ を決定する。

$$\phi(t+1) = F[\phi, r(t)]$$

F は $\Phi \times r \to \Phi$ 写像である。

（5）出力関数 G は，次式のように，時刻 t の状態 $\phi(t)$ より時刻 t のオートマトンの出力を決定する。

$$y(t) = G[\phi(t)]$$

G は $\Phi \to Y$ 写像である。

基本的に，オートマトンは入力系列の入力に対応する出力を出す。時刻 t の観測として，オートマトンの仕事は，非負である整数 $0, 1, 2, \ldots$ の集合上で継続的に次のように考えられる。初期状態 $\phi(0)$ が与えられ，出力関数 G によりアクション $y(0)$ が与えられる。そして，入力 $r(0)$ と遷移関数 F に基づき次の状態 $\phi(1)$ が決定される。これらの動作が再帰的に形成されるとき，状態系列とアクションは与えられる入力系列に対して獲得される。ここで，全時刻 t の状態とアクションは 1 つ前の時刻 $t-1$ の状態と入力にだけ依存し，その他の過去の状態や入力には依存しないことに注意する。

もし，遷移関数 F と出力関数 G が両方とも決定論的写像ならば，オートマトンは決定性オートマトン（Deterministic Automaton）と呼ばれる。そのような場合，与えられた初期状態と入力に対して，それに続く状態とアクションはただ 1 つだけ指定される。

もし，遷移関数 F または出力関数 G が確率的ならば，オートマトンは確率オートマトンと呼ばれる。この場合，一般的に与えられた初期状態と入力系列に従う絶対的な状態とアクションは存在せず，連続した状態とアクションに関連する確率だけを考えることができる。

確率オートマトン　2つの関数 F と G のうち少なくとも1つが確率的であるような確率オートマトンを考える。遷移関数 F が確率的ならば，現在の状態と入力が与えられると次の状態は確率的であり，遷移関数 F は様々な状態に到達する確率を与える。したがって，遷移関数 F は条件付き確率行列 $F(r_1), F(r_2), \ldots, F(r_u)$ として特定できる。ここで，各 $F(r)$ は入力記号 r に基づく $m \times m$ 行列であり，その構成要素は以下のように与えられる。

$$f_{ij}^{r} = Pr\left\{\phi(t+1) = \phi_j \mid \phi(t) = \phi_i, r(t) = r\right\} \quad \begin{matrix} i = 1, 2, \ldots, m \\ j = 1, 2, \ldots, m \\ r = r_1, r_2, \ldots, r_u \end{matrix} \tag{2.19}$$

f_{ij}^{r} はオートマトンが入力 r に基づき状態 ϕ_i から状態 ϕ_j へ移動する確率を表す。

同様に，確率的出力関数 G は $m \times s$ の条件付き確率行列により表現でき，その構成要素は以下のように与えられる。

$$g_{ij} = Pr\left\{y(t) = y_j \mid \phi(t) = \phi_i\right\} \quad \begin{matrix} i = 1, 2, \ldots, m \\ j = 1, 2, \ldots, s \end{matrix} \tag{2.20}$$

g_{ij} は状態 ϕ_i とアクション y_j が対応する確率を表す。

f_{ij}^{r} と g_{ij} は確率的であることから，それらは区間 $[0, 1]$ に閉じている。さらに，オートマトンは初期状態 ϕ_i から始まり，次の時刻には m 状態のうち1つの状態に必ず行かなくてはならない。したがって，以下のような確率を持つ。

$$\sum_{j=1}^{m} f_{ij}^{r} = 1 \qquad for \quad each \quad r \in R \quad i \tag{2.21}$$

同様に，

$$\sum_{j=1}^{s} g_{ij} = 1 \qquad for \quad each \quad i \tag{2.22}$$

上記の2つの等式において，各行列の各行の和は1であり，行列 F と G は確率であることを暗示している。

学習オートマトン　これまでに，環境の構造と特性，そしてオートマトンの構造と特性について個別に考えてきた。ここで，図2.1に示されるようなフィードバックにより2つを接続することを考える。これは環境とオートマトンが1つずつ組み合わされたもっとも簡潔な構成である基本モデルである。環境の出力 $r(t)$ をオートマトンの入力とし，オートマトンのアクション $y(t)$ を環境の入力とする。初期状態 $\phi(0)$ から始まり，オートマトンは対応するアクション $y(0)$ を形成する。この入力に対応する環境の応答は $r(0)$ であり，オートマトンは次の状態 $\phi(1)$ へ変化する。この動作は，状態，アクション，環境の応答の系列として繰り返される。ここで，オートマトンが可変構造確率オートマトンの場合，確率分布 $p(t)$ または遷移行列 $F(r)$ は各時刻で更新され，その性能を改善する。その目標は未知環境に対してもっとも有効な状態を出力しやすくする最適確率分布または最適遷移行列

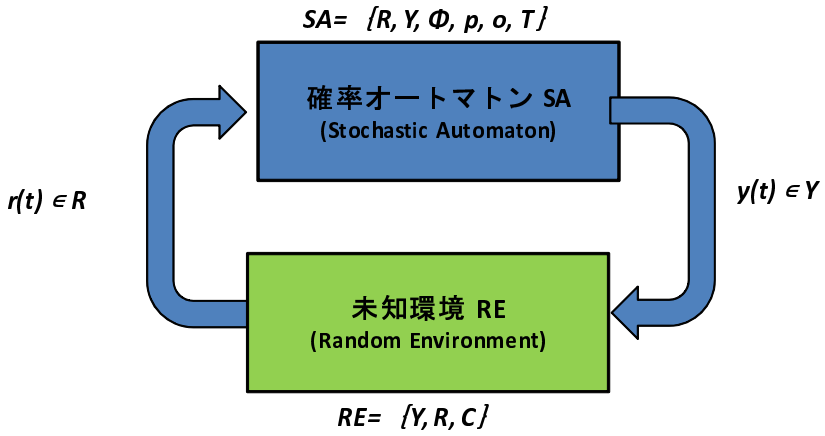

図 2.1: 学習オートマトン・基本表現 [12]

を見出すことである。以下のように未知環境上で動作する確率オートマトンを学習オートマトンとして定義する。

$$LA = \{SA, RE\} \tag{2.23}$$

ただし，SA は確率オートマトン，RE は未知環境である。

確率オートマトンが固定構造であるとき，学習オートマトンは固定構造オートマトンまたは固定構造学習オートマトンと呼ばれる。同様に，確率オートマトンが可変構造であるとき，学習オートマトンは可変構造オートマトンまたは可変構造学習オートマトンと呼ばれる。通常，学習オートマトンといえば可変構造学習オートマトンのことをさすことが多い。

強化法　一般的に，強化法は以下のような作用素 T の形で定義されている [26]。

$$p(t+1) = T[p(t), y(t), r(t)] \tag{2.24}$$

ここで，強化法 T は写像であり，アクション確率分布 $p(t+1)$ はその前回の値 $p(t)$，アクション $y(t)$，入力 $r(t)$ を基に更新される。$p(t+1)$ が $p(t)$ の線形関数あるいは非線形関数であるとき，強化法はそれぞれ線形強化法，非線形強化法と呼ばれる。また，複数の強化法からなるときは複合型強化法とも呼ばれる。強化法を構築するときの基本的アイディアは非常に簡単であり，時刻 t における出力を $y(t) = y_i$ とすると，その出力に対して未知環境からの応答 $r(t)$ がリワード応答であれば y_i を選択する確率 $p_i(t)$ を増やし，ペナルティ応答であれば $p_i(t)$ を減らせば良い。強化法の一般形は，時刻 t での出力を $y(t) = y_i$ としたとき以下のようになる。

$r(t) = 0$ のとき（リワード応答）

$$p_i(t+1) = p_i(t) + \sum_{j=1, j\neq i}^{s} \eta_j(p(t)) \tag{2.25}$$

$$p_j(t+1) = p_j(t) - \eta_j(p(t)) \tag{2.26}$$

$$j \neq i, j = 1, 2, \ldots, s \tag{2.27}$$

$r(t) = 1$ のとき（ペナルティ応答）

$$p_i(t+1) = p_i(t) - \sum_{j=1, j\neq i}^{s} \zeta_j(p(t)) \tag{2.28}$$

$$p_i(t+1) = p_i(t) + \zeta_j(p(t)) \tag{2.29}$$

$$j \neq i, j = 1, 2, \ldots, s \tag{2.30}$$

ただし，$\eta(\cdot)$ と $\zeta(\cdot)$ は正値関数であり $\sum_{i=1}^{s} p_i = 1$ である。

一般的に，強化法の一般形に関して絶対的良好を満たすための必要十分条件は以下に示す 2 つの式により与えられる。

$$\frac{\eta_1(p(t))}{p_1(t)} = \frac{\eta_2(p(t))}{p_2(t)} = \ldots = \frac{\eta_s(p(t))}{p_s(t)} = \lambda(p(t)) \tag{2.31}$$

$$\frac{\zeta_1(p(t))}{p_1(t)} = \frac{\zeta_2(p(t))}{p_2(t)} = \ldots = \frac{\zeta_s(p(t))}{p_s(t)} = \mu(p(t)) \tag{2.32}$$

ただし，$\lambda(\cdot)$ と $\mu(\cdot)$ は以下を満たす任意の連続関数である。

$$0 < \lambda(p(t)) < 1, \quad 1 < \mu(p(t)) < \min_i(\frac{p_i}{1-p_i}), \quad i = 1, 2, \ldots, s \tag{2.33}$$

これは新しい強化法を設計する際の基準を与えている。現在までに提案されているさまざまな強化法は，静的未知環境において $\epsilon-$ 最適を満たしており，その関数 $\eta(\cdot)$ と $\zeta(\cdot)$ は上記の条件を満たしている。

2.2 グラフ理論 [20,21]

グラフは，有限個のノードからなる集合と 2 つノードを接続するいくつかの枝からなる集合によって定義される。定義自体は単純であり，ノードと枝のつながりの様子を図に描くことができるため，応用範囲はきわめて広い。例えば，鉄道や地下鉄など交通網は典型的なグラフであり，インターネットや電力網などインフラも典型的なグラフである。また，対人関係やコミュニケーションなどの関係もグラフの一種である。また，最短経路探索問題はその動作によって，以下のように分類することができる。

- 単一始点最短経路探索問題

  特定の一つのノードから他の全ノードとの間の最短経路探索問題である。

- 2 頂点対最短経路探索問題
  特定の二つのノード間の最短経路探索問題である。本研究では，こちらの問題を用いて，提案手法の有効性を検証する。
- 全点対最短経路探索問題
  グラフの中のあらゆる二つノード の組み合わせについての最短経路探索問題である。

2.3 輻輳制御

2.3.1 TCP の基本概念 [32]

TCP（Transmission Control Protocol）は，IP（Internet Protocol）層上でアプリケーションに対して信頼性のあるストリーム型の全二重通信を実現するトランスポート層のプロトコルである。TCP には，これを実現するために次のような機能が実装されている [32,33]。

2 点間のエンドーエンドを結ぶバイトストリームコネクション　この機能により，TCP は 2 点間のアプリケーションに対してバイト単位の連続したストリーム転送を提供する。これは，アプリケーション側から見ると，TCP という伝送パイプを通して切れ目なくデータが転送されてくることになる。もう一つのトランスポート層プロトコルである UDP（User Datagram Protocol）が提供し，アプリケーションによって書き込まれた単位のデータグラムを転送する方式とは対照的である。TCP では，ストリーム転送を実現するために，データをセグメントと呼ばれるユニットに分割して送出する。セグメントへの分割はアプリケーションで責任を持つ必要はなく，TCP の実装系が自動的に行う。作成されるコネクションは，双方向にデータ転送が可能な全二重通信路となっている。

マルチプレクシング　この機能では，ボート番号によってアプリケーションのデータ通信が多重化される。多重化されることで，一つのサーバが複数のコレクションを同時に処理できる。これは，たとえ同じポート番号を持つサービスさえも多重化が可能であることを示している。

信頼性の確保　セグメントが到着した時に ACK(acknowledgement : 確認応答) を返すことで確保する。ただし，単純な応答確認モデルではネットワークの帯域を有効利用できないので，ウィンド方式を用いて帯域の有効利用を図るとともに，タイムアウトによる再転送処理，シーケンス番号による順序制御などを同時に行う。

フロー制御　TCP コネクションの両端において，受信バッファをあふれさせないようにするための制御のしくみで，基本的にはウィンドウサイズを可変長にして伸縮させること

で行う。同時に，中間ルータの輻輳を制御するために輻輳ウィンドウのアイディアも取り入れられている。

TCP の状態遷移図

ここで，TCP での接続を有限状態オートマトンとして考えた場合の状態遷移図を考える。図 2.2 に状態遷移図を示す。

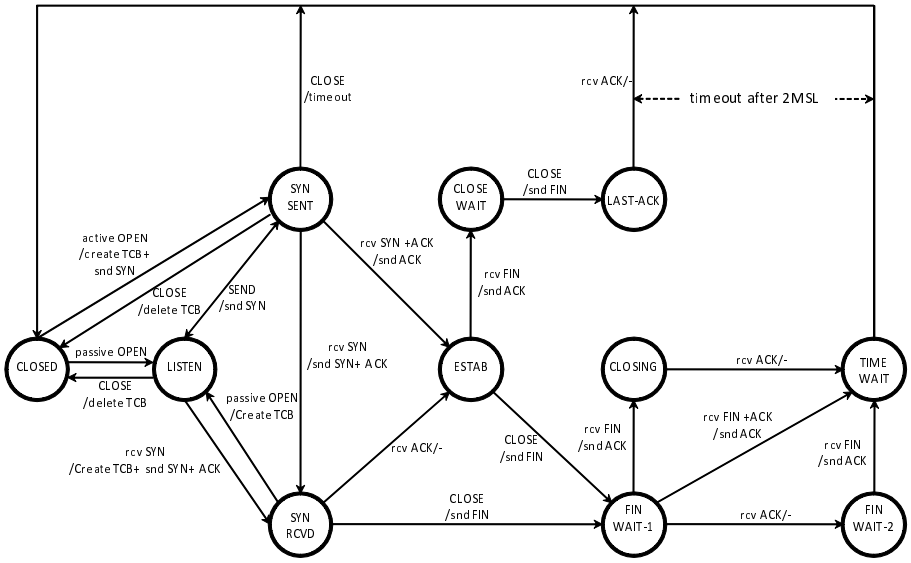

図 2.2: TCP の状態遷移図 [33]

CLOSED　図 2.2 の一番左にある CLOSED は，netstat では表示されない。未使用もしくは使用済みの TCB（Transmission Control Block。TCP の各接続の内部状態を保存しているデータ構造）がこれに該当する。

LISTEN　パッシブ・オープンで，待ち受け状態になっていることを表す。アクティブ・オープンの SYN を受けて「SYN RCVD」へ遷移する。

SYN RCVD　アクティブ・オープンの SYN に対して ACK と SYN で応答し，それに対する ACK を待っている状態である。ACK を受信すると「ESTAB」へ遷移する。

SYN SENT　アクティブ・オープンで，SYN（シーケンス番号同期要求）を送信した状態である。SYN と ACK を受信すれば，ACK を送信して「ESTAB」へ遷移する。

ESTAB　TCP 接続が確立した状態である。データの送受信を行うことができる。FIN を受けたり，上位アプリケーションからクローズが呼び出されたりすると，クローズ処理へ遷移する。

FIN WAIT-1　アクティブ・クローズの最初の段階である。FIN（転送終了要求）を送信し，それに対する応答を待っている状態である。

FIN WAIT-2　送信した FIN に対する ACK を受け取った状態である。送信側のクローズ処理が終了し，相手からの FIN の受信を待っている状態である。

CLOSING　アクティブ・クローズで FIN を送信した後，ACK が戻ってくるよりも先に，相手からも FIN を受けた状態である。両方でほぼ同時にアクティブ・クローズ処理を開始するとこの状態になる。送信した FIN の ACK を待って,「TIME WAIT」へ遷移する。

TIME WAIT　「CLOSING」で ACK を受けた状態である。アクティブ・クローズ後のタイムアウト待ち状態である。同じシーケンス番号やポート番号などを再利用しないように，しばらく待ってから（ネットワーク上で遅れていたパケットがこの時間内に到着する可能性があるので，それと衝突しないように待つ）,「CLOSED」へ遷移して終了する。

CLOSE WAIT　パッシブ・クローズの状態である。送信側に FIN を送信して「LAST-ACK」へ遷移する。

LAST-ACK　「CLOSE WAIT」で送信した FIN に対する ACK を待つ状態である。ACK の受信後,「CLOSED」へ遷移する。

TCP コネクションの確立

TCP コネクションの確立は，スリーウェイハンドシェイク（3WHS : three-way handshaking）と呼ばれるセグメントのやり取りにて行われる。3WHS とは図 2.3 に示したように，1 往復半，すなわち三つのセグメントのやり取りにてコネクションを確立することを指す。以下，図 2.2 と図 2.3 を参照しながらその三つのステップについて説明する。

(1) クライアントからのコネクションの要求（SYN によるシーケンス番号の同期を含む）
TCP コネクションを確立しようとするクライアントは，まず SYN フラグの立ったセグメントを送る。このとき，クライアントはサーバ側のサービスで使われるポート番号（一般にはウエルノンポート番号が使われる）を知っている必要がある。また，クライアントがサーバへ送るために用いられるシーケンス番号も同時に指定する。シーケンス番号は任意に選ばれる。

(2) サーバからの ACK による応答と SYN によるシーケンス番号の同期　クライアントからの SYN フラグによるコネクション確立のセグメントが到着すると，サーバでは確認応答とサーバからクライアントへの通信のためのシーケンス番号の通知が行われる。このために，サーバからクライアントに送られるセグメントには SYN と ACK の両方のフラグが立てられる。シーケンス番号にはサーバで任意に選ばれた初期値が入り，確認応答番号としては，クライアントから通知された番号の値+1 を設定する。

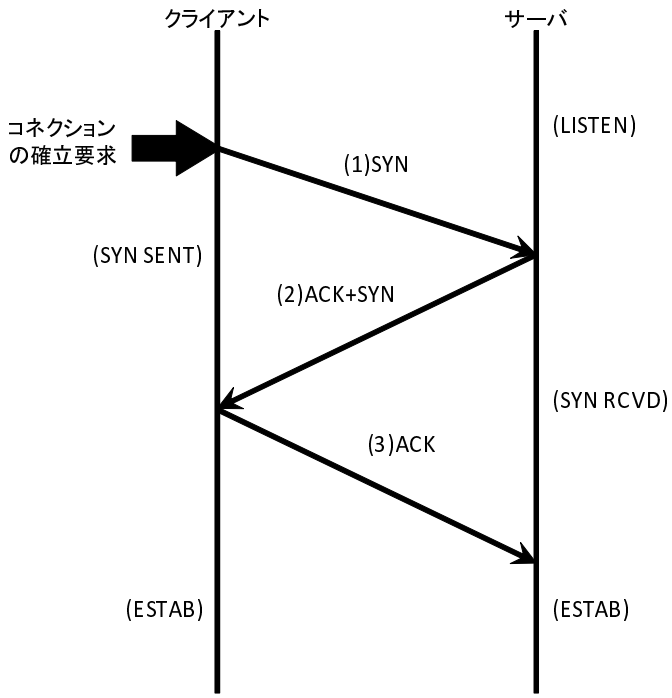

図 2.3: TCP コネクションの確立フロー [33]

(3) クライアントからの ACK による応答　サーバからの ACK+SYN のフラグの立ったセグメントを受け取ると，クライアントでは，それに対する ACK フラグの立ったセグメントを返す。

このように，クライアントは (1) と (2)，サーバは (2) と (3) のセグメントの送受信で，互いのシーケンス番号を交換し，無事にコネクションを確立できる。つまり，3 回のセグメントの送受信があるので，スリーウェイハンドシェイクと呼ばれる。

TCP コネクションの解放

TCP のコネクションの解放は，コネクションの確立時ほど単純ではない。図 2.2 の状態遷移図に示す通り，ESTAB から CLOSE を受けて FIN フラグを立てたセグメントを送り終了動作に入る。しかし，実際の終了処理にはいくつもの状態が存在する。これは，コネクションが解放されるに当たって種々の局面が存在するからである，コネクションの解放手順にはいくつかのバターンがある。まずは正常な解放バターンから見ていく。

この場合のセグメントの転送は図 2.4 のようになる。これは，3WHS の変形のような形をしているが，実際には全二重の通信路を片方ずつ解放するために，全部で 4 つのステップによって解放が行われることを示している。

解放の手順はクライアントからでもサーバからでも始められるが，どちらの場合でも，必ずこの 4 つのステップを踏む。図 2.4 はクライアント側からコネクションを解放する場合の手順を示している。以下に解放の手順を示す。

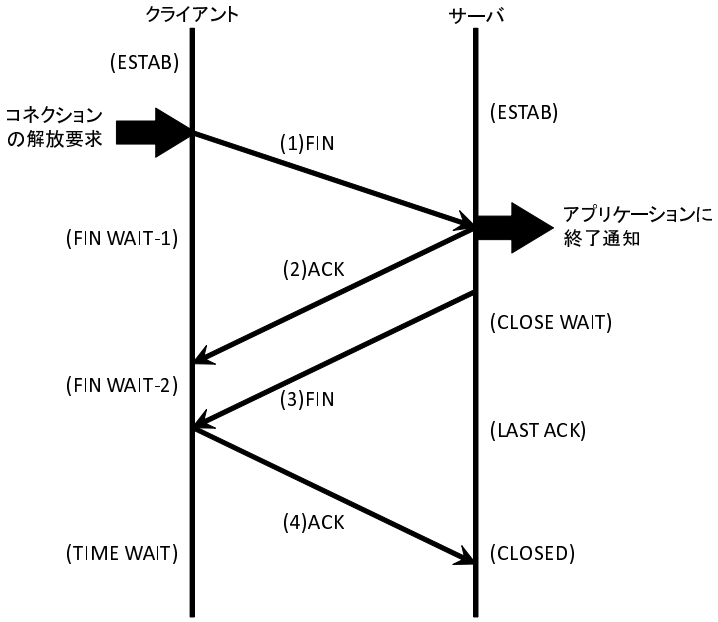

図 2.4: TCP コネクションの解放 [33]

(1) クライアントからコネクションの解放要求　クライアントのアプリケーションが，もう送るべきデータがないと認識すると，TCP にクローズを通知する。クローズが通知されると，TCP はその方向のコネクションを終了しようとする。つまり，TCP は送信中のデータをすべて送信し終えると，FIN フラグを立てたセグメントを送る。

(2) サーバのアプリケーションにクライアントからの通信終了の通知　サーバでは，FIN フラグの立ったセグメントを受け取ると，それに対する確認応答を返す。これにより，クライアントからサーバへの通信が終了する。このとき，TCP はサーバのアプリケーションにクライアントからの通信の終了を通知する。

(3) コネクションが終了し，クライアントに FIN を送る　サーバでも送るべきデータがなくなると，TCP にクローズを通知してコネクションが終了する。TCP はクライアントに FIN フラグの立ったセグメントを送る。

(4) クライアントから ACK を返す　クライアントは FIN フラグの立ったセグメントを受け取ると，ACK を返す。

クライアントからのコネクションの解放要求を受け取ったときに，サーバに送るべきデータがある場合，サーバは実行する前にそれらのデータを送出できる。このような解放状態はハーフクローズと呼ばれる。サーバは図 2.5 に示すように，送るべきデータをすべて送出後に FIN を送って ACK を待つ。 このような状態はしばしば考えられる。例えばリモートシステムからファイルをコピーしているような場合，コピー先（これをクライアントと読み替える）が転送すべきファイルの要求をすべて出し尽くしても，コピー元（これをサーバと読み替える）では要求されたファイルの内容を送っている（例えば，SSH 上

で scp を使っているような場合など）ときは，コピー元の TCP はファイルの内容をすべて送り終わるまで，コネクションをクローズできない。一方，コピー先の方は，もはや要求すべきファイルは残されていないため，その時点でコネクションをクローズしてもかまわないことになる。ところで，このとき，コネクションの先のプロセスはきちんと動作する。図 2.2 と図 2.4 にコネクションの解放手順を示す。ESTAB 状態にあるクライアントは CLOSE 命令を受け取ると，FIN フラグを立てたセグメントをサーバに送り，自身は FIN WAIT-1 という状態に遷移する。ここで，サーバから ACK を受け取ると，セグメントを何も送出せずに FIN WAIT-2 の状態に遷移して，サーバからの FIN フラグによるコネクションの終了通知を待つ。つまり，この時点でクライアント側の TCP，およびその上位のプロセスは FIN フラグの待機状態となっており，コネクションは完全に解放されているわけない。当然，この状態でデータが送られてくれば，正しいデータの送り先である特定のポート番号のプロセスにデータが届けられる。

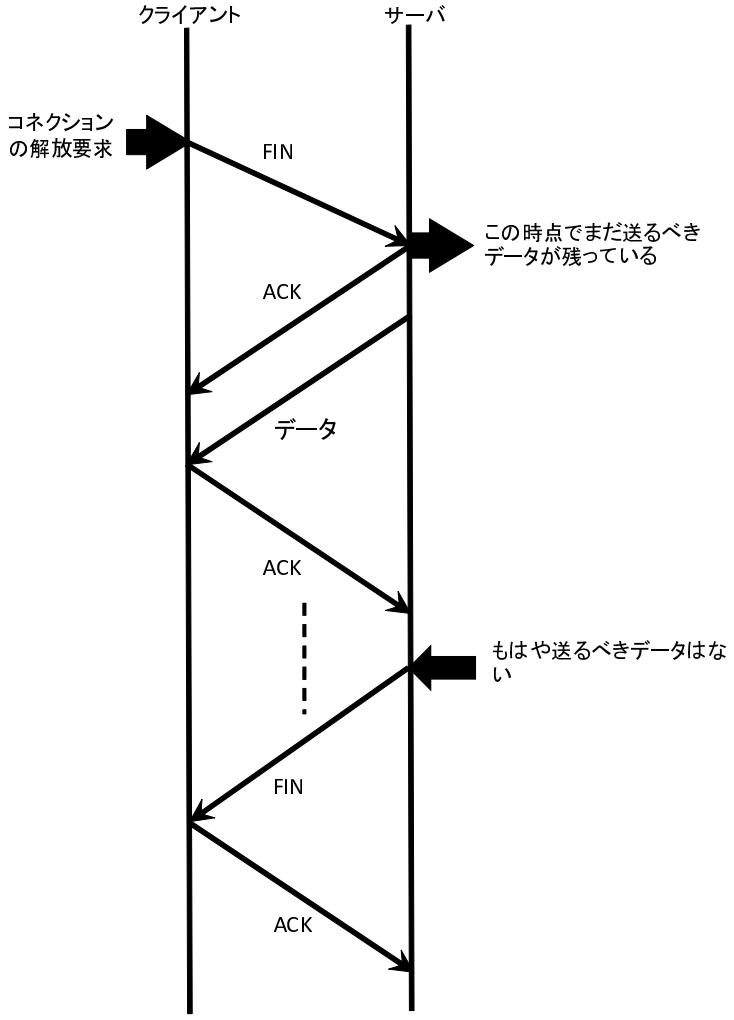

図 2.5: ハーフクローズ [33]

コネクションのリセット

2MSL（2 Maximum segment lifetime）状態にあるときにマシンがクラッシュし，システムが再起動した場合を考える必要がある。このときも，先ほどと同様に新旧のコネクションの混同が起きる恐れがある。これを避けるために，再起動後，MSL の間はいかなるコネクションも確立しないという規定が RFC にある。この時間は沈黙時間と呼ばれているが，この規定は実装する側の責任において回避してもよいことが RFC1122 [34] に明記されている。そのため，多くの実装ではこの規定を無視している。

TCP は何らかの異常状態を検出した場合，異常を検出した側が RST フラグを立てたセグメントを相手方に送ることでコネクションをリセットする。リセットはまさに強制終了なので，リセット時にはこれまで使われていた資源をすべて解放することになる。

2.3.2 輻輳制御と輻輳回避 [33]

TCP は，個々のコネクションが効率よくデータ転送できるように，あるいは受信側と途中のルータのバッファのあふれなどを避けるためにフロー制御機能を備えている。ただし，受信側のバッファのあふれに対するフロー制御と途中のルータを含めたネットワークのあふれに対する制御は大きく異なり，特に後者のルータを含めたネットワークのあふれに対処することを輻輳制御と呼ぶ。図 2.6 に示すように，負荷が Knee point を超えてから，

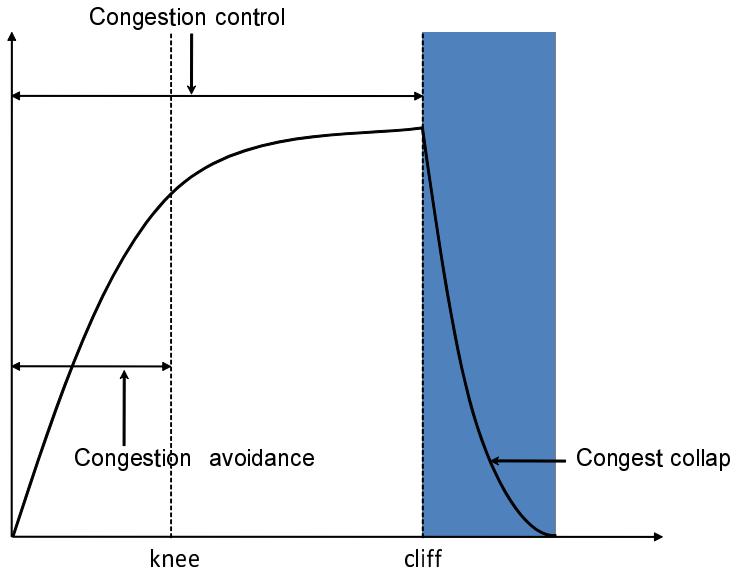

図 2.6: 輻輳制御と輻輳回避 [33]

スループットの増加速度が大幅に低下し，遅延の増加速度が速くなる。一方，負荷が cliff point を超えた場合，スループットは 0 まで急激的に低下し（輻輳崩壊）遅延が無限大になる。輻輳制御の目標として，負荷を cliff point の左側に維持することである。

受信側とのフロー制御

受信側は受信したデータを受信バッファ内に蓄積し，それをアプリケーションに渡す。多く場合，何らかの特別な事情がなければ，到着したデータはただちにアプリケーションに渡される。しかし，アプリケーションプログラムがビジー状態であったり，デッドロックなどの何らかの要因でアプリケーションがブロックされている場合には，データをただちに取り込むことができない。そのような状態が長く続くと，バッファに滞留するデータが多くなり，最終的にはバッファの空きがなくなる。このような状態になると受信側はデータを受け取ることができなくなるため，送信側にウィンドサイズを告知して送信側からの送信データを制御する。完全にバッファがフルになった場合は，ウィンドウサイズ 0 を通知して送信を止める。また，中途半端に小さなウィンドウサイズを送ると SWS（Silly Window Syndrome）の引き金になる可能性もある。

図 2.7 のように，スロースタート，輻輳回避，リカバリーの三つプロセスで，TCP が送出

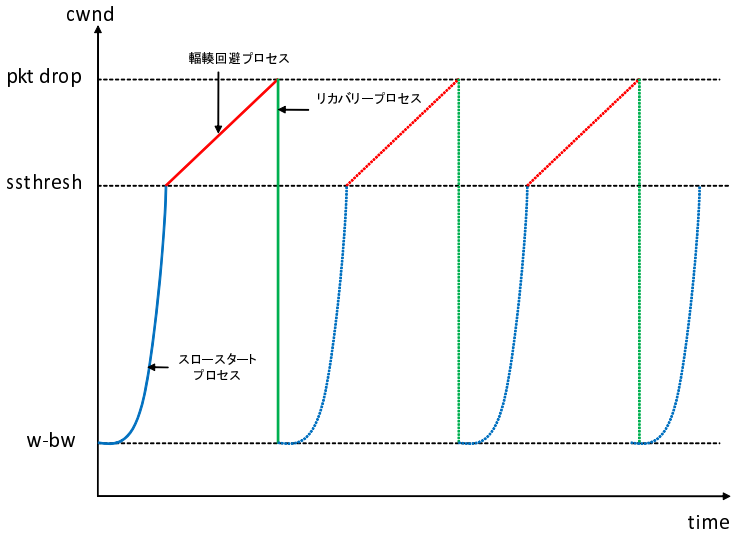

図 2.7: TCP 輻輳制御アルゴリズム

する輻輳ウィンドウの増減アルゴリズムを構成する。

スロースタートプロセス　送信側におけるウィンドウサイズは，受信側から告知されたウィンドウサイズが使われる。しかし，実際には，送信側における輻輳ウィンドウと告知ウィンドウのどちらか小さな値が取られる。TCP の開始直後などでは，輻輳ウィンドウは 1 にセットされている。それから，図 2.8 に示すように，確認応答が 1 つ返ってくる毎にウィンドウサイズを 1 つずつ大きくする。このようにして，TCP はネットワークが一挙にデータで埋め尽くされることがないようにしている。このような送信開始の方法をスロースタートと呼ぶ。ただし，出足はスローでも，スロースタートは図 2.7 に示すように指数関数的にウィンドウサイズが増加するために，そのままの状態にしておくといずれはネットワークを輻輳崩壊に追い込むことになるため，ある時点からは輻輳回避プロセスに移行する。

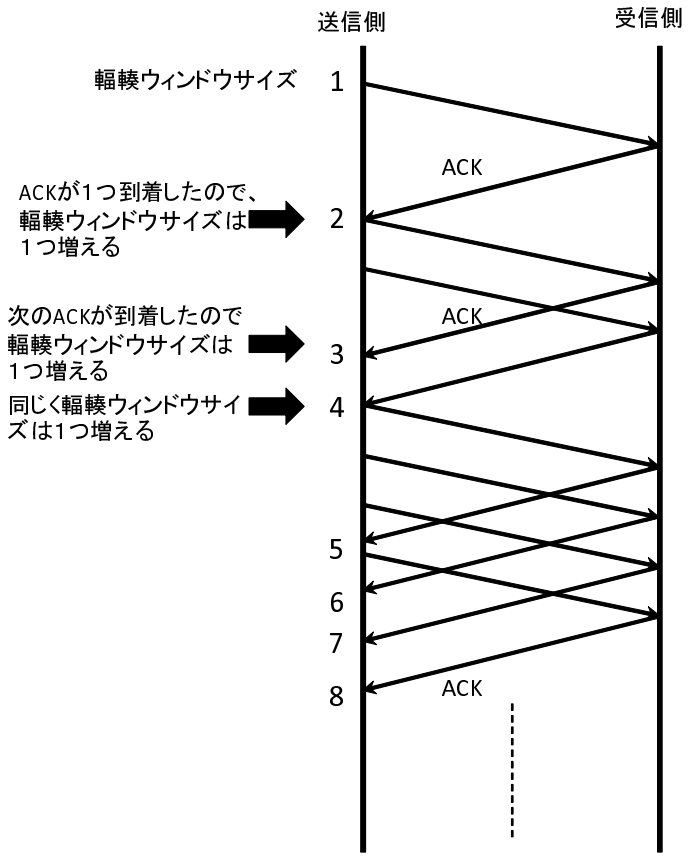

図 2.8: スロースタートによる輻輳ウィンドウの増加 [33]

輻輳回避アルゴリズム　輻輳回避アルゴリズムは，輻輳状態を緩和するためのアルゴリズムで，スロースタートが確認応答を受け取ると 1 ずつ輻輳ウィンドウサイズを増加させ，結果として指数関数的にウィンドウサイズが増えていく。それに対して輻輳回避は，確認応答を受け取ると現在の輻輳ウィンドウサイズの逆数分だけ増加させる。すると，現在の輻輳ウィンドウサイズと同じだけの確認応答を受け取るとウィンドウサイズが 1 増えることになる。このようなウィンドウサイズの増加はスロースタートに比べて格段に緩やかになる。このように緩やかに増加することで，輻輳崩壊を招くことを緩和する。

リカバリープロセス　パケット損失またはタイムアウトが発生してた場合に，移行するプロセスである。このプロセスでウィンドウサイズを急激的に下げる。このウィンドウサイズの下げかたは TCP のバージョンによって異なる。TCP Tahoe の場合はウィンドウサイズ *cwnd* を 1 とする。TCP Reno/New Reno の場合はウィンドウサイズをパケット損失またはタイムアウトが発生したときの最大ウィンドウサイズの半分まで下げる（$cwnd = cwnd/2$）。このように TCP ではウィンドウサイズを動的に制御することによって輻輳崩壊が起こらないようにしている。

また TCP は信頼性を保つためのアルゴリズムを備えている。それは Fast Retransmit アルゴリズムと Fast Recovery アルゴリズムである。

Fast Retransmit アルゴリズム　再送タイマのタイムアウトを待たずに迅速に再送パケットを転送する手法で，Repeat ACK が 3 つ到着した場合，データパケット損失の可能性が高いと判断して再送タイマのタイムアウトまで待機しないで当該パケットを迅速に再転送する。

Fast Recovery アルゴリズム　パケット再送後はさらに Repeat ACK を受信すると輻輳ウインドウサイズを一時的に増加させ，新しいパケットを送出する。新しい確認応答 ACK が到着すると，ウインドウサイズをパケット損失前の 1/2 にしスロースタートプロセスの閾値にセットし，スロースタート段階には入らず，いきなり輻輳回避段階に入る。これにより，転送速度をパケット損失検出前の 50%に落とす。

輻輳ウインドウサイズの適応戦略は 4 つ（AIMD（Additive Increase Multiplicative Decrease）[33]，AIAD（Additive Increase Additive Decrease）[33]，MIAD（Multiplicative Increase Additive Decrease）[33]，MIMD（Multiplicative Increase Multiplicative Decrease）[33]）あるが，現行の TCP のほとんどは AIMD 方式を採用している。AIMD 方式は，増やすときに加算的に，減らすときに乗算的に行い，公平性があるため採用されている。

2.3.3　AIMD [33]

AIMD 方式

TCP における輻輳窓の制御は以下のようになる。

$$ACK : w \leftarrow w + \frac{a}{w} \quad : \text{正常時} \tag{2.34}$$

$$Drop : w \leftarrow w - bw \quad : \text{損失時} \tag{2.35}$$

$$Slow - start : w \leftarrow w + c \tag{2.36}$$

各々のパラメータ a，b，c は次のようになる。$a = 1, b = 0.5$（Reno の場合），c=1。上の式からわかるように，AIMD 方式ではスロースタートプロセスで閾値を超えるまでは+1 ずつ増やすので線形的に増え，輻輳回避では増加スピードを緩め加算的にし，パケット損失を検出したら乗算的に減らすことがわかる。

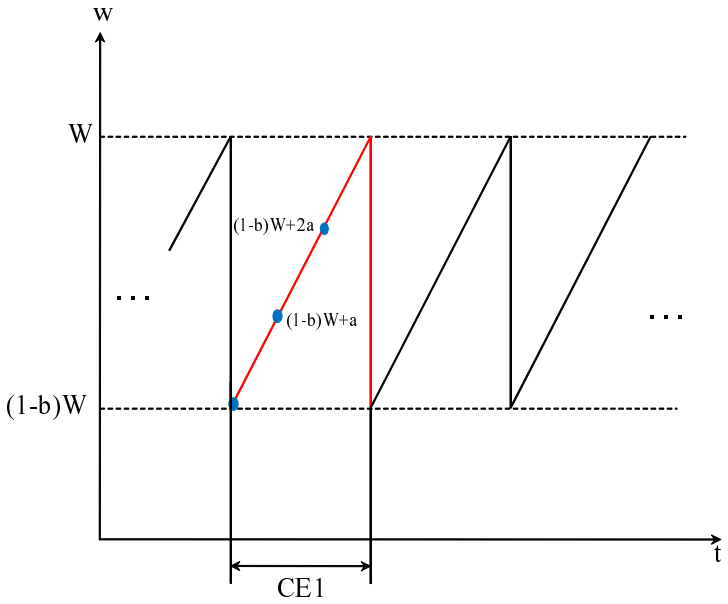

図 2.9: 衝突時間間隔

衝突時間間隔　各 CE 内で，ウィンドウサイズ W は各 RTT（Round Trip Time：リップ時間）において，$(1 - bW, 1 - bW + a, 1 - bW + 2a \ldots)$ ように増加するので，必要とする RTT の数 R は以下のように求められる。

$$(1 - b)W + xa = W \tag{2.37}$$

$$R = x + 1 = bW/a + 1 \tag{2.38}$$

つまり，各 CE 内で R 個の RTT が生じる。

平均送出率 ・ 送出率　定常状態において急激に変化する AIMD（a, b, c）モデルを用いた場合
平均送出率 S は，

$$S = \frac{2 - b}{2} W \quad (ppr) \tag{2.39}$$

送信率 T は，

$$T = \frac{2 - b}{2R} W \quad (pps) \tag{2.40}$$

各 CE 内での合計パケット数は,

$$\left(\frac{b}{a}W + 1\right) \times S = \left(\frac{b}{a}W + 1\right) \times \frac{2-b}{2}W \tag{2.41}$$

$$\approx \frac{b(2-b)}{2a}W^2 \tag{2.42}$$

パケット損失率　式（2.41），式（2.42）よりパケット損失率 p は,

$$P = \frac{2a}{b(2-b)W^2 + a(2-b)W} \tag{2.43}$$

or

$$p \approx \frac{2a}{b(2-b)W^2} \tag{2.44}$$

式（2.44）の近似式より,

$$W \approx \sqrt{\frac{2a}{b(2-b)p}} \tag{2.45}$$

となる。

AIMD 応答関数

AIMD フローの定常状態送信レート T は AIMD 応答関数のパラメータ（a, b, R, p）によって決定される。

a は増加パラメータ，b は減少パラメータ，R はラウンド・トリップ・タイムで，p はパケット損失率である。S は最大輻輳ウィンドウサイズである。

式（2.40），式（2.45）より，AIMD 応答関数の一般式は以下のように表せる。

$$T = AIMD(a, b, R, p) = \frac{\sqrt{a(2-b)}}{R\sqrt{2bp}} \tag{2.46}$$

$$S = AIMD(a, b, p) = \frac{\sqrt{a(2-b)}}{\sqrt{2bp}} = \frac{\sqrt{\frac{a(2-b)}{2b}}}{\sqrt{p}} \tag{2.47}$$

TCP Reno の場合は以下のように表せる（図 2.10，2.11）。

TCP Reno のパラメータ（$a = 1, b = 0.5$）を用いて，T と S は以下のように計算することができる。

$$T = AIMD(1, 0.5, R, p) = \frac{\sqrt{1.5}}{R\sqrt{p}} \tag{2.48}$$

$$S = AIMD(1, 0.5, p) = \frac{\sqrt{1.5}}{\sqrt{p}} \tag{2.49}$$

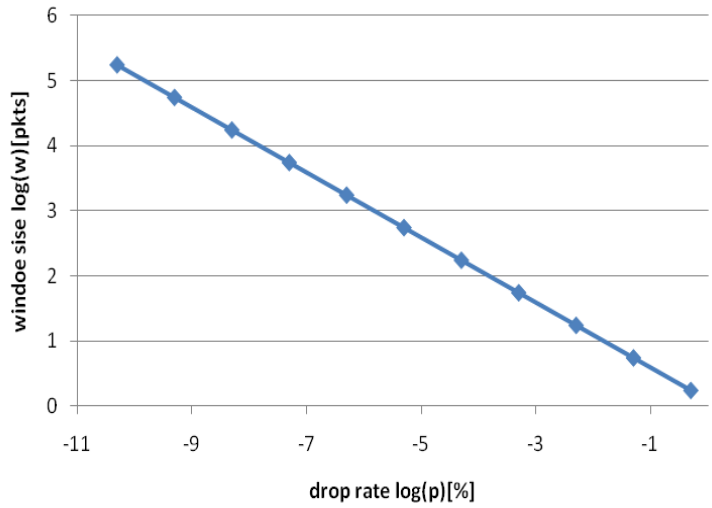

図 2.10: TCP の基本グラフ

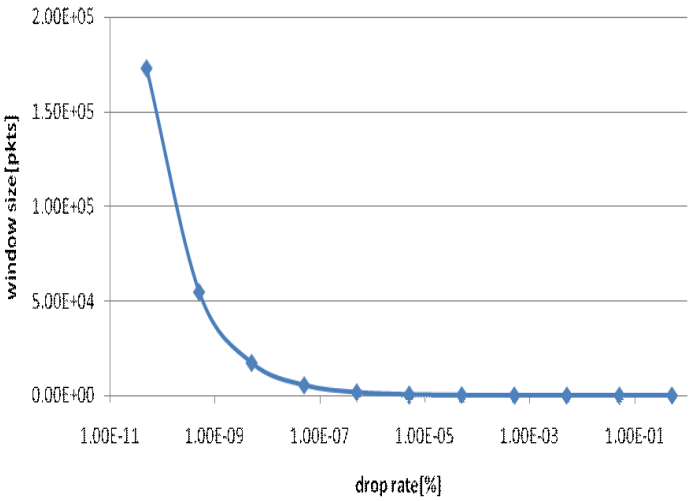

図 2.11: TCP の基本対数グラフ

スロースタートフェーズにおける応答機能　スロースタートフェーズにおける瞬時のスループット $T(t)$ は次のように与えられる。

$$T(t) = 2^{\frac{t}{R}} \quad (ppr.) \tag{2.50}$$

速度 T に達するには時間 $t(T)$ がかかる。

$$t(T) = R \times \log_2(T) \quad (s) \tag{2.51}$$

例えば，帯域幅 10[Gbps]，パケットサイズ 1500[byte] のとき必要となるウィンドウサイズは

$$\frac{10^{10}}{1500 \times 8} \approx 83333 \quad [packets] \tag{2.52}$$

この速度を達成するパケットは 83333[pakets] となり，それを達成するときにかかる時間 $t(T)$ は

$$t(T) = R\log_2(83333) \approx 16.3455R \approx 17[RTT] \tag{2.53}$$

となる。

2.3.4　TCP Reno の輻輳制御方式 [33]

現在の TCP における輻輳制御方式においての主流は TCP Reno である。TCP Reno は ACK パケットの到着をもとにして，転送レートの調整を行う，ウィンドウフロー制御を行っている。

TCP Reno は，まず送受信ホスト間で予め ACK の受信を待たずに連続して送信できるパケットの最大数を設定する。このパケットの最大数を輻輳ウィンドウサイズと呼ぶ。送信側ホストは，データパケットを送信済みであるが，まだ ACK パケットを受信していないパケットの数が輻輳ウィンドウサイズよりも小さい場合は，さらに続けてデータパケットを送信できる。ACK パケットの送信を停止し，ACK パケットの到着を待つ。TCP Reno の送信側ホストは，ACK パケットを受信すると，輻輳ウィンドウサイズを次式のように変更する。

$$cwnd \leftarrow cwnd + 1 \qquad (cwnd < ssth) \tag{2.54}$$

$$cwnd \leftarrow cwnd + \frac{1}{cwnd} \qquad (otherwise) \tag{2.55}$$

ここで，$ssth$ は TCP Reno が「スロースタートフェーズ」から「輻輳回避フェーズ」と呼ばれる動作モードに移行するときの閾値である。つまり，TCP Reno はスロースタートフェーズで輻輳ウィンドウサイズを指数的に増加させ，輻輳回避フェーズでは輻輳ウィンドウサイズを線形的に増加させる。

次に，TCP Reno がパケット棄却を検出した場合を説明する。まず，TCP Reno は重複 ACK（Duplicate ACK）とタイムアウトという 2 種類の方法で，ネットワーク中でのパ

ケット棄却を検出する。TCP Reno が重複 ACK によりパケット棄却を検出した場合（Fast Retrasmission），閾値と輻輳ウィンドウサイズを次式のように更新する。

$$ssth \leftarrow \frac{cwnd}{2} \tag{2.56}$$
$$cwnd \leftarrow ssth \tag{2.57}$$

このため，高速回復フェーズにおいて，輻輳ウインドウサイズは指数的に増加することになる。ただし，再送したパケットに対応した ACK パケットを受信すると，高速回復フェーズ移行前の値に戻す。

TCP Reno がタイムアウトによりパケット棄却を検出した場合，閾値と輻輳ウインドウサイズを次式のように更新する。

$$ssth \leftarrow \frac{cwnd}{2} \tag{2.58}$$
$$cwnd \leftarrow 1 \tag{2.59}$$

2.3.5 ロングファットパイプネットワーク問題 [16,17]

Long Fat Pipe Syndrome

従来のインターネットでは，遠距離間のホームページへアクセスすると反応が遅く，イメージデータ等のテキスト以外のコンテンツをダウンロードするのに長時間を要するといった問題があった。これは TCP がホスト間でパケットの ACK を頻繁に行うため，データリンクが ATM やギガビットイーサネットのように高速であっても転送距離が長い場合，高スループットが得られにくくなり，従来の TCP に対して，帯域増加すると同時に転送距離が長くなったとき，ネットワーク空き時間が発生しやすくなる。この現象は「Long Fat Pipe Syndrome」と呼ばれる。

従来 TCP の問題点 [33]

広帯域で遅延の大きいロングファットパイプネットワークにおいて TCP Reno で高い輻輳ウインドウサイズを次の条件（パケットサイズ : 1500[Byte]，RTT : 100[ms]，リンク帯域幅 : 10[Gbps]）で維持しようとしたとき，定常状態送信レート T は以下のように表せる。

$$T = \frac{\sqrt{1.5}}{100 \times 10^{-3}\sqrt{p}} \times 1500 \times 8 = \frac{\sqrt{1.5}}{\sqrt{p}} \times 1.2 \times 10^{5} \tag{2.60}$$

この場合 TCP が 10[Gbps] の送信レートを保つためには，以下に示す数値が必要になる。

$$\frac{\sqrt{1.5}}{\sqrt{p}} \times 1.2 \times 10^{5} = 10^{10} \tag{2.61}$$
$$p \approx 2 \times 10^{10} \tag{2.62}$$

ここで，p はパケット損失率を表す。つまり，これは 5×10^{9} パケットごとに一回パケット損失が発生することになる。これを時間になおすと，約 1.67[h] 毎ごとに一回しか輻輳を発生しないということになる。この値は現実的に不可能である。

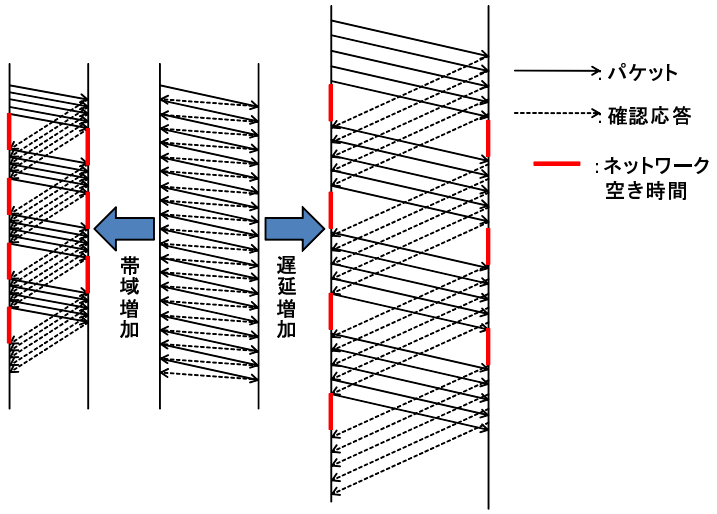

図 2.12: Long Fat Pipe Syndrome

2.3.6 HighSpeed TCP [24, 25]

HighSpeed TCP がウィンドウサイズを増減させるアルゴリズムは，基本的には TCP Reno と同じであるが，輻輳回避フェーズにおける 1RTT 毎のウィンドウサイズの増減幅，および，重複 ACK の受信によってパケット廃棄を検出した際のウィンドウサイズの減少幅が異なる。以下では，輻輳回避フェーズにおける HighSpeed TCP のウィンドウサイズの増減について説明する。

図 2.13 のように，HighSpeedTCP は，現在のウィンドウサイズが W_{low} より小さい場合，TCP Reno と同じアルゴリズムにしたがってウィンドウサイズを増減させる。一方，現在のウィンドウサイズが W_{low} よりも大きい場合，ACK パケットを受信した際に，TCP Reno よりもウィンドウサイズを大きく増加させ，重複 ACK によってパケット廃棄を検出した際には，TCP Reno のように大きくウィンドウサイズを減少させない。

このことにより，TCP コネクションのウィンドウサイズを大きな値に維持できるため，高いスループットを得られる。

HighSpeed TCP の応答関数

ウィンドウサイズの増加幅および減少幅は，現在のウィンドウサイズの大きさを用いて決定される。ウィンドウサイズが w のとき，1RTT 毎のウィンドウサイズの増加幅を $a(w)$，重複 ACK によってパケット廃棄を検出した際のウィンドウサイズの減少幅を $b(w)$ とする

と，$a(w)$ および $b(w)$ は次のように表される。

$$a(w) = w_{high}^2 P_{high} \times \frac{2b(w)}{2 - b(w)} \tag{2.63}$$

$$b(w) = 0.5 + (B - 0.5)\frac{\log w - \log W_{low}}{\log W_{high} - \log W_{low}} \tag{2.64}$$

$$p(w) = \exp\left[\frac{\log(w) - \log(W_{low})}{\log(W_{high}) - \log(W_{low})} \times (\log(P_{high}) - \log(P_{low})) + \log(P_{low})\right] \tag{2.65}$$

$$P_{low} = \frac{1.5}{{W_{low}}^2} \tag{2.66}$$

P_{low} は TCP Reno において平均輻輳ウインドウサイズが W_{low} となるときのパケット廃棄率を表す。また P_{high}，W_{high}，b_{high}，W_{low} は HighSpeed TCP が持つパラメータであり，以下のような意味を持つ。HighSpeed TCP は，ネットワークにおけるパケット廃棄率が P_{high} のときに，平均ウインドウが W_{high} になるように，輻輳ウインドウサイズの増加幅 $a(w)$，および減少幅 $b(w)$ を決定する。また，現在の輻輳ウインドウサイズが W_{high} 以上のときには，重複 ACK の受信によってパケット廃棄率を検出した際の輻輳ウインドウサイズを $(1 - b_{high}) \times W_{high}$ にする。

以上より，HighSpeed TCP は TCP Reno よりも高速ネットワークにおいても高いスループットを実現することが可能となる。

HighSpeedTCP の輻輳制御方式

HighSpeedTCP は図 2.13 により，輻輳ウインドウが低い場合（P_{low} 以下の時）は TCP Reno と同じ輻輳制御を行い，輻輳ウインドウが高い場合（P_{high}以上の時）には，TCP Reno より増加率を積極的に修復する TCP 反応機能を使用する。このとき

$$\log w = \tan\theta(\log p - \log P_{low}) + \log W_{low}) \tag{2.67}$$

$$S = \tan\theta = \frac{\log W_{high} - \log W_{low}}{\log P_{low} - \log P_{high}} \tag{2.68}$$

$$w = 10^{S(\log p - \log P_{low}) + \log W_{low}} = \frac{p^S}{P_{low}^S} W_{low} \tag{2.69}$$

$$w = \frac{\sqrt{1.5}}{\sqrt{p}} \qquad , if \quad w \le W_{low} \tag{2.70}$$

$$w = \frac{P^S}{P_{low}^S} W_{low} \qquad , if \quad w > W_{low} \tag{2.71}$$

となる。

次に AIMD（$a(w), b(w), c(w)$）型による，HighSpeedTCP のウインドウサイズ w を求める。標準値を W_{low}[pkts]，輻輳ウインドウサイズを W_{high} とすると，増加パラメータ $a(w)$ と減少パラメータ $b(w)$ の相関関係は以下で表せる。

$$W_{high} = \frac{\sqrt{\frac{a(w)(2-b(w))}{2b(w)}}}{\sqrt{P_{high}}} \tag{2.72}$$

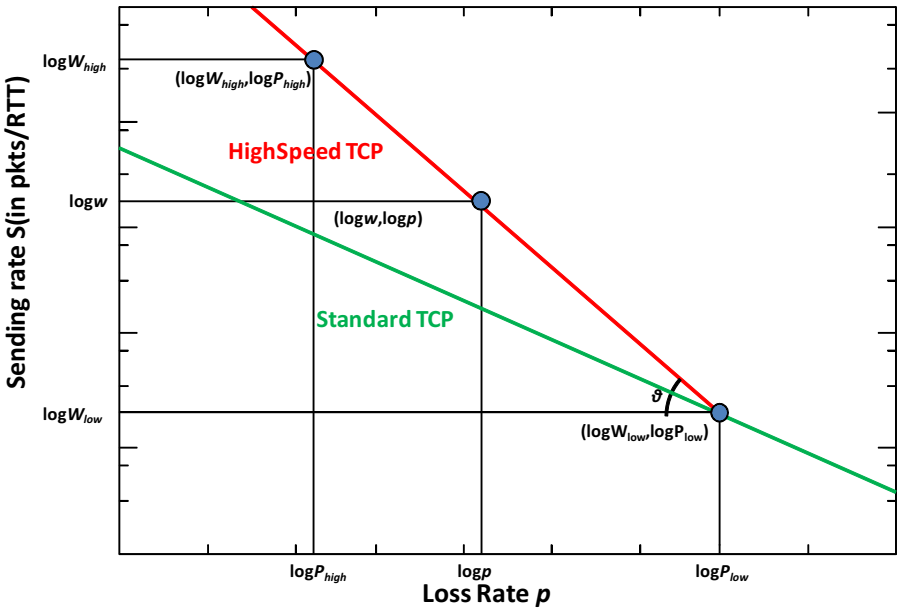

図 2.13: HighSpeed TCP と TCP

上式より，$a(w)$ に対する $b(w)$ と $w = W_{high}$ の関係式が求められる。

$$a(w) = W_{high}^2 P_{high} \times \frac{2b(w)}{2 - b(w)} \tag{2.73}$$

$a(w)$ と $b(w)$ を選ぶときの指標として W_{low} に対するパケット損失率 $p(W_{low} = P_{low})$ と W_{high} に対するパケット損失率 $p(W_{high}) = P_{high}$ がある。対数グラフより，以下で W_{low} に対するパケット損失率 $p(w)$ を求めることができる。

$$\begin{aligned}
\log p(w) &= \log P_{low} - (\log w - \log W_{low})/\tan\theta \\
&= \log P_{low} - (\log w - \log W_{low})\frac{\log P_{low} - \log P_{high}}{\log W_{high} - \log W_{low}} \\
&= \log P_{low} + (\log P_{high} - \log P_{low})\frac{\log w - \log W_{low}}{\log W_{high} - \log W_{low}}
\end{aligned} \tag{2.74}$$

減少パラメータ $b(w)$ は $b(W_{low}) = 0.5$，$b(W_{high}) = B$ とする。$w \le W_{low}$ としたとき，$b(w)$ は対数の $p(w)$ に対し直線的になると仮定する。

$$b(w) = 0.5 + (B - 0.5)\frac{\log w - \log W_{low}}{\log W_{high} - \log W_{low}} \tag{2.75}$$

ここで，AIMD 方式を使用する場合，HighSpeedTCP のウィンドウサイズの制御式は以下になる。

$$ACK : w \leftarrow w + \frac{a(w)}{w} \quad : 正常時 \tag{2.76}$$

$$Drop : w \leftarrow w - b(w)w \quad : 損失時 \tag{2.77}$$

ここで,

$$a(w) = W_{high}^2 P_{high} \times 2b(w)/(2 - b(w)) \tag{2.78}$$

$$b(w) = 0.5 + (B - 0.5)\frac{\log w - \log W_{low}}{\log W_{high} - \log W_{low}} \tag{2.79}$$

B, P_{high}, W_{low}, W_{high} はパラメータで，その推奨値:$B = 0.1$, $P_{high} = 0.0000001$, $W_{low} = 31$, $W_{high} = 83000[10Gbps]$ である。

このようにして HighSpeed TCP の応答関数が求められる。推奨値は定常状態において推奨されている数値である。

2.3.7 HighSpeed TCP の問題点

TCP Reno との公平性 [16,35]

HighSpeedTCP と TCP Reno の間の公平性を求める。それぞれの輻輳ウインドウサイズを w_{reno}，$w_{highspeed}$ とするとき,

$$w_{reno} = \frac{\sqrt{1.5}}{\sqrt{p}} \tag{2.80}$$

$$w_{highspeed} = \frac{p^S}{P_{low}^S} W_{low} \tag{2.81}$$

となる。このとき HighSpeedTCP のパラメータを帯域幅:10[Gbps], パケットサイズ:1500[Byte], RTT:100[ms] とし，ウインドウサイズが 31[pkts] のとき損失率 P_{low}，増加率 S を求める。

$$P_{low} = \left(\frac{\sqrt{1.5}}{W_{low}}\right)^2 = \left(\frac{\sqrt{1.5}}{31}\right)^2 \approx 0.0015 \tag{2.82}$$

$$S = \frac{\log W_{high} - \log W_{low}}{\log P_{low} - \log P_{high}} = \frac{\log 83000 - \log 31}{\log 0.0015 - \log 0.0000001} \tag{2.83}$$

P_{low} と S を式（2.81）に代入し，$w_{highspeed}$ を w_{reno} で除算し，公平性を求める。

$$\frac{W_{highspeed}}{W_{reno}} = \frac{0.15}{p^{0.82}} / \frac{0.18}{p^{0.5}} = \frac{0.11}{p^{0.32}} \tag{2.84}$$

上式より HighSpeed TCP の公平性は図 2.14 のようになる。図 2.14 により，パケット損失率が減少すると，HighSpeed TCP と TCP Reno の間が不公平になると考えられる。

しかし，将来，輻輳制御はすべて HighSpeed TCP に切り替えられることが予測できるため，HighSpeed TCP の公平性の改善は将来性がないと考えられる。そのため，本研究ではこの問題を見送ることとする。

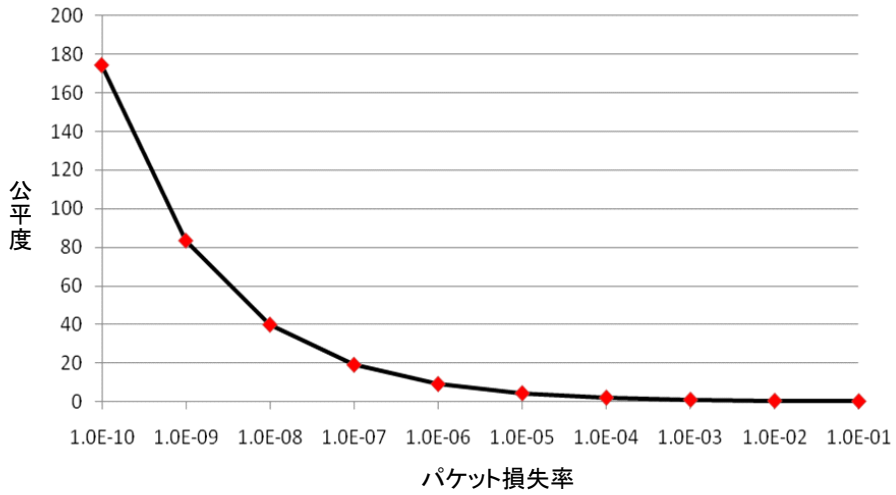

図 2.14: HighSpeedTCP の公平性

帯域幅に対する適応性 [16, 17]

帯域幅 10[Gbps]，パケットサイズ 1500[byte] の場合，式 (2.50)，式 (2.51) と式 (2.52) によって，平均輻輳ウインドウサイズは 83333[pkts] と算出した。その値の参考し，HighSpeed TCP の W_{high} の推奨値は 83000 と設定する。また，HighSpeed TCP は式 (2.77)，式 (2.77)，式 (2.79) と式 (2.79) によって制御し，TCP Reno より輻輳時間間隔が短くなる。

しかし，W_{high} の推奨値は帯域幅 10[Gbps]，パケットサイズ 1500[byte] の場合で算出し，設定した値である。そのために，帯域幅の増加を伴って，式 (2.52) によって，平均輻輳ウインドウサイズが増えると考えられる。また，式 (2.53) によって，帯域幅の増加に伴って，輻輳時間間隔は線形的に増加することが，第 4 章の検証実験で判明した。

第3章

提案手法

3.1 協調的学習オートマトンチームモデル [36,37]

協調的学習オートマトンチーム（The Collaborative Learning Automaton Team）LA^C_{TEAM} は，未知環境 RE 上で動作する N 個の確率オートマトンより構成される。

$$LA^C_{TEAM} = \{SA, RE\} \tag{3.1}$$

$$SA = \left\{SA^1, SA^2, ..., SA^N\right\} \tag{3.2}$$

各確率オートマトンは空間分散的に配置され，隣接するもの同士のみ情報交換を行う。各確率オートマトンは隣接確率オートマトンの情報を直接，または間接的に参照しながら，各自の強化学習プロセスを遂行する。

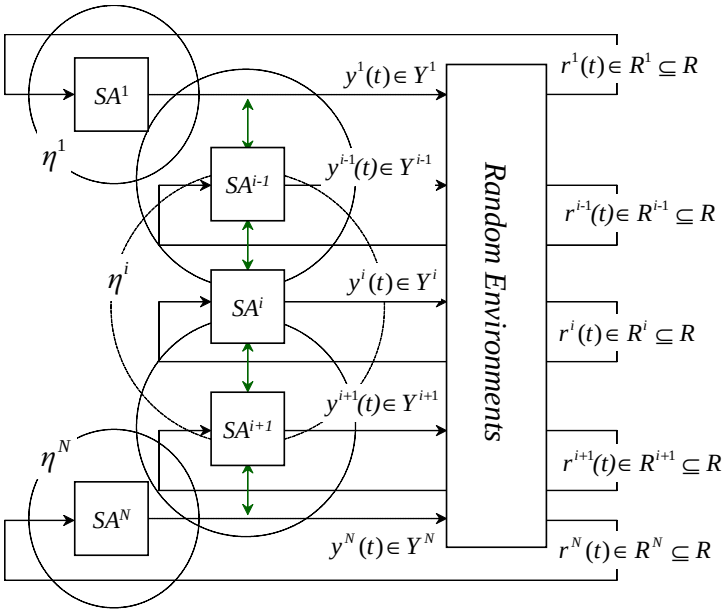

図 3.1: 協調的学習オートマトンチームモデルの基本構成

図 3.1 に，協調的学習オートマトンチームモデルの基本構成を示す。このモデルでは，各個体 SA^i は未知環境 RE からの強化信号入力 $r^i(t)$ だけではなく，隣接確率オートマトンである SA^{i-1}, SA^{i+1} からの情報入力も持つのが特徴である。従来の学習オートマトンチームモデル [13] と異なり，協調的学習オートマトンチームモデルにおける各個体 SA^i の学習目標は，隣接確率オートマトンの情報を参照（協調）しながら，未知環境からの強化信号を利用して自分の最適戦略を逐次的に決定することである。

3.1.1 協調的学習オートマトンチームモデルの個体表現

協調的学習オートマトンチームモデルにおける i 番目の個体 SA^i を以下のように定義する。

$$SA^i = \left\{R^i, Y^i, \Phi^i, p^i, o^i, T^i, \eta^i\right\}, \qquad i = 1, 2, \ldots, N \tag{3.3}$$

ただし，

R^i は SA^i への強化信号の集合
Y^i は SA^i の出力集合
Φ^i は SA^i の状態空間
p^i は SA^i の出力選択確率分布
o^i は SA^i の出力関数
T^i は SA^i の強化法
η^i は SA^i の隣接確率オートマトンの集合

である。詳細な説明を以下に述べる。

1. 強化信号
   $r^i(t) \in R^i$ は時刻 t における強化信号で，SA^i の出力に対する未知環境側からの評価を表す。未知環境全体 R は，各確率オートマトンの未知環境 R^i の和集合として以下のように構成される。
   $$R \equiv \bigcup_{i=1}^{N} R^i \tag{3.4}$$

2. 出力集合の表現
   SA^i の出力集合 Y^i を以下のように定義する。
   $$Y^i \equiv \{Y_I^i, Y_E^i\} = \left\{y_1^i, y_2^i, ..., y_{s^i}^i\right\} \tag{3.5}$$
   ここで，Y_I^i は SA^i の固有出力集合，すなわち内部出力集合であり，Y_E^i は SA^i の隣接 SA（$SA^j \in \eta^i$ and $SA^j \in SA$）からもらった情報に対応する出力の集合で，SA^i の外部出力集合と呼ぶ。出力 $y_j^i \in Y^i$ は出力関数 o^i により決定される。ここで，出力関数 o^i は出力選択確率 p_j^i で出力 $y_j^i \in Y^i$ を選んで返す関数である。

3. 状態空間の表現
   SA^i の状態空間 Φ^i は内部状態空間と外部状態空間から構成される。
   $$\Phi^i \equiv \left\{\Phi_I^i, \Phi_E^i\right\} = \left\{\phi_1^i, \phi_2^i, ..., \phi_{s^i}^i\right\} \tag{3.6}$$

ここで，Φ_I^i は SA^i の固有状態集合，すなわち内部状態集合であり，Φ_E^i は SA^i の隣接確率オートマトン $SA \in \eta^i$ からもらった情報に対応する状態集合で，外部状態集合と呼ぶ。

4. 出力選択確率分布の表現
出力選択確率分布 p^i は状態空間 Φ^i 上で定義される確率分布で，以下のように定義する。

$$p^i \equiv \left\{P_I^i, P_E^i\right\} = \left(p_1^i, p_2^i, \ldots, p_{s^i}^i\right)^T \tag{3.7}$$

ここで，P_I^i は SA^i の固有状態に対応する確率分布，P_E^i は SA^i の隣接確率オートマトン $\in \eta^i$ の外部状態に対応する確率分布である。$p_j^i \in p^i$ は状態 $\phi_j^i \in \Phi^i$ に対応する出力 $y_j^i \in Y^i$ を出力関数 o^i により選択する確率である。強化学習プロセスにおいて，出力選択確率分布 p^i を強化法 T^i により修正する際に，p^i 上で単体（simplex）条件が常に成り立つことを考慮する必要がある。

5. 出力関数の表現
出力関数 o^i は状態空間 Φ^i から出力空間 Y^i への写像として，$o^i : \Phi^i \to Y^i$ と定義されるが，学習オートマトンの研究では，状態空間と出力空間は一般に単射的関係が成立するとしているので，簡単のために，各出力 y_j^i は出力選択確率分布 p^i により決定されると見なす。

6. 強化法
SA^i の強化法 T^i を以下のようなオペレータの形で定義する。

$$p^i(t+1) = T^i\left(y^i(t), r^i(t), p^i(t)\right) \tag{3.8}$$

各個体 SA^i には従来の強化法をそのまま適用できるが，その利用形態により，協調的学習オートマトンチームモデルはさらに以下の 2 種類に分類できる。各個体 SA^i が同じ強化法を採用し，同じ学習率で学習するモデルを同質モデル（Homogeneous Model）と呼び，各個体 SA^i が異なる強化法を採用し，または異なる学習率で学習するモデルを異質モデル（Non-Homogeneous Model）と呼ぶ。

7. SA^i の隣接集合 [36]
SA^i の隣接集合 η^i は SA^i と直接情報交換する確率オートマトンから構成される。隣接確率オートマトン同士では各自の情報交換機構を利用して互いに情報交換を行う。隣接集合に自分以外のすべての確率オートマトンが含まれる場合，完全協調モデル，一部しか含まない場合，部分協調モデルと呼ぶ。

3.1.2 未知環境 [12–15]

未知環境 RE を以下のように定義する。

$$RE = \{Y, R, C\} \tag{3.9}$$

ただし，各記号の意味は以下の通りである。
Y は RE の入力集合，$y(t) \in Y$ は時刻 t における入力を表す。
同期モデルの場合，Y を以下のように定義する。

$$Y \equiv Y^1 \otimes Y^2 \otimes \ldots \otimes Y^N \tag{3.10}$$

ここで，$\otimes$ は直積を表す演算子である。
非同期モデルの場合，Y を以下のように定義する。

$$Y \equiv Y^1 \cup Y^2 \cup \ldots \cup Y^N \tag{3.11}$$

ここで，$\cup$ は和集合を表す演算子である。
R は強化信号の集合，$r(t) \in R$ は時刻 t における強化信号を表す。
C は R 上のペナルティ確率分布で以下のように定義する。

$$C = \left\{c_1^1, \ldots, c_{s^1}^1, \ldots, c_1^N, \ldots, c_{s^N}^N\right\} \tag{3.12}$$

従来の慣習により，未知環境は強化信号 r の表現により以下の 3 モデルに分類される。

- P-model
  $r \in \{0, 1\}$ の 2 値の離散値で，$r = 1$ をペナルティ応答，$r = 0$ をリワード応答と呼ぶ。
- Q-model
  $r \in \left\{r_j \mid j = 1, 2, ..., q; r_j \in [0, 1]\right\}$ の多値の離散値となる。
- S-model
  $r \in [0, 1]$ の連続値となる。

$c_j^i \in C$ は，SA^i からの入力 y_j^i に対して，ペナルティ信号（$r = 1$）を返す確率を表す。ここで，未知環境から協調的学習オートマトンチームへの強化信号の受け渡し方により，協調的学習オートマトンチームモデルを同期モデルと非同期モデルに分類する。

- 同期モデル（Synchronized model）

  $$r^1(t) = r^2(t) = \ldots = r^N(t) = r(t) \tag{3.13}$$

  同期モデルの場合，各確率オートマトンは未知環境から同じ強化信号を受け取る。
- 非同期モデル（Non-Synchronized model）

  $$\exists i \neq j : r^i(t) \neq r^j(t),\ i, j \in \{1, 2, \ldots, N\} \tag{3.14}$$

  非同期モデルの場合，各確率オートマトンは未知環境から異なる強化信号を受け取ることがある。

3.1.3 情報交換機構

確率オートマトン間で交換する情報は応用対象によってさまざまな形で定義できるが，ここでは一例として，$SA^j \in \eta^i$ が SA^i に転送する情報 h^{ji} を以下のように定義する。

$$h^{ji} = (y_{j^*}^j(t), p_{j^*}^j(t)),\ j = 1, 2, \ldots, |\eta^i| \tag{3.15}$$

ただし，$y^j_{j^*}(t)$ は SA^j において現在（時刻 t）までに推定された最適出力で，$p^j_{j^*}(t)$ はそれを選択する確率である。ここで，自分自身への情報転送は考慮しないので，$h^{ii} = 0$ となる。

学習の初期段階で，SA^i がまだ隣接確率オートマトンから情報を受け取っていない場合，出力集合 Y^i と出力選択確率 p^i を以下のように定義する。

$$Y^i = \left\{ Y^i_I, Y^i_E = \{0, \ldots, 0\} \right\} \tag{3.16}$$

$$p^i = \left(P^i_I, P^i_E = (0, \ldots, 0) \right) \tag{3.17}$$

SA^i が自分の隣接確率オートマトン $SA \in \eta^i$ から新しい情報を受け取った場合，$y^j_{j^*}(t)$ を外部出力集合 Y^i_E に挿入した後，確率の単体条件を維持するために，出力選択確率分布 p^i の各要素 p^i_j を以下のように修正する。

$$p^i_j \leftarrow p^i_j \Big/ \Big(\sum_{p \in P^i_I} p + \sum_{p' \in P^i_E} p' \Big), j = 1, \ldots, s^i \tag{3.18}$$

情報交換は随時，または一定時間毎に行うものとする。

3.1.4 協調的学習オートマトンチームの強化法 [12–15, 26]

協調的学習オートマトンチーム内の各確率オートマトンの強化学習プロセスは自身の全状態空間上で行われる。確率オートマトンの強化法は，理論的な解析時に用いる強化法 T^R，具体的な問題への応用時に用いる R-I タイプ強化法および R-P タイプ強化法が知られている。R-I タイプ強化法および R-P タイプ強化法は，いずれも強化法 T^R より派生した手法である。R-I タイプ強化法は強化法 T^R と同様に未知環境からのリワード応答時に出力選択確率を修正する強化法で，R-P タイプ強化法はリワード応答時のみならずペナルティ応答時にも出力選択確率を修正する強化法である。ここでは，一般的な議論を展開するために，最も一般的で理論的な解析が容易な強化法 T^R を取り上げる。

強化法 T^R

時刻 t で，SA^i が出力選択確率分布 $p^i(t)$ に従い，出力 $y^i_k \in Y^i$ を選択して未知環境に出力し，リワード応答の強化信号 $r^i(t) = 0$ を受け取ったとする。このとき，出力 y^i_k の選択確率 $p^i_k(t)$ および出力 $y^i_j (j \neq k, j = 1, \ldots, s^i)$ の選択確率 $p^i_j(t)$ を以下の強化法 T^R で修正する。

$$p^i_j(t+1) = p^i_j(t) - \theta(t)\delta^i_{j,k}(t), \ j \neq k \tag{3.19}$$

$$p^i_k(t+1) = p^i_k(t) + \theta(t) \sum_{j \neq k} \delta^i_{j,k}(t) \tag{3.20}$$

すなわち，リワード応答 $r^i(t) = 0$ を受け取った出力 y^i_k の選択確率 $p^i_k(t)$ を増加させ（式 (3.20)），y^i_k 以外の出力 $y^i_j (j \neq k)$ の選択確率 $p^i_j(t)$ を減少させる（式 (3.19)）。ただし，

$$\delta^i_{j,k}(t) = \xi(\bar{d}^i_j(t)) - \xi(\bar{d}^i_k(t)) \tag{3.21}$$

$$\bar{d}^i_j(t) = \frac{1}{l}\sum_{m=1}^{l} d^i_j(t-m) \tag{3.22}$$

$$\bar{d}^i_k(t) = \min_{j=1,2,\ldots,s^i} \bar{d}^i_j(t) \tag{3.23}$$

である。ここで，$\theta(t)$ は学習率，$\xi(\cdot)$ は区間 [0,1] 上で定義される単調増加関数である。$D^i_j(t) = (d^i_j(t-l), \ldots, d^i_j(t-1))$ を時刻 $t-l$ から $t-1$ までの，SA^i に保持される過去の出力 $y^i_j(t-l), \ldots, y^i_j(t-1)$ に対する未知環境からの強化信号（ペナルティ評価）ベクトルとする。$\bar{d}^i_j(t)$ はその平均値である。ここでは，未知環境のペナルティ応答 $r^i(t) = 1$ について考える。式 (3.22) に示すように $\bar{d}^i_j(t)$ は出力 y^i_j に対して，時刻 $t-l$ から $t-1$ までの平均ペナルティ評価で，式 (3.23) に示すように $\bar{d}^i_k(t)$ は各出力の平均ペナルティ評価の最小値である。

強化法 T^R の特徴として，時刻 t での出力 y^i_j を選択する確率分布 $p^i_j(t)$ を修正する際に，他の出力の現在までの評価値も利用する点があげられる。また，未知環境からの離散値 $\{0, 1\}$ の強化信号（ここではペナルティ評価値）をそのまま採用するのではなく，その平均値，すなわち連続値を採用するので，未知環境としては連続値の強化信号を出力する S-model の挙動を呈する。

強化法 T^R の動特性 [12, 13]

強化法 T^R の動特性を解析する際に，平均ペナルティ関数の収束性の検証が重要となる。

SA^i における T^R 単体としての動特性を考察するために，以下の平均ペナルティ関数 $M(t)$ を定義する。

$$M(t) = \sum_{j=1}^{s^i} E\left[d^i_j(t)\right] p^i_j(t) \tag{3.24}$$

ここで，オペレータ $E[\cdot]$ は期待値を表す。

SA^i の強化学習プロセスの目標は $M(t)$ を最小化する確率分布 $p^i(t)$ を決定することである。T^R における $p^i(t)$ の単体制約条件を考えるために，ここで，以下の制限値域を導入する。

$$p^i_{\min} \le p^i_j(t) \le p^i_{\max}, j \in \{1, 2, \ldots, s^i\} \tag{3.25}$$

$$0 \le p^i_{\min} \le p^i_{\max} \le 1 \tag{3.26}$$

$$p^i_{\max} = 1 - (s^i - 1)p^i_{\min} \tag{3.27}$$

明らかに，もし，y^i_{j*} が最適出力であれば，p^i_{j*} を $p^i_{\max}$ に収束させ，$M(t)$ を最小化することが望ましい。つまり，強化法 T^R において，任意に与えられた μ，ρ に対して，ある時間定数 τ が常に存在し，すべての $t > \tau$ について，

$$Pr\Big(\sup_{t'>t} \left|p^i_{j*}(t') - p^i_{\max}\right| > \mu\Big) < \rho \tag{3.28}$$

が成立することを証明すべきである。

定理 3.1
実数の数列 $\theta(t)$ について，

$$\theta(t) > 0, \sum_{t=1}^{\infty} \theta(t) = \infty, \sum_{t=1}^{\infty} \theta^2(t) < \infty \tag{3.29}$$

とする。
もし，y^i_{j*} が最適出力，つまり，

$$\forall j \neq j*, E[d^i_{j*}(t)] < E[d^i_j(t)] \tag{3.30}$$

であれば，強化法 T^R において，

$$T^R(p^i_{j*}(t)) = p^i_{\max} \tag{3.31}$$

が成立する。

ここで，$p^i_{j*}(t)$ は時刻 t における最適出力 y^i_{j*} の選択確率を表し，$T^R(p^i_{j*}(t))$ は強化法 T^R による $p^i_{j*}(t)$ の学習結果を表す。定理 3.1 は，強化法 T^R による最適出力 y^i_{j*} の選択確率 $p^i_{j*}(t)$ の学習結果 $T^R(p^i_{j*}(t))$ が最大値 $p^i_{\max}$ に収束することを意味する。また，定理 3.1 は SA^i の単体としての収束性を保証するもので，文献 [13] と同じ手法で証明できる。この証明は付録に示す。

一般的に学習オートマトンの研究では，理論的解析に強化法 T^R，具体的な問題への応用には強化法 T^R から派生した R-I タイプ強化法，または R-P タイプ強化法が用いられることが多い。本研究では，R-P タイプ強化法を用いて検証を行う。ここで，確率オートマトンの強化法は，取り扱う例題の性質を考慮して，未知環境からリワード応答の強化信号を受けたときのみ出力選択確率を修正する R-I タイプ強化法ではなく，リワード応答のみならずペナルティ応答の強化信号を受けたときにも出力選択確率を修正する R-P タイプ強化法 [4, 13] を採用する。時刻 t において，SA^i が出力 y^i_k を選択し，未知環境から強化信号 $r^i(t)$ を受け取ったとする。このとき出力 y^i_k の選択確率 $p^i_k(t)$，および y^i_k 以外の出力 $y^i_j (j \neq k, j = 1, \ldots, s^i)$ の選択確率 $p^i_j(t)$ を，以下の R-P タイプ強化法で修正する。

(i) $r^i(t) = 0$（リワード応答時）

$$\begin{aligned} & p^i_k(t+1) \\ & = (1-\alpha)p^i_k(t) + \alpha = p^i_k(t) + \alpha(1 - p^i_k(t)) = p^i_k(t) + \alpha \sum_{j \neq k} p^i_j(t) \end{aligned} \tag{3.32}$$

$$\begin{aligned} & p^i_j(t+1) \\ & = (1-\alpha)p^i_j(t) = p^i_j(t) - \alpha p^i_j(t) \end{aligned} \tag{3.33}$$

(ii) $r^i(t) = 1$（ペナルティ応答時）

$$
\begin{aligned}
&p_k^i(t+1) \\
&= (1-\beta)p_k^i(t) = p_k^i(t) + (-\beta)p_k^i(t) = p_k^i(t) + (-\beta)\Big(1 - \sum_{j \neq k} p_j^i(t)\Big) \qquad (3.34)
\end{aligned}
$$

$$
\begin{aligned}
&p_j^i(t+1) \\
&= (1-\beta)p_j^i(t) + \frac{\beta}{s^i - 1} \\
&= p_j^i(t) - \beta p_j^i(t) + \frac{\beta}{s^i - 1} \\
&= p_j^i(t) - (-\beta)\Big(\frac{1}{s^i - 1} - p_j^i(t)\Big) \qquad (3.35)
\end{aligned}
$$

R-P タイプ強化法はリワード応答時を例に述べると，強化法 T^R において，$\delta_{j,k}^i(t) = p^i(t), \theta(t) = \alpha$ のように簡略化したものといえる。ここで，学習率は $\alpha = 0.01$，$\beta = 0.001$ とし，確率オートマトン同士の情報交換の時間間隔は 1 ステップとする。なお，R-P タイプ強化法の学習率 α, β は，強化法 T^R の学習率 $\theta(t)$ に対応し，リワード応答時には α が $\theta(t)$ に対応し，ペナルティ応答時には $-\beta$ が $\theta(t)$ に対応する。

3.2 ハイブリッド型協調的学習オートマトンチームモデル

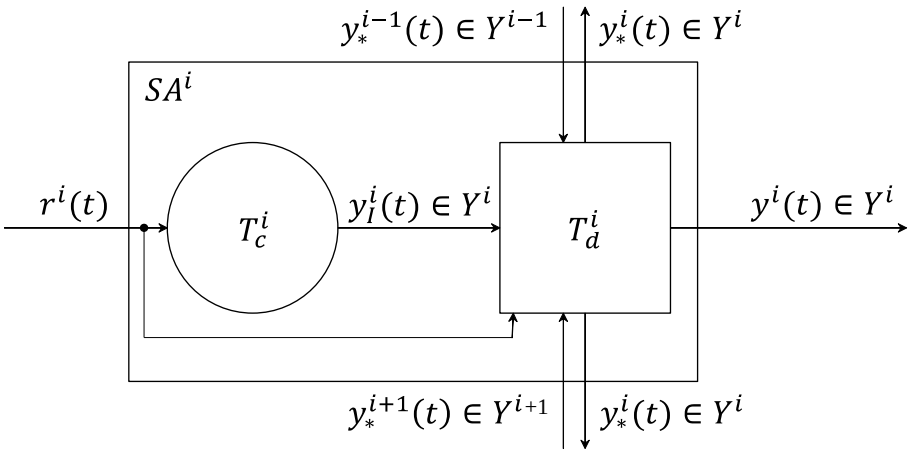

図 3.2: 確率オートマトンの構造

学習オートマトンチームを構成する各確率オートマトンは，具体的な応用問題に応じて様々な形を取ることができる。一方，応用問題によって，入力および出力に対して，離散の値に限りなく，連続的な入力および出力も多く存在している。本研究では，連続的な出力が必要である応用問題において，ハイブリッド型協調的学習オートマトンチームモデルを提案する。

本研究では，複数の確率オートマトンにより並列分散的に配置し，大域探索を行えるように，確率オートマトンの内部を図 3.2 のように構成する。また，論理的に探索領域を一

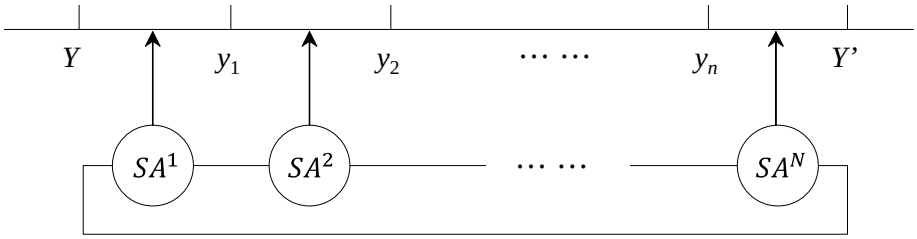

図 3.3: Parallel searching of y_{best} with LA^C_{TEAM}

直線で仮定し，$y_{best} \in [Y, Y']$ の最適値を推定するために，図 3.3 に示すようにその値域を複数（n 個）の部分空間（$[Y, y_1], [y_1, y_2] \ldots, [y_{n-1}, y_n], [y_n, Y']$）に分割し，複数の確率オートマトンでの同時推定を行う。複数の確率オートマトンは直線状に配列し，隣接する確率オートマトンのみ情報交換を行うようにする。

確率オートマトン SA^i は，自分の探索空間 $[y_{i-1}, y_i]$ 内で y^i_{best} の推定を行いながら，隣接する確率オートマトン SA^{i-1}, SA^{i+1} からの推定情報を参照して値域全体 $[Y, Y']$ での最適値を推定する。自分の探索空間内での探索は連続系強化法 T^i_c，自分の探索結果と隣接する確率オートマトンの探索結果から最適値を決めるためには離散系強化法 T^i_d を用いる。このように連続系強化法と離散系強化法の両方を用いる学習オートマトンを，本研究ではハイブリッド型学習オートマトンと呼ぶ。

時刻 t において，確率オートマトン SA^i は以下のように動作する。

1. まず，連続系強化法 T^i_c を利用して決定された出力選択確率分布に従い，ローカル探索に対応した出力 (y_{best} の推定値) を決める。本研究では，連続系強化法 T^i_c として CARLA（Continuous action reinforcement learning automata）[38] を採用する。CARLA の強化法は「3.2.1　ハイブリッド型協調的学習オートマトンチームモデルの個体表現」で詳細を述べる。また，他の詳細について，参考文献 [38] に参照する。
2. 自分の推定値と隣接する確率オートマトンから通知された推定値の中から最適値を決め，SA^i の出力として，未知環境に出力する。この際，各推定値を選択する確率分布を離散系強化法 T^i_d で修正する。本研究では，離散系強化法 T^i_d として，R-P 強化法 [1–4] を採用する。
3. 未知環境から強化信号 $r(t)$ を受け取り，各強化法 T^i_c, T^i_d を用いて出力選択確率の修正を行う。

3.2.1　ハイブリッド型協調的学習オートマトンチームモデルの個体表現

1. 強化信号
   時刻 t における，強化信号 $r^i(t) \in R^i$ は，SA^i の出力に対する未知環境側からの評価を表す。
2. 出力集合の表現

SA^i の出力集合を式 (3.36)-(3.38) のように定義する。

$$Y^i \equiv \{Y_I^i, Y_E^i\} \tag{3.36}$$

$$Y_I^i = [y_{\min}^i, y_{\max}^i] = [y_{\max}^{i-1}, y_{\min}^{i+1}] = [W_{i-1}, W_{i+1}] \tag{3.37}$$

$$Y_E^i = \{y_*^{i-1}, y_I^i, y_*^{i+1}\} \tag{3.38}$$

ここで，Y_I^i は SA^i のローカル探索に対応する出力集合で，W_{high} のローカル推定値で構成される連続空間となる。Y_E^i は SA^i の隣接 $SA \in \eta^i$，本研究では $\eta^i = \{SA^{i-1}, SA^{i+1}\}$ から得た情報に対応する出力集合である。また，y_*^{i-1}, y_*^{i+1} はぞれぞれ現時点で SA^{i-1}, SA^{i+1} の最良値であり，y_I^i は連続系強化法 T_c^i により決定された出力選択確率に従って選択された出力である。

SA^i の出力 $y^i(t) \in Y^i$ は離散系強化法 T_d^i により決定された出力選択確率で決定される。

3. 状態空間の表現

   SA^i の状態空間 Φ^i は SA^i のローカル探索に対応する状態空間 Φ_I^i と隣接確率オートマトン $\in \eta^i$ から得た情報に対応する状態空間 Φ_E^i から構成される。状態空間 Φ^i を式 (3.39) のように定義する。

$$\Phi^i \equiv \{\Phi_I^i, \Phi_E^i\} = \{\phi_1^i, \phi_2^i, \ldots, \phi_{s^i}^i\} \tag{3.39}$$

   ここで，状態空間 Φ_E^i だけは隣接確率オートマトン $\in \eta^i$ から参照可能である。

4. 出力選択確率分布の表現

   出力選択確率分布は状態空間上で定義される確率分布で，式 (3.40) のように定義する。

$$p^i \equiv \{p_c^i, p_d^i\} \tag{3.40}$$

   ここで，p_c^i は連続系強化法 T_c^i により修正される，連続探索空間上での推定値を出力する確率分布で，確率密度関数 $f_I^i(y,t)$ で表現される。一方，p_d^i は離散系強化法 T_d^i により修正される，自分の最適出力 y_I^i と隣接確率オートマトンから得た情報 $\{y_*^{i-1}, y_*^{i+1}\}$ を選択する確率分布である。

5. 出力関数の表現

   出力関数は状態空間から出力空間への写像として，$o^i : \Phi^i \to Y^i$ と定義される。学習オートマトンの研究では，状態空間と出力空間は一般に単射関係が成立するとしているので，簡単のために，各出力は出力選択確率分布 p^i により決定されると見なす。

6. 強化法

   SA^i の強化法 T^i を式 (3.41)-(3.43) のように定義する。

$$T^i = \{T_c^i, T_d^i\} \tag{3.41}$$

$$p_c^i(t+1) = T_c^i(y_I^i(t), r^i(t), p_c^i(t)) \tag{3.42}$$

$$p_d^i(t+1) = T_d^i(y_I^i(t), r^i(t), p_d^i(t)) \tag{3.43}$$

   離散系強化法 T_d^i による出力選択確率の更新式を式 (3.44)-(3.45) に示す。

$$p_j^i(t+1) = p_j^i(t) - \theta(t)\delta_{j,s}^i(t),\ j \neq k \tag{3.44}$$

$$p_s^i(t+1) = p_s^i(t) + \theta(t)\sum_{j \neq s}\delta_{j,s}^i(t) \tag{3.45}$$

連続系強化法 T_c^i では，$p_c^i(t+1)$ に対応する確率密度関数 $f_l^i(y,t+1)$ を式 (3.46) のように表現する [38]。

$$f(y,t+1) = \begin{cases} \alpha(t)[f(y,t) + G(t)H(y,r)] & (y \in (y_{\min}, y_{\max})) \\ 0 & (\text{otherwise}) \end{cases} \tag{3.46}$$

ここで，$G(t)$ はローカル探索に用いる評価関数である。また，$\alpha(t)$ は式 (3.47) のような正規化処理に関与するパラメータである。

$$\alpha(t) = \left(\int_{y_{\min}}^{y_{\max}} [f(y,t) + G(t)H(y,r)]dy \right)^{-1} \tag{3.47}$$

一方，$H(y,r)$ はガウシアン近傍関数（Symmetric Gaussian Neighborhood Function）で，式 (3.48) のように定義する。

$$H(y,r) = \lambda \exp\left(-\frac{(y-r)^2}{2\sigma}\right) \tag{3.48}$$

λ と σ はガウシアン近傍関数の平均値と分散を定めるパラメータで，式 (3.49),(3.50) のように定義する。

$$\lambda = \frac{g_h}{y_{\max} - y_{\min}} \tag{3.49}$$

$$\sigma = g_w(y_{\max} - y_{\min}) \tag{3.50}$$

ただし，g_h と g_w は学習速度と精度に関係するパラメータである。

3.2.2 未知環境および情報交換機構

ハイブリッド型協調的学習オートマトンチームモデルの未知環境と情報交換機構について，3.1 節の協調的学習オートマトンチームモデルの未知環境と情報交換機構の定義が同じため，ここでは省略する。

第4章

実験

4.1 最短経路探索問題を用いた検証実験

提案モデルである協調的学習オートマトンチームモデルの有効性を検証するために，ネットワークにおける分散型経路探索問題に提案モデルを適用する。具体的には，図 4.1 に示すメッシュネットワークにおける最短経路探索問題について考える。メッシュ内の各ノードは 8 近傍を持つように配置されている。送信元を S，宛先を T として，S から T に至る最短経路を探索する。メッシュネットワークの各ノードに確率オートマトンを 1 個ずつ配置し，確率オートマトン間の分散的協調により各確率オートマトンの出力選択確率が最短経路を選択するように収束することを示す。さらに，最短経路上のリンクを遮断した際，遮断リンクを回避した最短経路を選択するように，各確率オートマトンの出力選択確率が収束することを示す。

4.1.1 最短経路探索の実験グラフ

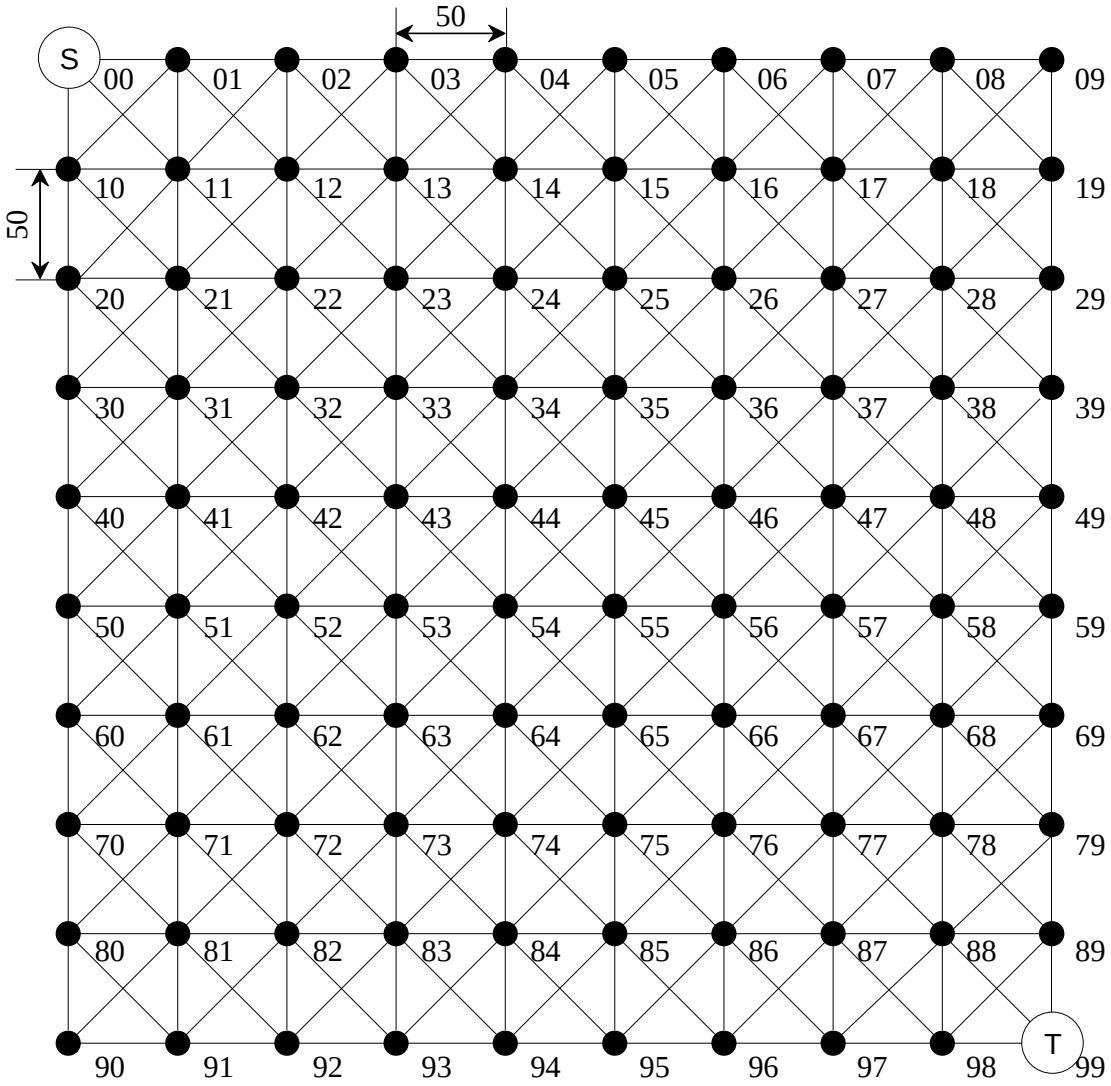

図 4.1: 最短経路探索の実験グラフ（メッシュ）[16, 36]

4.1.2 協調的学習オートマトンチームモデルの表現

SA^i の固有出力集合 Y_I^i を隣接ノードへのリンク番号に対応させ，強化信号 $r^i(t)$ を各隣接ノード $SA^j \in \eta^i$ を経由して宛先 T までの推定経路長 $D^i_{(j,T)*}$ で表現する。ここで，ノード間，すなわち SA^i と SA^j の間で交換される情報 h^{ji} を以下のように定義する

$$h^{ji} = D^j_{(j,T)*},\ j = 1, \ldots, |\eta^i|;\ SA^j \in \eta^i \tag{4.1}$$

ただし，$D^j_{(j,T)*}$ はノード SA^i の隣接ノードである SA^j が推定した，ノード SA^j から宛先 T までの最短経路長である。

ノード SA^i が情報 h^{ji} を受けると，自分から隣接ノード SA^j を経由して宛先 T までの推定経路長 $D^i_{(j,T)*}$ を以下のように算出する。

$$D^i_{(j,T)*} = D^i_{(i,j)} + D^j_{(j,T)*},\ j = 1, \ldots, |\eta^i|;\ SA^j \in \eta^i \tag{4.2}$$

ここで，$D^i_{(i,j)}$ はノード SA^i と隣接ノード $SA^j \in \eta^i$ の距離である。

4.1.3 実験 1　最短経路探索問題（環境不変）

実験環境

本問題では，横縦方向の $D^i_{i,j}$ は 50，斜め方向の $D^i_{i,j}$ は $50\sqrt{2}$ とする。

実験結果

始点（node0）から終点（node99）までの最短経路は

$$(S = 0, 11, 22, 33, 44, 55, 66, 77, 88, 99 = T)$$

である。

図 4.2 は始点（node0）の確率オートマトンの出力選択確率の収束の様子を示す。

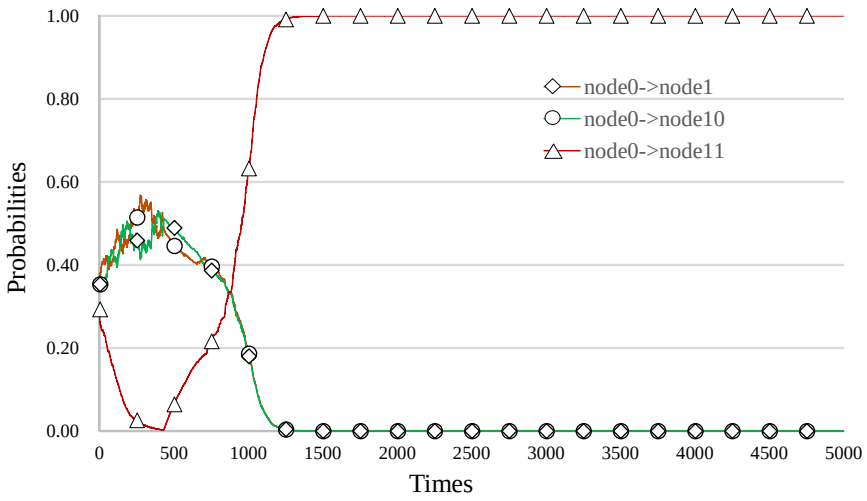

図 4.2: 始点（node0）の確率分布遷移

図 4.3 に中間点（node22）の確率オートマトンの出力選択確率の収束の様子を示す。

実験結果より，始点（node0）ではリンク（0,11），中間点（node22）ではリンク（22,33）すなわち最短経路上のノードを選択する確率が 1 に収束することがわかる。他のノードについても同様な結果を観測している。以上の結果により，最終的に，各ノード上に置かれた確率オートマトンの分散的協調により最適経路が構築されたことがわかる。

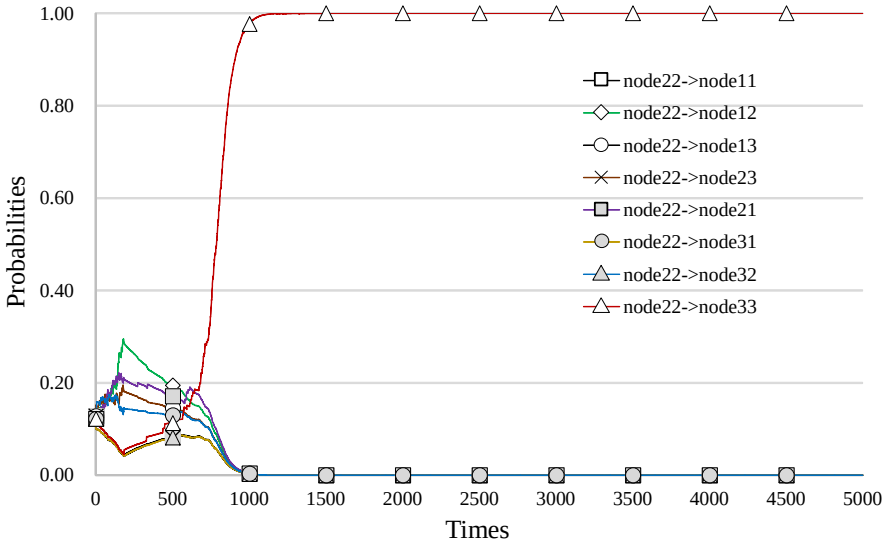

図 4.3: 中間点（node22）の確率分布遷移

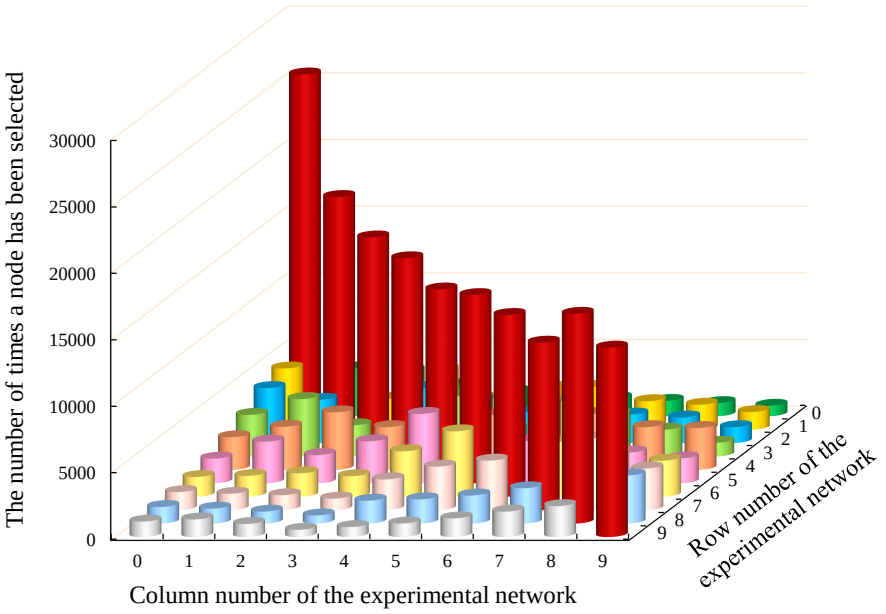

図 4.4: 各ノートの選択回数

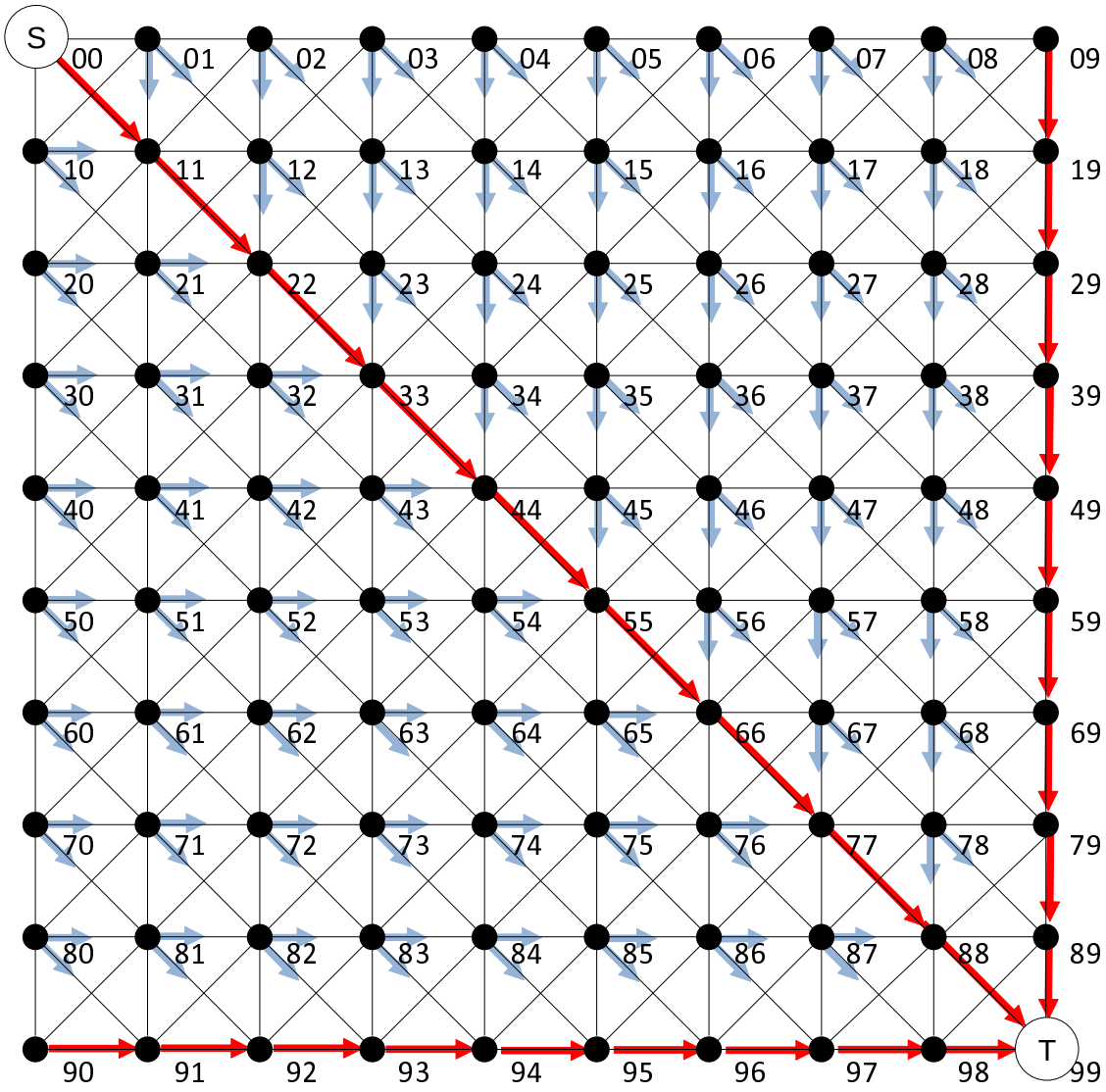

図 4.5: 各ノートにおいて，出力選択分布の様子

図 4.4 は提案モデルにより構築した最適経路を直観的に示すために，5,000 ステップまでに各ノードが選択された回数を示している。図 4.4 にみるように，最短経路上のノードの選択回数が多いことがわかる。また，最終的に，各ノードにおいて，出力選択分布の様子は図 4.5 のように示す。

4.1.4　実験 2　最短経路探索問題（経路可変）

実験環境

実験開始から 2,500 ステップを経過したところで，本来最短経路上にあるべきノード 44 とノード 55 間のリンクを遮断する。

実験結果

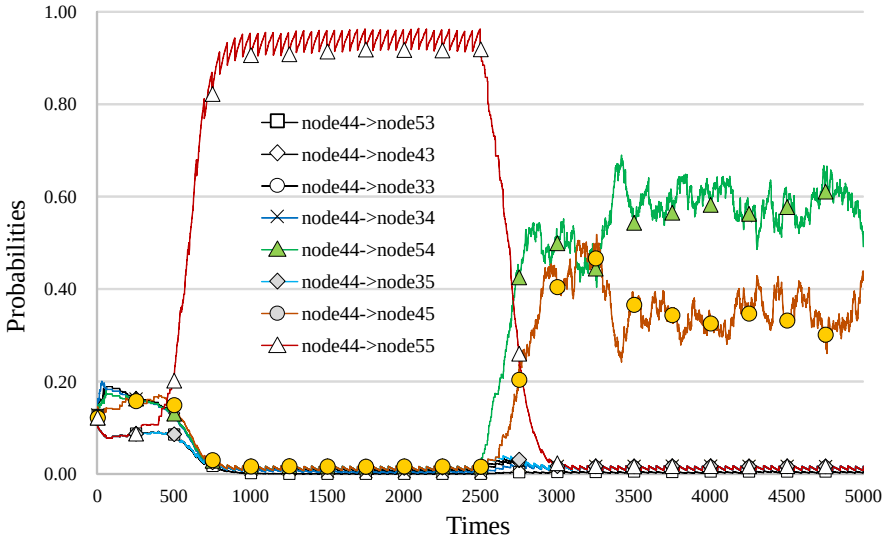

図 4.6: 遮断するリンクの始点（node44）の確率分布遷移

図 4.6 からわかるように，2,500 ステップまでに，ノード 44 では最適経路上にあるリンク（44,55）を選択する確率がおおむね 1 に収束しているが，リンク遮断が発生した後，代替経路の探索を始めている。このネットワークトポロジーの場合，宛先 T までに，ノード 44 からの代替経路として，リンク（44,45）とリンク（44,54）を経由する経路があるので，それぞれのリンクを選択する確率がおおむね 0.5 程度になることがわかる。また，図 4.7，図 4.8 からわかるように，すべてのノートにおいて，終点までの最短経路を確保しているため，リンク（44,55）を遮断した後に，代替経路を素早く見つけることができた。これにより，リンク変動が発生した場合でも，提案モデルの有効性を確認できる。

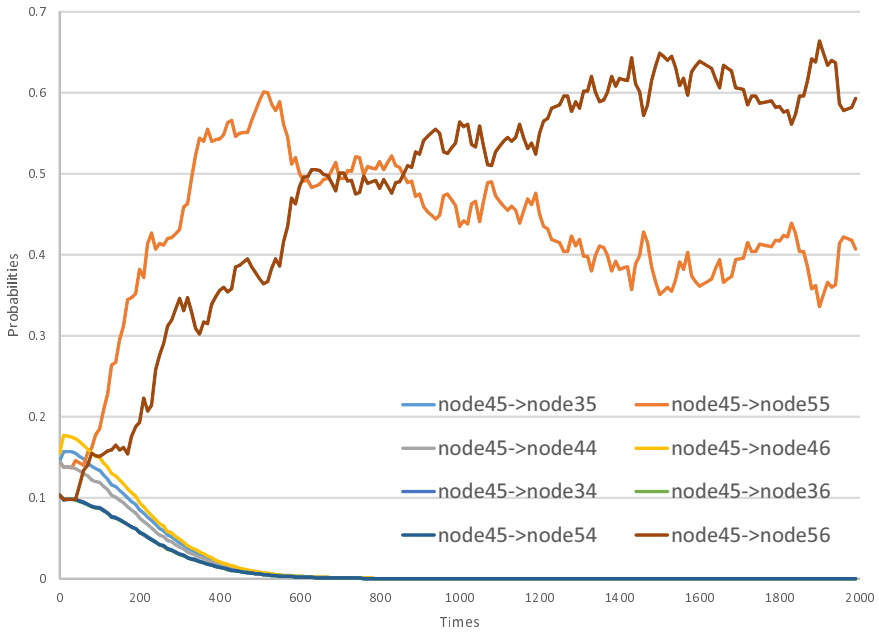

図 4.7: ほかの中間点（node45）の確率分布遷移

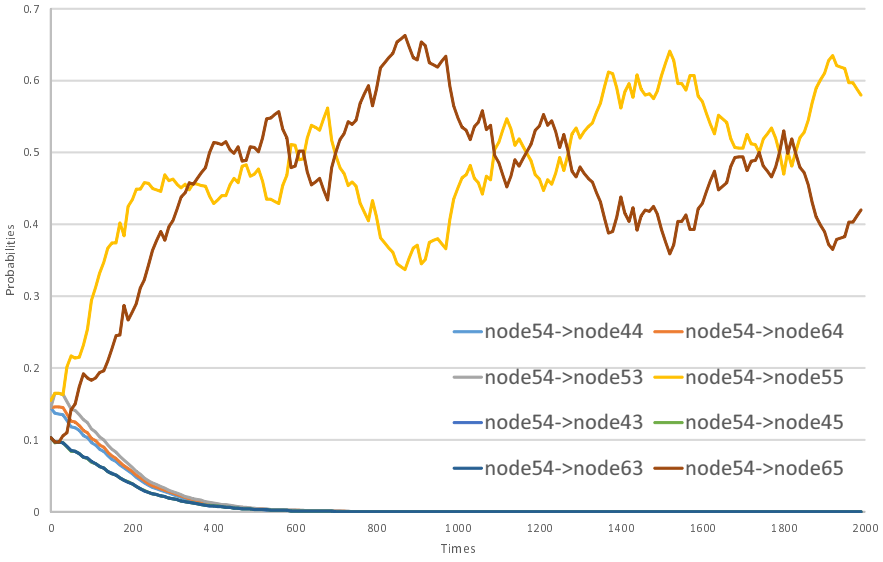

図 4.8: ほかの中間点（node54）の確率分布遷移

4.2 輻輳制御を用いた検証実験

4.2.1 輻輳制御実験のトポロジー

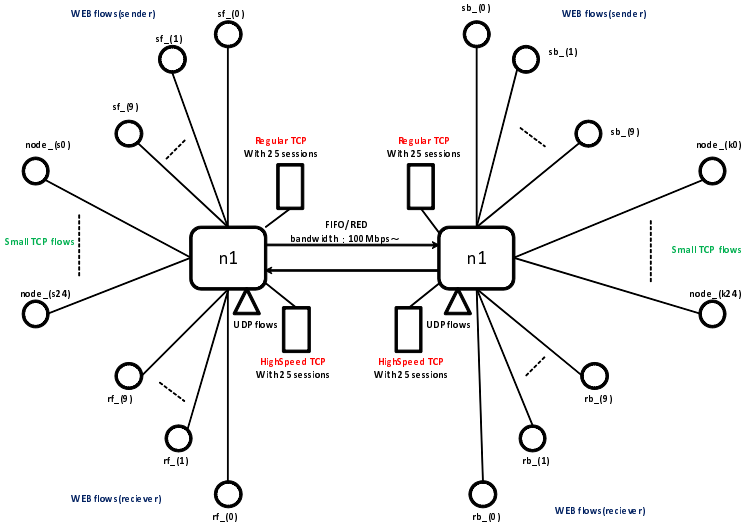

図 4.9: 輻輳制御実験のトポロジー

図 4.9 に輻輳制御実験に用いるトポロジーを示す。n1，n2 間のリンクをボトルネックリンクとて，帯域幅は各実験において 100Mbps から設定する。キューマネジメント方式は FIFO 方式または RED 方式を利用する。n1，n2 間では，HSTCP フローとレギュラー TCP フローを双方向からそれぞれ最大 50 本まで発生する。各フローの開始時間はランダムで自動的に設定する。さらに，n1，n2 の両サイドにそれぞれ，Small TCP（ここでは TCP Reno を利用する） ノードを 25 個，WEB 送受信ノードを 10 個ずつ用意する。また，n1，n2 間では UDP フローも発生できるように設定する。HighSpeed TCP に対して干渉効果を検証できるようにしている。本研究では，Network Simulator NS2 [39–41] を利用し，検証を行います。

4.2.2 ハイブリッド型学習オートマトンの協調的チームモデルの表現

出力集合の表現

W_{high} の変動域を $[W_1, W_2]$ とする。実験では，$W_1 = 8,300$（帯域幅 1Gbps の場合の W_{high} の推定値），$W_2 = 830,000$（帯域幅 100Gbps の場合の W_{high} の推定値）とする。まず，$[W_1, W_2]$ を 8 個の部分領域に対数値で等間隔に分割して，各確率オートマトンのローカル探索域（$[w_0, w_1], \ldots, [w_{n-1}, w_n]$）（図 3.3 の y_1, y_2 などに相当）とする。時刻 t において，確率オートマトン SA^i は，連続系強化法 T_c^i を利用したローカル探索で決定された値 $w^i(t) \in [w_{i-1}, w_i]$ と隣接する確率オートマトン（SA^{i-1}, SA^{i+1}）から得た隣接領域での探索結果 $w^{i-1}(t) \in [w_{i-2}, w_{i-1}], w^{i+1}(t) \in [w_{i+1}, w_{i+2}](y_*\ i-1, y_*^{i+1}$ に相当) に対して，さらに，離散

系強化法 T_d^i を利用して，3 個のうち最も適した値（SA^i の出力 $y^i(t)$ に対応）を出力として決定する。この場合，SA^i の出力集合は，ローカル探索値域 $[w_{i-1}, w_i]$ と 2 個の隣接情報 $Y_E^i = \{w^{i-1}(t), w^{i+1}(t)\}$ から構成される。

出力選択確率分布の表現

離散系強化法 T_d^i に対した出力選択確率分布 p_d^i は隣接確率オートマトンからの情報による出力選択確率 $[p^{i-1}, p^{i+1}]$ と連続系強化法に対する出力選択確率から構成する。連続系強化法に対する出力確率分布は確率密度関数で表現する。

未知環境の表現

送受信ペア間での帯域幅の変化を推定するために，輻輳ウィンドウサイズの変化と再転送率の変化に着目して，未知環境の表現を行う。さらに，輻輳ウインドウサイズの変化量と再転送率の変化量が大幅に異なると，各変化量の値をそのままで使用した場合，送受信ペア間の帯域幅の変化を推定できないと考えられるため，本研究では，輻輳ウインドウサイズと再転送率それぞれの変化の状況を変化量の値そのものの代わりに変化量の時間的な増減パターンで表し，S-model の学習オートマトンを採用して，未知環境の強化信号 $r(t)$ を表現する。ここで，時刻 t における輻輳ウィンドウサイズの変化を $\Delta w(t)$，再転送パケット数の変化を $\Delta e(t)$ とする。

$$
\begin{aligned}
f_1(t) &= \begin{cases} 1, \Delta w(t) > \Delta w(t-1) \\ 0, \Delta w(t) = \Delta w(t-1) \\ -1, \Delta w(t) < \Delta w(t-1) \end{cases} \\
f_2(t) &= \begin{cases} 1, \Delta e(t) > \Delta e(t-1) \\ 1, (\Delta e(t) = \Delta e(t-1)) \,\&\, (\Delta e(t) = 0) \\ 0, (\Delta e(t) = \Delta e(t-1)) \,\&\, (\Delta e(t) \neq 0) \\ -1, \Delta e(t) < \Delta e(t-1) \end{cases} \\
r(t) &= [1 + \exp(\theta_1 f_1(t) + \theta_2 f_2(t))]^{-1} \\
&\theta_1, \theta_2 \in [0,1], \theta_1 + \theta_2 = 1
\end{aligned}
\tag{4.3}
$$

強化法

離散系強化法 T_d^i では，R-P 型強化法 [1]〜 [4] を採用する。出力選択確率分布の更新式を式 (4.5),(4.6) に示す。

$$
\begin{aligned}
r(t) &= 0 \\
&\quad p_i(t+1) = (1-\alpha)p_i(t) + \alpha \\
&\quad p_j(t+1) = (1-\alpha)p_j(t) \\
&\quad j \neq i, j = 1, 2, \ldots, s \\
r(t) &= 1 \\
&\quad p_i(t+1) = (1-\beta)p_i(t) \\
&\quad p_j(t+1) = (1-\beta)p_j(t) + \frac{\beta}{s-1} \\
&\quad j \neq i, j = 1, 2, \ldots, s
\end{aligned}
\tag{4.4, 4.5}
$$

一方，連続系強化法 T_c^i は，出力選択確率分布，すなわち確率密度関数を式 (3.46) によって更新する。ただし，評価関数 $G(t)$ は，式 (4.4) の $r(t)$ とする。

また，連続系強化法 T_c^i については，$y_{\min} = W_1, y_{\max} = W_2$，$g_w = 0.02, g_h = 0.3$ とする。離散系強化法 T_d^i については，$\alpha = 0.01$，$\beta = 0.001$ とする。

4.2.3 実験 3　TCP Reno の輻輳ウィンドウ変動

実験検証用のトポロジー図 4.9 を用いて，TCP Reno の輻輳ウィンドウ変動について検証実験を行う。

実験環境

ここで高速回線における TCP Reno の挙動を検証するために，以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数:　1 本
- HighSpeed TCP を発生する本数：0 本
- 帯域幅：　1Gbps
- 転送遅延時間：　50ms
- シミュレーション時間：　500s

実験結果

TCP Reno の輻輳ウィンドウの変動について，実験結果を図 4.10 に示す。

実験結果より，帯域幅 1Gbps において，輻輳時間間隔は 7 分程度になる，帯域使用率が低くなることがわかる。ここで，輻輳時間間隔とは輻輳が発生した後，最大スループットを得るまでにかかる時間である。

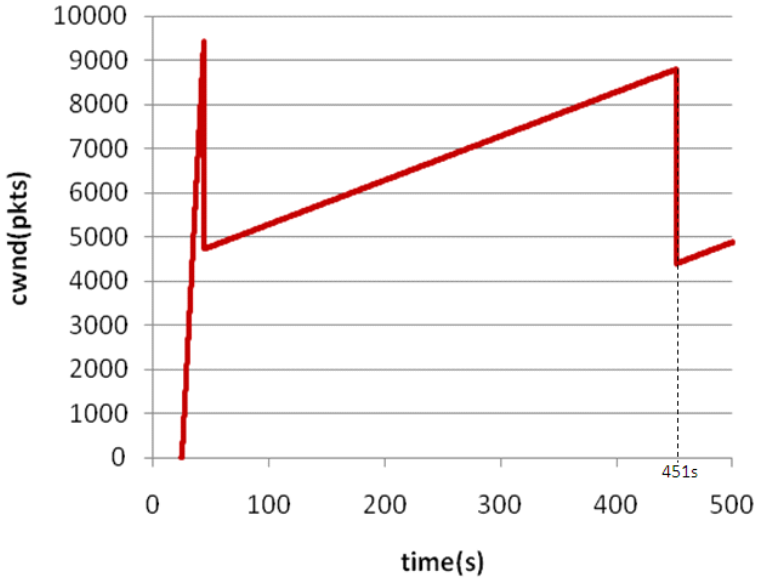

図 4.10: TCP Reno のウィンドウサイズ

4.2.4 実験 4　TCP Reno の回線利用率およびパケット損失率

実験検証用のトポロジー図 4.9 を用いて，TCP Reno の回線使用率とパケット損失率について検証実験を行う。

実験環境

以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数:　10 本
- HighSpeed TCP を発生する本数：0 本
- 帯域幅:　200Mbps，500Mbps，700Mbps，1Gbps，5Gbps，7Gbps，10Gbps
- 転送遅延時間：　50ms
- シミュレーション時間：　200s

実験結果

TCP Reno の回線使用率の実験結果は図 4.11 に示す。

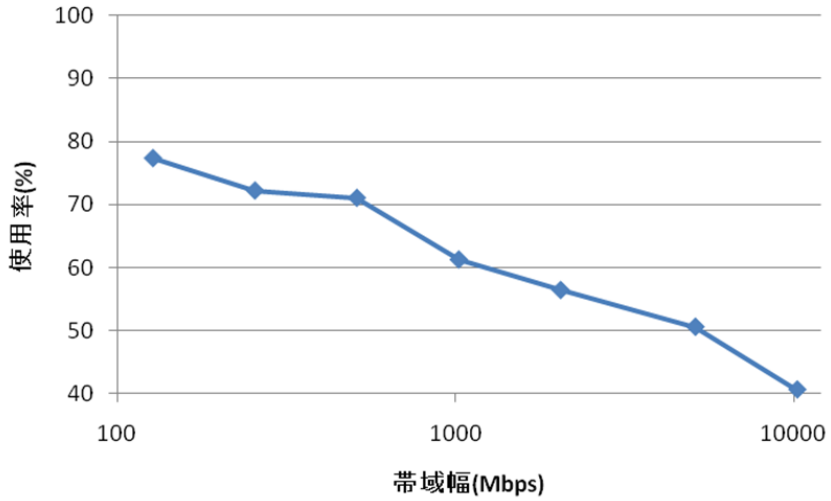

図 4.11: TCP Reno の回線使用率

実験結果より，帯域幅が 1Gbps に超える場合，回線使用率が低下することが判明した。ここで，図 4.12 に示すパケット損失率を確認したところで，以下のことを判明した。つまり，シミュレーション時間内に最大スループットを達成していないと推測され，輻輳が一回も発生していないと考えられる。

4.2.5 実験 5 HighSpeed TCP のウィンドウ変動

実験検証用のトポロジー図 4.9 を用いて，HighSpeed TCP のウィンドウ変動について検証実験を行う。

実験環境

高速回線における HighSpeed TCP の挙動を検証するために，以下の実験環境において検証実験を行う。

- TCP Reno を発生する本数: 0 本
- HighSpeed TCP を発生する本数：1 本
- 帯域幅： 1Gbps
- 転送遅延時間： 50ms
- シミュレーション時間： 500s

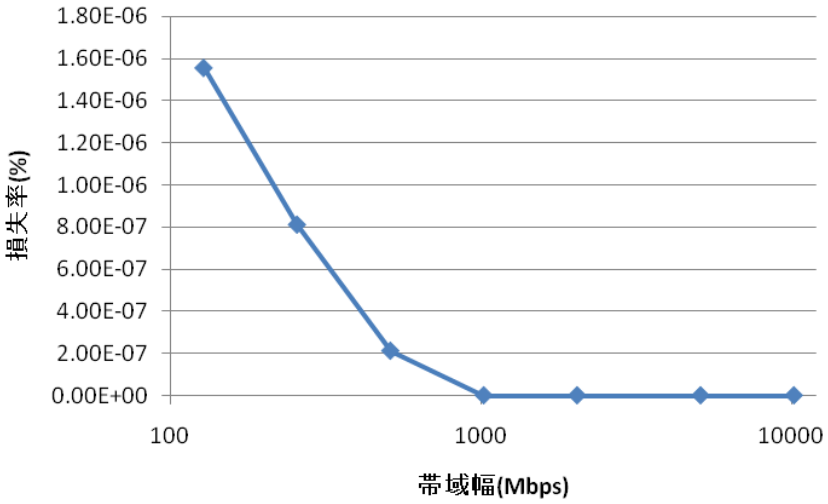

図 4.12: TCP Reno のパケット損失率

実験結果

HighSpeed TCP のウィンドウ変化に関して，実験結果を図 4.13 に示す。

実験結果より，輻輳ウィンドウサイズが速く収束することが判明した。また，TCP Reno に比べると輻輳時間間隔が短くなることを確認できた。

4.2.6 実験 6 HighSpeed TCP と TCP Reno の回線使用率

実験検証用のトポロジー図 4.9 を用いて，HighSpeed TCP と TCP Reno の回線使用率の遷移について検証実験を行う。

実験環境

以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数: 10 本
- HighSpeed TCP を発生する本数：10 本
- 帯域幅: 100Mbps，500Mbps，1Gbps，2Gbps，5Gbps，10Gbps
- 転送遅延時間： 50ms
- シミュレーション時間： 200s

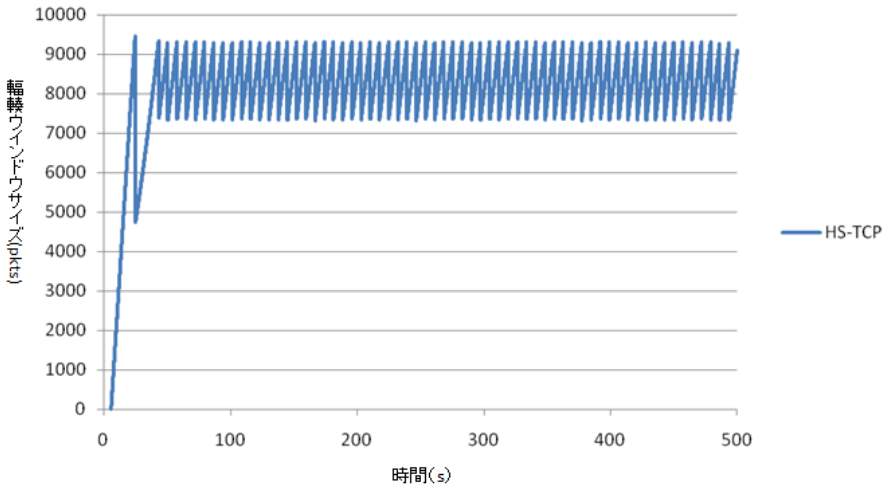

図 4.13: HighSpeed TCP のウィンドウサイズ

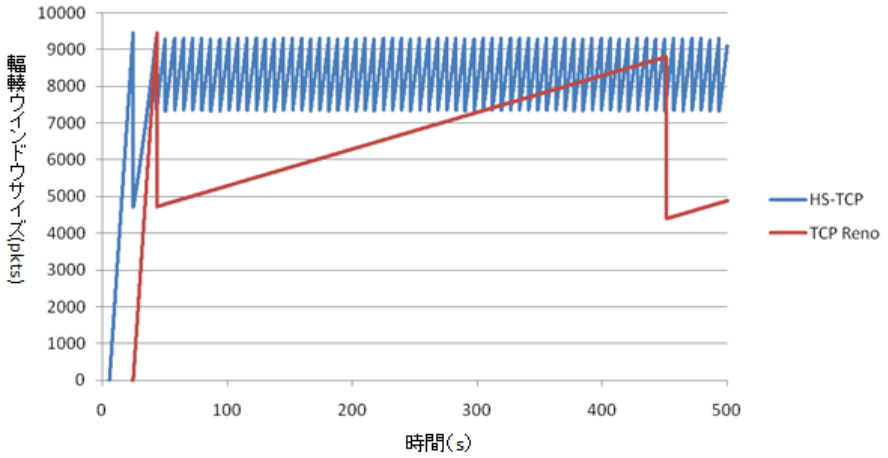

図 4.14: ウィンドウサイズ（HighSpeed TCP と TCP Reno の比較）

実験結果

HighSpeed TCP と TCP Reno の回線使用率の遷移に関して，実験結果を図 4.15 に示す。

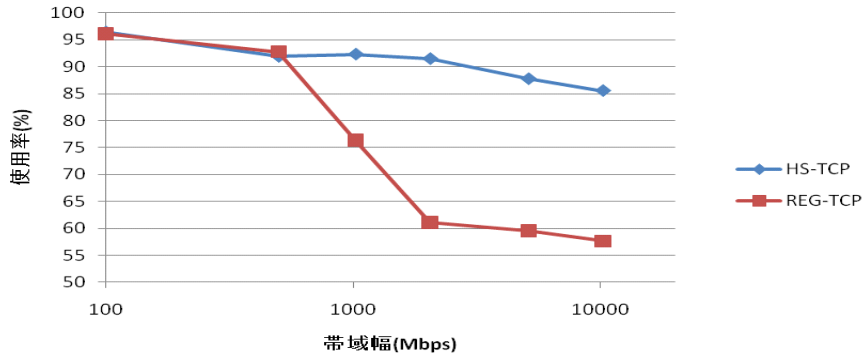

図 4.15: リンク使用率（HighSpeed TCP と TCP Reno の比較）

実験結果より，帯域幅は 1Gbps 以上の場合，HighSpeed TCP は TCP Reno より帯域使用率が大幅に改善されることができる。TCP Reno の輻輳制御方式では帯域使用率は帯域幅の増大に反比例して低下する問題点をある程度に改善したできたと考えられる。

4.2.7　実験 7　HighSpeed TCP の適応性

実験検証用のトポロジー図 4.9 を用いて，帯域幅に対する HighSpeed TCP の適応性について検証実験を行う。

実験環境

以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数:　0 本
- HighSpeed TCP を発生する本数：10 本
- 帯域幅：　1Gbps，2Gbps，5Gbps，7Gbps，10Gbps，20Gbps，50Gbps，70Gbps，100Gbps
- 転送遅延時間：　50ms
- シミュレーション時間：　200s

実験結果

帯域幅に対する HighSpeed TCP の適応性を図 4.16 を示す。

実験結果より，帯域幅は 10Gbps 以上になると帯域使用率が低下する問題が存在することが判明した。その原因は HighSpeed TCP が帯域幅の変動に適応できないため，帯域使用率が低下したためと考えられる。

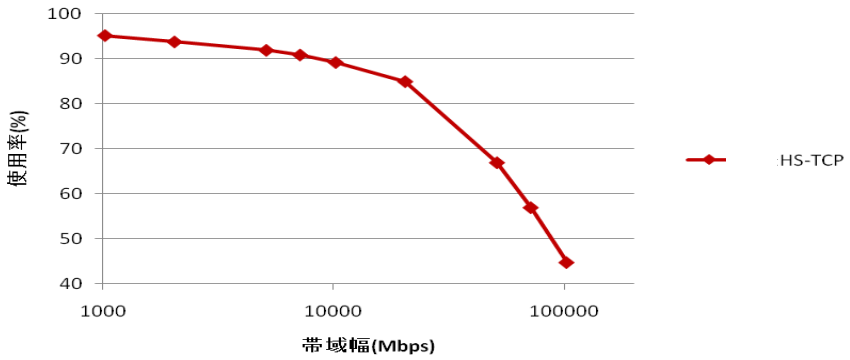

図 4.16: HighSpeed TCP のリンク使用率

4.2.8 実験 8 提案手法を用いた HighSpeed TCP のウィンドウ変動

実験検証用のトポロジー図 4.9 を用いて，帯域幅 1Gbps の場合，提案手法を用いた HighSpeed TCP のウィンドウ変動について検証実験を行う。

実験環境

以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数: 0 本
- HighSpeed TCP を発生する本数：0 本
- 提案手法の HighSpeed TCP を発生する本数：1 本
- 帯域幅： 1Gbps
- 転送遅延時間： 50ms
- シミュレーション時間： 100s

実験結果

帯域幅 1Gbps の場合，提案手法を用いた HighSpeed TCP のウィンドウ変動を図 4.17 に示す。

4.2.9 実験 9 提案手法を用いた HighSpeed TCP のウィンドウ変動

実験検証用のトポロジー図 4.9 を用いて，帯域幅は 50Gbps の場合，提案手法を用いた HighSpeed TCP のウィンドウ変動について検証実験を行う。

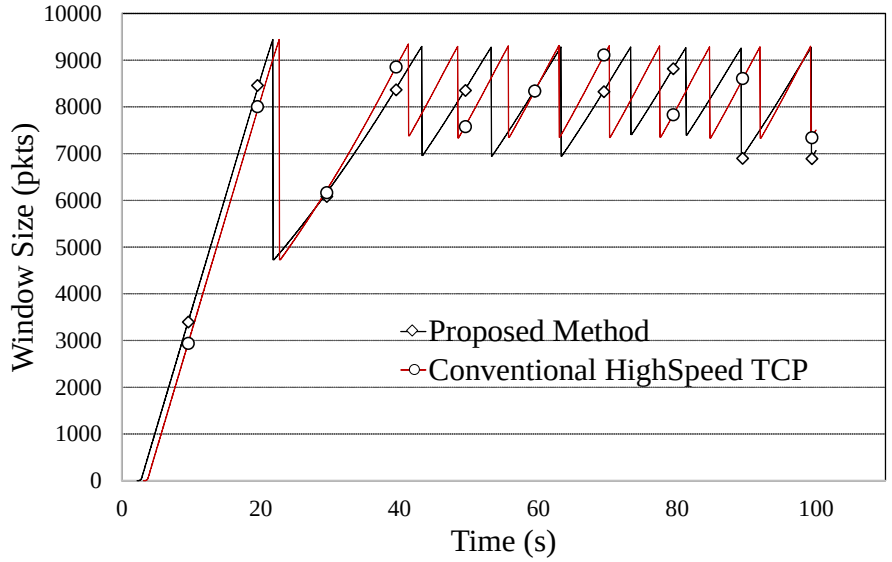

図 4.17: 提案手法を用いた HighSpeed TCP のウィンドウ変動（1Gbps）

実験環境

以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数: 0 本
- HighSpeed TCP を発生する本数：0 本
- 提案手法の HighSpeed TCP を発生する本数：1 本
- 帯域幅： 50Gbps
- 転送遅延時間： 50ms
- シミュレーション時間： 100s

実験結果

帯域幅 1Gbps の場合，提案手法を用いた HighSpeed TCP のウィンドウ変動を図 4.18 に示す。図 4.17，図 4.18 は帯域幅がそれぞれ 1Gbps 時, 50Gbps 時の輻輳ウィンドウの変動を表す。図 4.17 では提案手法は従来の HighSpeed TCP とあまり差がない。一方，図 4.18, すなわち帯域幅が 50Gbps の場合，静的なパラメータを採用した従来の HighSpeed TCP に比べ，提案手法の方が収束速度が大きく，早く安定することがわかる。これは従来の HighSpeed TCP では，提案手法のように，リンク帯域幅の変動に応じてパラメータ W_{high} を動的に調整するのではなく，10Gbps を対象にした推奨値（$W_{high} = 83,000$）を利用するからである。

4.2.10 実験 10 提案手法を用いた HighSpeed TCP の適応性

実験検証用のトポロジー図 4.9 を用いて，帯域幅に対する提案手法を用いた HighSpeed TCP の適応性について検証実験を行う。

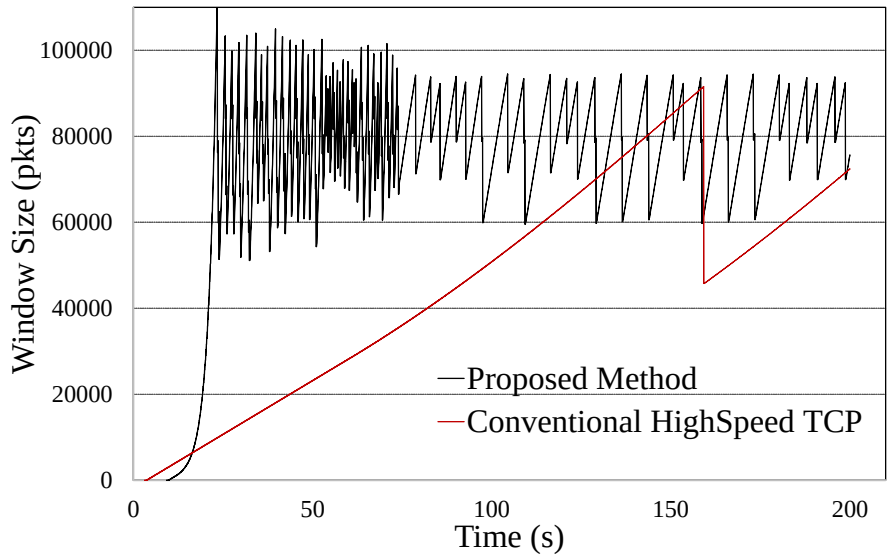

図 4.18: 提案手法を用いた HighSpeed TCP のウィンドウ変動（50Gbps）

実験環境

以下の実験環境において，検証実験を行う。

- TCP Reno を発生する本数: 0 本
- HighSpeed TCP を発生する本数：0 本
- 提案手法の HighSpeed TCP を発生する本数：1 本
- 帯域幅： 1Gbps，2Gbps，5Gbps，7Gbps，10Gbps，20Gbps，50Gbps，70Gbps，100Gbps
- 転送遅延時間： 50ms
- シミュレーション時間： 500s

実験結果

帯域幅に対する提案手法を用いた HighSpeed TCP の適応性を図 4.19 に示す。

図 4.19 は輻輳制御機構の帯域変動耐性を表す。帯域の変動に対して，従来の HighSpeed TCP の帯域使用率が大きく低下するのに対して，提案手法はほとんど低下しない。

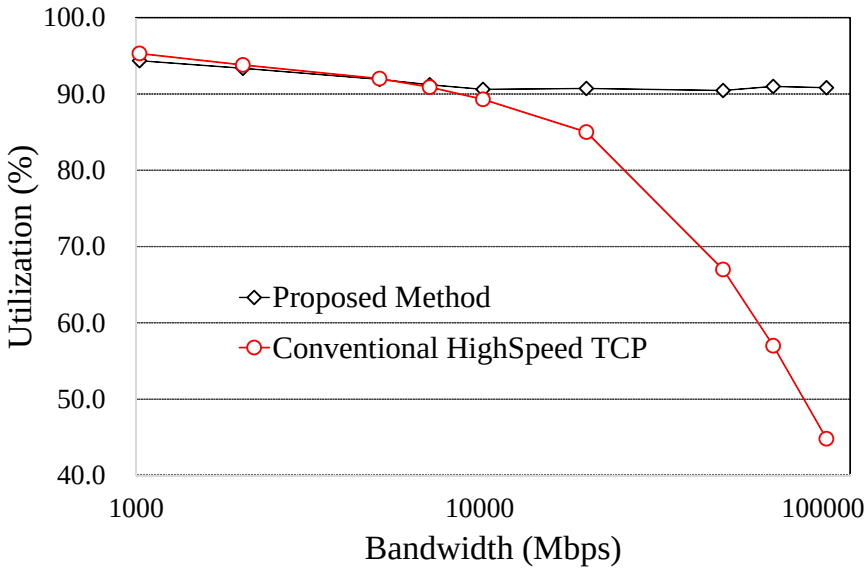

図 4.19: 提案手法を用いた HighSpeed TCP のリンク使用率

第5章

むすび

本研究では，マルチエージェントシステムをモデル化するための協調的学習オートマトンチームモデルを提案した。このモデルでは，従来の学習オートマトンチームモデルに隣接集合と情報交換機構を導入し，確率オートマトンの状態空間，出力集合，出力選択確率分布を内部と外部に特徴づけて，強化学習プロセスを遂行している。さらに，連続値を持つ確率オートマトンを用いたハイブリッド型学習オートマトンチームモデルを提案することによって，離散型オートマトンチームモデルで学習しにくい問題も解決することができた。

本研究では，提案モデルの有効性を示すために，簡単なメッシュネットワークにおける分散型経路探索問題について，$LA^C_{TEAM}(T^{R-P})$ を適用してシミュレーション検証を行った。実験結果からわかるように，各ノード上に置かれた確率オートマトン間の協調関係が局所的に確立され，大域的最適化の目標を実現できた。また，ハイブリッド型学習オートマトンチームモデルの有効性を検証するために，HighSpeed TCP の輻輳制御問題を用いて，シミュレーション検証を行った。実験結果からわかるように，HighSpeed TCP の帯域使用率が大幅に改善することができた。二つの応用問題を用いて，提案手法の有効性を検証することができた。

今後の課題として，以下があげられる。まず，情報交換機構について，本研究では，直接情報交換方式の例を示したが，並列分散的環境上で実装する場合，通信プロトコルと情報フォーマットの定義が必要となる。

一方，並列分散環境における非同期的集団行動について考える場合，間接的情報交換方式の表現が重要となる [16, 36]。その場合，各個体は交換したい情報を何らかの方法で環境に残すなどの方法が考えられる。隣接個体は自分のタイミングで環境に残される情報を参照する [42]。直接情報交換方式に比べ，この方式は自由度は高く柔軟性があると考えられるが，環境に情報を残す方法の定義と情報表現が必要となる。

また，システム全体の収束性について，本研究では，強化学習プロセスにおいて，ある状態で最小ペナルティ確率に対応する強化信号が存在する場合の各確率オートマトンの強化法 T^R の収束性を示したが，LA^C_{TEAM} 全体の収束性を検証する場合，さらに，以下のことを考える必要がある [12, 13]。全体の制御目標を表す評価関数 $U(x^1, x^2, \ldots, x^N)$ において，LA^C_{TEAM} の出力ベクトル

$(x^1_*, x^2_*, \ldots, x^N_*), x^j_* \in Y^j, j = 1, \ldots, N$

が存在して，以下の式が満たされることが要求される。

$$U(x_*^1, x_*^2, \ldots, x_*^N) = \underset{\forall x^1 \in Y^1, \ldots, \forall x^N \in Y^N}{Optimum} \{U(x^1, x^2, \ldots, x^N)\}$$

ここで，Optimum は最適を意味する。このシステム全体の動特性の理論検証については，将来の研究課題とする。

謝辞

本研究を進めるにあたり，直接御指導いただきました千葉大学大学院工学研究科 平田廣則名誉教授，小圷 成一教授，岡本 卓准教授に深く御礼申し上げます。また，日ごろより有益なご助言を賜りました関東学院大学大学院工学研究科 銭 飛教授に深く感謝致します。さらに，日頃から大変お世話になりました当研究室（システム数理教育研究分野）の皆様に篤く御礼申し上げます。

参考文献

[1] K.S. Narendra and M.A.L. Thathachar: Learning automata - A survey, *IEEE Trans. Systems, Man, and Cybernetics*, Vol.4, No.4, pp.323-334 (1974)

[2] S. Lakshmivarahan: Learning Alogorithms Theory and Applications, *Springer-Verlag* (1981)

[3] N. Baba: New Topics in Learning Automata Theory and Applications, *Springer-Verlag* (1985)

[4] Narendra, Kumpati S. and Mandayam AL Thathachar: Learning Automata: An Introduction, *Prentice-Hall, Inc.* (1989)

[5] M.A.Arbib.: Theories of Abstrct Automata, *Prentice-Hall,Inc.* (1969)

[6] E.F.Codd.: Cellular Automata, *Academic Press* (1968)

[7] S.Wolfram.: Theory and applications of Cellular Automata, *World Scientific* (1986)

[8] Von Neumann J.: Theory of Self-Reproducing Automata, *Univ. of Illinois Press* (1966)

[9] 高玉圭樹:「マルチエージェント学習 - 相互作用の謎に迫る - 」，コロナ社 (2004)

[10] Obaidat, Mohammad S., Georgios I. Papadimitriou, and Andreas S. Pomportsis: Guest editorial learning automata: theory, paradigms, and applications, *IEEE Trans. Systems, Man, and Cybernetics, Part B*, Vol.32, No.1, pp.706-709 (2002).

[11] H.Une, F.Qian and H. Hirata, Adaptive Routing Algorithm for Network Load Balancing, *IEEJ Transactions on Electrical and Electronic Engineering*, Vol.6, No.5, pp.441-449 (2011)

[12] 銭飛 :「学習オートマトンの集団モデルに関する研究」, 千葉大学大学院自然科学研究科博士論文 (1991)

[13] 銭飛，平田廣則:「分散型強化学習システム : 学習オートマトンのチームモデル」，電学論 C，Vol.118，No.5，pp.797-793 (1998)

[14] 三上貞芳，皆川雅章 :「強化学習」，森北出版 (2000)

[15] 電気学会 GA・ニューロを用いた学習とその応用調査専門委員会 :「学習とそのアルゴリズム」，森北出版 (2002)

[16] 汪岑：「強化学習を用いたHighSpeed TCPの輻輳制御に関する研究」，関東学院大学大学院工学研究科電気工学専攻修士論文 (2011)

[17] 汪岑，塩崇裕，銭飛：「強化学習を用いたHighSpeed TCPの輻輳制御」，平成22年電気学会電子・情報・システム部門大会，pp.1272-1273 (2010)

[18] 白石和章：「学習オートマトンを用いた自律分散制御の応用に関する研究」，千葉大学大学院自然科学研究科人工システム科学専攻電子機械システム博士論文 (2005)

[19] Dowling J., Curran E., Cunningham R. and Cahill V.: Using Feedback in Collaborative Reinforceme Learning to Adaptively Optimize MANET Routing, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol.35, No3, pp.360–372 (2005)

[20] 茨木俊秀，永持仁，石井利昌：「グラフ理論　-　連結構造とその応用　-」，朝倉書店 (2010)

[21] J. A. Bondy and U. S. R. Murty：「グラフ理論への入門」，共立出版 (1991)

[22] Misra Sudip and B. John Oommen: An Efficient Dynamic Algorithm for Maintaining All-Pairs Shortest Paths in Stochastic Networks, *IEEE Trans. Computers*, Vol.55, No.6, pp.686-702 (2006)

[23] C. Demetrescu G.F. Italiano: A New Approach to Dynamic All Pairs Shortest Paths, *Proc. 35th Ann. ACM Symp. Theory of Computing*, pp.159-166 (2003)

[24] Sally Floyd: RFC 3742: Limited Slow-Start for TCP with Large Congestion Windows, *http://www.ietf.org/rfc/rfc3742.txt* (2004)

[25] Sally Floyd: RFC 3649: HighSpeed TCP for Large Congestion Windows, *http://www.ietf.org/rfc/rfc3649.txt* (2003)

[26] 池坊屋：「組合せ最適化問題における学習オートマトン計算法に関する研究」, 広島国際学院大学大学院工学研究科博士論文 (2003)

[27] K. Doya K. Samejima K. Katagiri K. Kawato: Multiple model-based reinforcement learning, *Neural Comput.*, Vol.14, No.6, pp.1347-1369 (2002)

[28] Anatasios A. Economides: Multiple Response Learning Automata, *IEEE Transactions on Systems,Man, and Cybernetics, Part B*, Vol.26, No.1, pp.153-156 (1996)

[29] Geir Horn and B. john Oommen: Solving Multiconstraint Assignment Problems Using Learning Autiomata, *IEEE Transactions on Systems,Man, and Cybernetics, Part B*, Vol.40, No.1, pp.6-18 (2010)

[30] Farnaz Abtahi and Mohammad Reza Meybobi: Solving Multi-Agent Markov Decision Processes Using Learning Automata, *Intelligent Systems and Informatics, 2008. SISY 2008. 6th International Symposium on IEEE* (2008)

[31] K. S. Fu and G. J. McMurtry: A Study of Stochastic Automata as a Model for Learning and Adaptive Controllers, *EEE Trans. Automatic Control AC-11*, pp.379-387 (1966)

[32] 奥山徹 :「TCP のしくみと実装―RFC の詳細から実装系の解析まで」, CQ 出版 (2004)

[33] Fall Kevin and Sally Floyd: Simulation-based Comparisons of Tahoe, Reno and SACK TCP, *ACM SIGCOMM Computer Communication Review*, Vol.26, Issue 3,pp.5- 21 (1996)

[34] FreeS/WAN, *http://www.freeswan.org*

[35] 汪岑，丁亮，水島茂雄，銭飛 :「強化学習による High Speed TCP の公平性問題の改善」, 平成 23 年　電気学会電子・情報・システム部門大会，pp.1282-1283 (2011)

[36] 汪岑，小圷成一，岡本卓，銭飛 :「マルチエージェントシステムのモデル化のための協調的学習オートマトンチームモデル」, 電学論 C，Vol.137，No.5，pp.759-767 (2017)

[37] 銭飛，平田廣則 :「可変構造学習オートマトンネットワーク」, 電気学会論文誌 C, Vol.118，No.5，pp333-338 (1998)

[38] Howell, M. N. and T.J.Gordon: Continuous action reinforcement learning automata and their application to adaptive digital filter design, *Engineering Applications of Artificial Intelligence*, Vol.14, pp.549-561 (2001)

[39] 銭飛 :「NS2 によるネットワークシミュレーション - 実験で学ぶ QoS ネットワーク技術」, 森北出版　(2006)

[40] 水野秀樹 :「S2 によるネットワークシミュレーション入門 - 有線からワイヤレスアドホックネットワークまで」, 森北出版 (2011)

[41] The Network Simulator - ns-2, *https://www.isi.edu/nsnam/ns/*

[42] 汪 岑，小圷 成一，岡本 卓，銭 飛 :「間接情報共有を用いた協調的学習オートマトンのチームモデル TCP の輻輳制御」, 平成 28 年　電気学会電子・情報・システム部門大会，pp.1213-1217 (2016)

[43] B. Venkata Ramana, B. S. Manoj and C. Siva Ram Murthy: Learning-TCP: A Novel learning Automata Based Reliable Transport Protocol for Ad hoc Wireless Networks, *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on IEEE* (2005)

[44] Hamid Beigy and Mohammad Reza Meybobi: Cellular Learning Automata With Multiple Learning Automata in Each Cell and Its Applications, *IEEE Transactions on Systems,Man, and Cybernetics, Part B*, Vol.40, No.1, pp.54-65 (2010)

[45] Shuli Gao, Sidney N. Givigi and Alain JG Beaulieu: FPGA Implementation of Multiple Pursuit-Evasion Games with Decentralized Learning Automata, *Systems Conference, 2014 8th Annual IEEE* (2014)

[46] C. Mariano E. Morales: A new distributed reinforcement learning algorithm for multiple objective optimization problems, *Proc. Adv. Artif. Intell. Int. Joint Conf. 7th Ibero-Amer. Conf. Artif. Intell. 15th Brazilian Symp. AI*, pp.290-299 (2000)

著者関連論文リスト

論文

1. 汪　岑，小圷　成一，岡本　卓，銭　飛 :「学習オートマトンを用いた HighSpeed TCP の輻輳制御」，電気学会論文誌 C，Vol.137，No2，pp.333-334 (2017)

2. 汪　岑，小圷　成一，岡本　卓，銭　飛 :「マルチエージェントシステムのモデル化のため の協調的学習オートマトンチームモデル」，電気学会論文誌 C，Vol.137，No5，pp.759-767 (2017)

国際会議

1. C. Wang, F. Qian, S. Koakutsu, T. Okamoto: Congestion control of highspeed TCP using the partial collaborative learning automaton team mode, *Proc. of SICE Annual Conference 2015, pp. 300-305 (2015)*

国内学会発表

1. 汪 岑，塩 崇裕，銭 飛 :「強化学習を用いた HighSpeed TCP の輻輳制御」，平成 22 年　電気学会電子・情報・システム部門大会，pp.1272-1273 (2010)

2. 汪 岑，銭 飛，平田 廣則 :「部分協力的な学習オートマトンのチームモデル」，平成 24 年　電気学会電子・情報・システム部門大会，pp.1531-1534 (2012)

3. 汪 岑，銭 飛，平田 廣則 :「部分協力的な学習オートマトンのチームモデルを用いた最短経路探索」，平成 25 年　電気学会電子・情報・システム部門大会，pp.1471-1474 (2013)

4. 汪 岑，銭 飛，小圷 成一，岡本 卓 :「部分協力的なアナログ型学習オートマトンのチームモデル」，平成 26 年　電気学会電子・情報・システム部門大会，pp.1507-1510 (2014)

5. 汪 岑，銭 飛，小圷 成一，岡本 卓 :「アナログ型学習オートマトンのチームモデルを用いた HighSpeed TCP の輻輳制御」，平成 27 年　電気学会電子・情報・システム部門大会，pp.1271-1276 (2015)

6. 汪 岑，小圷 成一，岡本 卓，銭 飛 :「間接情報共有を用いた協調的学習オートマトンのチームモデル TCP の輻輳制御」，平成 28 年　電気学会電子・情報・システム部門大会，pp.1213-1217 (2016)

7. 汪 岑, 銭 飛, 小圷 成一, 岡本 卓 :「マルチエージェントシステムにおける学習オートマトンを用いたエージェントの表現」，計測自動制御学会 システム・情報部門学術講演会 2015 (SSI2015), pp.34-35 (2015)

8. 塩 崇裕，汪 岑，銭 飛 :「バッテリ消費問題を考慮したセンサネットワークの為の新しいルーティング方式」，平成 22 年　電気学会電子・情報・システム部門大会，pp.1416-1417 (2010)

9. 汪 岑，丁 亮，水島 茂雄，銭 飛 :「強化学習による HighSpeed TCP の公平性問題の改善」，平成 23 年　電気学会電子・情報・システム部門大会，pp.1282-1283 (2011)

10. 水島 茂雄，丁 亮，汪 岑，銭 飛 :「アドホックネットワークにおける省電力を考慮した新しいルーティングアルゴリズム」，平成 23 年　電気学会電子・情報・システム部門大会，pp.1275-127 (2011)

付 録A

補足

以降，簡単のため，SA の識別番号を表す添字 i は省略する。定理 3.1 を証明するための方針を図.A.1 に示す。任意の十分小さな ϵ を用いて，区間 $[p_{\min}, p_{\max}]$ を二つの部分区間

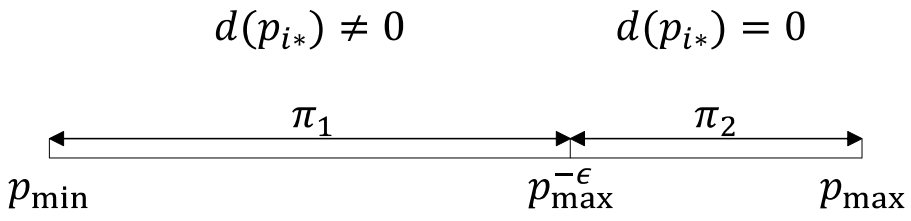

図 A.1: Roof of Convergence Proof

$$\pi_1 \triangleq [p_{\min}, p_{\max}^{-\epsilon}] \tag{A.1}$$

$$\pi_2 \triangleq [p_{\max}^{-\epsilon}, p_{\max}] \tag{A.2}$$

に分割する。ここで，$p_{\max} - p_{\max}^{-\epsilon} = \epsilon$ とする。
区間 π_2 を原点とする測度 $d(p_{j*}(t))$ を以下のように定義する。

$$d(p_{j*}(t)) \triangleq \begin{cases} 0, & p_{j*}(t) \in \pi_2 \\ p_{\max}^{-\epsilon} - p_{j*}(t), & p_{j*}(t) \in \pi_1 \end{cases} \tag{A.3}$$

また，$p_{j*}(t)$ が区間 π_1, π_2 に属する確率をそれぞれ

$$p_{\pi_1}(t) \triangleq Pr[p_{j*}(t) \in \pi_1] \tag{A.4}$$

$$p_{\pi_2}(t) \triangleq Pr[p_{j*}(t) \in \pi_2] \tag{A.5}$$

とおく。

$$\forall \epsilon, \eta > 0, \exists N : \forall t > N,$$
$$Pr[d(p_{j*}(t)) > 0] < \eta \Rightarrow Pr[p_{\max} - p_{j*}(t) > \epsilon] < \eta \tag{A.6}$$

の事実より，定理 1 を証明するためには，$E[|d(p_{j*}(t))|] \to 0$ を証明すればよいことがわかる。
$p_{j*}(t)$ が区間 π_2 に入るまでの間に，$t \to \infty$ に対して，$E[p_{j*}(t)]$ は単調増加であることを検証する必要があるので，定理 1 を証明するためには，以下の二つの補題が必要となる。

【補題 1】

T^R において，定理 1 と同じ条件下で，

$$\exists M : \forall t > M, E[\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] > 0 \tag{A.7}$$

が成立する。

【補題 2】

T^R において，定理 1 と同じ条件下で，

$$\sum_{t=1}^{\infty} p_{\pi_1}(t)\theta(t)\Big(E[d(p_{j*}(t))\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1]\Big) \tag{A.8}$$

が存在し，かつ有界である。
これらの補題および収束定理の詳しい厳密な証明を以下に述べる。
補題 1 の証明：
$d_i(t) \in [0, 1]$ より，$|\delta_{j*,j}(t)|$ は有界であることがわかる。また定理 1 の条件より $\lim_{t\to\infty} \theta(t) = 0$ であるので，$\theta(t) \max_i |\delta_{j*,j}(t)|$ も有界となる。
強化法 T^R において，$k(t)$ を時刻 t における最小ペナルティに対応する出力の番号とする。

$$p(k(t) = j*) = Pr[d_{j*}(t) \geq d_j(t), \forall j \neq j*] \tag{A.9}$$

となる。

$$\begin{aligned}
&E[\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] \\
&= \sum_{j\neq j*} E[\xi(d_{j*}(t)) - \xi(d_j(t))|k(t) = j*, p_{j*}(t) \in \pi_1]p(k(t) = j*) \\
&+ \sum_{j\neq j*} E[\xi(d_{j*}(t)) - \xi(d_j(t))|k(t) = j, p_{j*}(t) \in \pi_1]p(k(t) = j) \\
&= \sum_{j\neq j*} E[\xi(d_{j*}(t)) - \xi(d_j(t))|k(t) = j*, p_{j*}(t) \in \pi_1]p(k(t) = j*) \\
&- \sum_{j\neq j*} E[\xi(d_{j*}(t)) - \xi(d_j(t))|k(t) = j, p_{j*}(t) \in \pi_1]p(k(t) = j)
\end{aligned} \tag{A.10}$$

一方，$\xi(\cdot)$ はその定義より，区間 [0,1] の単調増加関数であり，$E[d_{j*}(t)] < E[d_j(t)]$ であるので，

$$\begin{aligned}
&E[\xi(d_{j*}(t)) - \xi(d_j(t))|k(t) = j*]p(k(t) = j*) \\
&\quad + E[\xi(d_{j*}(t)) - \xi(d_j(t))|k(t) = j]p(k(t) = j) > 0
\end{aligned} \tag{A.11}$$

が成立することがわかる。よって，

$$E[\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] > 0 \tag{A.12}$$

が成立する。
補題 2 の証明：

$$
\begin{aligned}
&E[d^2(p_{j*}(t+1))] = E[d^2(p_{j*}(t+1))|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) + E[d^2(p_{j*}(t+1))|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) \\
&= E[d^2(p_{j*}(t+1))|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) \\
&\quad + E[(p_{\max}^{-\epsilon} - p_{j*}(t) - \theta(t)\delta_{j*,j}(t))^2|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) + E[d^2(p_{j*}(t+1) \in \pi_2)|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) \\
&= E[d^2(p_{j*}(t+1)|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) + E[p_{\max}^{-\epsilon} - p_{j*}(t))^2|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) \\
&\quad + E[d^2(p_{j*}(t))|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) - 2\theta(t)E[p_{\max}^{-\epsilon} - p_{j*}(t)\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) \\
&\quad + \theta^2(t)E[\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) + E[d^2(p_{j*}(t+1) \in \pi_2)|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) \\
&= E[d^2(p_{j*}(t+1)|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) + E[d^2(p_{j*}(t))] + p_{\pi_1}(t)\{\theta^2(t)E[(\delta_{j*,j}(t))^2|p_{j*}(t) \in \pi_1] \\
&\quad - 2\theta(t)E[(p_{\max}^{-\epsilon} - p_{j*}(t))\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] + E[d^2(p_{j*}(t+1) \in \pi_2)|p_{j*}(t) \in \pi_1]\} \qquad \text{(A.13)}
\end{aligned}
$$

上式の両辺について時刻 T までの和をとると，以下の結果が得られる。

$$
\begin{aligned}
&\sum_{t=1}^{T} E[d^2(p_{j*}(t+1))] = \sum_{t=1}^{T} E[d^2(p_{j*}(t+1))|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) \\
&\quad + \sum_{t=1}^{T} E[d^2(p_{j*}(t))] + \sum_{t=1}^{T} p_{\pi_1}(t)\{\theta^2(t)E[(\delta_{j*,j}(t))^2|p_{j*} \in \pi_1] \\
&\quad - 2\theta(t)E[(p_{\max}^{-\epsilon} - p_{j*}(t))\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] + E[d^2(p_{j*}(t+1) \in \pi_2)|p_{j*}(t) \in \pi_1]\} \\
&= \sum_{t=1}^{T} E[d^2(p_{j*}(t))] + \sum_{t=1}^{T}\{a(t) + b(t) - c(t) + e(t)\} \qquad \text{(A.14)}
\end{aligned}
$$

ここで,

$$
\begin{aligned}
a(t) &= E[d^2(p_{j*}(t+1))|p_{j*}(t) \in \pi_2]p_{\pi_2}(t) && \text{(A.15)} \\
b(t) &= p_{\pi_1}(t)\theta^2(t)E[(\delta_{j*,j}(t))^2|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) && \text{(A.16)} \\
c(t) &= 2\theta(t)E[(p_{\max}^{-\epsilon} - p_{j*}(t))\delta_{j*,j}(t))|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) && \text{(A.17)} \\
e(t) &= E[d^2(p_{j*}(t+1) \in \pi_2)|p_{j*}(t) \in \pi_1]p_{\pi_1}(t) && \text{(A.18)}
\end{aligned}
$$

とする。式 (A.14) は以下のように置き換えられる。

$$
\begin{aligned}
&E[d^2(p_{j*}(T+1))] \\
&= E[d^2(p_{j*}(1))] + \sum_{t=1}^{T}\{a(t) + b(t) - c(t) + e(t)\} \qquad \text{(A.19)}
\end{aligned}
$$

$d(\cdot)$ の定義より,

$$
E[d^2(p_{j*}(t))] \geq 0 \qquad \text{(A.20)}
$$

が成り立つ。ゆえに,

$$\begin{aligned}\sum_{t=1}^{\infty} c(t) &= \sum_{t=1}^{\infty}\Big(E[d^2(p_{j*}(t))] - E[d^2(p_{j*}(t+1))]\Big) \\ &\quad + \sum_{t=1}^{\infty}\Big(a(t)+b(t)+e(t)\Big) \\ &\leq E[d^2(p_{j*}(1))] + \sum_{t=1}^{\infty}\Big(a(t)+b(t)+e(t)\Big) \end{aligned} \tag{A.21}$$

強化法 T^R の定義により $p(t+1)-p(t)$ は $\theta(t)\max(\delta_{j*,j}(t))$ を超えないことがわかる。

$$|a(t)| \leq \theta^2(t)\Big(\max(\delta_{j*,j}(t))\Big)^2 p_{\pi_2}(t) \tag{A.22}$$

$$|b(t)| \leq 2\theta^2(t)\Big(\max(\delta_{j*,j}(t))\Big)^2 p_{\pi_1}(t) \tag{A.23}$$

$$|e(t)| \leq 2\theta^2(t)\Big(\max(\delta_{j*,j}(t))\Big)^2 p_{\pi_1}(t) \tag{A.24}$$

が成立する。
従って，$\sum_{i=1}^{\infty} c(t)$ は有界である。一方，補題 1 により，ある M が存在して

$$\forall t > M, E[d^2(p_{j*}(t))\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] \geq 0 \tag{A.25}$$

になるので，$\sum_{i=1}^{\infty} c(t)$ は単調増加であることがわかる。
定理 1 の証明：

$$\xi(t) = \theta(t)E[d^2(p_{j*}(t)) \cdot \delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] \tag{A.26}$$

とする。
補題 2 により，$\sum_{i=1}^{\infty} p_{\pi_1}(t)\xi(t)$ が存在し，かつ有界であり単調増加で，$d^2(p_{j*}(t)) \geq 0, \sum_{i=1}^{\infty}\theta(t) \to 0, \xi(t) \geq 0$ であるので，定理 1 を証明するためには，$p_{\pi_1}(t)$ について考察すれば良い。
ここで，$\forall\mu \geq 0$ を用いて，$[p_{\min}, p_{\max}^{-\epsilon}]$ を $\tau_j(j = 1, 2, 3)$ に分割する。

$$\tau_1 \triangleq [p_{\min}, p_{\max}^{-\epsilon} - \mu) \tag{A.27}$$

$$\tau_2 \triangleq [p_{\max}^{-\epsilon} - \mu, p_{\max}^{-\epsilon}) \tag{A.28}$$

$$\tau_3 \triangleq [p_{\max}^{-\epsilon}, p_{\max}] \tag{A.29}$$

$\forall\mu, p_{j*}(t) \leq p_{\max}^{-\epsilon} - \mu$ について考える。明らかに，$\mu = 0$ ならば，$\lim_{t\to\infty}(p_{\max} - p_{j*}(t)) \to 0$ が成り立つので,

$$\forall\epsilon, \eta, \exists M : \forall t > M, Pr(p_{\max}^{-\epsilon} - p_{j*}(t) > \epsilon) < \eta \tag{A.30}$$

が成り立つ。
$\mu \neq 0$ とする。補題 1 より，$E[\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] > 0$ であるので,

$$\lim_{t\to\infty}\sum_{t=1}^{T} E[\delta_{j*,j}(t)|p_{j*}(t) \in \pi_1] \to \infty \tag{A.31}$$

が成立する。
これは，区間 τ_2 の中で，$\sum_{i=1}^{\infty} E[\delta_{j*,j}(t)|p_{j*} \in \pi_1] \geq 0$ が成り立つことも意味するため，

$$p_{j*}(t) \xrightarrow{p} p_{\max} \tag{A.32}$$

となる。
一方，$p_{j*}(t) \in p(t)$ は単体条件，すなわち，

$$p(t) = (p_1(t), \cdots, p_j(t), \cdots, p_s(t)),$$
$$0 \leq p_j \leq 1, j = 1, \cdots, s \tag{A.33}$$
$$\sum_{j=1}^{s} p_j(t) = 1 \tag{A.34}$$

を満たす確率分布 $p(t)$ の要素なので，

$$\sum_{t=1}^{\infty} \left(p_{j*}(t) - p_{\max}\right)^2 \leq \sum_{t=1}^{\infty} \theta(t)\delta_{j*,j}(t)^2 \leq \infty \tag{A.35}$$

が成り立つ。従って，Kolmogorov の強大数法則より [**?**] [**?**]，

$$T^R(p_{j*}(t)) = p_{\max} \tag{A.36}$$

が導かれ，定理 1 が証明された。