ネットワークコンピューティングシステムの負荷分散問題への 学習オートマトンの適用

学生員 横 山 隆 志 (広島国際学院大)
正 員 銭 飛 (広島国際学院大)
正 員 平 田 廣 則 (千葉大)

A Learning Automata Solution of Load Balancing Problems of Network Computing Systems

Takashi Yokoyama, Student Member, Fei Qian, Member (Hiroshima Kokusai Gakuin University),
Hironori Hirata, Member (Chiba University)

In network computing systems, load balancing is the problem of job distribution among multiple processors. In the area of job distribution, Round Robin method and random job division are the simplest. The demerit of these approaches is to be unable to adapt to the change of network environment. Therefore, a job scheduling mechanism is required to make reasonable job assignment for efficient use of the network. This mechanism should be able to not only adapt dynamically to the destination but also allocate jobs even without advance prediction of the changes of traffic, individual computer process speed, or the length of waiting queue.

We propose learning automaton based job scheduling method for the load distribution problem in parallel distributed computing systems. This method can dynamically respond to job addresses and make proper job assignment. We show the effectiveness of our method by computer simulation.

キーワード：ネットワーク、ジョブスケジューリング、負荷分散、学習オートマトン
Keywords: network computing, job-scheduling, load balancing, learning automata

1. はじめに

ネットワークコンピューティングシステム (Network Computing Systems) の狙いは、従来の高価なメインフレーム計算機を代表とする集中処理システムをネットワークにつながる複数の小型計算機上に分散的に表現し、これらの計算機により本来の大規模計算問題を分散的に処理するダウンサイジングである。

近年、インターネットの急速な普及に伴い、ネットワークコンピューティングシステムに関する研究が注目され、スーパーコンピューティング、グローバルコンピューティングなどの新しい研究分野が現れ、パラレルバーチャルマシン (PVM: Parallel Virtual Machine)⁽¹⁾、メッセージパッシングインターフェース (MPI: Message Passing Interface)⁽²⁾ などのソフトウェア群の開発も行われるようになってきている。PVMはマスタースレーブモデルをベースにして開発されているソフトウェアパッケージであり、このパッケージを利用すれば、ネットワーク上にある計算機を擬似的に一台の並列計算機にシミュレートするこ

とができる。PVMは事前に定義されたマスタースレーブ計算機集合とスレーブ計算機集合からなり、計算問題をマスタースレーブ計算機よりある規則に従い分割し、専門ごとに定義された各スレーブ計算機に割り当て、各スレーブ計算機での並列分散計算が行うことができる。

筆者らは、今まで PVM, MPI における並列分散計算の実現手法について試みてきた。並列分散計算法が実現できる点は確認できたが、その過程において、ネットワークの通信速度のボトルネック、処理速度の異なるスレーブ計算機群でのジョブ割り当て、つまり、負荷分散処理が必要となることが明らかとなった。^{(3)~(5)}

ネットワークコンピューティングシステムにおける負荷分散問題は、システムの全体の性能に影響を及ぼす重要な問題の一つとされている。負荷分散制御機構の主な目標は、システムにおいて各計算機の負荷バランスを調節することにより、トータルの平均処理時間を短縮することである。負荷分散制御方式としては、静的制御方式と動的制御方式がある。ネットワークのトラフィック変動など、事前に予測できない環境の下で、負荷分散を有効

図 1 集中制御型負荷分散方式のモデル
Fig.1. Centralized load balancing model of network computing systems.

的に行うために、トラヒックの変動や、各計算機の処理速度、入力待ち行列の長さの変化などに動的に適應しながら、適切なジョブ割り当てができる動的な分散型負荷分散制御機構が要求される。

そこで、本論文では、未知環境を同定できる学習オートマトン (Learning Automaton) に着目して、各計算機に置かれる学習オートマトンより構成される分散型動的な負荷分散制御アルゴリズムを提案する。

2. ジョブスケジューリング問題のモデル化

ここで、 N 台の計算機からなるネットワークコンピューティングシステムについて考える。

- μ_j : 計算機 j の処理速度 (処理率)。
- λ : 集中制御方式のサーバーでのジョブ到着率。
- λ_j : 計算機 j のジョブ到着率。
- d_{ij} : 計算機 $i, j (i \neq j)$ 間での情報転送遅延。
 $d_{ii} = 0$ とする。

ただし、 $j = 1, 2, \dots, N$ である。

PVMのようなマスター・スレーブ型、またはクライアント・サーバー型システムを考える時、負荷分散制御機構を置く場所、及びそれへの依存関係により、集中制御型 (図 1) と分散制御型 (図 2) の負荷分散方式に分類することができる。

集中制御型負荷分散方式 (図 1) では、ジョブのスケジューリング制御は全部マスター (サーバー) 側で管理し、処理するジョブを PtP (Point to Point) 方式で各スレーブ (クライアント) 側に、ネットワークを通じて転送する。この場合、マスター側ではネットワーク全体の情報を把握する必要がある。

一方、分散制御型負荷分散方式 (図 2) の場合、それぞれの計算機をジョブ入力ユニット (Job input unit)、到着待ち行列 (queue)、処理ユニット (job processing unit) の三つの部分に分けて考える。ブロードキャスト方式に

図 2 分散制御型負荷分散方式のモデル
Fig.2. Distributed load balancing model of network computing systems.

より各入力ユニット i に到着率 λ_i で到着したジョブを、現在の計算機の到着待ち行列の空き具合に応じて振り分ける。自分自身へ振り分ける場合には、通信遅延を 0 とし、ネットワークを通じて他の計算機に振り分ける場合には、通信遅延 d_{ij} が生ずるとする。計算機 i に受理されたジョブをいったん到着待ち行列 q_i で待機させ、処理ユニットの処理順番を待つ。集中処理式のスケジューリング手法においては、全てのジョブ振り分けは一台の処理機構により行うが、本論文は完全分散処理方式について考える。つまり、ネットワークコンピューティングシステムの中には、スケジューリング処理に関するスーパーバイザーは存在せず、全てのスケジューリング処理はそれぞれの計算機自身に任せ、自分の処理状況に応じて独立に行う。このような分散処理を実現するために、本論文では図 3 で示す学習オートマトンチームモデルを利用する。

各計算機のジョブ入力ユニットに確率オートマトン (SA) を一つずつ置き、待ち行列の状況を監視しながら、負荷分散の戦略 (ジョブの割当先) を決定する。具体的には、確率オートマトン SA^i は到着率 λ_i のジョブの割り当てを決定する。各計算機の待ち行列の変化や、ネットワークの負荷変化等、前述の学習オートマトンの未知環境として表現する。この場合、各オートマトン間でネットワークの「資源」を巡って競争が生じ、自分にとって都合が一番良い戦略を決めようとする。全てのオートマトンは環境の変化を同定しながら、スケジューリング処理を動的に行うので、システム全体的には一種のオートマトンのチーム型ゲームが構成される。しかも、分散処理を実現するため、各オートマトンはそれぞれ独立に自分の戦略を取るため、このようなゲームは非協力的であると考えられる。環境からの反応に応じて、最適反応原理の下で、各オートマトンは強化法を利用して自分にとっての利得の最大の戦略を見つけるまで戦略を換え続ける。最適混

図3 分散制御型負荷分散方式の学習オートマトンモデル

Fig. 3. A learning automata model for the job scheduling problems.

合戦略を見つける、つまり、非協力ゲームが均衡点に到達すると、ゲームを終了する。明らかに、このような均衡点は問題の最適解になる⁽⁸⁾。これは、各スケジューラにおいて、システムの最大スループットが得られるジョブ割り当ての方策を習得できたことを意味する。

学習オートマトンの基本定義については他の文献⁽⁶⁾⁽⁷⁾に譲るが、ここで、図3の学習オートマトンチームモデルの定義を与える。

〔定義1〕 学習オートマトンの N 人ゲームは、各確率オートマトンからの出力

$$\mathbf{y}(t) = (y^1(t), \dots, y^N(t))^T \dots\dots\dots (1)$$

$$y^i(t) = o^i(\phi^i(t))$$

$$i = 1, 2, \dots, N$$

と、それと対応する未知環境からの強化信号 $r^i(t+1)$ に関する確率分布 $Prob\{r^i(t+1)|r^i(t), \mathbf{y}(t)\}$ が存在し、かつ

$$\sum_{r^i(t+1) \in R} Prob\{r^i(t+1)|r^i(t), \mathbf{y}(t)\} = 1 \dots\dots\dots (2)$$

が満たされるとき成立する。このとき、 $\mathbf{y}(t)$ は時刻 t でのプレー、 $\phi^i(t)$ はプレーヤ (オートマトン) SA^i の時刻 t での状態、 $p^i(t)$ を時刻 t での SA^i の混合戦略、 $r^i(t+1)$ をプレーの結果と呼ぶ。 R は強化信号の集合である。

ここで、 N 人のプレーヤがそれぞれ定義1に従う $SA^1, \dots, SA^N$ からなり、各プレーヤの取りうる戦略集合は各々の確率オートマトンの出力集合からなる。各時刻で SA^i が戦略 j^i を採用することを $y^i(t) = y_{j^i}^i$ で表わし、計算機 i で発生したジョブを計算機 j に転送することを意味する。各プレーヤにとって、ゲームの中での目標とは、ある混合戦略 (mixed strategy) $\mathbf{p}(t) = (p^1(t), \dots, p^N(t))$

を見出し、自身が未知環境から受ける平均ペナルティを最小にしようとするのである。各プレーヤのゲームの中の戦略を評価するため、それぞれの効用関数を以下のように定義する。

〔定義2〕 時刻 t で確率オートマトン SA^i が出力選択確率 $p^i(t) = p_{j^i}^i$ で出力 $y^i(t) = y_{j^i}^i$ を選択し、未知環境に作用するとする。

SA^i のゲーム中の効用関数を以下のように定義する。

$$U^i(\mathbf{p}(t)) = U^i(p^1(t), \dots, p^N(t))$$

$$= \sum_{j^1, j^2, \dots, j^N} \prod_{i=1}^N p_{j^i}^i |\delta^i(\mathbf{y})| \dots\dots\dots (3)$$

ただし、

$$\mathbf{y} = (y_{j^1}^1, \dots, y_{j^N}^N)$$

$$\delta^i(\mathbf{y}) = E \left(\tau^i(\zeta^i(r^i, y_{j^i}^i)) \mid \{y_{j^k}^k\} \right)$$

$$= \int_{-\infty}^{\infty} \tau^i(\zeta^i(r^i, y_{j^i}^i)) dp(r^i \mid \{y_{j^k}^k\}) \dots\dots\dots (4)$$

$\zeta^i(\cdot)$ は、 r^i と $y_{j^i}^i$ に依存するペナルティ指標であり、 τ は以下のような符号保留関数である。

$$sign(\tau^i(\zeta^i)) = sign(\zeta^i), \quad \tau^i(0) = 0 \dots\dots\dots (5)$$

オートマトン間の相互関係は、明らかに、大局的目標が分散的に表現された各オートマトン内部の学習ルールの依存関係に左右されるが、効用関数は混合戦略 $\mathbf{p}(t)$ に依存する。

ここで、各オートマトン SA^i は外部純戦略 (foreign pure strategy) $\bar{\mathbf{y}}^i = (y^1, \dots, y^{i-1}, y^{i+1}, \dots, y^N)$ に対して、自分の純戦略 y^{i*} を決定しようとする。関数 $|\delta^i(\mathbf{y}^*)|$ の値を最小にすることができる

$$\mathbf{y}^* = y^{i*}(\bar{\mathbf{y}}^i) = y^{i*}(y^1, \dots, y^{i-1}, y^{i+1}, \dots, y^N) \dots\dots\dots (6)$$

が存在すれば、 $y^{i*}(\bar{\mathbf{y}}^i)$ を外部戦略 $\bar{\mathbf{y}}^i$ の下での SA^i の最適純戦略と呼ぶ。 $\delta^i(\mathbf{y})$ はこの意味で SA^i のペナルティ指標関数と呼ばれ、純戦略 y^i と最適純戦略 y^{i*} の間の「距離」を示す。

このゲームの静特性について、文献⁽⁹⁾より、ナッシュ均衡点が存在することが証明されている。

3. 強化法と環境表現

学習オートマトンのチームモデルを分散制御型負荷分散方式の制御機構に適用するために、動的に変動するネットワークの状況を監視し、ジョブ割り当てを行う各確率オートマトンの強化法、及び環境の表現が必要となる。

〈3・1〉 オートマトンモデル 時刻 t において、 SA^i が戦略 j^i を採用し、出力 $y_{j^i}^i$ を出すとする。各オートマトンは以下のような一般的な強化法⁽⁹⁾を利用して混合戦略の修正を行う。

強化法 T^G :

$$p_{ji}^i(t+1) = O_{S_{\epsilon^i(t+1)}} \left(p_{ji}^i(t) + \theta^i(t+1) \sum_{j \neq j^i} \Delta p_{ji}^i(t) \right) \quad \dots (7)$$

$$p_j^i(t+1) = O_{S_{\epsilon^i(t+1)}} \left(p_j^i(t) - \theta^i(t+1) \Delta p_j^i(t) \right) \quad \dots (8)$$

$$j \neq j^i, j = 1, \dots, s^i$$

ただし,

$$\Delta p_{ki}^i = \tau^i(\zeta^i(y_{ki}^i, r^i(t+1))) \quad \dots (9)$$

$O_{S_{\epsilon^i}}$ は以下のような単体射影作用素 (projection operator) である。

$$O_{S_{\epsilon^i}} = \left\{ \sum_{j=1}^{s^i} p_j^i = 1, p_j^i \geq \epsilon^i \geq 0, \right\} \quad \dots (10)$$

この強化法は、文献⁽⁹⁾の強化法の一般形から、添え字指標 i を省略して、簡単化したものであり、文献⁽⁹⁾の強化法と同型 (isomorphic) であることが容易に検証できる。

強化法 T^G は以下のような意味を持つ。

時刻 t において、 SA^i から出した出力 $y(t) = y_j^i$ に対して、未知環境からは強化信号を返してくる。これに対して、強化法ではペナルティ指標 $\zeta^i(\cdot)$ を利用して判断を行う。 $\zeta^i(\cdot) > 0$ であれば、自分のゲーム中の努力を強める。さもなくば、自分のゲーム中の努力を弱める。これにより、自分が受ける平均ペナルティをできるだけ最小にしようとする。

このモデルをジョブスケジューリング問題に適用する際、未知環境の表現形式、及び関数 $\tau^i(\cdot)$, $\zeta^i(\cdot)$ を決める必要がある。

〈3・2〉分散型スケジューリング問題の環境モデル

前述しているように、ネットワークコンピューティングシステムにおける分散制御型負荷分散方式の制御機構を学習オートマトンのチームモデルで表現するために、未知環境の定義が重要となる。

時刻 t で、オートマトンチーム側から出された混合戦略を $\mathbf{y}(t) = (y^1(t), \dots, y^N(t))$ とする。この時、ネットワークに生ずるフロー $f(t)$ は以下のように計算される。

$$f(t) = \sum_{j=1}^N \delta_{y^j, j} f_j(t) \quad \dots (11)$$

ここで、 $f_j(t)$ は時刻 t で j 番目の計算機により発生し、外部 (自分以外) に出されるフローである。また、

$$\delta_{y^j, j} = \begin{cases} 0, & \text{if } y^j(t) = y_j^j, \\ 1, & \text{otherwise.} \end{cases} \quad \dots (12)$$

この場合のネットワークでの転送遅延は M/M/1 待ち行列モデル⁽¹⁰⁾で解析すると、平均遅延 D は以下のように表現される。

$$D(f(t)) = \frac{f(t)}{C_{net} - f(t)} + \eta \quad \dots (13)$$

ここで、 C_{net} はネットワークの容量であり、 η は非待ち時間処理により生じた遅延である。

一方、この場合、各計算機での待ち行列の長さは以下のようになる。

$$q_j = \frac{\lambda_j}{\mu_j - \lambda_j}, j = 1, 2, \dots, N \quad \dots (14)$$

ここで、処理率 μ_j は計算機 j の処理速度に依存するが、到着率 λ_j はランダム変数である。

以上より、ネットワークコンピューティングシステムにおける分散制御型負荷分散方式の制御機構の制御目的は、以下のような最適化問題に置き換えることができる。

$$\min \left\{ aD(f(t)) + b \sum_{j=1}^N q_j \right\} \quad \dots (15)$$

ここで、 $0 < a, b < 1$ はそれぞれの制約条件の強度を示す実験パラメータである。

これを実現するために、未知環境のペナルティ強化信号を以下のように定める。

$$r(t) = (r^1(t), \dots, r^N(t)) \quad \dots (16)$$

$$r^j(t) = 1 - \exp \left\{ -D(\hat{f}(t)) - \hat{q}_j(t) \right\} \quad \dots (17)$$

ここで、 $\hat{f}(t)$, $\hat{q}_j(t)$ はそれぞれネットワークの平均フローと待ち行列の平均長さの時刻 t での統計値である。

このようにモデル化された未知環境において、各オートマトンのペナルティ指標関数 $\zeta^i(\cdot)$ を以下のようにすればよい。

$$\zeta^i(y_{ji}^i, r^i(t+1)) = f(\hat{d}_{ji}^i(t) - \hat{d}_{ji}^i(t)) \cdot \left(\delta_{ij}(t) p_j^i(t) + \delta_{ji}(t) \frac{(1 - p_j^i(t)) p_{ji}^i(t)}{r - 1} \right) \quad \dots (18)$$

ここで、 $f(\cdot)$ は $[-1, +1]$ 上での単調増加関数であり、 $f(0) = 0$ とする。

$$\hat{d}_{ji}^i(t) = \begin{cases} \hat{d}_{ji}^i(t-1), & \text{if } j \neq j^i, \\ W_j^i(t)/Z_j^i(t), & \text{if } j = j^i. \end{cases} \quad \dots (19)$$

$$W_j^i(t) = \begin{cases} W_j^i(t-1), & \text{if } j \neq j^i, \\ r^i(t), & \text{if } j = j^i. \end{cases} \quad \dots (20)$$

$$Z_j^i(t) = \begin{cases} Z_j^i(t-1), & \text{if } j \neq j^i, \\ Z_j^i(t-1) + 1, & \text{if } j = j^i. \end{cases} \quad \dots (21)$$

$$\delta_{ij}(t) = \begin{cases} 0, & \text{if } \hat{d}_{ji}^i < \hat{d}_{ji}^i, \\ 1, & \text{if } \hat{d}_{ji}^i \geq \hat{d}_{ji}^i. \end{cases} \quad \dots (22)$$

また、強化法 T^G にある符号保留関数を $\tau^i(x) = x$ のような線形関数とする。

〈3・3〉学習オートマトンスケジューラ 分散型ジョブスケジューリングを行うための学習オートマトンスケジューラを以下に示す。

Step 1° 各ジョブ入力ユニットに配置してある $SA^i, i = 1, 2, \dots, N$ の出力選択確率分布を一様分布で初期化する。

Step 2° SA^i で出力選択確率を用いて、出力 j^i を決定し、到着率 λ_i で到着するジョブを $y_{j^i}^i$ 番目の処理ユニットに割り当てる。実際のジョブパケットを転送する前に、送信元のタイムスタンプを付け加える。

Step 3° 各処理ユニットでは、以下の処理を行う。

- ・ 各時刻での待ち行列の平均長を計算する。
- ・ パケットを受信した場合、受信したパケット中のタイムスタンプと到着時間から転送時間を割り出し、送信元との間の平均転送時間の統計を計算し、保存しておく。
- ・ 式 (17) で強化信号を計算して送信側に応答する。

Step 4° 各入力ユニットでは、処理ユニットから受け取った強化信号を利用して、以下の処理を行う。

- ・ 式 (19) ~ (22) を利用して未知環境の統計を取る。
- ・ 式 (18) を用いて、ペナルティ指標関数を計算する。
- ・ 強化法 T^G を用いて、出力選択確率分布の修正を行う。

Step 5° Step 2° に戻る。

学習オートマトンスケジューラを用いる提案アルゴリズムを以下では LA 法と呼ぶ。LA 手法では、平均転送遅延を見積もるために、NTP(Network Time Protocol) など、各処理ユニットでの時間同期が必要となる。

4. 計算機実験

前述のモデルを検証するために、本論文では、以下のネットワーク環境について考える。

ネットワークの転送速度：10 Mbps,

η ：確率変数であり、平均値 0.2 の正規分布に従うものとする。

トータルジョブ数：25,000 個[†]。

各ノードの処理率 $\mu_j (j = 1, 2, \dots, 5)$ は以下のとおりである。

$$\mu_1 = \mu_2 = 4Mbps, \mu_3 = 2Mbps$$

$$\mu_4 = \mu_5 = 1Mbps$$

LA 法の有効性を示すために、ここで、ランダム割り当て法 (RND)、ラウンドロビン法 (RR)、リストコネクション法 (LC)^{(11) (12)} と比較する。

ランダム割り当て法は到着したジョブをある確率分布に従いそれぞれのサーバーに割り当てる方法である。ここで、その割り当て確率を 0.2 (1/ノード数) とする。ラウンドロビン法は各ノードに番号を付け、ジョブが到着したとき、各ノード番号順に順番的に割り当てていく手

法である。リストコネクション法は過去一定時間内で利用された回数のもっとも少ないノードに優先的に割り当てる方法である。

実験 1：スループット

LA 法のジョブ処理能力を考察するために、全ての処理ユニットが稼動し、全てのジョブを処理した後の平均スループットを統計してみた。

図 4 に示すように、提案した学習オートマトンスケジューラの方は RND, LC, RR のずれよりも高いスループットを持つことがわかる。そして、ネットワーク転送速度 (10Mbps) の制限で、最大平均スループットは 8Mbps 程度であり、システム全体の最大期待スループットの 12Mbps を下回っていることがわかる。

一方図 5 では、各ジョブの平均応答時間 (ターンアラウンド・タイム) の比較を示している。図 5 により、平均応答時間において LA 法は他の手法より有効であることがわかる。

実験 2：システム変動

ネットワークトポロジーの変動、またはシステムの変動について、LA 法の適応性を示すために、システム稼動中に処理ユニットのアップ/ダウンに対して実験 (図 6, 7) を行った。

図 6 では、実験開始時に 4 台の処理ユニットで処理を行い、稼動 1,000 秒後、新しい処理ユニットを 1 台増やし、さらに、2,000 秒後に 1 台の処理ユニットがダウンするというシステム環境上での平均スループット変化を示している。図 6 により、LA 法と LC はシステムの変化に追従することができるが、RND と RR 法はシステムの変化に鈍感であることがわかる。それは、RND と RR 法は環境変化への適応が遅く、追従していく前に、システムが変化したことが原因になると考えられる。

一方、LC 法と比べてみると、LA 法の方はさらに耐故障性が優れていることがわかる。

図 7 では、この場合の平均応答時間の変動を示している。

実験 3：負荷分散

LA 法の負荷分散能力を考察するために、ここでは各処理ユニットが処理したジョブ数を一定時間間隔で統計した。図 8 に示しているように、各処理ユニットの処理率に比例したジョブが処理されていることがわかる。図 8 にある一番上の 2 本の曲線が処理率 $\mu_1 = \mu_2 = 4Mbps$ の 2 つの処理ユニット、中央の 1 本は処理率 $\mu_3 = 2Mbps$ の処理ユニット、一番下の 2 本は処理率 $\mu_4 = \mu_5 = 1Mbps$ の 2 つの処理ユニットが処理されていたそれぞれの平均ジョブ数である。

[†]ここでは簡単のために、各ジョブのサイズを均一にする。

図 4 実験結果 (スループット) の比較
Fig. 4. Experiment results for throughput properties.

図 5 実験結果 (平均応答時間) の比較
Fig. 5. Experiment results for mean response times.

図 6 実験結果 (システム変動・スループット) の比較
Fig. 6. Experiment results for throughput properties in dynamic changed environments.

5. おわりに

本論文では、並列分散処理システムにおけるジョブス

図 7 実験結果 (システム変動・応答時間) の比較
Fig. 7. Experiment results for mean response times in dynamic changed environments.

図 8 実験結果 (負荷分散)
Fig. 8. Experiment results for load balancing properties.

ケジューリング問題について考え、学習オートマトンのチームモデルより構成される分散型動的なスケジューリングアルゴリズムを提案した。

並列分散処理システムのそれぞれの処理ユニットに学習オートマトンスケジューラを取り入れることにより、通信ネットワークの変動や、処理ユニットの負荷変動に応じたダイナミックジョブスケジューリングが行える。

提案したアルゴリズムは学習オートマトンのチーム的ゲームモデルを用いている。このモデルをネットワークコンピューティングなどのシステムに適用する場合、未知環境の表現が重要となる。本論文では、M/M/1 待ち行列モデルで表現される待ち行列の長さの変動で強化信号を表わし、ジョブスケジューリングするための強化学習アルゴリズムを構築した。

計算機実験では、スループット、耐故障性、負荷分散能力について考察し、他の幾つかの手法と比較した。実験により、到着ジョブのサイズが均一である環境につい

て、LA法の有効性を示した。実際のネットワーク環境においては、到着ジョブのサイズが必ずしも均一ではないので、より詳細な評価を行うために、ネットワーク環境モデルの改善や、各スケジューラの強化法の選別、及び、ネットワークシミュレータなどによる量的な解析が必要と思われる。

(平成13年10月24日受付, 同14年1月15日再受付)

文 献

- (1) A.Geist, A.Beguelin, J.Dongarra, W. Jiang, R. Manchek and V. Sunderam: PVM: Parallel Virtual Machine, *MIT Press*, (1994).
- (2) M. Snir, S. Otto, S. Huss-Ledernam, D. Walker and J. Dongarra: MPI-The Complete Reference, *The MIT Press* (1998).
- (3) F.Qian and H.Hirata: Computer Networking Representations for Parallel Distributed Computing Algorithms, *Proceedings of 1993 International Joint Conference on Neural Networks* Vol.2, pp.1577-1581,(1993).
- (4) 本行圭介・銭飛: PVMを用いた平均場近似手法の実現, 第5回計測自動制御学会中国支部学術講演会論文集, pp.214-215,(1996).
- (5) 本行圭介・銭飛, 並列分散計算法を用いたPVMへの実装における一考察, 平成9年度電気学会電子情報システム部門大会論文集, pp.563-566, (1997).
- (6) K.S. Narendra & M.A.L. Thathachar: Learning Automata - A survey, *IEEE Trans. on SMC*, Vol.4, No.4, pp.323-334 (1974).
- (7) K.S. Narendra & M.A.L. Thathachar: Learning Automata - An Introduction, *Prentice-Hall International* (1989).
- (8) Bacharach, M: Economics and the Theory of Games, Curtis Brown,(1976).
- (9) 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, 電気学会論文誌, Vol. 115-C, No. 8, pp.787-793 (1998).
- (10) L.Kleinrock: Queuing System volume 1: Theory, *New York, John Wiley & Sons, Inc.*, (1975).
- (11) C.Xu and Francis C.M. Lau: Load Balancing in Parallel Computers Theory and Practice, *Kluwer Academic Publishers*, (1997).
- (12) H.Kameda, J. Li, C. Kim and Y.Zhang: Optimal Load Balancing in Distributed Computer Systems, *Springer-Verlag*, (1997).

横山 隆 志 (学生員) 1999年広島国際学院大学(旧広島電機大学)大学院電子工学専攻修士課程修了。同年同大学大学院博士課程計測制御専攻に入学, 現在に至る。並列分散型システム, 学習オートマトンの理論とその応用の研究に従事。電子情報通信学会会員。

銭 飛 (正員) 1987年東京工業大学大学院修士課程修了。1991年千葉大学大学院自然科学研究科博士課程修了。同年広島国際学院大学(旧広島電機大学)電子工学科講師, その後, 同情報工学科助教授を経て, 現在同教授。学習オートマトンの基礎理論, 大規模システムのモデリング, 解析, 及びコンピュータネットワークなど分散システムの基礎理論の研究に従事。Ph.D., IEEE, 情報処理学会, 計測自動制御学会, 電子情報通信学会会員。

平 田 廣 則 (正員) 1976年東京工業大学大学院博士課程電気工学専攻修了。同年千葉大学工学部助手, 1981年同助教授, 1994年同教授 1997年千葉大学大学院自然科学研究科教授となり, 現在に至る。大規模システムのモデリング, 解析と設計, 特に生態システム, VLSIレイアウト, 並びに分散型システムの基礎理論に興味を持つ。2001年電気学会業績賞受賞。工学博士。IEEE, INNS, ISEM, 電子情報通信学会, 計測自動制御学会, 情報処理学会, システム制御情報学会, 人工知能学会会員。