

コミュニティに基づく大規模エージェント配置手法の評価

宮田 直輝^{†a)} 石田 亨^{†b)}

Assessment of Community-based Load Balancing for Massively Multi-Agent Systems

Naoki MIYATA^{†a)} and Toru ISHIDA^{†b)}

Abstract. 近年,多数のエージェントを扱う大規模マルチエージェントシステムが開発されている。これらのマルチエージェントシステムは,しばしば分散サーバ上で実行される。エージェントが複数のサーバに分散される場合,システムの性能を改善するためには,サーバの計算負荷とインタラクションコストを考慮しなければならない。こうした問題に対する先行研究の多くは,各エージェントに対する評価を基に,エージェントの配置を決定する。それに対して,我々が提案する Community-based Load Balancing アルゴリズム (CLB) は,エージェントのコミュニティを評価して,その配置を決定する。本研究では,エージェント配置のシミュレーションを,エージェントのインタラクションが構成するネットワークを変動させて行った。そして,エージェントのネットワーク構造が正則,または小世界ネットワークの場合に,CLB を用いた場合のインタラクションコストが,エージェント単体を評価する場合よりも小さく,ネットワークが正則な場合,約 27%改善することが示された。

Keywords. 大規模マルチエージェントシステム,分散マルチエージェントシステム,エージェント配置,モバイルエージェント

1. ま え が き

近年,多数のエージェントを扱う大規模マルチエージェントシステムが開発されている [1]。大規模マルチエージェントシステムを構築することで,より多くの利用者にサービスを行うシステムを構築することや,より複雑な系を対象とするシミュレーションを行うことができる。例えば,数百万規模の交通をマルチエージェントシステムにより再現するシミュレーション [2] や,多数の利用者を想定したサービスをマルチエージェントシステムによって提供するシステム [3] の開発が行われている。こうした大規模なマルチエージェントシステムを構築する際に発生する問題を解決するための研究が,現在進められている [4]。

大規模マルチエージェントシステムにおける問題点の 1 つは,エージェント数の増加に従ってシステムの性能が低下することである。システムが扱うエージェ

ントの数が増加すると,エージェントの計算負荷や利用者からのアクセス負荷が増加し,リアルタイムなサービスを提供することや,効率的にシミュレーションを行うことが困難になる。

大規模マルチエージェントシステムの性能が低下する問題に対して,エージェントシステムを分散化するアプローチをとることができる。システムを分散化することで,システムのリソース不足による性能の低下を解決することができる。こうしたシステムの分散配置は,分散 OS などに適用されており,その有効性が示されている [5]。

しかし,サーバを分散配置する場合,分散 OS などに適用されてきた手法では不十分である。エージェントが分散配置されることで,インタラクションがサーバ間で発生し,システムのインタラクションコストは増加する。インタラクションが頻繁に発生する場合,このようなインタラクションコストの増加を無視することはできない。従って,エージェントの計算負荷とインタラクションコストに基づいて,その配置を決定しなければならない。

本稿では,分散マルチエージェントシステムにおい

[†] 京都大学大学院情報学専攻, 〒 606-8501 京都市左京区吉田本町

a) E-mail: miyata@ai.soc.i.kyoto-u.ac.jp

b) E-mail: ishida@i.kyoto-u.ac.jp

て計算負荷を分散し、通信コストを小さくするようなエージェント配置を決定する手法を提案する。まず、本研究では上記の問題を、各サーバの計算負荷が閾値より小さく、サーバ間で発生するインタラクションコストを最小にするエージェントの配置を求める、エージェント配置問題として定式化する。本研究は、エージェント配置問題に対して、エージェントのコミュニティに対する評価を基に、その配置を決定する Community-based Load Balancing を提案する。本稿におけるコミュニティとは、互いに頻りにインタラクションを行うエージェントの集合である。コミュニティに注目してエージェントの配置を決定するため、エージェントのインタラクションが構成するネットワークの構造によって、その効果が変動することが予想される。そこで、エージェントのネットワーク構造の変動に対する、CLB の効果をシミュレーションにより検証した。

2. 関連研究

分散環境におけるプロセスまたはエージェントの配置手法は、モバイルエージェントやマルチエージェントの分野で提案されている。これらの手法は、エージェントの計算負荷と通信コストに基づいてその配置を決定することで、システムの性能を改善する手法を提案している。

[6]、[7] では、エージェントが一連のタスクを実行する際の通信コストを最小にするように、移動を行う手法が提案されている。これらの手法は主に 1 体のエージェントに注目し、タスクを実行するのに必要な通信から、そのコストが小さくなるようにエージェントの移動を決定する。しかし、エージェント間でインタラクションが発生するマルチエージェントシステムにおいては、あるエージェントの移動は、他のエージェントの通信コストに影響を与える。従って、複数のエージェントの通信コストを考慮してその配置を決定しなければならない。

[8]、[9]、[10] では、エージェントの計算負荷と通信コストに基づいて、エージェントの配置を決定する手法が提案されている。これらの手法に共通する特徴として、エージェント単体の評価に基づいて、その移動と移動先のサーバを決定していることが挙げられる。例えば、[9] が提案する Comet アルゴリズムは、エージェント単体に対する評価を基に、移動するエージェントとその移動先を決定し、過負荷のサーバの計算負

荷が閾値よりも小さくなるまで移動を反復する。この操作を、過負荷のサーバの計算負荷が閾値を下回るまで反復し、エージェントの計算負荷を分散させる。しかし、4.3 節で述べるように、エージェントの集合に対する評価を基に配置を決定することで、よりインタラクションコストを小さくすることができる。

3. エージェント配置問題

本章では、エージェントが複数のサーバに分散する際に考慮すべき点を述べ、エージェント配置問題として定式化を行う。

3.1 分散環境におけるエージェント配置

本研究で考えるマルチエージェントシステムは、ネットワーク上で接続された m 台のサーバに n 体のエージェントが分散配置されているものとする。このマルチエージェントシステム上では、エージェントは必要に応じて相互にインタラクションを行う。互いに異なるサーバに存在するエージェント間のインタラクションは、互いのエージェントが存在するサーバ間でメッセージを交換することで実現可能であるとする。各サーバの性能は同じであり、サーバ間の接続は全て等価であると仮定する。このようなシステムにおいて達成すべき事柄は、以下の 2 点である。

- 各サーバにエージェントの計算負荷を分散する
- サーバ間で発生するインタラクションコストを最小にする

過負荷になったサーバは、以下の情報が取得できるとする。

- 他のサーバの計算負荷
- 内部エージェントの計算負荷
- 内部エージェント間のインタラクションコスト
- 各内部エージェントと外部サーバ間のインタラクションコスト

まず、エージェントの計算負荷を各サーバに分散する必要がある。あるサーバに対して負荷が集中することによる性能の低下を防ぐため、各サーバにエージェントを分散配置する必要がある。

次に、サーバ間で発生するインタラクションコストを抑える必要がある。あるエージェントが異なるサーバに存在するエージェントに対してインタラクションを行う場合、インタラクションのためのメッセージをサーバ間で通信する必要がある。従って、サーバ間のインタラクションが発生することで、システムにおけるインタラクションコストが増加する。こうしたイン

タラクションコストの増加を抑えるために、頻繁にインタラクションを行うエージェント同士を同じサーバに配置し、サーバ間で発生するインタラクションの頻度を少なくする必要がある。

エージェント間のインタラクションには、頻繁にその相手と頻度が変化する場合と、変動はあるが、統計的にはその変動が小さい場合が想定される。本研究におけるインタラクションは、統計的には変動が小さい場合を想定する。変動が小さい場合、それまでのインタラクションの傾向から、インタラクションコストが小さくなるような配置を行うことができる。一方で、インタラクションの変動が大きい場合、その変動を予測して、エージェントの配置を決定しなければならない。

本研究では、各エージェントは移動可能で、かつ長い間隔でエージェントの移動を行うことを想定している。そのため、エージェント間のインタラクションコストは、エージェント移動に伴う通信コストより十分大きく、エージェント移動に伴うコストを0と仮定している。

3.2 エージェント配置問題の定式化

表1の表記に基づいて、エージェント配置問題を以下のように定式化する。

エージェント集合 A の計算負荷 $W(A)$ は、(1) に示すように、エージェント a_j の計算負荷 w_j の総和で与えられるとする。

$$W(A) = \sum_{a_j \in A} w_j \quad (1)$$

ここで、サーバ s_1 が過負荷であり、他のサーバ $s_i (i = 2, 3, \dots, m)$ は過負荷でないとする。そして、過負荷のサーバから過負荷でないサーバにエージェントを移動することで、各サーバの計算負荷を閾値以下にすることができるものとする。つまり、初期条件として(2)、(3)、(4)が満たされているものとする。

$$W(A_1) > T_h \quad (2)$$

$$W(A_i) < T_h (i = 2, 3, \dots, m) \quad (3)$$

$$\sum_{1 \leq i \leq m} W(A_i) < mT_h \quad (4)$$

このとき、過負荷のサーバ s_1 から他のサーバ $s_i (i = 2, 3, \dots, m)$ にエージェントを移動して、負荷の

表1 表記の定義
Table 1 Notations

a_i	エージェント $i (i = 1, 2, \dots, n)$
s_i	エージェントサーバ $i (i = 1, 2, \dots, m)$
w_i	エージェント a_i の計算負荷
$W(A)$	エージェント集合 A の計算負荷
$p(a_i, a_j)$	a_i と a_j の間のインタラクション頻度
A_i	移動前に s_i が保持するエージェント集合
A'_i	移動後に s_i が保持するエージェント集合
T_h	エージェントサーバの計算量閾値

分散を行うことを考える。 s_1 から s_i に移動するエージェントの集合を $M_{1,i}$ とおくと、エージェントを移動した後の各サーバが保持するエージェント集合 A'_i は、(5) 式で与えられる。

$$A'_i = \begin{cases} A_1 - \sum_{2 \leq j \leq m} M_{1,j} & (i = 1) \\ A_i \cup M_{1,i} & (i = 2, 3, \dots, m) \end{cases} \quad (5)$$

上記の式を用いて、サーバの負荷を均等にし、かつサーバ間で発生するインタラクション量を抑えるという目的は(6)、(7)式として定式化することができる。

$$\min \sum_{a_i \in A'_k \wedge a_j \in A'_l (k < l)} p(a_i, a_j) \quad (6)$$

$$s.t. W(A'_i) < T_h (i = 1, 2, \dots, m) \quad (7)$$

(6)、(7)式を満たす $M_{1,i} (i = 2, 3, \dots, m)$ が、エージェント配置問題における最適なエージェント移動集合である。

$|A_1|$ が大きい場合、全ての配置の中から(6)式を満たすエージェントの移動を決定することは困難である。なぜならば、 $|A_1|$ 体のエージェントに対して、エージェントの移動集合の選び方は $m^{|A_1|}$ 通り存在するからである。そこで4.章で、エージェント配置問題に関する近似アルゴリズムを定める。

4. エージェント配置問題の近似解法

本章では、エージェント単体の評価に基づいてエージェントの配置を決定する Sequentially Load Balancing アルゴリズム (SLB) と、本研究が提案する Community-based Load Balancing アルゴリズム (CLB)、SLB に対する CLB の特徴について述べる。

4.1 Sequentially Load Balancing

Sequentially Load Balancing アルゴリズム (SLB) は、(8)に基づいてエージェントを評価する。 $gain(a_i)$

は, a_i を移動しても計算負荷が閾値を超えないサーバのうち, a_i がもっともインタラクションを行うサーバに移動した際の, インタラクションコストの変化量である.

$$gain(a_i) = \sum_{a_j \in A_1} p(a_i, a_j) - \max_{w_i + W(A_k) < T_h} \sum_{a_j \in A_k} p(a_i, a_j) \quad (8)$$

SLB は, 評価値が最も小さいエージェントを選択し, そのエージェントが最もインタラクションを行うサーバに移動する操作を, 移動元のサーバの計算負荷が閾値を下回り, かつインタラクションコストを改善するエージェントが見つからなくなるまで繰り返す.

4.2 Community-based Load Balancing

エージェント集合 A の評価値として, $C(A)$ を定義する. $C(A)$ は, A を加えても計算負荷が閾値を超えないサーバのうち, A が最もインタラクションを行うサーバに移動した場合の, インタラクションコストの変化量である. A を移動可能なサーバが存在しない場合, その評価値は無限度大であるとする.

$$C(A) = \begin{cases} \sum_{a_i \in A_1 - A, a_j \in A} p(a_i, a_j) - \\ \max_{W(A_k \cup A) < T_h} \sum_{a_i \in A_k, a_j \in A} p(a_i, a_j) & (\exists s_i | W(A_i \cup A) \leq T_h) \\ \infty & (\forall s_i | W(A_i \cup A) > T_h) \end{cases} \quad (9)$$

CLB はエージェントの移動集合 $M_i (i = 2, 3, \dots, 4)$ の近似解を, 山登り法で求める. まず, OptimalSet という操作を定義する. OptimalSet は 1 体のエージェントを引数にとり, そのエージェントが共に移動する最適なエージェント集合と, その移動先を返す. CLB は, 各エージェントに対して OptimalSet を適用し, $C(A)$ が最も小さいエージェント集合を, OptimalSet により得られる移動先サーバに移動する. この操作を移動元サーバの計算負荷が閾値を下回り, かつ評価値が負の (移動によりインタラクションコストが減少する) エージェント集合が見つからなくなるまで繰り返す.

次に, 操作 OptimalSet について述べる. OptimalSet は引数としてエージェントを受け取り, そのエージェントと共に移動する最適なエージェント集合と, 移動先サーバを返す. あるエージェントが与えられ

たときに, 共に移動するエージェント集合は $O(2^{|A_1|})$ 通り存在し, 最適なエージェント集合を求めることは困難である. そこで本稿では, OptimalSet の近似解を求める手順として, 近似解法 ApproximateOptimalSet を提案する. まず, 移動エージェント集合 $Move$ と, 移動候補エージェント集合 $Candidate$ を定義する. 初期状態で $Move$ は空集合であり, $Candidate$ は引数で与えられたエージェントのみを含む集合である. そして以下の操作を, $W(Move)$ が全てのサーバの許容量を上回るまで反復する. まず, 候補エージェントの中から, (10) で定められる評価値 $c(a_i)$ が最も小さいエージェントを選択し, 移動エージェント集合に加える. $c(a_i)$ は, そのエージェントを $Move$ に加えることによる, $C(Move)$ の変化量である.

$$c(a_i) = C(Move + a_i) - C(Move) \quad (10)$$

次に, 移動エージェント集合に加えたエージェントと同じサーバに存在し, かつインタラクションを行い, まだ $Candidate$ や $Move$ に含まれないエージェントを候補エージェントに加える. 以上の操作の反復の中で, 最も評価値が小さいエージェント集合を選択し, $C(A)$ が得られる移動先のサーバと共に返す.

4.3 CLB の特徴

CLB は, エージェントの配置を決定するために, 移動するエージェント集合を考えている. 通常, 大規模なマルチエージェントシステムにおいては, 過負荷なサーバから他のサーバに負荷を分散するためには, 複数のエージェントを移動することが必要である. 複数のエージェントを移動する場合, 移動するエージェントの集合を考えることで, 単体ごとにエージェントを移動した場合よりも, インタラクションコストを小さくすることができる.

エージェント単体を評価する場合と, コミュニティを評価する場合で, 移動後のインタラクションコストが異なる例を図 1 と図 2 に示す. 各図には, エージェントと, その間のインタラクション, サーバが表されている. 初期状態でサーバ O はエージェント 3 体分過負荷であるとし, サーバ L は 4 体分の許容量があるとする.

図 1 は, エージェント単体の評価値に基づいて, その移動を決定する場合を示している. エージェントの移動によるインタラクションコストの変化量を評価値とし, 評価値が小さいエージェントから移動を決定する場合, 左図の場合はエージェント a, b, c の順に移

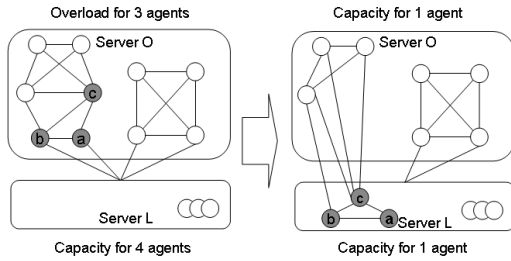


図 1 エージェント単体に対する評価に基づいてエージェントの移動を決定した場合
Fig.1 move agents based on the evaluation of agent

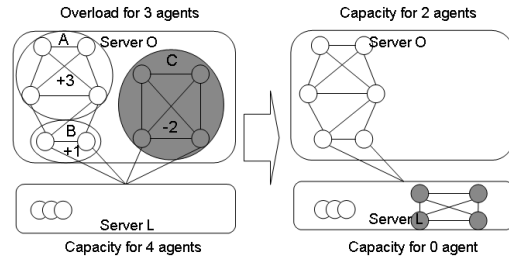


図 2 コミュニティに対する評価に基づいてエージェントの移動を決定した場合
Fig.2 move agents based on the evaluation of community

動が行われる．エージェント c を移動した後，サーバ O は過負荷でなくなり，かつサーバ O が保持するどのエージェントを移動してもインタラクションコストが増加するため，エージェントの移動を終了する．エージェントを移動した後の外部インタラクションコストは，開始前の 4 から 6 に増加する．これは， a が属するコミュニティの大きさが，移動先のサーバの許容量よりも大きく，移動によりコミュニティが分割され，インタラクションコストが増加したためである．

図 2 は，コミュニティに対して評価を行い，エージェントの移動を決定する場合を示している． A ， B ， C のコミュニティの評価値は，移動することによるインタラクションコストの変化量に基づくとき，それぞれ $+3$ ， $+1$ ， -2 である．このとき，評価値が最も小さいコミュニティ C を移動することで，サーバ O は過負荷でなくなり，インタラクションコストは 4 から 2 に減少する．従って，エージェントの評価に基づいて移動を行う場合と比較すると，インタラクションコストが 4 小さくなる．これは，コミュニティの評価に基づいて移動を決定することで，コミュニティの分割によるインタラクションコストの増加を防ぐことができたためである．

インタラクションコストが減少する一方で，エージェントの配置を決定するために必要となる計算量は大きくなる．ここでは，エージェントがインタラクションを行うエージェントの平均数を c とし，過負荷のサーバが計算負荷を閾値以下にするために移動しなければならないエージェントの数を N とする．SLB では，各エージェントの評価値を求める計算量は c に比例し， s_1 が保持する全てのエージェントの評価値を求めるための計算量は $O(c|A_i|)$ である．そして，最も評価値が小さいエージェントを選択して移動する操作を，移動元のサーバの負荷が閾値を下回るまで繰り返すため，

エージェントの移動に必要な計算量は $O(Nc|A_i|)$ である．

次に CLB の計算時間について述べる．まず，OptimalSet の操作に必要な計算量を S とする．CLB は各エージェントに対して OptimalSet を適用し，最も評価値が小さいコミュニティを選択して移動を行うため，エージェントの移動に必要な計算量は $O(NS|A_1|)$ である．次に，本稿で提案した OptimalSet の近似解法である ApproximateOptimalSet に必要な計算時間について述べる．Candidate に含まれる 1 体のエージェントの評価値を求める計算時間は $O(c)$ である．その操作を Candidate に含まれるエージェントに対して反復する．クラスタリング係数が高く，Candidate に含まれるエージェント数を $O(c)$ とすると，1 体のエージェントを移動エージェント集合に追加するために必要な時間は $O(c^2)$ である．その反復をコミュニティの移動先サーバがなくなるまで反復するため，その反復回数を $O(N)$ とすると，総計算時間は $O(Nc^2)$ となる．よって，CLB の総計算時間は $O(N^2c^2|A_i|)$ である．従って，1 台のサーバが保持するエージェント数が増加するにつれて， $|A_i|$ と N が増加し，CLB の計算量は大きく増加する．

5. エージェント配置シミュレーション

本章では，CLB の有効性を検証するために行ったシミュレーションについて述べる．

5.1 インタラクションコストの比較

本節では，CLB を用いた場合の，システムのインタラクションコストの変動を測定する．そして，エージェントを順に選択するアルゴリズムを適用した場合との比較を行う．

5.1.1 シミュレーション設定

このシミュレーションは、複数のサーバを接続した分散マルチエージェントシステムをシミュレートしたものである。このシミュレーションでは、エージェント数 $n = 1,000$ 、サーバ数 $m = 10$ としている。各エージェント間のインタラクション頻度 $p(a_i, a_j)$ は、小世界ネットワーク [11] の概念を用いて設定した。小世界ネットワークとは、高度に構造化され、かつ任意のノード間の経路長が短いネットワークである。実世界では、人間のコミュニティやインターネットなどに見られる。マルチエージェントシステムは対象とする系を反映するため、エージェント間のインタラクションは小世界ネットワークの性質を持っているといえる。本シミュレーションにおけるエージェントのインタラクションが構成するネットワークは、まず正則なネットワークを構成し、確率 p で枝を張り替えることで小世界ネットワークを構成した。本稿では、エージェントの隣接エージェント数を 6 に設定し、隣接エージェントとのインタラクションコストを 1 とした。

まず初期配置として、エージェントをクラスタごとに分割し、各サーバに配置する。具体的には、 $s_i (i = 1, 2, \dots, m)$ に存在するエージェント集合を $\{a_j | (i-1) * n/m + 1 \leq j \leq i * n/m\}$ とした。

この初期配置は、エージェント移動のために適用する各アルゴリズムに対して共通である。次に、各サーバに対して負荷の平均値をランダム値で定める。そして、各エージェントには、サーバに設定した負荷の平均値にランダム値をかけたものを設定する。エージェントの計算負荷とインタラクションコストには相関がないものとした。本シミュレーションでは、 T_h をエージェントの負荷の総和に 1.1 を掛けた値を設定した。

エージェントの計算負荷を定めた後、計算負荷が閾値 T_h を超えたサーバに対して各手法を適用し、負荷の分散を行う。そして、(11) によって与えられる移動後のシステム全体のインタラクションコストの変化を記録する。

$$\sum_{a_i \in A_k' \wedge a_j \in A_l' (k < l)} p(a_i, a_j) \quad (11)$$

以上の操作を 1 回の反復として、負荷分散によるエージェントの移動数と、システム全体のインタラクションコストの比較を行う。

エージェント移動の際に適用したアルゴリズムは、以下の 2 手法である。

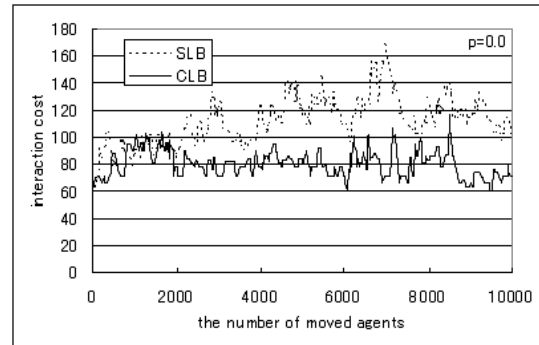


図 3 インタラクションコストの変化
Fig. 3 The change of the interaction cost

- Sequentially Load Balancing
- Community-based Load Balancing

5.1.2 シミュレーション結果

図 3 は適用手法ごとにシミュレーションを行い、その結果を記録したグラフである。このグラフの縦軸はサーバ間で発生するインタラクションコストの総和であり、(11) 式で与えられる。横軸は、過負荷のサーバから移動を行ったエージェントの数を表す。このグラフ中には、SLB と CLB をそれぞれ適用した場合のインタラクションのコストを反復回数ごとに記録している。

CLB を適用することで、SLB を適用する場合よりも、インタラクションコストが小さな配置を行うことができる。具体的には、SLB を適用した場合のインタラクションコストの最大値は 168、平均値が 110.45 であるのに対して、CLB を適用した場合のインタラクションコストの最大値は 117、平均値は 79.77 であった。

次に、小世界ネットワークを構成する際の確率 p を変動させ、手法ごとのインタラクションコストの比較を行った。本稿では、各設定に対して、6 回のシミュレーションを行った。そのシミュレーション結果を表 2 に示す。 $I(p)$ は (12) で定義され、確率 p で小世界を構成する際の、SLB の CLB に対する改善度を表す。(12) における Average interaction cost とは、エージェントの移動数が 10,000 から 20,000 までのインタラクションコストの平均値である。

$$I(p) = 1 - \frac{\text{Average interaction cost in CLB}}{\text{Average interaction cost in SLB}} \quad (12)$$

10,000 から 20,000 までの平均値を取った理由は、ある程度エージェントの移動を繰り返した後の、定常状態の平均値を測定するためである。

表 2 平均インタラクションコストと改善度の比較
Table 2 the average interaction cost and improvement

probability p	SLB	CLB	$I(p)$
0	110.946	80.880	0.27099
0.0001	111.296	84.373	0.24190
0.001	111.680	81.719	0.26827
0.01	129.515	105.013	0.18918
0.1	327.989	288.883	0.11923
0.5	1192.078	1176.377	0.01317
1.0	1418.101	1422.055	-0.0027

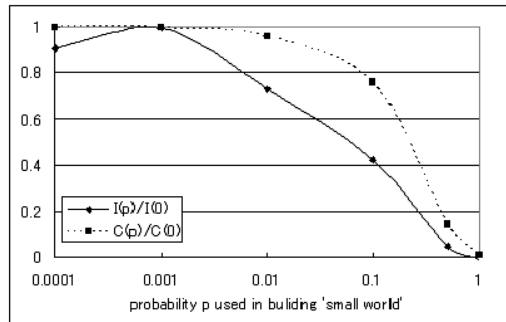


図 4 小世界ネットワーク構成時の確率 p と改善度の比較
Fig. 4 The proportion of Improvement

表 2 より，エージェントのインタラクションが構成するネットワークが正則，または小世界ネットワークの場合に，インタラクションコストはより改善する．具体的には，ネットワークが正則な場合に最も大きく，インタラクションコストは約 10%から 27%改善している．図 4 では，小世界ネットワーク構成時の確率 p を変動させた場合の改善度 $I(p)$ とネットワークのクラスタリング係数 $C(p)$ を比較した． $C(p)$ は，あるエージェントの隣接エージェント間に存在する枝の数の平均値である．つまり， $C(p)$ は，エージェントが構成するネットワークにおいて，あるエージェントの隣接エージェントが，どの程度互いに隣接しているのかを表す．図 4 より，ネットワークのクラスタリング係数が高い場合，移動元のエージェントにコミュニティが存在する確率が高くなり，CLB が SLB よりも効率的な配置を行うことができることを示している．

5.2 計算時間の測定

本節では，サーバ数が一定で，各サーバが保持するエージェント数が増加した場合の計算時間の変動と，各サーバが保持するエージェント数が一定で，サーバ数が増加した場合の CLB の計算時間の変動を計測した結果を示す．本計測は CPU1.7GHz，メモリ 512MB

表 3 サーバ数を増加させた場合の，CLB の平均計算時間
Table 3 the average computation time in changing the number of servers

サーバ数 (台)	10	20	50	100
CLB の平均計算時間 (分)	0.03	0.05	0.09	0.16

表 4 サーバが保持するエージェント数を増加させた場合の，CLB の平均計算時間

Table 4 the average computation time in changing the number of agents in a server

サーバ 1 台のエージェント数 (体)	100	200	500	1000
CLB の平均計算時間 (分)	0.03	0.3	5.8	101

の計算機で，Java を用いて実装されたプログラム上で行い，エージェントのインタラクションが構成するネットワークを正則ネットワーク，エージェントの隣接点数を 6 に設定した．

1 台のエージェントサーバに存在するエージェント数を 100 とし，サーバ数を 10, 20, 50, 100 とした場合，CLB の平均計算時間を表 3 に示す．表 4 には，サーバ数を 10 台とし，サーバ 1 台あたりに存在するエージェント数を 100, 200, 500, 1,000 と変化させた場合の CLB の平均計算時間を示す．これらの結果より，エージェント数に比例してサーバが存在する場合，CLB の計算量の増加は小さい．しかし，サーバ 1 台あたりのエージェント数が増加するに従って，CLB の計算量は大きく増加する．

6. む す び

本研究では，分散マルチエージェントシステムにおいて，システムの性能を改善するためのエージェント配置に関する議論を行った．エージェントが複数のサーバ上に分散配置される場合，システムのインタラクションコストは増加する．そのため，エージェントの計算負荷とインタラクションコストを考慮して，エージェントの配置を決定しなければならない．

本研究では，コミュニティの評価に基づいて配置を決定する Community-based Load Balancing の評価を行った．CLB の特徴は，互いに頻繁にインタラクションを行うエージェント集合であるコミュニティに注目し，コミュニティに対する評価を用いてエージェントの配置を決定することである．コミュニティを単位として移動を決定することで，コミュニティの分断によるインタラクションコストの増加を防ぐことができ，かつエージェント単体を評価するよりも，より適切な移動を行うことができる．本稿では，エー

エージェント配置に関するシミュレーションを用いることで、エージェントのインタラクションが構成するネットワークが正則、または小世界ネットワークである場合に、エージェント単体に対する評価に基づいて配置を決定する SLB アルゴリズムよりも、インタラクションコストが小さくなることを示した。CLB の計算時間は、サーバに存在するエージェント数が増加するのに従って著しく大きくなる。しかし、エージェント数に応じた数のサーバを用いて、サーバに存在するエージェント数を一定にした場合、その計算時間の増加量は大きくないことを示した。

一方で、CLB はエージェントの移動による安定性や収束性を保証することはできない。安定性や収束性は、分散アルゴリズムにおける重要な性質である。こうした性質を保証するためにエージェントの移動時にサーバ間で協調を行う必要があるが、本研究ではその方法に言及していない。サーバ間でエージェントの配置を決定する際に、こうした性質を保証するための協調方法を検討する必要がある。

CLB の計算量も解決すべき問題の一つである。本稿では、エージェント集合に対する評価に基づいて配置を決定する基本的なアルゴリズムとして CLB を提案した。しかし、CLB はサーバ 1 台あたりのエージェント数が増加するにつれて計算量が大きく増加するという問題がある。従って、CLB をより効率的に実行するための手法を提案する必要がある。例えば、OptimalSet を適用するエージェントを評価値に基づいて選択することや、OptimalSet の近似解をより効率的に求める手法を提案することが必要である。

謝辞 本研究は、日本学術振興会科学研究費基盤研究(A)(18200009, 2006-2008) 戦略的情報通信研究開発推進制度地域情報通信技術振興型研究開発(2005-2007)の助成を受けて行われた。

文 献

- [1] L. Gasser and K. Kakugawa: "Mace3j: fast flexible distributed simulation of large, large-grain multi-agent systems.", AAMAS, pp. 745-752 (2002).
- [2] M. Balmer, N. Cetin, K. Nagel and B. Raney: "Towards truly agent-based traffic and mobility simulations", AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, Washington, DC, USA, IEEE Computer Society, pp. 60-67 (2004).
- [3] H. Tai and G. Yamamoto: "An agent server for the next generation of web applications", DEXA '00: Proceedings of the 11th International Workshop on Database and Expert Systems Applications, Washington, DC, USA, IEEE Computer Society, p. 717 (2000).
- [4] T. Ishida, L. Gasser and H. Nakashima Eds.: "Massively Multi-Agent Systems I, First International Workshop, MMAS 2004, Kyoto, Japan, December 10-11, 2004, Revised Selected and Invited Papers", Vol. 3446 of Lecture Notes in Computer ScienceSpringer (2005).
- [5] I. Ahmad and A. Ghafoor: "Semi-distributed load balancing for massively parallel multicomputer systems", IEEE Transactions on Software Engineering, **17**, 10, pp. 987-1004 (1991).
- [6] T.-H. Chia and S. Kannapan: "Strategically mobile agents", MA '97: Proceedings of the First International Workshop on Mobile Agents, London, UK, Springer-Verlag, pp. 149-161 (1997).
- [7] T. Kawamura, S. Joseph, A. Ohsuya and S. Honiden: "Quantitative evaluation of pairwise interactions between agents", ASA/MA 2000: Proceedings of the Second International Symposium on Agent Systems and Applications and Fourth International Symposium on Mobile Agents, London, UK, Springer-Verlag, pp. 192-205 (2000).
- [8] 能登正人, 沼澤政信, 栗原正仁: "エージェントの移動性を考慮したエージェント間通信のトラフィック量に関する実験と評価", 電気学会論文誌 C, **124-C**, 3, pp. 904-911 (2004).
- [9] K.-P. Chow and Y.-K. Kwok: "On load balancing for distributed multiagent computing.", IEEE Transactions on Parallel and Distributed Systems, **13**, 8, pp. 787-801 (2002).
- [10] M.-W. Jang and G. Agha: "Adaptive agent allocation for massively multi-agent applications.", in T. Ishida, et al. [4], pp. 25-39.
- [11] D. J. Watts and S. H. Strogatz: "Collective dynamics of 'small-world' networks.", Nature, **393**, 6684, pp. 440-442 (1998).

Abstract Recently, large-scale multi-agent systems handling massive agents have been developed. These large-scale multi-agent systems may be executed on distributed agent servers for the benefit from distributed computing. When agents are distributed among multiple agent servers, both the computation load of agents and the interaction cost among agents have to be taken into account to improve the performance of the system effectively. Most of previous works evaluate agents one by one to select the most appropriate agent to be moved to a different server. Contrary, we propose "community-based load balancing" (CLB), which evaluates the community of agents to select a set of agents to be moved. We conducted multiagent simulations of replacing agents in different network structures of agents. The simulation results show that the interaction cost with CLB is much less than that with previous works when the network structure of agents is 'regular' or forming a 'small world'. When the network is regular, a preliminary simulation result shows that the interaction cost with CLB is about 27% smaller than that with previous works.

Key words Massively Multi-Agent Systems, Distributed Multi-Agent Systems, Agent Placement, Mobile Agent