大規模な分散環境での Agent-Based Simulation フレームワーク構築における効果的なエージェント配置手法

高橋俊博[†] 水田秀行[†]

近年，大規模な現象をシミュレーションできる Agent-Based Simulation（ABS）システムの必要性が高まっている．我々は，BlueGene 上に大規模な ABS フレームワークを構築した．エージェント間の通信量が大きいとき，ノード間の通信量が増大し ABS のパフォーマンスに影響を与える．エージェント間の通信量が大きいエージェントどうしを同一ノードに配置することで，ノード間の通信量を削減することができる．この問題に対し，非常に単純で効果的なアルゴリズムを提案し，実験によって有効性を示した．

Efficient Agent-based Simulation Framework for Multi-node Supercomputers

Toshihiro Takahashi[†] and Hideyuki Mizuta[†]

In recent years the importance of a large-scale Agent-Based Simulation (ABS) that can handle large complex systems is increasing. We developed a large-scale ABS framework on BlueGene. When the number of transmissions among the agents is large, the transmission costs seriously affect the performance of the simulation. It is possible to reduce the amount of transmission among the nodes by clustering the agents which communicate heavily with each other. This problem can be formulated as a Maximum-Flow and Minimum-Cut Problem. In this paper we present an efficient algorithm to find an approximate solution. Our algorithm is reliable, simple and efficient in ABS. We demonstrate its beneficial effects with some experiments.

1. はじめに

Agent-Based Simulation（ABS）は，多種多様な現象を理解するための強力な手段として注目されている．近年，興味の対象が大規模な現象にまで及んでおり，大規模な現象をシミュレーションできるシステムの必要性が高まっている．

すでに，ABS システムに関するいくつかの先進的な研究がなされている．たとえば，シカゴ大学とアルゴンヌ国立研究所が中心となって開発している Repast，サンタフェ研究所の Swarm，ジョージ・メイソン大学の MASON，東京工業大学の SOARS などがあげられる．MASON と SOARS は，大規模な ABS を実行することを目的の 1 つとしている．

本論では，マルチノード・スーパコンピュータである BlueGene 上に，数百万エージェントを動作させることを目的とした大規模な ABS フレームワークを構築した．その際，ノード間のネットワークの負荷を削減するアルゴリズムを考案，実装し，実験により有効性を示した．

† 日本 IBM 東京基礎研究所
Tokyo Research Laboratory, IBM Japan

2. 大規模 ABS フレームワークのアーキテクチャ

本論では，BlueGene 上に大規模な ABS フレームワークを構築した．BlueGene は，複数のノードから構成される．各ノードは，CPU，メモリ，その他のコンピュータとして必要ないくつかのリソースから構成される．各ノード間は，ギガビットイーサネットのような高速なネットワークで接続されている．各ノードには，Linux を BlueGene 向けにカスタマイズした OS がのっている．ノード内またはノード間の通信には，Message Passing Interface（MPI）を利用した．

ABS では，各エージェントの様々な判断処理や，各エージェント間の通信を繰り返し実行することによりシミュレーションが進行する．**図 1** に，本 ABS フレームワークにおけるアーキテクチャの概略を示す．本 ABS フレームワークでは，各ノードがシミュレー

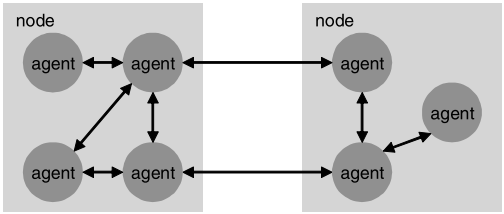

図 1 ABS フレームワークのアーキテクチャ概略
Fig. 1 The design of our ABS framework.

ションに登場する全エージェントの一部を担当する．同じノードに所属するエージェントどうしは，メモリを利用して通信を行う．違うノードに所属するエージェントどうしは，ネットワークを利用して通信を行う．ノード内通信とノード間通信との間に，通信速度の差が生じる．

本 ABS フレームワークにおける具体的な処理手順を述べる．シミュレーションが開始されると，各ノードは，初期化処理を行う．初期化処理では，全ノードが協調してエージェントを生成，分担する．また，各エージェントに一意のエージェント ID を振り，どのエージェント ID のエージェントがどのノードに属しているかを示すアドレステーブルを作成する．初期化処理が終わると，各ノードは，そのノードに所属する全エージェントの判断処理を実行し，それにともない発生したメッセージを送信バッファに格納する．次に，送信バッファにたまったメッセージを，宛先に応じて各ノードに送信する．次に，受信バッファにたまったメッセージを，宛先に応じて各エージェントに振り分ける．これらの処理を一定回数繰り返した後，得られた結果を外部記憶装置に書き出し，シミュレーションを終了する．

3. パフォーマンスボトルネック

前述のアーキテクチャでは，エージェント間の通信量が大きいとき，ノード間の通信量が増大し ABS のパフォーマンスに影響を与える．ABS では，各エージェントが他のすべてのエージェントと一様に通信する状況は稀で，エージェント間の通信量にばらつきがある状況が一般的である．この状況のもとでは，エージェント間の通信量が大きいエージェントどうしを同一ノードに配置することで，ノード間の通信量を削減することができる．また，シミュレーションの進行とともに，エージェントが動的に通信相手を変える状況を考える必要がある．このような状況にも対応できる工夫が必要である．

BlueGene のネットワークのパフォーマンスを測定

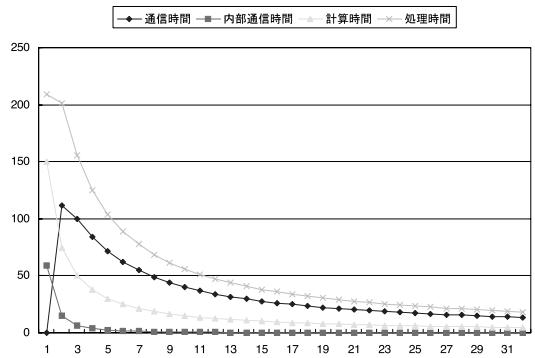

図 2 シミュレーションの処理時間
Fig. 2 Estimated computation times.

し，その結果をもとに，様々な仮定におけるシミュレーションの処理時間を大雑把に見積もった．**図 2** に，見積もった処理時間の一例を示す．縦軸は時間を表す．横軸はノード数を表す．ここでは，各エージェントは，1 ステップごとに全体のエージェントのうち 5%のエージェントと 1 回あたり 10 byte の通信を行い，3 ミリ秒の計算時間を要すると仮定した．また，シミュレーション全体の処理時間を，エージェントの計算時間，ノード内の通信時間，ノード間の通信時間の和とした．ノード数が 1 のとき，ノード間の通信時間は 0 である．ノード数が 2 のとき，ノード間の通信が発生する．結果として，この例においては，ノード数が 1 のときとノード数が 2 のときとで，シミュレーション全体の処理時間がおよそ同じになることが分かる．ノード間の通信時間がシミュレーション全体の処理時間に大きな影響を与えていることが分かる．

通信量の多いエージェントどうしを同一ノードにまとめる問題は，エージェントを頂点，エージェント間の通信量を辺として，グラフの最小カットを求める問題として定式化できる．この問題の厳密解を求めるためには，$O(n^2 \times m^{\frac{1}{2}})$ の計算量が必要である．ただし，n は頂点の数，m は辺の数を表す．

本論では，パフォーマンスの改善が目的であり，厳密解を求める必要はない．少ない計算量と通信量で良い近似解を求めるアルゴリズムが必要である．Jang ら[11] は，4 個のコンピュータがギガビットイーサネットで接続されている分散環境で ABS フレームワークを構築した．彼らは，いくつかの戦略に従って通信量の多いエージェントどうしを同一ノードにまとめ，ノード間の通信量の削減に成功している．ABS フレームワークにきわめて高い拡張性を持たせるためには，各ノードが自律的に動作し，ノード数が増加してもノード間のネゴシエーションが複雑化しないアルゴリズム

が必要である．また，エージェントが通信相手を動的に変える状況に対応できるアルゴリズムが必要である．

4. 通信コストを削減するアルゴリズム

ここでは，ノード間の通信コストを削減するアルゴリズムについて述べる．各ノードは，己に所属する各エージェントが，どのノードとどれだけの量の通信を行ったかを記録する．この情報をもとに，他のノードと交換する一定数のエージェントを選出，交換する．本アルゴリズムは，この操作を繰り返すことによってノード間の通信量を削減する．

以下に，ノード k における処理を述べる．A をすべてのエージェントの集合，C をノードの集合，$B_k \subset A$ をノード $k \in C$ が処理するエージェントの集合とする．また，$S(i,k)$ をエージェント $i \in A$ とノード k との間に発生する通信量の予測値とする．

- $\forall i \in B_k$，$\forall k' \in C$ について，$S(i,k')$ を推定する．具体的な推定法は後述する．
- $\forall i \in B_k$，$\forall k' \in C$ について
  $$\Delta R(i,k,k') = S(i,k) - S(i,k')$$
  を求める．$\Delta R(i,k,k')$ は，ノード k に所属しているエージェント i が，ノード k' に移動したときに変化する通信量を表す．
- $\forall i \in B_k$ について
  $$k'_{max}(i,k) = argmax_{k'} \Delta R(i,k,k')$$
  を求める．
- $\forall i \in B_k$ について，i を $Bpool(k, k'_{max}(i,k)) \subset B_k$ に加える．ただし，$Bpool(k,k')$ は，ノード k からノード k' に移動させるエージェント候補の集合を表す．
- $\forall k' \in C$ について
  $$n(k,k') = min(n_{max}, \#(Bpool(k,k')))$$
  を求める．ただし，$\#(Bpool(k,k'))$ は集合 $Bpool(k,k')$ の要素の個数を表す．また，n_{max} は，1 回の操作で交換するエージェントの最大数とする．n_{max} には，全体のエージェントの数に対しある程度小さな値を事前に設定しておく．
- $\forall k' \in C$ から，$n(k',k)$ を受信し，
  $$m(k,k') = min(n(k,k'), n(k',k))$$
  を求める．ここで，$m(k,k') = 0$ のとき，次に続くエージェントの交換処理は行わない．
- $\forall k' \in C$ について，上位 $m(k,k')$ 個のエージェントを $Bpool(k,k')$ から取り出し，ノード k' に送信する．
- $\forall k' \in C$ について，k' から $m(k,k')$ 個のエージェントを受信する．

以上の操作を繰り返すことで，ノード間の通信量を削減する．

いくつかの $S(i,k)$ の具体的な推定法が考えられる．1 つ目は，今までの 1 ステップあたりの平均通信量を，今後発生する通信量の予測値 $S(i,k)$ とする方法である．このとき，$S(i,k)$ は次の漸化式で計算できる．

$$S(i,k) \leftarrow \frac{S(i,k) \times (t-1) + s(i,k)}{t}$$

2 つ目は，今までの 1 ステップあたりの忘却付平均通信量を，今後発生する通信量の予測値 $S(i,k)$ とする方法である．このとき，$S(i,k)$ は次の漸化式で計算できる．

$$S(i,k) \leftarrow S(i,k) \times r + s(i,k) \times (1-r)$$

ただし，t はシミュレーションにおける現在の経過ステップ数を表す．$s(i,k)$ は，ステップ t においてエージェント i とノード k との間に発生した通信量を表す．r は，ユーザが設定する 0 から 1 までの実数値で忘却率を表す．

本アルゴリズムは，非常に単純であり，少ない計算量で高速に動作する．各ノードは，己に所属するエージェントの情報だけから他のノードへ送信するエージェントを選出する．このため，ノード間の複雑なネゴシエーションを必要としない．また，逐次的にエージェントを配置しなおすアルゴリズムであるため，エージェントの通信相手が動的に変化する場合にも対応できる．

5. アルゴリズムの振舞い

ここでは，いくつかの典型的な ABS モデルを構築し，本アルゴリズムの有効性を実験により検証する．

5.1 基本モデル

基本的なモデルとして，次の仮定をおいて ABS モデルを構築した．

- ノードの数は 2 とする．
- 1,000 個のエージェントが，$[0,1) \times [0,1)$ の二次元空間にランダムに配置されている．
- 各エージェントは，自分を中心とした 1 辺 0.1 の矩形内に入るエージェントとのみ通信する．
- エージェント間の 1 回の通信量は，1 byte とする．
- 1 回の操作で交換するエージェントの最大数は 1 とする．

図 3 に，二次元空間にランダムに配置されたエージェントの状態を示す．ここでは，左上の 0 ステップ目から右下の 280 ステップ目まで，40 ステップごとのエージェントの状態のスナップショットを示した．点はエージェントの位置を表す．点の種類は所属するノー

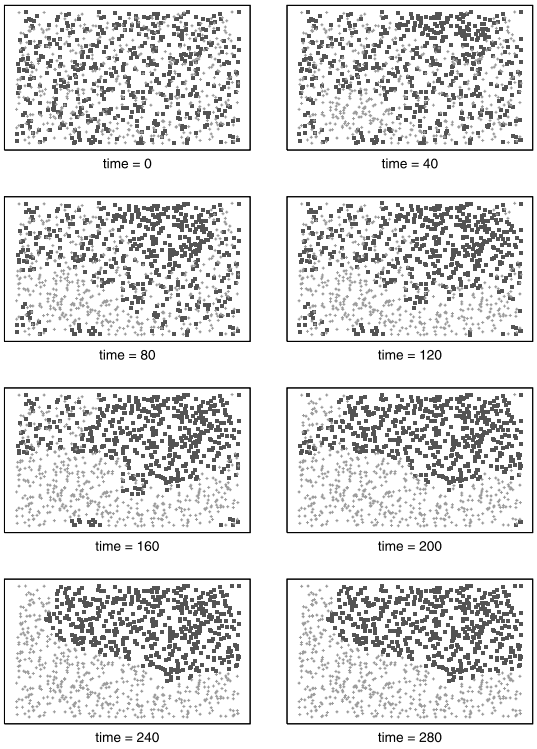

図 3 基本モデルにおけるエージェントの状態
Fig. 3 The state of agents in the basic model.

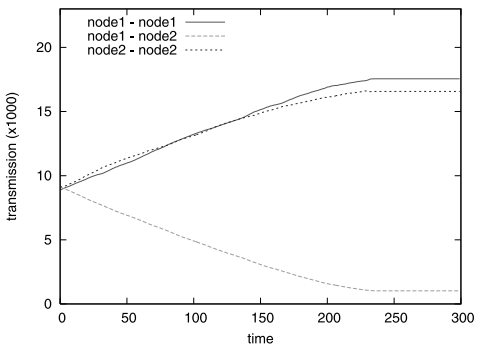

図 4 基本モデルにおける通信量の変化
Fig. 4 The amount of data transmitted in the basic model.

ドを表す．**図 4** に，エージェント間の通信により発生したノード間の通信量およびノード内の通信量の変化の様子を示す．縦軸は通信量を表す．横軸はシミュレーションのステップ数を表す．シミュレーションが 1 ステップ進むごとに，前述の操作を 1 回実行した．ノード間の通信量は，8,997 byte から 1,132 byte まで減少した．ノード内の通信量は，およそ 9,000 byte から 16,000〜17,000 byte まで増加した．2 つのエージェントのクラスタが安定するまでに，およそ 220 ステップを要した．本アルゴリズムが良好に動作していることが分かる．

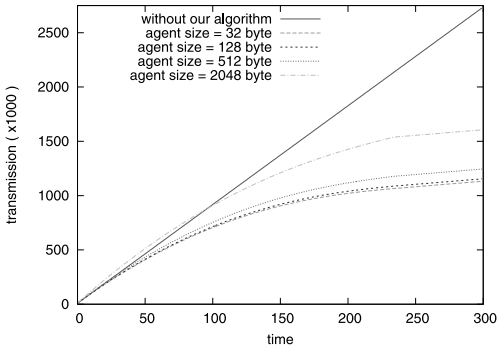

図 5 基本モデルにおける累積通信量
Fig. 5 The accumulated amount of data transmitted in the basic model.

図 4 に示した通信量には，本アルゴリズムにおけるエージェント交換の際に発生する通信量が含まれていない．**図 5** に，本アルゴリズムを無効にしたとき，本アルゴリズムを有効にし，エージェントのサイズを 32 byte，128 byte，512 byte，2,048 byte と仮定したときのシミュレーション開始からステップ t までの累積通信量を示す．この累積通信量には，エージェント間のメッセージの送受信により発生する通信量と，本アルゴリズムにおけるエージェント交換の際に発生する通信量の両方が含まれる．縦軸は累積通信量を表す．横軸はシミュレーションのステップ数を表す．ここでは，1 ステップあたりのノード間の通信量は，8,997 byte から 1,132 byte まで減少しているので，減少量は 7,865 byte である．シミュレーションの総ステップ数が十分に大きく，かつ各エージェントの通信相手が静的である場合，1 ステップで交換するエージェントのサイズの総量がこの減少量よりも小さければ，本アルゴリズムを動作させることで，必ずシミュレーション全体の累積通信量を削減することができる．

5.2 複数のエージェントを交換するモデル

2 番目のモデルは，ほぼ基本モデルと同じであるが，1 回の操作で交換するエージェントの最大数を 5 とした．

図 6 に，エージェントの状態を示す．**図 7** に，通信量の変化の様子を示す．ノード間の通信量は，8,997 byte から 1,059 byte まで減少した．2 つのエージェントのクラスタが安定するまでに，およそ 60 ステップを要した．基本モデルと比べ，必要としたステップ数が減少している．

5.3 ノードの数が 3 以上のモデル

3 番目のモデルは，ほぼ基本モデルと同じであるが，

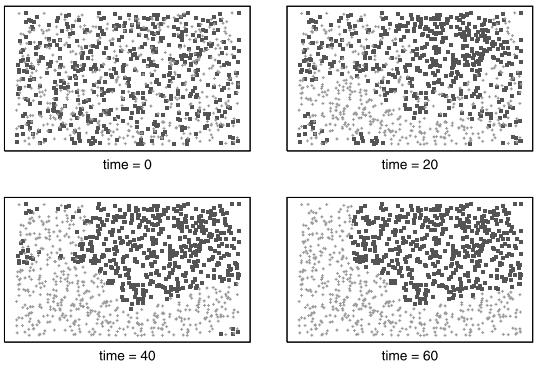

図 6 2 番目のモデルにおけるエージェントの状態
Fig. 6 The state of agents in the multi-exchange model.

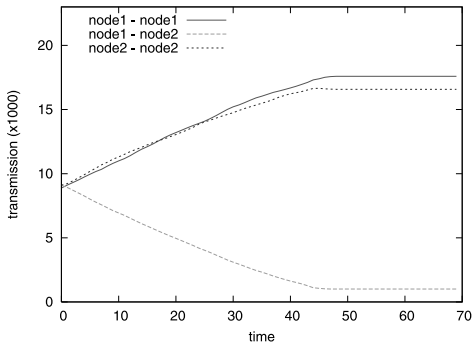

図 7 2 番目のモデルにおける通信量の変化
Fig. 7 The amount of data transmitted in the multi-exchange model.

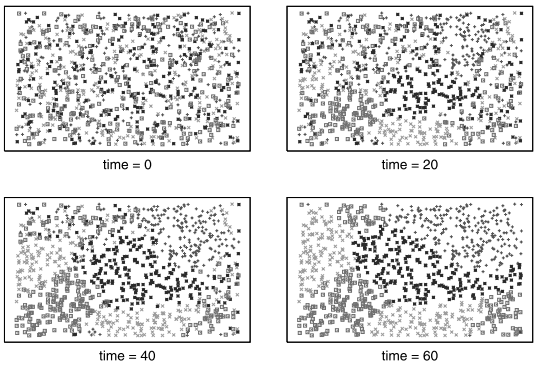

図 8 3 番目のモデルにおけるエージェントの状態
Fig. 8 The state of agents in the multi-node model.

ノードの数を 4 とした．

図 8 に，エージェントの状態を示す．**図 9** に，通信量の変化の様子を示す．ノード間の通信量は，およそ 2,100 byte から 500 byte まで減少した．ノードの数が 3 以上になっても，本アルゴリズムが良好に動作していることが分かる．

5.4 通信相手が動的に変わるモデル

4 番目のモデルは，ほぼ基本モデルと同じであるが，

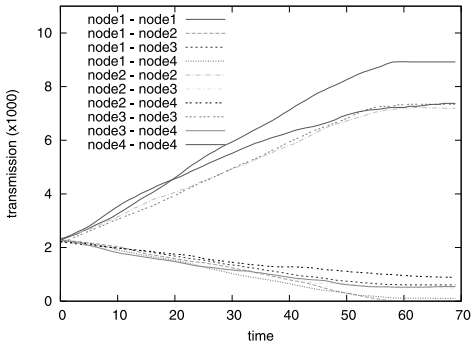

図 9 3 番目のモデルにおける通信量の変化
Fig. 9 The amount of data transmitted in the multi-node model.

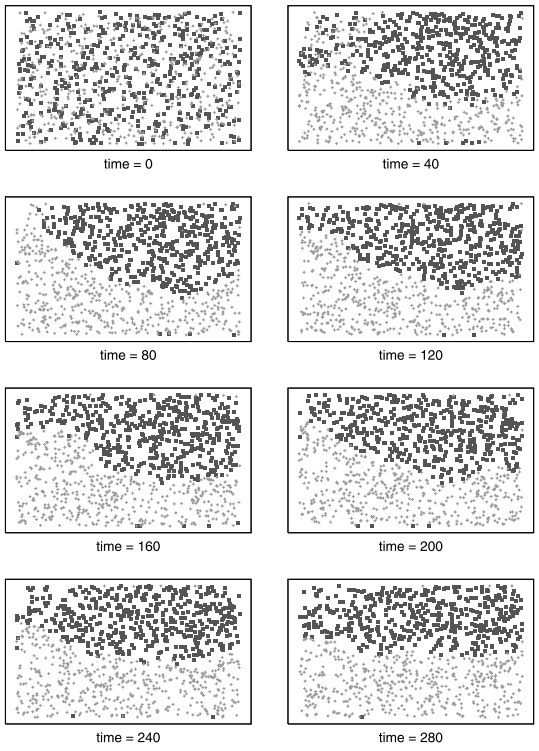

図 10 4 番目のモデルにおけるエージェントの状態
Fig. 10 The state of agents in the random walk model.

1 回の操作で交換するエージェントの最大数を 5 とした．また，シミュレーションが 1 ステップ進むごとに，各エージェントはランダムな方向に微小な距離を移動する．具体的には，各エージェントに対して −0.1 から 0.1 までの一様乱数を発生させ，得られた値をエージェントの現在位置に足しこむことでエージェントを移動させた．これにより，エージェントの通信相手がシミュレーションの進行とともに動的に変化する．

図 10 に，エージェントの様子を示す．**図 11** に，通

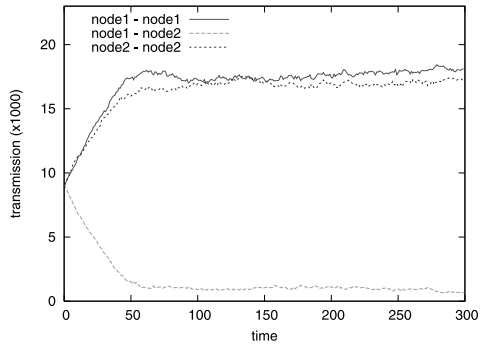

図 11 4 番目のモデルにおける通信量の変化
Fig. 11 The amount of data transmitted in the random walk model.

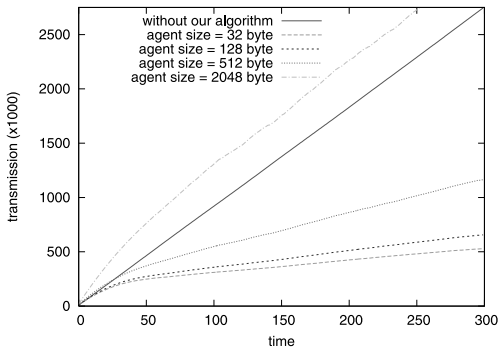

図 12 4 番目のモデルにおける累積通信量
Fig. 12 The accumulated amount of data transmitted in the random walk model.

信量の変化の様子を示す．2 つのエージェントのクラスタが安定するまでに，およそ 60 ステップを要した．その後も，ノード間の通信量が低く保たれていることが分かる．

図 12 に，累積通信量を示す．エージェントのサイズが 2,048 byte のとき，本アルゴリズムを動作させたときの累積通信量のほうが，動作させないときの累積通信量より多くなっている．ここでは，1 ステップあたりのノード間の通信量は，およそ 9,000 byte から 1,000 byte まで減少しているので，減少量はおよそ 8,000 byte である．また，シミュレーションを通して，1 ステップあたりに交換されるエージェントの平均個数は，4.41 個であった．1 ステップで交換するエージェントのサイズの総量が 8,000/4.41 ≈ 1,800 byte 程度より小さければ，本アルゴリズムを動作させることによって累積通信量を低く抑えることができると推測される．

5.5 オンラインオークションモデル

より実践的なモデルとしてオンラインオークションモデル（水田ら[10]）を構築し，本アルゴリズムの有効性を検証した．

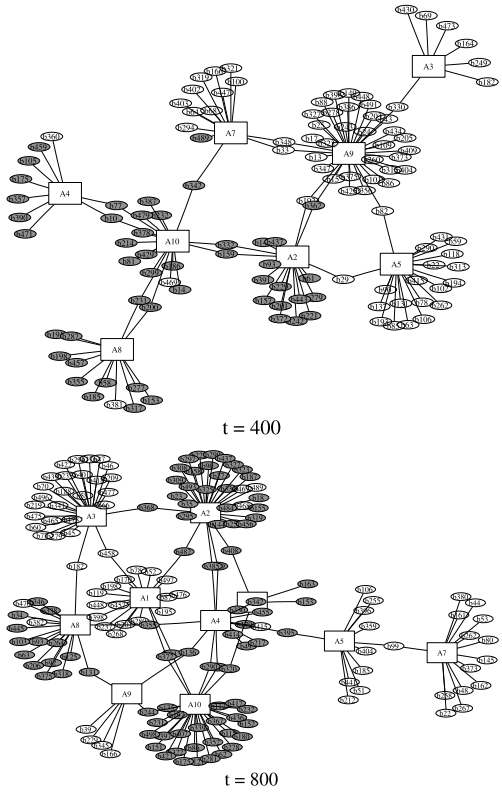

図 13 オンラインオークションモデルにおけるエージェントの状態
Fig. 13 The state of agents in the online auction model.

オンラインオークションモデルの概要を述べる．オンラインオークションモデルは，オークションと参加者の 2 つの構成要素からなる．オークションと参加者は，それぞれエージェントで表現される．オークションエージェントの個数は 100 個，参加者エージェントの個数は 2,000 個である．オークションエージェントは，開催中，休止中の 2 つの状態を持ち，休止中のオークションエージェントは，各ステップごとに 1/20 の確率で開催中に変わる．開催中になってから 100 ステップ後，再び休止中に変わる．参加者エージェントは，同時に最大 2 つまでのオークションに参加することができる．2 つのオークションに参加していない参加者エージェントは，各ステップごとに 1/300 の確率で開催中のオークションに参加する．このとき，一様な確率で参加するオークションを選択する．参加者エージェントは，Early Bidder と Sniper という戦略の異なる 2 種類のタイプに分かれる．Early Bidder は，オークション開催期間全体にわたって入札を行う．Sniper は，オークション終了直前に集中して入札を行う．オークションエージェントと，そのオークションに

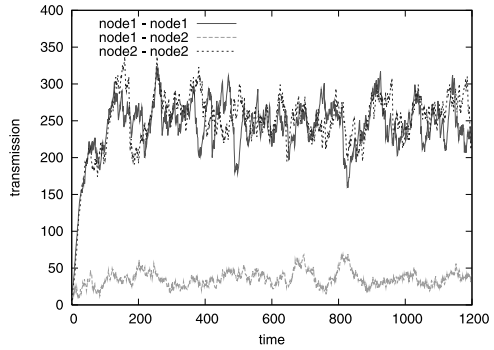

図 14 オンラインオークションモデルにおける通信量の変化（アルゴリズム動作時）
Fig. 14 The amount of data transmitted in the online auction model (with our algorithm).

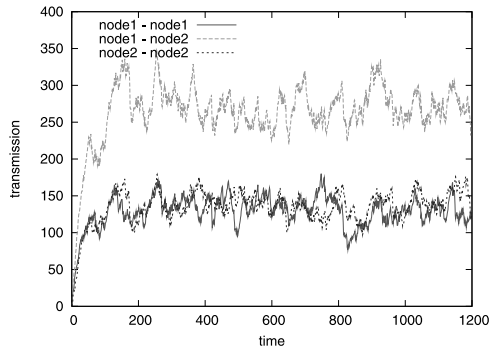

図 15 オンラインオークションモデルにおける通信量の変化（アルゴリズム非動作時）
Fig. 15 The amount of data transmitted in the online auction model (without our algorithm).

参加している参加者エージェントとの間で通信が発生する．エージェント間でやりとりされるメッセージは，送信元エージェント ID，宛先エージェント ID，メッセージ内容から構成され，サイズは 12 byte である．

図 13 に，エージェントの様子を示す．四角がオークション，白い丸がノード 1 に所属する参加者，黒い丸がノード 2 に所属する参加者を表す．同一のオークションに参加している参加者は，同じノードにまとめられていることが分かる．**図 14** に，本アルゴリズム動作時における，通信量の変化の様子を示す．**図 15** に，本アルゴリズム非動作時における，通信量の変化の様子を示す．シミュレーションの進行とともに参加者の参加しているオークションが変わっても，ノード間の通信量が低く保たれていることが分かる．

6. おわりに

本論では，BlueGene 上で大規模な ABS フレームワークを構築した．その際，ノード間の通信量を削減するアルゴリズムを提案した．

本アルゴリズムは，非常に単純であり実装が容易である．ノードの数が 3 以上のときでも容易に拡張が可能である．また，少ない計算量で高速に動作し，ノード間の複雑なネゴシエーションを必要としない．エージェントの通信相手が動的に変化する場合にも対応できる．本アルゴリズムの有効性を，実験により示した．

また，今回は BlueGene 上で本アルゴリズムを実装および検証したが，本アルゴリズムは BlueGene 以外のマルチノード・コンピュータやコンピュータ・クラスタでも有用であると思われる．特にノード間のネットワークが高速でないシステムにおいては，本手法によるパフォーマンス改善の効果は大きい．本手法は，様々な分散システムにおいて，そのシステムが扱うことのできる ABS モデルのクラスを広げることに貢献すると考えられる．

参 考 文 献

1) Banicescu, I. and Velusamy, V.: Load Balancing Highly Irregular Computations with the Adaptive Factoring, *Proc. 16th International Parallel and Distributed Processing Symposium*, p.195 (2002).
2) Bononi, L., D'Angelo, G. and Donatiello, L.: HLA-based adaptive distributed simulation of wireless mobile systems, *Proc. 17th ACM/IEEE/SCS Workshop on Parallel and Distributed Simulation* (2003).
3) Bononi, L., Bracuto, M., D'Angelo, G. and Donatiello, L.: A New Adaptive Middleware for Parallel and Distributed Simulation of Dynamically Interacting Systems, *Proc. 8th IEEE International Symposium on Distributed Simulation and Real Time Applications*, pp.178–187 (2004).
4) Chavez, A., Moukas, A. and Maes, P.: Challenger: A multiagent system for distributed resource allocation, *Proc. 1st International Conference on Autonomous Agents* (1997).
5) Deguchi, H., Tanuma, H. and Shimizu, T.: SOARS: Spot Oriented Agent Role Simulator — Design and Agent Based Dynamical System, *Proc. 3rd International Workshop on Agent-based Approaches in Economic and Social Complex Systems*, pp.49–56 (2004).
6) Goldberg, A.V. and Rao, S.: Beyond the flow decomposition barrier, *Proc. 38th IEEE Annual Symposium on Foundations of Computer Science*, pp.2–11 (1997).
7) Goldberg, A.V. and Tarjan, R.E.: A new approach to the maximum flow problem, *J. As-*

soc. Comp. Mach., Vol.35, pp.921–940 (1988).
8) Logan, B. and Theodoropoulos, G.: Dynamic interest management in the distributed simulation of agentbased systems, *Proc. 10th Conference on AI, Simulation and Planning*, Society for Computer Simulation International and ACM SIGSIM, pp.45–50 (2000).
9) Logan, B. and Theodoropoulos, G.: The distributed simulation of multiagent systems, *Proc. IEEE*, Vol.89, Issue 2, pp.174–185 (2001).
10) Mizuta, H. and Steiglitz, K.: Agent-Based Simulation of Dynamic On-Line Auctions, *Proc. 2000 Winter Simulation Conference* (2000).
11) Jang, M-W. and Agha, G.: Agent framework servives to reduce agent communication overhead in large-scale agent-based simulations, *Simulation Modeling Practice and Theory*, pp.679–694 (2006).
12) Nowe, A. and Verbeeck, K.: Distributed Reinforcement learning, Loadbased Routing a case study, *Proc. Neural, Symbolic and Reinforcement Methods for sequence Learning Workshop* (1999).
13) Parent, J., Verbeeck, K. and Lemeire, J.: Adaptive Load Balancing of Parallel Applications with Reinforcement Learning on Heterogeneous Networks, *DCABES* (2002).
14) Schaerf, A., Shoham, Y. and Tennenholtz, M.: Adaptive Load Balancing: A Study in Multi-Agent Learning, *Journal of Artificial Intelligence Research 2*, pp.475–500 (1995).
15) MASON.
http://cs.gmu.edu/ eclab/projects/mason/
16) Repast.
http://repast.sourceforge.net/
17) Swarm.
http://www.swarm.org/

(平成 18 年 8 月 18 日受付)
(平成 18 年 12 月 29 日再受付)
(平成 19 年 1 月 6 日採録)

高橋 俊博

2004 年早稲田大学大学院理工学研究科修士課程修了．同年日本アイ・ビー・エム（株）に入社．東京基礎研究所に所属．エージェントベース・シミュレーションの研究に従事．

水田 秀行（正会員）

1997 年東京大学大学院博士課程修了．博士（理学）．同年日本アイ・ビー・エム（株）に入社．東京基礎研究所に所属．エージェントベース・シミュレーションの研究に従事．