PG Labディスカッションペーパーシリーズ
第3号　2006年5月

Discussion Paper Series
No.3 May, 2006

ISSN 1880-9146

環境改善資金調達マルチエージェントシミュレーションのMPIによる並列計算

蟻川浩・森下仙一・村田忠彦

文部科学省私立大学社会連携研究推進拠点
関西大学政策グリッドコンピューティング実験センター

Policy Grid Computing Laboratory,
Kansai University
Suita, Osaka 564-8680 Japan
URL : http://www.pglab.kansai-u.ac.jp/
e-mail : pglab@jm.kansai-u.ac.jp
tel. 06-6368-1177
fax. 06-6330-3304

関西大学政策グリッドコンピューティング実験センターからのお願い

本ディスカッションペーパーシリーズを転載，引用，参照されたい場合には、
ご面倒ですが、弊センター（pglab@jm.kansai-u.ac.jp）宛に
ご連絡いただきますようお願い申しあげます。

Attention from Policy Grid Computing Laboratory, Kansai University

Please reprint, cite or quote WITH consulting Kansai University Policy Grid Computing Laboratory (pglab@jm.kansai-u.ac.jp).

環境改善資金調達マルチエージェントシミュレーションの MPI による並列計算

蟻川 浩[1]，森下 仙一[2]，村田 忠彦[1,2]

Distributed Mulitagent Simulation for Financing Model for Preservation of the Global Commons Using MPI

Hiroshi Arikawa[1], Sen-ichi Morishita[2], Tadahiko Murata[1,2]

概要

環境改善資金調達マルチエージェントシミュレーションを PC クラスタやグリッドシステム上で実装するための MPI を用いた並列計算法を提案する．近年，より現実的な MAS による社会シミュレーションが必要とされている．我々はグリッド環境による大規模な社会シミュレーションを可能にするツールの作成を行っている．そこで，本論文では，MPI を用いた MAS の並列計算手法を提案し，その効果を確認した．その結果，単体の計算機での結果と比較して類似度が高いことがわかった

Abstract

This paper describes a distributed multi agent simulation (MAS) for financing model using Message Passing Interface. Recently, the social simulation with MAS is widely noticed social scientists, a more realistic simulation is necessary for them. So, we have been developing a social simulation tool for the Grid computing environment that enables a large-scale computing. In this paper, we propose a parallel algorithm for MAS with Message Passing Interface (MPI) on the Grid. The simulation results show that there is no significant gap between the experimental data obtained by the proposed algorithm and the traditional algorithm.

キーワード：マルチエージェントシミュレーション，社会シミュレーション，並列計算，メッセージ・パッシング・インターフェイス
Keyword: Multi-Agent Simulation, Social Simulation, Parallel Computing, Message Passing Interface

[1] 関西大学政策グリッドコンピューティング実験センター，[2] 関西大学総合情報学部
[1] Policy Grid Computing Laboratory, Kansai University, [2] Faculty of Informatics, Kansai University

１．はじめに

マルチエージェントシミュレーション（MAS）は，環境と自律的な行動をするエージェントから構成され，多数のエージェントが相互に作用しながら，環境も変化するという特徴をもっている．近年，MAS の枠組みによりコンピュータ上で人工的に構成した「社会」（環境）の中で，人流や交通流，経済現象などの社会現象の再現，分析が試みられている[1-3].

MAS を単一の計算資源で処理する場合，エージェント数が増加するにつれて，計算時間が長時間になるという問題がある．エージェント数を増やした社会シミュレーションを実現するためには，グリッド・コンピューティングの環境で実行できることが要求される．

グリッド・コンピューティングとは，地理的，組織的に広範囲に分散した複数のコンピュータをネットワークで接続，共有することにより，大規模演算向けコンピュータとして仮想的に構成した手法である[4]．グリッド・コンピューティングによるシミュレーションを行う場合，どのように処理を分散させるかを考慮しなければならない．本論文では MAS を実現するモデルとして知られている Sugarscape モデルにおいて，MPI を用いた並列計算手法を提案する．具体的には，Sugarscape モデルにおける環境を分割し，異なる計算機間にエージェントが移動する場合に，必要なデータの送受信を行う．提案した並列計算手法の有効性を示すために，MPI を用いた並列計算の場合と非並列計算の場合において，結果の類似性と有効性を検証する．

２．環境改善資金調達シミュレーション

２．１　Sugarscape モデル

本論文では，MAS の例として，西崎らによって考案された環境改善資金調達モデル[3]を採用する．このシミュレーションモデルは，人工社会モデルの 1 つである Sugarscape モデル[1]を拡張したものである．Sugarscape モデルは Joshua M. Epstein と Robert Axtell が開発した人工社会モデルで，エージェント，環境，ルールという 3 つの要素から構成されている．

Sugarscape モデルには，多数のエージェントが存在し，各エージェントは内部状態と行動ルールをもっている．エージェントの内部状態は，生涯を通じて不変である視力，代謝率，性別，初期財産，寿命，交配可能年齢と，状況によって変動する現在位置，財産，年齢とがある．代謝率とは，1 期間毎にエージェントが消費する資源量のことである．初期状態では，性別は男女数がほぼ等しくなるように定められ，各エージェントには初期財産が与えられる．エージェントは資源収集を行うと同時に代謝も行い，代謝量を差し引いた分の資源が財産として蓄積されていく．

また，Sugarscape 環境は格子状の 2 次元空間で表現され，上下と左右がそれぞれ連続するトーラス空間になっている．2 次元空間上にはエージェン

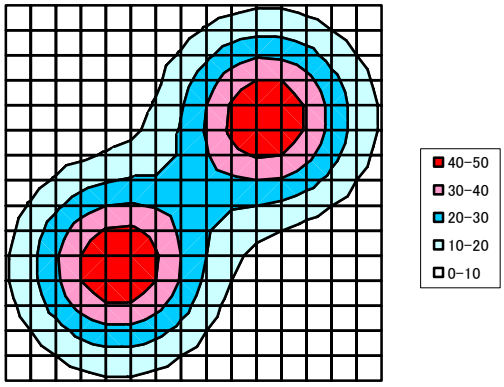

図 1　資源量の分布

トが生きるために必要な資源が配置されており，エージェントに収集されても再生する．配置される資源は図 1 のようになっている．2 次元空間上の各地点は，資源の現在量と最大容量と，エージェントの存在有無を要素としてもつ．

ここで，Sugarscape 環境上に汚染物質を発生させることにより公害を表現する．エージェントの資源収集を生産活動とみなし，資源収集によって汚染物質が生じるものとする．資源が m 単位収集される時，汚染物質が λm 生成されるとする．ある地点の時刻 t における汚染物質の合計量 p_t を式(1)に示す．

$$p_t = p_{t-1} + \lambda m \tag{1}$$

汚染物質は資源の再生率に影響を与える．蓄積された汚染物質量に応じて，各地点の資源再生率が変化する．したがって，汚染物質が増えるにつれ，エージェントの生存に必要な資源が再生されなくなる．

現実社会では環境汚染が著しい場合，汚染を軽減するような措置をとる．これを考慮し，汚染物質量に応じて表 1 のような公害発生率 λ を設定する．ここで，汚染物質量がある一定量を超えると公害発生率が半分になっているのは，ある一定量以上の公害が発生した場合，なんらかの特別の処置がとられるものとしているためである[3]．

エージェントと環境は，それぞれルールをもっており，それらのルールに従ってエージェントは自律的に行動する．

エージェントの行動ルールの例を示す．各エージェントは 1 期間毎に 2 次元空間上の直交する 4 方向を視力に基づいて見渡し，最も資源が多く，他

表１　公害発生率

Accumulated pollutants p_t	Generation rate λ
Under 50 units	0.001
Over 50 units	0.0005

のエージェントがいない地点に移動する．そして，その地点にある資源を全て収集し，代謝しながら財産の増減を行う．エージェントの財産が 0 以下となった場合は，寿命に達していなくても死亡する．また，エージェントの移動後，そのエージェントが交配可能年齢であること，その時点での財産が初期財産より多いことを交配の条件とし，その条件を満たしているエージェントは，交配条件を満たした近隣のエージェントと交配を行う．交配時には両親の初期財産の平均を子供に与える．

環境におけるルールとして，2 次元空間上の各地点の資源は，最大容量に達していない場合，単位期間あたり一定量ずつ再生される，と設定する．

３．並列計算への適用

３．１　Sugarscape 環境の分割

MAS では，エージェントの数が増えるとシミュレーション時間も増える．グリッド･コンピューティングの環境でシミュレーションを行う目的はシミュレーション時間の短縮であるため，エージェントの意思決定にかかる時間を短縮することが要求される．本論文では，文献[5]で提案された手法を採用する．具体的には，Sugarscape 環境を複数の区域に分割し，各区域におけるエージェント意思決定の処理を各コンピュータに行わせるものである．

３．２　隣接区域との通信

Sugarscape 環境を複数の区域に分割した場合，区域間の移動を定義しなければ，エージェントは初期区域だけでしか行動できなくなる．Sugarscape 環境の分割数が増加するに従って，エージェントの移動範囲が狭くなるため，環境を分割しない場合と比べエージェントの分布が大きく異なる．よって，従来の Sugarscape モデルとの結果と大幅にかけ離れてしまうことになる．そこで，各区域でエージェントが移動する前に隣接区域と通信を行い，隣接区域の環境情報を付加する．その後，各区域でエージェントの移動を行う．区域を出たエージェントの情報は蓄積しておき，全てのエージェントの行動後，蓄積されたエージェント情報を該当する隣接区域に送信し，移動前の区

域のエージェント情報を削除する．これにより，各エージェントの移動毎に通信を行う場合に比べ通信回数が少なくなる．

図 2 に上記の隣接区域のエージェント情報交換の一例を示す. この図では, 1 つの Sugarscape 環境を 9 区域に分割している．本来 Sugarscape 環境は 1 つのトーラス状の 2 次元空間であるので, たとえば区域 7 に注目した場合, 斜線部分の環境情報を受信する．全エージェントの移動後は，斜線部分に入ったエージェントの情報を該当区域に送信する．そして，隣接区域に出たエージェントの情報を区域 7 から削除する．

３．３　エージェント衝突時の処理

隣接区域との通信により，区域を移動してきたエージェント情報を受信し，その情報に基づいて各エージェントを予定移動地点に配置する．この場合エージェント同士が衝突するという問題が発生する．通常，同一区域内（同一計算機内）でエージェントが移動する時には必ず他のエージェントがいない地点を選択するが，Sugarscape の環境を分割することによって，区域を移動してきたエージェントを配置する前に，移動先の区域のエージェントが移動してくることがある．この場合，予定移動地点にすでに他のエージェントが存在するため，衝突が発生する．本論文では先にその区域にいたエージェントを優先するルールを採用する．区域を移動してきたエージェントは予定移動地点の近隣をランダムに選択して移動させる．ここで，元の区域に戻すことも考えられるが，エージェントの移動のために，再度通信を行う必要があるため，区域を移動してきたエージェントは元の区域に戻るという選択はしない．

４．MPI を用いた並列処理の実現

本論文では MAS の並列処理を可能にするための方法について説明する．具体的には，MPI ライブラリを用いた場合の MAS の動作フロー，MPI ライブラリを利用して実装する際に工夫した点について述べる．

４．１　MPI ライブラリ

MPI ライブラリは分散メモリ型並列処理環境で並列プログラムを実装するために必要不可欠となる, プロセス間同士でメッセージを交換するための通信ライブラリの総称である. Message Passing Intarface Forum [6]にて通信ライブラリの規格が策定され, 規格に基づいて様々なライブラリが実装されている．MPI ライブラリとして有名なものとしては，MPICH [7], LAM [8], GridMPI [9]がある．

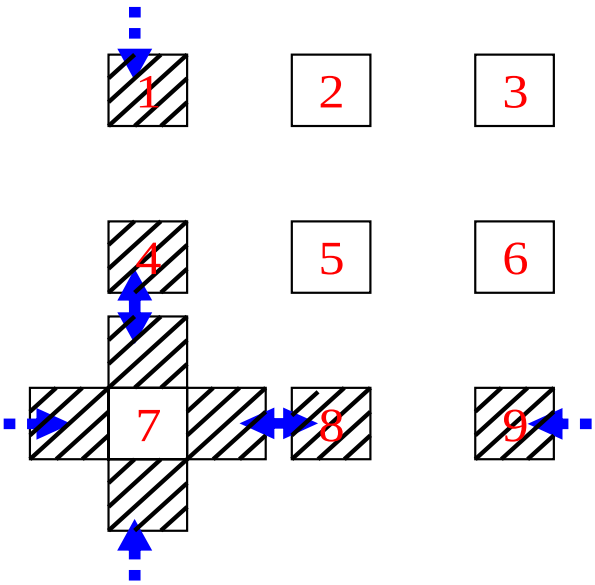

図 2　隣接区域との通信例

MPI は，主に 1 対のプロセス間の通信を行う「1 対 1 通信」とプロセスのグループ間で通信を行う「集団通信」があり，これらの命令を適切に用いて並列プログラムを実装する．

４．２　MPI を用いた MAS の動作の流れ

MPI を用いた MAS では，複数台のプロセッサを用いてシミュレーションが行われる．本論文では，Sugarscape 環境を分割する方式を採用するため，プロセッサとプロセスの実行との関係は図 3 のようになる．図 3 において，縦方向はプロセスの実行位置，横方向はプロセッサの数を意味する．図中 P1 から P9 はプロセッサを意味する．以下では，各番号のそれぞれの処理について説明する．

まず，処理(1)において Sugarscape 環境を分割した情報(各区域の情報)を自分を含む他のプロセッサに分配する．同時にエージェントの初期情報も送信する．その後，処理(2)において各プロセッサが処理をするのに必要となる隣接区域の情報を交換する．

Sugarscape 環境情報とエージェントの初期情報がすべてのプロセッサに行き渡ると，プロセッサ毎に処理(3)のエージェント意思決定処理が行われる．

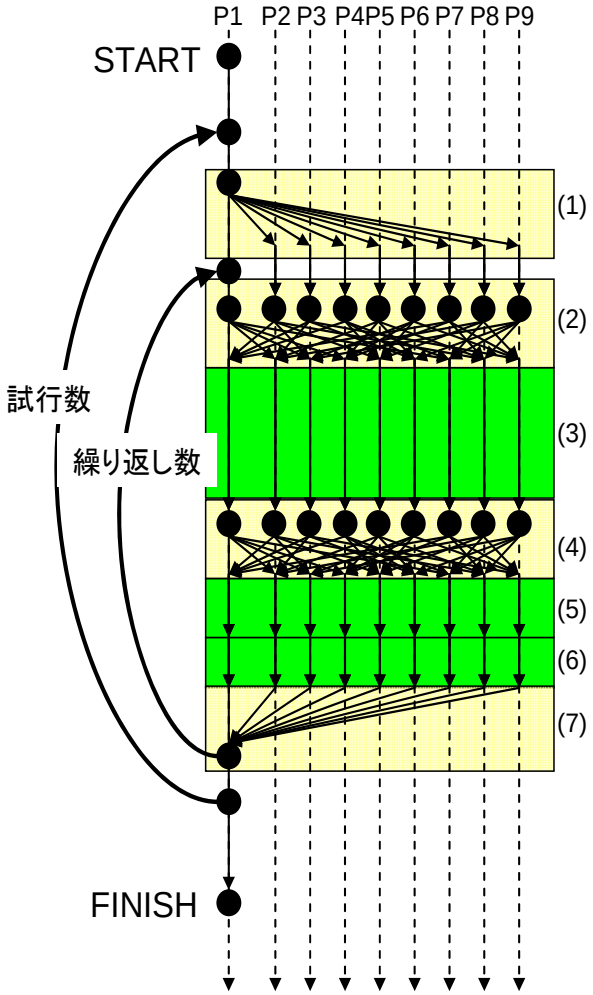

図 3　MPI を用いた並列計算の動作フロー

処理(4)では，各プロセッサがもつ Sugarscape の領域外に移動するエージェントの情報をエージェント毎に該当する領域を担当するプロセッサに通信する．その後，処理(5)では 3.2 で示したエージェントの衝突判定を行い，処理(6)で各領域における資源回復処理を行う．処理(7)で各領域のデータを収集し，各ステップごとのデータ集計を行う．

処理(2)~(7)を所定の回数まで繰り返し実行して，1 試行の計算が完了する．そして，統計的に十分な数のシミュレーションを行うことで，シミュレーション結果を得る．

ここで，プロセス間で通信が必要となる処理は処理(1)，処理(2)，処理(4)，処理(7)の部分である．これらの処理に MPI ライブラリを用いている．以下で具体的な説明をする．

処理(1)では各プロセス毎に異なるデータを送信する必要があるため，送信側であるプロセッサ 1 では MPI_Send()を，受信側であるプロセッサ 2 からプロセッサ 9 は MPI_Recv()を用いる．

処理(2)と処理(4)では例えば，プロセッサ 1 はプロセッサ 2,3,4,7 にというように，送信側と受信側が対応しているため，MPI_Sendrecv()を用いる．

処理(7)ではすべてのプロセッサでの処理結果を集計する作業である．MPI では通信と同時に演算を行う命令として MPI_Reduce()がある．処理(7)では，MPI_Reduce()を用いた．

MPI では同期通信と非同期通信の命令がある．同期通信はすべてのプロセスである処理が完了するまで通信を始めないが，非同期通信は通信に関係するプロセスが通信可能であれば通信するという違いがある．本論文で対象とした MAS は，エージェントの移動に伴う衝突判定においてデータの同期を考慮する必要があるため，同期通信の命令を利用してプログラムを実装した．MPI_Send(),MPI_Recv()，MPI_Sendrecv()，MPI_Reduce()はすべて同期通信の命令である．

４．３　MTU による通信の制限とその対処

一口に並列計算環境といっても，複数のプロセッサが搭載されている並列計算機から，PC クラスタやグリッド・コンピューティング環境のようにコモディティなコンピュータでインターネット技術を利用して構成した並列計算環境まで幅広い．本論文では，PC クラスタやグリッド・コンピューティング環境を並列計算環境と想定して話を進める．

PC クラスタやグリッド・コンピューティング環境を並列計算環境と想定した場合，これらの環境でしばし用いられる MPI ライブラリはインターネット技術を用いてデータ通信が行われる．したがって通信プロトコルは TCP/IP をベースとしたものである．TCP/IP では Max Transfer Unit (MTU)と呼ばれる，1 パケットあたり送信可能なデータの最大値を示すものがある．MPI による MAS プログラムの実装の過程において，MTU を超えて一度に

大量の通信を行おうとすると送信エラーが起こるといった症状が発生した．

そこで，我々は MTU 値を超えないようにデータ送信するように通信部分を工夫した．具体的には，送信したいデータを複数回に分けて，通信命令を複数回呼び出して通信する方法を採用した．

本来，このような問題は MPI ライブラリ側で対処するべき問題であり，我々プログラムを実装する側が対処するべき内容ではない．しかし，MPI ライブラリは環境等に応じて様々な実装例があるため, MPI ライブラリに依存しないデータの通信方法として有効であると考える．

５．比較実験

本論文で提案したSugarscapeモデルの並列処理手法の有効性を確認するため，単体での計算の場合と提案手法の場合でシミュレーションを行い，実験結果の類似性を検討する．

５．１　シミュレーション時のパラメータ

Sugarscape 環境情報については，単体での計算プログラムでは 150×150 の Sugarscape モデルを採用する．また，提案する並列計算プログラムでは図 2 に示した 150×150 の区画を 9 区域に分割した Sugarscape モデル(50×50×9 台)の Sugarscape モデルを採用する．

単体での計算プログラム，並列計算プログラムともに，初期エージェント数は 3600 とする．また，資源の現在量が最大容量に満たない場合は 1 期間に最大容量×再生率分まで回復するというルールを用いる．

５．２　並列計算環境

1 試行あたりのシミュレーション期間は 2000 期間とする．シミュレーションの試行回数は 50 試行とし，シミュレーション結果はその平均とする．エージェントに関係するパラメータ設定は以下のとおりである. エージェントの内部状態としては，視力は 1 から 6 まで，代謝率は 15 から 35 まで，寿命は 60 から 100 まで，交配開始年齢は 12 から 15 まで，交配終了年齢は男なら 40 から 50 まで，女なら 30 から 40 までとする．初期エージェントの初期財産は代謝率の 1 倍から 3 倍とする．

表 2　計算機環境

CPU	Pentium4　3EGHz
メモリ	2GB
OS	Linux 2.4.7.
MPI ライブラリ	LAM 7.1.1.
ネットワーク	1000base-T

表 3　人口推移の平均 2 乗誤差

	平均 2 乗誤差	平方根
エージェントの人口の推移	11507.2	107.3
汚染物質量の推移	1636700.3	1279.3
総資源量の推移	24530.8	156.6

５．３　実験結果

５．３．１　シミュレーション結果の比較

図 4～図 6 に，並列処理を行わない場合（Not Distributed: ND）と提案手法による場合（Distributed: D）のエージェント人口，汚染物質量，総資源量の推移を示す.

また，図 4～図 6 のグラフにおいて，それぞれ 200 期間毎の 2 乗誤差を計算し，50 試行に対して得られた平均 2 乗誤差とその平方根を表 3 に示す.

これらの図から並列処理を行わない場合と提案手法による場合とを比べると，ほとんど差がないことがわかる．また，表 3 の 2 乗誤差も低い値となっている．これらの結果から，分割を行った場合と行わなかった場合の推移の類似性が高く，並列化によるシミュレーション結果への影響は少ないことがわかる.

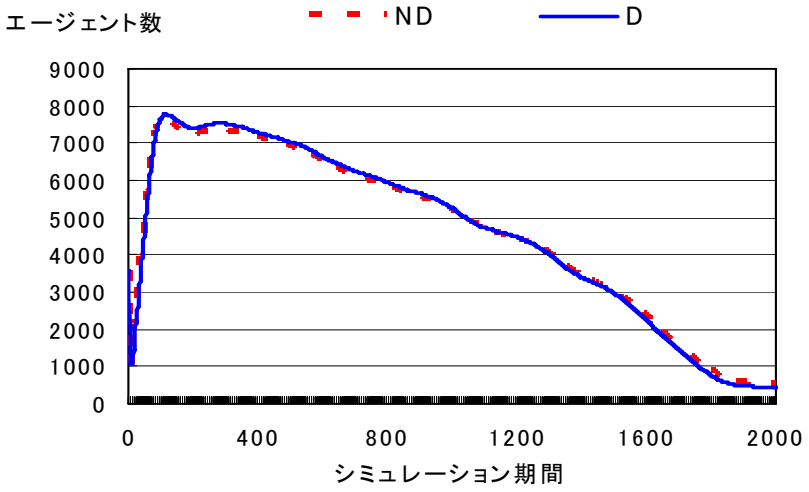

図 4　エージェント人口の推移

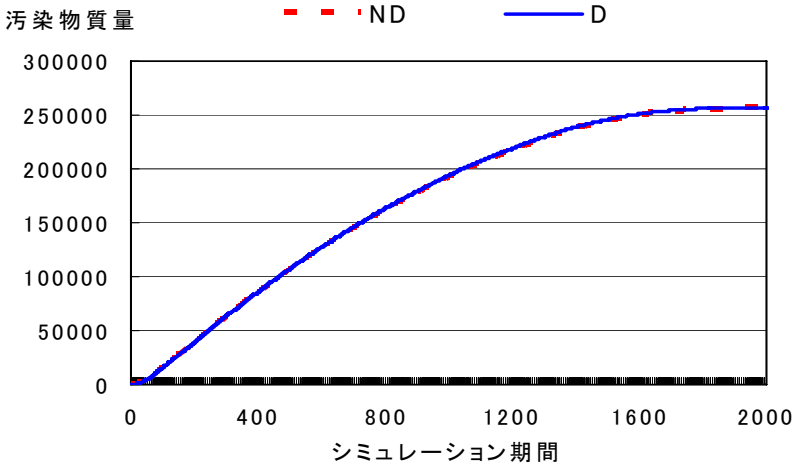

図 5　汚染物質量の推移

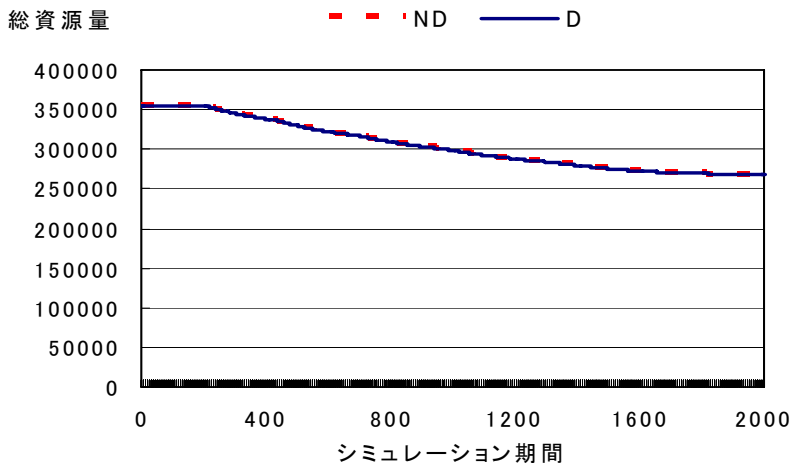

図 6　総資源量の推移

５．３．２　シミュレーション実行時間

図 7 は並列処理を行った場合の実計算時間，通信時間，通信待ち時間の内訳を示す．これらは MPI_Wtime()を用いて計測した．MPI_Wtime()はマイクロ秒の精度で計測できるライブラリである．図 7 から，通信時間が全体の実行時間の 86%占めていることがわかる．エージェントの意思決定の計算よりもデータの同期等による通信にかかる時間のほうが大きいことが判明した．

本論文における実験では，単体でのシミュレーションとの比較のためエージェント数を単体と同じ値で実験したため，並列計算の有用性を示すまでに至っていない．並列計算の有用性を示すためには，次の問題点と解決方法が考えられる．

まず，提案手法では複数の計算機を用いているため，より大きな環境情報とエージェント数による計算が可能であるが，本実験では単体でのシミュレーションと同じ値を採用している．エージェント数を増やした場合のシミュレーションを行い，どの程度の規模の計算であれば提案手法が有用であるか調査する．

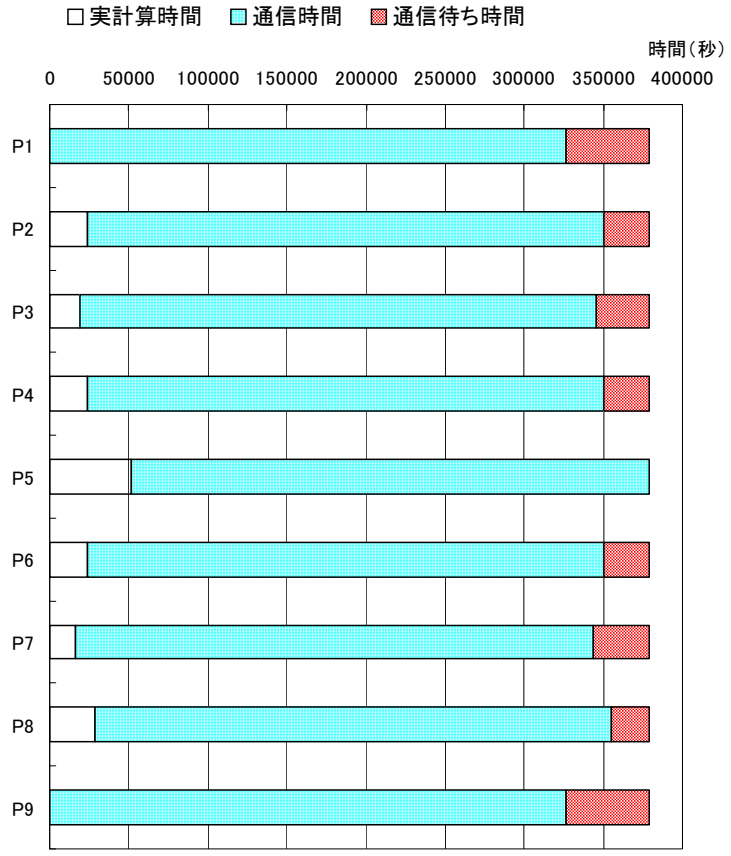

図 7　各プロセッサにおける実行時間の内訳

次に，本論文で提案した手法は隣接区域の情報を丸ごと受信する手法を採用している．エージェントの知覚範囲を考慮すると，隣接区域の情報を丸ごと受信する必要はない．例えば，図 8 のように，必要な分だけ通信する手法を採用することで，通信量の削減が可能になる．その結果，通信時間を削減できると考えられる．

最後に，本実験では Sugarscape 環境を均一に分散しただけなので，エージェントの処理が均一に分散できていないことが明らかになった．図 7 より，プロセッサ 5 に負荷が集中している．一方で，プロセッサ 1 と 9 ではほと

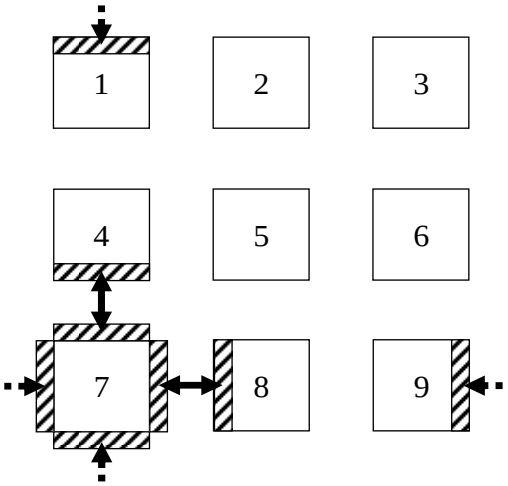

図 8　隣接区域との通信例

んど負荷がない. 並列処理では各プロセッサに均等な計算負荷を与えることが望ましいので, 今後はエージェントの数に応じた負荷分散手法を検討したいと考えている.

６．おわりに

本論文では, Sugarscape モデルの MPI を用いた並列計算手法について提案した．複数の環境に分割し MPI を用いてデータの交換を行う場合と，単体の計算資源を用いる場合で実験を行った．実験結果をもとに，並列計算と単体の計算との類似性を示した後, 並列計算による計算結果の有効性を示した.

今後は，通信量の削減による通信時間の縮小や，エージェント数を増加させることによるエージェントの振る舞いの変化と並列効果の確認，MTU の違いによる処理時間の短縮など，大規模 MAS の並列処理について，総合的な見地から研究を進めていく．また，他の MAS の並列処理の適用についても検討していく.

参考文献

[1] Joshua M. Epstein, Robert Axtell (服部 正太, 木村 香代子訳), 人工社会 －複雑系とマルチエージェント・シミュレーション－, 共立出版(1999).

[2] 並列分散処理研究室（筑波大学大学院システム情報工学研究科）, http://www.padc.cs.tsukuba.ac.jp/index.html (2004).

[3] 西崎 一郎, 上田 良文, 佐々木 智彦, “慈善くじによるグローバル・コモンズの保全のための資金調達と人工社会モデルを用いたシミュレーション分析”, システム制御情報学会論文誌, Vol.17, No.7, pp. 288-296 (2004).
[4] Ian Foster, Carl Kesselman, “The Grid: Blueprint for a New Computing Infrastructure” Morgan Kaufmann Publishers, Inc., (1999).
[5] 村田 忠彦, 松本 優美, 蟻川 浩, 山口 正聡, 小橋 博道, 門岡 良昌,“グリッドによるマルチエージェントシミュレーションの分散処理”, 計算工学講演会論文集, Vol.11, No.2, pp.545-548 (June 2006).
[6] Message Passing Interface Forum, http://www.mpi-forum.org
[7] LAM/MPI Parallel Computing, http://www.lam-mpi.org
[8] MPICH, http://www-unix.mcs.anl.gov/mpi/mpich1/
[9] GridMPI, http://www.gridmpi.org/index.jsp