PG Lab ディスカッションペーパーシリーズ
第41号　2009年7月

ISSN 1880-9146

Discussion Paper Series
No.41 July, 2009

環境情報を考慮したマルチエージェントシミュレーションの並列計算機環境への実装

蟻川 浩，村田 忠彦

文部科学大臣認定　共同利用・共同研究拠点
関西大学ソシオネットワーク戦略研究機構
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
（文部科学省私立大学社会連携研究推進拠点）

Policy Grid Computing Laboratory,
The Research Institute for Socionetwork Strategies,
Joint Usage / Research Center, MEXT, Japan
Kansai University
Suita, Osaka 564-8680, Japan
URL: https://www.pglab.kansai-u.ac.jp/
http://www.kansai-u.ac.jp/riss/
e-mail: pglab.@ml.kandai.jp
tel. 06-6368-1228
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@ml.kandai.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@ml.kandai.jp).

環境情報を考慮したマルチエージェントシミュレーションの並列計算機環境への実装

蟻川 浩[1]，村田　忠彦[1,2]

A Implementation of A Multi Agent Simulation Considering Environmental Information on Parallel Computer

Hiroshi ARIKAWA[1] and Tadahiko MURATA[1,2]

概要

本稿は、環境情報を考慮した環境を有する大規模マルチエージェントシミュレーションを実現することを目的として、大規模な環境情報を複数の環境情報に分割して表現する方法を提案する。提案手法を人工社会モデルのひとつである Sugarscape モデルに対して適用し、提案手法に基づく並列計算機向けシミュレーションプログラムの実装方法と並列処理向けエージェント間の調停ルールを説明する。提案手法に基づく並列処理プログラムを実装し、評価を行った。その結果、提案手法によるシミュレーション結果は非並列シミュレーション結果に類似し、かつシミュレーション時間の短縮が可能である。

Abstract

This paper describes a method to design a large-scale social simulation with the environmental information on parallel computer. To implement a large-scale Multi-Agent Simulation (MAS), a large number of agents and a large size of environmental information should be implemented in the simulation program. In this paper, we propose a representation technique to implement a simulation program on parallel computer. Our proposed technique is to divide environmental information into some sub-environmental information. We firstly explain the proposed technique and a rule which mediate between agents for parallel MAS. We illustrate how to implement sugarscape-based simulation program with the proposed technique for parallel computer system. Then we evaluate the simulation program on a PC cluster system for the large-size environmental information. The simulation results show that there is no significant gap between the experimental data obtained by the proposed technique and the non-parallel technique. We also show that the computation time of the simulation program with the proposed technique can be improved on a PC cluster system.

キーワード：マルチエージェントシミュレーション、シュガースケープモデル、並列処理

Keyword: Multi-Agent Simulation, sugarscape model, parallel computing

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　Policy Grid Computing Laboratory,　Kansai University

2 関西大学 総合情報学部　Faculty of Informatics,　Kansai University

1. はじめに

様々な社会現象を解明する手法として、エージェントシミュレーション技術による社会シミュレーションが注目されている[1,2]。Epstein と Axtell によって提案された人工社会を表現するための Sugarscape モデル[3]はエージェント、環境、ルールの 3 種類によって社会の変化を説明するもので、環境改善資金調達シミュレーション[4]などの応用にみられるように広く支持されている。また、現実社会に適した制度設計を可能にするためのエージェントシミュレーション技術の応用事例として、例えば、社会システムにおける人間の選択行動における実験とエージェントシミュレーションとの比較事例[5]や、人間を含む社会情報システムの挙動把握のためのエージェントモデル利用による分析[6]などがある。一方、マルチエージェントシミュレーションを行うためには、エージェントシミュレーションを実現するための基盤が必要である。マルチエージェントシミュレーションのソフトウェアとしては、Swarm[7]、MASON[8]、RePast[9]が有名であるが、プロトコル記述による大規模マルチエージェントシミュレーションの実現[10]、社会シミュレーション用ツールキットの開発[11,12,13]、スーパーコンピュータ向けエージェントシミュレーションフレームワークの提案と実装[14]、グリッド・コンピューティング技術[15]を活用したエージェントシミュレーションツールキットの拡張[16]、Sugarscape モデルにおけるグリッド・コンピューティングシステムへの適用[17]、既存並列分散処理用通信ライブラリ活用による Sugarscape モデルの実装[18]など、様々な角度から研究開発が行われている。

マルチエージェントシミュレーションによる社会シミュレーションでは通常、エージェントが存在する空間を環境と呼ぶ。この環境は、エージェントだけで構成されることもあるが、エージェントが存在する空間の振る舞いを観察する場合には、その空間をシミュレーションにおいて考慮する必要がある。空間を考慮した環境を想定してシミュレーションを行う場合、表現される環境の大きさと精度に比例して扱われる情報量が大きくなる。例えば、Sugarscape モデルによる大規模マルチエージェントシミュレーションを実現するためには、Sugarscape を表現する 2 次元セル空間のサイズを拡大できなければならない。また、環境情報のサイズ拡大は、シミュレーションで取り扱うエージェント数の増加にもつながる。問題規模の拡大はエージェント意思決定処理等にかかる演算負荷が増大することを意味する。数十万および数百万のエージェントによるシミュレーションを Sugarscape モデルによって実現する場合には、大規模な記憶領域の確保および演算処理にかかる負荷低減の 2 つを同時に達成することが要求される。

これらの要求に対処するため、近年、PC クラスタシステムなどの分散メモリ型並列計算機の導入が進んでいること、またその計算機を有効利用するための並列計算プログラムの実装手法が浸透してきたことを踏まえ、本論文では分散メモリ型並列計算機を指向した大規模マルチエージェントシミュレーションに適した環境情報の表現方法を提案する。提案手法を Sugarscape モデルに適用するための方法について示すとともに Sugarscape モデルの

並列化によるシミュレーションプログラムの実装上の課題と解決策、評価実験による知見を示す。さらに、提案手法の有用性、大規模問題への適用可能性、並列処理によるシミュレーション時間削減の効果および並列処理の有用性について示す。

2. 並列処理に適した環境情報の拡張表現

2.1 環境情報を表現するための方法と並列化への拡張

マルチエージェントシミュレーションではエージェントによって構成される空間を環境とし、他エージェントからの相互影響、および環境から受ける影響によって生じるエージェントおよび環境の変化を社会の変化ととらえる。エージェントシミュレーションにおける環境について、物理空間として表現するか否かの 2 種類が考えられる。本論文では、物理空間を表現する場合とは Sugarscape モデルや交通流のように空間をある座標系に置き換えて表現することと定義する。一方、物理空間を表現しない場合とは金融市場やオークションなどのように取引を仲介する場として表現されることと定義する。

物理空間を表現した環境を有するマルチエージェントシミュレーションプログラムを実装する場合、通常、ELASTIC[12]の実装方法にみられるようにエージェントとは別に環境情報を定義する。シミュレーションで表現したいエージェント数および環境の規模に基づいてそれぞれの記憶領域を計算機上に確保することになる。

本論文は、PC クラスタやスカラ型スーパーコンピュータなどの分散メモリ型並列計算機およびグリッド・コンピューティング等の広域分散環境に存在する計算機資源に基づく大規模計算機環境を利用することを考慮して、環境情報の拡張表現方法を提案する。

図 1 に本論文で提案する環境情報の拡張表現方法の概念を示す。本論文では、シミュレーションの問題サイズによって規定される環境情報をグローバル領域と定義する。これに対し、グローバル領域を複数の領域に分割したものをローカル領域と定義する。図 1 に示すように、ローカル領域は他のローカル領域と連携をとることができる。ローカル領域と連携がある領域を隣接領域と定義する。本論文で提案する環境情報の拡張表現方法により、問題サイズの拡大はローカル領域の数を増やすことで対応する。

本論文で提案する拡張表現方法は、Swarm[7]や SOARS[11]等で扱われている、環境情報をエージェントもしくはそれに準じたものとして表現する方法にも適用できる。生物エージェントと無生物エージェントの 2 つを定義している Swarm では、通常、環境情報は無生物エージェントの 1 つとして環境エージェントを定義する。提案手法を適用する場合は、ローカル領域の数だけ環境エージェントを定義するとともに、各環境エージェントが他の環境エージェントと情報交換する仕組みを設ける。

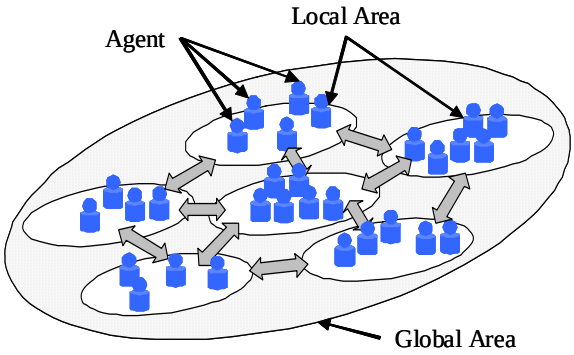


図 1: 環境情報の拡張表現

2.2 Sugarscape における環境情報の拡張

本節では Epstein と Axtell が開発した人工社会モデルである Sugarscape モデル[3]を例として、2.1 節に示した環境情報の拡張表現方法を説明する。

Sugarscape モデルは、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る。エージェントおよび環境の各種ルールによってもたらされるエージェントの時間変化による行動を観察し、人工社会の変化をとらえるものである。エージェントは視力、代謝率、性別、初期財産、寿命、交配可能年齢、状況によって変動する現在位置、財産、年齢などを内部状態としてもつ。環境は格子状の 2 次元空間で表現される。各格子をセルと定義する。セルにはエージェントが生きるために必要な資源が定義されている。本論文における環境空間は 2 次元トーラス空間と定義する。また、1 セルあたり最大 1 エージェントしか存在できないと定義する。エージェントには移動、資源収集および代謝、交配、寿命の各ルールが、環境には資源回復ルールがある。

Sugarscape モデルによる大規模マルチエージェントシミュレーションの実現において、問題規模の拡大は、通常、環境のサイズを拡大することである。具体的にはセル数を増加させることである。Sugarscape モデルは単純なルールに基づく演算であるがエージェント数およびセル数分だけ繰り返し演算することになる。問題規模の拡大による演算回数はエージェント数、セル数、時間変化量に比例する。また、セルおよびエージェントに比例してセルおよびエージェントの各種情報を記憶しなければならない。Sugarscape モデルによる大規模シミュレーションを実現するためには、問題規模の変化に対応できるようなシミュレーションプログラム実装上の工夫が求められる。具体的には、環境を表現するセル数およびエージェント数の拡大に対して大規模な記憶領域を柔軟に確保できる方法を、また演算回数の増加に対する技術的な解決が求められる。

本論文は、2.1 節にて提案した環境情報の拡張表現にもとづいて Sugarscape モデルの環境情報を拡張する。以後、提案する環境情報の拡張表現により表現された Sugarscape モデルを並列 Sugarscape モデルと呼ぶ。

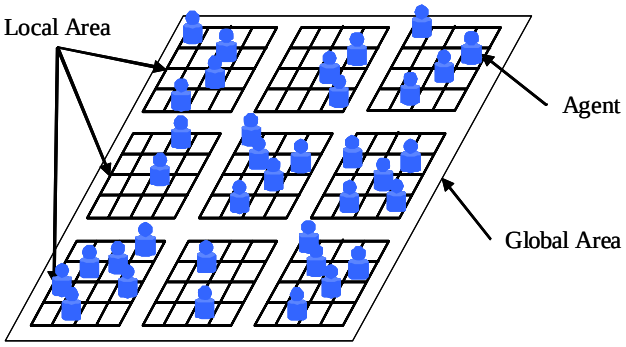


図 2: 並列 Sugarscape モデル

並列 Sugarscape モデルは並列計算機上で実行することを前提として、図 2 のようにシミュレーションで想定するサイズの環境を複数の小環境に分割する。分割された小環境がローカル領域になる。また、分割された小環境の集合および隣り合う小環境がそれぞれグローバル領域、隣接領域となる。並列 Sugarscape モデルにおいて、ローカル領域の記憶領域は並列計算機を構成する各演算ノードの主記憶領域に確保される。

エージェントは、シミュレーションの初期状態によりいずれかのローカル領域に必ず配置される。隣接領域の近くに存在するエージェントは隣接領域の情報を得ることを可能とするとともに、意思決定処理の結果隣接領域に移動するのが妥当である場合には、隣接領域に移動することができる。各種ルールに基づく演算処理はすべてのローカル領域で共通とする。

3. 並列 Sugarscape モデルによるシミュレーションプログラムの実装

3.1 対象とする並列計算機と通信モデル

並列分散処理プログラムの実装を行うために、対象とする計算機システムおよび通信モデルを明確にする。本論文では分散メモリ型並列計算機を対象計算機とする。分散メモリ型並列計算機は PC クラスタのように独立した主記憶装置を有する計算機が複数あり、それらが相互結合ネットワークで接続されているものとする。本論文では、独立した主記憶を有する計算機を演算ノードと定義する。スカラ型スーパーコンピュータや PC クラスタの構成により、1 計算機あたりマルチコアプロセッサと呼ばれる複数の演算コアが内蔵されたプロセッサが複数搭載されている、もしくはプロセッサが複数搭載されている場合がある。演算コアもしくはプロセッサに対して独立したプロセスが割り当てられる場合は、演算コア 1 つもしくはプロセッサを 1 演算ノードとみなす。一方、グリッド・コンピューティング技術によって構築した分散メモリ型並列計算機環境も前述の定義に該当するが、並列プログラムの実装方法および実行方法が複雑になるため、本論文では対象としない。なお、分散メモリ型並列計算機について、以後、並列計算機と省略する。

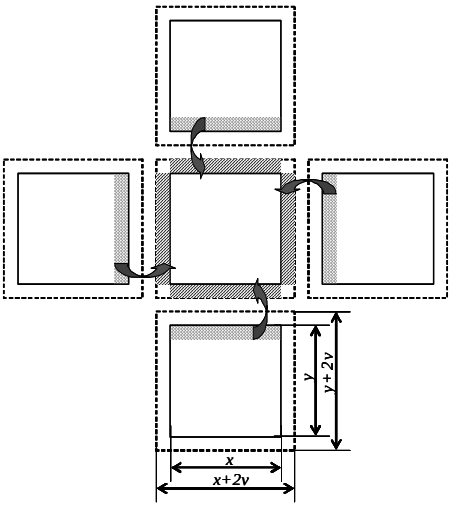


図 3: ローカル領域におけるセル空間の表現と隣接領域との関係

並列計算機で利用する通信モデルは、メッセージ交換モデルを採用する。メッセージ交換モデルは分散メモリ型並列計算機で広く使われるモデルのひとつで、プロセス間、もしくはプロセス全体で通信が行われるモデルである。特に、Message Passing Interface (MPI) は仕様が策定されており、それに基づいて実装された通信ライブラリ、いわゆる MPI 通信ライブラリが数多く存在する。本論文は MPI 通信ライブラリを用いて実装することを前提とする。

3.2 並列 Sugarscape モデルにおける環境情報の主記憶上での確保

並列 Sugarscape モデルでは複数のローカル領域で構成される。従来の Sugarscape モデルでは、エージェントは自身の意思決定によって環境情報内を移動できる。従来の Sugarscape モデルでのエージェントの移動を維持すべく、エージェントが複数のローカル領域で構成される環境情報内を移動できるようにする仕組みを導入する。

まず、エージェントの移動可能範囲と同一方向にあるローカル領域を隣接領域と定義する。次に、エージェントが隣接領域へ移動することを考慮して、エージェントの移動可能範囲と同一方向の隣接領域と情報交換する。図 3 にローカル領域におけるセル空間の表現と隣接領域間との関係を示す。本論文では、あるローカル領域の隣接領域が上下左右に存在することを前提に説明する。並列 Sugarscape モデルで表現されるローカル領域は各方向の長さをそれぞれ x および y としたとき、$x \times y$ で表現される領域である。エージェントの隣接領域間の移動を可能にするためには、隣接領域間の情報を得られなければならない。そこで、隣接領域間のセル情報をオーバラップさせる。具体的には、エージェントの最大視野 v として、上下左右にエージェントの最大視野分の情報をオーバラップさせる。図 3 の斜線部が隣接領域間でオーバラップされる分である。ローカル領域が保持するセル数の全体は $(x+2v)(y+2v)$ 分となる。演算ノードは $(x+2v)(y+2v)$ で表現されるセル数に相当する

主記憶空間を確保する。オーバラップさせる領域の情報は隣接領域から得る。図 3 網掛け部がオーバラップさせる領域の情報源である。図 3 網掛け部の各セルの資源量、セルの占有状況、他の領域へ移動するエージェントの各情報が図 3 斜線部に送信される。

3.3 先住民優先による衝突回避ルールの導入

各ローカル領域の境界近くに存在するエージェントは隣接領域の空きセルを移動候補とすることができる。意思決定処理の結果、隣接領域の空きセルを移動予定先とすることができる。しかし、隣接領域に存在するエージェントの意思決定と衝突する場合がある。

通常、Sugarscape モデルでは唯一の環境にすべてのエージェントが存在することが前提なので、意思決定処理の段階で衝突が回避できる。一方、並列 Sugarscape モデルは、大規模シミュレーションの実現のために環境を複数の小環境として分割するので、唯一の環境にすべてのエージェントが存在するという前提が成り立たない。並列 Sugarscape モデルにおいては、 (1)グローバル領域において衝突回避の調停を実現する、(2) 各ローカル領域において衝突回避の調停を実現する、のいずれかを選択してエージェントの衝突回避を実現しなければならない。(1)は、グローバル領域がすべてのエージェントの移動先およびセルの空き状況を把握するため、衝突回避の調停はこれまでの Sugarscape モデルと同様でよい。但し、環境情報とエージェント情報を複数の演算ノードで管理する場合、グローバル領域を実現するために新たな記憶領域の確保が必要になる。具体的には、すべてのエージェントの移動予定先、すべてのセルの空き状況を把握するための記憶領域の確保が必要になるため、(1)を並列計算機で実現するのは事実上困難である。したがって、本論文では(2)の方法を採用する。各ローカル領域にて衝突回避の調停を行い、隣接領域から移動してきたエージェントに対して追加の衝突回避ルールを適用する。隣接領域から移動してきたエージェントに対して追加の衝突回避ルールを適用することで、グローバル領域を表現することなく並列計算機上で衝突回避の調停ができる。本節では、隣接領域から移動してきたエージェントに対する追加の衝突回避ルールとして、「ローカル領域に存在するエージェントの選択結果は、隣接領域から移動してきたエージェントの選択結果よりも優先される」というルールを導入する。本稿では、先住民優先による衝突回避ルールと呼ぶ。

図 4 を用いて衝突回避ルールを説明する。図 4 では隣接領域 N と N+1 において、領域 N+1 から領域 N に移動するエージェントが領域 N に存在するエージェントとあるセルで衝突が発生する場合(図 4 の before の状態)、先住民である領域 N に存在していたエージェントが当該セルの確保できる。一方、領域 N+1 から移動してきたエージェントは元の領域である領域 N+1 には戻らず、領域 N 内における空きセルを新たな移動先とする。具体的には、衝突発生したセルを中心とした斜線部内で最寄りの空きセルを新たな移動先とする（図 4 の after における状態）。先住民優先による衝突回避ルールにより、1 シミュレーション期間内において、隣接領域間でのエージェントの情報交換回数が高々1 回で完了する。

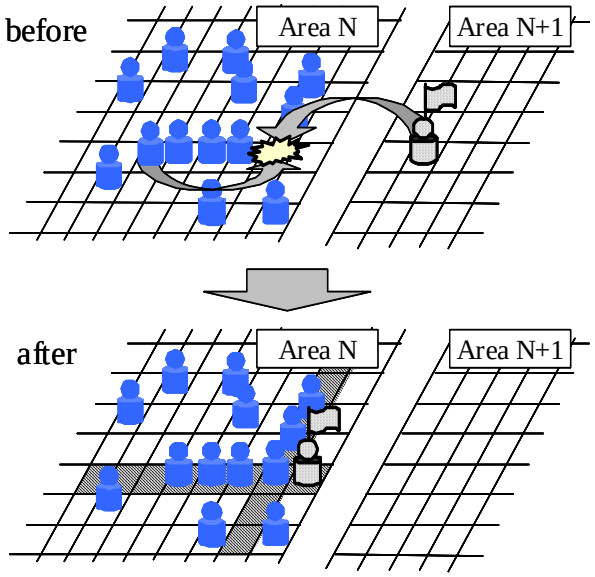


図 4: 先住民優先による衝突回避ルール

3.4 並列 Sugarscape モデルにおける動作フロー

通常、Sugarscape モデルによるマルチエージェントシミュレーションのアルゴリズムは、以下に示す Step 1 から Step 6 をシミュレーション期間分だけ繰り返す。

Step 1: すべてのエージェントは自身の視野にもとづいて最大の資源を確保できるセルを探索し、移動先を決定する。複数のエージェントがある 1 つのセルを移動先として選択した場合、いわゆる衝突が発生した場合は移動元のセルに戻る。

Step 2: すべてのエージェントは資源回収ルールに基づいて移動先から資源を回収する。

Step 3: すべてのエージェントは交配ルールに基づいて交配を行う。交配により、新たなエージェントが誕生する。

Step 4: すべてのエージェントに対して寿命ルールを適用する。寿命を迎えたエージェントはこの時点で排除される。

Step 5: すべてのセルに対して、資源回復ルールを適用し、セルの資源量を更新する。また、エージェントが存在するセルは、汚染物質生成ルールに基づいて算出した汚染物質量を資源量から相殺する。

Step 6: データの集計を行う。

並列 Sugarscape モデルでは、先に示した 6 つのステップは各ローカル領域で実行される。また、並列 Sugarscape モデルでは各ローカル領域で情報交換が発生する必要がある。そこ

で、Step 1, 5, 6 に対して以下の拡張を行う。

Step 1 を次のように修正する。

Step 1-1: すべてのエージェントは自身の視野にもとづいて最大の資源を確保するセルを探索する。この時点では、移動予定先とする。

Step 1-2: ローカル領域に留まるエージェントと隣接領域に移動するエージェントを分離する。ローカル領域に留まるエージェントについては、移動予定先を移動先とする。

Step 1-3: 隣接領域に移動するエージェントについて、隣接領域に情報を送信する。また、隣接領域から情報を受信する。

Step 1-4: 隣接領域から移動してきたエージェントに対して、3.3 節で示した先住民優先による衝突回避ルールを適用し、移動先を決定する。

Step 5 を次のように修正する。

Step 5-1: すべてのセルに対して、資源回復ルールを適用し、セルの資源量を更新する。また、エージェントが存在するセルは、汚染物質生成ルールに基づいて算出した汚染物質量を資源量から相殺する。

Step 5-2: 各ローカル領域は隣接領域に資源量の最新情報を送信する。また、隣接領域から受信した資源量の最新情報をローカル領域の環境情報に反映させる。

Step 6 を次のように修正する。

Step 6-1: 各ローカル領域において、エージェント数、資源量、汚染物質量などの情報を集計する。

Step 6-2: 各ローカル領域において集計した各種結果を、集計担当のローカル領域に集約する。

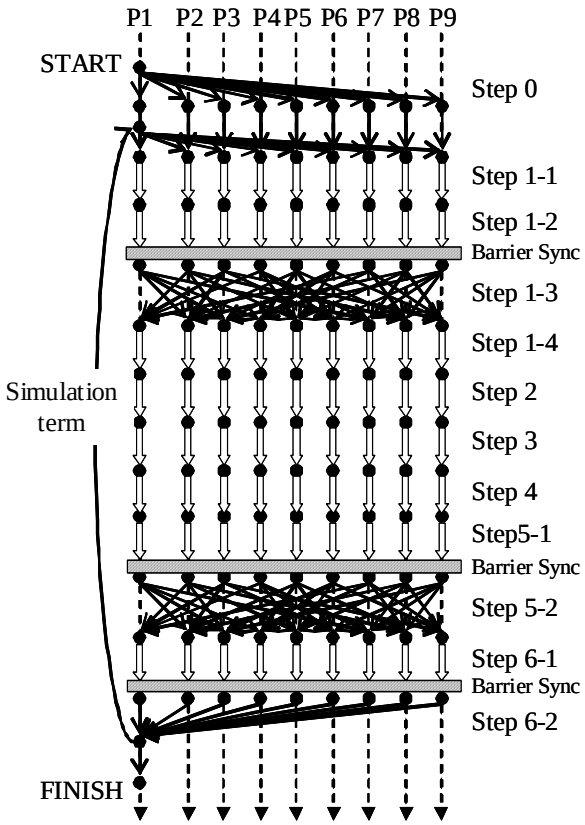


図 5: 並列 Sugarscape における動作フロー

図 5 に並列 Sugarscape モデルおける並列計算機上での動作フローを示す。Step 0 では MPI ライブラリを用いるための手続きを行う。まず、演算ノードとローカル領域との対応づけが必要になる。MPI では rank と呼ばれる識別子によってプロセスを認識するので、ローカル領域と rank 値とを対応づける。次に、隣接領域の rank 値を通知する。最後に、初期条件のデータファイルを読み込む。シミュレーション時間の同期および隣接領域間のデータ交換タイミングを図るために、Step 1-3、Step 5-2、Step 6-2 実施前にバリア同期を実施する。先住民優先による衝突回避ルールの導入により、隣接領域との情報交換は、Step 1-3 および Step 5-2 の高々2 回で完了する。

4. 実験および評価

4.1 シミュレーション結果への影響

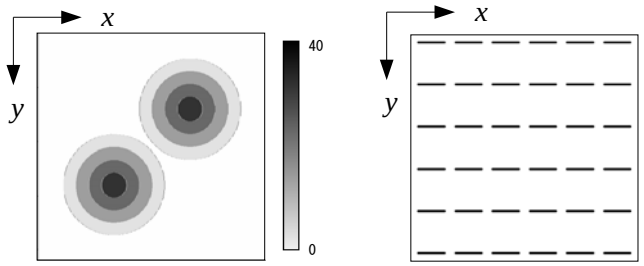
提案する並列 Sugarscape モデルに基づいて実装したシミュレーションプログラムのシミュレーション結果への影響について評価する。非並列シミュレーション結果との比較することで、先住民優先による衝突回避ルールの影響を検証する。

計算時間短縮の効果を測定する際に用いた並列計算機システムを以下に示す。演算ノー

ドは Intel 社 3.0EGHz Hyper Threading Technology (HTT) 搭載 Pentium4 プロセッサ 1 基および 2GByte の主記憶を搭載した計算機 40 台、ファイルサーバは Intel 社 3.2GHz Hyper Threading Technology (HTT) 搭載 Xeon プロセッサ 1 基および 1GByte の主記憶を搭載した計算機 2 台、タスク管理ノードは Intel 社 3.2GHz Hyper Threading Technology (HTT) 搭載 Xeon プロセッサ 2 基および 2GByte の主記憶を搭載した計算機 1 台で構成されている。すべての演算ノードとファイルサーバは 1000Base-T のネットワークカードを有しており、1000Base-T 対応のネットワーク機器に接続されている。したがって、ノード間通信の最大通信速度は 1Gbps である。すべての演算ノードは HTT を有効にしているため、演算ノードのオペレーティングシステムにより 2 つの演算コアを有するプロセッサとして認識されている。したがって、評価実験で用いた計算機システムは最大 80 プロセスの並列実行が可能である。

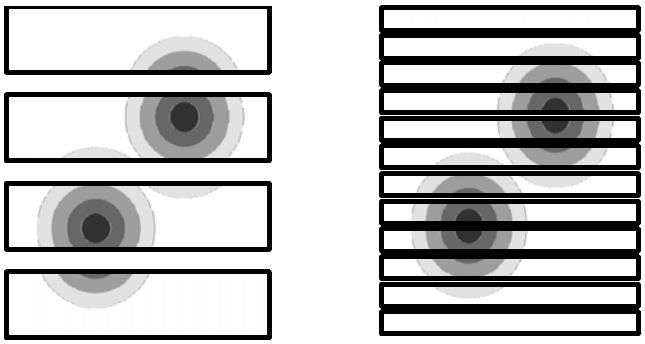
本論文では、環境改善資金調達シミュレーションモデル[4]を参照モデルとして提案モデルに基づくシミュレーションプログラムの実装を行った。シミュレーションプログラムは C 言語を用いて実装した。C コンパイラおよび MPI ライブラリはそれぞれ GNU Compiler version 3.4.6 および GridMPI 2.0 である。評価実験で用いたシミュレーションプログラムでは、環境情報の一部およびエージェント情報の記憶領域を確保するために mmap()システムコール[19]を利用した。mmap()システムコールはファイルシステム上に必要な大きさの記憶空間を確保するためのシステムコールである。mmap()システムコールの挙動はオペレーティングシステムが提供するページング機構およびスワッピング機構と同じである。mmap()システムコールの特徴としては、主記憶装置を部分的に利用するため、データアクセスの動作はファイルを常にアクセスする場合に比べて高速である。また、オペレーティングシステムのページングおよびスワッピングをプロセスが独立して行うため、プロセスがオペレーティングシステムの挙動に左右されない。さらに、通常、計算機上での記憶領域の確保は malloc()関数を利用するが、malloc()関数では大規模かつ異なるサイズの記憶領域を確保できない場合がある。一方、mmap()システムコールでは大規模かつ異なるサイズの記憶領域を確保できる。このような利点に着目し、本論文では mmap()システムコールを用いた。

評価実験におけるシミュレーションの条件を以下に示す。環境情報のサイズ、シミュレーション期間はそれぞれ 2100×2100 (4410000 セル)、1000 期間とする。環境情報である資源量の初期分布は図 6-(a)に示すように、(1400,700)および(700,1400)を頂点として同心円上に分布する。初期エージェント数は 27000 とし、図 6-(b)のように配置する。その他の条件およびルールは文献[4]に従う。文献[4]では資金調達のための社会システムを導入しない場合、募金システムを導入した場合、事前くじシステムを導入した場合の 3 種類のシミュレーションが例示されているが、本論文は資金調達のための社会システムを導入しない場合のシミュレーションのみ取り扱う。



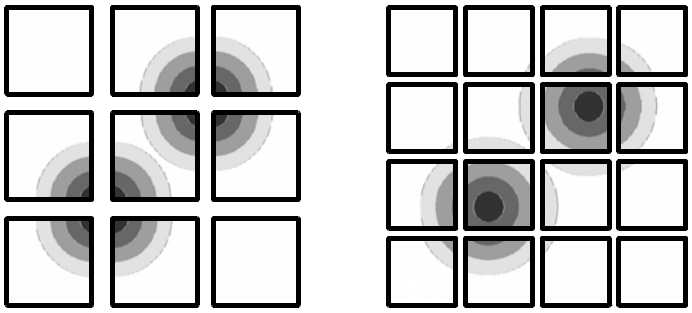
(a) resource map (b) agent position

図 6: 初期状態における資源量およびエージェント位置



A: (1 ×4) B: (1 ×12)

図 7: 環境情報の分割方法(1 次元分割)



A: (3 ×3) B: (4 ×4)

図 8: 環境情報の分割方法(2 次元分割)

Sugarscape モデルにおける環境情報の分割方法として、本論文は図 7 および図 8 に示すように、1 次元分割および 2 次元分割の 2 種類を採用する。前者は図 7 のように y 方向を m 分割する。一方、後者は図 8 のように x, y 各方向に対して n 分割する。以後、1 次元分割および 2 次元分割をそれぞれ(1×m)、 (n×n)と表記する。

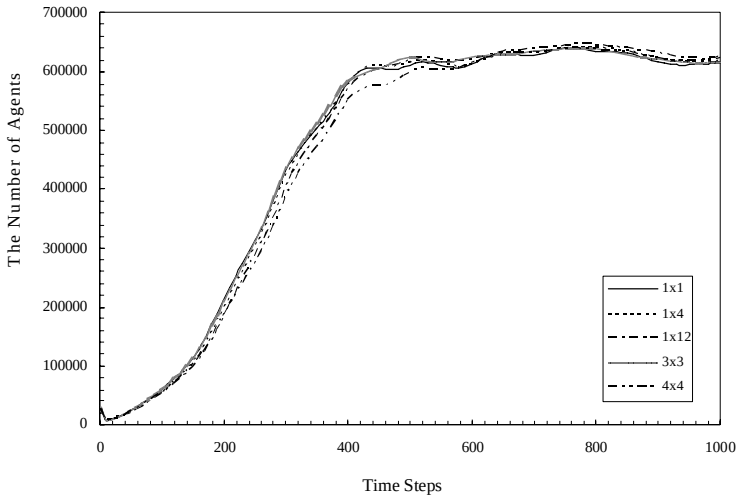


図 9: エージェント数の推移

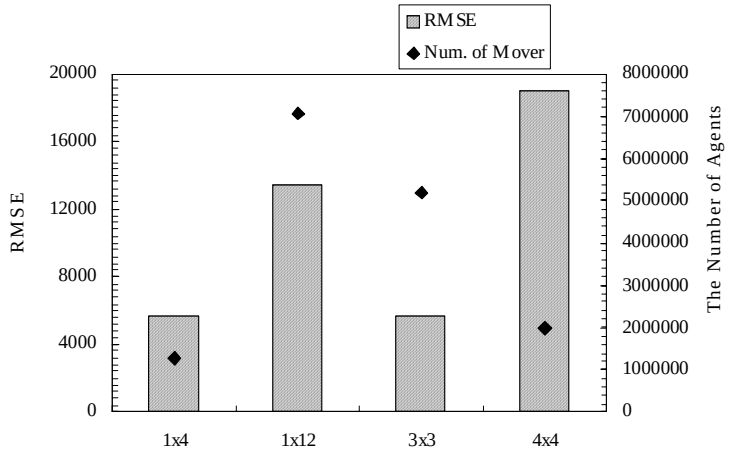


図 10: 非並列シミュレーション結果との平均二乗誤差の平方根と隣接領域に移動したエージェント数

図 9 および図 10 はそれぞれ 5 試行におけるエージェント数、非並列シミュレーションの結果を基準とした場合の平均二乗誤差の平方根と隣接領域に移動したエージェント数の結果である。図 9 および図 10 において、1 次元分割については(1×4)、(1×12)を、2 次元分割については(3×3)、(4×4)の実験結果を示す。また、図 9 において、(1×1)は非並列シミュレーション、つまり分割しなかった場合の結果である。図 10 の縦軸に示す RMSE は式(1)で表わされる。

$$RMSE = \sqrt{\frac{\sum_{i=1}^{i=n} \{A_P(i) - A_{np}(i)\}^2}{n}} \qquad (1)$$

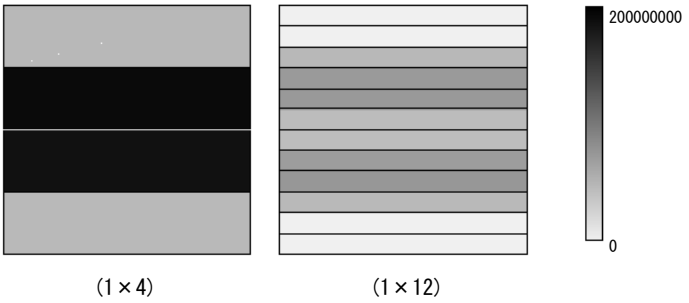


図 11 : 領域別エージェント意思決定処理実行回数(1 次元分割)

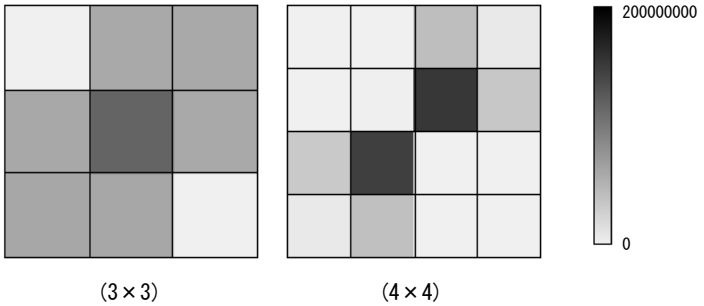


図 12 : 領域別エージェント意思決定処理実行回数(2 次元分割)

ここで、$A_p(i)$ および $A_{np}(i)$ はそれぞれシミュレーション期間 i における並列および非並列シミュレーションのエージェント数を意味する。

図 11 および図 12 はエージェント意思決定処理実行回数を領域別に示したものである。図 11 および図 12 はそれぞれ 1 次元分割および 2 次元分割のものである。図中の実線で囲まれた領域はローカル領域を意味する。

非並列シミュレーションとの計算結果の比較について、図 9 よりシミュレーション期間が 20 から 640 の間は交配ルールと寿命ルールによってエージェント数の増減が著しいため、非並列シミュレーションと並列シミュレーションとのエージェント数の比較で差が生じやすいことがわかる。一方、シミュレーション期間が 640 以降においてエージェント数の増減が安定するため、非並列シミュレーションと並列シミュレーションとのエージェント数の差はほとんどみられない。この結果から、並列処理によるシミュレーションは非並列シミュレーションの結果に類似した結果を得られるといえる。

次に、並列 Sugarscape モデルにおける、並列プロセス数とシミュレーション結果の関係について、図 10 より 1 次元分割、2 次元分割において並列プロセス数の増加により非並列シミュレーションとのエージェント数に差が生じやすくなることが伺える。一方、図 9 のエージェント数の推移と図 10 の RMSE の値から、エージェントの推移のオーダに対して、RMSE の値が高々3%程度である。また、図 10 の隣接領域に移動したエージェント数(Num.

of Movers の値)について、(1×4)および(4×4)の結果が低く、(1×12)および(3×3)の結果が高いのは、特定のローカル領域に資源量が多く配置されているからと読み取れる。図 10 の隣接領域に移動したエージェント数、図 11、図 12 より、環境情報の分割は初期条件に影響されることが言える。これらの結果から、環境情報は 1 次元分割で、かつ分割数が少ないほど非並列シミュレーションの結果に類似することがわかった。

4.2 計算時間短縮の効果

並列 Sugarscape モデルに基づいて実装した大規模マルチエージェントシミュレーションプログラムによる計算時間短縮の効果について評価する。並列 Sugarscape モデルでは、環境情報を複数のプロセッサに分散配置させるとともに各種計算を並列実行させるため、環境情報の分割方法が計算時間に影響を及ぼす。本節では、環境情報の分割方法が計算時間に及ぼす効果について考察する。

シミュレーション結果の影響を測定する際に用いた並列計算機システムは 4.1 節の実験と同様である。また、シミュレーションプログラムおよびシミュレーション条件は 4.1 節と同様である。

表 1 は(1×1)、(1×4)、(1×12)、(3×3)、(4×4)の 5 種類について 5 試行の平均実行時間(Time)、高速化率(Speedup)、並列化効率(Efficiency)を示す。表中 *P*、1D、2D はそれぞれプロセス数、1 次元分割、2 次元分割を意味する。本論文では、実行時間は図 4 の Step 1-1 から Step 6-2 までの各処理にかかる時間の合計と定義する。各処理の時間は各処理実行前のチェックポイントを通過してから、各処理実行後のチェックポイントを通過するまでとする。時間測定用のチェックポイントは MPI_Wtime()を利用した。

表 1 中の高速化率および並列化効率はそれぞれ式(2)および式(3)で定義される。

$$S(P) = \frac{T(1)}{T(P)} \tag{2}$$

$$E(P) = \frac{S(P)}{P} = \frac{T(1)}{P \cdot T(P)} \tag{3}$$

ここで、$T(1)$ および $T(P)$ はそれぞれ 1 プロセスにおける計算時間、プロセス数 P による計算時間と定義する。高速化率および並列化効率は、それぞれプロセス数の増加により計算時間が短縮できることを表す指標、各演算ノードが担当する演算量の均質さを表す指標である。

表 1: 計算時間

		P	Time T(P)	Speedup S(P)	Efficiency E(P)
	1x1	1	2375388.00	1.00	-
1D	1x4	4	500395.48	4.75	1.19
	1x12	12	93647.18	25.37	2.11
2D	3x3	9	200367.04	11.86	1.32
	4x4	16	263638.76	9.01	0.56

並列 Sugarscape モデルに基づく大規模マルチエージェントシミュレーションプログラムの計算時間短縮効果について、本論文では、並列プロセス数が及ぼす効果および分割方法の違いが及ぼす効果について評価する。

並列プロセス数の増加が及ぼす効果について、並列プロセス数の実行時間が 1 プロセスのそれに比べて大幅に短縮した。並列プロセス数の増加により計算時間短縮の効果がある。また、(4×4)の場合を除いて高速化率がプロセス数を超えた。通常、プロセス間通信によって情報を同期するような並列処理プログラムは高速化率がプロセス数を超えない。評価実験において高速化率が概ねプロセス数を超えたのは、mmap()システムコールを利用したことが理由のひとつである。4.1 節で述べたように、mmap()システムコールはページングおよびスワッピングはプロセス自身のタイミングで発生する。並列処理によって各プロセスが確保する記憶領域のサイズは(1×1)よりも小さくなること、また mmap()システムコールによるページング等の処理時間が並列処理により複数のプロセスで同時に発生することから、1 シミュレーション期間においてページング等の発生による計算時間の遅延が複数のプロセスで分散できた。ゆえに、多くの場合において高速化率が高い値を示したといえる。

分割方法の違いにより計算時間に及ぼす効果について、(4×4)の場合を除いて並列化効率が 1.0 を超えた。 (4×4)において並列化効率が 1.0 を下回ったのは、エージェント情報が特定の演算ノードに集約されたために起きたと説明できる。通常、バリア同期を有する並列プログラムは、演算量の多い演算ノードの影響を受ける。ゆえに、シミュレーション時間は演算量の多い演算ノードの処理時間に支配される。本論文での評価実験では、すべての演算ノードが担当するローカル領域のサイズを均等にしたので、演算処理時間は各ローカル領域のエージェント意思決定処理回数に支配される。図 11 および図 12 より、(1×12)、(3×3)においてはエージェント意思決定処理回数が複数の領域で分散しているが、(4×4)においてはエージェント意思決定処理回数が特定の領域に集中している。結果、(4×4)では特定の演算ノードに影響されたために、並列化効率が低くなった。並列化効率を向上させるためには、資源量の分布に応じて分割方法を選択することが望ましい。

5. おわりに

本論文は、環境情報を考慮した大規模マルチエージェントシミュレーションの実現のために、並列処理に適した環境情報の拡張表現を提案し、社会シミュレーションの一手法として人工社会を表現するモデルである Sugarscape モデルに適用した。並列 Sugarscape モデルは PC クラスタ等にみられる分散メモリ型並列計算機上で実行されることを考慮して、ひとつの大規模な環境情報を複数の小環境情報にて構成されるモデルとして拡張した。Sugarscape モデルへの適用に際して、隣接領域を移動するエージェントに対する新たに衝突回避ルールを提案した。また、提案する拡張表現方法に基づいて表現した並列 Sugarscape モデルによる大規模マルチエージェントシミュレーションプログラムの実装について、分散メモリ型並列計算機上で動作するためのプログラムの実装方法を明らかにした。

並列 Sugarscape モデルに基づくマルチエージェントシミュレーションについて、環境改善資金調達シミュレーションを例題としてプログラムを実装し、分散メモリ型並列計算機を用いて評価を行った。本論文では 2100×2100 の環境情報のサイズについて非並列シミュレーションと並列シミュレーションの 2 種類を実施した。非並列シミュレーションと並列シミュレーションの 2 種類の計算結果を比較した結果、従来の Sugarscape モデルによるシミュレーション結果と同様の傾向が得られることを示した。また、先行研究と比較してセル数にして 1764 倍となる問題規模のシミュレーションを可能にし、分割方法および並列プロセス数によるものの最大 25 倍の高速化率を達成することができた。

今後の課題は、衝突回避ルールによるシミュレーション誤差の更なる削減について資源量の分布、領域分割、エージェント移動の観点から検討することである。また、さらに大規模な問題サイズにおける並列 Sugarscape モデルでの社会シミュレーションを通じて、Sugarscape モデルにおける大規模シミュレーション実現のための問題点と解決策について明らかにする。

参 考 文 献

[1] R. Conte, N. Gilbert, J. S. Sichman, MAS and social simulation: A suitable commitment, *Proc. of the First International Workshop on Multi-Agent Systems and Agent Based Simulation* (Lecture notes in computer science 1534, Springer, Berlin, Germany), pp. 1-9, 1998.

[2] T. Terano, S. Takahashi, D. L. Sallach, J. Rouchier (eds), *Proc. of the First World Congress on Social Simulation*, August 21-25, 2006.

[3] J. M. Epstein, R. Axtell, *Growing Artificial Societies: Social science from the bottom up*, MIT Press, 1996.

[4] 西﨑 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.

[5] R. Selten, T. Chmura, T. Pitz, S. Kubec, M. Schreckenberg, Commuters route choice behaviour, *Games and Economic Behavior*, Vol.58, pp-394-406,2007

[6] Y. Murakami, T. Ishida, T. Kawasoe and R. Hishiyama, Scenario description for multi-agent simulation, *Proc. Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pp.369-376, 2003

[7] N. Minar, R. Burkhart, C. Langton and M. Askenazi, The Swarm simulation system: A toolkit for building multi-agent simulations, http://www.santafe.edu/projects/
swarm/overview.ps, 1996

[8] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, MASON: A Multiagent Simulation Environment, *Simulation*, Vol. 81, Issue 7, pp.517-527, 2005

[9] M. J. North, N. T. Collier and J. R. Vos, Experiences creating three implementations of the repast agent modeling toolkit, *ACM Transaction on Modeling and Computer Simulation*, Vol.16, Issue 1, pp. 1–25, 2006.

[10] 中島 悠, 椎名 宏徳, 山根 昇平, 八槇 博史, 石田 亨, 大規模マルチエージェントシミュレーションにおけるプロトコル記述と実行基盤, 電子情報通信学会論文誌, Vol. J89-D, No.10, pp.2229-2236, 2006

[11] 田沼 英樹, 出口 弘, エージェントベース社会シミュレーション言語 SOARS の開発, 電子情報通信学会論文誌, Vol.J90-D, No.9, pp.2415-2422, 2007

[12] H. Arikawa, S. Morishita, T. Murata, Policy-Making Assistance Using ELASTIC: Enhanced Large-Scale Agent Simulation Toolkit for Innovative Community, *Proc. of International Conference of Socionetwork Strategies and Policy Grid Computing 2008*, pp.109-122, 2008

[13] 井庭 崇, 中鉢 欣秀, 松澤 芳昭, 海保 研, 武藤 佳恭, Boxed Economy Foundation Model: 社会・経済のエージェントベースモデリングのためのフレームワーク, 情報処理学

会論文誌, pp.20-30, Vol.44, No.SIG14 (TOM9), 2003
[14] T. Takahashi, H. Mizuta, Efficient agent-based simulation framework for multi-node supercomputers, *Proc. of 2006 Winter Simulation Conference*, pp. 919-925, 2006.
[15] I. Foster and C. Kesselman (eds.), *The Grid: Blueprint for a New Computing Infrastructure* Second Edition, Morgan Kaufmann, 2004.
[16] D. Chen, K.G. Theodoropoulos, T.S. Turner, W.T. Cai, R. Minson, Y. Zhang, Large scale agent-based simulation on the grid, *Future Generation Computer Systems*, Vol.24, Issue 7, pp.58-71, 2008
[17] H. Arikawa, S. Morishita, T. Murata, Performance Improvement of GridRPC-based Multi-Agent Simulation Software, *PG Lab Discussion Paper Series*, No.20, pp.1-14, 2007
[18] T. Murata, H. Arikawa, S. Morishita, T. Maeda, A Design of Problem Solving Environments for Policy Making Asssistance Using MAS-based Social Simulation, *Proc. of Third IEEE International Conference on e-Science and Grid Computing*, pp.521-528, 2007
[19] B.O. Gallmeister, *POSIX.4 Programmers Guide*, O'Reilly, pp. 128-129 and 389-391, 1995.