原著論文

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並列計算機上での実現[†]

蟻川 浩^{*1・2}・村田 忠彦^{*2・3・4}

本論文では、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。また、大規模な問題サイズのシミュレーションによる評価実験を PC クラスタ上で実施し、提案手法によるシミュレーション結果の非並列シミュレーション結果との類似傾向と、シミュレーション時間の短縮可能性を示す。

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

1. まえがき

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

[†] A Large-scale Multi Agent Simulation Considering Environmental Information and Its Implementation on Parallel Computer
Hiroshi ARIKAWA and Tadahiko MURATA

*1 関西大学 研究推進部

Division of Research Development, Kansai University

*2 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

*3 関西大学 総合情報学部

Faculty of Informatics, Kansai University

*4 関西大学 ソシオネットワーク戦略研究機構

Research Institute for Socionetwork Strategies, Kansai University

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の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図を示す。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

図1 環境情報の拡張表現 (概念図)

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ふたつを定義している Swarm では、通常、環境情報は無生物エージェントのひと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。

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持つ。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。環境はセルが2次元空間に分布しており、エージェントが生きてするために必要な資源が配置されている。なお、本論文では環境は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。エージェントおよび環境には、それぞれの内部状態を変更するルールがある。前者には移動、資源収集および代謝、交配、寿命の各ルールが、後者には資源回復ルールがある。Sugarscape モデルによるマルチエージェントシミュレーションは、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。

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基づいて 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。

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が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。

図2 並列 Sugarscape モデル

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。スカラー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、演算コアひとつもしくはプロセッサを1 演算ノードとみなす。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。

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が、プロセス間およびプロセス全体で通信できるライブラリであれば適用可能である。

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、環境情報が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。そこで、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。

例えば、エージェントの移動可能範囲が上下左右の4方向の場合、各ローカル領域における隣接領域は上

下左右にあるローカル領域が該当し、それらのローカル領域と情報を交換する。エージェントが上下左右と斜め方向の8方向に移動可能であると定義した場合、上下左右方向および斜め方向にある隣接領域と情報交換を行う。

各ローカル領域では、隣接領域から得た環境情報の一部を保持できるように主記憶を確保する。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。図3は、エージェントの移動可能方向が上下左右の4方向に移動する場合のローカル領域と隣接領域の関係を示す。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ーラップさせる。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ーラップさせる。図3の斜線部が隣接領域間でオーバー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。オーバーラップさせる領域の情報は隣接領域から得る。図3網掛け部がオーバー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。

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

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。各ローカル領域におい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。これにより、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、先住民優先による衝突回避ルールを導入する。先住民優先による衝突回避ルールとは、「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」というルールである。

図4を用いて衝突回避ルールを説明する。図4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)である。ここで、領域 N に存在していたエージェントは、先住民として衝突が発生したセルを移動先とする。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

図4 先住民優先による衝突回避ルール

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。

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基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。本論文では、エージェントの視野を以下のように定義する。方向は上下左右の4方向とする。可視範囲は、エージェントに予め与えられる視力の値に基づいて決定される。例えば、視力4のエージェントは現在のセルから上下左右各方向4セル、合計16セルから次の移動先を決定する。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隣接領域と情報交換がなされる。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基づいて最大の資源を確保できるセルを探索する。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に並列 Sugarscape モデルに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付けが必要になる。MPI では

図5 並列 Sugarscape における動作フロー

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 値とを対応付ける。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 実験条件

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および計算時間短縮の効果を評価する。非並列シミュレーション結果との比較をすることで、隣接領域間での通信回数を意識して導入した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 3.0EG Hz Hyper Threading Technology (HTT) 搭載 Pentium 4 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、プロセスは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シミュレーションが例示されている。これらの社会システムの有効性はシミュレーションにおいて、エージェント数により評価されており、本論文でもエージェント数の変化を評価尺度として用いる。いずれの社会システムも Sugarscape モデルに基づいており、ある時刻の汚染物質や資源量はある時刻に存在するエージェントの内部状態の値に基づいて算出される。また、募金システムや事前くじシステムのアルゴリズムを導入したシミュレーションと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との違いは

図6 初期状態における資源量およびエージェント位置

図7 環境情報の分割方法（1次元分割）

図8 環境情報の分割方法（2次元分割）

エージェントの行動結果から得られる汚染物質量の抑制を表現しているか否かである。これは、図5 Step 5-1の箇所を変更するか否かの議論で済むことから、いずれの社会システムを利用しても提案手法の検証において影響を与えない。そこで、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い、シミュレーションへの影響および計算時間短縮の効果について検証する。

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 $(1 \times m)$ 、 $(n \times n)$ と表記する。

4.2 シミュレーション結果への影響

提案手法のシミュレーション結果を、図9から図13に示す。図9はエージェント数の推移の平均値と標準偏差を示したものである。図10は標準偏差の縦軸のスケールを変更したものである。図9および図10は非並列シミュレーション (1×1) と並列シミュレーション (1×4) 、 (1×12) 、 (3×3) 、 (4×4) の5種類について、それぞれ5試行の平均と標準偏差の結果である。

また、図11は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平均値を示したものである。図11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。図11の縦軸に示すRMSEは式(1)で表わされる。

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

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。なお、本論文で用いたRMSEは数値が小さいほど類似していることを意味する。

図12および図13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。色が濃くなるほど実行回数が多いことを示す。図12および図13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

図9 エージェント数の推移 (5 試行の平均と標準偏差)

図10 エージェント数の推移 (5 試行の標準偏差)

図11 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の平均

図12 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13 領域別エージェント意思決定処理実行回数 (2次元分割)

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。図9下部に示す標準偏差は平均値に対して2%程度の小さなものであり、特に大きな影響を与えていないといえる。標準偏差について縦軸のスケールを変化させた図10よりエージェント数推移の標準偏差は、(1×4)、(1×12)、(3×3)については、非並列シミュレーションとよく似た標準偏差の推移を示しているが、(4×4)については、偏差のピークがやや遅れて表れていることがわかる。偏差のピークの出現パターンについては、環境情報とローカル領域の境界位置との兼ね合いなどが考えられるが、今後検討する必要がある。

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11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11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%程度であることがわかる。

図11の隣接領域に移動したエージェント数 (Num. of Movers の値) より隣接領域を移動するエージェント数は並列プロセス数に必ずしも比例するとは限らない。環境情報の分割において、(1×4)および(4×4)はそれぞれ図7-Aおよび図8-Bに示すように特定のローカル領域に資源量の多い箇所が配置されている。結果、図12-Aおよび図13-Bから、エージェントの移動が特定の領域で留まったと説明できる。一方、(1×12)および(3×3)はそれぞれ図7-Bおよび図8-Aに示すように資源量の多い箇所を複数の隣接領域で分割している。図12-Bおよび図13-Aからエージェント意思決定処理実行回数は複数の領域で緩やかに分散してい

ることが伺えることから、隣接領域間を移動するエージェントが増加したといえる。図11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いえる。図11の隣接領域に移動したエージェント数、図12、図13より、環境情報の分割は初期条件に影響されることがいえる。また、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といえる。

4.3 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ス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5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。

表1 計算時間

		P	Time	Speedup	Efficiency
			$T(P)$	$S(P)$	$E(P)$
1D	1x1	1	2375388.00	1.00	-
	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

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

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

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。

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、高速化率が高い値を示し、特に(4×4)以外は高速化率がプロセス数を超えた。

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12および図13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が他のノードに影響を与えたため、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と考えられる。

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す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な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した。

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。

謝辞

本研究は文部科学省 私立大学学術研究高度化推進事業 社会連携研究推進事業(平成17年度～平成21年度)の助成を得て行われた。また、本研究の一部は文部科学省「人文学および社会科学における共同研究拠点の整備の推進事業」の委託を受けて行われた。

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.
- [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, 八横 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 Assistance 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.
- (2009年7月18日 受付)
(2010年2月16日 採録)
- [問い合わせ先]
〒564-8680 大阪府吹田市山手町3-3-35
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
蟻川 浩
TEL : 06-6368-1228
FAX : 06-6330-3304
E-mail : hiro-ari@pglab.org

著者紹介

ありかわ ひろし
蟻川 浩 [非会員]

1976年生。2006年9月 奈良先端科学技術大学院大学情報科学研究科修了。博士(工学)。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ポスト・ドクトラル・フェローを経て、現在、関西大学研究推進部 研究ネットワークアドミニストレータ。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研究員を兼務。2010年4月より奈良産業大学情報学部専任講師。広域並列分散処理システム構築技術、社会シミュレーション向け大規模エージェントシミュレーションソフトウェア実装技術の各研究、研究用計算機システムおよびネットワークシステムの運用業務に従事。情報処理学会、ACM各会員。

むらた ただひこ
村田 忠彦 [正会員]

1971年生。1997年3月大阪府立大学大学院工学研究科博士後期課程修了。2001年 関西大学総合情報学部准教授。2005年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長。2008年共同利用・共同研究拠点ソシオネットワーク戦略研究機構研究員。2009年同教授。現在に至る。博士(工学)。2010年～2011年シカゴ大学計算研究所客員教授。1997年システム制御情報学会奨励賞、2006年IEEE SMCS 最優秀技術委員会賞受賞。現在、本学会編集委員、同事業委員、人工知能学会進化計算フロンティア研究会幹事、IEEE SMCS 日本支部長、Review of Socionetwork Strategies 副編集長などを務める。

A Large-scale Multi Agent Simulation Considering Environmental Information and Its Implementation on Parallel Computer

by

Hiroshi ARIKAWA and Tadahiko MURATA

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.

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

Contact Address : **Hiroshi ARIKAWA**

Policy Grid Computing Laboratory, Kansai University

Yamate-cho 3-3-35, Suita-shi, Osaka-fu, 564-8680, JAPAN

TEL : 06-6368-1228

FAX : 06-6330-3304

E-mail : hiro-ari@pglab.org