

移動エージェントに基づく
分散アントコロニークラスタリングの
直線化法

東京理科大学
理工学研究科情報科学専攻
李志祥

平成 23 年 2 月 1 日

目次

第1章	序論	3
1.1	本研究の背景	4
1.2	本研究の概要	6
1.3	論文の構成	8
第2章	基礎知識	9
2.1	移動エージェントについて	9
2.1.1	エージェントと環境	9
2.1.2	移動エージェントの定義	11
2.1.3	移動エージェントの特徴	13
2.1.4	移動エージェントの状態	15
2.1.5	移動エージェントの実行環境	16
2.2	アリの群行動	18
2.2.1	フェロモンの基本概念	18
2.2.2	フェロモンコミュニケーション	18
2.2.3	アリの食料収集	19
2.2.4	マルチエージェントシステムとしてのアリ	20
2.3	アントコロニー最適化法	22
2.4	アントコロニークラスタリング	24
第3章	アプローチ	26
3.1	カートロボット	27
3.2	アントエージェント (AA)	27
3.3	フェロモンエージェント (PA)	28
3.4	システム全体の動作概要	31
第4章	実装と評価	32
4.1	実装	32
4.2	実験	34

4.3	結果	34
第 5 章	まとめ	41
5.1	結論	41
5.2	考察	41
5.2.1	問題点	41
5.3	おわりに	45
第 6 章	謝辞	46
付 録 A		49

第1章 序論

現在の科学技術が発達し、ロボットという存在が身近なものとなるだけでなく、小型化、高性能化など、ハードウェアの目覚ましい進歩が見られる現代において、ロボット制御システムも様々な研究が進められ、急速な進歩を遂げてきた。特にマルチエージェントに基づくロボット制御システムでは、ロボット自身が相互作用を行い、周囲の環境に適応するために学習することを可能とした。これに加え、モジュール方式、再構築可能性、拡張性の導入により、マルチロボットシステムの分散環境における制御システムの開発はより簡単なものとなった。

また、エージェントという概念が研究、応用されてきた。例えば、WWWブラウザで検索サイトに移動し、何か検索したい単語を打ち込めば、いろいろな情報が画面に表示される。ウェブに限らず、Eメールが届けば「新しいメールが届きました」などと表示されたり、ダイアルアップでインターネットに接続していて、一定時間以上何もしていないでいるとコンピュータが接続を維持するのかそれとも切断してもよいのかと問うメッセージを表示するなど。コンピュータの中では様々なエージェントが動いていて、必要な処理を行い、適宜ユーザに情報を示す。そして、それらのエージェントを「デスクトップエージェント」とか「インターフェースエージェント」と呼んでいる。非常に古典的で簡単なエージェントではあるが、このようなものもユーザと外部の世界との仲介をしている、エージェントである。

しかし、マルチエージェントシステムの発達によって、各エージェント間で過度の相互作用が行われ、マルチロボット環境において様々な問題を起こす可能性が現れた。例えば、各ロボットがエージェントによって制御され、ロボット間の相互作用が無線ネットワークで達成されるマルチロボットシステムを考えたとする。この場合、ロボットが移動する際に周囲の環境も変化するので、ロボット間のそれぞれの接続状態も変化する。この環境において、ひとたびネットワークにおけるいくつかの接続に障害が生じると、システムはロボットの状態の一貫性を維持するこ

とが困難になる．さらにこの問題は，相互作用の数に比例して増加する傾向があった．

以上のような過度のロボット同士の通信の問題を解消し，分散環境でロボット制御を行うために移動エージェント技術を用いた方法が開発された．移動エージェント技術とは，仮想的な主体によって人間の情報処理を代行させる技術の総称で，実行主体となるプログラム自身がネットワーク上を移動するという点で従来のプログラム制御と異なっている．この技術は，利用者のネットワーク上での情報処理の支援や肩代わりをする，インターネットエージェントや，共同作業を支援するエージェントグループウェアなど，様々な応用が試みられてきた．

移動エージェントシステムにおいて，各エージェントはあるコンピュータから別のコンピュータに移動することができる．移動エージェントは，自身が移動することによって必要な機能を与えるだけでなく，自律的にタスクを実行することができるので，他のコンピュータとの通信量を減少させることができる．例えば，通信量が最小となる場合，移動エージェントが移動するときだけに，コンピュータ間のネットワークが接続していればよい[1]．この特性は，信頼性の低いネットワークや，断続的な通信によって動作しなければならない状況のロボットを制御するときに役立つ．また，移動エージェントを用いれば，マルチロボットシステム上にあるアクセスが容易な任意のロボットを通して，システム外のホストコンピュータやコントローラから，全体のマルチエージェントシステムに新しい機能や情報を取り入れることも可能になる．

これらのことから，移動エージェントと群ロボットの組み合わせは，システム全体の動作を調整する新たな方法を提供し，群ロボットが物理的な移動を行う代わりに，移動エージェントがロボット間を移動することで，全体としてエネルギー消費を減少させることができた[2-3]．しかし，更なるロボット数の増大は管制管理を困難にし，新たなコントロールの方法が求められるようになった．

1.1 本研究の背景

神林と滝本ら[4]は，階層型移動エージェント（図1.1）を使用した群ロボットの制御を提案した．これは，移動するエージェント自身が階層構造となっているので，それらが稼働している間，制御ソフトウェアも階層的に組み立てることができ，ロボット制御ソフトウェアの構築をユー

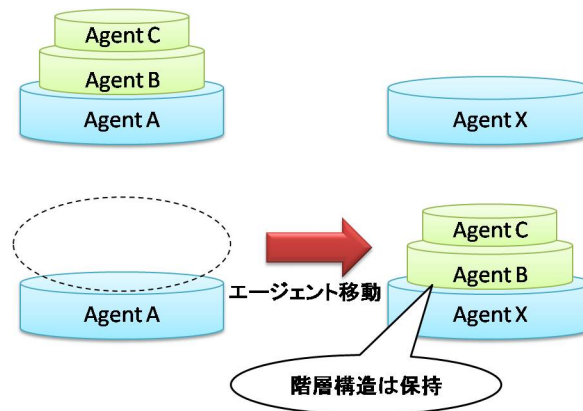


図 1.1: 階層型移動エージェント

ザはエージェントの移動によって行うことができる。つまり、エージェントの移動によって動的に制御ソフトウェアを拡張することで、一つずつ機能を追加することができるだけでなく、基礎制御ソフトウェアを比較的単純にすることができる。したがって、それらはロボットを始めから高機能にする必要がなく、ロボットを単独で学習させる必要もない。それらは、新しいエージェントとして機能を後で送ることができるのである。これによって、移動エージェント技術の利用は群ロボットの省エネルギー化に寄与することが実証された。一方、ロボット制御における別の手法として、生命複雑系や群知能^③を用いたアプローチがある。Deneubourg は、生物学におけるアリの軍団を、集合と、巣へ帰る振る舞いの分類でシミュレートするアルゴリズムを公式化した [5]。さらに、フェロモンに基づくアリの行動を実際の問題解決に応用する試みがある。Dorigo らが提案した、アントコロニー最適化法 (Ant Colony Optimization, ACO) である [6]。この試みでは、いくつかの組み合わせ最適化問題や巡回セールスマン問題で有用性を示している。また、Wang と Zhang は、群ロボットのオブジェクトを分類する研究の延長で、アリの群行動からヒントを得たアプローチを提案した [7]。Lumer は Deneubourg のモデルを改善し、アントコロニークラスタリングと呼ばれる新しいシミュレーションモデルを提案し、あるオブジェクトを同種のクラスタに分けることを可能とした [8]。このアントコロニークラスタリング (Ant Colony Clustering, ACC) を用いたシミュレーションでは、人工アリが実際のアリを模倣し、徐々にいくつかのクラスタを形成する。ACC の応用例の一つにカートロボット

がある．空港のターミナルを考えてもらいたい．利用者が置いたままにしたカートを集めるのに苦労する係員を見たことがあるだろう．もし係員が集める前に，カートがある程度集められていれば，はるかに楽になる．この場合，各カートの集合場所をあらかじめ設定しておき，知的なカート（知能ロボット）が自動的に戻るという簡単な実装方法がある．しかし，いくつかのカートは他の集合地点に近くとも，設定された場所への長距離移動を強いられることとなり，強力なバッテリーが必要となるなど，省エネルギーの観点から重要な問題となる．これらの問題を解決するため，フィールド上に散在したロボットの位置を移動エージェントで把握し，アントコロニー最適化法（ACO）を基にしたクラスタリングアルゴリズムであるアントコロニークラスタリング（ACC）を用いて集合場所を決定し，自律的に行動を決定する方法がある [9]．しかしこの方法は，カートの位置を管理し ACC を実行するホストが必要となるため，本来 ACC が持っている分散システムとしての優位点が制限されていた．

1.2 本研究の概要

本論文では，Mizutani ら [10] が提案した移動エージェントに基づく分散アントコロニークラスタリング（Ant colony clustering using mobile agents as ants and pheromone）を引き継ぎ，ネットワークで接続された群ロボットを制御する新たなアプローチを示す．移動エージェントと群ロボットの組合せは，群ロボットが物理的な移動を行う代わりに，移動エージェントがロボット間を移動することで，全体としてエネルギー消費を減少させることができる，新たな方法を提供した．

しかし，規模が大きく複雑なロボットシステムに対して，どのように移動エージェントとロボットを組み合わせるかが課題となっていた．これらの問題を解決するために，蟻の群行動におけるフェロモンコミュニケーションをモデル化したクラスタリングアルゴリズムであるアントコロニークラスタリング（ACC）を参考に，新たな制御手法を提案する．本研究では，フィールド上に散在したカートロボットに対し，移動エージェントを用いて集合場所へ一直線に集合させることを考える．ここで，蟻に対応する移動エージェント（アントエージェント，AA）とフェロモンの効果を提供する移動エージェント（フェロモンエージェント，PA）の 2 種類の移動エージェントを用いて，カートロボットを制御する．カートロボット上にこれら 2 つの移動エージェントを横断させ，AA がカート

ロボットをフェロモン濃度の濃い場所へ移動させることによって、集合場所を自律的に決定する。これによって、ロボット制御のホストを必要とせずに、準最適な場所へカートを集合させることができ、集合位置を計算するための特別な装置も必要としない、分散化した Stigmergy(場との相互作用) に基づくロボット制御が実現できる。本手法の有効性を証明するためにシミュレータを作成し実験を行った。その結果、多くのロボットから小数かつ線形的なクラスタを生成できることを確認した。

1.3 論文の構成

本論文における，第2章以降の構成を説明する．

- 第2章では本研究の基礎知識を説明する．
- 第3章では本研究の提案手法について説明する．
- 第4章では提案手法の実装について説明し，その結果と評価について述べる
- 第5章まとめ

第2章 基礎知識



本章では今回の研究を理解するうえでの基礎知識について解説する．

2.1 移動エージェントについて

本節では，移動エージェント技術に関する説明を行う．

2.1.1 エージェントと環境

まず，エージェントという言葉聞いたとき，どのようなものを想像するだろうか．「代理人」という訳から，人間から依頼されたタスクを実行する文字通りの代理人や，人間の代理をする知的なソフトウェアというイメージがなされるだろう．この様々な意味がエージェントという概念を広く，分かりにくいものにしてきた．

分散オブジェクト技術の業界標準である CORBA(Common Object Request Broker Architecture) を取りまとめている OMG(Object Management Group) という団体がある．OMG は IBM，Sun などのハードウェアメーカ，ベンダなどが集まって結集した団体であるが，この OMG が MASIF(Mobile Agent System Interoperability Facility) という，エージェントなどの共通ファシリティに関しての案をまとめている．MASIF によると「エージェントとは人や組織のために自律的に活動するコンピュータプログラムである」と規定している．ただ，コンピュータプログラムにとらわれずに，エージェントというものに統一的な概念を導入するならば，「環境の状態を感知し，行動を行うことによって，環境に対して影響を与えることができる自律的主体である」とすることができるだろう [11]．

ここにおける主体という言葉は，人間である場合やロボットやソフトウェアなど，それ自体で 1 個体をなすものを指す．自律的とは，自身の経験とそれが働く環境に組み込まれた知識の両方に基づいて行動できる

ことを指す。つまりエージェントとは、図 2.1 であらわすように、自らの知覚と行動を介して、環境と相互作用する自律的で知的な主体なのである。図 2.1: 環境とエージェントでは環境とはいったいどのようなものだろうか。基本的にはエージェントが環境を知覚し意思決定を行う部分があるとする、環境とはそれ以外をさすことになる。環境に対する重要な性質を以下に挙げる。

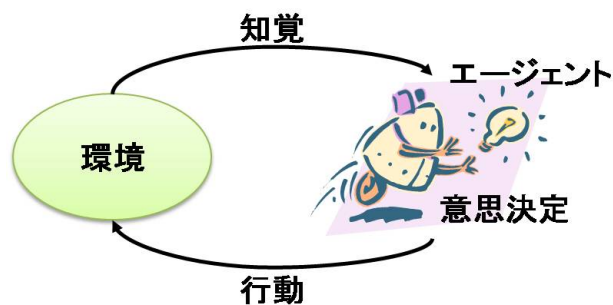


図 2.1: 環境とエージェント

アクセス可能と不可能

エージェントが環境の最新の状態を完全に、かつ正確に知覚できるとき、その環境はそのエージェントにとって、アクセス可能 (accessible) であると呼ぶ。一方、環境から知覚できる情報が限定されたり、不正確だったり、遅れをとまったりする場合、アクセス不可能 (inaccessible) であると呼ぶ。

決定的と非決定的

ある環境の状態で、エージェントが同じ行動を起こすとそれに応じた環境の変化が一意に定まる場合、その環境を決定的 (deterministic) であると呼ぶ。自らの行動に対して確率的に次の状態が定まるなどの場合、その環境は非決定的 (non-deterministic) であると呼ばれる。一般にエージェントが複数存在し、各エージェントが自律的に行動を起こす場合、その環境は非決定的となる。

エピソード的と非エピソード的

エージェントの経験がエピソードと呼ばれる単位に分割できる場合、その環境をエピソード的 (episodic) と呼ぶ。エピソードとはエージェントの知覚と行動の列からなり、試行と同様の意味で用いることもある。エピソード的なとき、それぞれのエピソードは独立であるから、エピソードに分割できない、非エピソード的 (non-episodic) な環境より単純な環境となる。

静的と動的

エージェントの行動以外によって環境が変化する場合、その環境は動的 (dynamic) であると呼ぶ。逆に、エージェントの行動の結果だけでしか環境が変化しない場合、その環境を静的 (static) であると呼ぶ。静的な環境では、意思決定に時間がかかっても、環境は変化しないため、その間の時間経過を考慮する必要がなく、非常に単純になる。しかし、エージェントが複数存在する場合は、環境が他のエージェントによって変化する可能性もあるため、静的な環境であっても見かけ上、動的な環境の様な変化を生じる場合がある。

離散的と連続的

エージェントの知覚と行動を有限に区別できるとき、その環境は離散的 (discrete) であると呼び、区別できないとき、その環境を連続的 (continuous) であると呼ぶ。

2.1.2 移動エージェントの定義

移動エージェントとは、ソフトウェアエージェント技術のひとつで、ネットワーク上のコンピュータ間を移動しながら計算処理を継続できる自律的で能動的なプログラムであり、非同期処理の実現や、通信コストの削減などの特徴を持つ。

これまで、プログラミングパラダイムは主にアセンブラ言語、手続き型言語、オブジェクト指向言語という流れで変遷してきた。アセンブラ言語は計算機のアーキテクチャを強く意識して計算機の挙動を記述するものであったが、手続き型言語では処理を主体にプログラムが書けるよ

うになり、オブジェクト指向言語では計算対象 (オブジェクト) を主体とする記述が可能となった。

これは、手続き型言語においては計算機のアーキテクチャを隠蔽し、オブジェクト指向言語においてはデータ抽象の考え方によってデータ実装の詳細を隠蔽した抽象化の歴史とみることができる。この抽象化は、計算の世界と現実の世界を近づける方向で続けられてきた。オブジェクトの次に位置づけられるエージェントもこの抽象化の延長上にあると考えられる。さらなる抽象化によって、計算の世界と現実の世界をより近づけるのがエージェント指向パラダイムのねらいである。

移動エージェントは、プログラム本体だけではなく、プログラムの実行状態も保存したまま移動し、移動前の実行状態から処理を再開できる。そのため、移動エージェントは、非同期なネットワークプログラミングの実現や、次世代的な分散処理の実現、通信コストの削減、通信障害時における柔軟な対応などの利点がある。[12, 13]

エージェントの定義は様々だが、主に以下が挙げられる。これら全ての特徴を持つ必要はなく、このうちいくつかを持てば、エージェントであるといえる。

- 自律性 (Autonomy)
自主的にアクションを実行できる
- 反応性 (Reactivity)
環境の変化に対してアクションを実行できる
- 自発性 (Pro-activity)
コンテキストや意図を考慮して適切なアクションを実行できる
- 学習 (Learning capability)
知識・経験に基づいた処理ができる
- 高レベル通信 (High-level communication)
専用または自然言語によるエージェント間または対人間通信ができる。
- 協調性 (Cooperation)
他のエージェントまたは人間と協調して目標を達成する能力をもつ
- 個性 (Character)
他のエージェントと区別する名前や特徴、目的をもつ

- 移動性 (Mobility)
ネットワークを介して他のコンピュータを移動できる

2.1.3 移動エージェントの特徴

まず、従来のサーバ・クライアント型のアプリケーションと、移動エージェントの通信方式の違いを図で示す。

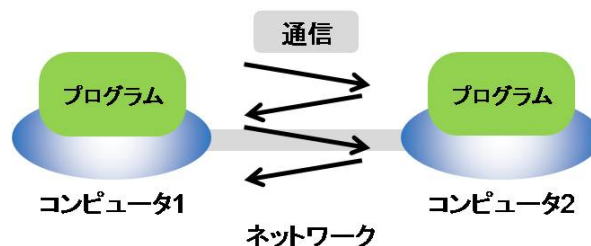


図 2.2: サーバ・クライアント型の通信形態

図 2.2 で示すように、サーバ・クライアント型のアプリケーションでは、コンピュータ 1 上のプログラムと、コンピュータ 2 上のプログラムとで、通信を繰り返しながら処理を進めていく。サーバ・クライアント型であれば、サーバでデータを一元管理できるという利点がある。しかし、サーバ・クライアント型のモデルにも欠点があり、例えば次のようなものが挙げられるだろう。

- クライアント側にアプリケーションがインストールされているため、アップデート等が容易ではない
- サーバ側に障害が発生すると、クライアントはシステムを実行できない
- 通信障害に弱い
- 通信コストが高い

一方、移動エージェントシステムの動作を図 2.3 で示す。移動エージェントではコンピュータ 1 上で動作するプログラムが、ネットワークを介して、コンピュータ 2 上に移動し、直接コンピュータ 2 上で動作することが可能である。

移動エージェントの大きな利点のひとつに、ネットワークバンド幅の有効活用が挙げられる。移動エージェントは移動した先の資源にローカルにアクセスできるので、大量のデータに対して高速にローカルアクセスを行ったあと、得られた結果だけを移動して持ち帰るといった動作が可能である。また、必要に応じて移動先のマシン環境の情報を活用したり、負荷の小さいマシンに移動して負荷分散を行うことも可能である。無線アクセスのような通信が切れやすい環境においては、通信が切断されても処理を継続できることから、移動エージェントが特に有効と言えるだろう。

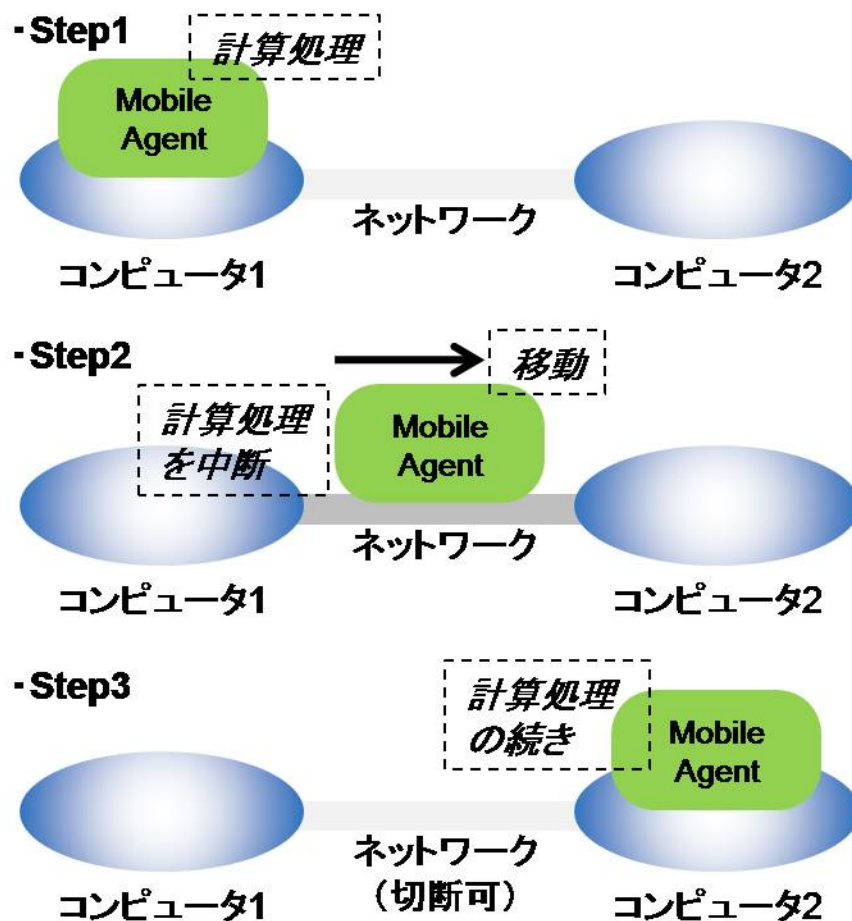


図 2.3: 移動エージェントの通信形態



以下に移動エージェントの利点を挙げる。

通信コストの削減

サーバ・クライアント型アプリケーションのように，コンピュータ間のネットワークを常時接続して作業を行う場合，通信コストが高くなるのは明白である．移動エージェントを用いた場合では，エージェントの移動後，そのネットワークが再び必要となるまで，ネットワークを切断しておくことが可能である．これによって通信コストを削減することが可能となる．

ネットワーク負荷の削減

サーバ・クライアント型と異なり，移動エージェントはプログラム自体を移動させる．すなわちエージェントを移動させるときだけ，ネットワークに接続し利用すればよく，ネットワークの負荷を低減させることができる．

非同期処理が可能

上記の利点と同様に，移動エージェントは，ネットワークには常時接続している必要がないため，非同期処理が可能となる．負荷分散の実現処理能力の高いマシンを選択し，そのマシンに移動して処理を行うことで，マシンに対する負荷の分散が可能である．

2.1.4 移動エージェントの状態

本節の冒頭で述べた MASIF によると，移動エージェントの状態として以下の二つのどちらか（もしくは両方）があると定義している．

1. エージェント実行状態
2. 移動先での実行を再開する際に必要な処理を確定する手助けとなる属性値

ここにおいて，エージェント実行状態とはプログラムカウンタの値やスタックフレームの内容，あるいは CPU 使用率や仮想マシンのレジスタ情報などのランタイム情報のことを指し，上記 1,2 においては，エージェントに関連する状態に関する定義を与えている．状態とは，プログラムで使用

されている変数や配列などに保持されているデータのことだと考えてよい。移動エージェントが「移動する」とは、移動エージェントの状態とエージェントを構成するプログラムコードを移動先のコンピュータシステムに転送することである。転送が完了すると、エージェントのプログラムコードをコンピュータにロードするとともに、移動直前の状態を復元させる。

Java 言語においては、プログラマが仮想マシンのレジスタや実行時のスタックの参照や変更を行うことはできない。従って Java では、移動時に保存できるデータは、インスタンス変数の値だけであり、プログラムカウンタやスタックの内容は保持できない。

2.1.5 移動エージェントの実行環境

ここまでで移動エージェントの利点や特徴を述べた。しかし、このような移動エージェントの実行には基盤となるシステムが必要であり、移動エージェント環境を一から作成するのは大変であるが、移動エージェントを構築するためのシステムは既にいくつか存在し、代表的なものには、General Magic 社が開発した Telescript 言語・日本 IBM による Aglets・東芝による picoPlangent などが挙げられる。我々の研究でロボット実機を利用する際は、佐藤一郎氏によって開発された移動エージェントシステム「AgentSpace」(図 2.4) や階層型移動エージェントをシステム「MobileSpace」を利用している。

AgentSpace や MobileSpace は Java 言語を利用した、分散計算用ミドルウェアであり、フリーのオープンソースプログラムである。このシステムには、移動エージェントの作成を可能にする API 群、及びエージェント実行や移動などの処理を行うランタイムシステムの 2 つの部分からなり、他の移動エージェントシステムと比較して、エージェントの移動が高速であること、エージェントが自己完備化された計算実体である、などの特徴がある [14]。この特徴を具体的に述べると、他のシステムでは実行状態を転送した後に、必要なクラスファイルを転送するという方式をとっており、一つのクラスファイルの転送に際して転送要求と受け取りという二回の通信を要する。この方法だとある通信の際に通信遅延が生じると全体としての転送時間の増大を招く恐れがある。一方 AgentSpace は、一つのエージェントを構成する全ての情報（実行状態やクラスファイル）などをひとつにまとめてからエージェントを転送する方法をとって

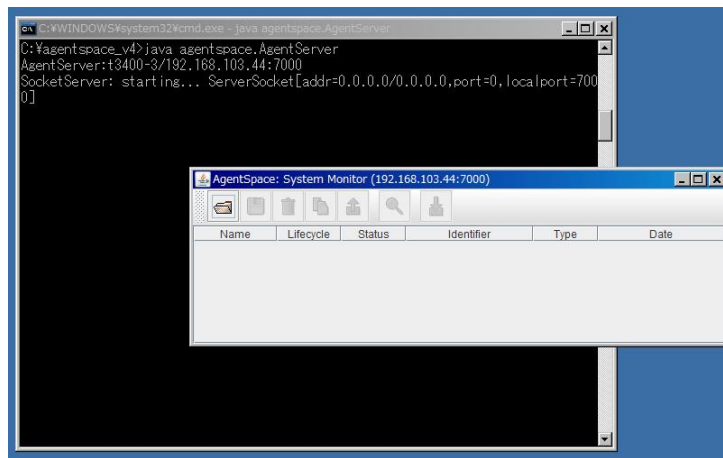


図 2.4: AgentSpace

いる。これにより通信メッセージ数が削減され、高速化が可能となっており、前述の通信遅延による全体の転送時間の増大という問題も避けることができる。このように一つにまとめてエージェントを転送する方法は、不安定な状態の通信システムにおいても非常に有効となる。AgentSpaceでは、ひとたび転送が完了すれば、エージェントを実行するのに必要な全ての情報が転送済みとなるので、通信環境の不安定さによるリスクも軽減できる。なお、本研究においてはシミュレータの作成に留まっているので、AgentSpace は用いていないが、今後、実機での実証実験を行う際には必要になると予想される。

2.2 アリの群行動

本節では、今回移動エージェントでのロボット制御に応用するアリの群行動について説明する。

2.2.1 フェロモンの基本概念

生態科学の分野において、フェロモンとは、「生物が体外に分泌し同種の個体間におけるコミュニケーションを媒介する化学物質である」と定めている。このフェロモンは、その種類も様々であり、例えばフェロモンを感知した生物を引き寄せる要素をもったフェロモンや、周りに危険を知らせる要素をもったフェロモンなどがある。このとき、フェロモンの「蒸発」と「拡散」という性質が大変重要となる。

「蒸発」は、古くなった情報が、時間経過とともに消えてゆくという性質である。この性質は、他個体に対して常に最新の情報を提供するという利点をもたらす。

「拡散」はフェロモンが発生した場所から周辺に対して広がり散る性質である。この性質により、フェロモンの発生源からある程度離れた場所に対しても、情報を提供できるという利点をもたらす。

2.2.2 フェロモンコミュニケーション

社会性昆虫であるアリは役割分担に基づいて、あたかも一個体として振る舞いながら、巨大な社会システムを見事に維持している。そこでは、自己利益のためではなく、自己を犠牲にしてまで群全体のために行動し、時には自分の命さえ投げ出して群を守る。

また、分散的・局所的な多数の要素に基づき、それぞれの判断に従って行動しているにも関わらず、マクロなレベルから見た場合、その振る舞いは目的に合った行動のように見える。

多くのアリは視力が弱く、触覚からの刺激に頼っている。したがって仲間同士のコミュニケーションは視覚的なものではなく、触覚から感じられる科学的な信号を用いることが知られている。~~この信号は一般的にフェロモンと呼ばれ、~~アリなどの社会性昆虫はこのフェロモンによって、言葉や視覚のかわりにコミュニケーションを行っている。アリのコロニー

では、数十種類ものフェロモンを用いて非常に多くの成員からなる群をコントロールしている。

アリのコロニーでは、各個体が自分の直面する状況に合わせて分散的にフェロモンを放出する。そして、そのフェロモンが科学的な特性やアリ同士の接触などでコロニーを取り巻く環境に、時間的・空間的に分散していき、複数の個体が出すフェロモンが相互作用することで、分散的な情報が統合される。各アリはその統合情報としてのフェロモンに反応することで、各個体があたかも全体の情報を知っていて、コロニーにとって最適な行動をとっているかのような行動を見せる。

2.2.3 アリの食料収集

食料収集は、働きアリの仕事の中で最も重要な仕事の一つである。食料の探索や運搬などに必要な労働力は働きアリが携わる仕事の中で一番大きく、できるだけ余分な労働力をかけずに効率よく食料を集めるということは、コロニーを維持する上でとても重要となる。

一般的にアリの食料収集は探索アリと収集アリの協力によって行われる。探索アリは食料を発見すると、道標フェロモンを散布することで、帰り道をマーキングしながら巣へ戻る。探索アリが帰巢した後、収集アリは触覚からの感覚に頼りながら、道標フェロモンに従って進んでいく。食料を発見すると、すぐにそれを取り上げて、真っ直ぐに帰巢する。このとき収集アリもフェロモンを散布しながら帰る。これらの行動は一見単純であり、個々のアリが従うルールも単純であるのにも関わらず、そのコロニーに属する複数のアリ達がこのルールに従うとき、まさに群知能と呼ぶべき振る舞いが見られる。これが、アリの最短経路行進である。

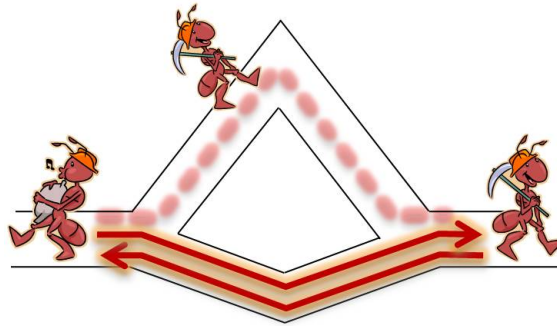


図 2.5: 分岐点でのアリの行進

では、分岐点があった場合アリはどのように行動するのだろうか．最初に分岐点に達したアリは、経路をランダムに選択する．後続のアリもフェロモンによってマーキングされている経路を選択する確率が高くなる．しかし、アリの行動速度は一定なので、短い経路を選択したアリは、長い経路を選択したアリよりも先に目的地に到着する．よって、長い経路のフェロモンの方が早く蒸発し、その経路は自動的に消滅し、フェロモン濃度の濃い短い経路のフェロモンは時間とともにさらに強化される．結果、一つの経路を集中的に選択することとなり、その経路が最適である可能性が高い．図 2.5 にそのイメージを示す．

この短い経路がわずかに早くマーキングされる現象と、経路選択のポジティブフィードバックがアリの経路選択のメカニズムであり、アリ達はどちらの経路が短い意識することなく、群全体の行動を最適化することを可能としている．

2.2.4 マルチエージェントシステムとしてのアリ

これまで述べたように、アリの群行動の中では全体を統率するようなスーパーバイザは存在しない．強いて言えば女王アリが中心的存在となるが、女王アリは働きアリの制御をしているわけではなく、複数の役割を持った個体たちが、各自で知覚する情報に基づいて局所的に行動する．それに関わらず、全体としては恒常的にシステム全体の状態を、ある一定のクオリティに保っているとみなすことができる．

つまり、各個体は自分の局所的な行動ルールに基づく行動を行い、フェロモンという物質を用いて環境を変化させて、間接的に過去の探索の成果

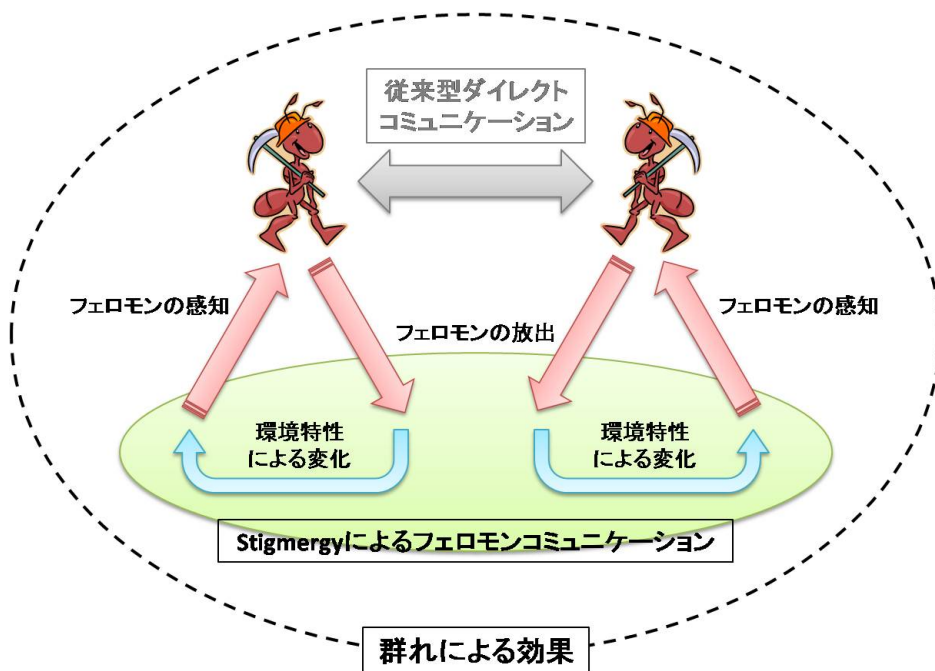


図 2.6: マルチエージェントとしてのアリの模式図

を伝達している．このような場との相互作用による通信の概念を動物行動学では Stigmergy と呼ぶ．これは，Stigma (Sign の意) と Ergon (Work の意) を合わせた言葉であり、「行動 (Work) から次の行動の合図 (Sign) を得る」という意味である．つまり，各個体間の協調行動を調整する (ローカル変数を介した) 間接的なコミュニケーションであり，アリにおいては，自分が知覚した局所的な情報からフェロモンを発信し，さらにそれをアリが再び知覚することで，アリ 環境 アリ...といったフィードバックが生まれ，複雑な環境の変化に柔軟に適應することができる．図 2.6 はこの，マルチエージェントとしてのアリの模式図である．

アリのコロニーの様な特徴と柔軟性は，人工的なシステム設計においても取り上げられることが多く，このようなシステムはマルチエージェントシステムと呼ばれ，人工知能や人工生命，ロボット工学だけでなく，様々なコンピュータサイエンスの分野で盛んに研究がなされている．

2.3 アントコロニー最適化法

前節で述べたようなアリの経路探索をはじめとする、フェロモンコミュニケーションモデルはマルチエージェントシステムのコミュニケーションモデルとしてモデル化が容易であり、いくつもの研究がなされている。特に実用的なのが、巡回セールスマン問題の解法や、ネットワークのルーティング問題などへの応用に提案された、アントコロニー最適化法 (Ant Colony Optimization, ACO) である [6]。このアントコロニー最適化法はメタヒューリスティック (特定の計算問題に依存せず、近似的最適解を得ることができる方法) であり、二つの重要な概念を導入することで解決を図っている。

1 つ目は、人工アリエージェントの確率的行動選択である。人工アリの行動を確率的に決定することによって、同じ情報を利用しながらも、各エージェントの行動選択に幅を持たせ、エージェントが同じ動作に陥ってしまわないように工夫されている。つまり、エージェントの行動を確率的に制御することによって、どちらの選択がよいかわからないときは広く探索を行い、どちらかの選択のほうがよいと分かるときは集中的に探索を行うのだ。

2 つ目は、フェロモンの蒸発である。これは現実のフェロモンを模倣したものであり、各エージェントが発信するフェロモンの情報の効力を時間とともに減少させる。これにより、新しい情報を速やかに利用しながら、探索領域にスタックされるのを防止し、ある限定的な状況で探索した情報に基づくフェロモンの情報は、後に探索領域が遷移した状況ですでに役立たなくなるという状況を防ぐ。このような古い情報は ACO ではフェロモンの蒸発の概念によって、自然に淘汰されるのである。ACO のアルゴリズムは対象問題によっていくつかバリエーションがあるが、概念的には共通であるため、以下に巡回セールスマン問題への応用のアルゴリズム例を示す。なお、この ACO の詳細や最新の研究は Ant Colony Optimization Home Page [15] で公開されている。

ACO アルゴリズムの例

複数の人工アリエージェントが，都市間の距離及びフェロモン値を参照しながら都市を移動し，それぞれのエージェントが全ての都市を巡回することで，エージェントの数だけ巡回路が生成され，それぞれの巡回路長に応じて，通過した巡回路の各経路上に，フェロモンが置かれる．なお各経路のフェロモンは時間とともに蒸発する．このとき，都市数を n ，人工アリエージェント数を m ，都市 i, j 間のフェロモン量を $\tau(i, j)$ ，初期時刻におけるフェロモン量を τ_0 ，繰り返し回数 t ，終了ステップを t_{\max} ，とすると，

1. $t := 0$ とし，すべての経路のフェロモンの値を τ_0 で初期化
2. 全ての人工アリエージェントを初期都市に配置
3. 各アリは巡回路が完成するまで都市の選択を繰り返す
4. 各アリの巡回路と巡回路の総距離に基づいて，各経路のフェロモン情報 $\tau(i, j)$ を更新
5. $t := t + 1$ とし， t が t_{\max} に達していなければ2へ
6. 得られた結果のうち最良の巡回路を出力し終了

この ACO はクラスタリングやソーティングにも応用されており，次でアントコロニークラスタリングについて説明する．

2.4 アントコロニークラスタリング

アリが巣の中で行う行動にアブラムシの畜産や幼虫の飼育がある．アリの巣の中の牧場や保育所を覗いてみると，家畜や幼虫が大きさなどによって空間的に整理され，配置されているのが分かる．

このような生態はエサを与える作業を効率化するように進化したと考えられている．このような行動は，エージェントが物体を，持ち上げる（下ろす）確率， P_{pick} (P_{drop}) として次の式で実現できる．

$$P_{\text{pick}} = (1 - \chi) \cdot \left(\frac{k_{\text{pick}}}{k_{\text{pick}} + f(i)} \right)^2 \quad (2.1)$$

$$P_{\text{drop}} = \chi \cdot \left(\frac{k_{\text{drop}}}{k_{\text{drop}} + f(i)} \right)^2 \quad (2.2)$$

ここで， $f(i)$ は周囲にある i の類似物体の密度である．より厳密には周囲にある類似物体の数が増えるほど減少する関数として，次のように定義される．

$$f(i) = \begin{cases} \frac{\sum_{j \in N(i)} d(i,j)}{|N(i)|} & (N(i) \neq \phi) \\ 1 & (N(i) = \phi) \end{cases} \quad (2.3)$$

$d(i, j)$ は物体 i と j の類似度（特徴空間での距離）であり， $N(i)$ は物体 i の近傍の集合を表す．なお， $d(i, j)$ は $0 \leq d(i, j) \leq 1$ となるように正規化されている． $d(i, j) = 0$ となる時は 2 つの物体が完全に一致したときである．

k_{pick} と k_{drop} はそれぞれ，持ち上げる（下ろす）行動の閾値を与えるパラメータである． χ は，アリのムーア近傍（アリを中心として，周囲 8 方向の隣接セル）における，物体の数 n によって決まる反応係数である．

$$\chi = \frac{n^2}{n^2 + k_{\text{crowd}}^2} \quad (2.4)$$


ここで， k_{crowd} は閾値であり， $n = k_{\text{crowd}}$ のとき， $\chi = 1/2$ となる．周りに物体が多いほど χ は 1 に近づき，下ろす行動に偏る．逆に少ないほど持ち上げる行動をしやすくなる．

式 2.1 と式 2.2 は，平面内でランダムに行動するアリが環境変数に応じて物体を移動するモデルとなっている．つまりアリが物体を上げる（下ろす）確率は，周囲の類似物体の密度によって決まる．これによって，アリは密度の薄い所で物を持ち上げ，濃い所で離すことを繰り返す．その結果類似した特徴のクラスタに分離する．これをアントコロニークラスタリング（Ant Colony Clustering，ACC）と呼ぶ．

第3章 アプローチ

本章では、移動エージェントを用いた新たなロボットコントロールへの取り組みを提案する。まず、本手法が想定する環境について説明する。本手法では、第1.1章本研究の背景において例として述べたような空港のターミナルを考える。広い空港を利用したことがある人ならば、乗客が使用したまま散乱したカートと、それらを集めてカート置場に戻す仕事をする係員を見たことがあるだろう。もし係員が集める前に、カート自体が自動的に列に集められていれば、その係員の仕事ははるかに楽になるだろう。この場合、各カートの集合場所をあらかじめ設定しておき、知的なカート(知能ロボット)が自動的に戻るという簡単な実装方法がある。

しかし、いくつかのカートは他の集合地点に近くとも、設定された場所への長距離移動を強いられる問題が起こる。カートロボットに搭載されているバッテリーを有効に使うには、移動距離をより少なくしつつ集合されることが望ましい。このとき、特定の集合地点に集めることは問題としない。それは、ある程度集合していれば、あとは人力で整理することは容易だからである。今回、この空港の様なフィールド上において、移動エージェントを用いてカートロボットを制御し、ACCを参考に各カートロボットがそれぞれの判断で集合場所を決定し集合するようにすることで、分散制御を実現する。

しかし、本研究でクラスタリングの対象となるオブジェクト(群ロボット)には特性による差異はなく、フィールドも有限線形空間であるために、従来のACCをそのまま用いることができない。そこで、ロボットを制御する移動エージェントだけではなく、フェロモンの効果を提供する移動エージェントも実装する。これにより人工アリは強くフェロモンの感じる場所への経路を高い確率で選択するようになり、フェロモン濃度が高い所にオブジェクトが集まりやすくなる。つまり、アリに対応する移動エージェント(アントエージェント, AA)とフェロモンの効果を提供する移動エージェント(フェロモンエージェント, PA)の2種類の移動エージェントをオブジェクトに対応するカート上に横断させ、PAのフェ

ロモンから得られる情報によって AA がカートを運転し，準最適な場所へカートを集合させる．

3.1 カートロボット

本手法を行うとき，カートはエージェントが移動可能な知的なカートでなくてはならない．カートにはカメラと方位センサが内蔵され，これらのセンサからの情報によって各種計算を行う．なおカートは以下の 3 状態である．

- 使用中（利用者，または他のエージェントにより使われている）
- ロック（集合場所にある）
- フリー（集合場所にない）

このカートロボットは，前進もしくは回転が可能で，アリに対応する移動エージェントである AA によって操作される．AA から操作されるとき，ランダムウォークまたはフェロモンウォークを行う．フェロモンウォークとはフェロモンの効果を提供する PA の指示によるもので，フェロモン濃度の濃い方へ移動する．

3.2 アントエージェント (AA)

AA は，ACC におけるアリに対応する移動エージェントであり，カートロボットに対し指示を出し運転を行う．AA は，すべてのカートの IP リストを持ち，このリストを基にして，図 3.1 のようにカートを横断する．AA がカートに到着した際行う動作を，表 3.1 に示す．

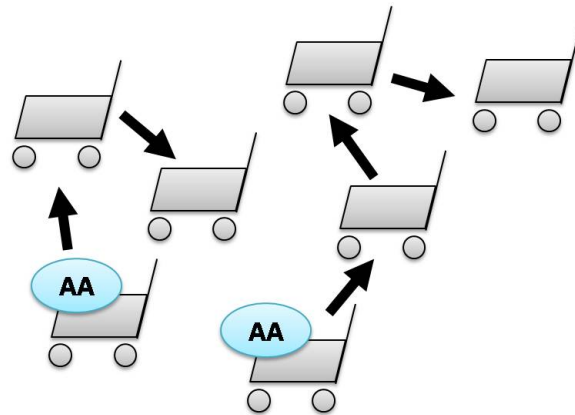


図 3.1: AA の移動

AA が訪れたカートが使用中であったとき，AA は何もせずに次のカートへ移動する．AA がフリー状態のカートを訪れたとき，そのカートに PA が存在した場合，これはアリがオブジェクトを見つけ拾い上げたことと対応し，PA に従ってカートを制御する．PA が存在しないときは，~~アリがオブジェクトを見つけていないことと対応し~~，ランダムウォークを行う．また，ロック状態のカートを訪れた場合は PA を生成し，次のカートへ移動する．ここで生成された PA によるフェロモンの振る舞いについては次節で示す．

カートの状態	AA の動作	
使用中	何もせず別のカートへ	
フリー	PA あり	フェロモンウォーク
	PA なし	ランダムウォーク
ロック	PA を生成し別のカートへ	

表 3.1: AA の動作

3.3 フェロモンエージェント (PA)

PA はフェロモンの効果を提供する移動エージェントであり，集合場所へのベクトル値を持つ．PA はロックされたカート，つまり集合場所にい

るカートから生成される．集合場所にいるカートの数によってフェロモンが強化され，クラスタリングの結果に影響を及ぼす．これはフェロモンの特性からであり，図 3.2 で示すように，フェロモンには以下の効果がある．

- オブジェクトの数によって比例増加する
- 距離によって比例減少する
- 有効範囲外に効果がない
- 有効期限が切れると消滅する

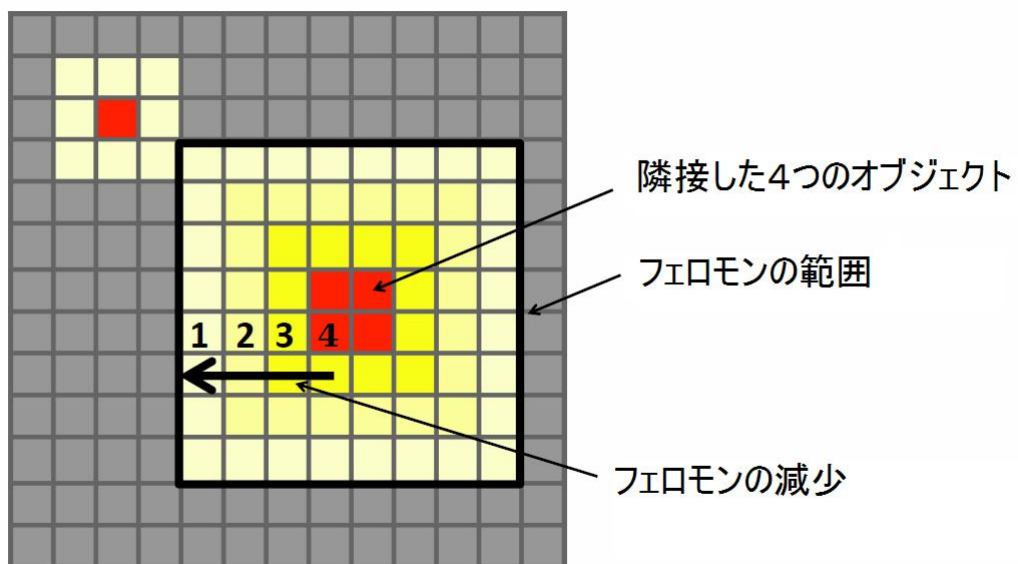


図 3.2: フェロモンの強さと有効範囲

また PA は自身の複製を，有効範囲内にあるカートへ転送することができる（図 3.3）．その際，図 3.4 であらわすように，集合場所へのベクトルと，転送したカートのベクトルとの合成を行うことで，集合場所の方向を更新する．

また範囲を考えるとベクトル v の絶対値，つまりフェロモンの濃度は式 3.1 のようにあらわされる．

$$|v| = \begin{cases} 0 & (\text{範囲外}) \\ \frac{C}{\text{distance}} & (\text{その他}) \end{cases} \quad (3.1)$$

ここで C は、任意の定数であり、フェロモンの特性によって変わる。アリがフェロモンの強い方へ導かれるように、複数の PA が到着した場合は、PA のもつベクトルの大きい方に更新される。また、有効期限の過ぎた PA は消滅し、常に新しい情報によって相互作用がなされる。

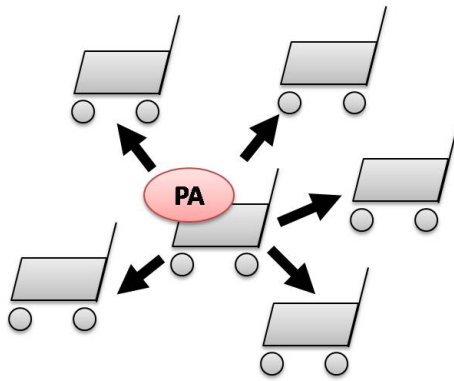


図 3.3: PA の移動

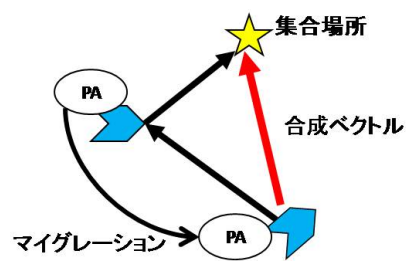


図 3.4: フェロモンのマイグレーション

3.4 システム全体の動作概要

ここまで述べた，提案手法をまとめる．図 3.5 に示すフローチャートが，今回提案するシステムの簡略図である．このような流れで，カートロボット上に移動エージェントを横断させることによって，自律的に集合地点へと移動させるのである．

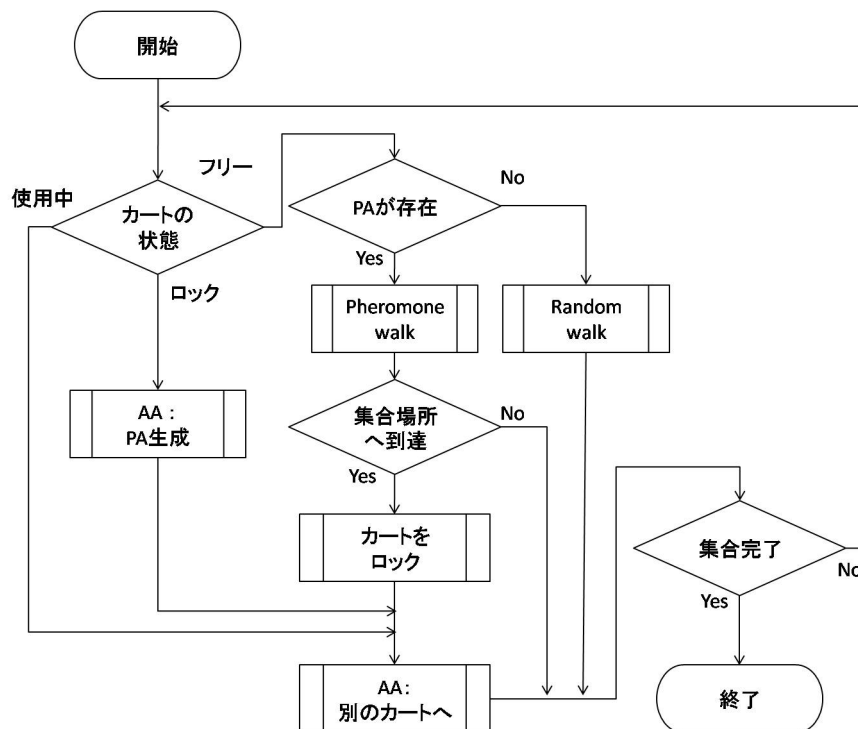


図 3.5: システム動作概要

第4章 実装と評価

本手法の有用性を確認するために，シミュレータを作成し結果を得た．本章では，実際に作成したシミュレータにおける実装の重要な部分とその実験結果について説明する．

4.1 実装

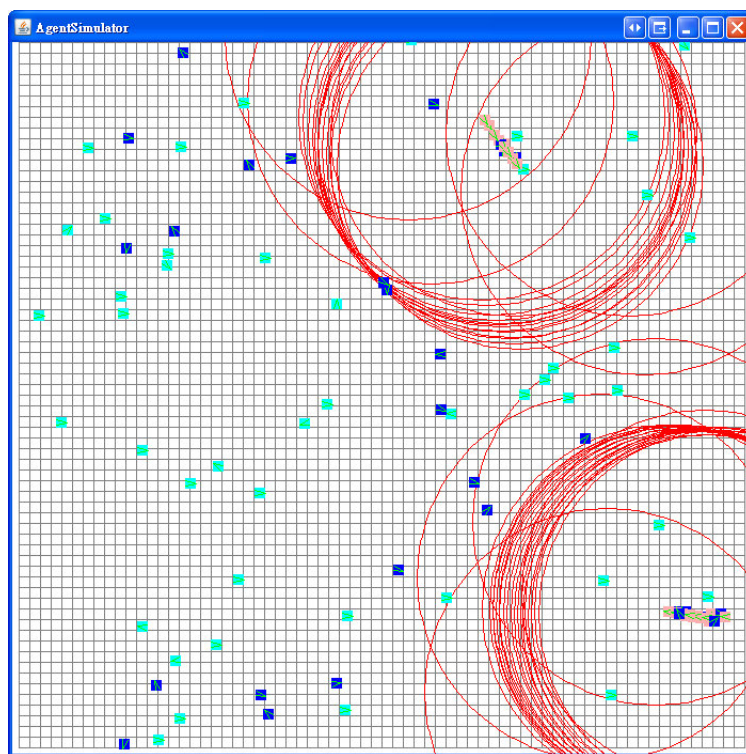


図 4.1: シミュレータ動作中の表示

本研究で実装したシミュレータについて述べる．シミュレータは Java を用いて作成し，各カート，各エージェントをオブジェクトとして実装した．フィールドは縦横 200 グリッドで定義した正方形フィールドを使用した．フィールドにはカートの初期位置をランダムに配置し，それらのカートに対し AA も同様にランダムに配置した．図 4.1 は今回作成したシミュレータの実行中の表示である．グリッドで表示されたフィールド上に色で塗りつぶされているのがカートであり，青色が AA によってコントロールされているカート，灰色が AA が訪れていないフリーなカート，濃灰色が集合地点であるクラスタの中心である．また赤色でフェロモンの有効範囲を表示している．

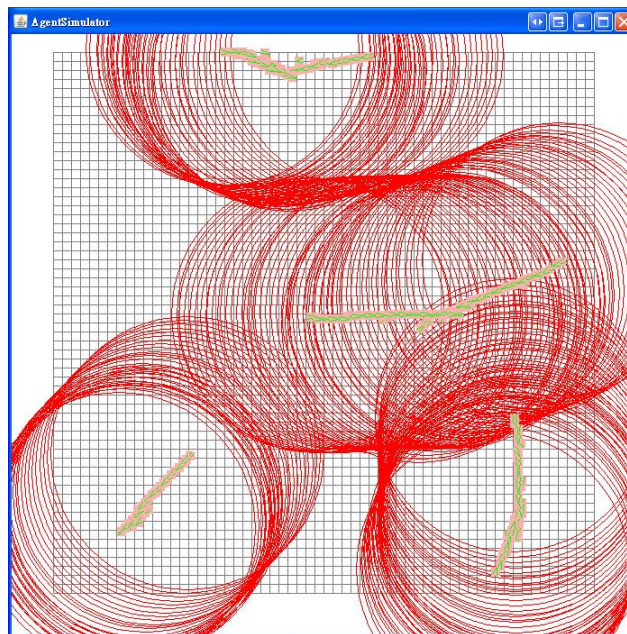


図 4.2: 終了条件を満たした際のシミュレータ

図 4.2 は終了条件を満足した際のシミュレータである．図 4.2 の場合だと，1 つの直線のクラスタに分かれていることが分かる．なおこれらシミュレータの内部の詳細な動作については，巻末にソースファイルを添付するのでそちらを参照していただきたい．

4.2 実験

実装及び実験は表 4.1 の環境で行った．

ワークステーション	Dell Vostro 400
CPU / Memory	Intel(R) Core 2 Duo E8200 (2.66GHz) / 2GB RAM
OS / 言語環境	Windows XP Professional SP3 / Java1.6

表 4.1: 実験環境

実験内容としては，カートの数 を 50, 100, 150, 200 の 4 パターン．また，Mizutani ら [10] の結論から (AA の数はほとんどクラスタ数に影響を与えていないことが分かる)，ここでは AA の数を 30 に固定し，シミュレータを実行し，まず正常に直線なクラスタに分けられるかの実験を行った．また各実験は 100 回ずつ試行し，終了条件を満たした際のクラスタ数，カート最大数，カート最小数，PA 数，カート平均移動距離，経過ステップ，などを調べ，平均値を用いて比較することとした．

4.3 結果

まず，シミュレータの実行に関しては問題なく実行され，結果を得ることができた．多数のカートを与えた場合でも，少数のクラスタに直線に集めることができた．

また，図 4.3，図 4.4，図 4.5，図 4.6 は上で述べた 4 パターン × 試行 100 回の結果を表したものである．Cart の数のそれぞれの場合に分け，実行した結果のクラスタの数，カート最大数，カート最小数，PA 数，カート平均移動距離，経過ステップなどを，試行回数 n について表にしている．またこの表より，試行回数 100 回の平均から，クラスタ数とカート最大数などを比較したグラフが図 4.7 である．また，平均移動距離と経過ステップについて比較したグラフが図 4.8 である．

n	クラス	カート最大数	カート最小数	FA数	カート平均移動距離	経過ステップ
1	2	28	22	1241	416	1405
2	2	36	14	1479	348	1287
3	4	18	8	667	479	1224
4	1	50	50	1926	1120	2843
5	3	37	2	1526	347	2367
6	2	40	10	1661	305	1047
7	1	50	50	2197	490	2084
8	1	50	50	2449	584	1423
9	1	50	50	2451	1001	3513
10	1	50	50	2451	905	4360
11	2	29	21	1254	217	1444
12	2	25	25	1226	664	1815
13	3	23	11	944	460	1379
14	2	45	5	1993	1003	2490
15	2	39	11	1630	381	1694
16	2	31	19	1308	296	1237
17	2	36	14	1479	275	689
18	2	27	23	1252	804	1976
19	2	32	18	1317	706	1496
20	2	33	17	1316	385	1458
21	3	20	14	1248	604	2043
22	1	50	50	2093	338	1410
23	1	50	50	2450	293	1135
24	1	50	50	2401	1229	3767
25	2	36	14	1444	209	603
26	2	42	8	1794	330	1797
27	1	50	50	2451	425	1918
28	1	50	50	2021	788	3684
29	2	28	22	1241	399	1146
30	3	22	12	1391	642	2752
31	1	50	50	2172	357	2253
32	3	23	13	872	366	1166
33	3	27	5	1061	286	1015
34	2	28	22	14449	179	739
35	3	26	12	953	288	866
36	3	25	9	1397	437	1294
37	2	29	21	1262	472	1888
38	1	50	50	2441	443	1476
39	1	50	50	2449	676	2132
40	2	34	16	1910	781	3478
41	1	50	50	2451	669	2041
42	1	50	50	2451	472	1725
43	2	26	24	1227	622	1679
44	2	29	21	1254	706	1595
45	4	15	7	653	364	851
46	3	21	12	863	383	1355
47	2	46	4	2075	415	1913
48	2	41	9	2049	354	1719
49	1	50	50	2451	325	1831
50	2	33	17	1362	359	1201
51	1	50	50	2159	636	2487
52	2	34	16	1397	441	1381
53	1	50	50	2321	678	2593
54	1	50	50	2074	907	2839
55	1	50	50	2395	511	1880
56	1	50	50	2451	421	2187
57	1	50	50	2451	522	4087
58	2	26	24	1227	432	2098
59	2	28	22	1247	625	2485
60	4	24	6	1281	249	815
61	3	34	5	1689	531	1345
62	1	50	50	2451	1284	4336
63	2	34	16	1379	440	1543
64	3	21	10	1658	510	1414
65	2	38	12	1551	345	1970
66	2	37	13	1526	332	1204
67	2	31	19	1304	291	733
68	3	22	11	873	306	1004
69	2	40	10	2425	346	1310
70	4	37	3	1502	264	1488
71	1	50	50	2431	411	1287
72	2	30	20	1281	419	1854
73	2	28	22	1241	442	1576
74	2	29	21	1254	284	1373
75	3	31	8	1139	184	824
76	3	29	3	1317	511	1271
77	2	35	15	1416	515	2068
78	1	50	50	2317	490	1439
79	1	50	50	2451	951	3757
80	2	25	25	2000	533	3377
81	2	43	7	1836	546	1493
82	2	35	15	1416	390	1571
83	3	25	7	983	279	794
84	2	36	14	1457	241	2054
85	2	46	4	1881	395	1623
86	2	37	13	1494	473	1344
87	1	50	50	2445	354	1967
88	2	28	22	1247	328	993
89	2	39	11	1604	379	1870
90	2	48	2	2426	582	3891
91	3	23	6	1231	533	1502
92	3	18	16	1111	348	1064
93	1	50	50	2438	228	1064
94	3	20	14	1443	336	1334
95	2	47	3	2451	577	2068
96	4	26	1	1229	259	1058
97	4	20	5	929	242	977
98	1	50	50	2451	742	4366
99	2	45	5	2002	501	1459
100	2	28	22	1247	219	859
n	クラス	カート最大数	カート最小数	FA数	カート平均移動距離	経過ステップ
average	2.01	36.57	23.8	1796.32	487.03	1810.39
max	4	50	50	12449	1284	4366

図 4.3: cart 50

n	クラス	カート最大数	カート最小数	FA数	カート平均移動距離	経過ステップ
1	4	35	11	2818	122	555
2	10	20	2	2237	175	741
3	5	55	1	4214	237	2784
4	2	81	19	5428	182	1060
5	6	40	1	3686	174	1123
6	4	39	9	3103	133	621
7	4	57	10	3887	141	919
8	8	44	2	3031	143	706
9	5	36	6	2865	107	605
10	4	75	3	4586	195	1112
11	3	63	14	4357	161	917
12	6	44	5	2846	126	597
13	3	56	14	4219	188	1178
14	4	46	2	3261	186	1127
15	4	40	14	2842	118	689
16	3	64	1	5372	269	1463
17	3	47	18	3741	190	957
18	7	21	6	1969	59	316
19	3	43	24	3472	189	1085
20	3	62	12	4362	245	1105
21	3	37	27	3358	258	1194
22	6	29	1	2460	119	558
23	4	60	3	4009	203	1532
24	4	38	15	2696	117	639
25	5	38	2	379	142	1079
26	5	37	1	2812	163	952
27	4	42	11	3319	129	718
28	3	41	25	3438	122	696
29	5	37	9	2438	154	770
30	6	54	3	3576	143	1068
31	3	39	23	3455	146	747
32	5	28	3	3610	122	764
33	3	63	7	5211	175	1076
34	7	29	1	1944	113	623
35	4	53	5	4405	133	782
36	4	43	6	3732	108	744
37	5	28	12	2531	98	783
38	4	50	12	3326	135	639
39	5	43	6	3400	193	910
40	6	31	4	3376	190	1067
41	4	31	19	2544	146	1036
42	3	47	25	3561	129	758
43	4	39	10	3340	87	422
44	5	40	9	2587	134	584
45	5	34	5	2706	155	673
46	5	35	1	2754	131	600
47	4	59	3	4234	168	929
48	4	38	11	2917	129	797
49	6	32	3	4070	166	938
50	4	42	10	3146	192	1036
51	4	39	16	2769	257	1271
52	3	44	24	3480	219	1167
53	4	55	9	4037	134	1027
54	4	51	3	5420	151	845
55	6	29	2	2891	111	526
56	5	38	2	2721	150	1161
57	4	45	2	3755	158	909
58	2	50	50	4951	197	1732
59	4	30	16	2598	109	609
60	3	48	9	3753	171	883
61	3	64	9	4843	243	1022
62	4	56	6	3965	171	1338
63	5	47	2	3263	130	654
64	4	35	12	3537	119	1015
65	6	24	11	2517	202	890
66	5	29	10	2266	198	1000
67	4	40	7	3374	177	1203
68	2	66	34	4352	202	1165
69	8	29	1	3123	212	1113
70	6	28	1	2275	117	641
71	6	26	1	2079	112	668
72	4	35	17	2789	131	756
73	5	53	1	3890	140	807
74	4	39	19	2745	127	675
75	4	36	7	3746	257	2623
76	7	31	3	2081	129	657
77	5	43	3	3305	137	778
78	4	52	7	3558	188	921
79	3	44	19	3600	259	2078
80	3	36	31	3392	172	915
81	4	52	11	3751	157	1065
82	3	43	19	3636	181	1055
83	3	44	27	3476	180	952
84	4	34	18	2963	132	627
85	4	29	16	2582	224	1611
86	6	29	7	2150	139	697
87	4	40	16	3024	139	673
88	7	46	3	2999	182	846
89	5	34	1	2917	131	707
90	2	68	32	5615	246	1319
91	2	85	15	4954	165	1127
92	4	34	16	2666	137	838
93	7	47	1	4130	137	895
94	6	42	4	3005	201	846
95	5	27	12	2187	97	460
96	5	36	4	2845	138	616
97	3	44	19	3652	160	793
98	3	55	12	4055	206	1062
99	4	61	8	4535	226	1294
100	8	31	1	2312	142	695
n	クラス	カート最大数	カート最小数	FA数	カート平均移動距離	経過ステップ
average	4.46	43.04	10.22	3358.49	161.4	938.81
max	10	85	50	5615	269	2784

図 4.4: cart 100

n	クラス	カート最大数	カート最小数	PA数	カート平均移動距離	経過ステップ
1	7	41	3	4450	91	708
2	6	42	15	4808	88	579
3	4	45	21	5966	92	853
4	5	46	15	5489	77	599
5	7	41	11	4752	80	561
6	4	54	21	6495	94	778
7	9	31	1	5756	108	1132
8	6	68	4	7263	120	1315
9	4	52	21	6135	107	786
10	6	59	7	6388	79	813
11	4	42	35	5611	108	787
12	5	44	17	7354	86	675
13	8	35	4	4222	87	681
14	7	36	2	5656	122	1001
15	8	41	1	5680	72	662
16	6	37	11	5017	104	815
17	8	34	3	7274	91	617
18	6	34	1	4836	96	792
19	8	43	2	6950	103	898
20	5	45	17	4914	94	660
21	6	43	1	5163	92	735
22	8	38	2	4933	109	899
23	8	29	10	3345	74	626
24	5	41	8	5704	134	1442
25	8	32	2	5439	93	808
26	7	50	4	5393	117	929
27	6	63	2	6405	92	826
28	6	67	5	8423	132	922
29	6	72	5	7152	103	760
30	7	36	5	5226	84	633
31	7	40	13	5278	112	711
32	3	71	29	8690	128	1229
33	6	53	11	6072	154	1162
34	7	41	2	4650	81	691
35	5	37	11	5008	79	607
36	7	37	2	5036	114	787
37	7	33	1	4441	88	815
38	8	45	5	5408	89	701
39	6	39	7	8506	148	1159
40	7	43	1	6328	119	871
41	10	31	2	5567	84	697
42	6	52	10	6231	109	828
43	9	44	1	4887	88	650
44	5	47	20	5073	84	587
45	5	80	2	9677	131	1046
46	10	39	1	6159	121	878
47	5	75	11	7190	108	738
48	5	49	12	5319	80	633
49	8	37	1	6105	83	575
50	8	30	2	4000	77	666
51	8	43	1	4479	88	595
52	9	33	2	5036	90	685
53	4	53	25	5823	102	746
54	8	33	9	4972	100	713
55	8	43	3	7613	123	816
56	10	45	1	4626	89	699
57	7	46	3	5222	119	951
58	4	80	12	7589	167	1315
59	10	32	1	4628	74	527
60	7	33	4	4039	90	735
61	5	51	10	5525	127	917
62	10	27	2	3378	75	603
63	6	51	3	5854	130	934
64	4	96	1	9678	134	1374
65	4	47	20	6256	93	807
66	9	46	1	5770	103	714
67	8	38	3	5836	110	780
68	8	37	1	4532	104	877
69	8	31	1	3772	70	536
70	5	45	5	6960	81	572
71	8	38	1	5132	84	629
72	6	35	20	4417	120	795
73	7	42	1	5067	129	1010
74	5	74	9	7664	126	1006
75	4	53	17	6390	123	1361
76	4	46	32	6044	117	916
77	10	38	2	5482	86	715
78	5	41	8	6730	96	1038
79	8	41	2	4820	155	1081
80	6	63	2	7281	114	791
81	5	62	8	5954	96	812
82	4	54	22	5984	94	910
83	5	58	7	10179	125	1003
84	9	46	2	5542	125	888
85	4	52	16	6336	177	1185
86	9	56	4	5235	72	548
87	13	42	1	4003	139	950
88	6	47	2	5941	111	849
89	5	55	15	5756	106	737
90	8	38	7	5825	116	944
91	5	49	15	5280	73	571
92	9	35	3	5764	93	678
93	5	54	16	5229	91	603
94	8	48	1	5397	95	744
95	5	62	12	6501	81	708
96	10	29	2	5027	76	565
97	6	45	2	5952	96	847
98	6	57	1	6042	124	1077
99	10	42	2	4950	99	766
100	7	47	2	5148	96	714
n	クラス	カート最大数	カート最小数	PA数	カート平均移動距離	経過ステップ
average	6.68	46.43	7.4	5784.14	103.1	816.4
max	13	96	35	10179	177	1442

図 4.5: cart 150

n	クラス	カート最大数	カート最小数	PA数	カート平均移動距離	経過ステップ
1	8	44	5	9730	76	769
2	9	63	1	9252	98	912
3	12	61	1	7918	81	778
4	10	46	1	6917	77	783
5	6	48	9	8777	69	710
6	9	60	2	8666	81	873
7	5	63	10	9439	100	937
8	8	66	6	10480	74	792
9	13	52	1	6280	89	789
10	9	46	1	8024	69	725
11	11	31	1	9136	76	755
12	9	62	3	9531	100	951
13	13	26	1	7919	73	755
14	8	59	5	7783	67	623
15	5	66	6	10327	88	971
16	9	44	1	7167	81	741
17	10	31	4	8852	61	621
18	11	40	2	7080	73	842
19	11	43	3	6495	76	783
20	9	41	2	7442	69	747
21	12	34	1	6224	66	678
22	10	43	4	8576	80	808
23	7	58	7	8202	96	901
24	13	41	2	7407	82	626
25	7	60	2	8841	87	862
26	8	53	6	9596	76	758
27	9	60	1	8718	81	762
28	8	43	11	8500	71	683
29	11	44	2	9899	80	827
30	7	61	12	8644	93	966
31	12	94	1	12141	92	856
32	8	71	9	9026	82	723
33	9	43	4	9159	89	869
34	13	43	1	6260	58	657
35	8	50	1	9715	80	759
36	9	49	3	7578	72	820
37	9	50	2	7217	60	698
38	6	66	1	9861	82	836
39	5	50	28	8440	73	787
40	11	46	6	8445	64	598
41	11	39	2	5715	75	769
42	12	38	2	7424	58	700
43	8	38	9	8530	61	687
44	9	48	7	6811	86	838
45	9	46	2	7070	87	801
46	9	51	5	8345	72	754
47	8	50	3	8325	66	729
48	8	50	1	11769	85	826
49	10	55	2	9086	71	745
50	11	59	1	8196	95	947
51	10	78	5	9852	93	951
52	8	56	1	10211	93	968
53	11	38	2	6549	79	860
54	9	38	8	7816	79	717
55	11	44	1	8374	75	786
56	12	55	5	8620	73	780
57	8	45	3	7317	76	746
58	6	53	7	11043	85	894
59	9	36	10	7474	66	635
60	7	44	16	10088	90	994
61	13	47	1	6197	72	750
62	10	42	1	8146	66	715
63	11	32	3	6248	76	694
64	9	43	11	7071	71	705
65	10	59	1	10765	89	924
66	12	37	2	7515	71	823
67	8	65	2	8708	92	954
68	8	63	3	10655	88	890
69	11	39	1	7903	67	726
70	7	78	2	11233	95	873
71	6	68	18	11018	64	664
72	9	39	2	9086	78	811
73	7	63	14	7894	76	759
74	11	33	5	5790	53	548
75	7	53	11	9196	74	688
76	10	47	2	7440	69	703
77	10	46	1	7473	66	708
78	5	68	15	9220	69	728
79	10	44	2	7615	86	929
80	12	42	2	7839	69	698
81	9	37	2	6793	86	809
82	11	50	2	8121	61	689
83	10	55	4	9188	80	867
84	8	56	2	10009	82	766
85	13	66	1	8358	81	759
86	10	61	3	8520	82	840
87	8	38	16	6475	66	727
88	14	30	2	6756	61	765
89	10	54	1	7203	88	795
90	12	41	2	9407	74	730
91	6	64	2	10026	90	985
92	9	37	3	9211	73	715
93	9	30	11	6590	54	663
94	6	44	22	9006	73	691
95	10	30	4	6694	60	665
96	9	47	7	9949	75	755
97	7	53	2	8667	67	622
98	10	60	1	8432	80	818
99	10	46	1	8614	74	748
100	8	57	1	8622	76	760
n	クラス	カート最大数	カート最小数	PA数	カート平均移動距離	経過ステップ
average	9.28	49.76	4.49	8435.12	76.81	782.67
max	14	94	28	12141	100	994

図 4.6: cart 200

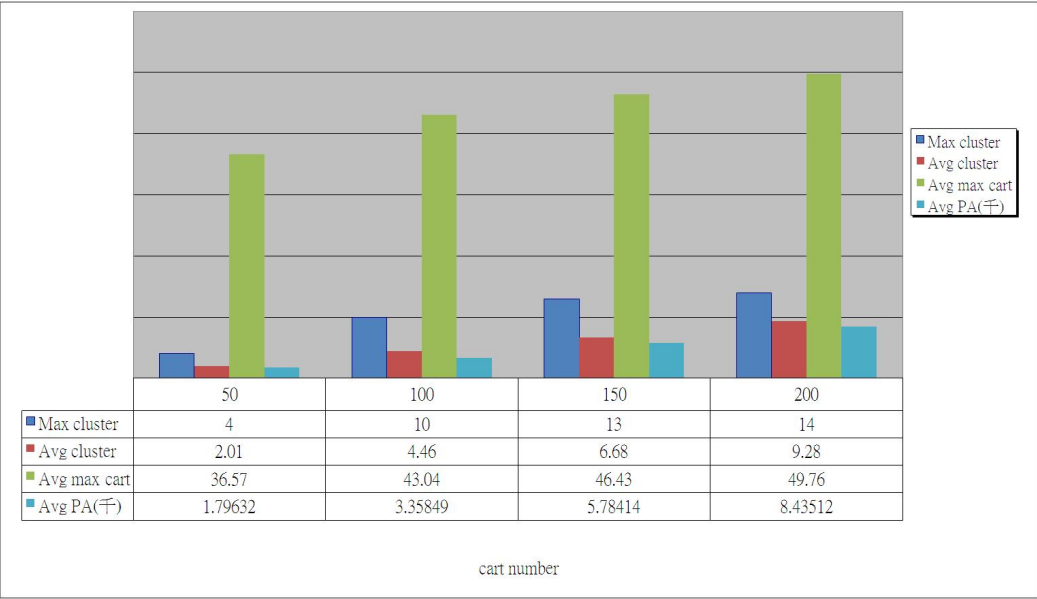


图 4.7: cart 別比較表

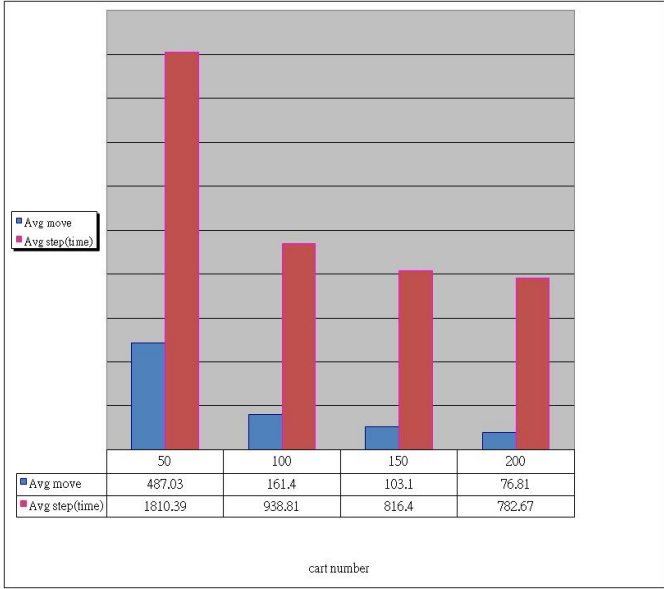



图 4.8: cart 別比較表

今回行った試行の結果を以上の図から詳細な分析を行う。

クラスタの数について

クラスタの数について比較を行うと、カート数を増やすとともにクラスタ数はほぼその比例で増えていく。要するにもっと多くのカートが置いてあるフィールド場だとカートの集合場所（クラスタ）が増えていく。これに対し、旅客の利用やスタッフの収集動作も便利になる。


カート最大数について

これは一つのクラスタの中で最大いくつかのカートがに集まることのできるということ。つまり、一つのクラスタで多ければ多いほどの~~カートが集めれば~~、空港のスタッフがますます手間を省くことができる。グラフによるとカート数を増やすとともに一つのクラスタの中のカートも増えていく。しかし、その増加率が対数関数的に徐々に減少していく。フィールドの大きさにも関係してると考えられる。

カート平均移動距離について

カート数を増やすとともにカートの移動距離が減少していく。つまり、フィールド対カート数の比例の飽和状態になってない限り、カートの移動距離が減り続ける。そうすると、カートのバッテリーの電力とカートのタイヤの消費を減少するため、カート数と空港の大きさがある程度の比例になると、そういった消費が減少できると考えられる。

経過ステップ（時間）について

 カート数を増やすとともに、実験が始まるから終了するまでの時間が徐々に減少していく。これはクラスタの数が増えていくと関係性があると考えられる。

以上の分析から見ると、カートのバッテリーの電力やカートのタイヤなどの消費を省くために、カートの数とフィールド（空港の大きさ）がある比例に維持できれば望ましい結果につながると考えられる。

第5章 まとめ

5.1 結論

本研究では、移動エージェントに基づく分散コロニークラスタリングの直線化ができた。ネットワークによって接続している群ロボットの制御について新たなアプローチを提案した。このアプローチでは、フィールド上に散在した群ロボット（カートロボット）が、アントエージェントとフェロモンエージェントの二つのエージェントからのポジティブフィードバックによる Stigmergy に基づいて、自律的にいくつかの直線なクラスタを生成し、このクラスタが、カートロボットが集合する準最適な場所となった。また、従来の研究にあったような、ACC を実行し集合場所を計算するホストコンピュータも不要であり、分散化された Stigmergy に基づくロボット制御が実現できる可能性を示した。さらに、シミュレータを作成し実験を行った結果、各エージェントが自律的に集合地点を決定し、その地点へ集合することができた。これは、フェロモンコミュニケーションモデルを応用した、群ロボット制御システムの実現可能性を示めている。

5.2 考察

5.2.1 問題点

この実験を経ていくつかの問題点を発見した。

- 収束しないケースがある

どうしても収束しない（カートが移動できない）ケースとして、カートの後ろが壁のときやカートに囲まれてしまったときといったものがあり、どうしようもないので、そのような状況に陥ったカートは強制的にロックするように工夫した。しかし、こうするとクラスタの数は

- 唯一のクラスタになってしまう

実際に実験を繰り返すと，カート数が少ない場合，すべてのカートが一直線になってしまうケースが少なくない．カート数 50 の場合，100 回試行の結果で，唯一のクラスタに集中したケースは 29 回．クラスタが直線になれると収集するときの手間が省けると思うけれども，1 つのクラスタのカート数があまりに多すぎると，通行の妨げとなってしまうかもしれない．参考図 5.1 ．

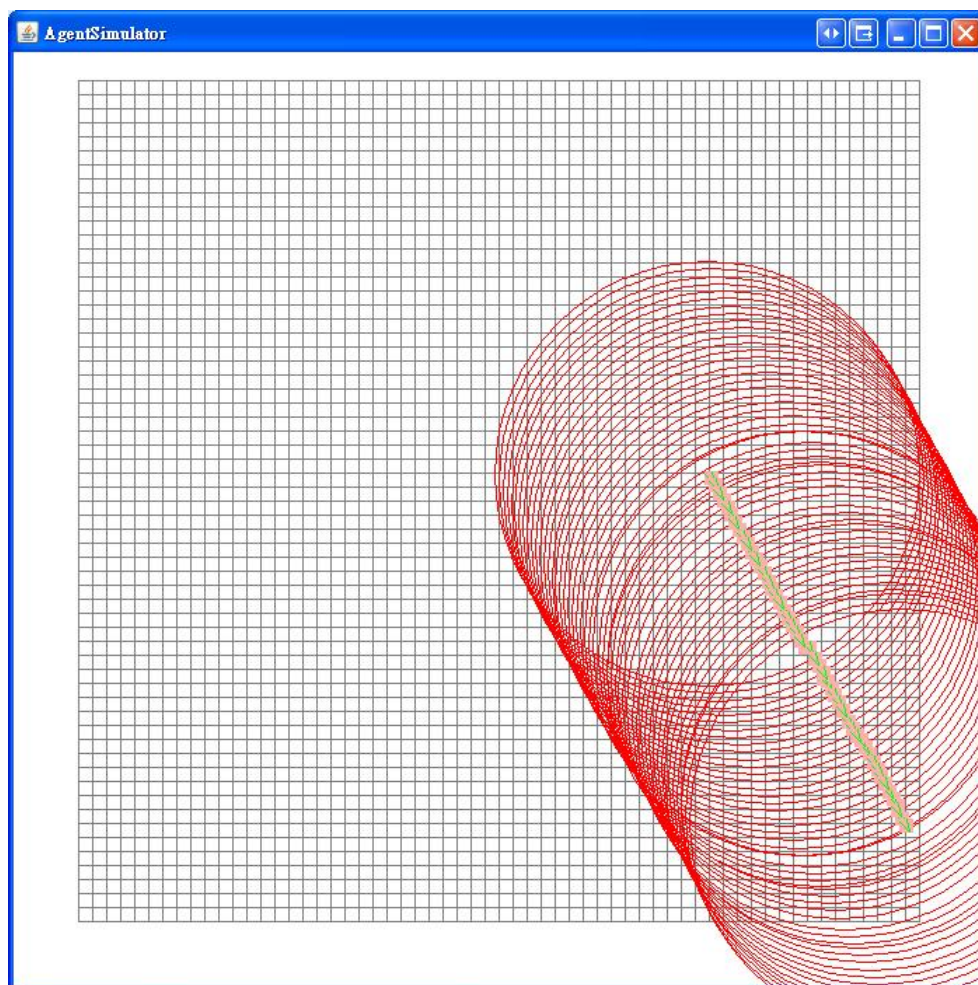


図 5.1: 唯一のクラスタになる

- フェロモンによる孤立化

その一、図 5.2 のように初期状態でカートがランダム配置され、AA を持ちカート が AA と PA とともに持っていないカート、カート と接触し、隣接カート数が 2 になったので、カート がすでに集合場所へ到達できたと判断し、カートをロックし PA が生成される。そして、一定時間が経過して、PA が消滅した後、AA と PA とともに持っていないカート、カートの近くにフェロモンが発生し、周りの二つのカートに AA がきて、このとき、カート とカート が飛んできた方向に移動する場合だと、カート とカート が立ち去って、カート がそのまま 1 つだけのクラスタになってしまう。

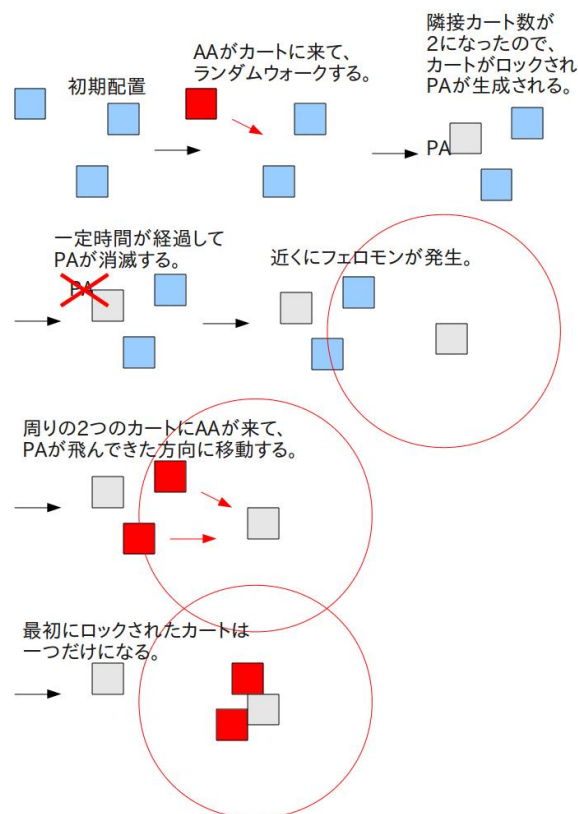


図 5.2: フェロモンによる孤立化

その二、図 5.3 のように初期状態でカートがランダムに配置され、AA がカート 1 にきて、カートがランダムウォークし始め、その後カート 1 とカート 2 の真ん中を通って、そのとき、隣接カートが 2 になったので、カート 1 がロックされ PA が生成される。そして、一定の時間が経過して、PA が消滅した後、AA と PA とともに持っていないカート 3、カート 4 に AA が来るが、隣接カート数が 1 なので、ランダムウォークを始める。この場合だと、カート 3 とカート 4 が立ち去って、カート 1 がそのまま 1 つだけのクラスタになってしまう。

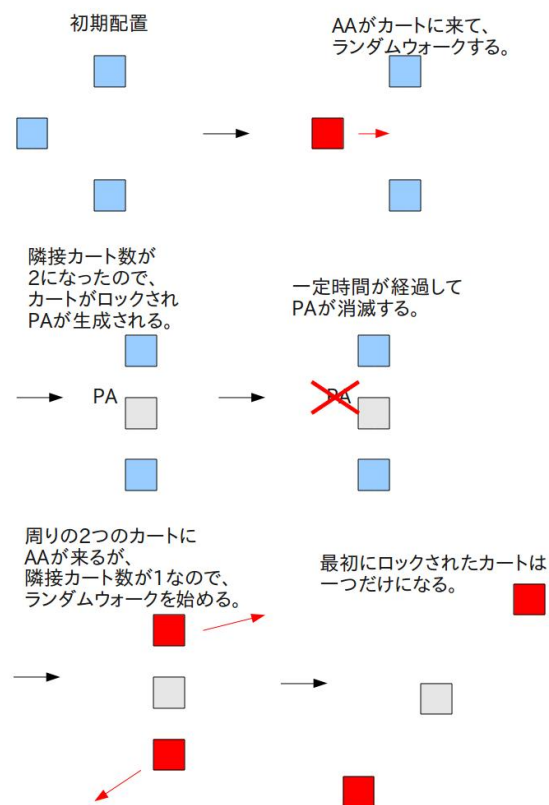


図 5.3: フェロモンによる孤立化

5.3 おわりに

科学技術の発達によって、ロボットという存在が身近になるだけでなく、例えばカートの様な今まで人が扱うために作られたものが自動化されロボットになっていく近未来において、今まで以上にロボット制御の効率化や新たな取り組みが必要となってくるのは明白である。しかし、本手法が示したように、外から見ると複雑なロボット制御システムが必要に見える場合でも、フェロモンコミュニケーションなどの Stigmergy に基づいた取り組みを行うことで、ひとつひとつの単体としては、非常に単純な動作ですむ。今後は、遺伝的アルゴリズム (GA) や群知能分野における最新のアルゴリズムだけでなく、他分野で研究されている最適化アルゴリズムとの併用によって、クラスタリング能力の向上やロボット制御の処理時間や消費電力などを減少させることができるであろう。

第6章 謝辞

本研究を行うにあたり，至らない私に基礎から根気良くご指導を賜りました滝本宗宏准教授にお礼申し上げます．また，研究の場を与えてくださった東京理科大学に感謝の意を表します．

参考文献

- [1] W.J. Binder • CG. Hulaas • Cand A. Villazon • CPortable Resource Control in the J-SEAL2 Mobile Agent System • CProceedings of International Conference on Autonomous Agents , pp. 222-223 (2001)
- [2] Takimoto,M., Mizuno,M., Kurio,M., Kambayashi,Y. : Saving energy consumption of multi-robots using higher-order mobile agents. In: Proceedings of the first KES International Symposium on Agent and Multi-Agent Systems, Lecture Notes in Artificial Intelligence 4496, Springer-Verlag , pp. 549-558 (2007)
- [3] Nagata,T., Takimoto,M., Kambayashi,Y. : Suppressing the total costs of executing tasks using mobile agents. In: Proceedings of the 42nd HICSS, IEEE Computer Society (2009)
- [4] Kambayashi, Y., Takimoto, M.: Higher-order mobile agents for controlling intelligent robots. International Journal of Intelligent Information Technologies 1(2), pp. 28-42 (2005)
- [5] Deneubourg, J., Goss, S., Franks, N.R., Sendova-Franks, A.B., Detrain, C., Chreien, L.: The dynamics of collective sorting: Robot-like ant and ant-like robot. In: Proceedings of the First Conference on Simulation of Adaptive Behavior: From Animals to Animats, MIT Press, Cambridge , pp. 356-363 (1991)
- [6] Dorigo, M. and Gambardella, L. M.: "Ant Algorithms for Discreate Optimization", Artifical Life Vol.5 No. 2, MIT Press, pp.137-172 (1999)
- [7] Wand, T., Zhang, H.: Collective sorting with multi-robot. In: Proceedings of the First IEEE International Conference on Robotics and Biomimetics, pp. 716-720 (2004)

- [8] Lumer, E.D., Faiesta, B.: Diversity and adaptation in populations of clustering ants, from animals to animats 3. In: Proceedings of the 3rd International Conference on the Simulation of Adaptive Behavior, MIT Press, pp. 501-508 (1994)
- [9] Kambayashi,Y., Tsujimura,Y., Yamachi,H., Takimoto,M., Yamamoto,H. : Design of Multi-Robot System Using Mobile Agents with Ant Colony Clustering. In: Proceedings of the 42nd HICSS, IEEE Computer Society (2009)
- [10] Masashi Mizutani, Munehiro Takimoto, Yasushi Kambayashi ;March 2010,ACIIDS'10: Proceedings of the Second international conference on Intelligent information and database systems: Part I
- [11] 大内東, 山本雅人, 川村秀憲著. : 「マルチエージェントシステムの基礎と応用」, コロナ社, (2002)
- [12] 沼岡千里, 大沢英一, 長尾確著. : 「エージェントテクノロジー最前線」, 共立出版社, (2000)
- [13] 大須賀昭彦. : エージェントの研究動向. 人間主体の知的情報技術に関する調査研究?, (財) 日本情報処理開発協会先端情報技術研究所,(2002).
- [14] AgentSpace - A Mobile Agent System. (2010/2/25 現在アクセス可)<http://research.nii.ac.jp/?ichiro/>
- [15] Ant Colony Optimization Home Page. (2010/2/25 現在アクセス可)<http://iridia.ulb.ac.be/?mdorigo/ACO/ACO.html>
- [16] F.Steele,Jr., G.Thomas : Directed Stigmergy-Based Control forMulti-Robot System. In: Proceedings of HCI'07, ACM (2007)
- [17] 安藤靖志, 深澤良彰, 増谷修, 佐々木宏, 岩崎弘利, 本位田真一. : フェロモンモデル: 交通渋滞予測への適用, 電子情報通信学会論文誌 DVol.J88-D1 No.9 pp.1287-1298 (2005)

付 録 A

巻末に本研究で作成したプログラムのソースコード添付する．以下は添付するソースの簡単な解説である．

- MainFrame.java
シミュレータの本体である．main() メソッド, paint() メソッド等をもち終了条件を満たすまでこのプログラムが実行される．
- Config.java
設定ファイルである．初期設定を記述してある．
- PA.java
PA クラスを記述してある．PA は PA クラスのオブジェクトとして実装される．
- AA.java
AA クラスである．AA は AA クラスのオブジェクトとして実装され, 各カートに指示を出す．
- AASState.java
AA の状態を示すための列挙型である．
- Cart.java
Cart クラスである．カートロボットは Cart クラスのオブジェクトとして実装され, AA からの指示で動くためのメソッドを宣言してある．
- CartState.java
Cart の状態を示すための列挙型である．
- CartAngle.java
Cart の向きを示すための列挙型である．

- Address.java
AA がカートを巡回するためにもつ「カートのリスト」を実装する ,
Address クラスである .
- Adj.java
クラスタにいるロボットの ID を格納するための Adj クラスである .