平成 24 年度卒業論文

論文題目

artisoc を用いた群れのふるまいのモデル

神奈川大学 工学部 電子情報フロンティア学科
学籍番号　200902740
小泉　駿

指導担当者　木下宏揚　教授

目 次

図 目 次

第1章　序論

インターネット上で様々な情報を交換しながら築かれる人間関係によって，多様なコミュニティ（群れ）が形成される．例えば，現代の企業活動において，企業間や顧客との多様な関係の中でクラウドを介して情報のやりとりをするようになってきている．

従来のコンピュータ利用は，ユーザー（企業、個人など）がコンピュータのハードウェア，ソフトウェア、データなどを，自分自身で保有・管理していたのに対し、クラウドコンピューティングでは「ユーザーはインターネットの向こう側からサービスを受け、サービス利用料金を払う」形になる．ユーザーが用意すべきものは最低限の接続環境（パーソナルコンピュータや携帯情報端末などのクライアント，その上で動くブラウザ，インターネット接続環境など）のみであり，加えてクラウドサービス利用料金を支払う．実際に処理が実行されるコンピュータおよびコンピュータ間のネットワークは，サービスを提供する企業側に設置されており，それらのコンピュータ本体およびネットワークの購入・管理運営費用や蓄積されるデータの管理の手間は軽減される．[1]

その中で情報漏えいの問題が出てくる．情報漏えいのひとつとしてCovertChannelから生じている．CovertChannelはアクセス制御の設定を行っているにも拘わらず，情報の読み書きの連続によって想定外の相手に伝わるという情報流（不正通信路）である．例として，複数の企業が業務提携をする場合を考える．それぞれの企業が有する情報の中に，外部と共有可能な情報と外部と共有可能ではない機密情報が入り混じっていると考えられる．この時，直接機密情報へのアクセスが制御されていたとしても，ほかの客体（object）とオペレーション（Read,Write）を複数介したときに機密情報へのアクセスができてしまうというものである．これがクラウドシステムの中でさらに顕在化している．

情報漏えいが問題になってくるが，個別の情報リソースを情報フィルターで制御してしまうと，システムの利便性の低下を引き起こしてしまう恐れがある．

前年度の研究ではコミュニティからの情報漏えいを制御するシステムを研究をしていた．従来のセキュリティは，客体自体を護るという概念である．完全に護ろうとすることが強い制約になり，情報を創発する振舞いを阻害する方向性を持つ．提案するセキュリティは，客体を使って主体の群れが行為することを護る，という概念である．行為し続けることを護るので、主体の群れの振舞い自体が進化し，情報を創発し続けるシステムである．

前年度の研究はクラウドシステムの情報セキュリティを目的としていた。常に活動し続ける社会の中で引き起こされる情報漏洩の調和を、個々の情報リソースを護るのではなく、群れの進化的な作用自体を動きの中で保つシステムを探求する。つまり、動的な社会の中に公私の価値の循環（調和）を実現するシステムを研究したい。そこで、「群知能」に着目し、進化と制約の矛盾を解決するような抽象的なシステムをシミュレーションし、多様な社会をアナロジーとして記述するための構造を研究をしていた.

また、社会の中に公私の価値の循環性を色で表現しようと試みていた.「色の混色」と「鳥の群れ」を社会システムのアナロジーとして考える. 着するアナロジーは,「色の混色」という操作と,「鳥の群れ」という相互作用である．情報の意味を創発することのアナロジー，群れをなして意味を創発するアナロジーは，色光の三原色の加法混色と色料の三原色の減法混色を使って表現される．公私の価値循環という相互作用において情報漏えいがないように複雑系のマルチエージェントで情報流を制御する.

複雑系というのは，ただ単なる「要素が多くて複雑」「理解できなくて複雑」ということではなく要素を切り分けていくと本質が抜け落ちてしまうといった事柄へのアプローチ手法として「複雑系」がクローズアップされてきた．[9][10][12] クラウドという複雑系において，内在する論理の間の相互作用を解明し，複雑系における生成と消滅の相互依存関係を，色彩の世界の色光３原色，色料３原色の循環の比喩にゆって記述を試みている.

コミュニティ同士の連携や競争といった相互作用を研究することで, コミュニティの価値の漏洩を防ぎつつ，常に活動し続ける社会を表現するということが研究の目的である．それを実現するためには，artisoc というマルチエージェント・シミュレータが使えるのではないかと考えた．[2]

しかし，artisoc の様々なサンプルモデルを組み合わせたモデルの作成はされていなかった.

そこで，artisoc の様々なサンプルモデルを組み合わせたモデルの作成し，群れを成すモデルを作成しようと考えた．一つ一つのモデルが研究に適している要素があるので，この要素を組み合わせることで研究に必要なモデルができるのではないかと考えた.

friendship モデルではエージェントの群れを作ることができるが，ただ群れを作るものであり，ふるまいが集まり情報漏えいを起こしてしまう可能性がある．そこで，classmate モデルという関係性を持たせるものを利用することで，情報漏えいを起こすふるまいを群れから離れさせ，情報漏えいを防止できるのではないかと考えた.

この研究は情報の読み書きの連鎖を”ふるまい”として，個々の情報リソースを守るのではなく，群れの作用（”ふるまい”）自体を守るシステムを探求する．ふるまいに着目し，群れが集まる研究する．群れの表現方法として「群知能」を用

いてシミュレーションする．

これにより，群れを成立させる適切なパラメータを求め，現実の情報漏えい問題との対応付けの検討が可能となる．まず私は，artisoc というマルチエージェント・シミュレータを用いて，様々なサンプルモデルの中から，friendship モデルと classmate モデルを適用することで”ふるまい”の群れが形成されることをシミュレーションで確認する．

第２章　基礎知識

2.1　Boid

1987年，アメリカのリレイグ・レイノズルによって考案・製作された人工生命シミュレーションプログラムである．Boidのモデル名は鳥もどきという意味（Bird-android）．Boidの群れを実現させる振る舞いは３つの要素からなる．「衝突の回避」,「群れの中心に向かう」,「向きを合わせる」といった３つのルールを規定するだけで鳥の群れをシミュレーションすることができる．[3] [15] [16]

2.1.1　「衝突の回避」Separation

引き離し．近くの鳥や物体に近づきすぎたらぶつからないように離れるルールである．Boid同士が近づきすぎたら，前を飛んでいるBoidはスピードを速くして，後ろを飛んでいるBoidはスピードを遅くするといったように，衝突の回避のためにBoidはそれぞれ自分にとっての最適距離を持っている．障害物に対しては，それにぶつからないように方向転換して衝突を避けるようにする．[3]

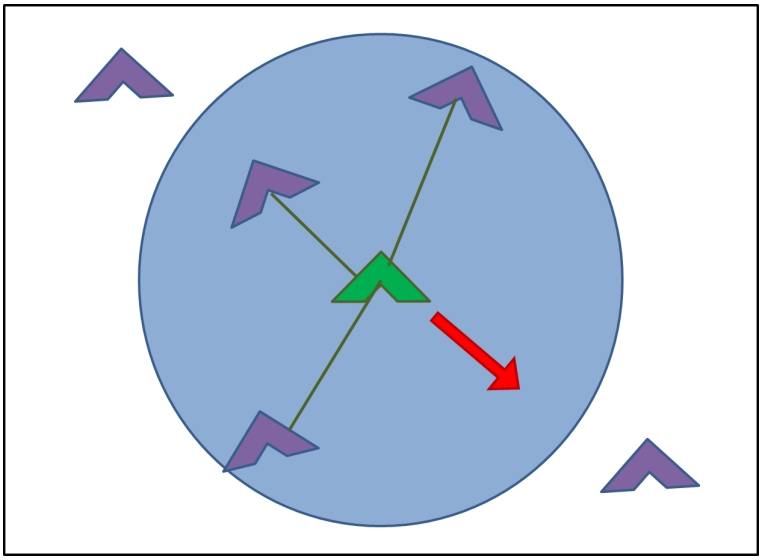

図 2.1: 衝突の回避

2.1.2 「群れの中心に向かう」Cohision

結合．鳥たちが多くいる方向へ向かって飛ぶルール．鳥たちが多くいる方へ向かって飛ぶというのは，おおざっぱにいうと群れの中心（重心）方向にむかうということ．つまりこのルールは,Boid に群れの中心の方向へ飛んでいくとを指示している．この群れの中心は, 全 Boid の位置（座標）の平均として求める．[3]

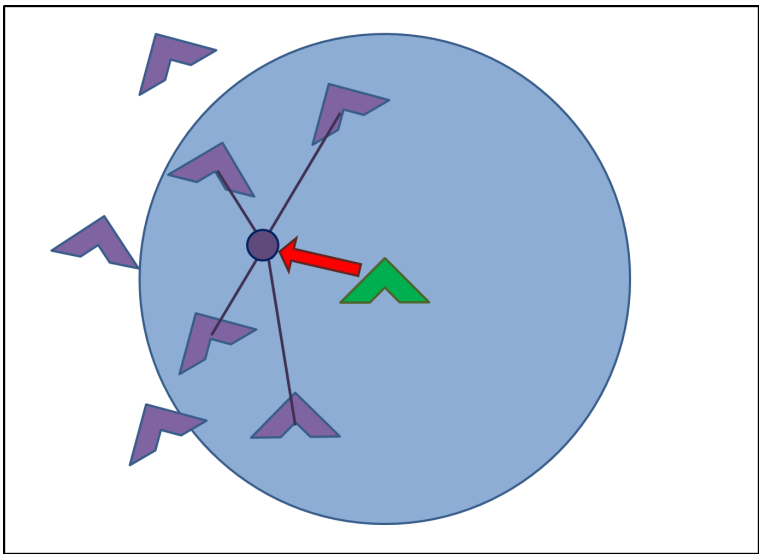

図 2.2: 群れの中心に向かう

2.1.3 「向きを合わせる」Alingment

整列．近くの鳥たちと飛ぶスピードや方向を合わせようとするルール．同じ方向にあまり距離を空けないように飛ぶようにする．このルールはある一定の距離より遠ざかり過ぎてしまったら，前の Boid はスピードを遅くし，後ろを飛んでいる Boid はスピードを速くするといったようにすることで実現することができる．[3]

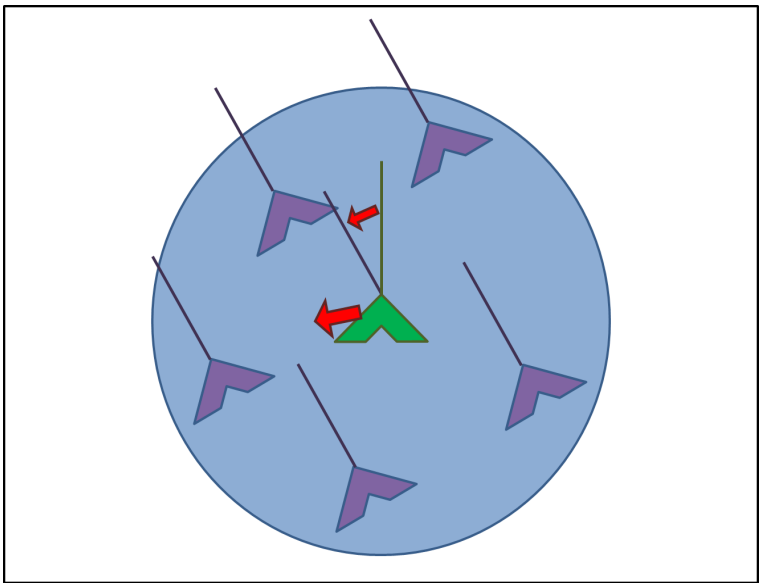

図 2.3: 向きを合わせる

2.1.4 ACO(Ant Colony Optimization)

アリの摂食行動から着想を得たアルゴリズム.

フェロモンという揮発性物質を模したパラメータを最適化する.

組合せ最適化やネットワークルーティングなどに応用.

実世界では，アリは始めランダムにうろつき，食物を見つけるとフェロモンの跡を付けながらコロニーへ戻る．他のアリがその経路を見つけると，アリはランダムな彷徨を止めてその跡を辿り始め，食物を見つけると経路を補強しながら戻る．しかし，時間とともにフェロモンの痕跡は蒸発しはじめ，その吸引力がなくなっていく．その経路が長いほどフェロモンは蒸発しやすい．それに対して，経路が短ければ行進にも時間がかからず，フェロモンが蒸発するよりも早く補強されるため，フェロモン濃度は高いまま保たれる．従って，あるアリがコロニーから食料源までの良い（すなわち短い）経路を見つけると，他のアリもその経路を辿る可能性が高くなり，正のフィードバック効果によって結局すべてのアリが 1 つの経路を辿ることになる．[4][11]

2.2 CovertChannel

アクセス行列において，subject,object, パミッションをアクセストリプルと定義し，そのアクセストリプル間で起きる不正な情報の経路（CovertChannel）を定義する．

定義 1 *Covert Channel* アクセス行列において，アクセス禁止のパミッションに矛盾する情報フローを *Covert Channel* と呼ぶ．

即ち，
subjectS_i(i=1,2, ⋯)，S_j(j=1,2, ⋯)，ただし $i \neq j$
objectO_n(n=1,2, ⋯)，O_m(m=1,2, ⋯)，ただし $n \neq m$
パミッション P {RW,¬RW，¬WR,¬ (RW)}，ただし R (READ)，W (WRITE) とするとき，アクセストリプル (S_i, O_n, P) について，
If $(S_i, O_n, \neg R)$,
And if $(S_j, O_n, R) \wedge (S_j, O_m, W) \wedge (S_i, O_m, R)$
Then covert channel$(S_i, S_j, O_m(O_n))$
と定義する．この定義では，subjectS_i はobjectO_n という名前とその内容をREAD禁止し $(S_i, O_n, \neg R)$ であるにも拘わらず，subjectS_i が objectO_n の内容を読み込み (READ)，objectO_m にそれを書き込み (WRITE)，subjectS_i が objectO_m の内容からobjectO_n の内容を読み込む (READ) 事 ((S_i, O_n, R) と表現される) によって，アクセストリプル $(S_i, O_n, \neg R)$ と矛盾する結果が生じることが示される．[2][13]

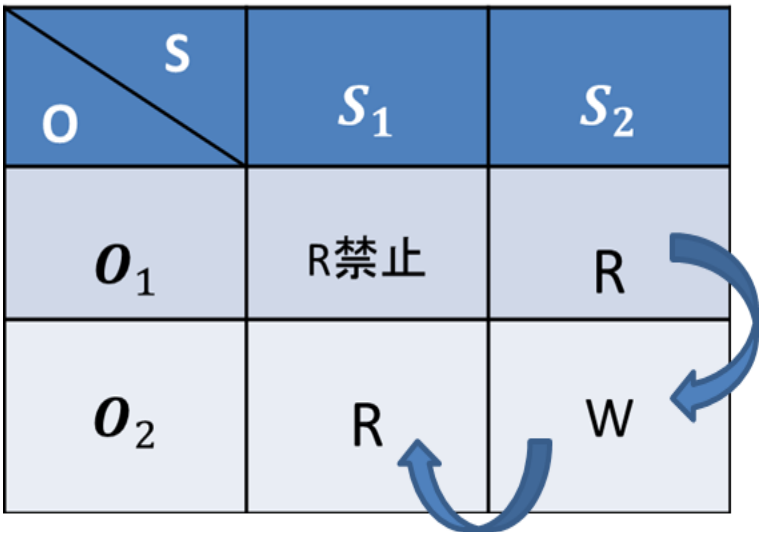

図 2.4: CovertChannel の例

2.3 言語ゲーム

2.3.1 ウィトゲンシュタインの基本概念

ウィトゲンシュタイン著『哲学探究』において言語活動をゲームで捉え，言葉の意味を外延（対象）や内包（共通性質）ではなく，特定のゲームにおける機能として理解すべきことを提唱したもの.「言語の機動的なふるまい」といったニュアンスであり，トランプのジョーカーの意味がそれを用いて遊ぶゲームによってことなるといったようなもの．この言語ゲームそのものもすべてに共通する内包を持たず，親戚関係のようにゆるい連鎖によって一体化している．[5]

2.3.2 言語の限界

規則に基づく行為は言語の限界であり，言語世界に住む＜私＞の観察視点からは，言語的に量子化された＜私＞は言語の規則に盲目に従うように見える．その意味で規則や規則に基づく行為は見直され，社会システムのために転回される必要がある．すなわち，言語から規則や規則に基づく行為“ 言語ゲーム ”へと思索の視点を変えねばならない．しかし，そのような“ 見直し ”には限界がある．ひとたび行為世界に入ったとしても“ 見直し ”は限界のある言語によって記述されなければシステムを構成する事が出来ない．[2]

2.3.3 家族的類似

言語ゲームで振る舞う particle は，家族的類似性によって群れを作る，と定義した．家族的類似性は言語と行為の類似性を表現するものである．家族的類似性は同値関係でもないし，等価関係でもない．常に変動しつつ，少しづつ似ているエンティティの集まりであるが，それは自己から見れば，同値関係であってもよいし等価関係であってもよい．そのような個人個人の意味論を統合して世界を記述する意味論が無い，ということである．[2][5]

2.3.4 群れるふるまい [2]

集まる力の源は ”家族的類似 ” であり，群知能のパラメータとして表現される．
群れる正の力

- 群れの中心に向かう力： Cohision
- 隣人と家族的類似行為をする： Alingment
- 行為の濃度： Pheromone

Pheromone：行為の重要性を表現する．揮発性．濃度が濃い Pheromone は重要な行為を表す．

群れる負の力

- 群れから排除する力，離れる力： Separation

2.4 マルチエージェント・シミュレータ

2.4.1 マルチエージェント

自らの価値基準に従って自分の行為を自由に選択できるような自律的なが多数共存する環境である．エージェントは，構成要素のうち特に主体性を持ち，自律的に行動するものである．エージェント同士の相互依存関係があり，エージェント同士の相互作用により，やがてシステム全体の流れのようなものが創発され，その流れが今度は逆にエージェントにフィードバックされ，また個々のエージェントのふるまいを決定していく．[6]

2.4.2 artisoc

人工社会（artifical society）を縮めた名前．もっぱら理科系のツールとして開発されていたマルチエージェント・シミュレーションが, 社会科学の方法のひとつとしても有望ではないかと言われはじめてから 10 年以上たつ．しかしこの手法が今でもあまり浸透していないのは, 社会科学の研究者や学生にとって、まずプログラミング言語から学ぶ必要があり, 要するに敷居が高い手法だったことに起因すると思われる．この事情は, アプリケーションソフトが多数市販されている統計分析の手法と比べると一目瞭然である．

たしかにアメリカを中心に, 人工社会を念頭においた汎用シミュレータはいくつか存在しているが, 使いやすいシミュレータでは単純で定型的なモデルしかつくれない. 複雑で様々なモデルを作れる強力なシミュレータを操作するには高度な専門知識が必要である．

そのような状況で, 数年前に登場したものが Windows パソコン用のシミュレータ KK-MAS である. これは構造計画研究所が開発した, 日本語環境のパソコンの上で, プログラミング言語やプログラミング技法を学ばないでも利用できる, しかも汎用性のあるマルチエージェント・シミュレータである. この KK-MAS をプロトタイプにしつつ，シミュレーションを実行することを念頭において開発された第二世代のシミュレータである．

人間同士の相互作用をコンピュータ上で誰もが簡単に再現することができ，ダイナミックに変化する社会現象を生きたまま分析できる．

デバック機能の追加やエージェント関数の拡充，マルチプラットフォーム対応やデータベース連携，web ブラウザからの操作，複数モデルの同期実行など，様々な機能が拡張している．

人工社会という新しい捉え方が従来のものとは異なる最大の特徴は，主体の相互作用に焦点を当てたボトムアップ・アプローチである．人工社会がボトムアップ・アプローチであるという特徴は，主体間の相互作用にせよ主体と環境との相

互作用にせよ，局所的な関係をモデル化しさえすれば，全体についての性質は自ずと現れるという社会の捉え方である．[2][7]

2.5 その他のマルチエージェント・シミュレータ

2.5.1 swarm

生物エージェント（虫など）と無生物エージェント（壁や障害物）からなる人工世界のボトムアップモデルに基ずくシミュレータ．エージェントの動きを簡単な規則で記述し，エージェント間の相互作用による創発現象を観察する．

特徴は，シミュレーション・プログラミングを補助するオブジェクト指向型のソフトウェアライブラリの集まり．ユーザは自分のプログラムから swarm ライブラリのオブジェクトを取り込むことでシミュレーションを構築．swarm のライブラリは Java と ObjectiveC に対して提供されている．

swarm のシミュレーションは

- Model：人工社会を抽象敵にモデル化し，シミュレートする．
- Observer：モデルのシミュレーションを観察し，表示などを行う．

の 2 つの部分からなる．[2][8]

2.5.2 Mason

2 次元・3 次元の空間を使用した物理・社会モデル用のシミュレーションライブラリ. 適切なクラスを継承してモデルを実装することでそれに合わせた GUI が提供される. モデル開発というより, MASON ライブラリを使用したアプリケーション開発という色合いがどうしても強くなってしまう. 分析機能はなく, 動きを見て楽しむというのが主眼のようである．[2][8]

2.6 公共性・私性の価値の循環的調和 [2]

公私の価値循環という相互作用において情報漏えいがないように複雑系のマルチエージェントで情報流を制御する.

複雑系において，内在する論理の間の相互作用を解明し，複雑系における生成と消滅の相互依存関係を，色彩の世界の色光３原色，色料３原色の循環の比喩にゆって記述を試みる.

1. エージェントは色彩 Object を取り込み，あるいは生成.
2. 混色，編集する.
3. 同系色相の陣取りゲームをする.
4. 仲間を増やすことができる.
5. 濃い色彩が他者に漏えいしてはならない.
6. 同系色彩が異系色彩に漏えいしてはならない.

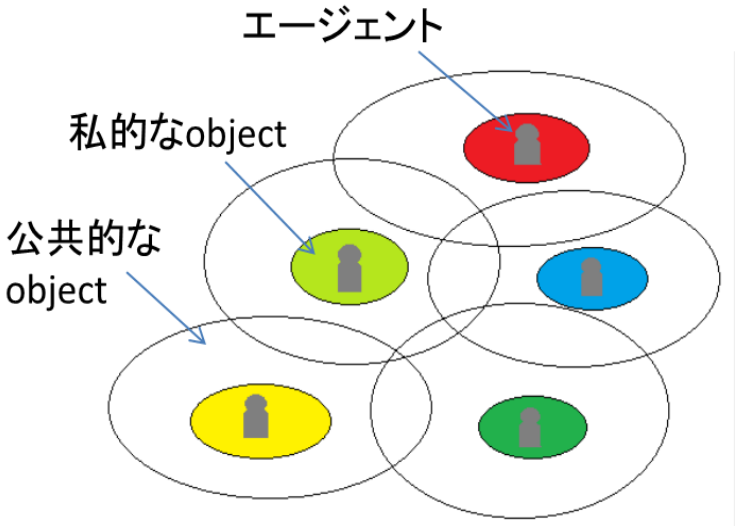

図 2.5: 公共性・私性の価値の循環的調和

2.7 公共性・私性の循環を模擬する色光・色料の循環

公共的・私的な価値の循環の比喩を色彩の濃淡（明度）の比喩で表す．私的な場で採色した淡い色彩は、公共的な場では，減法混色で濃く彩色される．公共的な場で濃くなった色彩は，私的な場の加法混色によって，淡く彩色される．[2]

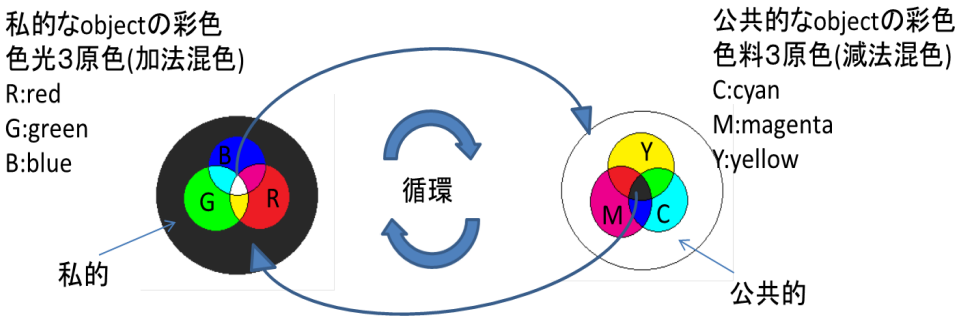

図 2.6: 色彩世界の循環比喩

第3章　提案

3.1　提案するモデル

ふるまいに着目し，群れが集まる研究する．群れの表現方法として「群知能」を用いてシミュレーションする．

これにより，群れを成立させる条件を求め，現実の情報漏えい問題との対応付けの検討が可能となる．

Boid 構造は連携を表現し, Particle は群れを作り動き回るための「衝突の回避」,「群れの中心に向かう」,「向きをあわせる」といった規則を持つ. 群れからの情報漏えいを調和するシステムを研究していくにあたって、群れ同士の相互作用、特に「関係性」と「連携作用」に着目する．

群れを表現するにあたって，Boid と ACO のフェロモンの概念を利用する．Boid は群れを作る連携の作用，フェロモンは行為の重要性を表す．これらから，群れが集まる”ふるまい”をマルチエージェント・シミュレータで記述する．情報漏えいを少なくするように常に群れを成り立たせ，その条件を求めたい．

artisoc の様々なサンプルモデルの中から friendship モデルと classmate モデルの２つのモデルを組み合わせて，距離ではなく関係のパラメータによって群れるモデルを作成したい．

friendship モデルではただ近いエージェントに近づいて群れを作るので，群れは成り立つが，意味もなく群れを作ってしまう．そこで classmate モデルを用いることでエージェントに関係を持たせることがわかった．よって，群れを作る friendship モデルに関係を持たせる classmate モデルを組み合わせることにより，関係性によって群れを成り立たせようと考えた．群れを情報の交流を表し，距離が近いほど情報交換をしていることを意味する．従来のアクセス制御では情報漏えい防止を重点において情報フィルターを適応させ，情報を完全に護るという強い制約をしていたが，提案としては情報リソースに対する読み書きの系列をエージェントとし、その集合（群れ）から関係によって離れていったエージェントに対して情報フィルターを適応させることで深刻な情報漏えいを防ぐことが可能である．

例えば，企業同士や企業と顧客とで情報のやり取りをする際に，共有可能な情報を交換することができ，外部と共有可能ではない機密情報，見られてはいけない情報を守ることが可能である．

3.1.1 friendship モデル

friendship モデルは近くにいるエージェントを見つけ，そこに近づくモデル．エージェントを集合させる機能，Boid のような動きをするので，これを利用できないかと考えた．

friendship モデルは，好みの近い者同士が友達になる過程をモデル化したものである．[7]

friendship モデルの入力コマンドと実行結果の図 3.1 を下記に記述する．

```
Agt_Init{ //シミュレーションの最初だけに実行されるルール
My.X = Rnd()*50 //50 × 50 の空間の中で
My.Y = Rnd()*50
My.Direction = Rnd()*360 //ランダムな方角に進む
ClearAgtset(My.friends) //friends を初期化（中を空に）する
}
Agt_Step{ //毎ステップ実行されるルール
Dim s As Integer //s という変数を整数型として（As Integer）
Dim one As Agt
Dim temp As Agtset
Dim close As Agtset
Dim neighbor As Agtset
Turn(Rnd()*360) //とりあえず、好みはランダムに変わろうとする
//自分の好みと似ている人を友人にする
MakeAllAgtSetAroundOwn(neighbor, 3, False)
If CountAgtset(neighbor) > 0 then
one = GetAgt(neighbor, Int(Rnd()*CountAgtset(neighbor)))
AddAgt(My.friends, one) //one を My.friends に追加
TurnAgt(one) //one の好みにあわせようとする
//My.friends の重複をなくす
DuplicateAgtset(My.friends, temp)
End if
Forward(1)
}
```

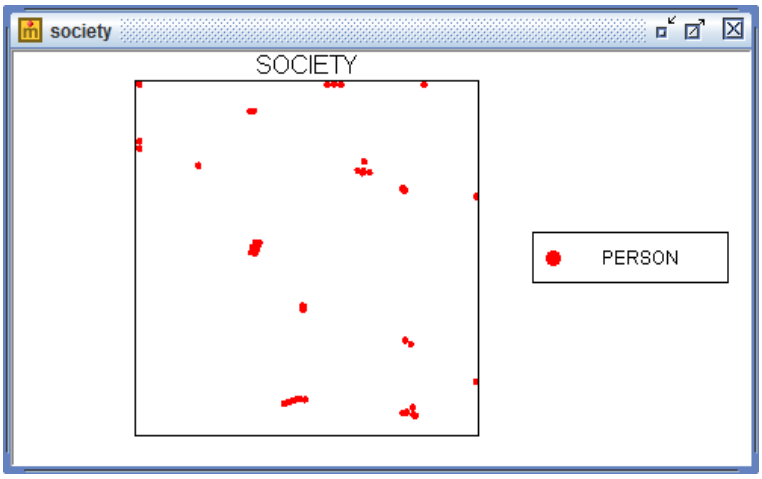

図 3.1: friendship モデルの実行結果

この friendship モデルを少し複雑にしたものもある．いったん友達同士になると，お互いの好みが離れると近づけようとするが、あまり似すぎているかえって好みを変えようとする動きである．[7]

friendship モデルのプログラムの「forward(1)」の直前に、下のルールを書き込む．

```
\\離れた友人の好みに合わせようとする
DuplicateAgtset(temp, My.friends)  \\友達全員コピー
DelAgtset(temp, neighbor)  \\好みの近い人を除く
If CountAgtset(temp) > 0 then    \\好みの離れた友人がいれば
one = GetAgt(temp, Int(Rnd() * CountAgtset(temp)))
TurnAgt(one)    \\友人の好みに合わせようとする
Else
\\好みの近すぎる友人から離れようとする
MakeAllAgtsetAroundOwn(close, 1.5, False)    \\好みが近すぎる人
MakeCommonAgtset(temp, close, My.friends)    \\好みが近すぎる友達
If CountAgtset(temp) > 0 then    \\好みが近すぎる友人がいれば
one = GetAgt(temp, Int(Rnd() * CountAgtset(temp)))
TurnAgt(one)
Turn(180)    \\結局，反対方向に向く
End if
End if
```

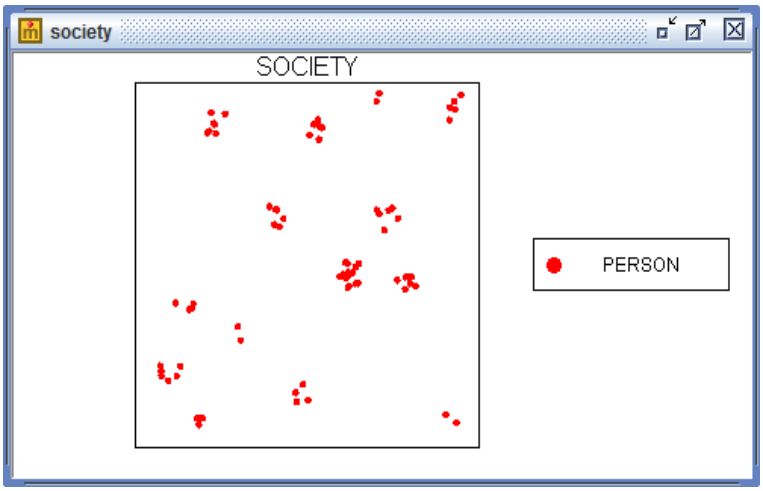

図 3.2: friendship モデルを少し複雑にしたモデルの実行結果

3.1.2 classmate モデル

エージェントに関係を持たせるモデル．フェロモンの概念に利用できないかと考えた．フェロモンは行為の重要性を表すので，重要性を関係性に表せないかと考えた．

エージェント同士が衝突したらエージェント同士が遊ぶか喧嘩をする．その時にエージェントに値を蓄積（遊んだら＋１，喧嘩したら＋０）する．その経験を数回行い，蓄積された数値の差でどれが好きなエージェントか，どれが嫌いなエージェントかを決める．[7]

classmate モデルの入力コマンドと実行結果の図 3.3 を下記に記述する．

```
Agt_Init{
My.X=Rnd()*20
My.Y=Rnd()*20
My.Direction=Rnd()*360
}
Agt_Step{
Dim i As Integer
Dim one As Agt
Dim neighbor As Agtset
Turn(Rnd()*180-90)
If Forward(1)<>-1 Then //前進してぶつかったら U ターン
     Turn(180)
End if
//近くにいる同級生と遊ぶかケンカする
MakeAllAgtsetAroundOwn(neighbor,2,False)
For each one in neighbor
i=Round(Rnd()) //四捨五入して０（遊ぶ）か（ケンカ）
My.relation(one.ID,i)=My.relation(one.ID,i)+1 //関係を蓄積
Next one
//同級生と恒久的関係の構築
ClearAgtset(My.friends) //好きな相手を初期化
ClearAgtset(My.enemies) //嫌いな相手を初期化
MakeAgtsetspace(neighbor,Universe.classroom)
For each one in neighbor
If My.relation(one.ID,0)>=My.relation(one.ID,1) +3 Then
AddAgt(My.friends,one) //好きな相手に加える
Elseif My.relation(one.ID,1)>=My.relation(one.ID,0) +3 Then
AddAgt(My.enemies,one)
```

```
End if
Next one
}
```

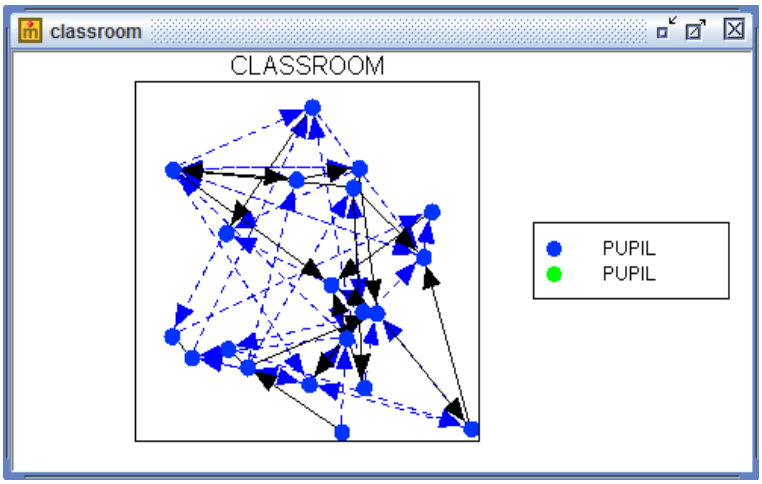

図 3.3: classmate モデルの実行結果

関係は矢印で表される．黒い矢印は好きな相手を表し，青い点線の矢印は嫌いな相手を表す.

3.2 組み合わせたモデル

friendshipモデルとclassmateモデルを組み合わせたモデルの作成を試みた．classmateモデルをベースに作成し，friendshipモデルの集まるプログラムともう一つ，少し複雑にしたfriendshipモデルの一部を参考にして作成した．あるエージェントの周り好きな相手がいたらその相手に近づき，周りに嫌いな相手が数人いたら離れていくというものを考えた．関係を示す値（蓄積させる値）は変えてはいないが，周りにいる好きなエージェントの数と嫌いな相手の数を変えてプログラムを組み立てた．

組み合わせたモデルの入力コマンドと実行結果の図3.4を下記に記述する．

```
Agt_Init{
My.X=Rnd()*20
My.Y=Rnd()*20
My.Direction=Rnd()*360
}
Agt_Step{
Dim i As Integer
Dim one As Agt
Dim neighbor As Agtset
Turn(Rnd()*180-90)
If Forward(1)<>-1 Then //前進してぶつかったら U ターン
     Turn(180)
End if
//近くにいる同級生と遊ぶかケンカする
MakeAllAgtsetAroundOwn(neighbor,2,False)
For each one in neighbor
i=Round(Rnd()) //四捨五入して0（遊ぶ）か（ケンカ）
My.relation(one.ID,i)=My.relation(one.ID,i)+1 //関係を蓄積
Next one
//同級生と恒久的関係の構築
ClearAgtset(My.friends) //好きな相手を初期化
ClearAgtset(My.enemies) //嫌いな相手を初期化
MakeAgtsetspace(neighbor,Universe.classroom)
For each one in neighbor
If My.relation(one.ID,0)>=My.relation(one.ID,1) +3 Then
AddAgt(My.friends,one) //好きな相手に加える
Elseif My.relation(one.ID,1)>=My.relation(one.ID,0) +3 Then
AddAgt(My.enemies,one)
```

```
End if
Next one

DuplicateAgtset(temp, My.friends)  \\好きな相手をコピー
DuplicateAgtset(temp2, My.enemies)  \\嫌いな相手をコピー
MakeAllAgtsetAroundOwn(neighbor, 3, False)
if CountAgtset(temp) > 0 then  \\近くに好きな相手がいると
one = GetAgt(temp, Int(Rnd() * CountAgtset(temp)))
TurnAgt(one)  \\その相手の方に向く
Elseif CountAgtset(temp2) > 1 then  \\近くに嫌いな相手が数人いたら
one = GetAgt(temp2, Int(Rnd() * CountAgtset(temp2)))
TurnAgt(one)
Turn(180)  \\反対に向く
End if

}
```

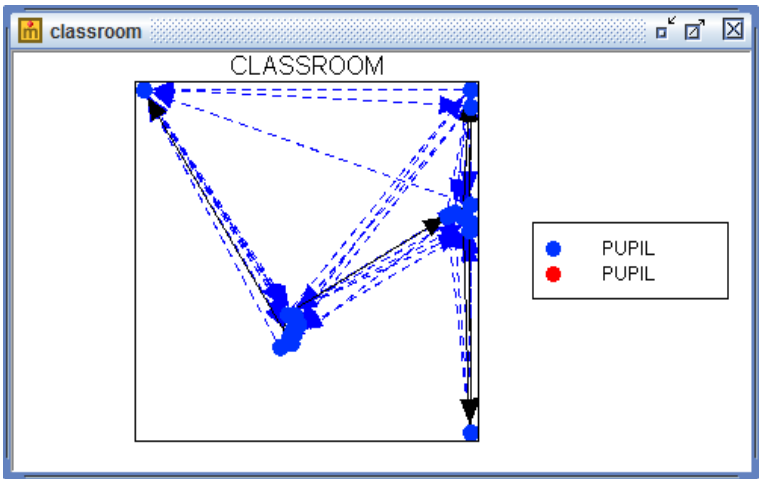

図 3.4: 組み合わせたモデルの実行結果

関係は矢印で表され，黒い矢印は好きな相手を表し，青い点線の矢印は嫌いな相手を表す．図 3.4 を見ると，いくつか群れを成しており群れの間に青い点線が多くある．なので嫌いな相手と距離を置いていることになる．より遠くのエージェントに対して関係によって群れから離れていったエージェントを群れと離れたエージェントの距離に応じて情報フィルターを適応させることで深刻な情報漏えいを防ぐことが可能である．

この作成したプログラムの関係を示す値（蓄積させる値），周りにいる好きな相

手，嫌いな相手の数の値を変え，シミュレーションを何度も重ね，常に群れを成す状態を求めたい．

また，このプログラムは空間を小さく設定してシミュレーションをしている．エージェントの数を増やし，空間を大きくしてシミュレーションする際は関係を示す値や周りにいる好きな相手，嫌いな相手がどの程度の数を認識させてエージェントの動きを制御するか検討をする必要である．

第4章　結論

情報の読み書きの連鎖を”ふるまい”として，個々の情報リソースを守るのではなく，群れの作用自体を守るシステムを探求するため，ふるまいに着目した．群れの表現方法として「群知能」を用いてシミュレーションし，群れを成立させる適切な条件を求めることで，現実の情報漏えい問題との対応付けの検討が可能となる．

図 3.4 より，関係性によって群れを成すことが確認できた．今後の課題として，作成したプログラムの数値，エージェントに蓄積させる値，周りにいる好きな相手，嫌いな相手の数などの値を変え，シミュレーションを何度も重ね，統計データを収集し常に群れが成り立つ条件を求める必要がある．そして，現実の情報漏えい問題との関係を明確にし，現実のシステム制御の研究に繋げたい．

謝辞

本研究を行本研究を行うにあたり，終止熱心にご指導して頂いた木下宏揚教授，ご多忙の折研究室に足を運び様々な面で有益なご助言をして頂いた森住哲也氏に深く感謝いたします．さらに，公私にわたり良き研究生活送らせて頂いた木下研究室の方々に感謝いたします．

2013 年　2 月

小泉　駿

参考文献

[1] 日経 BP 社出版局編，クラウド大全 The Complete Cloud Computing　サービス詳細から基盤技術まで　，2009

[2] 内山竜佑，木下宏揚，多様性を実現する群知能の振舞いのモデル，神奈川大学 2011 年度卒業論文

[3] ”Boid とは”
http://members.jcom.home.ne.jp/ibot/boid.html

[4] 磯村淳，木下宏揚，クラウドファイルシステム，神奈川大学 2011 年度卒業論文

[5] 藤本隆志訳，大修館書店ウィトゲンシュタイン全集　第 8 巻　哲学探究，1976

[6] ”MAS コミュニティ　マルチエージェントとは”
http://mas.kke.co.jp/modules/tinyd4/index.php?id=3

[7] 山影進，人工社会構築指南 - artisoc によるマルチエージェント・シミュレーション入門 -，2010

[8] ”マルチエージェントシミュレータ比較”
http://www.gpgsim.net/gpgsim/comp-mas.html

[9] ”MAS コミュニティ これまでのシミュレーションの考え方”
http://mas.kke.co.jp/modules/tinyd4/

[10] ミッチェル・ワールドロップ，複雑系—生命現象から政治、経済までを統合する知の革命，新潮社，1996

[11] 伊庭斉志，進化論的計算手法 (知の科学)，人工知能学会編，オーム社，2005

[12] ”MAS コミュニティ 複雑系とは”
http://mas.kke.co.jp/modules/tinyd4/index.php?id=2

[13] 小松充史, 木下宏揚，Covert Channel 分析制御のために推論を導入した情報フィルタに関する研究，神奈川大学 2006 年度卒業論文

[14] Cameron, Peter J. (1991), Projective and polar spaces, QMW Maths Notes, 13, London: Queen Mary and Westfield College School of Mathematical Sciences, MR:1153019

[15] Delgado-Mata C, Ibanez J, Bee S, et al. (2007). "On the use of Virtual Animals with Artificial Fear in Virtual Environments". New Generation Computing 25 (2): 145?169. doi:10.1007/s00354-007-0009-5

[16] Hartman C, Benes B (2006). "Autonomous boids". Computer Animation and Virtual Worlds 17 (3-4): 199?206. doi:10.1002/cav.123

[17] Lebar Bajec, I. and F. H. Heppner (2009). "Organized flight in birds"

[18] Ibanez J, Gomez-Skarmeta A F, Blat J (2003). "DJ-boids: emergent collective behavior as multichannel radio station programming"

[19] Moere A V (2004). "Time-Varying Data Visualization Using Information Flocking Boids".

第5章　質疑応答

Q.artisoc の空間はなんですか？

A. アフィン空間（ベクトル空間）というもの．アフィン空間というのは幾何

ベクトルの存在の場であり，ユーグリット空間から絶対的な原点，座標と標準的な長さや角度などといった計量の概念を取り除いたアフィン構造を抽象化した幾何学的構造である．

（代数的な）ベクトル空間からどの点が原点であるかを忘れたものと考えることができる．[14]

Q. エージェントが値を蓄積する？

A. エージェントそれぞれに値が蓄積される．エージェント一つ一つにそれぞれの値が蓄積され，各エージェントがどのエージェントが好きか嫌いかを決める．