人　工　知　能 95−11
(1994.　7.　21)

2次元平面内を移動するエージェント群の挙動の解釈の生成

車谷浩一，中村真理
電子技術総合研究所
kurumatani@etl.go.jp, mari@osaka.etl.go.jp

平面上を移動し，お互いに協調して系全体の目的を達成するために作業を行なうマルチエージェントシステム（蟻のコロニー，自動車の社会，工場のロボット群，マイクロマシンシステムなど）が，時間の経過とともにどのような挙動を示すか，また何故そのような挙動を示すかという，挙動の解釈・説明の自動生成の構想について述べる．具体的には，1) 数値シミュレーションにより得られる解の特徴の抽出，2) 偏微分方程式系を扱う定性推論システムによる大局的な挙動の推定，を組み合わせ，自然言語による解釈・説明の生成を目指す．

GENERATING INTERPRETATION OF MULTI-AGENTS BEHAVIOR MOVING IN TWO-DIMENSIONAL PLANE

Koichi Kurumatani (Cooperative Architecture Section)
Mari Nakamura (Life Electronics Research Center)
Electrotechnical Laboratory
kurumatani@etl.go.jp, mari@osaka.etl.go.jp

A multi-agents system's behavior can be very complex, though the algorithm of each agent is quite simple. We propose a new method of generating the explanation of a multi-agents system's behavior, e.g., how and why the system shows such a behavior. The system consists of three components: 1) numerical simulator with function value supervisor, 2) qualitative reasoner with region grammar which handles partial differential equations, 3) explanation generator. The method is expected to become one basis of a self-reorganization mechanism of the multi-agents system's configuration and behavior algorithm.

1 はじめに

平面上を移動しながらお互いに通信し，相互の協調動作によって全体としての目的を達成するようなマルチエージェントシステムを考える．例えば，蟻のコロニーに代表されるような社会性昆虫は，各個体は単純なアルゴリズムに従って行動しているのにもかかわらず，コロニー全体として，餌の採集，繁殖，育児などの高度な作業を効率良く実行することができる．また自動車の社会，工場のロボット群，マイクロマシンシステムなどのシステムでは，各個体の行動は単純でも全体としての行動は十分に複雑であり，各個体の協調が適切に行なわれれば全体として効率良く動作出来るが，さもなくば非能率的なものとなってしまう．

筆者らは，1) 系全体としての目標が与えられたとき，それを達成するための各個体の行動の決定，2) 環境に何らかの変化があった場合，系全体としての目標を引続き達成できるような各個体の行動の修正，といった意味でのマルチエージェントシステムの自己再編機能について研究を進めているが，その第一歩として本稿では，「各個体がどのように振舞っているのか？」「全体としてどのような行動が行なわれているのか？」そして「それらは何故起こっているのか？」といった挙動の解釈・説明を自動生成する一つの構想について述べる．

従来，複雑なシステムの解析は，各個体の行動を規定する方程式群を数値シミュレーションで解き，時間とともにシステムがどのように変化していくかを観察することによって行なわれてきた．さらに，近年飛躍的に発展したコンピュータグラフィクスを利用したシミュレーション結果の表示は，視覚を通した直感的な理解の助けとして大いに有効ではあるが，「対象システムにおいていったい何が起こっているのか？」ないしは「なぜそのような挙動を示すのか？」といった説明ないしは解釈は人間の手に委ねられている．

本稿の内容は，数値シミュレーションの結果の解釈・説明を定性推論の手法を用いて自動的に生成する手法であり，いわば「数値シミュレーション結果の自然言語による説明」と言えるものである．

なお本稿の以下の部分では「解釈 (interpretation)」と「説明 (explanation)」とを特に区別せず，同様な意味で用いている．

2 蟻コロニーの採餌行動のアルゴリズム

例題として蟻のコロニーの採餌行動（集団で餌を見つけ出し運搬する行動）を取り上げ，その説明の生成を目標とする．この例題では，各個体は実に単純なアルゴリズムで動作していることが知られているが，その集団的振舞いは十分に複雑なものである．

本稿では各蟻の行動アルゴリズムを，以下のように単純化して用いる．

[個体レベルの行動アルゴリズム]

- 各蟻は「探索」「誘引」「輸送」のいずれかのモードを取る．
- 通常は探索モードにあり，ランダムウォークをしている．
- 餌を見つけると輸送モードに入り，餌を巣に運ぶ（巣に向かっての直線的な移動）．この際に，自分が移動している経路上に液体の集合フェロモンを残す（このように平面内に描かれたフェロモンの軌跡をトレイルと呼ぶ）．巣に戻った輸送モードの蟻は，探索モードに入る．
- 液体フェロモンは時間とともに蒸発し，気体フェロモンとして拡散する．
- 探索モードにある蟻は，ある一定値以上の気体フェロモン濃度の場所に達すると誘引モードに入り，濃度の高い方向へと移動する．さらにトレイルに遭遇すると，トレイルを辿って餌へと向かう．

これによりコロニー全体としては，

[系レベルでの行動（アルゴリズム）]

- 餌場を見つける．
- 複数の蟻を動員する．
- 餌がなくなるまで輸送を行なう．

を実現している．

3　解釈・説明の生成

3.1　基本的な考え方

以上のようなアルゴリズムで個体を動作させると，系全体の挙動は一般に極めて複雑なものとなり得る．一例を挙げると，「個体が動員されることによって，一か所の餌場に加速度的（指数関数的）に個体が集まる．これにより餌場が複数存在するにも拘らず，一か所の餌場に個体が集中してしまい，他の餌場の存在が忘れられてしまう」といったものである．

このような，挙動の解釈の自動生成を行なうため，本研究では，

1. （数値）解の特徴の抽出
   数値計算によって挙動シミュレーションを実行し，その解の特徴を抽出する．
2. 偏微分方程式系を扱える定性推論
   偏微分方程式で記述される広がり（空間）を扱える定性推論の手法によって，大局的な挙動を推定する．
3. 解釈の生成
   この2種類の情報を統合して解釈・説明の生成を行なう．

というアプローチを取る．特に 2) の手法は，従来の定性推論の偏微分方程式系への拡張である．図1に本研究のアプローチの全体図を示す．

3.2　数値解の特徴の抽出

微分方程式系の数値シミュレーションの基本的なアイデアは，積分・微分といった関数関係を四則演算によって近似し，それらの繰り返し演算によって挙動を近似することである．近似とはいえ，よく配慮された数値シミュレーションは十分な精度で現実を模倣することができる．

本研究では，数値シミュレーションの結果から解釈を生成するのではなく，その実行中において必要な情報を抽出するメカニズムを組み込むことにより，対象システムの挙動の中から部分的な解釈を生成する．具体的には，数値シミュレーションを実行するプログラムに「監視ルーチン」を組み込み，解の特徴の抽出を行なう．ここでいう解の特徴とは，解関数の極値の位置とその移動，閾値を定めたときの解関数の形状とその変化，などである．

3.3　偏微分方程式系を対象とした定性推論

常微分方程式系（いわゆる「力学系」）で記述できる，比較的単純な対象システムに関しては， QSIM [1] に代表されるような定性推論の手法を用いて対象システムの挙動の記号表現を生成し，それとは並列に実行される数値シミュレーションの結果の説明を生成する手法が，既に研究されている [2].

従来の常微分方程式系を対象とする定性推論では，系を記述する変数の値として定性値（‘境界標’とそれに囲まれた‘区間値’）を用いる．系は有限個の変数で記述されるため， QSIM のような生成検査法 (generate and test) で挙動推論が行える．

一方偏微分方程式系では，常微分の変数に相当するものが関数となる．この関数の独立変数は，通常，時間変数ならびに空間的な広がりを表現する位置変数である．すなわち偏微分方程式系の場合には，常微分での変数が無限個存在するという状況にあり，生成検査法を用いた従来の手法の適用は難しい．

本研究では，与えられた定性値に対応する空間領域を表現する記号レベルでの表現（領域文法）を用意し，そのような領域の変化を推論する手法によって，この問題を解決する．

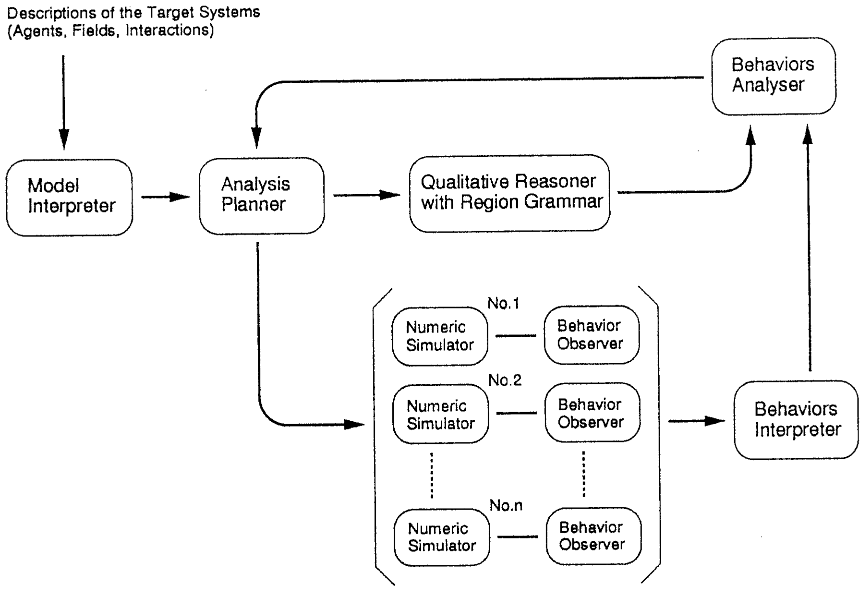

図１． 挙動解析全体の処理の流れ

4 領域文法による定性推論

偏微分方程式系の表現の基礎となる空間的広がりを定性的に扱うための手法として，本研究では領域 (region) と呼ばれる記号的表現を用いる．すなわち２変数 x, y の関数 $f(x, y)$ が与えられたとき，なんらかの意味ある値 l （境界標：landmark）を関数値としてとるような x, y の領域（つまり $f(x, y) = l$ なる xy 平面に平行な平面領域）に注目し，その記号的表現を用意する．

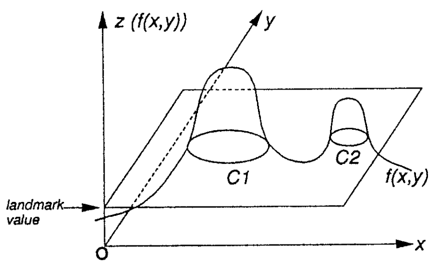

図２． 領域 (region) の定義

一般にこのような平面領域は 1) 閉曲線 C， 2) 点 P， 3) 空集合 Φ，のいずれかとなる． 2), 3) の扱いは簡単であるから除いて考えると，関数 $f(x, y)$ の時間変化は，閉曲線 C の変化で表すことが可能である．

領域文法 (region grammar) とは，このような閉曲線 C の変化の様子を定性的に記述するための枠組にほかならない．本研究では閉曲線の変化を記述するために，領域間の静的な関係として同一／分離／共有／包含など，動的な関係として生成／消滅／拡大／縮小などを用意し，各々述語で記述する．

[領域文法 (region grammar)]

○ 領域の定義

```
region(r, a, v)
```

r は a なる関数が v なる定性値を取るような領域である．このような領域が非連結な場合には，各々独立した領域として扱う．

○ 領域間の静的関係
（2つの領域間の空間的関係）

```
identical(ra, rb)      同一
isolated(ra, rb)       分離
intersect(ra, rb)      共有部分を持つ
include(ra, rb)        包含する
broader(ra, rb)        面積が広い

broader(Ra, Rb) :- include(Ra, Rb).
```

○ 領域の動的変化

```
 (t1 に比べ t2 において)
expand(r, t1, t2)      拡大している
shrink(r, t1, t2)      縮小している
no_change(r, t1, t2)   同一

 (時刻 t1 において)
expanding(r, t1)       拡大しつつある
shrinking(r, t1)       縮小しつつある

 (時刻 t0 において)
appear(r, t0)          生成する
disappear(r, t0)       消滅する

no_change(R, T1, T2) :- constant(R).
expand(R, t1, t2) :-
  broader(region(R, t2),
          region(R, t1)).
expand(R, t1, t2) :-
  include(region(R, t2),
          region(R, t1)).
no_change(R, t1, t2) :-
  identical(region(R, t1),
            region(R, t2)).
expanding(R, t1) :-
  expand(R, t1, t1 + delta_t).
```

5 領域の変化に関する推論と解釈の生成

対象システムは従来の定性値と領域文法を用いて記述され，定性値と領域の変化としてその挙動が推論される．定性値の変化に関してはQSIM のような生成検査法を用いる．

領域の変化も基本的には生成検査法を用いる．まず領域の変化の可能性を，

G1) 定性値と領域との関係

G2) 問題の定義から得られる情報

から生成する． G1) としては「輸送モードの蟻が存在すれば，トレイルを表す領域が存在する」などであり， G2) は対象システムに関する問題定義の一部としての記述から翻訳される．例えば「誘引モードの蟻が餌場に達すると輸送モードに変化する」は以下のように記述される．

```
appear(a_trns) :-
 region(b, bates, plus),
 region(a_att, attracted_ants, plus),
 region(a_trns, transport_ants, plus),
 intersect(a_att, b).
```

次に，生成された領域の変化の可能性を，

T1) 数学的拘束条件

T2) 問題によって与えられた拘束条件

によりフィルタリングする． T1) の数学的拘束条件による拘束条件とは， QSIM の test 部で用いられる拘束条件を領域文法に拡大したものに相当する．これは対象システムを支配している方程式群によって与えられる制約を拘束条件として記述したものであり，対象システムに依存しない．例えば「関数 d は単調に拡散していく」は以下のように記述される．

```
shrinking(R, _) :-
 region(R, F, plus),
 function_type(F, diffusion).
```

T2) は G2) と同じく問題定義から翻訳されるものである．

G2), T2) は単なるヒューリスティクスというよりも，領域文法の表現力ならびにその上で表現される数学的拘束条件によっては制御できない定性的な冗長性を解消するために用いる．また非連続的変化のような，生成検査法での処理に向かない変化の表現・推論にも用いる．

5.1 蟻コロニーの採餌行動の例題

本稿での例題である蟻コロニーの挙動の推論は以下のように行なわれる．蟻の密度を表現するため，各モード毎に蟻の密度を表す領域を用意する．境界標として，探索モードの蟻に関しては平均密度 dm，輸送・誘引モードは 0 を用いる．また餌場・巣を表す領域を用意し，境界標を 0 とする．

初期状態は以下のように表現される (時刻 $t0$).

```
region(b, bates, plus).
region(n, nest, plus).
appear(b, t0). appear(n, t0).
constant(n).    /* 巣は変化しない */
isolated(b, n).
```

時刻 $t1$ において初めて餌場に蟻が到達すると[1]，その蟻は餌場と巣の間にトレイルを形成する．その後トレイルはフェロモンとして拡散する．

```
region(d, pheromone_dense, plus).
appear(d, t1).
/* フェロモン濃度の高い領域が出現 */
include(d, b). include(d, n).
```

このフェロモンに誘引され蟻が集まってくる．それらの蟻は餌場にたどり着くと，輸送モードへと変化する．

```
region(a_att, attracted_ants, plus).
appear(a_att, t2).
  . . .
region(a_trns, transport_ants, plus).
appear(a_trns, t3).
```

このような定性推論の過程を辿ることにより，対象システムの挙動に関する説明を生成することができる．例えば以下のようになる．

[1]この情報は，並列に実行されている数値シミュレーションの途上で検出されるが，問題の定義から「ランダムウォークを行なっている探索モードの蟻は，いつかは餌に到達するはずだ」というヒューリスティクスを用いて推論することも可能である．ただしこの推論は，本稿で扱う定性推論の範囲外である．

探索モードの蟻が餌場に到達する →
その蟻が輸送モードに入り，
餌場と巣の間にトレイルを形成する →
トレイルが蒸発しフェロモンが拡散する →
探索モードの蟻がフェロモンに触れ，
誘引モードに入る →
. . .

定性推論の実行の途上においては一般に枝分れが生じ，その各々の経路は全て正しい解釈の可能性がある．並列に実行されている数値シミュレータに組み込まれた監視ルーチンが生成する解の特徴と定性推論の結果を照合することにより，数値シミュレーションの正しい解釈を選択する．

現在のところ，領域文法の表現力が不足しているために定性的には異なる状態として表現されるべき状態が区別できないという問題があり，今後の課題である．

6 まとめ

数値シミュレーションと偏微分方程式系を扱える定性推論の手法とを併用し，平面上を移動するマルチエージェントシステムの挙動の解釈を自動生成する構想について述べた．

今後の課題としては，1) 領域文法の表現力を高めより精密な定性表現を行なうこと，2) 数値シミュレーションの結果を用いた，定性推論で得られる推論結果の正当性の検証，3) 挙動の説明結果による，個体の行動アルゴリズムの自動修正，などが挙げられる．

参考文献

[1] B.J. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[2] K.D. Forbus and B. Falkenhainer. Self-explanatory simulations: An integration of qualitative and quantitative knowledge. In *Proc. of AAAI-90*, pages 380–387, 1990.