プロダクション規則と局所評価関数による最適化の方法とその計算過程におけるマクロなふるまい*

金田　泰**
新情報処理開発機構

著者は自己組織的計算のための「化学的キャスティング・モデル (CCM)」とそれにもとづく計算言語 SOOC を提案している．このモデルは仕様も明確にかきくだせない開放系の問題への適用を本来の目的として開発した．しかしこれを仕様が明確な最適化問題に適用しても，巡回セールスマン問題などの単純な最適化問題のばあい，唯一のプロダクション規則と唯一の評価関数 (局所秩序度という) をあたえるだけで，局所的な情報の参照だけにもとづいて問題がとけるという利点がある．この報告では CCM にもとづく最適化の例として巡回セールスマン問題をとりあげ，解法と実験結果とをしめす．また，この方法による計算過程におけるマクロなふるまいを理解するため，マルコフ連鎖にもとづくモデルにあてはめた結果をしめす．

1. はじめに

金田 [Kan 92a, Kan 92b] は開放系 (環境に対してひらかれたシステム) における仕様も明確でない現実世界の問題をとくための自己組織的計算をめざして，化学的キャスティング・モデル (Chemical Casting Model, CCM) という計算モデルを提案している[注1]．CCM はプロダクション・システムにもとづくモデルだが，従来のそれとはちがって局所秩序度という一種の評価関数と非決定性 (確率的制御) をとりいれている．局所的情報だけにもとづいて動作するプロダクション規則と局所秩序度からなる単純かつ汎用的な “プログラム” により，自己組織的にすなわちミクロ・レベルとマクロ・レベルとのあいだの相互作用によって [Pri 77, Hak 78]，“大域的な秩序” をつくりだすことをめざしている．またこのモデルによって，遺伝的アルゴリズムなどと同様に記号的計算とパタン的計算との中間領域の開拓をめざしている．

CCM はミクロ・レベルで計算を駆動するためのミクロ・モデルだが，そこに (数学的な意味での) 位相あるいは距離を導入してマクロ・モデルとくに確率過程論にもとづくモデルを構築し，システムのふるまいをマクロ・レベルでみることを可能にすることを意図している．マクロ・モデルの導入で意図していることはいくつかあるが [Kan 93]，ここでは計算の自動制御の問題についてだけのべておく．現在の計算システムは，わずかなバグや外部からのノイズ (不正な入力など) といった攪乱によって不正な結果がえられたり，システム・ダウンをおこしたりというように，重大な影響をうける．自然のシステムや自動制御システムでは通常このようなことはおこらない．それは，これらのシステムにおいてはマクロな量にもとづいてフィードバック制御がおこなわれているからである．これらのシステムが安定点の近傍で動作しているときには，マクロな量が攪乱によっておおきな影響をうけることはない．このようなすぐれた特性を計算システムにももたせる方向に研究をすすめることは重要だとかんがえられる．

CCM は上記のように開放系における自己組織的計算をめざして開発されたモデルだが，現在のところは古典的な制約充足問題や最適化問題への適用をこころみている段階である．制約充足問題に関しては，すでに N クウィーン問題とグラフ彩色問題について報告した [Kan 92a, Kan 92b].

この報告では CCM の最適化への適用についてのべる．最適化においても，アルゴリズムを記述せず，できるだけ単純かつ汎用的な “プログラム” によってロバストな最適化をおこなうことをめざしている．この点では，この研究はニューラル・ネットや遺伝的アルゴリズムによる最適化と同様の方向をめざしている．第 2 章では準備のため，かんたんに CCM について説明する．第 3 章では，CCM にもとづく最適化の例として巡回セールスマン問題をとりあげる．第 4 章ではその実験結果についてのべるとともに，それをマルコフ連鎖モデル [Kan 92b] にあてはめて，その計算過程におけるマクロなふるまいをし

---

* この研究の一部は著者が日立製作所中央研究所においておこなったものである．

** E-mail: kanada@trc.rwcp.or.jp

注1 このモデルは金田 [Kan 92a] においては「化学的プログラミング・モデル」とよばれていた.

2. 計算モデル CCM

この章では，化学的キャスティング・モデルについて説明する．CCM は化学反応系とのアナロジにもとづく計算モデルであり [Kan 92a]，不完全な情報，あるいは非決定的な計画にもとづく計算のためのモデルである．

CCM の構成要素についてかんたんに説明する．CCM はプロダクション・システムにもとづくモデルである． OPS5 [For 81] のような古典的なプロダクション・システムにおける作業記憶に相当する短期記憶は，CCM においても**作業記憶**とよぶ．すなわち，CCM が作用するデータは作業記憶にふくまれる．そして，プロダクション・システムにおける規則ベースすなわちプログラムに相当するものを**キャスタ**とよぶ．CCM は不完全な計画にもとづく計算のためのモデルなので，完全な計画を意味するプログラムということばのかわりにキャスタということばを使用する．いまのところ，キャスタはユーザによって記述され，そのままのかたちでつかわれる．

作業記憶にふくまれるべき**オブジェクト**あるいは**データ**としては，つぎのようなものがある (**図** 1 参照)．**原子**はデータの単位であり，内部状態をもつ．原子にはデータ型があり，それを**元素**ともよぶ．原子どうしを**リンク**によって結合することができ，結合された全体を**分子**とよぶ．リンクは無向でも有向でもよい．リンクの存在は通常のプロダクション・システムにない CCM の特徴のひとつである．無向のリンクは化学結合に似ているが，化学結合には有向のリンクに相当するものはない．また，リンクにはラベルをつけることもできる．

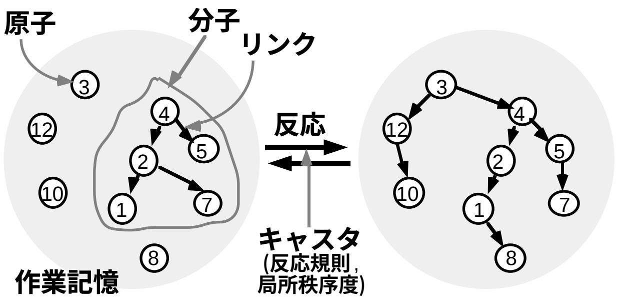


図 1 **化学的キャスティング・モデルの構成要素**

キャスタは反応規則と局所秩序度とで構成される．**反応規則**はシステムの局所的な (ミクロな) 変化のしかたをきめる規則であり，ユーザにより定義される．ここで「局所的」ということばは，その反応規則によって参照される原子数がすくないということを意味する[注2]．反応規則は前向き推論によるプロダクション規則として記述される．したがって，つぎのようなかたちをしている．

LHS → RHS.

反応規則は化学反応式に相当するものだといえる．後述する *N* クウィーン問題やグラフ彩色問題 [Kan 92b] などをはじめとするおおくの単純なシステムにおいては反応規則は 1 個だけ存在するが，複数の変化のしかたをみとめるより複雑なシステムにおいては複数個の反応規則が存在する．

局所秩序度は局所的な “組織化” あるいは “秩序化” の程度をあらわす一種の評価関数であり，作業記憶の局所的な状態が “のぞましい” ほどおおきな値をとるように，ユーザにより定義される．局所秩序度の存在は，通常のプロダクション・システムにくらべたときの CCM のもっともおおきな特徴である．局所秩序度はつぎの 2 つのうちのいずれかのかたちで定義される．

(1) **自己秩序度** *o*(*e*)
1 個の原子 *e* に対して定義される．

(2) **相互秩序度** *o*(*e1*, *e2*)
2 個の原子からなる対 <*e1*, *e2*> に対して定義される．

後述の *N* クウィーン問題のキャスタは相互秩序度を使用しているが，以下の説明においては，かんたんのため自己秩序度だけをかんがえる．自己秩序度は規則の適用時に原子ごとに計算されるが，その値は当該原子の内部状態だけでなく，そこからでるリンクがつながったさきの原子の状態にも依存しうる．

反応はつぎの 2 つの条件をみたすときにおこる．反応規則の左辺 LHS および右辺 RHS には原子とマッチする 1 個または複数個のパタンがあらわれるが，第 1 の条件は左辺にあらわれるすべてのパタンのそれぞれにマッチする原子が存在することである．

反応がおこるとこれらの原子は消滅して，そのかわりに右辺にあらわれる原子が生成される．ただし，左辺と右辺とに対応する原子があらわれるばあいは，その原子は生成・消滅するかわりにかきかえられる．このような規則とそれにあらわれる (左辺および右辺の) パタンにマッチするすべての原子との組を**インスタンス**とよぶ．ひとつのインスタンスがふくむ原子のうち，反応前に存在するものすなわち左辺にあらわれるものの局所秩序度の総和を “反応前のインスタンス秩序度”，反応後に存在するものすなわ

[注2] CCM においては，化学反応系のように (物理的な意味での) 距離の概念が導入されていないから，局所的ということばは距離がちかいということを意味しない．

ち右辺にあらわれるものの局所秩序度の総和を “反応後のインスタンス秩序度” とよぶ．反応後のインスタンス秩序度をあらかじめ計算したものが反応前のインスタンス秩序度よりおおきいとき，すなわち反応によって局所秩序度の和が増加する時だけ反応がおこるというのが第 2 の条件である．

そして，いずれかのインスタンスについて上記の 2 条件がみたされているかぎり，反応はくりかえしおこる．これらの条件をみたすインスタンスが存在しなくなると実行は中断される．

ただし，一般には上記の 2 つの条件をみたすインスタンスは複数個存在する．条件をみたすインスタンスが複数個生成される原因としては，ひとつの規則の条件部をみたす原子の組が複数個存在するばあいと，複数の規則についてその条件部をみたす原子の組が存在するばあいとがある．いずれのばあいでも，これらのインスタンスのうちのいずれがどのような順序で，あるいは並列に反応するかは非決定的であり，反応の順序はシステムが自発的にきめる．このような自発性あるいは非決定性を CCM にあたえているひとつの理由は，非決定性のないアルゴリズミックな計算においては，プログラマがあたえた “よけいな制御” によってプログラムの動作が制約され，自己組織的な計算や並列度のたかい計算を不可能にしているばあいがあるとかんがえられるからである [Kan 92b].

しかし，反応の順序をある程度はユーザが制御することができないと，のぞんだ計算を実現できないばあいがある．ユーザは**スケジュリング戦略**というものを指定することによってインスタンスの選択順序を制御し，反応の順序を部分的に制御することができる．スケジュリング戦略にはインスタンスを系統的に選択する**系統的戦略**と，ランダムに選択する**ランダム戦略**とがある．スケジュリング戦略について，くわしくは金田 [Kan 92a] を参照されたい．

3. CCM による TSP

CCM にもとづくシステムによる最適化の例として，2 次元ユークリッド空間の巡回セールスマン問題 (Travelling Salesperson Problem. 以下 TSP と略す) の近似解をもとめるためのシステム (TSP **ソルバ**とよぶ) をしめす．10 都市のばあいの作業記憶の初期状態と終状態の例を**図** 2 にしめす．都市は原子として表現され，(図にはしめさないが) 内部状態としてその座標 (x, y) をもっている．巡路は都市を *next* というなまえのリンクでつなぐことによって表現される．初期状態は，全都市をランダムに接続して巡路をつくった状態か，または図 2 にしめしたようにただひとつの都市が巡路にふくまれる状態のうちのいずれかとする．後者のばあいでももちろん，終状態では全都市が巡路にふくまれるようにする．

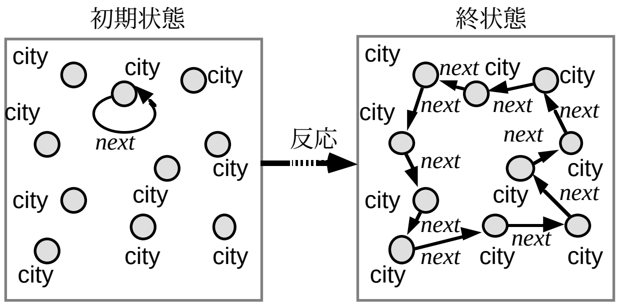


図 2　10 **都市の** TSP **の初期状態と終状態の例**

キャスタは 2 個の規則と 1 個の局所秩序度とによって構成される．第 1 の規則はここにはしめさないが，それは巡路にランダムに 1 個の都市を挿入する規則である．すなわち，この規則が適用できなくなるまで反復実行することによって，巡回セールスマン問題が要求する制約が充足される．初期状態において巡路に全都市がふくまれるばあいは，この規則は不要である．第 2 の規則を**図** 3 にしめす．これは巡路を最適化するための規則である．この規則は巡路中の 1 都市を他の位置に移動させる．図 3 の左辺 (左側) が条件部であり，実線でかかれた 3 個のパタンに都市がマッチする．図 3 の右辺が動作部であり，マッチした都市のリンクが図のようにかきかえられる．図 3 の規則は左辺と右辺とが対称であるから，可逆である．すなわち，右辺を条件部，左辺を動作部とみなしてもまったく同様に機能する[注3].

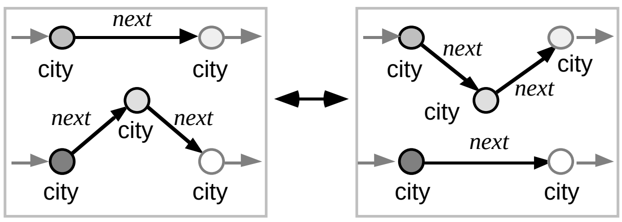


図 3　**巡路の最適化のための反応規則**

都市をあらわす原子に関する局所秩序度は自己秩序度 o_{city} として定義する．o_{city} はその都市からリンクされた都市までの距離に負号をつけたものをあらわす．ただし，リンクが存在しないばあいは o_{city} は負のおおきな値をとる[注4]．すなわち o_{city} はつぎのように定義される．

[注3] 局所秩序度の存在により可逆な規則によって計算できることが，CCM のひとつの特徴である．

[注4] このようにするのは，すべての都市が巡路にとりこまれるようにするためである．

$$o_{\text{city}}(c) = -\,sqrt\big((c.x - c.next.x)^2 + (c.y - c.next.y)^2\big) \quad \textbf{when } c.next \neq \textbf{nil}$$
$$-\infty \quad \textbf{when } c.next = \textbf{nil}$$

作業記憶全体にわたる局所秩序度の和を**大域秩序度**とよぶが，上記のように局所秩序度を定義すると，全都市が巡路にとりこまれた状態においては大域秩序度は巡路のながさに負号をつけたもの，すなわち最大化すべき量に一致する[注5]．なお，計算言語 SOOC-93 によって記述したキャスタを**付録**にしめす．

ある原子または原子対に関する局所秩序度を増加させることが大域秩序度の減少をひきおこすばあいがあるシステム (キャスタと作業記憶をあわせたもの) は**競合型**であるという．また，局所秩序度が増加するときには大域秩序度が減少することがないシステムは**協調型**であるという．上記の TSP ソルバは協調型である．なぜなら，ある隣接する 2 都市の距離をかえても，それによって他の 2 都市のあいだの距離がかわることはないからである．最適化問題ではないが，*N* クウィーン問題やグラフ彩色問題のばあい [Kan 92a, Kan 92b] にはこのような条件がなりたたない．したがって，これらのシステムは競合型である．これらの競合型のシステムはシミュレーテッド・アニーリングのような特性をもっていることがわかっている [Kan 92a]．この特性は解探索のためにはのぞましいとかんがえられるが，協調型のシステムはこの特性をもっていない．

次章でしめすように，上記のキャスタにより TSP の近似解をもとめることができる．また，このキャスタをつかえば，いったん近似解がもとめられて計算が中断されたあとも，外部から都市の追加，既存の都市の座標変更などがおこなわれれば，(ばあいによってはそれまでとは逆方向に) 再動作してあらたな近似解がもとめられる[注6]．しかしこのキャスタには，おおくのばあい巡路の交叉が除去できないという問題点がある．**図** 4 の (1) のように交叉したさきが 2 点だけからなるばあいは第 2 の規則を適用することによって交叉が除去できるが，これ以外のばあい，たとえば図 4 の (2) のようなばあいには交叉は除去できない．この問題については第 5 章でよりくわしくのべる．

[注5] 大域秩序度は CCM の計算 (処理系) じたいにおいては計算されないということを念のためにかきくわえておく．

[注6] 規則の動作が局所秩序度によってきめられ，規則じたいは可逆でもよいということが，インクリメンタルに動作 (漸動性 [Kan 92a]) しやすくしているといえるであろう．

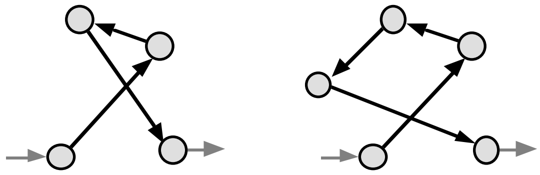


図 4　TSP **ソルバにおける交叉の除去可能性**

4. 巡回セールスマン問題の実験結果

前章でしめした TSP ソルバを CCM にもとづく計算言語の処理系 SOOC-92 (Self-Organization-Oriented Computing) によって記述し実行させた．SOOC-92 は Common Lisp によって記述されていて，逐次処理をおこなう．その結果を計算時間，解の質という点から評価するとともに，マクロなふるまいを記述するマルコフ連鎖モデル [Kan 92b, Kan 93] にあてはめる．

4.1 計算時間の評価

N 個の都市を正方形の領域に配置し，その座標をランダムにきめたばあいの反応回数 (規則の実行回数)，規則左辺のマッチング回数および SUN4 における計算時間を *N* を変化させて測定した結果を**図** 5 にしめす．スケジュリング戦略としてはランダム戦略を使用している．計算時間はほぼマッチング回数に比例し，$O(N^3)$ であることがわかる．

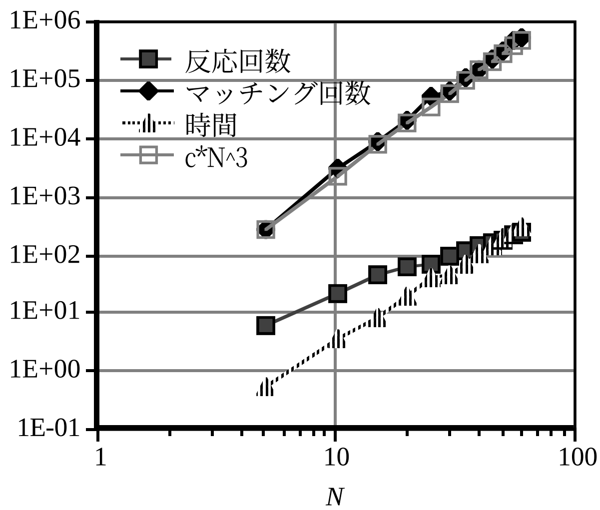
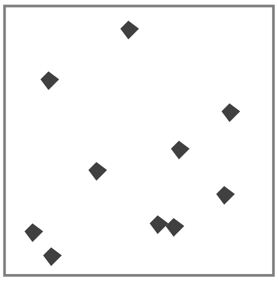


図 5　*N* **都市** TSP **の計算時間** (SUN4)

4.2 解の質の評価

TSP の解の質を一般的に評価するのは困難なので，**図** 6 にしめす 10 都市の例題 [Hop 85, Mur 91] と 20 都市の例題 ([Mur 91] の都市配置 (c)) について 100 回の計算をおこない，巡路長の分布をしらべた結果をしめす．10 都市の例題に関しては 97 回は最適解

がえられ，のこりの 3 回も最適解に非常にちかい解がえられた．よい結果がえられたのは，問題が比較的容易だからであろう．



都市の座標
(0.1707　0.2293)
(0.2293　0.7610)
(0.2439　0.1463)
(0.4000　0.4439)
(0.5171　0.9414)
(0.6195　0.2634)
(0.6683　0.2536)
(0.6878　0.5219)
(0.8488　0.3609)
(0.8732　0.6536)

図 6　実験に使用した 10 都市 TSP

20 都市の例題の結果を**図** 7 にしめす．最左端の棒が最適解かそれに非常にちかい近似解がえられた頻度をあらわす．各棒のうち色のうすい部分は交叉をふくむ解がえられた頻度である．この問題のばあいには解の質はよいとはいえない．交叉が除去できないことが解の質を低下させるおおきな原因となっていることがわかる．このばあいにかぎらず，前記のキャスタは，これまでに実験したかぎりでは都市数がおおいばあいに質のよい解をあたえない．

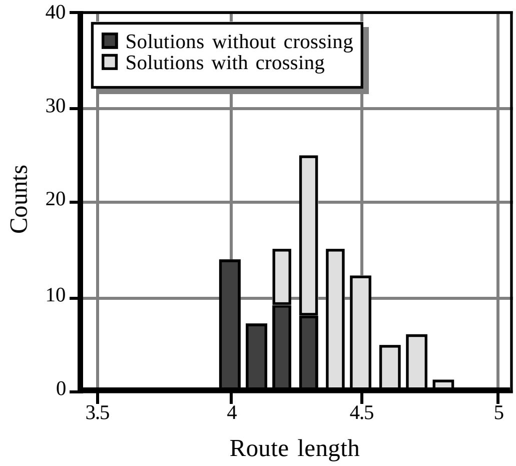


図 7　20 都市 TSP の計算結果における巡路長分布

4.3 マルコフ連鎖モデルのあてはめ

金田 [Kan 92b] は CCM にもとづく計算過程のマクロなふるまいを説明するためのモデルとして，離散状態離散時間のマルコフ連鎖モデルを提案している．このモデルでは，大域秩序度 O が $0 \le O \le \mathrm{I}$ という範囲の整数値をとるばあいに，O が時刻 t に i という値をとる確率を $\mathrm{p}(O(t) = i)$ $\left(\sum_{i=0}^{\mathrm{I}} \mathrm{p}(O(t) = i) = 1\right.$ がなりたつ$\left.\right)$ とし，$\mathrm{p}(O(t) = i)$ $(i = 0, 1, \ldots, \mathrm{I})$ を要素とするベクトルを $\mathbf{p}_t$ とする．このとき，$\mathbf{p}_t$ に関してマルコフ性がなりたつ，つまり $\mathbf{p}_t$ と $\mathbf{p}_{t+1}$ とのあいだにつぎのような関係がなりたつと仮定する．

$$\mathbf{p}_{t+1} = \mathrm{T}\,\mathbf{p}_t .$$

ここで遷移行列 T は I 行 I 列の行列であり，その値は時刻にはよらないとする．

上記のモデルは大域秩序度が一定範囲の離散値をとるばあいにしか適用できない．ところが，TSP においては大域秩序度が連続値をとるうえ，その最大値はあらかじめわからない[注7]．TSP に適用するためにはこのモデルを連続状態のばあいになおさなければならない．しかし，理論上は連続状態としてあつかっても，計算のためには離散化が必要になるとかんがえられる．そこで，つぎのようにして状態の離散化をはかることにより，前記のマルコフ連鎖モデルをあてはめる．すなわち，特定の問題をとく計算をくりかえしおこない，そこでえられた解の大域秩序度の範囲を適当にくぎる．各範囲をひとつのマクロな状態とかんがえることによって，計算過程を離散状態のマルコフ連鎖とみなす[注8]．

現在のところ，遷移行列 T は実測値から推定するほかはない (推定の方法は金田 [Kan 92b] がのべている)．したがって，マルコフ連鎖モデルが妥当であるかどうかをしらべるため，金田 [Kan 92b] が N クウィーン問題についておこなったのと同様に，つぎのような比較をおこなう．ランダムな初期状態における大域秩序度の分布を $\mathbf{p}_0$ とし，そこから計算をはじめたときの大域秩序度分布の時系列 $\mathbf{p}_0$, $\mathbf{p}_1$, … を実測値 (頻度) から直接推定したものと，これを $\mathbf{p}_0$ と T をつかって推定したものとを比較する (本来は後者においては $\mathbf{p}_0$ を理論的に予測するのがよいとかんがえられるが，この実験においては前者とおなじ値を使用している)．この実験においては前記の 10 都市の TSP を使用し，大域秩序度は実験でえられた最大値 –2.59 から最小値 –5.93 までのあいだを 20 のマクロな状態に分割した[注9]．

図 8 〜 9 は，大域秩序度のひくいほうから 10 個の状態について $\mathbf{p}_t$ の推定値をしめしている．図 8

[注7] ただし，都市の座標が固定されていることを仮定すれば，大域秩序度は離散値をとる．しかし，それらの値をかぞえあげることは最適解をもとめるより計算量がおおく，したがって一般にはできない．

[注8] このような近似をおこなうことによって生じる誤差を理論的に解析すべきだが，まだ解析していない．ただし，定性的にはつぎのようなことがいえるであろう．このモデルにあてはめることによる誤差としては，離散化による誤差，推定に使用するデータ数が有限であるための誤差 (ばらつき)，最大値・最小値の推定誤差などがある．

[注9] 状態をよりこまかく分割することにより離散化による誤差は減少するが，そうすると第 1 にばらつきによる誤差が増加し，第 2 にグラフに表現したときによみとりにくくなるため，20 個ときめた．

は実測値から直接推定した値であり，図 9 はマルコフ連鎖モデルをつかって推定した値である．各おれ線はひとつの状態をあらわし，それにラベルづけされた値はその状態における大域秩序度の最小値である．たとえばラベルが –5.93 の状態は，大域秩序度 O が $-5.93 \le O < -5.77$ という値をとる状態である．

図 10 〜 11 は，のこりの 10 個の状態について $\mathbf{p}_t$ の推定値をしめしている．図 8 は実測値から直接推定した値であり，図 9 はマルコフ連鎖モデルから推定した値である．図 10 と図 11 とをくらべると，最適解をふくんでいる状態への収束のはやさなどに多少のちがいがあるものの，大半の状態について，その確率がよく一致しているのがわかる．

以上の結果から，すくなくともこの問題についてはは大域秩序度の時系列がマルコフ連鎖により近似できると結論できるであろう．ただし，競合型の問題である N クウィーン問題やグラフ彩色問題のばあいとはちがって，定常確率過程にちかいふるまいをする**準定常状態** [Kan 92b] は存在せず，解がもとまって停止するすなわち定常状態に達するまで，つよい非定常性がある．また，この問題においてはほとんどのばあいに最適解またはそれに非常にちかい値に収束するということがいえる[注10]．これらの結論のうちとくに後者は，交叉が生じやすい，より都市数がおおいばあいにはなりたたない[注11]．

図 8 〜 11 から確率分布そのものをよみとるのがむずかしいので，**図** 12 〜 13 にこれらを 3 次元グラフとしてえがいている．各リボンが時刻 t における確率分布をあらわしている．

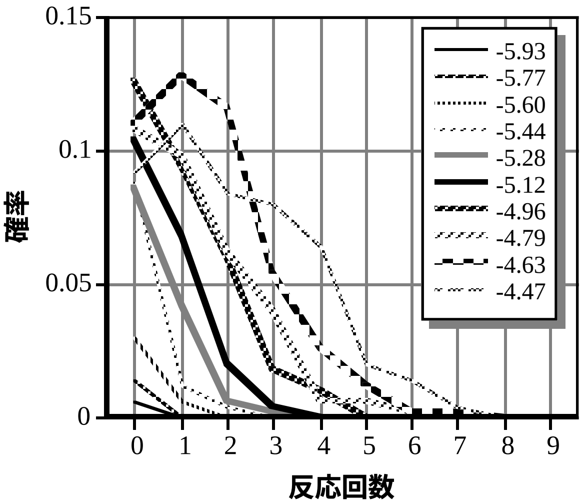


図 8　TSP **の計算過程における実測値から直接推定した大域秩序度の分布の変化** 1

[注10] より都市数がおおいばあいのマルコフ連鎖モデルの推定は現在試行中である．

[注11] このことは，図 6 をみてもあきらかである．

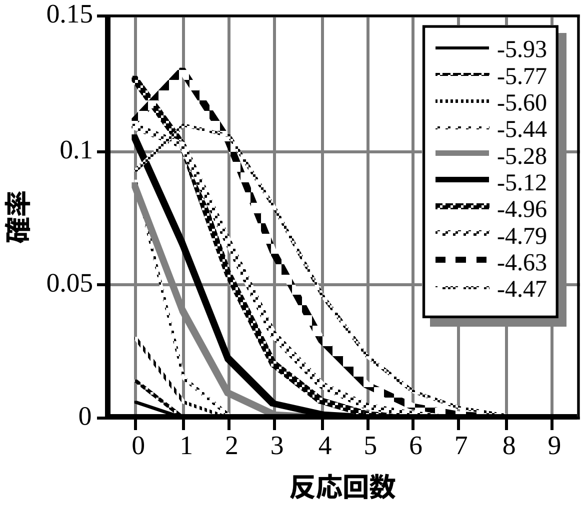


図 9　**マルコフ連鎖モデルから推定した大域秩序度の分布の変化** 1

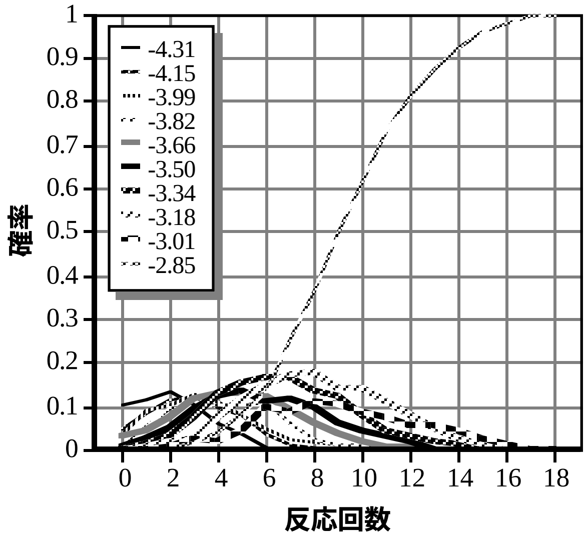


図 10　TSP **の計算過程における実測値から直接推定した大域秩序度の分布の変化** 2

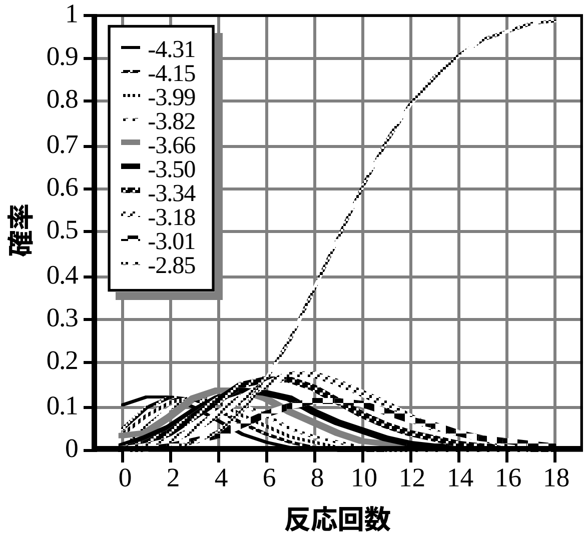


図 11　**マルコフ連鎖モデルから推定した大域秩序度の分布の変化** 2

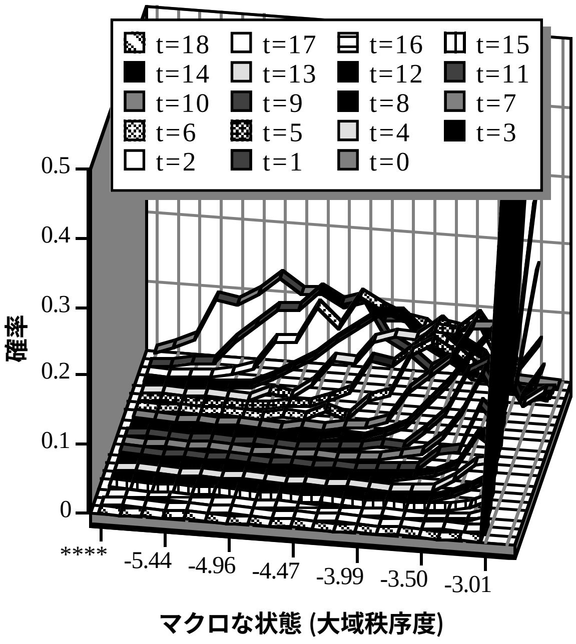


図 12 TSP **の計算過程における実測値から直接推定した大域秩序度の分布の変化**

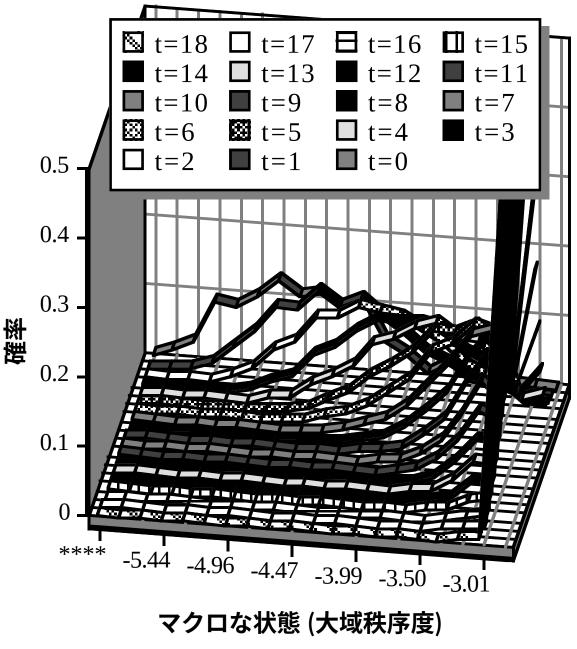


図 13 **マルコフ連鎖モデルから推定した大域秩序度の分布の変化**

5. 交叉の除去と大域的制約の保持

この章では，第 3 章でふれた交叉の問題をよりくわしく議論するとともに，それを拡大して CCM における大域的な制約の保持の問題について論じる．

第 3 章でしめしたキャスタにおいては都市をつなぐリンクにむきがあり，したがって巡路にもむきがあった．むきのある巡路において交叉をとるためには，**図** 14 のように非局所的なむきの変更が必要になるが，これを CCM のような局所計算のアプローチで実現するのはむずかしい．そこで，巡路のむきをなくすことをかんがえる．CCM においては，むきのないリンクを使用するか，または双方向のリンクを使用することによって，データ表現上むきをなくすことができる．ところが，このようにすると交叉を除去するときに**図** 15 にしめすような 2 つの可能性が生じる．これらのうちの一方はただしいが，他方を選択すると巡路を 2 つに分割させてしまう．ただしい選択がどちらであるかは，局所的な情報だけではわからない．

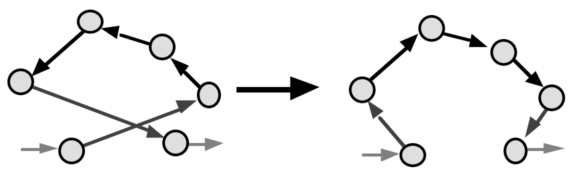


図 14 **交叉を除去するために必要な非局所的なむきの変更**

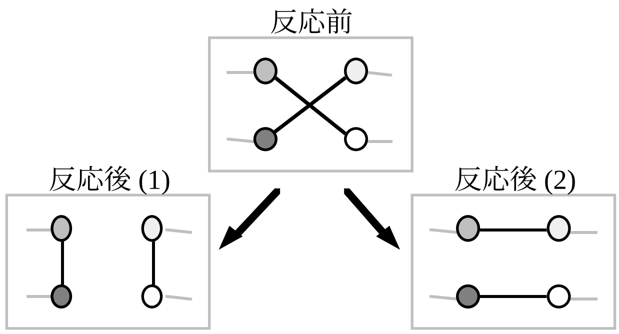


図 15 **むきのないリンクをつかった** TSP **ソルバにおける交叉の除去の** 2 **つの可能性**

交叉をとるばあいだけでなく，全都市がひとつの巡路にふくまれるという大域的な制約をみたしたまま巡路に自由な局所的操作をくわえることは困難である．この問題点は，CCM にかぎらず，ニューラル・ネットをつかっても，遺伝的アルゴリズムをつかっても，TSP，あるいはそれと似た大域的な制約をもつ問題をとくときにぶつかりやすい問題点だとかんがえられる (たとえば村島ら [Mur 91])．CCM において，大域的な情報を参照せずにこの問題点を解決する方法はまだわかっていない．

6. 結論

この報告では，TSP を例として，CCM による最適化についてのべた．CCM にもとづく計算言語とその処理系を使用すれば，TSP のばあいには，非常に単純な “プログラム” によって近似解をもとめることができる．また，この方法における計算過程は，マルコフ連鎖によって近似することができる．

しかし，CCM にもとづく計算は局所的におこなわれるため，大域的な制約をみたしつつ十分な最適化をおこなうのはむずかしい．TSP においてはこの問題点が，交叉の除去ができず，したがって解の質が低下するというかたちであらわれる．この問題点の解決策をみいだすことが，今後の重要な課題だとかんがえる．また，これまでに CCM を適用してきた問題は，制約充足問題にせよ最適化問題にせよ，閉鎖系の問題である．もともと CCM によってとくことをめざしていた開放系の問題に今後とりくんでいきたい．

謝辞

この研究をはじめるきっかけをつくっていただいた日立製作所日立研究所の 坂東 忠秋 部長，研究の継続をゆるしていただいている新情報処理開発機構の 岡 隆一 部長，マクロ・モデルにもとづく観測と制御というかんがえかたに示唆をあたえていただいた日立製作所中央研究所の 小島 啓二 氏，マルコフ連鎖モデルなどに関していろいろと議論していただいた同基礎研究所の 廣川 真男 氏，さらに第 33 回プログラミング・シンポジウム，同年の夏のプログラミング・シンポジウム，あるいはそのほかの機会に議論にくわわっていただいた，おおくの方々に感謝する．

参考文献

[For 81] Forgy, C. L.: *OPS5 User’s Manual*, Technical Report CMU-CS-81-135, Carnegie Mellon University, Dept. of Computer Science, 1981.

[Hak 78] ハーケン，H.: *協同現象の数理* (小森・相沢 訳), 東海大学出版会, 1980.

[Hop 85] Hopfield, J. J., and Tan, D. W.: Neural Computation of Decisions in Optimization Problems, *Biological Cybernetics*, Vol. 52, pp. 141–152, 1985.

[Kan 92a] 金田 泰: コンピュータによる自己組織系のモデルをめざして, *第 33 回プログラミング・シンポジウム報告集*, 1992.

[Kan 92b] 金田 泰: 自己組織系としての計算システム — ソフトウェア研究への 2 つの提案 —, *夏のプログラミング・シンポジウム報告集*, 1992.

[Kan 93] 金田 泰: 確率過程としての計算 — 計算過程のマクロ・モデルの必要性とその例 —, *情報処理学会プログラミング研究会*, 1993 (予定).

[Mur 91] 村島 定行，萬膳 義久: 巡路中の都市隣接性に基づいた巡回セールスマン問題のニューラルネットへの埋込み, *電子情報通信学会論文誌* D-II, Vol. J74-D-II, No. 8, pp. 1080–1089, 1991.

[Pri 77] ニコリス，G.，プリゴジーヌ，I.: *散逸構造* (小畠・相沢 訳), 岩波書店, 1980.

付録 SOOC による TSP のキャスタ

計算言語 SOOC-93 によって記述した TSP のキャスタをしめす[注12]．まず，データ構造をつぎのように定義する (ここで，defelement マクロの構文は Common Lisp の defstruct マクロにちかい).

```
(defelement city              ; 元素 city (city 型) の宣言
   id
   x y
   next)
```

第 2 の規則はつぎのように定義する．

```
(defrule tsp
   (var C1 C2 C3 C4 C5)                    ; 変数宣言
   (reaction
      (exist city C1 (:next C2))
      (exist city C2 (:next C3))
      (exist city C4 (:next C5))         ; ここまでが左辺
      <–>                                ; (この規則は可逆)
      (exist city C1 (:next C3))
      (exist city C2 (:next C5))
      (exist city C4 (:next C2))))       ; ここまでが右辺
```

また，局所秩序度はつぎのように定義する (ここで，deforder マクロの構文は CLOS (Common Lisp Object System) の defmethod マクロにちかい).

```
(deforder ((c city))
                    ; city 型の原子 c の自己秩序度の宣言
   (let ((next (city-next c)))
      (if (null next)
         –999999999
         (– (distance next c)))))
```

ただし，distance は Lisp によって記述されたつぎのような関数である．

```
(defun distance (city1 city2)
   (sqrt (+ (expt (– (city-x city1) (city-x city2)) 2)
   (expt (– (city-y city1) (city-y city2)) 2))))
```

以上のほかに，実行させるためには初期状態の設定をおこない，システムを起動させなければならないが，そのためのプログラムはここでは省略する．

[注12] SOOC-93 は，すでにインプリメントされている計算言語 SOOC-92 とは一部，仕様がことなっている．実験に使用したのは SOOC-92 である．