解説

複雑系による経済モデル分析

第2回　遺伝的アルゴリズムによる最適化と経済への応用

時永　祥三

1. はじめに

今回の解説では，遺伝的アルゴリズム（GA: Genetic Algorithm，以下ではGAとよぶ）による経済モデル分析に関して述べていく[1]．GAは単純な要素の集合を用いて複雑な機能を実現する方法であり，複雑系の理論の中でも最も適用可能性が高いとされている[4]．

これまで，GAに関しては極めて多数の研究がありこれらを全て網羅することはできないので，以下では，次のような分野に限定して話を進める．

(1) GAによる計画問題の数値解法
(2) GAによる組合せ問題の解法
(3) クラシファイヤによるルール合成
(4) 遺伝的プログラミングの基礎と応用

2. 遺伝的アルゴリズムの原理

遺伝的アルゴリズム（GA）は生物の遺伝子に作用する各種の遺伝子操作を模擬して，この方法を数理的に定式化された問題を解く手法として応用するものであり，現在でも，さまざまな分野で，また種々の問題解決に適用されている[1～4, 9]．GAの原理を簡単に述べれば，与えられた問題の解を遺伝子（あるいは個体ともよばれる）として複数個準備しておいて，これらの遺伝子が与える問題解決への貢献度（例えば最大化問題であれば関数値が大きなもの）を計算し，これらの中で貢献度の高いものどうしを取り出し，そのペアに対して遺伝子操作を行い，次の世代における個体として残していく．貢献度の高い個体に対して遺伝子操作をすることにより，より良好な解を与えるといった方針のもとで生成操作が実行される．

GAでは最適化するべき変数値やパラメータ値は，通常，2進数（ビット）として表現される．例えば，n個の変数x_iにより記述される関数$f(x), x=(x_1, x_2, \cdots, x_n)$を最大化する問題を考えると，それぞれの変数$x_i$は$x_1, x_2, \cdots, x_n$の順に並べられる．これを個体とよぶ．1つの個体は，1つの解を意味する．

x_iは複数個のビットにより表現されるが，10進化した数がx_iとなると考えてもよいし，例えば3ビットで0～7の8個の数を表現し，このそれぞれに，あらかじめ設定した数値を対応させることもできる．個体により表現された解を採用した場合に，関数値$f(x)$が得られる．$f(x)$が大きな個体ほど，より最適な解に近いと判断することができる．これを適合度とよぶ．

GAのもとでは最初の個体は乱数により発生された数値を配列したものでよく，解としては良好なものである必要はない．個体を複数個生成しておく．この解を相互に融合することにより解を改善していく．

こののち，次の3つの操作，すなわち，(1)選択（selection），(2)交叉（crossover），(3)突然変異（mutation）を実行する．最初の「選択」は個体の適合度に応じて交叉処理を行う個体を選ぶ方法であり，通常，適合度に応じた確率で個体を選択する．

次の「交叉」処理は，図1に示すように2つの個体をランダムな位置で切断し，それぞれの前半と後半を相互に入れ換えて新しい個体を生成することを意味す

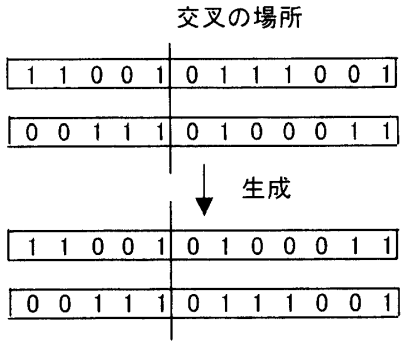

図1　交叉処理の概念

ときなが　しょうぞう
九州大学 大学院経済学研究院経済工学部門
〒812-8581 福岡市東区箱崎6-19-1

る．これら新しく生成された個体はoffspringとよばれる．個体の総数を一定に維持するため，個体のプールにある適合度の小さい個体を，これらのoffspringにより置き換える．

次の「突然変異」は，個体の交叉処理だけでは同じような個体だけになり改善が起こらない状態（数値最適化におけるlocal minimumに相当）に陥るため，これを回避するための方法であり，ランダムに選んだ個体のランダムな位置を，ランダムな数値で置き換える．

なお，これらの選択，交叉，突然変異にはさまざまな拡張が行われ，目的に応じて使用されている（詳細は省略する）．

3. 財務計画問題への応用

次に，GAを用いて企業経営分野における財務計画問題を解く方法について示していく．財務計画問題といっても漠然としているが，ここでは企業が生産拡大などの目的で資金運用や設備投資を実施する場合に，あらかじめいくつかの計画を検討して，その中で最も望ましいものを採用することを指している．このような問題を解く場合に，一般的な制約付きの非線形最適化問題であることに加えて，次のような経営に固有の問題を考慮する必要がある．

(1) スプレッドシートによる多期間モデル記述

企業の財務計画は，設備投資などが現実に数ヵ月や数ヵ年など複数の期間（多期間）にわたって実行されたと仮定して，シミュレーションなどにより検討される．計画に含まれるパラメータをGAにより最適化する．

(2) 各種の指標の比較分析

設備投資などを考えた場合，投入する資金に対する投資計画の有効性をはかる尺度としては，利益の最大化などのほかに，財務指標への制約条件（例えば総資本回転率0.7以上5.0以下，総資本利益率0.09以上1.0以下など）を加えて，評価関数を作成する．

ここでは，問題をやや簡単化し，商品の生産を6年間にわたり実行して，制約条件を満足する範囲で，それぞれの年度に投資すべき金額を求める問題を考える．したがって，GAにおける個体は次のようになる．

1年目投資金額，2年目投資金額，…，6年目投資金額

これらの変数には上限が定められていると同時に，予算の制約，在庫コストなどが与えられているとし，必ずしも多く生産すれば最適解となるものではない．また，資金が不足する場合には借入を行い，余剰が生じた場合には返済することとするが，負債比率が一定の数値より大きくならないなどの条件を加えておく．

財務計画の記述は等式表現として記述できる．以下に簡単な例を示す．

売上高＝500＋5×投資金額

売上高利益率＝25/(1＋8 exp(−2t))，

tは年度でt＝1, 2, …, 6

利益＝売上高×売上高利益率/100

売上原価＝売上高－利益

流動資産＝前年度流動資産，次年度の計算では

前年度流動資産＋流動資産増加

固定資産＝前年度固定資産＋投資金額

流動負債＝前年度流動負債，次年度の計算では

前年度流動負債＋流動負債増加

なお，これらの式では，単年度の業績を表す利益のほかに，次年度以降の負債比率，固定比率などの指標を計算するための式を含んでいる．全体の式の数は約60で，資金の借入と返済（金融）の部分は省略している．

等式表現では，生産によって生じる利益や原価，あるいはこれらから得られる貸借対照表の項目の計算をモデルとして記述している．この表現では当該の期間のほかに，前期として記述している量は，現在の1つ前の期における値である．ここに示すように，商品の製造，販売は複数の期間にわたって影響するので，Excelのセル表現が便利である．

図2にはこの財務計画における投資金額の最適化による利益の改善状況を示している．

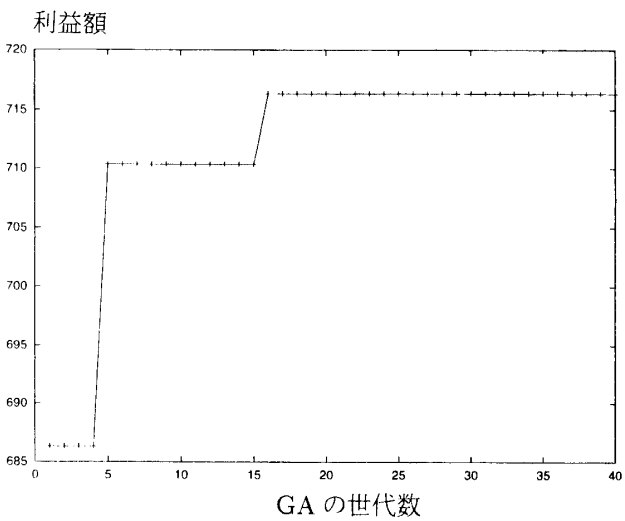

図2 投資金額の最適化による利益の変化

3.1 製鉄所における充当問題

GAは数値最適化や非線形最適化問題を解く手法として便利であるほかに，組合せ問題を解く方法としても応用されている．例えば，複数の製品を加工する場合に，その順序を入れ換えることにより，総コストを減らす問題に適用したり，商品の配送ルートが複数個あり，途中でこれらのルートを交互に入れ換えた新しいルートを求め，これが従来のルートより効率的であるかどうかを判断する．このルートの交叉をGAであると考える方法である．

ORの基礎的な問題としてナップザック問題があり，これは有限のナップザックに荷物を詰める問題で，価格などが最高となるようにする方法である．この拡張として，文献[12]に紹介されている製鉄所における鋼板からの商品の最適切断の事例をとりあげる．

製鉄所では，スラブ（鉄の塊）を圧延して延べ板を作り，これを注文の大きさに切断して多種類の厚板を作成する．厚板はその性質（鋼種とよばれる）に応じて分類されているので，注文の多い厚板は大部分のスラブを用いて生産されるが，スラブは一定の量として生産されるので，予備のスラブ（余ったスラブ）が発生する．これを用いて少量の注文の厚板は予備のスラブを用いて生産している（厚板充当とよばれる）．厚板充当の作業では，スラブと注文との組合せ，延べ板での製品の配置を決める必要がある．

この場合，製品の納期をできるだけ守ることや，延べ板を切った残りが少なくなることが最適化の目的となる．また，鋼種ごとのグループ化，延べ板の寸法(厚さ，幅，長さ）での設備制限などの拘束条件が存在する．

いまスラブへの注文割当を単純化した例を図3に示す．この例では，スラブ2を延べ板として使用したとき，注文1を処理する．もし，充当可能であれば，スラブ1に再び次の注文2が追加充当可能であるかを調べ，順次このような計算を繰り返す．

問題をGAにより解くために，スラブの固有の番号と，注文の固有の番号を1つの個体に表現し，これを部分一致交差処理：PMX（Partly Matched Crossover）により最適化していく手順を用いる．これは，交叉により発生する不都合な部分を修復して実行可能な個体とする方法である．

親A，Bから子供1，2を生成する例を考える．子供1は親Aより交叉の前半部分をそのまま引き継ぎ，後半について親Bから遺伝子を継承とした場合，その遺伝子の位置の値がすでに子供1の前半に含まれている場合には，これ以外の値を親Bから選択する．例えば，子供1は前半を親Aから継承し，後半の第1番目の数値aを親Bからもらう．しかし，後半第2番目の数値bがすでに存在している場合は，親Bにおいてaの前にあるcをもらう．同様の操作を繰り返し，同じ数値を継承しない工夫を行う．

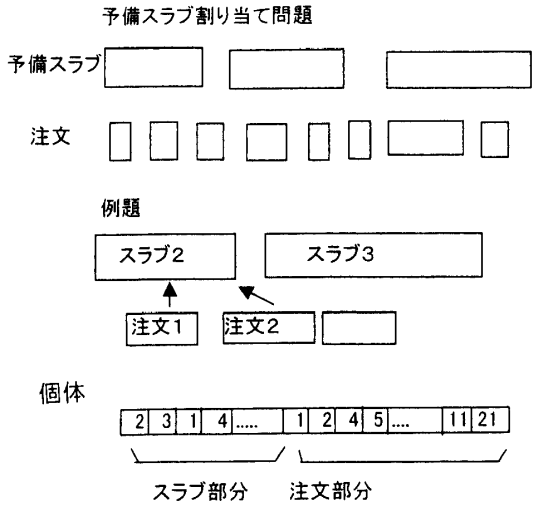

図3　予備スラブによる厚板充当

GAにおける個体は，スラブ番号部分と注文順番部分との2つからなり，10進数をコードとしてそのまま用いている．図3は，同時にこの例を示しており，スラブの選択順序はスラブ2，スラブ3，…であり，注文の割当順序は，注文1，注文2，…の順序である．途中でスラブが充当できなくなったら，次のスラブを投入する．これにより最も無駄の小さな計画（個体）が採用される．

評価関数は注文の納期への満足度と，切断する場合の歩留りで行い，次のように定義される．

$$T = w_1 \sum 1/T_o + w_2(a/b) \tag{1}$$

T_oは現在の日付から納期の迫っている充当された注文の納期までの日数，a, bはそれぞれ，充当された注文の総重量，充当に使用したスラブの総重量である．

以下では，問題を簡単化して，スラブは長さだけが特徴量であり，製品ごとに，厚さや幅を考慮しないと仮定する．ただし，個々のスラブは異なる長さを持っているとする．製品も，同様に，長さと納期だけが規定されていると仮定する．

個体の構成は，最初の部分にはスラブ順番部分があり，その後半に製品の注文順番部分がある．図4には評価関数Tが変化していく様子を示す．これより分かるように，ほぼ，120世代で適切な解のレベルに到

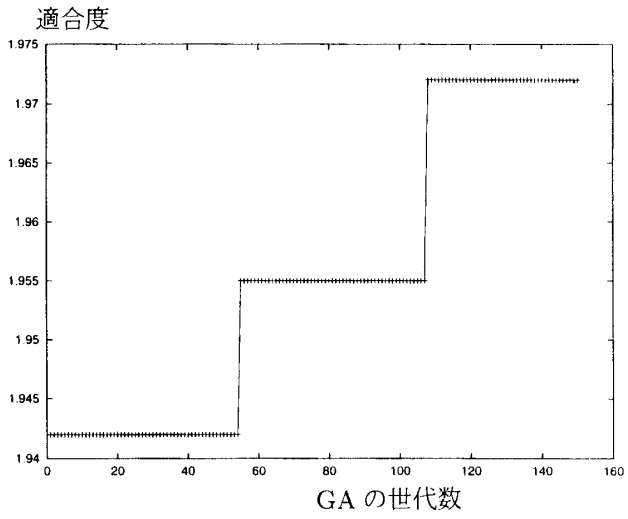

図4　評価関数の改善状況

達すると言える．

4. LCS によるルールの生成

GA における遺伝子（個体）に if-then ルール（プロダクションルールとよばれる）を埋め込んで用いる方法も提案されている．その代表的なシステム構成が LCS（Learning Classifier System）である[1, 10]．LCS においては，1つの最適な個体を求めることが問題とされるのではなく，複数の個体のグループが存在し，それぞれは外部の条件（環境）に依存して最適な解を与えるように構成されている．したがって，時間的に変化する環境に応じて加えるべき操作が変化する場合などの最適化問題に応用される．

プロダクションルールには，条件部（LHS: Light Hand Side）と実行部（RHS: Right Hand Side）があり，現在の環境にルールの LHS が一致した場合に，対応する RHS に記述された手続き（プログラムなど）が動作する規則の集合として記述される．

if　LHS　then　RHS

LCS の動作原理は，単純な GA に比べて，やや複雑である．この動作をアルゴリズムとして書くと，次のようになる．

(1)　初期設定として最初に乱数を発生して，初期のクラシファイヤの集合（プール）を構成する．クラシファイヤの適合度などを計測する手段として強度（strength）という量が定義され，最初の段階で同じ強度をすべてのクラシファイヤに与えておく．

(2)　環境の現状をコード化して，メッセージとして構成してメッセージリストに加える．このメッセージは次のステップでクラシファイヤの LHS と照合される．

(3)　メッセージに適合する LHS を持つクラシファイヤが検出される．これを活性化されたクラシファイヤとよんでおく．

(4)　活性化されたクラシファイヤは，次の段階で賭け金を払ってメッセージを出す権利を得る賭けをする．賭けで競合する相手がいる場合には，強度の大きなクラシファイヤがこの権利を獲得する．

(5)　このようにして競争に勝ったクラシファイヤのメッセージにより，以前のメッセージリストの内容をすべて入れ換える．

(6)　活性化されたクラシファイヤの中で，強度が最大となるクラシファイヤの出したメッセージの中に含まれる情報を用いてシステムへの動作がなされる（RHS に書かれている具体的な操作を実行）．

(7)　上の(2)～(6)に示された動作を一定の回数繰り返したのちに，個体に GA を適用して，次の世代を生成する．

以下では問題を簡単にし，メッセージは外部からのみ供給され，クラシファイヤの RHS には直接的に外部に操作する量が記述されている場合を考察する[10]．そのため，競合関係の解消は1回で終了し，クラシファイヤはメッセージをリストに格納しないと仮定する．したがって，LHS は環境からのメッセージとの一致度をはかるだけのものとする．ここでは，株式の取引ルールを作成する問題を考察してみる．

まず，学習に用いるデータとしては，次のようなものが観測されていると仮定する．すなわち，(1)株価の上昇下降のデータ，(2)トレンドの形状，(3)過渡的な変化，(4)取引数量，(5)金利動向である．

LCS のデータの構成は次のようになる．ビット位置1～10は第1～5番目の情報に対する LHS を2ビット単位で表現（4段階），ビット位置11～13は LHS が満足される場合の株価予測を3ビット単位で表現（8段階）する．

データを LCS に繰り返し与えることにより学習を行う．なお，学習の結果を検証するには別のデータが必要であるが，ここでは話を簡単にするために同じデータを用いて検証している．

また，LCS のシステムに含まれるパラメータは次のように設定しておく．クラシファイヤ数は50，GA の適用間隔は10，学習ステップ数は500，突然変異確率は0.1である．

LCS を適用した結果として得られる利益額を縦軸

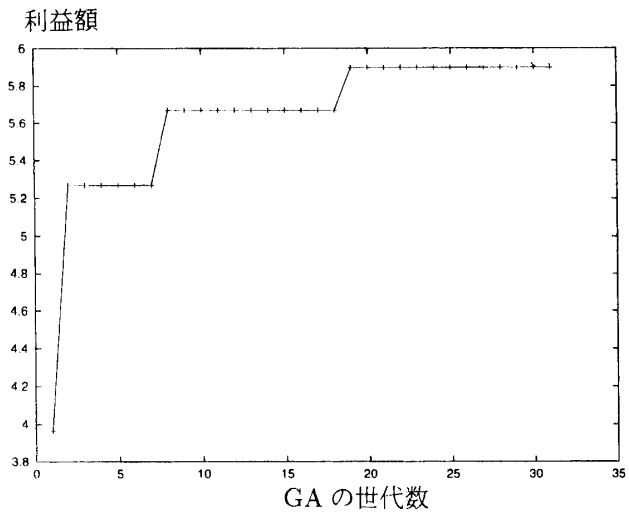

図5　LCS による株式取り引きのルール学習

に，ステップ数を横軸にとった場合の図5を示している．図5より分かるように，LCS での学習には最低30回の繰り返し計算が必要であり，GA による数値最適化にくらべて長い時間が必要であることがわかる．

5. 遺伝的プログラミングによる方程式系の近似

遺伝的プログラミング（GP: Genatic Programming）は，方程式などの形を木構造により表現しておいて，演算記号や対象とする変数を変更しながら，望ましい形に近似する問題に適用可能である[5, 6]．

GP は GA の1つの拡張形であると言え，GP では1つのコンピュータプログラムである．具体的には，計算する演算記号，数学関数，およびこれらの対象となる変数を含んだ木構造で表現されている．以下では，観測された時系列データ $x(t)$ をもとに，この観測データ $x(t)$ を生成しているシステム（方程式の体系）を推定する問題を中心に考えていく．

個体に対応する木構造は，前置表現（prefix representation）とよばれるもので表現される[7, 8]．これは，演算子と被演算子を適切な順序に一列にならべる表現である．例を次に示す．いまシステムの方程式を

$$x(t)=(6.43\times x(t-1)-x(t-2))\times(x(t-3)-3.54) \quad (2)$$

とする．これを変換した式は次のようになる．

$$\times-\times 6.43x(t-1)x(t-2)-x(t-3)3.54 \quad (3)$$

この前置表現を解釈するには次のようにする．まず変数 $x(t-1), x(t-2), x(t-3)$ に値を代入する．この式を左から見ていき，演算子の後ろに被演算子（変数か定数）が2個連続している場合に限り，その演算を実施する．結果は1つの被演算子になる．この操作を，対象がなくなるまで繰り返す．この操作を実行するにはスタックを用いると便利である．スタックとは，上から順にものを乗せていって，適当なときに，その乗せたものを取り出す仕組みである．

個体の適応度の定義として，方程式による予測値と観測値との2乗誤差の逆数を用いる．個体の交叉処理，突然変異に関しては，やや注意が必要である．GA の場合には，個体の構造は基本的に同じであるので，乱数を発生させておいて，2つの個体を交叉する位置として用いればよかった．これに対して，GP では，一般に個体の長さが異なるので，GA のような手法は適用できない．また，木構造を適当な位置で切断して，相互に結合することになるが，切断することにより意味をなさなくなるケースも発生するので，任意の位置で切断することはできない．同様に，突然変異も任意の場所で任意の形式で実施することはできない．

交叉処理を矛盾なく進めるため，*StackCount* という変数を導入する[8]．*StackCount* は，スタックの中に存在する演算子の数から，演算記号の数を引いた数値である．したがって，前置表現において，その途中まで，この *StackCount* をかぞえて，その数値が同じであれば，この位置で2つの個体を切断し交叉処理をしても，意味のある方程式表現を与えていることになる．図6に例を示す．

以上のような準備のもので，GP を適用するアルゴリズムを整理すると以下のようになる．

（ステップ1）　個体の初期値生成

最初の個体の集合（プール）を乱数をもとにして発生させる．この場合，すでに述べた *StackCount* を用いて，システム記述の方程式として意味をなすものが得られるまで繰り返す．

（ステップ2）　個体の適応度の計算

すでに述べた関数近似における予測値と観測値との2乗誤差の逆数を，個体 i の適合度 S_i として定義する．

（ステップ3）　交叉処理

適応度の大きさに応じた確率で2つの個体を取り出し，交叉処理を実施する．個体の長さは一般に異なるので，乱数を1個発生させておいて，一方の個体の切断個所とし，ここまでの *StackCount*（図6では演算子数3－被演算子数1＝2）を計算する．こののち，もう一方の個体の *StackCount*（図6では演算子数

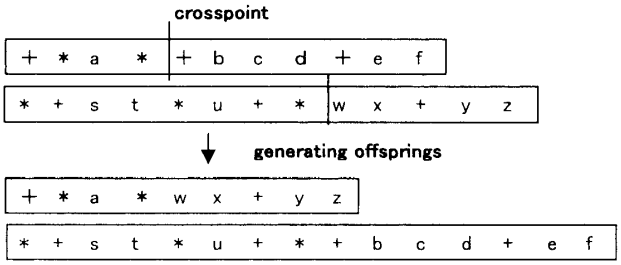

図6 GPにおける交叉処理の例

5－被演算子数3＝2）を計算しながら，同じ*Stack-Count*の値となる場所を検出しこれらから任意に1個を選択する．このように，それぞれの個体の切断位置を決める．

（ステップ4） 突然変異

GPにおける突然変異として，次の2つのものを用いることとし，一定の確率で適用する．

グローバル突然変異

ある個体（木構造）I_sを生成しておいて，目的とする木構造の葉の部分を，このI_sにより置き換える方法である．もとの木構造は，やや複雑なものへと変化する．

ローカル突然変異

木構造の葉の部分を，別の変数に置き換える操作である．あるいは，木構造における根の部分に相当する原始関数を，別のものに置き換える操作であることもある．GAにおける局所探索（local search）に相当している．

応用として，カオス時系列を発生させる方程式を推定する問題を考える．カオス時系列とは，一見不規則に変動する時系列でありながら，その発生メカニズムが1つの方程式で記述できるものを指しており，特に，この方程式の中には，確率過程のような不確実な要素を含んでいない．このような方程式を，特に決定論的な方程式（deterministic equations）とよんでいる．

以下では，問題を簡単にするために，すでに方程式系が既知である次のケースを考察する[5]．ここでは，次の式により与えられるロジスティック写像（logistic map）により生成された時系列を用いる．

$$x(t)=3.87x(t-1)[1-x(t-1)] \tag{4}$$

シミュレーションでは，次のような条件設定を行う．個体プールのサイズ＝500，配列の最大値＝20，GP最大適用回数＝500，データサンプル数＝20．

十分なGPの繰り返し回数の後に，式(4)の方程式とまったく同じ式を得る．このときの，観測値と予測値の2乗平方誤差を時系列の標準偏差で割った予測誤差は$rsme=0.000002$である．近似された方程式系により発生される時系列（予測値）は，もとのロジスティック写像により生成された観測時系列の値と極めて近い．

ここまでの議論では，カオス時系列を生成する方程式系は1次元であると仮定してきた．しかし，カオス力学系では，通常，3次元以上でカオスとなることが知られているため，観測された時系列を3次元の方程式系で推定することが望ましい．多次元カオスに対するGP適用方法とその応用は，この解説の最後の回で示す．

参考文献

[1] L. B. Brooker, D. E. Goldberg and J. H. Holland: "Classifier system and genetic algorithm", Artificial Intelligence, 40, pp. 235-282 (1989).

[2] D. E. Goldberg: *Genetic Algorithm in Search, Optimization and Machine Learning*, Los Altos, California, Morgan Kaufmann (1989).

[3] D. E. Goldberg: "Computer-aided pipeline operation using genetic algorithm and rule learning. Part I: Genetic algorithm in pipeline optimization, Part II: Rule leraning control of a pipeline under normal and abnormal conditions", Engineering with Computers, 3, pp. 35-58 (1987).

[4] J. H. Holland: *Adaption in Natural and Artificial Systems*, Univ. of Michigan Press (1975), MIT Press (1992).

[5] Y. Ikeda and S. Tokinaga: "Approximation of Chaotic Dynamics by using smaller number of data based upon the Genetic Programming and its Applications", Trans. IEICE, vol. E 83-A. no. 8, pp. 1599-1607 (2000).

[6] Y. Ikeda and S. Tokinaga: "Controlling the chaotic dynamics by using approximated system equations obtained by the Genetic Programming", Trans. IEICE, vol. E 84-A. no. 9, pp. 2118-2127 (2001).

[7] J. R. Koza: *Genatic Programming*, MIT Press (1992).

[8] M. J. Keith and M. C. Martin: "Genetic programming in C++: Implementation issues", in (ed) K. E. Kinnerar, Jr., Advance in Genetic Programming, MIT Press (1994).

[9] K. Tan and S. Tokinaga: "Optimization of fuzzy inference rules by using the genetic algorithm and its application to the bond rating", JORSJ, vol. 42, no. 3,

pp. 302-315 (1999).
[10] S. Tokinaga: "Applying adaptive credit assignment algorithm for the learning classifier system based upon the genetic algorithm", Trans. IEICE, vol. E 75-A, no. 5, pp. 568-577 (1992).
[11] 時永祥三:『複雑系による経済モデル分析』, 九州大学出版会 (2000).
[12] 吉田耕作ほか6名:"遺伝的アルゴリズムを用いた厚板充当システム", 日本オペレーションズリサーチ学会1995年度春季研究発表会アブストラクト集, pp. 62-63 (1995).