論　文

遺伝的アルゴリズムによる有限オートマトン設計の自動化

非会員　横井智幸　(福岡工業大学)
正員　小野俊彦　(福岡工業大学)

Generation of Finite Automaton by Genetic Algorithms

Yokoi Tomoyuki, Non-member, Ono Toshihiko, Member (Fukuoka Institute of Technology)

This study uses genetic algorithms(GAs) for the system which generates a finite automaton automatically. This system can build an automaton by incorporating automaton construction algorithms(ACAs) into the GAs. ACAs are algorithms that construct a finite automaton on the basis of gene information. The performance of this system is realized by integration algorithms and minimization algorithms. The integration algorithms are applied when an automaton is made by combining several automata, each of which is created by divided requirements of the original automaton. The minimization algorithms are used to minimize the number of nodes of the automaton which GAs and the integration algorithms generate. The most suitable finite automaton is formed automatically by using these algorithms collectively. An automaton drawing program can show the finite automaton as a state transition diagram to confirm the result easily. By simulations, the effectiveness of this method was confirmed.

キーワード：遺伝的アルゴリズム, 有限オートマトン
Keywords: genetic algorithms, finite automaton

1. まえがき

遺伝的アルゴリズム（Genetic Algorithms:GA）は1975年にJ.Hollandの著書Adaptation in Natural and Artificial Systems[1]に導入され一般に公開された。その概念は生物進化の過程を遺伝子に着目しモデル化したもので，遺伝子を一つの解候補と見なして集団を作り，その集団で選択淘汰，交叉，突然変異などの遺伝的操作によって進化をシミュレートし解を求める方法である。このアルゴリズムは確率的かつ経験的な側面を持つ探索法で，解候補を遺伝子に表現できればモデル化困難な問題にも適用できるため工学分野だけでなく経済分野などの様々な問題に適用され成果を上げている。

オートマトンはコンピュータの働きを抽象化した数学的モデルであり，有限オートマトン[9]は基本的なモデルとして，さまざまな事柄に応用されている。有限オートマトンは，はじめ神経網とスイッチング回路を念頭において構成されたが，最近ではコンパイラの一部である語彙解析プログラムの設計のための有用な手段としても用いられている。その他にもテキスト編集，パターン・マッチング，各種のテキスト処理やファイル探索プログラムなどに用いられ，さらに論理学のような分野にも有用な数学的概念の一つとして使われている。

有限オートマトンを構成する場合に問題となるのは，いくつの状態数でどのように状態遷移させれば効率の良いオートマトンが構成できるかである。これまで，この問題に対してヒューリスティックアルゴリズム用いた手法[3]や，遺伝的アルゴリズムによる方法[7]，遺伝的プログラミング（Genetic Programming:GP）を用いた手法[4][5]，ニューラルネットワークを用いた手法[6]などの研究が発表されている。

本研究の目的は有限オートマトンを自動生成するシステムを構築することである。本システムはGAとオートマトン構成アルゴルズム（Automaton Construction Algorithms:ACA）を主とし，合成アルゴリズム、最小化アルゴリズム及び描画プログラムより成り立っている。ACAでは遺伝子の情報よりオートマトンを構成した後，生成したオートマトンが満たすべき条件を規定した受理・却下信号データファイルをもとに評価を行う。この評価値よりGAは条件を満足するオートマトンを求める。合成アルゴリズムは生成されたオートマトンの合成を行い，最小化アルゴリズムは生成されたオートマトンを最小の構成に変換する。オートマトン描画プログラムは結果を状態遷移図として出力する。

このシステムの特徴はいくつかのアルゴリズムを統合的に用いることで，GAのみでは難しい受理条件のオートマトンの生成に対しても適用できることである。

以下，システムの構成，GAの適用法，各種アルゴリズムの内容，シミュレーション結果について報告する。

2. 問題設定

有限オートマトンを構築する場合に受理条件に合わせて次のようなことを決定しなければならない。

- 構成する状態（ノード）の数
- 入力信号に対する状態の遷移先
- 受理する状態

有限オートマトンの受理条件は，受理される信号及び却下される信号によって表す。つまり，受理条件・却下条件がすべて満足するオートマトンを構成することが目的となる。

システムを構築するにあたって以下の点を考慮した。

- 受理条件を満足するオートマトンは多数考えられるが最適化の観点から最小の状態数のオートマトンを構成する。
- 複雑な受理条件に対応させる。
- 状態遷移図で表示させる。

なお本研究では，入力信号はa・bの2種類の記号とする。

3. システム構成

本研究では，遺伝子の情報に従ってオートマトンを構成するアルゴリズムACAを導入し，GAと組み合わせシステムを構築した。また，生成するオートマトンの構成ノード数が未知数なので，GAを適用するにあたって遺伝子の長さが問題となるため本方式では，構成ノード数を最小の2より開始しオートマトンが求まるまで順次増加させる方法を用いた。さらに複雑な条件のオートマトンの生成はGAのみでは難しいので合成アルゴリズム及び最小化アルゴリズムを用いることで対応している。本システムの構成を図1に示す。図のようにノード数制御部，GA部，ACA部及び受理・却下信号データファイル，合成アルゴリズム，最小化アルゴリズム，オートマトン描画プログラムで構成され，主な動作は次のようになっている。

ノード数制御部ではノード数を決定する。開始ノード数はオートマトンの最小ノード数である2ノードより開始し，所定の条件を満足するオートマトンが得られない場合に順次1づつ増やして行く。

次にGAでは選択・交叉・突然変異等の遺伝的操作を行うことによって新しい遺伝子を生成する。

ACAでは遺伝子情報をもとにオートマトンを構築し，その評価を行う。このときACAは以下に説明する受理・却下信号データファイルを参照してオートマトンの評価を行う。その評価を評価値としてGAに返す。ACAの詳細は後述する。

受理・却下データファイルには生成したオートマトンに入力を与えた場合の結果（受理・却下）の情報を規定している。入力信号はあらゆる条件を考慮して一般的に無限の長さが必要であるが、実用上は不可能であるので，ある一定の長さまでの取り得るすべての信号列としている。このデータにより，生成されたオートマトンの正確な評価を行うことができる。図2は入力信号の長さが3個の信号までの取り得るすべての信号列での受理・却下データファイルの例である。このようなデータファイルの作成は信号列が長くなると手作業では困難となるので，受理・却下各条件をもとにプログラムで作成できるようにしている。

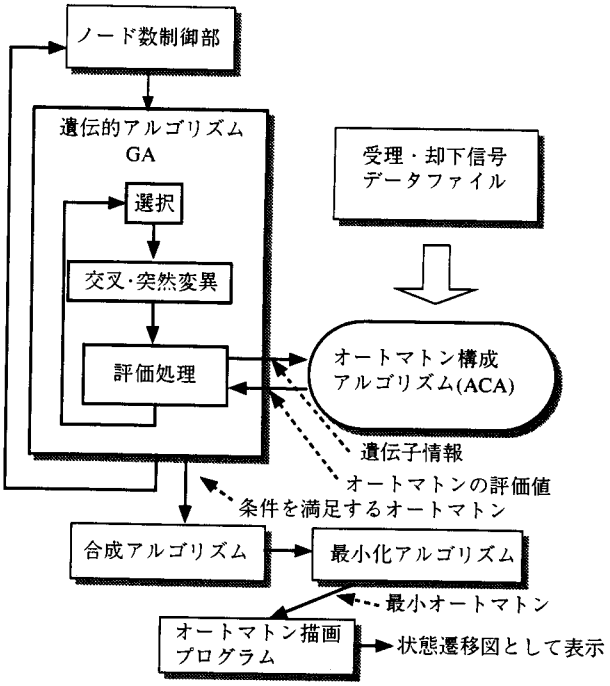

図1 系統図
Fig. 1. Schematic diagram

入力信号	結果(受理1・却下0)
なし	0
a	0
b	1
aa	0
ab	0
ba	1
bb	1
aaa	0
aab	0
aba	1
abb	0
baa	0
bab	1
bba	1
bbb	1

図2 受理・却下信号データファイルの例
Fig. 2. An example of data file for acceptance and rejection signal

合成アルゴリズムでは，合成するためのオートマトンをこれまでの操作を繰り返すことで求め，そのすべてのAND及びOR合成を行い1つのオートマトンにする。

最小化アルゴリズムでは生成されたオートマトンの同値ノードや孤立ノードをなくし最小化オートマトンに変換

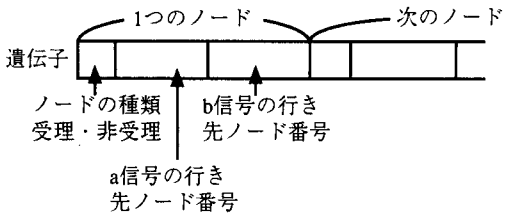

図3　遺伝子表現
Fig. 3.　Representation of genes

する。

オートマトン描画プログラムは生成されたオートマトンを容易に確認できるように状態遷移図で表示する。

4. GAの適用法

本研究ではGA処理部にGenesis[7]を採用した。Genesisには基本的な遺伝的操作が用意されているので，これに必要とする改造、追加を行った。以下，GAの適用における遺伝子表現，選択，交叉，突然変異について述べる。

〈4・1〉 遺伝子表現　GAを適用するにあたって特に問題となる遺伝子表現はオートマトンを構成するために必要な情報を表していなければならない。本研究では問題設定で述べた3点の情報を遺伝子で表現しGAを適用した。

遺伝子表現の具体的な内容は図3のようになる。表現方法はビットストリング表現とし，これをノードごとに分割し，各ノードについて以下のように表現する。最初のビットで受理・非受理ノードの区別を表し，続いてのnビット（nはノード番号を表すのに必要なビット数）は"a"の信号が入力された場合の行き先ノード番号，次のnビットは"b"の信号が入力された場合の行き先ノード番号を表している。このようなビットストリングを必要なノード数だけ並べたものを一つの遺伝子としている。

この遺伝子情報をもとに，ACAによってオートマトンが構成される。

〈4・2〉 選択操作　選択操作はエリート保存方式を加えたランク選択を用いた。エリート保存とは各世代において最も優れた個体を必ず次の世代に残す方式である。ランク選択とは各個体を評価値の順に並べ，その順番に比例した確率で選択する方式である。

〈4・3〉 交叉法　本研究の交叉法はオートマトンの構成を考慮した2点交叉を用いた。

一般的な2点交叉法ではランダムに選んだ2点間で交叉を行うため，図4（a）のようにノード内の点を交叉点とした場合，親のオートマトンのノード情報(受理・非受理，信号による行き先）が破壊されてしまう。そこで交叉の場所を図4（b）に示すようにノード間の境界点よりランダムに選んだ2点間で行うようにした。この交叉法によって親の特性を破壊することなく子に受け継がせることができる。

〈4・4〉 突然変異操作　本研究で使用した突然変異操作として，遺伝子の各遺伝子座ごとにある確率でその値を変

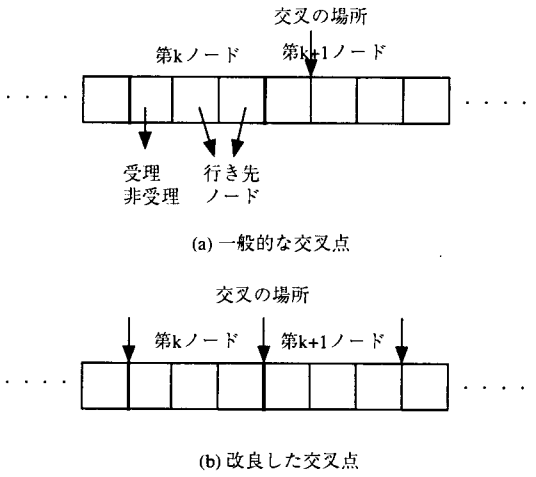

図4　交叉点
Fig. 4.　Points of crossover

化させる方式を用いた。

〈4・5〉 評価処理　評価値の計算はACAにて行い，その値を利用してGAは最適化を行う。

5. オートマトン構成アルゴリズムACA

ACAはGAから送られてくる1つの遺伝子情報をもとにオートマトンを作成し，その評価を行うアルゴリズムである。図1システム系統図に示されているACAの動作は以下のようになる。

ACAの全体の流れをフローチャートで表すと図5となる。まずGAから遺伝子を受け取り，その情報（各ノード

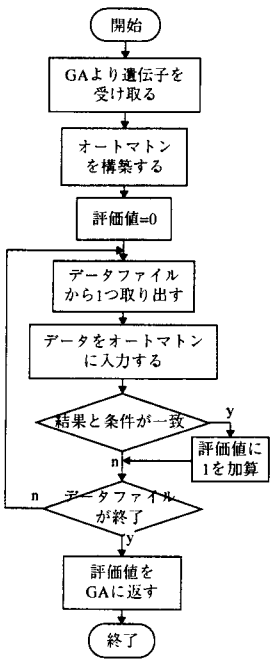

図5　ACAのフローチャート
Fig. 5.　Frowchart of ACA

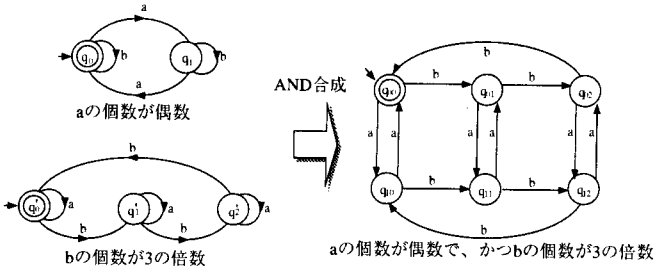

図6　合成例
Fig. 6.　Example of composition

の受理・非受理状態，行き先ノード番号）をもとにオートマトンを作成する。次に作成したオートマトンの評価を受理・却下信号データファイルを参照して行い，その評価を評価値としてGAに送る。評価値は作成されたオートマトンがどれだけ規定の条件を満たしているかを表している。この評価法をステップの順に説明すると次のようになる。

ステップ1.　受理・却下信号データファイルから1つデータを取りだす。

ステップ2.　取り出したデータを遺伝子より構築したオートマトンに入力し受理か却下の判定を行う。

ステップ3.　ステップ2によって得られた判定結果とデータの受理・却下条件が一致した場合に評価値に1を加算する。

ステップ4.　データファイルが終了するまでステップ1～3を繰り返す。

ステップ5.　GAに評価値を返す。

以上より得られた評価値とデータの総数が同じ値となれば所定のオートマトンが求まったことになる。またGAではこの評価値が最大になるように探索を行う。

6.　合成アルゴリズム

複雑な条件のオートマトンの作成を行う場合，GAの探索空間が広くなりオートマトンが求まり難くなる。そこで本システムでは複雑な条件をいくつかの条件に分け，それぞれのオートマトンをGAによって作成する。そして，それらを合成アルゴリズムによって合成することで全体のオートマトンを求める方法を導入した。なお，合成アルゴリズムはオートマトンの合成の定理[18]を参考にした。この合成アルゴリズムによる合成の例を図6に示す。図ではAND合成して2つの受理条件が合成されている。

7.　最小化アルゴリズム

複雑な受理条件のオートマトンをGAで生成しようとしたとき，孤立ノードや意味的に同じである同値ノードなどが含まれ最小な構成ではない解が生成される場合がある。また合成アルゴリズムを使用した場合も同様である。そこで本システムでは最小化アルゴリズム[8][9]を導入し最小オートマトンを求める。図7にその例を示す。図では最小化によってノード数6のオートマトンがノード数3に減少している。このアルゴリズムを使うことでオートマトンが得ることができれば常に最小の構成にすることができる。

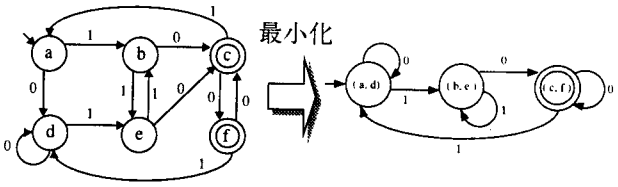

図7　最小化例
Fig. 7.　Example of minimizing

8.　シミュレーション

本方式の特性を検証するためにシミュレーションを行った。なお，すべてのシミュレーションはワークステーション Sun Sparc Classic にて実行した。以下その結果を報告する。

〈8·1〉 受理条件を満足するオートマトンが得られるかの検証　本システムを検証するために文献[3]で紹介されている7種の受理条件でシミュレーションを行った。表1に生成するオートマトンの条件を示す。初期ノード数は2とし，ノード数が各150世代間で受理・却下条件を満足するオートマトンが得られない場合に1つ増加させる方式とした。この時，増加させるノード数の上限を10ノードとし，ノード数増加時に最良の遺伝子を1個残すことにした。その他の設定は個体数100，交叉確率1.0，突然変異確率0.03とし，受理・却下データファイルは長さ12ビットまでの取り得るすべてのビット列（$2^0+2^1+\cdots+2^{12}=8191$列）とした。なお，このファイルは受理条件よりプログラムにより生成した。

表1　受理条件
Table 1.　Acceptance and rejection conditions of automaton

シミュレーション	受理条件	受理例	却下例
1	b*	bbbbb	bbaba
2	(ba)*	babababa	abbaaab
3	奇数個のbのあとのaは奇数個ではない	aaabaabb	abbbaaaaa
4	aaa列が無い	bbabbaa	abbaaaba
5	ab列の数とba列の数の和が偶数	aabbaaaba	aabbaaab
6	bの数とaの数の差が3の倍数	abbaabbbb	bbabaaa
7	a*b*a*b*	aababbb	abbaabba

表 2　検証結果
Table 2.　Result of verification

シミュレーション	得られた回数	得られた確率	ノード数	最小世代数	最大世代数	平均世代数
1	300	100.0	2	0	2	0.07
2	300	100.0	3	0	135	7.87
3	300	100.0	5	3	150	83.04
4	299	99.7	4	0	149	38.56
5	300	100.0	5	8	150	77.22
6	300	100.0	3	0	147	16.71
7	300	100.0	5	6	150	76.70

以上の条件で 300 回実行し検証を行った結果を表 2に示す。第 2，3 列はそれぞれの正しいオートマトンの得られた回数と確率，第 4 列は得られたオートマトンのノード数，第 5 列以降は正しいオートマトンが得られたノード数における最小世代数，最大世代数，300 回の単純平均値を示す。GA において収束特性は初期集団による影響が大きいため初期集団生成の際の乱数系列を変えて 300 回のシミュレーションを各受理条件ごとに行った。乱数の核として時間の整数を用いることにより，全て違った乱数系列を発生させた。7 種のうちシミュレーション 4 を除く 6 種に対して 100%即ち全てにおいて正しい結果が得られた。またシミュレーション 4 は 300 回中 299 回は正解が得られた。実行時間も本検証では一つの問題に対して 1 分以内でオートマトンを生成することができた。これらの結果の一例として，シミュレーション 3，および 5 において生成されたオートマトンを図 8，および 9に示す。なお，この図はオートマトン描画プログラムを用いて作成した。

〈8・2〉 合成および最小化の効果の検証　本システムでは複雑な受理条件のオートマトンの作成の効率化を図るため，合成アルゴリズムを導入した。その効果の検証を行うために，合成を用いない手法と合成を行った方法との比較シミュレーションを行った。具体的な方法は以下のとおりである。

（1）　受理条件は「奇数個の b の後の a は奇数個でなく，かつ b の数と a の数の差が 3 の倍数のものを受理する」が受理条件とした。GA のみではこの条件で実行する。

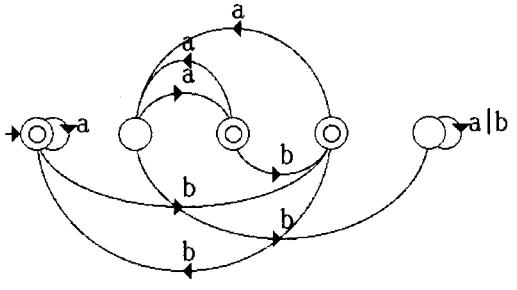

図 8　シミュレーション 3
Fig. 8.　Created automaton-3

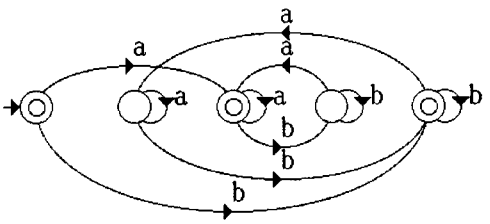

図 9　シミュレーション 5
Fig. 9.　Created automaton-5

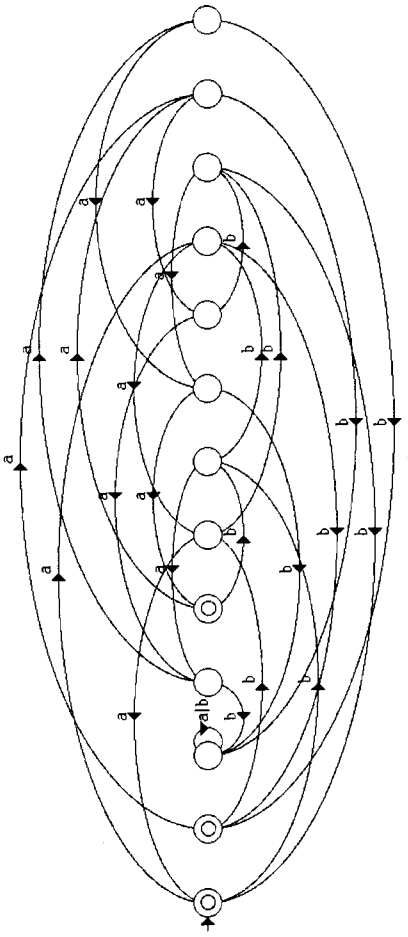

図 10　合成・最小化による結果
Fig. 10.　Result of composition and minimizing

（2）　合成を行う方は「奇数個の b の後の a は奇数個でないものを受理する」及び「b の数と a の数の差が 3 の倍数のものを受理する」が条件のオートマトンをそれぞれ生成させ，それらを合成することで解を求める。

（3）　この際，各々のデータファイルは 10 ビットまでの取り得るすべてのビット列を用いる。

（4）　得られた解は最小化アルゴリズムによって最小化する。

表3 ノード数の変化
Table 3. A change of number of nodes

合成時のノード数	最小化後のノード数
15	13

以上の方法で実行した結果，GAのみで行った場合は1時間の計算時間でも結果が得られなかったが，合成方式では4分以内の実行時間で条件を満足するオートマトンが求まった。得られた結果の状態遷移図を図10に示し，表3に最小化アルゴリズムによるノード数の変化を示す。

この表より最小化アルゴリズムによってのノード数の減少が確認できた。

9. まとめ

本論文は有限オートマトンを自動生成するシステムにGAを適用する研究について報告した。本システムでは遺伝的アルゴリズムとオートマトン構成アルゴルズムを組み合わせることでオートマトンが自動構成でき，またノード数を制御することで最小ノード数で構築することが可能となった。さらに，合成アルゴリズムや最小化アルゴリズムを用いてより性能を向上させることができ，オートマトン描画プログラムにより表示される状態遷移図によって容易に生成したオートマトンを確認することができた。これらの有効性をシミュレーションによる検証で確認できた。

（平成12年7月26日受付，同12年12月22日再受付）

文献

（1） J. H. Holland : Adaptation in Natural and Artifical Systems, The University of Michigan Press, p.211, 1975.
（2） D. E. Goldbeerg : Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Pub. co., p.412, 1989
（3） Masaru Tomita : Dynamic construction of finite-state automata from examples using hill-climbing, Proceedings of the Fourth Annual Cognitive Science Conference, Ann Arbor, MI, pp.105 - 108, 1982.
（4） Bertrand Daniel Dunay, Frederic E. Petry, Bill P. Buckles : Regular language induction with genetic programming, Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE Press, Volume I, pp. 396 - 400, 1994.
（5） Scott Brave : Evolving Deterministic Finite Automata Using Cellular Encoding, Proc. of The First Annual Conference on Genetic Programming, pp.39-44, 1996.
（6） C. L. Giles, C. B. Miller, D. Chen, H. H. Chen, G. Z. Sun, Y. C. Lee : Learning and Extracting Finite State Automata with Second-Order Recurrent Neural Networks, Neural Commputation, Volume 4, pp.393 - 405, 1992.
（7） Peter J. Angeline, David B. Fogel, Lawrence J. Fogel : A Comparison of Self-Adaptation Methods for Finite State Machines in Dynamic Environments, Evolutionary Programming V, Proceedings the Fifth Annual Conference on Evolutionary Programming, pp.441 - 449, 1996.
（8） 岩田 茂樹，笹井 塚美 : 有限オートマトン入門，森北出版株式会社,1986.
（9） J. ポップクロフト，J. ウルマン : オートマトン 言語理論 計算論1，サイエンス社，1984.
（10） 北野 宏明 : 遺伝的アルゴリズム，産業図書，1993.
（11） 北野 宏明 : 遺伝的アルゴリズム2，産業図書，1995.
（12） リプシュッツ，成嶋 弘 : 離散数学，マグロウヒル，1984.
（13） 小野 俊彦，田代順一，横井智幸 : 遺伝的アルゴリズムと最適問題，福岡工業大学情報科学研究所所報 第9巻，pp.11 - 22，1998.
（14） 小野 俊彦，横井 智幸，山田 正和，河村 将史 : 遺伝的アルゴリズムの応用，福岡工業大学情報科学研究所所報 第10巻，pp.49 - 58，1999.
（15） 横井 智幸，小野 俊彦 : GAによる回転を考慮した二次元パターンの最適切断，第17回計測自動制御学会九州支部学術講演会予稿集，pp.287 - 288，1998.
（16） 横井 智幸，小野 俊彦 : 遺伝的アルゴリズムによる有限オートマトンの自動生成，第18回計測自動制御学会九州支部学術講演会予稿集，pp.333 - 334，1999.
（17） 横井 智幸，小野 俊彦 : 有限オートマトン自動生成システムへの遺伝的アルゴリズムの適用，第44回システム制御情報学会研究発表講演会予稿集，pp.453 - 454，2000.
（18） Toshihiko Ono, Tomoyuki Yokoi : AUTOMATIC GENERATION OF FINITE AUTOMATON BY GENETIC ALGORITHMS, 3rd International Workshop on Emergent Synthesis, pp.31 - 36, 2001.

横井智幸（非会員） 1975年7月15日生。1998年3月福岡工業大学情報工学科卒業。2000年3月福岡工業大学大学院工学研究科修士課程情報工学専攻修了。同年4月福岡工業大学工学研究科博士後期課程知能情報システム工学専攻に入学，現在に至る。遺伝的アルゴリズムの応用の研究に従事。計測自動制御学会学生会員。

小野俊彦（正員） 1931年10月10日生。1954年3月九州大学工学部電気工学科卒業。同年4月東京芝浦電気（株）（現（株）東芝）技術部，1978年7月より同社重電技術研究所勤務。1987年10月福岡工業大学工学部通信工学科教授，1990年4月情報工学科教授，現在に至る。システムの故障診断，知識工学。遺伝的アルゴリズムの応用などの研究に従事。工学博士，技術士。電気学会，計測自動制御学会，システム制御情報学会，情報処理学会，人工知能学会会員。