Fuzzy Systems and Data Mining VI
A.J. Tallón-Ballesteros (Ed.)

doi:10.3233/FAIA200711

The Overview of Genetic Algorithm with Tree Chromosome Structure to Identify Functions

Mitsukuni MATAYOSHI [a,1]
[a] *Department of Industry and Information Science, Japan*

Abstract. This paper is a collection of previous studies for function identification by simple genetic algorithm (GA) [1] with tree chromosome structure which has been proposed in [2]-[7], and gives the details more than survey paper. This paper also aims to introduce the studies which were written in Japanese. In this paper, there are five main points. First, a tree chromosome structure, which is the core idea of the studies, is introduced. The tree chromosome structure makes GA succeed in function identification called symbolic regression. Second, the proposed GA with tree chromosome structure succeeded in identifying the target functions from the observed data are shown indeed. The target functions are algebraic functions, primary transcendental functions, time series functions including chaos function, and user-defined one-variable functions. Third, to find function represented with some parentheses, a hierarchical tree chromosome structure is introduced. Forth, some local search methods to aim at the improvement for identification success rate and shortening identification time are introduced. In the end of this paper, the proposed tree and hierarchical tree chromosome structure can be adapted for identifying Boolean functions are laid out.

Keywords. Genetic Algorithm, Evolutionary Algorithm, Function Identification, Symbolic Regression, Local Search, Time Series Function, Boolean Function

1. Introduction

The scientist finds natural laws from observed data. It has long been human's dream to automate this process. One such attempt is Genetic Programming (GP) [8]-[12], which treat structural representations of functions directly as gene code using computer language LISP. It can search for a target function by applying basic operators of genetic algorithm to the concept tree coded in LISP. However, in real experiments, the high number of initial groups needed to maintain the variety of a graph causes the huge amount of calculation necessary, that is the big problem in GP. And, by crossover and/or mutation of genetic manipulation approach, a profitable partial tree structure may be destroyed. Thereby the search process is not steady, and much time is needed in many cases.

However, in 1998, a unique approach using simple GA [2], which can find the laws from the observed data in a very short time, has been proposed. Subsequently, its local

[1]Corresponding Author: Mitsukuni Matayoshi, Okinawa International University, 2-6-1 Ginowan Ginowanshi Okinawa, Japan; E-mail: matayosi@okiu.ac.jp

search methods and a modified chromosome structure to find Boolean function have been also proposed [3]-[7]. Nevertheless, the approach is unknown to researchers who are engaged in studying of genetic algorithms. Even though the evolutionary computing research area becomes wide, and recently there are so many applied research areas such as [13]-[15]. Presumably, the reason is that the study and some related studies were written in Japanese. Therefore, the aim of this paper is to introduce the studies in English.

In this paper, first of all, a sophisticated chromosome structure for a simple genetic algorithm [2] to identify function from observed data is introduced, and that is required for understanding the studies, which have been proposed by Matayoshi [2]-[7]. The proposed GA-based function identification methods with tree chromosome structure can, for example, identify $f(r,x,m) = r \cdot tan(x) + m$ (obtaining the height of a point from its elevations viewed from another point) in a few minutes even when Pentium 350MHz computer is used. After that, some local search methods for the previous study are introduced in English because the original paper is only written in Japanese. In the rest of this paper, Boolean function identification by using tree chromosome structure in GA is described and shown the results.

2. Function Identification

The purpose of symbolic regression is to obtain a function's form, in many cases, with coefficients from the observed data. Before the proposed GA approach with sophisticated chromosome structure, automatic identification of unknown functions was a monopoly of GP. The method introduced in this paper allows symbolic regression and various identifications by the implementation of the tree structure as a chromosome, which is called Tree Chromosome Structure (TCS), of GA based approach. Three chromosome structure consists of function chromosome, pointer chromosome, constant chromosome, and operator chromosome. These are shown below.

3. Tree Chromosome Structure

One of the main points of the study is the unique chromosome structure (refer to Figure 1). GA can find the form of the target function by using it as a chromosome. For example, in Figure 1, when the fourth gene of the function chromosome is $Fm[4] = 2$, and the corresponding gene of the pointer chromosome is $Pm[4] = 4$, then the operator "÷" which is division symbol is selected as the gene of the operator chromosome. That is the "divide" operation is set.

$$f(x) = \begin{cases} odd(Refer\ to\ variable\ chromosome\ or\ constant\ chromosome\ set) \\ even(Refer\ to\ operator\ chrosome) \end{cases} \quad (1)$$

$$F_m[f] = \begin{cases} 0\ (f\ :\ odd,\ Refer\ to\ variable\ chromosome) \\ 1\ (f\ :\ odd,\ Refer\ to\ constant\ chromosome) \\ 2\ (f\ :\ even,\ Refer\ to\ operator\ chromosome) \end{cases} \quad (2)$$

$$P_m[f] = \begin{cases} \in V_c\ (f\ :\ odd,\ V_c \in Positive\ integer) \\ \in C_c\ (f\ :\ odd,\ C_c \in Positive\ integer) \\ \in O_p (f\ :\ even,\ 0 \leq\ O_p \leq\ 5) \end{cases} \quad (3)$$

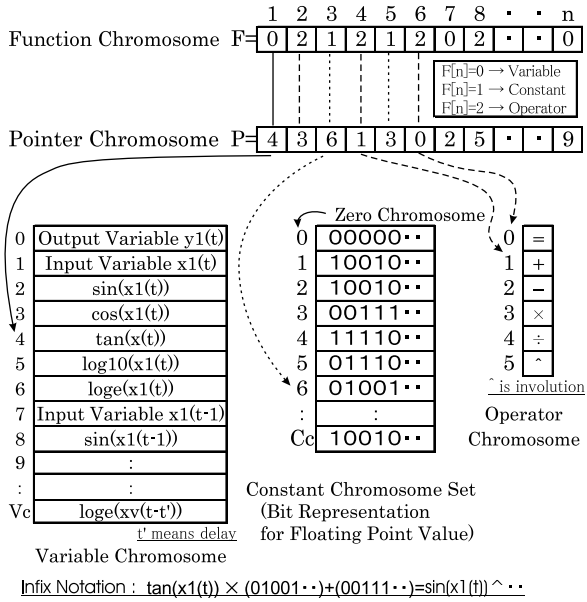

Figure 1. The tree chromosome structure.

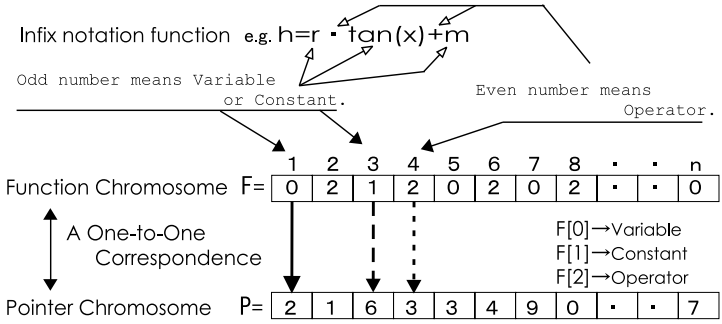

Figure 2. The Function Chromosome and Pointer Chromosome.

Here, $m \in$ natural number, $n \in$ positive odd number, V_c: variable chromosome length, C_c: number of representable constants, O_p: operator chromosome length.

The basic operators of GA are applied to the function chromosome, the pointer chromosome, and the constant chromosome set.

3.1. Function chromosome and pointer chromosome

The function chromosome is described by infix notation without parentheses (refer to Section 6). The pointer chromosome and the function chromosome have a one-to-one correspondence. The correspondence by the function chromosome and the pointer chromosome makes a form of function (see Figure 1, Figure 2).

3.2. Variable Chromosome

The variable chromosome is a fixed length chromosome and does not itself undergo GA operations. It is generated according to the observed data.

3.3. Operator Chromosome

Tree-Chromosome structure has one operator chromosome that cannot allow GA's operators such as selection. The operator chromosome has six genes including "=" and that is an irrevocably fixed chromosome (refer to Figure 1). Here " ^ " means power function. The operators are defined to be left-associative, and each gene of operator chromosome takes mathematical operation priority as follows: $+,- < \times,\div < \hat{} < =$.

3.4. Structure of constant chromosome set

The structure of the constant chromosome set is a two-dimensional array. Rows in this array represents different constant floating-point value (refer to Figure 3). This chromosome is manipulated by GA operators to get the appropriate constant value.

$$\text{Chromosome: } \underbrace{\overset{i=m}{0} + \cdots + \overset{i=1}{1} + \overset{i=0}{0}}_{Integer\ part:\ m\ bits} \ \underbrace{\overset{j=1}{0} + \overset{j=2}{1} + \cdots + \overset{j=n}{0}}_{Floating\ point part:\ n\ bits}.$$

The decimal notation: $0 \cdot 2^m + \cdots + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + \cdots + 0 \cdot 2^{-n}$.
Here, $m, n \in$ natural number.

Figure 3. Representation of constant chromosome with decimal notation.

4. The genetic operators and the fitness function

To find appropriate function, the proposed method applies the simple well-known basic GA operators (selection, crossover, and mutation) to the function chromosome, pointer chromosome, and the constant chromosome sets. The three basic operators on each chromosome adopt the following methods consistently:

Crossover method: One point crossover.
Mutation method: Compulsory conversion of gene.
Selection method: Elite preservation method.

Here, the function chromosome and the pointer chromosome have a one-to-one correspondence. Therefore, the crossover point is made the same. GA operators are repeated until the end condition, such as an iteration is set in advance or the convergence condition calculated by the fitness function given in (4), is satisfied. The elite solution survives reliably into the next generation by the fitness. Then, there is no change effected by GA operators because the elite solution's function chromosome,

$$Fittness = \sum_{i=0}^{n} \frac{1.0}{(|R_i - Observed\_data_i| + 1.0)} \tag{4}$$

(R: output of a chromosome, n: number of observed data)

5. Experiment

5.1. Experimental function

The test function and observed data used for the experiments are as follows:

(a) $Y = \frac{1}{2}gt^2$. (the law of free fall, $g = 9.8$)
(b) $r^2 = x^2 + y^2$. (circle at origin)
(c) $h = r \cdot tan(x) + m$. (refer to Section 1 and Section 3)
(d) $Y = x_1 \cdot x_2 \cdot x_3 - x_4 \cdot x_5 \cdot x_6$. (two-box problem [10])
(e) $Y = (x+z)^{(x-z)}$. (calculation with parentheses)
(f) $Y = x + (w-z)f(x)$. (user-defined function:$f(x) = (x + \frac{1}{x})(x^2 + ln(x)), x \neq 0$)
(g) $a_{n+2} = a_{n+1} + a_n$. ($n \in$ positive integer, $a_0 = 1, a1 =_1$: Fibonacci sequence)
(h) $a_{n+1} = \alpha \cdot a_n(1 - a_n)$.
($n \in$positive integer, $0 < a_0 < 1.0, 3.57 < \alpha \leq 4.0$:chaos phenomenon in logistic map)

Table 1. Restrictions for Test Functions.

Function	Observed variable	Unknown variable	Noise	Note
(a)	Y, t	$\frac{1}{2}, g$	$Y \pm 1 \sim 5\%$	$0 \leq t \leq 24(s)$ $g : 9.8(m/s^2)$
(b)	r, x, y		$r \pm 1 \sim 5\%$	r : radius $\lvert x, y \rvert \leq 10$
(c)	h, r, x	m	$h \pm 1 \sim 5\%$	x : elevation $0 \leq r \leq 10$ m : 1.5 view height
(d)	$Y, x_{1\sim 6}$		$Y \pm 1 \sim 5\%$	$0 \leq x_{1\sim 6} \leq 10$
(e)	Y, x, z		$Y \pm 1 \sim 5\%$	$-5 \leq z - x \leq 10$
(f)	Y, x, z		$Y \pm 1 \sim 5\%$	$0 \leq x \leq 10$ $0 \leq z \leq 10$
(g)	a_n		$a_n \pm 1 \sim 5\%$ Here, $2 \leq n$	$n = 0 \sim 24$ $a_0 = 1.0$ $a_1 = 1.0$
(h)	a_n	α	$a_n \pm 1 \sim 5\%$ Here, $2 \leq n$	$\alpha = 3.8$ $a_0 = 0.2$ $n = 0 \sim 24$

Note: An unknown variable means the data which cannot be observed independently.

5.2. Objectives

Function identification succeeds when the following objectives are achieved (see Section 5.1):

(a): Obtain the law and unknown constant value.
(b): Obtain a complete solution when both positive and negative one-time solutions are given as input.
(c): Obtain a complete solution which contains the transcendental function and unknown constant value (view height).
(d): Obtain the complete solution when there are a lot of observation variables.

(e): Obtain the approximation solution of factorized functions.
(f): Obtain a complete solution of the function which contains the user-defined function.
(g): Obtain a complete solution of the Fibonacci sequence (time series function).
(h): Obtain a complete solution and the constant value (= 3.8 for a fixed chaotic behavior) of the logistic map.

5.3. Experimental conditions

In all experiments, the upper bound of generations is 5000. Each mutation rate of the function chromosome, the pointer chromosome, and constant chromosome is 10%. The number of individuals in the function chromosome and pointer chromosome are $F_m = P_m = 500$. The number of constant chromosome individuals is $C_c = 200$. The constant chromosome length is $T = 10$ (5 bits for integer part, 5 bits for after the decimal point). Then experimental runs were executed for each problem in Table 1. The number of input data was 25 sets of the observed data. The experiment ends when the iteration reaches the upper bound or the absolute error value of Eq.(4) is $|R_i - Observed\_data_i| < 0.05$.

5.4. Experimental results: No noise

Experimental results are indicated in Table 2; the proposed method identifies the target functions at very high speed (Pentium II 350 MHz). In (a) and (c), the approximate value of unknown constant $\frac{1}{2} \cdot g$ and, in consideration of the end condition, accurate value of m are obtained.

Table 2. Results, (No Noise in Observation Data).

	The best detected solution			Average of detected solutions (include approximation solutions)			
Func.	Detected Function	Time (s)	Detected generation	Number of times detected	Error	Time (s)	Detected generation
(a)	$t \cdot 19.96/4.062 \cdot t$	155.0	574	$1 + 8^* + 1^\sharp$	0.112	432.8	1558.3
(b)	$x \cdot x + y \cdot y$	1.84	18	10	0	6.9	59.7
(c)	$r \cdot tan(x) + 1.531$	494.96	2682	$1 + 6^* + 3^\sharp$	0.024	270.2	1430.4
(d)	$x_1 \cdot x_2 \cdot x_3 - x_4 \cdot x_5 \cdot x_6$	11.04	119	$6 + 3^\sharp$	0.005	103.9	919.4
(e)	–	–	–	–	–	–	–
(f)	$x + w \cdot f(x) - z \cdot f(x)$	30.25	721	10	0	75.7	1681.5
(g)	$a_{n+1} + a_n$	0.14	0	$9 + 1^\sharp$	0	0.6	2.8
(h)	–	–	–	–	–	–	–

Note: * is a solution when the rounding error is considered. # is an approximate solution.

The proposed method with tree chromosome structure has succeeded in calculating values of constants with high accuracy (although this is difficult in GP). However, the identification both of (e) and (h) have failed. All detected functions in (h) are oscillatory functions (e.g., $a_{n+1} = a_n^{4.875}$). Indeed, the logistic map oscillates in the range $1 + \sqrt{6} < \alpha \leq \alpha_* (\simeq 3.57)$. The reason why the method could not find it is later mentioned (refer to Section 5.6). In regard to (e), the method is insufficient to solve it. To get the solution of factorized functions, a sensible approach has been also proposed (refer to Section 6).

5.5. Experimental results: Noise

Experimental results when the observed data contains noise are shown in Table 3. The proposed method identifies functions (a), (b), (c), (f), and (g), and succeeds in obtaining of constant values with high accuracy. In the real world, noise is often included in the observed data. In such cases, this method obtains reasonable results.

Incidentally, a solution of the form $x_1 \cdot x_2 \cdot x_3 - x_4 \cdot x_5 \cdot x_6 + \beta$ appeared in all experiments in (d). Because the error value ($\cong 5.02$) is small, β is a kind of an extra expression which absorbs and/or represents noise term.

Table 3. Results, (Noise in Observation Data).

	The best detected solution			Average of detected solutions (include approximation solutions)		
Func.	Detected Function	Error	Time (s)	Number of times detected	Error	Time (s)
(a)	$t/10.250 \cdot 28.219/9.875 \cdot t$ $-t \cdot t/3.031$	35.16	1408.32	$7^* + 3^\sharp$	35.36	1425.03
(b)	$x \cdot x + y \cdot y + y/19.562 \cdot y$	1.06	912.67	9^*	1.41	915.72
(c)	$r \cdot tan(x) + 1.5$	0.97	609.34	$3 + 4^* + 3^\sharp$	0.99	581.37
(d)	–	–	–	$10^\sharp$	5.02	581.599
(e)	–	–	–	–	–	–
(f)	$1.688 + w \cdot f(x) - z \cdot f(x) + x$	55.99	238.59	$4^* + 4^\sharp$	56.23	219.10
(g)	$0.906 \cdot a_n$ $+a_{n+1}/27.031/30.906 + a_{n+1}$	463.14	1229.66	$3^* + 5^\sharp$	434.17	1235.948
(h)	–	–	–	–	–	–

Note: * is a solution when the rounding error is considered. # is an approximate solution.

5.6. Logistic map experiment

In Sections 5.4 and 5.5, the identification of the logistic map failed. This indicates falling into local solution, however, that is not so far from original function, of a wide search space. Two results were as follows:

No noise: $a_{n+1} = 3.938 \cdot a_n - 3.962 \cdot a_n \cdot a_n$ (detecting rate 10%, error: 0.017).
Noise: $a_{n+1} = 3.969 \cdot a_n - 4.031 \cdot a_n \cdot a_n$ (detecting rate 10%, error: 0.044).
Here, variable numbers are rounded off to four decimal places.

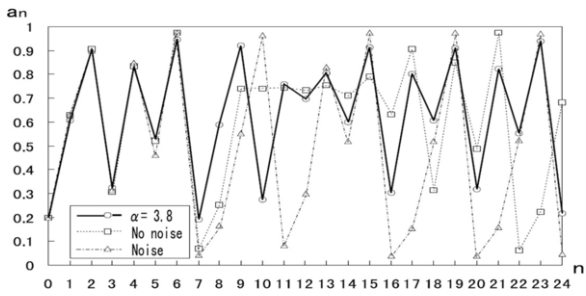

Figure 4. Comparison of Predicted and Actual Values for Test Function(h), The Logistic Mapping.

However, the obtained value for α, which determines the chaotic behavior, is not 3.8 exactly. The butterfly effect makes prediction difficult. In original paper, the constant chromosome cannot make constant value $\alpha = 3.8$ because five bits for decimal value has been assigned unfortunately. Hence, in this experiment, $\alpha = 0.813$ or $\alpha = 0.781$ should have been assigned.

6. Hierarchical Tree Chromosome Structure

The tree chromosome structure identified functions (f) and (h) in their expanded forms. However, the identification of (e) failed because its expansion was not possible. In order to identify general expressions in their factorized form, the tree chromosome structure needs to be expanded.

6.1. Hierarchical tree chromosome

The function chromosome shown in Section 3 is without parentheses. When one function chromosome is regarded as a formula in a pair of parentheses, that is, it is factorized form, the general expression can be obtained by connecting the factorized forms with operators. Thus, a new chromosome called HTCSF (Hierarchical Tree Chromosome Structure for Function) has been designed and proposed (see Figure 5). This method can represent a complex form function by hierarchizing.

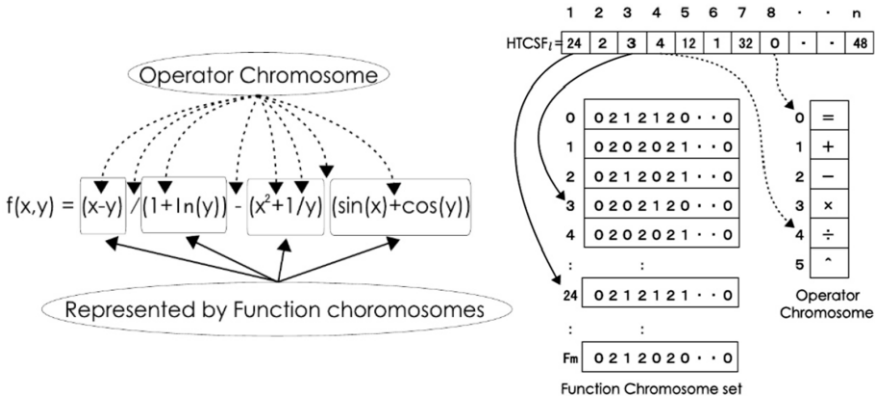

Figure 5. Representation of the hierarchical tree chromosome structure for factorized function chromosomes.

6.2. Experiments using HTCSF

The experimental results for using HTCSF are shown in Table 4. Reliable identification of (5) is seen; however, the identification of expandable functions failed. To get appropriate function formula in reasonable time, local search method might be efficient. In next Section 7, the proposed method with some local search methods get good results are shown.

Table 4. Results for Identification of Factored Expressions. Using HTCSF.

	The best detected solution				Average of detected solutions			
Func.	Detected Function	Error	Time (*s*)	Detected generation	Number of times detected	Error	Time (*s*)	Detected generation
(e)	$(x+y)^{(x-y)}$	0.0	1.21	5	10	0.0	3100.69	1085.8
(f)	–	–	–	–	–	–	–	–
(h)	–	–	–	–	–	–	–	–

7. Local search methods to Identify function

Tree Chromosome Structure succeeds in function identifications. However, there are not exact solution in the results. For example, in Table 2 the results of target function (c) are one exact solution, six solutions with rounding error, and three approximate solutions. Thus, to get more sophisticated form of target function, some local search methods have been proposed [7] (written in Japanese). Of course, the proposed local search methods also aim at the identification success rate improvement and shortening identification time.

7.1. Local search methods

Three different Local Search Methods for Tree Chromosome Structure (LSMs-TCS) have been proposed in [7]. And each LSMs-TCS takes two moving strategies. Therefore, in total, there are six different approaches to find good solution. In other words, six methods to avoid trapping into local solution are given. LSMs-TCS is the method which can change variable, constant value, or operator compulsory. The first method changes variable or constant value to another variable. Second method switches an operator to an another operator including power operator. The last one exchanges variable or constant value for ZERO. Here, the behavior of LSMs-TCS is shown in as follows:

- Compulsory changing procedure for variable and/or constant to variable
  This local search method changes variable and/or constant to another variable compulsory, and simultaneously changes the related gene of the pointer chromosome to a gene of variable chromosome so that one-to-one correspondence is kept.

  e.g., when $f(x,y) = x \cdot y + y \cdot y$ is found during function identification, this local search method is able to change the y of $x \cdot y$ to x and get the right form of circle at origin function $f(x,y) = x \cdot x + y \cdot y$.

- Compulsory changing procedure for variable and/or constant to ZERO
  The behavior of this local search method is to replace the target variable and/or constant by ZERO. This is a special case of method mentioned above.

  e.g., when $f(x,r,m) = m \cdot r + tan(x) + x - 1/x^2$ is obtained by the process of function identification, $f(x,r,m) = m \cdot r + tan(x) + 0.0 - 0.0/x^2$ is made by this local search method by replacing x and 1 with 0.0 respectively.

- Compulsory changing procedure for operator
  This local search method changes operator to another operator. However, if the value of right side of the target operator is near ZERO, then the target operator

never changes to division operator because of to avoid dividing by ZERO.

e.g., when $f(x,r,m) = m \cdot r + tan(x) + x \cdot 0.0$ was gotten by during function identification, the local search method could obtain the right function formula by changing the multiplication operator of $m \cdot r$ to addition operator, and the addition operator of $r + tan(x)$ to multiplication operator. Thereby, the found function expression finally becomes $f(x,r,m) = m + r \cdot tan(x) + x \cdot 0.0$. The found function formula has a redundant part which is multiplied by ZERO in this case. Incidentally, the multiplication operator of this redundant part is never exchanged by division operator. Here, when m is the height of another point, r is the distance from the another point to the point directly underneath of the objective point, and x is the angle of elevation from another point. Then, the found function gets the height of an objective point from its elevations viewed from another point.

Each LSMs-TCS can take two moving strategies. The one of moving strategy is Fast Admissible Move Strategy (FAMS) which changes the solution to another good solution immediately when a better fitness individual is gotten by applying the local search method. The other strategy called the Best Admissible Move Strategy (BAMS) takes the Best solution from the neighborhood solution sets. The both strategies are repeated until there is nothing of exchangeable object in neighborhood.

In original paper [7], the six approaches are described in detail but are written in Japanese. However, hereinto, for want of space, the essence of two strategies are given instead of the detailed description of each six approaches. The essence of local search methods with moving strategies are shown as follows:

Step1: Take an individual at random from the population.
Step2: Take a genetic locus on the function chromosome and apply LMS. Simultaneously, change the gene of pointer chromosome because of one-to-one correspondence.
Step3: Calculate the fitness.

In FAMS;
Step4: If the fitness is improved, then the alteration is fixed, and back to Step 2.
Step5: If there is no good individual in neighborhoods, then quit the FAMS operation.

In BAMS;
Step4: If an improved fitness is obtained, then the altered individual is copied as the best so far.
Step5: Restore to the previous individual, and back to Step 2.
Step6: If there is no good individual in neighborhoods, then quit the BAMS operation.
Step7: Replace the best as a new individual to the previous individual.

7.2. New fitness

Differences of fitness function make some interesting differences for obtained solutions, which were reported and cleared in [7]. Specifically, when the identification system uses the original fitness function (4), undesirable approximate solutions are obtained in many cases (refer to Table 2, Table 3). Hereupon next two fitnesses have been tested.

$$Fitness\_2 = \frac{1}{n}\sum_{i=1}^{n}\frac{|input(i) - output(i)|}{|input(i)|} * 100. \tag{5}$$

$$New\_fitness = max\left(\frac{|input(i) - output(i)|}{|input(i)|} * 100\right). \tag{6}$$

($input(i)$: i-th observed datum, $output(i)$: i-th output datum obtained by the system)

As a result, a new very simple fitness function (6) has been introduced in [7]. In formula (4) and (5), it is cleared that more observed data are given, the more averaging fitness is obtained because the fitness is divided by the total number of observations n. In other words, it is reasonable to assume that approximate solutions might be brought instead of the right solution.

Table 5. Testing results when the fitness value functions are different.

	Fitness function	
Obtained function	(6)	(5)
$Y = \frac{1}{2}gt^2$	47	22
$Y = \frac{1}{2}gt^2 \pm \beta t$	1	14
$Y = \frac{1}{2}gt^2 \pm \beta$	0	4
$Y = \frac{1}{2}gt^2 \pm \beta t + \gamma$	0	9
Don't find	2	1

β, γ: constant value

7.3. Experimental conditions

In all experiments, the limited time is 5 minutes. Therefore, the upper bound of generations does not set. The evaluation of convergence is $New\_fitness < 5\%$ The number of individuals in the function chromosome and pointer chromosome are $F_m = P_m = 100$. The number of constant chromosome individuals is $C_c = 50$. The initial length of function chromosome and pointer chromosome is 31. The constant chromosome length is $T = 10$ (5 bits for integer part, 5 bits for after the decimal point). The length of operator chromosome is 6 including equal operator. Each mutation rate of the function chromosome, the pointer chromosome, and constant chromosome is 30%. The rate of crossover for function chromosome and pointer chromosome, and constant chromosome is around 70% because of excluding mutated chromosomes. To keep diversity of population, the most of chromosomes excluded the elite take gene manipulation in each generation. The proposed method has three exception processes are shown as follows:

- Multiple power operator prohibition: e.g. $f(x) = x^{3^5}$
- Overflow prevention: e.g. $f(x) = x \cdot 100^{1000}$
- Zero-divide prevention: e.g. $f(x) = x \cdot (1/100)^{1000}$

7.4. Selection from LSMs-TCS

It is necessary for the proposed GA-based identification system to take one of LSMs-TCS in each generation. However, it is not known in advance which LSM is the best one

to get good solution at the moment. Therefore, the experiments (W), (F), and (B) were performed in [7]. Here, (W) means with no LSMs and is for the verification experiment, (F) uses local search method with FAMS, and (B) uses local search method with BAMS. In the experiments, the only elite chromosome is chosen for manipulation by LSMs-TCS because many "0.0" gene were frequently appeared in the population in preliminary experiments. Even under such conditions, "0.0" gene were often made in the experiment. For example, the "0.0" gene appearance frequency of function identification for (b) were 3/69 in (W), 44/73 in (F), and 57/80 in (B) (refer to Table 6).

7.5. Target functions for experiments

The target functions of identification are composed of (a)-(h) excluding (e) in Section 5. And a new famous natural law called Kepler's the third law, that is $T = k \cdot a^{2/3}, k \cong 1$, is added as (e).

7.6. Experiment using local search methods

The experimental results using proposed local search methods are shown in Table 6 and Table 7. Table 6 and Table 7 show better results in comparison with original identification method ((W) in tables) in Table 2, 3, and 4. Table 6 shows that LSMs-TCS gets good results in many cases excluding identification for function (c). Table 7 shows LSMs-TCS gets good improvement rate in finding time and the good results of statistical test. Here, +, * symbols in Table 7 are the results of Wilcoxon rank-sum test which is a Non-parametric test of mean difference. (F)-imp. and (B)-imp. are the improvement rate.

Table 6. Number of successes and improvement rate.

Func.	(W)	(F)	(B)	(F)-imp. (%)	(B)-imp. (%)
(a)	88	94	94	6.8	6.8
(b)	69	73	80	5.7	15.9
(c)	28	26	17	-7.1	-39.2
(d)	82	82	78	0.0	-4.8
(e)	95	99	100	4.2	5.2
(f)	0	1	2	-	-
(g)	53	59	68	11.3	28.3
(h)	0	0	0	-	-

Table 7. Average times(sec.) and improvement rate.

Func.	(W)	(F)	(B)	(F)-imp. (%)	(B)-imp. (%)
(a)	11.60	8.76*	8.75**	24.4	24.5
(b)	30.71	10.41**	11.72**	66.1	61.8
(c)	163.71	146.28	124.05	10.6	24.2
(d)	53.43	71.19^{++}	88.95^{++}	-33.2	-66.4
(e)	15.48	16.86	19.61	-8.9	-26.6
(f)	-	90.68	275.77	-	-
(g)	1.16	0.89	0.64**	23.2	44.8
(h)	-	-	-	-	-

*,+ / **,++: significance level 5%/1%(two-tailed test).

Table 8. Average length of expression of successfully identified functions.

Func.	ML	(W)	(F)	(B)	(F)-imp. (%)	(B)-imp. (%)
(a)	5	6.63	6.51	7.23	1.8	-9.0
(b)	7	9.17	11.57	12.30	-26.1	-34.1
(c)	5	8.42	8.23	9.47	2.2	-12.4
(d)	11	11.17	11.75	12.15	-5.1	-8.7
(e)	3	4.43	4.49	4.86	-1.3	-9.7
(f)	9	-	27.0	11.0	-	-
(g)	3	4.01	4.32	5.23	-7.7	-30.4
(h)	9	-	-	-	-	-

ML:Minimum Length of function.

Table 8 shows that when LSMs-TCS is used, the length of found function become longer. For example, in the meaning of function, both function $f(x,r,m) = m \cdot r + tan(x)$ and $f(x,r,m) = m \cdot r + tan(x) + 0.0 - 0.0/x^2$ are the same. However, the length of former function is 5, but the later one becomes 13. It is certain that the found function has some 0.0 term more than expected when zeroizing method of LMSs is used.

8. Boolean function identification

The target Boolean functions have single output and single/multi variable input. The identification of a Boolean function is different from function identification through symbolic regression in the following respect: Abbreviated forms of Boolean function are critical when constructing circuits, and one aim of the proposed method is to produce the most efficient Boolean representation. To find Boolean function by the proposed approach, Tree Chromosome Structure and Hierarchical Tree Chromosome Structure described above have to be modified for Boolean function identification.

8.1. Tree chromosome structure for identifying Boolean function

Figure 6 shows that the modified chromosome structure has one function chromosome and one pointer chromosome of one-to-one correspondence, one variable chromosome, and one operator chromosome. The variable chromosome is composed only of the variable and its negative form. The operator chromosome has (NAND, NOR, AND, OR, =). Here, Boolean operators are defined to be left-associative, and their order of priority is as follows: $\uparrow, \downarrow < \cap, \cup < =$. The factors and function chromosome is shown in Figure 7. Figure 8 is the HTCSF for Boolean function.

8.2. Fitness for identifying Boolean function

The fitness is calculated by the following two target functions. Here, $Fitness\_b1$ means the fitness until GA detects a complete form. Only when the derived expression is a complete form, is $Fitness\_b2$ used. In the Boolean function identification, the aspired goal is a complete expression and length minimization. The former is essential as an existence of error in a detected Boolean function is fatal. One the other hand, the latter is also an important concept when considering the manufacturing costs for an actual electric circuit.

$$Fitness\_b1 = \frac{1}{n}\sum_{i=1}^{n}\frac{1.0}{1.0+\alpha\cdot|R_i - Truth\_table_i|} \tag{7}$$

$$Fitness\_b2 = (7) + \frac{1.0}{(LDF+1.0)} \tag{8}$$

(n: Truth table, R: Output of detected function, α: constant > 0, LDF: Length of Detected function)

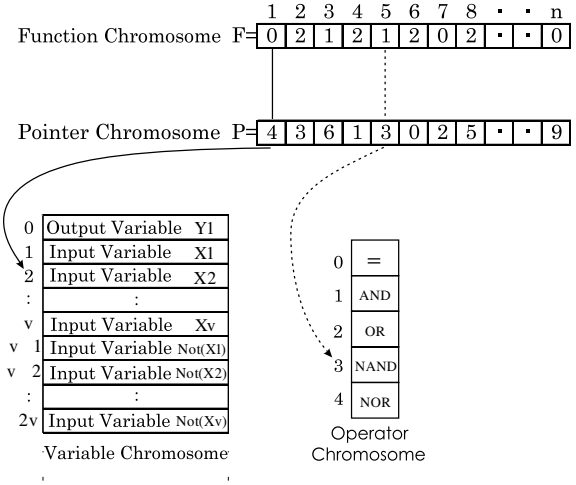

Figure 6. TCS for Boolean function identification.

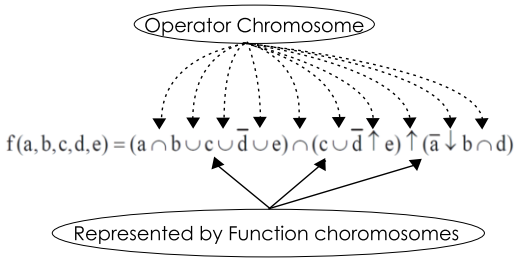

Figure 7. Factors and function chromosome.

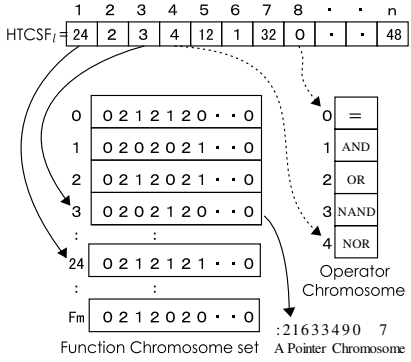

Figure 8. Representation of the HTCSF for Boolean function.

8.3. Experiments

The test functions used for the experiments in the paper are as follows:

(A) $f = a \oplus b$ (Exclusive - OR)
(B) $f = \overline{b} \cap \overline{c} \cup a$
(C) $f = (a \cup \overline{b}) \cap (a \cup \overline{c})$
(D) $f = (a \uparrow b \downarrow c \cup d)$
(E) $f = (a \cap b \cup c \cup \overline{d} \cup e) \cap (c \cup \overline{d} \uparrow e) \uparrow (\overline{a} \downarrow b \cap d)$
(F) $f = (a \cap b \cup c \cup \overline{d} \cup e) \cap (c \cup \overline{d} \uparrow b) \uparrow (\overline{a} \downarrow b \cap d)$
(G) $f = (a \cap b \cup c \cup \overline{d} \cup e) \cap (c \cup \overline{d} \uparrow d \cap e) \uparrow (\overline{a} \downarrow b \cap d)$

Where (B) and (C) have the same truth tables; (B) is the disjunctive canonical form, and (C) is the conjunctive canonical form. (D), (E), (F), and (G) are Boolean functions prepared for experiment, possible abbreviations for wich are unknown. The truth tables of the above functions act as the input observation data for the identification system.

8.4. Condition and results

Conditions are indicated in Table 9. In all experiments, the upper bound of generations is 1000. Mutation rates of both the function chromosome and the pointer chromosome are 10%. The number of individuals in the function chromosome and the pointer chromosome are $F_m = P_m = 500$. Further, each chromosome length is 31, and $\alpha = 5.0$. Ten experimental runs were executed for each problem in Table 9. The number of input data varied with the size of the relevant truth table. The experiment ends when the generation reaches the upper bound or the value of $Fitness\_b2$ equal or exceeds 1.1. Here, 'Rate of [1](%)' means the appearance rate of output 1 in each truth table, and 'Variables' means the total number of variables (the sum of input and output variables).

Table 9. Experimental Examination Conditions for Three Chromosome Structure.

Function	(A)	(B)	(C)	(D)	(E)	(F)	(G)
Variables	3	4	4	5	6	6	6
Rate of [1](%)	50.0	62.5	62.5	6.25	81.25	78.125	87.5

Table 10 indicates that the proposed method succeeds in the identification of abbreviated forms of Boolean functions. However, functions (E) could not be identified, despite function (E) being very similar in form to (F) and (G) (refer to Section 8.3).

Table 10. Results for identification of factored expression of Boolean functions. : Using HTCSF.

	The best detected function				Average of detected functions		
Number	Function	Length	Generation	Time(s)	Length	Generation	Time(s)
(A)	$a \downarrow b \downarrow a \cap b$	7	5	0.19	8.0	38.8	1.27
(B)	$\overline{a} \uparrow c \cup b$	5	0	0.06	6.4	36.6	2.1
(C)	$\overline{b} \cap \overline{c} \cup a$	5	1	0.11	6.6	36.0	2.06
(D)	$\overline{d} \uparrow b \downarrow c \cup \overline{a}$	7	0	0.10	7.6	127.4	13.34
(E)	–	–	–	–	–	–	–
(F)	$\overline{e} \uparrow \overline{c} \downarrow e \cup \overline{d} \downarrow \overline{a} \uparrow \overline{b}$	11	10^3	198.2	19.4	10^3	199.1
(G)	$d \downarrow \overline{e} \uparrow a$	5	0	0.20	6.0	61.8	12.6

9. Conclusion and future work

This paper introduces the studies of function identification by simple GA with Tree Chromosome Structure, and also aims to show and to make understand some directly related studies written in Japanese. The unique point of the introduced GA-based identification approach is the Tree Chromosome Structure. By using the chromosome structure, GA can identify algebraic functions, primary transcendental functions, time series functions including chaos function, user-defined one-variable functions, and Boolean functions. The results show good performances in each function identification, so that it is not too much to say that the introduced approaches are so powerful.

An eventual research topic is to detect concealed laws from observed raw data such as empirical data of engineering, psychological, social research, and so on, by the use of the introduced methods that still remains to be done as a future work.

References

[1] D. E. Goldberg. Genetic Algorithms in Search, Optimization, and Machine Learning: Boston, MA: Addison-Wesley; 1989. 372 p.
[2] Mitsukuni Matayoshi. A Method for Function Identification using a Genetic Algorithm. IEICE Technical Report. 1998;98(199):25-31. (in Japanese)
[3] Mitsukuni Matayoshi. Three Chromosome Structure in a Genetic Algorithm to Identify Functions. IEICE. 1999 Nov; J82-D-I:1327-1335. (in Japanese)
[4] Mitsukuni Matayoshi. Boolean function identification in GA with Tree-chromosome structure. Proceedings of the 13th Annual Conference of JSAI; 1999 Jun 16-18; JSAI; p. 248-250. (in Japanese)
[5] Mitsukuni Matayoshi. Three Chromosome Structure in a Genetic Algorithm to Identify Functions. Systems and Computers in Japan. 2000; 31(10): 32-40.
[6] Mitsukuni Matayoshi. Boolean function identification in GA with Tree-chromosome structure. The Journal of General Industrial Research. 2002 Mar;10;1-8.
[7] Mitsukuni Matayoshi, et al. Local Search Methods for Tree Chromosome Structure in a GA to Identify Functions. IEEJ Transactions on Electrical and Electronic Engineering. 2006 Jan;C126(1):123-131. (in Japanese)
[8] Koza, J. R. Genetic Programming: On the Programming of Computers by Means of Natural Selection and Genetics. Cambridge. MA: MIT Press; 1992.
[9] Hitoshi, Iba et al. System Identification based on structured genetic algorithms. Proceedings 5th International Conference on Genetic Algorithms (ICGA 93); 1993 Jun; Urbana-Champaign, IL. p. 279-286.
[10] Hitoshi, Iba et al. System Identification Approach to Genetic Programming; Journal of Japanese Society for Artificial Intelligence (JSAI). 1995 Jul;10(4):590-600. (in Japanese)
[11] Hitoshi, Iba et al. Temporal Data Processing Using Genetic Programming. Proceedings 6th International Conference on Genetic Algorithms (ICGA 95); 1995 Jul; San Francisco, CA. p. 279-286.
[12] Les M. H and Donna J. D'Angelo. The GA-P: A Genetic Algorithm and Genetic Programming Hybrid. IEEE Intelligent Systems. 1995 Jun; 10(3):11-15.
[13] Gudar J. Ibrahim, Tarik A. Rashid, Ahmed T. Sadiq. Evolutionary DNA Computing Algorithm for Job Scheduling Problem. IETE Journal of Research. 2018 May; 64(4):514-527.
[14] J. M. Abdullah and T. Ahme. Fitness Dependent Optimizer: Inspired by the Bee Swarming Reproductive Process. IEEE Access. 2019;7;43473-43486.
[15] Ahmed S. Shamsaldin, Tarik A. Rashid, Rawan A. Al-Rashid Agha, Nawzad K. Al-Salihi, Mokhtar Mohammadi. Donkey and smuggler optimization algorithm: A collaborative working approach to path finding. Journal of Computational Design and Engineering. 2019 Oct;6(4):562-583.