Fourth International Workshop on Computational Intelligence & Applications
IEEE SMC Hiroshima Chapter, Hiroshima University, Japan, December 10 & 11, 2008

Multi-chromosomal Grammatical Evolution

Akira Hara[*1], Tomohisa Yamaguchi[*2], Takumi Ichimura[*1], Tetsuyuki Takahama[*1]
1) Graduate School of Information Sciences, Hiroshima City University
3-4-1, Ozuka-higashi, Asaminami-ku, Hiroshima, 731-3194 Japan
email:{ahara, ichimura, takahama}@hiroshima-cu.ac.jp
2) Faculty of Information Sciences, Hiroshima City University
Presently with KURE COMPUTING CENTER CO.,LTD.

***Abstract*—Grammatical Evolution (GE) is an evolutionary method for optimizing a program generated by a one-dimensional chromosome and grammatical rules. The grammars consist of terminals, which are items that can appear in the language, and nonterminals, which can be expanded into one or more terminals and nonterminals. The genes are translated into a program based on the grammar. If the genes are used up for generating complete program, the chromosome is wrapped and reused. GE has an advantage that illegal individuals are not generated by the genetic operations. When a certain gene changes, however, the successive genes might be used for the different production rule from the rule applied before even if they are not changed. Therefore, it is difficult to preserve the characteristics of parents. To solve this problem, we propose GE using multiple chromosomes. In this method, multiple chromosomes as many as the nonterminal symbols in the grammatical rules are prepared. A chromosome correspondent to the expanded non-terminal symbol is selected and used for mapping. Moreover, a new technique of the wrapping is also introduced so that the grammatical rules which increase the number of nonterminal symbols can not be applied when the wrapping happens. We performed some experiments, and showed the effectiveness of our proposed method.**

I. Introduction

Genetic Programming (GP)[1], [2], [3] is an evolutionary method for optimizing the program represented by the tree structure. To apply GP to a problem, it is necessary to design the appropriate symbols (terminals and nonterminals) and genetic operators (crossover, mutation and so on), so that individuals with illegal representation for solutions can not be generated in the evolutionary process. Grammatical Evolution (GE)[4], [5] has been proposed in order to get rid of the restrictions and to optimize the arbitrary programs. GE was inspired by the biological process of generating a protein from DNA. The DNA is the sequence of nucleotides, and groups of three nucleotides, called codons, are used to specify the building blocks of proteins. This genotype-to-phenotype mapping is utilized for automatic programming. In GE, a binary bit string is used as a genotype. Therefore, GE is free from the restriction on crossover and mutation. The grammatical rule represented by the BNF (Backus Naur Form) notation[6] is used for the translation from the genotype to the phenotype. The genes are translated into a program based on the grammar. If the genes are used up for generating complete program, the chromosome is wrapped and reused. As a result, it is possible to convert a chromosome into a valid program grammatically.

However, there are some problems in GE. When an gene changes by genetic operations, the production rules derived by the successive genes also change in chain reaction. Therefore, the effective characteristics of the parent individuals can not be preserved. Another problem is the occurrence of invalid individuals. When a genotype of individual is converted into a phenotype by using the grammatical rules, the translation process is finished if all nonterminals are converted into terminals. However, the process occasionally may not be finished due to the endless expansion on nonterminals, where a nonterminal is expanded into two or more nonterminals repeatedly. The individual can not be evaluated. Such an individual is called the invalid individual, because it will never undergo a complete mapping to a set of terminals. The invalid individuals are often observed in the initial population. The occurrence of invalid individuals causes ill effect such as degradation of search performance.

In this research, in order to preserve the characteristics of the parental individuals, we propose an improved GE, which uses the individuals with multiple chromosomes. Each individual has the same number of chromosomes as the kinds of nonterminals in the given grammar. Each chromosome is exclusively for the expansion of the corresponding nonterminal. Therefore, even if a chromosome for an nonterminal changed by genetic operations, the change does not affect the expansion of other nonterminals. Moreover, in order to suppress the generation of invalid individuals, the handling of the rules also changes in our proposed method. When the wrapping occurs, rules which increase the number of nonterminals are eliminated from the rule set. This modification prevents expressions from growing continually, and increases the possibility of generating complete programs. We verify the effectiveness of our proposed method by comparison to the conventional GE.

The contents of this paper are as follows. Section II describes the conventional GE. Then, Section III proposes the multi-chromosomal GE. Section IV shows the experimental results on three kinds of problems. Finally, Section V describes the conclusions and future work.

II. Grammatical Evolution

A. Structure of Individuals in GE

The genotype of an individual in GE is represented by a variable-length binary string as shown in Fig.1. The binary string is separated in every eight bits. Each eight-bit group is

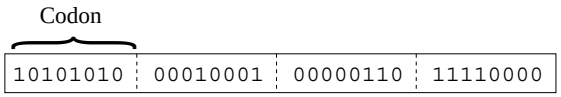

Fig. 1. Individual representation in GE.

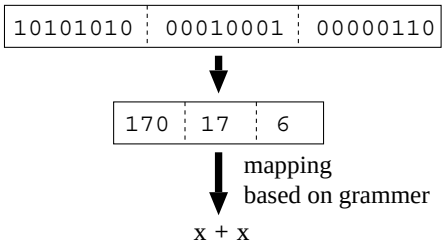

Fig. 2. Mapping from genotype to phenotype.

called codon. That is, an individual in GE is composed of the sequence of the codons.

Each codon represents an integer value. Therefore, the genotype expressed in a binary string is converted into the integer values as shown in Fig.2. Then, production rules are selected based on the integers, and the individual is converted into the phenotype.

Because the binary string is used as the genotype in GE, the genetic operators (*e.g.* crossover, mutation, and selection) for Genetic Algorithms[7], [8] can be used in GE.

B. Mapping from Genotype to Phenotype

In GE, the grammatical rule expressed by the BNF notation is used for the genotype-to-phenotype mapping. As a result, the genotype is converted into the valid program grammatically. A grammar can be represented by the tuple $\{N, T, P, S\}$, where N is the set of nonterminals, T is the set of terminals, P is the set of production rules that maps the elements of N to T, and S is the start symbol that is a member of N. Below is an example BNF which generates the numerical expression.
$N = \{\text{expr}, \text{op}, \text{var}\}$
$T = \{+, -, /, *, (,), \text{x}, 1\}$
$S = \langle\text{expr}\rangle$
P can be represented as follows:

```
(1)<expr>::=<expr><op><expr>   (0)
           |(<expr><op><expr>) (1)
           |<var>              (2)
(2)<op>::=+   (0)
          |-   (1)
          |/   (2)
          |*   (3)
(3)<var>::=x    (0)
           |1    (1)
```

The translation from a chromosome to a program is performed by reading codons of eight bits to generate a corresponding integer value. The rule for expanding a nonterminal is selected based on the integer value and the following

Fig. 3. An example of codon sequence.

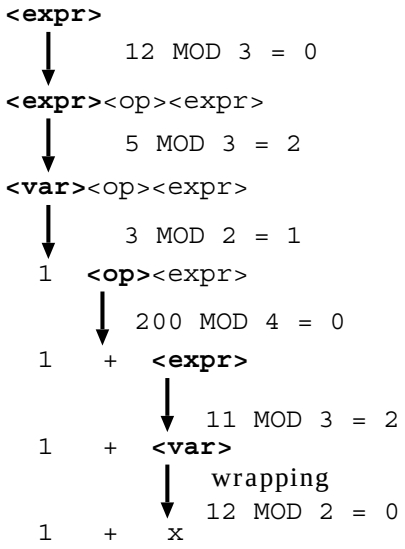

Fig. 4. An example of mapping in standard GE.

mapping function:

$$rule = \frac{\text{(codon integer value)}}{MOD} \\ \text{(number of rules for the current nonterminal)} \quad (1)$$

Considering the expansion of the nonterminal <op>, there are four selectable production rules. If we assume the read codon produces the integer 6, then (6 MOD 4 = 2) would select rule (2) '/' from the rule set for <op>. Each time a production rule has to be selected to map from a nonterminal, successive codon is read. If there are two or more nonterminals in the program, the leftmost nonterminal is dealt with first. For example, in the case of <var><op><exp>, the expansion of <var> is performed first. This mapping process is performed until all the nonterminals are converted into terminals. If an individual runs out of codons for mapping, the codons are reused from the head of the chromosome. This technique is called wrapping. In the biological systems, the gene-overlapping phenomenon has been observed in many organisms[9].

Consider the individual with codons as shown in Fig.3 for an example. Fig.4 shows the mapping process of the individual. The start symbol <expr> is converted into $1+x$ by mapping.

There may be a case where the same production rule is used iteratively and the mapping process can not be finished even after several wrapping events. In this case, the individual with incomplete mapping is regarded as an invalid individual and is given the lowest possible fitness value.

Though each codon in GE can represent 256 distinct integer values, many of these integer values represent the

same production rule. A similar phenomenon can be observed in the genetic code of biological organisms, referred to as degenerate genetic code[9]. This means that subtle changes in the search space (genotype) may have no effect on the solution space (phenotype). This phenomenon has an effect of the maintenance of genotypic diversity throughout a run of GE[10].

C. Problems in GE

Compared with GP, GE has some merits: the guaranty of the generation and preservation of valid programs, and genetic diversity which enables neutral mutation[10]. However, there are some problems in GE. It is difficult to preserve useful characteristics of the parental individuals. When the value of a codon changes by a genetic operator, the production rules selected by the successive genes also change in chain reaction. For example, consider the case in Fig.3. The integer value 5 in the second codon is used for expanding the nonterminal <expr>. When the value of the first codon changes from 12 to 14 by a mutation, however, the second codon is used for expanding the nonterminal <var>. As described in this example, the change of a codon value has a great influence on actions of the successive codons even if they do not change.

Another problem is the occurrence of invalid individuals. The invalid individuals can not be evaluated properly. It is considered that the number of actual effective individuals in population is less than the population size. Therefore, the invalid individuals have a bad influence on the evolution. Especially, the invalid individuals are often observed in the initial population because the initial population is generated at random. This may cause the decline in the search performance in the early stage of evolution.

III. Multi-chromosomal Grammatical Evolution

A. Use of Multiple Chromosomes

In this paper, we propose an improved GE where each individual has two or more chromosomes. In this method, each individual has the same number of chromosomes as nonterminals in a given BNF notation. Consider the case of the production rule P in the previous section as an example. In this case, there are three kinds of nonterminals. Therefore, an individual has three different chromosomes as shown in Fig.5. The first chromosome among them is utilized for expanding <expr>. The second chromosome is used for <op>, and the third one is used for <var>. That is, the chromosomes are switched according to the current nonterminal. This method has an effect of preserving the characteristics of the parental individuals. Fig.6 shows a mapping from the genotype in Fig.5 to phenotype under the grammar P in the previous section. By using multiple chromosomes, the actions of codons for <op> and <var> have been preserved even if a codon for expanding <expr> changed by genetic operators. Therefore, the issue that the characteristics of the parental individuals are greatly lost in the conventional GE is solved.

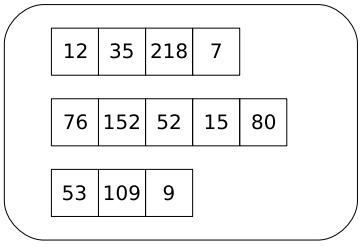

Fig. 5. An individual with multiple chromosomes.

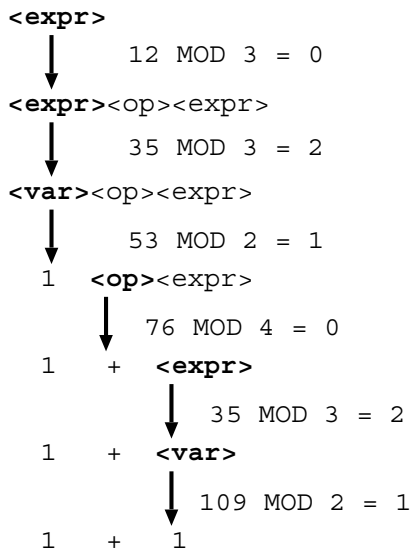

Fig. 6. An example of a mapping in multi-chromosomal GE.

B. Rule Change in Wrapping

An incomplete mapping could arise if the production rules which increase the number of nonterminals were applied over and over. For example, consider the individual where all of codons specify rule (0) of <expr>. In this case, the same mapping rule is applied over and over again, and the individual becomes invalid. In order to solve the problem, the rule set is changed at the occurrence of wrapping so that the rules which increase the number of nonterminals will not be selectable. As a result, the wrapping works efficiently, and it becomes more probable to success the mapping to phenotype. This technique has an effect to decrease the number of invalid individuals.

For example, consider the rule set for <expr> as follows.

```
(1)<expr>::=<expr><op><expr>   (0)
          |(<expr><op><expr>)(1)
          |<var>               (2)
```

If the wrapping occurs, the rules (0) and (1), which increase the number of nonterminals, are deleted. As a result, the rule set is changed as follows.

```
(1)<expr>::=<var>              (0)
```

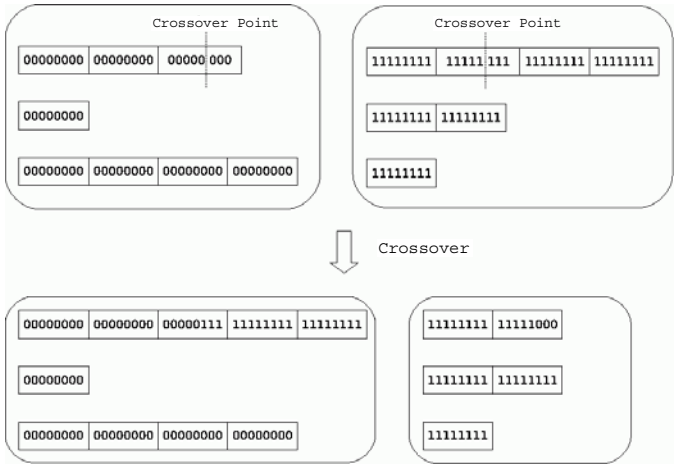

Fig. 7. An example of crossover in multi-chromosomal GE.

TABLE I
PARAMETER SETTINGS.

Parameters	Values
Population size	500
Max generation	51
Crossover rate	90%
Mutation rate	1%
Duplication rate	1%
Max wrapping in each chromosome	10
Max number of codons in each chromosome in initial individual	12
Max number of codons in each chromosome	100
Number of bits for one codon	8
Tournament size	5
Number of elite individual	20

C. Genetic Operations for Multiple Chromosomes

In performing a crossover operation, a couple of chromosomes correspondent to the same nonterminal is chosen at random from two parental individuals. Then, one-point crossover is performed for the chromosomes. Fig.7 shows the case where the first chromosome in each individual is chosen for crossover.

IV. EXPERIMENTS

Three kinds of problems; Symbolic regression, 3 Multiplexer, and Mastermind are used for experiments. Table I shows the parameters for our proposed and conventional methods. These parameter values are based on the previous researches[4], [5], [11], [12], [13].

A. Symbolic Regression

The aim of this problem is to find mathematical expression that represents a given set of input and output pairs. The target function is $g(x) = x^4 + x^3 + x^2 + x$, and the 20 sample points $(x_1, g(x_1)), \ldots, (x_{20}, g(x_{20}))$ in $-1 \leq x \leq 1$ are given for generating the approximate function $h(x)$. The fitness f is calculated by the equation (2).

$$f = -\sum_{k=1}^{20} \mid (g(x_k) - h(x_k)) \mid \qquad (2)$$

The max fitness is $f = 0$ in the case of $g(x_k) = h(x_k)$ for all k.

1) BNF Definition: The grammar used in this problem is given below:
$N = \{\text{expr}, \text{op}, \text{var}\}$
$T = \{+, -, /, *, (,), \text{x}, 1.0\}$
$S = \langle \text{expr} \rangle$
and production rules P can be represented as

```
(1)<expr>::=<expr><op><expr>   (0)
            |(<expr><op><expr>)(1)
            |<var>             (2)
(2)<op>::=+   (0)
          |-   (1)
          |/   (2)
          |*   (3)
(3)<var>::=x    (0)
           |1.0 (1)
```

2) Experimental results: Fig.8 shows the cumulative frequency measure of success over 50 runs of respective methods. The cumulative frequency of success means the number of runs where the optimal solution (target function) can be acquired up to the respective generations. Fig.9 shows the number of invalid individuals in each generation averaged over 50 runs. In these figures, "GE" means the conventional GE, "GE(new-wrapping)" means the GE in which only the proposed wrapping technique is introduced, and "newGE" means the proposed multi-chromosomal GE with the new wrapping technique.

Fig.8 indicates that the proposed method (newGE) has better performance than the other two methods in the frequency of discovering optimal solution. Especially, at the beginning of search, the proposed method and GE with new wrapping method are much superior to the conventional GE. This is because many of individuals can be converted into the valid programs by the wrapping with rule change, though they might become invalid programs in the conventional GE. At initial population, individuals more than the half of the population were invalid ones in the conventional GE. Many invalid individuals also exist in the latter generations. On the other hand, the methods with the proposed wrapping technique scarcely generated invalid individuals. Furthermore, the proposed multi-chromosomal GE is superior to the GE with new wrapping technique in cumulative frequency measure of success. This difference seems to be due to the effect of preserving parental characteristics by multiple chromosomes.

B. 3 Multiplexer

The aim of this problem is to discover a boolean expression that behaves as a 3 Multiplexer. The three inputs (input1, input2 and input3) are binary variables. The input1 represents an address which specifies either data A or data B, and input2 and input3 represent the data respectively. The boolean expression must output the value of data correspondent to the specified address. There are eight fitness cases, representing all possible input-output pairs. Fitness is the number of input

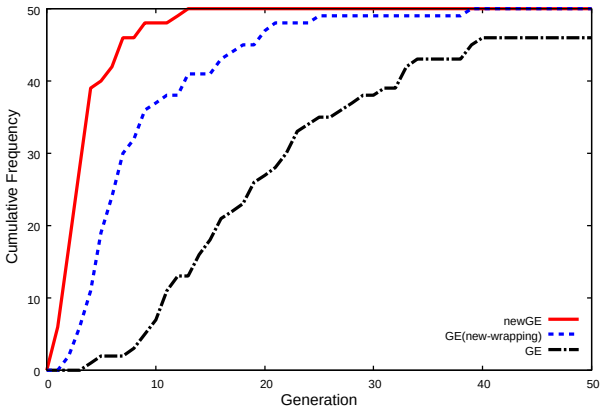

Fig. 8. Cumulative frequency of success measures on the symbolic regression.

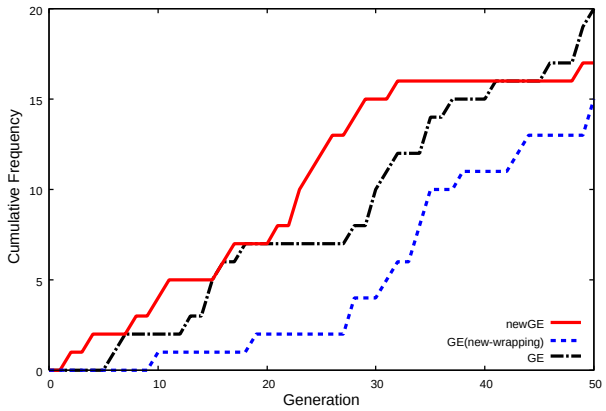

Fig. 10. Cumulative frequency of success measures on the 3 multiplexer.

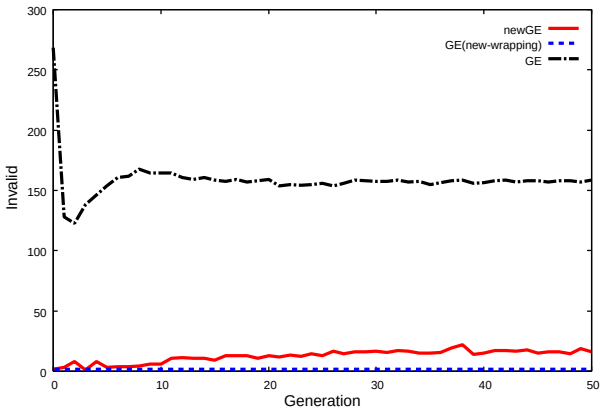

Fig. 9. The number of invalid individuals on the symbolic regression.

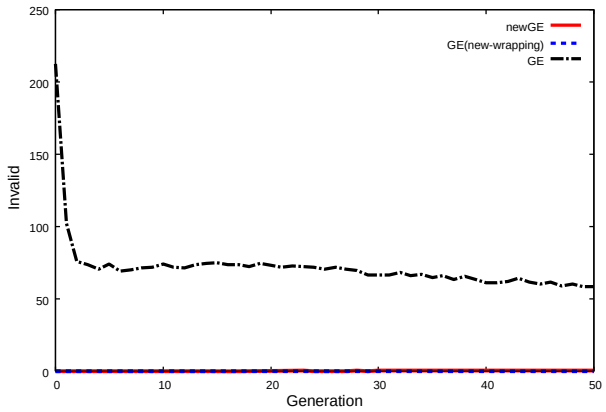

Fig. 11. The number of invalid individuals on the 3 multiplexer.

cases for which the expression returns the correct output. Therefore, the max value of fitness is eight.

1) BNF Definition: The grammar used is given below:
$N = \{\text{bexpr}, \text{bilop}, \text{ulop}, \text{input}\}$
$T = \{\text{and}, \text{or}, \text{not}, \text{input1}, \text{input2}, \text{input3}\}$
$S = \langle \text{bexpr} \rangle$
and production rules P can be represented as

```
(1)<bexpr>::=(<bexpr><bilop><bexpr>)   (0)
            |<ulop>(<bexpr>)           (1)
            |<input>                   (2)
(2)<bilop>::=and   (0)
            |or    (1)
(3)<ulop>::=not    (0)
(4)<input>::=input1   (0)
            |input2   (1)
            |input3   (2)
```

2) Experimental results: Fig.10 and Fig.11 show the results by 50 runs. In the beginning of search, the proposed method (newGE) is superior to the conventional GE in the cumulative frequency of success. In the last stage of search, however, the conventional method shows better performance than the proposed method. The population by conventional GE evolves without stagnation though the performance of initial search stage is lower than the other methods. The reason why the evolution by the proposed method stagnated seems to be as follows: In the proposed method, four chromosomes are used for four nonterminals in the given grammar. The crossover operation is applied only to the couple of chromosomes, and other chromosomes are succeeded to offspring without change. Therefore, it might be difficult to give the drastic change to the population when evolution stagnates. In the conventional GE, more than 50 invalid individuals exist at every generation as well as the symbolic regression problem. On the other hand, in the proposed method and the GE with new wrapping technique, invalid individuals were scarcely generated from the initial generation.

C. Mastermind

This problem is to find the correct combination of colored pins. The instance tackled here uses 4 colors and 10 pins with the following values “4 3 2 4 2 4 3 1 2 4”. Respective figures represent different colors. The fitness of an individual is calculated as follows: The individual receives one point for each pin that has the correct color, regardless of its position. If the colored pin is at the correct position, one point is also added. In addition, if the generated sequence is the completely same as the correct one, a bonus point

is also added. Therefore, the maximum fitness is equal to (the number of figures in correct combination) $* 2 + 1$.

1) BNF Definition: The grammar used is given below:
$N = \{\text{pin}, \text{num}\}$
$T = \{1, 2, 3, 4\}$
$S = \langle \text{num} \rangle$
and production rules P can be represented as

```
(1)<pin>::=<pin><pin>   (0)
          |<num>        (1)
(2)<num>::=1    (0)
          |2    (1)
          |3    (2)
          |4    (3)
```

2) Experimental results: Fig.12 and Fig.13 show the results of 50 runs. In the cumulative frequency of success, the proposed method (newGE) is superior to the conventional GE. In the proposed method, the chromosome for extracting `<num>` is preserved even if the chromosome for `<pin>` has changed by genetic operators. This effect would yield the good result. In the conventional GE, more than 100 individuals were invalid ones at initial population, and more than 20 individuals exist also in the latter generations. On the other hand, in the proposed method and the GE with new wrapping technique, invalid individuals were scarcely generated from the initial generation.

V. Conclusions

In this paper, to preserve the characteristics of the parental individuals, we proposed the multi-chromosomal GE where each chromosome is exclusively for the corresponding nonterminal. Moreover, we proposed new wrapping technique where the rule set is changed so that rules increasing the number of nonterminals can not be used. The proposed method was compared with the conventional GE by the experiments on Symbolic regression, 3 Multiplexer and Mastermind. In the Symbolic regression and Mastermind problems, the proposed method showed faster evolutional speed and higher success rate than the conventional method. Invalid individuals could scarcely generated by the new wrapping method. The GE in which only the new wrapping method was introduced also could prevent the generation of invalid individuals, but showed lower performance than the proposed method. This shows the effect by the use of multiple chromosomes.

Our experimental problems have relatively small number of nonterminals, at most four nonterminals. From now on, we have to apply our proposed method to the problems with many nonterminals and complex optimal solution in order to examine its generality. In addition, the crossover used in our proposed method is applied to only a couple of chromosomes, and other chromosomes do not change. This might cause the stagnation of evolution because the drastic change of genotypes does not happen in problems with many nonterminals. We have to consider the appropriate genetic operators for evolving multi-chromosomal individuals.

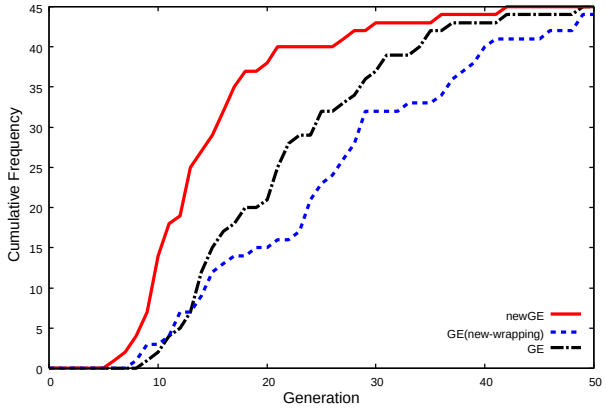

Fig. 12. Cumulative frequency of success measures on the mastermind.

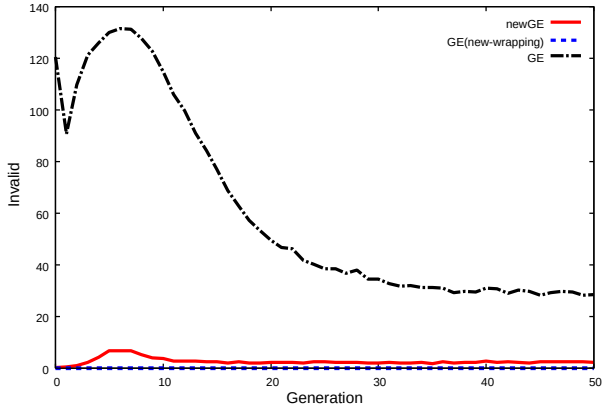

Fig. 13. The number of invalid individuals on the mastermind.

References

[1] John R. Koza, *Genetic Programming, On the Programming of Computers by means of Natural Selection*, MIT Press, 1992.
[2] John R. Koza, *Genetic Programming II, Automatic Discovery of Reusable Programs*, MIT Press, 1994.
[3] John R. Koza, *Genetic Programming III, Darwinian Invention and Problem Solving*, Morgan Kaufmann, 1999.
[4] Michael O'Neill and Conor Ryan, Grammatical Evolution, *IEEE Transactions on Evolutionary Computation*, Vol.5, No.4, pp.349-358, 2001.
[5] Michael O'Neill and Conor Ryan, *Grammatical Evolution*, Kluwer Academic Publishers, 2003.
[6] Peter Naur, Revised report on the algorithmic language ALGOL 60, *Communications of the ACM*, Vol.6, No.1, pp.1-17, 1963.
[7] John H. Holland, *Adaptation in Natural and Artificial Systems*, The Univ. Michigan Press, 1975.
[8] David E. Goldberg, *Genetic Algorithms in search, optimization, and machine learning*, Addison Wesley, 1989.
[9] Gerald D. Elseth and Kandy D. Baumgardner, *Principles of Modern Genetics*, West Publishing Company, 1995.
[10] Michael O'Neill and Conor Ryan, Genetic Code Degeneracy: Implementations for Grammatical Evolution and Beyond, *ECAL'99: Proceedings of the Fifth European Conference on Artificial Life*, pp.149-153, 1999.
[11] Michael O'Neill, Anthony Brabazon, Miguel Nicolau, Sean Mc Garraghy and Peter Keenan, πGrammatical Evolution, *GECOO 2004: Proceedings of the Genetic and Evolutionary Computation Conference*, pp.617-629, 2004.
[12] Conor Ryan, Miguel Nicolau and Michael O'Neill, Genetic Algorithms Using Grammatical Evolution, *EuroGP 2002: Proceedings of the Fifth European Conference on Genetic Programming*, pp.278-287, 2002.
[13] Michael O'Neill and Anthony Brabazon, Grammatical Swarm, *GECCO 2004: Proceedings of the Genetic and Evolutionary Computation Conference*, pp.163-174, 2004.