A Linear Genetic Programming with Reusable Gene

Zongyue Wang [1,2], Hongchao Ma [2]
[1]School of Computer Engineering College,
Jimei University, Xiamen, Fujian 361021 P.R.China
[2] School of Remote Sensing & Information Engineering,
Wuhan University, Wuhan, Hubei 430079 P.R.China
wangzongyue1979@163.com, hchma@whu.edu.cn

Abstract

In this paper, a novel genetic programming named linear genetic programming with reusable gene (LGPRG) has been proposed. This new method absorbed the merits of many other linear genetic programming. It codes with a simple, nearly unrestrained string. Based on its character of reused, more expressions could be contained in one chromosome without the increase of computation task. Further more, the expression segments in a chromosome are always integrated. This method is proved to be effective and stable from the complexity analysis and experiment.

1. Introduction

After the birth of human beings, modeling has become an indispensable tools for people to explore the world. However, it is still too hard to handle modeling for complex problem. In 1990, J.Koza proposed genetic programming (GP) [1,2,3], a significant branch of evolutionary computation, inspired by biological evolution to find computer programs that perform a user-defined task. Based on genetic programming, evolutionary modeling could easily evolve an expectant expression model by tree structure chromosome. Many applications have already proved its validity. But some times, tree structure is hard to control the illegal expressions which can be produced in any step of evolution.

After 2000, many new genetic programming methods have been proposed especially for various linear genetic programming [4,5]. Ferreira proposed a linear gene expression programming (GEP) [6,7,8] in 2001. She took string structure chromosome instead of former tree structure to store expression. Although there are still some restrains for the string's validity, it makes the expression simpler and its efficiency has also improved in some levels. In the year of 2003, M.Oltean proposed another linear genetic programming called multi expression programming (MEP) [9~13]. He introduced the conception of reused gene into chromosome, so more expressions can be contained in one chromosome without the increase of the computation task. Though more expressions are included in one chromosome, many of the expressions maybe short. In another words, longer expression are not easy to appear in one chromosome. Because the addresses of operation are hard to stay in a stage that makes the expression longer. Recently, a Chinese researcher Peng proposed multi-gene evolutionary algorithm based on overlapped expression (MEOE) [14]. Illegal chromosome disappeared in this linear genetic programming and it has reused character in some degree.

Efficiency and results of genetic programming can be affected by many factors such as coding structure, genetic operation, parameter setting and so on. This paper mainly discussed the coding factor for genetic programming. A new algorithm absorbed the merits of other genetic programmings are described. The new method presents a simple coding pattern and strong reused character. Further more, it is good at present long expression.

In part 2, the new algorithm LGPRG is described in details from coding method, genetic operation, selection strategy and so on.. In part 3, experiment is given firstly. The following is quality and quantity analyzing for LGPRG and other algorithm. In part 4, conclusion and future expectation are described.

2. Linear Genetic Programming with Reusable Gene

2.1. Code Designing

Efficiency and stability of genetic algorithm can be affected by the structures and decoding manner. Like the structure of GEP and MEOE, LGPRG also takes linear string to encode the expression, but the distinct decoding way makes algorithm different from others.

Definition of the LGPRG chromosome coding can be described as follows: genes of chromosome are constituted by constant, variable and operator; assuming that the highest argument of the operators is n, the last genes in chromosome of size n should be variable or

978-0-7695-3505-0/08 $25.00 © 2008 IEEE
DOI 10.1109/IITA.Workshops.2008.184

constant. They should never change into operator during the evolution process; each gene stands for an expression in chromosome. Fig.1shows a example chromosome.

*+a/-b+/ab

Figure 1 A chromosome

Define the chromosome decoding as following: decode the chromosome form the bottom to the top; if the gene is constant or variable, decode the gene as the same. If the gene is operator, its operation numbers are the former decoding genes next to its position. Fig.2 is the expressions according to each gene in Fig.1. Genes are in the left column, expressions are in the right column.

chromosome	expression
*	(a+(b-(a/b+a)}/b)*a
+	a+(b-(a/b+a)}/b
a	a
/	(b-(a/b+a)}/b
-	b-(a/b+a)
b	b
+	a/b+a
/	a/b
a	a
b	b

Figure 2 Genes in chromosome and their expressions

When coding the chromosome, the last n genes should be terminals such as constants or variables. It is sufficient but not necessary. The main purpose for this restrain is to keep the validity for chromosome. For instance, if the last gene is "+", it will have no operation number for its last position. There is no other restrict for this coding method.

While decoding the chromosome, direction should be confirmed. We set the direction from the bottom to the top. For the last gene "b" in fig.2, its expression is "b". For the upper gene "a" in fig.2, its expression is "a". For the last third gene "/" in fig.2, its operation number is "a" and "b" just mentioned, which is next to it. Its expression is "a/b". Further more, the last fourth gene "+" stands an expression "a/b+b". Its operation numbers are last third gene "/"and last second gene "a". Because the expression of gene "/" is "a/b", the expression for gene "+" is "a/b+b". The other gene's expressions are also translated by this way. It can be proved that all functions can be expressed by this way.

2.2. Recombination Designing

New algorithm's recombination designing is similar to genetic algorithm. There are many types of recombination such as one point recombination, two point recombination and so on.

2.3. Mutation Designing

There are many types of mutation such as one point mutation, two point mutation and so on. Assuming that the highest argument of the operators is n, to keep chromosome legal, the last n genes should not be changed.

2.4. Transposition Designing

Transposition means a part of the chromosome moved to another part of the same chromosome. There are many kinds of transposition operation for linear genetic programming with reusable gene. We can take reference from gene expression programming.

2.5. Algorithm Flow

LGPRG algorithm's flow can be described in fig.3 like the traditional genetic algorithm.

```
ALOGRITHM
begin
    t:=0;
    initialize P(t);
    P(t) = {x1(t), x2(t),…, xn(t)}
    evaluate(t);
    F (P(t)) = {F (x1(t)), F (x2(t)),…, F (xn(t))}
    while (not termination condition) do
        Pr(t) = recombination {P(t)};
        Pm(t)=mutation {Pr(t)};
        Pn(t)=transposition {Pm(t)};
        evaluate [Pn(t)];
        P(t+1) = select [Pn(t)];
        t:=t+1
    print xbest, F (xbest);
end
```

Figure 3 Algorithm of LGPRG

3. Experiment

3.1. Success Rate Comparing Experiment o f Simple Function Finding

Taking $y = x^4 + x^3 + x^2 + x^1 + x$ as the experimental function. We give a comparison experiment between several linear genetic programming – GEP, MEP, MEOE and LGPRG.

Data sets are in table 1, Considering that different algorithms need different parameter setting, we give the setting as table 2 to assure they can perform the best in each running.

Table 1 Data sets

x	y
2.81	95.2425
6	1554
7.043	2866.55
8	4680

10	11110
11.38	18386
12	22620
14	41370
15	54240
20	168420

Table 2 Parameter setting

Parameter name	GEP	MEP	MEOE	LGPRG
Operator set	+, -, *, /	+, -, *, /	+, -, *, /	+, -, *, /
Terminal Set	a	a	a	a
Length of chromosome	20	20	20	20
Population size	50	50	50	50
Recombination probability	0.66	0.70	0.70	0.70
Mutation probability	0.044	0.10	0.10	0.10
Transposition probability	0.10	0.10	0.10	0.10
Generation for each run	50	50	50	50
Runtimes	100,000	100,000	100,000	100,000

Table 3 is the success rate of each algorithm. From the table, linear genetic programming with reusable gene performs well in stability.

Table 3 Success Rate

Algorithm Name	Success Rate
GEP	96.9%
MEP	97.3%
MEOE	95.4%
LGPRG	97.0%

3.2. Searching Efficiency Comparing Experiment o f Complex Function Finding

The test function is $y = \log(\tan(x_0 x_1 + x_0)) + x_0$. We also give a comparison experiment between several linear genetic programmings as GEP, MEP, MEOE and LGPRG. The parameter setting is in table 4. The dataset is randomly selected in condition $x_0, x_1 \in [0,30]$. The number of sample data is 10.

Table 4 Parameter setting

Parameter name	GEP	MEP	MEOE	LGPRG
Operator set	+, -, *, /,tan,log	+, -, *, /,tan,log	+, -, *, /,tan,log	+, -, *, /,tan,log
Terminal Set	x0, x1	x0, x1	x0, x1	x0, x1
Length of chromosome	20	20	20	20
Population size	50	50	50	50
Recombination probability	0.66	0.70	0.70	0.70
Mutation probability	0.044	0.10	0.10	0.10
Transposition probability	0.10	0.10	0.10	0.10
Generation	1000	1000	1000	1000

Fig.4 is the absolute error of each algorithm in different generation. From the figure, we can see that LGPRG is effective. Considering the effect of probability, we do several repeated experiment of above. The performance is nearly same. LGPRP still performs best in the above algorithms.

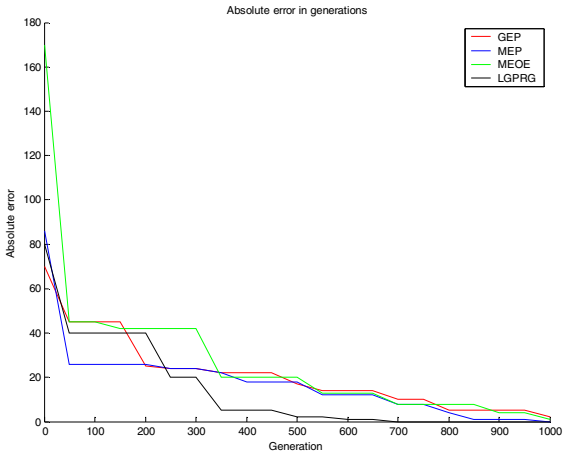

Figure 4 Absolute error in each generation

3.3. Quality Analyzing for LGPRG and Other Linear Genetic Programming

The obvious merit of genetic programming is the clear structure of the expression which has been presented. One expression could be presented to a multi layer tree. But genetic programming of tree structure is not convenient to store. During the process of evolution, illegal tree may always occur in initial step, recombination step and mutation step.

Gene expression programming, which is a classic linear genetic programming, performs better than tree based genetic programming. Comparing GEP to GP, GEP take a simple coding method, which just use a simple string. GEP's efficiency is usually higher than GP's. But GEP still has some unfriendly character. First, there is a strong restrict for legal chromosome that chromosome should be divided into two part and operators can not exist in the tail part. Second, we can not get the fitness directly form the tree, it is necessary to change the tree to other structure. Third, genes in chromosome have not been reused in chromosome. If the first gene in GEP is a terminal, the expression of the chromosome is the terminal. The other genes in chromosome will have no

sense. In fact, the other genes still could be used to extend the presentation ability of chromosome.

Multi expression programming, a highly reusable algorithm, could contain more expressions in a chromosome, but its computation task will not increase. Further more, it is unnecessary to change its string structure to other when computing the fitness. However, the expressions in multi expression programming are easy to be short in chromosome especially during the initial period.

MEOE, a novel simple linear genetic programming, has reused character in some degree. A gene in MEOE could reuse its neighborhoods.

LGPRG absorbs the merits of other linear genetic programmings while abandoning the shortcomings. It codes simply nearly without restrict. It takes has highly reused character than GEP and MEOE. Comparing to MEP, it could contain more completely expression. But a chromosome in MEOE could only stand for an expression, some of gene nodes may be wasted for this restriction.

3.4. Quantity Analyzing for New Algorithm and Other Linear Genetic Programming

Considering from the expressing ability, we give the comparing result in table 5. Assuming there are n genes in a chromosome.

Table 5 Ability of Expression for Chromosome

Algorithm Name	Number of expressions	Number of fitness computation
GEP	1	1
MEP	n	1
MEOE	1	1
LGPRG	n	1

An expression tree of genetic programming can be seen as a chromosome here. Genetic programming contains one expression in a chromosome. Gene expression programm-ing contains one expression in a chromosome. Multi ex-pression contains n expression in a chromosome. Multi-gene evolutionary algorithm contains one expression in a chromosome. The new algorithm in this paper contains n expression in a chromosome.

4. Conclusion

This paper proposed a novel reused linear genetic programming – LGPRU, which absorbed merits of other algorithm. Experiment and complex indicate that the new algorithm is stable and effective. However, genetic programming is not be influenced just by coding method. Many other aspects, such as designing of genetic operation, parameter setting and so on, are all the influencing factors for genetic programming. The algorithm will perform best just when all aspects cooperate well.

Acknowledgement

This project is supported by Ministry of Education of the People's Republic of China (Project No.A1420060213).

References

[1] J.Koza, Genetic programming Ⅰ,The MIT Press,1992.
[2] J.Koza, Genetic programming Ⅱ,The MIT Press,1994.
[3] J.Koza, Genetic programmingⅢ,The MIT Press,1999.
[4] Ryan C,Collins JJ, O'Neill M (1998). "Grammatical evolution: Evolving programs for an arbitrary language". Proceedings of the First European Work- Shop on Genetic Programming. Springer-Verlag, pp.83-95.
[5] Brameier M, Banzhaf W (2001). "A comparison of linear genetic programming and neural networks in medical data mining". IEEE Transactions on Evolutionary Computation. No. 5, pp.17-26.
[6] Ferreira C. (2001). "Gene expression programming: a new adaptive algorithm for solving problems". Complex Systems. Vol . 13, No.2, pp.87-129.
[7] Ferreira C. (2002). "Gene expression programming: Mathematical Modeling by an Artificial Intelligence". Angra do Heroismo, Portugal.
[8] Ferreira C.(2006). "Gene expression programming: Mathematical modeling by an Artificial Intelligence". 2nd Edition, Springer-Verlag, Germany.
[9] Oltean Mihai, D. Dumitrescu (2002). "Multi expression programming". technical-report.
[10] Oltean Mihai, Grosan C.(2003). "A Comparison of Several Linear Genetic Programming Techniques". Complex-Systems. Vol. 14, No. 4, pp. 285-313.
[11] Mihai Oltean, Crina Grosan, Mihaela Oltean (2004). "Encoding Multiple Solutions in a Linear Genetic Programming Chromosome". In Marian Bubak and Geert Dick van Albada and Peter M. A. Sloot and Jack Dongarra editors, Computational Science - ICCS 2004: 4th International Conference, Part III, volume 3038, pages 1281-1288, Krakow, Poland, 2004.
[12] Oltean Mihai, Grosan C. (2003). "Evolving Evolutionary Algorithms using Multi Expression Programming". The 7th European Conference on Artificial Life, September 14-17, 2003, Dortmund, Edited by W. Banzhaf (et al), LNAI 2801, pp. 651-658, Springer-Verlag, Berlin, 2003.
[13] Grosan C, Abraham (2005). "Multi-expression programming for intrusion detection system". International Work-conference on the Interplay between Natural and Artificial Computation. Span: J. Mira and J.R. Alvarez (Eds.),pp.163-172.
[14] Jing Peng, Changjie Tang, Changan Yuan (2007). "Multi-gene evolutionary algorithm based on overlapped expression". Journal of computer (in Chinese).Vol 25 No 12,pp.775-785