International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys.14, No.1 (Oct-23)

http://dx.doi.org/10.12785/ijcds/140198

Adaptive Length Gene Expression Programming

Samsul Amar[1,2], Andi Sudiarso[2] and Muhammad Kusumawan Herliansyah[2]

[1]*Department of Industrial and Mechanical Engineering, Faculty of Engineering, Universitas Trunojoyo Madura, Indonesia*
[2]*Department of Mechanical and industrial Engineering, Faculty of Engineering, Universitas Gadjah Mada, Yogyakarta, Indonesia*

Received 13 Dec. 2022, Revised 8 Aug. 2023, Accepted 21 Sep. 2023, Published 1 Oct. 2023

Abstract: Gene expression programming (GEP) is capable of solving many prediction, classification, and optimization problem effectively. It uses a fixed-length chromosome representing a set of equations. However, the chromosome length significantly affects the algorithm's performance. Different problems may require varied chromosome lengths to achieve good results. Only a few studies have been conducted to deal with chromosome length in GEP. Therefore, this study aimed to develop an adaptive GEP to find proper chromosome length during the evolutionary process. The study proposed that the chromosome length may be varied for each individual instead of using the same length in the population. The evolutionary process would adjust the chromosome length and the chromosome with proper length will tend to survive. Furthermore, the study proposed a contraction operator that could delete or insert an allele in the chromosome to make it short or extended. This operator is expected to adjust the chromosome length to its optimal. A special slice crossover was also proposed to accommodate the crossover between parents with different chromosome lengths. The proposed algorithms' performance was investigated by solving three symbolic regression problems. Additionally, the performance was compared to related previous gene expression programming algorithms.

Keywords: adaptive length, gene expression programming, symbolic regression problem

1. INTRODUCTION

Popular machine learning methods have been applied for prediction and classification, including artificial neural networks, k-nearest neighbors, decision trees, and adaboost. Although these methods could produce good prediction accuracy, they create a black box problem [1] [2]. The methods could probably provide a better prediction or classification than humans, but they cannot communicate or explain the reason underlying that decision. This could make the decision maker reject applying machine learning or cause tracing difficulties. Also, it could lead to law problems regarding the decision guided by machine learning methods.

The black box problem could be avoided using an algorithm called gene expression programming (GEP). This algorithm has been applied successfully to solve many real problems effectively [3] [4]. GEP is a search algorithm inspired by the natural or biological mechanism of adaptation and survival. Other popular machine learning methods encounter black box problems. In contrast, GEP produces explicit mathematical equations or model explanations. This enables users to get the answer from a black box and know how the variables are interconnected.

GEP adopts gene expression in an organism and improves genetic algorithm and genetic programming. It uses and denotes a fixed-length chromosome into an expression tree representing a set of equations or a specific model. A chromosome consists of one or more genes. Fix length string chromosome is easy to manipulate using various evolutionary operators. The GEP procedure is similar to the genetic algorithm processes, including initialization, fitness evaluation, selection and replication, mutation, and crossover. The chromosome design makes the decoding process always produce valid equations. This is one advantage of GEP compared with genetic programming, where an invalid function is probably produced in a generic operation. Also, the GEP's expression tree representation is better than the genetic algorithm in solving complex models.

GEP was developed by Fereira in 1999 and has been implemented or improved by many studies [4]. For instance, Mwaura and Keedwell [5] developed a heuristic method to adjust the mutation and crossover rate during the evolutionary process to avoid an unfit rate-setting problem. Some studies combine GEP with other metaheuristic methods to increase the algorithm's performance. Zhang et al. [6] and Zuo et al. [7] combined GEP with differential evolution and found that their proposed algorithm performed better than the original GEP. Yang and Ma [8] used an orthogonal design for a multiple-parent crossover operator in chromosome reproduction. The study also introduced an evolutionary stable strategy to maintain population diver-

E-mail address: samsul.amar@trunojoyo.ac.id, a.sudiarso@ugm.ac.id, herliansyah@ugm.ac.id

sity during evolution. Furthermore, Fajfar and Tuma [9] developed a special numeric crossover operator to improve the constant creation ability of RGEP. Mehr [10] combined GEP with GA for streamflow forecasting. The study refined the best individual from the GEP algorithm by GA to get better fitness. Similarly, Deng et al. [11] combined GEP with an artificial fish swarm algorithm and found that the combination performed better than the original GEP. Yang et al. [12] developed a hybrid Kalman filter GEP to reconstruct drawing and handwriting.

GEP, with its developments, are capable of solving various problems. However, some problems still need to be undertaken [4], such as the encoding design. The basic encoding of GEP adopts K-expression to represent the computer program in a fixed-length string. One of the K-expression disadvantages is that a good building block could be easily destroyed in the genetic operation. Efforts have been made to overcome this problem, including prefix GEP (P-GEP) [13], robust GEP (RGEP) [14], and self-learning GEP (SL-GEP) [15]. P-GEP applied depth-first decoding instead of width-first decoding as in the original K-expression. Experiments showed that this decoding method could protect the good structure more than the original GEP [13]. RGEP implements the P-GEP with certain development to simplify the gene structure and avoid invalid syntax. SL-GEP proposes a representation of sub-function as more efficient for complex problems.

Another problem of GEP is chromosome length. According to Ferreira [3], the algorithm performs better when the chromosome length expands to a certain threshold. The performance then decreases significantly because of the large solution space to be investigated. Therefore, Bautu et al. [16] developed AdaGEP that could adaptively adjust the active genes in the chromosome through genemap, a binary string representing gene activation. The gene is activated when the value of a correspondent gene in the genemap is 1. Otherwise, the gene is deactivated or ignored in the decoding process. The study showed that AdaGEP could perform better than the original GEP even when the number of genes surpasses the threshold. Using AdaGEP, the performance decrease after the chromosome length threshold is not as sharp as the original GEP. However, the user must still decide the number of genes. The performance is also influenced by the number and length of genes.

There is a need for a better adaptive chromosome length GEP that could produce a good performance. This study aimed to contribute to reducing the chromosome length problem by proposing an adaptive length gene expression programming (ALGEP). It adopted RGEP encoding to enhance its performance with some developments. First, the study proposed varying the chromosome length for each individual instead of using the same length in the population. The chromosome length is randomly chosen from a specified range when generating the initial population. The evolutionary process adjusts the length to ensure that the chromosome with the proper length survives. Second, the study proposed a contraction operator that could delete or insert an allele in the chromosome to make it short or extended. This operator is expected to adjust the chromosome length to its optimal. A special slice crossover was also proposed to accommodate the crossover between parents with different gene lengths.

The performance was investigated by applying the proposed algorithm to solve three non-linear symbolic regression problems (SRP). The result was compared with the standard GEP and RGEP, the previous related algorithms.

2. ALGORITHM

This section discusses the basic GEP, P-GEP, RGEP, and the proposed ALGEP.

A. GEP

GEP was invented by Ferreira in 1999 [3] and used a fixed-length string to represent and decode the gene to an expression tree to produce a mathematical equation. The solution is expressed using the Karva language by adopting a width-first scheme to translate a gene into an expression tree. In GEP, a gene contains a head and a tail section. The head could contain function and terminal string, while the tail has only the terminal. Function and terminal strings are defined by the user. Examples of function strings are *, /, +, -, sqrt, sin, and cos, while terminal strings could include constants and variables. The length of a tail depends on the length of the head, calculated using (1).

$$t = h(n_{\max} - 1) + 1 \tag{1}$$

Where t is the length of the tail, h is the length of the head, and $n_{\max}$ is the maximum number of arguments in the functions. For instance, sqrt (square root) has one input argument while * (multiplying operator) has two input arguments. Using this scheme, all generated genes and the result of all generic operators are translated into valid functions. This is one advantage of GEP, where there is no need to iteratively find a valid syntax after genetic manipulation of a gene, as in genetic programming. Fig. 1 shows the gene decoding process, where a fixed-length string containing a head and tail is decoded into an expression tree and translated into a function. In the example, the head length is six, and the maximum number of arguments in the function is two. Therefore, the tail length (t) is:

$$t = 6(2 - 1) + 1 = 7$$

Fig. 2 shows the typical GEP process, where the initial population is created randomly. The number of chromosomes in the population, called population size, is predefined. Each chromosome is then expressed or translated into a mathematical function or computer program. The program is executed to obtain the fitness function of each chromosome.

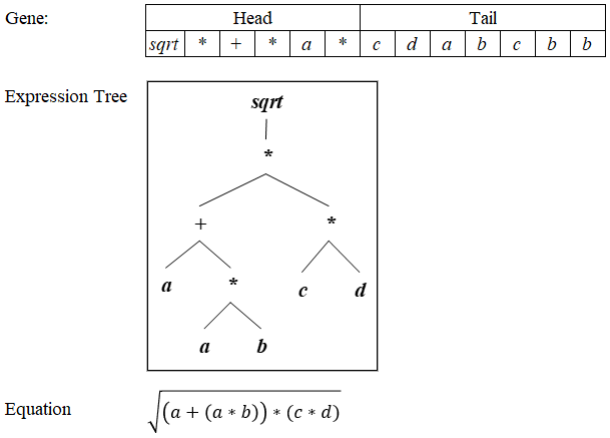

Figure 1. Example of a GEP decoding

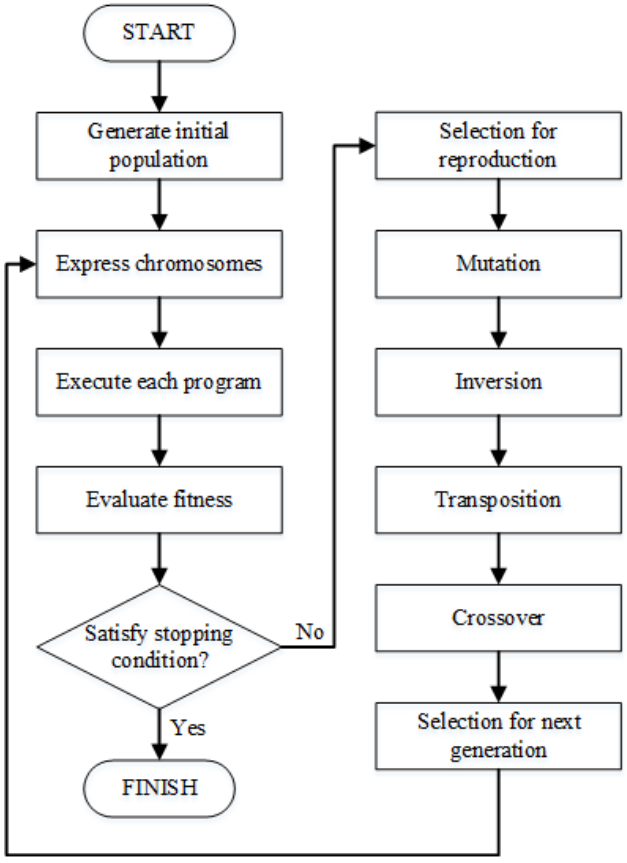

Figure 2. Typical GEP process

The selection scheme is similar to genetic algorithm, where the roulette wheel is the commonly used operator for selection. A mutation is the most effective evolutionary operator [3], though it could be constructive or destructive. The mutation rate is usually low because a rate close to 1 would make the algorithm similar to a random search. Ferreira proposed a guide to determine the mutation rate (pm) equal to 2/ (chromosome length). For instance, the rate is 0.2 when the chromosome length is 10. Mutation could occur in both the head and tail and the process replaces an allele with another allele. When the mutation occurs in the head, a function or terminal is replaced by another function or terminal. In contrast, the replacement is only for the terminal when the mutation occurs in the tail. Fig. 3 a) and b) show a mutation in the head and tail sections, respectively.

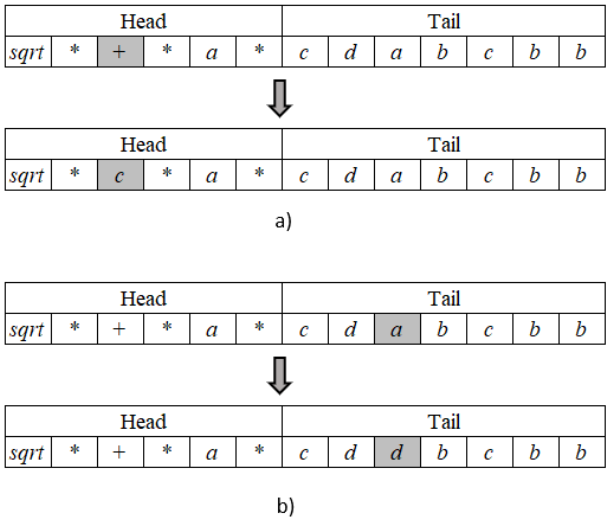

Figure 3. Example of mutation: a) in the head. b) In the tail

Inversion is the process of inverting a sequence part of a gene at a rate usually less than 0.1. Fig. 4 shows an example of an inversion process that drastically changes fitness. Transposition is copying and inserting a chromosome fragment into another position, as shown in Fig 5.

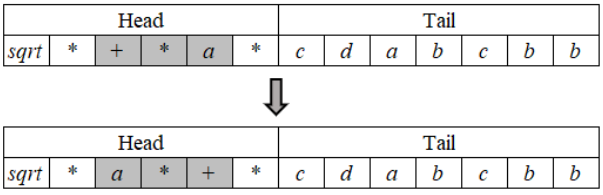

Figure 4. Example of an inversion

GEP crossover is performed similarly to crossover in genetic algorithm. One-point crossover slices two parents in a point and exchanges each section to produce two children. The two-point crossover is the same process but with two slice points. Fig. 6 and 7 show examples of one-point and two-point crossover.

The process is iterated to satisfy one of the termination

conditions, including reaching a predefined maximum generation, passing the maximum time, no change in the best fitness for a specific number of generations, and reaching an optimal solution (if known).

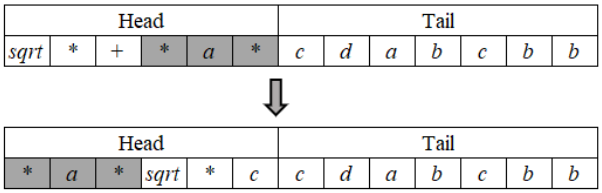

Figure 5. Example of a transposition

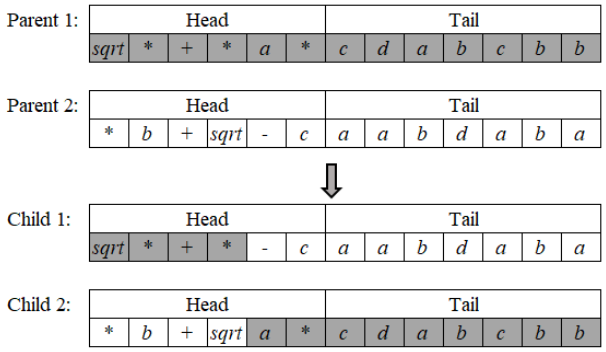

Figure 6. Example of a one-point crossover

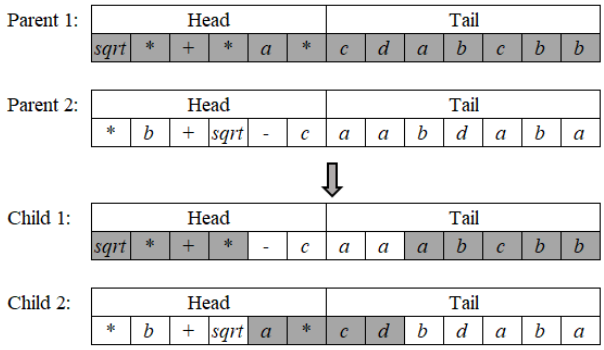

Figure 7. Example of a two-point crossover

B. P-GEP

P-GEP adopts a linear genotype representation in prefix notation and a different mapping mechanism between its genotype and phenotype. It applies depth-first decoding instead of width-first decoding as in the original K-expression. The experiments showed that this decoding method could protect the good structure more than the original GEP [13]. Fig. 8 shows an example of P-GEP decoding.

C. RGEP

RGEP implements the P-GEP encoding scheme with some development to simplify the gene structure and avoid invalid syntax. It does not use separate head and tail sections in a gene and does not re-create and reject invalid individuals. The method ignores functions with no sufficient functions or terminal strings. Fig. 9 shows an example of RGEP decoding. In the example, the grey strings "*" and "-" have insufficient input arguments and are ignored in the decoding process.

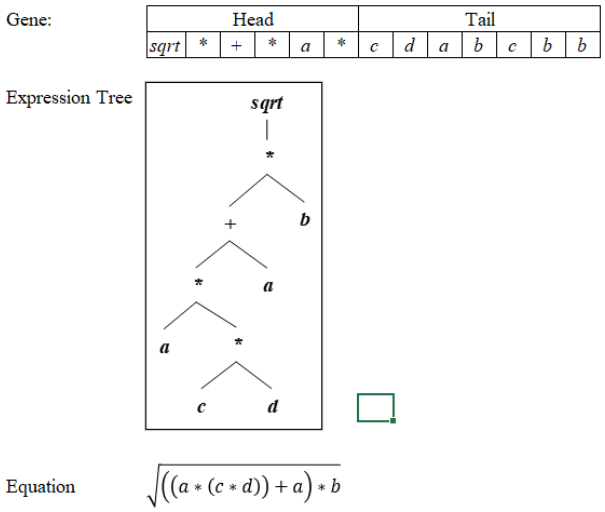

Figure 8. Example of a P-GEP encoding

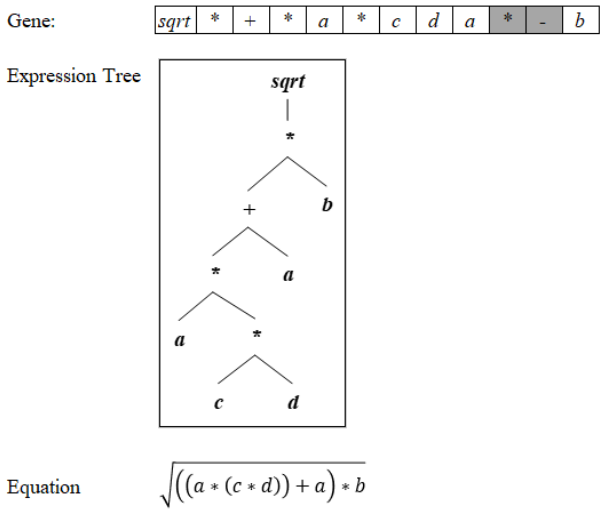

Figure 9. Example of a RGEP encoding

D. The Proposed ALGEP

ALGEP is a development of RGEP with an adaptive chromosome length and some revised evolutionary operators. The chromosome length is not fixed and could

be varied for each individual in the population. When generating the initial population, the length of a gene is randomly chosen from a specified range. The evolutionary process adjusts the gene length to ensure that the individual with the proper length survives. Fig. 10 shows an example of a population in ALGEP. Each individual may have a different gene length in the specified range. Users could also choose whether the first allele is always a function or random randomly. When the first allele is chosen as a function, it is replaced by a random function in the initial population or along the evolutionary process. For instance, individual 2 in Fig. 10 may be changed to "+ a * b +". The function "+" in the first allele is chosen randomly from the function set.

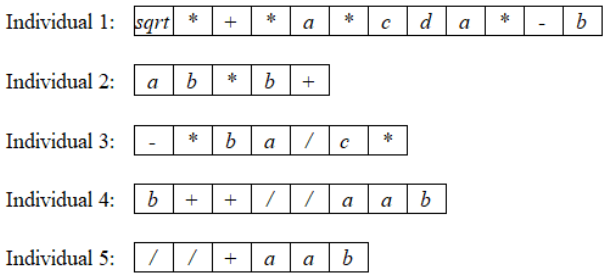

Figure 10. Example of a population in ALGEP

E. The Proposed Contraction Operator

The contraction operator could delete a bit of the chromosome to shorten it or insert a bit to extend it with a specified probability, pcon. A random number is generated in the contraction process. The contraction is conducted when the random number is less than or equal to pcon. Moreover, a simple rule is used to decide whether the individual is to be contracted or expanded. When the random number generated is less than *pcon/2*, a bit in the random position is deleted, and the gene length is shortened. Otherwise, a bit of a random string is inserted in a random position. This operation is expected to result in a proper gene length in the evolutionary process. Fig. 11 shows an example of the contraction operation of an individual. In the example, a bit in position 5 is deleted, shortening the chromosome length from 10 to 9 bits. A random bit could also be inserted in position 5 to extend the gene length to 11 bits.

F. The Proposed Slice Crossover

The chromosome length in ALGEP may be varied for each individual, though this makes the crossover process ineffective. Fig. 12 shows a two-point crossover for two individuals with gene lengths of 12 and 7 at positions 3 and 6, respectively. Using two-point crossover, the position from bit 8 to 12 in parent 1 cannot become the subject of crossover. This condition could reduce the probability of producing a better child since it might be a good gene in that position.

Slice crossover was proposed to overcome this problem, where the crossover position in parents 1 and 2 could be different and chosen randomly. For instance, Fig. 13 shows that crossover points in parents 1 and 2 are at positions 7 to 8 and 3 to 4, respectively. Therefore, all positions of the parents could become the subjects of crossover and increase the probability of producing good children.

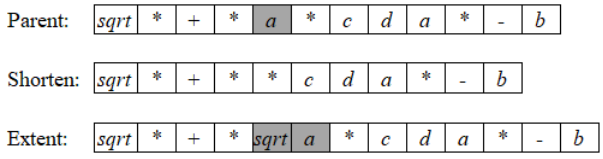

Figure 11. Example of a contraction

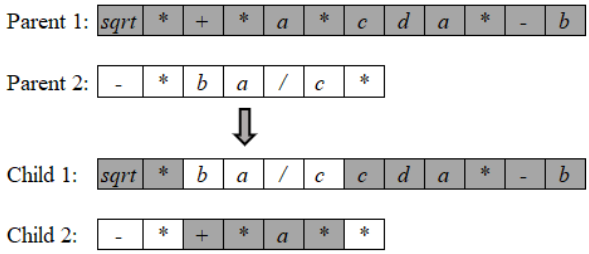

Figure 12. Example of a two-point crossover in ALGEP

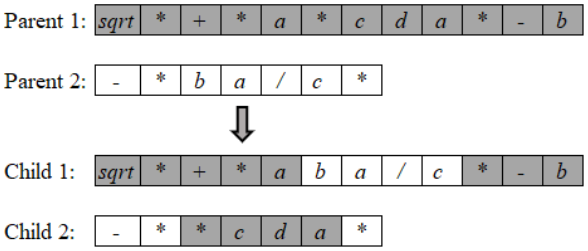

Figure 13. Example of a slice crossover

3. EXPERIMENT

The performance of ALGEP, contraction operator and slice crossover were analyzed by applying the algorithms to solve 3 SRPs and comparing the result with basic GEP and RGEP. The first SRP (2) is a simple polynomial function with one variable taken from [17]. The second SRP (3) is a sequence induction problem from [18] and is relatively difficult to solve. The third SRP (4) is the most difficult, with three variables and a complex structure modified from [19]. It was modified to avoid using a constant in the GEP. This is because the experiment aimed to evaluate the basic chromosome structure and avoid other factors, including constants.

$$f(x) = x^4 + x^3 + x^2 + x \quad (2)$$

$$f(x) = 4x^4 + 3x^3 + 2x^2 + x \quad (3)$$

$$f(x) = \frac{(x_1 x_2)}{((x_1 - 1)x_3^2)} \quad (4)$$

For all functions, 20 random values were generated as the training data set without noise. The x values of the first (2) and second (3) functions were generated from uniform random probability in the range $[-10, 10]$. For the third function (4), uniform random values were generated in the range $[-1, 1]$ for variables x_1 and x_2, whereas x_3 was obtained from the range $[1, 2]$.

The performance of each proposed algorithm was investigated by setting the six method treatments, including 1) GEP, 2) RGEP, 3) basic ALGEP without contraction operator and slice crossover, 4) ALGEP with contraction operator, 5) ALGEP with slice crossover, and 6) ALGEP with a combination of contraction operator and slice crossover.

Each treatment's effectiveness was tested using various chromosome lengths varied from 11 to 103 with a 4-step interval. This means that chromosome length for the first experiment was 11, then 15, until 103. Each treatment was run 100 times, and the success rate percentage was recorded. A run was successful when the best result had an error less than the specified precision. The program for the experiment was coded in Python.

Ineffective genes were avoided by making the first gene a function for all treatments. When the initial population generation or the evolutionary process produced a non-function in the first gene, that gene was changed to function randomly. The study also used a two-point crossover method, except for treatments 5) and 6), which used the proposed slice crossover. The probability of contraction was set to 0.2 for treatments 4) and 6). Table 1 shows the parameter setting of the experiment.

4. RESULT AND DISCUSSION

The success rate was recorded for each treatment. Fig. 14, 15 and 16 show the success rate comparison of GEP, RGEP and ALGEP for solving SRP 1, SRP 2 and SRP 3, respectively. Table II compares the success rate for each SRP. The results indicate that ALGEP performs better in solving all SRPs. RGEP performs better than basic GEP, supporting [14]. This conclusion is supported by paired t-test results, as in Table III, where the success rate of the ALGEP algorithm is significantly higher than the other algorithms. The success rate of SRP 1 is higher than the success rate of SRP2 and the success rate of SRP 2 is higher than the success rate of SRP 3. This result indicate that SRP 1 is the easiest problem of the other, while SRP 3 is the most difficult problem to be solved among the others. Moreover, Fig. 14 indicates that the success rate of ALGEP does not decline after some specific length, while the decline appears when using GEP and RGEP.

Table IV and Fig. 17 – 19 indicate that applying slice crossover makes the performance of ALGEP better than when using the two-point crossover. Moreover, Table V shows that slice crossover performs significantly better than two-point crossover for all SRPs. Therefore, the slice operator is effective in increasing the performance of ALGEP.

The proposed contraction operator does not perform as well as expected and does not significantly affect basic ALGEP. The combination of contraction operator and slice crossover also does not increase the performance of ALGEP compared to slice crossover alone. Table III shows that the contraction operator does not significantly affect ALGEP performance. It may have both destructive and constructive effect on the gene and consequently cannot improve the population's fitness.

Figure 14. The success rate of GEP, REG and ALGEP for solving SRP1

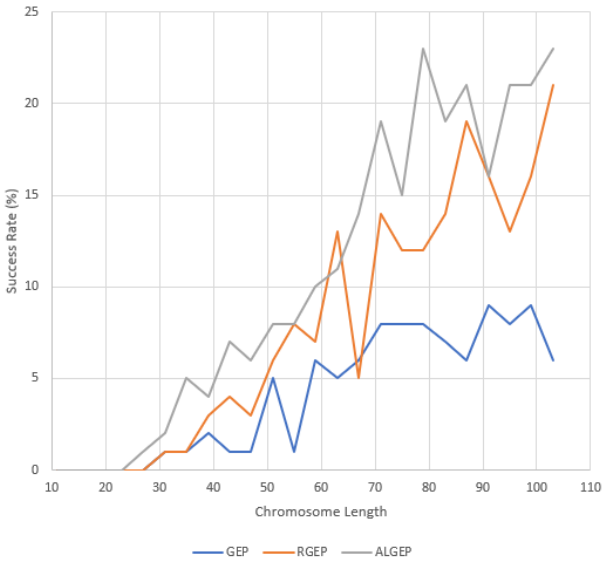

Figure 15. The success rate of GEP, REG and ALGEP for solving SRP2

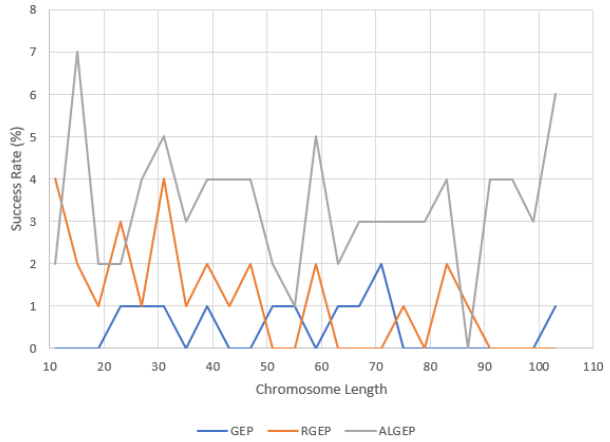

Figure 16. The success rate of GEP, REG and ALGEP for solving SRP3

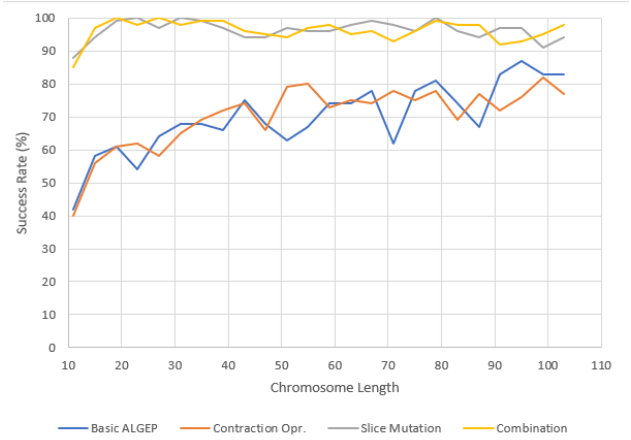

Figure 17. The success rate of contraction operator and slice crossover for solving SRP 1

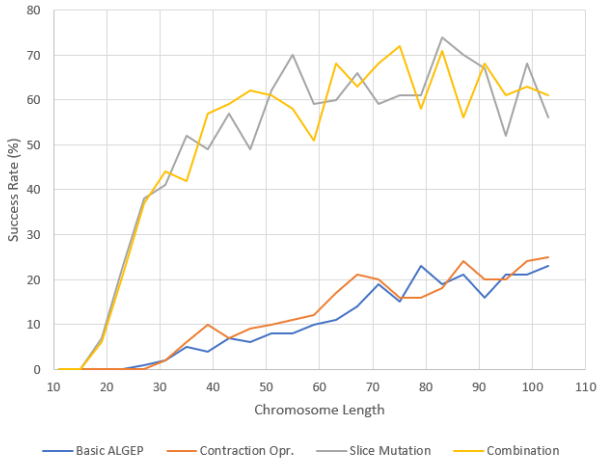

Figure 18. The success rate of contraction operator and slice crossover for solving SRP 2

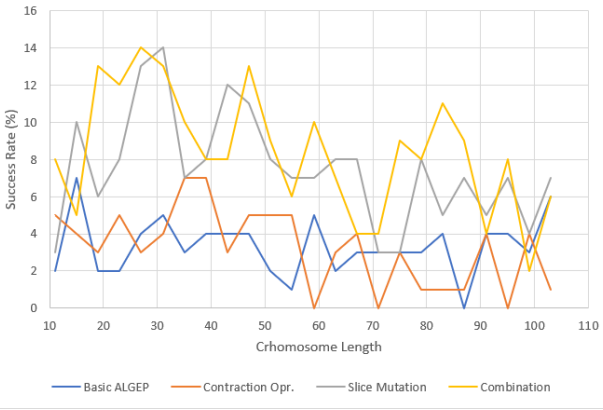

Figure 19. The success rate of contraction operator and slice crossover for solving SRP 3

5. CONCLUSSION

One of the problem in GEP is deciding proper chromosome length. This paper proposed an adaptive length GEP that can adaptively find a proper chromosome length during the evolutionary process. The chromosome length was generated randomly in the initial population and the chromosome with the proper length would tend to survive and produce offspring in the next generations. Success rate of the proposed ALGEP was compared with basic GEP and RGEP in solving three SRPs. The result demonstrated that ALGEP performs significantly better than the benchmarks for the all problems. For example, ALGEP produced 69.92% success rate in solving SRP 1 while RGEP and GEP produced 49.96% and 31.33% success rate respectively. The superior performance of ALGEP indicates that the population's adaptive length works well to find the proper chromosome length. This result means that ALGEP is promising in future studies and applications. With some developments and combined with other methods, ALGEP may become a good choice in prediction field.

This paper also proposed slice crossover to deal with different chromosome length during crossover process. This paper also proposed a contraction operator to make the proper length finding ran faster. The performance of slice crossover and contraction operator in ALGEP was investigated in solving three SRPs. As a result, the slice crossover application in ALGEP increased the performance of the algorithm significantly. For example, ALGEP using slice crossover could get 96.29% success rate in solving SRP1, significantly better compared with 69.92% success rate if using two-point crossover. Therefore, future studies could explore more and improve the slice crossover. However, the results showed that applying the contraction operator in ALGEP does not gives a significant effect. The combination of contraction operator and slice crossover also did not perform better than slice crossover solely.

References

[1] Y. Bathaee, "The artificial intelligence black box and the failure of intent and causation," *Harv. J. Law Technol.*, vol. 31, no. 2, p. 50, 2018.

[2] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM comput. surv.*, vol. 51, no. 5, pp. 1–42, Aug. 2018.

[3] C. Ferreira, *Gene expression programming: mathematical modeling by an artificial intelligence*. 2nd rev. and Extended ed. Berlin; New York: Springer-Verlag, 2006.

[4] J. Zhong, L. Feng, and Y.-S. Ong, "Gene expression programming: A survey [review article]," *IEEE Comput. Intell. Mag.*, vol. 12, no. 3, pp. 54–72, 2017.

[5] J. Mwaura and E. Keedwell, "Adaptive gene expression programming using a simple feedback heuristic," *Proceedings of the AISB conference*, p. 6, 2009.

[6] Y. Zhang and J. Xiao, "A new strategy for gene expression programming and its applications in function mining," *Univers. J. Comput. Sci. Eng. Technol.*, vol. 1, no. 2, p. 122, 2010.

[7] J. Zuo, C.-j. Tang, C. Li, C.-a. Yuan, and A.-l. Chen, "Time series prediction based on gene expression programming," in *Advances in Web-Age Information Management: 5th International Conference, WAIM 2004, Dalian, China, July 15-17, 2004 5*. Springer, 2004, pp. 55–64.

[8] J. Yang and J. Ma, "A hybrid gene expression programming algorithm based on orthogonal design," *Int. J. of Comput. Intell Sys*, vol. 9, no. 4, pp. 778–787, Jul. 2016.

[9] I. Fajfar and T. Tuma, "Creation of numerical constants in robust gene expression programming," *Entropy*, vol. 20, no. 10, p. 756, Oct. 2018.

[10] A. D. Mehr, "An improved gene expression programming model for streamflow forecasting in intermittent streams," *J. Hydrol.,*, vol. 563, pp. 669–678, Aug. 2018.

[11] S. Deng, C. Yuan, L. Yang, and L. Zhang, "Distributed electricity load forecasting model mining based on hybrid gene expression programming and cloud computing," *Pattern Recognit. Lett.*, vol. 109, pp. 72–80, Jul. 2018.

[12] Z. Yang, Y. Wen, and Y. Chen, "semg-based drawing trace reconstruction: A novel hybrid algorithm fusing gene expression programming into kalman filter," *Sensors*, vol. 18, no. 10, p. 3296, Sep. 2018.

[13] X. Li, C. Zhou, W. Xiao, and P. C. Nelson, "Prefix gene expression programming," in *Proc. Genetic and Evolutionary Computation Conference, (GECCO),05*, 2005, p. 7.

[14] N. Ryan and D. Hibler, "Robust gene expression programming," *Procedia comput. Sci.*, vol. 6, pp. 165–170, 2011.

[15] J. Zhong, Y.-S. Ong, and W. Cai, "Self-learning gene expression programming," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 65–80, Feb. 2016.

[16] E. Bautu, A. Bautu, and H. Luchian, "Adagep-an adaptive gene expression programming algorithm," in *Ninth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2007)*. IEEE, Sep. 2007, pp. 403–406.

[17] J. R. Koza, "Genetic programming: on the programming of computers by means of natural selection," in *Cambridge, Mass: MIT Press*, 1992.

[18] M. F. Korns, "Accuracy in symbolic regression," *Genetic Programming Theory and Practice IX, R. Riolo, E. Vladislavleva, and J. H. Moore, Eds. New York, NY: Spriger New York*, pp. 129–151, 2011.

[19] M. Keijzer, "Improving symbolic regression with interval arithmetic and linear scaling," in *Genetic Programming: 6th European Conference, EuroGP 2003 Essex, UK, April 14–16, 2003 Proceedings*. Springer, 2003, pp. 70–82.

Samsul Amar (Member, IEEE) receive bachelor's degree in industrial engineering from Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia, in 2002, and master degree in industrial engineering from Universitas Gadjah Mada (UGM), Yogyakarta, Indonesia, in 2012. His master thesis research was optimization for clinical rotation scheduling. He is currently a doctoral student at Mechanical and Industrial Engineering Department UGM with the research focus in artificial intelligent especially in developing adjusted length gene expression programming.

From 2006 until now, he is a lecturer in Universitas Trunojoyo Madura, East Java, Indonesia in Industrial and Mechanical Engineering department. His research interest are in field of modelling, optimization, metaheuristic, and artificial intelligence.

Andi Sudiarso (Member, IEEE) received the bachelor's and master degree (M.T.) in electrical engineering from Universitas Gadjah Mada (UGM), Indonesia, master degree (M.Sc.) in manufacturing engineering from UMIST, UK, and Ph.D. in Mechanical Engineering from The University of Manchester, UK.

He is currently a lecturer and researcher at the Department of Mechanical and Industrial Engineering, UGM. His research interest are manufacturing engineering, production system, and artificial intelligence.

Muhammad Kusumawan Herliansyah (Member, IEEE) receive bachelor's degree in mechanical engineering from Universitas Gadjah Mada (UGM), Yogyakarta, Indonesia in 1996 and master degree in manufacturing system from Institut Teknologi Bandung (ITB), Indonesia, in 2002. He receive doctoral double degree in advance material processing from University of Malaya, Malaysia dan Graduate School of Engineering Kyoto University, was born in Greenwich Village, Japan, in 2008. He is currently a lecturer and researcher in UGM at the Department of Mechanical and Industrial. His research interests are advanced material engineering, biomaterial, robotic and automation, and artificial intelligence.

Mr. Herliansyah receive certificate as a professional engineer from the Institution of Engineers Indonesia (PII) and ASEAN Engineer from ASEAN Federation of Engineering Organisations (AFEO). He receive award from

TABLE I. PARAMETERS SETTING

Parameter	Value
Population size	100
Max generation	100
Number of run	100
Length of chromosome	11-103
Function in all first gene	Yes
Probability of mutation	0.1
Probability of inversion	0.1
Probability of transposition	0.1
Probability of crossover	0.9
Parent selection method for crossover	Roulette Wheel
Selection method for next generation	Roulette Wheel without replacement
Best possible fitness	1
Accuracy performance measurement	Normalize Mean Square Error
Selection Range	1
Precision	0.0001
With constant	No
Random method for gene generation	Uniform
Mathematical operator	+ - * /
Problem to be solved	SRP 1, SRP 2, SRP 3

TABLE II. THE SUCCESS RATE OF GEP, REG AND ALGEP FOR SOLVING THE SRPS

Chrom Length	Success Rate (%)								
	SRP 1			SRP 2			SRP 3		
	GEP	RGEP	ALGEP	GEP	RGEP	ALGEP	GEP	RGEP	ALGEP
11	0	0	42	0	0	0	0	4	2
15	0	13	58	0	0	0	0	2	7
19	8	31	61	0	0	0	0	1	2
23	13	28	54	0	0	0	1	3	2
27	29	41	64	0	0	1	1	1	4
31	24	32	68	1	1	2	1	4	5
35	27	44	68	1	1	5	0	1	3
39	31	48	66	2	3	4	1	2	4
43	37	52	75	1	4	7	0	1	4
47	43	58	68	1	3	6	0	2	4
51	44	53	63	5	6	8	1	0	2
55	49	59	67	1	8	8	1	0	1
59	37	53	74	6	7	10	0	2	5
63	38	57	74	5	13	11	1	0	2
67	39	55	78	6	5	14	1	0	3
71	40	68	62	8	14	19	2	0	3
75	42	59	78	8	12	15	0	1	3
79	45	67	81	8	12	23	0	0	3
83	37	69	74	7	14	19	0	2	4
87	36	69	67	6	19	21	0	1	0
91	42	64	83	9	16	16	0	0	4
95	35	62	87	8	13	21	0	0	4
99	28	59	83	9	16	21	0	0	3
103	28	58	83	6	21	23	1	0	6
Average	31.33	49.96	69.92	4.08	7.83	10.58	0.46	1.13	3.33

TABLE III. PAIRED T-TEST FOR GEP, RGEP AND ALGEP

SRP	Success Rate Mean Comparison	p-value	Significance different at $\alpha = 0.05$
SRP 1	RGEP >GEP	1.65E-10	Significant
	ALGEP >GEP	1.08E-14	Significant
	ALGEP >RGEP	3.83E-08	Significant
SRP 2	RGEP >GEP	1.65E-10	Significant
	ALGEP >GEP	1.08E-14	Significant
	ALGEP >RGEP	3.83E-08	Significant
SRP 3	RGEP >GEP	1.65E-10	Significant
	ALGEP >GEP	1.08E-14	Significant
	ALGEP >RGEP	3.83E-08	Significant

TABLE IV. THE SUCCESS RATE OF CONTRACTION OPERATOR, SLICE CROSSOVER AND COMBINATION FOR SOLVING THE SRPS

Chrom Length	Success Rate (%)								
	SRP 1			SRP 2			SRP 3		
	Contr. Opr.	Slice Cross.	Comb.	Contr. Opr.	Slice Cross.	Comb.	Contr. Opr.	Slice Cross.	Comb.
11	40	88	85	0	0	0	5	3	8
	56	94	97	0	0	0	4	10	5
15	61	99	100	0	7	6	3	6	13
	62	100	98	0	23	21	5	8	12
	58	97	100	0	38	37	3	13	14
27	65	100	98	2	41	44	4	14	13
	69	99	99	6	52	42	7	7	10
39	72	97	99	10	49	57	7	8	8
43	74	94	96	7	57	59	3	12	8
47	66	94	95	9	49	62	5	11	13
51	79	97	94	10	62	61	5	8	9
55	80	96	97	11	70	58	5	7	6
59	73	96	98	12	59	51	0	7	10
63	75	98	95	17	60	68	3	8	7
67	74	99	96	21	66	63	4	8	4
71	78	98	93	20	59	68	0	3	4
75	75	96	96	16	61	72	3	3	9
79	78	100	99	16	61	58	1	8	8
83	69	96	98	18	74	71	1	5	11
87	77	94	98	24	70	56	1	7	9
91	72	97	92	20	67	68	4	5	4
95	76	97	93	20	52	61	0	7	8
99	82	91	95	24	68	63	4	4	2
103	77	94	98	25	56	61	1	7	6
Average	70.33	96.29	96.21	12.00	50.04	50.29	3.25	7.46	8.38

TABLE V. PAIRED T-TEST FOR BASIC GEP, CONTRACTION OPERATOR, SLICE CROSSOVER AND COMBINATION

SRP	Success Rate Mean Comparison	p-value	Significance (α = 0.05)
SRP 1	Contraction >Basic ALGEP	0.79	Not Significant
	Slice crossover >Basic ALGEP	1.26E-11	Significant
	Combination >Basic ALGEP	1.17E-11	Significant
	Slice crossover >Contraction	1.35E-12	Significant
	Combination >Contraction	1.53E-12	Significant
	Combination <Slice crossover	0.89	Not Significant
SRP 2	Contraction >Basic ALGEP	0.02	Significant
	Slice crossover >Basic ALGEP	4.39E-11	Significant
	Combination >Basic ALGEP	5.86E-11	Significant
	Slice crossover >Contraction	3.26E-11	Significant
	Combination >Contraction	4.04E-11	Significant
	Combination >Slice crossover	0.87	Not Significant
SRP 3	Contraction >Basic ALGEP	0.88	Not Significant
	Slice crossover >Basic ALGEP	2.90E-07	Significant
	Combination >Basic ALGEP	7.49E-07	Significant
	Slice crossover >Contraction	2.22E-06	Significant
	Combination >Contraction	7.36E-07	Significant
	Combination >Slice crossover	0.18	Not Significant