A Hierarchical Cooperative Genetic Programming for Complex Piecewise Symbolic Regression

***Abstract*—In regression analysis, methodologies range from black-box approaches like artificial neural networks to white-box techniques like symbolic regression. Renowned for its transparency and interpretability, symbolic regression has become increasingly prominent in elucidating complex data relationships. Nevertheless, its effectiveness in managing complex piecewise symbolic regression tasks poses significant challenges. This paper introduces a novel Hierarchical Cooperative Genetic Programming (HCGP) framework to address this issue. The HCGP model utilizes a unique hierarchical structure, incorporating dual cooperative genetic programming (GP) populations. This innovative design significantly enhances the capability to solve complex piecewise symbolic regression problems. Implementing a scenario-based GP is central to the HCGP framework, which strategically selects the appropriate underlying calculation GP. This feature enables the system to autonomously learn and adapt to complex scenarios, selecting the most suitable calculation GPs for each case. Our HCGP approach distinguishes itself from traditional and state-of-the-art methods. It demonstrates particular proficiency in modeling piecewise expressions within complex scenarios. The empirical evaluation of our model, conducted using benchmark datasets, has exhibited its superior accuracy and computational efficiency. This progress emphasizes the potential of HCGP in sophisticated data modeling and marks a substantial advancement in hierarchical structure in complex piecewise symbolic regression.**

***Index Terms*—genetic programming, symbolic regression, hierarchical structure, evolutionary algorithm**

I. Introduction

The evolution of regression analysis, a cornerstone in statistical data modeling, has witnessed significant advancements tailored to address the complexities of contemporary data structures. Amidst these developments, symbolic regression is a pivotal methodology, offering a level of interpretability often lacking in black-box approaches, such as conventional artificial neural networks [1]. Its ability to generate explicit mathematical models that reveal intricate data relationships has positioned symbolic regression as a preferred tool in various scientific and engineering applications.

In real-world problem-solving, the demand often extends beyond mere performance; factors such as solutions' interpretability, modifiability, and traceability are increasingly valued. Symbolic regression caters to these requirements, presenting a versatile, transparent approach that resonates with practical applications. Its emphasis on creating understandable and adaptable models aligns with the growing need for solutions that are effective, comprehensible, and accountable in real-life scenarios. This aspect of symbolic regression underscores its significance and broad applicability in addressing real-world challenges, where the clarity and adaptability of solutions often take precedence over performance metrics alone.

Notwithstanding its advantages, symbolic regression faces considerable challenges, mainly when applied to complex, piecewise symbolic regression tasks. These tasks, characterized by distinct data behaviors in different input domains, demand a modeling approach capable of discerning and accurately representing these segmented relationships. Existing methods often struggle with the nuanced demands of such problems, especially those with complex conditions.

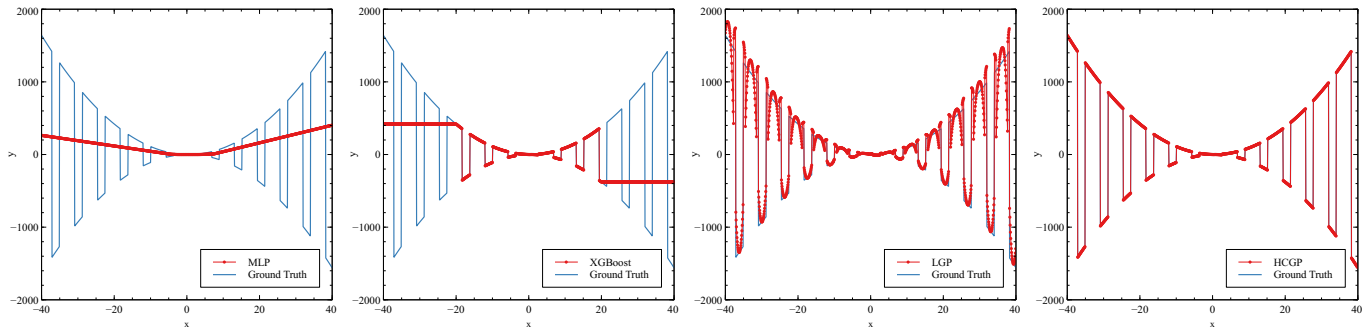
In response to this challenge, this paper introduces a Hierarchical Cooperative Genetic Programming (HCGP) framework, a novel tailored for complex piecewise symbolic regression. The HCGP framework sets itself apart through a unique hierarchical structure incorporating dual cooperative GP populations. This configuration substantially enhances the model's ability to efficiently navigate and resolve segmented symbolic regression tasks. Central to HCGP is the adoption of a scenario-based GP approach. This method effectively navigates various complex scenarios, autonomously selecting GP individuals' most appropriate calculation. This feature reduces the search space and boosts the model's comprehensibility, handling complex piecewise symbolic regression challenges adeptly.

In our investigation, we considered a simple expression under a complex condition, as detailed in Equation 1. We tested several models, including Multilayer Perceptron (MLP), Extreme Gradient Boosting (XGBoost), Logic Genetic Programming (LGP), and our proposed HCGP on this symbolic function regression problem. The findings in Fig. 1 revealed that conventional regression methods struggle with complex scenario problems, even when the scenario involves just an essential sine function. This challenge has led us to develop the HCGP method tailored to address such issues.

$$y = \begin{cases} -x^2 + x, & sin(x) \geq 0.5 \\ x^2 - x, & sin(x) < 0.5 \end{cases} \tag{1}$$

A notable feature of our HCGP approach is the utilization of the population dedicated to evolving the scenario selection of GP individuals. This innovation significantly enhances the model's capability to fit problems characterized by complex scenarios. Furthermore, the adaptability and robustness of this method make it particularly well-suited for addressing a variety of real-world problems, which often encompass similarly complex scenarios. Therefore, our approach demonstrates improved performance in theoretical models and holds

* Corresponding author



(a) MLP (b) XGBoost (c) LGP (d) HCGP

Fig. 1. MLP, XGBoost, LGP and HCGP's Best Performance on Equation 1

substantial promise for practical applications across diverse domains.

In this article, we introduce a novel approach to symbolic regression - the Hierarchical Cooperative Genetic Programming (HCGP), inspired by research on cooperative coevolution GP [2], [3], novel GP representations [4], [5], [6] and principally based on the double-layer cooperative genetic programming framework [7]. We have applied this method to complex piecewise symbolic regression problems, and considering the unique characteristics of symbolic regression, we have refined its evolutionary process. This enhancement significantly improves its performance. The main contributions of this paper are manifold.

- Innovative Hierarchical Genetic Programming Architecture: We propose a novel hierarchical genetic programming framework to address piecewise symbolic regression problems. This approach significantly improves the fitting of expressions with complex conditions, showcasing a marked advancement in symbolic regression techniques.
- Redesigned Evolutionary Methods for HCGP: We have redeveloped the Hierarchical Cooperative Genetic Programming (HCGP) evolutionary process, including enhanced crossover and mutation strategies. These modifications have resulted in a notable improvement in the algorithm's performance.
- Extensive Comparative Analysis with State-of-the-Art Methods: The efficacy of the HCGP has been rigorously tested across multiple symbolic regression datasets. Our comprehensive comparisons with various state-of-the-art methods demonstrate the superiority of HCGP in terms of accuracy and efficiency.

The remainder of this paper is organized as follows. Section II presents the background and literature review on symbolic regression and the applications of the symbolic regression methods in real-world applications. Section III describes the proposed HCGP method, outlining its structure and evolution process. Section IV discusses the experimental results, providing a comprehensive analysis of the performance of the HCGP and other state-of-the-art methods over several datasets. Finally, Section V concludes the paper, summarizing the essential findings and suggesting potential avenues for future research in this area.

II. Background and Literature Review

Symbolic regression, a unique approach within the broader spectrum of regression analysis, distinguishes itself by autonomously constructing mathematical models that best describe a dataset [8]. This process is typically carried out using evolutionary algorithms [9] such as genetic programming. Unlike conventional regression techniques that fit data to predefined models, symbolic regression explores various possible models, making it exceptionally versatile and powerful in uncovering underlying data relationships [10].

The objective of symbolic regression is to discover a mapping $\hat{y} = f(\mathbf{x}, \theta) : \mathbb{R}^d \rightarrow \mathbb{R}$, utilizing a dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Symbolic regression endeavors to identify an expression for f and corresponding parameters θ that encapsulate the relationship between all instances of $\mathbf{x}_i$ and y_i. Once the appropriate f and θ are determined, it becomes possible to predict $\hat{y}_i$ using $\mathbf{x}_i$, even for values of $\mathbf{x}_i$ that are out of the range of the original dataset. Moreover, the training process of symbolic regression involves minimizing the difference between the predicted values $\hat{y}$ and the actual values y.

The applications of symbolic regression are impressively varied and extensive. It has been instrumental in rediscovering physical laws from experimental data in physics [11], [12], [13]. This capability to extract fundamental relationships from raw data points makes scientific discovery and verification invaluable. In the ecological domain [14], [15], symbolic regression aids in the modeling of complex ecological interactions, which are often nonlinear and involve multiple interacting factors. Its ability to handle such complexity is critical for ecological research and conservation efforts.

Symbolic regression has also been applied in forecasting financial markets in the economic sphere [16]. Here, the flexibility of symbolic regression to model nonlinear and non-stationary time series data offers a significant advantage over traditional linear models. Financial analysts and economists leverage this to predict market trends, assess risks, and make informed investment decisions [17], [18].

Engineering applications of symbolic regression encompass system optimization and predictive maintenance. This methodology enables engineers to model intricate systems, discern vital factors influencing performance, and anticipate failures before they manifest [19]. Furthermore, symbolic regression has been effectively applied in areas such as truck dispatching [7], [20], job shop scheduling [21], [22], vehicle routing [23], and robotic control [24], [25], among others. These diverse applications highlight its versatility and utility in addressing complex engineering challenges.

Symbolic regression's role in data analytics is also noteworthy [26]. It provides a tool for data scientists to derive meaningful insights from large and complex datasets. Its ability to generate human-readable models interprets results more efficiently, enhancing the decision-making process in various business and research contexts.

Traditional symbolic regression methods, while powerful, exhibit limitations when dealing with piecewise symbolic regression, especially in scenarios characterized by complex piecewise functions. Piecewise functions consisting of several sub-functions defined on a specific interval are commonplace in real-world data representing distinct regimes or operational modes [27].

One core limitation of conventional approaches is their struggle with identifying the boundaries and appropriate models for each segment in piecewise data. This challenge is compounded in non-linear, discontinuous, or high-dimensional data scenarios, where the transition points between different regimes are not easily discernible [28], [29].

Moreover, traditional methods typically excel only within the range of data on which they are trained, lacking the capability to uncover the underlying rules that govern the data. This limitation constrains their predictive power to scenarios that fall within the bounds of the training data, rendering them ineffective for extrapolation or predictions beyond the training range [30], [31].

In summary, while symbolic regression offers a potent tool for data modeling in diverse real-world applications, traditional methods face significant challenges in handling complex piecewise regression problems. This limitation necessitates the development of more advanced, flexible, and autonomous symbolic regression techniques capable of effectively deciphering and modeling complex, piecewise scenarios without extensive manual intervention.

III. Methodologies

Addressing the challenges of traditional symbolic regression in managing complex conditions within piecewise symbolic regression, this section is dedicated to exploring the conventional methods of piecewise symbolic regression, the Logic Genetic Programming. Concurrently, it introduces and elaborates on our innovative HCGP approach. This section aims to clarify the significance of logic operators in symbolic regression and offers an in-depth examination of the HCGP's structure. It emphasizes the distinct benefits of HCGP, particularly in comparison to traditional LGP, underscoring its enhanced capabilities in handling the intricacies of piecewise symbolic regression.

A. Logic Genetic Programming

As previously mentioned, evolutionary algorithms are a predominant method for solving symbolic regression problems. While alternatives like decision trees [32], recurrent neural networks [33], and even transformers [34] exist for symbolic regression, GP-based methods are the most commonly employed. According to SRbench data [35], GP-based methods rank among the best performers among all approaches. This high performance is due to GP's extensive search capabilities. Utilizing evolutionary algorithms with large populations, GP is adept at finding suitable solutions within the expansive search space characteristic of symbolic regression [36]. Additionally, the symbolic nature of these problems often hinders the generation of a differentiable objective function, a requirement for traditional learning methods. Evolutionary algorithms, which do not necessitate a differentiable objective function, are particularly effective in this domain, contributing to the success of GP in symbolic regression tasks.

As depicted in Fig. 1, traditional GP can be divided into two main categories: Arithmetical GP (AGP), which solely employs arithmetic operators, and Logic GP (LGP), which integrates logic operators. This paper is concerned with complex piecewise symbolic regression problems. The absence of logic operators in AGP is a notable limitation for fitting piecewise functions. Consequently, our discussion primarily centers on the LGP method, which we also use as a comparison in our study. In LGP, ternary operators like "If-Else" are utilized. These operators enable the selection of outputs from the second or third subtree, depending on the condition set in the first subtree, thereby effectively tackling the intricacies of piecewise functions. However, as observed in Fig. 1, while LGP can fit complex piecewise functions, it does not perform as well on more straightforward fitting challenges, such as those presented in Equation 1. In contrast to AGP, the introduction of logic operators in LGP significantly enlarges the search space. Although LGP can identify a solution, the immense size of the search space poses a significant challenge in efficiently finding the optimal one.

In this research, we utilize the R-Square (coefficient of determination) as the performance evaluation criterion for all algorithms, as shown in Equation (2). In the R-squared formula, y_i denotes the actual values, whereas $\hat{y}_i$ represents the predicted values. Additionally, $\overline{y}$ signifies the mean of

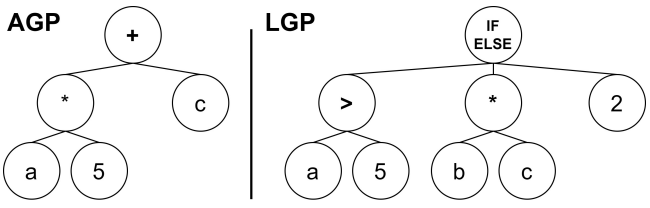


Fig. 2. Arithmetical GP and Logic GP Structure

the actual values, and n is the number of data points. R-Squared indicates the proportion of variance in the dependent variable that can be predicted from the independent variables. It measures how much the regression predictions align with the actual data points. The value of R-squared ranges from 0 to 1, where a value of 0 implies that the model does not account for any of the variability of the actual data around its mean, while a value of 1 indicates the model accounts for all the variability of the response data around its mean.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2}$$

Algorithm 1 outlines the main steps and components of LGP and HCGP. The process begins with creating an initial population determined by a predefined population size. During the evolutionary process, the fitness of each individual is calculated based on the R-squared value (R^2) as shown in Equation 2, with an additional penalty for oversized GP trees. A new population is then produced, where a random genetic operator—either crossover, mutation, or reproduction—is selected to generate offspring for the new population. This study adopts a non-elitist tournament selection method to enhance diversity. This evolutionary cycle continues until the maximum number of generations is reached. The subsequent subsections detail the technical aspects of these components.

Algorithm 1 LGP, and HCGP Evolution Process

Require: Initial Parameters $initial$
  $p \leftarrow NewPopulation$
  $p.initial\_individuals(initial.population\_size)$
  $generation \leftarrow 0$
  while $generation < initial.max\_generation$ **do**
    $p.calculate\_fitness()$
    $p.penalize\_long\_individuals()$
    $next\_generation \leftarrow NewEmptyPopulation$
    while $next\_generation.size() < p.size()$ **do**
      Insert an $individual$ to $next\_generation$ by Crossover, Mutation, or Reproduction on p
    end while
    $p \leftarrow next\_generation$
    $generation \leftarrow generation + 1$
  end while

1) Crossover: In the crossover operation, two parent individuals are chosen through `tournament selection` size 7. To produce two offspring, these parents undergo a `single point crossover` operation. Fig. 3 provides an example of this process: a subtree from parent 2 is combined with parent 1 to create offspring 1, while offspring 2 is formed by merging a subtree from parent 1 into parent 2.

2) Mutation: The mutation operation involves modifying a single individual to produce a new offspring. A mutation point is randomly selected, and a new, randomly generated subtree is grown from this point. This modification ensures the overall tree remains within the depth limit, as illustrated in Fig. 4.

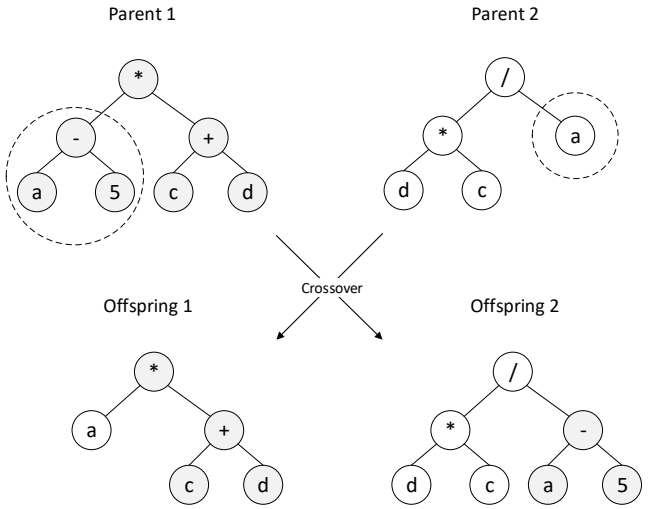


Fig. 3. Crossover operation in AGP & LGP

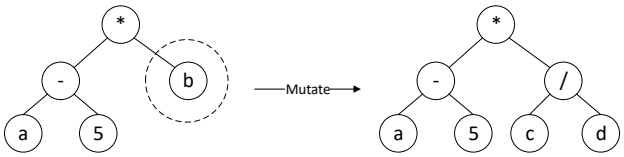


Fig. 4. Mutation operation in AGP & LGP

B. Hierarchical Cooperative Genetic Programming

Building upon the framework of LGP and its inherent limitations, our objective was to preserve LGP's adeptness in handling complex multi-scenario piecewise problems yet streamline the search space for greater efficiency. To accomplish this, we adopted a strategy that split the search space into two distinct realms: the scenario space and the calculation space. This division is operationalized by training two cooperative GP populations, each specializing in its respective search domain. As illustrated in Fig. 5, the structure of an HCGP individual is conceptually split into two parts. The upper layer is dedicated to exploring the scenario space and identifying the relevant scenarios for the problem. In contrast, the lower layer concentrates on the calculation, focusing on the computational aspects required for the regression. This division reduces the search space and leverages the strengths of both domains, fostering a more effective and efficient symbolic regression approach.

However, this bifurcated approach presents a new challenge. The individuals in the scenario and calculation layers cannot independently execute the symbolic regression task. To overcome this, we have devised a method where one individual from the scenario layer is paired with another from the calculation layer, forming a cohesive rule. An HCGP individual is typically composed of several such composite rules. When an input $\mathbf{x}_i$ is introduced, the scenario individual evaluates whether this input aligns with its designated scenario. If yes, the corresponding calculation individual within that rule

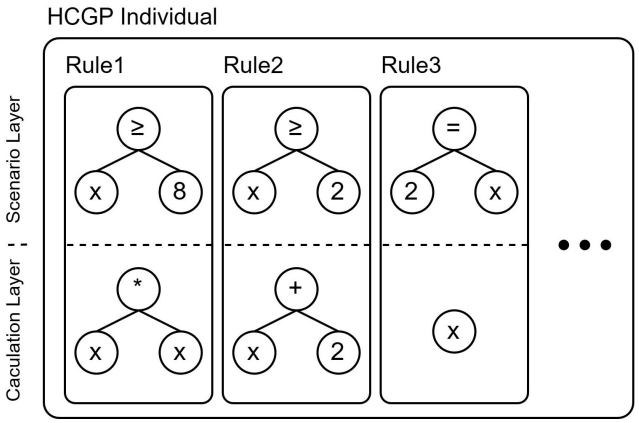


Fig. 5. Structure of HCGP Individual

determines the outcome. In cases where there is no match, the process iteratively progresses to the subsequent rule. This procedure is repeated until an appropriate scenario is found or all the rules have been considered. When no matching scenario is identified, the output is computed using the calculation individual associated with the final rule. This systematic approach ensures a comprehensive evaluation, enhancing the effectiveness of the symbolic regression process.

With the introduction of this novel structure, a redesign of the evolutionary process is imperative. Regarding fitness evaluation, we maintain the same overall fitness metric employed in LGP for the entire HCGP individual. Additionally, we have introduced a new fitness formula for each rule within the HCGP individual, as detailed in Equation 3. In this rule-specific fitness calculation, m_j indicates the count of instances $\mathbf{x}$ successfully matched by the scenario individual in $rule_k$ during testing. Concurrently, R_k^2 reflects the R^2 value corresponding to the predictions made by the calculation individual in $rule_k$. This tailored fitness metric is crucial in the evolutionary process. It represents the specific performance of each rule during the fitting process and its effectiveness. This facilitates a practical assessment of each rule's performance within the HCGP individual. This evaluation is used in the ranking, mutation, crossover, or removal of rules within the individual, optimizing the overall efficacy of the HCGP framework.

$$fitness_k = m_j \times R_k^2 \tag{3}$$

The overall evolutionary process of HCGP is similar to that of LGP, detailed in Algorithm 1. However, the crossover and mutation methods have explicitly been redesigned to align with the unique structure and requirements of the HCGP framework:

1) Crossover: In the crossover process, like the method used in LGP, two parent individuals are initially chosen by tournament selection. Following this, using `roulette wheel selection` based on rule-specific $fitness_k$, exchange rules are selected from each parent, favoring those with

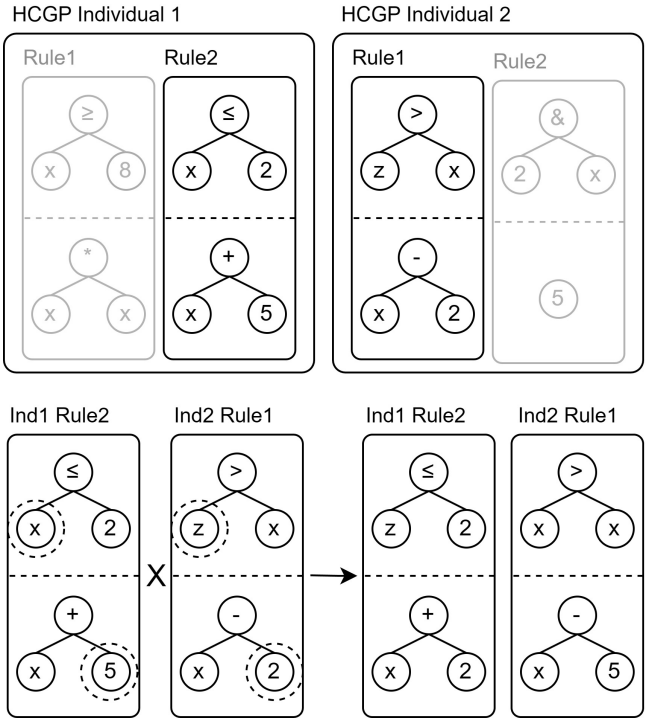


Fig. 6. Crossover operation in HCGP

higher fitness. The crossover operation is divided into four distinct types. The first type exchanges two rules without crossing over the individuals in either the scenario or calculation layers. The second type involves crossover between scenario individuals, while the third type is focused on crossover among calculation individuals. The fourth and most comprehensive type involves a crossover between the scenario and calculation layers. When a new rule is formulated, it either randomly replaces its corresponding original rule in the parent individual or is inserted at the beginning or end of the parent, thus creating a new child individual. It is important to note that if the rule count of an HCGP individual reaches the maximum rule limit, new rules will only replace existing ones rather than being added. An example of the fourth type of crossover is depicted in Fig. 6. This approach to crossover in HCGP individuals aligns with the established principles in LGP.

2) Mutation: The mutation operation engages a single parent, from which a rule within the HCGP individual is selected for mutation. The selection is based on the rule's specific $fitness_k$, utilizing `roulette wheel selection`. Rules with lower fitness are more likely to be chosen for mutation. The mutation process itself is divided into two types. The first type involves deleting the selected rule. The second type, more complex, includes mutating both the scenario and calculation individuals within a rule, thus creating a new rule. Following the creation of this new rule, it is reinserted into its original position in the parent individual. An illustration of the mutation process is provided in Fig.7.

Besides the crossover and mutation processes, the remaining

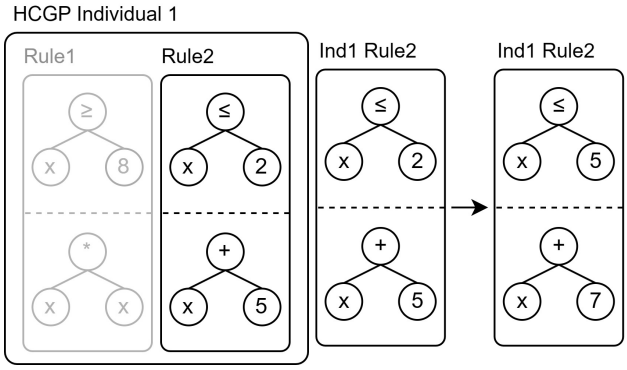


Fig. 7. Mutation operation in HCGP

aspects of the HCGP evolutionary mechanism are maintained consistent with those of LGP to ensure a fair comparison. The following section will detail two experimental parts designed to reveal the performance of the HCGP method. These experiments focus on a simple function problem and benchmark datasets, comprehensively evaluating HCGP's capabilities.

IV. Experiments and Discussion

This section outlines two types of tests. The first is a piecewise function fitting problem, as shown in Equation 1, and the second involves testing on the PMLB benchmark set [37]. The performance of HCGP in these tests demonstrates the superiority of the novel method proposed. Furthermore, comparisons between HCGP and other state-of-the-art symbolic regression methods highlight HCGP's efficacy in fitting real, complex piecewise symbolic regression problems, showcasing its potential in solving real-world challenges. The proposed HCGP method performs better than others in complex piecewise symbolic regression scenarios.

The configuration of LGP and HCGP adheres to the parameters established in the referenced study [7]. The crossover, mutation, and reproduction rates are set at 0.6, 0.3, and 0.1, respectively. We have constituted a population size of 1024, with the initial population generated using the ramped half-and-half method. Parent selection is conducted via a tournament process with a size of 7, and the evolution is capped at 100 generations. The maximum depth for all trees is limited to 10. Within the HCGP framework, the number of rules is adjustable, ranging from 1 to 10. The operators incorporated into this research consist of addition, subtraction, multiplication, protected division, if-else, and relational operators. For fitting simple piecewise functions, sine and cosine operators are included, enhancing the model's capability to capture the intricacies of the problem. The terminal set encompasses both the input variable and an integer constant, with values spanning from 1 to 10.

A. Piecewise Function Fitting

This simple piecewise fitting problem shown in Equation 1, while seemingly straightforward, incorporates a complex condition due to the sine function, making it a piecewise function fitting problem with complex conditions. For this function, we randomly sampled 200 points within the range of x from -20 to 20 for training and endeavored to reconstruct the relationship between x and y through regression. Our comparison involved four methodologies: MLP, XGBoost, LGP, and our proposed HCGP. These represent traditional neural networks, ensemble learning methods, logic-based GP approaches, and our novel method. Table I shows that each algorithm underwent 10 iterations with varying random seeds. HCGP and XGBoost emerged as the most effective, nearly achieving an R^2 value of 1, indicative of near-perfect fitting for this complex condition. Contrarily, MLP struggled significantly with this segmented problem. LGP showed commendable performance but did not match the efficacy of HCGP.

TABLE I
MLP, XGBoost, LGP and HCGP Test Results on Equation 1 (R^2)

	MLP	**XGBoost**	**LGP**	**HCGP**
Min	0.06	0.98	0.32	0.98
Mean	0.07	0.99	0.71	0.99
Max	**0.07**	**1**	**0.96**	**1**

Particularly intriguing was the performance of the XGBoost algorithm, which, despite registering the highest scores in Table I with an R^2 value of 1, displayed limitations, as shown in Fig. 1. Within the training range of -20 to 20, XGBoost precisely predicted all data points. However, its predictions did not align well with the y outputs beyond the training set range, suggesting that while XGBoost excels in regression, it primarily learns from the training set without delving into the fundamental relationship between x and y. In contrast, symbolic regression methods like LGP and HCGP demonstrated adeptness within and outside the training interval. This underscores a critical distinction between symbolic regression and traditional regression techniques: the former delves into the core relationship between inputs and outputs, providing a universal model adaptable to various scenarios vital in the uncertainty-ridden real world. The robustness of diverse inputs is incredibly crucial. LGP, for instance, despite not perfectly matching the function model, showed robust performance for extra inputs, a trend even more pronounced in HCGP. Therefore, we posit that symbolic regression-based methods hold significant promise for broader application in real-world settings.

B. Benchmark Datasets Test

As previously mentioned, we utilized the PMLB dataset as our benchmark in this study. Within PMLB, we randomly selected five datasets that are representative of evaluating the performance of our HCGP method against other state-of-the-art techniques. For methods other than LGP and HCGP, we directly used data from SRBench [35]. SRBench is a comprehensive benchmark that includes a range of advanced ensemble learning and symbolic regression algorithms, such as XGBoost and Operon. In SRBench, hyperparameters for all algorithms, including the number of trees for XGBoost and

TABLE II
EXPERIMENT RESULTS (R^2)

	Titanic	Banana	Slenth Case	Faculty Salaries	US Crime
HCGP	**0.32***	0.51	**0.76***	**0.95***	**0.85***
LGP	0.19	0.31	0.58	0.78	0.80
AFP	0.28	0.61	0.62	0.75	0.64
AFP_FE	0.27	0.62	0.66	0.80	0.71
AIFeynman	-0.03	-0.80	-5.98	-3.66	-0.19
AdaBoost	0.27	0.49	0.60	0.72	0.82
BSR	0.24	0.01	0.32	0.70	0.18
DSR	0.24	0.47	0.58	0.81	0.66
EPLEX	0.24	0.53	0.54	0.78	0.77
FEAT	0.27	0.48	0.56	0.65	0.68
FFX	0.29	0.51	-25.08	0.05	0.71
GP-GOMEA	0.30	0.63	0.58	0.75	0.73
ITEA	0.28	0.63	0.35	0.76	0.68
KernelRidge	0.30	**0.73***	0.60	0.74	0.78
LGBM	0.26	0.71	-0.01	-0.06	-0.06
Linear	0.22	0.00	0.57	0.78	0.78
MLP	0.29	0.70	0.65	0.66	0.76
MRGP	0.30	0.72	-0.54	0.80	0.29
Operon	0.30	0.68	0.34	**0.95***	0.54
RandomForest	0.30	0.69	0.45	0.66	0.77
SBP-GP	0.30	0.67	0.47	0.75	0.69
XGB	0.22	0.71	0.53	0.71	0.74
gplearn	0.05	0.57	0.66	0.76	0.72

* Best algorithm.

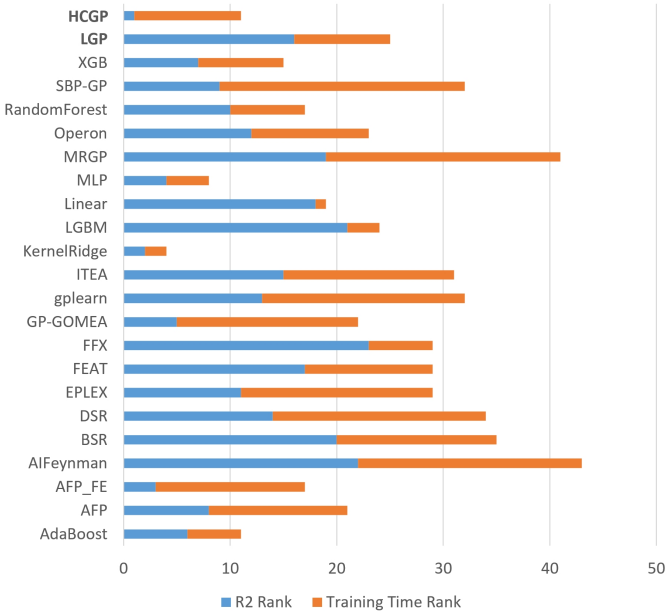


Fig. 8. Performance vs. Training Time

population sizes for Operon, have been thoroughly optimized using the halving grid search method. Our study employs the experimental protocols and results provided by SRBench, ensuring a fair and typical comparison between our algorithm and other leading methods.

The results showcased in Table 2 indicate that our proposed HCGP method exhibits strong performance across most datasets. This affirms the efficacy of the hierarchical structure integral to our approach. By differentiating between scenario and calculation layers, we have surpassed the performance of existing methods in various real-world datasets. Furthermore, when LGP is considered an ablation test, lacking the hierarchical structure, it is clear that HCGP consistently outperforms across all datasets. This suggests that integrating a hierarchical structure in addressing piecewise symbolic regression problems effectively narrows down the search space for GP, thereby aiding in identifying superior solutions. These outcomes underscore the importance of incorporating a hierarchical structure and reinforce the superiority of our developed HCGP method.

We further extensively evaluated all methods, comparing both performance and computational cost. As depicted in Fig. 8, our method achieves notable performance and does so with a reasonable training duration. In this ranking, a lower number indicates a better outcome. Our method excels not just in terms of performance but also in maintaining a modest increase in training time.

V. CONCLUSION AND FUTURE WORK

This study introduces the HCGP framework to address the complexities inherent in piecewise symbolic regression tasks. Diverging from traditional methods, HCGP's innovative hierarchical structure, featuring dual cooperative genetic programming populations, has demonstrated exceptional proficiency in modeling piecewise expressions within complex scenarios. The empirical evaluations of our model on benchmark datasets highlighted its superior accuracy, computational efficiency, and training cost, marking a significant advancement in symbolic regression techniques.

Looking ahead, there are several promising avenues for future work. One potential exploration area involves enhancing the HCGP framework by dynamically adjusting each terminal's appearance rate of different layers based on learned knowledge during training. This adaptation could further expedite the convergence speed and improve the overall performance of the HCGP model. Additionally, the research could refine the scenario-based genetic programming approach, exploring more sophisticated selection and combination strategies for various scenarios. Another intriguing direction is the application of HCGP in diverse real-world domains, such as bioinformatics, financial modeling, environmental science, and job shop scheduling, where its ability to extract interpretable models from complex data can be particularly valuable. These future endeavors will enhance the capabilities of HCGP and contribute significantly to the field of symbolic regression and data analysis at large.

REFERENCES

[1] D. Maulud and A. M. Abdulazeez, "A review on linear regression comprehensive in machine learning," *Journal of Applied Science and Technology Trends*, vol. 1, no. 4, pp. 140–147, 2020.

[2] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 193–208, 2013.

[3] M. A. Potter and K. A. D. Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary computation*, vol. 8, no. 1, pp. 1–29, 2000.

[4] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "A computational study of representations in genetic programming to evolve dispatching rules for the job shop scheduling problem," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 621–639, 2012.

[5] N. X. Hoai, R. I. McKay, and D. Essam, "Representation and structural difficulty in genetic programming," *IEEE Transactions on evolutionary computation*, vol. 10, no. 2, pp. 157–166, 2006.

[6] Y. Bi, B. Xue, and M. Zhang, "Genetic programming with a new representation to automatically learn features and evolve ensembles for image classification," *IEEE transactions on cybernetics*, vol. 51, no. 4, pp. 1769–1783, 2020.

[7] X. Chen, R. Bai, R. Qu, and H. Dong, "Cooperative double-layer genetic programming hyper-heuristic for online container terminal truck dispatching," *IEEE Transactions on Evolutionary Computation*, 2022.

[8] A. Diveev and E. Shmalko, *Machine learning control by symbolic regression*. Springer, 2021.

[9] T. Bartz-Beielstein, J. Branke, J. Mehnen, and O. Mersmann, "Evolutionary algorithms," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 4, no. 3, pp. 178–195, 2014.

[10] E. P. Wigner, "The unreasonable effectiveness of mathematics in the natural sciences," in *Mathematics and science*, pp. 291–306, World Scientific, 1990.

[11] N. Makke and S. Chawla, "Interpretable scientific discovery with symbolic regression: a review," *Artificial Intelligence Review*, vol. 57, no. 1, p. 2, 2024.

[12] S.-M. Udrescu and M. Tegmark, "Ai feynman: A physics-inspired method for symbolic regression," *Science Advances*, vol. 6, no. 16, p. eaay2631, 2020.

[13] L. S. Keren, A. Liberzon, and T. Lazebnik, "A computational framework for physics-informed symbolic regression with straightforward integration of domain knowledge," *Scientific Reports*, vol. 13, no. 1, p. 1249, 2023.

[14] Y. Chen, M. T. Angulo, and Y.-Y. Liu, "Revealing complex ecological dynamics via symbolic regression," *BioEssays*, vol. 41, no. 12, p. 1900069, 2019.

[15] D. Vázquez, R. Guimerà, M. Sales-Pardo, and G. Guillén-Gosálbez, "Automatic modeling of socioeconomic drivers of energy consumption and pollution using bayesian symbolic regression," *Sustainable Production and Consumption*, vol. 30, pp. 596–607, 2022.

[16] P. Truscott and M. F. Korns, "Explaining unemployment rates with symbolic regression," *Genetic Programming Theory and Practice XI*, pp. 119–135, 2014.

[17] A. F. Sheta, S. E. M. Ahmed, and H. Faris, "Evolving stock market prediction models using multi-gene symbolic regression genetic programming," *Artificial Intelligence and Machine Learning*, vol. 15, no. 1, pp. 11–20, 2015.

[18] P. Venegas, I. Britez, and F. Gobet, "Ensemble models using symbolic regression and genetic programming for uncertainty estimation in esg and alternative investments," *Big Data in Finance: Opportunities and Challenges of Financial Digitalization*, pp. 69–91, 2022.

[19] W. T. Hale, E. Safikou, and G. M. Bollas, "Inference of faults through symbolic regression of system data," *Computers & Chemical Engineering*, vol. 157, p. 107619, 2022.

[20] X. Chen, R. Bai, R. Qu, H. Dong, and J. Chen, "A data-driven genetic programming heuristic for real-world dynamic seaport container terminal truck dispatching," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, 2020.

[21] Z. Huang, Y. Mei, and M. Zhang, "Investigation of linear genetic programming for dynamic job shop scheduling," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, IEEE, 2021.

[22] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Genetic programming with lexicase selection for large-scale dynamic flexible job shop scheduling," *IEEE Transactions on Evolutionary Computation*, 2023.

[23] W. Yi, R. Qu, L. Jiao, and B. Niu, "Automated design of metaheuristics using reinforcement learning within a novel general search framework," *IEEE Transactions on Evolutionary Computation*, 2022.

[24] M. A. Lewis, A. H. Fagg, A. Solidum, *et al.*, "Genetic programming approach to the construction of a neural network for control of a walking robot.," in *ICRA*, pp. 2618–2623, Citeseer, 1992.

[25] P. Silva, C. P. Santos, V. Matos, and L. Costa, "Automatic generation of biped locomotion controllers using genetic programming," *Robotics and Autonomous Systems*, vol. 62, no. 10, pp. 1531–1548, 2014.

[26] N. J. Christensen, S. Demharter, M. Machado, L. Pedersen, M. Salvatore, V. Stentoft-Hansen, and M. T. Iglesias, "Identifying interactions in omics data for clinical biomarker discovery using symbolic regression," *Bioinformatics*, vol. 38, no. 15, pp. 3749–3758, 2022.

[27] A. Makady, A. de Boer, H. Hillege, O. Klungel, W. Goettsch, *et al.*, "What is real-world data? a review of definitions based on literature and stakeholder interviews," *Value in health*, vol. 20, no. 7, pp. 858–865, 2017.

[28] M. K. Transtrum, B. B. Machta, and J. P. Sethna, "Why are nonlinear fits to data so challenging?," *Physical review letters*, vol. 104, no. 6, p. 060201, 2010.

[29] D. Mayne, "Nonlinear model predictive control: Challenges and opportunities," *Nonlinear model predictive control*, pp. 23–44, 2000.

[30] M. Quade, M. Abel, K. Shafi, R. K. Niven, and B. R. Noack, "Prediction of dynamical systems by symbolic regression," *Physical Review E*, vol. 94, no. 1, p. 012214, 2016.

[31] C. Wilstrup and J. Kasak, "Symbolic regression outperforms other models for small data sets," *arXiv preprint arXiv:2103.15147*, 2021.

[32] Y.-Y. Song and L. Ying, "Decision tree methods: applications for classification and prediction," *Shanghai archives of psychiatry*, vol. 27, no. 2, p. 130, 2015.

[33] X. Chen, R. Bai, R. Qu, and J. Dong, "Neural network assisted genetic programming in dynamic container port truck dispatching," in *2023 IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2023.

[34] P.-A. Kamienny, S. d'Ascoli, G. Lample, and F. Charton, "End-to-end symbolic regression with transformers," *Advances in Neural Information Processing Systems*, vol. 35, pp. 10269–10281, 2022.

[35] W. La Cava, P. Orzechowski, B. Burlacu, F. O. de França, M. Virgolin, Y. Jin, M. Kommenda, and J. H. Moore, "Contemporary symbolic regression methods and their relative performance," *arXiv preprint arXiv:2107.14351*, 2021.

[36] J. He and X. Yao, "From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 495–511, 2002.

[37] R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore, "Pmlb: a large benchmark suite for machine learning evaluation and comparison," *BioData mining*, vol. 10, pp. 1–13, 2017.