

# Report for the Automatically Generated Graph Evolutionary Algorithm

Hongxiang Geng Q23201173@stu.ahu.edu.cn, Institutes of Physical Science and Information Technology, Anhui University, Hefei, China	Sizhe Zhu Q23301275@stu.ahu.edu.cn, Institutes of Physical Science and Information Technology, Anhui University, Hefei, China	Jun Wang 18260231386@163.com, School of Computer Science and Technology, Anhui University, Hefei, China
Huaiyuan Gu guw98874@gmail.com, School of Computer Science and Technology, Anhui University, Hefei, China	Xiuming Zou 1751034987@qq.com, School of Computer Science and Technology, Anhui University, Hefei, China	Hao Jiang haojiang@ahu.edu.cn, School of Computer Science and Technology, Anhui University, Hefei, China
Cheng He chenghe_seee@hust.edu.cn, School of Electrical and Electronic Engineering, Huazhong University of Science and Technology, Wuhan, China		

## I. Introduction

**E**volutionary Algorithms (EAs) are widely considered as effective tools for solving optimization problems, owing to their population-based nature and the flexibility to incorporate a wide variety of variation operators. These operators such as crossover, mutation, and selection can be adapted or customized to explore and exploit the search space efficiently, making EAs particularly suitable for complex, multimodal, and high-dimensional optimization tasks. However, most traditional EAs like DE [1], PSO [2], CMA-ES [3] have poor performance because of the fixed operator configuration throughout the optimization process. In fact, determining the most suitable configuration of EA for a specific problem is almost relied on expertise and labor-extensive. To address this issue, this framework utilizes a new perspective to organize these modules into trainable functional components, the Automatically Generated Graph-based Evolutionary Algorithms (AG-GEAs) can be trained like a neural network style to advance candidate solutions. In AG-GEAs, key evolutionary operators such as tournament selection, exchange, crossover, and mutation—are modularized

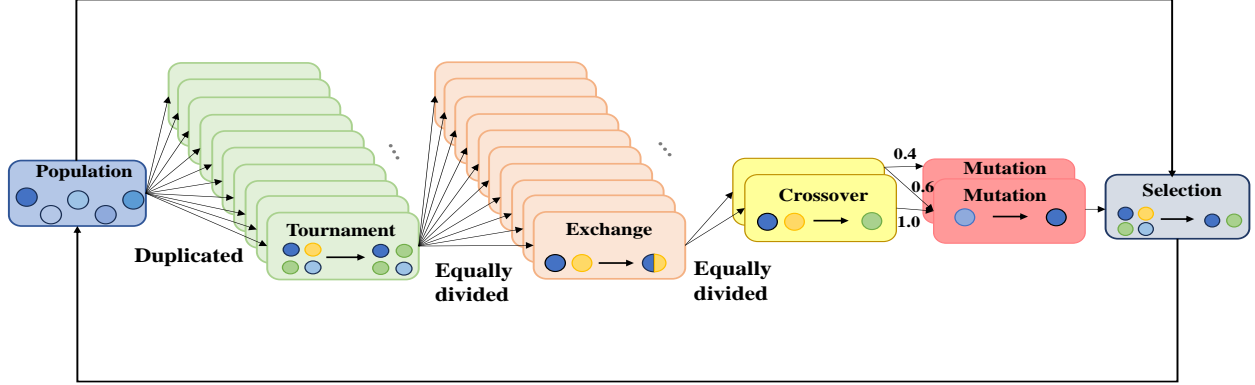


Fig. 1. Framework of the proposed automatically generated graph evolutionary algorithms (AG-GEAs)

into individual blocks. These blocks are then connected according to an adjacency matrix, forming flexible algorithm structures. By exploring different block connections, AG-GEAs can automatically discover optimal parameter settings and algorithm configurations that are well-suited to a wide category of problems.

## II. Proposed Algorithms

### A. Framework of AG-GEA

As depicted in Fig. 1, the AG-GEAs consist of six types of functional blocks, i.e. *Population*, *Tournament*, *Mutation*, and *Selection* connected in sequence. The candidate solutions are divided into several parts refer to the ratio from the front block to back and there are various number of blocks in parallel. The solutions in the *Population* block are duplicated and passed to all *Tournament* blocks, the solutions in each *Tournament* block are equally divided and passed to all *Exchange* blocks, the solutions in each *Exchange* block are equally divided and passed to all *Crossover* blocks, and the solutions in each *Crossover* block are divided with different ratio and passed to all *Mutation* blocks. Note that only the solutions passed from the *Mutation* blocks to the *Selection* block will be evaluated.

---

Algorithm 1: Algorithm Framework of the Proposed AG-GEA

---

Input:  $N$  (population size),  $K$  (K-Tournament Selection),  $nParent$  (Number of parents generated),  $nP_i$  (number of parents to generate a offspring,  $i=1,2$  denotes in Exchange, Crossover respectively),  $N^*$  (output population size)

Output:  $P^*$  (final population)

```

1  $P \leftarrow \text{Initialization}(N)$  ;
2 while termination criterion not fulfilled do
    //duplicated  $P$  into ten Tournament blocks
3    $[T_1, T_2, \dots, T_{10}] \leftarrow \text{Duplicate}(P)$ ;
    //Performing K-Tournament Selection on  $T_i, i = 1, \dots, 10$ 
4    $[P_1, P_2, \dots, P_{10}] \leftarrow \text{Tournament}(K, nParent, T_i)$ ;
5    $[P_{i1}, \dots, P_{i10}] \leftarrow \text{Divide}(P_i); i = 1, \dots, 10$ 
    //Performing  $i$ -th Exchange block,  $i = 1, \dots, 10$ 
6    $E_i \leftarrow \text{Exchange}(nP_1, [P_{1i} \cup \dots \cup P_{10i}])$ ;
7    $[E_{i1}, E_{i2}] \leftarrow \text{Divide}(E_i)$ ;
    //Performing  $j$ -th Crossover block,  $j = 1, 2$ 
8    $C_j \leftarrow \text{Crossover}(nP_2, [E_{1j} \cup \dots \cup E_{10j}])$ ;
9    $[C_{j1}, C_{j2}] \leftarrow \text{Divide}(C_j)$ ;
    //Performing Mutation block
10   $M_1 \leftarrow \text{Mutation}([C_{11} \cup C_{12}])$ ;
    //Performing Mutation block
11   $M_2 \leftarrow \text{Mutation}([C_{21} \cup C_{22}])$ ;
    //Do fitness sorting
12   $P^* \leftarrow \text{Selection}(N^*, [P \cup M_1 \cup M_2])$ ;
13 return  $P^*$ ;

```

---

## B. Translation, scale, and rotation invariant operator

A continuously differentiable operator  $h(\mathbf{x}^1, \mathbf{x}^2, \dots)$  is translation, scale, and rotation invariant operator (TSRI) [4] if and only if it has the following form:

$$h(x_k^1, x_k^2, \dots) = w_1 x_k^1 + w_2 x_k^2 + \dots \quad (1)$$

where  $x_k^1$  denotes the  $k$ -th variable of solution  $\mathbf{x}^1$  with  $k = 1, \dots, d$ , and  $w_1, w_2, \dots$  can be any constants with  $w_1 + w_2 + \dots = 1$ . For formula derivation and more details, please refer to the original paper: Principled design of translation, scale, and rotation invariant variation operators for metaheuristics.

### C. Details of blocks

Each block may include both hyperparameters and parameters. The hyperparameters are predefined manually, while the parameters can be automatically optimized through a training process. Blocks of the same type share identical hyperparameters but may differ in their parameter values. The specific functionality of each block is described as follows:

- *Population* : This block stores the solutions of each generation and outputs them upon termination. It does not require any hyperparameters or parameters.
- *Tournament* : Selecting  $n_t$  parents using m-tournament selection according to solutions' objective values (single-objective) or non-dominated front numbers (multi-objective), where  $n_t$  is a hyperparameter and  $m \in [1, 10]$  is a parameter.
- *Exchange* : A TSRI operator using  $n_e$  solutions to generate one offspring solution. When generating each variable of each offspring solution, only one of the  $w_1, w_2, \dots, w_{n_e}$  in Eq.(1) is 1 and the rest are 0, where the parameters being 1 are selected according to the probabilities

$$(p_1, p_2, \dots, p_{n_e}) \quad (2)$$

That is,  $n_e$  is the hyperparameter and  $p_1, p_2, \dots, p_n \in [0.1]$  are the parameters.

- *Crossover* : A TSRI operator using  $n_c$  solutions to generate one offspring solution. When generating each variable of each offspring solution, the  $w_2, \dots, w_{n_c}$  in Eq. (1) are sampled according to normal distributions and the  $w_1$  is directly set to  $1 - w_2 - \dots - w_{n_c}$ , where the normal distributions are determined by

$$\begin{pmatrix} \mu_{12}, \sigma_{12}, p_{12}, \dots, \mu_{1n_c}, \sigma_{1n_c}, p_{1n_c} \\ \mu_{22}, \sigma_{22}, p_{22}, \dots, \mu_{2n_c}, \sigma_{2n_c}, p_{2n_c} \\ \dots & \dots \\ \mu_{s_c 2}, \sigma_{s_c 2}, p_{s_c 2}, \dots, \mu_{s_c n_c}, \sigma_{s_c n_c}, p_{s_c n_c} \end{pmatrix}, \quad (3)$$

- *Mutation* : A TSRI operator using one solution  $x$  to generate one offspring solution  $o$ , where each offspring variable  $o_k$  is calculated by

$$o_k = x_k + w_0 \cdot (u_k - l_k) \quad (4)$$

where  $u_k$  is the upper bound and  $l_k$  is the lower bound. The upper and lower bounds can be regarded as special solutions, which implies that  $w_1 = 1$  and  $w_2 = -w_3 = w_0$  here. The  $w_0$  is sampled by normal distributions determined by

$$(\sigma_1, p_1, \dots, \sigma_{s_m}, p_{s_m}). \quad (5)$$

- *Selection* : Selecting N solutions according to solutions' objective values (single-objective) or the fitness defined in SPEA2 (multi-objective) [5]. It has one hyperparameter N denoting the population size and no parameter.

## References

- [1] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, pp. 341–359, 1997.
- [2] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the sixth international symposium on micro machine and human science*. Ieee, 1995, pp. 39–43.
- [3] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001.
- [4] Y. Tian, X. Zhang, C. He, K. C. Tan, and Y. Jin, "Principled design of translation, scale, and rotation invariant variation operators for metaheuristics," *Chinese Journal of Electronics*, vol. 32, no. 1, pp. 111–129, 2023.
- [5] E. Zitzler, "Spea2: Improving the strength pareto evolutionary algorithm," 2001.