Algorithm report for Block Evolutionary Algorithm

I. Introduction

Evolutionary algorithms(EAs) have become one of the most effective techniques for optimization problems, where a number of variation operators have been developed to handle the problems with various difficulties. While most EAs use a fixed operator all the time, it is a labor-intensive process to determine the best EA for a new problem. Therefore, it is particularly important to develop an evolutionary algorithm for automatic design algorithms.BlockEA encapsulates operators into blocks and generates new structures through the adjacency matrix between blocks as a new algorithm for solving different types of problems. The operators encapsulated into blocks include tournament selection, exchange operator, crossover operator, mutation operator, etc.different block structures can be used to obtain a set of optimal parameters and obtain the best algorithm to solve the problem.Furthermore, to boost BlockEA's convergence speed, a new differential evolution algorithm is connected during the latter stages of the optimization process.

II. Proposed Algorithm

A. Framework of BlockEA

The framework of BlockEA is given in Algorithm1 and Fig.1, which starts with the random initialization of a population with size N. There are three tournament selection blocks for parent selection, and the three parents are evenly divided into three parts and passed into the exchange operator blocks. Then, half of the populations of exchange operator1 and exchange operator2 are passed into the crossover operator1 for cross operation. All the populations output by exchange operator 3 are input into the crossover operator block. Finally, the populations obtained from the two crossover operators are selected for environment selection.The environment selection in BlockEA preserves the best N solutions.

Algorithm 1: Framework of the proposed BlockEA

Input: N (population size)

Output: S (final solution)

$P \leftarrow Initialization(N)$;

while *termination criterion not fulfilled* **do**

  $P_1 \leftarrow Tournament1(P)$;

  $P_2 \leftarrow Tournament2(P)$;

  $P_3 \leftarrow Tournament3(P)$;//Then divide P_i(i=1,2,3)into j subpopulation called P_{ij}(j=1,2,3)

  $P_4 \leftarrow Exchange1(P_{11} + P_{21} + P_{31})$;

  $P_5 \leftarrow Exchange2(P_{12} + P_{22} + P_{32})$;

  $P_6 \leftarrow Exchange3(P_{13} + P_{23} + P_{33})$;

  $P_7 \leftarrow Crossover1(P'_4 + P'_5)$;//$P'_i$ is half of P_i(i=4,5)

  $P_8 \leftarrow Crossover2(P_6)$;

  $P \leftarrow Selection(P_7 + P_8)$;

$[F_1, F_2, \ldots] \leftarrow$ Do fitness sorting on P; //F_i denotes the solution set in the i-th fitness sort

$S \leftarrow argmin|F_1 \bigcup \ldots \bigcup F_i|$;

return S;

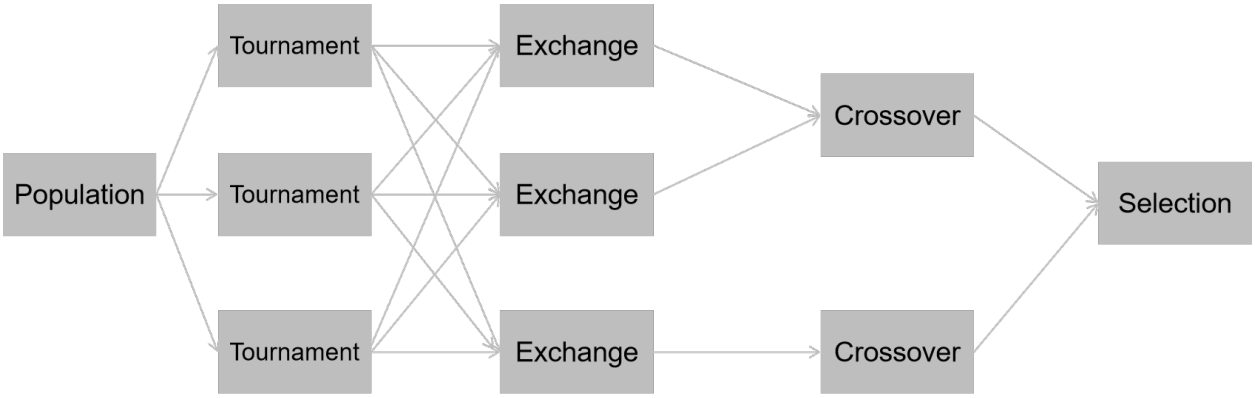

Fig. 1: Framework of BlockEA

B. Exchange Operators of BlockEA

The Block_Exchange is intricately designed to facilitate the parent exchange process in evolutionary algorithms. It efficiently manages parent selection by dynamically adjusting the probability of each parent's contribution to offspring based on their designated parameters. the block gathers decision variables from parent blocks and selectively samples parents according to the computed Fitness probabilities. The selection mechanism of parents is not based solely on ranking, but on their normalized fitness value as the selection probability. This not only helps to improve the fitness of the population, but also maintains the necessary genetic diversity.The procedure of Block_Exchange is given in Algorithm 2.

Algorithm 2: Block_Exchange

Input: P (population),N (population size),$nParents$ (Number of parents for generating an offspring)

Output: P' (final population)

$D \leftarrow$ Number of decision variables;

$K \leftarrow \frac{N}{nParents}$

$select \leftarrow K \times D$ zero matrix;

$select \leftarrow select + [\alpha_1 \dots \alpha_D]$;

$//\alpha_i$ `is` $[0 \dots K-1]^{\mathrm{T}}$

$select \leftarrow select + [\beta_1 \dots \beta_K]^{\mathrm{T}}$;

$//\beta_i$ `is` $[0 \dots D-1]$

$P' \leftarrow P(select)$

return P';

C. Crossover Operators of BlockEA

Block_Crossover implements a specialized form of the crossover operation in evolutionary algorithms. This operation is a core aspect of evolutionary strategies, where offspring are generated by combining genetic information from multiple parents. And provides a parameterized way to adjust the crossover process to better adapt to specific problem requirements.Unlike traditional two-parent crossover processes, involving multiple parents can provide a richer genetic pool. This approach helps enhance the diversity within the population, potentially leading to more robust solutions.The block manages the influence of each parent on the offspring through dynamic weights. These weights are adjusted in real-time based on current parameters, allowing the crossover operation to adapt flexibly to the state of the population.And the block employs a strategy of using cumulative sums and normalization to manage weights, ensuring that the crossover process remains effective and reasonable.The procedure of Block_Crossover is given in Algorithm 3.

Algorithm 3: Block_Crossover

Input: P (population),N (population size),$nParents$ (Number of parents for generating an offspring)

Output: P' (final population)

$D \leftarrow$ Number of decision variables;

$M \leftarrow N - \frac{N}{nParents}$

$R \leftarrow M \times D$ random matrix range [-1,1];

for $i = 1$ *to* $nParents - 1$ **do**

    R=w_1 * R + w_2; //w_1, w_2 is the assigned parameter

$K \leftarrow N \div nParents$;//Then divide R and P into K groups

$R_i \leftarrow$ [1-sum(R_i,1);R_i];//R_i denotes the i-th group in R

for $i = 1$ *to* K **do**

    $P_i \leftarrow P_i * R_i$;

    $P_i \leftarrow$ sum(P_i,1);

$P' \leftarrow P_1 \bigcup \ldots \bigcup P_K$;

Adjusting P' to within boundaries

return P';