An improved Decomposition-based Multi-Objective Evolutionary Algorithm for Network Architecture Search

1.Participant Information

Bingsen Wang ; Xiaofeng Han; Xiang Li; Tao Chao

School of Astronautics ,Harbin Institute of Technology, Harbin 150001, China

Email: 809214248@qq.com; hanx63520@gmail.com

2. Problem Analysis and Algorithm Description

2.1 Problem Analysis

The competition this year revolves around 15 unique customized instances in the CitySeg/MOP section of EvoXBench, each tailored for semantic segmentation of the Cityscapes dataset. As summarized in Table I, these instances vary in complexity, with each instance having between two and seven optimization objectives. These objectives cover a range of key performance metrics: f^e represents model segmentation error; f^c includes model complexity parameters such as floating-point operations (FLOPs, denoted as f_1^c and the number of model parameters or weights (f_2^c). Additionally, f^h focuses on hardware performance, capturing critical aspects such as latency (f_1^h), reference latency on a given hardware) and energy consumption (f_2^h), measured in joules on the corresponding hardware). These hardware-related objectives encompass h_1 for GPU performance and h_2 for edge device efficiency, thus covering a wide range of real-world application scenarios.

Table 1:Introduction of CitySegMOP Test Suit

Problem	D	M	Objectives
CitySeg/MOP1	32	2	$f^e, f_1^{h_1}$
CitySeg/MOP2	32	3	$f^e, f_1^{h_1}, f_1^c$
CitySeg/MOP3	32	3	$f^e, f_1^{h_1}, f_2^c$
CitySeg/MOP4	32	4	$f^e, f_1^{h_1}, f_2^{h_1}, f_1^c$
CitySeg/MOP5	32	5	$f^e, f_1^{h_1}, f_2^{h_1}, f_1^c, f_2^c$
CitySeg/MOP6	32	2	$f^e, f_1^{h_2}$
CitySeg/MOP7	32	3	$f^e, f_1^{h_2}, f_1^c$
CitySeg/MOP8	32	3	$f^e, f_1^{h_2}, f_2^c$
CitySeg/MOP9	32	4	$f^e, f_1^{h_2}, f_2^{h_2}, f_1^c$
CitySeg/MOP10	32	5	$f^e, f_1^{h_2}, f_2^{h_2}, f_1^c, f_2^c$
CitySeg/MOP11	32	3	$f^e, f_1^{h_1}, f_1^{h_2}$
CitySeg/MOP12	32	5	$f^e, f_1^{h_1}, f_1^{h_2}, f_2^{h_1}, f_2^{h_2}$
CitySeg/MOP13	32	6	$f^e, f_1^{h_1}, f_1^{h_2}, f_2^{h_1}, f_2^{h_2}, f_1^c$
CitySeg/MOP14	32	6	$f^e, f_1^{h_1}, f_1^{h_2}, f_2^{h_1}, f_2^{h_2}, f_2^c$
CitySeg/MOP15	32	7	$f^e, f_1^{h_1}, f_1^{h_2}, f_2^{h_1}, f_2^{h_2}, f_1^c, f_2^c$

Table 2:Objectives of CitySegMOP Test Suit

Objectives	Definition
f^e	prediction error
$f_1^{h_1}$	h_1's inference latency
$f_1^{h_2}$	h_2's inference latency
$f_2^{h_1}$	h_1's inference energy consumption
$f_2^{h_2}$	h_2's inference energy consumption
f_1^c	# of floating point operations
f_2^c	# of parameters/weights

Firstly, as shown in Table 2, among the seven objectives mentioned above, the evaluation of hardware-related objectives (i.e., inference speed on relevant hardware $f_1^{h_1}, f_1^{h_2}$, and energy consumption $f_2^{h_1}, f_2^{h_2}$ as well as model segmentation error (f_e) involves randomness. For hardware-related objectives, the evaluation values fluctuate within the range of $[1-\beta, 1+\beta]$ relative to the evaluation values obtained from the lookup table. For inference speed-related f_1 objectives, the random coefficient $\beta = 0.02$; for energy consumption-related f_2 objectives, the random coefficient $\beta = 0.05$. Due to the use of surrogate models for neural network performance prediction, the evaluation values of model segmentation error also exhibit uncertainty. Although the specific fluctuation range cannot be clearly identified, repeated evaluations of the same neural network architecture show that model segmentation error f_e has greater uncertainty than hardware-related objectives f^{h}.

Based on the analysis of evaluation uncertainties in the problem, this report proposes an adaptive noise algorithm to address the uncertainty assessment issue, specifically adopting the ε-dominance method to replace the conventional Pareto dominance. The ε-dominance introduces an additional parameter ε to relax the dominance definition, enhancing the robustness of dominance relationships. Moreover, in solving high-dimensional optimization problems, ε-dominance can increase environmental selection pressure, filter out well-performing solutions, and improve the algorithm's convergence and uniformity.

Furthermore, the aforementioned 15 CitySeg/MOP instances exhibit irregular Pareto fronts, typically characterized by highly nonlinear, degenerate, incomplete, large scale differences, and high-dimensional traits[3]. CitySeg/MOP instances often feature complex multimodal landscapes, with solutions very close to adaptive values in the objective space, which may represent vastly different architectures in the decision space, leading to highly nonlinear Pareto fronts. Conflicts may exist among different objectives; for instance, models with lower prediction errors typically have higher complexity and lower hardware performance, although conflicts may not necessarily arise between model complexity and hardware-related objectives, resulting in the degeneration of the Pareto front. The prediction errors of neural network architectures fall within the [0,1] range; however, due to objectives related to

complexity and hardware being formulated to represent different physical quantities, their target value scales differ significantly, thus exhibiting adverse scaling issues. CitySeg/MOP instances involve seven types of objectives, which in most cases lead to high-dimensional Pareto fronts. The irregular Pareto front affects the distribution of reference vectors and the selection of scalarization functions in the MOEAD algorithm, thereby reducing algorithm performance.

In response to this irregular Pareto front, this report employs the Growing Neural Gas (GNG) approach to guide decomposition-based multi-objective evolutionary algorithms. The GNG can learn the topology of the Pareto front. The GNG adaptively organizes a topological network through a set of nodes and edges. Edges connect nodes and represent their relationships. Known solutions are used as input signals, and as the population evolves, the network grows closer to reflecting the topology of the Pareto front. Nodes in the GNG represent positions on the Pareto front, and edges emanating from nodes provide curvature information for these positions. Therefore, they can be used to adapt reference vectors and scalarization functions, with each reference vector using a different scalarization function.

2.2 Algorithm Description

A. The algorithm IDEA_GNG provides a general framework for the algorithm.

Algorithm: IDEA_GNG

INPUT：P $(Population)$, N $(Population\ Size)$, A_S $(Input$ Signal Archive), N_S (Maximum Size of A_S), $G_{\max}$ (Maximum Generation)

1: $g = 0$;
2: $P = Initialize\ (P)$;
3: $z^* = Ideal\_Point\ (P)$;
4: $R = R_u = Uniform\_Reference\_Vector\_Generation\ (N)$;
5: $A_S = \emptyset$;
6: **while** the stopping criterion is not met, i.e., $g < G_{\max}$ **do**
7: $P' = Mating\_Selection\ (P)$;
8: $P'' = Reproduction\ (P')$;
9: $z^* = Ideal\_Point\ (P'' \cup \{z^*\})$;
10: **if** $g < (1-\alpha) \times G_{\max}$ **then**
11: $A_S = Input\_Signal\_Archive\_Update(A_S \cup P'', R, N_S, z^*)$;
12: $[R_{\text{node}}, V_{\text{edge}}] = GNG\_Update(A_S)$;
13: $R = Reference\_Vector\_Adaptation\ (R_u, R_{\text{node}}, A_S)$;
14: $F^S = Scalarizing\_Function\_Adaptation\ (R, V_{\text{edge}})$;
15: **end if**
16: $P = Environmental\_Selection(P \cup P'', R, F^S, N, z^*)$;
17: $g = g + 1$;
18: **end while**

Output: P

Lines 1-5 in Algorithm are the initialization phase. $g = 0$ is the count value of generation (line 1). The population P is randomly initialized in the decision space (line 2). Function $Ideal\_Point$ returns the minimum objective values as the estimated ideal point $z^* = (z_1^*, \dots, z_M^*)$ (line 3).R_u is a set of uniformly distributed reference

vectors whose size is equal (or similar) to the population size N. R is the reference vectors to be used in selection (line 4). The input signal archive A_S with the maximum size N_S is initialized to be an empty set (line 5).

In the DEA-based optimization model[2], there are three operations iterated until the stopping criterion defined by the maximum number of generations $G_{\max}$ is satisfied (line 6)

1) Parents P' are selected by mating selection (line 7).
2) Offspring P'' are generated by a reproduction operator(line 8)
3) New population is obtained by environmental selection (line 16).

In addition, z^* is updated once the offspring are produced (line 9). We will describe environmental and mating selection in Section III-B.

There are four operators in the GNG-based learning model:

1) input signal archive update (line 11);
2) GNG update (line 12);
3) reference vector adaptation (line 13);
4) scalarizing function adaptation (line 14).

Note that these operators are not employed in the last $\alpha \times G_{\max}$ generations (line 10). This is because that we expect the evolution to focus on exploitation by keeping the reference vectors and the scalarizing functions unchanged.

B. The main steps of GNG

In Section 2.2.A, lines 11-14 of the algorithm framework are all about the GNG-based learning model. GNG is able to learn the topological structure in a given set of input signals (vectors) adaptively. The network of GNG consists of: 1) a set of nodes which accumulate input signals and 2) a set of edges among pairs of nodes which represent the relationships among connected nodes. In this paper, they are denoted as R_{node} and V_{edge}, respectively. The main steps of GNG are given as follows[1]:

Step 0: (Initialization) start with two connected nodes at random positions r_1 and r_2. Note that the age of a new generated edge and the error variable of a new generated node are 0.

Step 1: Input a signal ξ.

Step 2: Find the nearest node r_a and the second-nearest node r_b to ξ.

Step 3: Increment the age of all edges emanating from r_a.

Step 4: Increase the error variable of r_a by adding the squared distance of r_a and ξ

$$\text{error}(\mathbf{r}_a) = \text{error}(\mathbf{r}_a) + \| \mathbf{r}_a - \xi \|^2.$$

Step 5: Move r_a and its direct topological neighbors, $\mathbf{r}_k^{nb}, k = 1, \dots, K$,(i.e., the nodes connected with $\mathbf{r}_a$ by an edge) toward ξ by learning rates ϵ_a and ϵ_{nb}, respectively, of the total distance

$$r_a = r_a + \epsilon_a(\xi - r_a)$$
$$r_k^{nb} = r_k^{nb} + \epsilon_{nb}\left(\xi - r_k^{nb}\right).$$

Step 6: If $\mathbf{r}_a$ and $\mathbf{r}_b$ are connected by an edge, set the age of this edge to zero. If such an edge does not exist, create it.

Step 7: Remove edges with an age larger than $\text{age}_{\max}$. If this results in nodes having no emanating edges, remove them as well.

Step 8: If the number of signals input so far is a multiple of a parameter λ and the number of nodes has not reached to maximum, insert a new node as follows.

1) Determine the node $r_1^{\max}$ with the largest error variable.

2) Insert a new node r^{new} halfway between $r_1^{\max}$ and its neighbor $r_2^{\max}$ with the largest error variable

$$r^{\text{new}} = 0.5(r_1^{\max} + r_2^{\max}).$$

3) Insert edges connecting r^{new} with $r_1^{\max}$ and $r_2^{\max}$, and remove the original edge between $r_1^{\max}$ and $r_2^{\max}$.

4) Decrease the error variables of $\mathbf{r}_1^{\max}$ by multiplying them by a constant α. Set the error variable of r^{new} to be the same as that of $r_1^{\max}$.

Step 9: Decrease all error variables by multiplying them by a constant δ.

Step 10: If a stopping criterion (e.g., reaching the maximum number of iterations) is not yet fulfilled, go to step 1.

C. ε-dominance

ε-dominance is a method that relaxes the definition of dominance in multi-objective optimization problems. In traditional Pareto dominance, a solution is said to dominate another if it is superior in all objectives. However, in ε-dominance, a solution is still considered dominant even if it is slightly inferior in some objectives. Specifically, solution A is said to ε-dominate solution B if A is not worse than B by more than ε in all objectives and is better than B by more than ε in at least one objective. This approach can increase the diversity of solutions and aid in exploring a broader search space in multi-objective optimization.

Let $\varepsilon \in \mathbb{R}^+$ be a positive real number. We define that an objective vector u ε-dominates v, denoted by $u \succcurlyeq_\varepsilon v$,precisely if $(1+\varepsilon) \cdot u_i \geq v_i$ for all $i \in \{1, \ldots, m\}$.

In order to explore the distribution pattern of objective function values with evaluation uncertainty, we randomly selected 10 solutions and repeated the evaluation of the neural network architectures represented by these solutions 1000 times to investigate the statistical characteristics of these noisy objectives. We calculated the coefficient of variation (CV) for each target dimension of each data group according to the formula $CV = \frac{\mu}{\sigma} \times 100\%$. Here, σ represents the standard deviation, and μ represents the mean. The obtained values are referred to as the coefficient of variation (CV), used to measure the relative dispersion of data. A smaller coefficient of

variation indicates a relatively low data dispersion and greater data stability, while a higher coefficient of variation suggests a relatively high data dispersion and greater data volatility.

Subsequently, we calculated the average of the 10 data sets for each target dimension, resulting in the final coefficient of variation for each target dimension as shown in Table 3.

Table 3 : CV of Each Objectives

$Objectives$	f_e	$f_1^{h_1}, f_1^{h_2}$	$f_2^{h_1}, f_2^{h_2}$
CV	5.6841%	1.1924%	2.9155%

Using the coefficient of variation directly as ε would result in excessively lenient dominance conditions, leading to the inclusion of some poorly performing solutions in the Pareto front, thus affecting algorithm performance. The final selection of ε was appropriately reduced, as shown in Table 4.

Table4 : ε of Each Objectives

$Objectives$	f_e	$f_1^{h_1}, f_1^{h_2}$	$f_2^{h_1}, f_2^{h_2}$
ε	0.03	0.005	0.015

3. EXPERIMENTAL RESULT

All experiments in this report are implemented in the PlatEMO4.6, Pymoo 0.6.1.1 and EvoXBench 1.0.5. Table 5 presents the HV values obtained by IDEA-GNG on CitySeg/MOP test suites.

Table5：HV Values Obtained by IDEA_GNG on CitySeg/MOP

Problem	M	D	HV
CitySeg/MOP1	2		8.9886e-1 (1.00e-3)
CitySeg/MOP2	3		7.9892e-1 (1.38e-3)
CitySeg/MOP3	3		8.2245e-1 (3.37e-3)
CitySeg/MOP4	4		6.9670e-1 (1.99e-3)
CitySeg/MOP5	5		6.5537e-1 (1.17e-3)
CitySeg/MOP6	2		7.7193e-1 (2.79e-4)
CitySeg/MOP7	3		7.3107e-1 (3.48e-4)
CitySeg/MOP8	3	32	7.3223e-1 (4.75e-4)
CitySeg/MOP9	4		5.7641e-1 (5.57e-4)
CitySeg/MOP10	5		5.4687e-1 (6.71e-4)
CitySeg/MOP11	3		6.8526e-1 (5.26e-3)
CitySeg/MOP12	5		4.7139e-1 (2.24e-3)
CitySeg/MOP13	6		4.2117e-1 (1.98e-3)
CitySeg/MOP14	6		4.3261e-1 (2.43e-3)
CitySeg/MOP15	7		3.9851e-1 (5.63e-4)

REFERENCE

[1] B. Fritzke, “A growing neural gas network learns topologies,” in Proc. Adv. Neural Inf. Process. Syst., 1995, pp. 625–632.

[2] Liu Y, Ishibuchi H, Masuyama N, et al. Adapting reference vectors and scalarizing functions by growing neural gas to handle irregular Pareto fronts[J]. IEEE Transactions on Evolutionary Computation, 2019, 24(3): 439-453.

[3] Lu Z, Cheng R, ** Y, et al. Neural architecture search as multiobjective optimization benchmarks: Problem formulation and performance assessment[J]. IEEE transactions on evolutionary computation, 2023.