An application of Preference-Inspired Co-Evolutionary Algorithm to Sectorization

Elif Öztürk[3,5][0000−0002−8407−1871], Pedro Rocha[3][0000−0002−2871−5070], Filipe Sousa[1][0000−0001−9349−3454], Margarida Lima[2][0000−0003−2231−4978], Ana M. Rodrigues[2,3][0000−0003−1070−3626], José Soeiro Ferreira[3,4][0000−0002−7552−9924], Ana C. Nunes[1,6][0000−0002−4025−1985], Cristina Lopes[2][0000−0002−4833−470X], and Cristina Oliveira[2][0000−0002−8407−1871]

[1] ISCTE - University Institute of Lisbon, Lisbon, Portugal
[2] CEOS.PP, ISCAP, Porto, Portugal
[3] INESCTEC - Technology and Science, Porto, Portugal
[4] FEUP - Faculty of Engineering, University of Porto, Porto, Portugal
[5] FEP.UP - Faculty of Economics, University of Porto, Porto, Portugal
[6] CMAFcIO - Faculty of Sciences, University of Lisbon, Lisbon, Portugal

Abstract. Sectorization problems have significant challenges arising from the many objectives that must be optimised simultaneously. Several methods exist to deal with these many-objective optimisation problems, but each has its limitations. This paper analyses an application of Preference Inspired Co-Evolutionary Algorithms, with goal vectors (PICEA-g) to sectorization problems. The method is tested on instances of different size difficulty levels and various configurations for mutation rate and population number. The main purpose is to find the best configuration for PICEA-g to solve sectorization problems. Performance metrics are used to evaluate these configurations regarding the solutions' spread, convergence, and diversity in the solution space. Several test trials showed that big and medium-sized instances perform better with low mutation rates and large population sizes. The opposite is valid for the small size instances.

Keywords: Sectorization Problems, Co-Evolutionary Algorithms, Many-Objective Optimisation

1 Introduction

Sectorization, the division of a whole – region, network, area – into subsets, usually appears in real-life situations, such as school/health districting, maintenance operations, political districting or design of sales territories. These are multi-objective problems since it is common to wish for balanced, compact or connected sectors.

Multi-objective optimisation problems (MOP) require optimising several conflicting objectives simultaneously. Various algorithms have been developed and among the most well known are MOEA, MOGA, NSGA-II, and SPEA, which

work well for up to 3 objectives (being NSGA-II the most adopted method to solve MOPs, and also used by many authors to deal with sectorization and related problems [21,3]). However, more than 3 objectives causes the performance of these algorithms to degrade [7].

The problems containing four or more objectives are called many-objective problems (MaOP) and emerge as a particular case of MOPs. MaOPs have harder challenges and require significantly more effort into the solution strategy [13]. The deterioration of MOP methods performance arises due to the following difficulties when many objectives are included: decreasing search capacity of Pareto dominance, increasing complexity of the approximation to the Pareto Front (PF), and the complications in the solution visualisation [7]. The evaluation of the solutions fitness is done through an important concept known as Pareto Dominance, which enables classifying solutions as dominated or non-dominated. However, in some situations, the entire solution set may be non-dominated, collecting in the same Pareto frontier. This reduces the search abilities of Pareto dominance, which is already very challenging due to the dimensionality of the objective space increasing in proportion to the number of objectives. When this happens, the hyper-surface of the PF gets larger, and the number of solutions to approximate the entire PF extensively increases. Ultimately, decision-making becomes harder, as the visualisation of solutions is complicated by many objectives. Hence, the algorithms developed to solve MaOPs try to erase these difficulties where MOP algorithms stand weak.

In the literature, multiple techniques deal with these challenges. The present paper overviews them and selects a Co-Evolutionary Algorithm to solve sectorization problems, called Preference-Inspired Co-Evolutionary Algorithms, with goal vectors (PICEA-g). This method uses preferences to lead the solutions in the solution space to obtain more desirable solutions and facilitate decision-making. Since sectorization problems can involve many conflicting objectives, PICEA-g emerged as a promising exploration path. The current work contains the first application of the method for sectorization problems and offers preliminary results on its performance. Therefore, it constitutes a relevant contribution to the sectorization literature.

The remainder of the paper is structured as follows. Section 2 presents the literature review about methods proposed to cope with MaOPs. Section 3 includes the framework of PICEA-g as well as the genetic operators and objectives selected. Section 4 shows the results and critically discusses PICEA-g performance in different instances and configurations. The conclusions are in Section 5.

2 Literature Review

Some techniques are proposed in the literature that enhance the performance of well-known MOP algorithms when many objectives are in question. For instance, to improve Pareto-dominance evaluation, the following modifications are presented: (i) the use of modified dominance instead of Pareto dominance to reduce the number of non-dominated solutions, such as ϵ-dominance [5], α-domination

[6], (ii) the introduction of different ranks to non-dominated solutions to create higher selection pressure in the PF [8], and (iii) the use of different performance evaluation mechanisms than Pareto dominance. These mechanisms can be fit in two main groups: indicator-based and scalarising function-based. The former uses an indicator function to measure the quality of the solutions. The best known indicator based algorithm is Hypervolume Estimation (HypE) [1]. HypE employs Monte-Carlo simulation to measure the exact hypervolume values used as an indicator to evaluate the solutions. On the other hand, the latter evaluates the fitness through a scalarising function, such as weighted sum or weighted Tchebycheff (Chebyshev). The most well-known scalar function-based algorithm is Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D) [22] which decomposes the problem within scalar sub-problems and optimises them together. Moreover, NSGA-III can be included in the same category [2] where the predefined reference points are distributed to the objective space to keep a diverse solution set and help them converge.

A proposed alternative to improve the PF approximation uses preference-based procedures [4,17]. Priory integration of the preferences into the algorithm reduces the search in the objective space, by concentrating in a more representative sub-space, increasing the chance of finding improved solutions. The concept of simultaneous evolution of the candidate solutions regarding the preferences in practice is called preference inspired algorithms. The preference points used in these algorithms are randomly generated and are only used to increase the selection pressure of the candidate solutions [15].

Purshouse et al. [14] presented preference-inspired evolutionary algorithms that used target objective vectors as preference solutions (PICEA). The candidate solutions are then co-evolved according to their dominance on the preference solutions. Soon after, Wang et al. [18] proposed the idea of goal vectors to lead the candidate solutions, and called it a preference-inspired evolutionary algorithm with goals (PICEA-g). If a candidate solution dominates more goal vectors (while fewer candidate solutions dominate those goal vectors) it has a higher chance to proceed to the next generation [10]. PICEA-g is compared with several MaOP algorithms, and its superiority is shown in [18].

In the literature, it is likely to find some applications of PICEA-g addressing real-world problems that show it may be modified and adapted for a new application on a specific problem, as seen in the dynamic districting and routing problem by Lei et al. [9]. The authors implemented the method with a minor modification by mating neighbouring solutions to improve the offsprings during the coevolution. Moreover, Long et al. [11] implemented PICEA-g for multi-period location routing problem by integrating the Tchebycheff method to decompose the objective space while improving the diversity of the solutions. It is also possible to find some papers on workflow scheduling [10,12] with modifications of the original PICEA-g, appropriate to their specific problems. In all these applications, PICEA-g gave promising results and exemplary performance in MaOPs. For this reason, this paper focuses on this method to tackle current challenges in sectorization problems.

3 Approximation Method: PICEA-g

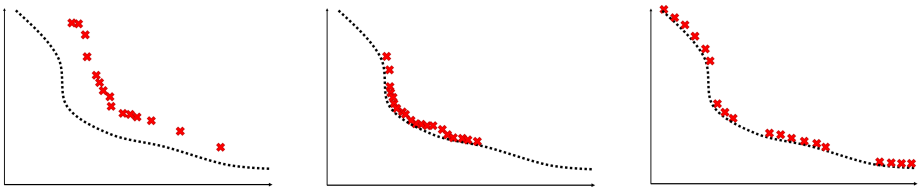
This section explains the PICEA-g method step by step, following procedures and equations based on Wang et al. [18].

The generation of the full PF is a challenging problem due to existing limitations (convergence efficiency and computational cost), so an adequate representation is desired. A good representation of the PF requires having a sufficient number of solutions that provide ample coverage across its length while in close proximity, but maintaining the solutions with a certain degree of dispersion. When these requirements are not met, as in Fig. 1, the representation quality of the PF is significantly reduced.

In Fig. 1a the solutions are shown as not converging to the PF, where close proximity is desired. Alternatively, Fig. 1b shows a situation where the solutions converged to a specific region, very close to the PF, being all very close to one another. This leads to almost redundant solutions, with a very small distinction between each adjacent solution, where having solutions with higher distinction, or separation, between the adjacent ones is desired. Finally, in Fig.1c the solutions are in close proximity to the PF and dispersed enough so that few redundant solutions exist, but insufficient coverage of the full PF as seen by the empty gaps along with it. These complications arise due to, among other factors, the maximum size of the population, the total number of generations, and the guiding procedure that pushes the solutions into new unexplored regions of the solution space. Tweaking the size of population and number of generations will not overcome these challenges due to heavy computational constraints.

An efficient approach to guide the solutions can dramatically improve the convergence efficiency to the PF, generating better quality solutions and reducing the cost of dealing with tradeoffs between solution quality, solution variety, and computational cost. This convergence process can be improved using the PICEA-g algorithm, which produces goal vectors that enable guiding the generation of new solutions into different regions of the solution space, closer to the PF, and into its regions without solution representation.

The PICEA-g uses the evolution process that is common in all Evolutionary Algorithms, that start with an initial solution population of size N defined by $S(t)$ (where t is the generation index), and through the use of a crossover and



(a) No convergence (b) No dispersion (c) No representation

Fig. 1: The different problems can appear in the PF convergence

mutation operation produces offspring $S_c(t)$ that are evaluated using the fitness function F_s (in Eq. 1, using the goal vectors of PICEA-g method) which then allows filtering the full population (parent S and offspring S_c) using a truncation operation, producing a new solution population of size N, defined as $S(t+1)$. This cycle is repeated until stopping criteria is met, usually a specific number of generations. A diagram with this process is exemplified in Fig. 2.

$$F_s = 0 + \sum_{g \in G \uplus G_c | s \preceq g} \frac{1}{n_g} \tag{1}$$

The PICEA-g algorithm generates new goal vectors, defined as $G_c(t)$ using random selection. The new goal vectors G_c are added to the population of existing goal vectors G, evaluated using the fitness function F_g (in Eq. 2) and truncated considering their fitness and population size. The fitness evaluation procedure follows a dominance metric where the solutions population is compared against the preference population, and vice-versa. This process happens in the Evaluation block of Fig. 2.

$$F_g = \frac{1}{1+\alpha}, \qquad \alpha = \begin{cases} 1, & n_g = 0 \\ \frac{n_g - 1}{2N-1}, & \text{otherwise} \end{cases} \tag{2}$$

Depending on the goal vectors' distance to the ideal PF and their location on the solution space, their usefulness as a mean for comparability between different solutions varies significantly. As specified on [19], goal vectors closer to the PF help the convergence of solutions to it, and depending on the solution space region where the goal vectors are located, they push the solutions to cover new sections of the PF. Using a random generation process for the goal vectors is a reasonable starting approach (since it does not have any specific optimisation that facilitates the convergence of the solution population to the PF) that produces a baseline performance that allows improvements in the goal vectors generation process to be compared to.

The fitness of a given preference (F_g) is given by the expression defined in F_g shown in Eq.2, which attributes the fitness value for each one of the goal vectors.

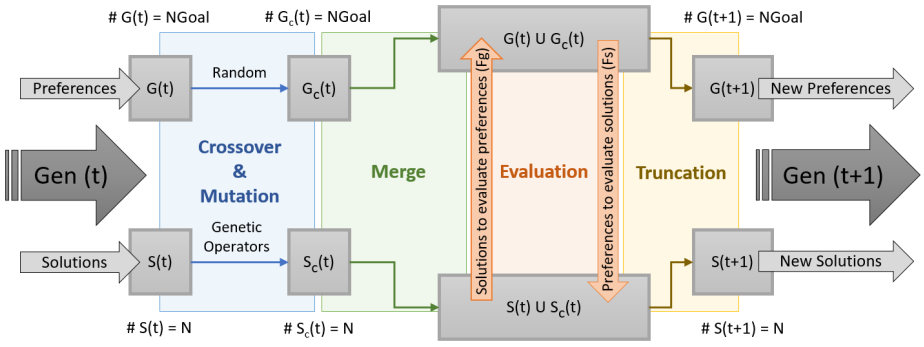


Fig. 2: Solution and preference population evolution process in PICEA-g.

The parameters N and n_g represent, respectively, the maximum population size, and the number of solutions that satisfy preference g. By penalising the goal vectors that are satisfied by a large number of solutions, this process creates an incentive to eliminate them from the population. Goals that are satisfied (dominated) by fewer solutions are given high fitness value, which remain on the solution space, pushing the generation of new solutions that try to satisfy (dominate) them.

The fitness of a given solution (F_s) is dependent on the aggregated quality of all individual preferences g it satisfies. This fitness expression, seen in Eq.1, states that the fitness for a given s is the sum of the relative quality of each solution that it satisfies (as in $1/n_g$, where n_g is the number of solutions that satisfy preference g). If s does not satisfy any g, its fitness value is 0. A preference g that satisfies many solutions provides a very small contribution to the individual fitness of each one, but a preference g that satisfies only a few contributes significantly more. This creates the incentive to generate solutions outside of the solution space region where most other solutions are concentrated. This constant interaction between both populations (solutions and preferences) is the basis for the PICEA-g approach.

3.1 Genetic Operators

PICEA-g is based on evolutionary algorithms composed of specific operators, such as encoding scheme, crossover and mutation.

In the encoding scheme, the solutions were encoded using a 'matrix form binary grouping' (MFBG) genetic encoding system. MFBG is a binary matrix where the rows represent the total number of basic units, while the columns represent the sectors. The feasibility of MFBG requires: (i) a basic unit cannot be assigned more than one sector, and (ii) each sector must have at least one basic unit, and cannot be empty. Thus, the sum of each row must be one, and the sum of each column must be one or more.

The crossover operator is based on a multi-point crossover, where it randomly selects multiple rows from the two-parent solutions and switches them to generate two offspring solutions. This eliminates the need to set a crossover probability due to the advantageous design of the genetic encoding system used. Although simple, this method also improves the diversity in the following generations due to the random selection of multiple points.

The mutation operator is applied to every offspring population with a given probability in order to provide some randomness to the population. The mutation is implemented row by row in a whole chromosome by assigning the basic unit to another sector when the mutation value exceeds the defined probability threshold.

3.2 Objectives

Three objectives commonly used in sectorization problems are considered in the solution method: Equilibrium, Compactness and Contiguity.

Equilibrium is the balance between sectors. This indicator can refer to multiple sector characteristics, such as balance in demand, balance in workload or working hours, etc. Considering the current approach, the equilibrium was defined as the deviation from the mean, adopted from [16] and is shown in Eq.3 and Eq.4.

$$\bar{q} = \frac{\sum_{j=1}^{J} \sum_{i} x_{ij} \times y_i}{J} \tag{3}$$

In Eq. 3, $\bar{q}$ represents the mean demand of each sector, where J is the total number of sectors, x_{ij} represents if the basic unit i is assigned to sector j and y_i is its corresponding demand. Eq. 4 refers to the deviation from the mean demand of the sectors. Here, q_j is the sum of the basic units in sector j. A better equilibrium level represents a smaller standard deviation value from the mean.

$$std'_{eq} = \sqrt{\frac{1}{J-1} \sum_{j=1}^{J} (q_j - \bar{q})^2} \tag{4}$$

Compactness refers to density in each sector. That is quite a relevant objective for most sectorization problems, especially if they are further concerned about routing or travelling. Eq. 5 shows the mathematical representation of the compactness in this study. Here d gives the compactness level of the chromosome or solution. It is the sum of the distances between the centroid o_j and the furthest point to the centroid p_j in each sector j. The smaller the d gets, the more compact the sectors are.

$$d = \sum_{j=1}^{J} dist(o_j, p_j) \tag{5}$$

Contiguity indicates the connectivity of a sector. It is a common objective in sectorization problems that evaluates the flexibility of moving from one basic unit to another within the sectors. The measure for this objective is also adopted from [16]. The authors represent contiguity through a square matrix of size equal to the number of basic units, and set the value 1 to all feasible paths between two nodes in the same sector, 0 otherwise. Eq. 6 represents the contiguity in sector j. Here, the nominator is the sum of the links in a sector, and the denominator is the maximum number of links if all the basic units are linked. Thus, c_j takes the value 1 when the sector is fully connected.

$$c_j = \frac{\sum_{i=1}^{i_j} (\sum_{w=1}^{i_j} m_{wi}^j)}{i_j(i_j - 1)}, \qquad m_{wi}^j = \begin{cases} 1 & \text{if path between w and i in sector j} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The contiguity of the chromosome ($\bar{c}$) is calculated by the formula represented in Eq. 7. Here, the numerator is the sum of sector base contiguity calculated in Eq. 6, and the denominator is the total number of basic units I. Value $\bar{c}$ varies between 0 and 1, being 1 the best, and 0 worst. In order to evaluate all objectives as minimisation, it is refactored as $(1-\bar{c})$ while also switching the best and worst limit values.

$$\bar{c} = \frac{\sum_{j=1}^{k} c_j i_j}{I} \tag{7}$$

4 Results and Discussion

The performance of PICEA-g is tested using the 3 specified objectives over 10 different sectorization instances, each with a different size and difficulty. These instances are generated following gamma distribution, where shape and scale parameters are created randomly for each basic unit produced. The demand for each basic unit is created following a uniform distribution. Finally, connected graph theory is considered while the links between the basic units are generated. The aim is to find a pattern that shows the best parameter composition for each instance type. For that, 3 performance metrics are used, namely, Error Ratio (ER), Inverted Generational Distance (IGD) and Spacing (S).

ER is a cardinality metric and measures the proportion of the solutions on the approximation PF over the population size. IGD is a convergence and distribution metric. The distance of the approximation PF from the reference PF and the distribution of the solutions in the objective space are estimated by IGD. Finally, S is a spread metric that measures the deviations of the distances between the solutions in the approximation PF. All these metrics are assumed better when they have lower values. These metrics were adopted from Yen and He [20].

Table 1 shows the instances characteristics. The configuration parameters that were selected consisted on a combination of mutation rates (using 0.00, 0.02, 0.04, 0.06, 0.08 and 0.10) and population size (with 50, 60, 70, 80, 90, 100), which were evolved for a total number of 1000 generations. $Name$ refers to the Gamma (γ) instances set. $Nodes$ represent the number of basic units to be sectorized, while $Sectors$ are the sector number.

Each instance was tested with every combination of population size and mutation rate through PICEA-g, being run 20 times each. In total, every instance is tested 720 times for 36 configurations with 20 trials each. It is possible to obtain the instances and observe the results for all instances from the following link: https://drive.inesctec.pt/s/EQn6yCD3jdap3TW.

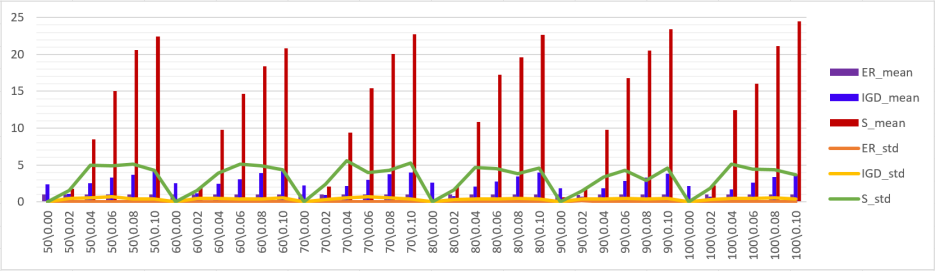
According to the results presented in the link, large and middle-size instances performed better regarding selected performance metrics when the mutation rate was lower. However, smaller instances produced better results with higher mutation rate. This behaviour is not unexpected, since the large instance solutions set shows a higher degree of population diversity compared to the smaller ones and manages to converge to the PF successfully when no significant disturbances (mutations) are present. With high mutation probability, this convergence reduces or even stops. On the other hand, in the smaller instances, the diversity in the initial solution set is more constrained. In this situation, a higher mutation

Table 1: Instances characteristics

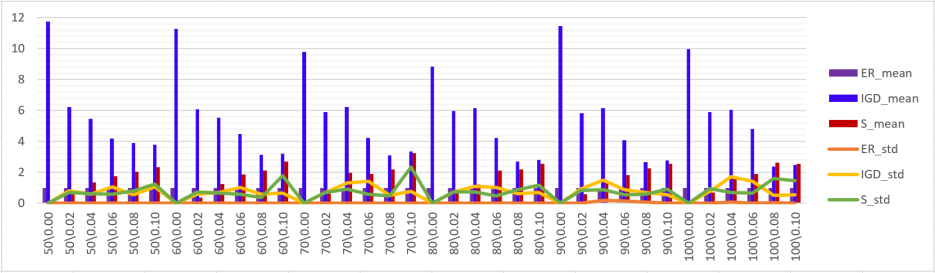
$Name$	$\gamma 2$	$\gamma 3$	$\gamma 8$	$\gamma 9$	$\gamma 10$	$\gamma 11$	$\gamma 14$	$\gamma 18$	$\gamma 28$	$\gamma 49$
$Nodes$	690	56	873	432	102	288	204	528	350	1000
$Sectors$	30	5	30	10	10	10	10	30	10	30

probability compensated this limitation, which helps creating disturbances that increase the chances of finding better solutions.

In the remainder of the section, we focused on the performance of two instances as a representative examples of our experiments. Fig. 3a and 3b show the performance of the two instances, $\gamma 3$ and $\gamma 8$ respectively, on different metrics. We selected these instances to show how PICEA-g behaves in small and big instances. The bars show the mean value of the performance metrics for 20 runs, and the lines are the standard deviations (std) from the mean for each configuration.



(a) 873 nodes



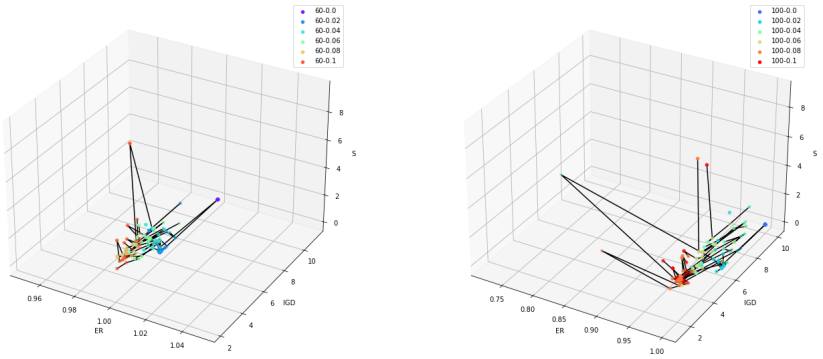
(b) 56 nodes

Fig. 3: The performance of two instances for selected performance metrics

Fig. 3a shows that when the mutation rate is higher, the performance of IGD and S get worse for the same population size, although ER remains in the same range. Thus, the solutions' convergence, distribution, and spread are better with the lower mutation rate applied to the algorithm. On the other hand, Fig. 3b shows that higher mutation rates improve the performance of IGD while worsening the performance of S. ER again appears to be in similar ranges for different configurations. The IGD is a convergence and distribution metric. It is possible to say that the convergence of the solutions is better with a higher mutation probability, although the spread of the solutions on the PF decreases. This result shows that the solutions may converge well but not cover the objective space sufficiently.

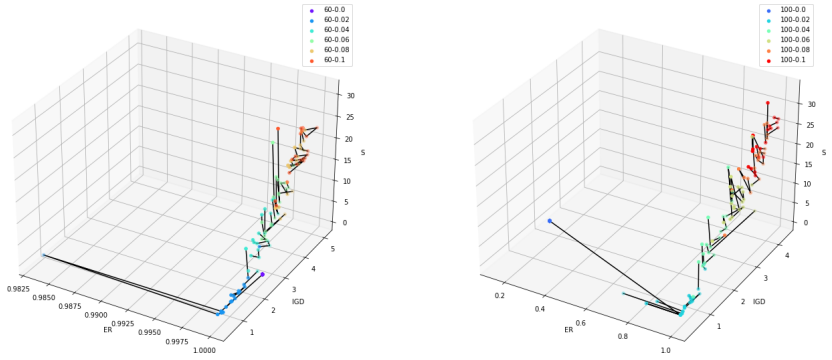
Moreover, Fig.4 and 5 shows the performance of selected metrics over each other. The Pareto dominance concept is used for this comparison. In other words, after performance metrics are measured for each configuration considering each run, they are located on Pareto frontiers according to their dominance for three selected metrics. As seen, for $\gamma 3$ with 56 nodes the sequence of the Pareto frontiers show that the solutions perform better with higher mutation probability. On the other hand, the lower mutation probability results better in the $\gamma 8$ with 873 nodes. Unfortunately, the produced results were not sufficient to clearly identify the best parameters for each instance type, requiring further research.

The algorithm is implemented through Python 3.9.6, and the method is executed on a PC Intel Xeon gold 6148 @ 2.4GHz, 20 cores, 40 threads, 96Gb ram and Win X64 operating system. The computation time for the PICEA-g algorithm was 3.5 seconds per generation for a 100 population size.



(a) Population size 60 (b) Population size 100

Fig. 4: The performance of selected metrics over each other: case of $\gamma 3$



(a) Population size 60 (b) Population size 100

Fig. 5: The performance of selected metrics over each other: case of $\gamma 8$

5 Conclusion

This work presented the first application of the PICEA-g to Sectorization problems. The current configuration used three performance metrics for the evaluation: ER, IGD and S. The instances were selected based on their small, medium and large sizes (number of basic units) and distinct features (connectivity and number of sectors), producing different difficulty levels. In order to estimate the performance of PICEA-g on different sectorization problems, a combination of diverse configuration parameters was selected (mutation rates, maximum population sizes, total number of generations).

According to the preliminary results obtained, smaller instances show better performance when higher mutation rates are selected, while larger instances show better performance with lower mutation rates. A possible explanation for this behaviour is that the low degree of distinction between solutions (individuals) in smaller instances leads them to become stuck on a local minimum, which requires a significant amount of disturbance (mutation) to change. On the other hand, the high degree of distinction of the solutions in larger instances has difficulties converging to a neighbouring local minimum solution if the mutation rate is very high, showing better performance with a smaller rate of disturbance.

Albeit these are preliminary results, the experiments revealed that the PICEA-g is a promising method to solve sectorization and related problems. More precise insights require further analysis with different configuration parameters and instance types. Thus, future work consists of improving the parametrisation of the method depending on the instance type, improving performance, and comparing with other methods usually referenced in the sectorization literature.

Acknowledgements

This work is financed by the ERDF - European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project 'POCI-01-0145-FEDER-031671'.

References

1. Bader, J., Zitzler, E.: Hype: An algorithm for fast hypervolume-based many-objective optimization. Evolutionary computation 19(1), 45–76 (2011)
2. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. IEEE transactions on evolutionary computation 18, 577–601 (2013)
3. Farughi, H., Mostafayi, S., Arkat, J.: Healthcare districting optimization using gray wolf optimizer and ant lion optimizer algorithms. Journal of Optimization in Industrial Engineering 12(1), 119–131 (2019)
4. Fonseca, C.M., Fleming, P.J.: Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 28, 26–37 (1998)

5. Hernández-Díaz, A.G., Santana-Quintero, L.V., Coello, C.A.C., Molina, J.: Pareto-adaptive -dominance. Evolutionary computation 15(4), 493–517 (2007)
6. Ikeda, K., Kita, H., Kobayashi, S.: Failure of pareto-based moeas: Does non-dominated really mean near to optimal? In: Proceedings of the 2001 congress on evolutionary computation. vol. 2, pp. 957–962. IEEE (2001)
7. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: A short review. In: 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence). pp. 2419–2426. IEEE (2008)
8. Kukkonen, S., Lampinen, J.: Ranking-dominance and many-objective optimization. In: Congress on Evolutionary Computation. pp. 3983–3990. IEEE (2007)
9. Lei, H., Wang, R., Laporte, G.: Solving a multi-objective dynamic stochastic districting and routing problem with a co-evolutionary algorithm. Computers & Operations Research 67, 12–24 (2016)
10. Lei, H., Wang, R., Zhang, T., Liu, Y., Zha, Y.: A multi-objective co-evolutionary algorithm for energy-efficient scheduling on a green data center. Computers & Operations Research 75, 103–117 (2016)
11. Long, S., Zhang, D., Liang, Y., Li, S., Chen, W.: Robust optimization of the multi-objective multi-period location-routing problem for epidemic logistics system with uncertain demand. IEEE Access 9, 151912–151930 (2021)
12. Paknejad, P., Khorsand, R., Ramezanpour, M.: Chaotic improved picea-g-based multi-objective optimization for workflow scheduling in cloud environment. Future Generation Computer Systems 117, 12–28 (2021)
13. Purshouse, R.C., Fleming, P.J.: Evolutionary many-objective optimisation: An exploratory analysis. In: The 2003 Congress on Evolutionary Computation, 2003. CEC'03. vol. 3, pp. 2066–2073. IEEE (2003)
14. Purshouse, R.C., Jalbă, C., Fleming, P.J.: Preference-driven co-evolutionary algorithms show promise for many-objective optimisation. In: International Conference on Evolutionary Multi-Criterion Optimization. pp. 136–150. Springer (2011)
15. Qiu, Q., Yu, W., Wang, L., Chen, H., Pan, X.: Preference-inspired coevolutionary algorithm based on differentiated resource allocation strategy. IEEE Access 8, 205798–205813 (2020)
16. Rodrigues, A.M., Ferreira, J.S.: Measures in sectorization problems. In: Operations research and big data, pp. 203–211. Springer (2015)
17. Thiele, L., Miettinen, K., Korhonen, P.J., Molina, J.: A preference-based evolutionary algorithm for multi-objective optimization. Evolutionary computation 17(3), 411–436 (2009)
18. Wang, R., Purshouse, R.C., Fleming, P.J.: Preference-inspired coevolutionary algorithms for many-objective optimization. IEEE Transactions on Evolutionary Computation 17(4), 474–494 (2012)
19. Wang, R., Purshouse, R.C., Fleming, P.J.: Preference-inspired co-evolutionary algorithm using adaptively generated goal vectors. In: 2013 IEEE Congress on Evolutionary Computation. pp. 916–923 (2013)
20. Yen, G.G., He, Z.: Performance metric ensemble for multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation 18, 131–144 (2014)
21. Zhang, K., Yan, H., Zeng, H., Xin, K., Tao, T.: A practical multi-objective optimization sectorization method for water distribution network. Science of The Total Environment 656, 1401–1412 (2019)
22. Zhang, Q., Li, H.: MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on evolutionary computation 11, 712–731 (2007)