The Effect of Offspring Population Size on NSGA-II: A Preliminary Study

Max Hort
University College London
London, United Kingdom
max.hort.19@ucl.ac.uk

Federica Sarro
University College London
London, United Kingdom
f.sarro@ucl.ac.uk

ABSTRACT

Non-Dominated Sorting Genetic Algorithm (NSGA-II) is one of the most popular Multi-Objective Evolutionary Algorithms (MOEA) and has been applied to a large range of problems.

Previous studies have shown that parameter tuning can improve NSGA-II performance. However, the tuning of the offspring population size, which guides the exploration-exploitation trade-off in NSGA-II, has been overlooked so far. Previous work has generally used the population size as the default offspring population size for NSGA-II.

We therefore investigate the impact of offspring population size on the performance of NSGA-II. We carry out an empirical study by comparing the effectiveness of three configurations vs. the default NSGA-II configuration on six optimization problems based on four Pareto front quality indicators and statistical tests.

Our findings show that the performance of NSGA-II can be improved by reducing the offspring population size and in turn increasing the number of generations. This leads to similar or statistically significant better results than those obtained by using the default NSGA-II configuration in 92% of the experiments performed.

CCS CONCEPTS

• **Computing methodologies** → **Genetic algorithms**.

KEYWORDS

Genetic algorithms, multi-objective optimization, NSGA-II, offspring population

ACM Reference Format:
Max Hort and Federica Sarro. 2021. The Effect of Offspring Population Size on NSGA-II: A Preliminary Study. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459479

1 INTRODUCTION

Evolutionary Algorithms (EA) have shown a wide applicability to a broad range of problems. One reason for the popularity of EAs is their ability to adapt to particular problems by tuning their parameters [3]. Finding optimal choices for parameters, such as population size and mutation rate, is difficult and significantly changes the behavior of EAs and their achieved performance (fitness landscape) [3, 5].

Genetic algorithms (GA) [6] are the most popular type of EA. Generally speaking, GAs perform global search by emulating biological evolution. When implementing GAs, developers have to make important decisions, including the population size, offspring population size, and mutation rate [3]. The choice of parameters has a a critical impact on algorithmic performance.

The offspring population size plays an important role in guiding the exploration-exploitation trade-off for GAs [3]. While the parent population determines the focus of the GA within the search space, the size of the offspring population determines the rate of exploration performed. A high offspring population size allows for the exploration of a larger search space surrounding the parent population. However, increasing the size of the offspring population size is accompanied by an increasing amount of fitness evaluations on the offspring population, which effects the runtime of GAs.

This raises the question whether population size and offspring population size in GAs should remain coupled. In particular, we are interested in the effect of offspring population size on the performance of Multi-Objective Evolutionary Algorithms (MOEAs). Among those, we are interested in NSGA-II [4], which is one of the most popular MOEAs. Just as in canonical GAs, the offspring population size in NSGA-II is equal to the population size by default, and previous work has not considered offspring population size when tuning NSGA-II parameters (see e.g., [9, 10, 16]). We therefore investigate the tuning of offspring population size for NSGA-II as explained in the following.

2 EXPERIMENTAL DESIGN

In this section, we outline the design of the experiments we carry out to investigate the effect of offspring population size on the performance of NSGA-II. We present the research question as well as the settings and the problems used in our experiments.

Research Question. To evaluate the effect of offspring population size on the performance of NSGA-II, we answer the following research question: *RQ: How does offspring population size influence the performance of NSGA-II?*. For this purpose we adapt the offspring population size of NSGA-II, while maintaining an identical amount of fitness computations (e.g., reduce offspring population size and increase number of generations).

Computational search. NSGA-II [4] is a well-known MOEA based on Pareto-optimality. NSGA-II can be considered as an extension of GAs for multiple objective function optimization. The fitness of an individual is determined for each objective and they are ranked based on Pareto fronts and crowding distance. NSGA-II

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459479

proceeds with an initial population P of size N. Based on P, an offspring population Q of identical size is generated, using selection, crossover and mutation procedures.

Performance Metrics We selected four Pareto Front quality indicators that are suitable for theoretical problems with known Pareto fronts [8]. In particular, we consider: Generational Distance (GD), Generational Distance Plus (GD+), Inverted Generational Distance (IGD), Inverted Generational Distance Plus (IGD+) [7].

Test Problems. We investigate six test problems in our experiments, which have been investigated in the past and are available in PYMOO, a framework for Multi-Objective Optimization (MOO) in Python [2]: ZDT1-6. [18]. The selected six problems are MOO problems with two objective functions.

Settings. For each of the NSGA-II configurations investigated in our study (see Section 3) we set a parent population of size 100, a budget of 25,000 fitness evaluations, a crossover probability of 0.9 and a mutation probability of 1/n (where n is the number of decision variables), according to Deb et al. [4]. We used the NSGA-II implementation provided in pymoo (version 0.4.2.2) [2].

Validation and Evaluation Method. To evaluate the effect of offspring population size on the offspring ratio, we perform experiments on six test problems. These experiments apply different NSGA-II configurations and are repeated 100 times (each with different random seeds), to account for randomness [1]. We average results achieved across multiple runs. We then use statistical significance tests to assess performance achieved with different offspring population sizes and summarise the results following a win-tie-loss procedure as done in previous work [11, 12, 14]. For this purpose, we use the Wilcoxon Signed-Rank test [17] ($\alpha < 0.01$), which is a non-parametric test that makes no assumption about underlying data distribution as done in previous work [13, 15].

3 RESULTS

To answer our research question, we compute the performance of different configurations for NSGA-II. In particular, we compare the default configuration (offspring population size = 100, generations = 250) against three different configurations which change offspring population size and perform 25,500 fitness evaluations: NSGA-II$_{O_{50}G_{500}}$ (offspring population size = 50, generations = 500), NSGA-II$_{O_{150}G_{166}}$ (offspring population size = 150, generations = 166), NSGA-II$_{O_{200}G_{125}}$ (offspring population size = 200, generations = 125).

Each pair of configurations is compared on six problems and four performance metrics, resulting in 24 comparisons. For each comparison, we determine if one of the two configurations is statistically better than the other, according to the Wilcoxon Signed-Rank test. Table 1 summarises these results.

NSGA-II$_{O_{50}G_{500}}$ is the only configuration able to outperform the default one. It is statistically better than the default one in 12/24 comparisons and worse in only 2/24. The remaining 10/24 comparisons result in a tie as we cannot conclude whether the differences are statistically significant.

4 CONCLUSIONS

Our findings show that the performance of NSGA-II, which generally sets the offspring population size equal to the population size (i.e., by default both are set to 100), can be improved by reducing the size of the offspring population. Such a reduction allows for more generations and ultimately a better performance than using the default configuration.

We therefore hope that in the future offspring population sizing attracts more attention than previously, such that it will be included in parameter tuning.

	Default	NSGA-II$_{O_{50}G_{500}}$	NSGA-II$_{O_{150}G_{166}}$	NSGA-II$_{O_{200}G_{125}}$
Default	-	12-10-2	1-9-14	4-2-18
NSGA-II$_{O_{50}G_{500}}$	2-10-12	-	4-0-20	3-1-20
NSGA-II$_{O_{150}G_{166}}$	14-9-1	20-0-4	-	2-9-13
NSGA-II$_{O_{200}G_{125}}$	18-2-4	20-1-3	13-9-2	-

Table 1: Win-tie-loss summary of the Wilcoxon tests comparing performance (GD, GD+, IGD, IGD+) by each pair of methods (columns vs. rows).

ACKNOWLEDGMENTS

M. Hort and F. Sarro are supported by the ERC grant 741278 (EPIC).

REFERENCES

[1] A. Arcuri and L. Briand. 2014. A Hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *STVR* 24, 3 (2014), 219–250.

[2] J. Blank and K. Deb. 2020. Pymoo: Multi-Objective Optimization in Python. *IEEE Access* 8 (2020), 89497–89509.

[3] K. De Jong. 2007. Parameter setting in EAs: a 30 year perspective. In *Parameter setting in Evolutionary Algorithms*. Springer, 1–18.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE TEVC* 6, 2 (2002), 182–197.

[5] B. Doerr, C. Gießen, C. Witt, and J. Yang. 2019. The (1+\lambda) Evolutionary Algorithm with Self-Adjusting Mutation Rate. *Algorithmica* 81, 2 (2019), 593–631.

[6] D. E. Goldenberg. 1989. Genetic algorithms in search, optimization and machine learning.

[7] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima. 2015. Modified distance calculation in generational distance and inverted generational distance. In *International Conference on Evolutionary Multi-criterion Optimization*. Springer, 110–125.

[8] M. Li, T. Chen, and X. Yao. 2020. How to Evaluate Solutions in Pareto-based Search-Based Software Engineering? A Critical Review and Methodological Guidance. *IEEE TSE* (2020). https://doi.org/10.1109/TSE.2020.3036108

[9] S. Ramesh, S. Kannan, and S. Baskar. 2012. Application of modified NSGA-II algorithm to multi-objective reactive power planning. *Applied Soft Computing* 12, 2 (2012), 741–753.

[10] J. Sadeghi, S. Sadeghi, and S. Taghi A. Niaki. 2014. A hybrid vendor managed inventory and redundancy allocation optimization problem in supply chain management: An NSGA-II with tuned parameters. *Computers & Operations Research* 41 (2014), 53–64.

[11] F. Sarro, F. Ferrucci, M. Harman, A. Manna, and J. Ren. 2017. Adaptive Multi-Objective Evolutionary Algorithms for Overtime Planning in Software Projects. *IEEE TSE* 43, 10 (2017), 898–917.

[12] F. Sarro, M. Harman, Y. Jia, and Y. Zhang. 2018. Customer rating reactions can be predicted purely using app features. In *IEEE International Requirements Engineering Conference*. 76–87.

[13] F. Sarro, R. Moussa, A. Petrozziello, and M. Harman. 2020. Learning From Mistakes: Machine Learning Enhanced Human Expert Effort Estimates. *IEEE TSE* (2020). https://doi.org/10.1109/TSE.2020.3040793

[14] F. Sarro and A. Petrozziello. 2018. Linear Programming As a Baseline for Software Effort Estimation. *ACM TOSEM* 27, 3 (2018), 12:1–12:28.

[15] F. Sarro, A. Petrozziello, and M. Harman. 2016. Multi-objective software effort estimation. In *IEEE International Conference on Software Engineering*. 619–630.

[16] A. S. Sayyad, K. Goseva-Popstojanova, T. Menzies, and H. Ammar. 2013. On parameter tuning in search based software engineering: A replicated empirical study. In *IEEE International Workshop on Replication in Empirical Software Engineering Research*. 84–90.

[17] F. Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics*. Springer, 196–202.

[18] E. Zitzler, K. Deb, and L. Thiele. 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation* 8, 2 (2000), 173–195.