On the configuration of multi-objective evolutionary algorithms for PLA design optimization

Willian M. Freire, Simone de F. Tonhão, Tiago P. Bonetti, Marcelo Y. Shigenaga, William de A. Cadette, Fernando dos S. Felizardo, Aline M. M. M. Amaral, Edson OliveiraJr, Thelma E. Colanzi
Informatics Department, State University of Maringá (UEM)
Maringá, Paraná, Brazil
{willianmarquesfreire,siimone.franca,tiagopiperno,marcelo.shigenaga,williamakdt}@gmail.com,fsf50@hotmail.com
ammmamaral@uem.br,edson@din.uem.br,thelma@din.uem.br

ABSTRACT

Search-based algorithms have been successfully applied in the Product Line Architecture (PLA) optimization using the seminal approach called Multi-Objective Approach for Product-Line Architecture Design (MOA4PLA). This approach produces a set of alternative PLA designs intending to improve the different factors being optimized. Currently, the MOA4PLA uses the NSGA-II algorithm , a multi-objective evolutionary algorithm (MOEA) that can optimize several architectural properties simultaneously. Despite the promising results, studying the best values for the algorithm parameters is essential to obtain even better results. This is also crucial to ease the adoption of MOA4PLA by newcomers or non-expert companies willing to start using search-based software engineering to PLA design. Three crossover operators for the PLA design optimization were proposed recently. However, reference values for parameters have not been defined for PLA design optimization using crossover operators. In this context, the objective of this work is conducting an experimental study to discover which are the most effective crossover operators and the best values to configure the MOEA parameters, such as population size, number of generations, and mutation and crossover rates. A quantitative analysis based on quality indicators and statistical tests was performed using four PLA designs to determine the most suitable parameter values to the search-based algorithm. Empirical results pointed out the best combination of crossover operators and the most suitable values to configure MOA4PLA.

CCS CONCEPTS

• **Software and its engineering → Software product lines**; **Software architectures**; **Search-based software engineering**; • **Computing methodologies → Search methodologies**.

KEYWORDS

Multi-objective evolutionary algorithm, software product line, software architecture, recombination operators

SBCARS '21, September 27-October 1, 2021, Joinville, Brazil

ACM ISBN 978-1-4503-8419-3/21/09…$15.00
https://doi.org/10.1145/3483899.3483905

ACM Reference Format:
Willian M. Freire, Simone de F. Tonhão, Tiago P. Bonetti, Marcelo Y. Shigenaga, William de A. Cadette, Fernando dos S. Felizardo, Aline M. M. M. Amaral, Edson OliveiraJr, Thelma E. Colanzi. 2021. On the configuration of multi-objective evolutionary algorithms for PLA design optimization. In *15th Brazilian Symposium on Software Components, Architectures, and Reuse (SBCARS '21), September 27-October 1, 2021, Joinville, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3483899.3483905

1 INTRODUCTION

Software Product Line (SPL) consists of techniques, methods and tools that aim the development of similar systems to maximize software reuse, and minimize costs through the artifacts reuse, focusing on fast software delivery [10]. An SPL shares a common set of features and components satisfying the specific needs of a particular market segment, which can be developed throughout a generic architectural design.

Designing a Product Line Architecture (PLA) is a complex task, as the architect must consider different and conflicting metrics, such as extensibility, modularity, and reusability. Specifically, feature modularization is an important characteristic of PLA design, as it can directly influence extensibility and reusability [21]. As various factors can influence the PLA design, there is not a single, ideal solution, leading to several possibilities for modelling a PLA [6].

In this context, the PLA design can be modeled as a multi-objective optimization problem [15]. Search Based Software Engineering (SBSE) is a research field that allows converting a software engineering problem into a computational search problem, which can be solved with metaheuristics, to automatically find near-optimal solutions [16]. Genetic algorithms are a widely used metaheuristic in the SBSE field. These algorithms are based on the analogy with the processes of natural selection and evolutionary genetics [14]. The purpose of these algorithms is to copy nature considering the adaptation and survival of the fittest. They include search operators of three types: selection, crossover and mutation [14]. All metaheuristics, including genetic algorithms, are stochastic algorithms. Therefore, there is a randomness factor associated with the application of search operators.

MOA4PLA (*Multi-Objective Approach for Product-Line Architecture Design*) [6] applies SBSE techniques to optimize several architectural properties of a PLA design simultaneously. This approach was implemented in OPLA-Tool [13] using the NSGA-II algorithm (Non-dominated Sorting Genetic Algorithm II) [9], the most used

Multi-Objective Evolutionary Algorithm (MOEA) in SBSE studies [4]. NSGA-II is a multi-objective algorithm as it allows the simultaneous optimization of more than one objective (factor). In the context of PLA design optimization, the objectives are a compound of software metrics related to the architectural properties to be optimized [6]. As NSGA-II is an extension of genetic algorithms, it applies selection, crossover and mutation operators. MOA4PLA has crossover and mutation operators specific for PLA design, which were implemented in OPLA-Tool to be used together with NSGA-II.

To achieve the best results from NSGA-II, it is imperative to configure the main parameters of the algorithm, namely: the application rates for mutation and crossover operators, the population size and the number of generations (or number of fitness evaluations). In general, these values vary according to both (i) the problem under optimization and (ii) the characteristics of the individuals provided as input to the optimization process [14]. As MOEAs are stochastic algorithms, it is recommended the execution of several runs and the comparison of the possible algorithm's configurations using quality indicators [1, 2].

Previous studies [2, 3] pointed out that the canonical values to configure the main parameters of NSGA-II for software engineering problems are not the most appropriate ones. Thus, the values to configure NSGA-II for PLA design were investigated in a previous study [22], when only mutation operators were available in OPLA-Tool. As a result, the best values for mutation rate, population size and number of generations were found.

More recently, three crossover operators were proposed for the PLA design optimization [7, 8]. However, reference values have not been investigated for PLA design optimization using crossover operators. Preliminary results pointed out that we do not need to apply the three operators at the same time [7]. However, the best combination of the crossover operators has not been studied yet.

In this context, we conducted an exploratory study with a twofold goal: (i) investigating what is the best combination of crossover operators for search-based PLA design, and (ii) discovering the best values to configure the MOEA parameters, such as population size, number of generations, and mutation and crossover rates. To accomplish such goals, we carried out experiments with four PLA using six different configurations for NSGA-II using OPLA-Tool. To define each configuration we varied the pair of crossover operators and the values to configure the algorithm's parameters. A quantitative analysis based on quality indicators and statistical tests was performed to determine the most suitable parameter values and the best combination of crossover operators to configure MOA4PLA.

In summary, our contributions are as follows: (i) experimental evidence revealed that the most effective configuration is: population with 100 individuals, 300 generations, crossover and mutation rates equal to 0.4 and 0.8, respectively; (ii) experimental evidence also revealed that the joint application of the *Feature-driven Crossover* and *Modular Crossover* operators is more beneficial for PLA design optimization, corroborating preliminary results [7]; (iii) we increased the state of the art by finding the best values for the algorithm parameters to PLA optimization, what enables to obtain even better results; and, (iv) knowing the best configuration for MOA4PLA is crucial to ease the adoption of search-based PLA design by newcomers or non-expert companies.

This paper is organized in sections. Section 2 presents the main concepts involved in this work. Section 3.3 addresses the experiment design. The experiment results are presented and discussed in Section 4 and the research questions are answered in Section 5. In Section 6 some guidelines related to the parameter configuration are presented. Finally, Section 7 addresses the threats to validity, whereas Section 8 concludes the paper and points for future works.

2 PLA DESIGN OPTIMIZATION

PLA design is influenced by several factors, such as feature modularization, reusability, variability, and PLA extensibility. Some of these factors may be in conflict, leading to various modelling possibilities for a specific PLA design. SBSE techniques have achieved satisfactory results to discover near-optimal solutions for problems like PLA design [15]. When search-based algorithms are used to optimize PLAs, metrics to evaluate those factors are used in objective functions that guide the search process. Each solution obtained in the search is evaluated and ranked by the objective functions. So those functions consist of the objectives to be optimized. At the end of the search, several alternative designs may be automatically achieved by the algorithms according to the selected objectives.

Relying on these ideas, the MOA4PLA [6] is an approach that aims to automate the search for better PLA designs using MOEAs. This approach allows the use of different MOEAs in the optimization of PLAs. The first step to apply MOEAs in PLA optimization is instantiating an initial solution, from the original PLA design, to be used by the algorithm as the starting point of the optimization process. In MOA4PLA such an instantiation is called *PLA representation*. Software metrics, related to the factors that influence PLA design, are used as objectives to evaluate the solutions obtained during the optimization process. For doing so, MOA4PLA has a set of *objective functions* to evaluate the generated solutions. During the optimization process, *search operators* are applied on the PLA representations over a lot of generations, generating as output a set of PLA design alternatives with the best trade-off among the optimized objectives.

PLA representation. In MOA4PLA, the PLA is a class diagram that depicts the static structural aspect of the architecture [6]. It is represented using a metamodel, which allows the handling of architectural elements by the search operators. A PLA contains architectural elements such as components, interfaces, operations and their inter-relationships. Each architectural element is associated with feature(s) by using UML stereotypes. Each element can be common to all SPL products or variables, being present only in some product(s). Variable elements are associated with variabilities that have variation points and their variants. A XMI (XML Metadata Interchange) file containing the original PLA design is the input to our approach. During the XMI processing to generate the PLA representation, each identified XMI element is instantiated as an object of the metamodel according to its type. So, the PLA representation will contain objects to represent the architectural elements, their relationships, variabilities and features associated with each architectural element [5].

Objective Functions. The evaluation model [26] of MOA4PLA encompasses several objective functions based on software metrics,

which provide indicators about feature modularization, PLA extensibility, variability, design elegance, coupling, cohesion and size. The entire evaluation model is presented in [26]. The three objective functions used in our study are FM (Feature Modularization), COE (Relational Cohesion), and ACLASS (Class Coupling). *FM* provides an indicator for feature modularization of a PLA design based on the sum of the metrics for feature-driven cohesion, and feature scattering and feature interlacing over architectural elements. The *COE* objective function evaluates the cohesion of PLA design in terms of the internal relationship of the classes of the PLA design. *ACLASS* measures class coupling by the number of architectural elements that depend on other classes of the design, added to the number of elements on which each class depends. The equations of the objective functions and their metrics are presented in [26]. In MOA4PLA all objective functions must be minimized. Thus, the lower the fitness value, the better.

Search Operators. MOA4PLA uses mutation and crossover operators to optimize the PLA design provided as input. The mutation operators are also used to create the initial population from the original PLA design. Currently, seven mutation operators are available, namely Move Method, Move Attribute, Add Class, Move Operation, Add Manager Class, Feature-driven Operator, and Feature-driven Operator for Class. *Move Attribute* and *Move Method* [6] move an attribute and a method to another class, respectively. *Add Class* [6] moves a method or an attribute to a new class. *Move Operation* [6] moves operations between interfaces. *Add Manager Class* [6] creates an interface which is realized by a new manager class, and then moves an operation to this interface. *Feature-driven Operator* [6] focuses on the PLA feature modularization by modularizing a feature tangled with others in a component. Finally, *Feature-driven Operator For Class* [20] modularizes pieces of the PLA design where two or more features are realized by the same class. For the previous operators, the relationships between source and target architectural elements are properly added.

The crossover operation is used to generate offspring by exchanging values in a pair of parent solutions chosen from the population. There are three crossover operators specific for PLA optimization. *Feature-driven Crossover* (FdC) [8] is concerned with maintaining and improving feature modularization of a PLA design. This operator selects a feature at random and then creates children by swapping the architectural elements that realize the selected feature. *Complementary Crossover* (CC) [8] aims to create offspring from parents that are complementary in different good fitness values. For doing this, at the beginning of a generation cycle, all individuals are sorted in ascending order based on their objective fitness values. For each objective to be optimized, one list is created to store information related to a particular objective. Then, two lists are randomly selected. Each parent is picked from one list, preferably selecting the best ones for each objective. The crossover point is determined only at one parent. To generate offspring, the child receives the part that precedes the crossover point of the first parent. Then, it receives elements from the other parent that are not yet in the child [8]. *Modular Crossover* (MC) [7] focuses on preserving allocations of complete or partial modules (packages or components) from a parent to a child. The rationale is that if a module in a PLA may already be well modularized, then the child will likely benefit from having this package as it is. Preliminary results [7, 8] revealed that the application of FdC together with MC or CC is more beneficial for PLA design optimization since FdC impacts on feature modularization, what is a very important property for PLA design, whereas the other crossover operators improve other architectural properties.

OPLA-Tool [13] is the open source tool that automates the application of MOA4PLA. It allows the usage of different multi-objective algorithms to optimize PLA designs, such as NSGA-II [9], which was used in this study. Using this tool, the user can analyze the set of solutions generated after the optimization process. Implementations for the existing crossover operators were added in the latest version of this tool.

3 RESEARCH METHOD

This section addresses the research method in terms of the study design (Section 3.3), the PLA designs used in the study, and the quality indicators and statistical tests used to evaluate the results.

3.1 PLAs used in the empirical study

Table 1 presents the numbers of components, classes, features and variabilities of the PLAs used in the experiments.

Table 1: Numbers of Architectural Elements of the PLAs

PLA	Components	Interfaces	Classes	Variabilities	Features
AGM1	9	14	20	4	9
AGM2	9	14	21	5	11
MM1	11	16	13	7	13
MM2	8	13	10	7	13

Arcade Game Maker (AGM) [23] is a SPL created by the Software Engineering Institute (SEI) that encompasses three arcade games: Brickles, Bowling, and Pong. AGM1 is presented in [18] and AGM2 [5] was obtained from AGM1 by adding two new features: ranking and logging. These features aim at maintaining information about both the best ranking for each game available in a AGM product and the users that achieve the rankings.

Mobile Media (MM) [18] is a SPL composed of features that handle music, videos, and photo for portable devices. It provides support for managing different types of media. MM1 was developed in [18] and MM-2 [5] is equivalent to the release R8 presented in [11]. MM1 and MM2 are equivalent in terms of features but MM2 has a lower number of architectural elements.

3.2 Quality Indicators and Statistical Tests

The results of the MOEAs are quantitatively evaluated using quality indicators and appropriate statistical analysis. Quality indicators allow analysis of the convergence and diversity of the algorithms regarding the Pareto front. As it is hard to find the optimal Pareto Front, we define the reference front (PF_{true}) as the set of all non-dominated solutions obtained by all algorithms in all runs as it is commonly done in practice [28]. In each run, an algorithm generates a set of solutions that is an approximation to PF_{true}, which is called PF_{approx}. The front formed by all non-dominated solutions obtained by each algorithm in all runs is called PF_{known}.

The quality indicators used in this work were Hypervolume (HV) and Euclidean Distance to the ideal solution (ED). HV measures the area that a Pareto front covers on the objectives space. The higher the HV value, the greater the coverage area, reflecting a better front. When working with a minimization problem, a reference point greater than the worst solution achieved should be defined [28]. The ED measures the distance between a solution *n* and the ideal solution in the solutions space [27]. Considering that this work involves a minimization problem, the ideal solution has the minimum value obtained for each objective. Solutions with low ED are better because they have the best trade-off between the optimized objectives [27].

The statistical tests used were Shapiro–Wilk, Kruskal-Wallis and Vargha-Delaney. The Shapiro–Wilk [24] is used to investigate if the sample sets have normal distribution. Based on the *p-value* calculus and the confidence level defined, the null hypothesis is evaluated. Thus, if the *p-value* is less than the chosen confidence level, then the null hypothesis is rejected and there is evidence that the data tested are not normally distributed. On the other hand, if the *p-value* is greater than the chosen confidence level, then the null hypothesis can not be rejected. In the context of this work, we used a confidence level of 95%.

Another statistical test used in this work is the Kruskal-Wallis [19]. This test is a non-parametric method for testing whether samples originate from the same distribution. It is used for comparing two or more independent samples of equal or different sample sizes. A significant Kruskal–Wallis test indicates that at least one sample stochastically dominates one other sample. The test does not identify where this stochastic dominance occurs or for how many pairs of groups stochastic dominance obtains.

Finally, we also used the Vargha-Delaney test. This test, through A-statistic, is a measure of effect size [25]. This test evaluates the probability of an aleatory value selected from a sample 1 being numerically greater than other aleatory values of a sample 2. It is possible, using this measure to compare the performance of two algorithms, and to quantify how much better one algorithm is than the other, in a statistically-sound fashion.

3.3 Design of the Experiment

This section describes the design of the experiment, which **aims to** analyze different combinations of crossover operators and parameter values for PLA optimization, **with the purpose of** characterizing the best configuration, **from the point of view of** PLA architects, **in the context of** software engineering Master's students and researchers of the State University of Maringa.

As mentioned in Section 2, the PLA solutions obtained throughout an optimization process are evaluated based on the values of the objective functions (fitness of the solutions). Such functions express the quality of generated solutions in relation to specific metrics for the PLA context.

In this sense, the objects of this exploratory study are the PLA alternatives resulting from the optimization process. The experimental package containing all the objects of analysis, as well as the graphs, spreadsheets, and measuring instruments are organized in the Experimental Package [12].

To achieve our goal we defined two research questions for this exploratory study:

RQ1: What is the best combination of crossover operators for search-based PLA design? Preliminary results [7] indicate the crossover operators MC and CC lead to better results when combined with the FdC operator. Therefore, it is important to investigate which pair of crossover operators – FdC + CC or FdC + MC – contributes to achieving better results;

RQ2: What is the best parameter configuration for the PLA optimization process? In this RQ we investigate the most appropriate values for the parameters: population size, number of generations, and application rates for mutation and crossover operators. Knowing these values is important as the application of crossover operators together with mutation operators impacts on the performance and convergence of the search-based algorithms. In our context, the performance of a configuration refers to how effective the configuration is in obtaining high-quality solutions in terms of the values of the objective functions.

3.3.1 Planning. **Instrumentation:** we used two variations of the Arcade Game Maker (AGM) and the Mobile Media (MM) PLAs (see Section 3.1), all of them available in [12].

Hypotheses Formulation: based on the RQs, we defined the following hypotheses:

- **Hypotheses for RQ1:**
  - **H0**$_{RQ1}$**:** there is no difference in the performance of the crossover operators FdC, CC, and MC;
  - **H1**$_{RQ1}$**:** the crossover operators FdC and MC have better performance;
  - **H2**$_{RQ1}$**:** the crossover operators FdC and CC have better performance.
- **Hypotheses for RQ2:**
  - **H0**$_{RQ2}$**:** there is no difference in the performance of any chosen configurations;
  - **H1**$_{RQ2}$**:** configuration with population size of 100 individuals, with 30,000 fitness evaluations (300 generations), with 40% of crossover rate, and 80% of mutation rate, using all mutation operators has better performance;
  - **H2**$_{RQ2}$**:** configuration with population size of 200 individuals, with 3,000 fitness evaluations (15 generations), with 40% of crossover rate, and 80% of mutation rate, using all mutation operators has better performance;
  - **H3**$_{RQ2}$**:** configuration with population size of 200 individuals, with 3,000 fitness evaluations (15 generations), with 40% of crossover rate, and 90% of mutation rate, using all mutation operators has better performance.

Selection of Variables: Figure 1 depicts the selected independent (PLA and Algorithm Configuration) and dependent (ACLASS, COE, FM, HV, and ED) variables of the experiment. The PLA factor has four treatments, being two versions of the AGM and two versions of the MM PLAs, and the Algorithm Configuration factor has six treatments, which are configurations from Config1 to Config6 (see Table 2). ACLASS, COE, and FM are the three objective functions used, from which the quality indicators HV and ED are analyzed (see Section 2).

Table 2 presents a summary of all used configurations. The first column specifies what configuration uses the respective parameters.

Table 2: Experiment Configurations

Config.	Population Size	Number of Generations	Number of Fitness Evaluations	Crossover Operators	Mutation Operators	Crossover Rate	Mutation Rate
Config1	100	300	30,000	FdC + CC	All	0.4	0.8
Config2	100	300	30,000	FdC + MC	All	0.4	0.8
Config3	200	15	3,000	FdC + CC	All	0.4	0.8
Config4	200	15	3,000	FdC + MC	All	0.4	0.8
Config5	200	15	3,000	FdC + CC	All	0.4	0.9
Config6	200	15	3,000	FdC + MC	All	0.4	0.9

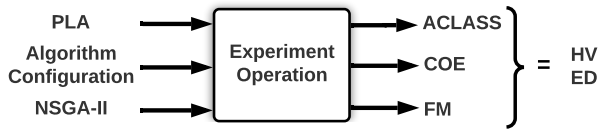


Figure 1: Variables of the experiment

The second, third and fourth columns show the population size, number of generations and fitness evaluations of the optimization algorithm (NSGA-II), respectively. The fifth and the eighth columns present the crossover operators and mutation rate used in each configuration. The combinations of these configurations were based on the best values for optimizing PLAs using only mutation operators [22] and on the best values achieved in [7, 8] when applying the crossover operators. The crossover rate did not vary because 0.4 was the best value found in [7]. All mutation operators were applied in every configuration because they complement each other.

3.3.2 Operation. We execute the experiment using the OPLA-Tool v2.0 [13]. For each configuration of the PLA optimization process, we executed 30 runs of the optimization algorithm, as recommended in [2, 3]. Considering six configurations with four PLAs (two variations of AGM and MM), we analyzed a total of four PLAs with six configurations running 30 times with a total of 720 runs of the optimization algorithm.

We carried out this process twice (Trial 1 and Trial 2) to increase the accuracy in the analysis of general results – as the optimization process can work with the randomness factor in certain situations, leading to different choices for the same executions – and to corroborate the results of the first trial. By running Trial 2 we obtain statistically samples with values tending to central dispersion measures, such as the mean.

3.3.3 Analysis and Interpretation. We collected and analyzed the data resulting from the experiment execution using statistical measures. Therefore, these data were organized in several files to serve as input for the R Studio and Google Spreadsheet software for statistical analysis.

The first step was the maintenance of spreadsheets with the values of the objective functions of the used PLAs resulting from the optimization process. These values were re-scaled, using min-max normalization as shown in Equation 1, for calculating HV. The main goal of this normalization was to define the reference point parameter of the HV. We also calculated the ED values for each execution of the configurations using Euclidean Distance for three dimensions as shown in Equation 2, since x, y and z are the values of the objective function, x', y' and z' are the ideal values for each objective function. Then, we compared the solutions with lower ED to find out which configuration had the best performance for each PLA.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

$$d = \sqrt{(x - x')^2 + (y - y')^2 + (z - z')^2} \tag{2}$$

HV was calculated using the PF_{Known} (see Section 2) generated by each configuration per PLA. As the values used to calculate HV were normalized between 0 and 1, the reference point was 1.01, to represent the worst solution that could be obtained. To obtain the ideal solution, which is used to calculate the ED value for each generated solution, we used PF_{true}. In our work, the ideal solution has the minimum value obtained for each objective, considering that MOA4PLA addresses the PLA design as a minimization problem. The fitness of the ideal solution for each PLA design is presented in Tables 3 and 4.

After calculating the HV for each run of the optimization algorithm, we used the results to check whether the samples follow a normal distribution using Shapiro-Wilk. Then, the statistical difference between the same PLA was analyzed with the Kruskal-Wallis test. From the results of this test, as presented in Section 4, we verified that the Vargha-Delaney test should be used to analyze the effect size of the results of the same PLA. Thus, we could check the best configuration among all the defined configurations.

However, the Vargha-Delaney test makes the comparison between two variables and in this study the comparison should be made over 6 configurations. Therefore, we paired all configurations and compared which pair had the best performance.

To complement the analysis of the configurations, we analyzed median values of ED per configuration. We then could verify which configurations had the best optimized solutions in terms of ED, confirming what had already been observed in previous statistical tests.

Section 4 describes and discusses the obtained results supported by the quality indicators and the statistical measures.

4 RESULTS AND DISCUSSION

In this section, we present and discuss the results obtained in our exploratory study. As we already mentioned, we extracted the values of the objective functions of the solutions resulting from the optimization process to analyze the results using quality indicators and

Table 3: Trial 1 - The lowest ED value per configuration (Fitness values are presented as (COE, ACLASS, FM))

PLA	Original Fitness	Ideal Fitness	Config1 ED	Config1 Fitness	Config2 ED	Config2 Fitness	Config3 ED	Config3 Fitness	Config4 ED	Config4 Fitness	Config5 ED	Config5 Fitness	Config6 ED	Config6 Fitness
AGM1	(7, 29, 914)	(1, 1, 327)	**7.00**	(7, 3, 330)	7.21	(7, 1, 331)	35.23	(1, 21, 356)	35.23	(1, 21, 356)	30.02	(7, 18, 351)	34.93	(1, 15, 359)
AGM2	(7, 30, 1040)	(1, 1, 102)	231.24	(9, 8, 333)	238.13	(9, 1, 340)	311.66	(4, 21, 413)	284.45	(7, 29, 385)	**28.02**	(2, 29, 102)	273.99	(11, 22, 375)
MM1	(5, 17, 885)	(2, 7, 165)	18.47	(11, 15, 179)	**13.89**	(9, 19, 165)	28.50	(8, 17, 191)	28.50	(8, 17, 191)	28.50	(8, 17, 191)	21.75	(17, 9, 183)
MM2	(5, 14, 1013)	(2, 4, 196)	**7.28**	(9, 6, 196)	12.57	(8, 5, 207)	51.15	(6, 14, 246)	51.59	(7, 10, 247)	43.15	(9, 11, 238)	49.53	(6, 10, 245)

Table 4: Trial 2 - The lowest ED value per configuration (Fitness values are presented as (COE, ACLASS, FM))

PLA	Original Fitness	Ideal Fitness	Config1 ED	Config1 Fitness	Config2 ED	Config2 Fitness	Config3 ED	Config3 Fitness	Config4 ED	Config4 Fitness	Config5 ED	Config5 Fitness	Config6 ED	Config6 Fitness
AGM1	(7, 29, 914)	(2, 1, 326)	32.45	(6, 20, 352)	**7.00**	(9, 1, 326)	32.26	(6, 21, 351)	35.96	(7, 23, 354)	33.78	(8, 24, 350)	32.45	(6, 20, 352)
AGM2	(7, 30, 1040)	(1, 1, 333)	**10.63**	(9, 8, 333)	**10.63**	(9, 1, 340)	59.08	(6, 22, 388)	59.41	(9, 22, 388)	58.74	(6, 18, 389)	56.05	(11, 22, 394)
MM1	(5, 17, 885)	(2, 7, 159)	**12.08**	(9, 16, 163)	14.53	(11, 10, 170)	34.06	(8, 17, 191)	43.46	(7, 17, 201)	26.93	(9, 17, 183)	34.06	(8, 17, 191)
MM2	(5, 14, 1013)	(2, 5, 192)	**6.00**	(8, 5, 192)	7.28	(9, 5, 194)	50.41	(7, 9, 242)	50.61	(8, 10, 242)	57.51	(9, 8, 249)	49.09	(11, 10, 240)

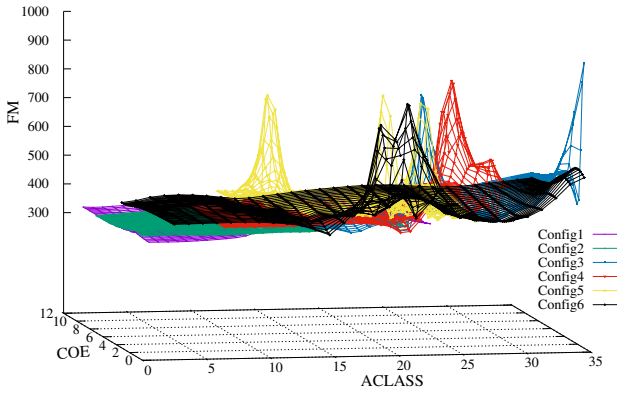


Figure 2: Trial 1 - Solutions Space Covered for AGM1

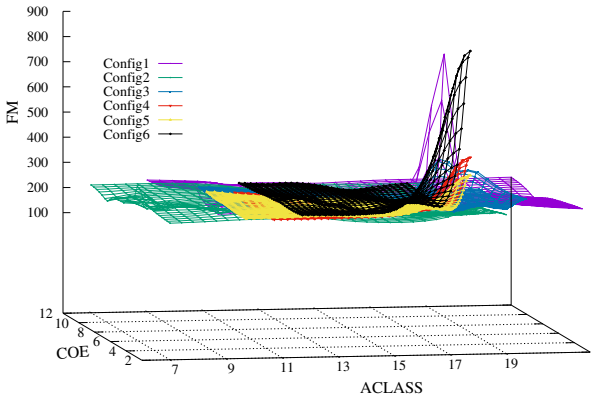


Figure 3: Trial 1 - Solutions Space Covered for MM1

applying statistical tests. We provide the solutions obtained by the configurations, the original PLA designs, and the results obtained by the quality indicators in the experimental package [12].

Diversity of solutions. Figure 2 shows the dispersion graph of the solutions obtained from all configurations for the PLA AGM1 in Trial 1. We can verify the configurations Config1 and Config2 cover the solutions space more efficiently than the other configurations, as they generate solutions with lower fitness values. This is justified by the fact that these configurations have the largest number of generations, which allows the algorithm to evolve further, generating better optimized solutions.

For MM1, in Figure 3, we can verify configurations Config1 and Config2 have a range very close to we observed in AGM1 (Figure 2). Furthermore, Config2 still has a greater number of better optimized solutions than Config1. The configurations had similar behavior for the other two PLAs (AGM2 and MM2) regarding the diversity of solutions. We then observe the same pattern of behavior in Trial 2, confirming the findings of Trial 1 regarding solutions space coverage.

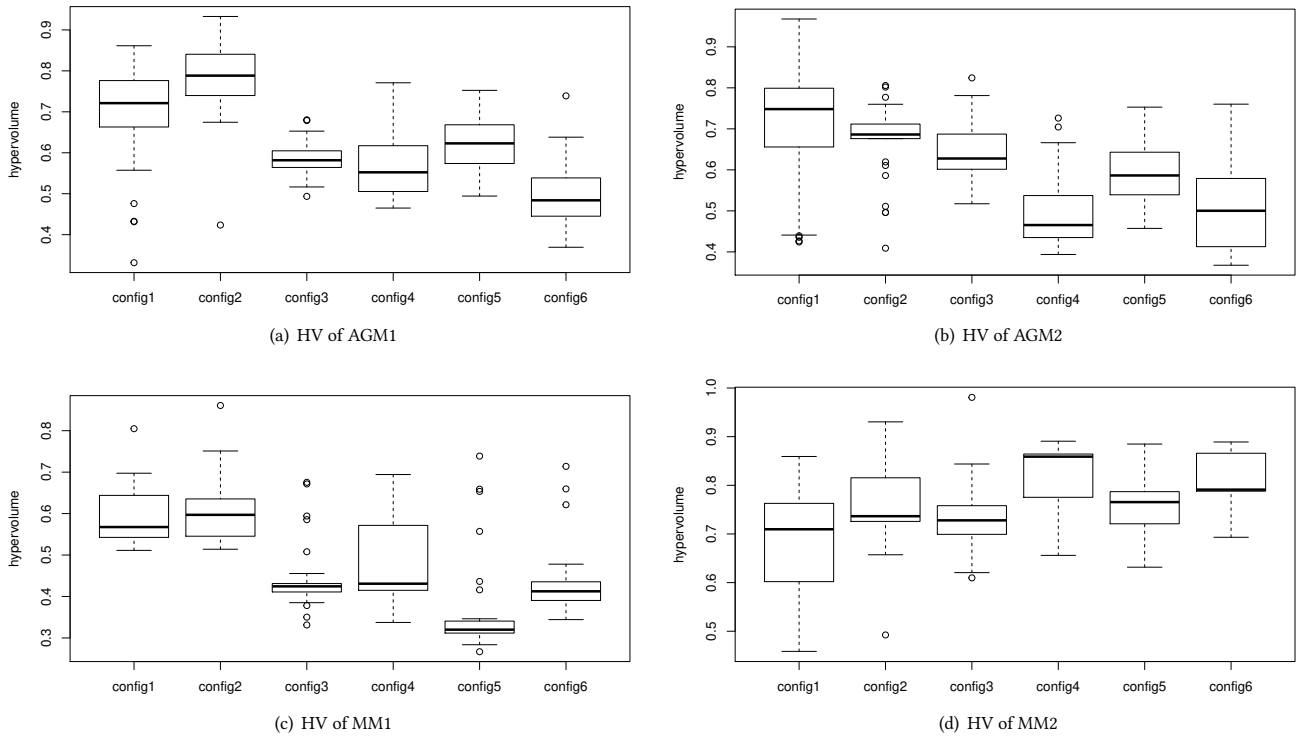
Hypervolume. We used the HV values in the normality, statistical difference, and effect size tests. First, we applied Shapiro-Wilk to each configuration to verify their distributions. Throughout this test we identified 67% (32) of the samples had non-normal distribution.

Thus, we used the results of the Kruskal-Wallis test (Table 5) to investigate the statistical difference among configurations for each PLA. We grouped data by PLA, and we verified in all cases tested there was a statistical difference, with a confidence level of 95%.

Table 5: Kruskal-Wallis *p-values*

PLA	Trial 1	Trial 2
AGM1	0.00000000000000022	0.00000000000514700
AGM2	0.00000000000020400	0.00000000000000022
MM1	0.00000000000000022	0.00000000000000022
MM2	0.00000000000672500	0.00000000000000022

Considering that in all cases we obtained statistical differences, we analyzed the pairs of configurations. To facilitate the observation of this difference between the configurations, Figure 4 presents the boxplots of the HV values used in the Kruskal-Wallis test. In such a figure, the closer to value 1, the better optimized are the solutions in the search space. Furthermore, the greater the spacing between



(a) HV of AGM1

(b) HV of AGM2

(c) HV of MM1

(d) HV of MM2

Figure 4: Trial 1 - Boxplots of Hypervolume (HV)

the minimum and maximum value, the greater the diversity of solutions for a given configuration.

We could verify by applying the Kruskal-Wallis test that Config3, Config4, Config5, and Config6 have similar statistical differences in some cases. The biggest difference is in Config1 and Config2, which have better optimized solutions in relation to the other configurations. As it can be seen in Figure 4, Config2 stands out over the other configurations in PLAs AGM1, MM1, and MM2. In addition, Config1 and Config2 solutions are better optimized. In 100% of the comparisons, the statistical test showed no evidence of a significant difference between Config1 and Config2.

However, in the Config1 and Config2 for AGM1, in Trial 1, and MM2, in both trials, the *p-value* was around 0.2, which does not indicate that there is a statistical difference between them, but it is closer than other configurations. It is noteworthy that Config1 and Config2 do not use the same crossover operators, but the same settings to number of generations and crossover and mutation rates. Considering that a larger number of generations is used (Figure 2) in these configurations, the optimization algorithm took longer to optimize the solutions and, consequently, to obtain a greater diversity of better optimized solutions.

Based on Figure 4, we observed configurations with even indexes (Config2, Config4, and Config6) have a similar behavior, as they perform better than configurations with odd indexes (Config1, Config3 and Config5). As shown in Figure 2, Config2, Config4, and Config6 use the FdC and MC crossover operators in the optimization process, which may have impacted the results.

Thus, we applied the Vargha-Delaney test comparison in pairs to verify effect size of samples. Considering this test allows comparing only two samples, all configurations were combined, applying the test to each pair. In Figures 5 and 6 the results obtained through comparisons using the Vargha-Delaney test are presented. In these scatter plots, if the point (value of A) referring to the comparison between two configurations is at 0.5, both have the same performance, that is, the resulting solutions are similar regarding how well optimized they are in the two configurations. When it is less than 0.5, the second configuration of the comparison obtained better solutions, and conversely, when the point is above 0.5, the first configuration presented better results.

We observed that Config2 in AGM1 in Trial 1 and Trial 2 showed better performance, as the effect size was large (A > 0.9), compared to Config3, Config4, Config5 and Config6. Regarding Config1, the effect size was medium (A ~ 0.7) in both trials. On the PLA AGM2 in Trial 1, the Config1 had better performance, because compared to Config4 and Config6 a large effect was obtained (A > 0.8) and compared to Config3 and Config5, it presented a medium effect (A > 0.7). Finally, comparing Config1 with Config2, it was found that the effect size between the two configurations was small (A = 0.65).

When analyzing the two versions of PLA MM in Trial 1 and Trial 2, there was a difference in results. In the case of MM1 in Trial 1, Config2 had better performance with a moderate effect size (A > 0.78) compared to the other configurations. In Trial 2, for MM1 the Config1 obtained large (A > 0.8) effect size compared to Config3, Config4, Config5 and Config6 and negligible difference (A > 0.54)

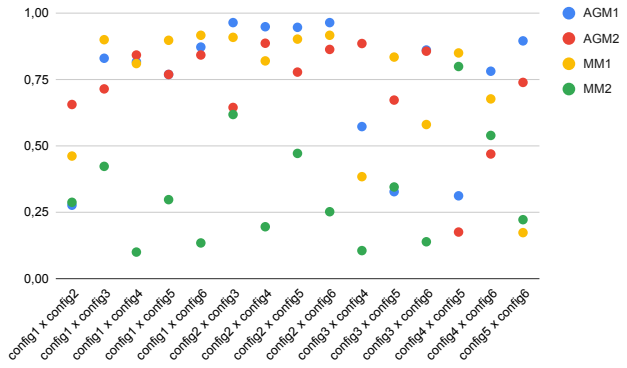


Figure 5: Vargha-Delaney Results for Trial 1

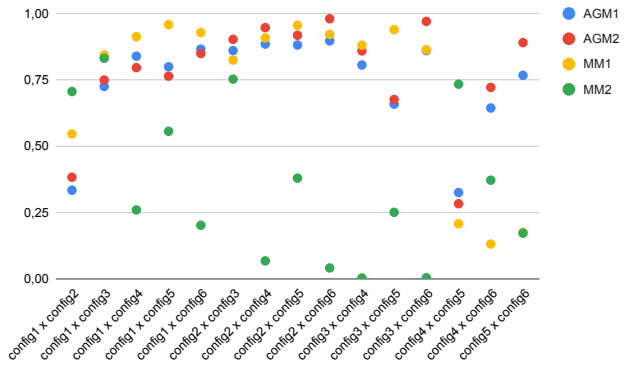


Figure 6: Vargha-Delaney Results for Trial 2

compared to Config2. In the case of MM2 in Trial 1, the Config4 and Config6 obtained a large difference. Finally, for MM2 in Trial 2, Config6 presented a large difference compared to Config2, Config3 and Config5 and a small difference compared to Config4 (A = 0.63).

The summary of the hypervolume results is that Config1 and Config2 show better results in PLAs AGM1, AGM2, and MM1 in both trials. Config2 obtained better results in both Trial 1 and Trial 2. In the case of MM2, Config4 and Config6 obtained better results in Trial 1 and Trial 2, respectively. Thus, configurations using FdC and MC operators performed better in 75% of configurations. Likewise, the use of configurations with 300 generations in the optimization algorithm showed better results in 75% of PLAs in the two trials.

The lowest Euclidean Distance to the Ideal Solution. To complement the aforementioned evidence, we identified which are the solutions that have the best trade-off among the objectives (solutions with the lowest ED). Tables 3 and 4 present the EDs of these solutions and their respective fitness value, calculated for each configuration of each PLA optimized in Trial 1 and Trial 2, respectively. These tables also present the fitness of the original PLA, provided as input to the optimization process, and the fitness of the ideal solution, which was used to obtain the ED value of each solution obtained by the samples in the two trials. In these tables, the lowest ED value obtained by PLA is highlighted in bold.

In Trial 1, Config1 obtained the solutions with the lowest ED for PLAs AGM1 and MM2, whereas Config2 and Config5 had the best performance for MM1 and AGM2, respectively. It is interesting to note that, even when they did not have the best performance, Config1 and Config2 found solutions with lower ED very close to the solution found by the configuration with the best performance.

The Trial 2 confirmed the results of Trial 1, as Config1 obtained the lowest ED solutions in 3 PLAs: AGM2, MM1, and MM2. An important observation is that both Config1 and Config2 generated solutions with the same distance to the ideal solution for AGM2, with an ED value equal to 10.63. In addition, in AGM1 the solution with the best ED was Config2. Thus, in both trials, Config1 and Config2 stood out over the others, obtaining the best optimized solutions.

Considering that solutions with ED value with better balance between objective functions are commonly chosen by software architects [27], we calculated the median of the EDs for each configuration (Table 6) to check the trend of these values. Through the median we observed Config1 stood out in obtaining better optimized solutions in Trial 1, whereas in Trial 2, Config2 tends to obtain better solutions. This confirms the superiority of Config1 and Config2 over other configurations.

Table 6: ED Median of Lowest Values

Config.	Trial 1	Trial 2
Config1	**12.8731476**	11.35659589
Config2	13.23112454	**8.955127851**
Config3	43.18733866	42.23355269
Config4	43.41120163	47.03447567
Config5	29.25613787	45.64260644
Config6	42.22813419	41.57526178

Runtime. The optimization algorithm execution time for each configuration varied according to the configuration. Config1 and Config2 required more time, as they have a greater number of generations (300), lasting around 12 hours to execute 30 runs, which is equivalent to around 24 minutes per run. On the other hand, the average time taken by Config3, Config4, Config5, and Config6, which evolved over only 15 generations, was 2 hours for 30 runs.

5 ANSWERING THE RESEARCH QUESTIONS

We answered in this section the research questions and the hypotheses posed in Section 3.3. Table 7 summarizes the results obtained using the quality indicators, including the configuration(s) that achieved the best result per indicator in each of the trial performed. The second and third lines of Table 7 present the Hypervolume (HV) values and the results of the lowest Euclidean Distance (ED), respectively.

It is noticeable that the behavior is similar in the two trials, in which Config1 and Config2 are better in most cases. Each of these configurations performed better in 44% of cases (including the tie in the ED indicator for AGM2 in Trial 2). Configurations Config4, Config5 and Config6 obtained the best result in only one case, which equates to 6% of the possibilities.

Table 7: The Best Configuration by Quality Indicator

Ind.	Trial 1				Trial 2			
	AGM1	AGM2	MM1	MM2	AGM1	AGM2	MM1	MM2
HV	Config2	Config2	Config2	Config4	Config1	Config2	Config1	Config6
ED	Config1	Config5	Config2	Config1	Config2	Config1/2	Config1	Config1

We analyzed the original designs of the PLAs to verify the design peculiarities that would aid in understanding the results, because in some cases the configurations with fewer generations performed well. Although AGM2 has two additional features compared to AGM1, they are well modularized. The MM1 and MM2 designs have worse feature modularization than the two AGM designs. However, MM2 has fewer architectural elements (components, interfaces, and classes) than MM1, making it easier to optimize. Thus, the optimization of PLAs AGM2 and MM2 is simpler, facilitating the convergence of the algorithm in a smaller number of generations. This justifies the fact that configurations such as Config4, Config5 and Config6 achieved the best performance in both quality indicators for AGM2 or MM2.

Another fact that supports this argumentation is the impact on the FM objective function value (Tables 3 and 4). It can be seen from the fitness of the original solution and from the fitness of the lower ED solutions that the FM value was optimized (decreased) in greater proportion for the PLAs AGM2 and MM2 than for their other respective versions.

RQ1: What is the best combination of crossover operators for search-based PLA design? In general, it is possible to notice in Table 7 that 56% of better configurations are Config2, Config4 and Config6. Such configurations use the combination of FdC and MC crossover operators (Table 2). When considering only HV results that involve statistical validation, configurations Config2, Config4 and Config6 performed better in 75% of cases. When considering the trend of ED, *i.e.* medians presented in Table 6, Config2 achieved the best performance for Trial 2, and the second best performance for Trial 1 (quite close to Config1). Hence, hypothesis H1 was accepted, indicating that the pair of crossover operators FdC and MC is more appropriate for the PLA optimization.

RQ2: What is the best parameter configuration for the PLA optimization process? When observing Table 7, the superiority of Config1 and Config2 configurations is evident in most cases. These two configurations use a population size of 100 individuals, with 300 generations, which results in 30,000 fitness assessments. These settings adopt 40% of crossover probability and 80% of mutation probability, using all mutation operators. Therefore, hypothesis H1 was accepted, indicating that the parameter values mentioned above constitute the best configuration for the optimization process.

6 GUIDELINES

This section presents some guidelines derived from the learning resulting from the experiment conducted in this work. It is intended that these guidelines aid the PLA architect in the task of PLA optimization using MOA4PLA by means of OPLA-Tool v2.0.

(1) ***Using the pair of crossover operators FdC and MC allows reaching better solutions both from the point of view of feature modularization and other architectural properties.*** As discussed in Section 2, the FdC operator aims to improve the feature modularization of the PLA designs obtained during the optimization process, and the MC operator seeks to maintain the modularization of certain packages of the parent PLAs in their offspring. The results of this and previous works [7, 8] evidence both that the operators fulfill their role and that their joint usage is beneficial for the PLA optimization. Thus, the use of these operators is beneficial for all types of PLA design, with well-modularized features or with poor feature modularization.

(2) ***Using 300 generations allows the generation of a greater diversity of solutions as well as obtaining better PLA solutions, even for small PLAs such as AGM1, AGM2 and MM1.*** This can be observed in Table 7 and happens because PLAs can evolve for longer, from generation to generation. It is noticed that Config1, which uses the combination of FdC and CC operators, presented good results in several cases, but this is more related to the number of generations than to the operators themselves, because the design of AGM1, AGM2 and MM1 have already well-modularized features. Furthermore, in Trial 1, Config2 showed better HV performance for these three PLAs.

(3) ***A small number (15) of generations combined with the FdC and MC crossover operators is sufficient to satisfactorily optimize small PLA design with poor feature modularization.*** This was observed in MM2, where Config4 and Config6 obtained greater diversity of solutions (HV indicator) over 15 generations, with populations of 200 individuals and using the FdC and MC crossover operators. The disadvantage is that this type of configuration could not find solutions so close to the ideal solution (ED indicator), such as Config1 that uses 300 generations and populations with 100 individuals. This guideline is also interesting when the architect considers the time to generate good PLA design alternatives because by adopting this configuration it is possible to generate solutions in few minutes (four minutes for running the MM2 PLA).

7 THREATS TO VALIDITY

In this section, we discuss the threats to validity related to our work, based on the taxonomy of Wohlin et al. [17]. We also describe how we mitigated possible threats.

Internal Validity. The sample is considered the main threat to validity identified in this work, since we used academic PLAs in the experiment. However, this threat was mitigated using PLAs from different domains, one PLA is related to media management for mobile devices (MM) and another to arcade games (AGM). Another threat is caused by the randomness of the algorithms and was minimized by the 30 independent runs that were performed in each trial, according to the recommendations of Arcuri and Briand [1].

External Validity. The sample size is identified as a threat to external validity, since a small sample with four PLAs was used. This sample consists of two versions of the AGM and two versions of the MM, as described in Section 3.1. Thus, it is not possible to generalize the experiment results for a larger population. However, the sample allowed the identification of guidelines, which can be refined and validated in studies with PLAs of other size and domain.

Construct Validity. The definition of the parameter values adopted in the configurations involved in the experiment can be

considered as a threat to validity. This threat was controlled by using values adopted in related works [7, 8, 22], which include calibration of the NSGA-II algorithm for the same problem addressed in this work.

Conclusion Validity. We minimize the main threats to the conclusion validity identified during the experiment using quality indicators and statistical tests commonly used in SBSE studies [4].

8 CONCLUSION

In this work we carried out an exploratory study to evaluate six different configurations to the multi-objective evolutionary algorithm used to optimize four PLAs. In this sense, we formulated two research questions to verify (i) which is the best combination of crossover operators, and (ii) the best values to configure the parameters to the PLA optimization process.

Empirical results pointed out that better solutions are obtained by the joint usage of the crossover operators FdC (*Feature-driven Crossover*) and MC (*Modular Crossover)*, corroborating the evidence raised in [7]. Furthermore, the number of generations performed by the evolutionary algorithm impacts on the evolution of PLAs, as expected, thus the higher the number of generations, the better the solutions, due to the time for PLAs to evolve.

These results increase the state of the art on PLA optimization using SBSE techniques with respect to the best values for the algorithm parameters configuration. In addition, the results of this work contribute to ease the adoption of search-based PLA design as newcomers (or non-expert companies) can use the findings of this work to configure MOA4PLA to optimize PLA design.

As future work, we aim to carry out a further study involving industrial and larger PLAs, to validate the guidelines proposed in this paper and to investigate whether the results obtained in this study might be generalized.

ACKNOWLEDGMENTS

This work is supported by CNPq grant 428994/2018-0, and CAPES grants 88887.615209/2021-00 and 88887.615213/2021-00.

REFERENCES

[1] Andrea Arcuri and Lionel Briand. 2014. A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing, Verification and Reliability* 24, 3 (2014), 219–250.

[2] Andrea Arcuri and Gordon Fraser. 2011. On Parameter Tuning in Search Based Software Engineering. In *3rd Symposium on Search Based Software Engineering (SSBSE)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 33–47.

[3] Andrea Arcuri and Gordon Fraser. 2013. Parameter tuning or default values? An empirical investigation in search-based software engineering. *Empirical Software Engineering* 18, 3 (2013), 594–623.

[4] Thelma Elita Colanzi, Wesley K. G. Assunção, Silvia R. Vergilio, Paulo Roberto Farah, and Giovani Guizzo. 2020. The Symposium on Search-Based Software Engineering: Past, Present and Future. *Information and Software Technology* 127 (2020), 106372.

[5] Thelma Elita Colanzi and Silvia Regina Vergilio. 2016. A feature-driven crossover operator for multi-objective and evolutionary optimization of product line architectures. *Journal of Systems and Software* 121 (2016), 126–143.

[6] Thelma Elita Colanzi, Silvia Regina Vergilio, Itana Gimenes, and Willian Nalepa Oizumi. 2014. A search-based approach for software product line design. In *18th International Software Product Line Conference (SPLC)*. Association for Computing Machinery, New York, NY, USA, 237–241.

[7] Diego F. da Silva. 2021. *Operadores de Cruzamento para aprimorar a Otimização de Arquitetura de Linha de Produto de Software.* Master's thesis. Universidade Estadual de Maringá, Programa de Pós-Graduação em Ciência da Computação, Departamento de Informática, Maringá.

[8] Diego F. da Silva, Luiz F. Okada, Thelma Elita Colanzi, and Wesley KG Assunção. 2020. Enhancing search-based product line design with crossover operators. In *22th Genetic and Evolutionary Computation Conference (GECCO)*. Association for Computing Machinery, New York, NY, USA, 1250–1258.

[9] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and Tamt Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[10] Frank Van der Linden, Klaus Schmid, and Eelco Rommes. 2007. *Software product lines in action: the best industrial practice in product line engineering.* Springer Science & Business Media, Berlin, Heidelberg.

[11] Eduardo Figueiredo, Nelio Cacho, Claudio Sant'Anna, Mario Monteiro, Uira Kulesza, Alessandro Garcia, Sergio Soares, Fabiano Ferrari, Safoora Khan, Fernando Castor Filho, et al. 2008. Evolving software product lines with aspects. In *30th International Conference on Software Engineering (ICSE)*. IEEE, Leipzig, Germany, 261–270.

[12] Willian Marques Freire, Simone de França Tonhão, Tiago Piperno Bonetti, Marcelo Yudi Shigenaga, William de Araujo Cadette, Fernando dos Santos Felizardo, Aline Maria Malachini Miotto Amaral, Edson OliveiraJr, and Thelma E. Colanzi. 2021. *Experimental package - On the configuration of multi-objective evolutionary algorithmsfor PLA design optimization.* Otimizes (UEM). https://doi.org/10.6084/m9.figshare.14903286

[13] Willian Marques Freire, Mamoru Massago, Arthur Cattaneo Zavadski, Aline Maria Malachini Miotto Amaral, and Thelma Elita Colanzi. 2020. OPLA-Tool v2.0: a Tool for Product Line Architecture Design Optimization. In *34th Brazilian Symposium on Software Engineering (SBES)*. Association for Computing Machinery, New York, NY, USA, 818–823.

[14] David E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., United States.

[15] Mark Harman, Yue Jia, Jens Krinke, William B. Langdon, Justyna Petke, and Yuanyuan Zhang. 2014. Search based software engineering for software product line engineering: a survey and directions for future work. In *18th International Software Product Line Conference (SPLC)*. Association for Computing Machinery, New York, NY, USA, 5–18.

[16] Mark Harman and Bryan F. Jones. 2001. Search-based software engineering. *Information and Soft. Technology* 43, 14 (2001), 833–839.

[17] Martin Höst, Björn Regnell, and Claes Wohlin. 2000. Using students as subjects—a comparative study of students and professionals in lead-time impact assessment. *Empirical Software Engineering* 5, 3 (2000), 201–214.

[18] Antonio Contieri Jr, Guilherme Correia, Thelma Elita Colanzi, Itana Gimenes, Edson OliveiraJr, Sandra Ferrari, Paulo Masiero, and Alessandro Garcia. 2011. Extending UML components to develop software product-line architectures: Lessons learned. In *5th European Conference on Software Architecture (ECSA)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 130–138.

[19] William Kruskal and Allen Wallis. 1952. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association* 47, 260 (1952), 583–621.

[20] Luciane Baldo Nicolodi, Thelma Elita Colanzi, and Wesley K. G. Assunção. 2020. Architectural Feature Re-Modularization for Software Product Line Evolution. In *14th Brazilian Symposium on Software Components, Architectures, and Reuse (SBCARS)*. Association for Computing Machinery, New York, NY, USA, 31–40.

[21] Camila Nunes, Uirá Kulesza, Cláudio Sant'Anna, Ingrid Nunes, Alessandro Garcia, and Carlos de Lucena. 2009. Assessment of the Design Modularity and Stability of Multi-Agent System Product Lines. *J. UCS* 15, 11 (2009), 2254–2283.

[22] Narcizo Palioto and Thelma Elita Colanzi. 2020. Configuração de Algoritmos Genéticos Multiobjetivos para Otimização de Projeto de Arquitetura de Linha de Produto. In *Anais da IV Escola Regional de Engenharia de Software (ERES)*. SBC, Porto Alegre, RS, Brasil, 204–213.

[23] SEI. 2009. *Arcade Game Maker pedagogical product line.* SEI. https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=485941

[24] Samuel Shapiro and Martin Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika* 52, 3/4 (1965), 591–611.

[25] András Vargha and Harold Delaney. 2000. A critique and improvement of the CL common language effect size statistics of McGraw and Wong. *Journal of Educational and Behavioral Statistics* 25, 2 (2000), 101–132.

[26] Yenisei Delgado Verdecia, Thelma Elita Colanzi, Silvia Regina Vergilio, and Marcelo C. Santos. 2017. An Enhanced Evaluation Model for Search-based Product Line Architecture Design. In *20th Iberoamerican Conference on Software Engineering (CIbSE)*. CIbSE, San Jose, Costa Rica, 155–168.

[27] Milan Zeleny and James L. Cochrane. 1973. *Multiple criteria decision making.* University of South Carolina Press, United States.

[28] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert Da Fonseca. 2003. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation* 7, 2 (2003), 117–132.