Multi-Objective Metamorphic Test Case Selection: an Industrial Case Study (Practical Experience Report)

Jon Ayerdi*, Aitor Arrieta*, Ernest Bota Pobee* and Maite Arratibel †
Mondragon Unibertsitatea*, Orona †
*{jayerdi,aarrieta,ebpobee}@mondragon.edu, †marratibel@orona-group.com

***Abstract*—Metamorphic testing is a technique that has shown great potential to alleviate the test oracle problem by exploiting the relations among the inputs and outputs of different executions of a system. However, this approach requires multiple test executions. In applications like Cyber-Physical Systems (CPSs), where the test executions can be very expensive in terms of time and resources needed, this can supose a problem. Therefore, it is paramount to optimize the test suite to reduce the costs of verifying the system. Test case selection is an optimization technique which accomplishes this by selecting a subset of test cases while aiming to preserve the effectiveness of the original test suite as much as possible. While there are many approaches for test case selection in the existing literature, none of them has been proposed for the metamorphic test case selection problem, where each metamorphic test case consists of a source and, at least, a follow-up test case pair.**

In this work, we present an evolutionary multi-objective approach for the metamorphic test case selection problem, adapting existing multi-objective test selection techniques and proposing new evolutionary operators and objective functions. Furthermore, we evaluate our approach with a set of metamorphic tests developed for an industrial case study from the elevation domain. The results suggest that our approach outperforms both Random Search and the same metaheuristic algorithm without the new evolutionary operators we propose.

***Index Terms*—Cyber-Physical Systems, Elevators, Metamorphic Testing, Test Selection**

I. Introduction

Cyber-Physical Systems (CPSs) are heterogeneous systems that integrate physical and software components [1], [2], [3]. Multi-elevator installations are highly configurable CPSs that must satisfy transportation demands while providing the best possible Quality of Service (QoS) to the passengers. For instance, the Average Waiting Time (AWT) of the passengers is considered to be one of the most important QoS metrics to measure passenger satisfaction [4]. The values of such metrics are, however, very volatile in short time frames, since slight variations in timing can change how the elevators are dispatched, which in turn changes the state of the installation for the rest of the scenario. Given how many parameters this system has and how complex it is, determining whether a test outcome is correct or not is very difficult. This is one of the fundamental problems in software testing, known as the *oracle problem* [5]. In practice, this is often solved by employing human oracles, i.e., manual verification by the test engineers. However, this is an expensive solution that does not scale well.

Metamorphic testing [6] is an alternative testing technique which can be used to alleviate the oracle problem. It consists in defining relations among *multiple* test executions that the system must hold, the so called Metamorphic Relations (MRs). This technique has reportedly been used in many domains with great success. Specifically in the domain of CPSs, it has been used for testing wireless sensor networks [7], autonomous drones [8], and self-driving cars [9], [10], among others. Segura et al. proposed the use of metamorphic testing for performance testing [11], [12], and Ayerdi et al. applied it in the context of industrial elevator systems [13].

Unfortunately, testing this type of system is very costly, even if simulation-based environments are used. In addition, metamorphic testing requires multiple test cases to check the output relations, which usually translates to even higher costs. Test selection is a technique which aims to reduce the cost of testing while maintaining the effectiveness of the test suite by selecting a subset of the available test cases.

In this practical experience report, we present our findings and lessons learned from applying metamorphic test selection in an industrial case study from the elevation domain. To the best of our knowledge, this is the first work in the literature proposing an approach for metamorphic test selection, and also the first to report on the usage of this technique in an industrial case study.

We propose and evaluate an approach to metamorphic test selection based on the NSGA-II evolutionary algorithm. We present the problem representation, genetic operators, and multiple possible objective functions to guide the search towards better solutions. Then, we compare our approach with a Random Search baseline, and we also evaluate the effectiveness of the new genetic operators we propose. We also compare the different possible objective function combinations. We employ the test suite's execution cost and mutation score as metrics to determine the quality of the solutions. We conclude that our approach is effective wrt the baselines and that the new genetic operators we propose result in consistently better results. We then evaluate and discuss the best objective function combinations to obtain cost-effective test suites.

In order to facilitate further research on this topic, we provide a replication package for our experiments, including the source code, experimental data, and our results [14].

The rest of the paper is structured as follows: Section II provides some background on metamorphic testing and our case study. Section III describes our test selection approach. Section IV presents the empirical evaluation. Section V summarizes the lessons learned and some future prospects. Section VI points out some threats to validity and how we mitigated them. Section VII describes the related work in the literature. Section VIII concludes the paper.

II. Background

A. Metamorphic Testing

Metamorphic Testing (MT) [6] is a technique that consists in checking known relations among the inputs and outputs of two or more test executions from the system under test, the so called *Metamorphic Relations* (MRs). For example, consider the function $abs(x)$, which computes the absolute value of x. We can define the following simple MR:

$$(x_f = -x_s) \Rightarrow (abs(x_f) = abs(x_s))$$

This MR implies that if the value of the input is negated ($x_f = -x_s$), then the output of the function must not change ($abs(x_f) = abs(x_s)$). Here, x_s is the *source test case*, and x_f, created by negating x_s, is the *follow-up test case*. The first part of the implication, which determines the relation between both test cases, is the *input relation*, and the assertion over the outputs from both test cases is the *output relation*. MRs are only applicable to test pairs which satisfy the input relation.

Segura et al. proposed the use of MT in order to detect performance-related issues in the program under test [11], [12]. For instance, rendering a larger bitmap image on a browser is expected to result in a higher memory usage. This type of MRs had already been suggested in earlier work [7]. In [15], a MR based on page load times is used to reveal bugs in the Adobe Launch Tag Manager software. More recently, Ayerdi et al. applied this concept in the elevation domain [13], and presented an approach to identify bugs on an elevator dispatcher by using MRs based on performance metrics.

B. Application Domain

Our application domain is a CPS provided by Orona, one of the largest elevator companies in Europe. Elevator systems integrate different computing units, as can be seen in Figure 1. All these computing elements are communicated through the Controller Area Network (CAN) bus. Each elevator has its own microprocessor, which is in charge of controlling different aspects of the (individual) elevator (e.g., speed of the elevator, opening and closing of the doors). In each of the floors, there are elevator calling buttons. These can be either conventional (in which the passenger only provides the traveling direction, i.e., up or down), or destination-aware (in which passengers select their destination floor). In this paper we focus on conventional installations. There is a last computing unit called traffic master. This microprocessor receives information of the status of each of the elevators (e.g., floor, estimation of the number of passengers), as well as information of each of the calls. Based on this information, the traffic master decides which elevator should attend each of the calls. This decision is taken by the traffic dispatching algorithm, which is the System Under Test (SUT) in this paper. The traffic dispatching algorithm continuously evolves [16] to deal with the inclusion of new functionalities, repairing of bugs, etc. Therefore, carefully testing this system is paramount.

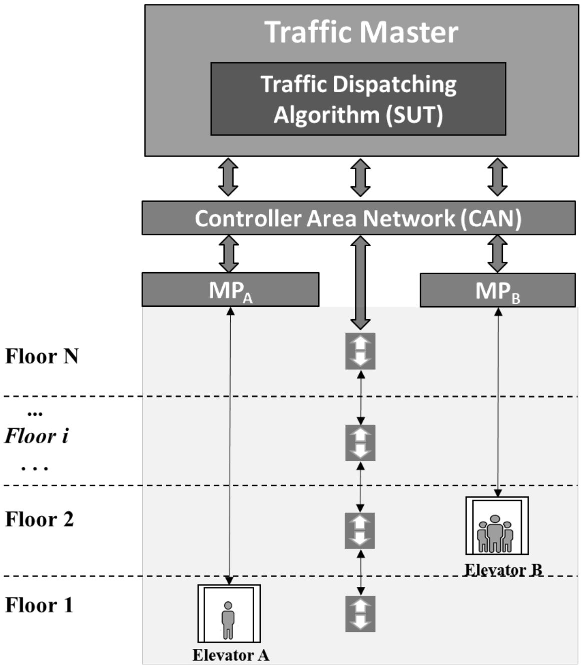


Fig. 1: Illustration of the industry case study

To test the traffic dispatching algorithm, simulation-based testing is employed [16]. The first test execution level is called Software-in-the-Loop (SiL). At this test level, Elevate[1] is employed to run the tests, which is a commercial and domain-specific simulation tool to test elevator traffic dispatching algorithms. The second test level is Hardware-in-the-Loop (HiL). In this case, the SUT is embedded in the real target processor, integrated with the real-time infrastructure (e.g., drivers, operating systems, communications). The remaining computing units are real, and the physical layer of the CPS (e.g., engines, mechanical parts) are emulated in some real-time test benches. The last test level refers to the real elevator. It is noteworthy that as the test levels increase, the cost for executing tests becomes more expensive. Therefore, it is important to detect faults early at the SiL test level, if possible.

In this paper we focus on the SiL test level. A test case at this test level for our industrial case study involves two files. On the one hand, the installation file, which provides all the necessary information related to the building in which the elevators are (e.g., number of floors, number and features of the elevators). On the other hand, the passenger file, which refers to the passengers traveling through the building. This file involves a set of passengers, each with certain attributes (e.g., the floor at which they arrive, the destination floor, the weight of the passengers, the time they take to enter and exit

[1] https://elevate.helpdocsonline.com/home

the floor). With these two files, it is possible to execute a test in the form of a simulation. Elevate returns a file, with different output information (e.g., the time required to wait by each passenger, the consumed energy). This information is parsed by the metamorphic oracles to provide a test verdict.

In a previous study, it was found that metamorphic testing was effective at detecting faults in this application domain [13]. However, since multiple test executions are required, and thousands of test cases exist, running the full test suite for every new version of the system is infeasible. Therefore, test selection approaches are required. In this paper we propose a multi-objective test case selection approach for metamorphic testing, specifically adapted to this application context.

C. Metamorphic Testing of Dispatching Algorithms

In this work, we evaluate metamorphic test case selection approaches on the elevation domain case study and MRs presented in [13].

The MRs we use employ the following performance metrics for elevator systems:

Average Waiting Time (AWT). This is the average time from the moment a landing call is issued until an elevator stops to attend the call. This is among the most important metrics for providing a good user experience [4].

Total Distance (TD). This is the sum of the distances traversed by all the elevators of the building, measured in floors. This metric may help reveal issues such as dispatching multiple elevators for a single call.

Total Movements (TM). This is the count of all the movements (i.e., engine start-ups) of all the elevators of the building. This metric may help reveal issues such as dispatching multiple elevators for a single call.

The MRs are based on the following Metamorphic Relation Input Patterns (MRIPs), which define the transformations performed to the source test cases in order to generate the follow-up test cases.

MRIP1: Additional call. This MRIP consists in introducing an additional passenger call to the source test case. Formally, we can define the input relation as $C_f = C_s \cup c'$, where c' is the additional call. Since test executions are highly volatile, the additional passenger call c' will always arrive after the rest of the passengers in order to ensure that the follow-up test execution does not diverge from the source test execution in unforeseen ways.

MRIP2: Additional elevators. This MRIP consists in enabling one or more extra elevators for the follow-up test case. Formally, this can be defined as $Ef = Es \cup E'$, where E' is a non-empty set of elevators.

MRIP3: Initial position change. This MRIP consists in changing the initial positions of the elevators without changing their number. Formally, we can express this as $E_f \neq E_s$, constrained by $|E_f| = |E_s|$.

III. Approach

A. Problem formulation

We define metamorphic test selection as an optimization problem which aims to maximize the effectiveness of a test suite while minimizing its cost. We specifically use the mutation score (MS) as the metric to determine the effectiveness and the execution time as the cost metric for this work, although different metrics could be used with our approach. The test suite TS consists of a set of test case pairs ($TC^m \in TS$), each of which belongs to a specific MR. There is a finite set of MRs ($MR_n \in MR$), each of which has one or more test case pairs in TS ($TC^m_{MR_n} \in TS$). In this context, the test case pairs consist of inputs only. We assume that the test outputs are not available yet, since the goal of the selection process is to reduce the cost of executing the test cases. Hence, the test selection must select a subset of the test cases $TS' \subset TS$ while reducing the cost and maintaining a high MS as best as possible. Only the test inputs and MRs of the test pairs are known, so the expected cost and effectiveness of a given set of test cases must be predicted or approximated based on that information alone.

B. Algorithm

Given our problem formulation, we chose a multi-objective search algorithm which will explore the search space of the problem (selections of the metamorphic tests) based on various features of the test inputs.

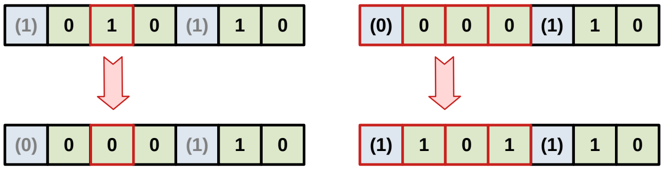
Inspired by the existing literature on test selection for similar contexts [17], we chose the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [18] as the metaheuristic for our approach. This algorithm is known to not scale well beyond three objectives, and there are other genetic algorithms that have been developed to overcome this limitation [19], but we do not consider it necessary to combine four or more objective functions for this problem.

C. Genetic Operators

In this section, we describe the specific solution representation and genetic operators we use for the metamorphic test selection problem. Some of the mutation and crossover operators are all based on traditional genetic operators [20]. We also define new operators based on source test case groups, considering the fact that all the test cases within the same source group will be very similar to each other (all the follow-ups will have been derived from the same source test case).

1) Solution representation: We represent a metamorphic test selection solution as a bit set indicating whether each of the follow-up test cases has been selected or not. The source test-cases are not part of the solution because they can be implicitly selected based on the follow-ups: If one or more of the follow-ups corresponding with the source test case were selected, then the source test case must be selected in order to evaluate the MR, and otherwise, the source test case should not be selected because it will not be used. This representation allows for source test cases which have multiple follow-ups, but assumes that each follow-up test case has a single corresponding source test case. To the best of our knowledge, there is no metamorphic testing work which contradicts this assumption in the existing literature.

Besides the solution representations (bit sets representing selected follow-ups), the test selection program must keep the set of "source groups", i.e., a mapping of source test cases and their corresponding follow-up test cases. This mapping is needed in order to calculate some fitness functions, such as the cost of executing the selected tests, and it is also used by some of the genetic operators we use.



(a) Follow-up flip

(b) Source group flip

Fig. 2: Mutation operators

2) Mutation:

- *Follow-up flip*. This operator selects or deselects a single follow-up test case from the test suite. With our solution representation, this is equivalent to a standard bit-flip mutation. Figure 2a shows a single follow-up test case being chosen as a mutation point and getting deselected. Note that the source test cases (bits between parentheses with blue background) cannot be mutation points, and their values are simply recomputed based on whether at least one of their follow-ups is selected or not. In this example, the first source test case becomes deselected after the mutation because none of its follow-ups is selected.
- *Source group flip*. This operator selects or deselects all the follow-up test cases corresponding with a single source test case. In the case of selecting, one of the follow-ups is selected first, and then every other follow-up for the source test case has a fixed probability of being selected. Figure 2b shows the first source test case being chosen as a mutation point and getting selected. Once selected, every single one of its follow-ups can be selected with a fixed probability. In this case, its first and third follow-ups are selected, but the second one is not.

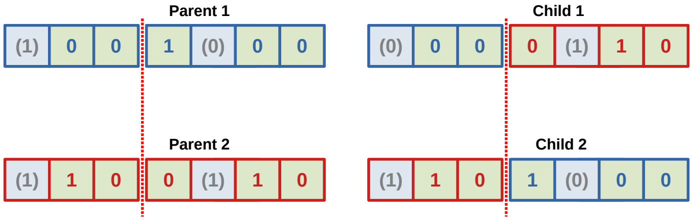


Fig. 3: Follow-up crossover

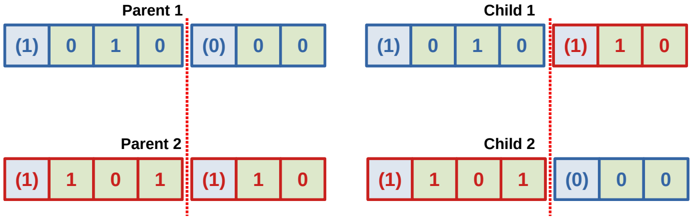


Fig. 4: Source group crossover

3) Crossover:

- *Follow-up crossover*. This operator performs a single-point crossover over the follow-up test cases. With our solution representation, this is equivalent to a standard single-point crossover. Figure 3 shows an example where the crossover point (dotted vertical red line) is after the second follow-up test case. Note that the source test cases (bits between parentheses with blue background) are ignored and recomputed after the crossover. In this example, Child 1 ends up having the first source test case deselected despite both parents having it selected, because none of its follow-ups is selected after the crossover.
- *Source group crossover*. This operator performs a single point crossover over the source test-case groups. Figure 4 shows an example where the crossover point (dotted vertical red line) is after the first source test case. Child 1 ends up with Parent 1's selections for source test case 1 and Parent 2's selections for source test case 2, while Child 2 is constructed the other way around.

4) Multiple Operators: Since our approach uses multiple mutation and crossover operators, a strategy for applying them must be defined. In this case, when performing a mutation or crossover, we simply choose one of the operators randomly with equal probability, and then apply the chosen operator.

D. Fitness Functions

In order to guide the search algorithm towards the best possible solutions, we define a set of fitness functions specific to the metamorphic test selection problem. Furthermore, some of the fitness functions are based on features of the test cases which are specific to the domain of our case study. All of the fitness functions other than the cost return values to be maximized, so in our implementation their values are negated in order to make all of them minimization objectives.

- *Cost*. This fitness function aims to minimize the cost of the execution for the selected test cases. It is calculated as the sum of the costs of the selected test cases. The objective is to minimize this value. For the elevation domain, we employ the following value in order to approximate the execution time in seconds for a given test case:

$$T_{last} - T_{first} + 60$$

  where T_{last} is the arrival time of the last passenger, and T_{first} is the arrival time of the first passenger, both measured in seconds. This formula simply assumes that all the passengers will be transported to their destination 60 seconds after the last call to the elevators, which was found to be a good approximation after some preliminary tests. A cost fitness function should be defined for any domain (e.g. the cost of each test case could just be 1 if there is no better approximation or if all test cases have similar costs), but the estimated test case cost we use here is specific to the elevation domain.

- *MR Coverage*. This fitness function aims to balance the number of test case pairs selected for each of the MRs. Since different MRs might be able to detect different types of failures, diversifying the MRs checked is expected to result in a higher failure detection capability. The value of this function is specifically calculated as:

$$\min_{\forall MR_n \in MR}\left(\frac{|TC^{selected}_{MR_n}|}{|TC^{all}_{MR_n}|}\right)$$

where MR is the set of all MRs used in our test suite, $|TC^{selected}_{MR_n}|$ is the count of test cases for MR_n selected, and $|TC^{all}_{MR_n}|$ is the total count of test cases for MR_n. The objective is to maximize this value. This fitness function is domain-agnostic.

- *Input Diversity*. This fitness function aims to maximize the diversity of the inputs [21]. The input diversity of a solution TS' is calculated as the sum of the minimum distance of each input test case from every other input test case in the test suite:

$$\sum_{t_1 \in TS'}\left(\min_{t_2 \in TS' \setminus t_1}(dist(t_1, t_2))\right)$$

The $dist(t_1, t_2)$ function computes the euclidean distance between t_1 and t_2, which are numeric vectors representing test inputs (not test input pairs). The objective is to maximize this value. This fitness function is domain-agnostic, but if the test cases do not consist of numeric inputs, it may require defining numeric features over the inputs which might be domain-specific. In this work, we employ domain-specific features such as elevators count, passenger count, or the ratio of passengers going up/down from lower/middle/upper floors in order to compute input diversity.

- *Passenger Density*. This fitness function aims to maximize the passenger density of the selected test cases, given the expectation that more dense test cases will result in more operations from the elevator system, and therefore more potential to reveal failures. The passenger density of a set of test cases is calculated as the sum of the passenger counts on each test case divided by the sum of the costs of each test case. The objective is to maximize this value. This fitness function is specific to the elevation domain.

- *Passenger Count*. This fitness function aims to maximize the total count of passengers in the selected test cases, because more passengers might force the elevator system to make more complex decisions, potentially revealing more failures. The total count of passengers is calculated as the sum of passengers counts on each test case. The objective is to maximize this value. This fitness function is specific to the elevation domain.

- *Passenger Distance Traveled*. This fitness function aims to maximize the total distance traveled by the passengers in the selected test cases, since longer trips might force the elevator system to make more complex decisions, potentially revealing more failures. The total distance traveled is calculated as the sum of travel distances from each passenger on each test case, measured in floors. The objective is to maximize this value. This fitness function is specific to the elevation domain.

TABLE I: Objective combinations derived from the proposed fitness functions. Each combination had *Cost* as an additional objective.

Combination	Fitness 1	Fitness 2
c1	MR Coverage	
c2	Input Diversity	
c3	Passenger Density	
c4	Passenger Count	
c5	Passenger Distance Traveled	
c6	MR Coverage	Input Diversity
c7	MR Coverage	Passenger Density
c8	MR Coverage	Passenger Count
c9	MR Coverage	Passenger Distance Traveled
c10	Input Diversity	Passenger Density
c11	Input Diversity	Passenger Count
c12	Input Diversity	Passenger Distance Traveled
c13	Passenger Density	Passenger Count
c14	Passenger Density	Passenger Distance Traveled
c15	Passenger Count	Passenger Distance Traveled

The maximum number of fitness functions per combination was three, as NSGA-II does not scale well for more than three objective functions [19]. Therefore, each combination was configured with the *Cost* objective combined with one or two additional objectives. Given our other five proposed fitness functions, we formed 15 combinations in total, summarized in Table I.

IV. Empirical Evaluation

This section details our evaluation and analysis of the results obtained from our experiments.

A. Research Questions

Based on the data obtained from our experiment and subsequent evaluations, we defined the following research questions (RQs):

- RQ1 - *How does our proposed approach perform when compared with Random Search (RS)?* We define this RQ to assess the performance of our approach (NSGA-II-MET) in the selection of cost-effective test cases, and whether it is an improvement over RS. To answer this RQ, we pair the 15 different objective combinations with our approach and RS and compare the results.
- RQ2 - *How do the new genetic operators we propose perform?* We define this RQ to assess the effectiveness of the source group based mutation and crossover operators from our approach (Source group flip and Source group crossover), and whether it is an improvement over just using the traditional operators (Follow-up flip and Follow-up crossover). We refer to the variant of our approach which only uses the traditional genetic operators as NSGA-II-TR. In order to answer this RQ, we pair each of the 15 different

objective combinations with NSGA-II-MET and NSGA-II-TR to compare the results.

- RQ3 - *Which of the different objective combinations performs the best when paired with our approach?* This RQ helps us to determine the best objective combination for selecting the most cost-effective test cases. To answer this RQ, we report and compare the results of all 15 objective combinations for NSGA-II-MET.

B. Experimental Set-up

1) Algorithm Configuration: NSGA-II-MET and NSGA-II-TR were configured with populations of 100 with the total number of fitness evaluations set to 25,000. The NSGA-II approaches were configured with a crossover rate of 1.0, and a mutation probability of *1/N*, where *N* is the count of test cases. All the other parameters were chosen based on existing work related to multi-objective test case selection [20], [22], [23], [24], [25]. The selection operator used was binary tournament [18]. For a fair comparison, RS was also configured to run 25,000 fitness evaluations.

2) Evaluation Dataset: For this evaluation, we employed the same set of test cases and mutants as the ones presented in [13]. The metamorphic tests are based on the performance metrics and MRIPs described in Section II-C. The test suite consists of 140 randomly generated source test cases and 1200 follow-up test cases: 420 for MRIP1 (Additional call), 360 for MRIP2 (Additional elevators), and 420 for MRIP3 (Initial position change). These are short-scenario test cases with an average duration of approximately 3 minutes. As for the mutants, there are 89 mutants of Orona's most common dispatching algorithm, which were generated by seeding faults based on traditional operator mutations [26]. The test suite achieves a total mutation score of 83.1% for the provided mutants (74 out of 89 mutants detected) [13].

The evaluation dataset consists of two separate groups of artifacts: (1) The set of test case inputs and the list of test input pairs for each MR, and (2) a table indicating which mutants are killed by each test case pair. Only the data from group (1) is used for the test selection itself, since the data from group (2) would not be available without executing the entire test suite. The data from (2) is used exclusively to compute the mutation score of a subset of the test case pairs to evaluate the effectiveness of metamorphic test selection solutions.

3) Evaluation Metrics: Pareto-based search algorithms provide more than one solution. The provided solutions are non-dominated among themselves. For each of these solutions we measured (1) the *cost* and (2) the *mutation score*:

Cost - The cost metric determines how expensive it is to execute the test suite. To this end, we measure the time required by all the test cases in the test suite to execute.

Mutation score - The mutation score is the ratio of mutants that are killed (i.e., identified as incorrect) by a test suite. A mutant is considered killed if at least one metamorphic test identifies it as incorrect.

It is important to note that the Pareto-frontier provided by the search algorithm is non-dominated based on the selected objective functions. However, the mutation score of each of the solutions is unknown. Therefore, with both the *cost* and *mutation score* of each objective function we derived a second Pareto-frontier. Similar to other multi-objective test case selection studies [27], [25], [28], [29], with this second Pareto-frontier, we derived the **Hypervolume** (HV). This metric is a quality indicator which is often used in the assessment of Pareto-based search algorithms [30], [31], [32]. The higher the HV, the better the performance of the Pareto-based search algorithm.

While the HV is a widely used metric to assess multi-objective search algorithms [30], in order to transfer multi-objective test case selection studies to the industry, decision makers (DMs) are necessary. A DM takes as input a Pareto-frontier returned by the search algorithms, and decides which the best solution is. To this end, we implemented a time-budget based DM. Our DM takes as input the Pareto-frontier and a time budget, which is given by the user. With this, the DM returns the solution that is closest to the time budget without exceeding it. We believe that these types of DMs are of interest for the industry. For example, in a domain analysis carried out in an industrial set-up, they showed interest in executing as many test cases as possible within a given time budget [33]. The time budgets are specified in percentages (i.e., a 10% of time budget means that the 10% of the the execution cost of the entire test suite is permitted). In the remainder of the paper, we refer to a specific DM configuration as DM_X, X being the percentage time budget. For each DM, we measure the mutation score obtained by the solution returned by it.

4) Statistical Tests: Since the algorithms we used are stochastic, we run each algorithm with each objective combination 50 times, as recommended by Arcuri and Briand [34]. The results were analyzed by means of statistical tests. Specifically we applied the *Shapiro-Wilk* test [35] to determine the distribution of the data.

After determining that the data was normally distributed, the t-test was used to assess whether there was statistical significance between two different algorithms. We considered there was statistical significance if the p-value was below 0.05. In addition, we used the Vargha and Delaney $\hat{A}_{12}$ test [36] to assess the effect size of the difference between approaches. We categorize the $\hat{A}_{12}$ values as **N** (*negligible*) if $d < 0.147$, **S** (*Small*) if $d < 0.33$, **M** (*Medium*) if $d < 0.474$, or **L** (*Large*) if $d >= 0.474$, where $d = 2 \cdot |\hat{A}_{12} - 0.5|$ [37].

C. Analysis of the Results

In this section, we present the evaluation results obtained and we discuss the answers to the RQs.

RQ1 - How does our proposed approach perform when compared with Random Search (RS)?

In this RQ, we compare our proposed approach (NSGA-II-MET) with Random Search implementation (RS) as a sanity check of our study.

Table II presents the results of the statistical comparison between the HV results from our approach (NSGA-II-MET) and Random Search (RS) in the "Effect size" column under

TABLE II: (RQ1 + RQ2) Statistically significant effect sizes and average improvement of NSGA-II-MET with respect to RS and NSGA-II-TR for the HV metric.

Combination	NSGA-II-MET vs RS		NSGA-II-MET vs NSGA-II-TR	
	Effect size	Improvement	Effect size	Improvement
c1	L+	52.53%	-	0.32%
c2	L+	53.05%	-	-0.39%
c3	L+	41.81%	L+	14.45%
c4	L+	69.70%	L+	2.91%
c5	L+	71.62%	M+	1.36%
c6	L+	50.99%	-	0.10%
c7	L+	47.99%	L+	7.24%
c8	L+	64.87%	S+	1.22%
c9	L+	63.74%	M+	2.07%
c10	L+	51.57%	-	0.79%
c11	L+	66.42%	L+	2.37%
c12	L+	66.57%	L+	1.97%
c13	L+	60.44%	L+	4.66%
c14	L+	60.65%	L+	5.09%
c15	L+	70.05%	L+	2.37%
Average		59.47%		3.10%

TABLE III: (RQ1 + RQ2) - Statistically significant effect sizes for mutation scores across different DMs.

		DM_1	DM_5	DM_10	DM_15	DM_20	DM_25	DM_30	DM_35	DM_40	DM_45	DM_50	DM_55	DM_60
NSGA-II-MET vs RS	c1	M+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L-	L-	L-
	c2	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	-	-	L+
	c3	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	M+	L+	L+
	c4	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+
	c5	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+
	c6	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L-	L-	M-
	c7	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	-	-	S+
	c8	-	L+	L+	L+	L+	L+	L+	L+	L+	L+	-	-	-
	c9	N+	L+	L+	L+	L+	L+	L+	L+	L+	L+	-	-	-
	c10	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	M-	M-	S-
	c11	N+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+
	c12	N+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+
	c13	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	S+	-	N+
	c14	N+	L+	L+	L+	L+	L+	L+	L+	L+	L+	S+	-	-
	c15	S+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	L+	-
NSGA-II-MET vs NSGA-II-TR	c1	-	-	L+	M+	M+	S+	-	-	-	M-	L-	L-	L-
	c2	M+	L+	M+	S+	-	-	-	S-	S-	-	-	-	-
	c3	S+	-	-	M+	L+	L+	L+	L+	L+	L+	L+	L+	L+
	c4	-	-	S+	L+	L+	L+	L+	M+	L+	L+	L+	L+	M+
	c5	-	S+	S+	-	M+	L+	L+	L+	-	S+	-	-	-
	c6	-	-	M+	S+	-	-	-	-	-	-	S-	M-	M-
	c7	-	-	M+	S+	-	M+	L+	L+	L+	L+	L+	L+	M+
	c8	-	-	M+	L+	-	-	-	-	-	-	-	-	S+
	c9	-	-	L+	L+	L+	-	S+	-	-	M+	-	-	-
	c10	-	-	-	-	-	-	-	-	M+	-	-	-	-
	c11	-	-	L+	M+	M+	S+	-	-	-	-	-	-	
	c12	-	-	L+	M+	L+	-	-	-	-	-	-	-	M+
	c13	-	-	M+	L+	M+	M+	M+	L+	M+	L+	L+	M+	L+
	c14	-	S+	M+	L+	L+	L+	L+	-	L+	M+	M+	L+	L+
	c15	-	S+	-	M+	L+	L+	L+	L+	M+	L+	L+	L+	M+

“NSGA-II-MET vs RS”. The results from the $\hat{A}_{12}$ values show that our approach is more effective with statistical significance and a large effect size for every single objective function combination.

On the other hand, the “Improvement” column besides it shows the average improvement of our approach wrt RS. We can observe that the improvement is approximately between 40% and 70%, with the average improvement for all the objective function combinations being around 60%.

Using the Time Budget Decision Makers with the best objective function combination (c15), Figure 5 reveals that RS fails to find suitable solutions for under 45% of the maximum cost. The efficiency is also lower for higher cost solutions up to 60% of the cost. Table III shows a detailed view of every possible mutation score comparisons across all objective and DM combinations. This view reveals that RS obtains better mutation scores for the three highest time budgets in c1, c6 and c10, but consistently worse results for every other objective combination. As we will later show in RQ3, these three objective combinations are among the least effective ones, so RS obtaining some better results with them is not very meaningful.

> **Answer to RQ1:** Our approach obtained significantly better overall results than RS for every single objective function combination.

RQ2 - How do the new genetic operators we propose perform?

In this RQ, we compare our approach (NSGA-II-MET) with a reduced variant where only the traditional genetic operators are used (NSGA-II-TR).

The “Effect size” column under “NSGA-II-MET vs NSGA-II-TR” from Table II shows that the results from our approach with the new genetic operators dominate over the version without them. The results show statistically significant improvements for 11 out of 15 objective combinations and no statistical significance for the rest of them.

On the other hand, the “Improvement” column next to it shows the average improvement of our approach wrt the variant without the new genetic operators. We can observe that the improvement is approximately between 0% and 14%, with the average improvement for all the objective function combinations being around 3%.

Figure 5 shows that the new genetic operators provide a modest but consistent improvement of the mutation score for all the time budgets with the best objective combination. The lower rows from Table III reveal similar results from the comparison with RS, with NSGA-II-MET dominating NSGA-II-TR for all objective combinations except three of them: c1, c2 and c6 in this case. Similarly to the previous case, the results for RQ3 will reveal that these objective functions are among the least effective ones, rendering this favorable comparison irrelevant.

> **Answer to RQ2:** Our approach with the new genetic operators obtained similar or moderately better results for all the objective function combinations, revealing that the new operators improve the search process.

RQ3 - Which of the different objective combinations performs the best when paired with our approach?

This RQ compares the performances of the different objective function combinations to identify the most effective ones.

Table IV shows the effect size comparison of all the objective combinations with each other for our approach. The results show that c4, c5 and c15 dominate the rest of the

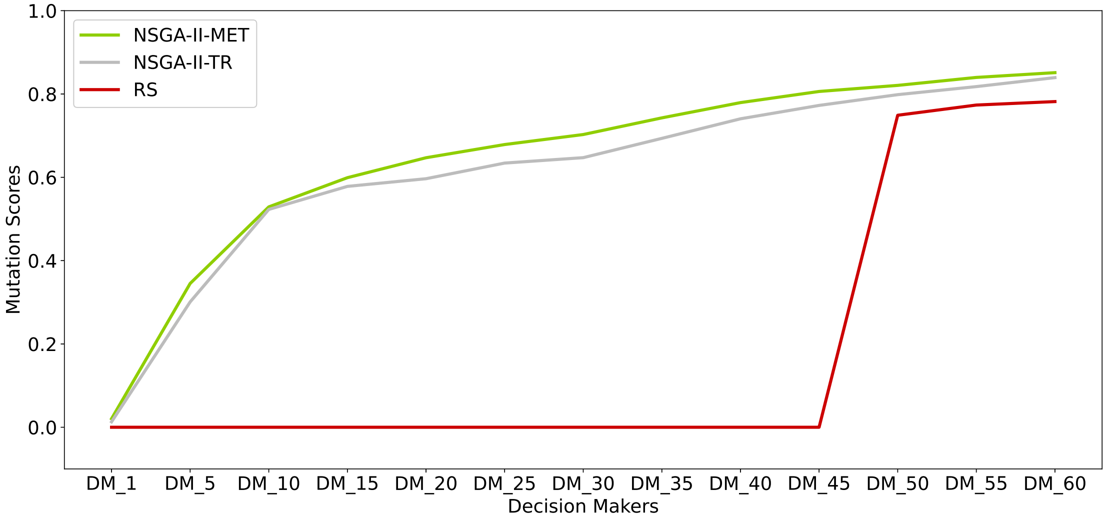


Fig. 5: Average mutation scores for the best performing objective combination (c15).

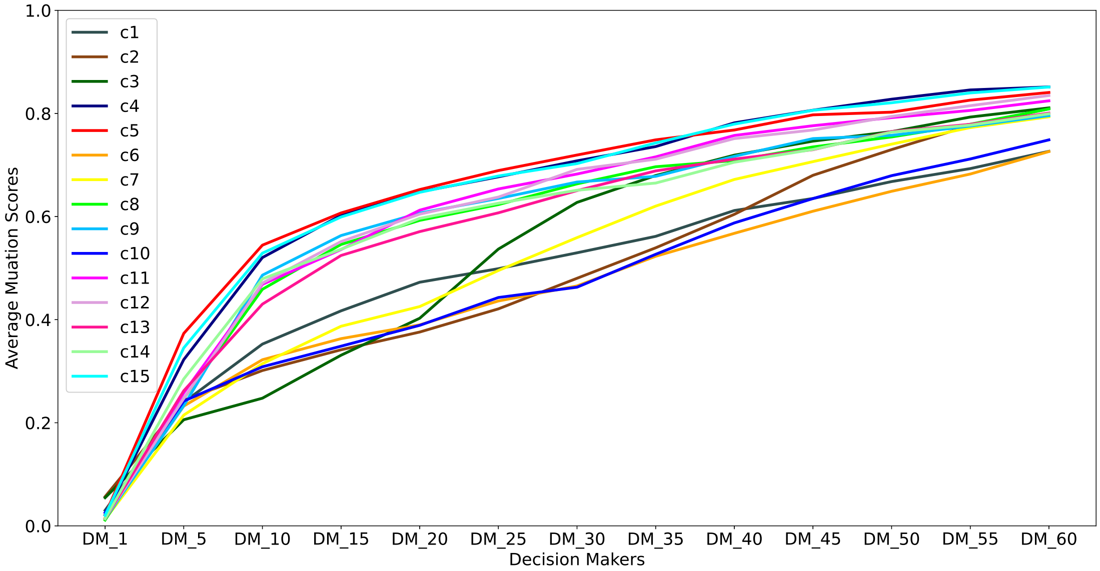


Fig. 6: Comparison of average mutation scores for all objective combinations.

objective function combinations, while none of them dominate each other. Recall that these objective functions correspond with Passenger Count (c4), Passenger Distance Traveled (c5) and a compound of both of these objectives (c15).

The next best objective combinations are c11 and c12, which are compounds of Input Diversity with Passenger Count and Passenger Distance Traveled respectively. Following these, the next best results are obtained by c8 and c9, which are compounds of MR Coverage with Passenger Count and Passenger Distance Traveled. On the other side of the spectrum, the worst results are obtained by c2 (Input Diversity), followed by c1 (MR Coverage), c6 (MR Coverage + Input Diversity) and c10 (Input Diversity + Passenger Density).

Figure 6 shows a graph of the average mutation scores obtained with every objective function combination for all the decision makers. Overall, c15 is the objective combination with the best average HV result, although c4 and c5 obtain similar results for all the execution time budgets.

The results clearly show that the domain-specific Passenger Count and Passenger Distance Traveled fitness functions are the most effective, and outperform generic fitness functions such as MR Coverage or Input diversity.

TABLE IV: (RQ3) Statistically significant HV effect size comparison of objective combinations for NSGA-II-MET. "+" suffix indicates better HV for the objective in the first column.

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15
c1	-	S+	L-	L-	L-	-	L-	L-	L-	-	L-	L-	L-	L-	L-
c2	S–	-	L-	L-	L-	S-	L-	L-	L-	M-	L-	L-	L-	L-	L-
c3	L+	L+	-	L-	L-	L+	S-	L-	L-	M+	L-	L-	L-	L-	L-
c4	L+	L+	L+	-	-	L+	L+	L+	L+	L+	S+	M+	L+	L+	-
c5	L+	L+	L+	-	-	L+	L+	L+	L+	L+	S+	S+	L+	M+	-
c6	-	S+	L-	L-	L-	-	L-	L-	L-	-	L-	L-	L-	L-	L-
c7	L+	L+	S+	L-	L-	L+	-	L-	L-	L+	L-	L-	L-	L-	L-
c8	L+	L+	L+	L-	L-	L+	L+	-	-	L+	L-	L-	-	M-	L-
c9	L+	L+	L+	L-	L-	L+	L+	N+	-	L+	L-	L-	S-	S-	L-
c10	-	M+	M-	L-	L-	-	L-	L-	L-	-	L-	L-	L-	L-	L-
c11	L+	L+	L+	S-	S-	L+	L+	L+	L+	L+	-	-	M+	S+	M-
c12	L+	L+	L+	M-	S-	L+	L+	L+	L+	L+	-	-	M+	-	M-
c13	L+	L+	L+	L-	L-	L+	L+	-	-	L+	M-	M-	-	-	L-
c14	L+	L+	L+	L-	M-	L+	L+	M+	S+	L+	S-	-	-	-	L-
c15	L+	L+	L+	-	-	L+	L+	L+	L+	L+	M+	M+	L+	L+	-

Answer to RQ3: Maximizing Passenger Count, Passenger Distance Traveled, or a combination of both are the best possible objective function combinations, while Input Diversity and MR Coverage are the worst.

V. Lessons Learned and Future Prospects

Next, we describe the lessons learned from this work and the future prospects we foresee.

Lesson 1: Effectiveness of the approach. Our experiments have shown us that there is a lot of potential to minimize the cost of metamorphic test suites without compromising their fault detection capability. With our approach and the best objective functions, on average, over 60% of the whole test suite's mutation score can be achieved for only 10% of the total test execution cost, and over 90% of the mutation score can be achieved for only 50% of the total cost. These insights have convinced Orona of the benefits of employing metamorphic test selection in their workflow. Metamorphic testing itself is a new addition to Orona's testing process, so the test selection approach presented in this paper will be adopted initially as the only currently implemented alternative. In the future, new fitness functions and other potential improvements will also be evaluated, especially because the current random test generation approach is likely to change.

Lesson 2: Domain-specific objective functions may be more effective. We have learned that the objective functions used for the test selection have a high impact on the final outcome. For our case study, the experimental results have shown that maximizing the number of passengers and the distance they traverse, which are domain-specific, are the best objective functions. We must note that our elevation case study has a somewhat special input space, since the inputs are tasks for the systems to perform. Under this generalization, it appears that the best objectives are maximizing count and the total complexity of the tasks to perform, which makes sense. On the other hand, it appears that diversifying the inputs is not very effective in comparison, which might indicate that simpler test cases, in this case test cases with fewer passengers or smaller travel distances, are not very useful in general.

Lesson 3: Different objective functions depending on time budget. The results also reveal that different objective functions might be more effective depending on the target time budget. Figure 6 shows that c5 (Passenger Distance Traveled) obtains the best results up to DM_35, but beyond that point c4 (Passenger Count) becomes more effective. Overall, however, c15 (which is a compound of c4 and c5) happens to have the best HV metric. The difference is rather small for this particular case, so always using c15 would be acceptable. Nevertheless, using different objective functions for the test selection depending on the target time budget might be more worthwhile for other case studies. For instance, the graph shows cases such as c3 (Passenger Density), which has a particularly low effectiveness between DM_5 and DM_35, but catches up with more effective objective combinations such as c13 beyond that point.

Future prospect 1: Generalizability to other systems. Since this work is based on a single industrial case study, it would be interesting to analyze whether the results can be generalized to CPSs from other domains. In particular, we suspect that the selection of the most effective objective functions may be domain-specific. Thus, a more thorough evaluation with multiple case studies would be needed to derive guidelines on how to select the best objective functions based on the characteristics of the case study.

Future prospect 2: MR Coverage based on historical results. Our results have shown that the MR Coverage objective is not a very effective fitness function in comparison with other ones. However, this function merely attempts to balance the usage of all the MRs in the test suite equally, which might not be the best objective. As the results from the MRs for our case study have shown, there is a great disparity between the effectiveness of the different MRs [13]. This means that skewing the MR Coverage objective to aim for more coverage of the most effective MRs may be more appropriate. This is not possible to do under our current problem formulation, since the outcome of executing the tests is unknown. Nevertheless, as the same MRs are used to test different versions of the system, data on the relative effectiveness of the different MRs would gradually become available, which would allow us to tune the MR Coverage. An analysis on how the MR Coverage objective can be tuned as more information becomes available might reveal ways to greatly increase the cost-effectiveness of the metamorphic test selection.

Future prospect 3: Online test generation. Taking the idea of using historical results a step further, it might be possible to obtain a better test suite by guiding the test generation based on previous test execution results. Instead of selecting metamorphic test cases from a large test suite, the test cases could be generated and selected incrementally, such that the test generation and selection process can use information from the previous iteration. This way, for example, the MR Coverage can be dynamically adjusted as more metamorphic tests are executed. Moreover, test cases that are deemed ineffective can be discarded to minimize the cost of future usages of the final test suite. The general idea of updating the test suite to maximize its effectiveness while minimizing its cost is known as the whole test suite generation approach [38].

VI. Threats to Validity

An *internal validity* threat in our evaluation is related to the generated mutants. In the field of CPSs, using mutants is compute-intensive. Therefore, using a large set of mutants is often not feasible [39]. However, the number of mutants used in this study was similar to those used in other empirical evaluations of testing CPSs [39], [40], [41]. The employed parameters (e.g., population size) is another threat of our study. We used the same parameters as other studies tackling the multi-objective test case selection problem [39], [20].

A *conclusion validity* threat of our evaluation relates to the non-deterministic nature of the employed search algorithms. To deal with it, we run each algorithm 50 times. Furthermore, we applied statistical tests to carefully analyze the results.

An *external validity* threat of our evaluation is related to the generalization of the results. While it is true that we only applied our approach into a single case study, it must also be acknowledged that this is a complex industrial case study.

One threat to the *construct validity* of our experiment was that parameters varied across the selected algorithms. We alleviated this threat by configuring all the algorithms to evaluate the fitness functions 25,000 times for a fair comparison.

VII. Related Work

A. Test Case Selection

The problem of test case selection is multi-objective in nature [42]. For this reason, Pareto-based search algorithms have been widely applied to tackle this problem. Yoo and Harman [20] were the first to propose the use of Pareto-based search algorithms for test case selection. Since then, different multi-objective test case selection approaches have been proposed for a wide range of applications, including software-product lines [23], autonomous vehicles [43] and java programs [44]. In our case, the targeted system is the elevator traffic dispatching algorithm from Orona, which is considered a CPS. In the field of CPSs, test case selection has been applied in different areas, such as configurable CPSs [20] and MATLAB/Simulink models [17]. These approaches are based on the outputs from previous executions. Our approach is different to these in various aspects. Firstly, it is designed for selecting metamorphic test cases, for which we propose new genetic operators. Secondly, our approach does not need any previous test executions, as it is based solely on test inputs. Lastly, we apply our approach to an industrial case study.

Other approaches have proposed different genetic operators for test case selection. For instance, Panichella et al. [28] propose operators to inject diversity through the search process. Arrieta et al. [27] propose seeding the initial population of the search algorithm. Olsthoorn and Panichella [29] propose a novel crossover based on linkage learning. Unlike all of these studies, our proposed genetic operators are designed for the selection of metamorphic tests.

B. Metamorphic Testing Cost Minimization

To the best of our knowledge, this work is the first application of metamorphic test selection. Nevertheless, some related work that attempts to reduce the cost of metamorphic testing by similar means has already been presented in the literature.

Some studies have proposed different criteria to select the most effective MRs, such as the features of the MR (e.g. output relations that are simple equalities are generally considered weaker) or their ability to trigger different execution paths [45], [46], [47]. More recently, Srinivasan et al. explored coverage and fault based criteria to prioritize MRs [48]. While this approaches can reduce the cost of metamorphic testing, they imply completely accepting or rejecting MRs. The possibility of generating more test cases for the most effective MRs but still using a few test cases for less effective MRs could potentially increase the coverage of the test suite.

As for test selection strategies, Barus et al. noted that random testing is often adopted as the source test case selection strategy, and reported that employing Adaptive Random Testing (ART) can enhance the effectiveness of metamorphic testing [49]. As for white-box approaches, Alatawi et al. proposed the use of dynamic symbolic execution (DSE) to generate source test cases [50], whereas Saha et al. explored using various coverage-based criteria from EvoSuite [51]. However, none of the existing works addresses the metamorphic test selection problem (source and follow-up test pairs).

VIII. Conclusion

In this work, we have evaluated the usefulness of metamorphic test selection in order to minimize the cost of metamorphic testing while maintaining its effectiveness as much as possible. Our experience with an industrial case study from the elevation domain has shown that there is a great potential in this technique.

We propose a general multi-objective search-based approach for this, and we define both generic and domain-specific objective functions to guide the metamorphic test selection. In our empirical evaluation, we find out that our approach beats all the baselines. In addition, we discover that the most effective objective functions are domain-specific metrics that favor higher complexity scenarios in our system. Finally, we propose future research avenues in order to improve the generalizability of our approach, as well as to further optimize the cost-effectiveness of metamorphic testing for CPSs.

Acknowledgments

This work is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871319. Jon Ayerdi, Aitor Arrieta and Ernest Pobee are part of the Software and Systems Engineering research group of Mondragon Unibertsitatea (IT1519-22), supported by the Department of Education, Universities and Research of the Basque Country.

References

[1] R. Baheti and H. Gill, "Cyber-physical systems," *The impact of control technology*, vol. 12, no. 1, pp. 161–166, 2011.

[2] R. Alur, *Principles of cyber-physical systems*. MIT Press, 2015.

[3] E. A. Lee and S. A. Seshia, *Introduction to embedded systems: A cyber-physical systems approach*. Mit Press, 2016.

[4] G. Barney and L. Al-Sharif, *Elevator traffic handbook: theory and practice*. Routledge, 2015.

[5] E. T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo, "The oracle problem in software testing: A survey," *IEEE transactions on software engineering*, vol. 41, no. 5, pp. 507–525, 2014.

[6] T. Chen, S. Cheung, and S. Yiu, "Metamorphic testing: a new approach for generating next test cases. technical report hkust-cs98-01," *Hong Kong Univ. of Science and Technology*, 1998.

[7] W. Chan, T. Y. Chen, S. C. Cheung, T. Tse, and Z. Zhang, "Towards the testing of power-aware software applications for wireless sensor networks," in *International Conference on Reliable Software Technologies*. Springer, 2007, pp. 84–99.

[8] M. Lindvall, A. Porter, G. Magnusson, and C. Schulze, "Metamorphic model-based testing of autonomous systems," in *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*. IEEE, 2017, pp. 35–41.

[9] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th international conference on software engineering*. ACM, 2018, pp. 303–314.

[10] Z. Q. Zhou and L. Sun, "Metamorphic testing of driverless cars," *Communications of the ACM*, vol. 62, no. 3, pp. 61–67, 2019.

[11] S. Segura, J. Troya, A. Durán, and A. Ruiz-Cortés, "Performance metamorphic testing: Motivation and challenges," in *2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER)*, 2017, pp. 7–10.

[12] S. Segura, J. Troya, A. Durán, and A. Ruiz-Cortés, "Performance metamorphic testing: A proof of concept," *Information and Software Technology*, vol. 98, pp. 1 – 4, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095058491830017X

[13] J. Ayerdi, S. Segura, A. Arrieta, G. Sagardui, and M. Arratibel, "Qos-aware metamorphic testing: An elevation case study," in *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2020.

[14] J. Ayerdi, A. Arrieta, E. B. Pobee, and M. Arratibel, "Replication package," 2022, last access: Aug 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6971634

[15] O. Johnston, D. Jarman, J. Berry, Z. Q. Zhou, and T. Y. Chen, "Metamorphic relations for detection of performance anomalies," in *2019 IEEE/ACM 4th International Workshop on Metamorphic Testing (MET)*. IEEE, 2019, pp. 63–69.

[16] J. Ayerdi, A. Garciandia, A. Arrieta, W. Afzal, E. Enoiu, A. Agirre, G. Sagardui, M. Arratibel, and O. Sellin, "Towards a taxonomy for eliciting design-operation continuum requirements of cyber-physical systems," in *2020 IEEE 28th International Requirements Engineering Conference (RE)*. IEEE, 2020, pp. 280–290.

[17] A. Arrieta, S. Wang, U. Markiegi, A. Arruabarrena, L. Etxeberria, and G. Sagardui, "Pareto efficient multi-objective black-box test case selection for simulation-based testing," *Information and Software Technology*, vol. 114, pp. 137–154, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584918301721

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] A. Panichella, F. M. Kifetew, and P. Tonella, "Automated test case generation as a many-objective optimisation problem with dynamic selection of the targets," *IEEE Transactions on Software Engineering*, vol. 44, no. 2, pp. 122–158, 2017.

[20] S. Yoo and M. Harman, "Pareto efficient multi-objective test case selection," in *Proceedings of the 2007 international symposium on Software testing and analysis*, 2007, pp. 140–150.

[21] T. Y. Chen, H. Leung, and I. Mak, "Adaptive random testing," in *Annual Asian Computing Science Conference*. Springer, 2004, pp. 320–329.

[22] D. Pradhan, S. Wang, S. Ali, and T. Yue, "Search-based cost-effective test case selection within a time budget: An empirical study," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1085–1092. [Online]. Available: https://doi.org/10.1145/2908812.2908850

[23] S. Wang, S. Ali, and A. Gotlieb, "Minimizing test suites in software product lines using weight-based genetic algorithms," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 1493–1500. [Online]. Available: https://doi.org/10.1145/2463372.2463545

[24] A. Arrieta, S. Wang, G. Sagardui, and L. Etxeberria, "Search-based test case selection of cyber-physical system product lines for simulation-based validation," in *Proceedings of the 20th International Systems and Software Product Line Conference*, ser. SPLC '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 297–306. [Online]. Available: https://doi.org/10.1145/2934466.2946046

[25] A. Arrieta, S. Wang, U. Markiegi, A. Arruabarrena, L. Etxeberria, and G. Sagardui, "Pareto efficient multi-objective black-box test case selection for simulation-based testing," *Information and Software Technology*, vol. 114, pp. 137–154, 2019.

[26] H. Agrawal, R. DeMillo, R. Hathaway, W. Hsu, W. Hsu, E. W. Krauser, R. J. Martin, A. P. Mathur, and E. Spafford, "Design of mutant operators for the c programming language," Technical Report SERC-TR-41-P, Software Engineering Research Center, Purdue ..., Tech. Rep., 1989.

[27] A. Arrieta, P. Valle, J. A. Agirre, and G. Sagardui, "Some seeds are strong: Seeding strategies for search-based test case selection," *ACM Trans. Softw. Eng. Methodol.*, apr 2022, just Accepted. [Online]. Available: https://doi.org/10.1145/3532182

[28] A. Panichella, R. Oliveto, M. Di Penta, and A. De Lucia, "Improving multi-objective test case selection by injecting diversity in genetic algorithms," *IEEE Transactions on Software Engineering*, vol. 41, no. 4, pp. 358–383, 2014.

[29] M. Olsthoorn and A. Panichella, "Multi-objective test case selection through linkage learning-based crossover," in *International Symposium on Search Based Software Engineering*. Springer, 2021, pp. 87–102.

[30] S. Wang, S. Ali, T. Yue, Y. Li, and M. Liaaen, "A practical guide to select quality indicators for assessing pareto-based search algorithms in search-based software engineering," in *Proceedings of the 38th International Conference on Software Engineering*, 2016, pp. 631–642.

[31] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[32] J. J. Durillo and A. J. Nebro, "jmetal: A java framework for multi-objective optimization," *Advances in Engineering Software*, vol. 42, no. 10, pp. 760–771, 2011.

[33] D. Pradhan, S. Wang, S. Ali, and T. Yue, "Search-based cost-effective test case selection within a time budget: An empirical study," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 2016, pp. 1085–1092.

[34] A. Arcuri and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering," in *Proceedings of the 33rd International Conference on Software Engineering*, ser. ICSE '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 1–10. [Online]. Available: https://doi.org/10.1145/1985793.1985795

[35] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[36] A. Vargha and H. D. Delaney, "A critique and improvement of the cl common language effect size statistics of mcgraw and wong," *Journal of Educational and Behavioral Statistics*, vol. 25, no. 2, pp. 101–132, 2000.

[37] J. Romano, J. D. Kromrey, J. Coraggio, J. Skowronek, and L. Devine, "Exploring methods for evaluating group differences on the nsse and other surveys: Are the t-test and cohen'sd indices the most appropriate choices," in *annual meeting of the Southern Association for Institutional Research*, 2006, pp. 1–51.

[38] G. Fraser and A. Arcuri, "Whole test suite generation," *IEEE Transactions on Software Engineering*, vol. 39, no. 2, pp. 276–291, 2012.

[39] A. Arrieta, S. Wang, G. Sagardui, and L. Etxeberria, "Search-based test case prioritization for simulation-based testing of cyber-physical system product lines," *Journal of Systems and Software*, vol. 149, pp. 1–34, 2019.

[40] B. Liu, S. Nejati, L. C. Briand *et al.*, "Improving fault localization for simulink models using search-based testing and prediction models," in *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2017, pp. 359–370.

[41] R. Matinnejad, S. Nejati, L. C. Briand, and T. Bruckmann, "Test generation and test prioritization for simulink models with dynamic behavior," *IEEE Transactions on Software Engineering*, vol. 45, no. 9, pp. 919–944, 2018.

[42] Y. T. Chen, R. Gopinath, A. Tadakamalla, M. D. Ernst, R. Holmes, G. Fraser, P. Ammann, and R. Just, "Revisiting the relationship between fault detection, test adequacy criteria, and test set size," in *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, 2020, pp. 237–249.

[43] C. Birchler, N. Ganz, S. Khatiri, A. Gambi, and S. Panichella, "Cost-effective simulation-based test selection in self-driving cars software with sdc-scissor," in *29th IEEE International Conference on Software Analysis, Evolution, and Reengineering, Honolulu, USA (online), 15-18 March 2022*. ZHAW Zürcher Hochschule für Angewandte Wissenschaften, 2022.

[44] D. Mondal, H. Hemmati, and S. Durocher, "Exploring test suite diversification and code coverage in multi-objective test case selection," in *2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2015, pp. 1–10.

[45] T. Y. Chen, D. Huang, T. Tse, and Z. Q. Zhou, "Case studies on the selection of useful relations in metamorphic testing," in *Proceedings of the 4th Ibero-American Symposium on Software Engineering and Knowledge Engineering (JIISIC 2004)*. Citeseer, 2004, pp. 569–583.

[46] J. Mayer and R. Guderlei, "An empirical study on the selection of good metamorphic relations," in *30th Annual International Computer Software and Applications Conference (COMPSAC'06)*, vol. 1. IEEE, 2006, pp. 475–484.

[47] Y. Cao, Z. Q. Zhou, and T. Y. Chen, "On the correlation between the effectiveness of metamorphic relations and dissimilarities of test case executions," in *2013 13th International Conference on Quality Software*. IEEE, 2013, pp. 153–162.

[48] M. Srinivasan and U. Kanewala, "Metamorphic relation prioritization for effective regression testing," *Journal of Software: Testing, Verification and Reliability*, 2022.

[49] A. C. Barus, T. Y. Chen, F.-C. Kuo, H. Liu, and H. W. Schmidt, "The impact of source test case selection on the effectiveness of metamorphic testing," in *2016 IEEE/ACM 1st International Workshop on Metamorphic Testing (MET)*. IEEE, 2016, pp. 5–11.

[50] E. Alatawi, T. Miller, and H. Søndergaard, "Generating source inputs for metamorphic testing using dynamic symbolic execution," in *Proceedings of the 1st International Workshop on Metamorphic Testing*, 2016, pp. 19–25.

[51] P. Saha and U. Kanewala, "Fault detection effectiveness of source test case generation strategies for metamorphic testing," in *Proceedings of the 3rd International Workshop on Metamorphic Testing*, 2018, pp. 2–9.