Employing Multi-Objective Search and Machine Learning to Mine Cross Product Line Rules: A Technical Report

Safdar Aqeel Safdar[1,2], Tao Yue[1], Shaukat Ali[1], Hong Lu[1]
[1]Simula Research Laboratory, Oslo, Norway
[2]University of Oslo, Oslo, Norway
{safdar, tao, shaukat, honglu}@simula.no

ABSTRACT

Modern systems are being developed by integrating multiple products within/across product lines that communicate with each other through information networks. Runtime behaviors of such systems are related to product configurations and information networks. Cost-effectively supporting Product Line Engineering (PLE) of such systems is challenging mainly because of lacking the support of automation of the configuration process. Capturing rules is the key for automating the configuration process in PLE. However, there does not exist explicitly-specified rules constraining configurable parameter values of such products and product lines. Manually specifying such rules is tedious and time-consuming. To address this challenge, in this paper, we present an improved version (named as *SBRM*$^+$) of our previously proposed Search-based Rule Mining (*SBRM)* approach. *SBRM*$^+$ incorporates two machine learning algorithms (i.e., C4.5 and PART) and two multi-objective search algorithms (i.e., NSGA-II and NSGA-III), employs a clustering algorithm (i.e., *k*-means) for classifying rules as high or low confidence rules, which are used for defining three objectives to guide the search. To evaluate *SBRM*$^+$ (i.e., $SBRM^+_{NSGA\text{-}II}$-*C45*, $SBRM^+_{NSGA\text{-}III}$-*C45*, $SBRM^+_{NSGA\text{-}II}$-*PART*, and $SBRM^+_{NSGA\text{-}III}$-*PART*), we performed two case studies (Cisco and Jitsi) and conducted three types of analyses of results: *difference analysis*, *correlation analysis*, and *trend analysis*. Results of the analyses show that all the *SBRM*$^+$ approaches performed significantly better than two Random Search-based approaches (*RBRM*$^+$*-C45* and *RBRM*$^+$*-PART*) in terms of fitness values, six quality indicators, and 17 machine learning quality measurements (MLQMs). As compared to *RBRM*$^+$ approaches, *SBRM*$^+$ approaches have improved the quality of rules based on MLQMs up to 27% for the Cisco case study and 28% for the Jitsi case study.

KEYWORDS

Product Line; Configuration; Rule Mining; Multi-Objective Search; Machine Learning; Interacting Products

1. INTRODUCTION

Product Line Engineering (PLE) is a well-acknowledged paradigm to improve the productivity of developing products with higher quality and at a lower cost. By benefiting from PLE, more and more systems are developed by integrating different products, which often belong to different product lines, and communicate and interact with each other through information networks [1, 2]. An example of such systems is video conferencing systems [3] (VCSs). These systems are highly configurable as each product has a large number of configurable parameters (e.g., a VCS product developed by Cisco[1] can have more than 120 configurable parameters) offering different configuration options to users (Figure 1). For example, in case of VCSs, users can select different protocols for making a call. Each product has a set of operations that enable it to communicate/interact with other products (Figure 1). Each product has state variables for defining system states. At runtime, configured products belonging to multiple product lines communicate (e.g., via information networks) with each other [1, 2] (Figure 1). Thus, runtime behaviors of such systems not only depend on the configuration of these communicating products but are also influenced by information networks (also named as communication medium). Note that configuration in our context is about assigning a set of values to configurable parameters (i.e., including communication medium specific) of communicating products.

Cost-effective PLE is challenging mainly because of the lack of the support for automation of the configuration process [4, 5]. Capturing rules is the key to enabling automation of various configuration functionalities (e.g., consistency checking,

[1] www.cisco.com/c/en/us/products/collaboration-endpoints/index.html

decision propagation, and decision ordering) [6-9]. In our context, such rules describe how configurations of communicating products within/across product lines impact their runtime interactions via information networks. We name such rules as Cross Product Lines (CPL) rules. CPL rules are important for two reasons. First, they can be used to identify invalid configurations where products may fail to interact due to, for example, violated dependencies among features of interacting products [10]. Identified invalid configurations can help developers to maintain current product lines or develop future product lines. Second, CPL rules can provide support for enabling (automated or semi-automated) configurations of products of future deployments. However, the literature does not provide sufficient support to mine such rules, as it mainly focuses on mining rules constraining product configurations within a single product line [5, 11].

As mentioned in [10], rules (i.e., configuration constraints) can be identified from either domain knowledge or testing of the system. Manually specifying such rules based on domain knowledge is tedious and time-consuming, and heavily relies on experts' knowledge of the domain. Moreover, certain information is only known at runtime (e.g., network related information such as bandwidth, traffic congestion , and maximum transmission unit (MTU) size) [10], which makes CPL rules only possible to be captured at runtime. Identifying CPL rules via testing has its own challenges, as the configuration space is typically very large and testing all possible configurations is infeasible. Besides, in practice, testers often use valid configurations to test a system [10]. Therefore, identifying CPL rules requires an automated approach without exhaustively exploring all possible configurations of communicating products within/across product lines.

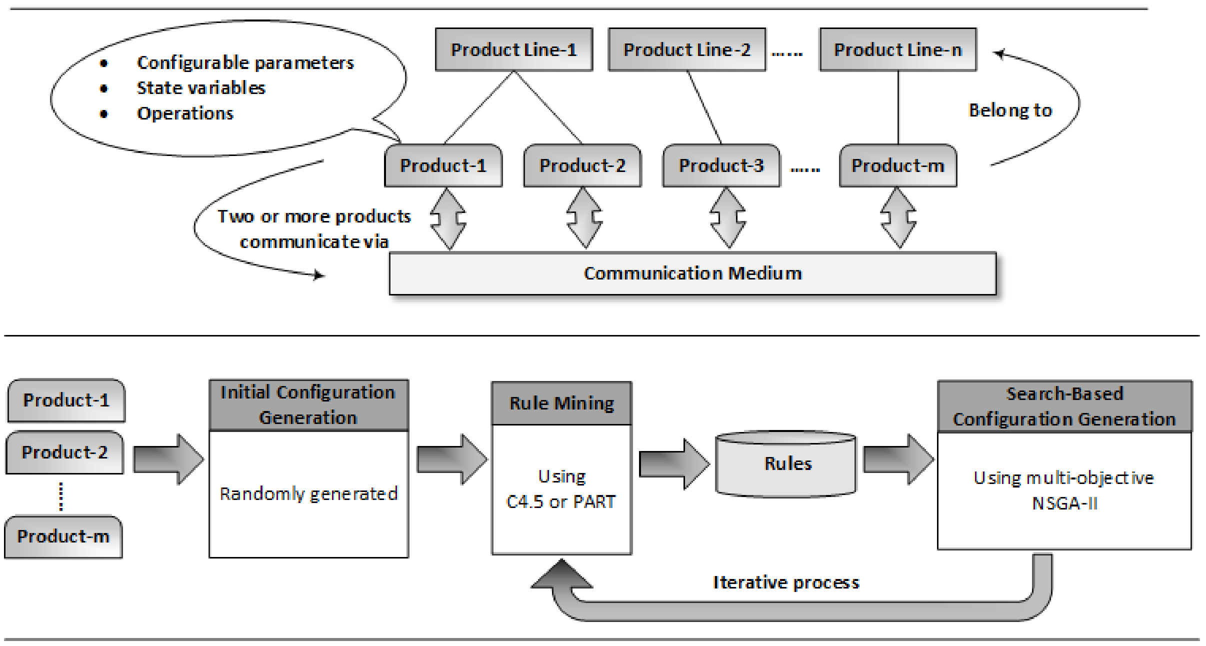


Figure 1:The overall context and scope of SBRM and SBRM$^+$

In [11], a rule mining approach was proposed to mine rules for a product line where product configurations are generated randomly and labeled as faulty or non-faulty. Labeled product configurations are the input to the classification algorithm of J48 [12] to mine rules. However, randomly generating configurations to mine rules is inefficient, as rules with all classes are not equally important (i.e., rules with faulty classes are more important than non-faulty ones). We advanced one step further by employing search to generate product configurations with three heuristics and our initial investigation was presented in our previous work [13], where we proposed an approach, named as Search-based Rule Mining (*SBRM*), combining multi-objective search with machine-learning to mine CPL rules. The three heuristics aim to generate configurations maximally violating high confidence rules with non-faulty classes and satisfying low confidence rules with non-faulty classes and rules with faulty classes). *SBRM* has three major components (Figure 1): 1) *Initial Configuration Generation*: randomly generating an initial set of configurations for communicating products; 2) *Rule Mining*: taking the generated configurations as input along with corresponding system states and applying the machine learning algorithm to mine CPL rules; and 3) *Search-based Configuration Generation*: taking the mined CPL rules as input and generating an another set of configurations using multi-objective search algorithm, which are combined with the previously generated configurations to mine a refined set of CPL rules. *SBRM* obtains CPL rules with different degrees of confidence (i.e., the probability of being correct) with an emphasis on mining rules that can reveal invalid configurations by specifying the configurations that may lead to abnormal (i.e., unwanted) system states [14]. Instead of collecting a large amount of data required for machine learning all at once, we obtain input data incrementally with multiple iterations. During each iteration, we use rules mined from the previous iteration to guide the search for generating configurations. Newly generated configurations are combined with those from all the previous iterations to incrementally refine the aforementioned rules.

In our previous investigation [13], we applied the PART algorithm as the learning algorithm and NSGA-II as the search algorithm in *SBRM*. We validated the approach using a relatively small-sized real-world case study of two communicating VCS products belonging to two different product lines with 17 configurable parameters. In this paper, we extend the prior work by making several additional contributions:

- A significantly improved version of *SBRM* (called $SBRM^+$) is proposed for mining CPL rules constraining configurations of communicating products across/within product lines.
  - A clustering algorithm (i.e., *k*-means) is employed in $SBRM^+$ (as compared to using thresholds in *SBRM*) for classifying rules as high and low confidence rules, which are used for defining the three heuristics/objectives.
  - Two multi-objective search algorithms NSGA-II and NSGA-III are integrated into $SBRM^+$, whereas in *SBRM*, we used only NSGA-II.
  - Two decision tree based rule mining algorithms PART and C4.5 are integrated into $SBRM^+$ (referred as $SBRM^+_{NSGA\text{-}II}\text{-}C45$, $SBRM^+_{NSGA\text{-}III}\text{-}C45$, $SBRM^+_{NSGA\text{-}II}\text{-}PART$, and $SBRM^+_{NSGA\text{-}III}\text{-}PART$ in the rest of the paper), whereas in *SBRM*, we used only PART.
- The $SBRM^+$ approaches were evaluated by performing a real-world case study of three communicating VCS products belonging to three different product lines (Cisco) with 27 configurable parameters and a real-world open source case study of three products of Audio/Video Internet Phone and Instant Messenger, belonging to the same product line (Jitsi) with 39 configurable parameters. Note that evaluation results presented in this paper are based on new experiments conducted using two relatively larger case studies (with 27 and 39 configurable parameters) of three communicating products and no results were taken from our previous work [13].
- We conducted three types of analyses of results for both case studies: *difference analysis*, *correlation analysis*, and *trend analysis*.
  - We conducted *difference analysis* to compare the performance of NSGA-II and NSGA-III combined with PART and C4.5 with Random Search (RS) combined with PART and C4.5 in terms of fitness values, six commonly used quality indicators (i.e., *Hyper Volume* (*HV*), Inverted Generational Distance (*IGD*), Epsilon (ϵ), Euclidean Distance from the Ideal Solution (*ED*), Generational Distance (*GD*), and Generalized Spread (*GS*)), and 17 machine learning quality measurements (MLQMs), in comparison with the prior work where we evaluated NSGA-II and RS using fitness values, *HV*, and six MLQMs. Furthermore, we have also compared the performance of the four $SBRM^+$ approaches in terms of the 17 MLQMs to identify the best-suited approach for mining CPL rules.
  - We conducted *correlation analysis* to study the correlation between the quality of rules in terms of MLQMs and average fitness values and quality indicators, which was not performed in our prior work.
  - We conducted *trend analysis* to see the trend in the quality of rules based on MLQMs across different iterations of $SBRM^+$ (also not performed in the prior work).
- We also significantly extended the related work.

Evaluation results show that $SBRM^+$ is effective to produce high-quality rules as compared to RS based rule mining approach (i.e., $RBRM^+$), as in 7 out of 8 comparisons for the two case studies, the $SBRM^+$ approaches significantly outperformed the $RBRM^+$ approaches in terms of the majority of MLQMs. In eighth comparison, neither one of the two approaches dominates other. Among the four $SBRM^+$ approaches, $SBRM^+_{NSGA\text{-}II}\text{-}C45$ produced the highest quality rules based on MLQMs for the Cisco case study and $SBRM^+_{NSGA\text{-}II}\text{-}PART$ for the Jitsi case study. *Correlation analysis* suggests that in most of the cases lower average fitness values, lower values of quality indicators (except for *HV*) and higher *HV* values mean overall higher quality rules in terms of MLQMs. Moreover, *trend analysis* shows an increasing trend of the quality of rules in terms of MLQMs for all the four $SBRM^+$ approaches across the five iterations.

The rest of the paper is organized as follows: Section 2 provides the background knowledge. In Section 3, we give an overview of $SBRM^+$ followed by the search-based approach for generating configurations in Section 4. In Section 5, we present experiment design and execution. In Section 6, we present results and analyses, followed by the overall discussion and threats to validity. Section 7 summarizes the literature review, and finally, in Section 8, we conclude the paper.

2. Background

In this section, we briefly introduce relevant knowledge on multi-objective search (Section 2.1), machine learning techniques for rule mining and clustering (Section 2.2), and branch distance calculation heuristic (Section 2.3).

2.1 Multi-objective Search

Multi-objective search has been widely applied to address different software engineering optimization problems such as test case prioritization, cost estimation, and configuration generation [15-18]. Multi-objective search algorithms are designed to solve problems where various objectives are competing with each other, and no single optimal solution exists. They aim to find a set of non-dominated solutions for trading off different objectives.

To address our problem, we selected the most commonly used Non-dominated Sorting Genetic Algorithm (NSGA-II) [19, 20], which has proven to be effective for solving various software engineering problems such as test case prioritization and cost estimation [20, 21]. NSGA-II relies on the Pareto dominance theory, which yields a set of non-dominated solutions for multiple objectives [19]. At first, candidate solutions (i.e., the population) are sorted into various non-dominated fronts using a ranking algorithm. Then, individual solutions are selected from non-dominated fronts based on the crowd distance, which measures the distance between the individual solutions and the rest of the solutions in the population [22]. If two solutions belong to the same non-dominated front, then the solution with a higher crowd distance will be selected to increase the diversity of solutions.

We also selected a relatively new multi-objective algorithm NSGA-III [23, 24], which has shown to perform better than NSGA-II in several contexts [25]. The basic working procedure of NSGA-III is quite similar to the NSGA-II but with significant changes in its selection operator. Unlike NSGA-II, NSGA-III's selection process utilizes well-spread reference points to apply the selection pressure to maintain the diversity among population members. We use Random Search (RS) as the comparison baseline.

2.2 Machine Learning

Machine learning is typically used for classifying, clustering, and identifying/predicting patterns in data [26]. It has also been used for inferring rules [11, 27]. Machine learning techniques can be classified into supervised learning (i.e., for labeled data) and unsupervised learning (i.e., for unlabeled data). Supervised learning aims to find relations between input data and its outcome whereas unsupervised learning is for identifying hidden patterns inside input data without labeled responses. We adopted supervised learning in our approach, as we aim to find rules between product configurations (i.e., input) and system states indicating the states of products' interaction (i.e., outcome).

There are two major paradigms of rule generation: 1) creating rules from decision trees, converting the trees into rules and pruning them as opted by C4.5 [28]; 2) employing the separate-and-conquer rule learning technique used by Repeated Incremental Pruning to Produce Error Reduction [29]. Creating rules from decision trees is computationally expensive in the presence of noisy data, and the separate-and-conquer rule learning technique has the over pruning (hasty generalization) problem [30]. The Pruning Rule-Based Classification algorithm (PART) combines the two paradigms mentioned above of rule generation while avoiding their shortcomings. PART generates partial decision trees, and corresponding to each partial tree, a single rule is extracted for the branch that covers maximum nodes [30]. Therefore, in our previous study [13], we opted for PART. In this study, we also included C4.5, as it is the most popular algorithm in the research community as well as industry [31].

We used Lloyd's algorithm [32] for clustering rules, a commonly used *k*-means algorithm, which minimizes the average squared distance between points within the same cluster. Initially, it selects *k* data points randomly as centers of *k* clusters. Furthermore, it uses the Euclidean distance function [33] to calculate the distances between each data point and centers of *k* clusters, and assign each data point to its nearest cluster. After assigning all the data points to *k* clusters, it updates the centers of *k* clusters by calculating the mean of all the data points within each cluster. Once centers are updated, it recalculates the Euclidean distance for all the data points and reassigns them to *k* clusters. This process continues until the centers of *k* cluster do not change in two consecutive iterations.

2.3 Branch Distance Calculation Heuristic

Branch distance is a heuristic used in search-based software engineering, which indicates to what extent the given data satisfy the predicate (aka condition or clause) in the rule/constraint. For measuring the branch distance between a configuration of a configurable parameter and a predicate in the rule, we opted the branch distance calculation approach provided in [34, 35]. In Table 1, we summarize the distance calculation formula corresponding to different operations for numerical and enumerated data.

Table 1: Branch distance functions [34] *

Predicate type	Operation	Distance function
Predicates with relational operators	a=b	0
	a!=b	a!=b → 0 **else** nor(|a−b| +1) *k
Predicate with a Boolean condition		True → 0 **else** k
Logical connective of two predicates	$Pr_1 \wedge Pr_2$	$Pr_1 + Pr_2$ (sum of branch distances for both predicates)

* *k* is a positive constant greater than zero, we used *k*=1; *nor* gives a normalized value between zero and one.

3. OVERVIEW

Figure 2 presents an overview of *SBRM*$^+$, which relies on machine learning and multi-objective search to mine CPL rules in an iterative and incremental process. As shown in Figure 2, the whole process consists of seven steps, which are organized into four types of activities: *Generation*, *Execution*, *Mining*, and *Clustering*. *Generation* related steps (i.e., Steps 1 and 5) are about generating configurations for the selected products within/across product lines, using a search algorithm (e.g., NSGA-II) or RS. *Execution*-related steps (i.e., Steps 2 and 6) configure the selected products with generated configurations and obtaining their consequent system states to label the configurations. *Mining*-related steps (i.e., Steps 3 and 7) combine all generated configurations with system states and apply machine learning algorithms (e.g., PART) to mine rules. *Clustering* Step 4 clusters and classifies the mined rules into categories with a clustering algorithm (e.g., *k*-means). Based on the categories, we defined three search objectives for guiding the search for generating configurations (Step 5).

As shown in Figure 2, in the first step, an initial set of configurations is randomly generated for configurable parameters of the selected products within/across product lines. The second step obtains the system states indicating the states of interaction among selected products. During the second step, we configure the selected products with randomly generated configurations, execute certain functionalities to enable the communication among the selected products, and capture the system states to know if the products communicated successfully (as intended). In our context, an *interaction* can be defined as a communication between two or more products communicating via a communication medium. An interaction can be enabled by executing certain functionalities (i.e., a sequence of operations) of the communicating products to make them communicate with each other.

In step 3, we feed the set of generated configurations (as Attributes) and their corresponding system states (as Classes) to Weka [12] as the initial input and apply a rule mining algorithm (e.g., PART or C4.5) to mine the initial set of rules. Normally, a classifier (e.g., C4.5 and PART) trains a model using a training dataset and then validates the model using a test dataset. In our case, the input configurations are used as the training dataset. For validation, we used 10 times 10-fold stratified cross-validation as it presents all classes (approximately) equally across each test fold [31, 36]. This means both PART and C4.5 use 10% of the training data (i.e., generated configurations with corresponding system states provided as input) in each test fold to validate the model. Note, PART gives a set of rules as the outcome. However, C4.5 gives a decision tree, where a non-leaf node in a branch represents a predicate specifying the configuration value for a particular configurable parameter and the leaf node represents the predicted Class (e.g., the call status *ConnectedConnected* in our context). From each branch of a generated tree, we extract a rule by joining all non-leaf nodes in the branch with the AND operator to form the antecedent of the rule and using the leaf node as its consequent. We provide the code for extracting the rules from the tree in the Bitbucket repository[2].

In step 4, the mined rules are clustered using the *k*-means clustering algorithm (Section 2.2) and classified into different categories. The classified rules are fed to NSGA-II or NSGA-III for generating configurations for the next iteration in step 5. In step 6, we repeat step 2 but take the configurations generated from the search instead of the random one. In step 7, we

[2] https://bitbucket.org/safdaraqeel/ase-ruleextraction

combine all the configurations generated from steps 1 and 5 and collected system states captured from steps 2 and 6, and feed all the data to Weka to mine a refined set of rules. This rule set is then used in the next iteration (starting from step 4) to generate more configurations and further refine the rules.

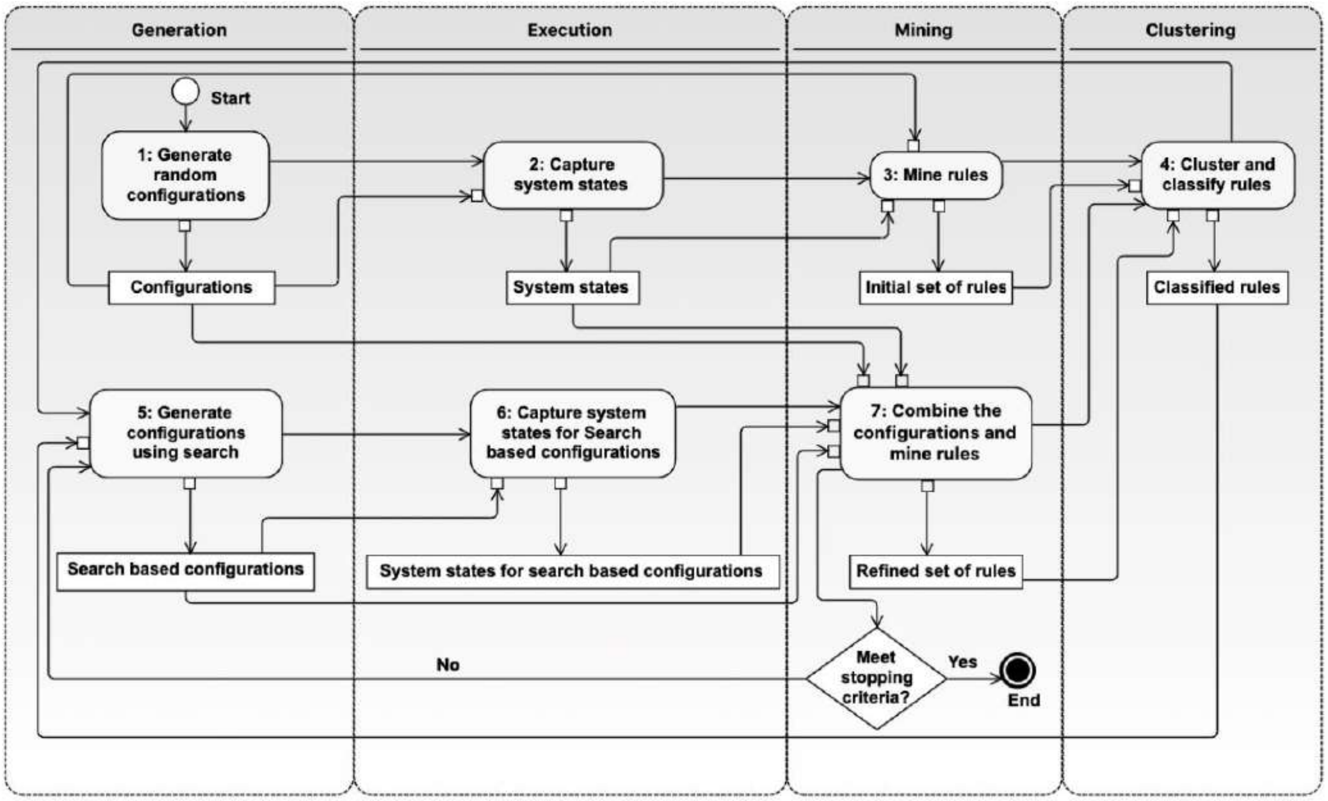

Generation
Execution
Mining
Clustering
Start
1: Generate random configurations
2: Capture system states
3: Mine rules
4: Cluster and classify rules
Configurations
System states
Initial set of rules
Classified rules
5: Generate configurations using search
6: Capture system states for Search based configurations
7: Combine the configurations and mine rules
Search based configurations
System states for search based configurations
Refined set of rules
Meet stopping criteria?
No
Yes
End

Figure 2: Overview of the proposed approach (SBRM$^+$)

In each iteration, newly generated configurations with collected system states are added to the dataset from the previous iteration to mine a new set of rules. We repeat the process (step 4 to step 7) until we meet the stopping criteria, e.g., a fixed number of iterations and/or when the rules mined from two consecutive iterations are similar. We used a fixed number of iterations in our experiments, as we have limited available resources for mining rules. Getting similar rules from consecutive iterations indicates that it is very unlikely to refine the rules further. All the iterations (e.g., five iterations in our experiment) used for refining the rules and repeated before meeting the stopping criteria make a complete cycle. We consider step 4 (i.e., classification of rules) and step 5 (i.e., using search to generate configurations), as the innovative part of the whole approach, i.e., *SBRM$^+$*. This is because using Weka to mine rules is an application of the rule mining algorithm (e.g., PART or C4.5), but applying search requires carefully designing a fitness function. Similarly, classification of rules requires applying the *k*-means clustering algorithm based on certain attributes, ranking the clusters using specific formula and classify the rules into different categories, which are consequently fed to the search algorithm (e.g., NSGA-III) to generate configurations. Both steps 4 and 5 are discussed in detail in the following section.

Pseudocode 1 is the pseudocode of SBRM$^+$, where in L1, we encode the configuration generation problem by representing all the configurable parameters as numerical variables (Integer or Real) and restricting their domains by defining their upper and lower limits. In L2-L5 (i.e., Zero-Iteration), we generate the initial set of configurations randomly, decode them, and mine the initial set of rules. Similarly, in L6-L19, we cluster and classify the rules (L7), generate configurations (L8) using the search (e.g., NSGA-II), decode the configurations (L9), and mine the refined set of rules (L10-L11). Note, encoding and decoding are discussed in detail in Section 4.3, whereas the mining and clustering are introduced in Section 4.2.

```
Input: A set of $n$ configurable parameters $CP = \{cp_1, cp_2, .., cp_n\}$ with their sets of possible values $CPV = \{CPV_1, CPV_2, .., CPV_n\}$, Number of intial randomly generated configurations $NC_{RG}$, Number of iterations $NI$, and Number of configurations to be generated per iteration $NC_{PI}$, a set of parameters for search algorithm $P_{SA}$
Output: A set of rules $RRS$
Begin

// ………………… Encoding Configuration Generation Problem …………….
L1.  $ECP, UpperLimits, LowerLimits \leftarrow$ Encode_Configurations_Generation_Problem ($CP$, $CPV$)

// ………………… Generating Initial Set of Rules based on Randomly Generated Configurations…………….
L2.  $I := 0$          // Zero-Iteration where we use configurations generated randomly
L3.  $EC_{RG} \leftarrow$ Generate_Configurations_Randomly ($ECP, UpperLimits, LowerLimits, NC_{RG}$)
L4.  $DC_{RG} \leftarrow$ Decode_Configurations ($EC_{RG}, UpperLimits, LowerLimits$)
L5.  $RS_{Initial} \leftarrow$ Mine_Initial_RuleSet ($DC_{RG}$)

// ……… Generating Refined Set of Rules Based on Configurations Generated Using Search Algorithms………
L6.  $I := I + 1$          // First iteration where we use configurations generated by search algorithms
L7.  $CRS_{Initial} \leftarrow$ Cluster_And_Catergorize_Rules($RS_{Initial}$)
L8.  $EC_{SBG} \leftarrow$ Generate_Configurations_Using_Search ($ECP, UpperLimits, LowerLimits, NC_{PI}, CRS_{Initial}, P_{SA}$)
L9.  $DC_{SBG} \leftarrow$ Decode_Configurations ($EC_{SBG}, UpperLimits, LowerLimits$)
L10. $DC_{RG+SBG} \leftarrow$ Combine_Configurations ($DC_{RG}, DC_{SBG}$)
L11. $RS_{Refined} \leftarrow$ Mine_Refine_RuleSet ($DC_{SBG+RG}$)

L12. while ($I \leq NI$) do
L13.     $I := I + 1$
L14.     $CRS_{Refined} \leftarrow$ Cluster_And_Classify_Rules($RS_{Refined}$)
L15.     $EC_{SBG} \leftarrow$ Generate_Configurations_Using_Search ($ECP, UpperLimits, LowerLimits, NC_{PI}, CRS_{Refined}, P_{SA}$)
L16.     $DC_{SBG} \leftarrow$ Decode_Configurations ($EC_{SBG}, UpperLimits, LowerLimits$)
L17.     $DC_{RG+SBG} \leftarrow$ Combine_Configurations ($DC_{RG+SBG}, DC_{SBG}$)
L18.     $RS_{Refined} \leftarrow$ Mine_Refine_RuleSet ($DC_{RG+SBG}$)

L19. return $RS_{Refined}$
```

Pseudocode 1: Search Based Rule Mining (SBRM$^+$)

4. SEARCH-BASED CONFIGURATION GENERATION APPROACH

Sections 4.1 presents formal definitions required to define the configuration generation problem. In Section 4.2, we present details about the classification of CPL rules whereas solution encoding in Section 4.3. Section 4.4 presents the objectives and effectiveness measures, followed by the fitness function in Section 4.5.

4.1 Formalization of Configuration Generation Problem

We formalize relevant concepts and exemplify them with an example of three communicating VCS products belonging to two different product lines (Figure 4). The definitions, formal representations, and examples of the concepts related to the product lines and rule mining are presented in Table 2. Moreover, we also constructed a class diagram shown in Figure 3 to conceptually describe how the defined concepts are related to each other.

As shown in Figure 3, each product line has two or more products, which are communicating via a communication medium (e.g., Wired Internet, Wireless Internet, Bluetooth). A product has one or more configurable parameters, state variables, and operations. Each configurable parameter has two or more configurable parameter values. Similarly, each state variable has two or more state values. An operation can take zero or more operation parameters as input, where each operation parameter has two or more operation parameter values. The operation parameter values assigned to the operation parameters of the operation may affect the behavior of the operation. Different products can communicate with each other by enabling a particular interaction. Enabling a particular interaction requires executing a sequence of operations belonging to one or more communicating products. State rules defined on state variables of the communicating products can be used to define the system states, which indicate whether products interact/communicate successfully (as intended). The configurable parameter values assigned to the configurable parameters of the communicating products determine the success of the interaction. Moreover, the communication medium may also influence the interaction. An interaction has at least one source product and one or more target products. An interaction is homogeneous if the communicating products belong to the same product line otherwise heterogeneous. The communication between products enabled by an interaction can be unidirectional or bidirectional.

In Figure 4**,** *VCS-PL1* and *VCS-PL2* are two VCS product lines. *VCS1*, *VCS2*, and *VCS3* are three products communicating through *WiredInternet*, where *VCS1* and *VCS2* belong to *VCS-PL1*, and *VCS3* belongs to *VCS-PL2*. *VCS1* has three configurable parameters (e.g., *VCS1.defaultProtocol*), three state variables (e.g., *VCS1.callStatus*), and five operations (e.g., *VCS1.dial()*). Similarly, *VCS2* has three configurable parameters, one state variable, and three operations whereas *VCS3* has four configurable parameters, one state variables, and three operations. *dial()* operation of all three VCS products has three operation parameters including *protocol*, *callRate*, and *callType*.

Table 2. Formalization of concepts*

Def#	Concept	Definition and formal representation with examples
1	Product line	A product line can be defined as a set of products sharing explicitly defined and managed common and variable features and relying on the same domain architecture. A set of npl product lines for a particular application domain can be presented as: $PL = \{pl_1, pl_2, .., pl_{npl}\}$, where pl_i represents the *ith* product line. For example, {*VCS-PL1*, *VCS-PL2*} is a set of two product lines.
2	Product	A product can be defined as a triplet (CP, SV, OP), where CP, SV, and OP are sets of configurable parameters, state variables, and operations. A set of inp products for a product line pl_i can be presented as: $P_i = \{p_{i1}, p_{i2}, .., p_{inp}\}$, where p_{ij}represents the *jth* product of pl_i. For example, {*VCS1, VCS2*} represent a set of products for *VCS-PL1*.
3	Configurable parameter	A configurable parameter is a numerical (e.g., integer, real) or non-numerical (e.g., binary, ordinal, nominal) type variable, which can take different values [37]. The possible values of a numerical type configurable parameter can be specified by defining the constraints on its upper and lower limits whereas, for a non-numerical type configurable parameter, they can be specified as a set of predefined values. A set of $incp$ configurable parameters for a product p_i can be presented as: $CP_i = \{cp_{i1}, cp_{i2}, .., cp_{incp}\}$, where cp_{ij} represents the *jth* configurable parameter of p_i. For example, {*defaultProtocol, encryption, defaultCallRate*} is a set of configurable parameters of product *VCS1*, where *defaultProtocol* and *encryption* are non-numerical and *defaultCallRate* a numerical type configurable parameters.
4	Configurable parameter value	A configurable parameter value is a value that can be assigned to a configurable parameter. A set of $incpv$ configurable parameter values for each cp_i can be presented as: $CPV_i = \{cpv_{i1}, cpv_{i2}, .., cpv_{incpv}\}$, where cpv_{ij} represents the *jth* value of cp_i. For a real type configurable parameter, $incpv$ will be infinity because such configurable parameter can take infinite values between its upper and lower limits. For example, *defaultProtocol* can be configured with *SIP*, *H323*, and *AIM* whereas *defaultCallRate* can take a value between 64 to 6000.
5	State variable	State variables are variables used to describe the state of the system. A set of $insv$ state variables for a product p_i can be presented as $SV_i = \{sv_{i1}, sv_{i2}, .., sv_{insv}\}$, where sv_{ij} represents the *jth* state variable for the product p_i. For example, {*callStatus, numberOfActiveCalls, isPresentationShared*} is a set of state variables of *VCS1*.
6	State value	A state value is a possible value of a state variable, specific to one product. A set of inv possible state values for a state variable sv_i, can be presented as $V_i = \{v_{i1}, v_{i2}, .., v_{inv}\}$, where v_{ij} represents the *jth* state value of sv_i. For example, {*Connected, Failed*} represents a set of state values for *callStatus* of *VCS1*.
7	Operation	An operation is a function implemented in a product. A set of $inop$ operations for a product p_i can be presented as: $OP_i = \{op_{i1}, op_{i2}, .., op_{inop}\}$, where op_{ij} represents the *jth* operation of p_i. For example, {*dial(), accept(), disconnect(), startPresentation(), stopPresentation()*} is a set of operations for *VCS1*.
8	Operation parameter	An operation parameter is a variable of numerical (e.g., integer, real) or non-numerical (e.g., binary, ordinal, nominal) type, provided as input to an operation. A set of $inpm$ parameters for an operation op_i can be presented as $PM_i = \{pm_{i1}, pm_{i2}, .., pm_{inpm}\}$, where pm_{ij} represents the *jth* operation parameter for the operation op_i. For example, {*protocol, callRate, callType*} represents a set of operation parameters for *dial()* operation of *VCS1*.
9	Operation parameter value	An operation parameter value is a value that can be assigned to an operation parameter. A set of $inpv$ parameter values for an operation parameter pm_i can be presented as $PV_i = \{pv_{i1}, pv_{i2}, .., pv_{inpv}\}$, where pv_{ij} represents the *jth* operation parameter value of pm_i. For example, {*Audio, Video*} represents a set of operation parameter values for *callType* operation parameter corresponding to the *dial()* operation of *VCS1*.
10	Interaction	An interaction is communication between at least one source product and one or more target products communicating via a communication medium, enabled by a sequence of operations belonging to the source and target products. A set of $inin$ interactions supported by a product p_i to communicate with other products can be presented as: $IN_i = \{in_{i1}, in_{i2}, .., in_{inin}\}$, where in_{ij} represents the *jth* interaction supported by product p_i. For example, {*making-call, sharing-presentation*} represents a set of interactions supported by *VCS1*.
11	Selected products	A set of nsp communicating products under study can be presented as: $SP = \{p_1, p_2, .., p_{nsp}\}$, where p_i represents the *ith* product in the set of communicating products. Such products may belong to different product lines or same product line. For example, SP = {*VCS1, VCS2, VCS3*} represented a set of selected products where *VCS1* and *VCS2* belong to *VCS-PL1*, and *VCS3* belongs to *VCS-PL2*.
12	Selected interactions	A set of $nsin$ selected interactions for the selected products can be defined as: $SIN_{SP} = \{in_1, in_2, .., in_{nsin}\}$, where in_j represents the *jth* selected interaction. For example, SIN_{SP} = {*making-call*} represents the set of selected interactions.
13	Selected operations	For each selected interaction in_i, a sequence of operations required to enable interaction in_i can be defined as: $OPS_i = (op_{i1}, op_{i2}, .., op_{insop})$, where op_{ij} represents the *jth* operation (in order) required to enable interaction in_i. For example, (*VCS1.dial(), VCS2.accept(), VCS3.accept(), VCS1.disconnect()*) represents the sequence of operations required to enable *making-call* interaction.

14	Selected configurable parameters	A set of $nscp$ selected configurable parameters for all the selected products can be defined as: $SCP_{SP} = \{cp_1, cp_2, .., cp_{nscp}\}$, where cp_i represents the ith configurable parameter of the selected product. For example, SCP_{SP}={*VCS1.defaultProtocol*, *VCS1.defaultCallRate*, *VCS1.encryption*, *VCS2.encryption*, *VCS3.encryption*} represents the set of selected configurable parameters for the selected products.
15	Selected state variables	A set of $nssv$ selected state variables for all the selected products related to the selected interaction can be defined as $SSV_{SP} = \{sv_1, sv_2, .., sv_{nssv}\}$, where vs_i represents the ith state variable. The selected state variables may belong to different products. For example, {*VCS2.callStatus, VCS3.callStatus*} represents the set of selected state variables.
16	System states	System states are the combinatorial states for all the products involved in the selected interactions, which indicate whether products communicated successfully (as intended). Such states are described by defining the state rules on the selected state variables. A set of nis possible system states corresponding to the selected interactions can be defined as: $SS = \{ss_1, ss_2, .., ss_{nis}\}$, where ss_i represents the jth system state. For example, {*ConnectedConnected*, *ConnectedFailed*, *FailedConnected*, *FailedFailed*} represents a set of system states, which are specified by concatenating (state rule) the states values of the selected state variables.
17	Predicate	A predicate is a conditional statement in a rule with one configurable parameter and its value joined by one of the relational operators (i.e., $=, \neq, <, \leq, >, \geq$). For example, "*VCS1.encryption = On"* and "*VCS1. defaultCallRate* > 1000" are two predicates.
18	Rule	In the context of rule mining, a rule with npr predicates can be represented as: $r_i = pr_1 \ AND \ pr_2 \ AND \ ... AND \ pr_{npr} : ss_k$, where pr_j represents the jth predicate of rule r_i and ss_k represents kth system state. For example, r_1**:** "*VCS1.encryption = On* AND *VCS2.encryption = Off* AND *VCS3.encryption* = BestEffort: *ConnectedFailed*" and r_2**:** "*VCS1.encryption = On* AND *VCS2.encryption = BestEffort* AND *VCS3.encryption = On*: *ConnectedConnected*" are two rules.
19	Confidence of a rule	For a rule r_i, $Cf(r_i)$ represents the *confidence* of r_i, which is between 0 and 1. *Confidence* for a rule r_i can be calculated as: $Cf(r_i) = \frac{(SP_i - V_i)}{(SP_i + V_i)}$, where SP_i represents the number of instances for which r_i holds true (i.e., *support*) and V_i represents the number of instances that violate r_i (i.e., *violation*). An instance represents a set of configurable parameter values for the selected configurable parameters of the communicating products and corresponding system state.
20	Configuration solution	A configuration solution $\{s_j\}$ is a set of configurable parameter values assigned to all the selected configurable parameters, which mathematically can be represented as: $s_j = \{cpv_{j1}, .., cpv_{jnscp}\}$, where cpv_{ji} represents the configurable parameter value assigned to the ith configurable parameter (i.e., cp_i) in $\{s_j\}$. For example, {*SIP*, 5000, *On*, *BestEffort*, *Off*} is a configuration solution.
21	Configuration space	A set of ns potential configuration solutions (i.e., configuration space) can be defined as: $S = \{\{s_1\}, \{s_2\}, .., \{s_{ns}\}\}$, where $\{s_i\}$ represents the ith configuration solution. ns can be calculated as the cardinality of the Cartesian product of configurable parameter values' sets for all the selected configurable parameters, which can be represented mathematically as: $\lvert CPV_1 \times .. \times CPV_{nscp} \rvert$. The configuration space for the Cisco's case study contains approximately 1.03e33 configuration solutions and 6.54e60 for the Jitsi case study
22	Effectiveness measures	A set of ne effectiveness measures can be defined as: $E = \{e_1, e_2, .., e_{ne}\}$, where e_i represents the ith effectiveness measure. For example, a set of three effectiveness measures (i.e., AHNS, NLNS, and NAS) defined in Section 4.3
23	Explored solutions	A set of nes configuration solutions explored during the search is a proper subset of S, which mathematically can be represented as: $S_{Ex} = \{\{s_1\}, \{s_2\}, .., \{s_{nes}\}\}$, where $nes < ns$.

* Note: All the examples provided are based on the running example. Also, by selected elements (e.g., products, configurable parameters), we mean elements under study for learning CPL rules.

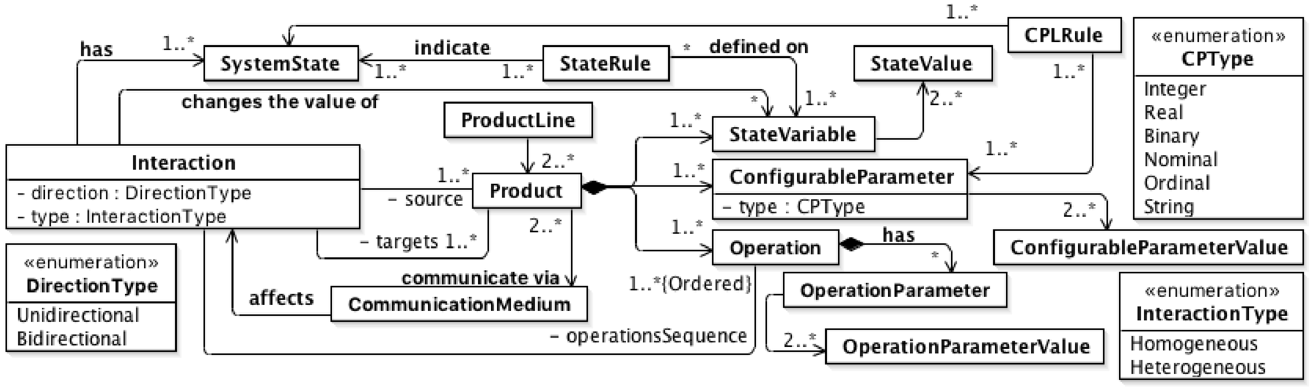


Figure 3. A conceptual model for interacting products

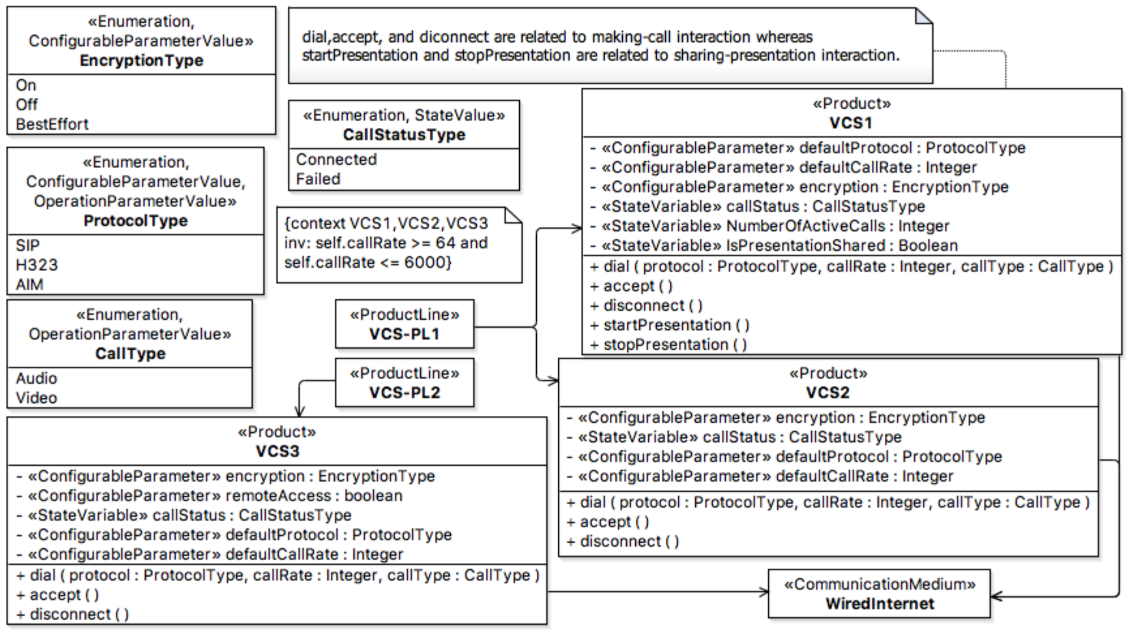

«Enumeration, ConfigurableParameterValue» EncryptionType
On
Off
BestEffort
«Enumeration, ConfigurableParameterValue, OperationParameterValue» ProtocolType
SIP
H323
AIM
«Enumeration, OperationParameterValue» CallType
Audio
Video
«Enumeration, StateValue» CallStatusType
Connected
Failed
dial,accept, and diconnect are related to making-call interaction whereas startPresentation and stopPresentation are related to sharing-presentation interaction.
{context VCS1,VCS2,VCS3 inv: self.callRate >= 64 and self.callRate <= 6000}
«ProductLine» VCS-PL1
«ProductLine» VCS-PL2
«Product» VCS1
- «ConfigurableParameter» defaultProtocol : ProtocolType
- «ConfigurableParameter» defaultCallRate : Integer
- «ConfigurableParameter» encryption : EncryptionType
- «StateVariable» callStatus : CallStatusType
- «StateVariable» NumberOfActiveCalls : Integer
- «StateVariable» IsPresentationShared : Boolean
+ dial (protocol : ProtocolType, callRate : Integer, callType : CallType)
+ accept ( )
+ disconnect ( )
+ startPresentation ( )
+ stopPresentation ( )
«Product» VCS2
- «ConfigurableParameter» encryption : EncryptionType
- «StateVariable» callStatus : CallStatusType
- «ConfigurableParameter» defaultProtocol : ProtocolType
- «ConfigurableParameter» defaultCallRate : Integer
+ dial (protocol : ProtocolType, callRate : Integer, callType : CallType)
+ accept ( )
+ disconnect ( )
«Product» VCS3
- «ConfigurableParameter» encryption : EncryptionType
- «ConfigurableParameter» remoteAccess : boolean
- «StateVariable» callStatus : CallStatusType
- «ConfigurableParameter» defaultProtocol : ProtocolType
- «ConfigurableParameter» defaultCallRate : Integer
+ dial (protocol : ProtocolType, callRate : Integer, callType : CallType)
+ accept ( )
+ disconnect ( )
«CommunicationMedium» WiredInternet

Figure 4. Exemplifying concepts related to the product interaction

Based on the concepts presented in Table 2, our configuration generation problem can be formulated as searching a solution set S_R from a set of explored solutions (i.e., S_{EX}) such that $S_R \subset S_{Ex}$, and all the solutions in S_R have highest effectiveness in terms of effectiveness measures E than all the other explored solutions in $\{S_R \backslash S_{Ex}\}$.

$$\forall_{s_r \in S_R} \forall_{s_i \in S_{Ex}} \forall_{e_j \in \mathrm{E}}\, s_i \notin S_R \,\Lambda\, Effect(s_r, e_j) \geq Effect\,(s_i, e_j)$$
$$\Lambda\, \exists_{e_k \in \mathrm{E}}\; Effect(s_r, e_k) > Effect\,(s_i, e_k) \qquad (1)$$

where $Effect\,(s_i, e_j)$ gives the value of the *j*th effectiveness measure (Section 4.4) for configuration solution s_i.

4.2 Clustering and Classification of CPL Rules

Generally, from the user perspective, the system states (Table 2) can be categorized as normal states and abnormal states. Normal states indicate that interaction was enabled successfully and selected products interacted/communicated successfully as intended whereas abnormal states show that interaction failed and selected products did not interact/communicate successfully. Consequently, CPL rules can be classified into two categories: $R_A = \{r_{a1}, r_{a2}, r_{a3}, \ldots, r_{nar}\}$ for abnormal states (Category-I), where r_{ai} represents the *i*th rule with abnormal state and nar represents the total number of rules with abnormal states; $R_N = \{r_{n1}, r_{n2}, r_{n3}, \ldots, r_{nnr}\}$ is for normal states, where r_{ni} represents the *i*th rule with normal state and nnr represents the total number of rules with normal states.

We apply *k*-means (Section 2.2) to cluster R_N into three clusters based on three attributes of rules: *confidence*, *support*, and *violation*. *Support* and *violation* have a different scale than *confidence*, and generally, clustering algorithm does not work with attributes of different scales [38]. Thus, we divided *support* and *violation* by the sum of maximum support and maximum violation in order to normalize *support* and *violation*. After clustering the rules, we calculate the rank for each cluster as:

$$Rank\ (c_i) = (Support(c_i) + Confidence(c_i) - Violation(c_i)) \qquad (2)$$

Where $Support(c_i)$, $Violation(c_i)$, and $Confidence(c_i)$ are mean values of normalized *support*, *violation*, and *confidence* for all the rules belonging to cluster c_i. Based on the calculated ranks of the three clusters, all the rules are classified into two categories: R_{N1} represents high-confidence rules belonging to a cluster with the highest rank (Category-II) whereas R_{N2} represents low-confidence rules belonging to other two clusters with the lowest and medium ranks (Category-III). In Table 2 (Def# 18), we present two CPL rules r_1 and r_2 where r_1 is a rule with an abnormal state *ConnectedFailed* and r_2 is a rule with a normal state *ConnectedConnected*. For example, r_1 describes that if the encryption of *VCS1* (i.e., *Caller*) is set to be "On", encryption of *VCS2* (i.e., *Callee1*) is set to be "Off", and encryption of *VCS3* (i.e., *Callee2*) is set to be "BestEffort", the conference call will connect to the *VCS2* but will fail to connect with the *VCS3*. Rule r_1 is an abnormal state rule, as the consequent of r_1 (i.e., *ConnectedFailed*) is an abnormal system state. Similarly, r_2 is a normal state rule because its consequent (i.e., *ConnectedConnected*) is a normal system state.

4.3 Solution Encoding and Decoding

As mentioned in Table 2 (Def#3), a configurable parameter can be a numerical (e.g., Integer) or non-numerical (e.g., Boolean, Nominal) type variable. Thus, to apply search algorithms for the configuration generation problem, we encode all the configurable parameters as a vector of integer variables to represent the configuration solutions. Considering three configurable parameters *encryption* (i.e., Nominal), *remoteAccess* (i.e., Boolean), and *callRate* (i.e., Integer) of three communicating products *VCS1*, *VCS2*, and *VCS3* in Figure 5, *encryption* can take one of the three values (*On*, *Off*, and *BestEffort*) and *remoteAccess* can take *True* or *False* whereas *callRate* can take a value from 64 to 6000.

To encode the non-numerical configurable parameters, we map all the configurable parameter values to a sequence of numbers (Figure 5). For example, we mapped *On*, *Off*, and *BestEffort* to *1*, *2*, and *3* respectively in order to encode *encryption*. The configuration solution is represented as a vector of integer variables (i.e., *e_encryption*, *e_remoteAccess*, and *e_callRate*) where each variable represents a particular configurable parameter. For example, *e_encryption* represents *encryption* in Figure 5. To decode a particular configuration solution, we replace the integer values in the vector with the configurable parameter values of corresponding configurable parameters. For example, in Figure 5, we replace values *3* and *1* with *BestEffort* and *True* to get the final decoded configuration solution: <*BestEffort*, *True*, *5000*>.

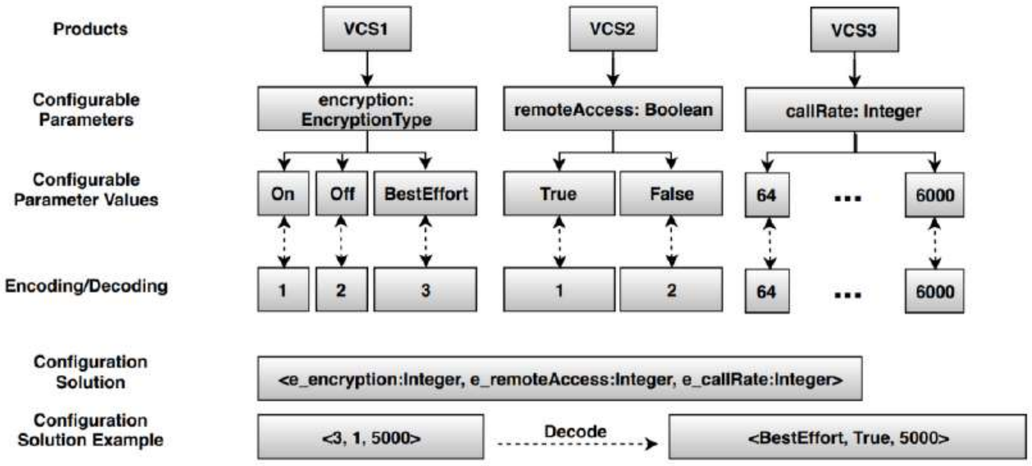


Figure 5: Exemplifying the encoding and decoding mechanism employed in SBRM$^+$

4.4 Objectives and Effectiveness Measures

CPL rules could reveal invalid configurations that lead to unwanted states of the system (i.e., abnormal states) are more important, therefore, the invalid configurations are of more interest. This encouraged us to use the search to generate configurations in a smart way. To be more specific, by applying search heuristics, we embrace configurations under which communicating products may fail to interact/communicate with each other and avoid configurations that lead to successful interactions among products. To achieve this goal, we define three objectives based on the distances between a configuration solution and the three categories of rules (Category-I, Category-II, Category-III). Before presenting the objectives and effectiveness measures, we first define the distance function that is used to assess the effectiveness measures. The distance function indicates to what extent a configuration solution conforms to a rule.

$$D(r_i, s_r) = \frac{\sum_{j=1}^{npr} d(pr_j, cpv_r)}{mnp} \qquad (3)$$

where $D(r_i, s_r)$ calculates the distance between rule r_i and configuration solution s_r . In equation (3), $d(pr_j,\ cpv_r)$ calculates the branch distance between *j*th predicate pr_j from rule r_i and corresponding configurable parameter value cpv_r of the configurable parameter involved in predicate pr_j from configuration solution s_r. npr represents the total number of predicates in rule r_i whereas mnp represents the number of predicates in a rule with the maximum number of predicates. To calculate the distance between pr_j and cpv_r as a branch distance, we use the distance calculation formula provided in [34] (Section 2.3).

Objective-1: This objective is to avoid generating configurations that completely or close to satisfy rules in Category-II. The effectiveness measure $AHNS$ corresponding to this objective can be calculated as:

$$AHNS(R_N, s_r) = \sum_{i=1}^{nnr} Cf(r_i) * D(r_i, s_r) \mid r_i \in R_{N1} \quad (4)$$

where $AHNS(R_N, s_r)$ takes R_N (the set of rules related to the normal states) and one configuration solution s_r as input and gives the effectiveness measure as output. To determine $AHNS$, we calculate the sum of weighted distances for all the rules in Category-II (i.e., R_{N1}), where each rule belongs to the cluster with the highest rank. The weighted distance of r_i is calculated by multiplying $Cf(r_i)$ with $D(r_i, s_r)$.

Objective-2: This objective is to generate configurations within the configuration space that satisfy Category-III (i.e., R_{N2}) as well as its nearby space. The nearby space contains configurations for which the distance to the rules in Category-III is close to 0 but not exactly 0. These configurations might help to either improve the confidence of correct rules by increasing their support or filter out incorrect ones by increasing their violation and hence reducing their confidence. The effectiveness measure $NLNS$ related to the second objective can be calculated as:

$$NLNS(R_N, s_r) = \sum_{i=1}^{nnr} Cf(r_i) * (1 - D(r_i, s_r)) \mid r_i \in R_{N2} \quad (5)$$

where $NLNS(R_N, s_r)$ takes R_N (the set of rules associated with the normal states) and configuration solution s_r as input and outputs $NLNS$. Since we want to explore the configuration space near the configurations satisfying the rules in Category-III, configurations with a smaller distance to the rules in Category-III are preferred. Therefore, we use $(1 - D(r_i, s_r))$ in the $NLNS(R_N, s_r)$. To calculate $NLNS$, we calculate the sum of the weighted distance (i.e., calculated by multiplying $Cf(r_i)$ with $(1 - D(r_i, s_r)))$ of a configuration solution to all the rules in Category-III (i.e., R_{N2}), where each rule belongs to a cluster with middle rank or lowest rank.

Objective-3: This objective is to generate configurations within the configuration space that satisfy Category-I and its nearby space. The rules in Category-I are of high interest in our context because they indicate situations where interactions of the selected products fail. The effectiveness measure NAS for this objective can be calculated as:

$$NAS(R_A, s_r) = \sum_{i=1}^{nar} Cf(r_i) * \left(1 - D(r_i, s_r)\right) \quad (6)$$

where $NAS(R_A, s_r)$ takes rule set R_A (related to the abnormal states) and configuration solution s_r as input. To calculate NAS, we calculate the sum of weighted distances for all the rules in R_A (Category-I).

4.5 Fitness Function

We first normalize the three effectiveness measures using the simple yet robust unity-based normalization function $nor\left(F(x)\right) = \left(\frac{F(x) - F_{min}}{F_{max} - F_{min}}\right)$[39, 40], where $F(x)$ is an effectiveness measure function, F_{max} and F_{min} are the maximum and minimum values of the effectiveness measure. For $AHNS$, F_{min} is 0 when the distance between all the rules in Category-II and configuration solution s_r is 0. F_{max} can be calculated as $\sum_{i=1}^{nnr} Cf(r_i)$ where the distance between all the rules in Category-II and configuration solution s_r is 1. For $NLNS$ and NAS, F_{min} is 0 when the distance between all the rules in the corresponding category and configuration solution s_r is 1. Corresponding to $NLNS$ and NAS, F_{max} can be calculated as $\sum_{i=1}^{nnr} Cf(r_i)$ and $\sum_{i=1}^{nar} Cf(r_i)$ respectively, where the distance between all the rules and configuration solution s_r is 0.

With the three effectiveness measures, we define the fitness function based on the three objectives as follow:

$$F(O_1) = 1 - Nor\ (AHNS(R_N, s_r)) \quad (7)$$

$$F(O_2) = 1 - Nor\ (NLNS(R_N, s_r)) \quad (8)$$

$$F(O_3) = 1 - Nor\ (NAS(R_A, s_r) \quad (9)$$

Note that, in the above equations, we define our search problem as a minimization problem by subtracting each normalized effectiveness measure from 1 to ensure that a configuration solution with a value closer to 0 is better.

The fitness function with the three objectives is combined with NSGA-II and NSGA-III to address the configuration generation optimization problem. We implemented our problem in jMetal by encoding all the configurable parameters in the configuration solution s_r as integer variables (Section 4.3). Besides the possible values for all the variables that are specified by constraining their upper and lower limits, there are no additional constraints. Initially, all the variables in s_r are initialized with random values between their upper and lower limits. During the search, *SBRM*$^+$ generates optimized solutions guided by the fitness function. The jMetal based implementation of our configuration generation problems for both of the case studies are provided in the Bitbucket repositories[3].

5. Evaluation

The overall objective of the evaluation is to assess the effectiveness of combining two different machine learning algorithms (i.e., PART and C4.5) with NSGA-II and NSGA-III to mine CPL rules. In Section 5.1, we present experiment design, followed by the experiment execution (Section 5.2).

5.1 Experiment Design

We present research questions in Section 5.1.1, the two case studies in Section 5.1.2, evaluation metrics in Section 5.1.3, evaluation tasks and parameter settings in Section 5.1.4, and statistical tests used for analysis in Section 5.1.5. In Table 3, we provide a summary of the experiment design.

5.1.1 Research Questions

The overall objective of the evaluation is to investigate if NSGA-II and NSGA-III are effective, as compared to RS, in terms of solving the configuration generation problem, and assess the quality of rules mined using two machine learning algorithms (PART and C4.5) when combined with NSGA-II and NSGA-III. The overall objective can be achieved by answering the following research questions:

RQ1. *Are NSGA-II and NSGA-III effective to generate configurations for the purpose of mining rules as compared to RS?*
RQ2. *Does SBRM$^+$ produce better quality rules (in terms of machine learning measurements) than RBRM$^+$?*
RQ3. *To what extent the quality of rules improved using SBRM$^+$ in comparison to RBRM$^+$ (after the final iteration)?*
RQ4. *Which one of NSGA-II and NSGA-III is more effective to generate configurations for mining rules?*
RQ5. *Which one of PART and C4.5, when combined with NSGA-II and NSGA-III, produces better quality rules*?
RQ6. *How is the quality of rules correlated with average fitness values and quality indicators?*
RQ7. *What is the trend of the quality of rules produced by SBRM$^+$ across the iterations?*
RQ8. *Is it feasible to apply SBRM$^+$ in practice in terms of time required for employing search to generate configurations?*

Table 3: Overall design of the experiment*

RQs	Tasks	Evaluation metrics	Comparison/Treatment	Statistical tests and plot types
1	T$_1$-comparing fitness values and six quality indicators for SBRM$^+_{\text{NSGA-II}}$-PART and SBRM$^+_{\text{NSGA-III}}$-PART with RBRM$^+$-PART, and SBRM$^+_{\text{NSGA-II}}$-C45 and SBRM$^+_{\text{NSGA-III}}$-C45 with RBRM$^+$-C45	– FV-O1 – FV-O2 – FV-O3 – OFV – Hyper Volume (HV) – Inverted Generational Distance (IGD) – Epsilon (ϵ) – Euclidean Distance (ED) – Generational Distance (GD) – Generated Spread (GS)	– SBRM$^+_{\text{NSGA-II}}$-PART vs. RBRM$^+$-PART – SBRM$^+_{\text{NSGA-III}}$-PART vs. RBRM$^+$-PART – SBRM$^+_{\text{NSGA-II}}$-C45 vs. RBRM$^+$-C45 – SBRM$^+_{\text{NSGA-III}}$-C45 vs. RBRM$^+$-C45	– Mann-Whitney U-test – Vargha and Delaney's $\hat{A}_{12}$ statistics
2	T$_2$-comparing the quality of rules for SBRM$^+_{\text{NSGA-II}}$-PART and SBRM$^+_{\text{NSGA-III}}$-PART with RBRM$^+$-PART, and SBRM$^+_{\text{NSGA-II}}$-C45 and SBRM$^+_{\text{NSGA-III}}$-C45 with RBRM$^+$-C45	– Accuracy – MAE – RMSE – RAE – RRSE – CC/FF/CF/FC-Precision – CC/FF/CF/FC-Recall – CC/FF/CF/FC-FMeasure		
3	T$_3$-quantifying the average relative improvements in the quality of rules based on MLQMs at the end of a cycle for SBRM$^+_{\text{NSGA-II}}$-PART, SBRM$^+_{\text{NSGA-}}$	– ARI in Accuracy – ARI in MAE – ARI in RMSE – ARI in RAE – ARI in RRSE		– Column plot using average values

[3] https://bitbucket.org/safdaraqeel/sbrm-jitsi/, https://bitbucket.org/safdaraqeel/sbrm-cisco

	III-PART, SBRM$^+_{NSGA-II}$-C45, and SBRM$^+_{NSGA-III}$-C45	– ARI in CC/FF/CF/FC-Precision – ARI in CC/FF/CF/FC-Recall – ARI in CC/FF/CF/FC-FMeasure		
4	T$_4$-comparing fitness values and six quality indicators for SBRM$^+_{NSGA-II}$-C45 and SBRM$^+_{NSGA-II}$-PART with SBRM$^+_{NSGA-III}$-C45 and SBRM$^+_{NSGA-III}$-PART	– FV-O1 – FV-O2 – FV-O3 – OFV – Hyper Volume (HV) – Inverted Generational Distance (IGD) – Epsilon (ϵ) – Euclidean Distance (ED) – Generational Distance (GD) – Generated Spread (GS)	– SBRM$^+_{NSGA-II}$-C45 vs. SBRM$^+_{NSGA-III}$-C45 – SBRM$^+_{NSGA-II}$-PART vs. SBRM$^+_{NSGA-III}$-PART	– Mann-Whitney U-test – Vargha and Delaney's $\widehat{A}_{12}$ statistics
5	T$_5$-comparing the quality of rules for SBRM$^+_{NSGA-II}$-PART, SBRM$^+_{NSGA-III}$-PART, SBRM$^+_{NSGA-II}$-C45, and SBRM$^+_{NSGA-III}$-C45	– Accuracy – MAE – RMSE – RAE – RRSE – CC/FF/CF/FC-Precision – CC/FF/CF/FC-Recall – CC/FF/CF/FC-FMeasure	– SBRM$^+_{NSGA-II}$-C45 vs. SBRM$^+_{NSGA-II}$-PART – SBRM$^+_{NSGA-III}$-C45 vs. SBRM$^+_{NSGA-III}$-PART – Winner (SBRM$^+_{NSGA-II}$-C45 vs. SBRM$^+_{NSGA-II}$-PART) vs. Winner (SBRM$^+_{NSGA-III}$-C45 vs. SBRM$^+_{NSGA-III}$-PART)	– Mann-Whitney U-test Vargha and Delaney's $\widehat{A}_{12}$ statistics
6	T$_6$-assessing the correlation of average fitness values and quality indicators with MLQMs for SBRM$^+_{NSGA-II}$-PART, SBRM$^+_{NSGA-III}$-PART, SBRM$^+_{NSGA-II}$-C45, and SBRM$^+_{NSGA-III}$-C45	– All the MQLMS vs. AFV-O1 – All the MQLMS vs. AFV-O2 – All the MQLMS vs. AFV-O3 – All the MQLMS vs. OAFV – All the MQLMS vs. HV – All the MQLMS vs. IGD – All the MQLMS vs. ϵ – All the MQLMS vs. ED – All the MQLMS vs. GD – All the MQLMS vs. GS	– SBRM$^+_{NSGA-II}$-C45 – SBRM$^+_{NSGA-III}$-C45 – SBRM$^+_{NSGA-II}$-PART – SBRM$^+_{NSGA-III}$-PART	– Spearman's correlation
7	T$_7$-assessing the trend of the quality of rules based on MLQMs across the iterations for SBRM$^+_{NSGA-II}$-PART, SBRM$^+_{NSGA-III}$-PART, SBRM$^+_{NSGA-II}$-C45, and SBRM$^+_{NSGA-III}$-C45	– Accuracy – MAE – RMSE – RAE – RRSE – CC/FF/CF/FC-Precision – CC/FF/CF/FC-Recall – CC/FF/CF/FC-FMeasure		– Scatter plot – Linear Regression
8	T$_8$-assessing the feasibility of applying search based on the average time required to generate configurations	– ATPI – ATPC	– SBRM$^+_{NSGA-II}$-C45 – SBRM$^+_{NSGA-III}$-C45 – RBRM$^+$-C45 – SBRM$^+_{NSGA-II}$-PART SBRM$^+_{NSGA-III}$-PART – RBRM$^+$-PART	– Average values

* FV-O1= Fitness values for the first objective, FV-O2= Fitness values for the second objective, FV-O3= Fitness values for the third objective, OFV = Overall fitness values, MAE= Mean Absolute Error, RMSE= Root Mean Squared Error, RAE= Relative Absolute Error, RRSE= Root Relative Squared Error, CC=ConnectedConnected, FF= FailedFailed, FC= FailedConnected, CF= ConnectedFailed, ARI= Average Relative Improvement, AFV-O1= Average fitness values for the first objective, AFV-O2= Average fitness values for the second objective, AFV-O3= Average fitness values for the third objective, OAFV= Overall average fitness values, ATPI= Average time (minutes) required to generate configurations per iteration, ATPC= Average time (minutes) required to generate configurations per cycle.

5.1.2 Case Studies

Cisco Systems[4], Norway provides a variety of VCSs to facilitate high-quality virtual meetings [41]. Cisco has developed several product lines for VCS including C-Series, MX-Series, and SX-Series. Each product from these different product lines has several configurable parameters (e.g., *defaultProtocol* and *encryption*), which need to be configured before making calls. For each VCS, we have a set of state variables representing the states of VCS (e.g., *callStatus*, *numberOfActiveCalls*, *cameraConnected*) that vary according to different hardware and software configurations. Each product has several operations (e.g., *dial()*, *disconnect()*, *hold()*, *accept()*, *transfer()*) to support different interactions (e.g., making a call, sharing presentation)

[4] www.cisco.com/c/en/us/products/collaboration-endpoints/index.html

supported by the product. An operation can also take several parameters as input (e.g., *callType*, *callRate*, and *Protocol* for *dial()* operation). For our experiment, we used three real products C60, SX20, and MX300 developed by Cisco, which belong to three different product lines C-series, SX-series, and MX-series. We selected 27 configurable parameters (i.e., including network specific ones) for the Cisco case study, which were related to the call functionality. Simula Research Laboratory has a long-term collaboration with Cisco, Norway under Certus-SFI [42]. As part of our collaboration, we have access to several VCSs at our lab, and thus we used these systems for our experiments. Therefore, our case study is real, but the experiment was not performed in the real industrial setting of Cisco.

Jitsi [43] is a real-world open source Audio/Video Internet Phone, and Instant Messenger developed in Java, which supports several known protocols including *SIP*, *AIM*, and *ICQ*. Jitsi was developed based on the OSGI architecture using Apache-Felix implementation. Jitsi provides a large number of features such as encrypted audio/video conference calls, messaging, desktop sharing, call hold, transfer, and call recording. Jitsi has several configurable parameters (e.g., *sIPZtpr*, *defaultProtocol*, *audioCodec*) and state variables (e.g., *callStatus*, *numberOfConferenceParticipants*). Just like the Cisco case study, Jitsi also has several operations such as *dial()*, *accept()*, and *hold()*. We extended the case study by adding a new OSGI bundle to introduce several new configurable parameters (e.g., *defaultCallRate*, *MTU*) and implemented several rules constraining the configurable parameter values. These implemented rules determine the success of a call connection based on configurable parameter values assigned to the configurable parameters of the caller and two callees, as we used three instances (products) of Jitsi in our experiment as for the Cisco case study. The total number of the configurable parameters selected for the Jitsi case study is 39.

For both Cisco and Jitsi case studies, we selected making a call as the interaction because making a call is the main functionality of a VCS/VoIP and other functionalities depend on it. The call statuses of both callees were therefore selected as the state variables. Based on the two-state variables (i.e., call statuses for both callees) system states were defined by concatenating their state values, which were used to classify the configurations. For both case studies (i.e., Cisco and Jitsi), we have one normal system state *ConnectedConnected* (CC) and three abnormal system states *FailedFailed* (FF), *FailedConnected* (FC), and *ConnectedFailed* (CF), constituting four classes in our rule-mining problem, which is in nature a classification problem in machine learning. The *ConnectedConnected* shows that caller is connected to both of the callees successfully and *FailedFailed* indicates that the caller is failed to establish connections with the two callees. *FailedConnected* shows that the caller is connected to the second callee and failed to connect with the first callee whereas *ConnectedFailed* shows that caller is successfully connected with the first callee but failed to connect with the second callee. For both case study, to enable the making a call interaction, we used two operations *dial()* and *disconnect()* of the caller, one operation *accept()* for both callees.

5.1.3 Evaluation Metrics

To answer RQ1 (Table 3), we compared NSGA-II and NSGA-III with RS in terms of FV-O1, FV-O2, FV-O3, and OFV. FV-O1, FV-O2, and FV-O3 are fitness values of Objective-1, Objective-2, and Objective-3 respectively (Section 4.4) whereas OFV is the overall fitness. OFV is calculated by taking the average of FV-O1, FV-O2, and FV-O3, as common practice [44]. Additionally, we compared NSGA-II and NSGA-III with RS in terms of six quality indicators: *Hypervolume (HV)*, *Inverted Generational Distance (IGD)*, *Epsilon (ϵ)*, *Euclidean Distance from the Ideal Solution (ED)*, *Generational Distance (GD)*, and *Generated Spread (GS)*. These quality indicators have been used in the existing literature [16, 25, 45-65] to measure the quality of solutions produced by the search algorithms in terms of convergence and diversity. The selected quality indicators are as follow:

- *Hypervolume (HV)* calculates the volume in the objective space covered by members of a Pareto front PF_c produced by a particular search algorithm to measure both convergence and diversity [66]. A higher value of *HV* indicates better performance of the algorithm in terms of convergence and diversity.
- *Inverted Generational Distance (IGD)* calculates the average distance of the solutions in optimal Pareto front PF_o to the closest solution in computed Pareto front PF_c [67]. A lower *IGD* shows better performance of the algorithm in terms of convergence and diversity.
- *Epsilon (ϵ)* calculates the shortest distance required to transform every solution in the computed Pareto front PF_c to the closest solution in optimal Pareto front PF_o [68, 69]. A lower value of ε indicates better performance of the algorithm in terms of convergence and diversity.

- *Euclidean Distance from the Ideal Solution (ED)* measures the Euclidean distance between the ideal solution and the closest solution in PF_c [70]. The ideal solution is created by selecting the optimal value of each objective (e.g., minimum values for a minimization problem) obtained from all the non-dominated solutions in PF_c. A value of 0 for *ED* shows that PF_c includes the ideal solution.
- *Generational Distance (GD)* calculates the average Euclidean distance between solutions in the computed Pareto front PF_c and the optimal Pareto front PF_o. A lower value of *GD* shows better performance of the algorithm in terms of convergence.
- *Generated Spread (GS)* measures the extent of spread for the solutions in computed Pareto front PF_c produced by a search algorithm [71-73]. A lower value of *GS* shows better performance of the algorithm in terms of diversity.

Since the optimal Pareto font PF_o is not known for our problem like most of the real-world problems, thus, we used reference Pareto front to compute the values of indicators. To compute the reference Pareto front, we combined the Pareto fronts produced by all the search algorithms. Note, we computed two separate reference Pareto fronts for the approaches using C45 and PART as rule mining algorithms.

To answer RQ2 (Table 3), we compared *SBRM*$^+_{\text{NSGA-II}}$-*C45* and *SBRM*$^+_{\text{NSGA-III}}$-*C45* with *RBRM*$^+$-*C45*, and *SBRM*$^+_{\text{NSGA-II}}$-*PART* and *SBRM*$^+_{\text{NSGA-III}}$-*PART* with *RBRM*$^+$-*PART* based on 17 (i.e., five related to the classifier and 12 related to the four classes) machine-learning quality measurements (MLQMs): *Accuracy*, *Mean Absolute Error* (*MAE*), Root *Mean Squared Error* (*RMSE*), *Relative Absolute Error* (*RAE*), and *Root Relative Squared Error* (*RRSE*) of a classifier and *Precision*, *Recall*, and *FMeasure* for the four classes [31]. To differentiate *Precision*, *Recall*, and *FMeasure* corresponding to four classes, we used abbreviations of the classes with *Precision*, *Recall*, and *FMeasure*. For example, *Precision* for *ConnectedConnected* is represented as *CC-Precision*.

- *Accuracy* indicates the overall performance of rule mining algorithms (e.g., C4.5, PART) by specifying the percentage of instances that conform to mined rules [26], where one instance contains one specific configuration (i.e., a set of configurable parameter values for the selected configurable parameters of the communicating products) and corresponding system state.
- *Precision* represents the percentage of instances that are correctly classified divided by the total number of instances covered by rules associated with a specific system state (i.e., defined based on the call statuses of both callees in our case). For example, 98% *FF-Precision* means that, according to the mined rules, there are 2% of instances whose configurations are identified as invalid ones, which led to the *FailedFailed* state. But actually, they lead to the other states (e.g., *ConnectedConnected*, *FailedConnected*, *ConnectedFailed*).
- *Recall* represents the percentage of instances that are correctly classified divided by the total number of instances corresponding to a particular system state. For example, 90% *FF-Recall* means that configurations of 10% instances are not associated with the *FailedFailed* state according to the mined rules, but these instances actually lead to the *FailedFailed* state.
- *FMeasure* is the harmonic mean of *Precision* and *Recall* [26].
- *Mean Absolute Error (MAE)* represents an average of individual errors (i.e., differences between values predicted by the classifier and the actual observed values) without considering the sign of the error.
- *Root Mean Squared Error (RMSE)* is the square root of the mean of the absolute squared error (i.e., square of *MAE*). *RMSE* amplify the effect of outliers (i.e., individuals with large errors) by squaring their errors.
- *Relative Absolute Error (RAE)* is calculated as *MAE* divided by the error of the default predictor (i.e., ZeroR classifier, which simply selects the most frequent value from training dataset (if nominal) or the average value (if numerical).
- *Root Relative Squared Error (RRSE)* is the square root of the relative mean squared error (i.e., square of *RAE*) [31].

For calculating the values for the MLQMs mentioned above, we used 10 times 10-fold stratified cross-validation [31, 36], as stratified cross-validation ensures that each class is (approximately) equally represented across each test fold [31] (Section 3).

For answering RQ3 (Table 3), we calculate the average relative improvements (ARIs) in terms of 17 MLQMs mentioned above achieved at the end of each cycle (i.e., after *iteration-5*) using *SBRM*$^+_{\text{NSGA-II}}$-*C45* and *SBRM*$^+_{\text{NSGA-III}}$-*C45* in comparison to *RBRM*$^+$-*C45*, and *SBRM*$^+_{\text{NSGA-II}}$-*PART* and *SBRM*$^+_{\text{NSGA-III}}$-*PART* in comparison to *RBRM*$^+$-*PART*. We calculated the ARIs for *Accuracy* of classifier and *Precision*, *Recall*, and *FMeasure* for all the classes with respect to *SBRM*$^+_{\text{NSGA-II}}$*-C45*, *SBRM*$^+_{\text{NSGA-III}}$*-C45*, *SBRM*$^+_{\text{NSGA-II}}$*-PART*, and *SBRM*$^+_{\text{NSGA-III}}$*-PART* as:

$$ARI = \frac{\sum_{c=1}^{10}(S(x_{ic}) - R(x_{ic}))}{10} \qquad (10)$$

where $S(x_{ic})$ and $R(x_{ic})$ give the values of ith MLQM in *iteration-5* for cth cycle corresponding to *SBRM*$^+_{\text{NSGA-II}}$*-C45* or *SBRM*$^+_{\text{NSGA-III}}$*-C45* (*SBRM*$^+_{\text{NSGA-II}}$*-PART* or *SBRM*$^+_{\text{NSGA-III}}$*-PART*) and *RBRM*$^+$*-C45*(*RBRM*$^+$*-PART*), respectively. To calculate the ARIs for *MAE*, *RAE*, *RMSE*, and *RRSE* with respect to *SBRM*$^+_{\text{NSGA-II}}$*-C45*, *SBRM*$^+_{\text{NSGA-III}}$*-C45*, *SBRM*$^+_{\text{NSGA-II}}$*-PART*, and *SBRM*$^+_{\text{NSGA-III}}$*-PART* we used the following formula:

$$ARI = \frac{\sum_{c=1}^{10}(R(x_{ic}) - S(x_{ic}))}{10} \qquad (11)$$

For RQ4 (Table 3), we compared NSGA-II with NSGA-III in terms of FV-O1, FV-O2, FV-O3, OFV, and six quality indicators as we did in RQ1 for comparing NSGA-II and NSGA-III with RS. For RQ5, we compared the quality of the rules produced from *SBRM*$^+_{\textbf{NSGA-II}}$*-C45*, *SBRM*$^+_{\textbf{NSGA-III}}$*-C45*, *SBRM*$^+_{\textbf{NSGA-II}}$*-PART*, and *SBRM*$^+_{\textbf{NSGA-III}}$*-PART*, based on 17 MLQMs mentioned above to find the best-suited search algorithm combined with rule mining algorithm for mining CPL rules. To answer RQ6 (Table 3), we computed the correlation estimates (ρ) and the *p*-values using the Spearman's test corresponding to all the 17 MLQMs in correlation to the average fitness values for the three individual objectives (i.e., *AFV-O1*, *AFV-O2*, and *AFV-O3*), overall average fitness (*OAFV*), and six quality indicators (i.e., *HV*, *IGD*, ϵ, *ED*, *GD*, and *GS*). *AFV-O1*, *AFV-O2*, *AFV-O3*, and *OAFV* are calculated based on the values of *FV-O1*, *FV-O2*, *FV-O3*, and *OFV* respectively, corresponding to each iteration of all the runs. For example, *AFV-O1* corresponding to one iteration can be calculated as: $AFV - O1 = \frac{\sum_{i=1}^{500} FV - O1_i}{500}$, where 500 is the total number of fitness values. For RQ7 (Table 3), we assessed the trend of the quality of rules in terms of above-mentioned 17 MLQMs for *SBRM*$^+_{\textbf{NSGA-II}}$*-C45*, *SBRM*$^+_{\textbf{NSGA-III}}$*-C45*, *SBRM*$^+_{\textbf{NSGA-II}}$*-PART*, and *SBRM*$^+_{\textbf{NSGA-III}}$*-PART*, across the five iterations. To answer RQ8 (Table 3), we calculated the average time required by NSGA-II in *SBRM*$^+_{\textbf{NSGA-II}}$*-C45* and *SBRM*$^+_{\textbf{NSGA-II}}$*-PART*, NSGA-III in *SBRM*$^+_{\textbf{NSGA-III}}$*-C45* and *SBRM*$^+_{\textbf{NSGA-III}}$*-PART*, and RS in *RBRM*$^+$*-C45* and *RBRM*$^+$*-PART* for generating configurations per iteration (*ATPI*) and per cycle (*ATPC*). *ATPI* is calculated as: $ATPI = \frac{\sum_{i=1}^{5}\sum_{c=1}^{10} T_{ic}}{50}$, where T_{ic} represents the time required by the approach in the ith iteration of the cth cycle. *ATPC* is calculated as: $ATPC = \sum_{i=1}^{5}(\frac{\sum_{c=1}^{10} T_{ic}}{10})$.

5.1.4 Experimental Tasks and Parameter Settings

As shown in Table 3, we designed eight tasks (T_1-T_8) for addressing RQ1-RQ8. We used the default settings for NSGA-II and NSGA-III as implemented in jMetal [71, 74]. The single point crossover and bit-flip mutation, implemented in jMetal were applied as crossover and mutation operators, respectively with 0.9 crossover rate and (1/total number of configurable parameters) mutation rate. We used a population size of 500 and 50,000 fitness evaluations where we selected all the Pareto Non-dominated configuration solutions for mining the rules. NSGA-III produces 92 solutions for three objective problems regardless the larger population size [23], thus, we executed it using multiple threads to get 500 solutions in one run. We used five iterations per cycle, and in each iteration, we run the search algorithm (NSGA-II, NSGA-III, or RS) once, which means we have five runs of the search algorithm in a complete cycle. We used total 10 cycles (i.e., 50 runs of the search algorithm) for our experiment to cater the randomness inherited in the search algorithms.

Since selecting the best set of parameters is application dependent [11], we used the default settings provided by Weka [12] for both PART and C4.5. Default settings have been used in various contexts such as mining rules for video generator product line [11] and comparing the performance of different classification algorithms [75]. We used 0.25 and 2 for minConfidence (i.e., the minimum confidence for a rule) and minNumObj (i.e., the minimum number of instances for a rule) respectively.

5.1.5 Statistical Analyses

As inspired by [76], we systematically conducted three types of analyses: *difference analysis*, *correlation analysis*, and *trend analysis* to answer RQ1-RQ2 and RQ4-RQ7 (Section 5.1.1). To answer RQ3 and RQ8, we report descriptive statistics, as for RQ3 we intend to assess the magnitude of ARI achieved by *SBRM*$^+_{\textbf{NSGA-II}}$*-C45*, *SBRM*$^+_{\textbf{NSGA-III}}$*-C45*, *SBRM*$^+_{\textbf{NSGA-II}}$*-PART*, and *SBRM*$^+_{\textbf{NSGA-III}}$*-PART* whereas, for RQ8, we aim to show the total time required for generating configurations.

Difference analysis is the most commonly used analysis, which studies the distributions of a single measure between two groups. We carried out the difference analysis to compare rule mining approaches (e.g., *SBRM*$^+_{NSGA\text{-}II}$*-C45*, *RBRM*$^+$*-C45*) in terms of fitness values, quality indicators, and MLQMs (Table 3) to answer RQ1, RQ2, RQ4, and RQ5. To conduct the *difference analysis*, we use the non-parametric Mann-Whitney U-test as recommended in [77] with a significance level of 0.05 and the Vargha and Delaney's $\widehat{A}_{12}$ statistics as an effect size measure [78]. For comparing the *Accuracy, Precision*, *Recall*, *F-measure*, and *HV* for a comparison pair (A_i vs. A_j), if $\widehat{A}_{12}$ is greater than 0.5, A_i is better than A_j, and a value less than 0.5 means vice versa. Similarly, in the case of fitness values, *IGD*, ϵ, *ED*, *GD*, *GS MAE*, *RMSE*, *RAE*, and *RRSE*, if $\widehat{A}_{12}$ is less than 0.5, A_i is better than A_j, otherwise, A_j is better than A_i. $A_i > A_j$ shows that approach A_i performed significantly better than A_j based on the results of the Mann-Whitney U-test and Vargha and Delaney's $\widehat{A}_{12}$ statistics. Similarly, $A_i < A_j$ shows that A_j significantly outperformed A_j whereas $A_i = A_j$ indicates no significant difference between the two approaches being compared.

Correlation analysis evaluates the correlation (positive/negative) between two variables (e.g., x and y) and its statistical significance. To find the correlation of MLQMs with average fitness values and six quality indicators (RQ6), we applied the nonparametric Spearman's test [79] and reported the correlation coefficients (ρ) and *p*-values. The value of ρ ranges from -1 to 1 where a value of $\rho > 0$ (or $\rho < 0$) shows a positive (or negative) correlation between x and y, whereas $\rho = 0$ indicates no correlation. The p-value lower than 0.05 shows the correlation is statistically significant. The analysis aims to test whether *Accuracy, Precision*, *Recall*, and *FMeasure* are positively correlated with *HV* and negatively correlated with average fitness values, *IGD*, ϵ, *ED*, *GD*, and *GS* and *MAE*, *RMSE*, *RAE*, and RRSE have a negative correlation with *HV* and a positive correlation with average fitness values, *IGD*, ϵ, *ED*, *GD*, and *GS* (i.e., hypothesis). Satisfying the hypothesis is regarded as good performance of the approach because we believe that smaller fitness and indicator values (except for *HV*, as the larger *HV* is better) lead to better quality of rules in terms of MLQMs.

To discover the trend of the quality of rules based on MLQMs (RQ7), we constructed 2D scatter plots and fitted linear regression lines. In 2D plots, the x-axis represents the iteration number in one cycle, and the y-axis represents different machine learning quality measurements such as *Accuracy*. This kind of analyses indicates variation in the quality of rules across the iterations.

For assessing the magnitude of average relative improvements (ARIs) in the quality of rules in terms of MLQMs (RQ3), we reported mean, min, and max values. Similarly, for assessing the feasibility of applying NSGA-II in *SBRM*$^+_{NSGA\text{-}II}$*-C45* and *SBRM*$^+_{NSGA\text{-}II}$*-PART*, NSGA-III in *SBRM*$^+_{NSGA\text{-}III}$*-C45* and *SBRM*$^+_{NSGA\text{-}III}$*-PART* based on the time required for generating configurations (RQ8), we reported average values of time required to generate configurations per iteration (i.e., *ATPI*) and per cycle (*ATPC*).

5.2 Experiment Execution

Figure 6 presents an overview of the experiment execution. As shown in Figure 6, at the first step, we randomly generated a set of 2000 configurations corresponding to the three selected products (*Caller*, *Callee1*, and *Callee2*) for each case study. For the Cisco case study, we selected C60 (i.e., *Caller*), MX300 (i.e., *Callee1*), and SX20 (i.e., *Callee2*). For the Jitsi case study, we used three instances (products) belonging to the same product line. At the second step, we configured the *Caller*, *Callee1*, and *Callee2* using randomly generated configurations and made a call from *Caller* to *Callee1* and *Callee2* for 10 seconds (step 3). We made the call for 10 seconds to give sufficient time for establishing the call connection. To make the call, first, we execute the *dial()* operation of *Caller* and then *accept()* operation of *Callee1* and *Callee2*. In step 4, we captured call statuses of *Callee1* and *Callee2* to get the system state and added the current configuration being executed and its corresponding system state to the executed configurations (step 5) whereas, in step 6, we disconnected the call by executing the *disconnect()* operation of *Caller*. We repeated step 2 to step 6 until all the configurations (2000 configurations) are executed. In step 7, we input executed configurations containing 2000 configurations along with their corresponding system states to Weka [12] and applied PART [14] and C4.5 to mine the initial set of rules.

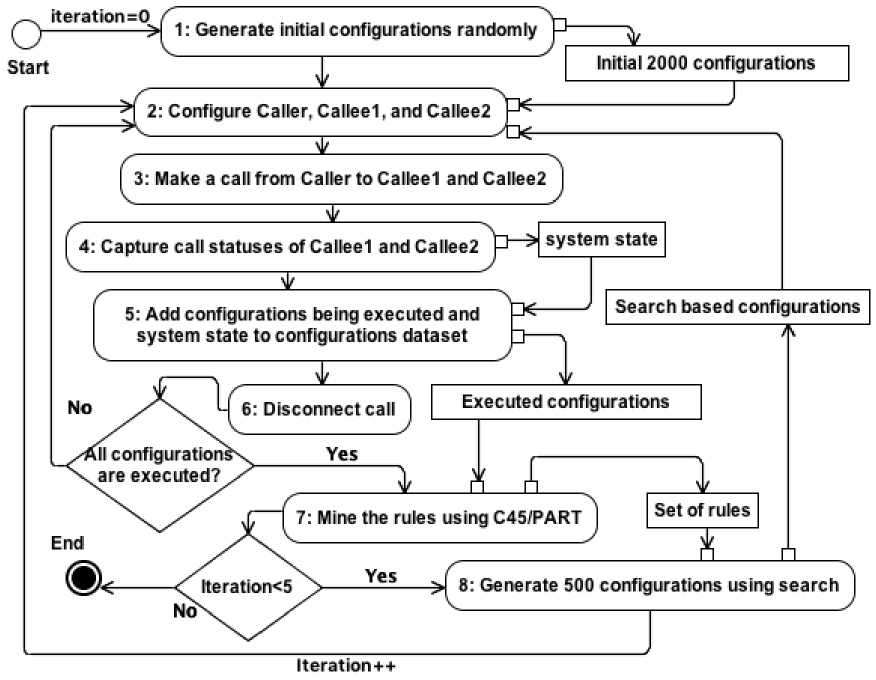

iteration=0
Start
1: Generate initial configurations randomly
Initial 2000 configurations
2: Configure Caller, Callee1, and Callee2
3: Make a call from Caller to Callee1 and Callee2
4: Capture call statuses of Callee1 and Callee2
system state
5: Add configurations being executed and system state to configurations dataset
Search based configurations
6: Disconnect call
Executed configurations
No
All configurations are executed?
Yes
7: Mine the rules using C45/PART
Set of rules
End
Iteration<5
Yes
No
8: Generate 500 configurations using search
Iteration++

Figure 6: An overview of the experiment execution

To refine the rules, we used the initial set of rules to guide the search algorithms (i.e., NAGA-II, NAGA-III, and RS) to generate 500 more configurations (step 8). For mining the refined set of rules, we repeated the same process starting from step 2 to step 7 (i.e., configuring the products, making the calls, adding the configurations and associated system states to the executed configurations, disconnecting the calls, and mining the rules using all the executed configurations). We repeated this incremental and iterative process for five iterations in a complete cycle and mined the final set of rules based on a dataset (i.e., represented as executed configurations in Figure 6) containing 4500 configurations and corresponding system states. We used five iterations as a stopping criterion.

For generating configurations using NSGA-II, NSGA-III, or RS for both case studies, we ran the experiment on a laptop with Intel Core i7 2.8 GHz CPU and 16GB RAM running the macOS Sierra v10.12.5 operating system. To make calls for the Jitsi case study, we installed three instances of Jitsi (*Caller*, *Callee1*, *Callee2*) on three computers. *Caller* was installed on the laptop mentioned above (i.e., the one used for generating configurations). *Caller1* was installed on a desktop (iMac) with Intel Core i5 CPU 2.7 GHz and 8GB RAM running the macOS Sierra v10.12.4 operating system. *Caller2* was installed on a laptop with Intel Core i7 CPU 2.5 GHz and 16GB RAM running the Windows-7 (x64) operating system. For the Cisco case study, all the three products have their dedicated hardware.

6. RESULTS AND ANALYSIS

In this section, we present the results and analysis of the evaluation and answer the research questions for both of the case studies (i.e., Cisco and Jitsi).

6.1 Effectiveness of Search (RQ1)

To answer RQ1, we compare $SBRM^{+}_{NSGA\text{-}II}$-*C45* and $SBRM^{+}_{NSGA\text{-}III}$-*C45* with $RBRM^{+}$-*C45* and $SBRM^{+}_{NSGA\text{-}II}$-*PART* and $SBRM^{+}_{NSGA\text{-}III}$-*PART* with $RBRM^{+}$-*PART* regarding the fitness values (i.e., *FV-O1*, *FV-O2*, *FV-O3*, and *OFV*) and six quality indicators corresponding to five individual iterations as well as overall, for both of the two case studies. In Table 4, we summarize the results for answering RQ1 whereas whereas the detailed results can be found in Appendix B.

The results of Man-Whitney U-test and Vargha and Delaney's $\hat{A}_{12}$ for all the fitness values (i.e., FV-O1, FV-O2, FV-O3, and OFV) show that $SBRM^{+}$ (i.e., $SBRM^{+}_{NSGA\text{-}II}$-*C45*, $SBRM^{+}_{NSGA\text{-}III}$-*C45*, $SBRM^{+}_{NSGA\text{-}II}$-*PART* and $SBRM^{+}_{NSGA\text{-}III}$-*PART*) significantly outperformed $RBRM^{+}$ (i.e., $RBRM^{+}$-*C45* and $RBRM^{+}$-*PART*) corresponding to both the Cisco and Jitsi case studies. Similarly, from the results of the quality indicators (Table 4), we noticed that *SBRM*$^+$ significantly outperformed *RBRM*$^+$ in terms of the majority of the comparisons (i.e., minimum 25 and maximum 32 out of 36 comparisons). Note, for each comparison pair, we have six comparisons (five individual iterations and overall) in terms of a particular quality indicator for one case study (i.e., total 36 comparisons for six indicators per case study and 48 comparisons for one quality indicator for all comparisons pairs and two case studies). We observed that for five indicators (except for *GS*), *SBRM*$^+$ significantly outperformed *RBRM*$^+$ for 221 out of 240 comparisons whereas *RBRM*$^+$ significantly outperformed *SBRM*$^+$ in terms of *GS* for 32 out of 48 comparisons for both of the case studies. Based on the results of RQ1, it can be concluded that NSGA-II and NSGA-III are more effective than RS for configuration generation problem. The detailed results of RQ1 can be found in Appendix B

Table 4: Comparing SBRM$^+$ with RBRM in terms of the quality indicators*

Case study	Comparison Pair											
	A_1 vs A_5			A_2 vs A_5			A_3 vs A_6			A_4 vs A_6		
	$A_1>A_5$	$A_1<A_5$	$A_1=A_5$	$A_2>A_5$	$A_2<A_5$	$A_2=A_5$	$A_3>A_6$	$A_3<A_6$	$A_3=A_6$	$A_4>A_6$	$A_4<A_6$	$A_4=A_6$
Cisco	25/36	6/36	5/36	32/36	0/36	4/36	28/36	6/36	2/36	30/36	3/36	3/36
Jitsi	32/36	1/36	3/36	29/36	6/36	1/36	27/36	4/36	5/36	27/36	6/36	3/36

*A_1= SBRM$^+_{NSGA-II}$-C45, A_2= SBRM$^+_{NSGA-III}$-C45, A_2= SBRM$^+_{NSGA-II}$-PART, A_4= SBRM$^+_{NSGA-III}$-PART, A_5= RBRM$^+$-C45, A_6= RBRM$^+$-PART

6.2 Comparing SBRM$^+$ with RBRM$^+$ (RQ2)

To answer RQ2, we compare *SBRM*$^+_{NSGA-II}$*-C45* and *SBRM*$^+_{NSGA-III}$*-C45* with *RBRM*$^+$*-C45*, and *SBRM*$^+_{NSGA-II}$*-PART* and *SBRM*$^+_{NSGA-III}$*-PART* with *RBRM*$^+$*-PART* in terms of MLQMs based on the rules mined from each iteration as well as *Overall* (i.e., combining the results of all the five iterations), for both case studies. In Table 5, we summarize the results for answering RQ2. Detailed results are provided in Appendix B for reference.

Table 5: Comparing SBRM$^+$ with RBRM in terms of MLQMs*

Case study	Comparison Pair											
	A_1 vs A_5			A_2 vs A_5			A_3 vs A_6			A_4 vs A_6		
	$A_1>A_5$	$A_1<A_5$	$A_1=A_5$	$A_2>A_5$	$A_2<A_5$	$A_2=A_5$	$A_3>A_6$	$A_3<A_6$	$A_3=A_6$	$A_4>A_6$	$A_4<A_6$	$A_4=A_6$
Cisco	78/90	0/90	12/90	54/90	7/90	29/90	70/93	3/93	20/93	57/93	3/93	33/93
Jitsi	86/102	0/102	16/102	19/102	20/102	63/102	88/102	0/102	14/102	48/102	38/102	16/102

*A_1= SBRM$^+_{NSGA-II}$-C45, A_2= SBRM$^+_{NSGA-III}$-C45, A_2= SBRM$^+_{NSGA-II}$-PART, A_4= SBRM$^+_{NSGA-III}$-PART, A_5= RBRM$^+$-C45, A_6= RBRM$^+$-PART

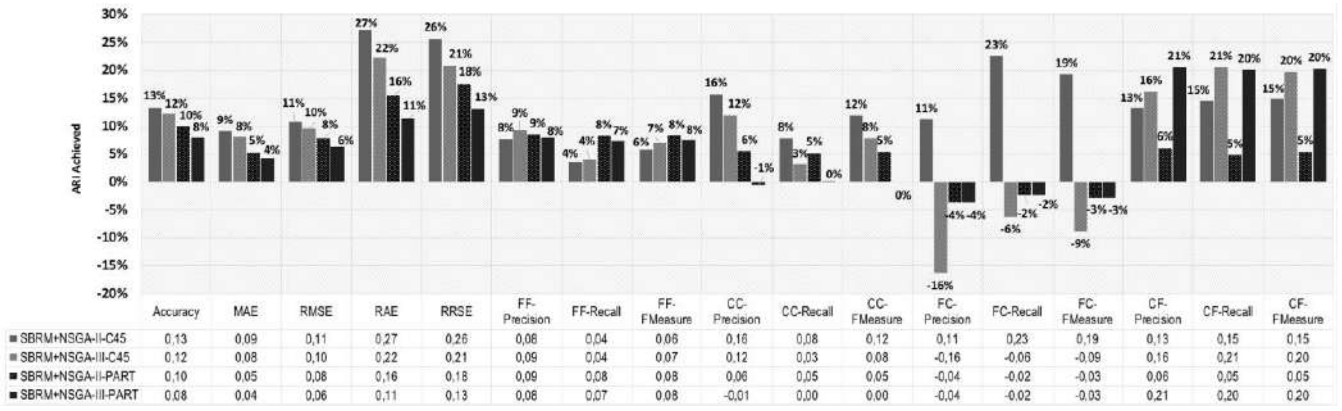
As shown in Table 5, for the Cisco case study, *SBRM*$^+_{\text{NSGA-II}}$*-C45* and *SBRM*$^+_{\text{NSGA-III}}$*-C45* significantly outperformed *RBRM*$^+$*-C45* in 87% (i.e., 78/90) and 60% (i.e., 54/90) of the total comparisons. Respectively. *SBRM*$^+_{\text{NSGA-II}}$*-PART* and *SBRM*$^+_{\text{NSGA-III}}$*-PART* significantly outperformed *RBRM*$^+$*-PART* in 75% (i.e., 70/93) and 61% (i.e., 57/93) of the total comparisons. In 8% (i.e., 7/90) of the total comparisons, *RBRM*$^+$*-C45* significantly outperformed *SBRM*$^+_{\text{NSGA-III}}$*-C45* whereas for 3% (3/90) and 3%(3/93) of the total comparisons RBRM$^+$*-PART* significantly outperformed *SBRM*$^+_{\text{NSGA-II}}$*-PART* and *SBRM*$^+_{\text{NSGA-III}}$*-PART*, respectively. For the remaining comparisons, there was no significant difference between the *SBRM*$^+$ approaches and the *RBRM*$^+$ approaches.

Corresponding to the Jitsi case study, *SBRM*$^+_{\text{NSGA-II}}$*-C45* and *SBRM*$^+_{\text{NSGA-III}}$*-C45* significantly outperformed *RBRM*$^+$*-C45* in 84% (i.e., 86/102) and 19% (i.e., 19/102) of the total comparisons respectively, whereas *SBRM*$^+_{\text{NSGA-II}}$*-PART* and *SBRM*$^+_{\text{NSGA-III}}$*-PART* significantly outperformed *RBRM*$^+$*-PART* in 86% (i.e., 88/102) and 47% (i.e., 48/102) of the total comparisons. In 20% (i.e., 20/102) and 37% (i.e., 38/102) of total comparisons *RBRM*$^+$*-C45* and *RBRM*$^+$*-PART* significantly outperformed *SBRM*$^+_{\text{NSGA-III}}$*-C45* and *SBRM*$^+_{\text{NSGA-III}}$*-PART* respectively, whereas for the remaining comparisons there was no significant difference between the *SBRM*$^+$ (*SBRM*$^+_{\text{NSGA-II}}$*-C45*, *SBRM*$^+_{\text{NSGA-III}}$*-C45*, *SBRM*$^+_{\text{NSGA-II}}$*-PART*, and *SBRM*$^+_{\text{NSGA-III}}$*-PART*) and *RBRM*$^+$(*RBRM*$^+$ *-C45* and *RBRM*$^+$*-PART*).

Since for both of the case studies, *SBRM*$^+$ significantly outperformed *RBRM*$^+$ in terms of the majority of MLQMs (i.e., 84% for *SBRM*$^+_{\text{NSGA-II}}$*-C45*, 86% for *SBRM*$^+_{\text{NSGA-II}}$*-PART*, and 47% for *SBRM*$^+_{\text{NSGA-III}}$*-PART*) except for *SBRM*$^+_{\text{NSGA-III}}$*-C45* corresponding to the Jitsi case study where neither one of the two approaches (i.e., *SBRM*$^+_{\text{NSGA-III}}$*-C45* and *RBRM*$^+$*-C45)* dominates the other. Thus, we can conclude that given the same context (i.e., the same case study, machine learning algorithm and its parameter settings) *SBRM*$^+$ tends to produce rules with higher quality as compared to *RBRM*$^+$ with respect to the MLQMs. In the worst case, *SBRM*$^+$ produces rules with the same quality as for *RBRM*$^+$.

6.3 Average Relative Improvements in the Quality of Rules (RQ3)

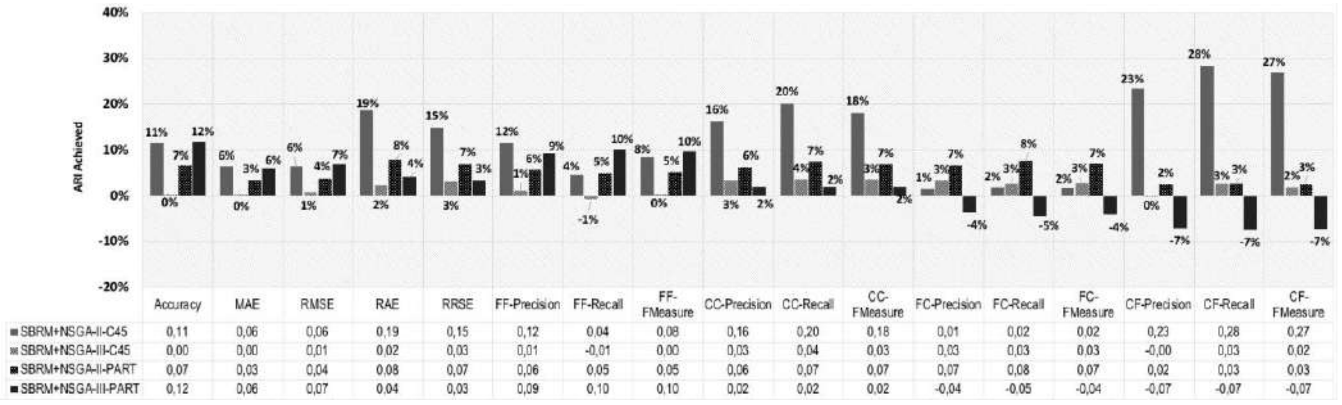
For RQ3, we computed the average relative improvements (ARIs) in terms of MLQMs achieved at the end of the cycle (i.e., after *iteration-5*) using *SBRM*$^+$ (*SBRM*$^+_{\text{NSGA-II}}$-*C45*, *SBRM*$^+_{\text{NSGA-III}}$-*C45*, *SBRM*$^+_{\text{NSGA-II}}$-*PART*, and *SBRM*$^+_{\text{NSGA-III}}$-*PART*) in comparison to *RBRM*$^+$(*RBRM*$^+$ -*C45* and *RBRM*$^+$-*PART*) (Section 5.1.3). In Figure 7 and Figure 8, we present the ARIs in terms of all the MLQMs for *SBRM*$^+_{\text{NSGA-II}}$-*C45*, *SBRM*$^+_{\text{NSGA-III}}$-*C45*, *SBRM*$^+_{\text{NSGA-II}}$-*PART*, and *SBRM*$^+_{\text{NSGA-III}}$-*PART* corresponding to the Cisco and Jitsi case studies respectively. Moreover, the detailed results are presented in Appendix B.



	Accuracy	MAE	RMSE	RAE	RRSE	FF-Precision	FF-Recall	FF-FMeasure	CC-Precision	CC-Recall	CC-FMeasure	FC-Precision	FC-Recall	FC-FMeasure	CF-Precision	CF-Recall	CF-FMeasure
■ SBRM+NSGA-II-C45	0,13	0,09	0,11	0,27	0,26	0,08	0,04	0,06	0,16	0,08	0,12	0,11	0,23	0,19	0,13	0,15	0,15
■ SBRM+NSGA-III-C45	0,12	0,08	0,10	0,22	0,21	0,09	0,04	0,07	0,12	0,03	0,08	-0,16	-0,06	-0,09	0,16	0,21	0,20
■ SBRM+NSGA-II-PART	0,10	0,05	0,08	0,16	0,18	0,09	0,08	0,08	0,06	0,05	0,05	-0,04	-0,02	-0,03	0,06	0,05	0,05
■ SBRM+NSGA-III-PART	0,08	0,04	0,06	0,11	0,13	0,08	0,07	0,08	-0,01	0,00	0,00	-0,04	-0,02	-0,03	0,21	0,20	0,20

Figure 7: ARI achieved by SBRM$^+_{\text{NSGA-II}}$-C45, SBRM$^+_{\text{NSGA-III}}$-C45, SBRM$^+_{\text{NSGA-II}}$-PART, and SBRM$^+_{\text{NSGA-III}}$-PART for the Cisco case study

As shown in Figure 7, for the Cisco case study, on average *SBRM*$^+$ achieved 8% to 13% higher *Accuracy* than *RBRM*$^+$ and 4% to 27% lower values for the four error-related MLQMs (i.e., *MAE*, *RMSE*, *RAE*, and *RRSE*). The ARIs in terms of *FF-Precision*, *FF-Recall*, and *FF-FMeasure* for *SBRM*$^+$ range between 4% and 9% and for *CC-Precision*, *CC-Recall*, and *CC-FMeasure*, the ARIs are up to 16%. The ARIs corresponding to *FC-Precision*, *FC-Recall*, and *FC-FMeasure* for *SBRM*$^+_{\text{NSGA-II}}$-*C45* range between 11% and 23%, while *SBRM*$^+_{\text{NSGA-III}}$-*C45*, *SBRM*$^+_{\text{NSGA-II}}$-*PART*, and *SBRM*$^+_{\text{NSGA-III}}$-*PART* have negative ARIs ranging from -16% to -2%. This is because they did not produce rules related to *FailedConnected* due to less number of configurations leading to *FailedConnected* system state. About *CF-Precision*, *CF-Recall*, and *CF-FMeasure*, the ARIs for *SBRM*$^+$ are between 5% and 21%.



	Accuracy	MAE	RMSE	RAE	RRSE	FF-Precision	FF-Recall	FF-FMeasure	CC-Precision	CC-Recall	CC-FMeasure	FC-Precision	FC-Recall	FC-FMeasure	CF-Precision	CF-Recall	CF-FMeasure
■ SBRM+NSGA-II-C45	0,11	0,06	0,06	0,19	0,15	0,12	0,04	0,08	0,16	0,20	0,18	0,01	0,02	0,02	0,23	0,28	0,27
▨ SBRM+NSGA-III-C45	0,00	0,00	0,01	0,02	0,03	0,01	-0,01	0,00	0,03	0,04	0,03	0,03	0,03	0,03	-0,00	0,03	0,02
■ SBRM+NSGA-II-PART	0,07	0,03	0,04	0,08	0,07	0,06	0,05	0,05	0,06	0,07	0,07	0,07	0,08	0,07	0,02	0,03	0,03
■ SBRM+NSGA-III-PART	0,12	0,06	0,07	0,04	0,03	0,09	0,10	0,10	0,02	0,02	0,02	-0,04	-0,05	-0,04	-0,07	-0,07	-0,07

Figure 8: ARI achieved by SBRM$^+_{\text{NSGA-II}}$-C45, SBRM$^+_{\text{NSGA-III}}$-C45, SBRM$^+_{\text{NSGA-II}}$-PART, and SBRM$^+_{\text{NSGA-III}}$-PART for the Jitsi case study

As shown in Figure 8, for the Jitsi case study, on average *SBRM*$^+$ achieved up to 12% higher *Accuracy* and 19% lower values for error-related MLQMs (i.e., *MAE*, *RMSE*, *RAE*, and *RRSE*) as compared to *RBRM*$^+$. In terms of *FF-Precision*, *FF-Recall*, and *FF-FMeasure*, the ARIs for *SBRM*$^+$ are between -1% and 12% whereas for *CC-Precision*, *CC-Recall*, and *CC-FMeasure*, the ARIs range between 2% and 20%. Concerning *FC-Precision*, *FC-Recall*, and *FC-FMeasure*, the ARIs for *SBRM*$^+$ are up to 8% whereas the ARIs in terms of *CF-Precision*, *CF-Recall*, and *CF-FMeasure* are up to 28%. However, we observed that for some MLQMs (e.g., *CF-Precision*, *CF-Recall*) for *SBRM*$^+_{\text{NSGA-III}}$-*PART* have negative ARIs.

From Figure 7, one can observe that *SBRM*$^+$ has positive improvements for the majority of the MLQMs (i.e., 85%) with an ARI up to 27% for the Cisco case study. Similarly, for the Jitsi case study, Figure 8 shows that *SBRM*$^+$ has positive values for ARIs corresponding to the majority of the MLQMs (i.e., 90%) with an ARI up to 28%. This shows that for both of the case studies, *SBRM*$^+$ has significantly improved the quality of rules in terms of MLQMs as compared to *RBRM*$^+$, as also suggested by the statistical analysis results (Section 6.2).

6.4 Comparing the Effectiveness of NSGA-II and NSGA-III (RQ4)

To answer RQ4, we compare *SBRM*$^+_{NSGA\text{-}II}$*-C45* with *SBRM*$^+_{NSGA\text{-}III}$*-C45* and *SBRM*$^+_{NSGA\text{-}II}$*-PART* with *SBRM*$^+_{NSGA\text{-}III}$*-PART* in terms of the fitness values (i.e., *FV-O1*, *FV-O2*, *FV-O3*, and *OFV*) and the six quality indicators (Section 5.1.3) for both of the two case studies. Table 6 summarizes the results of RQ4 whereas detailed results are presented in Appendix B.

Table 6: Comparing SBRM$^+_{NSGA\text{-}II}$-C45 with SBRM$^+_{NSGA\text{-}III}$-C45 and SBRM$^+_{NSGA\text{-}II}$-PART with SBRM$^+_{NSGA\text{-}III}$-PART in terms of fitness values and quality indicators*

Case study	Fitness value based comparison						Quality indicators based comparison					
	A_1 vs. A_2			A_3 vs. A_4			A_1 vs. A_2			A_3 vs. A_4		
	$A_1>A_2$	$A_1<A_2$	$A_1=A_2$	$A_3>A_4$	$A_3<A_4$	$A_3=A_4$	$A_1>A_2$	$A_1<A_2$	$A_1=A_2$	$A_3>A_4$	$A_3<A_4$	$A_3=A_4$
Cisco	7/24	16/24	1/24	4/24	16/24	4/24	2/36	23/36	11/36	2/36	28/36	6/36
Jitsi	7/24	17/24	0/24	5/24	15/24	4/24	18/36	4/36	14/36	5/36	4/36	27/36

*A_1= SBRM$^+_{NSGA\text{-}II}$-C45, A_2= SBRM$^+_{NSGA\text{-}III}$-C45, A_2= SBRM$^+_{NSGA\text{-}II}$-PART, A_4= SBRM$^+_{NSGA\text{-}III}$-PART

As shown in Table 6, for the Cisco (Jitsi) case study, *SBRM*$^+_{NSGA\text{-}III}$*-C45* significantly outperformed *SBRM*$^+_{NSGA\text{-}II}$*-C45* for 16/24 (17/24) fitness-based comparisons and *SBRM*$^+_{NSGA\text{-}III}$*-PART* significantly outperformed *SBRM*$^+_{NSGA\text{-}II}$*-PART* for 16/24 (15/24) comparisons whereas in only 7/24 (7/24) and 4/24 (5/24) fitness-based comparisons *SBRM*$^+_{NSGA\text{-}II}$*-C45* and *SBRM*$^+_{NSGA\text{-}II}$*-PART* significantly outperformed *SBRM*$^+_{NSGA\text{-}III}$*-C45* and *SBRM*$^+_{NSGA\text{-}III}$*-PART* respectively.

In terms of quality indicators, Table 6 shows that for the Cisco case study, *SBRM*$^+_{NSGA\text{-}III}$*-C45* (*SBRM*$^+_{NSGA\text{-}III}$*-PART*) significantly outperformed *SBRM*$^+_{NSGA\text{-}II}$*-C45* (*SBRM*$^+_{NSGA\text{-}II}$*-PART*) for 23/36 (28/36) indicator-based comparisons whereas for only 2/36 (2/36) indicator-based comparisons *SBRM*$^+_{NSGA\text{-}II}$*-C45* (*SBRM*$^+_{NSGA\text{-}II}$*-PART*) significantly outperformed *SBRM*$^+_{NSGA\text{-}III}$*-C45* (*SBRM*$^+_{NSGA\text{-}III}$*-PART*). Similarly, for the Jitsi case study, *SBRM*$^+_{NSGA\text{-}II}$*-C45* (*SBRM*$^+_{NSGA\text{-}II}$*-PART*) significantly outperformed *SBRM*$^+_{NSGA\text{-}III}$*-C45* (*SBRM*$^+_{NSGA\text{-}III}$*-PART*) in terms of the quality indicators for 18/36 (5/36) comparisons whereas for only 4/36 (4/36) *SBRM*$^+_{NSGA\text{-}III}$*-C45* (*SBRM*$^+_{NSGA\text{-}III}$*-PART*) significantly outperformed *SBRM*$^+_{NSGA\text{-}II}$*-C45* (*SBRM*$^+_{NSGA\text{-}II}$*-PART*). To summarize the results of RQ4, we can notice that in most of the cases NSGA-III significantly outperformed NSGA-II in terms of fitness values and quality indicators, however, in some cases (e.g., for *GS*) we observed otherwise.

6.5 Comparing the quality of rules for SBRM$^+$ (RQ5)

To answer RQ5, we compare the four *SBRM*$^+$ approaches in terms of MLQMs based on the rules from each iteration and *Overall* (i.e., the rules of all the five iterations) for both of the case studies. To do so, first, we compare *SBRM*$^+_{NSGA\text{-}II}$*-C45* with *SBRM*$^+_{NSGA\text{-}III}$*-C45* and *SBRM*$^+_{NSGA\text{-}II}$*-PART* with *SBRM*$^+_{NSGA\text{-}III}$*-PART* and then we compare the two better performing approaches from these two comparisons to find the best. Table 7 summarizes the results of RQ5 whereas the details results can be found in Appendix B.

Table 7: Comparing the quality of rules for the SBRM$^+$ approaches in terms of MLQMs*

Case study	Comparison Pair								
	A_1 vs. A_2			A_3 vs. A_4			W_1 vs. W_2		
	$A_1>A_2$	$A_1<A_2$	$A_1=A_2$	$A_3>A_4$	$A_3<A_4$	$A_3=A_4$	$W_1>W_2$	$W_1<W_2$	$W_1=W_2$
Cisco	49/90	6/90	35/90	24/84	6/84	54/84	74/90	3/90	13/90
Jitsi	85/102	2/102	15/102	60/102	35/102	7/102	2/102	82/102	18/102

*A_1= SBRM$^+_{NSGA\text{-}II}$-C45, A_2= SBRM$^+_{NSGA\text{-}III}$-C45, A_2= SBRM$^+_{NSGA\text{-}II}$-PART, A_4= SBRM$^+_{NSGA\text{-}III}$-PART, W_1= Winner of A_1 vs. A_2, W_2= Winner of A_3 vs. A_4

As shown in Table 7, for the two case studies, *SBRM*$^+_{\text{NSGA-II}}$*-C45* significantly outperformed *SBRM*$^+_{\text{NSGA-III}}$*-C45* in 54% (i.e., 49/90) and 83% (i.e., 85/102) of the total comparisons respectively whereas in only 7% (i.e., 6/90) and 2% (i.e., 2/102) of the total comparisons *SBRM*$^+_{\text{NSGA-III}}$-C45 significantly outperformed *SBRM*$^+_{\text{NSGA-II}}$*-C45*. Similarly, *SBRM*$^+_{\text{NSGA-II}}$*-PART* significantly outperformed *SBRM*$^+_{\text{NSGA-III}}$*-PART* in 29% (i.e., 24/84) and 59% (i.e., 60/102) of total comparisons for the Cisco and Jitsi case studies respectively whereas in 7% (i.e., 6/84) and 34% (i.e., 35/102) of total comparisons *SBRM*$^+$NSGA-III-*PART* significantly performed better than *SBRM*$^+$NSGA-II-*PART*. Since *SBRM*$^+$NSGA-II-*C45* and *SBRM*$^+$NSGA-II-*PART* are two winners from the first two comparisons, we use these two approaches as the third comparison pair to find the best for both case studies.

Table 7 indicates that in 82% (i.e., 74/90) of the total comparisons, *SBRM*$^+$NSGA-II-*C45* significantly outperformed *SBRM*$^+$NSGA-II-*PART* whereas in only 3% (i.e., 3/90) *SBRM*$^+$NSGA-II-*PART* significantly performed better than *SBRM*$^+$NSGA-II-*C45* for the Cisco case study. Similarly, for Jitsi, in 80% (i.e., 82/102) of the total comparisons, *SBRM*$^+$NSGA-II-*PART* significantly outperformed *SBRM*$^+$NSGA-II-*C45* while in only 2% (i.e., 2/102) of the total comparisons *SBRM*$^+$NSGA-II-*C45* significantly performed better than *SBRM*$^+$NSGA-II-*PART*.

Since *SBRM*$^+$*NSGA-II*-*C45* significantly outperformed other the other three *SBRM*$^+$ approaches in terms of MLQMs for the Cisco case study and *SBRM*$^+$NSGA-II-*PART* for the Jitsi case study, we can conclude that given the default parameter settings for both the machine learning algorithms and the search algorithms, *SBRM*$^+$*NSGA-II*-*C45* and *SBRM*$^+$NSGA-II-*PART* produce better rules with respect to MLQMs for the Cisco and Jitsi case studies, respectively.

6.6 Correlation Analysis (RQ6)

To answer RQ6, we compute the correlation coefficients (ρ) and p-values using Non-Parametric Spearman's test for all the MLQMs in correlation to the average fitness values (*AFV*) for the three individual objectives (i.e., *AFV-O1*, *AFV-O2* and *AFV-O3*), overall average fitness values (*OAFV*) and six quality indicators corresponding to both case studies. Through correlation analysis, we intend to test our hypothesis, i.e., *Accuracy, Precision*, *Recall*, and *FMeasure* have positive correlations with *HV* and negative correlations with the average fitness values and the other five quality indicators whereas *MAE*, *RMSE*, *RAE*, and RRSE are negatively correlated with *HV* and positively correlated with the average fitness values and the other five quality indicators (Section 5.1.5). The results of RQ6 are summarized in Table 8 for both Cisco and Jitsi case studies whereas the detailed results can be found in Appendix B.

Table 8: Summary of the correlation analysis' results (RQ6) *

	SBRM*$^+$*NSGA-II*-*C45			***SBRM*$^+$*NSGA-III*-*C45***			***SBRM*$^+$*NSGA-II*-*PART***			***SBRM*$^+$*NSGA-III*-*PART***		
Case study	HS	HR	NS	HS	HR	NS	HS	HR	NS	HS	HR	NS
Cisco	28/170	11/170	131/170	30/140	52/140	58/140	25/140	44/140	71/140	39/140	11/140	90/140
Jitsi	91/170	11/170	68/170	60/170	16/170	94/170	74/170	16/170	80/170	29/170	22/170	119/170

*HS= Hypothesis satisfied, HR= Hypothesis rejected, NS= Not significant

As shown in Table 8, for the Cisco case study, 23% (i.e., 39/170), 59% (i.e., 82/140), 49% (i.e., 69/140), and 36% (i.e., 50/140) of the total correlations are significant for *SBRM*$^+$*NSGA-II*-*C45*, *SBRM*$^+$*NSGA-III*-*C45*, *SBRM*$^+$*NSGA-II*-*PART*, and *SBRM*$^+$*NSGA-III*-*PART* respectively, where 72% (i.e., 28/39), 37% (i.e., 30/82), 36% (i.e., 25/69), and 78% (i.e., 39/50) of significant correlations satisfy our hypothesis (Section 5.1.5). Similarly, for the Jitsi case study, 60% (i.e., 102/170), 45% (i.e., 76/170), 53% (i.e., 90/170), and 30% (i.e., 51/170) of the total correlations are significant for *SBRM*$^+$*NSGA-II*-*C45*, *SBRM*$^+$*NSGA-III*-*C45*, *SBRM*$^+$*NSGA-II*-*PART*, and *SBRM*$^+$*NSGA-III*-*PART* respectively, where 89% (i.e., 91/102), 79% (i.e., 60/76), 82% (i.e., 74/90), and 57% (i.e., 29/51) of significant correlations satisfy our hypothesis (Section 5.1.5).

6.7 Trend Analysis of the Quality of Rules Across the Iterations (RQ7)

To answer RQ7, we study the variation in the quality of rules in terms of MLQMs across the iterations (from *iteration-1* to *iteration-5*) for the *SBRM*$^+$ approaches for both case studies. To do so, we plotted the scatter plots and fitted Linear Regression lines for all the MLQMs. The results of the trend analysis are summarized below, and the plotted graphs are provided in Appendix B.

Table 9: Summary of trend analysis' results (RQ7) *

	SBRM*$^+$*NSGA-II*-*C45			***SBRM*$^+$*NSGA-III*-*C45***			***SBRM*$^+$*NSGA-II*-*PART***			***SBRM*$^+$*NSGA-III*-*PART***		
Case study	IT	DT	ST	IT	DT	ST	IT	DT	ST	IT	DT	ST
Cisco	15/17	1/17	1/17	10/17	0/17	7/17	14/17	0/17	3/17	10/17	2/17	5/17
Jitsi	14/17	0/17	3/17	16/17	1/17	0/17	17/17	0/17	0/17	14/17	0/17	3/17

*IT= Increasing trend of the quality of rules, DT= Decreasing trend of the quality of rules, ST= Straight line (no change in the quality of rules)

As shown in Table 9, for both case studies, we observed an increasing trend of quality of rules in terms of the majority (81%, i.e., 110/136) of the MLQMs for all the four *SBRM*$^+$ approaches across the iterations. Also, for both case studies, we witnessed a slightly decreasing trend in only 3% (i.e., 4/136) of MLQMs for all the four *SBRM*$^+$ approaches whereas in the remaining 16% (i.e., 22/136), we observed a straight line. Note that the quality of rules in terms of the MLQMs increases if values of error related MLQMs (i.e., *MAE*, *RAE*, *RMSE*, and *RRSE*) decrease and other MLQMs increase.

6.8 Cost of Applying Search to Generate Configurations (RQ8)

To answer RQ8, we calculated the average time required by NSGA-II (in *SBRM*$^+_{NSGA-II}$*-C45* and *SBRM*$^+_{NSGA-II}$*-PART*), NSGA-III (in *SBRM*$^+_{NSGA-III}$*-C45* and *SBRM*$^+_{NSGA-III}$*-PART*), and RS (in *RBRM*$^+$*-C45* and *RBRM*$^+$*-PART*) to generate configurations per iteration (i.e., *ATPI*) as well as per cycle (i.e., *ATPC*) (Section 5.1.3). Table 10 shows the average time required by *SBRM*$^+_{NSGA-II}$*-C45*, *SBRM*$^+_{NSGA-III}$*-C45*, *SBRM*$^+_{NSGA-II}$*-PART*, *SBRM*$^+_{NSGA-III}$*-PART*, *RBRM*$^+$*-C45*, and *RBRM*$^+$*-PART* to generate configurations per iteration and per cycle for both of the case studies.

Table 10: Average time (minutes) required for generating configurations

Case Study	Metric	*SBRM*$^+_{NSGA-II}$*-C45*	*SBRM*$^+_{NSGA-III}$*-C45*	RBRM$^+$-C45	*SBRM*$^+_{NSGA-II}$*-PART*	*SBRM*$^+_{NSGA-III}$*-PART*	RBRM$^+$-PART
Cisco	ATPI	22	3224	18	23	8765	22
	ATPC	108	16118	90	116	43824	108
Jitsi	ATPI	32	22527	40	10	10179	21
	ATPC	159	112636	199	52	50896	103

From Table 10, we can observe that the costs of generating configurations using *SBRM*$^+_{NSGA-II}$*-C45*, *SBRM*$^+_{NSGA-II}$*-PART*, *RBRM*$^+$*-C45*, and *RBRM*$^+$*-PART* are quite comparable. However, *SBRM*$^+_{NSGA-III}$*-C45* and *SBRM*$^+_{NSGA-III}$*-PART* took significantly more time than the others, because NSGA-III is significantly slower than NSGA-II and RS. Also, NSGA-III produces only 92 solutions for the three objective problems regardless of its population size [23], thus, we executed it multiple times to get 500 configuration solutions corresponding to each iteration. We used a fixed number of fitness evaluations instead of time budget as the termination criterion of the search because 1) different frameworks for multi-objective optimization with metaheuristics (e.g., jMetal [71]) use fitness evaluations instead of time budget; 2) A fixed number of fitness evaluations are widely applied in SBSE [80-83]; 3) We used 50,000 fitness evaluations as termination criterion, because we were able to obtain good results in our earlier studies involving industrial datasets [41, 81]; and 4) We think comparing search algorithms based on fixed time is biased towards faster algorithms, as a slower one gets less chance to evolve towards a better solution, particularly in the context where the time cost of executing an approach is not important which is the case of applying our approach.

From Table 10, we can also notice that *SBRM*$^+_{NSGA-II}$*-C45*, *SBRM*$^+_{NSGA-III}$*-C45*, and *RBRM*$^+$*-C45* took more time than *SBRM*$^+_{NSGA-II}$*- PART*, *SBRM*$^+_{NSGA-III}$*- PART*, and *RBRM*$^+$*- PART* respectively. This can be explained as C4.5 produced lengthier rules (i.e., more predicates) than PART (Table 11). Thus, approaches producing lengthier rules have a higher cost of calculating fitness values and consequently higher execution time. On average, C4.5 produced 1.7 and 2 more predicates per rule than PART for the Cisco and Jitsi case studies, respectively.

Table 11: Average number of predicates for the Cisco and Jitsi case studies

Approach	Cisco		Jitsi	
	Avg. predicates per rule	**Avg. predicates per run**	**Avg. predicates per rule**	**Avg. predicates per run**
SBRM$^+_{NSGA-II}$-C45	**5.2**	14420	**4.6**	100095
SBRM$^+_{NSGA-II}$*-PART*	**3.6**	17102	**2.7**	22764
SBRM$^+_{NSGA-III}$-C45	**5.7**	14853	**5.5**	134451
SBRM$^+_{NSGA-III}$*-PART*	**4.0**	20668	**2.7**	20498
RBRM$^+$-C45	**5.6**	21763	**4.0**	106454
RBRM$^+$*-PART*	**3.9**	26632	**2.6**	25208

6.9 Discussion

For RQ1, we noticed that NSGA-II and NSGA-III significantly outperformed RS in terms of all the fitness values and majority of the quality indicators for both of the case studies (Section 6.1). This can be simply explained as NSGA-II and NSGA-III generate and select better solutions using operators such as mutation and crossover. We also noticed that RS performed better than NSGA-II and NSGA-III in terms of *GS* (representing the diversity of obtained solutions) for 15/24 and 17/24 comparisons for the Cisco and Jitsi case studies, respectively. This is because 1) for our problem, higher convergence to the objectives (e.g., *Objective-1* avoids generating configurations satisfying high confidence rules with normal states) may reduce the search space to be explored, which consequently affects diversity negatively; and 2) RS explores the search space more uniformly as compared to other algorithms [84], thus, the solutions produced by RS has high diversity but low convergence as shown by the results of RQ1 (Section 6.1).

For RQ2, we observed that in 7 out of 8 comparisons for both of the case studies, *SBRM*$^+$ performed significantly better than the two *RBRM*$^+$ approaches in terms of the majority of MLQMs whereas in one of the 8 comparisons there was no significant difference observed between the two approaches (i.e., *SBRM*$^+_{\text{NSGA-III}}$-*C45* and *RBRM*$^+$-*C45*) (Section 6.2). *SBRM*$^+_{NSGA\text{-}II}$-*C45*, *SBRM*$^+_{NSGA\text{-}III}$-*C45*, *SBRM*$^+_{NSGA\text{-}II}$-*PART*, and *SBRM*$^+_{NSGA\text{-}III}$-*PART* have achieved an ARI up to 27%, 22%, 18%, and 21% for the Cisco case study respectively (Section 6.3). Similarly, for the Jitsi case study, *SBRM*$^+_{NSGA\text{-}II}$-*C45*, *SBRM*$^+_{NSGA\text{-}III}$-*C45*, *SBRM*$^+_{NSGA\text{-}II}$-*PART*, and *SBRM*$^+_{NSGA\text{-}III}$-*PART* have achieved an ARI up to 28%, 4%, 8%, and 12% respectively (Section 6.3). This is because the three objectives use previously mined rules for guiding the search to generate configurations that increase the support of the correct rules and filter out incorrect ones. In addition, the operators of NSGA-II and NSGA-III help *SBRM*$^+$ to converge faster than *RBRM*$^+$.

For RQ4, NSGA-III significantly outperformed NSGA-II in terms of fitness values and the quality indicators in most of the cases. For RQ5, *SBRM*$^+_{\boldsymbol{NSGA\text{-}II}}$-*C45* significantly outperformed other three *SBRM*$^+$ approaches in producing better quality rules in terms of MLQMs for the Cisco case study and *SBRM*$^+_{\text{NSGA-II}}$-*PART* for the Jitsi case study. This deviation in the results for the two case studies could be explained as follow: 1) The number of categorical configurable parameters is different (15 for Cisco and 27 for Jitsi); 2) The maximum number of possible configurations for a categorical configuration parameter is different (4 for Cisco and 16 for Jitsi); 3) The total number of configurable parameters is different (27 for Cisco and 39 for Jitsi); and 4) the configuration spaces are different ($1.03e^{33}$ for Cisco and $6.54e^{60}$ for Jitsi). The categorical parameters are of more importance because satisfying the predicates with the categorical parameters in the rules is more difficult than satisfying the predicates with numerical parameters. This is because usually in the rules, predicates with numerical parameters allow a large number of values to satisfy the predicates, whereas satisfying predicates with categorical parameters requires exact values from predefined candidate values. The different characteristics of the case studies could make different algorithms suitable for mining the rules. Based on the characteristics of the two selected case studies and their corresponding results, we can argue that PART is a preferred choice to integrate with NSGA-II (i.e., *SBRM*$^+_{NSGA\text{-}II}$-*PART*) for mining rules for a relatively larger case study whereas C4.5 (*SBRM*$^+_{NSGA\text{-}II}$-*C45*) is a better choice in the case of a smaller sized case study. Nevertheless, these results cannot be generalized based on the evaluation of merely two case studies. Besides, the selection of the machine learning algorithms and their parameter settings are usually application dependent. Thus, generalizing the results further requires a much larger scale empirical evaluation with more case studies.

From the *correlation analysis* for RQ6, we noticed that the majority of cases satisfy our hypothesis (Section 5.1.5) that the overall quality of rules in terms of MLQMs improves by reducing the average fitness values and quality indicators (except for *HV*) and increasing *HV*. However, smaller average fitness values and quality indicators (except for *HV*) and larger *HV* do not mean that all the MLQMs will always be improved, as we observed several cases (e.g., correlations of *FF-Precision* and *CF-Recall* with *AFV-O3* corresponding to *SBRM*$^+_{\boldsymbol{NSGA\text{-}II}}$-*PART* for the Cisco case study, correlations of *CC-Recall* and *CC-FMeasure* with *AFV-O1* corresponding to *SBRM*$^+_{\boldsymbol{NSGA\text{-}II}}$-*PART* for the Jitsi case study) that reject our hypothesis. It is quite possible that certain MLQMs are affected negatively due to several reasons, 1) *Objective-1* avoids generating the configurations satisfying high confidence rules with *ConnectedConnected* class due to which mining algorithm will give more preference to other classes (i.e., *FailedFailed*, *FailedConnected*, and *ConnectedFailed*), therefore, MLQMs such as *CC-Recall* and *CC-FMeasure* may decrease with the decrement in *AFV-O1* as it did for the Jitsi case study; 2) *Objective-2* and *Objective-3* generate configurations satisfying low confidence (i.e., higher violation and lower support) rules with normal and abnormal states, which increase the violation of low confidence rules that may affect MLQMs negatively in certain cases (e.g., when violation of rules increased but not enough to remove them from rule set) as it did for the Cisco case study. In such cases, MLQMs may decrease with the reduction in *AFV-O2* and *AFV-O3*.

For RQ7, we noticed an increasing trend of the quality of rules based on the majority of MLQMs for all the four *SBRM*$^+$ approaches for both case studies. This is because, in each new iteration, we refined the rules by generating the configurations based on the rules mined from the previous iteration and mining a new set of refined rules, which improves the quality based on MLQMs in each new iteration. Thus, the incremental, iterative process refines rules across iterations, and the number of iterations does have an impact on the results. For RQ8, the best performing *SBRM*$^+_{NSGA\text{-}II}$*-C45* took 108 minutes for the Cisco case study whereas *SBRM*$^+_{NSGA\text{-}II}$*-PART* took 52 minutes corresponding to the Jitsi case study, for generating configurations for a complete cycle, which is acceptable as it is a one-time process.

Furthermore, to know the distribution of the mined rules associated with the four system states (*ConnectedConnected*, *FailedFailed*, *ConnectedFailed*, and *FailedConnected*) in the five iterations for both case studies, we plotted stacked column plots. Note, we have also presented the distribution of the rules for the iteration zero, to be complete. Figure 9 presents the average numbers of rules mined with the different approaches for the Cisco case study. From Figure 9, we can see that *RBRM*$^+$*-C45* (*RBRM*$^+$*-PART*) produced more rules than *SBRM*$^+_{NSGA\text{-}II}$*-C45* and *SBRM*$^+_{NSGA\text{-}III}$*-C45* (*SBRM*$^+_{NSGA\text{-}II}$*-PART* and *SBRM*$^+_{NSGA\text{-}III}$*-PART*) in all the five iterations except that *SBRM*$^+_{NSGA\text{-}II}$*-C45* produced slightly more rules than *RBRM*$^+$*-C45* in *iteration-1* and *iteration-2*. We can also notice that no rules were produced for *FailedConnected* in the first three iterations and significantly fewer numbers of rules produced for *FailedConnected* (to compare with the other categories) in only *iteration-4* and *iteration-5*.

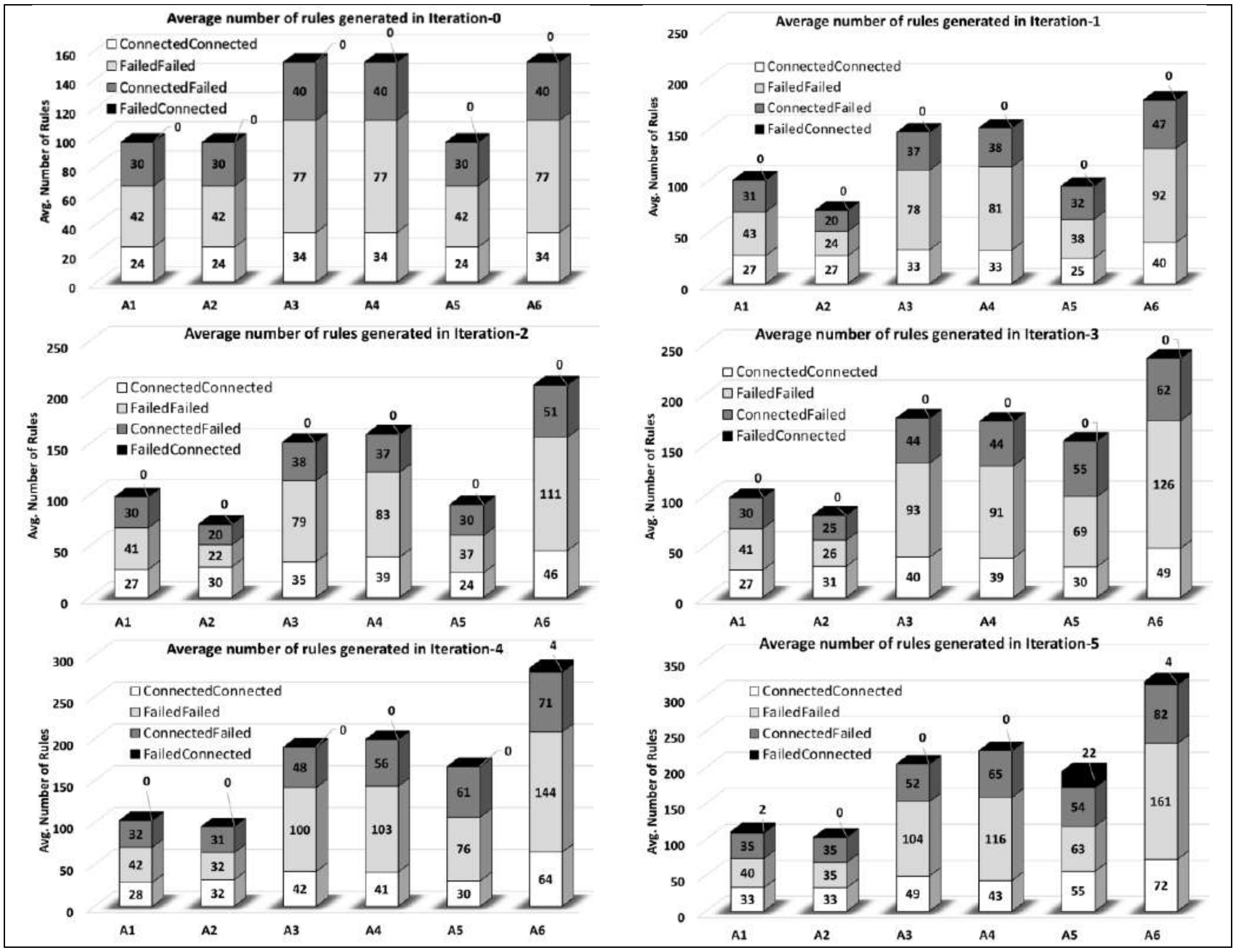


Figure 9: Average numbers of rules mined in each iteration for the Cisco case study*

*A_1= SBRM$^+_{NSGA\text{-}II}$-C45, A_2= SBRM$^+_{NSGA\text{-}III}$-C45, A_3= SBRM$^+_{NSGA\text{-}II}$-PART, A_4= SBRM$^+_{NSGA\text{-}III}$-PART, A_5= RBRM$^+$-C45, A_6= RBRM$^+$-PART

Figure 10 presents the average numbers of rules mined for the Jitsi case study. From the figure, we can observe that *RBRM*$^+$-*C45* (*RBRM*$^+$-*PART*) produced more rules than *SBRM*$^+_{NSGA\text{-}II}$-*C45* and *SBRM*$^+_{NSGA\text{-}III}$-*C45* (*SBRM*$^+_{NSGA\text{-}II}$-*PART* and *SBRM*$^+_{NSGA\text{-}III}$-*PART*) in all the iterations just as for the Cisco case study. For both of the case studies, we observed that the *SBRM*$^+$ approaches produced less number of rules than the two *RBRM*$^+$ approaches. This is because the three objectives refine the rules by removing low confidence incorrect rules and the search operators (i.e., mutation, crossover, and selection) help *SBRM*$^+_{NSGA\text{-}II}$-*C45* and *SBRM*$^+_{NSGA\text{-}II}$-*PART* to get optimal configurations in terms of three objectives.

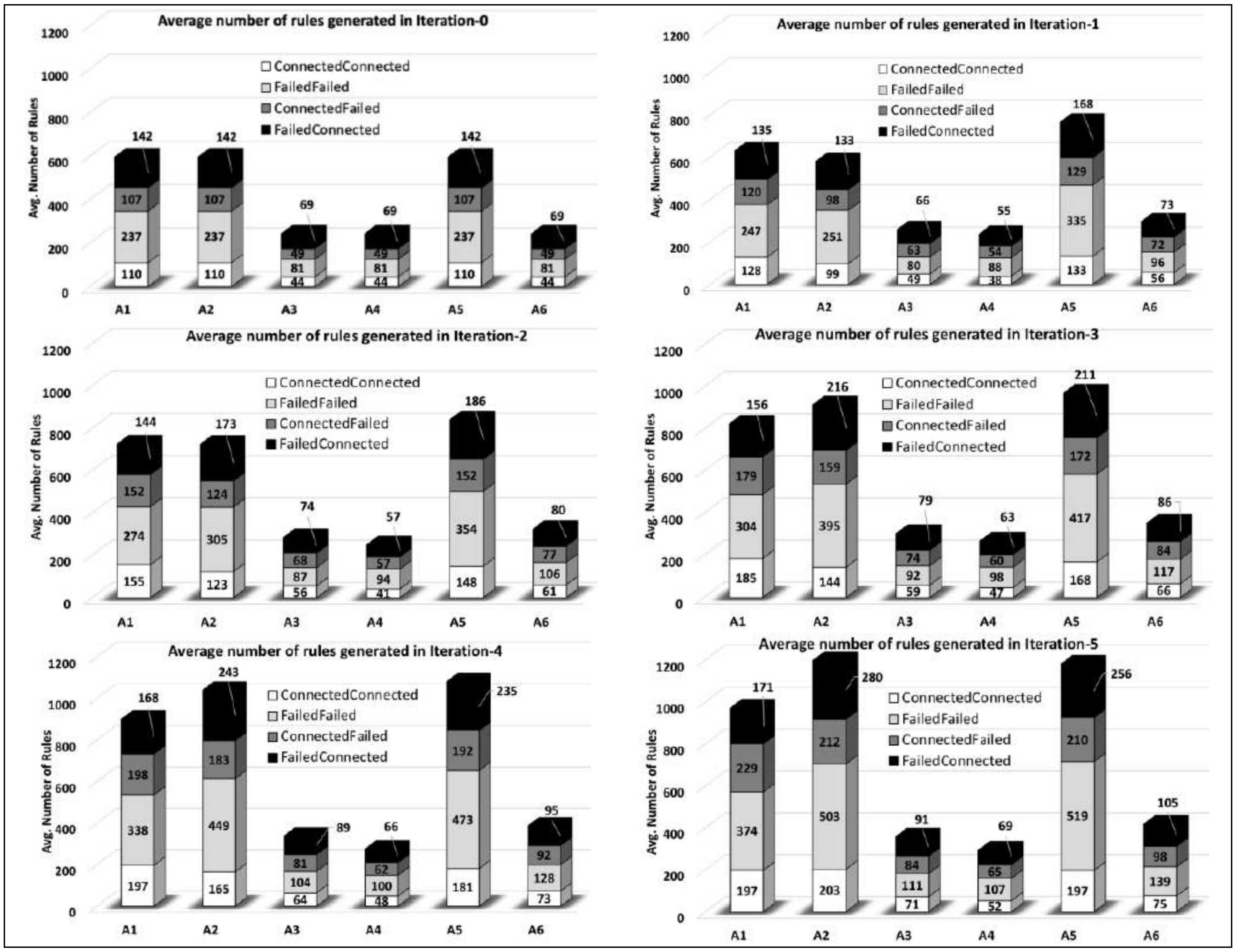


Figure 10: Average numbers of rules mined in each iteration for the Jitsi case study*

*A_1= SBRM$^+_{NSGA\text{-}II}$-C45, A_2= SBRM$^+_{NSGA\text{-}III}$-C45, A_3= SBRM$^+_{NSGA\text{-}II}$-PART, A_4= SBRM$^+_{NSGA\text{-}III}$-PART, A_5= RBRM$^+$-C45, A_6= RBRM$^+$-PART

Figure 11 shows the distribution of rules with respect to normal and abnormal system states, obtained using *SBRM*$^+$ for both case studies. From Figure 12, one can see that the majority of the rules produced are rules with abnormal system state, which is expected because *SBRM*$^+$ focused on generating invalid configurations.

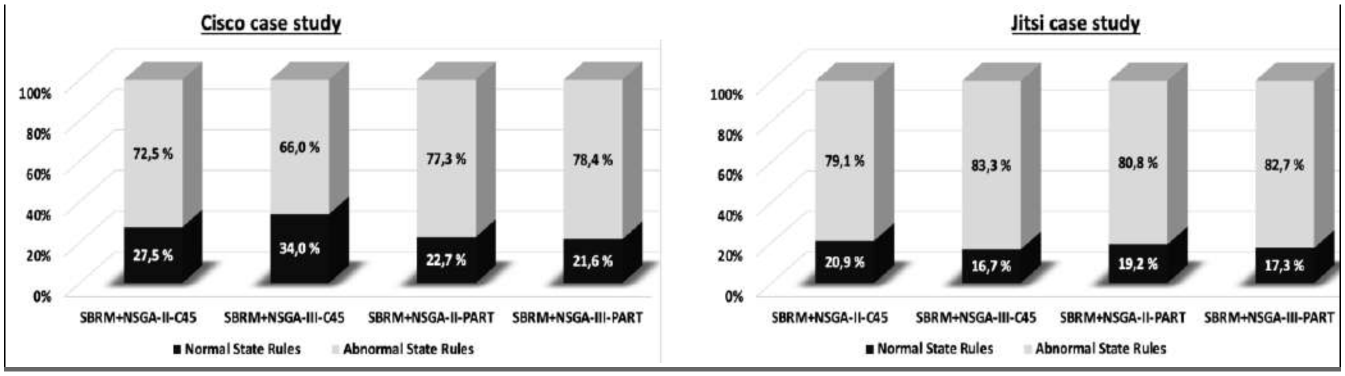


Figure 11: Rules distribution w.r.t. system states

To see the distribution of configurations with respect to the corresponding system states (i.e., valid or invalid), we collected the statistics about configurations generated using *SBRM*$^+$ for both case studies, which are shown in Figure 12. However, it is worth mentioning that the distribution of generated configurations is greatly influenced by the input rules provided to the search algorithms.

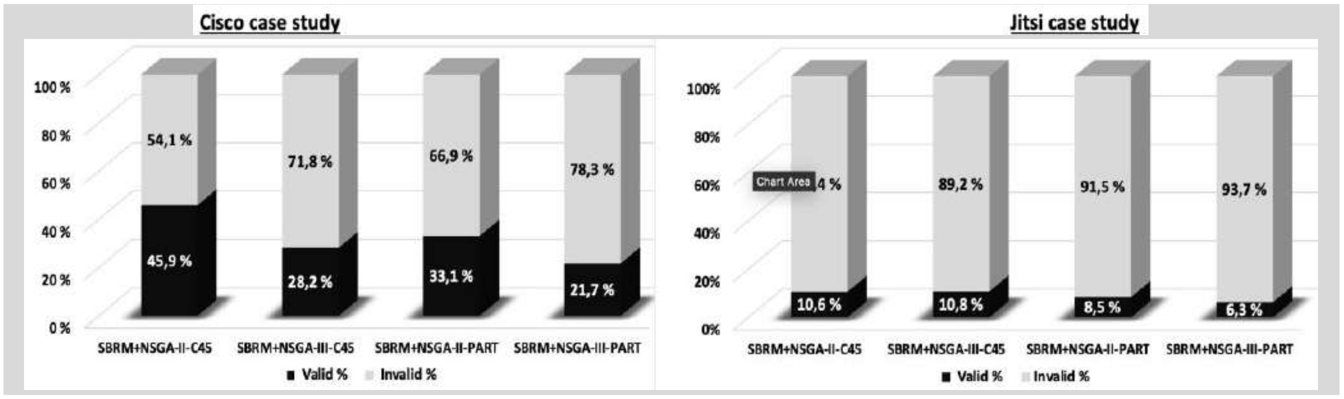
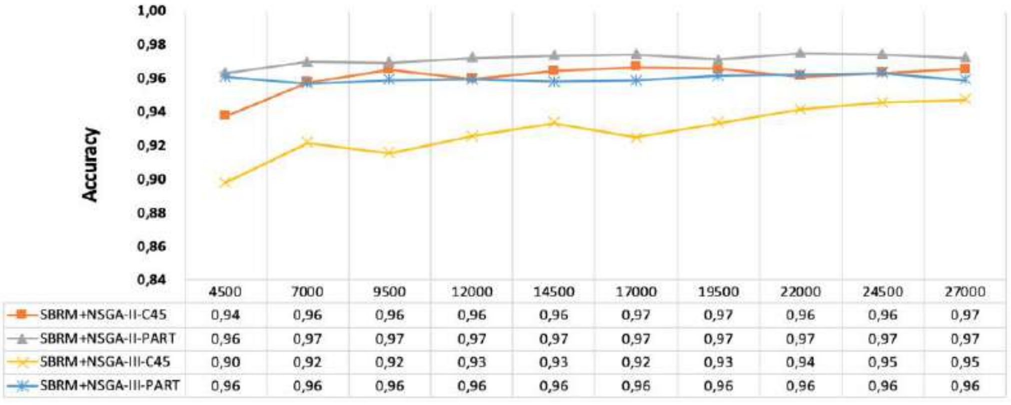


Figure 12: Average numbers configurations with of valid and invalid system states per run

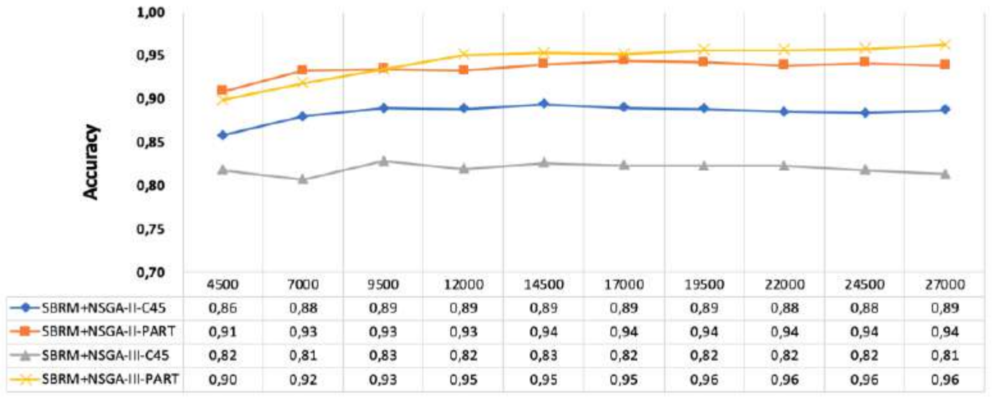
Moreover, we intend to assess if adding more iterations increases the quality of rules significantly. Due to high execution cost of the experiments, we combined configurations from 10 runs of already executed experiments and mine rules to see the trend of quality improvement of rules with respect to the dataset size. More specifically, first, we mine the rules using configurations of the first run and then incrementally add the configurations from other nine runs and mine the rules. Note, for the first run we used all the 4500 configurations whereas, for other 9 runs, we have added only 2500 configurations per run (i.e., for five iterations) because the initial 2000 configurations (i.e., randomly generated) are common across all the runs. To show the trend, we plotted the MLQMs against the number of instances (i.e., configurations) in the dataset. Due to limited space, we have selected *Accuracy* as a representative MLQM to illustrate the trend (Figure 13 and Figure 14).



	4500	7000	9500	12000	14500	17000	19500	22000	24500	27000
SBRM+NSGA-II-C45	0,94	0,96	0,96	0,96	0,96	0,97	0,97	0,96	0,96	0,97
SBRM+NSGA-II-PART	0,96	0,97	0,97	0,97	0,97	0,97	0,97	0,97	0,97	0,97
SBRM+NSGA-III-C45	0,90	0,92	0,92	0,93	0,93	0,92	0,93	0,94	0,95	0,95
SBRM+NSGA-III-PART	0,96	0,96	0,96	0,96	0,96	0,96	0,96	0,96	0,96	0,96

Figure 13: Accuracy vs. number of instances in the dataset for Cisco

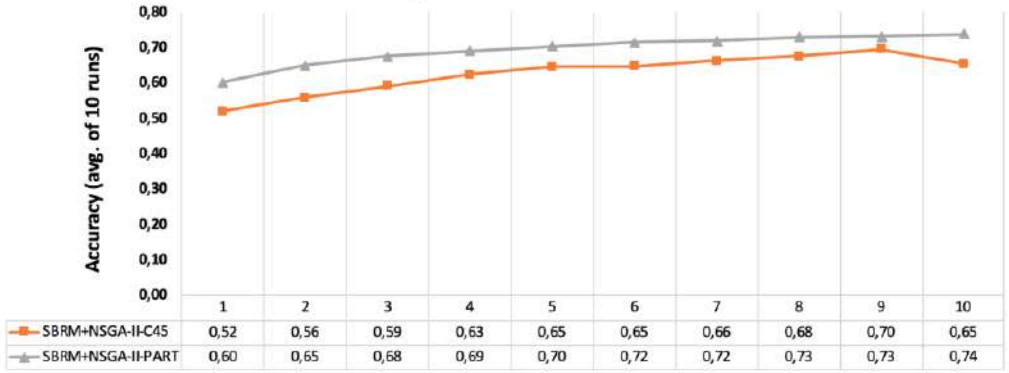
As shown in Figure 13 and Figure 14, for both of the case studies, there is an improvement in the quality of rules, but not significant. From 4500 to 22,500 instances (i.e., configurations), we get up to 5% of improvement for the Cisco case study and 6% for the Jitsi case study. Note, for the other MLQMs, we also observed similar results.



	4500	7000	9500	12000	14500	17000	19500	22000	24500	27000
SBRM+NSGA-II-C45	0,86	0,88	0,89	0,89	0,89	0,89	0,89	0,88	0,88	0,89
SBRM+NSGA-II-PART	0,91	0,93	0,93	0,93	0,94	0,94	0,94	0,94	0,94	0,94
SBRM+NSGA-III-C45	0,82	0,81	0,83	0,82	0,83	0,82	0,82	0,82	0,82	0,81
SBRM+NSGA-III-PART	0,90	0,92	0,93	0,95	0,95	0,95	0,96	0,96	0,96	0,96

Figure 14: Accuracy vs. number of instances in the dataset for Jitsi

We assess the trend of quality of rules against different dataset sizes. However, one can argue that we added the configurations generated using rules from iteration zero to iteration-4 and adding configurations generated using rules from iteration 5 and onwards would have improved the quality of rules significantly. To cater this argument, we selected the best performing approaches *SBRM*$^{+}_{\textit{NSGA-II}}$*-C45* for the Cisco case study and *SBRM*$^{+}_{\textit{NSGA-II}}$*-PART* for the Jitsi case study and conducted the experiment with these two approaches to obtain five more iterations (i.e., in total 10 iterations) with the Jitsi case study. This is done only for the Jitsi case study because the experiment can be run on a cluster. However, for the Cisco case study, running the experiment needs dedicated hardware equipment and we cannot run the experiment in parallel due to the limited number of VCSs available, which makes the experiment extremely time-consuming. Figure 15 shows the average *Accuracy* (i.e., calculated as the average of 10 runs for each iteration) across the 10 iterations. From Figure 15, we can observe an improvement in the quality of rules across the iterations, however, we got an improvement of 4% at maximum for any approach from iteration-5 to iteration-10. On the other hand, when looking at the improvement from iteration-1 to iteration-5, we got an improvement of 13% for *SBRM*$^{+}_{\textit{NSGA-II}}$*-C45* and 10% for *SBRM*$^{+}_{\textit{NSGA-II}}$*-PART*. Thus, it would be fair to say that after a number of iterations (e.g., five in our case), the improvement will be very slow. This suggests that using a fixed number of iterations is a practical and wise approach to terminate the process.



	1	2	3	4	5	6	7	8	9	10
SBRM+NSGA-II-C45	0,52	0,56	0,59	0,63	0,65	0,65	0,66	0,68	0,70	0,65
SBRM+NSGA-II-PART	0,60	0,65	0,68	0,69	0,70	0,72	0,72	0,73	0,73	0,74

Figure 15: Average accuracy across 10 iterations for the Jitsi case study

6.10 Threats to Validity

In Section 6.10.1, we discuss threats to the internal validity followed by threats to the construct validity in Section 6.10.2. We discuss threats to the conclusion validity and external validity in Section 6.10.3 and Section 6.10.4, respectively.

6.10.1 Internal Validity

Threats to the internal validity exist when the results are influenced by the internal factors such as parameter settings [85]. The first threat to the internal validity is the selection of search algorithms in our study. To mitigate this threat, we selected the most widely used NSGA-II algorithm, which has shown promising results in different contexts [19, 20]. Moreover, we have selected a relatively new multi-objective search algorithms, i.e., NSGA-III, which also has good performance on addressing many objective problems [25]. The second threat is the selection of algorithms for rule mining. We selected PART as it has been proven to be more effective than many well-known algorithms [14, 30] and C4.5, the most popular algorithm in industry and the research community [31, 86]. The third threat is the selection of parameter settings for the selected search algorithm. To mitigate this threat, we used default parameter settings, which have exhibited promising results [87]. Similarly, for the machine-learning algorithms, we also used the default parameters settings, which perform reasonably well [12, 31]. Another threat is the selection of the *Confidence* measure for calculating fitness values, as there exist other measures (e.g., *Lift*). We acknowledge that this is a threat to the internal validity and dedicated experiments are needed for further investigation.

6.10.2 Construct Validity

Threats to the construct validity exist when the measurement metrics do not sufficiently cover the concepts they are supposed to measure [85, 88]. To mitigate this threat, we compared different approaches using the same comprehensive set of measures: fitness values, quality indicators, and 17 MLQMs, which are commonly used in the literature [31, 36, 89].

6.10.3 Conclusion Validity

Threats to the conclusion validity concern with the factors influencing the conclusion drawn from the results of the experiment [90]. The most probable threat to the conclusion validity is due to the random variation inherent in search algorithms. To minimize this threat, we repeated the experiment 10 times (i.e., total 50 runs of each search algorithm) to reduce the effect caused by randomness, as recommended in [74, 91]. Moreover, we also applied the Mann-Whitney test to determine the statistical significance of the results and the Vargha and Delaney $\hat{A}_{12}$ statistics as the effect size measure, which are recommended for randomized algorithms [74, 91].

6.10.4 External Validity

The external validity concerns with the generalization of the experiment results to other contexts [85]. The threat to the external validity for our experiment is the case studies selected for the evaluation. In our study, we used a real-world case study (i.e., Cisco Video Conferencing Systems) and an open source case study Jitsi of different sizes. Furthermore, one can argue that the complexity of case studies (i.e., a large number of configurable parameters and system states) may affect the performance of proposed approach. We would like to argue that multi-objective search algorithms such as NSGA-II and NSGA-III have been applied to problems of different complexity, and they have proven to be quite effective [15-17, 25, 46, 88]. However, higher dimensional datasets (more attributes) for complex case studies, may reduce the performance (e.g., accuracy, precision) of machine learning algorithms but the impact will be the same for both *SBRM*$^+$ and *RBRM*$^+$, as both approaches employ a machine learning algorithm.

7. RELATED WORK

Search algorithms have been used to solve many problems in the context of PLE [15-17, 46, 88]. In this paper, we also combined the search with machine learning techniques to mine the rules in the context of PLE. The related work to this research stream focuses on existing studies presenting the approaches to mine the rules in the context of PLE. In Section 7.1, we discuss dedicated approaches that focus on mining rules from different artifacts (e.g., source code, configuration file, feature model) of product lines. Furthermore, in Section 7.2, we discuss approaches such as feature extraction, feature construction and feature recommendation, which mine crosstree constraints. Finally, in Section 7.3, we summarize the related work and compare it with our work.

Table 12: Characteristics of existing rule mining approaches*

Reference	Topic	Input	Output	ML Technique for mining rules	Configurable parameter type	Data generation/selection	# of classes	Evaluation	Case study
[11]	Configuration constraint extraction for PL	A FM and an oracle (computer vision algorithm)	A set of configuration constraints	Binary Decision Tree-J48 (implementation of C4.5)	Numerical and Categorical	Randomly	2	Precision, Recall, and expert opinion	A real-world PL of video generator
[92]	Constraint extraction for FM	A FM (features, feature description, and known binary crosstree constraints)	A set of crosstree constraints	LIBSVM classifier and genetic algorithm	Categorical	Randomly	3	Precision, Recall, and FMeasure	Two open source feature models of Weather Station and Graph PL
[93]	Constraint extraction for FM	Configuration files	A set of crosstree constraints	Apriori algorithm	Categorical	NA	NA	Support and confidence	An industrial PL of embedded systems
[94], [10]	Configuration constraint extraction	C code	A set of hierarchy and crosstree constraints	Static analysis (Build and code analysis)	Categorical	NA	NA	Accuracy measured in reference to the rules defined by the expert and recoverability	Four open source case studies (uClibc, BusyBox, eCos, and the Linux kernel)
[95]	Introducing probabilistic FM and provide a process to extract the crosstree constraints for FM	Formally defined FM inform of propositional formula	A set of crosstree constraints	Apriori algorithm and an algorithm presented in [96]	Categorical	NA	NA	Support and confidence	A small sized PL of Java applets
[97], **[98]**	Feature extraction and constraint extraction for recommending features	Product descriptions available on internet and user requirements	FM along with crosstree constraints	CFP-growth algorithm and Apriori algorithm	Categorical	NA	NA	Support and confidence	A PL of remote collaboration and description of 20 PL collected from SoftPedia
[5]	Feature extraction and constraint extraction	Product description, product comparison metrics, manually added	Attributed FM with crosstree constraints	Proposed algorithms implemented in Scala and SAT4J solver for computation of or-group	Numerical and Categorical	Randomly	NA	Quality of the rules was not evaluated	242 configuration matrices generated randomly and a real-world case study of

		domain knowledge							bestbuy.com
[27]	Feature extraction and constraint extraction	Product descriptions	FM with crosstree constraints	CFP-growth algorithm and Apriori algorithm	Categorical	NA	NA	Support, confidence, accuracy with respect to manually constructed FM	A PL of antivirus collected from SoftPedia
[99]	Feature extraction and recommendation	Feature description and user requirements	Feature recommendation based on the association rules	Apriori algorithm	Categorical	NA	NA	Support and confidence	A large number of datasets collected from three repositories SoftPedia, SourceForge, and FreeCode

* FM= feature model, PL= product line, NA= Not applicable, ML= Machine learning, LDA= Latent Dirichlet Allocation

7.1 Dedicated Rule Mining Approaches

The work in [11] applies Binary Decision Tree-J48 (machine learning algorithm) to infer the constraints from a set of randomly generated product configurations. To classify the configurations as faulty and non-faulty, a computer vision algorithm was used as an oracle. To validate the approach, it was applied to an industrial video generator product line. Rules were evaluated based on expert's opinion and machine-learning measurements such as Precision and Recall. Results show that on average 86% Precision and 80% Recall rate can be achieved using the proposed approach.

In [92], an approach for mining the crosstree binary constraints (i.e., requires, excludes) corresponding to a feature model is presented. The approach takes a feature model as input containing the features, their descriptions, and some known crosstree binary constraints. First, it trains LIBSVM classifier (an extension of support vector machine) with existing crosstree binary constraints where the parameters of the classifier are optimized using the genetic algorithm to minimize the error rate of the classifier. Second, it extracts all the feature pairs, and finally, the optimized classifier finds the candidate features of binary constraints. The approach was validated using two feature models collected from SPLOT repository. Results show that rules with high *Recall* (i.e., close to 100%) and the variable low *Precision* (on average 42%) can be achieved using proposed approach.

In [93], another approach is presented for mining the crosstree constraints. It constructs configuration matrix (i.e., product-features matrix) from configuration files and extracts crosstree constraints using an association rule mining technique (i.e., Apriori algorithm). Rules are pruned using minimum support and minimum confidence thresholds. The approach was evaluated using a large-scale industrial software product line for embedded systems. The evaluation shows that a large number of rules with variable support (i.e., 80% to 99%) and confidence (i.e., 90% to 100%) can be identified. The majority of the rules were identified with support ranging from 80% to 85%.

The work in [94] presents an approach to extract configuration constraints from existing C codebases using static analysis. It uses build time errors (e.g., preprocessor, parser, type, and link errors) as the oracle to classify the low-level system configurations (i.e., build and code files) and mine the constraints. To assess the accuracy of extracted rules, they were compared with the existing constraints specified in developer's created variability models. The approach was validated using four open source case studies (uClibc, BusyBox, eCos, and the Linux kernel). Results show that up to 19% of the total constraints can be recovered automatically from the source code, which assures successful build with the accuracy of 93%. In [10], an extension of [94] is presented in which the authors improved the static analysis and increased the recoverability rate by 9%. Additionally, an empirical study is also presented that identifies the sources of constraints.

7.2 Non-Dedicated Rule Mining Approaches

The work in [100] reported a Systematic Literature Review (SLR) of 13 approaches for feature extraction from natural language requirements. The results of SLR show that hybrid natural language processing approaches are commonly used in the overall feature extraction process. Various clustering approaches from data mining and information retrieval are used to group the common features. Moreover, several approaches have also employed association mining techniques to discover the pattern of the features to recommend the relevant features to the stakeholders. In [95], an extension of feature model called probabilistic feature model is introduced. To extract crosstree constraints from existing formally defined products, a rule mining process is presented that uses an association mining technique (i.e., Apriori Algorithm) to mine the conjunctive association rule and an algorithm proposed in [96] to mine the Disjunctive association rules. The proposed mining process was applied to a small case study of Java Applets. Rules were evaluated based on machine-learning measurements (i.e., support and confidence).

In [97], an approach is proposed to model and recommend product features for any particular domain based on the product description provided by the domain expert. To mine association rules between product features, association rule mining techniques are applied to configuration matrix (i.e., product-features matrix). The proposed approach was validated with 20 different product categories using product descriptions available at SoftPedia. Hariri et al. [98] extended the work presented in [97]. In [98], different clustering algorithms used to cluster the features and construct products by feature matrix were compared. The evaluation was also improved by applying the approach on diverse domains as well as a large project of a software suite for remote collaboration. Results show that rules with different *Precision* and *Recall* rates can be mined according to the threshold set for the confidence.

The work in [5] presents an approach to synthesize attributed feature models (AFM) from a set of product descriptions in the form of tables (i.e., configuration matrix). An algorithm is proposed that uses implication graph and mutex graph constructed from configuration matrix to extract the crosstree constraints. For extracting the relational constraints defined on values of attributes, the algorithm uses domain knowledge or selects the boundary values of attributes randomly when domain knowledge is not provided. The approach was validated using random configuration matrices as well as a real-world case study. Results show that the proposed algorithm can be used to mine a large number of rules for large-scale case studies.

The work in [27] proposed an approach to construct a feature model automatically from informal product descriptions available over the Internet. To mine the implication rules of features, CFP-growth algorithm and Apriori algorithm are applied to configuration matrix (i.e., product-features matrix). The proposed approach was applied to a case study of antivirus software using the product descriptions available at SoftPedia.

In [99], an approach is proposed to extract the features from multiple web repositories, organize, analyze, and recommend the high-quality features to the stakeholders. The proposed approach first extracts the information from the Internet repositories and then builds feature ontologies by employing Latent Dirichlet Allocation and clustering. To mine the hidden relationships among software features and to recommend high-quality features to the stakeholders, the proposed approach employs the association rule mining technique (i.e., the Apriori algorithm). The proposed approach is validated using a large number of datasets collected from three repositories (i.e., SoftPedia, SourceForge, and FreeCode).

7.3 Summary

In Table 12, we summarize the existing rule mining techniques and highlight their characteristics. From Table 12, one can see that, (1) all the techniques except [11] are focusing on mining binary crosstree constraints (requires and excludes) between different features of a product line or rules constraining the values of features' attributes in the case of [5]; (2) the majority of the approaches except two ([11] and [92]) are using unsupervised learning based association mining techniques such as Apriori algorithm and FP-growth algorithm; (3) none of the existing approaches have any sophisticated way to select/generate the configurations, and usually, configurations are generated/selected randomly or used existing configurations; (4) the majority of the approaches except two are focusing on only categorical type configurable parameters, however, [11] and [5] are also catering numerical configurable parameters; (5) all the existing approaches are using machine learning quality measurements such as Precision, Recall, Support, and Confidence; and 6) above all, none of the existing approaches are mining the rules for interacting products within/across the product lines.

In contrast to the existing rule mining techniques, we have proposed an incremental and iterative approach in which we generate the configurations smartly and feed the configurations to the machine-learning tool and apply supervised learning based rule mining techniques (i.e., PART and C45), to mine the rules between configurable parameters and system behaviors of interacting products across product lines. The innovative part of our approach is the data generation strategy and incremental, iterative nature, which helps to achieve rules with higher quality as compared to randomly selected configurations based approaches. To generate the configurations, we defined three objectives (Section 4.2) and combined them with the search algorithms (i.e., NSGA-II and NSGA-III). To evaluate the quality of rules, we used machine learning quality measurements, which are also used by existing rule-mining approaches in the literature.

8. CONCLUSION AND FUTURE WORK

Today, systems are being developed by integrating multiple products within/across the product lines that communicate with each other through different communication mediums (e.g., the Internet). The runtime behavior of these systems does not only depend on product configurations, but also on the communication medium. To identify the invalid configurations where these products may fail to communicate, we mine the Cross-Product Line (CPL) rules. To do so, in our previous work, we proposed an incremental and iterative approach named as Search-Based Rule Mining (*SBRM*), in which we combined the widely used multi-objective search algorithm (NSGA-II) with the machine learning algorithm (PART). To use the search in the rule mining process, we defined three objectives and integrated them with the multi-objective optimization algorithm NSGA-II. In this paper, we improved the previously proposed *SBRM* (named as *SBRM*$^+$) and incorporated two multi-objective search algorithms (i.e., NSGA-II and NSGA-III) and two machine learning algorithms (i.e., C4.5 and PART) to mine the rules. Moreover, in *SBRM*$^+$, we also integrated a clustering algorithm (i.e., *k*-means) to classify the CPL rules as high or low confidence rules, which are used for defining the three objectives to guide the search.

To evaluate the *SBRM*$^+$ (*SBRM*$^+_{NSGA\text{-}II}$*-C45*, *SBRM*$^+_{NSGA\text{-}III}$*-C45*, *SBRM*$^+_{NSGA\text{-}II}$*-PART*, and *SBRM*$^+_{NSGA\text{-}III}$*-PART*), we conducted experiments using two real case studies (Cisco and Jitsi) and performed three types of analyses: *difference analysis*, *correlation analysis*, and *trend analysis*. *Difference analysis* shows that *SBRM*$^+$ approaches performed significantly better than two random search-based approaches (*RBRM*$^+$*-C45* and *RBRM*$^+$*-PART*) in terms of the fitness values, six quality indicators, and 17 MLQMs corresponding to both case studies. Among the four *SBRM*$^+$ approaches, *SBRM*$^+_{NSGA\text{-}II}$*-C45* produced the highest quality rules based on MLQMs for the Cisco case study and *SBRM*$^+_{NSGA\text{-}II}$*-PART* for the Jitsi case study. *Correlation analysis* suggests that in most of the cases lower average fitness values and quality indicators (except for *HV*) and higher *HV* mean overall higher quality rules in terms of MLQMs. Furthermore, *trend analysis* shows an increasing trend of the quality of rules in terms of MLQMs for all the four *SBRM*$^+$ approaches across the five iterations.

Our future work includes: (1) Evaluating the performance of different search algorithms for generating configurations and mining the rules; (2) Using different parameter settings for machine learning algorithms and search algorithms; 3) Evaluating the performance of proposed approach using more complex case studies; and (4) Recommending configurations for the selected products based on the mined rules.

ACKNOWLEDGEMENT

This work was supported by the Zen-Configurator project funded by the Research Council of Norway (Grant No. 240024/F20) under the category of Young Research Talents of the FRIPO funding scheme. Tao Yue and Shaukat Ali are also supported by the Co-evolver project funded by the Research Council of Norway (grant no. 286898/LIS) under the category of Researcher Projects of the FRIPO funding scheme.“

REFERENCES

[1] G. Holl, P. Grünbacher, and R. Rabiser, “A systematic review and an expert survey on capabilities supporting multi product lines,” *Information and Software Technology (IST),* vol. 54, no. 8, pp. 828-852, 2012.
[2] M. Rosenmüller, and N. Siegmund, "Automating the Configuration of Multi Software Product Lines." pp. 123-30.
[3] S. Wang, A. Gotlieb, M. Liaaen, and L. C. Briand, “Automatic selection of test execution plans from a video conferencing system product line,” in Proceedings of the VARiability for You Workshop: Variability Modeling Made Useful for Everyone, Innsbruck, Austria, 2012, pp. 32-37.
[4] T. Yue, S. Ali, and B. Selic, "Cyber-Physical System Product Line Engineering: Comprehensive Domain Analysis and Experience Report." pp. 338-347.
[5] G. Bécan, R. Behjati, A. Gotlieb, and M. Acher, "Synthesis of attributed feature models from product descriptions." pp. 1-10.
[6] K. Nie, T. Yue, S. Ali, L. Zhang, and Z. Fan, "Constraints: the core of supporting automated product configuration of cyber-physical systems." pp. 370-387.
[7] E. Bagheri, T. Di Noia, A. Ragone, and D. Gasevic, "Configuring software product line feature models based on stakeholders’ soft and hard requirements." pp. 16-31.
[8] R. Mazo, C. Dumitrescu, C. Salinesi, and D. Diaz, "Recommendation heuristics for improving product line configuration processes," *Recommendation Systems in Software Engineering (RSSE)*, pp. 511-537: Springer, 2014.
[9] H. Lu, T. Yue, S. Ali, and L. Zhang, “Model-based Incremental Conformance Checking to Enable Interactive Product Configuration,” *Information and Software Technology (IST),* vol. 72, pp. 68-89, 2015.
[10] S. Nadi, T. Berger, C. Kästner, and K. Czarnecki, “Where do configuration constraints stem from? an extraction approach and an empirical study,” *IEEE Transactions on Software Engineering (TSE),* vol. 41, no. 8, pp. 820-841, 2015.
[11] P. Temple, J. A. G. Duarte, M. Acher, and J.-M. Jézéquel, "Using Machine Learning to Infer Constraints for Product Lines." pp. 209-218.
[12] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical machine learning tools and techniques*, 4th ed., Switzerland: Morgan Kaufmann, 2016.
[13] S. A. Safdar, L. Hong, Y. Tao, and A. Shaukat, "Mining Cross Product Line Rules with Multi-Objective Search and Machine Learning ". pp. 1319-1326.
[14] E. Frank, and I. H. Witten, "Generating accurate rule sets without global optimization." pp. 144-151.
[15] R. E. Lopez-Herrejon, L. Linsbauer, and A. Egyed, “A systematic mapping study of search-based software engineering for software product lines,” *Information and Software Technology (IST),* vol. 61, pp. 33-51, 2015.
[16] M. Harman, Y. Jia, J. Krinke, W. B. Langdon, J. Petke, and Y. Zhang, "Search based software engineering for software product line engineering: a survey and directions for future work." pp. 5-18.

[17] A. S. Sayyad, J. Ingram, T. Menzies, and H. Ammar, "Scalable product line configuration: A straw to break the camel's back." pp. 465-474.

[18] J. Guo, J. H. Liang, K. Shi, D. Yang, J. Zhang, K. Czarnecki, V. Ganesh, and H. Yu, "SMTIBEA: A hybrid multi-objective optimization algorithm for configuring large constrained software product lines," *Software & Systems Modeling (SoSyM)*, vol. 16, no. 4, pp. 1-20, 2017.

[19] 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.

[20] F. Sarro, A. Petrozziello, and M. Harman, "Multi-objective software effort estimation." pp. 619-630.

[21] D. Pradhan, S. Wang, S. Ali, T. Yue, and M. Liaaen, "STIPI: Using Search to Prioritize Test Cases Based on Multi-objectives Derived from Industrial Practice." pp. 172-190.

[22] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety (RESS)*, vol. 91, no. 9, pp. 992-1007, 9//, 2006.

[23] K. Deb, and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 577-601, 2014.

[24] H. Jain, and K. Deb, "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point Based Nondominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 602-622, 2014.

[25] M. W. Mkaouer, M. Kessentini, S. Bechikh, K. Deb, and M. Ó Cinnéide, "High dimensional search-based software engineering: finding tradeoffs among 15 objectives for automating software refactoring using NSGA-III." pp. 1263-1270.

[26] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*, 3rd ed.: Elsevier, 2012.

[27] J.-M. Davril, E. Delfosse, N. Hariri, M. Acher, J. Cleland-Huang, and P. Heymans, "Feature model extraction from large collections of informal product descriptions." pp. 290-300.

[28] J. R. Quinlan, *C4.5: Programming for machine learning*, 1st ed., London, UK: Morgan Kauffmann, 1993.

[29] W. W. Cohen, "Fast effective rule induction." pp. 115-123.

[30] G. Holmes, M. Hall, and E. Prank, "Generating rule sets from model trees." pp. 1-12.

[31] I. H. Witten, and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed., San Francisco,USA: Diane Cerra, 2005.

[32] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129-137, 1982.

[33] "Euclidean distance," https://wikipedia.org/wiki/Euclidean_distance.

[34] P. McMinn, "Search-based software test data generation: a survey," *Software Testing Verification and Reliability (STVR)*, vol. 14, no. 2, pp. 105-156, 2004.

[35] S. Ali, M. Z. Iqbal, A. Arcuri, and L. C. Briand, "Generating test data from OCL constraints with search techniques," *IEEE Transactions on Software Engineering (TSE)*, vol. 39, no. 10, pp. 1376-1402, 2013.

[36] M. Sokolova, and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management (IPM)*, vol. 45, no. 4, pp. 427-437, 2009.

[37] S. A. Safdar, T. Yue, S. Ali, and H. Lu, "Evaluating Variability Modeling Techniques for Supporting Cyber-Physical System Product Line Engineering." pp. 1-19.

[38] J. Guérin, O. Gibaru, S. Thiery, and E. Nyiri, "Clustering for different scales of measurement-the gap-ratio weighted k-means algorithm," *arXiv preprint arXiv:1703.07625*, 2017.

[39] C. Henard, M. Papadakis, G. Perrouin, J. Klein, and Y. L. Traon, "Multi-objective test generation for software product lines." pp. 62-71.

[40] R. T. Marler, and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization (SMO)*, vol. 26, no. 6, pp. 369-395, 2004.

[41] S. Wang, D. Buchmann, S. Ali, A. Gotlieb, D. Pradhan, and M. Liaaen, "Multi-objective test prioritization in software product line testing: an industrial case study." pp. 32-41.

[42] S. Ali, M. Liaaen, S. Wang, and T. Yue, "Empowering Testing Activities with Modeling-Achievements and Insights from Nine Years of Collaboration with Cisco." pp. 581-589.

[43] "Jitsi "; http://www.jitsi.org/.

[44] D. Pradhan, S. Wang, S. Ali, T. Yue, and M. Liaaen, "CBGA-ES: a cluster-based genetic algorithm with elitist selection for supporting multi-objective test optimization." pp. 367-378.

[45] C. Henard, M. Papadakis, M. Harman, and Y. Le Traon, "Combining multi-objective search and constraint solving for configuring large software product lines." pp. 517-528.

[46] A. S. Sayyad, T. Menzies, and H. Ammar, "On the Value of User Preferences in Search-Based Software Engineering: A Case Study in Software Product Lines." pp. 492-501.

[47] J. M. Chaves-González, and M. A. Pérez-Toledano, “Differential evolution with Pareto tournament for the multi-objective next release problem,” *Applied Mathematics and Computation*, vol. 252, pp. 1-13, 2015.
[48] J. K. Chhabra, "An empirical study of the sensitivity of quality indicator for software module clustering." pp. 206-211.
[49] W. K. G. Assunção, T. E. Colanzi, S. R. Vergilio, and A. Pozo, “A multi-objective optimization approach for the integration and test order problem,” *Information Sciences*, vol. 267, pp. 119-139, 2014.
[50] G. Guizzo, T. E. Colanzi, and S. R. Vergilio, "A pattern-driven mutation operator for search-based product line architecture design." pp. 77-91.
[51] V. Hrubá, B. Křena, Z. Letko, H. Pluháčková, and T. Vojnar, "Multi-objective genetic optimization for noise-based testing of concurrent software." pp. 107-122.
[52] M. R. Karim, and G. Ruhe, "Bi-objective genetic search for release planning in support of themes." pp. 123-137.
[53] L. Li, M. Harman, E. Letier, and Y. Zhang, "Robust next release problem: handling uncertainty during optimization." pp. 1247-1254.
[54] R. E. Lopez-Herrejon, J. Ferrer, F. Chicano, A. Egyed, and E. Alba, "Comparative analysis of classical multi-objective evolutionary algorithms and seeding strategies for pairwise testing of software product lines." pp. 387-396.
[55] F. Luna, D. L. González-Álvarez, F. Chicano, and M. A. Vega-Rodríguez, “The software project scheduling problem: A scalability analysis of multi-objective metaheuristics,” *Applied Soft Computing*, vol. 15, pp. 136-148, 2014.
[56] M. W. Mkaouer, M. Kessentini, S. Bechikh, and M. Ó. Cinnéide, "A robust multi-objective approach for software refactoring under uncertainty." pp. 168-183.
[57] S. Nejati, and L. C. Briand, "Identifying optimal trade-offs between cpu time usage and temporal constraints using search." pp. 351-361.
[58] A. Ramírez, J. R. Romero, and S. Ventura, "On the performance of multiple objective evolutionary algorithms for software architecture discovery." pp. 1287-1294.
[59] L. S. de Souza, R. B. Prudêncio, and F. d. A. Barros, "A comparison study of binary multi-objective particle swarm optimization approaches for test case selection." pp. 2164-2171.
[60] A. Sureka, "Requirements prioritization and next-release problem under Non-additive value conditions." pp. 120-123.
[61] W. K. G. Assunçao, T. E. Colanzi, S. R. Vergilio, and A. Pozo, "On the Application of the Multi-Evolutionary and Coupling-Based Approach with Different Aspect-Class Integration Testing Strategies." pp. 19-33.
[62] M. Bozkurt, "Cost-aware pareto optimal test suite minimisation for service-centric systems." pp. 1429-1436.
[63] L. Briand, Y. Labiche, and K. Chen, "A multi-objective genetic algorithm to rank state-based test cases." pp. 66-80.
[64] M. W. Mkaouer, M. Kessentini, S. Bechikh, and D. R. Tauritz, "Preference-based multi-objective software modelling." pp. 61-66.
[65] A. Ouni, M. Kessentini, H. Sahraoui, and M. S. Hamdi, "The use of development history in software refactoring using a multi-objective evolutionary algorithm." pp. 1461-1468.
[66] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham, “AbYSS: Adapting scatter search to multiobjective optimization,” *Evolutionary Computation, IEEE Transactions on*, vol. 12, no. 4, pp. 439-457, 2008.
[67] D. A. Van, V. Gary, and B. Lamont, “Multiobjective Evolutionary Algorithm Research: A History and Analysis,” *Evolutionary Computation* vol. 8, no. 2, 1999.
[68] C. M. Fonseca, and P. J. Fleming, “Multiobjective optimization and multiple constraint handling with evolutionary algorithms Part II: Application example,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 28 no. 1, pp. 38–47, 1998.
[69] M. Tanaka, H. Watanabe, Y. Furukawa, and T. Tanino, "GA-based decision support system for multicriteria optimization." pp. 1556-1561.
[70] J. L. Cochrane, and M. Zeleny, *Multiple Criteria Decision Making*: University of South Carolina Press, 1973.
[71] 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.
[72] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham, “AbYSS: Adapting scatter search to multiobjective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 4, pp. 439-457, 2008.
[73] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang, "Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion." pp. 892-899.
[74] A. Arcuri, and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering." pp. 1-10.
[75] S. Ali, and K. A. Smith, “On learning algorithm selection for classification,” *Applied Soft Computing*, vol. 6, no. 2, pp. 119-138, 2006.
[76] J. Wu, S. Ali, T. Yue, J. Tian, and C. Liu, “Assessing the quality of industrial avionics software: an extensive empirical evaluation,” *Empirical Software Engineering (EMSE)*, vol. 22, no. 4, pp. 1-50, 2016.

[77] H. B. Mann, and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics,* vol. 18, no. 1, pp. 50-60, 1947.

[78] 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 (JEBS),* vol. 25, no. 2, pp. 101-132, 2000.

[79] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 3rd ed., London,UK: Chapman and Hall, CRC Press, 2007.

[80] J. B. Kollat, and P. M. Reed, "Comparing state-of-the-art evolutionary multi-objective algorithms for long-term groundwater monitoring design," *Advances in Water Resources,* vol. 29, no. 6, pp. 792-807, 2006.

[81] D. Pradhan, S. Wang, S. Ali, T. Yue, and M. Liaaen, "CBGA-ES+: A Cluster-Based Genetic Algorithm with Non-Dominated Elitist Selection for Supporting Multi-Objective Test Optimization," *IEEE Transactions on Software Engineering*, 2018.

[82] A. E. Eiben, and S. K. Smit, "Parameter tuning for configuring and analyzing evolutionary algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 19-31, 2011.

[83] A. Baars, M. Harman, Y. Hassoun, K. Lakhotia, P. McMinn, P. Tonella, and T. Vos, "Symbolic search-based testing." pp. 53-62.

[84] J. N. Alarcón-Jaén, "Multi-objective approach for the minimization of test cases in Software Production Lines," University of Malaga, Spain. , 2018.

[85] P. Runeson, M. Host, A. Rainer, and B. Regnell, *Case study research in software engineering: Guidelines and examples*, 1st ed., New Jersey, USA: John Wiley & Sons, 2012.

[86] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, and P. S. Yu, "Top 10 algorithms in data mining," *Knowledge and Information Systems (KAIS),* vol. 14, no. 1, pp. 1-37, 2008.

[87] A. Arcuri, and G. Fraser, "On parameter tuning in search based software engineering." pp. 33-47.

[88] S. Wang, S. Ali, and A. Gotlieb, "Cost-effective test suite minimization in product lines using search techniques," *Journal of Systems and Software (JSS),* vol. 103, pp. 370-391, 2014.

[89] 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, Denver, Colorado, USA, 2016, pp. 1085-1092.

[90] C. Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, and A. Wesslen, *Experimentation in software engineering: an introduction*, 1st ed., Berlin, Germany: Kluwer Academic Publishers, 2000.

[91] 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." pp. 631-642.

[92] L. Yi, W. Zhang, H. Zhao, Z. Jin, and H. Mei, "Mining binary constraints in the construction of feature models." pp. 141-150.

[93] B. Zhang, and M. Becker, "Mining complex feature correlations from software product line configurations." pp. 19-25.

[94] S. Nadi, T. Berger, C. Kästner, and K. Czarnecki, "Mining configuration constraints: Static analyses and empirical results." pp. 140-151.

[95] K. Czarnecki, S. She, and A. Wasowski, "Sample spaces and feature models: There and back again." pp. 22-31.

[96] L. Zhao, M. J. Zaki, and N. Ramakrishnan, "BLOSOM: A framework for mining arbitrary boolean expressions." pp. 827-832.

[97] H. Dumitru, M. Gibiec, N. Hariri, J. Cleland-Huang, B. Mobasher, C. Castro-Herrera, and M. Mirakhorli, "On-demand feature recommendations derived from mining public product descriptions." pp. 181-190.

[98] N. Hariri, C. Castro-Herrera, M. Mirakhorli, J. Cleland-Huang, and B. Mobasher, "Supporting domain analysis through mining and recommending features from online product listings," *IEEE Transactions on Software Engineering (TSE),* vol. 39, no. 12, pp. 1736-1752, 2013.

[99] Y. Yu, H. Wang, G. Yin, and B. Liu, "Mining and recommending software features across multiple web repositories." pp. 1-9.

[100] N. H. Bakar, Z. M. Kasirun, and N. Salleh, "Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review," *Journal of Systems and Software (JSS),* vol. 106, pp. 132-149, 2015.

Appendix A: Examples of Generated Rules Using SBRM$^+$

Table 13: Examples of CPL rules from Cisco and Jitsi case studies

Case study	Rule example
Rule formate	Product.ConfigurableParameter = ConfigurableParameterValue **AND** … **AND** Product.ConfigurableParameter = ConfigurableParameterValue : SystemState (Support/Violation)
Cisco	VCS1.IP-Protocol = Sip **AND** VCS2.Listen-Port = Off **AND** VCS2.IP-Protocol = Sip **AND** Default-Transport = Tls : FailedFailed (34/5)
	VCS2.Max-Transmit-Callrate <= 5982 **AND** VCS1. IP-Protocol = Auto **AND** VCS1.Encryption = Off **AND** VCS2.Encryption = BestEffort **AND** VCS3.Encryption = BestEffort **AND** VCS3.Max-Transmit-Callrate > 135 : ConnectedConnected (103/1)
Jitsi	VCS1.IP-Pprotocol = AIM **AND** VCS3.Video-Codec = rtx **AND** VCS3.Audio-Codec = AMR-WB-16000 **AND** VCS2.Audio-Codec = SILK-12000 **AND** VCS1.Video-Codec = VP8 **AND** VCS1.Encryption = On **AND** VCS2.Encryption = BestEffort **AND** VCS1.Default-Callrate <= 5744 **AND** VCS2.Max-Receive-Callrate > 1680 **AND** VCS3.Max-Transmit-Callrate > 3005 : ConnectedFailed (30/1)
	VCS2.Video-Codec = VP8 **AND** VCS3.Video-Codec = h264 **AND** VCS2.MTU > 702 **AND** VCS1.MTU > 760 **AND** VCS1.Audio-Codec = SILK-16000 **AND** VCS1.SIP-Listen-Port = Off **AND** VCS1.Encryption = BestEffort **AND** VCS1.Video-Codec = VP8 **AND** VCS2.MTU > 806 : FailedConnected (32/9)

Appendix B: Descriptive Statistics and Detailed Results

In this section, we present the detailed results of our research questions corresponding to both of the case studies (i.e., Cisco and Jitsi).

Detailed Results of RQ1

Table 14. Comparing SBRM$^+_{NSGA-II}$-C45 and SBRM$^+_{NSGA-III}$-C45 with RBRM$^+$-C45 and SBRM$^+_{NSGA-II}$-PART and SBRM$^+_{NSGA-III}$-PART with RBRM$^+$-PART in terms of fitness values for both Cisco and Jitsi case studies*

MLQMs	FV-O1		FV-O2		FV-O3		OFV	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Comparing SBRM$^+_{NSGA-II}$-C45 with RBRM$^+$-C45 for the Cisco case study								
Iteration-1	**<0.05**	0.07	**<0.05**	0.30	**<0.05**	0.30	**<0.05**	0.04
Iteration-2	**<0.05**	0.31	**<0.05**	0.13	**<0.05**	0.08	**<0.05**	0.18
Iteration-3	**<0.05**	0.35	**<0.05**	0.10	**<0.05**	0.18	**<0.05**	0.27
Iteration-4	**<0.05**	0.31	**<0.05**	0.13	**<0.05**	0.12	**<0.05**	0.11
Iteration-5	**<0.05**	0.34	**<0.05**	0.33	**<0.05**	0.03	**<0.05**	0.08
Overall	**<0.05**	0.33	**<0.05**	0.19	**<0.05**	0.16	**<0.05**	0.16
Comparing SBRM$^+_{NSGA-III}$-C45 with RBRM$^+$-C45 for the Cisco case study								
Iteration-1	**<0.05**	0.19	**<0.05**	0.39	**<0.05**	0	**<0.05**	0
Iteration-2	**<0.05**	0.33	**<0.05**	0.32	**<0.05**	0	**<0.05**	0.05
Iteration-3	**<0.05**	0.22	**<0.05**	0.21	**<0.05**	0	**<0.05**	0.02
Iteration-4	**<0.05**	0.15	**<0.05**	0.20	**<0.05**	0	**<0.05**	0
Iteration-5	**<0.05**	0.34	**<0.05**	0.38	**<0.05**	0	**<0.05**	0
Overall	**<0.05**	0.28	**<0.05**	0.30	**<0.05**	0	**<0.05**	0.02
Comparing SBRM$^+_{NSGA-II}$-PART with RBRM$^+$-PART for the Cisco case study								
Iteration-1	**<0.05**	0.16	**<0.05**	0.15	**<0.05**	0.14	**<0.05**	0.01
Iteration-2	**<0.05**	0.15	**<0.05**	0.36	**<0.05**	0.23	**<0.05**	0.07
Iteration-3	**<0.05**	0.01	**<0.05**	0.18	**<0.05**	0.08	**<0.05**	0
Iteration-4	**<0.05**	0.32	**<0.05**	0.20	**<0.05**	0.15	**<0.05**	0.01
Iteration-5	**<0.05**	0.36	**<0.05**	0.09	**<0.05**	0.15	**<0.05**	0
Overall	**<0.05**	0.23	**<0.05**	0.19	**<0.05**	0.27	**<0.05**	0.17
Comparing SBRM$^+_{NSGA-III}$-PART with RBRM$^+$-PART for the Cisco case study								
Iteration-1	**<0.05**	0.07	**<0.05**	0.26	**<0.05**	0.03	**<0.05**	0.01
Iteration-2	**<0.05**	0.04	**<0.05**	0.35	**<0.05**	0	**<0.05**	0
Iteration-3	**<0.05**	0.11	**<0.05**	0.12	**<0.05**	0	**<0.05**	0
Iteration-4	**<0.05**	0.42	**<0.05**	0.08	**<0.05**	0	**<0.05**	0
Iteration-5	**<0.05**	0.41	**<0.05**	0.08	**<0.05**	0	**<0.05**	0
Overall	**<0.05**	0.24	**<0.05**	0.17	**<0.05**	0.02	**<0.05**	0

Comparing SBRM$^+_{NSGA-II}$-C45 with RBRM$^+$-C45 for the Jitsi case study								
Iteration-1	**<0.05**	0	**<0.05**	0.39	**<0.05**	0.33	**<0.05**	0
Iteration-2	**<0.05**	0.22	**<0.05**	0.02	**<0.05**	0	**<0.05**	0.01
Iteration-3	**<0.05**	0.18	**<0.05**	0	**<0.05**	0	**<0.05**	0
Iteration-4	**<0.05**	0.31	**<0.05**	0.01	**<0.05**	0	**<0.05**	0.01
Iteration-5	**<0.05**	0.15	**<0.05**	0.01	**<0.05**	0	**<0.05**	0
Overall	**<0.05**	0.17	**<0.05**	0.08	**<0.05**	0.08	**<0.05**	0.05
Comparing SBRM$^+_{NSGA-III}$-C45 with RBRM$^+$-C45 for the Jitsi case study								
Iteration-1	**<0.05**	0.06	**<0.05**	0.05	**<0.05**	0	**<0.05**	0
Iteration-2	**<0.05**	0.45	**<0.05**	0.01	**<0.05**	0	**<0.05**	0
Iteration-3	**<0.05**	0.43	**<0.05**	0.01	**<0.05**	0	**<0.05**	0
Iteration-4	**<0.05**	0.27	**<0.05**	0	**<0.05**	0	**<0.05**	0
Iteration-5	**<0.05**	0.41	**<0.05**	0.01	**<0.05**	0	**<0.05**	0
Overall	**<0.05**	0.34	**<0.05**	0.02	**<0.05**	0.01	**<0.05**	0.02
Comparing SBRM$^+_{NSGA-II}$-PART with RBRM$^+$-PART for the Jitsi case study								
Iteration-1	**<0.05**	0.16	**<0.05**	0.45	**<0.05**	0.47	**<0.05**	0
Iteration-2	**<0.05**	0.44	**<0.05**	0.01	**<0.05**	0	**<0.05**	0
Iteration-3	**<0.05**	0.36	**<0.05**	0	**<0.05**	0	**<0.05**	0
Iteration-4	**<0.05**	0.35	**<0.05**	0.02	**<0.05**	0.01	**<0.05**	0
Iteration-5	**<0.05**	0.31	**<0.05**	0.01	**<0.05**	0	**<0.05**	0
Overall	**<0.05**	0.38	**<0.05**	0.15	**<0.05**	0.17	**<0.05**	0.33
Comparing SBRM$^+_{NSGA-III}$-PART with RBRM$^+$-PART for the Jitsi case study								
Iteration-1	**<0.05**	0.41	**<0.05**	0.04	**<0.05**	0	**<0.05**	0
Iteration-2	**<0.05**	0.33	**<0.05**	0.08	**<0.05**	0.19	**<0.05**	0.12
Iteration-3	**<0.05**	0.08	**<0.05**	0.11	**<0.05**	0	**<0.05**	0
Iteration-4	**<0.05**	0.28	**<0.05**	0.09	**<0.05**	0.01	**<0.05**	0.02
Iteration-5	**<0.05**	0.24	**<0.05**	0.02	**<0.05**	0	**<0.05**	0.08
Overall	**<0.05**	0.30	**<0.05**	0.09	**<0.05**	0.07	**<0.05**	0.05

Table 15. Comparing SBRM+NSGA-II-C45 and SBRM+NSGA-III-C45 with RBRM+-C45 and SBRM+NSGA-II-PART and SBRM+NSGA-III-PART with RBRM+-PART in terms of indicators for both Cisco and Jitsi case studies*

MLQMs	**HV**		**IGD**		**ϵ**		**ED**		**GD**		**GS**	
	p-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$
Comparing SBRM$^+_{NSGA-II}$-C45 with RBRM$^+$-C45 for the Cisco case study												
Iteration-1	**<0.05**	1	**<0.05**	0	0.17	0.40	**<0.05**	0	**<0.05**	0	**<0.05**	1
Iteration-2	**<0.05**	0.89	**<0.05**	0.07	**<0.05**	0.07	**<0.05**	0.09	**<0.05**	0.14	**<0.05**	0.92
Iteration-3	**<0.05**	0.77	**<0.05**	0.16	**<0.05**	0.17	**<0.05**	0.09	**<0.05**	0.10	**<0.05**	0.84
Iteration-4	**<0.05**	0.78	**<0.05**	0.13	**<0.05**	0.13	0.44	0.39	0.39	0.38	**<0.05**	0.82
Iteration-5	**<0.05**	0.89	**<0.05**	0.01	**<0.05**	0.03	0.31	0.36	0.25	0.34	**<0.05**	0.97
Overall	**<0.05**	0.82	**<0.05**	0.11	**<0.05**	0.16	**<0.05**	0.26	**<0.05**	0.25	**<0.05**	0.91
Comparing SBRM$^+_{NSGA-III}$-C45 with RBRM$^+$-C45 for the Cisco case study												
Iteration-1	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
Iteration-2	**<0.05**	0.95	**<0.05**	0	**<0.05**	0	**<0.05**	0.04	**<0.05**	0	0.05	0.24
Iteration-3	**<0.05**	0.98	**<0.05**	0.02	**<0.05**	0	**<0.05**	0.08	**<0.05**	0	**<0.05**	0.17
Iteration-4	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.03	**<0.05**	0.01	**<0.05**	0.16
Iteration-5	**<0.05**	0.85	**<0.05**	0	**<0.05**	0	0.12	0.29	0.09	0.27	0.91	0.48
Overall	**<0.05**	0.95	**<0.05**	0.02	**<0.05**	0	**<0.05**	0.15	**<0.05**	0.09	**<0.05**	0.25
Comparing SBRM$^+_{NSGA-II}$-PART with RBRM$^+$-PART for the Cisco case study												
Iteration-1	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	1
Iteration-2	**<0.05**	0.90	**<0.05**	0.16	**<0.05**	0.22	0.16	0.31	**<0.05**	0.08	**<0.05**	0.86
Iteration-3	**<0.05**	1	**<0.05**	0.01	**<0.05**	0.06	**<0.05**	0.22	**<0.05**	0	**<0.05**	0.88
Iteration-4	**<0.05**	0.85	**<0.05**	0.10	**<0.05**	0.15	**<0.05**	0.16	**<0.05**	0	**<0.05**	0.78
Iteration-5	0.14	0.70	**<0.05**	0.13	**<0.05**	0.16	**<0.05**	0.22	**<0.05**	0.12	**<0.05**	0.81
Overall	**<0.05**	0.76	**<0.05**	0.22	**<0.05**	0.26	**<0.05**	0.27	**<0.05**	0.11	**<0.05**	0.84
Comparing SBRM$^+_{NSGA-III}$-PART with RBRM$^+$-PART for the Cisco case study												
Iteration-1	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	1
Iteration-2	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.17	**<0.05**	0.06	**<0.05**	0.79
Iteration-3	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.08	**<0.05**	0	0.74	0.55
Iteration-4	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.07	**<0.05**	0.01	0.62	0.43

Iteration-5	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.09	**<0.05**	0.03	0.97	0.49
Overall	**<0.05**	0.99	**<0.05**	0	**<0.05**	0.02	**<0.05**	0.08	**<0.05**	0.03	**<0.05**	0.73
Comparing SBRM$^+_{NSGA-II}$-C45 with RBRM$^+$-C45 for the Jitsi case study												
Iteration-1	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.85
Iteration-2	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.04	**<0.05**	0.14	**<0.05**	0.12
Iteration-3	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.12	**<0.05**	0.06	**<0.05**	0.03
Iteration-4	**<0.05**	0.97	**<0.05**	0	**<0.05**	0.06	0.19	0.33	0.05	0.24	**<0.05**	0.12
Iteration-5	**<0.05**	1	**<0.05**	0	**<0.05**	0	0.07	0.26	**<0.05**	0.05	**<0.05**	0.10
Overall	**<0.05**	0.98	**<0.05**	0.03	**<0.05**	0.03	**<0.05**	0.14	**<0.05**	0.07	**<0.05**	0.24
Comparing SBRM$^+_{NSGA-III}$-C45 with RBRM$^+$-C45 for the Jitsi case study												
Iteration-1	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	1
Iteration-2	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01	**<0.05**	1
Iteration-3	**<0.05**	1	**<0.05**	0	**<0.05**	0.12	**<0.05**	0.21	**<0.05**	0.07	**<0.05**	1
Iteration-4	**<0.05**	1	**<0.05**	0	**<0.05**	0	0.55	0.59	**<0.05**	0.20	**<0.05**	0.89
Iteration-5	**<0.05**	1	**<0.05**	0	**<0.05**	0.09	**<0.05**	0.22	**<0.05**	0.07	**<0.05**	1
Overall	**<0.05**	0.97	**<0.05**	0	**<0.05**	0.03	**<0.05**	0.17	**<0.05**	0.08	**<0.05**	0.98
Comparing SBRM$^+_{NSGA-II}$-PART with RBRM$^+$-PART for the Jitsi case study												
Iteration-1	**<0.05**	1	**<0.05**	0.10	**<0.05**	0.10	**<0.05**	0.10	**<0.05**	0	**<0.05**	1
Iteration-2	**<0.05**	0.83	**<0.05**	0	0.17	0.31	**<0.05**	0.09	**<0.05**	0.10	0.12	0.71
Iteration-3	**<0.05**	0.98	**<0.05**	0	0.08	0.26	**<0.05**	0.02	**<0.05**	0	**<0.05**	0.93
Iteration-4	**<0.05**	0.83	**<0.05**	0	0.06	0.25	**<0.05**	0.10	**<0.05**	0.09	**<0.05**	0.77
Iteration-5	**<0.05**	0.92	**<0.05**	0	**<0.05**	0.20	**<0.05**	0	**<0.05**	0	0.28	0.65
Overall	**<0.05**	0.89	**<0.05**	0.15	**<0.05**	0.26	**<0.05**	0.08	**<0.05**	0.07	**<0.05**	0.74
Comparing SBRM$^+_{NSGA-III}$-PART with RBRM$^+$-PART for the Jitsi case study												
Iteration-1	**<0.05**	1	**<0.05**	0	**<0.05**	0	**<0.05**	0.19	**<0.05**	0.05	**<0.05**	0.95
Iteration-2	**<0.05**	0.79	**<0.05**	0.10	0.44	0.39	**<0.05**	0.18	**<0.05**	0.13	**<0.05**	0.93
Iteration-3	**<0.05**	1	**<0.05**	0.01	**<0.05**	0.15	**<0.05**	0.13	**<0.05**	0.10	**<0.05**	1
Iteration-4	**<0.05**	0.92	**<0.05**	0.02	0.22	0.33	**<0.05**	0.11	**<0.05**	0.09	**<0.05**	0.95
Iteration-5	**<0.05**	0.90	**<0.05**	0	0.12	0.29	**<0.05**	0	**<0.05**	0	**<0.05**	0.91
Overall	**<0.05**	0.92	**<0.05**	0.03	**<0.05**	0.29	**<0.05**	0.11	**<0.05**	0.08	**<0.05**	0.95

Detailed Results of RQ2

Table 16: Comparing SBRM$^+_{NSGA-II}$-C45 with RBRM$^+$-C45 in terms of MLQMs for the Cisco case study*

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	0.104	0.72	**<0.05**	0.94	**<0.05**	0.99	**<0.05**	0.99	**<0.05**	1	**<0.05**	0.96
MAE	**<0.05**	0.06	**<0.05**	0.05	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01
RMSE	**<0.05**	0.20	**<0.05**	0.04	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.03
RAE	**<0.05**	0.01	**<0.05**	0.03	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01
RRSE	**<0.05**	0.08	**<0.05**	0.04	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.02
FF-Precision	0.405	0.62	**<0.05**	0.89	**<0.05**	0.98	**<0.05**	0.99	**<0.05**	0.99	**<0.05**	0.92
FF-Recall	0.256	0.66	0.226	0.67	**<0.05**	0.85	**<0.05**	0.79	0.063	0.75	**<0.05**	0.75
FF-FMeasure	0.307	0.64	**<0.05**	0.78	**<0.05**	0.98	**<0.05**	0.95	**<0.05**	0.91	**<0.05**	0.88
CC-Precision	**<0.05**	0.89	**<0.05**	0.91	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.96
CC-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99
CC-FMeasure	**<0.05**	0.97	**<0.05**	0.98	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99
FC-Precision	-	0.50	-	0.50	-	0.50	-	0.50	0.132	0.30	0.060	0.44
FC-Recall	-	0.50	-	0.50	-	0.50	-	0.50	0.132	0.30	0.060	0.44
FC-FMeasure	-	0.50	-	0.50	-	0.50	-	0.50	0.132	0.30	0.060	0.44
CF-Precision	**<0.05**	0.80	**<0.05**	0.80	**<0.05**	0.95	**<0.05**	0.92	**<0.05**	0.96	**<0.05**	0.88
CF-Recall	**<0.05**	0.98	**<0.05**	1	**<0.05**	0.95	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
CF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1

* "-" represents that a value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero for both of the approaches and therefore p-value cannot be calculated.

Table 17: Comparing SBRM$^+_{NSGA-III}$-C45 with RBRM$^+$-C45 in terms of MLQMs for the Cisco case study*

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$

Accuracy	0.21	0.67	**<0.05**	0.88	**<0.05**	0.96	**<0.05**	0.97	**<0.05**	1	**<0.05**	0.93
MAE	0.31	0.36	**<0.05**	0.07	**<0.05**	0	**<0.05**	0.01	**<0.05**	0	**<0.05**	0.06
RMSE	0.21	0.33	**<0.05**	0.13	**<0.05**	0.04	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.07
RAE	0.34	0.37	**<0.05**	0.12	**<0.05**	0.01	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.07
RRSE	0.25	0.34	0.06	0.25	**<0.05**	0.04	**<0.05**	0.04	**<0.05**	0	**<0.05**	0.10
FF-Precision	0.10	0.72	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.97
FF-Recall	0.17	0.69	0.16	0.69	**<0.05**	0.89	**<0.05**	0.86	0.05	0.76	**<0.05**	0.77
FF-FMeasure	0.27	0.65	**<0.05**	0.95	**<0.05**	0.98	**<0.05**	0.97	**<0.05**	0.95	**<0.05**	0.93
CC-Precision	0.43	0.61	0.97	0.49	0.16	0.69	0.14	0.70	**<0.05**	1	**<0.05**	0.71
CC-Recall	0.43	0.39	**<0.05**	0.20	0.71	0.45	0.82	0.47	**<0.05**	0.78	0.78	0.48
CC-FMeasure	0.60	0.58	0.23	0.34	0.33	0.64	0.41	0.62	**<0.05**	0.98	**<0.05**	0.63
FC-Precision	-	0.50	-	0.50	-	0.50	-	0.5	**<0.05**	0	**<0.05**	0.40
FC-Recall	-	0.50	-	0.50	-	0.50	-	0.50	**<0.05**	0	**<0.05**	0.40
FC-FMeasure	-	0.50	-	0.50	-	0.50	-	0.50	**<0.05**	0	**<0.05**	0.40
CF-Precision	0.22	0.67	0.50	0.60	**<0.05**	0.90	**<0.05**	0.90	**<0.05**	0.96	**<0.05**	0.82
CF-Recall	0.65	0.57	0.71	0.56	**<0.05**	0.81	**<0.05**	0.94	**<0.05**	1	**<0.05**	0.80
CF-FMeasure	0.23	0.67	0.34	0.63	**<0.05**	0.97	**<0.05**	1	**<0.05**	1	**<0.05**	0.88

Table 18: Comparing SBRM$^{+}_{NSGA-II}$-PART with RBRM^{+}-PART in terms of MLQMs for the Cisco case study*

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	0.93	**<0.05**	0.99	**<0.05**	0.99	**<0.05**	1	**<0.05**	0.99
MAE	**<0.05**	0	**<0.05**	0.06	**<0.05**	0.01	**<0.05**	0.01	**<0.05**	0	**<0.05**	0.01
RMSE	**<0.05**	0	**<0.05**	0.04	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01
RAE	**<0.05**	0	**<0.05**	0.02	**<0.05**	0.01	**<0.05**	0.01	**<0.05**	0	**<0.05**	0.01
RRSE	**<0.05**	0	**<0.05**	0.02	**<0.05**	0	**<0.05**	0.01	**<0.05**	0	**<0.05**	0.01
FF-Precision	**<0.05**	1	**<0.05**	0.92	**<0.05**	1	**<0.05**	0.99	**<0.05**	1	**<0.05**	0.99
FF-Recall	**<0.05**	1	**<0.05**	0.94	**<0.05**	0.97	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
FF-FMeasure	**<0.05**	1	**<0.05**	0.93	**<0.05**	0.99	**<0.05**	1	**<0.05**	1	**<0.05**	0.99
CC-Precision	0.427	0.61	**<0.05**	0.93	**<0.05**	0.80	**<0.05**	0.89	**<0.05**	0.96	**<0.05**	0.84
CC-Recall	**<0.05**	0.83	**<0.05**	0.86	**<0.05**	0.77	**<0.05**	0.93	**<0.05**	0.96	**<0.05**	0.86
CC-FMeasure	0.064	0.75	**<0.05**	0.90	**<0.05**	0.80	**<0.05**	0.93	**<0.05**	0.96	**<0.05**	0.85
FC-Precision	-	0.50	-	0.50	-	0.50	0.168	0.40	0.078	0.35	**<0.05**	0.45
FC-Recall	-	0.50	-	0.50	-	0.50	0.168	0.40	0.078	0.35	**<0.05**	0.45
FC-FMeasure	-	0.50	-	0.50	-	0.50	0.168	0.40	0.078	0.35	**<0.05**	0.45
CF-Precision	0.472	0.60	0.821	0.47	0.241	0.66	**<0.05**	0.80	**<0.05**	0.87	**<0.05**	0.65
CF-Recall	1	0.51	0.384	0.38	0.529	0.59	0.173	0.69	**<0.05**	0.84	0.213	0.57
CF-FMeasure	0.705	0.56	0.545	0.42	0.353	0.63	**<0.05**	0.77	**<0.05**	0.88	0.054	0.61

* "-" represents that a value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero for both of the approaches and therefore p-value cannot be calculated

Table 19: Comparing SBRM$^{+}_{NSGA-III}$-PART with RBRM^{+}-PART in terms of MLQMs for the Cisco case study

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	0.91	**<0.05**	0.95	**<0.05**	0.96	**<0.05**	1	**<0.05**	0.97
MAE	**<0.05**	0	**<0.05**	0.08	**<0.05**	0.04	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.02
RMSE	**<0.05**	0	**<0.05**	0.03	**<0.05**	0.01	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.01
RAE	**<0.05**	0.03	**<0.05**	0.10	**<0.05**	0.10	**<0.05**	0.04	**<0.05**	0	**<0.05**	0.06
RRSE	**<0.05**	0.05	**<0.05**	0.06	**<0.05**	0.04	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.04
FF-Precision	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.99	**<0.05**	0.99	**<0.05**	1	**<0.05**	1
FF-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98	**<0.05**	0.99	**<0.05**	0.99
FF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98	**<0.05**	1	**<0.05**	0.99
CC-Precision	0.76	0.46	0.47	0.40	0.29	0.36	0.60	0.43	0.68	0.44	0.14	0.42
CC-Recall	0.57	0.42	0.38	0.38	0.12	0.29	0.79	0.54	0.71	0.56	0.34	0.44
CC-FMeasure	0.65	0.44	0.38	0.38	0.16	0.31	0.68	0.44	0.97	0.49	0.19	0.42
FC-Precision	-	0.50	-	0.50	-	0.50	0.17	0.40	0.08	0.35	**<0.05**	0.45
FC-Recall	-	0.50	-	0.50	-	0.50	0.17	0.40	0.08	0.35	**<0.05**	0.45
FC-FMeasure	-	0.50	-	0.50	-	0.50	0.17	0.40	0.08	0.35	**<0.05**	0.45
CF-Precision	0.22	0.67	0.71	0.45	0.68	0.56	**<0.05**	0.82	**<0.05**	0.92	**<0.05**	0.65

CF-Recall	0.68	0.56	0.38	0.38	0.76	0.55	**<0.05**	0.79	**<0.05**	0.92	**<0.05**	0.62
CF-FMeasure	0.36	0.63	0.65	0.44	0.94	0.52	**<0.05**	0.80	**<0.05**	0.92	**<0.05**	0.63

Table 20: Comparing SBRM$^+_{\text{NSGA-II}}$-C45 with RBRM$^+$-C45 in terms of MLQMs for the Jitsi case study

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
MAE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01
RMSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01
RAE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
RRSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
FF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
FF-Recall	**<0.05**	0.99	**<0.05**	0.86	**<0.05**	0.92	**<0.05**	0.86	**<0.05**	0.84	**<0.05**	0.80
FF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.97	**<0.05**	0.94
CC-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CC-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CC-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
FC-Precision	0.384	0.62	0.570	0.42	0.226	0.67	0.289	0.65	0.240	0.66	0.072	0.60
FC-Recall	0.162	0.69	1	0.50	0.095	0.73	0.199	0.68	0.289	0.65	**<0.05**	0.63
FC-FMeasure	0.211	0.67	0.970	0.49	0.120	0.71	0.226	0.67	0.325	0.64	**<0.05**	0.63
CF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CF-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1

Table 21: Comparing SBRM$^+_{\text{NSGA-III}}$-C45 with RBRM$^+$-C45 in terms of MLQMs for the Jitsi case study

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	0.95	0.68	0.56	0.79	0.46	0.73	0.55	0.63	0.57	0.43	0.55
MAE	**<0.05**	0.80	**<0.05**	0.83	0.11	0.72	0.63	0.57	0.39	0.38	**<0.05**	0.62
RMSE	**<0.05**	0.02	**<0.05**	0.19	0.22	0.33	0.14	0.30	0.15	0.31	**<0.05**	0.29
RAE	**<0.05**	1	**<0.05**	0.89	0.17	0.69	0.74	0.45	**<0.05**	0.19	**<0.05**	0.63
RRSE	0.53	0.59	**<0.05**	0.21	**<0.05**	0.16	**<0.05**	0.11	**<0.05**	0.09	**<0.05**	0.20
FF-Precision	**<0.05**	0.93	0.76	0.46	0.94	0.52	0.65	0.57	0.19	0.68	0.54	0.54
FF-Recall	**<0.05**	1	**<0.05**	0.81	0.94	0.52	0.85	0.47	0.82	0.47	**<0.05**	0.64
FF-FMeasure	**<0.05**	1	0.34	0.63	0.91	0.48	0.71	0.56	0.57	0.58	0.13	0.59
CC-Precision	0.08	0.26	**<0.05**	0.77	0.52	0.59	0.26	0.66	0.25	0.66	0.10	0.60
CC-Recall	**<0.05**	0.10	0.91	0.52	0.73	0.45	0.34	0.63	0.23	0.67	0.81	0.49
CC-FMeasure	**<0.05**	0.13	0.43	0.61	1	0.50	0.27	0.65	0.25	0.66	0.70	0.52
FC-Precision	0.06	0.25	0.43	0.61	0.20	0.68	**<0.05**	0.76	**<0.05**	0.79	**<0.05**	0.63
FC-Recall	**<0.05**	0.14	0.82	0.47	0.38	0.62	0.35	0.63	0.16	0.69	0.99	0.50
FC-FMeasure	**<0.05**	0.15	0.85	0.53	0.27	0.65	0.20	0.68	0.13	0.71	0.33	0.56
CF-Precision	**<0.05**	0	**<0.05**	0	**<0.05**	0.12	0.21	0.33	0.91	0.48	**<0.05**	0.17
CF-Recall	**<0.05**	0	**<0.05**	0	0.17	0.31	0.91	0.48	0.39	0.62	**<0.05**	0.28
CF-FMeasure	**<0.05**	0	**<0.05**	0	0.05	0.24	0.48	0.40	0.52	0.59	**<0.05**	0.24

Table 22: Comparing SBRM$^+_{\text{NSGA-II}}$-PART with RBRM$^+$-PART in terms of MLQMs for the Jitsi case study

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.87
MAE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.12
RMSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.11
RAE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.06
RRSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.05
FF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.88
FF-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.88
FF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.88
CC-Precision	**<0.05**	0.77	**<0.05**	0.85	0.070	0.75	**<0.05**	0.81	**<0.05**	0.92	**<0.05**	0.82
CC-Recall	0.344	0.63	0.121	0.71	**<0.05**	0.79	**<0.05**	0.80	**<0.05**	0.92	**<0.05**	0.79

CC-FMeasure	0.212	0.67	**<0.05**	0.82	**<0.05**	0.79	**<0.05**	0.79	**<0.05**	0.93	**<0.05**	0.81
FC-Precision	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.97	**<0.05**	0.97	**<0.05**	1	**<0.05**	0.92
FC-Recall	**<0.05**	0.95	**<0.05**	0.99	**<0.05**	0.94	**<0.05**	0.94	**<0.05**	0.98	**<0.05**	0.90
FC-FMeasure	**<0.05**	1	**<0.05**	0.98	**<0.05**	0.96	**<0.05**	0.95	**<0.05**	1	**<0.05**	0.92
CF-Precision	**<0.05**	0.83	**<0.05**	0.77	0.129	0.71	0.472	0.60	0.130	0.71	**<0.05**	0.69
CF-Recall	0.241	0.66	**<0.05**	0.82	0.571	0.58	1	0.50	0.121	0.71	**<0.05**	0.63
CF-FMeasure	**<0.05**	0.78	**<0.05**	0.78	0.406	0.62	0.791	0.54	0.130	0.71	**<0.05**	0.67

Table 23: Comparing SBRM$^+_{NSGA-III}$-PART with RBRM$^+$-PART in terms of MLQMs for the Jitsi case study

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98	**<0.05**	1	**<0.05**	0.97
MAE	**<0.05**	<0.05	**<0.05**	<0.05	**<0.05**	0.02	**<0.05**	0.02	**<0.05**	<0.05	**<0.05**	0.03
RMSE	**<0.05**	<0.05	**<0.05**	<0.05	**<0.05**	<0.05	**<0.05**	0.02	**<0.05**	<0.05	**<0.05**	0.03
RAE	**<0.05**	<0.05	**<0.05**	0.11	**<0.05**	0.09	**<0.05**	0.15	**<0.05**	0.04	**<0.05**	0.25
RRSE	**<0.05**	<0.05	**<0.05**	0.16	**<0.05**	0.08	**<0.05**	0.07	**<0.05**	0.08	**<0.05**	0.24
FF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
FF-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
FF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CC-Precision	**<0.05**	0.10	0.05	0.24	0.68	0.44	1	0.50	0.24	0.66	**<0.05**	0.38
CC-Recall	**<0.05**	0.16	**<0.05**	0.23	0.50	0.41	0.73	0.45	0.62	0.57	**<0.05**	0.36
CC-FMeasure	**<0.05**	0.14	**<0.05**	0.23	0.39	0.38	0.74	0.45	0.55	0.59	**<0.05**	0.36
FC-Precision	0.41	0.39	**<0.05**	0.14	**<0.05**	0.20	0.15	0.31	0.05	0.24	**<0.05**	0.28
FC-Recall	0.10	0.28	**<0.05**	0.04	**<0.05**	0.14	**<0.05**	0.19	**<0.05**	0.20	**<0.05**	0.20
FC-FMeasure	0.16	0.31	**<0.05**	0.06	**<0.05**	0.16	0.07	0.26	**<0.05**	0.22	**<0.05**	0.23
CF-Precision	**<0.05**	0.12	**<0.05**	0.08	**<0.05**	0.07	**<0.05**	0.03	**<0.05**	0.04	**<0.05**	0.07
CF-Recall	**<0.05**	0.08	**<0.05**	0.09	**<0.05**	0.04	**<0.05**	0.03	**<0.05**	0.12	**<0.05**	0.07
CF-FMeasure	**<0.05**	0.11	**<0.05**	0.08	**<0.05**	0.04	**<0.05**	0.03	**<0.05**	0.09	**<0.05**	0.07

Detailed Results of RQ3

Table 24: Descriptive statistics for ARI of SBRM$^+_{NSGA-II}$-C45, SBRM$^+_{NSGA-III}$-C45, SBRM$^+_{NSGA-II}$-PART and SBRM$^+_{NSGA-III}$-PART for the Cisco case study (%)

MLQMs	SBRM$^+_{NSGA-II}$-C45			SBRM$^+_{NSGA-III}$-C45			SBRM$^+_{NSGA-II}$-PART			SBRM$^+_{NSGA-II}$-PART		
	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
Accuracy	26.5	5.3	13.4	18.7	3.0	12.2	16.9	4.6	10.1	11.4	0.5	8.0
MAE	14.3	4.9	9.1	11.0	3.3	8.1	8.7	2.2	5.3	5.8	0.5	4.3
RMSE	19.3	4.0	10.8	13.9	2.6	9.6	12.6	3.6	7.8	8.8	1.0	6.3
RAE	41.9	15.6	27.3	30.2	7.5	22.3	25.1	6.4	15.5	19.2	2.5	11.3
RRSE	45.8	10.2	25.7	32.0	4.6	20.9	28.6	8.3	17.6	22.8	3.6	13.1
FF-Precision	16.2	1.6	7.7	13.4	2.6	9.4	14.5	4.0	8.5	12.0	1.0	7.9
FF-Recall	17.9	-3.9	3.6	13.7	-4.5	4.0	14.2	3.4	8.3	10.8	1.0	7.4
FF-FMeasure	17.0	-0.9	5.8	13.6	-0.5	7.0	14.4	3.7	8.4	11.2	1.1	7.6
CC-Precision	25.5	11.6	15.7	16.2	4.9	12.0	11.0	1.3	5.5	8.1	-7.6	-0.5
CC-Recall	15.6	2.0	7.8	8.1	-2.8	3.1	10.2	1.0	5.2	7.1	-4.9	0.2
CC-FMeasure	20.9	7.1	11.9	11.6	1.3	7.8	10.7	1.2	5.4	7.6	-6.3	-0.1
FC-Precision	88.5	-18.2	11.2	-6.7	-34.3	-16.3	0.0	-16.7	-3.7	0.0	-16.7	-3.7
FC-Recall	94.1	-7.5	22.6	-1.1	-14.9	-6.2	0.0	-11.1	-2.4	0.0	-11.1	-2.4
FC-FMeasure	93.2	-10.7	19.3	-1.8	-17.9	-8.8	0.0	-12.5	-2.8	0.0	-12.5	-2.8
CF-Precision	35.6	3.4	13.3	41.0	-3.3	16.2	17.4	-1.1	6.1	36.1	-1.9	20.5
CF-Recall	33.2	4.4	14.5	54.0	5.2	20.5	15.2	-3.1	4.8	36.1	-1.1	20.1
CF-FMeasure	28.1	5.0	14.9	45.9	6.1	19.7	16.3	-2.2	5.4	35.8	-1.4	20.3

Table 25: Descriptive statistics for ARI of SBRM$^+_{NSGA-II}$-C45, SBRM$^+_{NSGA-III}$-C45, SBRM$^+_{NSGA-II}$-PART and SBRM$^+_{NSGA-III}$-PART for the Jitsi case study (%)

MLQMs	SBRM$^+_{NSGA-II}$-C45			SBRM$^+_{NSGA-III}$-C45			SBRM$^+_{NSGA-II}$-PART			SBRM$^+_{NSGA-III}$-PART		
	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean

Accuracy	15.0	3.9	11.4	4.9	-5.0	0.3	10.7	3.0	6.5	16.6	1.2	11.6
MAE	8.0	2.7	6.3	1.7	-2.3	0.2	5.9	1.4	3.3	8.3	0.9	5.9
RMSE	8.3	2.6	6.3	2.9	-1.4	0.8	6.5	1.3	3.7	9.9	1.1	6.8
RAE	23.1	10.3	18.6	6.0	-2.2	2.2	10.2	5.4	7.9	7.0	-1.2	4.1
RRSE	19.3	8.0	14.9	5.4	-0.2	3.1	9.3	4.0	6.9	6.1	-0.6	3.3
FF-Precision	15.7	3.8	11.5	5.4	-4.1	1.1	9.4	3.7	5.6	11.9	2.9	9.2
FF-Recall	10.5	-3.3	4.4	6.7	-6.0	-0.8	9.5	0.3	4.8	12.7	4.2	10.0
FF-FMeasure	13.4	0.6	8.4	5.4	-5.0	0.2	9.5	2.0	5.2	12.1	3.5	9.7
CC-Precision	23.9	12.5	16.2	10.1	-6.0	3.2	12.3	-2.9	6.2	11.7	-8.8	2.0
CC-Recall	31.4	13.8	20.1	12.6	-6.2	3.5	16.2	-3.4	7.3	18.6	-10.2	2.0
CC-FMeasure	27.7	13.2	18.1	11.4	-6.1	3.4	13.1	-3.1	6.8	14.8	-9.5	2.0
FC-Precision	6.9	-3.3	1.4	9.6	-1.8	3.2	10.7	1.7	6.5	2.6	-12.8	-3.6
FC-Recall	11.7	-3.7	1.7	12.1	-6.1	2.6	15.2	2.6	7.5	0.8	-12.0	-4.6
FC-FMeasure	10.0	-2.8	1.6	11.0	-4.6	2.8	12.3	2.4	7.0	1.6	-12.4	-4.1
CF-Precision	34.0	14.3	23.3	9.1	-7.0	-0.2	5.7	-1.2	2.4	-1.1	-11.8	-7.1
CF-Recall	44.1	15.7	28.3	16.1	-6.9	2.6	8.6	-3.5	2.6	1.9	-16.5	-7.5
CF-FMeasure	41.8	15.2	26.8	13.9	-7.1	1.8	7.2	-2.0	2.5	0.4	-13.6	-7.3

Detailed Results of RQ4

Table 26. Comparing SBRM$^+_{NSGA\text{-}II}$-C45 with SBRM$^+_{NSGA\text{-}III}$-C45 and SBRM$^+_{NSGA\text{-}II}$-PART with SBRM$^+_{NSGA\text{-}III}$-PART in terms of fitness values for both Cisco and Jitsi case studies*

MLQMs	FV-O1		FV-O2		FV-O3		OFV	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Comparing SBRM$^+_{NSGA\text{-}II}$-C45 with SBRM$^+_{NSGA\text{-}III}$-C45 for the Cisco case study								
Iteration-1	**<0.05**	0.33	**<0.05**	0.40	**<0.05**	1	**<0.05**	1
Iteration-2	0.53	0.50	**<0.05**	0.22	**<0.05**	0.90	**<0.05**	0.82
Iteration-3	**<0.05**	0.79	**<0.05**	0.24	**<0.05**	0.82	**<0.05**	0.89
Iteration-4	**<0.05**	0.63	**<0.05**	0.43	**<0.05**	1	**<0.05**	0.93
Iteration-5	**<0.05**	0.61	**<0.05**	0.45	**<0.05**	1	**<0.05**	1
Overall	**<0.05**	0.59	**<0.05**	0.35	**<0.05**	0.94	**<0.05**	0.92
Comparing SBRM$^+_{NSGA\text{-}II}$-PART with SBRM$^+_{NSGA\text{-}III}$-PART for the Cisco case study								
Iteration-1	**<0.05**	0.65	**<0.05**	0.31	**<0.05**	0.72	**<0.05**	0.39
Iteration-2	**<0.05**	0.59	0.21	0.49	**<0.05**	1	**<0.05**	1
Iteration-3	**<0.05**	0.45	**<0.05**	0.65	**<0.05**	1	**<0.05**	1
Iteration-4	**<0.05**	0.39	**<0.05**	0.67	**<0.05**	1	**<0.05**	1
Iteration-5	0.60	0.50	0.82	0.50	**<0.05**	1	**<0.05**	1
Overall	0.06	0.50	**<0.05**	0.53	**<0.05**	0.94	**<0.05**	0.97
Comparing SBRM$^+_{NSGA\text{-}II}$-C45 with SBRM$^+_{NSGA\text{-}III}$-C45 for the Jitsi case study								
Iteration-1	**<0.05**	0.11	**<0.05**	0.90	**<0.05**	0.98	**<0.05**	0.89
Iteration-2	**<0.05**	0.24	**<0.05**	0.69	**<0.05**	0.79	**<0.05**	0.59
Iteration-3	**<0.05**	0.24	**<0.05**	0.66	**<0.05**	0.75	**<0.05**	0.57
Iteration-4	**<0.05**	0.41	**<0.05**	0.64	**<0.05**	0.63	**<0.05**	0.57
Iteration-5	**<0.05**	0.16	**<0.05**	0.63	**<0.05**	0.68	**<0.05**	0.47
Overall	**<0.05**	0.24	**<0.05**	0.63	**<0.05**	0.64	**<0.05**	0.55
Comparing SBRM$^+_{NSGA\text{-}II}$-PART with SBRM$^+_{NSGA\text{-}III}$-PART for the Jitsi case study								
Iteration-1	**<0.05**	0.22	**<0.05**	0.93	**<0.05**	1	**<0.05**	1
Iteration-2	**<0.05**	0.62	**<0.05**	0.43	**<0.05**	0.29	0.21	0.49
Iteration-3	**<0.05**	0.74	**<0.05**	0.38	**<0.05**	0.39	**<0.05**	0.57
Iteration-4	**<0.05**	0.53	**<0.05**	0.55	0.28	0.51	**<0.05**	0.53
Iteration-5	0.81	0.50	**<0.05**	0.58	**<0.05**	0.63	**<0.05**	0.58
Overall	**<0.05**	0.55	**<0.05**	0.52	0.88	0.50	**<0.05**	0.57

Table 27. Comparing SBRM+NSGA-II-C45 with SBRM+NSGA-III-C45 and SBRM+NSGA-II-PART with SBRM+NSGA-III-PART in terms of indicators for both Cisco and Jitsi case studies*

MLQMs	HV		IGD		ϵ		ED		GD		GS	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Comparing SBRM$^+_{NSGA\text{-}II}$-C45 with SBRM$^+_{NSGA\text{-}III}$-C45 for the Cisco case study												

Iteration-1	**<0.05**	0	**<0.05**	1	**<0.05**	1	**<0.05**	0.10	**<0.05**	0	**<0.05**	1
Iteration-2	**<0.05**	0.15	**<0.05**	0.86	**<0.05**	0.90	0.97	0.49	0.22	0.67	**<0.05**	1
Iteration-3	**<0.05**	0.04	**<0.05**	0.95	**<0.05**	0.80	0.24	0.34	0.58	0.58	**<0.05**	0.93
Iteration-4	**<0.05**	0.08	**<0.05**	1	**<0.05**	1	**<0.05**	0.83	**<0.05**	0.87	**<0.05**	0.85
Iteration-5	**<0.05**	0.23	**<0.05**	1	**<0.05**	1	0.62	0.57	0.39	0.62	**<0.05**	0.96
Overall	**<0.05**	0.12	**<0.05**	0.94	**<0.05**	0.94	0.57	0.53	0.09	0.60	**<0.05**	0.95
Comparing SBRM$^+_{NSGA-II}$-PART with SBRM$^+_{NSGA-III}$-PART for the Cisco case study												
Iteration-1	**<0.05**	0.90	**<0.05**	0.91	**<0.05**	1	**<0.05**	0.91	0.31	0.64	**<0.05**	0
Iteration-2	**<0.05**	0	**<0.05**	1	**<0.05**	1	0.07	0.74	**<0.05**	0.84	0.25	0.34
Iteration-3	**<0.05**	0	**<0.05**	1	**<0.05**	1	**<0.05**	0.90	**<0.05**	0.93	0.06	0.75
Iteration-4	**<0.05**	0	**<0.05**	1	**<0.05**	1	0.06	0.75	**<0.05**	0.89	0.05	0.77
Iteration-5	**<0.05**	0.05	**<0.05**	1	**<0.05**	1	**<0.05**	0.77	**<0.05**	0.85	**<0.05**	0.77
Overall	**<0.05**	0.07	**<0.05**	0.99	**<0.05**	0.98	**<0.05**	0.72	**<0.05**	0.76	0.26	0.57
Comparing SBRM$^+_{NSGA-II}$-C45 with SBRM$^+_{NSGA-III}$-C45 for the Jitsi case study												
Iteration-1	**<0.05**	0	**<0.05**	1	**<0.05**	1	**<0.05**	0.82	**<0.05**	0	**<0.05**	0
Iteration-2	**<0.05**	0.79	0.19	0.68	**<0.05**	0	0.91	0.52	0.74	0.55	**<0.05**	0.09
Iteration-3	**<0.05**	0.85	0.80	0.54	**<0.05**	0.10	0.68	0.44	0.58	0.42	**<0.05**	0
Iteration-4	0.25	0.66	0.68	0.44	**<0.05**	0.18	0.07	0.26	0.74	0.45	**<0.05**	0.08
Iteration-5	**<0.05**	0.93	**<0.05**	0.10	**<0.05**	0.04	0.84	0.53	0.25	0.66	**<0.05**	0
Overall	**<0.05**	0.65	0.74	0.48	**<0.05**	0.27	0.29	0.44	**<0.05**	0.33	**<0.05**	0.03
Comparing SBRM$^+_{NSGA-II}$-PART with SBRM$^+_{NSGA-III}$-PART for the Jitsi case study												
Iteration-1	**<0.05**	0	**<0.05**	1	**<0.05**	1	0.76	0.55	**<0.05**	0.05	**<0.05**	0.84
Iteration-2	0.85	0.53	0.06	0.25	0.91	0.52	0.27	0.35	0.48	0.60	0.06	0.25
Iteration-3	0.68	0.44	0.25	0.34	0.53	0.59	0.57	0.42	0.28	0.65	**<0.05**	0.14
Iteration-4	0.80	0.54	0.25	0.34	0.68	0.44	0.73	0.45	0.11	0.72	**<0.05**	0.16
Iteration-5	0.48	0.60	0.80	0.46	0.17	0.31	0.14	0.70	0.06	0.75	**<0.05**	0.20
Overall	0.34	0.44	0.36	0.45	0.55	0.54	0.49	0.46	0.12	0.59	**<0.05**	0.21

Detailed Results of RQ5

Table 28: Comparing SBRM$^+_{NSGA-II}$-C45 with SBRM$^+_{NSGA-III}$-C45 in terms of MLQMs for the Cisco case study*

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	0.87	**<0.05**	0.85	0.17	0.69	0.14	0.70	0.53	0.59	**<0.05**	0.65
MAE	**<0.05**	0.02	**<0.05**	0.10	**<0.05**	0.14	**<0.05**	0.18	0.26	0.35	**<0.05**	0.31
RMSE	**<0.05**	0.07	**<0.05**	0.10	0.14	0.30	0.14	0.30	0.48	0.40	**<0.05**	0.33
RAE	**<0.05**	0	**<0.05**	0.09	**<0.05**	0.06	**<0.05**	0.14	0.03	0.21	**<0.05**	0.22
RRSE	**<0.05**	0.01	**<0.05**	0.08	**<0.05**	0.08	**<0.05**	0.20	0.11	0.28	**<0.05**	0.25
FF-Precision	**<0.05**	0.22	**<0.05**	0.10	**<0.05**	0.19	0.17	0.32	0.24	0.34	**<0.05**	0.32
FF-Recall	0.42	0.39	0.27	0.35	0.40	0.39	0.45	0.40	0.45	0.40	0.10	0.40
FF-FMeasure	0.06	0.25	**<0.05**	0.15	0.07	0.26	0.20	0.33	0.33	0.37	**<0.05**	0.35
CC-Precision	**<0.05**	0.89	**<0.05**	0.95	**<0.05**	0.96	**<0.05**	0.94	0.01	0.86	**<0.05**	0.89
CC-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.94	**<0.05**	0.97
CC-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.98	**<0.05**	0.92	**<0.05**	0.95
FC-Precision	-	0.50	-	0.50	-	0.50	-	0.50	0.08	0.65	0.08	0.53
FC-Recall	-	0.50	-	0.50	-	0.50	-	0.50	0.08	0.65	0.08	0.53
FC-FMeasure	-	0.50	-	0.50	-	0.50	-	0.50	0.08	0.65	0.08	0.53
CF-Precision	**<0.05**	0.91	**<0.05**	0.96	0.43	0.61	1	0.50	0.50	0.41	**<0.05**	0.67
CF-Recall	**<0.05**	0.99	**<0.05**	1	0.41	0.62	0.68	0.44	0.88	0.53	**<0.05**	0.71
CF-FMeasure	**<0.05**	1	**<0.05**	1	0.24	0.66	1	0.50	0.97	0.51	**<0.05**	0.73

* "-" represents that value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero for both approaches and p-value cannot be calculated

Table 29: Comparing SBRM$^+_{NSGA-II}$-PART with SBRM$^+_{NSGA-III}$-PART in terms of MLQMs for the Cisco case study*

MLQMs/Iterations	**Iteration-1**		**Iteration-2**		**Iteration-3**		**Iteration-4**		**Iteration-5**		**Overall**	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	0.31	0.36	0.32	0.64	0.74	0.55	0.17	0.69	0.08	0.74	0.10	0.60
MAE	0.23	0.67	0.19	0.32	0.85	0.47	0.22	0.33	0.08	0.26	0.12	0.41
RMSE	0.31	0.64	0.48	0.40	0.91	0.48	0.23	0.34	0.19	0.32	0.18	0.42

RAE	0.31	0.36	**<0.05**	0.18	**<0.05**	0.20	0.22	0.33	**<0.05**	0.18	**<0.05**	0.30
RRSE	0.53	0.41	**<0.05**	0.23	**<0.05**	0.20	0.31	0.36	0.05	0.24	**<0.05**	0.32
FF-Precision	**<0.05**	0.04	0.07	0.26	0.08	0.27	1	0.50	0.60	0.58	0.08	0.40
FF-Recall	**<0.05**	0.10	0.08	0.26	0.16	0.31	0.60	0.58	0.47	0.60	0.13	0.41
FF-FMeasure	**<0.05**	0.05	0.10	0.28	0.14	0.30	0.85	0.53	0.41	0.62	0.13	0.41
CC-Precision	0.10	0.72	**<0.05**	0.92	**<0.05**	0.86	**<0.05**	0.94	**<0.05**	0.91	**<0.05**	0.89
CC-Recall	**<0.05**	0.93	**<0.05**	0.89	**<0.05**	0.81	**<0.05**	0.90	**<0.05**	0.90	**<0.05**	0.87
CC-FMeasure	**<0.05**	0.89	**<0.05**	0.91	**<0.05**	0.84	**<0.05**	0.95	**<0.05**	0.91	**<0.05**	0.89
FC-Precision	-	0.50	-	0.50	-	0.50	-	0.50	-	0.50	-	0.50
FC-Recall	-	0.50	-	0.50	-	0.50	-	0.50	-	0.50	-	0.50
FC-FMeasure	-	0.50	-	0.50	-	0.50	-	0.50	-	0.50	-	0.50
CF-Precision	0.34	0.37	0.97	0.49	0.85	0.53	0.05	0.24	**<0.05**	0.20	0.26	0.43
CF-Recall	0.62	0.43	0.88	0.53	0.88	0.48	0.11	0.28	**<0.05**	0.16	0.21	0.43
CF-FMeasure	0.29	0.36	1	0.51	1	0.50	0.09	0.27	**<0.05**	0.18	0.24	0.43

* "-" represents that value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero for both approaches and p-value cannot be calculate

Table 30: Comparing SBRM$^{+}_{NSGA\text{-}II}$-C45 (W_1) with SBRM$^{+}_{NSGA\text{-}II}$-PART (W_2) in terms of MLQMs for the Cisco case study*

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	0.96	**<0.05**	0.99	**<0.05**	0.98	**<0.05**	0.93	**<0.05**	0.96
MAE	**<0.05**	0.09	**<0.05**	0.12	**<0.05**	0.08	**<0.05**	0.13	0.063	0.25	**<0.05**	0.26
RMSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.07	**<0.05**	0.02
RAE	**<0.05**	0	**<0.05**	0.08	**<0.05**	0.06	**<0.05**	0.08	**<0.05**	0.12	**<0.05**	0.16
RRSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.02	**<0.05**	0
FF-Precision	**<0.05**	0	**<0.05**	0.13	0.150	0.31	0.104	0.28	0.273	0.35	**<0.05**	0.28
FF-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98	**<0.05**	0.93	**<0.05**	0.99
FF-FMeasure	**<0.05**	1	**<0.05**	0.89	**<0.05**	0.93	**<0.05**	0.87	**<0.05**	0.78	**<0.05**	0.82
CC-Precision	**<0.05**	1	**<0.05**	0.98	**<0.05**	1	**<0.05**	1	**<0.05**	0.98	**<0.05**	0.99
CC-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.97	**<0.05**	0.98
CC-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.97	**<0.05**	0.99
FC-Precision	-	0.50	-	0.50	-	0.50	-	0.50	0.078	0.65	0.082	0.53
FC-Recall	-	0.50	-	0.50	-	0.50	-	0.50	0.078	0.65	0.082	0.53
FC-FMeasure	-	0.50	-	0.50	-	0.50	-	0.50	0.078	0.65	0.082	0.53
CF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CF-Recall	**<0.05**	1	**<0.05**	0.90	0.082	0.74	0.104	0.72	0.130	0.71	**<0.05**	0.81
CF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	1	**<0.05**	1	**<0.05**	1

* "-" represents that value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero for both approaches and p-value cannot be calculated, W_1 and W_2 are two winners of first two comparisons.

Table 31: Comparing SBRM$^{+}_{NSGA\text{-}II}$-C45 with SBRM$^{+}_{NSGA\text{-}III}$-C45 in terms of MLQMs for the Jitsi case study*

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$	*p*-value	$\hat{A}_{12}$
Accuracy	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.97
MAE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.02
RMSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.02
RAE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
RRSE	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
FF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.99	**<0.05**	0.94
FF-Recall	**<0.05**	0.21	0.31	0.64	**<0.05**	0.95	**<0.05**	0.86	**<0.05**	0.89	**<0.05**	0.72
FF-FMeasure	**<0.05**	1	**<0.05**	0.98	**<0.05**	0.99	**<0.05**	0.97	**<0.05**	0.94	**<0.05**	0.90
CC-Precision	**<0.05**	0.99	**<0.05**	0.94	**<0.05**	0.95	**<0.05**	0.92	**<0.05**	0.99	**<0.05**	0.95
CC-Recall	**<0.05**	0.98	**<0.05**	0.98	**<0.05**	0.98	**<0.05**	0.95	**<0.05**	1	**<0.05**	0.96
CC-FMeasure	**<0.05**	0.99	**<0.05**	0.95	**<0.05**	0.97	**<0.05**	0.95	**<0.05**	1	**<0.05**	0.95
FC-Precision	**<0.05**	0.82	0.29	0.36	0.85	0.47	0.47	0.40	**<0.05**	0.24	0.44	0.45
FC-Recall	**<0.05**	0.98	0.97	0.49	0.73	0.55	0.97	0.51	0.76	0.46	0.10	0.60
FC-FMeasure	**<0.05**	0.95	0.68	0.44	0.79	0.54	0.97	0.51	0.21	0.33	0.43	0.55
CF-Precision	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1

CF-Recall	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1
CF-FMeasure	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1

Table 32: Comparing SBRM$^{+}_{\text{NSGA-II}}$-PART with SBRM$^{+}_{\text{NSGA-III}}$-PART in terms of MLQMs for the Jitsi case study*

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$
Accuracy	**<0.05**	0	**<0.05**	0.21	**<0.05**	0.22	**<0.05**	0.18	**<0.05**	0.15	**<0.05**	0.30
MAE	**<0.05**	1	**<0.05**	0.80	**<0.05**	0.79	**<0.05**	0.82	**<0.05**	0.85	**<0.05**	0.71
RMSE	**<0.05**	1	**<0.05**	0.78	**<0.05**	0.79	**<0.05**	0.82	**<0.05**	0.85	**<0.05**	0.70
RAE	**<0.05**	0.10	**<0.05**	0	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.03	**<0.05**	0.21
RRSE	**<0.05**	0.10	**<0.05**	0	**<0.05**	0.03	**<0.05**	0	**<0.05**	0.03	**<0.05**	0.21
FF-Precision	**<0.05**	0	**<0.05**	0.24	0.06	0.25	**<0.05**	0.13	**<0.05**	0.10	**<0.05**	0.29
FF-Recall	**<0.05**	0	**<0.05**	0.04	**<0.05**	0.11	**<0.05**	0.12	**<0.05**	0.08	**<0.05**	0.18
FF-FMeasure	**<0.05**	0	**<0.05**	0.11	**<0.05**	0.16	**<0.05**	0.14	**<0.05**	0.10	**<0.05**	0.24
CC-Precision	**<0.05**	1	**<0.05**	0.82	0.14	0.70	0.10	0.73	0.12	0.71	**<0.05**	0.77
CC-Recall	**<0.05**	0.99	**<0.05**	0.81	**<0.05**	0.79	0.11	0.72	**<0.05**	0.77	**<0.05**	0.80
CC-FMeasure	**<0.05**	1	**<0.05**	0.81	**<0.05**	0.77	0.11	0.72	0.09	0.73	**<0.05**	0.79
FC-Precision	**<0.05**	0.98	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	1	**<0.05**	0.97
FC-Recall	**<0.05**	0.96	**<0.05**	1	**<0.05**	0.99	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
FC-FMeasure	**<0.05**	0.96	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
CF-Precision	**<0.05**	0.95	**<0.05**	0.98	**<0.05**	0.99	**<0.05**	0.97	**<0.05**	0.97	**<0.05**	0.97
CF-Recall	**<0.05**	0.96	**<0.05**	0.98	**<0.05**	1	**<0.05**	0.96	**<0.05**	0.93	**<0.05**	0.96
CF-FMeasure	**<0.05**	0.95	**<0.05**	0.98	**<0.05**	1	**<0.05**	0.97	**<0.05**	0.95	**<0.05**	0.97

Table 33: Comparing SBRM$^{+}_{\text{NSGA-II}}$-C45 with SBRM$^{+}_{\text{NSGA-II}}$-PART in terms of MLQMs for the Jitsi case study*

MLQMs/Iterations	Iteration-1		Iteration-2		Iteration-3		Iteration-4		Iteration-5		Overall	
	p-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$	*p*-value	$\widehat{A}_{12}$
Accuracy	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.08
MAE	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.95
RMSE	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.99	**<0.05**	0.88
RAE	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98
RRSE	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	1	**<0.05**	0.98	**<0.05**	0.89
FF-Precision	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.02
FF-Recall	**<0.05**	0.05	**<0.05**	0.06	**<0.05**	0.11	**<0.05**	0.15	**<0.05**	0.17	**<0.05**	0.24
FF-FMeasure	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0.01	**<0.05**	0.08
CC-Precision	**<0.05**	0.13	**<0.05**	0.18	0.173	0.32	0.190	0.32	0.121	0.29	**<0.05**	0.26
CC-Recall	**<0.05**	0.06	**<0.05**	0.14	**<0.05**	0.22	**<0.05**	0.21	**<0.05**	0.23	**<0.05**	0.19
CC-FMeasure	**<0.05**	0.08	**<0.05**	0.15	0.123	0.29	0.063	0.25	**<0.05**	0.22	**<0.05**	0.21
FC-Precision	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
FC-Recall	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
FC-FMeasure	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0	**<0.05**	0
CF-Precision	0.064	0.25	0.290	0.65	0.186	0.68	**<0.05**	0.87	0.064	0.75	**<0.05**	0.64
CF-Recall	**<0.05**	0	0.082	0.27	0.481	0.40	0.739	0.45	0.739	0.45	**<0.05**	0.33
CF-FMeasure	**<0.05**	0	0.345	0.37	1	0.50	0.529	0.59	0.650	0.57	0.268	0.44

* W_1 and W_2 are two winners of first two comparisons.

Detailed Results of RQ6

Table 34: Correlation analysis of MLQMs with AFVs corresponding to SBRM$^{+}_{\text{NSGA-II}}$-C45, SBRM$^{+}_{\text{NSGA-III}}$-C45, SBRM$^{+}_{\text{NSGA-II}}$-PART, and SBRM$^{+}_{\text{NSGA-III}}$-PART for the Cisco case study*

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-II}}$-C45										
MLQMs	**AFV-O1**		**AFV-O2**		**AFV-O3**		**OAFV**		**HV**	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	-0.18	0.20	-0.33	**<0.05**	-0.24	0.09	-0.39	**<0.05**	0.48	**<0.05**
MAE	0.15	0.30	0.34	**<0.05**	0.18	0.20	0.37	**<0.05**	-0.46	**<0.05**
RMSE	0.18	0.21	0.33	**<0.05**	0.23	0.11	0.40	**<0.05**	-0.49	**<0.05**
RAE	0.14	0.34	0.32	**<0.05**	0.19	0.18	0.37	**<0.05**	-0.46	**<0.05**

RRSE	0.17	0.24	0.33	**<0.05**	0.24	0.09	0.40	**<0.05**	-0.49	**<0.05**
FF-Precision	-0.15	0.31	-0.34	**<0.05**	-0.18	0.21	-0.35	**<0.05**	0.44	**<0.05**
FF-Recall	-0.23	0.10	-0.21	0.14	-0.43	**<0.05**	-0.42	**<0.05**	0.46	**<0.05**
FF-FMeasure	-0.23	0.11	-0.38	**<0.05**	-0.24	0.09	-0.41	**<0.05**	0.51	**<0.05**
CC-Precision	-0.24	0.09	-0.37	**<0.05**	-0.21	0.15	-0.37	**<0.05**	0.45	**<0.05**
CC-Recall	-0.15	0.30	-0.36	**<0.05**	-0.10	0.47	-0.35	**<0.05**	0.48	**<0.05**
CC-FMeasure	-0.19	0.18	-0.37	**<0.05**	-0.19	0.19	-0.39	**<0.05**	0.50	**<0.05**
FC-Precision	0.19	0.19	0.15	0.31	-0.25	0.08	-0.22	0.12	0.19	0.19
FC-Recall	0.19	0.17	0.15	0.29	-0.25	0.08	-0.23	0.12	0.19	0.18
FC-FMeasure	0.19	0.19	0.15	0.31	-0.25	0.08	-0.22	0.12	0.19	0.19
CF-Precision	-0.02	0.91	0.24	0.10	-0.32	**<0.05**	-0.03	0.84	0.04	0.81
CF-Recall	0.06	0.69	-0.01	0.93	0.10	0.49	0.16	0.28	-0.29	**<0.05**
CF-FMeasure	0.09	0.54	0.17	0.23	-0.13	0.37	0.09	0.55	-0.26	0.07

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-III}}$-C45

MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value
Accuracy	-0.21	0.14	-0.12	0.41	0.47	**<0.05**	0.42	**<0.05**	-0.22	0.13
MAE	0.23	0.10	0.09	0.55	-0.46	**<0.05**	-0.41	**<0.05**	0.23	0.11
RMSE	0.21	0.14	0.13	0.36	-0.47	**<0.05**	-0.42	**<0.05**	0.21	0.14
RAE	0.19	0.18	0.17	0.24	-0.44	**<0.05**	-0.41	**<0.05**	0.2	0.16
RRSE	0.17	0.23	0.21	0.15	-0.46	**<0.05**	-0.43	**<0.05**	0.2	0.16
FF-Precision	-0.24	0.09	0.12	0.39	0.54	**<0.05**	0.45	**<0.05**	-0.36	**<0.05**
FF-Recall	-0.17	0.24	0.02	0.90	0.44	**<0.05**	0.4	**<0.05**	-0.25	0.08
FF-FMeasure	-0.19	0.19	0.05	0.71	0.49	**<0.05**	0.42	**<0.05**	-0.29	**<0.05**
CC-Precision	-0.2	0.17	-0.18	0.20	0.26	0.06	0.25	0.08	-0.11	0.45
CC-Recall	-0.1	0.49	-0.28	0.05	0.19	0.19	0.14	0.32	0	1
CC-FMeasure	-0.16	0.27	-0.22	0.13	0.23	0.11	0.21	0.15	-0.06	0.65
FC-Precision	-	-	-	-	-	-	-	-	-	-
FC-Recall	-	-	-	-	-	-	-	-	-	-
FC-FMeasure	-	-	-	-	-	-	-	-	-	-
CF-Precision	0.11	0.46	-0.26	0.07	0.31	**<0.05**	0.33	**<0.05**	-0.09	0.53
CF-Recall	0.03	0.82	-0.18	0.2	0.38	**<0.05**	0.36	**<0.05**	-0.17	0.23
CF-FMeasure	0.08	0.58	-0.23	0.110	0.35	0.010	0.35	0.010	-0.12	0.400

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-II}}$-PART

MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value
Accuracy	-0.21	0.14	-0.12	0.41	0.47	**<0.05**	0.42	**<0.05**	-0.22	0.13
MAE	0.23	0.10	0.09	0.55	-0.46	**<0.05**	-0.41	**<0.05**	0.23	0.11
RMSE	0.21	0.14	0.13	0.36	-0.47	**<0.05**	-0.42	**<0.05**	0.21	0.14
RAE	0.19	0.18	0.17	0.24	-0.44	**<0.05**	-0.41	**<0.05**	0.20	0.16
RRSE	0.17	0.23	0.21	0.15	-0.46	**<0.05**	-0.43	**<0.05**	0.20	0.16
FF-Precision	-0.24	0.09	0.12	0.39	0.54	**<0.05**	0.45	**<0.05**	-0.36	**<0.05**
FF-Recall	-0.17	0.24	0.02	0.90	0.44	**<0.05**	0.40	**<0.05**	-0.25	0.08
FF-FMeasure	-0.19	0.19	0.05	0.71	0.49	**<0.05**	0.42	**<0.05**	-0.29	**<0.05**
CC-Precision	-0.20	0.17	-0.18	0.20	0.26	0.06	0.25	0.08	-0.11	0.45
CC-Recall	-0.10	0.49	-0.28	0.05	0.19	0.19	0.14	0.32	**<0.05**	1
CC-FMeasure	-0.16	0.27	-0.22	0.13	0.23	0.11	0.21	0.15	-0.06	0.65
FC-Precision	-	-	-	-	-	-	-	-	-	-
FC-Recall	-	-	-	-	-	-	-	-	-	-
FC-FMeasure	-	-	-	-	-	-	-	-	-	-
CF-Precision	0.11	0.46	-0.26	0.07	0.31	**<0.05**	0.33	**<0.05**	-0.09	0.53
CF-Recall	0.03	0.82	-0.18	0.20	0.38	**<0.05**	0.36	**<0.05**	-0.17	0.23
CF-FMeasure	0.08	0.58	-0.23	0.11	0.35	**<0.05**	0.35	**<0.05**	-0.12	0.40

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-III}}$-PART

MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value
Accuracy	-0.25	0.085	-0.04	0.770	0.25	0.078	-0.06	0.691	-0.25	0.076
MAE	0.24	0.098	0.08	0.578	-0.21	0.146	0.10	0.508	0.21	0.150
RMSE	0.28	**<0.05**	0.11	0.448	-0.15	0.292	0.18	0.222	0.14	0.319

RAE	0.09	0.553	0.29	**<0.05**	0.07	0.643	0.22	0.132	-0.10	0.483
RRSE	0.14	0.319	0.31	**<0.05**	0.09	0.522	0.30	**<0.05**	-0.13	0.354
FF-Precision	-0.35	**<0.05**	0.08	0.579	0.25	0.074	-0.11	0.465	-0.27	0.060
FF-Recall	-0.34	**<0.05**	0.18	0.217	0.34	**<0.05**	0	0.986	-0.36	**<0.05**
FF-FMeasure	-0.36	**<0.05**	0.14	0.340	0.32	**<0.05**	-0.05	0.731	-0.34	**<0.05**
CC-Precision	0.18	0.201	-0.01	0.931	0.15	0.285	0.33	**<0.05**	-0.19	0.186
CC-Recall	0.31	**<0.05**	-0.22	0.131	-0.08	0.601	0.18	0.200	0.01	0.934
CC-FMeasure	0.29	**<0.05**	-0.07	0.616	0.04	0.765	0.30	**<0.05**	-0.11	0.446
FC-Precision	-	-	-	-	-	-	-	-	-	-
FC-Recall	-	-	-	-	-	-	-	-	-	-
FC-FMeasure	-	-	-	-	-	-	-	-	-	-
CF-Precision	0.07	0.616	-0.39	**<0.05**	-0.34	**<0.05**	-0.42	**<0.05**	0.41	**<0.05**
CF-Recall	0.06	0.673	-0.38	**<0.05**	-0.38	**<0.05**	-0.44	**<0.05**	0.45	**<0.05**
CF-FMeasure	0.09	0.554	-0.38	**<0.05**	0.25	0.078	-0.43	**<0.05**	0.43	**<0.05**

* "-" = represents that the value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero

Table 35: Correlation analysis of MLQMs with six indicators corresponding to SBRM$^{+}_{NSGA-II}$-C45, SBRM$^{+}_{NSGA-III}$-C45, SBRM$^{+}_{NSGA-II}$-PART, and SBRM$^{+}_{NSGA-III}$-PART for the Cisco case study*

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{NSGA-II}$-C45

MLQMs	HV		IGD		ϵ		ED		GD		GS	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	0.13	0.38	-0.24	0.09	-0.28	0.05	0.17	0.25	0.34	0.02	-0.12	0.42
MAE	-0.09	0.55	0.19	0.18	0.22	0.12	-0.20	0.17	-0.36	0.01	0.12	0.40
RMSE	-0.12	0.41	0.25	0.08	0.27	0.05	-0.19	0.18	-0.36	0.01	0.13	0.36
RAE	-0.10	0.50	0.20	0.16	0.23	0.11	-0.19	0.18	-0.36	0.01	0.12	0.39
RRSE	-0.11	0.44	0.25	0.08	0.28	0.05	-0.19	0.18	-0.36	0.01	0.14	0.35
FF-Precision	0.08	0.57	-0.18	0.20	-0.21	0.14	0.18	0.21	0.33	0.02	-0.07	0.62
FF-Recall	0.21	0.15	-0.38	0.01	-0.46	0.00	0.11	0.44	0.19	0.20	-0.14	0.32
FF-FMeasure	0.14	0.33	-0.25	0.08	-0.27	0.05	0.14	0.33	0.29	0.04	-0.07	0.65
CC-Precision	0.17	0.24	-0.24	0.09	-0.26	0.07	0.10	0.49	0.31	0.03	-0.11	0.44
CC-Recall	0.10	0.47	-0.18	0.21	-0.15	0.28	0.29	0.04	0.44	0.00	-0.07	0.65
CC-FMeasure	0.13	0.37	-0.23	0.10	-0.24	0.09	0.20	0.16	0.38	0.01	-0.11	0.44
FC-Precision	-0.16	0.25	-0.22	0.13	-0.24	0.09	0.08	0.59	0.14	0.32	-0.04	0.80
FC-Recall	-0.17	0.23	-0.22	0.13	-0.24	0.09	0.09	0.55	0.15	0.30	-0.04	0.79
FC-FMeasure	-0.16	0.25	-0.22	0.13	-0.24	0.09	0.08	0.59	0.14	0.32	-0.04	0.80
CF-Precision	0.02	0.90	-0.17	0.23	-0.32	0.03	0.01	0.94	-0.08	0.59	-0.20	0.15
CF-Recall	0.06	0.70	0.15	0.30	0.11	0.43	-0.21	0.15	-0.19	0.20	-0.06	0.65
CF-FMeasure	0.12	0.41	-0.03	0.82	-0.12	0.42	-0.12	0.40	-0.15	0.30	-0.25	0.08

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{NSGA-III}$-C45

MLQMs	HV		IGD		ϵ		ED		GD		GS	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	-0.50	0.00	0.53	0.00	0.48	0.00	0.10	0.51	0.18	0.21	0.31	0.03
MAE	0.47	0.00	-0.52	0.00	-0.48	0.00	-0.09	0.55	-0.18	0.22	-0.30	0.04
RMSE	0.54	0.00	-0.54	0.00	-0.49	0.00	-0.12	0.41	-0.21	0.15	-0.34	0.02
RAE	0.50	0.00	-0.51	0.00	-0.46	0.00	-0.11	0.46	-0.20	0.17	-0.28	0.05
RRSE	0.55	0.00	-0.53	0.00	-0.48	0.00	-0.12	0.42	-0.22	0.12	-0.29	0.04
FF-Precision	-0.45	0.00	0.55	0.00	0.54	0.00	0.03	0.86	0.13	0.37	0.29	0.04
FF-Recall	-0.45	0.00	0.50	0.00	0.47	0.00	0.10	0.50	0.21	0.15	0.29	0.04
FF-FMeasure	-0.46	0.00	0.53	0.00	0.51	0.00	0.05	0.74	0.15	0.29	0.29	0.04
CC-Precision	-0.36	0.01	0.33	0.02	0.29	0.04	0.14	0.32	0.20	0.17	0.18	0.20
CC-Recall	-0.25	0.07	0.24	0.09	0.20	0.16	0.06	0.66	0.11	0.46	0.10	0.50
CC-FMeasure	-0.32	0.03	0.29	0.04	0.25	0.08	0.13	0.38	0.17	0.25	0.14	0.32
FC-Precision	-	-	-	-	-	-	-	-	-	-	-	-
FC-Recall	-	-	-	-	-	-	-	-	-	-	-	-
FC-FMeasure	-	-	-	-	-	-	-	-	-	-	-	-
CF-Precision	-0.53	0.00	0.37	0.01	0.31	0.03	0.28	0.05	0.34	0.01	0.22	0.12

CF-Recall	-0.53	0.00	0.41	0.00	0.38	0.01	0.20	0.16	0.25	0.08	0.20	0.16
CF-FMeasure	-0.54	0.00	0.41	0.00	0.35	0.01	0.23	0.10	0.28	0.05	0.21	0.13
Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{NSGA-II}$-PART												
MLQMs	**HV**		**IGD**		**ϵ**		**ED**		**GD**		**GS**	**HV**
	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value
Accuracy	0.49	0.00	-0.49	0.00	-0.27	0.06	0.64	0.00	0.61	0.00	-0.26	0.07
MAE	-0.54	0.00	0.55	0.00	0.37	0.01	-0.65	0.00	-0.66	0.00	0.26	0.07
RMSE	-0.50	0.00	0.49	0.00	0.27	0.06	-0.66	0.00	-0.63	0.00	0.25	0.08
RAE	-0.50	0.00	0.51	0.00	0.32	0.02	-0.61	0.00	-0.66	0.00	0.22	0.12
RRSE	-0.49	0.00	0.48	0.00	0.23	0.11	-0.61	0.00	-0.60	0.00	0.23	0.11
FF-Precision	0.51	0.00	-0.52	0.00	-0.44	0.00	0.62	0.00	0.70	0.00	-0.23	0.11
FF-Recall	0.35	0.01	-0.34	0.02	-0.13	0.36	0.55	0.00	0.45	0.00	-0.30	0.04
FF-FMeasure	0.50	0.00	-0.50	0.00	-0.32	0.02	0.64	0.00	0.62	0.00	-0.29	0.04
CC-Precision	0.18	0.20	-0.16	0.27	0.02	0.89	0.46	0.00	0.45	0.00	-0.18	0.21
CC-Recall	0.13	0.36	-0.10	0.47	0.05	0.73	0.36	0.01	0.33	0.02	-0.12	0.39
CC-FMeasure	0.16	0.28	-0.12	0.40	0.07	0.65	0.45	0.00	0.41	0.00	-0.14	0.33
FC-Precision	-	-	-	-	-	-	-	-	-	-	-	-
FC-Recall	-	-	-	-	-	-	-	-	-	-	-	-
FC-FMeasure	-	-	-	-	-	-	-	-	-	-	-	-
CF-Precision	0.26	0.07	-0.32	0.02	-0.34	0.02	0.12	0.39	0.17	0.25	-0.23	0.10
CF-Recall	0.05	0.74	-0.10	0.48	-0.24	0.10	-0.08	0.58	0.13	0.36	0.04	0.78
CF-FMeasure	0.13	0.37	-0.20	0.17	-0.32	0.02	-0.01	0.96	0.18	0.21	-0.05	0.71
Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{NSGA-III}$-PART												
MLQMs	**HV**		**IGD**		**ϵ**		**ED**		**GD**		**GS**	
	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value
Accuracy	0.29	0.04	-0.30	0.04	-0.30	0.03	-0.03	0.86	0.18	0.20	-0.23	0.11
MAE	-0.30	0.03	0.31	0.03	0.30	0.03	0.05	0.73	-0.18	0.21	0.22	0.13
RMSE	-0.34	0.02	0.36	0.01	0.34	0.02	0.00	1.00	-0.23	0.11	0.24	0.10
RAE	-0.18	0.21	0.26	0.07	0.19	0.20	-0.12	0.39	-0.29	0.04	0.23	0.11
RRSE	-0.24	0.09	0.33	0.02	0.24	0.09	-0.19	0.19	-0.35	0.01	0.26	0.07
FF-Precision	0.40	0.00	-0.41	0.00	-0.40	0.00	0.03	0.82	0.27	0.06	-0.10	0.51
FF-Recall	0.35	0.01	-0.33	0.02	-0.36	0.01	0.04	0.78	0.19	0.20	-0.01	0.92
FF-FMeasure	0.39	0.00	-0.39	0.01	-0.40	0.00	0.03	0.82	0.24	0.10	-0.06	0.67
CC-Precision	-0.10	0.49	0.19	0.18	0.07	0.65	-0.30	0.03	-0.30	0.04	-0.05	0.75
CC-Recall	-0.24	0.09	0.27	0.06	0.23	0.11	-0.22	0.12	-0.26	0.06	-0.24	0.09
CC-FMeasure	-0.20	0.16	0.27	0.06	0.18	0.22	-0.28	0.05	-0.32	0.02	-0.13	0.36
FC-Precision	-	-	-	-	-	-	-	-	-	-	-	-
FC-Recall	-	-	-	-	-	-	-	-	-	-	-	-
FC-FMeasure	-	-	-	-	-	-	-	-	-	-	-	-
CF-Precision	0.05	0.75	-0.24	0.10	-0.01	0.97	0.24	0.10	0.34	0.02	-0.22	0.12
CF-Recall	0.05	0.74	-0.23	0.10	0.00	0.99	0.21	0.14	0.38	0.01	-0.20	0.16
CF-FMeasure	0.03	0.85	-0.21	0.14	0.02	0.91	0.23	0.11	0.36	0.01	-0.20	0.16

* "-" = represents that the value of a particular MLQM corresponding to all the iterations in all the cycles is equal to zero

Table 36: Correlation analysis of MLQMs with AFVs corresponding to SBRM$^{+}_{NSGA-II}$-C45, SBRM$^{+}_{NSGA-III}$-C45, SBRM$^{+}_{NSGA-II}$-PART, and SBRM$^{+}_{NSGA-III}$-PART for the Jitsi case study

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{NSGA-II}$-C45										
MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value	ρ	*p*-value
Accuracy	0.03	0.836	-0.69	**<0.05**	-0.69	**<0.05**	-0.63	**<0.05**	0.66	**<0.05**
MAE	-0.04	0.804	0.69	**<0.05**	0.70	**<0.05**	0.63	**<0.05**	-0.66	**<0.05**
RMSE	-0.04	0.760	0.68	**<0.05**	0.69	**<0.05**	0.62	**<0.05**	-0.66	**<0.05**
RAE	-0.05	0.720	0.64	**<0.05**	0.64	**<0.05**	0.58	**<0.05**	-0.61	**<0.05**
RRSE	-0.12	0.424	0.60	**<0.05**	0.60	**<0.05**	0.53	**<0.05**	-0.58	**<0.05**
FF-Precision	0.02	0.902	-0.72	**<0.05**	-0.72	**<0.05**	-0.64	**<0.05**	0.68	**<0.05**
FF-Recall	0.01	0.943	-0.67	**<0.05**	-0.66	**<0.05**	-0.61	**<0.05**	0.65	**<0.05**
FF-FMeasure	0	0.996	-0.72	**<0.05**	-0.71	**<0.05**	-0.65	**<0.05**	0.69	**<0.05**

CC-Precision	0	0.980	-0.34	**<0.05**	-0.33	**<0.05**	-0.32	**<0.05**	0.32	**<0.05**
CC-Recall	0.09	0.522	-0.34	**<0.05**	-0.32	**<0.05**	-0.29	**<0.05**	0.30	**<0.05**
CC-FMeasure	0.07	0.609	-0.34	**<0.05**	-0.33	**<0.05**	-0.30	**<0.05**	0.31	**<0.05**
FC-Precision	-0.01	0.948	0.07	0.634	0	0.982	0.04	0.81	-0.07	0.606
FC-Recall	-0.01	0.930	0.07	0.649	0.01	0.951	0.05	0.74	-0.09	0.538
FC-FMeasure	-0.01	0.932	0.07	0.605	0.02	0.902	0.05	0.71	-0.10	0.510
CF-Precision	0.08	0.585	-0.47	**<0.05**	-0.46	**<0.05**	-0.41	**<0.05**	0.44	**<0.05**
CF-Recall	0.01	0.932	-0.50	**<0.05**	-0.49	**<0.05**	-0.45	**<0.05**	0.47	**<0.05**
CF-FMeasure	0.04	0.784	-0.50	**<0.05**	-0.49	**<0.05**	-0.45	**<0.05**	0.47	**<0.05**

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-III}}$-C45

MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	-0.23	0.11	-0.42	**<0.05**	-0.42	**<0.05**	-0.44	**<0.05**	0.37	**<0.05**
MAE	0.23	0.12	0.50	**<0.05**	0.51	**<0.05**	0.52	**<0.05**	-0.45	**<0.05**
RMSE	0.29	**<0.05**	0.46	**<0.05**	0.45	**<0.05**	0.49	**<0.05**	-0.40	**<0.05**
RAE	0.43	**<0.05**	0.49	**<0.05**	0.55	**<0.05**	0.58	**<0.05**	-0.47	**<0.05**
RRSE	0.54	**<0.05**	0.37	**<0.05**	0.43	**<0.05**	0.49	**<0.05**	-0.35	**<0.05**
FF-Precision	-0.13	0.37	-0.40	**<0.05**	-0.40	**<0.05**	-0.40	**<0.05**	0.37	**<0.05**
FF-Recall	-0.12	0.39	0.02	0.88	0.04	0.79	0.01	0.92	-0.07	0.63
FF-FMeasure	-0.12	0.40	-0.23	0.10	-0.23	0.11	-0.23	0.11	0.19	0.18
CC-Precision	-0.31	**<0.05**	-0.03	0.86	-0.11	0.43	-0.13	0.38	0.06	0.66
CC-Recall	-0.27	0.05	-0.12	0.40	-0.22	0.12	-0.22	0.13	0.16	0.25
CC-FMeasure	-0.30	**<0.05**	-0.09	0.54	-0.18	0.21	-0.19	0.18	0.13	0.37
FC-Precision	-0.21	0.15	-0.33	**<0.05**	-0.39	**<0.05**	-0.40	**<0.05**	0.30	**<0.05**
FC-Recall	-0.05	0.75	-0.33	**<0.05**	-0.36	**<0.05**	-0.34	**<0.05**	0.29	**<0.05**
FC-FMeasure	-0.10	0.48	-0.34	**<0.05**	-0.38	**<0.05**	-0.37	**<0.05**	0.30	**<0.05**
CF-Precision	-0.31	**<0.05**	-0.17	0.25	-0.25	0.09	-0.25	0.08	0.13	0.35
CF-Recall	-0.31	**<0.05**	-0.25	0.08	-0.32	**<0.05**	-0.32	**<0.05**	0.24	0.10
CF-FMeasure	-0.30	**<0.05**	-0.22	0.12	-0.30	**<0.05**	-0.29	**<0.05**	0.20	0.16

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-II}}$-PART

MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	0.29	**<0.05**	-0.65	**<0.05**	-0.65	-0.650	-0.62	**<0.05**	0.66	**<0.05**
MAE	-0.30	**<0.05**	0.65	**<0.05**	0.65	**<0.05**	0.61	**<0.05**	-0.66	**<0.05**
RMSE	-0.31	**<0.05**	0.67	**<0.05**	0.67	**<0.05**	0.62	**<0.05**	-0.66	**<0.05**
RAE	-0.28	**<0.05**	0.59	**<0.05**	0.58	**<0.05**	0.57	**<0.05**	-0.60	**<0.05**
RRSE	-0.29	**<0.05**	0.61	**<0.05**	0.61	**<0.05**	0.58	**<0.05**	-0.62	**<0.05**
FF-Precision	0.32	**<0.05**	-0.69	**<0.05**	-0.67	**<0.05**	-0.63	**<0.05**	0.66	**<0.05**
FF-Recall	0.26	0.073	-0.72	**<0.05**	-0.71	**<0.05**	-0.68	**<0.05**	0.72	**<0.05**
FF-FMeasure	0.26	0.066	-0.71	**<0.05**	-0.69	**<0.05**	-0.67	**<0.05**	0.71	**<0.05**
CC-Precision	0.14	0.350	-0.04	0.788	-0.05	0.750	-0.11	0.45	0.07	0.616
CC-Recall	0.14	0.336	0	0.985	-0.03	0.827	-0.09	0.53	0.06	0.687
CC-FMeasure	0.15	0.291	0	0.986	-0.02	0.870	-0.08	0.57	0.05	0.732
FC-Precision	0.30	**<0.05**	-0.36	**<0.05**	-0.39	**<0.05**	-0.36	**<0.05**	0.40	**<0.05**
FC-Recall	0.28	**<0.05**	-0.40	**<0.05**	-0.35	**<0.05**	-0.33	**<0.05**	0.37	**<0.05**
FC-FMeasure	0.34	**<0.05**	-0.42	**<0.05**	-0.40	**<0.05**	-0.34	**<0.05**	0.40	**<0.05**
CF-Precision	0.08	0.572	-0.15	0.306	-0.19	0.177	-0.18	0.22	0.23	0.109
CF-Recall	0.20	0.165	-0.06	0.678	-0.09	0.540	-0.09	0.54	0.13	0.362
CF-FMeasure	0.18	0.221	-0.11	0.462	-0.15	0.306	-0.13	0.36	0.18	0.207

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-III}}$-PART

MLQMs	AFV-O1		AFV-O2		AFV-O3		OAFV		HV	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	0.25	0.08	-0.33	**<0.05**	-0.40	**<0.05**	-0.27	0.06	0.43	**<0.05**
MAE	-0.24	0.10	0.32	**<0.05**	0.39	**<0.05**	0.27	0.06	-0.43	**<0.05**
RMSE	-0.24	0.10	0.32	**<0.05**	0.40	**<0.05**	0.28	**<0.05**	-0.44	**<0.05**
RAE	-0.36	**<0.05**	0.46	**<0.05**	0.60	**<0.05**	0.47	**<0.05**	-0.57	**<0.05**
RRSE	-0.37	**<0.05**	0.45	**<0.05**	0.61	**<0.05**	0.49	**<0.05**	-0.56	**<0.05**
FF-Precision	0.24	0.09	-0.34	**<0.05**	-0.42	**<0.05**	-0.34	**<0.05**	0.46	**<0.05**
FF-Recall	0.24	0.10	-0.33	**<0.05**	-0.40	**<0.05**	-0.31	**<0.05**	0.43	**<0.05**
FF-FMeasure	0.26	0.06	-0.36	**<0.05**	-0.44	**<0.05**	-0.34	**<0.05**	0.47	**<0.05**

CC-Precision	0.13	0.38	-0.11	0.46	-0.28	0.05	-0.29	**<0.05**	0.16	0.28
CC-Recall	0.06	0.67	-0.02	0.89	-0.20	0.15	-0.23	0.11	0.08	0.56
CC-FMeasure	0.11	0.46	-0.08	0.59	-0.26	0.07	-0.27	0.06	0.13	0.35
FC-Precision	0.29	**<0.05**	-0.26	0.07	-0.22	0.13	-0.08	0.58	0.24	0.10
FC-Recall	0.22	0.12	-0.25	0.08	-0.23	0.11	-0.16	0.28	0.30	**<0.05**
FC-FMeasure	0.27	0.06	-0.27	0.06	-0.23	0.11	-0.12	0.40	0.29	**<0.05**
CF-Precision	0.01	0.96	-0.15	0.30	-0.11	0.46	-0.10	0.49	0.18	0.21
CF-Recall	-0.01	0.95	-0.10	0.51	0.00	0.98	0.01	0.93	0.10	0.47
CF-FMeasure	-0.01	0.97	-0.11	0.44	-0.03	0.83	-0.03	0.86	0.13	0.37

Table 37: Correlation analysis of MLQMs with six indicators corresponding to $SBRM^{+}_{NSGA-II}$-C45, $SBRM^{+}_{NSGA-III}$-C45, $SBRM^{+}_{NSGA-II}$-PART, and $SBRM^{+}_{NSGA-III}$-PART for the Jitsi case study*

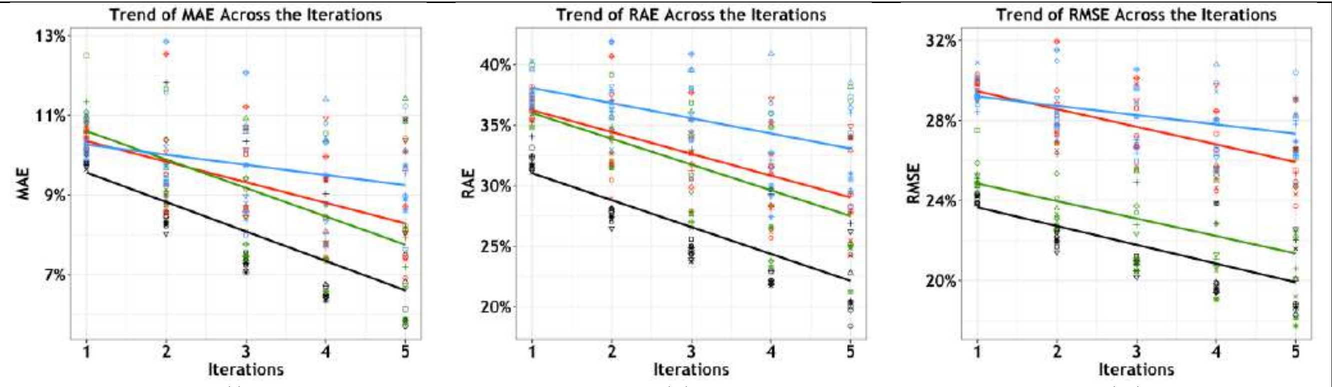
Correlation analysis of MLQMs with AFVs for $SBRM^{+}_{NSGA-II}$-C45												
MLQMs	**HV**		**IGD**		**ϵ**		**ED**		**GD**		**GS**	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	0.51	0.00	-0.64	0.00	-0.39	0.01	0.18	0.22	0.36	0.01	-0.42	0.00
MAE	-0.50	0.00	0.64	0.00	0.38	0.01	-0.18	0.22	-0.35	0.01	0.43	0.00
RMSE	-0.49	0.00	0.63	0.00	0.38	0.01	-0.18	0.21	-0.35	0.01	0.42	0.00
RAE	-0.48	0.00	0.59	0.00	0.38	0.01	-0.22	0.12	-0.39	0.01	0.49	0.00
RRSE	-0.42	0.00	0.54	0.00	0.32	0.02	-0.22	0.13	-0.38	0.01	0.47	0.00
FF-Precision	0.52	0.00	-0.67	0.00	-0.40	0.00	0.15	0.29	0.37	0.01	-0.38	0.01
FF-Recall	0.47	0.00	-0.63	0.00	-0.35	0.01	0.11	0.46	0.34	0.02	-0.30	0.03
FF-FMeasure	0.53	0.00	-0.68	0.00	-0.41	0.00	0.16	0.28	0.37	0.01	-0.38	0.01
CC-Precision	0.32	0.02	-0.31	0.03	-0.23	0.10	0.21	0.14	0.26	0.07	-0.24	0.09
CC-Recall	0.27	0.06	-0.29	0.04	-0.18	0.21	0.26	0.07	0.25	0.09	-0.25	0.08
CC-FMeasure	0.29	0.04	-0.30	0.03	-0.19	0.18	0.23	0.11	0.26	0.07	-0.24	0.09
FC-Precision	-0.03	0.84	0.07	0.63	0.09	0.52	-0.16	0.26	-0.13	0.37	-0.15	0.30
FC-Recall	-0.05	0.74	0.08	0.60	0.09	0.53	-0.26	0.07	-0.19	0.18	-0.08	0.59
FC-FMeasure	-0.04	0.76	0.09	0.55	0.10	0.48	-0.22	0.13	-0.16	0.25	-0.13	0.38
CF-Precision	0.35	0.01	-0.42	0.00	-0.29	0.04	0.20	0.16	0.41	0.00	-0.51	0.00
CF-Recall	0.40	0.00	-0.46	0.00	-0.35	0.01	0.14	0.34	0.40	0.00	-0.51	0.00
CF-FMeasure	0.39	0.01	-0.45	0.00	-0.33	0.02	0.15	0.31	0.41	0.00	-0.50	0.00
Correlation analysis of MLQMs with AFVs for $SBRM^{+}_{NSGA-III}$-C45												
MLQMs	**HV**		**IGD**		**ϵ**		**ED**		**GD**		**GS**	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	0.40	0.00	-0.63	0.00	-0.08	0.58	0.07	0.61	0.31	0.03	-0.21	0.14
MAE	-0.39	0.00	0.64	0.00	0.07	0.65	-0.08	0.58	-0.31	0.03	0.23	0.10
RMSE	-0.40	0.00	0.65	0.00	0.06	0.66	-0.09	0.53	-0.29	0.04	0.23	0.11
RAE	-0.39	0.01	0.59	0.00	0.10	0.51	-0.13	0.35	-0.39	0.01	0.24	0.10
RRSE	-0.41	0.00	0.61	0.00	0.11	0.46	-0.16	0.27	-0.35	0.01	0.22	0.12
FF-Precision	0.38	0.01	-0.64	0.00	-0.06	0.70	0.05	0.74	0.29	0.04	-0.19	0.18
FF-Recall	0.43	0.00	-0.70	0.00	-0.09	0.52	0.00	0.99	0.26	0.07	-0.13	0.36
FF-FMeasure	0.43	0.00	-0.68	0.00	-0.10	0.48	0.00	0.97	0.28	0.05	-0.15	0.31
CC-Precision	0.09	0.52	-0.09	0.53	0.00	0.98	0.35	0.01	0.19	0.18	-0.10	0.48
CC-Recall	0.12	0.43	-0.07	0.62	-0.03	0.84	0.39	0.01	0.21	0.14	-0.16	0.27
CC-FMeasure	0.09	0.54	-0.06	0.66	0.00	1.00	0.39	0.00	0.22	0.12	-0.15	0.31
FC-Precision	0.26	0.07	-0.38	0.01	-0.05	0.72	0.15	0.30	0.36	0.01	-0.29	0.04
FC-Recall	0.26	0.07	-0.35	0.01	-0.07	0.63	0.08	0.56	0.35	0.01	-0.23	0.11
FC-FMeasure	0.26	0.07	-0.37	0.01	-0.04	0.79	0.13	0.37	0.38	0.01	-0.28	0.05
CF-Precision	0.12	0.40	-0.17	0.25	-0.08	0.58	0.03	0.86	0.35	0.01	0.01	0.96
CF-Recall	0.00	0.99	-0.06	0.68	0.04	0.77	0.18	0.21	0.44	0.00	-0.11	0.45
CF-FMeasure	0.05	0.75	-0.11	0.45	0.00	1.00	0.13	0.39	0.42	0.00	-0.07	0.62
Correlation analysis of MLQMs with AFVs for $SBRM^{+}_{NSGA-II}$-PART												
MLQMs	**HV**		**IGD**		**ϵ**		**ED**		**GD**		**GS**	**HV**
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value

Accuracy	0.44	0.00	-0.44	0.00	-0.14	0.34	0.01	0.93	-0.38	0.01	-0.08	0.57
MAE	-0.50	0.00	0.52	0.00	0.12	0.41	-0.01	0.97	0.46	0.00	0.04	0.81
RMSE	-0.50	0.00	0.49	0.00	0.19	0.18	-0.02	0.92	0.39	0.00	-0.01	0.96
RAE	-0.68	0.00	0.65	0.00	0.28	0.05	-0.28	0.05	0.49	0.00	0.04	0.79
RRSE	-0.64	0.00	0.56	0.00	0.39	0.01	-0.35	0.01	0.33	0.02	0.09	0.53
FF-Precision	0.36	0.01	-0.37	0.01	-0.05	0.75	-0.02	0.91	-0.40	0.00	-0.04	0.77
FF-Recall	0.05	0.72	0.01	0.97	-0.12	0.41	-0.04	0.78	0.07	0.62	-0.28	0.05
FF-FMeasure	0.23	0.10	-0.22	0.13	-0.06	0.69	-0.01	0.92	-0.22	0.12	-0.17	0.24
CC-Precision	0.25	0.08	-0.22	0.12	-0.23	0.11	0.32	0.02	-0.06	0.69	-0.19	0.18
CC-Recall	0.33	0.02	-0.31	0.03	-0.20	0.17	0.32	0.02	-0.16	0.28	-0.19	0.19
CC-FMeasure	0.30	0.03	-0.28	0.05	-0.22	0.13	0.34	0.02	-0.12	0.40	-0.19	0.20
FC-Precision	0.45	0.00	-0.45	0.00	-0.14	0.32	0.21	0.14	-0.34	0.01	0.11	0.45
FC-Recall	0.33	0.02	-0.36	0.01	-0.01	0.96	0.18	0.21	-0.36	0.01	0.07	0.63
FC-FMeasure	0.39	0.01	-0.41	0.00	-0.05	0.71	0.18	0.20	-0.36	0.01	0.07	0.63
CF-Precision	0.46	0.00	-0.37	0.01	-0.29	0.04	0.37	0.01	-0.21	0.15	-0.13	0.38
CF-Recall	0.49	0.00	-0.41	0.00	-0.25	0.08	0.24	0.09	-0.29	0.04	-0.03	0.83
CF-FMeasure	0.47	0.00	-0.39	0.00	-0.25	0.08	0.29	0.04	-0.27	0.06	-0.06	0.69

Correlation analysis of MLQMs with AFVs for SBRM$^{+}_{\text{NSGA-III}}$-PART

MLQMs	HV		IGD		ϵ		ED		GD		GS	
	ρ	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value	*ρ*	*p*-value
Accuracy	-0.02	0.89	-0.23	0.11	0.20	0.16	0.31	0.03	0.47	0.00	0.18	0.22
MAE	0.01	0.94	0.23	0.11	-0.19	0.18	-0.30	0.03	-0.46	0.00	-0.18	0.21
RMSE	0.00	0.98	0.24	0.09	-0.18	0.21	-0.31	0.03	-0.47	0.00	-0.19	0.19
RAE	0.02	0.87	0.49	0.00	-0.25	0.08	-0.10	0.50	-0.30	0.04	-0.38	0.01
RRSE	0.00	0.99	0.53	0.00	-0.23	0.11	-0.05	0.71	-0.26	0.06	-0.39	0.00
FF-Precision	0.05	0.75	-0.29	0.04	0.15	0.29	0.29	0.04	0.45	0.00	0.27	0.06
FF-Recall	0.02	0.86	-0.24	0.09	0.18	0.22	0.35	0.01	0.53	0.00	0.24	0.10
FF-FMeasure	0.03	0.86	-0.29	0.04	0.18	0.21	0.30	0.04	0.47	0.00	0.25	0.08
CC-Precision	0.17	0.24	-0.31	0.03	-0.01	0.92	-0.15	0.30	-0.05	0.71	0.40	0.00
CC-Recall	0.18	0.20	-0.24	0.09	-0.08	0.58	-0.16	0.26	-0.07	0.62	0.40	0.00
CC-FMeasure	0.17	0.23	-0.29	0.04	-0.04	0.80	-0.16	0.26	-0.07	0.63	0.41	0.00
FC-Precision	-0.24	0.09	-0.15	0.29	0.29	0.04	-0.03	0.86	0.03	0.85	-0.11	0.46
FC-Recall	-0.14	0.34	-0.17	0.23	0.20	0.15	-0.03	0.85	0.06	0.66	-0.08	0.58
FC-FMeasure	-0.21	0.15	-0.17	0.25	0.25	0.08	-0.04	0.77	0.04	0.81	-0.10	0.47
CF-Precision	-0.10	0.47	-0.02	0.90	0.10	0.48	0.16	0.26	0.21	0.14	-0.04	0.81
CF-Recall	-0.15	0.30	0.07	0.64	0.09	0.52	0.12	0.42	0.11	0.45	-0.12	0.39
CF-FMeasure	-0.13	0.36	0.04	0.77	0.10	0.50	0.15	0.31	0.15	0.30	-0.09	0.53

Detailed Results of RQ7



(I) (II) (III)

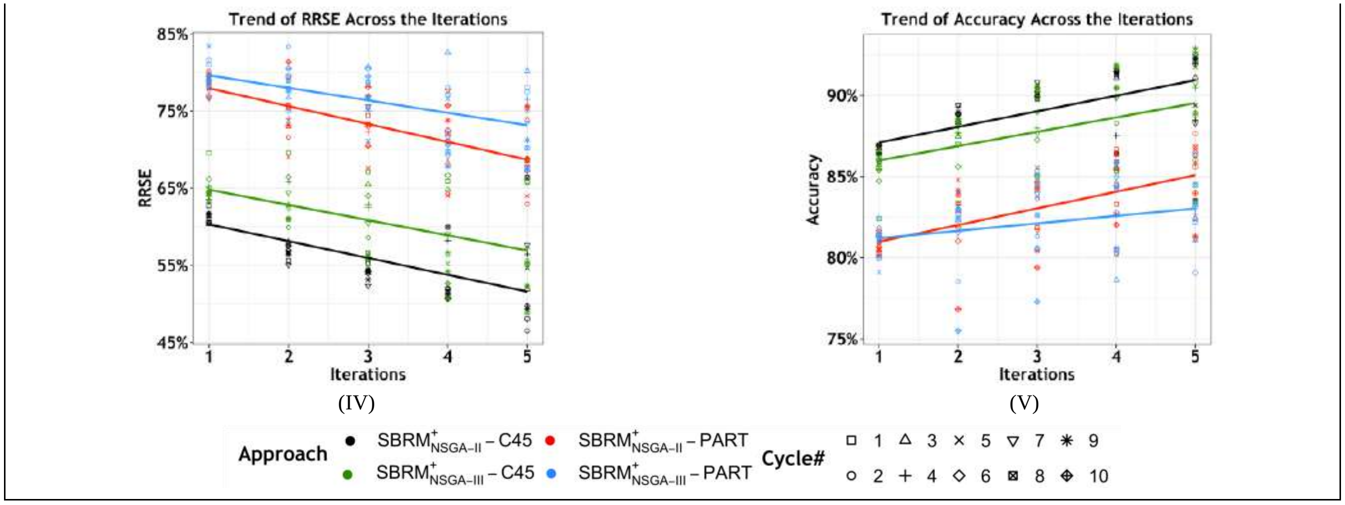


Figure 16: Trend of Accuracy, MAE, RAE, RMSE, and RRSE across iteration for the Cisco case study

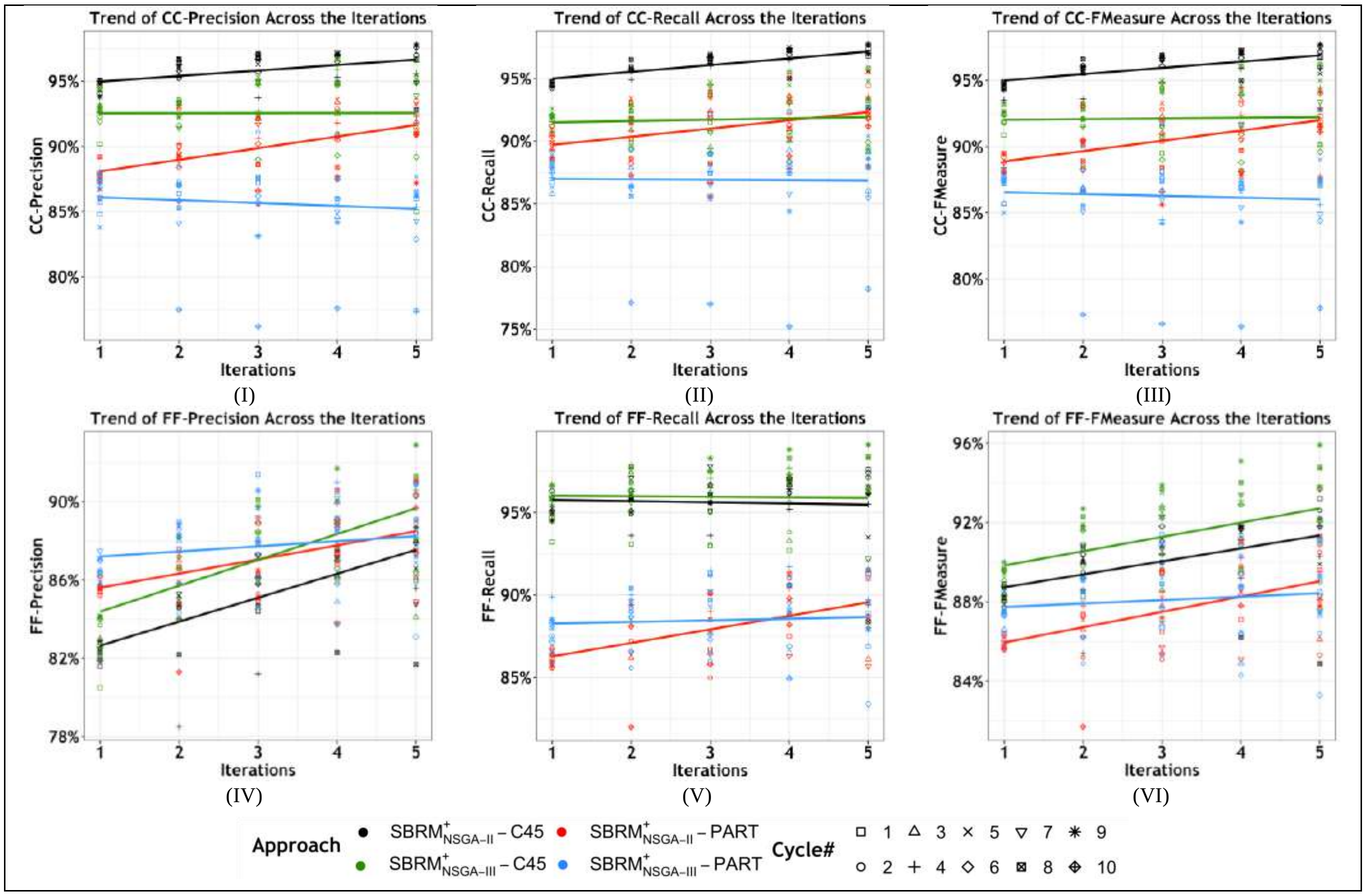


Figure 17: Trend of CC-Precision, CC-Recall, CC-FMeasure, FF-Precision, FF-Recall, and FF-FMeasure across iterations for the Cisco case study

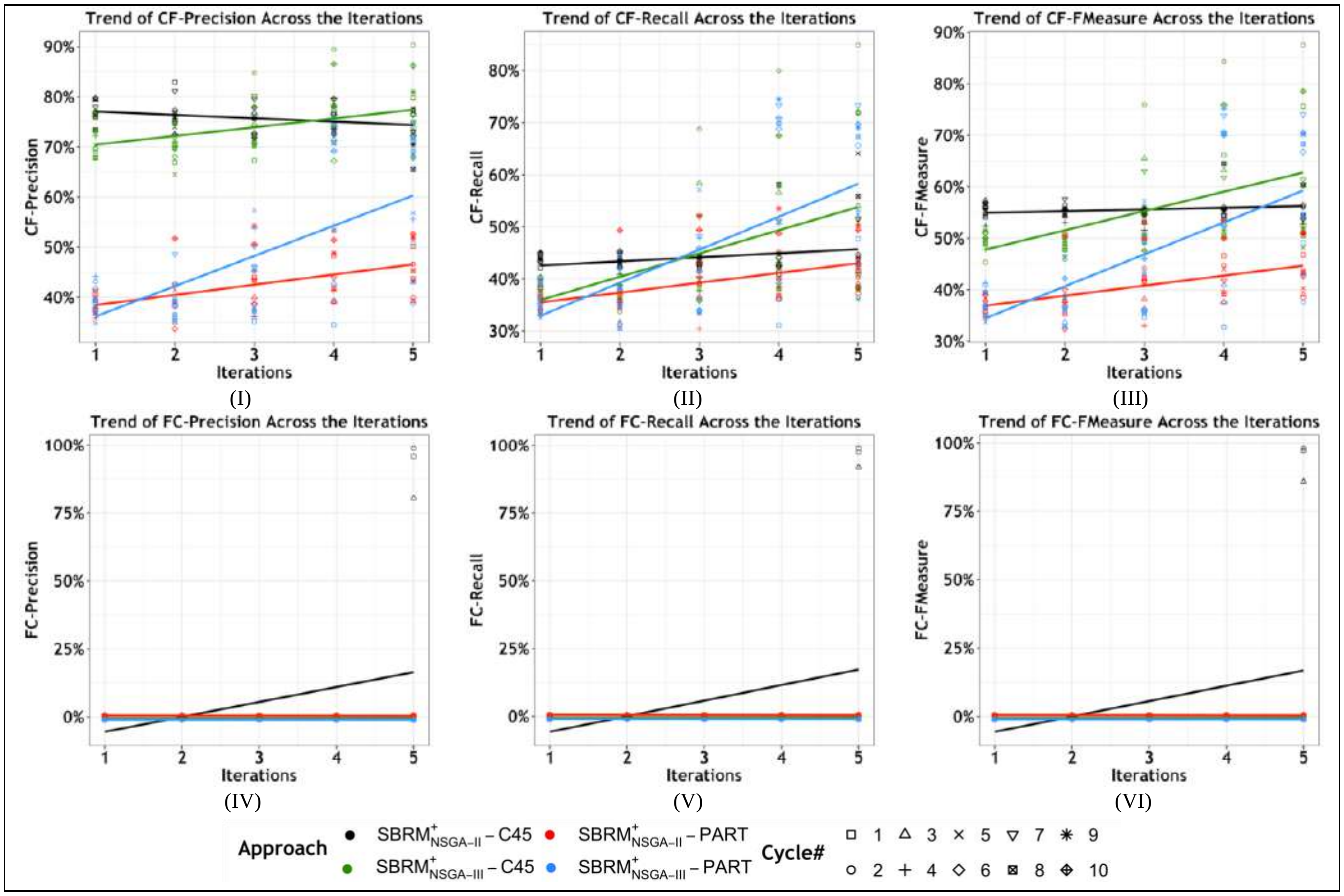
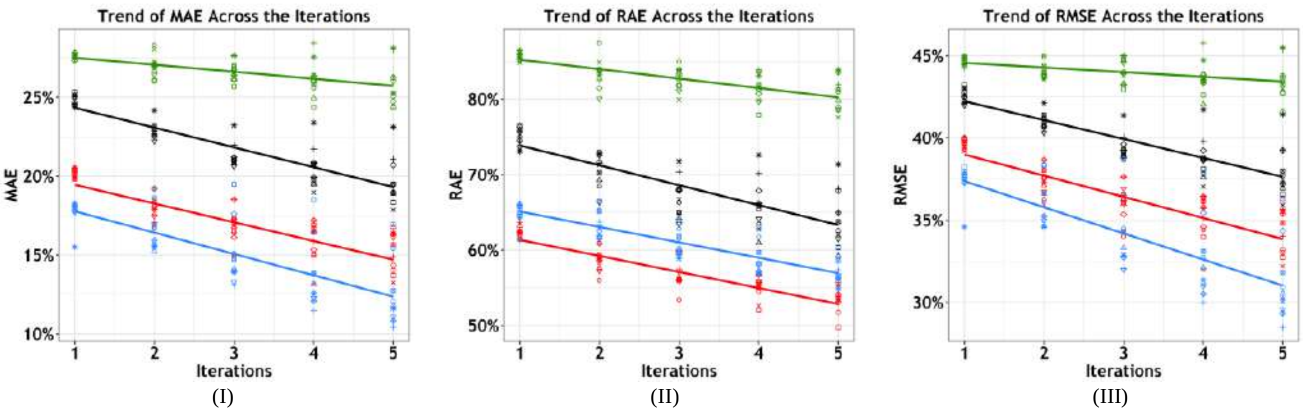


Figure 18: Trend of CF-Precision, CF-Recall, CF-FMeasure, FC-Precision, FC-Recall, and FC-FMeasure across iterations for the Cisco case study



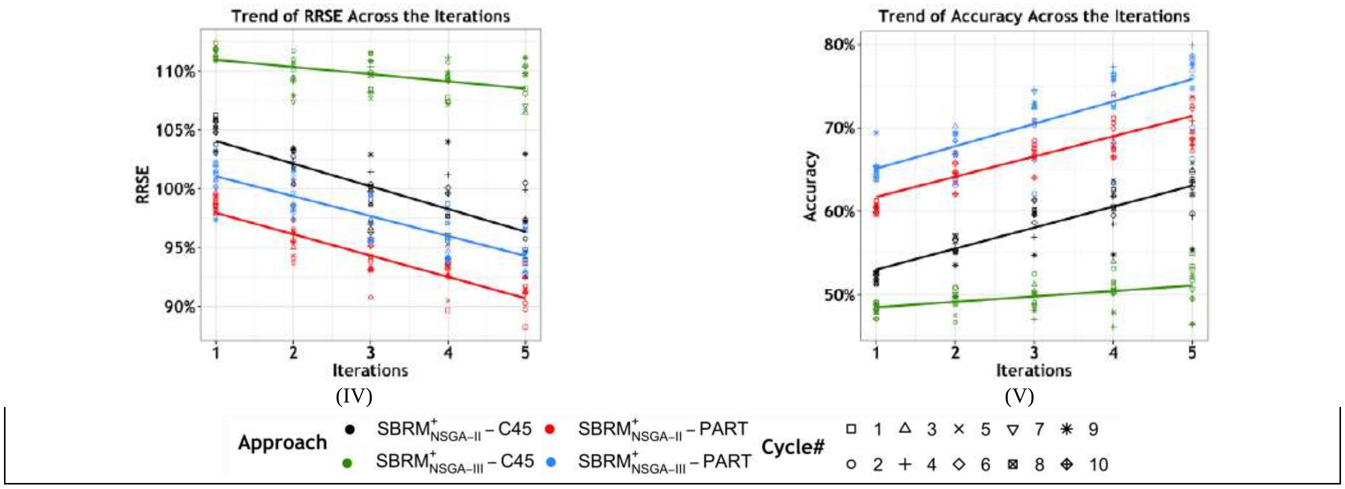


Figure 19: Trend of Accuracy, MAE, RAE, RMSE, and RRSE across iteration for the Jitsi case study

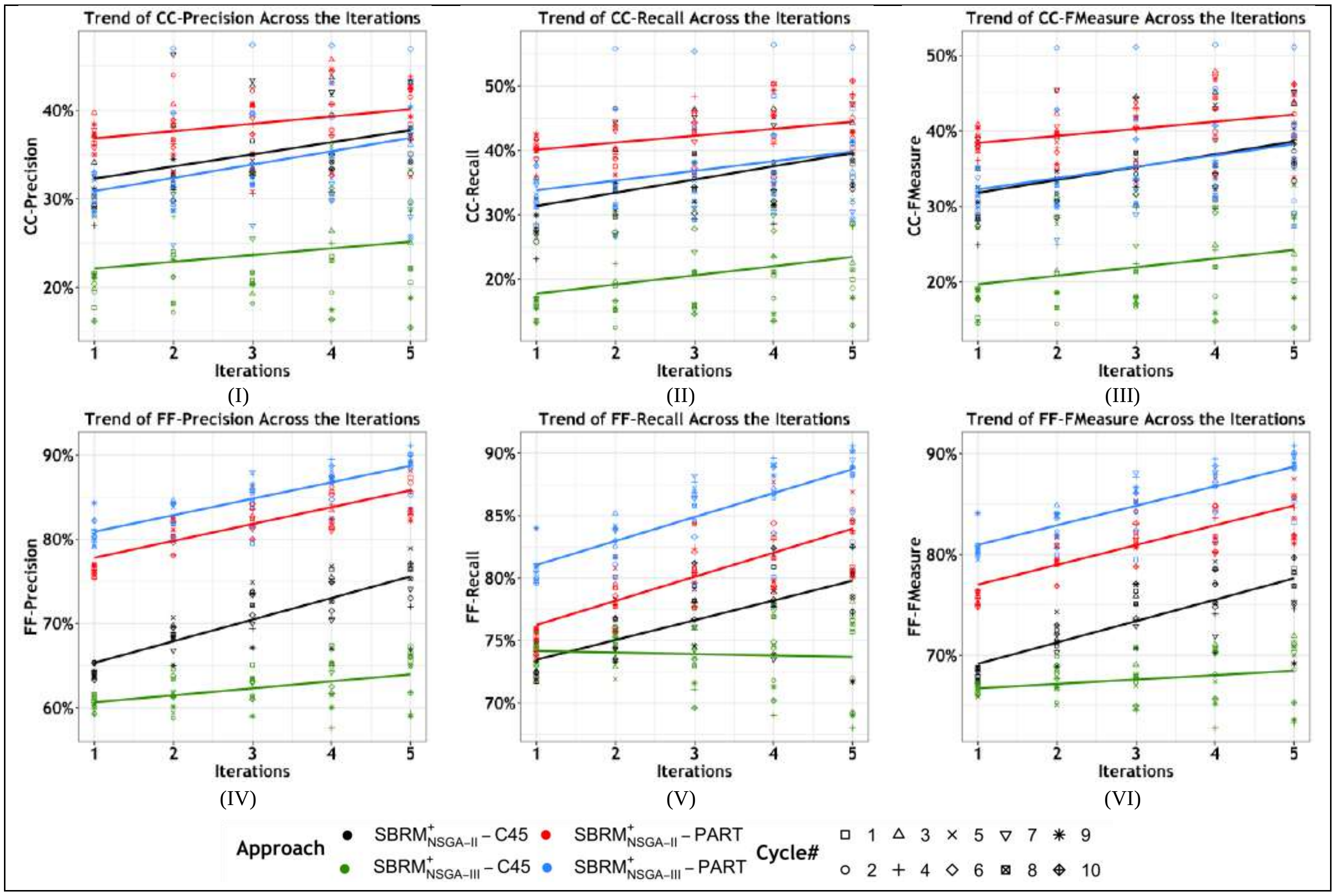


Figure 20: Trend of CC-Precision, CC-Recall, CC-FMeasure, FF-Precision, FF-Recall, and FF-FMeasure across iterations for the Jitsi case study

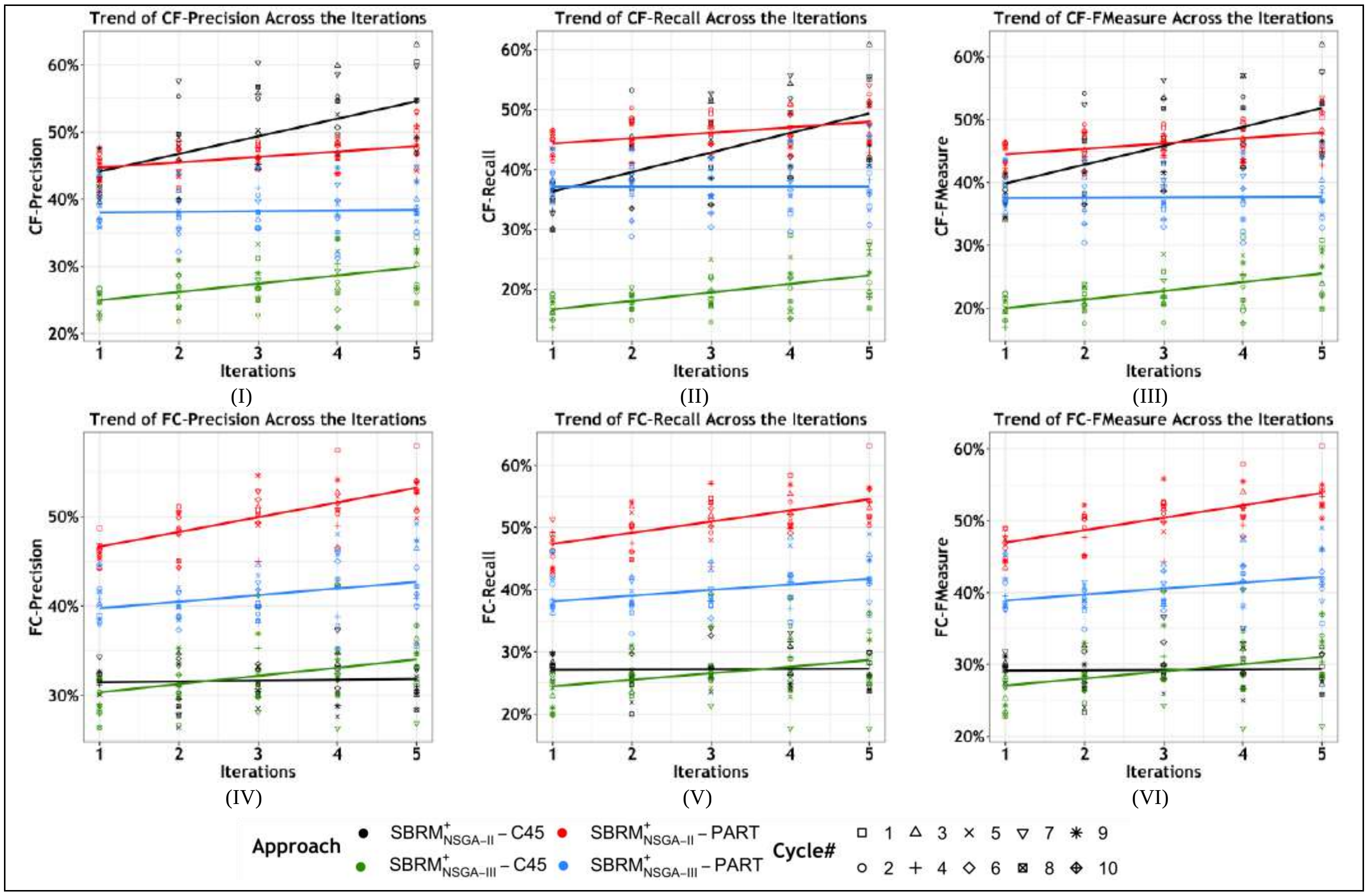


Figure 21: Trend of CF-Precision, CF-Recall, CF-FMeasure, FC-Precision, FC-Recall, and FC-FMeasure across iterations for the Jitsi case study