Cost-effective regression testing through Adaptive Test Prioritization strategies

Amanda Schwartz\textsuperscript{a},*, Hyunsook Do\textsuperscript{b}

\textsuperscript{a} University of South Carolina Upstate, Spartanburg, SC 29323, United States
\textsuperscript{b} University of North Texas, Denton, TX 76203, United States

\textbf{ARTICLE INFO}

Article history:
Received 20 January 2015
Revised 26 October 2015
Accepted 11 January 2016
Available online 27 January 2016

Keywords:
Regression testing
Adaptive regression testing strategy
Test case prioritization

\textbf{ABSTRACT}

Regression testing is an important part of the software development life cycle. It is also very expensive. Many different techniques have been proposed for reducing the cost of regression testing. However, research has shown that the effectiveness of different techniques varies under different testing environments and software change characteristics. In prior work, we developed strategies to investigate ways of choosing the most cost-effective regression testing technique for a particular regression testing session. In this work, we empirically study the existing strategies presented in prior work as well as develop two additional Adaptive Test Prioritization (ATP) strategies using fuzzy analytical hierarchy process (AHP) and the weighted sum model (WSM). We also provide a comparative study examining each of the ATP strategies presented to date. This research will provide researchers and practitioners with strategies to utilize in regression testing plans as well as provide data to use when deciding which of the strategies would best fit their testing needs. The empirical studies provided in this research show that utilizing these strategies can improve the cost-effectiveness of regression testing.

© 2016 Elsevier Inc. All rights reserved.

1. Introduction

As software systems evolve, it is important to test changes made to maintain the quality of modified software systems. Regression testing is the process of testing modified software systems. Regression testing is important, but it also can be very expensive. To date, many regression testing techniques have been proposed (i.e. regression test selection, test case prioritization, and test case minimization) to reduce the cost of regression testing (Engstrom et al., 2010; Yoo and Harman, 2010).

Many empirical studies have been conducted to evaluate the regression testing techniques that have been proposed. Recent studies (Do et al., 2010; Qu et al., 2008; Walcott et al., 2006) have shown that environmental and testing factors affect the cost-effectiveness of regression testing techniques. Further, the studies show that the cost–benefits differ based on the particular software release and that different techniques are most cost-effective in different regression testing sessions. The technique that is most cost-effective for one version may not be the most cost-effective for every version of a software system. Therefore, there is no single regression testing technique that is most cost-effective for every version of a system.

To address this problem, in our previous work (Arafeen and Do, 2011), we proposed one strategy to help identify the most cost-effective regression testing technique for each regression testing session. This work proposed and empirically studied the analytical hierarchy process (AHP) Saaty (1980) as one Adaptive Test Prioritization (ATP) strategy.\textsuperscript{1} The results indicated that the prioritization techniques selected by the AHP method can be more cost-effective than those that do not consider system lifetime and testing processes.

Although this study showed promising results, there are many advances that can be made in ATP. First, only one ATP strategy was considered in this study. New strategies can be developed and evaluated that have the potential to be even more cost-effective.

Second, although AHP has been widely used across many different fields, there are some disadvantages to the method. The AHP method is frequently criticized for being subjective to the judgments made by the human decision makers (Shen et al., 2011; Sun, 2010; Wu et al., 2011). Thus, the results can be inaccurate if the decision makers are inexperienced or if they lack knowledge about

\textsuperscript{1} In prior work, we called the strategies adaptive regression testing (ART) strategies but have renamed them Adaptive Test Prioritization (ATP) strategies in this work.
the application domain. Further, the study only used one decision maker, so the results are dependent upon the judgments made by one individual. Another weakness of the AHP method is that the comparisons made by the decision maker during the pairwise comparison process are often inconsistent (Benitez et al., 2011; Lin and Wang, 2012). Judgments made in one comparison often contradict judgments made in another comparison. In addition, the AHP method is very time consuming for the decision makers. Empirical studies have shown that decision makers prefer other methods because of the time required by the pairwise comparisons (Ahl, 2005; Hatton, 2007). Further, pairwise comparisons are not scalable. The work required by the pairwise comparisons limits the number of criteria and alternatives that can be considered (Saaty and Ozdemir, 2003). To address these problems, other strategies need to be developed.

This paper makes the following contributions:

- A new fuzzy AHP strategy is presented in Section 3.2 that utilizes fuzzy logic to address the imprecision in decision makers’ judgments in the AHP method.
- An empirical study investigating the fuzzy AHP and AHP strategies is presented in Section 4.5.
- A fuzzy expert system (FESART) is developed to reduce the time needed by decision makers. FESART is empirically studied in Section 4.6.
- A new strategy utilizing the WSM is presented in Section 3.4 and studied in Section 4.7.
- A comparative study of each of the strategies is performed and presented in Section 4.7.

The rest of the paper is organized as follows. In the next section, we provide background information and describe related work relevant to prioritization techniques and decision making strategies. The new strategies are described in Section 3, and the empirical studies are presented in Section 4. Finally, conclusions and future work are outlined in Section 5.

2. Background and related work

This section provides background information and related work relevant to regression testing techniques and decision making strategies. Specifically, we focus on work related to prioritization techniques and empirical studies conducted to evaluate regression testing techniques; the discussion of decision making strategies is limited to the methods that are directly related to this work.

2.1. Test case prioritization techniques

Test case prioritization techniques use various types of information (e.g., code coverage or code change information) to find the ideal ordering of test cases so that maximum benefit can be achieved even if testing is halted early. For example, one technique, total block coverage prioritization, simply sorts the test cases by the order of the number of blocks they cover. One variation of this technique, additional block coverage prioritization, iteratively selects a test case that yields the greatest block coverage, adjusts the coverage information for the remaining test cases to indicate their coverage for the blocks not yet covered, and then repeats this process until all blocks are covered by at least one test case.

The idea of reordering test cases was first mentioned by Wong et al. (1995). In their work, the test cases reordered were already selected by a test case selection technique. The work did note that even with a significant reduction in test size, different orders of test cases produced greater fault detection than larger test suites. The concept of test case prioritization as its own regression testing technique was more formally defined by Rothermel et al. (1999).

Since then, many test case prioritization techniques have been created and empirically studied, and a recent survey by Yoo and Harman (2010) provides an overview of these techniques. Further, several software organizations are now utilizing them in practice (Harrold and Orso, 2008; Srivastava and Thiagarajan, 2002).

While the goal of the proposed techniques is to improve the effectiveness of regression testing, to be useful in practice, techniques should be applicable within various testing environments and contexts. Recent research on test case prioritization has employed empirical studies to evaluate the cost–benefit tradeoffs among techniques by considering various factors and testing contexts (Do et al., 2010; Elbaum et al., 2002; Malishevsky et al., 2002; Qu et al., 2008; Walcott et al., 2006). These studies show that various techniques could be cost-effective and suggest tradeoffs among them. However, the studies also reveal wide variances in performance and attribute the differences to many different factors (e.g., the program under test, the test suites used to test them, the types of modifications made to the programs, and the testing processes).

2.2. Decision making strategies

Having many different prioritization techniques makes it a complex problem to decide which technique is going to be the most cost-effective technique for a specific regression testing session. Many different factors make one technique better than another. A problem that has multiple conflicting criteria is known as a multiple criteria decision making (MCDM) problem. Many different methods have been proposed and studied to solve MCDM problems. Here, we will limit our discussion to those that were used in our work.

2.2.1. AHP

One of the most widely used MCDM methods is the analytic hierarchy process (AHP) (Saaty, 1980). AHP has been used in many different areas. For instance, Kamal and Al-Harbi (2001) use AHP in project management to determine a contractor’s competence or ability to participate in a project bid. AHP has also been used to analyze and assess risks for a construction project (Mustafa and Al-Bahar, 1991), and to select the best maintenance strategy for an important oil refinery (Bevilacqua and Braglia, 2000).

More closely related to our work, AHP has also been used in the area of software engineering. Ahmad and Laplante (2006) use AHP to help select a software project management tool; Sadig et al. (2009) elicit and prioritize software requirements using AHP; and Zhang et al. (2012) use AHP to aid in early effort estimation of a project. Karlsson and Wohlin (1998) and Perini and Ricca (2009) compare AHP with other alternative methods in prioritizing software requirements. Yoo et al. (2009) use AHP to improve test case prioritization techniques by employing expert knowledge and by comparing the proposed approach with the conventional coverage-based test case prioritization technique. Arafeen and Do (2011) use AHP as an ATP strategy to select the most cost-effective regression testing strategy for a particular software version.

2.2.2. Fuzzy AHP

Although AHP has been shown to be useful in many different areas, there is one big drawback of AHP: it involves human judgment. Human judgment is subjective and imprecise. For this reason, it has often been suggested to apply fuzzy logic to the AHP process to manage the imprecision of the judgments made by decision makers (Lee, 2010; Shen et al., 2011).

Many fuzzy AHP methods have been proposed by various researchers. Early work on fuzzy AHP was done by Van Laarhoven and Pedrycz (1983), in which decision makers express their opinions in fuzzy numbers using triangular membership functions, and the mathematical model includes the logarithmic least squared
method. Buckley (1985) proposes a method using trapezoidal membership functions. Chang (1996) introduces a new approach using triangular fuzzy numbers (TFN) and the extent analysis method. Pan (2008) proposes a method that combines the use of triangular and trapezoidal fuzzy numbers. Of all of the methods proposed, Chang’s extent analysis is, by far, the most commonly used and suggested method to handle the inaccuracies in a decision maker’s judgments.

2.2.3. Fuzzy expert system

Fuzzy expert systems have been used in many different domains to aid in complex decision making problems. For example, fuzzy expert systems have been developed in the medical field to diagnose heart disease (Adeli and Neshat, 2010) and back pain (Kadhim et al., 2011), and in economics for choosing stocks in the stock exchange (Fasanghari and Montazer, 2010). Fuzzy expert systems have also been developed in the area of software engineering. They have been used frequently for software cost estimation (Kadhim et al., 2011; Kazemifard et al., 2011). There has been very little use in the area of software testing, however. Xu et al. developed a fuzzy expert system to build a new test selection technique (Xu et al., 2005). Our work develops a fuzzy expert system that helps choose the most cost-effective regression testing technique for regression testing sessions.

2.2.4. WSM

WSM (Fishburn, 1967) is one of the earliest and simplest MCDM methods developed, but it is still widely used in many widely different areas. In fact, some argue (Khorasani et al., 2012; Triantaphyllou, 2000; Zavadskas et al., 2012) that it is still one of the most popular and well known methods today. For example, just one area where it has recently been used is in the medical field in public health assessments (Su et al., 2013) and aiding with the scheduling of physicians (Gunawan and Lau, 2012). More closely related to this work, WSM has also been recently used in software engineering. A couple of examples of how it has been used is to assess risks in software maintenance (Abdelmoez et al., 2012) and instantiate a variability model in requirements engineering (Thurimella and Ramaswamy, 2012).

The main reason why the weighted sum model is popular is because of its simplicity. If WSM is used in an ATP strategy, its simplicity could provide a very low-cost strategy. For this reason, this research presents a new ATP strategy utilizing WSM to investigate the impact of using a simple, low-cost decision making method in an ATP strategy on the cost–benefit calculations for the strategy. The cost–benefit results of the ATP strategy utilizing WSM are compared with the results of the other ATP strategies presented in this work in Section 4.7.

3. Adaptive Test Prioritization (ATP) strategy

As we introduced in Section 1, the performance of regression testing techniques can vary across different software programs and even between versions of the same software system. This causes a problem when trying to decide which technique to choose for a particular software version. ATP strategies provide a method to attempt to choose the most cost-effective technique for a particular software version based on input from decision makers regarding characteristics, such as cost criteria, related to the particular software version under test. For example, consider a company that is preparing to release a new version of a large software system. For the previous releases, test engineers chose one particular testing technique and they have noticed with the last couple of releases that the time required by that technique has become very long. In order to reduce the time required (and therefore the cost), the company may want to consider other techniques to test the software system for their current release. In order to identify which technique would be most cost-effective for the particular release, they could use one of the ATP strategies presented in this work. At the end of each subsection, we use this example to explain how the strategy is applied.

Section 2 introduced four different decision making methods we use in this work to build ATP strategies. First, the AHP method is investigated to study how one MCDM strategy performs in regard to ATP. Second, we investigate the fuzzy AHP method in order to utilize fuzzy logic to reduce the imprecision in the judgments made by the decision maker. Third, we investigate a fuzzy expert system to examine the effects of a low cost strategy has on the final cost–benefit analysis. Fourth, we investigate the weighted sum model (WSM) in a comparative study to further understand how a very simple, low cost strategy can perform in regard to the other methods mentioned previously.

In this section, we present and explain the four ATP strategies used in this research.

3.1. AHP ATP strategy

The AHP strategy begins by building an AHP hierarchy. An AHP hierarchy consists of a goal the decision maker wishes to achieve, possible alternatives the decision maker is considering to reach that goal, and the evaluation criteria the decision maker will use to evaluate the alternatives. These three items are structured into an AHP hierarchy. Fig. 1a shows how the AHP hierarchy is structured. The goal is placed at the top, the middle tier consists of the criteria used to evaluate the alternatives, and the alternatives being considered are placed at the bottom of the hierarchy. For ATP, the goal is to choose the most cost-effective regression testing technique for a particular software version. The criteria consist of factors that affect the cost-effectiveness of the regression testing techniques being considered, and the alternatives are the regression testing techniques being considered.

After the AHP hierarchy has been established, a set of pairwise comparisons are performed. The pairwise comparisons are conducted between the evaluation criteria and between the alternatives in regards to the evaluation criteria. When conducting the pairwise comparisons for AHP, decision makers use a popular scale from 1 to 9, with the values represented in Table 1. For example, if the decision maker feels that one evaluation criterion (C1) is strongly more important than a second evaluation criterion (C2), C1 would receive a rating of 5 (and C2 would receive a rating of 1/5). Using the pairwise comparisons, calculations are performed to obtain global priorities for each of the alternatives. First a local priority is calculated using the following equation:

$$LP_i = \frac{\sum_{j=1}^{N} (RW_{ij})}{\sum_{i=1}^{M} \sum_{j=1}^{N} (RW_{ij})}$$

where $LP_i$ is a local priority of criterion $i$, $RW_{ij}$ is a relative weight of criterion $i$ over criterion $j$, and $N$ is the number of criteria (the local priorities of alternatives are calculated in the same way).

After calculating the local priorities for criteria and alternatives, an $M \times N$ matrix is constructed, where $M$ is the number of alternatives considered and $N$ is the number of criteria. The global priority

<table>
<thead>
<tr>
<th>Weight</th>
<th>Definition of weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Equally important</td>
</tr>
<tr>
<td>3</td>
<td>Moderately important</td>
</tr>
<tr>
<td>5</td>
<td>Strongly important</td>
</tr>
<tr>
<td>7</td>
<td>Very strongly important</td>
</tr>
<tr>
<td>9</td>
<td>Extremely important</td>
</tr>
</tbody>
</table>

Table 1: Scale of weights.
is calculated by the following equation:
\[ GP_k = \sum_{j=1}^{N} (LPA_{kj}) \times (LP_j) \],
where \( GP_k \) is a global priority for alternative \( k \), \( N \) is the number of criteria, \( LPA_{kj} \) is a local priority of alternative \( k \) for criterion \( j \) (1 \( \leq k \leq M \), \( M \) is the number of alternatives) for criterion \( j \) (see the tables on the top right of Fig. 1(b) for a visual depiction of \( LPA_{kj} \)), and \( LP_j \) is a local priority of criterion \( j \).

The alternative with the highest global priority is the preferred alternative and would be selected as the alternative to use. This process is illustrated in Fig. 1b. The top left of the figure shows an example of pairwise comparisons between the criteria, the top right illustrates the set of pairwise comparisons conducted between the alternatives in regard to each criterion, and the matrix at the bottom of the figure shows how these comparisons are used in the calculation for the global priority.

Consider the following as an example of the strategy being applied. A company is about to release its eighth software system. They are trying to decide whether to use the original test suite (A1), a block coverage technique (A2), an additional coverage technique (A3), or a random testing technique (A4). The decision maker has determined that the criteria most important in this decision is the time to run the prioritization technique (C1), the cost of software artifact analysis (C2), the cost of delayed fault detection (C3), and the cost of missed faults (C4). The decision maker weighed each criterion (see Section 4.4 for more information on how weights can be determined) and came up with the decision matrix in Table 2.

Next, the decision maker would weigh each testing technique considered against each other in regard to each alternative. A sample decision matrix for one criterion (C1) is shown in Table 3. When applying the strategy, a decision matrix for each criterion in regard to each alternative would need to be produced.

Based on these weights provided in the decision matrices, the local and global priority can be calculated, and the testing technique with the highest global priority is chosen for that software release.

### 3.2. Fuzzy AHP method

The success of the AHP process relies on human judgment made by the decision makers. The weights provided by the decision makers are subjective to their knowledge and experiences. If the decision makers are inexperienced and lack knowledge of the application domain under consideration, the data they produce would be imprecise. Fuzzy set theory was originally introduced and proved by Zadeh (1965) as a way to represent imprecise data.

As mentioned in Section 2, different methods have been used to apply fuzzy set theory to the AHP method. Chang’s extent analysis (Chang, 1996) is the most commonly used fuzzy AHP method and is considered to handle the inaccuracies of the judgments made by decision makers. Therefore, in this work, we use the extent analysis in the fuzzy AHP method for ATP.

The fuzzy AHP method begins in similar fashion to the AHP method by building the AHP hierarchy. The difference between AHP and fuzzy AHP comes in the processes that follow the pairwise comparisons. In fuzzy AHP, the pairwise comparisons are converted into triangular fuzzy numbers (TFNs). The TFNs that correspond to the AHP weights (on the popular scale of 1–9) are shown in Table 4.

Using the comparison matrices containing the TFNs, the extent analysis then performs the following steps:

**Step 1:** Find the fuzzy synthetic extent with respect to the ith object.

To calculate the fuzzy synthetic extent value, let \( C = \{ C_1, C_2, \ldots, C_n \} \) be a set of \( n \) criteria, and \( A = \{ A_1, A_2, \ldots, A_m \} \) be a set of \( m \) alternatives, and \( M_{ij} \) be TFNs for the \( i \)th criteria. The value of the fuzzy synthetic extent \( S_i \) with respect to the \( i \)th criteria is
\[ V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} \min(S_2 \geq S_1) \\ \min(S_2 \geq S_1) \\ \vdots \\ \min(S_n \geq S_1) \end{bmatrix} \]

where, for element \( i \), the subscript \( k \in \{1, 2, ..., n\} \) and \( k \neq i \). The degree of possibility of \( S_2 = (l_2, m_2, u_2) \geq S_1 = (l_1, m_1, u_1) \) is obtained by:

\[
V(S_2 \geq S_1) = \begin{cases} 1, & \text{if } m_2 \geq m_1 \\ 0, & \text{if } l_2 \geq u_2 \\ \frac{l_1 - u_2}{m_2 - u_2} - (m_1 - l_1), & \text{otherwise} \end{cases}
\]

Step 3: Normalize the weight vector

\[
W = \frac{1}{\sum_{i=1}^{n} (w_i)} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}
\]

Step 4: Choose the optimal alternative

The optimal alternative is the alternative with the highest global priority that is obtained from Step 3.

When applying this strategy, the procedure would be the same for the decision maker and an example of this process is given in Section 3.1. The fuzzy AHP strategy differs because the weights provided by the decision maker would be transformed into triangular fuzzy numbers. For example, the weights provided in Table 2 would be transformed into triangular fuzzy numbers as shown in Table 4.
3.3.4. Example applying FESART

As an example of applying FESART, consider the example about the company releasing its eighth software system from Section 3.1. The decision maker would not be required to complete any pairwise comparisons. Instead he or she would be required to give provide into the expert system. For this example, consider that the expert system was built to require input for four different cost criteria on a scale from 1 to 9 for each prioritization strategy being considered. The decision maker would simply provide a 1–9 judgment for each cost criterion for each prioritization strategy, and the suggested strategy would be provided as output by FESART.

3.3.2. Fuzzy inference

The fuzzy inference process takes the fuzzified input from the fuzzification process and determines the fuzzy output set. The fuzzy output set is determined by using fuzzy rules. In a fuzzy expert system, the fuzzy rules bring expert knowledge into the system to aid in the decision making process. The knowledge needed to construct fuzzy rules in a fuzzy expert system comes from a combination of several different sources. The most widely used sources are human knowledge and expertise, historical data analysis of a system, and engineering knowledge from existing literature. To develop rules for FESART, knowledge about the factors that influence cost–benefits for regression testing techniques is needed. To gain this knowledge, each of the previously mentioned methods can be used.

For example, if a FESART system was built that considered four cost criteria (the cost of applying prioritization technique, the cost of missed faults, the cost of delayed fault detection, and the cost of software artifact analysis), each criterion could be considered and evaluated through information gained from the methods listed previously. Using this knowledge, rules could be built to ensure that costs that have the strongest impact on the final cost will impact the fuzzy output set of the fuzzy inference process accordingly.

3.3.3. Defuzzification

The last step in FESART is to use the defuzzification process to provide decision makers with crisp output to use in their decision making process. Many different defuzzification techniques have been proposed, but the center of gravity is the most widely used method for the center of gravity, the defuzzification technique is calculated using the equation:

\[
WS = \frac{1}{n} \sum_{k=1}^{n} \frac{c_k}{\sum_{i=1}^{m} w_i x_{ki}}
\]

3.4. Weighted sum model (WSM)

The weighted sum model (WSM) is a simple MCDM technique in which the weighted sum is used to determine the best alternative. To use the WSM, given m variables \(x_1, ..., x_m\) and n weights \(w_1, ..., w_m\), the weighted sum is defined as:

\[
WS = \sum_{k=1}^{n} (w_k x_k)
\]

Using WSM as an ATP strategy, the decision maker begins by determining the criteria, which will become \(x_1, ..., x_m\) in the formula provided above, and weights for the criteria \((w_1, ..., w_m)\). Like the other strategies for ATP described previously, decision makers would need to determine criteria important to choosing cost-effective regression testing techniques. Criteria that are more important for meeting the goal are given higher weights. Then, each alternative is given a performance value for each criterion. The performance value in WSM can be a wide range of values. For example, a decision maker might feel most comfortable giving a performance value from 1 to 100 and would use this range of values as the performance value for each alternative in terms of each criterion.

In our experiments (provided in Section 4), the decision makers used a performance value of 1 to 9 since they were already comfortable using this type of a scale to make judgments on the criteria and alternatives in the other ATP strategies. A decision matrix is composed of the criteria, criteria weights, alternatives, and performance values. An example decision matrix is provided in Table 5. In this particular example, the decision maker has chosen three criteria \((C_1, C_2, C_3)\) and three alternatives \((A_1, A_2, A_3)\). Weights of .3 have been assigned to \(C_1\), .45 to \(C_2\), and .25 to \(C_3\). Based on this data, one can conclude that this decision maker felt that \(C_2\) was the most important toward reaching the goal because it received the highest ranking. The alternatives were given a performance value for each criterion. In this particular example, we can see that the decision maker felt that the alternative, \(A_2\), met the criterion, \(C_2\), the best out of the three alternatives.

Once the weights and performance values are determined, the weighted sum can be calculated using the decision matrix. For example, the weighted sum for \(A_1\) would be calculated as follows:

\[
WS(A_1) = .30 \times 5 + .45 \times 7 + .25 \times 5 = 5.9
\]

Similarly, \(A_2\) has a weighted sum of 6.2, and \(A_3\) has a weighted sum of 6.0. The alternative with the highest weighted sum is the preferred alternative. Here, \(A_2\) is the preferred alternative.

WSM has received many criticisms (Kim and De Weck, 2005; Scott and Antonsson, 2005), but it is still a widely used decision making method (Khorasani et al., 2012; Triantaphyllou, 2000). The WSM is popular because of its simplicity and scalability. For these reasons, this research studies WSM as an ATP strategy to investigate how effectively this simple, low-cost approach can identify the most cost-effective regression testing technique for a particular regression testing session.

4. Empirical studies

To investigate the cost-effectiveness of the strategies presented in Section 3, we conducted a series of empirical studies. These
empirical studies are presented in this section. First, the objects of analysis, variables, time constraints, and cost criteria used in the experiments are presented, and then the details of each experiment are outlined separately. The first experiment investigates the fuzzy AHP strategy, focusing on its effectiveness when compared to the traditional AHP strategy. The second experiment studies the fuzzy expert system, FESART, and the last study is a comparative study between each of the ATP strategies presented in this research.

4.1. Objects of analysis

For each of the empirical studies, five Java programs from the SIR infrastructure (Do et al., 2005) were used. The five programs used are ant, xml-security, jmeter, nanoxml, and galileo. Ant is a Java-based tool similar to the Unix tool make in which extensions are implemented as Java classes instead of shell-based commands. Jmeter is a load-testing tool. Xml-security is a component library that implements XML signature and encryption standards. Nanoxml is a small XML parser for Java, and galileo is a Java Bytecode analyzer.

Each program is outlined in Table 6. The first column, Versions, shows the number of versions of the object program. The second column, Classes, shows the number of class files in the latest version of that program. Size (KLOCs) is the number of lines of code (in thousands), and Test Cases shows the number of test cases available for the latest version of the program. The first three programs were provided with JUnit test suites, and the last two were provided with TSL (Test Specification Language) test suites. In this study, we required object programs containing faults to collect fault data to use in the cost–benefit calculations for the techniques. The programs we used, however, were not supplied with any such faults or fault data. Thus, we used mutation faults generated using the ByteME (Bytecode Mutation Engine) tool developed by one of the authors. It can be obtained from the SIR repository (Do and Rothermel, 2006b). The last column in Table 6 lists the sum of all mutation faults across all versions of the program. In actual testing scenarios, programs do not typically contain as many faults as these numbers of mutants. Thus, to simulate more realistic scenarios, our previous study (Do and Rothermel, 2006a) introduced mutant groups, which were formed by randomly selecting mutants from the pools of mutants created for each version; each mutant group size varied (randomly) between 1 and 10. In this study, we also performed our experiment considering the mutant groups.

4.2. Variables and measures

Each of the studies utilized one independent variable and one dependent variable. The independent and dependent variables are described in the next sections.

4.2.1. Independent variable

The independent variable studied in the experiments is the test case prioritization technique application mapping strategy. The mapping strategy assigns a test case prioritization technique to each version in a software system. The mapping strategies considered in each experiment vary, and will be discussed in the respective experiment sections.

When assigning test case prioritization techniques, the strategies considered four prioritization techniques and chose the technique that the strategy deemed most appropriate for a specific software version. The four prioritization techniques considered were total block coverage, additional block coverage, random order, and original order. Total block coverage sorts test cases by the order of the number of blocks they cover. Additional block coverage selects a test case that yields the greatest block coverage, adjusts the coverage information for the remaining test cases to indicate their coverage for the blocks not yet covered, and then repeats this process until all blocks are covered by at least one test case. Random order will randomly order test cases. When using random order for analysis, often the average of a number of runs with random ordering of test cases is considered to obtain unbiased results. In our experiments, we ran 30 times and used the average of 30 cost–benefit values explained in Section 4.2.2. Original order executes the test cases in the order they are written in the scripts.

4.2.2. Dependent variable

The dependent variable in the experiments is the relative cost–benefit value. This value determines whether one strategy is beneficial to another strategy by considering both the costs associated with testing activities and the benefits provided by the testing techniques. To calculate the cost–benefit for a prioritization technique, we used the EVOMO economic model (Do and Rothermel, 2008).

The EVOMO model contains two different equations: one to capture testing costs and one to capture benefits. The costs are captured by measuring the time required by testing activities and then translating that into cost by using the time to calculate how much it would cost for test engineers to perform those testing activities. The equation for capturing benefits is related to the revenue gains (or losses) that correspond to the change in system release time related with using a prioritization technique. The two equations for the EVOMO model are shown in Eqs. (9) and (10). Terms and potential measures that can be used to capture these are summarized in Table 7.

\[
\text{Cost} = \text{salary} \times \left( \sum_{i=2}^{n} (\text{testSetup}(i) + \text{obsTests}(i) + \text{resultVal}(i) + \text{missFaults}(i)) \right)
\]

\[
\text{Benefit} = \text{revenue} \times \left( \sum_{i=2}^{n} (\text{delivery}(i) - \text{testSetup}(i) + \text{obsTests}(i) + \text{analysis}(i - 1) + \text{techRun}(i) + \text{testExec}(i) + \text{resultVal}(i) + \text{delayedFD}(i)) \right)
\]

The major cost components that EVOMO captures are as follows: costs for applying regression testing techniques, costs associated with missed faults, costs for artifact analysis, costs of delayed fault detection feedback, and costs associated with obsolete tests. Measuring the costs for most components is straightforward, but determining the cost of missing faults is more difficult. Because we could not obtain this measure directly, we estimate it considering the survey data by Shull et al. (2002). Based on their survey data, we use 1.2 h as the time required to correct faults after delivery.2

2 Note that the survey data suggest that fixing “ordinary” faults require 1.2 h before delivery, but fixing faults after product release are much more expensive. In our study, we chose 1.2 h in order to not inflate results, given that our object programs are of small and medium size.

Table 6

<table>
<thead>
<tr>
<th>Objects</th>
<th>Versions</th>
<th>Classes</th>
<th>Size (KLOCs)</th>
<th>Test cases</th>
<th>Mutation faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>ant</td>
<td>9</td>
<td>914</td>
<td>61.7</td>
<td>877</td>
<td>412</td>
</tr>
<tr>
<td>jmeter</td>
<td>6</td>
<td>434</td>
<td>42.2</td>
<td>78</td>
<td>386</td>
</tr>
<tr>
<td>xml-sec</td>
<td>4</td>
<td>145</td>
<td>15.9</td>
<td>83</td>
<td>246</td>
</tr>
<tr>
<td>nanoxml</td>
<td>6</td>
<td>64</td>
<td>3.1</td>
<td>216</td>
<td>204</td>
</tr>
<tr>
<td>galileo</td>
<td>16</td>
<td>68</td>
<td>14.5</td>
<td>912</td>
<td>2494</td>
</tr>
</tbody>
</table>
### Table 7
Terms and potential measures.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>Software system</td>
</tr>
<tr>
<td>$i$</td>
<td>Index denoting a release $S_i$ of $S$</td>
</tr>
<tr>
<td>$n$</td>
<td>The number of releases of the software system</td>
</tr>
<tr>
<td>$u$</td>
<td>Unit of time (e.g., hours or days)</td>
</tr>
<tr>
<td>$\text{testSetup}(i)$</td>
<td>Time to perform setup activities required to test $S_i$</td>
</tr>
<tr>
<td>$\text{obliTests}(i)$</td>
<td>Time to identify and repair tests that are obsolete for $S_i$</td>
</tr>
<tr>
<td>$\text{analysis}(i)$</td>
<td>Time to instrument all units in $i$ and to collect traces for test cases in $S_i$</td>
</tr>
<tr>
<td>$\text{techRun}(i)$</td>
<td>Time to execute a prioritization technique on $S_i$</td>
</tr>
<tr>
<td>$\text{testExec}(i)$</td>
<td>Time to execute test cases on $S_i$</td>
</tr>
<tr>
<td>$\text{resultVal}(i)$</td>
<td>Time to check outputs of test cases on $S_i$</td>
</tr>
<tr>
<td>$\text{missFaults}(i)$</td>
<td>Cost of missed faults after delivery of $S_i$</td>
</tr>
<tr>
<td>$\text{delayFD}(i)$</td>
<td>Cost of delayed fault detection feedback on $S_i$</td>
</tr>
<tr>
<td>$\text{revenue}$</td>
<td>Revenue in dollars per unit $u$</td>
</tr>
<tr>
<td>$\text{salary}$</td>
<td>Average hourly programmer’s salary in dollars per unit $u$</td>
</tr>
<tr>
<td>$\text{deliveryT}(i)$</td>
<td>Expected time-to-delivery for $S_i$ when testing begins</td>
</tr>
</tbody>
</table>

Two approaches for comparing techniques can be considered. The first approach calculates absolute cost–benefit values for each technique; however, a drawback of this approach is that it requires data or estimates pertaining to the $deliveryT$ variable, and it is difficult to find such data or reasonable estimates for our object programs.

The second approach calculates relative cost–benefit values, in which the cost–benefits of techniques are determined relative to those of a baseline technique. This approach does not require values for $deliveryT$; moreover, it normalizes the cost–benefit values calculated for techniques relative to a shared baseline, rendering their comparison independent of particular choices of $deliveryT$.

The cost–benefit calculation from the EVOMO model for each prioritization technique can then be used to determine the relative cost–benefit value by comparing the test case prioritization technique application mapping strategies (the independent variable) considered in each experiment. To calculate the relative cost–benefit of a given strategy, $S$, the benefit and cost for the strategy using the equations in the EVOMO model are calculated, and the benefit and cost for a baseline strategy, $base$, are calculated. The relative cost–benefit, $\text{RelCB}$, of strategy, $S$, with respect to baseline strategy, $base$, is then calculated using the following equation:

$$\text{RelCB} = \frac{(\text{Benefit}_S - \text{Cost}_S)}{(\text{Benefit}_{base} - \text{Cost}_{base})}$$  \hspace{1cm} (11)

### 4.3. Considering time constraints

Testing activities are often cut short because of tight deadlines or budgetary constraints. Not all of the planned testing can always be completed. The amount of time by which testing activities are shortened varies by the circumstances of a particular release. Further, the degree of time constraints can vary as systems evolve. For instance, for a certain release, a company could suffer more time constraints compared to other releases due to the addition of a complex feature or the loss of technical personnel. To account for this in our experiments, we used varying degrees of time constraints with each version when we evaluated test case prioritization techniques. Specifically, for each version, a time constraint of either 25%, 50%, or 75% was randomly assigned for each version. This process was repeated four times (to give four runs) for the first two experiments, and 20 times (20 runs) for the last experiment. The last experiment required more runs in order to have enough data for a statistical analysis.

### 4.4. Evaluating cost criteria

Each of the ATP strategies presented in this research provides the means to consider different factors that affect the cost-effectiveness of regression testing when choosing the test case prioritization technique for a particular regression testing session. Each of the studies presented here considers four cost criteria:

- Cost of applying the test case prioritization technique: the time required to run a test case prioritization algorithm
- Cost of software artifact analysis: the costs of instrumenting programs and collecting test execution traces
- Cost of delayed fault detection: the waiting time for each fault to be exposed while executing test cases under a test case prioritization technique
- Cost of missed fault: the time required to correct missed faults.

In the experiments, when the decision maker was evaluating the cost criteria, many different sources were utilized. For example, the decision maker utilized knowledge from previous empirical studies, history data from previous versions of the system, and different software metrics (such as number of classes, number of tests, program size, and change characteristics). For example, Elbaum et al. (2003) report results of a multiple case study investigating the modifications made in the evolution of four software systems. The goal of their study was to determine how size, distribution, and location of the modifications made to a software system during maintenance impact the cost-effectiveness of
regression testing techniques. The results of their study provide helpful tradeoffs and constraints that affect the success of regression testing techniques. For example, they found that the distribution of changes greatly impacted the difference in performance between the additional coverage (Acov) and total coverage (Tcov) prioritization techniques. They found that when the changes were highly distributed, it greatly benefited the Acov technique but often hurt the performance of Tcov.

Another series of empirical studies performed by Elbaum et al. (2002) revealed useful information regarding the effectiveness of different techniques. In general, their studies provide information regarding the tradeoff between the benefits of early fault detection versus the cost of applying the regression testing technique itself. If the cost of performing the technique is more than the savings generated by a higher rate of fault detection, then the technique is not worth employing. A technique is only superior to another technique if the gains achieved by the first technique with respect to the second technique are greater than the additional costs (if any) of using the first technique.

The knowledge gained from these studies and additional studies (e.g., Do et al., 2010; Do and Rothermel, 2006a; Elbaum et al., 2004), along with knowledge of the systems under test, knowledge of previous versions of the system under test, and the decision maker's experience with regression testing can provide adequate knowledge for decision makers to use in each of the ATP strategies presented in this work. In addition to the decision makers in our study, the same process can be used by testers in industry to provide the weights necessary for the ATP strategies.

4.5. Empirical study 1: investigating the fuzzy AHP strategy

In the first empirical study, we studied the effectiveness of the fuzzy AHP strategy in comparison with the AHP strategy. This strategy was developed to address the inaccuracies introduced by the decision maker in the pairwise comparison process. This empirical study investigates whether the fuzzy AHP method is more effective than the AHP method in terms of ATP by studying the following research question:

RQ1: Is the fuzzy AHP method more effective than the AHP method for selecting appropriate test case prioritization techniques across the system lifetime?

The following subsections present the variables and measures, the experimental setup and design, and the results of the experiment.

4.5.1. Independent variable

The independent variable, the test case application mapping strategies, was discussed in Section 4.2.1. Specifically this study considers six mapping strategies consisting of the following:

- Orig-all: Uses the original technique across versions.
- Tcov-all: Uses the total block coverage technique across versions.
- Acov-all: Uses the additional coverage technique across versions.
- Rand-all: Uses the random technique across versions.
- AHP: Selects the best technique among four prioritization techniques (Tcov, Acov, Rand, and Orig) using the AHP method.
- Fuzzy AHP: Selects the best technique among four prioritization techniques (Tcov, Acov, Rand, and Orig) using the fuzzy AHP method.

4.5.2. Dependent variable

The dependent variable is the relative cost–benefit value as outlined in Section 4.2.2. For this experiment, the baseline strategy was chosen to be the Orig-all strategy.

4.5.3. Experiment setup and procedure

To conduct this experiment, two different decision makers (each with over 7 years of industry experience) performed the AHP and fuzzy AHP ATP strategies considering the four cost criteria listed in Section 4.4 and the four prioritization techniques explained in Section 4.2.1. Both the AHP strategy and the fuzzy AHP strategy were performed considering the random time constraint assigned (the process for assigning time constraints is explained in Section 4.3). Testing activities for each of the four prioritization techniques were performed for each time constraint, and the appropriate costs and benefits were calculated based on the results from the testing activities performed. The relative cost–benefit values were calculated using Eq. (11) in Section 4.2.2. The cost–benefit results for the prioritization technique chosen by each of the six strategies were recorded for the four runs of random time constraints.

4.5.4. Data and analysis

This section presents the results of the study. The relative cost–benefit results for the four runs of random time constraint assignments for each version of the five programs are broken down into two tables (Tables 8 and 9). To show how many faults have been missed by each strategy, Table 10 presents the total number of missed faults across all versions of each program for each strategy for Run 1. The results for ant, jmeter, and xml-security are presented in Table 8, and the results for nanoxml and galileo are shown in Table 9. In each of the tables, the results for decision maker 1 (DM1) are shown under the headings AHP-1 and fuzzy AHP-1, and decision maker 2 (DM2) are shown under AHP-2 and fuzzy AHP-2.

The results shown in Table 8 show that when looking at the total cost–benefit calculations, ant and jmeter have similar results. The fuzzy AHP method consistently performs better (has greater cost–benefit) than the AHP method and control strategies. Specifically, for ant, the totals for the fuzzy AHP method outperformed each of the other strategies for all four runs for both decision makers with only one exception in which AHP-2 tied with fuzzy AHP-2 on run 2. For jmeter, fuzzy AHP-1 had the greatest cost–benefits for all four runs, and fuzzy AHP-2 had the greatest cost benefits for three runs (having tied with AHP-2 and Acov-all on run 4), and was outperformed by the Rand-all strategy in Run 1 (AHP-1 was also outperformed by the Rand-all strategy on this run).

The results for xml-security were much different. First there were only three versions, so the total cost–benefits won't show as much of a difference as the other versions. Also, the results show that for both the AHP and fuzzy AHP strategies for both decision makers the Acov strategy was the chosen technique for all versions. One main reason for this is because for this particular program, the changes between subsequent versions were relatively small, and therefore both decision makers ranked Acov higher in terms of delayed fault detection. Both decision makers had also ranked the cost of delayed fault detection higher than other criteria in the criteria comparisons, so it had a large impact on the final global priorities.

The total cost–benefit results for nanoxml and galileo (shown in Table 9) follow a similar trend as ant and jmeter. For both programs, the fuzzy AHP strategy outperformed each of the other strategies in all cases except one. AHP-2 and fuzzy AHP-2 tied on the fourth run for nanoxml and AHP-2 was slightly more cost-effective than fuzzy AHP-2 in run 3 for galileo.

The total cost–benefit calculation helps provide some insight into the performance of the different strategies, but one version's

---

4 The rest of the runs show similar numbers, and the numbers produced by the strategies (FESART and WSM) in Experiments 2 and 3 are very close to those by AHP and fuzzy AHP.
cost–benefit calculation could skew the results. To account for this, we examined the results by the total number of versions that the strategies produced the best results. Fig. 3 shows the average (across all runs) of the total number of versions that produced the best results for every strategy for each object program. In the figure, the results of AHP-1 and AHP-2 are averaged and displayed as AHP-avg. Similarly, Fuzzy AHP-1 and fuzzy AHP-2 are averaged and denoted as FuzzyAHP-avg. There are cases in which a strategy did not have any versions in the object program where it was most cost–effective for any of the four runs (e.g., Tcov-all in nanoxml). For those cases, the figure contains the word zero instead of a vertical bar.

The results shown in this figure produced similar results as the total cost–benefit analysis, namely that the fuzzy AHP consistently performed better than the other strategies. FuzzyAHP-avg produced the best results across all programs except for one program: for xml-security, FuzzyAHP-avg tied with two other strategies (Acov-all and AHP-avg).
4.5.5. Discussion

The results of the study show that the prioritization techniques chosen by the fuzzy AHP process are consistently more cost-effective than the control strategies and the AHP strategy with only a few exceptions. Those results indicate that by using fuzzy set theory in the AHP process to handle imprecision by decision makers, we are able to provide a more cost-effective Adaptive Test Prioritization (ATP) strategy.

This conclusion is supported by examining the average of the total number of versions in which each strategy was most cost-effective, where we can see that the fuzzy AHP strategy is the highest for all five programs. This conclusion holds when the average of the number of most cost-effective versions across all runs is considered as well. The fuzzy AHP strategy is consistently more cost-effective than all other strategies considered in this study. The findings of the study help provide a more cost-effective strategy for researchers and practitioners to use in their regression testing planning.

4.6. Experiment 2: a fuzzy expert system for ATP

In this section, we discuss an empirical study conducted to evaluate FESART. A fuzzy expert system can address the time and scalability issues of the fuzzy AHP and AHP strategies because it does not require pairwise comparisons. To investigate whether a fuzzy expert system can provide a cost-effective ATP strategy, the following research question was studied:
RQ2: Is a fuzzy expert system that considers testing environments and contexts more cost-effective than the other ATP strategies presented to date at choosing the most cost-effective regression testing techniques across a system lifetime?

4.6.1. Independent variable

This study considers the following test case application mapping strategies:

- AHP: Uses the ATP strategy utilizing the AHP method across all versions. This technique is used as the baseline strategy.
- Fuzzy AHP: Uses the ATP strategy utilizing the fuzzy AHP method across all versions.
- FESART: A new ATP strategy that utilizes a fuzzy expert system to select the best technique across all versions.

4.6.2. Dependent variable

The dependent variable in the study is the relative cost–benefit value (discussed in Section 4.2.2). In this study, we made one modification to the EVOMO model. The costs considered in the EVOMO model account for the costs related to the regression testing techniques, but they do not consider the cost related to applying the ATP strategy. In the previous studies, this cost was not considered in the cost–benefit calculations. However, to evaluate the approaches properly, the cost of applying the strategy should be considered. In this research, this cost was added to the cost–benefit calculations. The EVOMO model was modified to include one additional cost: $C_{ATP}$ (the cost of applying the ATP strategy). $C_{ATP}$ is a cost related to human effort, so it is applied in the equations in the same way as other costs related to human effort (by capturing the cost related to the salary of the engineer who performed the activity).

4.6.3. Experiment setup and procedure

The experimental setup is similar to that of the first empirical study. Two decision makers with industry experience completed the AHP, fuzzy AHP, and FESART ATP strategies for each of the five programs (shown in Section 4.1) for four runs of random time constraints.

For this experiment a fuzzy expert system consisting of four input variables (corresponding to the four cost criteria described in Section 4.4) was built. The decision maker evaluated each cost criterion on a scale from 1 to 9, with 9 being a very high cost. Three membership functions, shown in Table 11, were used for each of the criteria being considered. For this experiment, we chose to use triangular membership functions. Each input was classified according to its degree of membership to these membership functions. This resulting fuzzy input set was used as input for the fuzzy inference process.

The fuzzy inference process takes the fuzzified input set and determines the fuzzy output set. The fuzzy output set consists of eight triangular membership functions. The output is rated on a scale from 1 to 9. The membership functions are shown in Table 12. The output set is built to categorize the overall cost for the regression testing technique and is categorized from low to high. L1, L2, and L3 are considered low costs, with L1 being the lowest. Then, A1 and A2 are categorized as average costs, with A1 being lower than A2. H1, H2, and H3 are all high costs, with H3 being the highest cost.
The fuzzy output set is determined by using fuzzy rules. In a fuzzy expert system, the fuzzy rules bring expert knowledge into the system to aid in the decision making process. The knowledge needed to construct fuzzy rules in a fuzzy expert system comes from a combination of several different sources. The most widely used sources are human knowledge and expertise, historical data analysis of a system, and engineering knowledge from existing literature. To develop rules for a fuzzy expert system in ATP, knowledge about the factors that influence cost–benefits for regression testing techniques is needed. To gain this knowledge, each of the previously mentioned methods was used. For example, from the literature (Do and Rothermel, 2008; Elbaum et al., 2001), it can be said that the costs of delayed fault detection are greater than the costs related to setting up and running the test cases, so the fuzzy rules involving these two items would give more importance to the cost of delayed fault detection.

Each criterion was considered and evaluated through information gained from the methods listed above. Using this knowledge, the criteria were ordered by their impact on cost–benefit tradeoffs. The order was determined to be the cost of missed faults (CF), the cost of delayed fault detection (CD), the cost of applying the prioritization techniques (CR), and the cost of software artifact analysis (CA), with the cost of missed faults having the strongest impact and the cost of software artifact analysis having the least impact.

For each of the criteria possible combinations of membership functions were considered. There are four input variables, and three membership functions for each one, so there are 81 unique combinations. Each combination was evaluated and assigned an appropriate output set. Then, the rules were studied to see if any of them could be combined or eliminated. We were able to reduce the rule set to 67. The following example demonstrates how we were able to reduce the rule set. In the original rule set, the following three rules existed:

- IF CF is H and CD is H and CR is A and CA is H then Cost is H3.
- IF CF is H and CD is H and CR is L and CA is L then Cost is H2.
- IF CF is H and CD is H and CR is A and CA is H then Cost is H1.

For each possible value for CA, the output set was still H3. The value of CA did not have any impact on these particular rules, so we can reduce the three rules into the following rule:

IF CF is H and CD is H and CR is H then Cost is H3.

A subset of the fuzzy rules is shown in Table 13. Because CF and CD were determined to have the most impact on the cost–benefit calculations and are classified as a high cost in these rules, the final cost is categorized in the different high cost output sets (H1, H2, and H3). When CR and CA have a higher cost, then the final cost is determined at the higher end of the high output sets (H2 or H3). When CR and CA have a low cost, the final cost is still considered high because CF and CD are high and have more impact, but the cost on the lower end of the high cost (H1).

After the fuzzy output set undergoes the fuzzification process, the number provided by the defuzzification process for one technique can be used to compare with numbers from the defuzzification process for other techniques. The output provided by the defuzzification process gives a crisp number in terms of the overall cost of the prioritization technique. The technique with the lowest number is the technique that is expected to be the most cost-effective for that particular software version.

The cost–benefit results for the prioritization technique chosen for each strategy were calculated for each version of every program across the four runs of random time constraints, and then the relative cost–benefit was calculated using AHP as the baseline strategy.

4.6.4. Data and analysis

This section provides the results of the study. Table 14 shows the relative cost–benefit results for the four runs of random time constraint assignments for each version of the five programs. Results for the first decision maker for the ATP strategy utilizing fuzzy AHP are labeled fuzzy AHP-1, and fuzzy AHP-2 is used for the second decision maker. Similarly, the results for the first decision maker for FESART are labeled FESART-1, and FESART-2 is used for the second decision maker. AHP is not shown in the table because it is the baseline strategy.

When examining the total cost–benefit values for all of the versions for an object program, FESART was more cost-effective than the other ATP strategies for all four runs of all five programs. Fig. 4 represents these results visually. The figure shows the average total cost–benefit for the fuzzy AHP strategy was slightly higher. Some programs show more dramatic results than others. For example, the results for galleon show a greater difference than the other programs. One obvious reason for this is because it is the program with the most versions. For xml-security, you will notice that the results for fuzzy AHP are negative (and appear in red in Fig. 4). This is because in this particular case the HAP and fuzzy AHP strategy chose the same prioritization technique, and the cost of applying the fuzzy AHP technique was slightly higher. In addition, examining the individual versions for each program also consistently shows FESART as the most cost-effective strategy with only a few exceptions.

4.6.5. Discussion

The results clearly showed FESART to be the most cost-effective ATP strategy considered in this study. Likely one of the biggest contributions to it being the most cost-effective strategy is that the time required by the decision makers is less, making it a less costly strategy to apply. Because it is less costly to apply, even if the AHP or fuzzy AHP strategy chose the same prioritization technique, the cost of applying the fuzzy AHP technique was slightly higher. In addition to being less costly to apply, FESART also chose the most cost-effective technique more frequently than the other strategies. One possible explanation for this is the role base used in a fuzzy expert system. The rule base provides knowledge needed for ATP, and thus the amount of knowledge needed by the decision maker is not as high as is needed with the other strategies.

The findings of this study provide practical implications for both practitioners and researchers in software engineering by providing a more cost-effective strategy to use in their regression
testing sessions. In addition, since it is a less time-consuming strategy, it could prove to be easier to adopt in regression testing plans.

4.7. Experiment 3: a comparative study

In this study, we conduct a comparative study of each of the ATP strategies presented in this research. Specifically, this study investigates the impact of a low cost strategy on its overall cost-benefit performance. In the second empirical study, we saw that FESART was consistently more cost-effective than the AHP and fuzzy AHP strategies. One major contribution to that was because its low cost was due to not requiring time-consuming pairwise comparisons. To investigate whether the cost-effectiveness of FESART can be entirely attributed to the low cost of applying the comparisons. To investigate whether the cost-effectiveness of FESART can be entirely attributed to the low cost of applying the comparisons.

Table 4 Experiment results: relative cost-benefit in dollars.
of the ATP strategies presented in this research. This study will not only help us investigate the impact of a low cost strategy, but also provide a comparative study to help us understand the tradeoffs and cost-effectiveness of each of the strategies. The goal of this study is to answer the following research question:

RQ3: How do each of the ATP strategies perform when the cost of applying the strategy is considered?

4.7.1. Independent variable

This study considers four test case application mapping strategies. The four strategies considered are:

- AHP: Uses the ATP strategy utilizing the AHP method across all versions. This strategy is used as the baseline strategy.
- Fuzzy AHP: Uses the ATP strategy utilizing the fuzzy AHP method across all versions.
- FESART: Uses the ATP strategy that utilizes a fuzzy expert system to select the best technique across all versions.
- WSM: Uses the ATP strategy that utilizes the weighted sum model (WSM) to select the best technique across all versions.

4.7.2. Dependent variable

The dependent variable in this study is the same as in the other two studies, the relative cost–benefit value (described in Section 4.2.2). Like the second empirical study evaluating FESART, the EVOMO model was extended to incorporate the cost of applying the ATP strategy itself into the study. Also, similar to the last study, the AHP strategy was used as the baseline strategy in this experiment.

4.7.3. Experiment setup and procedure

This section discusses the setup and procedure of the experiment. The same process for assigning random time constraints used in the other studies, and explained in Section 4.3, was used in this study. But for this study, in order to gain enough data to perform a statistical analysis, 20 runs of random time constraints were used instead of the four runs of the previous studies.

Similar to the other studies, two decision makers with industry experience performed each of the strategies considered in this experiment (AHP, fuzzy AHP, FESART, and WSM). The testing activities were performed and the cost–benefit results were measured for each of the strategies. These data were used to conduct a comparative study by performing a statistical analysis on each of the strategies. The procedure for the statistical analysis is described in the next section, and then the results for each program are presented.

4.7.4. Statistical analysis procedure

The statistical analysis was performed using the Statistical Analysis System (SAS). To perform the statistical analysis, this research begins with the Kruskal–Wallis test (Ramsey and Schafer, 1997). This test was chosen because the data did not meet the assumptions required for the ANOVA procedure. The ANOVA procedure assumes that the data are distributed normally with no severe outliers. The data from this experiment did not meet this assumption. When assumptions for ANOVA are not successfully met, the Kruskal–Wallis test is a commonly used alternative method.

The Kruskal–Wallis test begins by ranking the data in terms of its rank to the overall data set. The smallest value gets a rank of 1, the second-smallest gets a rank of 2, etc. If there are data that are the same, the tied observations get average ranks. For example, if there are four identical values occupying the second, third, fourth, and fifth smallest places, these rankings would get averaged, and each would receive a ranking of 3.5.

Then, the sum of the ranks is calculated for each group and a test statistic which considers the variance of the ranks among the groups is calculated. This test statistic is approximately chi-square distributed, which means that the probability of getting a particular value by chance is the p-value corresponding to the chi-square.

The results for the Kruskal–Wallis test performed on the cost–benefit calculation for 20 runs for each version of every program are presented in Table 15.

For each of the programs, the p-values are less than .05, so the results indicate that there is a statistical significance between the groups. The results of the Kruskal–Wallis test do not reveal which

---

groups (in this case, which strategies) have a statistically significant difference, only that one does exist between at least one group. In order to draw conclusions on each strategy, further testing needs to be performed to investigate which strategy is statistically different from the other strategies.

Boxplots for each of the programs are shown in Fig. 5 and are discussed in their respective sections. In each boxplot, FESART-1 represents the cost–benefit results for FESART for the first decision maker, and FESART-2 represents the cost–benefit results for the second decision maker. Fuzzy AHP-1 is for the results for the fuzzy AHP strategy for the first decision maker, and fuzzy AHP-2 shows the results for the fuzzy AHP strategy for the second decision maker. WSM-1 represents the cost–benefit results for the first decision maker, and WSM-2 shows the results for the second decision maker. The cost–benefit results for the traditional AHP method are used as the baseline strategy, and so they are not shown in the boxplots.

To examine which strategy (or strategies) are statistically different from the others, a multiple comparison method is required. We used the Bonferroni method. The Bonferroni method was applied to each program and the results for each program are shown in Table 15. Each group (in this case, ATP strategy) is given a group letter. Strategies with the same group letter indicate they are not statistically different.

4.7.5. Results for ant
The boxplots for ant show that FESART (for both decision makers) is noticeably more cost-effective than the other two strategies for both decision makers. Between the decision makers, DM2 shows greater cost–benefits than DM1.

Unlike the results for FESART, the results are not as straightforward when comparing the fuzzy AHP and WSM strategies. The results for fuzzy AHP-1 and WSM-2 appear to be normally distributed, but the results for Fuzzy AHP-2 have a severe outlier, and the upper half of the values for WSM-1 are much more spread out than the lower half.

WSM for DM1 appears to be just slightly more cost-effective than the fuzzy AHP strategy for DM1 and WSM for DM2. There is not a noticeable difference between these three. However, the results for fuzzy AHP-2 are noticeably different from the other three strategies (fuzzy AHP-1, WSM-1, and WSM-2). The cost–benefit results for fuzzy AHP-2 have a severe outlier (represented by the circle below the lower quartile), and the results for WSM-1 show the upper fifty percent having greater variability than the lower fifty percent. When comparing the median scores, WSM-2 performed the best (among the four) with fuzzy AHP-1 closely following. The median scores for WSM-1 and fuzzy AHP-2 are fairly close together, with the scores for WSM-1 being slightly higher. From the boxplot it appears that no significant conclusions can be made between fuzzy AHP and WSM.

The Bonferroni results for ant show that both decision makers have the same groupings. The results show that the FESART strategy is grouped differently than the fuzzy AHP and WSM strategies, so there is a statistically significant difference between FESART and the other two strategies. The fuzzy AHP and WSM strategies share the same group, so there is no statistically significant difference between those two strategies.

4.7.6. Results for jmeter
The boxplots results for jmeter differ between the decision makers and will be discussed separately. Like ant, the box plots show FESART as the most cost-effective strategy. Unlike ant there is also a clear cost-savings for WSM for DM1 when compared to the fuzzy AHP strategy (for both decision makers) and WSM for DM2. There appears to be very little difference between the cost–benefit calculations for fuzzy AHP-1, fuzzy AHP-2, and WSM-2. Also, the results for WSM-2 show some severe outliers.

The Bonferroni results show that for DM1, there is a statistically significant difference between the FESART and fuzzy AHP strategies, but not between FESART and WSM. These two strategies (FESART-1 and WSM-1) are statistically more cost-effective than fuzzy AHP-1. For DM2, there is a statistically significant difference between FESART and the other two strategies with FESART being more cost-effective than the other two strategies, and the other two strategies being placed in the same group.

One interesting thing to note about the results for jmeter is that there is quite a bit of variance between the two decision makers for the WSM strategy. For DM1, the difference in the cost benefits between FESART and WSM-1 was not statistically significant. However, for DM2 there was quite a bit of difference, and it was statistically significant. WSM has been known to be a somewhat volatile decision making strategy, with the results being strongly dependent on the decision maker, so results like these would make sense for WSM.

4.7.7. Results for xml-security
The boxplots for xml-security, as with jmeter, show quite a bit of variance for WSM. This is especially true when looking at the results for DM1. Some values in the boxplot for WSM-1 are actually higher than the values for both decision makers for FESART. The median value for WSM-1, however, is lower than the median value for each of the other strategies. With the exception of a few of the extreme values for WSM-1, like ant and jmeter, FESART is the most cost-effective strategy.

The Bonferroni results show that, for DM1, there is a statistically significant difference between FESART and fuzzy AHP, but there is not enough difference between FESART and WSM or WSM and fuzzy AHP to say that there is a statistically significant difference. The wide range of values shown in the boxplots for WSM for DM1 actually placed WSM for DM1 in both groups A and B. This means that we cannot say there is a statistical significance in the difference between WSM for DM1 and strategies categorized in Group A or Group B. For DM2, there is a statistical significance between FESART and the other two strategies, but not between fuzzy AHP and WSM.

4.7.8. Results for nanoxml
The boxplots for nanoxml show quite a few outliers. Only FESART-2 and WSM-2 do not include any outliers. From the boxplots, the cost–benefits for FESART are greater than for any of the other strategies for both decision makers. And like jmeter and xml-security, the values for WSM are inconsistent. WSM-1 shows a low outlier, and there is quite a difference between the values for WSM-1 and WSM-2.

The Bonferroni results for nanoxml show that, once again, the groups the strategies were placed in are different between the two decision makers. Unlike any of the previous programs, however, there is a statistically significant difference between each of the strategies for both decision makers. FESART-1 and FESART-2 were placed in Group A, being the most cost-effective strategy. For DM1, fuzzy AHP is placed in the next group, Group B, while the WSM
strategy is placed in Group B for DM2. One interesting thing to note is that, because of the wide variance in values for WSM for DM1, it was grouped differently than WSM-2, although the median values for both were very close.

4.7.9. Results for galileo

The boxplot values for FESART show to be the most cost-effective again. The values here are higher than the values of the other programs because galileo contains more versions, which shows that as the life of a program grows longer, the cost savings of utilizing the ATP strategies is greater. The boxplots for the WSM and fuzzy AHP strategies show that the values for WSM are higher than the fuzzy AHP strategy for DM2, but show little difference between the values for DM1.

The Bonferroni results show there is a statistically significant difference between FESART and the other strategies for both decision makers. FESART is statistically the most cost-effective strategy. The results for the remaining two strategies are different between the two decision makers. For DM1, there is no statistically significant difference between the fuzzy AHP and WSM strategies,
Table 16
Bonferroni results.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>ant</th>
<th>jmeter</th>
<th>xml-security</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DM1</td>
<td>DM2</td>
<td>DM1</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>Grp</td>
<td>Mean</td>
</tr>
<tr>
<td>FESART</td>
<td>438.06</td>
<td>A</td>
<td>487.48</td>
</tr>
<tr>
<td>Fuzzy AHP</td>
<td>197.12</td>
<td>B</td>
<td>163.04</td>
</tr>
<tr>
<td>WSM</td>
<td>163.11</td>
<td>B</td>
<td>115.14</td>
</tr>
<tr>
<td></td>
<td>DM1</td>
<td>DM2</td>
<td>DM1</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>Grp</td>
<td>Mean</td>
</tr>
<tr>
<td>FESART</td>
<td>225.06</td>
<td>A</td>
<td>262.32</td>
</tr>
<tr>
<td>Fuzzy AHP</td>
<td>28.57</td>
<td>B</td>
<td>51.43</td>
</tr>
<tr>
<td>WSM</td>
<td>-94.54</td>
<td>C</td>
<td>149.86</td>
</tr>
</tbody>
</table>

Fig. 6. Average means for all programs.

but there is a statistically significant difference between these two strategies for DM2.

4.7.10. General results for all programs

Although the results differ between the programs and decision makers, to attempt to gain a general trend in the data, Fig. 6 represents the average between the decision makers of the mean value of 20 runs for each program.

For each of the programs, FESART was the most cost-effective strategy. The WSM strategy came in second, being more cost-effective than the fuzzy AHP strategy for three of the five programs (jmeter, xml-security, and galileo). The fuzzy AHP strategy was more cost-effective than WSM for two of the five programs (ant and nanoxml).

4.7.11. Discussion

The goal of this study was to investigate the effectiveness of each of the proposed strategies when the cost of applying the strategy is considered. The results of the study indicate that the low-cost strategy WSM was, at times, more cost-effective than the fuzzy AHP strategy, and consistently more cost-effective than the AHP strategy. However, the results also show that FESART was consistently more cost-effective than the WSM strategy. These results indicate that the cost-effectiveness of FESART is not solely related to its low cost. Other possible factors that contribute to FESART’s increased performance is the expert knowledge-base used to generate the fuzzy rule set and the use of fuzzy logic to handle imprecision in the input provided by the decision makers. The variability in the cost–benefit results for the WSM strategy between decision makers shows that WSM is more sensitive to the input provided by the decision maker. For each program, the results for FESART only had minimal variation, but the variation was great for WSM.

To further investigate the research question of how each of the strategies perform when the cost of applying the strategy is considered, a statistical analysis was performed for each of the ART strategies discussed in this research. The results of the statistical analysis show a few different trends. One trend that was consistent among all programs and each decision maker was that FESART was statistically different from the other strategies, being the most cost-effective strategy. Other trends that were shown in the study include the volatility of the WSM strategy. The results for WSM differed not only among the different programs but also between the two decision makers. Another trend, which is somewhat related to the volatility of WSM, is that there was not a consistent statistically significant difference between the WSM and fuzzy AHP strategies. For example, the results for ant showed no statistically significant difference between the WSM and fuzzy AHP strategies for either decision maker. The results for jmeter showed a statistically significant difference between the results for WSM for DM1, but not for DM2. The results for nanoxml showed a statistically significant difference for both decision makers, but for DM1, fuzzy AHP was grouped as a more cost-effective strategy, whereas
for DM2 WSM was grouped as the more cost-effective strategy. Because of the varying performance between the fuzzy AHP and WSM strategies, no sound conclusions can be made as to which of these two strategies is more cost-effective.

When compared to the traditional AHP strategy, the results show that the strategies presented in this work are consistently more cost-effective. In this study, the AHP strategy was used as the baseline strategy, so the cost–benefit calculation for all 20 runs for the AHP strategy would be zero, and therefore it is not shown in the boxplots. The boxplots show that the majority of the values for each of the strategies are above zero, meaning that they are more cost-effective than the AHP (baseline) strategy.

4.8. Threats to validity

This section describes the internal, external, and construct threats to the validity of our empirical studies.

Internal validity: In each of the studies, calculations were performed to calculate the global priorities that selected the preferred prioritization technique. The calculations were double-checked, but the possibility of small marginal human errors still exists due to some of the required input being entered by hand into the different calculation tools.

External validity: The external validity of this experiment could be limited by the decision makers used in the studies. We used two decision makers in this study. Their backgrounds and experience levels could differ from those of other programmers. We sought to reduce this risk by selecting decision makers who have several years of industry experience.

Construct validity: The construct validity could be threatened by the number of criteria considered in this experiment. We considered four cost criteria, but additional criteria could be considered, which could change the results. While the EVOMO model is the most comprehensive model created to date for use in assessing regression testing techniques, other testing costs not captured by the model could influence overall costs and benefits, in particular testing situations and organizations (the costs of initial test case development, test suite maintenance, and opportunity cost). Finally, we used the Bytecode Mutation Engine (Do and Rothermel, 2006b) to create the mutants used in our experiments. There are other mutation tools (such as Major Just (2014), Pit, and Muljava Ma et al. (2006)) that have been used in recent research. The use of these tools could produce different mutants and possibly different results because of the varying mutants. Control for these threats can be achieved through future studies with additional cost criteria, testing costs, and different mutation tools.

5. Conclusions and future work

In this work, we presented new ATP strategies that address the limitations of the previous strategy and evaluated them through a series of empirical studies. In particular, three strategies were developed. One strategy utilized the fuzzy AHP method to address the issue of the results from the AHP method being subjective to the judgments made by decision makers. A second strategy used a fuzzy expert system to obtain the benefits of a strategy that does not require pairwise comparisons. A third strategy utilized the weighted sum model (WSM) to investigate the effectiveness of a simple, low-cost strategy for ATP.

Each of the empirical studies evaluated the strategies in terms of their cost–benefit results. The results of the studies indicated that the new strategies presented in this work provide for greater cost-savings for regression testing than the previously proposed strategy. The studies revealed some helpful trends; for instance, the FESART strategy, overall, appears to be the most cost-effective strategy among all the strategies presented in this research. One major contribution to that is its low cost. However, the results of the studies also show that any low cost strategy would not produce the same results (when WSM was evaluated, there was a wide variance in its cost–benefit results, and often it was the least cost-effective of the strategies presented in this work).

The results found in this research provide important practical implications for both researchers and practitioners. Since the cost of regression testing is very high, strategies to reduce the cost are important. This work provides multiple strategies that can help reduce the cost of regression testing. The dollar amounts shown in this work may seem insignificant to some, but if the strategies were utilized in practice, the savings could be substantial. For instance, only small and medium sized programs were used in the studies in this work. Industrial applications are very large, many of them containing millions of lines of code (the programs used in this study were only in the thousands to tens of thousands). If ATP strategies were used on these large applications, the cost savings could be much larger than those presented in these studies. Also, these studies only considered ordinary faults. Studies have shown that the costs associated with severe defects are much greater than ordinary defects. Considering severe defects could greatly increase the cost–benefit calculations.

Additional contributions of this work include the empirical studies provided in evaluating the performance of different decision making strategies, some of which were severely lacking in the literature (i.e., fuzzy AHP versus traditional AHP). The empirical studies performed in this work will provide researchers with data demonstrating the success (or lack of success) of the varying methods used in the context of regression testing. Further, the empirical studies provide data for researchers and practitioners to use when considering adopting ATP strategies. The data will help them see how ATP strategies may be beneficial and help them choose an appropriate strategy to meet their regression testing needs.

5.1. Future work

Although this research has provided some important contributions, there are some areas in which this work could be studied in the future. First, these studies only considered four test case prioritization techniques. Additional prioritization techniques could be considered that may have a greater cost-savings than the ones considered in this research. Also, other regression testing techniques, such as test case selection and test case minimization techniques, could be considered.

Another important area for future work is to investigate the scalability of the strategies by incorporating larger programs. Each of the experiments in this work used five small to medium sized Java programs. Even with small and medium sized programs there were cost-savings shown in the experiments. We plan to utilize larger programs from the SIR Repository Do et al. (2005) to investigate the scalability of the strategies. If the ATP strategies scale with larger programs the cost-savings would be even greater.

This work considered four cost criteria for each strategy. Future work could consider additional cost criteria. Further, additional empirical studies which evaluate other factors that could contribute to the cost-effectiveness of regression testing techniques could be conducted which would give more data to use when deciding on an appropriate regression testing technique for a particular regression testing session. Additional knowledge could then be used by the decision maker or even integrated into the ATP strategies.

With the information provided in this work and any future work performed in the areas just mentioned, there is strong potential for large cost-savings in regression testing through the use of ATP strategies.
Acknowledgments

This work was supported, in part, by NSF awards CNS-0855106 and CCF-1050343 North Dakota State University, and NSF CAREER Award CCF-1564238 to University of North Texas.

References


Amanda Schwartz is an assistant professor in Computer Science at the University of South Carolina Upstate.

Hyunsook Do is an associate professor of Computer Science at the University of North Texas.