Information and Software Technology 93 (2018) 74–93

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Test case prioritization approaches in regression testing: A systematic literature review

Muhammad Khatibsyarbini*, Mohd Adham Isa, Dayang N.A. Jawawi, Rooster Tumeng

Department of Software Engineering, Faculty of Computing, Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia

ARTICLE INFO

Article history:
Received 29 December 2016
Revised 2 August 2017
Accepted 25 August 2017
Available online 1 September 2017

Keyword:
Test case prioritization
Regression testing
Software testing
Systematic literature review

ABSTRACT

Context: Software quality can be assured by going through software testing process. However, software testing phase is an expensive process as it consumes a longer time. By scheduling test cases execution order through a prioritization approach, software testing efficiency can be improved especially during regression testing.
Objective: It is a notable step to be taken in constructing important software testing environment so that a system's commercial value can increase. The main idea of this review is to examine and classify the current test case prioritization approaches based on the articulated research questions.
Method: Set of search keywords with appropriate repositories were utilized to extract most important studies that fulfill all the criteria defined and classified under journal, conference paper, symposiums and workshops categories. 69 primary studies were nominated from the review strategy.
Results: There were 40 journal articles, 21 conference papers, three workshop articles, and five symposium articles collected from the primary studies. As for the result, it can be said that TCP approaches are still broadly open for improvements. Each approach in TCP has specified potential values, advantages, and limitation. Additionally, we found that variations in the starting point of TCP process among the approaches provide a different timeline and benefit to project manager to choose which approaches suite with the project schedule and available resources.
Conclusion: Test case prioritization has already been considerably discussed in the software testing domain. However, it is commonly learned that there are quite a number of existing prioritization techniques that can still be improved especially in data used and execution process for each approach.

1. Introduction

Software engineering is not just programming and software development. Software engineering itself is an implementation of engineering procedures in the development of any software in a systematic way [1]. Within a software development process, software testing consumes a longer time in execution and can be the most expensive phase [2]. Software testing itself is normally, repetitively, carried out even when there are time constraint and fixed resources. Software engineering groups are regularly compelled to end their testing activities because of financial and time necessities, which will trigger some difficulties such as problems with the software quality and client agreement. However, the application of test case prioritization (TCP) appears to enhance test viability in software testing activity [3].

Regression testing is an activity to confirm that progressions do not harm the previously functioning software [4,5]. As the software evolves, a software test suite has the tendency to increase in size which frequently makes it expensive to execute. Research shows regression testing is an expensive process which may require more than 33% of the cumulative expenses of the software [6]. In the work of Yoo and Harman [7], various regression test approaches were examined to supplement the importance of the accumulated test suite in regression testing. Those studies were then classified into three domains; minimization, selection, and prioritization.

Test suite minimization (TSM) approaches intend to distinguish repetitive experiments and to eliminate test cases from a test suite execution with a specific goal such as to decrease the number of tests to run [8]. Minimization is sometimes called 'test suite reduction', meaning the elimination of test cases are permanent.

Test case selection (TCS) approach also aims to decrease the number of test cases to be executed, however, the main idea of selection approach is that it is intended to be modification-aware [9]. TCS tries to recognize the test cases which would be important to the latest changes on a software.

* Corresponding author.
E-mail addresses: kmuhammad4@live.utm.my, fkmuhammad4@gmail.com (M. Khatibsyarbini).

http://dx.doi.org/10.1016/j.infsof.2017.08.014
0950-5849/

Table 1
Regression test approaches.

Component	Regression test approach		
	Minimization	Selection	Prioritization
Strategy	Eliminate test case.	Modification- aware test case.	Test case permutation by ordering and prioritizing.
Strength	Effective in reducing test cases.	Effective in selecting modification-aware test cases.	Useful when new test cases will always be considered in the test case permutation.
Limitation	Test cases are not modification-aware.	New test cases might be missed out in the temporary selection that is modification-aware.	Time-consuming, larger test suite.

Table 2
Summary of related studies in regression testing.

Study type	Study references	Study focus	Year of publication	Total studies reviewed	Years covered
SLR	Singh et al. [13]	Test Case Prioritization	2012	65	1997–2011
Mapping	Catal and Mishra [14]	Test Case Prioritization	2013	120	2001–2011
Surveys	Yoo and Harman [7]	Regression Testing	2012	159	1977–2009
	Kumar and Singh [15]	Literature Survey on TCP	2014	19	NA
	Kiran and Chandraprakash [16]	Literature Survey on TCP	2015	90	NA

Lastly, test case prioritization (TCP) aims to order a set of test cases to achieve an early optimization based on preferred properties [3,10]. It gives an approach the ability to execute highly significant test cases first according to some measure, and produce the desired outcome, such as revealing faults earlier and providing feedback to the testers. It also helps to find the ideal permutation of a series of test cases and could be executed accordingly [7].

Table 1 shows a general comparison of three approaches in regression testing. Test case minimization reduces the test case amount in a set of test suite continuously while selection technique performs a temporary selection of several test cases which related are to modification awareness. From selective selection, important test cases might be missed out from the test suite. These test cases could possibly contain an important priority that needs to be executed to reveal certain faults. In test case prioritization, every single test case including new test cases that are added into present test suite execution will be considered in prioritization. This is crucial as new test cases will be executed to test a modified part of the software, hence, any abnormalities in the functional output could easily be observed.

Despite the fact that there are numerous TCP approaches in the literature, there are no latest progressive literature reviews which illustrate recent TCP importance in software testing research. Therefore, this review attempts to perform a systematic literature review (SLR) on the latest TCP approaches as proposed by Kitchenham [11]. SLR is a specialized, uncompromising, study of research evidence [12]. The point of an SLR is not to simply summarize all current proofs based on research questions, it is also expected to bolster the improvement of evidence-based research recommendations for researchers.

This systematic literature review is structured as follows. Section 2 considers the previous studies related to TCP approaches. Section 3 describes the strategy embraced to direct this SLR. Next, result and discussion based on the research questions were discussed in Section 4. Research findings were then elaborated in Section 5. In Section 6, the threat of validity to this SLR was discussed. Finally, Section 7 presents conclusion with regard to this systematic literature review.

2. Background studies of test case prioritization

This section will discuss the previous studies that are related to TCP in regression testing. There were a few systematic reviews originated under the regression test case prioritization techniques domain. From the literature gathered, the authors collated one SLR, one mapping study, and three survey studies that are related to regression testing and TCP, as tabulated in Table 2.

The only SLR, work by Singh [13], offered a systematic review in regression test case prioritization study covering the time period from 1997 to 2011. In their work, from 65 studies, 49 were identified to initiate a different approach, two on augmentation of prior studies, and 14 on analyzing back earlier testified study results. The SLR also analyzed and identified about eight broad prioritization approaches. These approaches include; genetic-based, coverage, requirement, modification, history, fault, composite, and others which include several approaches. The SLR concludes that even as there were different kinds of approaches, the main objective of TCP in regression testing remains the same, which is to increase fault detection.

On the other hand, in the work of [14], the authors presented a systematic mapping in test case prioritization with a specific focus on TCP studies. It covered the time period between 2001 and 2011. A majority of the reviews recorded in their work were about the approval of different looks into TCP and solution recommendations. The results are the same as reported in the previous SLR with a similar inadequacy. The authors manage to identify 16 studies out of 120, which correlated with the strength and weakness of some prioritization techniques. To locate the finest prioritization method, additional reviews on the analysis of prioritization systems are needed to be done as different approaches are constantly being produced.

In addition to this SLR and one mapping study, there is one significant survey in regression testing domain and two minor surveys. The first significant survey is by Yoo and Harman [7], focusing on regression testing area which includes test suite minimization (TSM), test case selection (TCS), and test case prioritization (TCP). For TCP, they classified the current state-of-the-art approaches into several categories. Those approaches were classified based on requirement, model, coverage covered, historical data, probabilistic calculation, cost awareness, and others which include several minor approaches. The authors analyzed 159 studies covering the time period between 1977 and 2007. The authors also mentioned that experimental designs and evaluations process were still not harmonized but it is getting the attention of some researchers lately. In the first minor survey [15], which only analyzed 19 studies and only focused on coverage-based technique and cost-effective technique. The authors of the survey suggested that having a new technique applied during the early stages of software development life cycle may significantly reduce development cost. The next minor survey [16], identified 90 papers from IEEE publication which summarized that there is a need to have

Table 3
Summary of findings by related studies.

Study references	Covered findings	Similar findings compared to this SLR	Uncovered findings added into this SLR
Singh et al. [13]	- Empirical evidence for several TCP approaches (8 main approaches) - Gaps regarding usage of tools, metrics, and artifact used	- Empirical evidence for TCP approaches - Gaps regarding usage evaluation metrics	- Empirical evidence for other recent TCP approaches - Gaps regarding usage of artifacts on TCP approach
Catal and Mishra [14]	- Trends in TCP approaches - Trends of TCP publication - Trends of metric and dataset used in TCP	- Trends in TCP approaches - Trends of metric used	- Reasons behind the trends of each TCP approaches - Reasons behind the uses of evaluation metric
Yoo and Harman [7]	- Overall overview for regression testing (TCP, TSM, TCS) - Detailed overview of several popular approaches in TCP (4 approaches) - Type of evaluation metric used in TCP	- Detailed overview of TCP approaches - Type of evaluation metric used in TCP	- Detailed overview of other recent popular TCP approaches - Uncovered evaluation metric available with its reasons of creations
Kumar and Singh [15]	- Overview on two TCP approaches (coverage and cost oriented)	- Detailed overview of TCP approaches	- Detailed overview of other recent popular TCP approaches
Kiran and Chandraprakash [16]	- Overall overview for regression testing (TCP) - List of several TCP techniques, strategies, metrics, and algorithm	- Overview of TCP approaches (included techniques, strategies and algorithm used)	- Detailed overview of other recent popular TCP approaches - Reasons behind the uses of evaluation metric

an understanding of cost components and the advantages of having different parameters that could be taken into account. To summarize the background studies in previous related works, Table 3 shows the summary of findings in related studies in comparison to this SLR paper.

From Table 3, it can be seen that only one study provides an empirical evidence for some TCP approaches, while other studies only provide an overview of some approaches. Therefore, there is an incomplete detailed overview such as the reasons behind the trends regarding some TCP approaches, which need to be covered. It also can be noticed that most of the works only summarize the number of usage for each evaluation metric, but did not include in-depth discussions. Work by Singh [13] covered evaluation metrics but the gaps regarding usage of artifacts on specific TCP approach were not well discussed. In short, there are several uncovered findings that can be added to the current SLR work.

3. Research method

With a specific end goal, a structured method to perform this SLR, as shown in Fig. 1, was implemented in order to examine the studies that are related to TCP. The systematic and structured method was inspired by Kitchenham [11,12] and Achimugu [17].

Referring to Fig. 1, there are five main phases within the review protocol, itemized as follows. Research Question, Selection of Repositories, Search Strategy, Study Selection, and Data Synthesis and Extraction. In the first phase, four main research questions were generated to answer the main aim of this paper review. Next, selection of relevant repositories was performed. This is followed by employing a search strategy comprising specifying search strings and search process, which were planned based on the articulated research questions. The output of the search stage was then moved into the study selection phase. In this phase, the outcome of the search process underwent inclusion and exclusion criteria scrutiny to extract relevant studies. Quality assessments were then applied to evaluate the scrutinized studies further. Finally, the last phase dealt with data synthesis and extraction of primary studies utilized for this SLR.

3.1. Research questions and their motivations

This SLR aims to comprehend and review recent experimental evidence with respect to the most recent prioritization approaches in TCP area for further investigation, keeping in mind the end goal is to improvise the ability of present approaches. At the same time, the authors wish to review the empirical evaluations used in each reviewed approach. To accomplish this goal, four research questions with respective motivations were articulated as presented in Table 4.

All these research questions, that frame the reason for undertaking this research are relatively connected and concurrently explored. These research questions are used to answer the extra findings that will be covered in this SLR, as tabulated in Table 3. To be clear, Table 5 maps each research question to its respective extra finding and the finding's significance.

From Table 5, each RQ answers some uncovered findings from previous works, except for RQ 3. For each RQ, the questions are not only designed to answer the uncovered findings, but they are used to cover some extra findings that can be added to this SLR study. The significance of the findings for each RQ has also been detailed out as a guidance to achieve the goal of this SLR study.

3.2. Study strategy

A study strategy is crucial in every SLR to guarantee the broadness of the selected studies. The value of the SLR is generally realized according to the selected primary studies. Strategy for this review depended on these three stages:

a) Literature repository selection
b) Search string identification
c) Study selection process

3.3. Literature repository selection

The authors initiated this selection process by entering 'Test Case Prioritization' as search strings with the exact phrase on Google Scholar database. This database returns 2760 of studies available. From the search result, the authors identified several popular repositories in TCP research area and decided to gather the primary studies originated from recognized repositories. The chosen repositories are:

a) IEEE Xplore
b) ACM Digital Library
c) Science Direct (Elsevier only)
d) Wiley Online Library
e) Springer

The justification behind the selection of these online databases is that IEEE Xplore offers a number of important conference articles and symposium articles, while ACM Digital Library provides

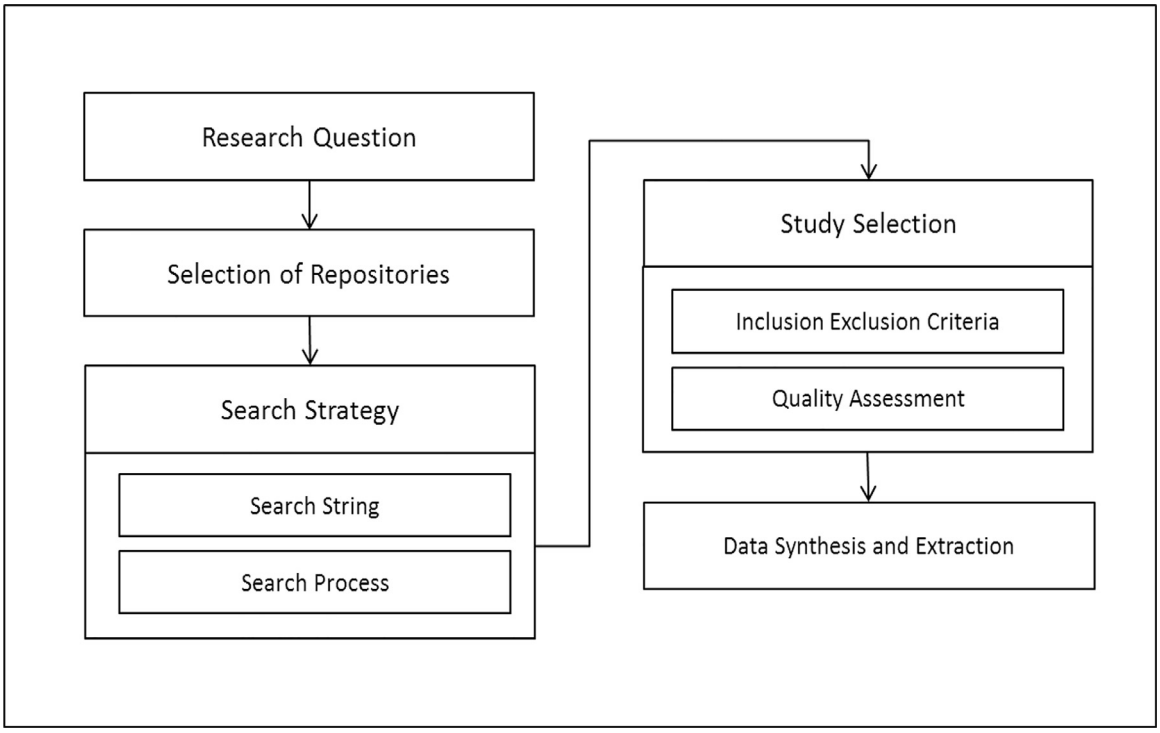

Fig. 1. Phases of review protocol.

Table 4
Research questions and their motivations.

Research questions	RQ statement	Motivation
RQ 1	What are the taxonomies of test case prioritization in regression testing?	
RQ 1.1	What is the research trend of TCP techniques in regression testing?	These research questions focus on characterizing the current domain of test case prioritization. The reason is to know the development of TCP in regression testing throughout the past years.
RQ 1.2	What are the distributions of approaches in TCP techniques?	
RQ 2	What are the differences in terms of approaches for each TCP technique?	
RQ 2.1	What are the descriptions, strength, and limitations of existing prioritization approaches?	The knowledge of differences in approaches is necessary to give a glimpse on how each prioritization approach functions, while the strength and weakness serve as the basis for improvement.
RQ 2.2	How were these approaches applied and how did they affect TCP results?	
RQ 3	What are the processes involved in TCP in regression testing?	This research question can also help to illustrate the basic process of TCP execution in all different approaches.
RQ 4	What are the evaluation metrics and suitable types of artifacts involved in TCP with the reasons for their creation?	This research question helps other researchers to choose which evaluation metric is suitable for their controlled experiment or case study.

more articles from workshops used for the primary studies. The remaining repositories are equally important as they host journal articles that are related to test case prioritization. There are also important journals extracted from IEEE Xplore and ACM Digital Library.

3.4. Search string identification

SLR is a well-known review technique for reviewing the literature with an extensive search aspect of the subject in the discussion from all relevant sources. Therefore, a systemic method to formulate search keywords in this SLR consists of the following steps:

a) Determination of significant terms based on RQs.
b) Determination of equivalent words for significant terms.
c) Determination of keywords in applicable studies.
d) Usage of the Boolean 'OR' and 'AND' operators as an alternative link between terms.

As the focus is to examine related studies in regression testing, TCP area to be precise, the results from previous studies are utilized in order to determine significant studies. The authors inten-

Table 5
Mapping of research questions to uncovered finding with its significance.

Research questions	Uncovered findings answered	Extra findings	Significance of the findings
RQ 1	Reasons behind the trends of each TCP approaches	Distributions of approaches in TCP techniques and its logic	To provide insights in the development of TCP in regression testing within the recent trends.
RQ 2	A detailed overview of other recent popular TCP approaches Empirical evidence for other recent TCP approaches	The strength and limitations of existing prioritization approaches	To provide a glimpse of how each prioritization approach functions, and serve information for improvement
RQ 3	–	Processes involved in TCP	To illustrate the basic process of TCP execution for improvement
RQ 4	An uncovered evaluation metric available with its reasons of creations Gaps regarding the usage of artifacts on TCP approach	Metric most likely used for specific TCP approach	Information of available evaluation metrics and artifacts with the reasons of creation and relation to a specific TCP approach

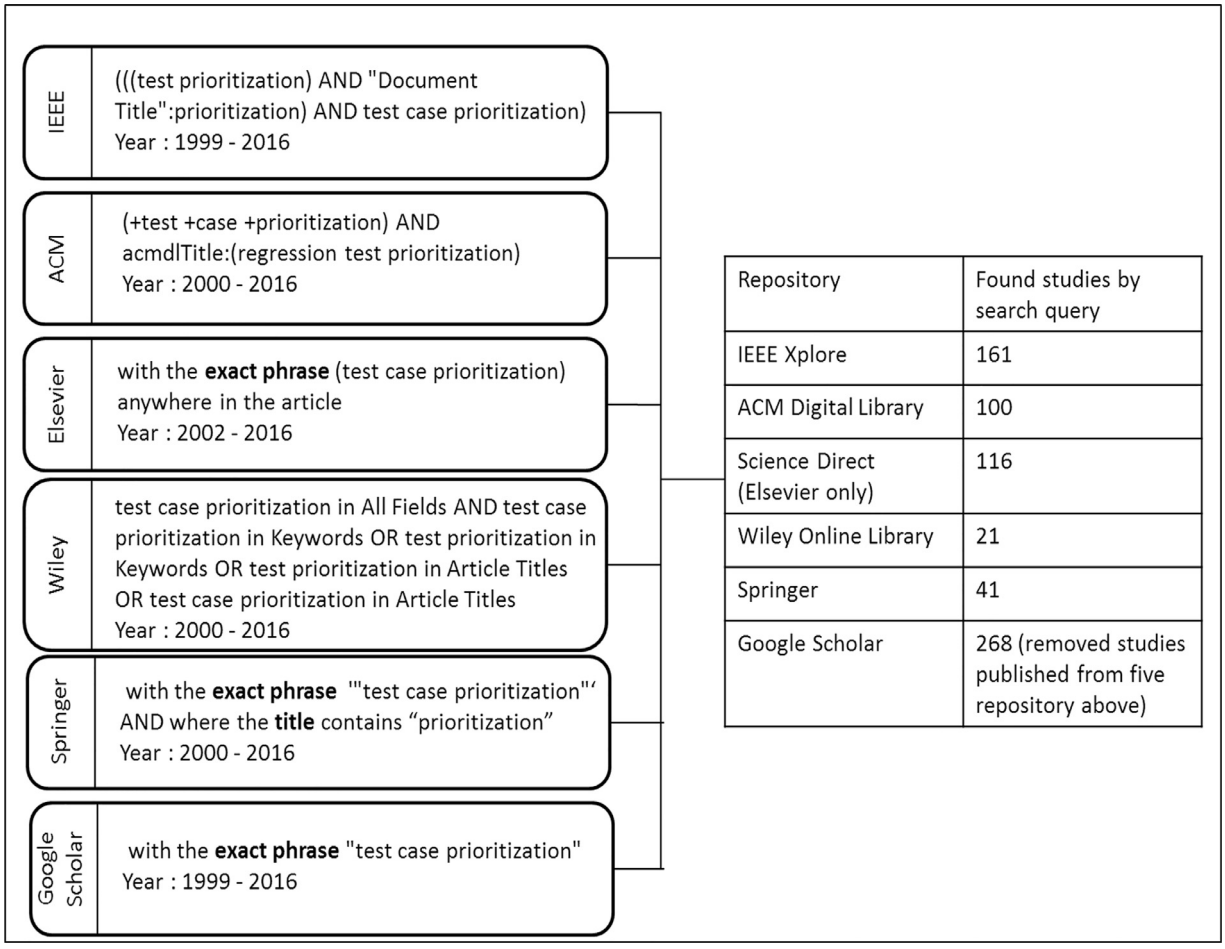

Fig. 2. Search strings for selecting studies from all repositories.

tionally used test case prioritization as the exact phrase in most of the search queries since there are numerous research works that are related to testing test cases in regression testing.

As illustrated in Fig. 2, different search strings were used in each repository. With a specific goal to retrieve conceivable significant reviews, the authors utilized these terms: "test prioritization", "regression testing", "test case prioritization", and "regression test prioritization". It ought to be noticed that if the authors utilized an exact phrase such as "test prioritization" alone, it will return an excessive number of unimportant reviews. Due to this, 'AND' operator was used to link "test prioritization" phrase and "test case prioritization" phrase to incorporate them into an alternative search term. As a result, quite a number of relevant studies were extracted from the combination of the phrases. To refine search outcomes, the authors used 'OR' operator for the phrases that appear in document titles and author keywords. The range of years published was set starting from 1999 until 2016.

3.5. Study selection process

As mentioned in the previous section, SLR requires to be conducted in an appropriate manner in order to produce a high impact review to the research domain itself. The SLR search process was initiated during the selection of the repositories. The first process was to identify several popular repositories in TCP research area. The next stage was a search stage, where, an exhaustive search was performed on all six selected repositories and all the prospec-

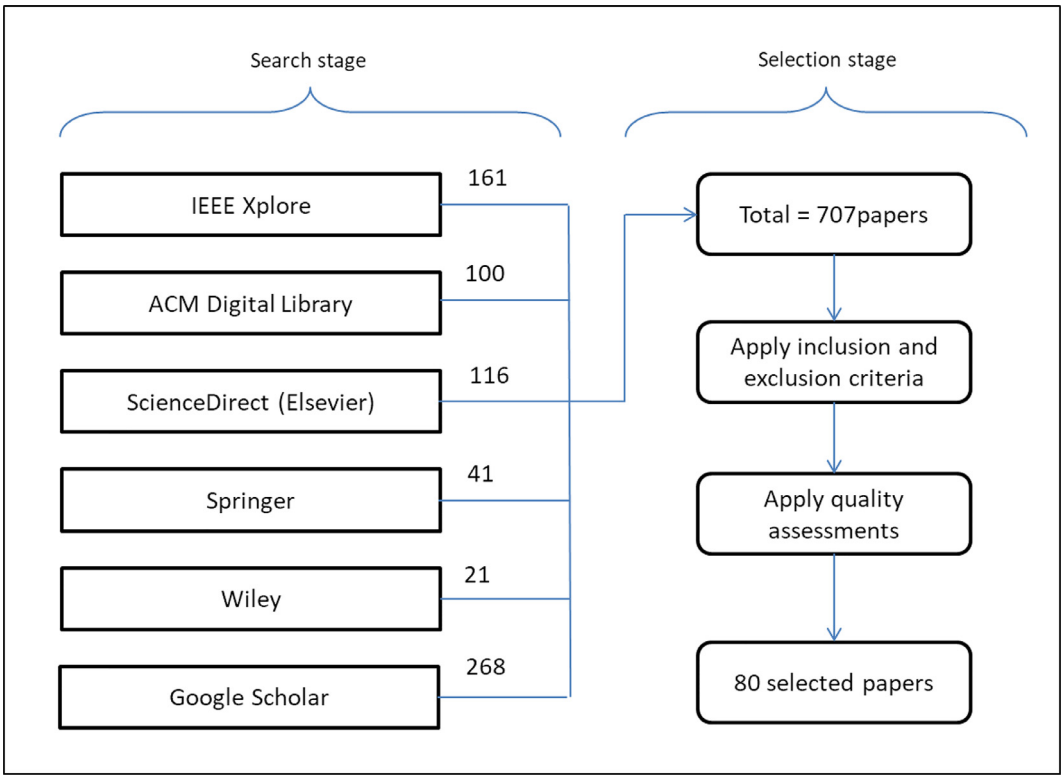

Fig. 3. Search and selection process.

Table 6
Inclusion and exclusion criteria.

Inclusion criteria	Exclusion criteria
English as publication language	Non-English as publication language
Paper focusing on test case prioritization approaches	Paper does not have any relation with TCP approaches
Paper with complete bibliographic information from 1999 to 2016	Paper without bibliographic information
Paper is able to correspond to at least one research question	Identical studies (latest paper is included)

tive papers were assembled together to ease the selection process. Then, all the prospective papers underwent a selection stage in order to choose for relevant papers that were going to be used in the primary studies. Fig. 3 illustrates the search and selection process conducted for this SLR.

From Fig. 3, 707 potential studies were identified from the search stage. To narrow down on the number of the papers to be reviewed, all the prospective studies were required to be justified to get the most significant studies. First of all, the prospective studies were required to go through inclusion and exclusion criteria. This process was essential to remove duplicates and unrelated studies. A detailed overview of the inclusion and exclusion criteria utilized for scrutiny is shown in Table 6. The first criteria to be satisfied were papers printed and issued in English only will be chosen. Studies that were not available in English were removed. Then, their abstract was briefly studied. The paper that does not have any association with research questions were excluded from the major studies list. For duplicated papers that appeared in different copies, the most recent ones would be the most completed and improved copies. They were selected while the others were removed.

After the inclusion and exclusion stage, quality assessment was applied. The quality evaluation of the chosen studies was accomplished by utilizing a weighting approach to examine significant studies that are adequate enough to answer all the RQs. The authors articulated five assessment questions shown in Table 7 to evaluate the comprehensiveness, reliability, and applicability of the nominated papers. Three optional answers with their respective score were given for each question: "Yes" = 1, "Partly" = 0.5, and "No" = 0. Subsequently, various papers were rejected from this quality assessment. Consequently, only 80 papers were chosen for the primary studies. The total scores of these chosen studies is portrayed in Table A1 in Appendix area.

Table 7
Quality assessment questions.

No	Question
1	Were the research objectives stated precisely?
2	Were the planned approaches stated clearly?
3	Was the experimental strategy appropriately designed?
4	Did the experiment apply on a case study or a controlled experiment?
5	Does the exploration enhance the scholarly world?

Upon the completion of this selection stage, 80 studies were identified to manifest the capability to answer all of the research questions derived earlier.

3.6. Data synthesis and extraction method

The principle of data synthesis is to simplify evidence presentation from the nominated papers to ease data extraction process. This is in order to answer all of the research questions. 80 selected primary studies underwent an additional inspection with respect

Table 8
Contents assessment measures.

Nominated paper	Description
Paper references	Paper title, publication year, and sources
Type of paper	Journal article, conference proceeding, symposiums, and workshop
Paper focus	Main ideas, complications, inspiration and purposes
Research methodology	Case study, experimentation, reviews, and literature surveys
Application domain	Depiction of the specific situation
Constraints	Study's limitations for future improvement

Table 9
Data collection for research questions framed.

Research questions (RQs)		Type of data extracted
RQ 1	RQ 1.1	Types of regression testing techniques, studies, and bibliographic references.
	RQ 1.2	Types of TCP approaches solution types used to conduct the techniques.
RQ 2	RQ 2.1	Brief description and test purpose of each approach, formal hypothesis, test strategy, threats to validity.
	RQ 2.2	Formal hypothesis, advantages, and a threat to validity.
RQ 3		Tool/environment for experimental setup, data collection method, process, and evidence type or measure for the effectiveness of TCP approaches.
RQ 4		Evaluation metrics and software artifact type.

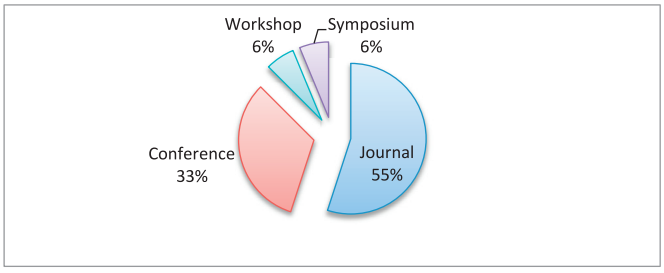

Fig. 4. Percentage of collated studies.

to the content assessment measures as shown in Table 8 to determine the specified matters for each paper.

The point of this measure is to synchronize the primary studies and enhance the assessed papers for clarity. This will help for the purpose of data extraction from papers which response exactly to the research questions. Brief important types of data showing the mapping of synthesized data to research questions are shown in Table 9 below.

4. Result and discussion

This section outlines the results with respect to the research questions. The summary of the primary studies was presented first, followed by each research question, answered in different subsection.

4.1. Overview of primary studies

80 primary studies in total were nominated for this review. From the primary studies, there were 44 journal articles, 26 conference papers, five workshop articles, and five symposium articles. The percentages of the collated studies are presented in Fig. 4, and the total papers issued per year are presented in Fig. 5. As for the overviews of the primary studies, the information is tabulated in Table A2 in Appendix section.

4.2. What is the research trend of TCP techniques in regression testing? (RQ 1.1)

As prioritization on test cases had only gone through test case selection in early studies [3], TCP was then suggested and assessed in a broader context. The first aspect of RQ1 was to determine the current trend of TCP techniques' studies. Referring to Fig. 5, the number of papers published through the year shows a positive increment beginning from 1999 until 2016. It can be concluded that TCP has been recognized as an important element in regression testing among researchers.

In real world scenarios, it is quite hard to realize which tests will actually detect faults. Hence, that is why test case prioritization approach needs to have other approach backups, expecting that a certain number of backup approaches will end in boosting fault discovery in different ways. There are many test case prioritization approaches which have been proposed by researchers. As many as eight broad approaches were described by Singh [15]. All these TCP techniques approaches were originally grouped based on some commonalities such as phases selected, available resources (requirement, test cases), and desired output (time execution, APFD). For example, Requirement-based TCP can be initiated during or at the end of requirement phase, by using the requirement resource itself. Fig. 6, shows other discovered approaches.

As shown in Fig. 6, the authors grouped the TCP approaches into seven main dimensions, which seem to be popular among researchers. These categories are reported in recent mapping studies and they have various common characteristics [14]. However, the authors combined some approaches under 'others-based' dimension, which were not really popular among researchers including topic model based [18], multi-criteria based [19,20], workflow based [21] and others [22–24]. Each approach presents different process and dataset in performing regression testing.

4.3. What are the distributions of approaches in TCP techniques? (RQ 1.2)

The second aspect of RQ1 is related to TCP techniques categories that have been proposed by previous researchers. The distributions for each approach are depicted in Fig. 7, while Fig. 8 shows the most utilized approaches. The list of citations for each discovered TCP approach is tabulated in Table A3 in Appendix section.

The results showed that search-based TCP is the most utilized method among the collated studies. It was used in 25% of the studies. Within search-based TCP itself, there were several algorithms used including Genetic Algorithm (GA) [25–33], Greedy [34,35], Ant-Colony [36–38], String Distance [39] and others [40,41]. Several observations are noted for artificial intelligence (AI) utilization. First, there are many publications on AI application and it is used in solving different problems contexts. Second, empirical data is easily available for AI experimental setup. This encourages researcher in executing and compiling results using search-based approach.

The second largest portion of the reported approach in the primary studies is coverage-based TCP with an 18% distribution

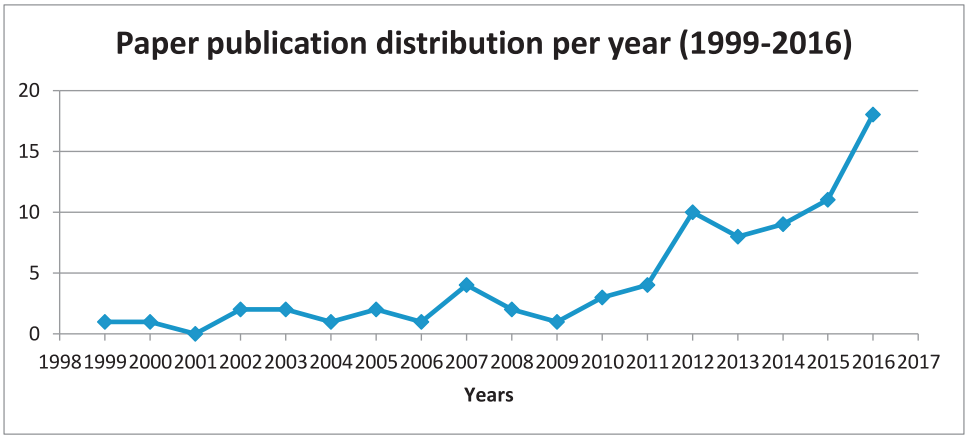

Fig. 5. Number of papers published per year.

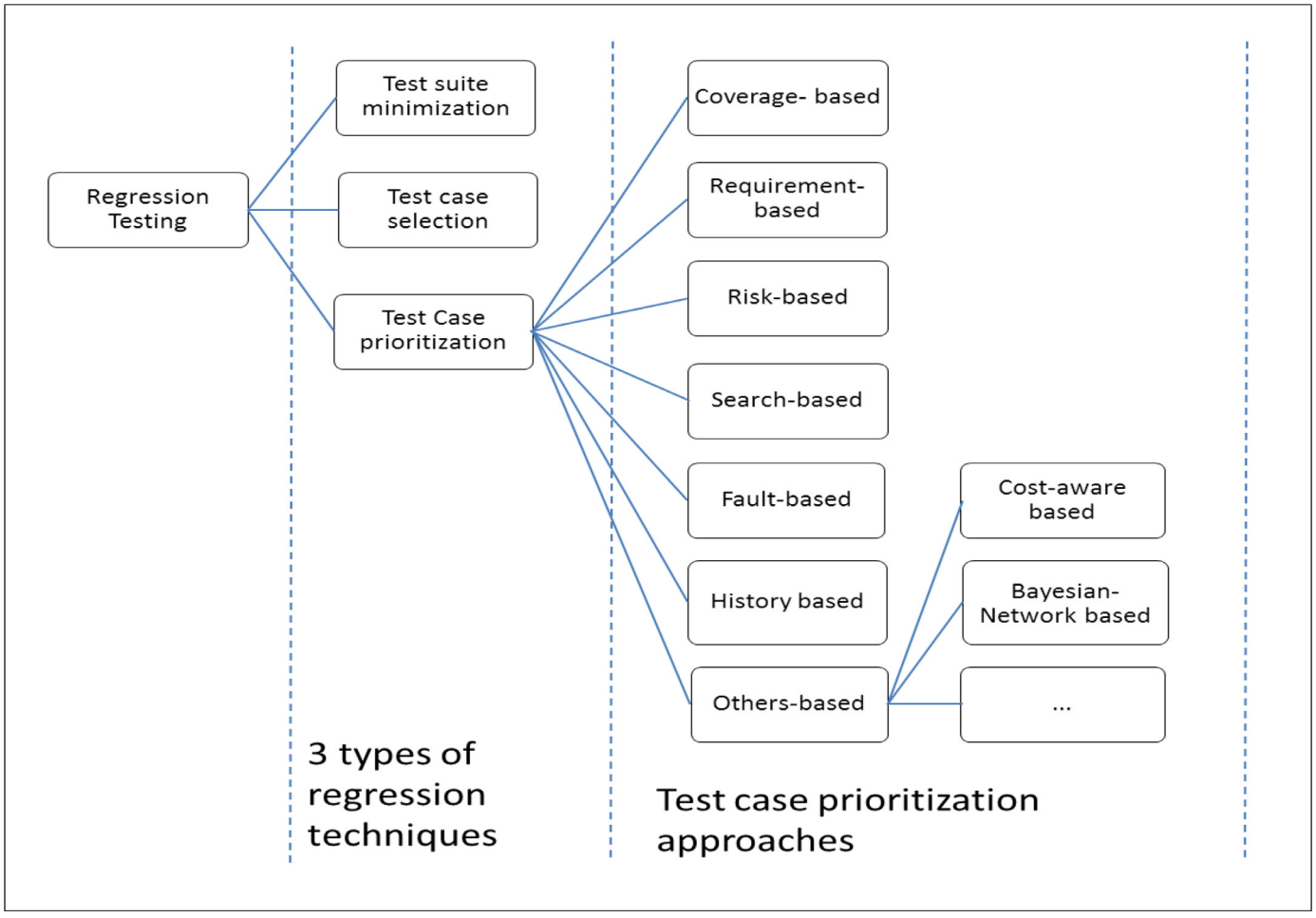

Fig. 6. The taxonomy of TCP.

[3,29,42–49,49–52]. Coverage-based TCP was the first proposed approach that resulted in high utilization until this day. Coverage-based TCP's main idea is to cover a 100% coverage of the system during regression testing, which logically will detect all possible faults within the system. However, to gain a 100% coverage, there is a time cost that needs to be considered, as testing execution time will be dragged and more resources will be required. This is the motivation that has led more publications on coverage based to improve it from time and resource-related costs aspects.

Fault-based TCP comes as the third most utilized approach reported in the primary studies [53–60]. The authors believe this approach hit quite a number of publication since fault-based prioritization approach was mentioned by Rothermel [3], in their work in TCP. The fault-based approach is able to construct a specific test suite that aims to detect the exactly estimated faults [58], which may ease researchers to target on faults that should be detected.

Requirement-based and history-based TCP share fourth most utilized approach in TCP studies with a 9% distribution respectively. From the primary studies, requirement-based TCP as ap-

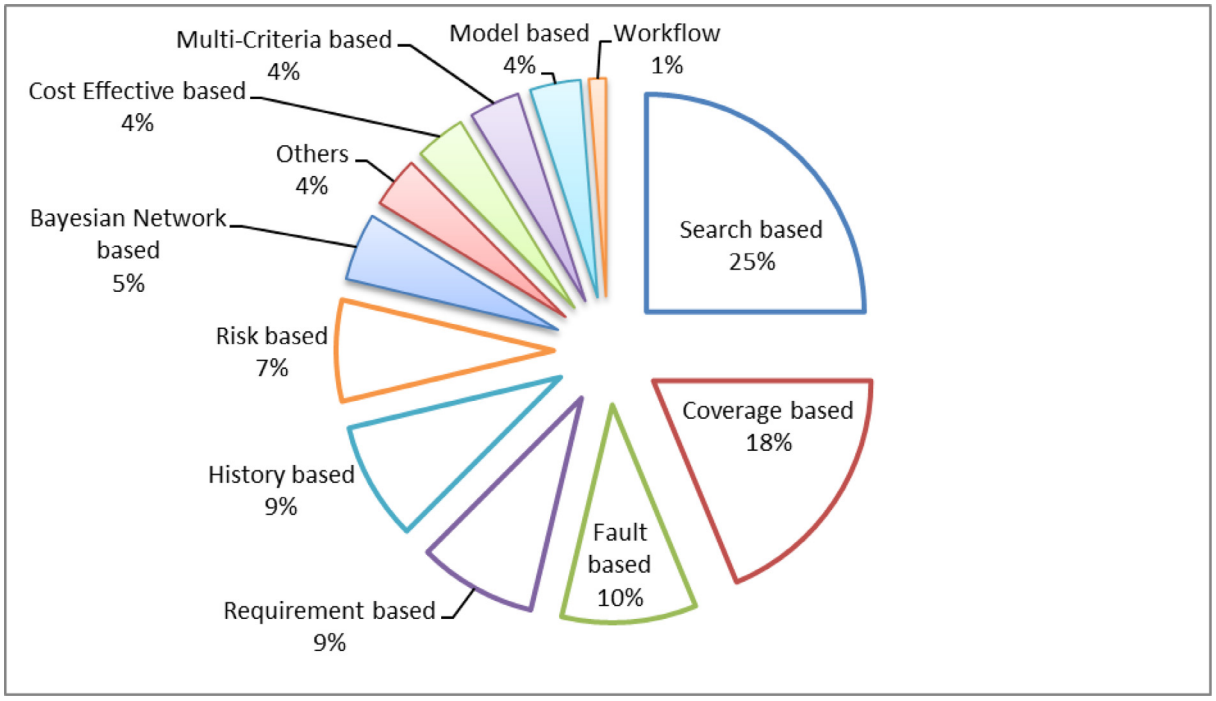

Fig. 7. Percentages of collated studies approaches.

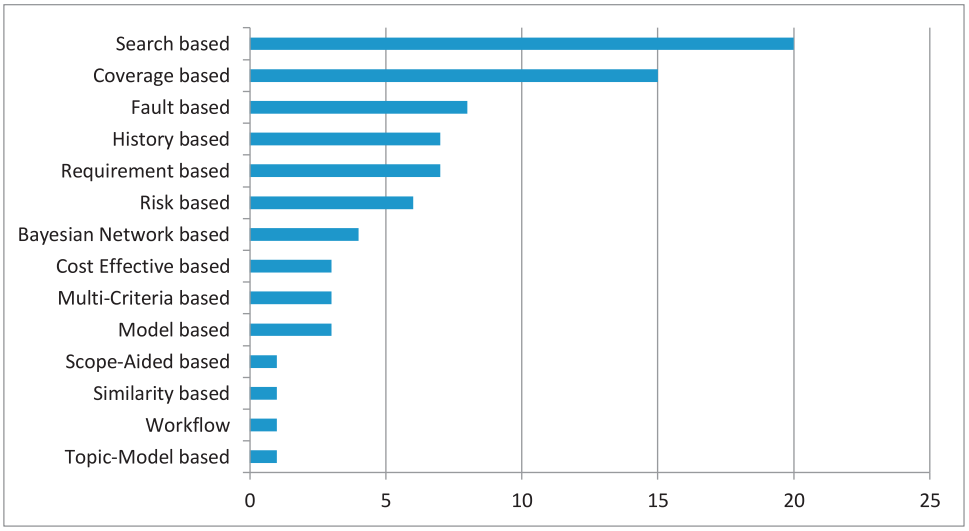

Fig. 8. Most utilized approaches.

peared in [52,61–67] utilize requirements information to prioritize test cases. A system is built based on its requirements. Therefore, by using the requirements information, it may possibly aid to classify several crucial test cases more than just by using code-related data. This statement may have been the factor for a requirement-based approach to be considered. On the other hand, history-based TCP appeared in [38,68–73], which utilize previous data or historical data in order to prioritize test cases. Based on the observation in the primary studies, history-based utilization factors has to do with its data that are easily available for any search-based experimental setup.

Risk-based TCP were reported with a 7% distribution in the primary studies, which appeared in [65,66,74–77]. They can be placed under two categories, either requirement risk or model risk. For requirement related risks, they used requirement to calculate their risk index for each requirement, while for model related risks, they calculated risk for each node in a model diagram (i.e. activity diagram, state transition diagram, etc.) and prioritize test cases based on these calculated risks. On top of that, Bayesian Network-based TCP appeared in [78–81], which is almost similar to model-based TCP, with a 5% distribution in the primary studies. Instead of using the model information only, Bayesian Network-based TCP foresees the possibility of each test case to discover any error by means of different accessible data [81].

For others-based TCP which did not seem to be popular, they were available in the literature, in the quantity of four or fewer. As mentioned before, the authors combined all these approaches under others-based TCP which are not really popular among the researchers such as cost-effective based [60,73,82], topic model based [18], multi-criteria based [19,20], workflow based [21], and others [22–24]. The authors also tabulated all the number of the papers for each approach depicted in Fig. 8. It shows the hierarchy

Table 10
Overviews of TCP approaches.

Approaches	Description
Search-based	- Search based approach can vary in terms of implementation - For example, Greedy algorithm was found highly related to coverage-based objective [35,45] - The application of AI strategy may differ, based on the selected test suite, input criteria, fitness function, and others
Coverage-based	- This approach is a fundamental testing approach and inspects the code directly [43] - Code, statement, branch, or function coverage lies within two main categories: total and additional coverage [7,45]
Fault-based	- Fault-based prioritization approach was initially mentioned by Rothermel [3] - Fault-based strategies produce a sequence of test cases to detect targeted faults [58], which may ease researchers to have a target on which faults that should be detected
Requirement-based	- Requirement-based utilizes requirements information during requirement elicitation such as customer priority, error prone value, volatility value, and execution difficulties for prioritization criteria [61,67]
History-based	- History-based uses history data as an input to execute test case prioritization - Based on the observation in the primary studies, history-based utilization factors has to do with its data that is easily available for any search-based experimental setup
Risk-based	- This approach is mainly used in a project that typically concerns on risk values related to the software to be developed [61] - Based on the primary studies, risks can be calculated from requirement itself and system model
Bayesian-Network-based	- Utilization of Bayesian Networks foresees the possibility of error to be discovered in every single test case by the means of different accessible data [81]
Cost-Effective-based	- Cost-aware test case prioritization technique was reported by Huang [73], which integrates historical data to reduce testing cost
Topic-Model-based	- Technique that uses linguistic data such as identifier markers, remarks, and string literals to help differentiate their functionality [18]

of the most utilized approaches in TCP and within it, search-based and others based approaches were separated into their respective category for easier analysis.

4.4. What are the descriptions, strength, and weakness of present prioritization approaches? (RQ 2.1)

The overview of description for each TCP approach as illustrated in Fig. 7 is listed in Table 10. The overview of these approaches are important, as it allows researchers to gain insights on how each prioritization approach works. However, for strength and limitations of each approach in TCP, they are detailed out in Table A4 in Appendix section. The overviews of these strengths and weaknesses serve as a motivation for researchers to look for a potential research area that can be focused on, in the future.

4.5. How were these approaches applied and affect TCP results? (RQ 2.2)

In order to answer this question, the primary studies were examined deeper into their experimental setup and results. For each approach, the authors elaborated certain work to give a view on how the approaches were applied and their effect on the TCP process. The next sub-sections will discuss each approach.

4.5.1. Coverage-based TCP

This approach is a fundamental testing approach which inspects the code directly [43]. The fundamental measures such as function, branch, and statement were the most widely used criteria in coverage approach [45]. All of the criteria were evaluated and can be categorized into two main coverage-based prioritization approaches [7]. Those coverage based are total and additional coverage based.

The total coverage approach organizes the test cases which manage to cover the most portions of the system accordingly by the mean of some preferred criterion. In the event that the test cases bear a similar coverage area, the conflict will be resolved arbitrarily. While for additional coverage approach, the main idea of this approach is to attain a complete coverage earlier. Therefore, this approach differs from the first highest coverage test case. The next test cases will consider the highest coverage of uncovered code or statement.

4.5.2. Requirement-based TCP

A system is built based on its requirements, therefore, by using the requirements information it may possibly aid to classify several crucial test cases more than just by using code-related data. The work in [67], assumes that prioritization of requirements for test (PORT) is a process of establishing the value of importance of a software requirement. Evaluation of PORT is based on four critical factors: client priority, error prone value, volatility value, and execution difficulties. Recent study by Srikanth [61], reported that a combination of two or more factors may produce a better result than a single factor prioritization in terms of testing effectiveness.

The work in [62], proposed a prioritization algorithm based on traceability, completeness, the impact of a fault in requirements, changes in requirements, customer priority, and developers views. The approach was claimed to yield a better result in term of fault detection rate in correlation with randomly ordered approaches. A prioritization approach that uses static data and requires a precise mapping from discrete test case may not be reliable.

4.5.3. Search-based TCP

Search-based prioritization approach has quite a number of implementation algorithms such as Genetic Algorithm (GA) [25–32], Greedy [34,35], Ant-Colony [36–38], String Distance [39] and others [40,41]. Experiment by Li [35] stated that GA application approach works poorer when compared to a greedy algorithm on computer-generated data. However, the application of a search-based algorithm may differ, based on the selected test suite, input criteria, fitness function, and others. While the collected result showed the major benefit of GA application in TCP approaches, there are certain disadvantages that exist, such as, execution time is a vast anxiety for GA applications and they are typically slow in the process of completion [28].

4.5.4. Risk-based TCP

Risk-based prioritization approach was mainly used in a project that typically concerns on risk-related values with the software to be developed [61]. Due to its prominence and practicality, some

researchers used requirements risk information in TCP process to categorize crucial test cases that were expected to distinguish the faults related to the system's risks [66,75]. The work of Hettiarachchi [65] proposed five main steps to prioritize test cases by using requirement risks values. Requirements priorities and risk values were calculated within the first four steps, while the test case prioritization process only started upon obtaining the result from the first four steps. It claims to have the ability to detect faults earlier since it is capable of being executed prior to code availability.

4.5.5. Fault-based TCP

Fault-based strategies try to produce a sequence of test cases to detect certainly targeted faults According to Rothermel [3], a specific fault can be revealed when executing a particular statement. There are also possibilities that the particular statement can reveal other error in other test cases. In fault-based prioritization demonstrated by Yu and Lau [59], they proposed Fault Adequate Test size (FATE) focusing in the specification. FATE is an effectiveness metric used to determine the size of the minimal fault adequate subset. FATE values range between 0 and n. The lower the FATE, the higher is the chance of detection of all aimed faults. Their test case prioritization can be implemented before the code information is available.

4.5.6. History-based TCP

History-based prioritization uses history data as an input to execute test case prioritization. Work presented by Khalilian [71], claimed that their improved method in prioritizing test cases by including historical test data manage to accelerate fault detection rate. The improved method is an extension of the existing history-based TCP approach put forward by Kim and Porter [72]. Their suggested work analyzes the weight for each test case by utilizing history data including the count of executions which revealed a fault. This proposed technique was measured using AFPDc utilizing Siemens suite and Space programs. Their technique showed an improvement over their previous method of performing selection probability calculation. It is agreed that historical based is an effective TCP approach, however, there are some limitations to this approach, including limited historic information availability and a small number of defect information.

4.5.7. Bayesian-Network-based TCP

For Bayesian-Network-based (BN) approach, a Bayesian Network's model foresees the possibility of each test case to discover any error by means of different accessible data [80]. Work presented by Mirarab and Tahvildari [80], proposed a BN approach by addressing three TCP difficulties: (1) determining the different collection of proof from the coding part, (2) incorporating all the data into a sole BN model, and (3) utilizing probabilistic interpretation to assess the possibility of accomplishment. As the results, this case study claimed to perform better than a random technique. However, standard execution of BN is not at the finest as error-prone data are not included and does not take advantage of any feedback.

4.5.8. Cost-aware-based TCP

A cost-aware-based test case prioritization technique was reported by Huang [73]. In their proposed work, Modified Cost-Cognizant Test Case Prioritization (MCCTCP) was utilized to estimate the units of faults discovered per unit testing rate by utilizing Genetic Algorithm (GA). Their experimental setup was implemented on *flex* and *sed* UNIX utility programs. They compared their techniques against random, optimal, historical fault, time-aware, total functional coverage, and cost cognizant total function coverage. Their prioritization was claimed to have a better value in average fault detection rate if the number of generations of software code is higher.

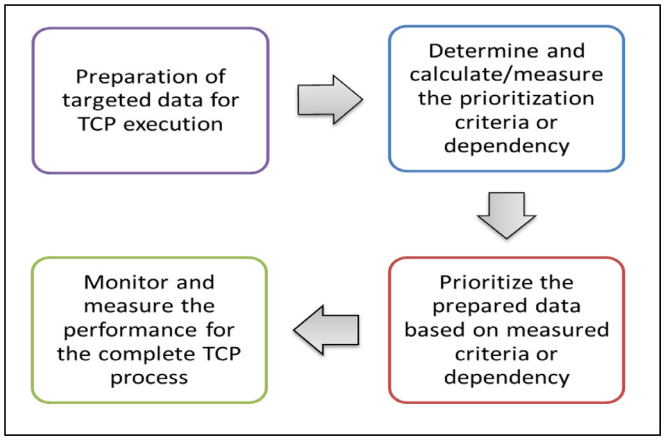

Fig. 9. Standard flow of TCP technique.

4.6. What are the processes involved in TCP? (RQ 3)

Software engineering highly concerns on how the engineering processes are applied to software development in a systematic way. Therefore, it is necessary to have this RQ to be investigated. In order to answer this question, the primary studies were examined further into their experiment flow. Every experiment should have their own process, however, in this SLR, processes involved in TCP are only highlighted from several studies. The reason behind this is because, numerous works exhibit similar process flow, with the only notable difference in terms of the addition of one extra step to an existing process flow. From the primary studies, the authors determine 19 processes proposed by several researchers. All these 19 works were selected as their processes were clearly presented. There were also other clearly stated processes in other studies, but they were not covered by the RQ's. Each of the process is detailed out in Table A5 in Appendix section. Based on the 19 processes realized, the authors illustrate the basic flow of TCP process as shown in Fig 9.

As shown in Fig 9, TCP process starts with the preparation of targeted data. Even though there is no single paper stating clearly that the TCP process starts with the preparation of data, it is compulsory for any experiment or research to identify which information or data that will be used. The data or information in TCP can be in the form of requirement statement, system models, and source code. The process is followed by, determining and calculating prioritization criteria or dependency based on the data chosen. From the primary studies, Bryce [49] and Eghbali [41] determine, first, the test cases that have most coverage and which are unique in coverage-based approach. While in risk-based approach in [65,66,74], the risk value related to each test case was defined and calculated before any prioritization was performed. After prioritization criteria determination stage, next is prioritization process. Test case prioritization activity can only be started after the criteria have been determined or measured. As in requirement based approach in [61,67], prioritization of test cases start after the requirement criteria such as customer priority is determined and calculated. In the work of Ma [63] and Wang [56], TCP process can be repetitive until all requirements, test cases, or prescribed faults are fully covered or satisfied. Finally, the result is monitored and the performance of a complete TCP process is measured. However, each approach may start this process in a different timeline within

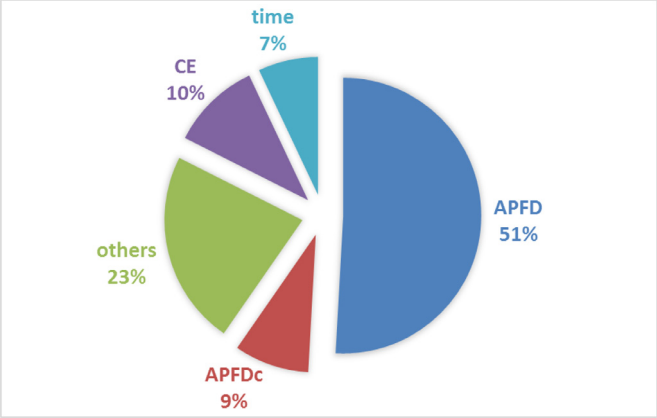

Fig. 10. TCP evaluation metrics type.

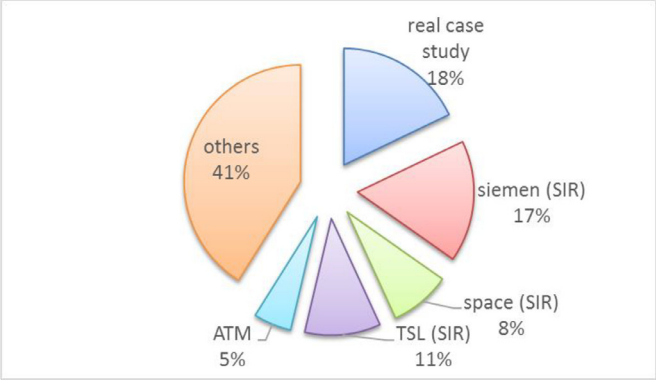

Fig. 11. TCP related artifact assessed.

a project schedule based on its chosen data. For requirement statement data, TCP can start during or right after requirement phase. For system models' data, it starts right after the design phase is completed and source code needs to wait until it is available.

4.7. What is the evaluation metric used in TCP along with the software artifact? (RQ 4)

To answer this research question, this section is divided into two sub-sections, comprising evaluation metric and software artifact sub-sections, with the relationship among them.

4.7.1. Evaluation metrics

It is essential for any approaches proposed in test case prioritization to perform metric measurement to assess their effectiveness. Evaluation metric is important to measure the efficacy of any proposed TCP approaches in prioritizing test cases and to benchmark its effectiveness against other existing approaches. Fig. 10 illustrates current widely utilized evaluation metrics in TCP area.

From Fig. 10, it shows that the most widely used metric is APFD with a 51% distribution, followed by Coverage Effectiveness (CE) 10%, APFDc (APFD with cost consideration) 9%, time execution 7%, and others 23%. Average Percentage of Faults Detected (APFD) is evidently the most utilized metric favored by researchers in the primary studies. This metric was originally introduced by [3] in early TCP research, and later used massively by other researchers [18,19,21,23–32,36,37,39,42–52,62,63,65–67,74,75,82–88]. APFD is a metric used to quantify on how quick an arranged and optimized test suite can discover defects [3,89]. The result of APFD ranges from zero to 100, where a greater value indicates a better fault revealing rate. From this metric, researchers have come up with other metric that evolves APFD metric by considering other factors such as cost or time constraints to fulfill different objectives of prioritization.

Coverage effectiveness (CE) comes as the second most utilized metric [18,21,23,24,28,39,48,49,55,56,58]. CE is a metric that integrates the size of test suites and the coverage of each test case. It is the ratio between the size of the whole test suite and the coverage of reordered test suite that reveals all faults or meets all requirements [90]. CE values range from zero to one, where a higher value indicates a better effectiveness in coverage. Next is APFDc with a 9% distribution, which is an APFD comparable metric with an inclusion of a cost factor. APFDc has been utilized by researchers [15,23,28,49,56,69,70,73,82,83], to measure the effectiveness of fault detection rate over cost. This metric is the most widely used metric in history and fault-based approaches, since history-based concerns on improvising previous testing that includes cost factor, while fault-based focuses on prescribing fault that are most likely to be the costly one.

Time execution metric is used by 7% of the distribution of researchers [18,25,28,36,37,39,58,88] in this SLR. Time execution metric is primarily used in search-based TCP approach to verify the effectiveness of a proposed algorithm in reducing TCP time. Finally, other metric comprises a 23% distribution, which includes several types of metric available. Most of the metrics are adapted from APFD, which has been turned into ASFD, WPFD, TSFD, APBC, APDC, APSC, and NAPFD to answer different TCP objectives.

To further support the significance of all evaluation metrics, ANOVA test has been utilized by several researchers [18,22,32,39,42,44,46,50,51,54,61,79,81,82]. Analysis of variance (ANOVA), is a statistical method that was introduced by Fisher [92]. ANOVA is a statistical test that aims to evaluate the difference of the means of three or more groups. The groups' data must be in numerical type. ANOVA hypothesizes the findings with both null hypothesis and alternative hypothesis. A null hypothesis is where all treatments are having equal values while alternative rejects the null hypothesis. ANOVA test in TCP hs been used to verify the impact of the proposed work by measuring a statistical significance of the investigated approach, by comparing any metrics, such as APFD or execution time.

4.7.2. Artifact

An artifact is required in completing a controlled research in testing practices. Artifacts can be in the form of test cases, software, coverage information, software requirements, fault records, history evidence, and others. The utilization of artifact depends on the type of research and its suitability to the experiment. A careful examination of the artifacts utilized by different TCP approaches is presented by Singh [13]. The study highlighted a number of artifacts assessed by researchers. In this SLR, the distribution of artifacts utilized is presented along with their usage in different TCP approaches. Fig. 11 shows the TCP-related artifacts assessed by researchers.

On the subject of the artifacts assessed, our results conform the findings reported in [13], where a 50% distribution of the artifacts are accessible with no restriction from Software Infrastructure Repository(SIR) [91]. In this SLR, a 36% distribution of the software artifacts has been extracted from SIR, as illustrated in Fig. 11. Meanwhile, real case study has been used in several studies [42,43,47,49,52–55,61,63,65,67], which is considered as an artifact in this SLR. The artifacts from SIR are mainly divided into three types comprising, siemen, space, and TLS. Siemen set consists of seven benchmark program which is written C programming language.

Table 11 shows the list of programs developed by Siemens Corporate Research, which constitutes a 17% distribution of ar-

Table 11
Siemen benchmark program.

Program	LOC	Description
tcas	173	Aircraft collision avoidance system
schedule2	374	Priority Scheduler
schedule	412	Priority Scheduler
tot_info	565	Information Statistics Measure
printokens	726	Lexical Analyzer
printokens2	570	Lexical Analyzer
replace	564	Pattern Replace

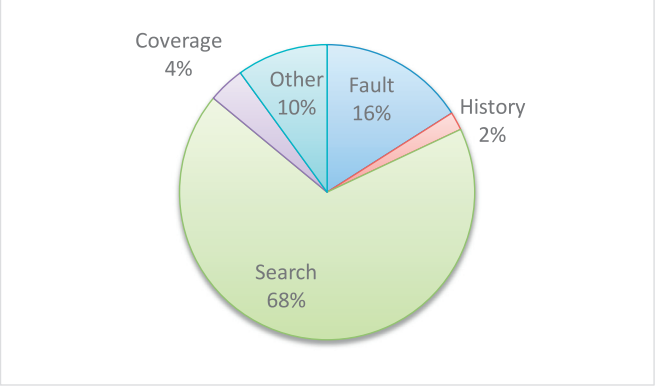

Fig. 12. Utilization of SIR programs in TCP approaches.

tifacts assessed by researchers [3,13,28,34,39,44,45,66,88]. The next popular SIR program among researchers is Test Specification Language (TSL) with an 11% distribution, as cited in [13,23,24,27,28,44,58,73,82,88]. This category is named TSL after a TSL tool that is used in executing the program. TSL program consists of four programs which are flex, grep, gzip, and make, where all of them are UNIX utilities. The last SIR is a 'Space' program with 9564 lines of code (6218 executables), which makes up 8% of the total artifacts assessed and applied in [13,27,28,33,34,44,45,71]. The space program has been used to validate the content within a file and functions as an interpreter for an array definition language (ADL) grammar. In short, SIR provides information for each of C object, excluding requirement specification documentation. Overall, SIR programs have been widely utilized in TCP approach, for studies that are not related to requirements. Fig. 12 shows the utilization percentage of SIR program across different TCP approaches.

From Fig. 12, we can see that 68% of utilization of SIR programs have been utilized under search-based TCP approach [25–40,93], while 16% of the studies have utilized SIR programs in fault-based TCP [56–60]. Other approaches that utilize SIR include, coverage-based 4%, history 2%, and others 10%. The reason behind this distribution of studies concerning utilization SIR is because SIR only provides artifacts in the form of source code, test suites, fault information, coverage information, and some history evidence. An artifact such as requirement is not provided, which explains why requirement-based TCP does not utilize SIR programs. Since SIR does not provide such information, a real case study or other controlled experiment has been used as shown in Fig. 11.

5. Research finding

In software development, prioritizing test case is an essential activity in testing phase [3,94]. At the point when client hopes are high, promised time is short, assets are constrained, and the developed software must have the capability to fulfill the software requirements with fewer faults, TCP technique is beneficial to reduce the cost, time, and software fault since TCP itself concerns to order test cases for early optimization in fault detection. A detailed approach of TCP is very important, not only to be used to optimize the test case execution, but also to aid project manager to organize project deliveries, and maybe to make some necessary adjustment. Therefore, the TCP impact in regression testing must be emphasized more.

In answering RQ1, all of the primary studies were used to answer this research question. As for the result, it can be said that TCP approaches are still broadly open for improvements. There were positive increments of TCP publication starting from 1999 until 2016. New approaches in TCP are introduced constantly every year, and implementation of artificial intelligence element has been a trend among researchers in the latest study. However, other approaches have their own supporters, with their own advantages. For example, in other approaches, a multi-objective or criteria technique has shown quite a number of supporters in a recent publication [95,96] as it has the capability to tackle two or more different kinds of objectives in one prioritization. This approach can also be easily combined with other techniques, which makes it a more interesting and promising approach. Therefore, it can be concluded that TCP is recognized as an important element in regression testing among researchers currently, as it has the capability to increase the effectiveness of testing in terms of fault detection rate, cost, and time.

For RQ2, 42% of the primary studies were examined thoroughly. As a conclusion, each approach has specified potential values, advantages, and limitation. The inputs and dataset type play an important role in the determination of their advantages and limitation. For example, the requirement-based approach uses customer priority during requirement elicitation as inputs to prioritize and generate a test case. Risk-based may also use requirement risk as one of the inputs to execute prioritization process. This indicates that both may have their own advantages against other approaches in terms of TCP execution starting point since both may start prior to code availability. The differences in term of strengths and limitations of these approaches are required to give a hint on how TCP approaches function and serve as a motivation for any changes in the future.

For RQ3, 19 out of 80 primary studies were evaluated regarding their experimental setup. The implementation of TCP process shows a significant role in certain project (RQ3). TCP techniques benefit project managers in adjusting their project schedules in order to counter the constraint that exists within the project development process. Variations in the starting point of TCP process among the approaches provide a different timeline and benefit to project manager to choose which approaches suite with the project schedule and available resources.

For RQ4, all of 80 primary studies have been examined thoroughly. The evaluation metrics utilized in these primary studies with the reasons behind their creations have been covered. APFD metric remains as the main metric used in all TCP approaches. However, there is quite a number of new metrics that have been introduced, which is adapted from APFD to support different objectives in different studies. As for the artifacts, the evidence suggests that programs extracted from SIR remains the most popular choice among researchers. However, SIR does not support requirement and model based TCP approaches.

6. Threats to validity

This SLR has some limitations recognized that may threaten its validity. Similar to previous reviews, the potential threats of this systematic review are associated with an imperfect collection of primary studies and imprecise data synthesis and derivation.

6.1. Selection of primary studies

In Section 3, the authors presented the research method in detail in the order used to select the relevant studies. However, the authors cannot completely ensure that the authors have gained all the accessible reviews related to TCP to answer all RQs. There are two possible issues. Firstly, a possible issue is the difficulty in finding appropriate search strings for different repositories. In Section 3.4, detailed explanations were given for the process of selection of repositories and search strings were used to find relevant studies to be used in this SLR. However, there might be other significant studies which may use other keywords. Secondly, some significant or related studies might have existed in non-English publication. It misses out the relevant non-English studies although it cannot be avoided.

6.2. Imprecise data synthesis and extraction

Imprecise data synthesis and fragmentary extraction process from the primary studies could be the next potential threat to the validity of this SLR. This may be due to unsystematic data extraction or invalid classification of data. To reduce this problem, manual scrutiny was applied to reduce the possibility of inaccurate data extraction by focusing on the data elements collected from the selected studies represented in Section 3.6. Furthermore, a set of specific quality assessments were applied to avoid inaccurate inclusion of desired studies.

6.3. TCP related field

Within regression testing, there are three techniques comprising, TCP, TCS, and TSM. Test case selection (TCS) is almost similar to TCP as the idea behind TCS is to localize fault, select related test cases, and execute it as a suite. However, in this SLR, the authors did not discuss the current trend of TCP in supporting fault localization issues which may be a potential threat to the validity of this SLR. To avoid this, the authors suggest that this issue could be discussed in future in Section 7. As an overview, TCP does support in fault localization as demonstrated in several recent works [97–100]. A TCP approach may also be turned into TCS after prioritization is completed and selection can be made based on prioritization result. Work by [97] proposed a technique known as diversity maximization speedup (DMS), which ranks up program unit while performing a fault localization for TCP and the results demonstrated that the technique was able to realize the aimed cost.

7. Conclusion

As the purpose of this study was to classify and criticize the current state and trend of test case prioritization approaches, SLR scheme was used to conduct this study. With this SLR scheme, some applicable RQs were formulated according to the aim of this study. This research was conducted through finding, classifying, evaluating, and understanding all of the primary studies. The motivation of this review was to discover any areas or fields that are likely to undergo any sort of improvement via systematic assessment of applicable and important studies in TCP approaches.

During the review process, the appropriate primary studies were recognized and evaluated. The data pulled out from the studies were synchronized. The authors structured the outcomes into tables and figures, which are intended to ease the understanding among the different research groups that work on the same TCP area. From the study, it was discovered that:

a) There were quite a number of prioritization techniques that exist and improvements are still required.
b) AI application is quite popular since it can be used for different problems and the empirical data availability for AI experimental setup is high.
c) The data or information classification in TCP such as requirement statement, system models, and source code needs to be investigated further in future.
d) The specific time frame for the execution process of TCP for each approach needs to be detailed out.
e) Early prioritization in the early stage of a system development life cycle is worth to be investigated further to ease project manager and development team in making the necessary adjustment.
f) Human involvement in decision-making and estimation needs to be changed into a computerized and automated reasoning to reduce human error.
g) How does TCP support fault localization compared to TCS? This issue could be an interesting point to be discussed in the next work.

Acknowledgements

The work is fully funded by Fundamental Research Grant Scheme, vote number 4F836 and Research University Grant Scheme, vote number 11H86 under the Ministry of Education Malaysia - KPM. We would also like to thank the members of Embedded & Real- *Time* Software Engineering Laboratory (EReTSEL) and Software Engineering Research Group (SERG), Faculty of Computing, UTM for their feedback and continuous support.

Appendix

This appendix section contains Tables A1–A5.

Table A1
Result quality scores of selected studies.

Paper Refs.	Q1	Q2	Q3	Q4	Q5	Score
Rothermel et al. [3]	1	1	0.5	0.5	1	4
Yoo and Harman [7]	1	0	1	0	1	3
Singh et al. [13]	1	0	1	0	1	3
Thomas et al. [18]	1	1	1	0.5	1	4.5
Sampath et al. [19]	1	1	1	0.5	1	4.5
Sanchez et al. [20]	1	1	0.5	0.5	1	4
Mei et al. [21]	1	1	1	0.5	1	4.5
Fang et al. [22]	1	1	1	0.5	1	4.5
Miranda and Bertolino [23]	1	1	1	0.5	1	4.5
Korel et al. [24]	1	1	0.5	0.5	1	4
Maheswari et al. [25]	1	1	1	0.5	1	4.5
Lou et al. [26]	1	1	0.5	0.5	1	4
Yuan et al. [27]	1	1	0.5	0.5	1	4
Catal [28]	1	1	0.5	0.5	1	4
Kaur and Goyal [29]	1	1	1	0.5	1	4.5
Jun et al. [30]	1	1	0.5	0.5	1	4
Sabharwal et al. [31]	1	1	0.5	0.5	1	4
Do et al. [33]	1	1	1	0.5	1	4.5
Deb et al. [32]	1	1	1	0.5	1	4.5
Li et al. [34]	1	1	1	0.5	1	4.5
Li et al. [35]	1	1	0.5	0.5	1	4
Solanki et al. [36]	1	1	0.5	0.5	1	4
Gao et al. [37]	1	1	0.5	0.5	1	4
Noguchi et al. [38]	1	1	0.5	0.5	1	4
Ledru et al. [39]	1	1	1	0.5	1	4.5
Jiang et al. [40]	1	1	1	0.5	1	4.5
Eghbali et al. [41]	1	1	1	0.5	1	4.5
Di Nardo et al. [42]	1	1	0.5	1	1	4.5
Nardo et al. [43]	1	1	1	1	1	5
Hao et al. [44]	1	1	1	0.5	1	4.5
Hao et al. [45]	1	1	1	0.5	1	4.5
Zhang et al. [46]	1	1	0.5	0.5	1	4
Miller [47]	1	0	1	0	1	3
Fang et al. [48]	1	1	1	0.5	1	4.5
Bryce et al. [49]	1	1	0.5	0.5	1	4
Bryce et al. [49]	1	1	1	0.5	1	4.5
Jones et al. [50]	1	1	1	0.5	1	4.5
Leon et al. [51]	1	1	1	0.5	1	4.5
Krishnamoorthi et al. [52]	1	1	1	1	1	5
Tahvili et al. [53]	1	1	1	0.5	1	4.5
Alves et al. [54]	1	1	1	0.5	1	4.5
Mei et al. [55]	1	1	1	0.5	1	4.5
Wang et al. [56]	1	1	0.5	0.5	1	4
Qi et al. [57]	1	1	0.5	0.5	1	4
Jiang et al. [58]	1	1	1	0.5	1	4.5
Yu and Lau [59]	1	1	1	0.5	1	4.5
Do et al. [60]	1	1	1	1	1	5
Srikanth et al. [61]	1	1	1	1	1	5
Muthusamy and Seetharaman [62]	1	1	1	0.5	1	4.5
Ma et al. [63]	1	1	0.5	0.5	1	4
Arafeen et al. [64]	1	1	0.5	0.5	1	4
Hettiarachchi et al. [65]	1	1	0.5	0.5	1	4
Yoon et al. [66]	1	1	1	1	1	5
Srikanth et al. [67]	1	1	0.5	1	1	4.5
Srikanth et al. [68]	1	1	1	1	1	5
Lin et al. [69]	1	1	0.5	0.5	1	4
Marijan et al. [70]	1	1	0.5	1	1	4.5
Khalilian et al. [71]	1	1	0.5	0.5	1	4
Kim and Porter [72]	1	1	0.5	0.5	1	4
Huang et al. [73]	1	1	0.5	0.5	1	4
Hettiarachchi et al. [74]	1	1	1	0.5	1	4.5
Yoon and Choi [75]	1	1	1	1	1	5
Stallbaum et al. [76]	1	1	0.5	0.5	1	4
Felderer et al. [77]	1	0	1	0	1	3
Ufuktepe et al. [78]	1	1	0.5	0.5	1	4
Zhao et al. [79]	1	1	0.5	0.5	1	4
Mirarab and Tahvildari [80]	1	1	0.5	0.5	1	4
Mirarab and Tahvildari [81]	1	1	0.5	0.5	1	4
Elbaum et al. [82]	1	1	1	0.5	1	4.5

Table A2
Overview of primary studies.

Publication Type	Publication Year	Publication Name	Refs.
Journal	2012	Software Testing, Verification, Reliability	[7]
Journal	2012	Informatica	[13]
Journal	2014	Software Engineering	[18]
Journal	2013	EEE Transactions on Software Engineering	[19]
Journal	2015	IEEE Transactions on Services Computing	[21]
Journal	2014	Software Quality Journal	[22]
Journal	2016	Journal of Systems and Software	[23]
Journal	2015	Indian Journal of Science and Technology	[25]
Journal	2011	International journal on computer science and engineering	[29]
Journal	2002	IEEE transactions on evolutionary computation	[32]
Journal	2006	IEEE Transactions on Software Engineering	[33]
Journal	2007	IEEE Transactions on software engineering	[34]
Journal	2016	Emerging Research in Computing, Information, Communication and Applications	[36]
Journal	2012	Automated Software Engineering	[39]
Journal	2015	Journal of Systems and Software	[40]
Journal	2016	IEEE Transactions on Software Engineering	[41]
Journal	2015	Software Testing, Verification and Reliability	[43]
Journal	2014	Transactions on Software Engineering and Methodology (TOSEM)	[44]
Journal	2016	IEEE Transactions on Software Engineering	[45]
Journal	2013	IEEE transactions on software engineering	[47]
Journal	2012	Science China Information Sciences	[48]
Journal	2011	International Journal of System Assurance Engineering and Management	[50]
Journal	2003	IEEE Transactions on Software Engineering	[51]
Journal	2009	Information and Software Technology	[53]
Journal	2016	Information Technology: New Generations	[54]
Journal	2016	Software Testing, Verification and Reliability	[55]
journal	2015	IEEE Transactions on Services Computing	[56]
Journal	2012	Information and Software Technology	[59]
Journal	2012	Information and Software Technology	[60]
Journal	2010	IEEE Transactions on Software Engineering	[61]
Journal	2016	Information and Software Technology	[62]
Journal	2014	International Journal of Applied	[63]
Journal	2012	Journal of Software Engineering and Applications	[67]
Journal	2016	Journal of Systems and Software	[69]
Journal	2012	Science of Computer Programming	[72]
Journal	2012	Journal of Systems and Software	[74]
Journal	2016	Information and Software Technology	[75]
Journal	2011	International Journal of Software Engineering and Knowledge Engineering	[76]
Journal	2014	International Journal on Software Tools for Technology Transfer	[78]
Journal	2004	Software Quality Journal	[83]
Conference	1999	Proceedings IEEE International Conference on Software Maintenance	[3]
Conference	2014	IEEE Seventh International Conference on Software Testing, Verification and Validation	[20]
Conference	2007	Proceedings of the 3rd international workshop on Advances in model-based testing	[24]
Conference	2011	Internet Computing & Information Services (ICICIS), 2011 International Conference	[30]
Conference	2010	Computer and Communication Technology (ICCCT)	[31]
Conference	2010	International Conference on Quality Software	[35]
Conference	2015	Software Engineering and Service Science (ICSESS)	[37]
Conference	2015	nternational Conference on Software Testing, Verification and Validation (ICST)	[38]
Conference	2013	International Conference on Software Testing, Verification and Validation	[42]
Conference	2013	International Conference on Software Engineering (ICSE)	[46]
Conference	2015	Software Engineering and Service Science (ICSESS)	[57]
Conference	2013	Software Maintenance (ICSM)	[58]
Conference	2013	International Conference on Software Testing, Verification and Validation	[65]
Conference	2014	Software Security and Reliability (SERE)	[66]
Conference	2013	Engineering of Complex Computer Systems (ICECCS)	[70]
Conference	2013	Software Maintenance (ICSM)	[71]
Conference	2002	Software Engineering, 2002. ICSE	[73]
Conference	2016	Computer Software and Applications Conference (COMPSAC)	[79]
Conference	2015	Computer Software and Applications Conference (COMPSAC)	[80]
Conference	2007	International Conference on Fundamental Approaches to Software Engineering	[81]
Conference	2008	International Conference on Software Testing, Verification, and Validation	[82]
Workshop	2012	Proceedings of the 2nd international workshop on Evidential assessment of software technologies	[28]
Workshop	2007	Workshop on Domain specific approaches to software test automation: in conjunction with the 6th ESEC/FSE joint meeting	[49]
Workshop	2008	Workshop on Automation of software test	[77]
Symposium	2015	Software Reliability Engineering (ISSRE)	[26]
Symposium	2015	International Symposium on Search Based Software Engineering	[27]
Symposium	2003	Software Reliability Engineering, 2003. ISSRE	[52]
Symposium	2016	Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)	[64]
Symposium	2005	International Symposium on Empirical Software Engineering	[68]

Table A3
Existing approaches and their citation indexes.

No.	Approaches	Source
1	Search based	[25–40]
2	Coverage based	[3,29,42–52]
3	Fault based	[53–60]
4	Requirement based	[52,61–67]
5	History based	[38,68–73]
6	Risk based	[65,66,74–77]
7	Bayesian Network based	[78–81]
8	Cost Effective based	[60,73,82]
9	Multi-Criteria based	[19,20],
10	Topic-Model based	[18]
11	Workflow	[21]
12	Lexicographical Ordering	[41]
13	Similarity based	[22]
14	Scope-Aided based	[23]
15	Model based	[24]

Table A4
The advantages and limitation of TCP approaches.

Approaches	Advantages	Limitation
Search-based -GA	Acquired result such as coverage and fault detection is the major benefit [28]	Speed execution of GA is likely to be slow [28,35] due to mutation process
Search based - Greedy	Results in high coverage and performance against other search algorithms [34,35]	Too much greedy type with different purposes [34]
Search-based Ant colony optimization (ACO)	ACO method has been confidently used in various optimization problems that benefits TCP as it has various kind of factors that may be combined [37]	As ants tend to follow the shorter path, total coverage would be the issues for this approach [37]
Search-based String Distance	Manage to detect and kill the strongest mutants in regression testing [39]	Not able to produce an optimum result when there are two optimal sequences produced [39]
Coverage-based	Perform full coverage which in theory is able to detect all possible fault [3,7,43]–[45] Coverage approaches are much predictable in terms of APFD [44]	Overall APFD score is still not optimal [43,44] Requires extra information such as test case fault history data or modification information to enhance fault detection rate [42,45] In other words, it is better to combine with other approaches
Fault-based	High possibility to detect the faults in a system earlier (including hidden faults) [53,59] Guaranteed to be free from prescribed faults [56,58]	Accuracy of fault detection is subjective that is prone to human error [53] Prescribed faults might not well defined [59]
Requirement-based	Uses of requirements information during requirement elicitation such as customer priority, error prone value, volatility value, and execution difficulties for prioritization criteria [61,67] Claim to achieve high efficiency (earlier execution starting time, no code dependency) [61,63,67]	Such information requires human involvement resulting in a possibility of human error [61,67] Uses static data that needs to have an perfect mapping may not reliable [62]
History-based	Utilizing historical data including the count of executions, the count which reveals a fault [71] Accelerate rate of fault-detection [71,72]	History-based has several bottlenecks such as historic information availability and small number of defect information
Risk-based	High possibility to detect specific error related to system's risks [66,75] Effective in early detection of serious fault [66,74]	Human expert dependency [75,76]
Bayesian -Network-based	Can use multiple kinds of information such as coverage info, fault proneness and etc. to predict the probability of fault detection using Bayesian network model [78,81]	Time consuming, requires algorithm improvement [81]
Cost-Effective- based	Promote cost awareness that leads to cost reduction [82]	Has several bottlenecks such as historic information availability and small number of defect information
Multi-Criteria based	Outperformed single criteria prioritization [19] Better in industrial case study [7]	
Topic-Model based	Can indicate test cases' functionality, and able to identify more related evidence [18]	Full natural language processing and complete with grammars, need to have a trained data which is ineffective, and is error prone [18]

Table A5
Major process proposed by researcher.

No	Source	Process
1	Miller [47]	1) Divide into two categories dependency 2) Test case dependents were calculated 3) Longest path of test case dependent were calculated 4) Calculating Test Case Priorities
2	Bryce et al [49]	1) Select test that covers the most unique 2) Iterate by random select test for tie breaking 3) Iterate test cover the most new pairs until last remaining
3	Leon and Podgurski [51]	1) Execute the tests based maximum coverage 2) Execute the tests by cluster sampling 3) Select test case by failure pursuit 4) Test case left ordered in random
4	Eghbali and Tahvildari [41]	1) Choose highest coverage test case 2) Choose the highest coverage and not covered yet 3) Repeat step 2 4) Repeat until the ordering is complete, random selection for ties
5	Tahvili et al. [53]	1) Specific criteria is determined 2) Calculate the criteria values 3) Prioritize the test case using the criteria values
6	Wang et al. [56]	1) Calculate fault severity 2) Selecting a highest fault severity of test case and record the covered test case and faults 3) Updating fault severity of the remaining test cases 4) Repeating step 2 to 4 until all the test cases are sorted to end
7	Jiang et al. [58]	1) Calculate cost to discover first fault localization 2) Calculate cost to discover second fault localization 3) Repeat for third and so on 4) Report the mean and variance of the cost for localizing faults
8	Yu and Lau [59]	1) Determine fault from prescribe model 2) Generate test cases using MUMCUT 3) The effectiveness is evaluated
9	Do et al. [60]	1) Obtain coverage information for each object program 2) Reorder test suites 3) Compute rate of fault detection and missed fault 4) Prioritize again with fault detection information and other cost
10	Srikanth et al. [68]	1) Analyze historical field failures 2) Identify rarity of use cases 3) Tag use cases 4) Prioritize use cases
11	Khalilian et al. [71]	1) Study historical data (test priority) 2) Test case coverage data is gathered with respect to coverage criteria 3) Prioritize based on percentage code coverage with its priority 4) Record the final data
12	Ma et al. [63]	1) Calculated priorities of test cases 2) Selection of test case with highest value 3) Prioritize and recalculation priorities 4) Repeat until all requirements are covered
13	Srikanth et al. [61]	1) Get customer priority scores and fault proneness scores 2) Find average of all scores 3) Divided the average of scores with sum of both to obtain weight 4) Compute combined scores by multiplying each factor score
14	Arafeen et al. [64]	1) Cluster requirement 2) Mapping the requirement 3) Prioritize in cluster 4) Full prioritization
15	Krishnamoorthi et al. [52]	1) Factor identification 2) Calculate total requirement and test cases 3) Calculate factor values 4) Calculate requirement weight 5) Analyze and map the test case to requirement 6) Recalculate requirement weight 7) Sorting the test cases
16	Srikanth et al. [67]	1) Determine the requirement criteria 2) Determine the requirement priorities 3) Defect and failure mapped to requirement 4) Determine the requirement complexity 5) Test case is run
17	Hettiarachchi et al. [74]	1) Risk is estimated 2) Risk requirement calculated 3) Risk item calculated 4) TCP process starts
18	Hettiarachchi et al. [65]	1) Risk is estimated 2) Risk weight calculated 3) Risk value calculated 4) Evaluate extra factors 5) TCP process start
19	Yoon et al. [66]	1) Identify risk item 2) Estimates the risk values 3) Calculate test case priority 4) Prioritize based on test case priority

References

[1] P. Ralph, Software engineering process theory: a multi-method comparison of sensemaking-coevolution-implementation theory and function-behavior-structure theory, Inf. Softw. Technol. 70 (2016) 232–250.
[2] G.J. Myers, T.M. Thomas, C. Sandler, The Art of Software Testing, vol. 1, John Wiley & Sons, 2004.
[3] G. Rothermel, R.H. Untch, C.C. Chu, M.J. Harrold, Test case prioritization: an empirical study, in: Software Maintenance, 1999. (ICSM '99) Proceedings. IEEE International Conference on, 1999, pp. 179–188.
[4] H.K.N. Leung, Insights into regression testing, in: Proceedings of the International Conference on Software Maintenance, 1989, pp. 60–69.
[5] S. Elbaum, G. Rothermel, J. Penix, Techniques for improving regression testing in continuous integration development environments, in: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering – FSE 2014, 2014, pp. 235–245.
[6] P.K. Chittimalli, M.J. Harrold, Recomputing coverage information to assist regression testing, IEEE Trans. Softw. Eng. 35 (4) (2009) 452–469.
[7] S. Yoo, M. Harman, Regression testing minimisation, selection and prioritisation: a survey, *Test Verification Reliab.* 0 (2007) 1–7.
[8] D. Jeffrey, N. Gupta, Improving fault detection capability by selectively retaining test cases during test suite reduction, IEEE Trans. Softw. Eng. 33 (2) (Feb. 2007) 108–123.
[9] S. Elbaum, P. Kallakuri, A.G. Malishevsky, G. Rothermel, S. Kanduri, Understanding the effects of changes on the cost-effectiveness of regression testing techniques, J. Softw. Test., Verification Reliab. 12 (2) (2003) 65–83.
[10] S. Elbaum, A.G. Malishevsky, G. Rothermel, Test case prioritization: a family of empirical studies, IEEE Trans. Softw. Eng. 28 (2) (2002) 159–182.
[11] B. Kitchenham, in: Procedures for Performing Systematic Reviews, 33, no. TR/SE-0401, Keele University, Keele, UK, 2004, p. 28.
[12] B. Kitchenham, O.P. Brereton, D. B., et al., Systematic literature reviews in software engineering – a systematic literature review, Inf. Softw. Technol. 51 (1) (2009) 7–15.
[13] Y. Singh, Systematic literature review on regression test prioritization techniques difference between literature review and systematic literature, Informatica 36 (2012) 379–408.
[14] C. Catal, D. Mishra, Test case prioritization: a systematic mapping study, Softw. Qual. J. 21 (3) (2012) 445–478.
[15] A. Kumar, K. Singh, A Literature Survey on Test Case Prioritization, Compusoft, 2014.
[16] P. Kiran, K. Chandraprakash, A literature survey on TCP-test case prioritization using the RT-regression techniques, Global J. (2015).
[17] P. Achimugu, A. Selamat, R. Ibrahim, M. Naz, A systematic literature review of software requirements prioritization research, Inf. Softw. 56 (2014) 568–585.
[18] S. Thomas, H. Hemmati, A. Hassan, Static test case prioritization using topic models, Software Engineering, 2014.
[19] S. Sampath, R. Bryce, A.M. Memon, A uniform representation of hybrid criteria for regression testing, IEEE Trans. Softw. Eng. 39 (10) (2013) 1326–1344.
[20] A.B. Sanchez, S. Segura, A. Ruiz-Cortes, A comparison of test case prioritization criteria for software product lines, in: Software Testing, Verification and Validation (ICST), 2014 IEEE Seventh International Conference, 2014, pp. 41–50.
[21] L. Mei, W.K. Chan, T.H. Tse, S. Member, B. Jiang, K. Zhai, Preemptive regression testing of workflow-based web services, IEEE Trans. 8 (5) (2015) 740–754.
[22] C. Fang, Z. Chen, K. Wu, Z. Zhao, Similarity-based test case prioritization using ordered sequences of program entities, Softw. Qual. J. 22 (2) (2014) 335–361.
[23] B. Miranda, A. Bertolino, Scope-aided test prioritization, selection and minimization for software reuse, J. Syst. Softw. 0 (2016) 1–22.
[24] B. Korel, G. Koutsogiannakis, L.H. Tahat, Model-based test prioritization heuristic methods and their evaluation, in: Proceedings of the 3rd International Workshop on Advances in Model-based Testing - A-MOST '07, 2007, pp. 34–43.
[25] R. Maheswari, D. Mala, Combined genetic and simulated annealing approach for test case prioritization, Indian Journal of Science and Technology, 2015.
[26] Y. Lou, D. Hao, L. Zhang, Mutation-based test-case prioritization in software evolution, in: 2015 IEEE 26th International Symposium on Software Reliability Engineering, ISSRE 2015, 2016, pp. 46–57.
[27] F. Yuan, Y. Bian, Z. Li, R. Zhao, Epistatic genetic algorithm for test case prioritization, International Symposium on Search Based, 2015.
[28] C. Catal, On the application of genetic algorithms for test case prioritization: a systematic literature review, in: Proceedings of the 2nd International Workshop on, 2012.
[29] A. Kaur, S. Goyal, A genetic algorithm for fault-based regression test case prioritization, Int. J. Comput. Appl. 32 (8) (2011) 975–8887.
[30] W. Jun, Z. Yan, J. Chen, Test case prioritization technique based on genetic algorithm, Internet Computing & Information, 2011.
[31] S. Sabharwal, R. Sibal, C. Sharma, Prioritization of test case scenarios derived from activity diagram using genetic algorithm, in: 2010 International Conference on Computer and Communication Technology, ICCCT-2010, 2010, pp. 481–485.
[32] K. Deb, S. Pratab, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NGSA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[33] H. Do, G. Rothermel, On the use of mutation faults in empirical assessments of test case prioritization techniques, IEEE Trans. Softw. Eng. 32 (9) (2006) 733–752.
[34] Z. Li, M. Harman, R.M. Hierons, Search algorithms for regression test case prioritization, IEEE Trans. Softw. Eng. 33 (4) (2007) 225–237.
[35] S. Li, N. Bian, Z. Chen, D. You, A simulation study on some search algorithms for regression test case prioritization, 2010 10th International, 2010.
[36] K. Solanki, Y. Singh, S. Dalal, P. Srivastava, Test case prioritization: an approach based on modified ant colony optimization, Emerg. Res. (2016).
[37] D. Gao, X. Guo, L. Zhao, Test case prioritization for regression testing based on ant colony optimization, Software Engineering and Service, 2015.
[38] T. Noguchi, H. Washizaki, Y. Fukazawa, History-based test case prioritization for black box testing using ant colony optimization, 2015 IEEE 8th, 2015.
[39] Y. Ledru, A. Petrenko, S. Boroday, N. Mandran, Prioritizing test cases with string distances, Autom. Softw. Eng. 19 (1) (2012) 65–95.
[40] B. Jiang, W.K. Chan, Input-based adaptive randomized test case prioritization, J. Syst. Softw. 105 (C) (2015) 91–106.
[41] S. Eghbali, L. Tahvildari, Test case prioritization using lexicographical ordering, IEEE Trans. Softw. Eng. 42 (12) (2016) 1178–1195.
[42] D. Di Nardo, N. Alshahwan, L. Briand, Y. Labiche, Coverage-based test case prioritization: an industrial case study, in: Proceedings - IEEE 6th International Conference on Software Testing, Verification and Validation, ICST 2013, 2013, pp. 302–311.
[43] D. Nardo, N. Alshahwan, L. Briand, Coverage-based regression test case selection, minimization, and prioritization: a case study on an industrial system, Software Testing, 2015.
[44] D. Hao, L. Zhang, L. Zhang, G. Rothermel, H. Mei, A unified test case prioritization approach, ACM Trans. Softw. Eng. Methodol. 24 (2) (2014) 10:1–10:31.
[45] D. Hao, L. Zhang, L. Zang, Y. Wang, X. Wu, T. Xie, To be optimal or not in test-case prioritization, IEEE Trans. Softw. Eng. 42 (5) (2016) 490–504.
[46] L. Zhang, D. Hao, L. Zhang, G. Rothermel, H. Mei, Bridging the gap between the total and additional test-case prioritization strategies, in: Proceedings - International Conference on Software Engineering, 2013, pp. 192–201.
[47] T. Miller, Using dependency structures for prioritisation of functional test suites, IEEE Trans. Softw. Eng. 39 (2) (2013) 258–275.
[48] C.R. Fang, Z.Y. Chen, B.W. Xu, Comparing logic coverage criteria on test case prioritization, Sci. China Inf. Sci. 55 (12) (2012) 2826–2840.
[49] R.C. Bryce, S. Sampath, J.B. Pedersen, S. Manchester, Test suite prioritization by cost-based combinatorial interaction coverage, Int. J. Syst. Assur. Eng. Manage. 2 (2) (2011) 126–134.
[50] J. Jones, M. Harrold, Test-suite reduction and prioritization for modified condition/decision coverage Georgia Institute of Technology, Test 3 (3) (2003) 101–195.
[51] D. Leon, A. Podgurski, A comparison of coverage-based and distribution-based techniques for filtering and prioritizing test cases, in: Proceedings - International Symposium on Software Reliability Engineering, ISSRE, January, 2003, 2003, pp. 442–453.
[52] R. Krishnamoorthi, S.A. Sahaaya Arul Mary, Factor oriented requirement coverage based system test case prioritization of new and regression test cases, Inf. Softw. Technol. 51 (4) (2009) 799–808.
[53] S. Tahvili, W. Afzal, M. Saadatmand, M. Bohlin, Towards earlier fault detection by value-driven prioritization of test cases using fuzzy TOPSIS, Inf. Technol. (2016).
[54] E.L.G. Alves, P.D.L. Machado, T. Massoni, M. Kim, Prioritizing test cases for early detection of refactoring faults, Softw. Test. Verification Reliab. 26 (5) (2016) 402–426.
[55] L. Mei, et al., A subsumption hierarchy of test case prioritization for composite services, IEEE Trans. Serv. Comput. 8 (5) (2015) 658–673.
[56] Y. Wang, X. Zhao, X. Ding, An effective test case prioritization method based on fault severity, Softw. Eng. Serv. (2015).
[57] Y. Qi, X. Mao, Y. Lei, Efficient automated program repair through fault-recorded testing prioritization, in: IEEE International Conference on Software Maintenance, ICSM, 2013, pp. 180–189.
[58] B. Jiang, Z. Zhang, W.K. Chan, T.H. Tse, T.Y. Chen, How well does test case prioritization integrate with statistical fault localization? Inf. Softw. Technol. 54 (7) (2012) 739–758.
[59] Y.T. Yu, M.F. Lau, Fault-based test suite prioritization for specification-based testing, Inf. Softw. Technol. 54 (2) (2012) 179–202.
[60] H. Do, S. Mirarab, L. Tahvildari, G. Rothermel, The effects of time constraints on test case prioritization: a series of controlled experiments, IEEE Trans. Softw. Eng. 36 (5) (2010) 593–617.
[61] H. Srikanth, C. Hettiarachchi, H. Do, Requirements based test prioritization using risk factors, Inf. Softw. Technol. 69 (C) (2016) 71–83.
[62] T. Muthusamy, A new effective test case prioritization for regression testing based on prioritization algorithm, Int. J. Appl. Inf. Syst. (IJAIS) 6 (7) (2014) 21–26.
[63] T. Ma, H. Zeng, X. Wang, Test case prioritization based on requirement correlations, in: 2016 IEEE/ACIS 17th International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD 2016, 2016, pp. 419–424.
[64] J. Badwal, H. Raperia, Test case prioritization using clustering, in: 2013 IEEE Sixth International Conference, 2013, pp. 488–492.
[65] C. Hettiarachchi, H. Do, B. Choi, Effective regression testing using requirements and risks, in: Proceedings – 8th International Conference on Software Security and Reliability, SERE 2014, 2014, pp. 157–166.
[66] M. Yoon, A test case prioritization through correlation of requirement and risk, J. Softw. Eng. Appl. 5 (10) (2012) 823–836.

[67] H. Srikanth, L. Williams, J. Osborne, System test case prioritization of new and regression test cases, in: Int'l Symp on Empirical Software Engineering, vol. 0, no. c, 2005, pp. 62–71.
[68] H. Srikanth, M. Cashman, M.B. Cohen, Test case prioritization of build acceptance tests for an enterprise cloud application: an industrial case study, J. Syst. Softw. 119 (2016) 122–135.
[69] C.T. Lin, C.D. Chen, C.S. Tsai, G.M. Kapfhammer, History-based test case prioritization with software version awareness, in: Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS, 2013, pp. 171–172.
[70] D. Marijan, A. Gotlieb, S. Sen, Test case prioritization for continuous regression testing: an industrial case study, Software Maintenance (ICSM), 2013.
[71] A. Khalilian, M. Azgomi, Y. Fazlalizadeh, An improved method for test case prioritization by incorporating historical test case data, Science of Computer, 2012.
[72] J.-M.K.J.-M. Kim, A. Porter, A history-based test prioritization technique for regression testing in resource constrained environments, in: Proceedings of the 24th International Conference on Software Engineering ICSE 2002, 2002, pp. 119–129.
[73] Y.C. Huang, K.L. Peng, C.Y. Huang, A history-based cost-cognizant test case prioritization technique in regression testing, J. Syst. Softw. 85 (3) (2012) 626–637.
[74] C. Hettiarachchi, H. Do, B. Choi, Risk-based test case prioritization using a fuzzy expert system, Inf. Softw. Technol. (2016).
[75] H. Yoon, B. Choi, A test case prioritization based on degree of risk exposure and its empirical study, Int. J. Softw. Eng. Knowl. Eng. 21 (2) (2011) 191–209.
[76] H. Stallbaum, A. Metzger, K. Pohl, An automated technique for risk-based test case generation and prioritization, in: Proceedings of the 3rd international workshop on Automation of software test, ACM, 2008, pp. 67–70.
[77] M. Felderer, I. Schieferdecker, A taxonomy of risk-based testing, Int. J. Softw. Tools Technol. Transfer 16 (5) (2014) 559–568.
[78] E. Ufuktepe, T. Tuglular, Automation architecture for Bayesian network based test case prioritization and execution, Computer Software and Applications, 2016.
[79] X. Zhao, Z. Wang, X. Fan, Z. Wang, A Clustering-Bayesian network based approach for test case prioritization, IEEE 39th Annual Computer Software and Applications Conference (COMPSAC), 2015.
[80] S. Mirarab, L. Tahvildari, A prioritization approach for software test cases based on Bayesian networks, Fundam. Approaches Softw. Eng. Springer Berlin Heidelberg 4422 (2007) 276–290.
[81] S. Mirarab, L. Tahvildari, An empirical study on Bayesian network-based approach for test case prioritization, in: 2008 International Conference on Software Testing, Verification, and Validation, 2008, pp. 278–287.
[82] S. Elbaum, G. Rothermel, S. Kanduri, A.G. Malishevsky, Selecting a cost-effective test case prioritization technique, Softw. Qual. J. 12 (3) (2004) 185–210.
[83] S. Yoo, M. Harman, Regression testing minimization, selection and prioritization: a survey, Softw. Test. Verification Reliab. 22 (2) (2012) 67–120.
[84] A.B. Sanchez, S. Segura, A. Ruiz-Cortes, A comparison of test case prioritization criteria for software product lines, in: 2014 IEEE Seventh International Conference on Software Testing, Verification and Validation, 2014, pp. 41–50.
[85] H. Srikanth, C. Hettiarachchi, H. Do, Requirements based test prioritization using risk factors: an industrial study, Inf. Softw. Technol. 69 (2016) 71–83.
[86] M.J. Arafeen, H. Do, Test case prioritization using requirements-based clustering, in: 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation, 2013, pp. 312–321.
[87] A. Shahbazi, J. Miller, Black-box string test case generation through a multi-objective optimization, IEEE Trans. Softw. Eng. 42 (4) (2016) 361–378.
[88] B. Jiang, W. Chan, Input-based adaptive randomized test case prioritization: a local beam search approach, J. Syst. Softw. (2015).
[89] S. Elbaum, A. Malishevsky, and G. Rothermel, Prioritizing test cases for regression testing. 2000.
[90] G. Kapfhammer, M. Soffa, Using coverage effectiveness to evaluate test suite prioritizations, in: Proceedings of the 1st ACM international, 2007.
[91] Software-artifact Infrastructure repository: home. [Online]. Available: http://sir.unl.edu/portal/index.php. [Accessed: 20-Mar-2017].
[92] R. Fisher, Statistical methods for research workers, 1925.
[93] D.K. Yadav, S. Dutta, Test case prioritization technique based on early fault detection using fuzzy logic, in: 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2016, pp. 1033–1036.
[94] A. Schwartz, H. Do, Cost-effective regression testing through adaptive test prioritization strategies, J. Syst. Softw. 115 (2016) 61–81.
[95] J.A. Parejo, A.B. Sánchez, S. Segura, A. Ruiz-Cortés, R.E. Lopez-Herrejon, A. Egyed, Multi-objective test case prioritization in highly configurable systems: a case study, J. Syst. Softw. 122 (2016) 287–310.
[96] A. Marchetto, M. Islam, W. Asghar, A multi-objective technique to prioritize test cases, IEEE Trans. 42 (10) (2016) 918–940.
[97] X. Xia, L. Gong, T.-D.B. Le, D. Lo, L. Jiang, H. Zhang, Diversity maximization speedup for localizing faults in single-fault and multi-fault programs, Autom. Softw. Eng. 23 (1) (Mar. 2016) 43–75.
[98] M. Laali, H. Liu, M. Hamilton, M. Spichkova, H.W. Schmidt, in: Test Case Prioritization Using Online Fault Detection Information, Springer, Cham, 2016, pp. 78–93.
[99] W. Fu, H. Yu, G. Fan, X. Ji, Test case prioritization approach to improving the effectiveness of fault localization, in: 2016 International Conference on Software Analysis, Testing and Evolution (SATE), 2016, pp. 60–65.
[100] X.-Y. Zhang, D. Towey, T.Y. Chen, Z. Zheng, K.-Y. Cai, A random and coverage-based approach for fault localization prioritization, in: 2016 Chinese Control and Decision Conference (CCDC), 2016, pp. 3354–3361.