SN Computer Science (2021) 2:421
https://doi.org/10.1007/s42979-021-00821-3

REVIEW ARTICLE

Functional Requirement-Based Test Case Prioritization in Regression Testing: A Systematic Literature Review

Muhammad Hasnain[1] · **Muhammad Fermi Pasha**[1] · **Imran Ghani**[2] · **Seung Ryul Jeong**[3]

Received: 21 November 2020 / Accepted: 13 August 2021 / Published online: 19 August 2021

Abstract

Regression testing, as an important part of the software life cycle, ensures the validity of modified software. Researchers' focus of this research is on functional requirement-based 'Test Case Prioritization' (TCP) because requirement specifications help keep the software correctness on customers' perceived priorities. This research study is aimed to investigate requirement-based TCP approaches, regression testing aspects, applications regarding the validation of proposed TCP approaches, systems' size under regression testing, test case size and relevant revealed faults, TCP related issues, TCP issues and types of primary studies. Researchers of this paper examined research publications, which have been published between 2009 and 2019, within the seven most significant digital repositories. These repositories are popular, and mostly used for searching papers on topics in software engineering domain. We have performed a meticulous screening of research studies and selected 35 research papers through which to investigate the answers to the proposed research questions. The final outcome of this paper showed that functional requirement-based TCP approaches have been widely covered in primary studies. The results indicated that fault size and the number of test cases are mostly discussed as regression testing aspects within primary studies. In this review paper, it has been identified that iTrust system is widely examined by researchers in primary studies. This paper's conclusion indicated that most of the primary studies have been demonstrated in the real-world settings by respective researchers of focused primary studies. The findings of this "Systematic Literature Review" (SLR) reveal some suggestions to be undertaken in future research works, such as improving the software quality, and conducting evaluations of larger systems.

Keywords Regression testing · Test case prioritization · Requirement priority · Customers' priority · Real-world settings · TCP issues

Introduction

Regression testing is performed when changes are made to existing functionality of software. The main focus of regression testing is to ensure that the newly introduced changes do not obstruct the behavior of existing and unchanged parts of software. To ensure this, regression testing helps one to execute test cases that have been newly built and determine the correctness of original features of a system. Regression testing concerns the testing of a system, and is carried out during development of a system as well as at the integration phases of a system. Moreover, regression testing practices are introduced in the evolution and maintenance of software in the agile software development environment [114]. Regression testing also ensures the software quality during the software evolution [115]. Hence, regression testing is also carried out at unit testing, component testing and system testing levels across the software development. Integration

✉ Muhammad Hasnain
muhammad.malik1@monash.edu

Muhammad Fermi Pasha
muhammad.fermipasha@monash.edu

Imran Ghani
ghanii@vmi.edu

Seung Ryul Jeong
srjeong@kookmin.ac.kr

[1] School of Information Technology, Monash University Malaysia, Subang Jaya, Malaysia

[2] Computer and Information Sciences Department, Virginia Military Institute, Lexington, VA, USA

[3] Kookmin University, Seoul 136, South Korea

testing is also mandatory for using and integrating third party's software, and ensuring the correct output. For both software development and integration, a series of test cases undergo regression testing. Regression testing involves the more expensive task of maintenance for software systems. It consumes more than 50% maintenance cost except the manual testing that costs less but is more prolong testing process [1]. In other words, regression testing preserves the quality of software systems that is validated through the execution of test cases. To retest all test cases of a system or its modified components is often impractical because of their additional development cost and time consumption. TCP technique is proposed to define the execution order of test cases in a test suite according to given criteria. In other words, TCP is aimed to find an optimal text case execution to achieve specific testing objectives [116]. Maximum faults can be revealed from the execution of reordered test cases. Test case prioritization (TCP) as a type of regression testing is used for the optimal reordering of test case [2], and to detect the maximum faults within a shorter time frame at reduced cost [3, 4]. TCP is the safe aspect of regression testing [5]. TCP reorders test cases and does not involve discarding of test cases for the test suite *T*. On the other hands, test case minimization aspect of regression testing uses the discard of duplicated test cases [6]. Several TCP approaches have been developed for scheduling and execution of test suits. These TCP approaches include coverage-based, users' session based, and requirements correlation-based TCP approaches, and risk-based TCP approaches [7–9]. TCP approaches use specific criteria and objectives for scheduling test cases in a prescribed order.

Requirements-based TCP approaches are proposed from the customers' requirements to order the test cases. While ordered test cases aim to reduce the fault identification time, and time needed for covering the functional requirements. On the other hand, a non-functional requirement TCP approach is aimed to validate the timing constraints and performance requirements [87]. The objectives of two types of requirements based approaches vary: the former type of TCP approach is proposed with the objectives of functional requirements coverage while the latter type of TCP approach covers the non-functional requirements coverage [6, 9].

Functional testing focuses on the input and expected out puts of a system. So, functional testing serves the functionality validation. Valid and invalid inputs are identified from the user's point of view [117]. Functional testing ensures that all functionalities of a system are correctly working. Subsequently, functional testing and regression testing are two distinct phases of software validation and verification [82]. Test case prioritization covers the test cases which can reveal changes in specifications [83].

System requirements and their associated risk can enable the software testers to identify test cases, which can reveal maximum faults [106]. Similar to earlier mentioned study, a recent research [108] propose to use requirements as a historical information of a system to perform TCP. A recent work [109] highlights the importance of acceptance testing driven by TCP to validate the system requirements. This approach is proposed to enhance users' trust in cyber physical systems (CPS).

Requirement engineering community keeps the records of requirement information and prefer to use it for the identification of faults quickly [110]. Recent research [111] proposes to utilize the changes in requirement information, size of the method, and complexity as risk indicating factors. This risk-based TCP shows better performance compared with the earlier proposed research studies.

We have presented a summary of review studies [16–21, 76, 85, 86, 89, 92], and [107] with respect to their limitations and differences from our study. In addition, we have identified some recent reviews on the regression testing and TCP topics. We state their contribution and highlight what is missed and can be performed in future works. A recently published research [112] covered the performance of evolutionary algorithms in TCP with time and cost factors. It would be exciting if more factors, including risk-related measures include in future works. Although risk-based TCP approaches were proposed, but they did not measures risk based on the evolutionary algorithms. Another review study [113] presents the review of genetic algorithm based TCP approaches. The latter research identified eight classes of genetic algorithms (GAs). Most of them were the multi-objective GAs proposed to solve the complex issues. While the most significant approaches have been summarized in the earlier mentioned review studies, we lack the emphasis on functional requirement-based testing. Our goal is to identify the primary studies in this innovative area of regression testing and analyze the literature to know the pros and cons of the proposed approaches.

Mechanism of Functional Requirement-Based TCP

The mechanism of functional requirement-based TCP requires structuring of the system under testing into functional units where a tree structure represents the functional units and their breaking down to corresponding functional requirements. Test cases are derived from the functional requirements. Test cases trace back to functional requirements, and test execution results can report the requirements' level [84]. As a results, failed status of test cases can determine that faults exist in a software.

Figure 1 illustrates the mechanism of functional requirements-based test case prioritization. Every "Functional Requirement" FR has a specific number of test cases at software testing phase in "Software Development Life Cycle" (SDLC). However, addition or change in FRs requires

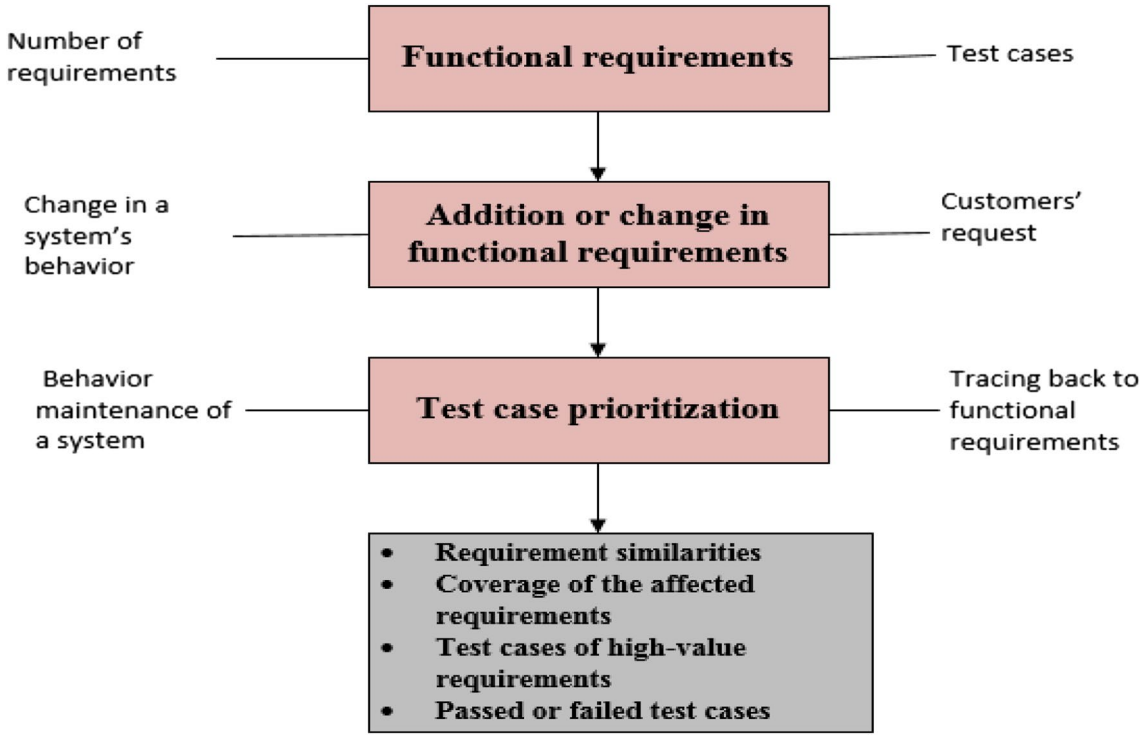

Fig. 1 Mechanism of functional requirements-based test case prioritization

regression testing. Therefore, proposed TCP approaches within the FRs-based criteria ensure that existing behavior of a software is maintained.

In practice, TCP approaches are based on certain conditions such as history of fault detection, source code information [10] and requirements' correlation [8]. These conditions imply the objective TCP, and many other factors including fault severity [11], testing paradigm [7] and rule-based execution of test cases [12], which are not taken into account for prioritization of test cases. Using these factors together, the objective of test case prioritization remains the identification of maximum faults within a short time. Other than above-mentioned factors, time and cost factors are significant for software development process [12]. In software development process, software testing takes more than 50% time for releasing a system. In similar way, both functional and non-functional requirements are verified before the deployment of a system. Therefore, quality improvement, verification, and validation of software are the crucial stages for its successful deployment. The software test quality cannot be measured directly, but related quality control, as well as defect management factors, help in determining the visibility of software test quality [13]. Functionality, engineering and adaptability are two sets of a software quality, which are considered as dimensions in the space of the software test quality [14]. To maintain quality of systems, existing techniques regarding the test case prioritization cannot provide the optimum results if there are a large number of test suites. In the earliest study on regression testing, Elbaum et al. [15] found that TCP techniques took a long time to prioritize each test case individually.

This SLR is focused on the functional requirements-based TCP in regression testing. We observed that a number of primary studies proposed different criteria-based TCP approaches, and few of them are presented together in the existing SLRs. Due to increase in number of studies over time regarding the TCP. We have chosen to present a SLR on FRs criteria-based proposed TCP approaches. "Test Case Selection" (TCS), "Test Case Reduction" (TCR) and "Test Case Prioritization" (TCP) are types of regression testing. We have also observed that research work on TCP has been widely examined in the literature. Therefore, we have decided to empirically reveal information on proposed requirement-based TCP approaches. We do not include the non-functional aspects of software because they are out of scope of this SLR. Non-functional requirements' aspects pertain to performance of software systems.

Motivation

Our SLR has motivation from existing functional requirements-based test case prioritization strategies. A FR-based strategy or approach is mostly derived from a system's requirements, relevant code and linkage between

requirements and code. Majority of the existing SLRs, and mapping studies focused on the regression testing, and its types TCS, TCR, and TCP. None of these studies showed a brief description on the functional requirement aspects of software systems. These aspects have been used to propose and evaluate the FR-based TCP approaches.

As evolution trends in software systems are increasing, ultimately modified and additional requirements are accommodated. To make successful modifications in software, TCP ensures that system behaves correctly and modification have no impacts on the unchanged portions of a program [98]. In the following table, we present summary of the existing SLRs, and mapping studies.

Despite the fact that numerous published works on TCP exist in the literature, according to our knowledge no review has been conducted on FR-based TCP techniques in regression testing. Prior to this SLR survey and mapping studies [20, 21, 76, 86] did not include FR-based TCP approaches at all (see Table 1). This might be due to their scope and inclusion criteria of studies. Other studies [16, 17, 89] did not include recent literature on FR-based TCP approaches. Besides these SLRs and mapping studies, [18, 19, 85] include limited discussion on the requirement criteria proposed for TCP approaches. On the other hand, survey studies with the inclusion of FRs-based TCP approaches provide us limited information about requirements change, test cases of modified requirements, and association between test cases and faults types. These concepts can be further explored to optimize TCP in regression testing. This optimization of TCP could result in the identification of maximum faults. Furthermore, none of survey, and mapping studies show the potential problems which still exist, and can be addressed in future works. To know which test case factors have been applied in the recent proposed TCP approaches is of great interest for researchers. They could address the identified issues in regression testing to optimize their TCP approaches. This SLR is aimed at highlighting the issues, which are discussed in the results section. Faults recovered from applications of proposed TCP approaches can help in ensuring the software quality. Earlier discovered faults can be fixed by lesser efforts, as researchers can focus the affected FRs rather than testing the complete set of functional requirements. This SLR will aid knowledge to software testers and managers to become aware of the updated information on requirement-based TCP approaches.

The aims and objectives of this study include as follows:

(i) The first objective of this SLR is to present an overview of the state-of-the-art FR-based approaches in regression testing. The proposed approaches might be considered proposed by academic researchers or both academic and industrial researchers.
(ii) The second objective of this SLR is to identify the issues addressed by the proposed FR-based TCP approaches. We also aim to highlight the issues which remain unresolved.
(iii) This SLR is aimed to identify the test case size and defect types that associate with each other.
(iv) This SLR also aims at identifying the linkage metrics in the chosen primary studies.
(v) This SLR identifies system under testing (SUT) and their domains used to validate the FR-based TCP approaches.

In the reminder of this SLR, "Related Work" presents the Related Work, including survey studies, and primary studies. "Research Method for Systematic Literature Review" presents the research method to execute the SLR. "Results and Discussion" reports "results and discussion". "Threats to Validity" presents threats to validity while "Conclusions and Implications" summarizes the results and presents our SLR's implications.

Related Work

In this section, authors of this paper provide related work on TCP in regression testing including discussion on the existing proposed TCP approaches. Researchers have distributed the related studies as survey studies and primary studies. In the subsequent part, we review the related survey studies.

Survey Studies

We have studied six important SLRs in the research area of regression testing and test case prioritization. Authors have collected the relevant survey articles which include discussion on regression testing test case prioritization.

In the first SLR study [16], researchers have stated that three factors, "Customer Priority" (CP), "Requirements Volatility" (RV), and "Developers' Priority" (DP), are used as properties of requirements. Moreover, we have identified "Requirement Change" (RC) and "Fault Proneness" (FP) factors in our SLR. They have categorized the prioritizing approaches into nine types. Requirement-based TCP approach is one of these types. They used primary studies which were published by 2009, and hence, a number of primary studies have been added in the literature on functional requirement-based TCP in regression testing. Since 2009 a number of TCP approaches regarding functional aspects of software have been proposed. These newly proposed TCP approaches use other than three above-mentioned requirements factors. Therefore, our SLR presents the updated information about requirement factors proposed in TCP approaches.

Table 1 Motivational aspect of this SLR and limitation of existing survey, and mapping studies

References	Research focus	Requirements-based TCP approaches' features	Industrial context (case studies, industrial programs)	Empirical aspects (test cases and faults)		Publication year
				Test cases	Faults	
[16]	Test case prioritization	Requirements-based TCP technique with factors Search strategy requirements	Limited support of industry-based case studies papers	×	×	2013
[17]	Regression testing of web services	Total and additional quota constraint TCP approaches	No support of industry-based case studies papers	√	√	2014
[18]	Test case prioritization	Prioritization of Requirements for Testing (PORT) technique and its critical factors Customer priority Developer's priority Traceability, and completeness algorithms	Limited support of industry-based case studies papers	√	√	2018
[19]	Regression test case selection	Requirement coverage with multi-objective criteria Minimum cost of execution time	No support of industry-based case studies papers	√	√	2017
[20]	Regression testing	–	Limited support of industry-based case studies papers	√	√	2016
[21]	Agent-based test generation	–	No support of industry-based case studies papers	×	×	2018
[76]	Regression testing	–	Limited support of industry-based case studies papers	√	√	2012
[85]	Test case prioritization of systems	PORT technique without brief discussion on its factors Requirement Clustering Requirement-risk aware TCP	Limited support of industry-based case studies papers	√	×	2018
[86]	Testing of semantic web services	–	Limited support of industry-based case studies papers	√	×	2018
[89]	Regression test case prioritization	PORT1.0 with four factors Quota constraint coverage requirements	No support of industry-based case studies papers	√	√	2012
[92]	Mutation testing	Mutation testing support for quality assurance	Very limited support of industry-based papers	×	×	2018
[107]	Test case generation (TCG)	Story generation from customers' specified requirements	State the importance of TCG	×	×	2018
This SLR	Test case prioritization	Extended versions of PORT techniques such as (PORT1.0), and (PORT2.0) Identification of regression testing aspects Text mining information in TCP Slicing algorithms for identification of affected nodes of FRs in TCP Empirical information of test case and defects Linkage metrics in test case prioritization	Full support of industry-based case studies papers	√	√	–

In the second SLR study [17], researchers explore the web services architecture and discuss how the process change results in the modification of functional requirements. However, researchers in [17] did not cover functional specifications of web services as a separate strategy that is used to prioritize test cases when web services are added or integrated. They have presented test case prioritization to an abstract level regarding functional requirement-based TCP. At abstract level, FR-based TCP means that no specific information about implementation of the proposed techniques have been provided. On the other hand, a FR-based TCP approach at detailed-level provides the precise explanation of proposed approaches and their implementation. They identified seven challenges of web services regression testing other than the challenges we have presented under RQ2 in results and discussion section. Therefore, we have updated the list of issues/challenges of test case prioritization in regression testing.

In the third SLR [18], researchers have mentioned three core studies in the area of requirement-based TCP approaches. In the same SLR study [18], researchers have verified that it has been claimed that a requirement-based TCP approach gives better results if maximum number of factors such as (CP, RC, FP and DP) are used. Among three studies, two studies are from the same group of researchers who have undertaken their research by using the requirement properties. To resolve the bias produced by the same group of researchers in using the requirement properties, we include other primary studies and compare the effectiveness of proposed TCP approaches.

In the fourth SLR study [19], researchers have covered functional requirements but use only two core primary studies for requirement coverage. Authors of the lateral-mentioned SLR provide a very specific definition of regression testing that verifies the specified requirements. They use coverage-based criteria instead of using requirement-based criteria to proceed with their survey study. As regression testing is a multi-facet research area, they preferred to use coverage criteria rather than using requirement criteria. In the fifth SLR, Rosero et al. [20], implicitly discuss the software functionalities, and identify 31 TCP approaches in the context of regression testing adaptation, identification of new algorithms, optimization of algorithms, and use of the artificial intelligence in TCP approaches. Until 2015, TCP approaches were not so generalized in academic and industrial settings as compared to date, because complexed nature of software presented barrier to generalize TCP approaches in industrial settings. Our SLR identifies TCP approaches, and their applications for academic and industrial projects.

In the sixth SLR study, Arora and Bhatia [21] discussed agent-based regression testing. Researchers identified regression testing approaches, methods and platforms. Moreover, they identified six testing categories, and functional testing as one of these categories pertains to specification of "System under Testing" SUT. They stated that test cases were derived from the specification and researchers missed to discuss that how various requirement properties help in test case prioritization. Rest of the SLR and survey studies including [76, 85, 86, 89] did not provide enough knowledge about FR-based TCP approaches. Therefore, researchers have potential to update the knowledge about recent development in FRs-based TCP in the literature.

Primary Studies

Gallagher et al. [22] developed the interclass testing technique based on the "Unified Modeling Language" (UML) diagrams, which was regarded as a data-flow-based approach. This data-flow-based approach determined the path used which made the input to test the software components. This testing technique affected the transparent implementation when it was used with the other models. In a recent study, Mohanty et al. [23] used a debugger to detect the faults as soon as possible. Their proposed technique was evaluated on a component-based system, where UML state chart diagram represented the state changes in the component-based system. The interrelation among the components was described by the conversion of UML state chart diagram into a component interaction graph. To further explore the idea of using UML 2.0 for test cases generation, Kundu et al. [24] proposed a metric named as "Rule-Based Matrix" RBM to generate, and manage test cases. This TCP approach has been proposed using the UML design specification and researchers have missed out the code and its characteristics such as the statement, relevant slice, function, control flow and data flow information. Zhang et al., [25] proposed TCP approaches alongside additional and total strategies, and researchers have used a number of covered and total elements per test for prioritization of test cases. Moreover, authors in the lateral study unified two strategies and showed that unified strategy of TCP outperformed the individual total and additional strategies.

In an earlier research work, Bryce and Colbourn [28] adapted the greedy method of 'one test at a time' for the pair-wise testing and tested the most important test cases among the large number of test cases. The main purpose of proposing the interaction testing was to screen out the faults. Faults screening enables the software testers for earlier detection of faults that is necessary for the successful deployment of the software. Test cases are used to detect the faults during the process of retesting. The prioritization of test cases was based on greedy method, which has been applied to a paired test case that enhanced the earlier detection capacity of the interaction testing. The proposed greedy method of TCP helped the researchers to provide simple and useful means to generate test cases. To accomplish this

research work, empirical studies were required to evaluate the performance of interaction testing. Before this research work was undertaken, Korel et al. [29] presented the state-based model for prioritization of test cases. The proposed model was found to be inexpensive, and with less overhead did not require the execution of the whole system. The proposed TCP technique in [29] remained successful for small systems and the same TCP technique required to be tested for larger systems. In this regression testing research area, a major contribution by Voinea and Sassenburg [30] was the proposal of model-based testing to achieve a slow acceptance in the market because model-based testing required the specification of the full behavioral functionality of a system.

Salehie et al. [26] used the regression testing to avoid the problems produced from changes in software evolution. Changes in software requirements violated the requirements in the previous release of a software. They adopted the "Goal Question Metric" (GQM) that aimed at setting the priority of test cases. Moreover, researchers in [26] found that interaction testing was widely used for the screening of the systems, however, budgetary and time constraints did not allow testing of the entire suite. To overcome this issue, TCP approach proposed in [6] used configuration model to extract the functional metrics. This multi-objective TCP approach was initially focused on the highly configurable systems and could be extended to other large scale systems. Prior to lateral discussed TCP approach, Arcaini et al. [27] also used feature model to propose a fault-based approach. Compact test suits were generated by involving this TCP approach. As a result, the number of test cases were reduced which detected faults in short duration.

To improve the fault detection rate in regression testing, total coverage requirements were concerned with the test cases. However, this phenomenon of total-coverage requirements consumed more cost and time for regression testing. In order to overcome this issue, Jaffarur-Rehman et al. [31] proposed slicing-based three heuristic TCP approaches and compared their performance with traditional TCP techniques. It was found that using the relevant slices for prioritization of test cases improved the fault detection rate. This illustrated that test size used in the experiments was smaller as compared to that used for other applications.

Chen et al. [32] considered the importance of random testing technique and modified the random testing technique for the execution of similar neighboring test cases. To accomplish this task of test case prioritization, researchers proposed the "Adaptive Random Testing" (ART) technique that was efficient to detect the first failure in software. The use of the ART in [32] has been advantageous as compared to 'Random Testing' RT technique because former ART technique involves a few test cases for detecting faults in test suits. However, fault detection capability of ART was found to be better than RT in terms of thoroughness and high coverage of percentages for fault detection. The ART was applied to numerical programs by researchers and it is required to apply the same ART for larger non-numerical programs in future research studies.

Zhang et al. [33] described the prioritization of test cases as a process to determine the fault rate, with a greater focus on fault detection for prioritization of test cases. They proposed two techniques for TCP and evaluated both proposed techniques. They found that their proposed techniques were superior to traditional TCP techniques. However, their proposed technique still needs more improvement regarding the test case prioritization to achieve the desired outcomes. Additional techniques proposed in the same study should be extended for larger systems in terms of time and cost. Time-aware TCP approach shows its limitation for total coverage because the proposed TCP approach has been not tried for the additional coverage. In a recent research work, Kumar and Chauhan [34] have taken dependencies and coupling between modules to propose a TCP approach for identification of modules that were highly affected by changes in other modules. The proposed TCP approach resolved the issue of identifying the critical errors that were propagated by changes in other modules.

In another research work, Maia et al. [35] considered the importance of software testing, but they were concerned with the execution of all test cases. To overcome this issue, they proposed a requirement-based TCP approach for prioritization of test cases. This requirement-based TCP approach was efficient in prioritizing the test cases as it did not require the prior knowledge about code and search-based TCP technique for prioritization of test cases.

According to Iskrenovic-Momcilovic and Micic [36], the purpose of software testing was to improve quality, verification, and validation of testing. The quality of a software testing has three dimensions, which are given as functionality, adaptability, and engineering. All three dimensions can be further classified into various factors, and each factor varies from application to application development.

As seen in the existing literature, researchers in a number of studies [37–39] used coverage-based criteria to propose test case prioritization approaches. A discussion about the coverage-based technique and a number of other criteria-based technique is out of scope of this SLR. Therefore, we keep discussing FR-based techniques. Moreover, we have a number of well-known SLRs in the research area of regression testing. Among these SLRs, four have particularly focused on test case prioritization, and rest of SLRs discuss regression testing. Although code coverage has been considered by a majority of researchers in the existing studies, in some cases code of system under testing is not available. Therefore, researchers consider sources other than code to propose their TCP approaches. Functional requirements of a system is one of these sources that we consider for this SLR.

We have chosen to only review primary studies on functional requirement-based TCP approaches in regression testing. There is no such SLR that discusses functional requirement-based TCP approaches. In the following section, researchers have provided the details for research method that has been used to conduct this SLR from the existing literature on functional requirement-based TCP in regression testing.

Research Method for Systematic Literature Review

The SLRs have been widely used as an instrument by researchers to identify, evaluate and interpret the existing research studies on particular research questions, as well as topic areas [40]. An explicit process is needed to conduct an SLR, and hence, researchers of this SLR have followed the guidelines recommended in [40–42], which are more appropriate for conducting SLRs in the area of software engineering. These guidelines include the enclosed three phases which are planning, conducting, and reporting reviews as shown in the Fig. 2. The following Fig. 2 gives us a clear description of this review process with the essential phases.

Figure 2 shows three phases with the activities performed in each phase. In the following, we provide a short description of each of three phases as follows:

- **The planning phase:** To plan the process review is a first phase, which includes activities that serve as refining research questions, while developing a review protocol. A quality assessment criteria also has been mentioned in the same phase of the process review.
- **The conducting phase:** This phase is mainly for selecting primary studies, which are identified after each refinement. The researchers of this SLR agreed to scan contents of primary studies through the consensus meetings. The contents of primary studies include title, abstract, keywords, conclusions, and other contents. Also, studies include discussion on the requirements and feasibility analysis. Data are extracted and maintained in the final step of the conduct phase.
- **The reporting phase:** The final phase of review process covers the synthesis of data to achieve review outcomes, which are discussed and concluded.

Research Questions

As stated by Kitchenham et al. [42], it is significant for all types of reviews to specify the research questions. We proposed research questions, and categorize them into two categories as given in the following Table 2:

Table 2 is the illustration of investigating various subsets of FR-based TCP approaches in the literature. The inclusion of various subsets is aimed to collect empirical and analytical evidences that may help to summarize strengths and weaknesses of primary studies. This SLR attempts to collect the empirical and analytical research on functional requirements and their role in conducting the regression testing. Also, empirical subsets can help in understanding the replication of results regarding TCP approaches and analyze problems encountered.

The next step is to establish a set of keywords in order to find the relevant primary studies on functional requirements-based TCP regarding regression testing. Finding the relevant primary studies helps researchers in answering the above-given "Research Questions" (RQs).

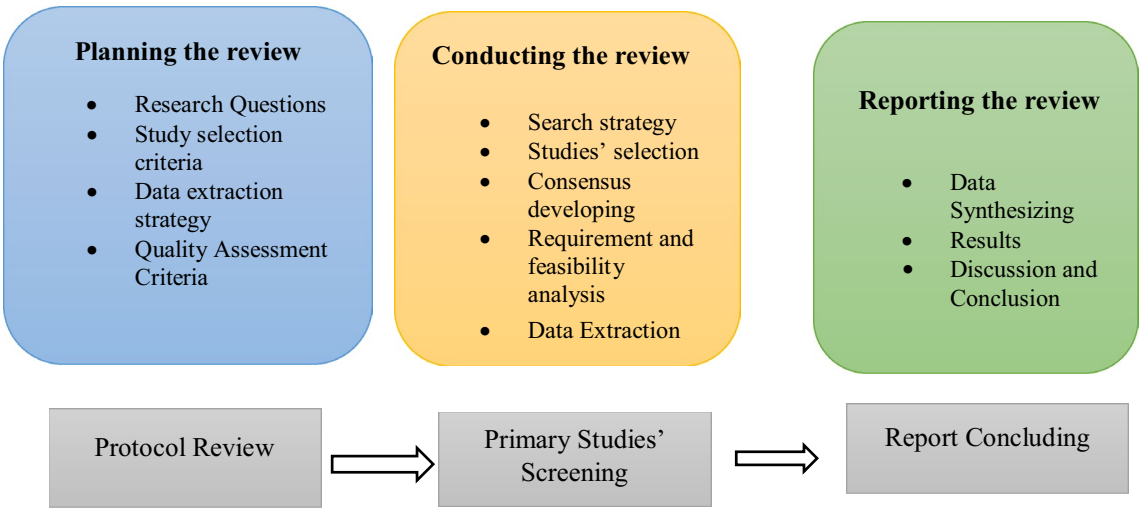

Fig. 2 Three phases of systematic literature review

Table 2 Research questions

Category	Research question
General FR-based TCP approaches	**RQ1.** What are the state-of-the-art techniques of functional requirement-based TCPs?
	RQ1.1: What are key requirements prioritization factors discussed in the selected primary studies?
	RQ1.2: What are important regression testing aspects considered for proposing and evaluating the TCP approaches?
Empirical FR-based TCP approaches	**RQ2.** What are regression testing issues that have been addressed by the proposed FR-based TCP approaches?
	RQ3. What are test case size and defects types that relate with each other in the selected primary studies?
	RQ4. What are linkage metrics used in the selected primary studies?
	RQ5. What systems under testing (SUT) are used to validate the requirement-based TCP approaches?
	RQ6. What are domains of systems under testing (SUT) used in primary studies?
	RQ7. Have primary studies been tried in the real-world settings or not?

Search Keywords Identification

In order to find relevant studies, the use of search strings within the popular digital libraries is very important. We selected databases including IEEE Xplore, Scopus, Taylor & Francis Online, SpringerLink, Wiley Online, ACM digital library, and World Scientific. Moreover papers related to our research topic were found in these databases. We used these databases on the basis of our experience for retrieving the primary studies. Other than these databases, dblp library was searched for papers that resulted in a few papers which were already covered by above-mentioned databases. We tested search strings and retrieved a number of relevant primary studies and survey studies. The researchers have used search the same strings in the well-known digital repositories, and executed search strings, which have been given as follows.

The search strings used in this SLR contained the combination of test case prioritization, regression testing, and functional requirements. Table 3 shows the search strings which were used in the digital libraries. We find that the same search string has different formats for used digital libraries. We observed that 'OR' and 'AND' operators were shown when using search strings on Springer and Scopus digital libraries.

Inclusion and Exclusion Criteria of Primary Studies

In the following, we present the inclusion and exclusion criteria of primary studies for this SLR.

Inclusion Criteria

Based on titles and abstracts, keywords, and conclusions of the primary studies, the "Inclusion Criteria" (IC) for primary research studies have been given as follows:

- **IC1**. Research studies in which regression testing has been discussed alongside test case prioritization.
- **IC2**. Research studies in which test case prioritization has been discussed.
- **IC3**. Research studies in which systems under testing that use requirements have been discussed.
- **IC4.** Research studies in which functional regression testing has been discussed.
- **IC5.** Research studies which discuss issues pertaining to requirements of system regarding regression testing or TCP.

If any of the above-given points were found in title, abstract, keywords, conclusions of research studies, those studies were included in this SLR.

Table 3 Search string applied to chosen repositories

Repository	Search string
IEEE Xplore	(((Test case prioritization) OR Regression Testing) AND Functional Requirements)
Springer	'Test AND case AND prioritization OR "regression testing" AND (Functional AND requirements)'
Scopus	ABS (test AND case AND prioritisationORregression AND testingANDfunctional AND requirements)
Taylor & Francis Online	[All: test case] AND [[All: prioritization] OR [All: regression]] AND [All: testing] AND [All: functional] AND [All: requirements]
ACM	(+test+case+prioritization) OR (regression+testing) AND (functional+requirements)
Wiley Online Library	"Test case prioritization" anywhere and "Regression testing" anywhere and "Functional requirements"
World Scientific	[All: test case prioritization] OR [All: regression testing] AND [All: functional requirements]

Exclusion Criteria

We have also presented an "Exclusion Criteria" (EC) as given in the following:

- **EC1**. Research studies which do not focus on regression testing
- **EC2.** Technical reports, Master and Ph.D. dissertations, and other studies of less than 4 pages were not included because sufficient information regarding test case prioritization was required.
- **EC3.** It was observed that some researchers published journal papers which were the extended versions of their workshop and conference papers. We excluded the earlier versions of workshop or conference papers of journal articles because research articles contain more approaches details, more experiments details, and more evaluation details.
- **EC4.** Primary studies which only contained conceptual contribution were also excluded because detailed experimental results, and discussion are not available.

Books chapters, Master and Ph.D. dissertations, and technical reports were not considered for current SLR as a complete review process is not undertaken for these studies. To assess the quality of selected primary studies, the quality assessment criteria used is given in the following section:

As suggested by Kitchenham et al. in [40], quality assessment criteria minimizes the bias while ensuring the external and internal validity of research studies. Therefore, we use quality assessment criteria which have been widely applied in a large number of SLRs. The quality assessment criteria is based on a few quality assessment questions which are as given in the following Table 4:

Each question in the above-given Table 4 has been scored on the scale from "Yes (1)", "Partly, 0.5" to "No, 0". Scores for all questions were summed up. We have involved other researchers in assessing the primary studies and increasing the reliability of this SLR.

Search Process

This research study is a systematic literature review of recently published papers on a certain research topic for a given period of time between 2009 and 2019. Engstrom et al. [43] stated that the so-called state of knowledge is the review of the literature which is aimed at identifying the aspects of regression testing and to examine how studies have been conducted. This is also aimed to present the overview of recent publications on requirement-based TCP in regression testing. To accomplish the objectives of this study, we have conducted the search in the databases as given in the following Fig. 3.

Since, we were interested in primary studies which employed key terms such as test case prioritization, therefore, keywords 'test case prioritization', 'regression testing', and 'functional requirements' were used English. These keywords should be in abstract, title or in articles.

Data Extraction

To extract data, primary studies in the final search underwent a process. 35 primary studies were inspected in detail regarding the content assessment to obtain results of all research questions. Data was collected on two different sheets. The first sheet contains the qualitative information about 35 selected papers. This qualitative information is taken from title, abstract, and main contents of the papers. The summary of papers have been given in the next section of Results and Discussion. In the second sheet, quantitative information taken from the selected primary studies was entered i.e. number of TCP approaches, how many approaches addresses the concerning issue of regression testing, and size of SUT. In the next section, we present the SLR's results and discussion from selected primary studies.

Results and Discussion

This section is focused on outlining the results of research questions. Before, answering to specified questions, we present summary of our selected primary studies specific to

Table 4 Quality assessment criteria questions

No	QAC question
1	Does the study explicitly involve regression testing and particularly test case prioritization?
2	Does the study use proposed functional requirement-based TCP?
3	Does the study show validity of proposed TCP technique?
4	Does the study contain information about systems which have been tested for relevant proposed TCP approaches?
5	Were the test cases adequately described in the study?

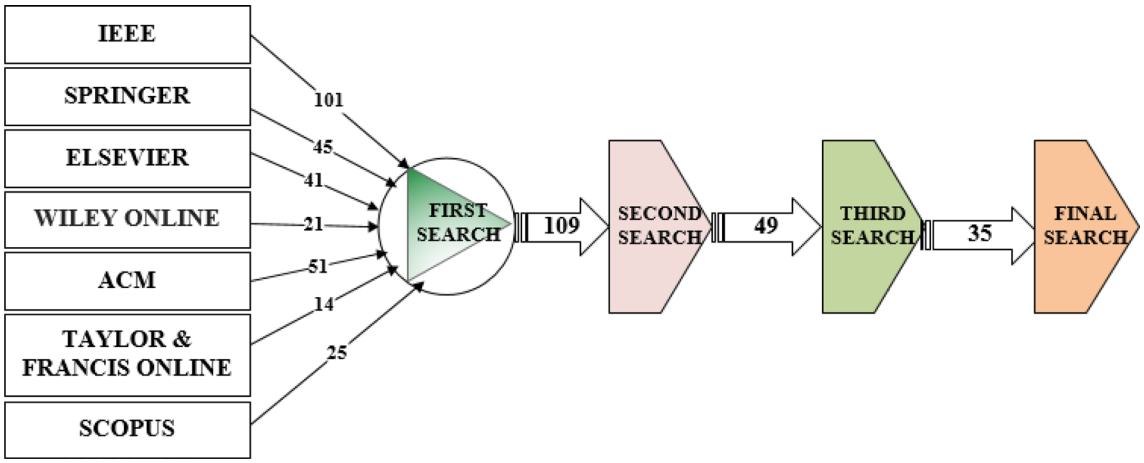

Fig. 3 Selection procedure of primary studies

functional requirements-based TCP as shown in the following Table 5.

Table 5 shows main research themes covered in the selected primary studies. Most significant themes include severe faults detection, requirement coverage, code coverage combined with the requirement coverage, historical and dissimilar information of requirements etc. 35 primary studies were nominated for this SLR. Total published papers per year are presented in Fig. 4.

Figure 4 shows that from 2009 to 2019, a total number of 35 studies were published on research topic functional requirement-based TCP approaches. It is observed that research publications for this SLR are not found in the high numbers from 2009 to 2015 because several studies were published on the other types of regression testing. Figure 4 illustrates that the maximum number of publications were found in year 2016. However, we cannot determine that research work is not taken on regression testing before 2009. The cause behind the absence of research papers before 2009 might be the interest of research community on other criteria-based types of TCP in regression testing. However, more number of FR-based approaches have been proposed after 2015 (see Fig. 4). FR-based TCP studies gained the highest attention of researchers between 2016 and 2018. It also surged during 2012. We can conclude that researchers are active on test case prioritization. Therefore, FR-based TCP approaches have been continuously proposed to address TCP problem.

RQ1: What are the State-of-the-Art Techniques of Functional Requirement-Based TCPs?

The discussion of RQ1 is longer than that of other RQs, as it covers the literature on functional requirement-based TCP approaches for addressing the TCP problem.

Tsai et al. [61] proposed a scenario-based regression testing that used traceability matrix between affected components and associated test cases. The scenarios were mainly based on different inputs (normal and abnormal inputs), and exception handling. Ripple effects were used to identify and analyze all affected scenarios. Proposed approach was one of the earliest TCP approaches that presented the need to use functional requirements and test them from different perspectives. In line with the later study, [21, 31] also explore the component traceability, which is specific for behavior, internal state, and execution of components. Component behavior is compared with the implementation in black box testing, while interface invocation is checked by the internal state of a software component in white box testing.

In the following, we give an overview of the influencing factors of TCP approaches such as time, funding, and market pressure.

Lack of Time and Funding, and Market Pressure

Time, and cost were stated as the major resources consumed for retesting of a software [23, 29]. However, researchers in [24] state them as time and cost constraints, which have been highlighted by the industrial collaborator in [37] and since the latter published study, time and cost are continuously focused in primary studies in regression testing. Besides, engineers, and testers have more market pressure for addressing the retesting of systems because of evolution in systems. Therefore, we discuss three constraints and their impacts regarding TCP approaches in as given follows:

Do et al. [88], conducted a series of experiments to assess whether time constraints have impacts on the cost and benefits of TCP optimization. They used four time constraint's levels (TL-0, TL-25, TL-50, and TL-75). The first TL-O level pertains to a scenario where no time constraint is

Table 5 Summary of selected primary studies

Sr. no	Reference Id	Research idea	Paper's venue	Year of publication
1	[9]	Requirements risk estimation using complexity, security, size and requirements modification attributes	Journal	2016
2	[44]	Requirements-based TCP approach	Conference	2010
3	[45]	Metric-based TCP approach	Journal	2011
4	[46]	Proposed a system level approach using requirements properties	Journal	2012
5	[47]	A multi-objective TCP approach that involves links between requirements, code and test cases	Conference	2012
6	[48]	A TCP approached incorporating clustering and requirements and code information	Conference	2013
7	[49]	Proposing requirements risk-based TCP approaches	Journal	2016
8	[50]	Proposing a TCP approach by involving requirements correlations	Conference	2016
9	[51]	Proposing a TCP approach using requirements as historical data	Conference	2016
10	[52]	An extended metric-based approach that identifies fault-prone parts of a software	Journal	2016
11	[53]	A TCP approach that uses requirements coverage, faults detection and execution time	Conference	2016
12	[54]	Proposing a metric-based TCP approach using requirement complexity, code complexity, and code coverage	Journal	2016
13	[55]	Proposing a TCP approach that uses user stories and clusters of test cases for regression testing of agile projects	Journal	2017
14	[56]	Proposing a TCP approach that uses dissimilarity historical information between various versions of software	Conference	2017
15	[57]	Proposing a TCP approach by associating test cases with the emerging requirements	Conference	2015
16	[58]	Requirements specification to model test case prioritization for identification of software severe faults	Journal	2009
17	[59]	Test cases are viewed as requirements for agile projects	Journal	2016
18	[60]	Genetic algorithm covers full coverage of changed requirements	Journal	2018
19	[62]	Traceability use for proposing a 'genetic software engineering' method and prioritizing functional requirements	Conference	2010
20	[63]	Detection of dependencies between requirements for test case prioritization	Conference	2018
21	[64]	Dependencies between requirements and incorporating elements like test cases, faults, cost, and test requirements	Conference	2017
22	[65]	A hybrid regression testing prioritizing approach that uses ant colony genetic algorithm and requirements properties	Conference	2018
23	[66]	Test case prioritization approach from requirements and code information	Conference	2016
24	[67]	Association rule mining and requirement modeling	Journal	2018
25	[68]	Dependency values between requirements for proposing a regression testing approach	Journal	2012
26	[76]	Regression testing with its three types	Journal	2012
27	[82]	Automated regression testing	Journal	2017
28	[83]	Regression testing process for evaluation of software protocol specification	Journal	2011
29	[85]	Test case prioritization approaches	Journal	2018
30	[86]	Semantic web services test case prioritization	Journal	2018
31	[87]	Search-based test case prioritization	Journal	2019
32	[88]	Time constraints and its impacts on test case prioritization	Journal	2010
33	[89]	Regression test case prioritization approaches	Journal	2012
34	[90]	Selective regression testing	Journal	2019
35	[99]	Model driven regression testing	Journal	2015

required for completion. On the other hand, TL-25, TL-50, and TL-75 represent scenarios where 25, 50, and 75 percent amount of testing is completed. Every testing process can have a time constraint. Both the process time constraint and market pressure force testing teams to suspend the testing before all test cases are executed. This helped testing teams in the identification of earlier faults. In spite of, its benefits in testing, the main limitation of this technique is that omitted test cases due to time constraints' levels could have faults which remain unrecovered.

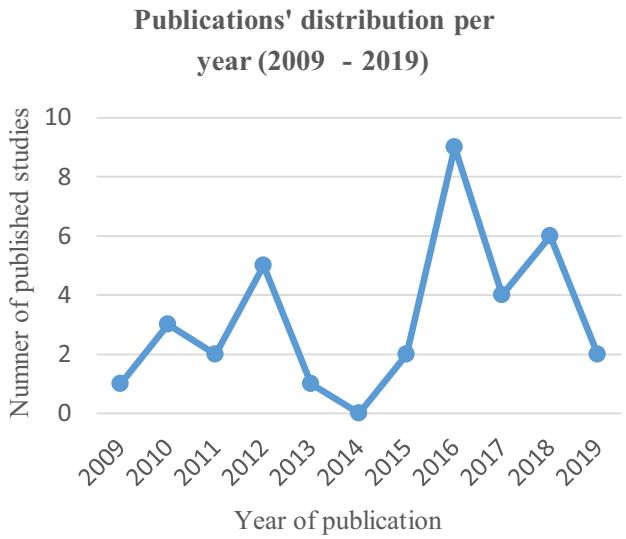

Fig. 4 Year wise published studies

Furthermore, Wang and Zeng in a primary study [51], examined the historical data to prioritize the test cases. They found that all test cases could not be executed due to cost and time constraints, therefore, they assigned probability values to test cases, and executed only those test cases with the high probabilities. The high probability is more relevant to requirement prioritization. The lateral given TCP methods outperform the random TCP in terms of "Average Percentage of Fault Detected" (APFD) value, but redundant fault issue has been not solved. The redundant fault issue was produced when faults were subdivided into each requirement.

Functional verification has been pointed as bottleneck in [91], for designing the embedded systems. Therefore, the simulation based functional verification is practiced due to its scalability and flexibility. However, dependency of quality functional tests on coverage criteria determines the success of functional verification. A clustering TCP approach proposed in the same study with the objective of achieving high coverage used three enhanced K-Means clustering algorithms. The problem of functional verification is addressed by re-ordering of test cases through enhanced K-Means clustering algorithm. The proposed TCP approach showed 90% saving in the simulation time. The main limitation of TCP approach was the k value (number of clusters), and hence future work can be taken to improve k value.

Since, time and cost remained the primary focus point of researchers in [36–39], and none of the earliest studies discuss the market pressure regarding TCP approaches. We found that an excellent research work was undertaken by researchers in [95] in which they included the market users' demands for large scale systems. They proposed fast fault detection TCP approach that met the fast delivery of a validated system. However, for every single change, it is impractical to retest the system. Since, a large scale system may require thousands of test cases to be re-executed. To overcome this issue, Hemmati et al. [95] proposed their work on the comparison of text-based, topic-based and history-based TCP technique. For the evaluation of three techniques, they used Mozilla-Firefox versions 5–13. Results showed that none of three techniques were dominating the other techniques. However, history-based techniques showed better results as compared to text-based and topic-based techniques in terms of fast fault detection within the short time of test case execution.

Recently, Arrieta et al. [87], found that "Cyber-physical Systems" (CPSs) with the high configuration requirements show a large variability. As a result, millions of configurations may be required to meet the customers' needs. It is, therefore, effective test processes are required to optimize the working of software product lines. The proposed search-based TCP approach remained effective in faults detection time, simulation time, and coverage of the functional requirements than the random TCP approach. Search algorithm used in testing reduced the fault detection time by 47%, and simulation time by 23% and functional requirements by 22%. For tight budget, random ordering TCP approaches are more appropriate [39], and rest of the TCP approaches become less stable because other than random TCP cannot guarantee the all faults detection within the limited budget. However, we observed that TCP approaches in [51, 87] show better time saving for faults detection than the random TCP approaches.

In another recently published study, Andrews et al. [90] extended their "Finite State Machine Web" (FSMWeb) behavioral and structural model to address the multiple changes in systems. They classified the failure mitigation tests into reusable, re-testable, and obsolete. With the objective of a full coverage, new tests were also regenerated. The proposed selective regression testing approach remained more efficient for failure scenarios of a specific function or portion of an application. However, the proposed approach need further work to cover more functions in an application.

From the analysis, we observed the FR-based TCP approaches show their advantages to other TCP approaches in terms of time and cost saving for test cases execution and faults detection. Moreover, FR-based TCP approaches show fast delivery of validated systems to increase users' demands in the market. Consequently, the merit of FR-based TCP is relied on the several prioritization factors which have been presented in the following.

Use of Factors to Prioritize Test Cases

Studies [23, 36, 39] earlier discussed were attempted to reduce time and cost factors. Both of these factors can be called as objectives of TCP approaches, as mentioned later

in [95]. We present FR-based TCP approaches that use other factors intending to achieve TCP objectives.

Kavitha et al. [44] proposed a requirement-based TCP approach with the aim of revealing more faults regarding the specifications of software. They focused on three important factors, including code implementation-related complexity, customers' priority for requirements, and changes in requirements. The same proposed TCP approach shows better results as compared to random ordering-based TCP approach. In spite of focus given on requirements' specification, a good achievement regarding the fault detection rate and fault severity has been missed in this earlier primary study. Previously several TCP approaches were shown with the priority factors of test cases [30–39] and did not mention the FR-based priority factors. For instance, Srikanth and Banerjee [46] proposed 'Prioritization of requirements for test' PORT 1.0 as a TCP technique which was based on four factors. These factors included as given in the following:

- Customer priority (CP)
- Fault propones (FP)
- Implementation complexity (IC)
- Requirement volatility (RV)

Priority of test cases execution was set according to score attained from each factor, and summing the values of four factors [46]. Although PORT 1.0 provided useful results for applications under the same domain, it showed its limitations when it was applied to large scale systems. To overcome this limitation, the extension of the PORT 1.0 TCP technique was delegated to a future research work on TCP. Srikanth et al. [49] extended the PORT 1.0 TCP technique for prioritizing test cases, and proposed a new PORT 2.0 TCP approach for regression testing of the enterprise level software application. In the newly proposed PORT 2.0 TCP approach, researchers used only two factors, FP and CP. In the same study, another risk-based TCP technique was also proposed for prioritization of test cases using the risk associated with the requirements. From PORT 1.0 results, CP was found to be the biggest contributor to detecting the faults. However, the latter proposed PORT 2.0 approach with the combination of CP and FP outperformed random TCP approaches in faults detection. The risk-based TCP approach proposed in the same study also determined the "Risk Exposure" (RE). Prior to proposed TCP approaches [44, 46, 49], the earliest study [37] highlighted the detection of high-risk fault from test case execution. In comparison with the latter study, the proposed TCP approach in [49] used "Risk Likelihood" (RL) and "Risk Impact" (RI) to determine the RE of each requirement. Furthermore, RE values of requirement categories are used for prioritization of requirements categories. Test cases corresponding to each of prioritized requirement categories are then prioritized for a final list.

In the context of risk-based TCP, Hettiarachchi et al. [9] have examined the assessment of requirements, and include four factors such as "Requirement Complexity" (RC), "Potential Security Threats" (PST) "Requirement Size" (RS), and "Requirements Modification Status" (RMS) to propose a risk-based TCP approach. Although some of requirement criteria factors vary in both primary studies [9, 49], the determination of risk exposure has been commonly studied in both primary studies.

Functional requirement correlation has been studied by Ma et al. [50] for prioritizing the test cases dynamically during execution of test cases. Perceived priorities of customers and developers were made essential factors that contributed to proposing a TCP method. To reorder the test cases, weights were assigned to each test case that helped to increase the testing efficiency. Prior to assigning the weights to each test case, a 'Functional Dependency Graph' (FDG) was established as per specifications of functional requirements. A path in FDG is known as the relationship between a pair of requirements, so each path between two requirements is known as a test case. Findings of this lateral primary study are in line with studies [9, 49] regarding the factors used for proposing the TCP approaches but does not involve the cost and severity of each test case. However, a functional requirement with its detected faults is assigned a high priority and code of relevant defected part of a system is changed which introduces new faults.

From the analysis, we have found some common criteria of defining test cases from functional requirements in studies [9, 46, 49]. This criteria is the use of test case factors derived from functional requirements. One of these factors is requirement change RCi as given follows:

$$RCi = \frac{\mathrm{N}}{\mathrm{M}} x10. \tag{1}$$

where N indicates the change occurring in a requirement i times, and M is the number of total number of requirements. If a requirement is changed more than 10 times then the volatile range of all requirement is calculated on 10-times scale, otherwise the range remains between 1, and 10 scale values.

Use of Text Mining and Clustering to Prioritize Test Cases

Arafeen and Do [48 2013] investigated the TCP approach that incorporated the information from requirements. A text-mining technique was used as means for clustering the relevant requirements, incorporating the code complexity of each cluster to prioritize test cases, and finally resulting into a set of test cases with a new order. By using the text mining approach, clusters of requirements were composed by calculating the frequency of terms in a requirement document. Code complexity was also incorporated in clustering the functional requirements of the system, which created a

new set of reordered test cases. Thus, requirement clustering is added in the existing literature on TCP. We also see that Haratay et al. [54] used requirements-based clustering approach to improve the number of faults. Bother of the latter-mentioned approaches achieve a higher number of test case execution.

Proposed TCP approaches show the increased reliability when requirement and code of SUT are available. However, a TCP approach is not scalable for web based system as code cannot be accessed. For instance, researchers in [24] examined test cases clustering based on the dissimilarity between profiles of test cases. The latter-mentioned TCP approach is more appropriate for reduction of test cases, and does not show prioritization of test cases. However, we find a primary study [26] where researchers have chosen the clusters of test cases with the worst cases, and test case clusters with the best cases and similar coverage of requirements were not considered. This is because the worst-case strategy resulted in the prioritization of test cases with the best cost.

From the analysis, we observed that researchers have missed the applications of their proposed TCP techniques for larger industrial systems, with the result that it has become a requirement for future studies to include the larger systems for regression testing.

Model-Based Test Case Prioritization

We present discussion of model-based TCP approaches derived from functional requirements.

We find several studies [22–25] in the existing literature which deal with regression testing by modeling TCP approaches. For instance, the earlier study [28] deals with TCP from users' profiles, and operational profiles, and study [27] deals with test case generation by proposing feature models. The latter-mentioned studies do not focus on the specifications for proposing TCP approaches.

To address regression testing from specification modeling, Mary and Krishnamoorthi [45] have proposed a metric-based approach on the varying priorities of requirements, test cases, and execution time of test cases for prioritization of test cases. The main aim of proposing the "Average percentage of Severe Fault Detection per Time" ($ASFD_T$) metric [45] has been to evaluate the prioritized testing empirically, depending on the varying prioritization of requirements. Results of the proposed metric indicated that all other techniques (function coverage, statement coverage and requirement-based prioritization) outperformed the random ordering technique. Hence, they extended APFD metric and formulated an $ASFD_T$ metric that showed better results for prioritized test case execution than the random prioritization. The central point of this proposed $ASFD_T$ metric was to measure the fault severity in the context of time and cost. The proposed $ASFD_T$ metric was also focused on measuring fault severity and their distribution for requirements. However, this $ASFD_T$ metric-based approach has been not yet applied to other coverage criteria-based techniques. Krishnamoorthi and Mary [58] investigated that how model-based TCP improved software quality. The model-based TCP used five key requirements properties and their values. Fraction value of each requirement was mapped to test cases, and test cases with higher values were executed before test cases with low values. Test cases with the severe faults could be detected rather than using random ordering of test cases.

Liu et al. [69] proposed a TCP technique combined with two parts. In the first part, initial prioritization of test case before execution was determined. In the second part, dynamic test case prioritization was done by adjusting the software failure. To evaluate the proposed TCP approach, researchers twice used the same168 test cases in their experiments. For the first time, authors have run no-sorted test cases, and the second time, they have reordered the test cases using the algorithm and adjusting the defects. So, authors in the same study identified the defects at the earlier phases that enabled the developers to avoid the faults. In a recently published research study, Mahali and Mohapatra [67] have identified the respective nodes of modified requirements and traced the affected nodes using the slicing algorithm. Researchers developed a pattern through mining of data from various nodes. This research remained in addressing the early fault detection and only functional requirements were undertaken. Both of these studies used algorithms to address the fast fault detection which is the primary objective of TCP approaches. However, we still need to investigate the regression testing regarding non-functional requirements in future works.

Marchetto et al. [52] proposed a TCP approach based on three artifacts namely codes, requirements, and test cases. In addition to these three artifacts, traceability links were used for identifying the most critical and fault-prone parts among the given three software artifacts. Authors have shown the behavioral view of their proposed approach in terms of objects' flow in a UML activity diagram. In fact, a link is established between requirements and source code that is used to identify the potentially fault-prone and critical sections. Credibility in this proposed technique is comprehensively focused in addressing the various issues regarding TCP in regression testing. The robustness issue, which has been not addressed by a majority of researchers, remained a fundamental objective of Marchetto et al. [52] and they verified how the proposed TCP approach worked in case of wrong/spurious traceability links. Experiments on 21 Java applications validated the effectiveness of the automated artifact analysis as well as weighting in TCP [52].

In the recent work Garousi et al. [60] proposed a custom-built genetic algorithm to provide a complete coverage of changed requirements. This research was aimed at improving

regression testing of critical defense software applications. More efficient test suits with the higher number of faults were gained through this approach. In comparison with UML test model in [31], the chosen studies [52, 60] particularly focus on the fault-prone parts of a system using UML modeling. However, the proposed approach [52] could not be evaluated regarding mutants which were not permitted to be injected in the chosen software applications. It could be a future direction for researchers to evaluate the proposed approach on systems which are feasible for mutation testing.

Haraty et al. [54] proposed a new change-based TCP method for generating the test cases and rerunning these test cases to ensure that software were bug free. Newly proposed TCP method is based on clustering, which is derived from three important test case factors: code complexity, "Code Coverage" (CC), and "Requirement Complexity" (RC). Last test case factor as RC is one of the primary complexity for TCP in regression testing. To overcome the RC issue, researchers in the same research study, prioritized the clusters of test cases and involved CC and RC factors. The proposed TCP technique used three complexity factors for specifying the effects of changes and showed a high inclusiveness in various experiments in the same research study. Validation of proposed cluster-based TCP was left for future research works.

Ayaz [93] proposed a Fourier-analysis-based TCP approach to prioritize "Modified Condition Decision Coverage" (MCDC) test cases. Formal definition of the proposed approach was based on the Boolean function. A correlation between the values and fault exposing potential of associating inputs can allow the reordering of test cases. Boolean functions supports both the test case generation and test case prioritization. Prior to this study was undertaken, the research work in [33] used Boolean decision variables to ensure that a test case was selected against a coverage statement. However, the proposed TCP approach [93] works faster in generating the minimal test suits as well as fault exposing potential. Moreover, correlation between Fourier analysis and noise related factors can be carried out in the future works to make the mathematical-model-based approach more robust and saleable to reorder test case of large size systems.

Awedikian and Yannou [94] proposed the model-based integrated statistical approach to generate automated functional test cases. Both quality, and cost objectives have been addressed by the proposed approach. Therefore, this approach is more nearer to solve the functional coverage of a software. A transition matrix has been used to demonstrate the changing functionality of a system. Information stored in the transition matrix is further used to generate offline as well as online test cases. For the evaluation of this approach, researchers carried out experiments at unit testing level on the two case studies. More fault detection in a lesser time period was achieved by the lateral-mentioned approach. In comparison with Jiang et al. [39] where researchers propose a statistical fault localization TCP approach to locate faults from debugging source code to determine suspiciousness of a system's elements, while former study [94] is based on the functional coverage. Most critical test cases execute and result in avoiding the higher cost of test cases execution. Therefore, a system without code availability can be retested by the functional coverage rather than using source code.

The trend of using the various factors and phases of software development has been increasingly introduced regarding the regression testing in various research studies. One of these studies that included requirement, design and code phases is by Kumar et al. [70] in which they gather information for early detection of faults during software development phases. The requirement phase has been examined as the most important phase of software development because faults in a requirement persist in the subsequent phases of software development. TCP technique proposed in the lateral study integrates the mapping of test cases from three phase, and uses weight priority for TCP and test case selection. The concept behind the proposal of the TCP technique is in line with another research study [63]. These both studies support earlier detection of faults to meet the end users' specifications. In a recently published work [65] a hybrid TCP approach has been proposed from requirements priority, intelligent techniques genetic and "Ant Colony Optimization" (ACO) algorithms. Requirement priority is used for test cases generation and ACO algorithm determined the best sequence for test cases execution. The hybrid TCP approach was found to be efficient than those TCP approaches that involved search based algorithms regarding earlier and maximum faults detection of applications. The hybrid approach aims to generate test cases based on their priorities. Priority to each test case is assigned using test factors. Next, an adequate sequence of test cases with the least execution time and a higher number of faults is determined by ACO algorithm. Prior to above-mentioned study, Tumeng et al. [57] proposed a technique that compared the text of test cases for emerging requirements. Both, the classical and non-classical comparison of texts were used to satisfy requirements.

In comparison with survey study [16] where authors identified four model-based TCP approaches, we include several model-based studies [63, 65]. These studies either use UML models instead of using source code of systems or apply ACO algorithm in combination with other criteria of TCP in regression testing. Due to less execution cost of mode-based TCP approaches in comparison with code-based TCP approaches, models are proposed earlier than the execution phase. Therefore, model-based TCP approaches outperform the code coverage-based TCP approaches.

User Stories in Test Case Prioritization

In a recently published work Kandil et al. [55] presented a TCP approach that has been designed from two techniques namely, 'Weighted Sprint Test Case Prioritization' (WSTP) and 'Cluster-Based Release Test Case Selection' (CRTS). WSTP is the novel technique in the context of regression testing. It applies to the users' stories in the projects and also considers the severity of stories from the previously executed test cases. Authors of the same study grouped the users' stories together in a cluster depending on the same modules and same functionalities. Most of the earlier discussed TCP techniques do not involve the agile projects for their experiments, but regression testing approach developed in [55, 59] include the agile parameters for test case prioritization as well as test case selection. Agile parameters, which are emphasized in the lateral research study, include the users' stories and their values, the score of issues, types of test cases, and practical weights given by the testers. User stories and acceptance test cases define user stories. An acceptance criteria defines how the program code should work. Testers document them and trace to user stories [59]. Business value of a user story represents what a team delivers in a number of iterations. The score of issue is determined from the previous execution of test cases and priority of an issue is also set by the agile team when they logging the issue. Each test case is also assigned a practical weight [55]. In future research studies, more agile parameters can be used for proposing TCP technique in regression testing. The latter-mentioned studies can be extended by merging the selection and prioritization techniques at the release level to keep them informed, which selected test case should be run first.

We observed that existing survey studies [20, 85] highlight the agile-based TCP while studies [86, 92] did not mention the agile-based TCP approaches. This might be due to scope of the studies. Also the lesser interest of researchers in TCP from agile parameters may be another reason of few published works.

Other TCP Techniques

In [70], authors have missed validation of their proposed approach to real-world systems. To bridge this gap in regression testing, and to verify the proposed TCP techniques on real-world or existing online systems, Hasan et al. [56] presented a conference paper with the experiment results on *Defects4j* dataset. They examined the test cases from two versions and involved the failure information regarding test cases. If failure history of the previous version has been missed for prioritization of test cases in the next version, it may impact the efficiency of the proposed technique. Although same proposed TCP techniques show better results as compared to similarity-based and random TCP techniques, the requirement prioritization or time constraint to make the proposed TCP technique more effective for regression testing has not been incorporated.

We have identified several FR-based TCP approaches in the existing literature. Majority of the chosen studies (11 of 35 studies) involve time, cost, and market pressure constraints, followed by model-based TCP approaches (13 of 35 studies) to address the issues given in RQ2. Moreover, we found that (six of 35 studies) include FR factors to propose TCP approaches in regression testing; (three of 35) studies which include discussion in agile factors; (one of 35) studies discuss text mining, and clustering along with TCP, and rest of (2 of 35) studies implicitly use functional requirements.

The chosen primary studies were aimed at prioritizing test cases for fast fault identification within context of functional requirements. The scope of TCP approaches is merely achieved because FR-based TCP approaches consider requirement factors, and weightage is given to each factor. However, human bias can result in error while allocating weights to factors. Moreover, a critical and safety related requirement may be given a less priority in comparison to other factors, and hence it can pose challenges for critical and safety systems. Our claim has an evidence from a recently published survey study [18] where researchers preferred the computerized and automated reasoning for decision-making over manually handled approaches which have more chances of human errors. In a recently published work, Banias [103] proposed to give a high priority to test cases linked with requirements which specify end users' health and safety. Test cases of end users' health has more priority than the safety unrelated test cases. Therefore, end users' health safety measure has been never considered in the selected FR-based TCP approaches. Primary study [9] highlights the safety and security risks related with the systems, and does not reveal information to safety risks of end users. Future works can gain popularity in the area of FR-based TCPs if critical safety factors are proposed to be added in the exiting factors. Therefore, some robust TCP techniques can be used for the both code and functional coverage of SUT [5]. Local beam search (LBS)-based TCP maps the new test cases to old test cases, and selects the best sequence of test cases. Research work recommends the use of LBS technique for both TCP and functional testing.

Limitations or Challenges of Existing FR-Based Techniques

To present the limitations of the existing FR-based technique, we have shown challenges as (C1-C7), mentioned above. In addition to these challenges, we observed the limitation of FR-based TCP approaches as given follows:

Evaluation of PORT 2.0 [49] is only performed on one system, and results in generalization to applications of other

domain is not explored. Before the proposal of the PORT 2 technique, PORT 1 [46] was evaluated on two domain applications. These domains are academia and industry. Applications from the rest of the domains listed in Table 6 were not considered. This could be a future research area regarding the application of benchmark FR-based TCP approaches in the software system from other domains. The second limitation we observed is that human experts assign value to requirements and risk factors, and they can bias while they assign values. This may impact the effectiveness of proposed FR-based TCP approaches. However, human involvement can be reduced by introducing the semi-automated approaches [9].

Only three test case prioritization factors such as CP, IC, and RC were focused in [44]. While FP and RV factors were not proposed for the TCP approach, that is why the results of the study cannot be generalized because more test cases prioritization factors exist in the literature. To prioritize test cases using test case execution time and the number of test cases is beneficial for the identification of maximum faults [45]. However, the challenge is the consumption of time on the execution of test cases which do not recover errors in a test suite.

RQ1.1 What are Key Requirements Prioritization Factors Discussed in the Selected Primary Studies?

Scope of our SLR is only to include requirements prioritization factors for testing. Therefore, RQ1.1 is aimed at identifying the requirement factors that have been used in proposing TCP approaches. Factors and attributes are interchangeably used in answering RQ1.1. Priority has been called as a metric in [26] where it relates with systems' bug history. We observed that test cases with maximum bugs have priority of their execution. TCP is aimed at reordering priority of a test case in order to ensure that maximum faults are covered from the highly prioritized test cases [97]. We include those factors which help in identification of higher faults. That is why the identification of fault-prone segments has potential to support developers for detecting the faults and fix them. Hence, developer's priority is a significant factor to prioritize test cases. Subsequently, for the fragment (part of a requirement) with the higher priority, software developers can perform debugging activities. Besides DP, other factors have their contribution in proposing TCP approaches. In the following Table 7, we present the requirement prioritization factors using requirement attributes/properties regarding primary studies.

Table 6 Domains of systems identified in primary studies

Domain of system under testing	References
Health	[9, 48, 67]
Education	[9, 58, 68]
Software development industry	[44, 50, 56–58]
Telecommunication	[45, 46]
Electrical & business management	[47, 54, 90]
IBM analytics application	[49]
Social work	[51]
Railway	[63]
Other	[52, 53, 55]

Table 7 shows the significant factors examined in the selected primary studies. These factors include CP, DP, RV, RC and FP which have been identified from the existing studies. In the following, we show frequency of identified factors in the selected primary studies. Factors and attributes are interchangeably used in the following.

As shown in above-given Fig. 5, FP factor followed by CP and DP factors have been widely employed while researchers proposed TCP approaches. In some of the primary studies five factors have been used in a combination of two or more than two attributes. Therefore, we show the number of individual factors examined in the primary studies. FP factor was examined in seven primary studies. Both CP, and DP attributes were studied in five primary studies of each, and RC was examined in two primary studies. Requirement volatility was least examined in primary studies.

From the findings, we observed that 20.00% (7 out of 35) primary studies used FP as a key factors of requirements for proposing TCP approaches. FP as an attribute of functional requirements is a measurable feature. For code coverage-based TCP approaches FP represented the past history of a system [24]. Since, researchers of the selected primary studies were more interested in finding the maximum faults through FR-based TCP approaches, and hence, we determine the frequency of FP which is highest of all attributes. Afterwards, CP and DP attributes contributed 14.28% of each (five out of 35) primary studies. RC and RV were least employed attributes 5.71% (two out of 35) and 2.85% (one out of 35) primary studies, respectively. Remaining 42.85% (15 out of 35) primary studies did not mention any of these five attributes. The above-mentioned factors were considered important factors while researchers mapped them to set the single priority value linked with test cases. These factors are the central items for the proposal of FR-based techniques, because requirement's priority is derived from these factors [49]. These factors helped in identifying 50% faults which were introduced during requirement phase, as revealed in [58]. Therefore, five attributes have been mostly focused for proposing the FR-based TCP approaches. More factors regarding requirements of a project can be focused while proposing the TCP approaches.

Table 7 Key factors of functional requirements

References	Functional requirement factors				
	Customer priority (CP)	Developer priority (DP)	Requirements change (RC)	Requirement volatility (RV)	Fault proneness (FP)
[9]	×	√	×	×	×
[44]	√	√	√	×	×
[45]	×	×	×	×	√
[46]	√	√	×	√	√
[47]	×	×	×	×	√
[48]	×	×	×	×	×
[49]	√	×	×	×	√
[50]	√	√	×	×	√
[51]	√	√	×	×	×
[52]	×	×	×	×	√
[53]	×	×	√	×	√
[54]	×	×	×	×	×
[55]	×	×	×	×	×
[56]	×	×	×	×	×
[63]	×	×	×	×	×
[67]	×	×	×	×	×

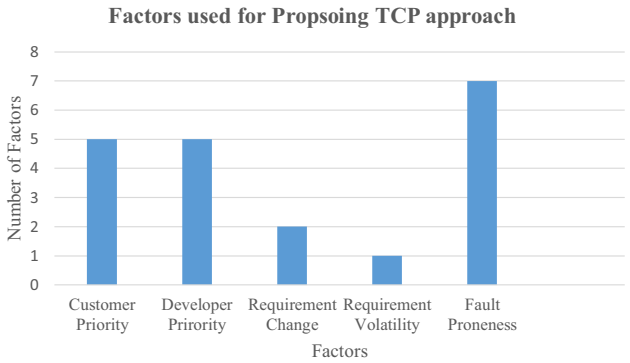

Fig. 5 Key factors used in proposing functional requirement-based TCP approach

RQ1.2. What are Important Regression Testing Aspects Considered for Proposing and Evaluating the TCP Approaches?

In regression testing, TCP has several aspects as stated by Rothermel et al. [37] where they discussed the prioritization regarding code coverage, and maximum faults detection from changed code, and reliability of a system. We extend the concept of TCP aspects to answer RQ1.2. Moreover, researchers in [38] discussed prioritization aspects, and their results regarding cost factor. We call FR, size of a system, faults, and test cases as regression testing aspects which were used for proposing and evaluation of TCP approaches. In other words, regression testing aspects are features of prioritization and its effectiveness.

In this regard, Yadav and Dutta [53] implemented a small object-oriented application and used their proposed TCP approach that was based on three factors. Among these three factors, requirement coverage was made an essential factor for TCP in regression testing. The value of requirement coverage along with other two factors' execution time and fault detection rate was determined through the 'Fuzzy Inference System' FIS in [53]. The FIS is a method to make decisions by using fuzzy rules [100]. To accomplish the former study results, 38 test cases were generated. APFD [15] metric was primarily used for determining the efficiency of proposed TCP approach [34, 36]. So, they used six programs with "Lines of Code" (LOC) that ranged between 193 and 1106, and determined the differences between actual APFD metric values and expected APFD metric values. The actual APFD metric value for each of the programs was found to be higher than the expected APFD metric values. All statistics as shown above highlight considered factors to propose and evaluate TCP approaches.

To explain the regression testing aspects for various industrial case studies, Srikanth and Banerjee [46] have focused on changed 75,213 LOC, 24 modified specifications, and 115 test cases, at the system level in the first case study. In the second case study, authors have analyzed 43,701 changed LOC, 54 modified/new requirements, and 87 test cases at the system level. From the analysis of the test cases taken from these two case studies, the PORT technique [46] that prioritizes test cases, which is based on four factors

(CP, IC, FP, and RV) showed better results than the random ordering of test cases. To see these results, authors have used 'Weighted Percentage of Failures Detected' WPFD [78] for four individual factors, and they found that testing efficiency of a system in consideration was equally dependent upon these four factors. To improve the regression testing, more factors are required to be included, while proposing new TCP approach that can help to improve the earlier fault detection.

Marchetto et al. [52] used 21 Java applications with the detailed description of test cases, LOC, number of faults, and number of requirements for each Java application. In other words, we can call them experimental objects or artifacts that have been used by authors. LaTazza is one of these 21 Java applications with 33 test cases, *2 K LOC* as the size of the application, 12 number of faults, and 10 requirements. We show the regression testing aspects identified in this study which have been detailed in the following Table 8.

Table 8 shows the primary studies along with regression testing aspects used in the primary studies. The idea behind using the regression testing aspects is to examine the effectiveness of primary studies regarding the TCP in regression testing. In the following Fig. 6, various regression testing aspects have been shown, which are tried out in the selected primary studies.

Figure 6 presents distribution of regression aspects by primary studies. We have a total of 35 primary studies

Table 8 Various regression testing aspects

References	Functional requirements	Size of system in lines of code	Faults	Test cases
[9]	√	√	√	×
[44]	√	√	√	×
[45]	√	√	√	×
[46]	√	√	√	√
[47]	√	√	√	√
[48]	√	√	√	√
[49]	√	√	√	√
[50]	√	×	√	√
[51]	√	√	√	√
[52]	√	√	√	√
[53]	√	√	√	√
[54]	×	×	√	√
[55]	√	×	√	√
[56]	×	√	√	√
[58]	√	√	√	√
[59]	√	×	√	√
[63]	√	×	√	√
[87]	√	×	√	√
[88]	√	√	√	√
[90]	√	×	√	√

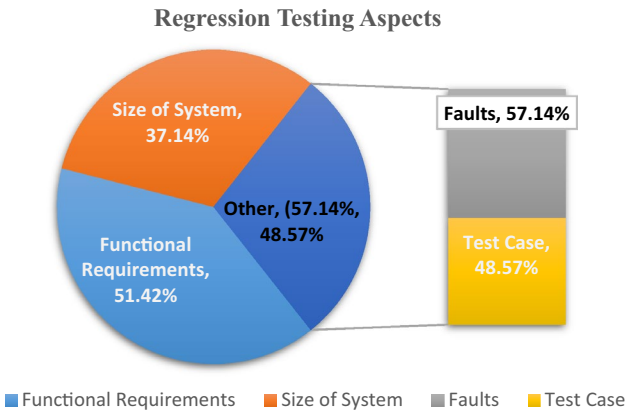

Fig. 6 Percentage distribution of regression testing aspects by primary studies

for this SLR. Four regression testing aspects as shown in Table 8 and Fig. 6 have been identified in 20 primary studies. We observed that four factors have been combined used in proposing TCP approaches. We noticed that faults and functional requirements were reported as 57.14% (20 of 35), and 51.42% (18 of 35) respectively in primary studies. Test case and size of system were reported as 48.57% (17 of 35), and 37.14% (13 of 35), respectively, in primary studies. The rest of primary studies do not include these aspects while proposing TCP approaches. Size of system was least mentioned in primary studies. The description of these regression testing aspects provide insight into four identified regression testing aspects. In comparison with study [19], where researchers showed LOC as size of a dataset while we identified it as key regression testing aspects because findings of [20] satisfy that a large system (200 KLoC) may contain more faults as compared to a small system.

The findings of our SLR indicate that majority of studies have clearly examined the FRs, faults, and test cases while proposing TCP approaches. However, the size of SUT has been least mentioned in the respective studies.

RQ2. What are Regression Testing Issues that have been Addressed by the Proposed FR-Based TCP Approaches?

The main objective of using this RQ2 is to know whether TCP approaches proposed from requirements features, address RT concerning issues/challenges given in the following. We identified seven challenges such as C1 (external validity), C2 (overhead), C3 (redundancy), C4 (scalability), C5 (test oracle), C6 (optimization), and C7 (robustness) in the selected primary studies. These challenges were of greater importance to be addressed by researchers. Prior to this SLR, survey study [17] highlights seven RT challenges such as cost, autonomy, test mining, dynamism,

concurrency, quota constraints, and privacy. Issues given in the latter study are more specific to TCP approaches regarding web services. Our SLR is rather focused on the issues, which have been addressed by proposed FR-based TCP approaches regarding traditional systems. To know that which issue is continuously prevailing in the regression testing, and which issues have been resolved so far can help researchers to design new testing approaches. We extracted all issues/challenges (C1–C7) from primary studies by reviewing the abstract, and conclusion sections and then reviewing main contents of primary studies.

In the following, we show the background of these problems in the context of regression testing approaches.

External Validity (C1)

External validity issue has remained a concern for researchers because proposed TCP approaches compete to retest the system as larger size systems require more effort than the smaller size systems [52]. Andrews et al. [90] were concerned with the external validity of their proposed approach. Typical results could not be achieved by the same type of applications. External validity that can be more precisely expressed as reliability centric TCP approach that detects software faults earlier during regression testing [71]. The external validity issue is observed for TCP techniques regardless of any criteria used to proposed approaches. It means that coverage-based approaches [104, 105] have the same issue of external validity as FR-based techniques. However, the difference is seen between performances of different criteria-based approaches because conditions vary, which are used to generalize results.

Overhead/Cost (C2)

Both cost and time efforts increase for regression testing due to changes in functional requirements. New commits emerge from last testing which require extra efforts and time. Hillah et al. [72] stated that the custom-built framework has more overhead for test cases. Similar definition of systems with customers' environment is given in [36] which is focused on the versions of operating systems. Overhead issue also emerges when a system needs changes, and new functions are added in it [90]. Most of TCP approaches directly address the cost issue regarding lesser number of test cases.

Redundancy (C3)

As software faults are linked to more than one requirement, test cases of the same fault might be executed which increase redundancy of test cases [51]. Redundant test cases exclusion was highly focused in [87], where researchers mentioned that redundant test cases need to be removed in the TCS regarding regression testing. However, redundancy has been linked with test case reduction in [39] where a test suit may contain two subsets, and test reduction approach eliminates the redundant set of test cases.

Scalability (C4)

This is a particular challenge when regression testing is performed for distributed systems. Although cloud services have addressed this challenge, future research work on new systems and how they operate on distributed channels is required. The same scalability challenge regarding regression testing was initially highlighted in [73] that still persists, although a number of TCP approaches have been proposed in regression testing. Scalability of proposed FR-based approaches is mostly seen in a number of studies [87]. Proposed TCP approaches do not scale up in handling of test cases of large size projects. There are industrial projects with millions of test suites, and simple heuristic TCP approaches are used instead [118].

Test Oracle (C5)

An issue/challenge emerged during the development of regression testing approach is to measure the correct output of test cases execution. To overcome the issue, test oracle is used to check the correctness of executed test cases. To expose the faults without test oracle in the automated test generation is an issue. This issue pertaining to test oracle has been mentioned in research studies [74–76, 79].

Optimization (C6)

To optimize a TCP technique for increasing the effectiveness of an approach is a primary objective of every research work undertaken in the test case prioritization. Regression test suit optimization is widely applied to reduce the regression testing cost. Optimization of test cases based on functional coverage criteria has been investigated by using Meta-Heuristic approach [70] that needs future investigation.

Robustness (C7)

TCP techniques face the robustness issue while complicated, and critical systems run the business by ensuring the working of related systems as they are changed due to new specifications [77]. Researchers always tried to address the TCP issue and optimized TCP approaches in achieving fast fault detection, and cost on reordering and execution of test cases [36–39]. In functional requirements criteria-based TCP, researchers have proposed TCP approaches and verify whether proposed TCP approaches are robust, scalable, and free of test oracle issue.

Based on the above-mentioned TCP issues/challenges we have shown distribution of issues regarding FR-based TCP approaches in following Fig. 7, and show which issues are greatly addressed in the selected studies.

Figure 7 shows the specific issues each primary study attempted to conquer. It indicates that majority of studies 57.14% (20/35) concentrate on the issue C1. Among these studies, mostly focused on the evaluation of proposed TCP approaches using various size systems. Seventeen studies (48.57%) concentrated on the issue C2. Only five studies (14.28%) have shown researchers' contribution in addressing the issue C3. Six studies (17.14%) were concerned on the issue C4. For issue C5, we noticed that four studies (11.42%) were more focused on the challenge C5. These studies represented test oracle (C5) as verification cost of functional requirements, and assign a weight to test oracle for manual and requirement bases regression testing. Sixteen studies (45.71%) focused on the issue C6 to identify earlier maximum faults by prioritizing test cases. Only four studies (11.42%) highlighted the issue C7. Majority of these studies have attempted to conquer more than one issue (see Fig. 7). We categorized the TCP issues into two classes named as "highly addressed issues", and "low addressed issues" as shown in Fig. 7. We can reveal that C3, C4, C5, and C7 are open research issues regarding regression TCP. Robust research regarding regression testing has been suggested, and therefore, future works can be carried to address these issues regarding test case prioritization.

Unique issues of FR-based TCP in comparison with other TCP approaches are described below:

Cost objectives issue discussed in [87], includes fault detection capability, test execution time, functional requirement coverage, and simulation time. Higher fault detection at a lower time is known as a TCP method with a high effectiveness regarding cost. The fault detection time of search-based TCP is lower than the random search-based techniques. Code availability of test cases on the shared distributed environment is costly because each tester cannot access straightforward the collected code information. One of the unique challenges of the FR-based TCP approach is regarding non-code information. For instance, a requirement written in natural language text needs to be mined with the help of text mining tools. It requires additional time to write test cases from extracted data of requirements [26]. Code-based TCP techniques do not face such a challenge. Regarding the users' satisfaction, the FR-based TCP approach proposed from requirement factors is beneficial because severe fault detection rate improves [58].

Moreover, risk-based approached which are proposed from requirement factors are more productive than random TCP approaches. PORT 2.0 [49] is one of the techniques, which requires less effort at the planning phase. The latter-mentioned approach also does not need the specific skills to apply it. However, this approach shows its limitation when each fault is taken equally severe.

RQ3. What are Test Case Size and Defects Types that Relate with Each Other in the Selected Primary Studies?

To answer RQ3, we have focused on the subject program, size of test cases and relevant fault/defect types. The main aim behind using the size of test cases is to see its association with the various defects types. We have identified various defects types which have been mentioned in the selected primary studies.

Table 9 shows the types of faults/defects detected by test cases in subject programs, which are derived by the proposed TCP approaches in regression testing. Primary studies [9, 44, 45, 58, 90] did not mention number of test cases,

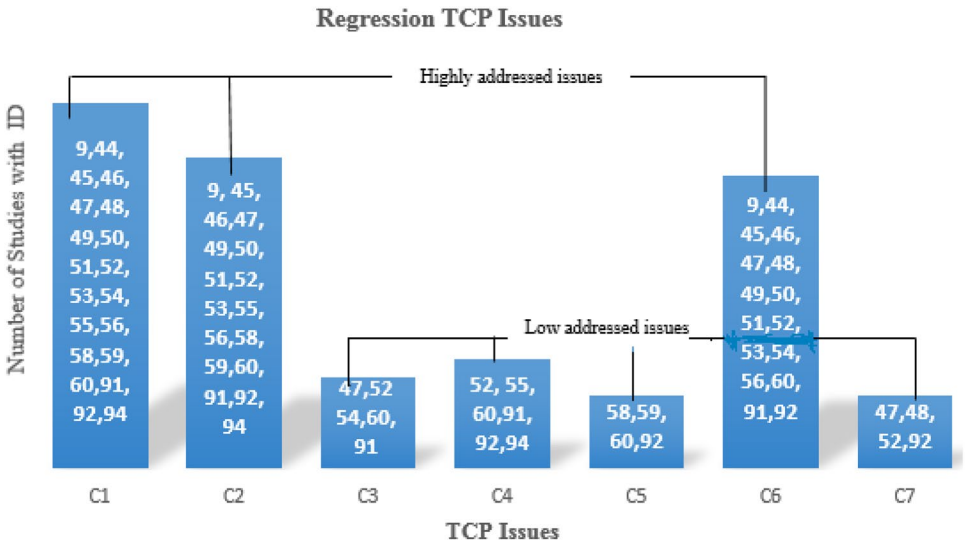

Fig. 7 Issues identified in primary studies

Table 9 Test case size, and defects types in subject programs

References	Subject program	Test case size	Defects types
[9]	–	–	Functional defects of systems Requirements-related risks
[44]	–	–	Requirement change defects
[45]	–	–	Systems' quality defects introduced in requirement phase Functional defects
[46]	Tekelec system	115	Requirement and code-related defects
	I-Cubed	75	
[47]	AveCalc:	47	Technical and critical business defects Functional and non-functional defects
	LaTazza:	33	
[48]	Capstone:	42	Requirements and code-related defects
	iTrust:	142	
[49]	–	100	Requirement-related risk
[50]	–	42	Requirement change defects
[51]	–	500	Functional defects of systems
[52]	Multiple Java systems	33–2307	Technical and critical business faults Functional and non-functional defect
[53]	–	38	Seeded faults in code of a system
[54]	–	48	Change segment faults
[55]	Release 1	359	Requirements-related defects
	Release 2	888	
	Release 3	549	
[56]	Jfreechart	2205	Functional defects of systems
	Joda-Time	2245	
	Closure compiler:	7827	
[58]	J2EEE, PHP and VB projects	–	Induced defects produced from unclear requirements communication
[59]	Subject Project	30,834	Mutants' produced defects
[63]	BR490 Sweden project:	1748	Testing failure-related problems
[90]	Mortgage System	–	Reusable artefacts' failure

but revealed information about defects types. Remaining primary studies mentioned number of test cases as well as revealed faults in requirements. From proposed TCP approaches in [46, 48, 53] test cases were used to detect the faults in source code of the systems. On the other hand, TCP approaches proposed in primary studies [9, 45, 52, 56] used test cases to identify the functional defects of systems. TCP approaches proposed in [9, 49] revealed requirements related risk defects of the systems under regression testing. As shown in Table 9, variant size of test cases have been used to identify faults in systems to evaluate proposed techniques. Arafeen and Do [48] mentioned the test cases of two systems in the context of severe faults in system requirements and source code. System level regression testing used test cases in a thousand test cases as given in primary studies [52, 56]. Reusable artefacts' failure concerning defects were mentioned in [90].

According to Table 9, several types of defects were identified in this SLR. However, we did not find any relationship between test case size and types of revealed defects. For instance, the most common type of defects as functional defects in [9, 51, 56] were identified by using variant number of test cases. This might be due to size of requirements' segments which were affected due to change or addition of new operations. Subsequently, the size of test case is increased if large proportion of requirements' segments is modified. Only two studies [53, 59] mentioned the size of test case for seeded and mutants faults. The size of test cases was identified 38 in [53], and 30,834 in [59]. However, same kind of defects were revealed from latter-mentioned studies. This provided evidence that no prominent connection existed between size of test cases and types of defects. We considered the number and defect type aspects in our SLR. The first aspect shows that a test case execution resulted in a number of maximum faults by following requirement-based TCP criteria. The defect type aspect shows the categories of identified faults which can be fixed to restore the appropriate functioning of software systems.

In comparison with study [18] where defects have been mapped to requirements, we further explore various types

of defects along with the number of test cases. Latter-mentioned survey study dealt with defects regarding cost, and found that large scale systems consumed more cost. Other studies [19–21, 76] did not reveal statistics about defects types.

RQ4. What are Linkage Metrics Used in the Selected Primary Studies?

Core of software engineering is to discuss the implementation of functional requirements accordingly by verifying source code of the SUTs. To keep track record of links between functional requirements and source code, linkage metrics such as traceability, requirement correlations, requirement complexity, code complexity and similar behavior were used in the selected primary studies.

The Following Fig. 8 shows matrices that have been either used for linking the system artefacts such as source code and requirements or any other key criteria used that helped in establishing correlation between functional requirements and faults etc. In the following Fig. 8, we show the number of linkage metrics reported in the selected 35 primary studies.

Figure 8 shows matrices used in primary studies in order to trace the links between faults and requirements, design and code. Among these matrices, traceability is used in 10 (29%) primary studies), followed by code complexity matrices in 3 (9%) primary studies. Requirement correlation and similarity behavior have been studied in each of 2 (6%) primary studies. Requirements complexity matrices was examined in one (3%) of total 35 primary studies. Rest of the 17 (47%) primary studies did not mention the usage of linkage metrics in the proposed FR-based TCP approaches.

Most studied traceability matrices have been used regarding the functional requirements, code source and test cases. Traceability matrices is employed in both directions i.e. forward traceability and backward traceability. The forward traceability maps requirements to test cases while backward traceability maps back test cases to requirements. Fundamental parameters employed in traceability matric include as requirement Id, faults, test cases, traces to source code, and trace to design specification [99]. We can conclude that test case property of traceability matrices ensures that execution of certain test cases verify the correct working of a module. Another means of using traceability matrices is to quickly locate the changed requirement and avoid retesting of test cases of unaffected requirements. Code complexity is another matric that predicts faults in SUT [30]. It provides empirical insights into code properties which cause complexity in code. Therefore, faults can be identified by tracing back to complex source code part of a system experienced by system developers. However, this practice is poorly employed in software industry. Requirement complexity is transferred to source code in programming languages. This matric is used to measure complexity of a program without examining the source code of a SUT. We can conclude that future works can be carried out in reference to the identified linkage metrics for improving TCP in regression testing.

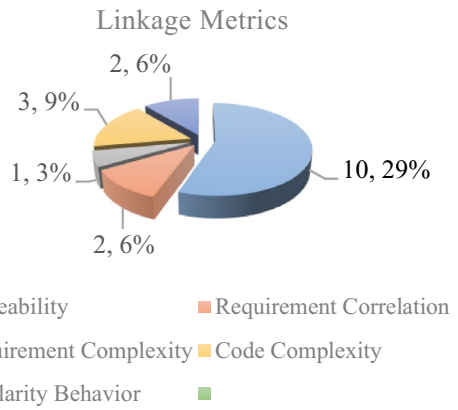

Fig. 8 Relative distribution of linkage metrics in primary studies

In comparison with the studies [16, 18] where researchers have included discussion on evaluation metrics such as APFD, and its extended metrics, we have discussed linkage metrics. The former type of matrices, and its extended version determine effectiveness of proposed TCP approaches while latter type of matrices represent the links of systems' requirements, design, and source code with test cases. Our SLR is the first study which identified the contribution of different linkage metrics in TCP regarding regression testing.

RQ5: What Systems Under Testing (SUT) are Used to Validate the Requirement-Based TCP Approaches?

To answer this question, we have examined, systems under testing, and dataset information used in primary studies. Researchers have validated their proposed TCP approaches regarding regression testing in various primary studies. It has been found that [9, 44–56, 58] primary studies mentioned the names of SUTs. Researchers noticed that 'iTrust' system was widely examined in the primary studies [9, 48]. Haraty et al. [54] also mentioned the number of methods that were used to validate the proposed TCP approach in the lateral discussed research study, and these methods were applied to the anti-missile system, which is known as "Launch Interceptor Program" (LIP). To further assess the SUTs, we have proposed the classification of SUTs used in the primary studies. Three levels of system's size as shown in the following have been proposed. Before this SLR, study [76] mentioned the size of datasets for test case minimization. We present classification of SUTs which shows us that to date which domain's system with lines of codes have been

retested regarding functional regression TCP. This reminds the researchers to consider the scalability of their proposed TCP approaches.

Therefore, in the following Table 10, we show, systems, size of systems, and proposed three levels such as small, medium and large size systems.

Table 10 is the illustration of our proposed three levels of system's size. This is indicating the proposed FR-based TCP approaches have been evaluated on three levels.

Figure 9 shows us the number of primary studies which have been undertaken to evaluate functional requirement-based TCP approaches using systems with different lines of code. We have major 35 primary studies with this information on LOC of systems that have been studied. However, in a primary study [52] researchers used two level systems. They wanted to examine the scalability performance of their proposed approach. Therefore, they used systems that belong to our proposed two levels. This is only a primitive effort by us regarding LOCs, and in future we would continue our research on a relationship between lines of codes, code complexity, and number of faults. Seven primary studies out of 35 selected primary studies reported the systems with LOCs $\leq$ 5000. Only three primary studies reported the systems with LOCs between 5001 and 10,000. Eight primary studies reported the system with LOCs > 10,000. As TCP research is increasingly carried out, researchers have optimized the proposed FR-based approaches to apply them on large size systems. Three out of 35 selected primary studies did not report number of lines of systems for mentioned SUTs. The size of programs has increased many-folds in newest versions than the older ones i.e. Mozilla Firefox new versions. Therefore, researchers used FR-based TCP approaches to maintain the functionality of extended and larger systems.

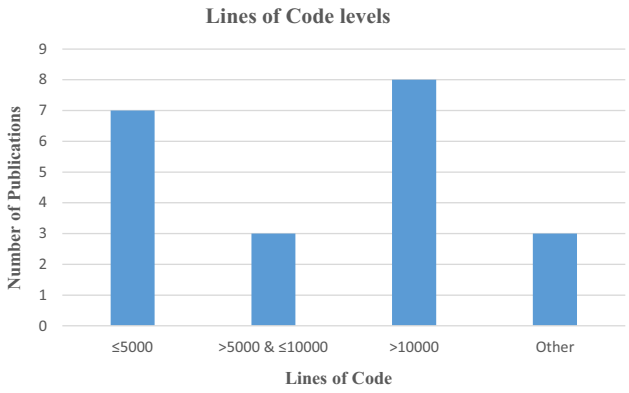

Fig. 9 Levels of systems under testing using lines of code

We observed that systems of industry leading organizations could not be accessed because an industry leading organization does not prefer to make their systems' SRS and source code available to public for confidentiality issues. In

Table 10 Systems classification into three levels of lines of code

References	System name	System size in lines of code	Small ($\leq$5000)	Medium ($\geq$5001& $\leq$10,000)	Large ($\geq$10,001)
[9]	iTrust	24.42–26.70 K	×	×	√
	Capstone	6.82 K	×	√	×
[44]	–	5000	√	×	×
[45]	–	7129	×	√	×
[46]	–	100,000	×	×	√
[47]	AveCalc	1827	√	×	×
	LaTazza	1121	√	×	×
[48]	iTrust	More than 10 K	×	×	√
[49]	–	Many millions	×	×	√
[51]	–	440,000	×	×	√
[52]	21 Java programs	2 K-138 K and	√	×	√
[53]	–	193–1106	√	×	×
[54]	Lunch interceptor System	–	–	–	–
[56]	Defects4j	Many thousands	×	×	√
[57]	Print token	726	√	×	×
[58]	J2EEE applications	5000	×	√	×
[63]	BR490	–	–	–	–
[67]	Hospital Management System	–	–	–	–
[68]	Student projects	2500	√	×	×
[90]	Mortgage system	257,592	×	×	√

this case, researchers prefer to use systems which have been previously regression tested by other researchers. Hence, they compare the results of their experiments to the results of existing studies with the similar systems. We have also found that iTrust system and various other open source Java programs have been used by a number of researchers in TCP regarding regression testing.

RQ6. What are Domains of System Under Testing (SUT) Used in Primary Studies?

We were interested to know whether systems under testing were used from different research domains. We identified eight domains of system used in primary studies. Moreover, the SUTs mentioned without any domain were kept in 'other' category of domains.

Figure 10 illustrate the domains of SUTs, which have been covered by the selected primary studies. Moreover, domain testing based on the input functionality and output functionality has been illustrated in Fig. 10. The results showed that SUTs from software domain remained most investigated in the selected research studies. This is because, many of the open source system are available on different repositories, and researchers have easier access to evaluate their approaches. Moreover, various research repositories update systems with the source code as well as functional requirements information. "Software-artifact Infrastructure Repository" (SIR) [25, 32, 33] is one of these repositories which contains updated versions of Java and C programs from different domains. Researchers also used SUTs from education, health, telecommunication, and electrical domains which reside in different repositories. Least systems from domains of IBM analytics, railway, and social work have been reported in primary studies. One of these primary studies [52], used 21 systems from various domains. Therefore, we kept the SUTs in another category of domains. Among selected 35 primary studies, only in four primary studies [59, 62, 64, 66], did researchers not explicitly mention the domains of systems they used to verify their proposed functional requirement-based approaches because they did not want disclose defects in those systems.

Figure 10 is also the illustration of SUTs of various research domains and its association with the domain testing. Domain testing is aimed at developing common artefacts which can be further used for software testing [96, 101]. Domain testing concept is based on input and output domains. Large input domains create complexity while generating and prioritizing test cases [102]. Domain testing is efficient as variables of both input and output domains can be mapped to identify risk to a system. Test cases can be written and ordered based on the risks induced from domain testing. This grouping of SUTs in nine categories can help researchers, and software practitioners to develop their domain knowledge of different industries. A software testers with a high domain knowledge of the industry application can perform efficiently regression testing, and avoids the irrelevant test suits while testing various integrated applications.

RQ7. Have Primary Studies been Tried in the Real-World Settings or Not?

This paper has mapped the evidence regarding the systems into two categories, which determines whether proposed requirement-based TCP approaches are tried out in the real-world settings or not. Real-world setting means that FR-based TCP approaches have been evaluated on the academic and industrial projects. Before this SLR, Catal and Mishra [16] and Kazmi et al. [17] presented statistics on industrial datasets used for evaluation of TCP approaches. Statistics presented in the latter-mentioned studies are outdated because several new FR-based TCP approaches have been proposed. Researchers of this paper have summarized

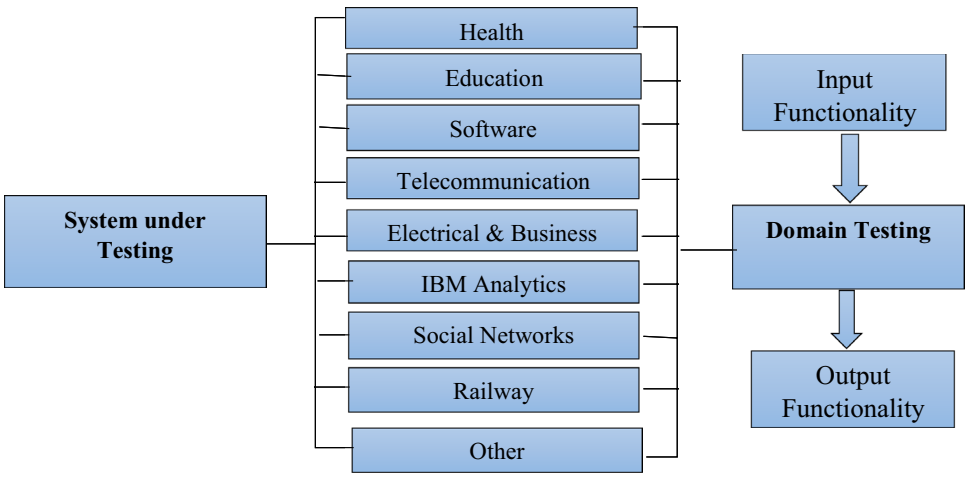

Fig. 10 Domains classification and domain testing

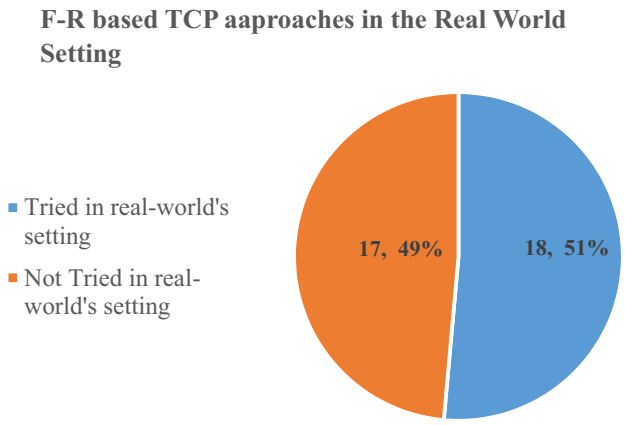

Fig. 11 Number of primary studies tried in the real-world settings

the information according to these two categories, as shown in the following Fig. 11.

Figure 11 shows the number of the primary studies, which proposed TCP approaches in regression testing. Figure 11 also shows that 18 primary studies include TCP approaches which have been tried out on the academic and industrial systems for their validation, and rest of 17 primary studies were not tried in the real-world settings.

For a number of publications regarding regression testing, and particularly TCP, it remained important for researchers to evaluate the proposed TCP approaches in real-world settings. Wohlin and Aurum [80] have emphasized methods, techniques, and frameworks in software engineering. The main aim behind using methods, techniques, and frameworks was to demonstrate the significant advantages and provide evidence that methods, techniques, and frameworks have their impacts in the software industry. Therefore, validation of methods, techniques, and frameworks in regression testing is required. It helps researchers in evaluating and testing of the proposed methods, techniques and framework in real-world settings. Krishnamoorthi and Mary [58] proposed factors oriented TCP approach, and evaluated their proposed technique on five student projects and two industrial projects. The proposed TCP approach revealed the better identification of severe faults in comparison with the random ordering [29] of test cases. Test cases with the injected faults revealed less number of faults than the prioritized test cases.

Hence in this SLR on requirement-based TCP, researchers have examined primary studies in perspective of validation of proposed TCP approaches on the academic and industrial projects. This is critical for researchers to extend the validated TCP approaches in regression testing in their future research works. Some of the TCP approaches are validated in a small software setting to illustrate the usability of these techniques, while other researchers have used small cases or sometimes provide examples with a rough idea about the use and applications of proposed TCP approaches. In order to make the assessment more practical, researchers have used the revised Kitchenham's classification [81] in which they presented the six level hierarchy from a weak level to the strong level as shown in the following Table 11.

Table 11 shows that "no evidence" and demonstration and toy examples have been added to the weak end of this six hierarchy level, while in the strong level industrial practices and industrial studies are essentially given. An industrial practice means that a TCP approach has been already used and adopted by some industrial organizations. Therefore, primary studies with the industrial practices were ranked as strongest in the six-level hierarchy.

Six correspondence hierarchy levels have been mapped to two categories, which has further explored that either proposed TCP approaches are tried out in the industrial or academic setting. So, primary studies in levels 4,5 and 6 are grouped together, which represent that proposed TCP approaches in these studies are tried out on the academic and industrial projects, while primary studies in level 1, 2 and 3 are grouped together and represent that proposed TCP approaches are not tried out any type of projects. TCP approaches proposed in majority of primary studies 51.42% (18 of 35 primary studies) have been tried out in the real-world academic and industrial projects. TCP approaches in the rest of primary studies 48.57% (17 of 35) were not tried out on the academic and industrial projects.

We observed that majority of TCP approaches were evaluated on the academic and industrial projects. Therefore, our findings elaborate the strengths of the proposed TCP approaches regarding the validation in the academia and industrial settings. However, TCP approaches still have potentials to optimize them for their validation on larger academic and industrial projects.

Threats to Validity

We were conscious of concerns that may impact outcomes as well as validity of this SLR. We tried to alleviate the threats to the validity of this SLR's results and conclusions.

Table 11 Revised six level hierarchy

Level	Description
1	No evidence was given
2	Demonstration or working out toy example was used as an evidence
3	Experts' opinions, and observations were used as an evidence
4	Academic studies were used as an evidence
5	Industrial studies provide the evidence
6	Industrial practices provide the evidence

Nevertheless, there are possible concerns and threats, which deserve discussion as follows:

There is possibility of bias in selecting the primary studies whether all of relevant papers were identified and included. This might have resulted due to following reasons.

First, search engines did not pick up some relevant papers. Second, some important papers were mistakenly omitted, and included in this SLR. To overcome these concerns and threats, we used multiple strategies. We used recognized data repositories such as IEEE Xplore, Scopus, ACM, Springer, World Scientific, Wiley Online, and Elsevier. The other data repositories were not used because of their scope, as well as time constraints. It might be possible that FR-based TCP papers from repositories other than the above-given repositories have been missed.

Finding the adequate search string is another possible concern. Justification about using search strings and data search repositories have been provided in the search method section of this SLR. We refined search strings many times to access a number of primary studies from relevant search repositories. We have chosen the iterative strategy to ensure the proper list of keywords was used.

There is another concern that this SLR may have insufficient number of primary studies, and hence used number of primary studies which lacked information about systems under regression testing. Although, we accessed most significant databases to retrieve information on requirement-based TCP approaches, however, there is probability that we missed some studies which primarily deal with the other subtopics of regression testing such as TCS, and TCR, and also contain discussion on TCP in regression testing.

The quality assessment may also result in concerns, thereby leading to inaccurate extraction of information from selected primary studies. Although a number of criteria-based artefacts were used in proposing TCP approaches regarding functional requirement, it might be possible that some aspects of proposing TCP approaches have been missed.

Conclusions and Implications

In this section we conclude our SLR, and also state its implications for researchers and software actioners.

In this SLR, we have provided a review of the requirement-based test case prioritization in regression testing. As SLR shows, the past more than ten years witnessed the dramatic increase in TCP activities, with a number of new research studies published. In this study, a systematic literature review was conducted regarding TCP in regression testing. 35 primary studies were selected from seven important databases. Regarding research questions RQ1 to RQ7 an investigation has been provided, as detailed in the following contents:

- **RQ1:** We have investigated the TCP approaches based on requirements from 35 selected primary studies. Among the identified requirement-based TCP approaches, PORT 1.0, PORT 2.0, risk-based TCP, change-based TCP, clustering-based TCP, WSTP TCP, and similarity-based TCP have been widely covered in primary studies. For year 2016, a number of requirement-based research studies have been published to address the TCP problem.
- **RQ1.1.** Our SLR identified five key requirements prioritization factors such as CP, DP, RV, RC and FP in the selected primary studies. Fault Proneness FP was greatly used by researchers as a measureable feature, and Requirement Volatility RV was least examined in the primary studies. 42.85% of total selected primary studies did not mention any of these attributes of functional requirements
- **RQ1.2.** This SLR identified a number of regression testing aspects for proposing and evaluating the proposed TCP approaches. The researchers presented the distribution of identified four factors in the selected primary studies. Fault size, and the number of test cases along with functional requirements are greatly covered in primary studies while size of SUT is least mentioned in the primary studies
- **RQ2.** We have identified that proposed TCP approaches addressed the external validity, overhead and optimization issues regarding TCP approaches. Rest of the issues such C3, C4, C5 and C7 as redundancy, scalability, test oracle, and robustness have been addressed in a lesser number of primary studies.
- **RQ3.** Researchers have shown that both defects and number of test cases have been widely examined in the selected primary studies. Test cases are used for identifying faults in SUTs in the context of requirements and source code. This SLR identified the various defects types. Among these defects requirements changes related risks and defects, and technical and critical business defects have been mostly examined in the selected primary studies.
- **RQ4.** This SLR shows that traceability matrices is widely employed in primary studies. Traceability has been found to establish a link between various defects types, and requirements and source code of SUTs regarding regression testing. Code complexity, requirement correlation and similarity behavior were also identified other matrices in the selected primary studies. Requirement complexity was least examined in the selected literature.
- **RQ5.** In the majority of primary studies, researchers have examined the systems for validating their proposed TCP techniques. Identified systems used to vali-

date the proposed TCP approaches in the mentioned primary studies include the I-Cubed and Tekelec, iTrust, CPMISS web application, Token Sending Program, LaTazza and JTidy, LIP and Defects4j. iTrust systems have been widely examined in a number of primary studies. 7 large size systems with more than 10,000 LOCs have been also investigated in this SLR.

- **RQ6**. Researchers identified nine system domains, which have been used for evaluation of FR-based TCP in primary studies. Open source systems under the domain of software industry were mainly used. Moreover, the researchers found that health, education, telecommunication and electrical & business, railway, social networks, and IBM analytics applications were used in primary studies.
- **RQ7.** This SLR's findings indicated that majority of primary studies were tried out in the real-world settings to validate the proposed TCP approaches. This SLR also shows that researchers mainly used academic and industrial systems to validate their proposed studies.

In the first of its kind SLR in FR-based test case prioritization, we have made an attempt to collect, a wide range information on the functional requirements pertaining to research questions. This SLR has a few research implications for researchers and software practitioners.

- The first implication is that challenges identified in this SLR can be further carried out by researchers in future research works. To overcome the challenges, hybrid criteria-based TCP approaches can be proposed which have lesser impacts from the identified challenges. New TCP approaches can be proposed with the comprehensive information on the identified challenges.
- The second implication is that software developers can keep record of changed segments of functional requirements and then may perform test case prioritization to identify the maximum faults.
- Another implication is that a new area of research can be initiated with the introduction of non-functional requirements to propose TCP approaches in regression testing.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Harrold M. Reduce, reuse, recycle, recover: techniques for improved regression testing. In: SoftwareMaintenance, 2009. ICSM 2009. IEEE International Conference on 20–26 Sept. 2009, Edmonton, AB, Canada 2009.
2. Memon A. Advances in computers, vol. 107. Salt Lake City: Academic Press; 2017.
3. Hao D, Zhang L, Zang L, Wang Y, Wu X, Xie T. To be optimal or not in test-case prioritization. IEEE Trans Software Eng. 2016;42(5):490–505.
4. Pandey A, Goyal N. Early software reliability prediction: a fuzzy logic approach. New York: Springer; 2013.
5. Jiang B, Chan WK. Input-based adaptive randomized test case prioritization: a local beam search approach. J Syst Softw. 2015;105:91–106.
6. Parejo JA, Sánchez AB, Segura S, Ruiz-Cortés A, Lopez-Herrejon RE, Egyed A. Multi- objective test case prioritization in highly configurable systems: a case study. J Syst Softw. 2016;122:287–310.
7. Hao D, Zhang L, Zhang L, Rothermel G, Mei H. A unified test case prioritization approach. ACM Trans Softw Eng Methodol (TOSEM). 2014;24(2):10.
8. Sampath S, Bryce RE, Memon AM. A uniform representation of hybrid criteria for regression testing. IEEE Trans Softw Eng. 2013;39(10):1326–44.
9. Hettiarachchi C, Do H, Choi B. Risk-based test case prioritization using a fuzzy expert system. Inf Softw Technol. 2016;69:1–15.
10. Lin CT, Chen CD, Tsai CS, Kapfhammer GM. History-based test case prioritization with software version awareness. In: Engineering of complex computer systems (ICECCS), 2013 18th International Conference on IEEE 2013;171–2.
11. Gao D, Guo X, Zhao L. Test case prioritization for regression testing based on ant colony optimization. In: Software Engineering and Service Science (ICSESS), 2015. 6th IEEE International Conference on IEEE 2015; 275–9.
12. Khalilian A, Azgomi MA, Fazlalizadeh Y. An improved method for test case prioritization by incorporating historical test case data. Sci Comput Program. 2012;78(1):93–116.
13. Lewis W. Software testing and continuous quality improvement. 3rd ed. Boca Raton: CRC Press; 2016.
14. Agarwal B, Tayal S. Software engineering. New Delhi: Laxmi Publications; 2009.
15. Elbaum S, Malishevsky AG, Rothermel G. Test case prioritization: a family of empirical studies. IEEE Trans Software Eng. 2002;28(2):159–82.
16. Catal C, Mishra D. Test case prioritization: a systematic mapping study. Softw Qual J. 2013;21:445–78.
17. Qiu D, Li B, Ji S, Leung H. Regression testing of web service: a systematic mapping study. ACM Comput Surv (CSUR). 2015;47(2):21.
18. Khatibsyarbini M, Isa M, Jawawi DA, Tumeng R. Test case prioritization approaches in regression testing: a systematic literature review. Inf Softw Technol. 2017;93:74–93.
19. Kazmi R, Jawawi DN, Mohamad R, Ghani I. Effective regression test case selection: a systematic literature review. ACM Comput Surv (CSUR). 2017;50(2):29.
20. Rosero R, Gomez O, Rodriguez G. 15 years of software regression testing techniques—a survey. Int J Software Eng Knowl Eng. 2016;26(5):675–89.
21. Arora P, Bhatia R. A systematic review of agent-based test case generation for regression testing. Arab J Sci Eng. 2018;43(2):447–70.

22. Gallagher L, Offutt J, Cincotta A. Integration testing of object-oriented components using finite state machines. Softw Test Verif Reliab. 2006;16(4):215–66.
23. Mohanty S, Acharya AA, Mohapatra DP. A model based prioritization technique for component based software retesting using UML state chart diagram. In: Electronics Computer Technology (ICECT), 2011 3rd International Conference on. IEEE 2011, Vol. 2, pp. 364–8.
24. Kundu D, Sarma M, Samanta D, Mall R. System testing for object-oriented systems with test case prioritization. Softw Test Verif Reliab. 2009;19(4):297–333.
25. Zhang L, Hao D, Zhang L, Rothermel G, Mei H. Bridging the gap between the total and additional test-case prioritization strategies. In *Proceedings of the 2013 International Conference on Software Engineering* IEEE Press 2013; 192–201.
26. Salehie M, Li S, Tahvildari L, Dara R, Li S, Moore M. Prioritizing requirements-based regression test cases: A goal-driven practice. In *Software Maintenance and Reengineering. (CSMR), 2011 15th European Conference on*. IEEE 2011;329–32.
27. Arcaini P, Gargantini A, Vavassori P. Generating tests for detecting faults in feature models. In *Software Testing, Verification and Validation (ICST), 2015 IEEE 8th International Conference on* IEEE 2015;1–10.
28. Bryce RC, Colbourn CJ. Prioritized interaction testing for pairwise coverage with seeding and constraints. Inf Softw Technol. 2006;48(10):960–70.
29. Korel B, Tahat LH, Harman M. Test prioritization using system models. In *Software. Maintenance, 2005. ICSM'05. Proceedings of the 21st IEEE International Conference on*. IEEE 2005;559–68.
30. Voinea L, Sassenburg H (2009) Improving testing efficiency: agile test case prioritization.
31. Jaffarur- Rehman JU, Jabeen F, Bertolino A, Polini A. Testing software components for integration: a survey of issues and techniques. Softw Test Verif Reliab. 2007;17(2):95–133.
32. Chen TY, Kuo FC, Liu H, Wong WE. Does adaptive random testing deliver a higher confidence than random testing?. In *Quality Software, 2008. QSIC'08. The Eighth International Conference on* (pp. 145–54). IEEE 2008.
33. Zhang L, Hou SS, Guo C, Xie T, Mei H. Time-aware test-case prioritization using integer linear programming. In *Proceedings of the eighteenth international symposium on Software testing and analysis* (pp. 213–24). ACM 2009.
34. Kumar H, Chauhan N. A module coupling slice based test case prioritization technique. Int J Modern Educ Comput Sci. 2015;7(7):8–16.
35. Maia CLB, Freitas FG, Souza JT. Applying search-based techniques for requirements based test case prioritization. In *I WOES-CBSoft* 2010; 24–31.
36. Iskrenovic-Momcilovic O, Micic A. Mechatronic software testing. In *Telecommunications in Modern Satellite, Cable and Broadcasting Services, 2007. TELSIKS 2007. 8th International Conference on* (pp. 486–9). IEEE 2007.
37. Rothermel G, Untch RH, Harrold MJ. Prioritizing test cases for regression testing. IEEE Trans Softw Eng. 2001;27(10):929–48.
38. Do H, Rothermel G. On the use of mutation faults in empirical assessments of test case prioritization techniques. IEEE Trans Softw Eng. 2006;32(9):733–52.
39. Jiang B, Zhang Z, Chan WK, Tse TH, Chen TY. How well does test case prioritization integrate with statistical fault localization? Inf Softw Technol. 2012;54:739–58.
40. Kitchenham B, Brereton P, David B, Mark T, Bailey J, Linkman S. Systematic literature reviews in software engineering—A systematic literature review. Inf Softw Technol. 2009;51(1):7–15.
41. Kitchenham BA, Brereton P, Turner M, Niazi MK, Linkman S, Pretorius R, Budgen D. Refining the systematic literature review process two participant-observer case studies. Empir Software Eng. 2010;15:618–53.
42. Kitchenham B. Guidelines for performing systematic literature reviews in software engineering version 2.3, Technical Report 2007.
43. Engstrom E, Runeson P, Skoglund M. A systematic review on regression test selection techniques. Inf Softw Technol. 2010;52(1):14–30.
44. Kavitha R, Kavitha VR, Kumar NS. Requirement based test case prioritization. Paper presented at the Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference, Ramanathapuram, India 2010.
45. Mary SA, Krishnamoorthi R. Time-aware and weighted fault severity based metrics for test case prioritization. Int J Software Eng Knowl Eng. 2011;21(1):129–42.
46. Srikanth H, Banerjee S. Improving test efficiency through system test prioritization. J Syst Softw. 2012;85:1176–87.
47. Islam M, Marchetto A, Susi A, Scanniello G. A multi-objective technique to prioritize test cases based on latent semantic indexing. In *2012 16th European Conference on Software Maintenance and Reengineering.* 2012;21–30.
48. Arafeen MJ, Do H. *Test Case Prioritization Using Requirements-Based Clustering*. Paper presented at the IEEE Sixth International Conference on Software Testing, Verification and Validation, Luxembourg, Luxembourg 2013.
49. Srikanth H, Hettiarachchi C, Do H. Requirements based test prioritization using risk factors: an industrial study. Inf Softw Technol. 2016;69:71–83.
50. Ma T, Zeng H, Wang X. Test case prioritization based on requirement correlations. Paper presented at the Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016 17th IEEE/ACIS International Conference, Shanghai, China 2016.
51. Wang X, Zeng H. History-based dynamic test case prioritization for requirement properties in regression testing. Paper presented at the Continuous Software Evolution and Delivery (CSED), IEEE/ACM International Workshop, Austin, TX, USA 2016.
52. Marchetto A, Islam MM, Asghar W, Susi A, Scanniello G. A multi-objective technique to prioritize test cases. IEEE Trans Softw Eng. 2016;42(10):918–40.
53. Yadav DK, Dutta S. Test case prioritization technique based on early fault detection using fuzzy logic Sign In orPurchase. Paper presented at the Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference. New Delhi, India 2016.
54. Haraty RA, Mansour N, Moukaha L, Khalil I. Regression test cases prioritization using clustering and code change relevance. Int J Softw Eng Knowl Eng. 2016;26(5):733–68.
55. Kandil P, Moussa S, Badr N. Cluster-based test cases prioritization and selection technique for agile regression testing. J Softw Evol Process. 2017;29:1–19.
56. Hasan MA, Rahman MA, Siddik S. Test case prioritization based on dissimilarity clustering using historical data analysis. Paper presented at the International Conference on Information, Communication and Computing Technology, Singapore 2017.
57. Tumeng R, Jawawi DNA, Isa MA. Test case prioritization with textual comparison metrics. In *Software Engineering Conference (MySEC), 2015 9th Malaysian* (pp. 7–12). IEEE 2015.
58. Krishnamoorthi R, Mary SSA. Factor oriented requirement coverage based system test case prioritization of new and regression test cases. Inf Softw Technol. 2009;51(4):799–808.
59. Bjarnason E, Unterkalmsteiner M, Borg M, Engström E. A multi-case study of agile requirements engineering and the use of test cases as requirements. Inf Softw Technol. 2016;77:61–79.

60. Garousi V, Ozkan R, Betin-Can A. Multi-objective regression test selection in practice: an empirical study in the defense industry. Inf Softw Technol. 2018;103:40–54.
61. Tsai WT, Bai X, Paul R, Yu L. Scenario-based functional regression testing. In *Computer Software and Applications Conference, 2001. COMPSAC 2001. 25th Annual International* (pp. 496–501). IEEE 2001.
62. Salem YI, Hassan R. Requirement-based test case generation and prioritization. In *Computer Engineering Conference (ICENCO), 2010 International* (pp. 152–7). IEEE 2010.
63. Tahvili S, Ahlberg M, Fornander E, Afzal W, Saadatmand M, Bohlin M, Sarabi M. Functional Dependency Detection for Integration Test Cases. In *Companion of the 18th IEEE International Conference on Software Quality, Reliability, and Security* 2018.
64. Vescan A, Şerban C, Chisăliţă-Cretu C, Dioşan L. Requirement dependencies-based formal approach for test case prioritization in regression testing. In: *Intelligent Computer Communication and Processing (ICCP), 2017 13th IEEE International Conference on*. IEEE 2017:181–8.
65. Ahmad SF, Singh DK, Suman P. Prioritization for regression testing using ant colony optimization based on test factors. In: *Intelligent Communication, Control and Devices*. Springer, Singapore; 2018:1353–60.
66. Kumar S, Ranjan P, Rajesh R. A concept for test case prioritization based upon the priority information of early phase. In: *Proceedings of the International Conference on Signal, Networks, Computing, and Systems*. Springer, New Delhi, 2016;213–23.
67. Mahali P, Mohapatra DP. Model based test case prioritization using UML behavioural diagrams and association rule mining. Int J Sys Assur Eng Manag. 2018;2:1–17.
68. Gupta V, Chauhan DS, Dutta K. Regression testing-based requirement prioritization of mobile applications. Int J Syst Serv Orient Eng (IJSSOE). 2012;3(4):20–39.
69. Liu W, Wu X, Zhang W, Xu Y. The research of the test case prioritization algorithm for black box testing. Paper presented at the Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference Beijing, China, 2014.
70. Kumar S, Ranjan P, Rajesh R. An overview of test case optimization using Meta-heuristic approach. In: Recent advances in mathematics, statistics and computer science. 2016:475–85.
71. Pandey AK, Goyal NK. Reliability centric test case prioritization. In: Early software reliability prediction. India: Springer; 2013. p. 105–15.
72. Hillah LM, Maesano A-P, Rosa FD, Kordon F, Wuillemin P-H, Fontanelli R, Maesano L. Automation and intelligent scheduling of distributed system functional testing. Int J Softw Tools Technol Transfer. 2017;19:281–308.
73. Chen Y, Probert RL, Sims DP. Specification-based regression test selection with risk analysis. In: *Proceedings of the 2002 conference of the Centre for Advanced Studies on Collaborative research* (p. 1) (2002). IBM Press.
74. Do H, Hossain M. An efficient regression testing approach for PHP web applications: a controlled experiment. Softw Test Verif Reliab. 2014;24:367–85.
75. Barr E, Harman M, McMinn P, Shahbaz M, Yoo S. The oracle problem in software testing: a survey. IEEE Trans Softw Eng. 2015;41(5):507–25.
76. Yoo S, Harman M. Regression testing minimization, selection and prioritization: a survey. Softw Test Verif Reliab. 2012;22:67–120.
77. Laranjeiro N, Vieira M, Madeira H. A robustness testing approach for SOAPWeb services. J Internet Serv Appl. 2012;3:215–32.
78. Srikanth H, Williams L, Osborne J. System test case prioritization of new and regression test cases. In: Empirical Software Engineering, 2005. 2005 International Symposium on (pp. 10-pp). (2005) IEEE.
79. Zhou ZQ, Zhang S, Hagenbuchner M, Tse TH, Kuo FC, Chen TY. Automated functional testing of online search services. Softw Test Verif Reliab. 2012;22(4):221–43.
80. Wohlin C, Aurum A. Towards a decision-making structure for selecting a research design in empirical software engineering. Empir Software Eng. 2015;20:1427–55.
81. Alves V, Niu N, Alves C, Valenca G. Requirements engineering for software product lines: a systematic literature review. Inf Softw Technol. 2010;52:806–20.
82. Gotlieb A, Marijan D. Using global constraints to automate regression testing. AI Mag. 2017;38(1):73.
83. Jiang B, Tse TH, Grieskamp W, Kicillof N, Cao Y, Li X, Chan WK. Assuring the model evolution of protocol software specifications by regression testing process improvement. Softw Pract Exp. 2011;41(10):1073–103.
84. Piazolo F, Felderer M. Innovation and future of enterprise information systems. New York: Springer; 2013.
85. Mukherjee R, Patnaik KS. A survey on different approaches for software test case prioritization. J King Saud Univ Comput Inf Sci. 2018;2:2.
86. de Souza Neto JB, Moreira AM, Musicante MA. Semantic web services testing: a systematic mapping study. Comput Sci Rev. 2018;28:140–56.
87. Arrieta A, Wang S, Sagardui G, Etxeberria L. Search-based test case prioritization for simulation-based testing of cyber-physical system product lines. J Syst Softw. 2019;149:1–34.
88. Do H, Mirarab S, Tahvildari L, Rothermel G. The effects of time constraints on test case prioritization: a series of controlled experiments. IEEE Trans Softw Eng. 2010;36(5):593–617.
89. Singh Y, Kaur A, Suri B, Singhal S. Systematic literature review on regression test prioritization techniques. Informatica. 2012;36:4.
90. Andrews A, Alhaddad A, Boukhris S. Black-Box model-based regression testing of fail-Safe behavior in web applications. J Syst Softw. 2019;149:318–39.
91. Wang S, Huang K. Improving the efficiency of functional verification based on test prioritization. Microprocess Microsyst. 2016;41:1–11.
92. Zhu Q, Panichella A, Zaidman A. A systematic literature review of how mutation testing supports quality assurance processes. Softw Test Verif Reliab. 2018;28(6):e1675.
93. Ayav T. Prioritizing MCDC test cases by spectral analysis of Boolean functions. Softw Test Verif Reliab. 2017;27(7):e1641.
94. Awedikian R, Yannou B. A practical model-based statistical approach for generating functional test cases: application in the automotive industry. Softw Testing Verif Reliab. 2014;24(2):85–123.
95. Hemmati H, Fang Z, Mäntylä MV, Adams B. Prioritizing manual test cases in rapid release environments. Softw Test Verif Reliab. 2017;27(6):e1609.
96. Lity S, Nieke M, Thüm T, Schaefer I. Retest test selection for product-line regression testing of variants and versions of variants. J Syst Softw. 2019;147:46–63.
97. Ansari A, Khan A, Khan A, Mukadam K. Optimized regression test using test case prioritization. Proc Comput Sci. 2016;79:152–60.
98. Wang X, Zeng H, Gao H, Miao H, Lin W. Location-based test case prioritization for software embedded in mobile devices using the law of gravitation. Mobile Inf Syst. 2019;2019(2):2.
99. Nooraei Abadeh M, Mirian-Hosseinabadi SH. Delta-based regression testing: a formal framework towards model-driven regression testing. J Softw Evol Process. 2015;27(12):913–52.
100. Mardani A, Van Fan Y, Nilashi M, Hooker RE, Ozkul S, Streimikiene D, Loganathan N. A two-stage methodology based on

ensemble adaptive neuro-fuzzy inference system to predict carbon dioxide emissions. J Clean Prod. 2019;231:446–61.
101. Neto PADMS, Machado I, McGregor JD, De Almeida ES, de Lemos Meira SR. A systematic mapping study of software product lines testing. Inf Softw Technol. 2011;53(5):407–23.
102. Ahmad T, Iqbal J, Ashraf A, Truscan D, Porres I. Model-based testing using UML activity diagrams: a systematic mapping study. Comput Sci Rev. 2019;33:98–112.
103. Banias O. Test case selection-prioritization approach based on memoization dynamic programming algorithm. Inf Softw Technol. 2019;2:2.
104. Yoo S, Harman M, Tonella P, Susi A. Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge. In: *Proceedings of the eighteenth international symposium on Software testing and analysis* (pp. 201–12). 2009. ACM.
105. Ledru Y, Petrenko A, Boroday S, Mandran N. Prioritizing test cases with string distances. Autom Softw Eng. 2012;19(1):65–95.
106. Harikarthik S, Palanisamy V, Ramanathan P. Optimal test suite selection in regression testing with testcase prioritization using modified Ann and Whale optimization algorithm. Clust Comput. 2019;22(5):11425–34.
107. Gupta N, Yadav V, Singh M. Automated regression test case generation for web application: a survey. ACM Comput Surv (CSUR). 2018;51(4):1–25.
108. Kim J, Jeong H, Lee E. Failure history data-based test case prioritization for effective regression test. in Proceedings of the Symposium on Applied Computing. 2017.
109. Shin SY, et al. Test case prioritization for acceptance testing of cyber physical systems: a multi-objective search-based approach. in Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis. 2018.
110. Dahiya O, Solanki K. Prevailing standards in requirement-based test case prioritization: an overview. ICT Anal Appl. 2020;2:467–74.
111. Jahan H, Feng Z, Mahmud S. Risk-based test case prioritization by correlating system methods and their associated risks. Arabi J Sci Eng. 2020;2:2.
112. Paygude P, Joshi SD. Use of evolutionary algorithm in regression test case prioritization: a review. In: International conference on computer networks, big data and IoT. Springer, 2018.
113. Bajaj A, Sangwan OP. A systematic literature review of test case prioritization using genetic algorithms. IEEE Access. 2019;7:126355–75.
114. Parsons D, Susnjak T, Lange M. Influences on regression testing strategies in agile software development environments. Software Qual J. 2014;22(4):717–39.
115. Sun X, et al. ComboRT: a new approach for generating regression test cases for evolving programs. Int J Software Eng Knowl Eng. 2016;26(06):1001–26.
116. Bian Y, et al. Concrete hyperheuristic framework for test case prioritization. J Softw Evol Process. 2018;30(11):e1992.
117. Ahmed BS, Abdulsamad TS, Potrus MY. Achievement of minimized combinatorial test suite for configuration-aware software functional testing using the cuckoo search algorithm. Inf Softw Technol. 2015;66:13–29.
118. Miranda B, et al. Fast approaches to scalable similarity-based test case prioritization. in 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE). 2018. IEEE.