Empirical Evaluations of Regression Test Selection Techniques: A Systematic Review

Emelie Engström
Department of Computer Science
Lund University
SE-221 00 LUND
+46 46 222 88 99
emelie.engstrom@cs.lth.se

Mats Skoglund
Department of Computer Science
Lund University
SE-221 00 LUND
mats.skoglund@cs.lth.se

Per Runeson
Department of Computer Science
Lund University
SE-221 00 LUND
+46 46 222 93 25
per.runeson@cs.lth.se

ABSTRACT

Regression testing is the verification that previously functioning software remains after a change. In this paper we report on a systematic review of empirical evaluations of regression test selection techniques, published in major software engineering journals and conferences. Out of 2 923 papers analyzed in this systematic review, we identified 28 papers reporting on empirical comparative evaluations of regression test selection techniques. They report on 38 unique studies (23 experiments and 15 case studies), and in total 32 different techniques for regression test selection are evaluated. Our study concludes that no clear picture of the evaluated techniques can be provided based on existing empirical evidence, except for a small group of related techniques. Instead, we identified a need for more and better empirical studies were concepts are evaluated rather than small variations. It is also necessary to carefully consider the context in which studies are undertaken.

Categories and Subject Descriptors

D.2.5 Testing and Debugging

General Terms

Measurement, Experimentation, Verification.

Keywords

Systematic review, regression testing, test selection

1. INTRODUCTION

Efficient regression testing is important, even crucial, for organizations with a large share of their cost in software development. It involves, among other tasks, determining which test cases need to be re-executed in order to verify the behavior of modified software. Iterative development strategies and reuse are common means of saving time and effort for the development. However they both require frequent retesting of previously tested functions due to changes in related code. The need for good testing strategies is thus becoming more and more important.

As regression testing tends to take a larger and larger share of total costs in the development of complex software systems, a great deal of research effort is being spent on finding cost-efficient methods for regression testing in a variety of topics. Examples include test case selection based on code changes [1, 10, 13, 17, 19, 39, 43, 51, 53, 57, 59] and specification changes [7, 36, 45, 58], evaluation of selection techniques [41], change impact analysis [40], regression tests for different applications e.g. on database application [15], regression testing of GUIs and test automation [35], and test process enhancement [29]. To deal with these many problems, researchers have typically divided the main ones into test selection, modification identification, test execution and test suite maintenance. This study is focused on the analysis of regression test selection techniques.

Although several techniques for regression test selection have been evaluated in previous work [3, 11, 33, 54], no general solution has been put forward since no technique could possibly respond adequately to the complexity of the problem and the great diversity in requirements and preconditions in software systems and development organizations. Neither does any single study evaluate every aspect of the problem, e.g. in [26] the effects of regression test application frequency is evaluated, [9] investigates the impact of different modifications on regression testing techniques, several studies examine the ability to reduce regression testing effort [3, 9, 11, 26, 33, 54, 56] and to reveal faults [9, 11, 12, 26].

By means of a systematic review, we collected and compared existing evidence on regression test selection. The use of systematic reviews in the software engineering domain has been subject of a growing interest in the last years. In 2004 Barbara Kitchenham proposed a guideline adapted to the specific problems of software engineering research. This guideline has been followed and evaluated [5, 28, 47] and updated accordingly in 2007 [27]. If several studies evaluate the same techniques under similar conditions on different subject programs, there is a possibility to perform an aggregation of findings and thus strengthen the ability to draw general conclusions. In this review however we found that the existing studies were diverse, thus hindering such aggregation. Instead we present a qualitative analysis of the findings, an overview of existing techniques for regression test selection and of the amount and quality of empirical evidence.

ESEM'08, October 9–10, 2008, Kaiserslautern, Germany.
Copyright 2008 ACM 978-1-59593-971-5/08/10...$5.00.

In 2004 Hyunsook et. al. presented a survey of empirical studies in software testing in general [21]. Their study covered two journals and four conferences. The starting point for our study is several electronic databases with the purpose to cover as many relevant journals and conference and workshop proceedings as possible. Our focus was on techniques for regression test selection while Hyunsook et al. included all studies regarding testing. Earlier reviews of regression test selection are not exhaustive but compare a small number of chosen regression test selection techniques. Rothermel and Harrold presented a framework for evaluating regression test techniques [41, 42] and evaluated some existing techniques. Juristo et al. aggregated results from unit testing experiments [24] of which some evaluate regression testing techniques. Binkley et al. reviewed research on the application of program slicing to the problem of regression testing [4]. Hartman et al. reports a survey and critical assessment of regression testing tools [18]. However, as far as we know, no systematic review on regression test selection research has been carried through.

This paper is organized as follows. In section 2 the method used for our study is described. Section 3 reports and discusses the results. Section 4 concludes the work.

2. RESEARCH METHOD

2.1 Research Questions

This review aims at summarizing the current state of the art in regression test selection research by proposing answers to the following questions:

1) Which techniques for regression test selection in the literature have been evaluated empirically?
2) Can these techniques be classified, and if so, how?
3) Are there significant differences between these techniques that can be established using empirical evidence?
4) Can technique *A* be shown to be superior to technique *B*, based on empirical evidence?

2.2 Sources of information

In order to gain a broad perspective, as recommended in Kitchenham's guidelines [27], we searched widely in electronic sources. The following databases were covered:

- Inspec (<www.theiet.org/publishing/inspec/>)
- Compendex (<www.engineeringvillage2.org>)
- ACM Digital Library (<portal.acm.org>)
- IEEE eXplore (<ieeexplore.ieee.org>)
- ScienceDirect (<www.sciencedirect.com>)
- Springer LNCS (<www.springer.com/lncs>)
- Web of Science(<www.isiknowledge.com>)

These databases cover the most relevant journals and conference and workshop proceedings within software engineering, as confirmed by Dybå et al. [8]. Grey literature (technical reports, some workshop reports, work in progress) was excluded from the analysis for two reasons: the quality of the grey literature is more difficult to assess and the volume of studies included in the first searches would have grown to an unreasonable amount. The searches in the sources selected resulted in overlap among the papers, where the duplicates were excluded primarily by manual filtering.

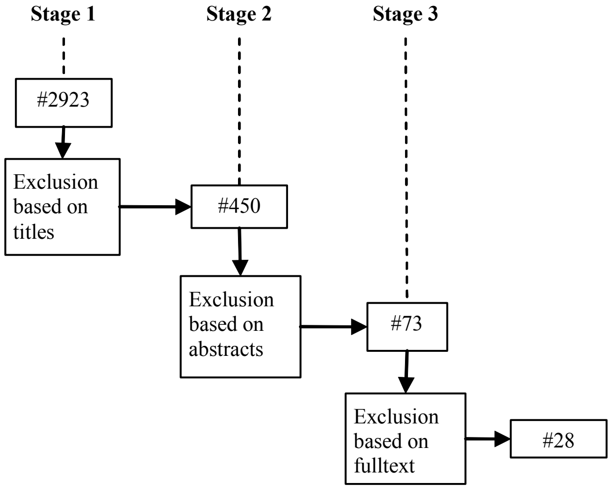
Stage 1
Stage 2
Stage 3
#2923
Exclusion based on titles
#450
Exclusion based on abstracts
#73
Exclusion based on fulltext
#28

Figure 1 Study selection procedure

2.3 Search criteria

The initial search criteria were broad in order to include articles with different uses of terminology. The key words used were <regression> and (<test> or <testing>) and <software>. The start year was set to 1969 to ensure that all relevant research within the field would be included, and the last date for inclusion is publications within 2006. Kitchenham recommends that exclusion based on languages should be avoided [27]. At least one paper in the original selection was in Japanese, and since we were not able to read Japanese, we could not follow this recommendation. Thus, only papers written in English are included. The initial search located 2 923 potentially relevant papers.

2.4 Study Selection

In order to obtain independent assessments, four researchers were involved in a three stage selection process, see Figure 1. In the first stage duplicates and irrelevant papers were excluded manually based on titles. In our case, the share of irrelevant papers was extremely large since papers on software for *statistical* regression testing could not be distinguished from papers on *software* regression testing in the database search. After the first stage 450 papers remained. In the second stage information in abstracts was analyzed and the papers were classified along two dimensions: research approach and regression testing approach. Papers not presenting an empirical research approach were excluded as well as papers not focusing on regression test selection, e.g. papers on test suite maintenance or test automation. In the third stage a full text analysis was performed and the empirical quality of the studies was further assessed. The following questions were asked in order to form an opinion about which studies to exclude or include for final data extraction:

- Is a specific regression test selection method evaluated?

- Are the metrics and the results relevant for a comparison of methods?
- Is data collected and analyzed in a sufficiently rigorous manner?

These questions are derived from a list of questions, used for a similar purpose, published by Dybå et al. [8]. However in our review context, quality requirements for inclusion had to be weaker than suggested by Dybå et al. in order to obtain a useful set of studies to compare. Abstract analysis and full text analysis were performed independently by two of the researchers with a third researcher acted as a checker. Using this procedure, 28 articles were finally selected that reported on 38 unique empirical studies, evaluating 32 different techniques.

2.5 Threats to validity

Threats to the validity of the systematic review are analyzed according to the following taxonomy; construct validity, reliability, internal validity and external validity.

Construct validity reflects to what extent the phenomenon under study really represents what the researchers have in mind and what is investigated according to the research questions. The main threat here is related to terminology. Since the systematic review is based on a hierarchical structure of terms – regression test/testing consists of the activities modification identification, test selection, test execution and test suite maintenance – we might miss other relevant studies on test selection. However, this is a consciously decided limitation which has to be taken into account in the use of the results.

Reliability focuses on whether the data is collected and the analysis is conducted in a way that it can be repeated by other researchers with the same results. In a systematic review, the inclusion and exclusion of studies is the major focus here, especially in this case where another domain (statistics) also uses the term regression testing. Our countermeasures taken were to set up criteria and to use two researchers to classify papers in stages 2 and 3. In cases of disagreement, a third opinion is used. One of the primary researchers was changed between stages 2 and 3. Still, the uncertainties in the classifications are prevalent and a major threat to reliability, especially since the quality standards for empirical studies in software engineering are not high enough. Research databases is another threat to reliability [8]. The threat is reduced by using multiple databases; still the non-determinism of some database searches is a major threat to the reliability of any systematic review.

Internal validity is concerned with the analysis of the data. Since no statistical analysis was possible due to the inconsistencies between studies, the analysis is mostly qualitative. Hence we link the conclusions as clearly as possible to the studies which underpin our discussions.

External validity is about generalizations of the findings. On the meta level, i.e. conducting systematic review, we have gained very much the same experiences as reported by other systematic reviews in software engineering [5, 8, 28, 47]. On the study level, i.e. regression testing, the threats to external validity are concerned with the context of the study. Most studies are conducted on small programs and hence generalizing them to a full industry context is not possible.

3. RESULTS AND DISCUSSION

The goal of this study was to determine whether the literature on regression test selection techniques provides a uniform and rigorous base of empirical evidence. The papers were initially obtained in a broad search in seven databases covering relevant journals, conference and workshop proceedings within software engineering. Then an extensive systematic selection process was carried out to identify papers describing empirical evaluations of regression test selection techniques. Only papers with very poorly reported or poorly conducted studies are excluded, as well as papers where the comparisons made were considered irrelevant to the goals of this study. The results presented here thus give a good picture of the existing evidence base.

Of 2 923 papers analyzed in the systematic review, we identified 28 on empirical evaluations of techniques for regression test selection. The papers report on 38 unique studies and provided in total 32 different techniques for regression test selection for evaluation. Five reference techniques are also identified, e.g. re-test all and random(25), which randomly selects 25% of the test cases. A paper may report on several studies and in some cases the same study is reported in more than one paper. This distribution is shown in Table 1. Note that many of the techniques are originally presented in papers without empirical evaluation. These papers are not included in the systematic review, but referenced in Section 3.1 as sources of information about the techniques as such.

Table 1 Distribution of number of papers after the number of studies each paper reports

# reported studies in each paper	# papers	# studies
0 (re-analysis of another study)	2	0
1	18	18
2	6	12
3	1	3
5	1	5
Total	**28**	**38**

Table 2 lists the different publication fora in which the articles have been published, and Table 3 lists authors with more than one publication. In addition to these 14 authors presented in the table 39 researchers have authored or co-authored one paper each.

It is worth noting regarding the publication fora, that the empirical regression testing papers are published in a wide variety of journals and conference proceedings. Limiting the search to fewer journals and proceedings would have missed many papers, see Table 2.

The major software engineering journals and conferences are represented among the fora. It is not surprising that a conference on software maintenance is on the top, but it is remarkable that the International Symposium on Software Testing and Analysis is not on the list at all. However, for testing in general, empirical studies have been published there [21] but apparently not on regression test selection.

Table 2 Number of papers in different publication fora

Publication Fora	Type	#	%
International Conference on Software Maintenance	Conference	5	17.9
ACM Transactions of Software Engineering and Methodology	Journal	4	14.2
International Symposium on Software Reliability Engineering	Conference	3	10.7
International Conference on Software Engineering	Conference	2	7.1
Journal of Software Maintenance and Evolution	Journal	2	7.1
Asia-Pacific Software Engineering Conference	Conference	2	7.1
International Symposium on Empirical Software Engineering	Conference	2	7.1
IEEE Transactions of Software Engineering		1	3.6
Journal of Systems and Software	Journal	1	3.6
Software Testing Verification and Reliability	Journal	1	3.6
ACM SIGSOFT Symposium on Foundations of SE	Conference	1	3.6
Automated Software Engineering	Conference	1	3.6
Australian SE Conference	Conference	1	3.6
International Conf on COTS-based Software Systems	Conference	1	3.6
Int. Conference on Object-Oriented Programming, Systems, Languages, and Applications	Conference	1	3.6
Total		**28**	**100**

Table 3 Researchers and number of publications

Name	#
Rothermel G.	9
Harrold M. J.	5
Robinson B.	5
Zheng J.	4
Elbaum S. G.	3
Kallakuri P.	3
Malishevsky A.	3
Mansour N.	3
Smiley K.	3

Name	#
Williams L.	3
Baradhi G.	2
Frankl P. G.	2
Kim J. M.	2
Orso A.	2
Porter A.	2
White L.	2
Vokolos F.	2

3.1 Existing techniques

Table 4 lists the 32 different regression test selection techniques, in chronological order according to date of first publication. In this review, the techniques, their origin and description, are identified in accordance to what is stated in each of the selected papers.

Table 4 Techniques for regression test selection

Technique	Origin	Description
T1	Harrold and Soffa (1988) [17]	Dataflow-coverage-based
T2	Fischer et al. (1981) [10] Hartman and Robson (1990) [19]	Modification-focused, minimization, branch and bound algorithm
T3	Leung and White (1990) [32]	Procedural-design firewall
T4	Gupta et al. (1992) [13]	Coverage-focused, slicing
T5	White and Leung (1992) [51]	Firewall
T6	Agraval et al. (1993) [1]	Incremental
T7	Chen and Rosenblum (1994) [6]	Modified entity
T8	Gupta et al. (1996) [14]	Slicing
T9	Pei et al. (1997) [39]	High level – identifies changes at the class and interface level
T10	White and Abdullah (1997) [49]	High level – identifies changes at the class and interface level
T11	Vokolos and Frankl (1997) [53]	Textual Differing
T12	Rothermel and Harrold (1997) [43]	Viewing statements, DejaVu
T13	Mansour and Fakih (1997) [34]	Genetic algorithm
T14	Mansour and Fakih (1997) [34]	Simulated annealing
T15	Wong et al. (1997) [55]	Hybrid: modification, minimization and prioritization- based selection
T16	Rothermel et al. (2000) [44]	Edge level-identifies changes at the edge level
T17	Wu et al. (1999) [57]	Analysis of program structure and function-calling sequences
T18	Harrold et al. (2001) [16]	Edge level – identifies changes at the edge level

T19	Orso et al. (2001) [36]	Use of metadata to represent links between changes and Test Cases
T20	Sajeev et al. (2003) [45]	Use of UML (OCL) to describe information changes
T21	Elbaum et al. (2003) [9]	Same as T7 but ignoring core functions modified-non-core
T22	Koju et al. (2003) [30]	Safe technique for virtual machine based programs
T23	Orso et al. (2004) [37]	Partitioning and selection Two Phases
T24	Pasala and Bhowmick (2005) [38]	Runtime dependencies captured and moduled into a graph (CIG)
T25	Skoglund and Runeson (2005) [46]	Change based selection
T26	Willmor and Embury (2005)[52]	Test selection for DB-driven applications (extension of T12) combined safety
T27	Willmor and Embury (2005) [52]	Database safety
T28	Chengying and Yansheng (2005) [7]	Enhanced representation of change information
T29	White et al. (2005) [50, 60]	Extended firewall additional data-paths
T30	Zheng (2005) [60]	I-BACCI v.1
T31	Zheng et al. (2006)[61]	I-BACCI v.2 (firewall + BACCI)
T32	Zheng et al. (2006)[31, 61]	I-BACCI v.3
REF1	Leung and White (1989) [31]	Retest-all
REF2		Random (25)
REF3		Random (50)
REF4		Random (75)
REF5		Intuitive, experience based selection

In the studied material we identify that there is no clear definition of what constitutes a *technique* and thus there is a problem in determining which regression test techniques exist. The first research question we stated in Section 2.1 regards which empirically evaluated regression test selection techniques exist in the literature. This research question depends on the possibility to uniquely identify each of the techniques used in the literature. Two different aspects where this problem of not having a clear definition manifests itself are regarding:

1. The uniqueness of techniques
2. The difference between the specifications of techniques and their implementations

There is a great variance regarding the uniqueness of the techniques identified in the studied papers. Some techniques may be regarded as novel at the time of their first presentation, while others may be regarded as only variants of already existing techniques. For example in [3] a regression test selection techniques is evaluated, T7, and the technique used is based on modified entities in the subject programs. In another evaluation, reported on in [9] it is stated that the same technique is used as in [3], but adapted to use a different scope of what parts of the subjects programs that is included in the analysis, T21. In [3] the complete subject programs are included in the analysis, while in [9] core functions of the subject programs are ignored. This difference of scope probably have an effect on the test cases selected using the two different approaches. The approach with ignored core functions is likely to select fewer test cases compared to the approach where all parts of the programs are included. It is not obvious whether the two approaches should be regarded as two different techniques or if they should be regarded as two very similar variants of the same technique.

Some techniques evaluated in the reviewed papers are specified to be used for a specific type of software, e.g. Java, T18 and T23 [16, 37], component based software, T20, T24, T28 and T32 [2, 7, 22, 23], or database-driven applications, T26, [52]. It is not clear whether they should be considered one technique applied to two types of software, or two distinctly different techniques. For example, a technique specified for Java, T18, is presented and evaluated in [16]. In [48] the same technique is used on MSIL (MicroSoft Intermediate Language) code, T22, however adapted to cope with programming language constructs not present in Java. Thus, it can be argued that the results of the two studies cannot be synthesized in order to draw conclusions regarding the performance of neither the technique presented in [16], T18, nor the adapted version, T22, used in [30].

There are also techniques specified in a somewhat abstract manner, e.g. techniques that handle object-oriented programs in general, T17 [57]. However, when evaluating a technique, the abstract specification of a technique must be concretized to handle the specific type of subjects selected for the evaluation. The concretization may look different depending on the programming language used for the subject programs. T17 is based on dependencies between functions in object-oriented programs in general. The technique is evaluated by first concretizing the abstract specification of the technique to C++ programs and then performing the evaluation on subject programs in C++. However, it is not clear how the concretization of the specification should be performed to evaluate the technique using other object-oriented programming languages, e.g. C# or Java. Thus, due to differences between programming languages, a concretization made for one specific programming language may have different general performance than a concretization made for another programming language.

The performance of a regression test selection technique may be very sensitive to changes in the implementation of the technique, the analysis scope used, on the type of input given to drive the technique or other issues regarding e.g. the use of a technique. This sensitivity makes it difficult to assert whether a specific study can be considered to have evaluated the exact same technique as another study. Without further studies of this sensitivity, little can be said about what degree of impact a subtle difference in implementing or use of a technique has on the performance of a technique. The difficulties of uniquely identifying which techniques that have been used in which studies also have the consequence that it is difficult to determine which techniques that have been empirically evaluated.

In this review, the techniques are identified in accordance to what is stated in each of the selected papers. Due to this, the number of identified techniques is relatively high compared to the number of studies, 32 techniques were found in 38 studies. In Table 5, the distribution of techniques over different studies is presented. One technique was present in 12 different studies, another technique in 7 studies etc. 18 techniques only appear in one study.

Table 5 Distribution of techniques after occurrences in number of studies

Represented in no studies	Number of techniques
12	1
7	1
6	0
5	1
4	1
3	3
2	7
1	18
Total	**32**

3.2 Classification of techniques

Since the techniques are sensitive to subtle changes in their implementation or use, we could compare classes of techniques, instead of comparing individual techniques. Some suggestions of classifications of regression test techniques exist. Graves et al. [11] present a classification scheme where techniques are classified as Minimization, Safe, Dataflow-Coverage-based, Ad-hoc/Random or Retest-All techniques. Orso et al. [37] separate between techniques that operate at a higher granularity e.g. method or class (called high-level) and techniques that operate at a finer granularity, e.g. statements (called low-level). In this review we searched for classifications in the papers themselves with the goal of finding common properties to be able to reason about groups of regression testing techniques.

One property found regards the type of input required by the techniques. The most common type of required input is source code text, e.g. T1-8, T11-14, T17 and T21. Other types of code analyzed by techniques are intermediate code for virtual machines, e.g. T9-10, T15-16, T18, T22 and T24, or machine code, e.g. T28-32**.** Some techniques require input of a certain format, e.g. T19 (meta data) and T20 (OCL) Techniques may also be classified according to the type of code used in the analysis (Java, C++…). A third type of classification that could be extracted from the papers regards the programming language paradigm. Some techniques are specified for use with procedural code, e.g. T1-2, T7, T11-12, and T21, while other techniques are specified for the object-oriented paradigm, e.g. T9, T10, T15-18, T22 and T24-T26 some techniques are independent of programming language, e.g. T3, and T28-32.

The most found property assigned to regression test selection techniques is whether they are *safe* or *unsafe*. With a safe technique the defects found with the full test suite are also found with the test cases picked by the regression test selection technique. This property may be used to classify all regression test selection techniques into either *safe* or *unsafe* techniques. Re-test all is an example of a safe technique since it selects all test cases, hence, it is guaranteed that all test cases that reveal defects are selected. Random selection of test cases is an example of an unsafe technique since there is a risk of test cases revealing defect may be missed. In our study seven techniques were stated by the authors to be safe, T7, T11, T12, T22, T25-27.

3.3 Empirical Evidence

Table 6 overviews the studies by research method, and the size of the system used as subject. We identified 23 unique controlled experiments and 15 unique case studies. Half of the experiments are conducted on the same set of small programs [20], often referred to as the Siemens programs, which are made available through the software infrastructure repository.[1] The number of large scale real life evaluations is sparse. In this systematic review we found four. Both types of studies have benefits and encounter problems, and it would be of interest to study the link between them, i.e. does a technique which is shown to have great advantages in a small controlled experiment show the same advantages in a large scale case study. Unfortunately no such link was found in this review.

Table 6 Studies of different type and size

Type of studies	Size of subjects under study	Number of studies	%
Experiment	Large	1	3
Experiment	Medium	7	18
Experiment	Small	15	39
Case study	Large	4	11
Case study	Medium	5	13
Case study	Small	4	11
Case study	Not reported	2	5
	Total	**38**	**100**

The empirical quality of the studies varies a lot (from well designed and reported experiments on medium to large sized systems to case studies on small programs, see Table 6). In order to obtain a sufficiently large amount of papers, our inclusion criteria regarding quality had to be weak. Included in our analysis was any empirical evaluation of regression test selection

[1] http://sir.unl.edu

techniques if data collection and analysis could be followed in the report. This was independently assessed by two researchers.

There is no common definition of what criteria defines a good regression test selection method. We identified two main categories of metrics used: *cost reduction* and *ability to detect faults*. Five different aspects of cost reduction and two of fault detection effectiveness have been evaluated in the studies. Table 7 gives an overview of the extent to which the different metrics are used in the studies. Size of test suit reduction is the most frequent, evaluated in 76% of the studies. Despite this it may not be the most important metric. If the cost for performing the selection is too large in relation to this reduction no savings are achieved. In 42% of the studies the total time (selection and test execution) is evaluated instead or as well. The value of using a certain metric is dependent on the design of the study.

Several of the studies concerning reduction of number of test cases are only compared to retest all, [16, 30, 55, 57, 60, 61] with the only conclusion that a reduction of test cases can be achieved, but nothing on the size of the effect in practice. This is a problem identified in experimental studies in general [25]. Several of the studies evaluating time reduction are conducted on small programs, and the size of the differences is measured in milliseconds. Only 30% of the studies consider both fault detection and cost reduction. Rothermel proposed a framework for evaluation of regression test selection techniques [41] which have been used in some evaluations. This framework defines four key metrics, inclusiveness, precision, efficiency, and generality. Inclusiveness and precision corresponds to relative fault detection and precision, respectively, in Table 7. Efficiency is related to space and time requirements and generality is more of a theoretical reasoning.

Table 7 Use of metrics in the studies

	Evaluated Metrics	**Number**	**%**
Cost reduction	Test suite reduction	29	76
	Test execution time	7	18
	Test selection time	5	13
	Total time	16	42
	Precision (omission of non-fault revealing tests)	1	3
Ability to detect faults	Relative Fault detection effectiveness	5	13
	Absolute Fault detection effectiveness	8	21

An overview of the empirically studied relations between techniques and studies are shown in Figure 2. Techniques are represented by circles and comparative studies are represented with connective lines between the techniques. CS on the lines refers to the number of case studies conducted in which the techniques are compared, and Exp denotes the number of experimental comparisons. Dashed circles encapsulate closely related techniques. Some techniques have not been compared to any of the other techniques in the diagram: T15, T17, T22, and T24. These techniques are still empirically evaluated in at least one study, typically a large scale case study. If no comparison between proposed techniques is made, the techniques are compared to a reference technique instead, e.g. the retest of all test cases, and in some cases a random selection of a certain percentage of test cases is used as a reference as well.

Researchers are more apt to evaluate new techniques or variants of techniques than to replicate studies, which is clearly indicated by that we identified 32 different techniques in 28 papers. This gives rise to several clusters of similar techniques compared among them selves and several techniques only compared to a reference method such as re-test all. Thus no coherent chain of evidence was identified.

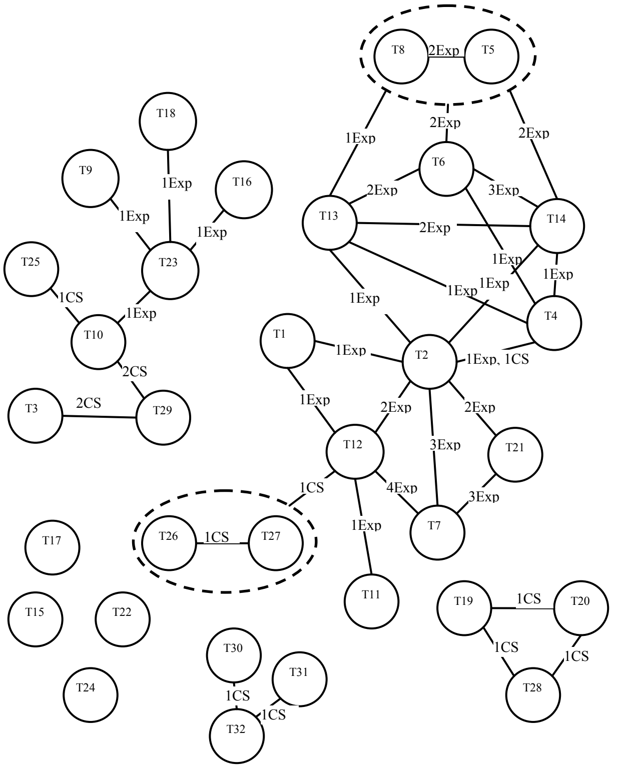

Figure 2 Techniques related to each other through empirical comparisons

One group of techniques has been evaluated sufficiently to allow for meaningful comparison, T2, T6, T7, T12, T14 and T21. Each of these techniques has been evaluated in at least three controlled experiments. If only the lines representing at least 2 experiments in Figure 2 are considered, the techniques constitute two clusters. A comparison of the techniques in the larger cluster indicates that the minimization technique, T2, is the most efficient in reducing time and/or number of test cases to run. However this is an unsafe technique (see Section 3.2) and all but one of six studies report on significant losses in fault detection. When it comes to safe techniques, the incremental technique presented by Agraval et al. [1] T6, is shown to be the most efficient in reducing test cases. However analysis time for T6 is shown to be too long (it exceeds the time for rerunning all test cases) in several early experiments but in later experiments, it is shown to be good. It is interesting to notice that the technique is not changed between the studies, but the subjects on which the experiments are conducted. This emphasizes the importance of the regression testing context in empirical studies.

4. CONCLUSIONS AND FUTURE WORK

In this paper we present results from a systematic review of empirical evaluations of regression test selection techniques. Related to our research questions we have identified that:

1) there are 32 empirically evaluated techniques on regression test selection published,
2) these techniques might be classified according to the input needed for the technique, type of code or programming paradigm. Classification in safe/unsafe is also an option,
3) the empirical evidence for differences between the techniques is not very strong, and sometimes contradictory, and
4) hence there is no basis for selecting one superior technique.

We have identified some very basic problems in the regression testing field which hinders a systematic review of the studies. Firstly, there is a great variance in the uniqueness of the techniques identified. Some techniques may be presented as novel at the time of their publications and others may be regarded as variants of already existing techniques. Combined with a tendency to consider replications as second class research, the case for cooperative learning on regression testing techniques is not good. In addition to this, some techniques are presented in a rather general manner, e.g. claimed to handle object-oriented programs, which gives much space for different interpretations on how they may be implemented due to e.g. different programming language constructs existing in different programming languages. This may lead to different (but similar) implementations of a specific technique in different studies depending on e.g. the programming languages used in the studies.

As mentioned in Section 1, to be able to select a strategy for regression testing, relevant empirical comparisons between different methods are required. Where such empirical comparisons exist, the quality of the evaluations must be considered. One goal of this study was to determine whether the literature on regression test selection techniques provides such uniform and rigorous base of empirical evidence on the topic that makes it possible to use it as a base for selecting a regression test selection method for a given software system.

Our study shows that most of the techniques presented are not evaluated sufficiently for a practitioner to make decisions based on research alone. In many studies, only one aspect of the problem is evaluated and the context is too specific to be easily applied directly by software developers. Few studies are replicated, and thus the possibility to draw conclusions based on variations in test context is limited. Of course even a limited evidence base could be used as guidance. In order for a practitioner to make use of these results, the study context must be considered and compared to the actual environment into which a technique is supposed to be applied.

Future work for the research community is 1) to more specifically define and agree on what is considered a regression test technique; 2) encourage systematic replications of studies in different context, preferably with a focus on gradually scaling up to more complex environments; 3) define how empirical evaluations of regression test selection techniques should be reported, which variation factors in the study context are important.

5. ACKNOWLEDGMENTS

The authors acknowledge Dr. Carina Andersson for her contribution to the first two stages of the study. The authors are thankful to librarian Maria Johnsson for excellent support in the search procedures. The work is partly funded by the Swedish Governmental Agency for Innovation Systems under grant 2005-02483 for the UPPREPA project, and partly funded by the Swedish Research Council under grant 622-2004-552 for a senior researcher position in software engineering.

6. REFERENCES

[1] Agrawal, H., Horgan, J.R., Krauser, E.W., and London, S.A. 1993. Incremental regression testing. In Proceedings. Conference on Software Maintenance 1993. CSM-93 (Cat. No.93CH3360-5). IEEE Comput. Soc. Press, 348-57.

[2] Anjaneyulu, P. and Animesh, B. 2005. An approach for test suite selection to validate applications on deployment of COTS upgrades. In Proceedings. 12th Asia-Pacific Software Engineering Conference. IEEE Computer Society, 7 pp.

[3] Bible, J., Rothermel, G., and Rosenblum, D.S. 2001. A comparative study of coarse- and fine-grained safe regression test-selection techniques. ACM Transactions on Software Engineering and Methodology. 10(2), 149-183.

[4] Binkley, D. 1998. The application of program slicing to regression testing. Information and Software Technology. 40(11-12), 583-94.

[5] Brereton, P., Kitchenham, B.A., Budgen, D., Turner, M., and Khalil, M. 2007. Lessons from applying the systematic literature review process within the software engineering domain. Journal of Systems and Software. 80(4), 571-83.

[6] Chen, Y.-F., Rosenblum, D.S., and Vo, K.-P. 1994. Test tube: A system for selective regression testing. In Proceedings - International Conference on Software Engineering. IEEE, Los Alamitos, CA, USA, 211-220.

[7] Chengying, M. and Yansheng, L. 2005. Regression testing for component-based software systems by enhancing change information. In Proceedings. 12th Asia-Pacific Software Engineering Conference. IEEE Computer Society, 8 pp.

[8] Dybå, T., Dingsöyr, T., and Hanssen, G.K. 2007. Applying Systematic Reviews to Diverse Study Types: An Experience Report. In First International Symposium on Empirical Software Engineering and Measurement, 2007, ESEM 2007. 225-234.

[9] Elbaum, S., Kallakuri, P., Malishevsky, A., Rothermel, G., and Kanduri, S. 2003. Understanding the effects of changes on the cost-effectiveness of regression testing techniques. Software Testing, Verification and Reliability. 13(2), 65-83.

[10] Fischer, K., Raji, F., and Chruscicki, A. 1981. A methodology for retesting modified software. In NTC '81. IEEE 1981

National Telecommunications Conference. Innovative Telecommunications - Key to the Future. IEEE, 6-3.

[11] Graves, T.L., Harrold, M.J., Kim, J.M., Porter, A., and Rothermel, G. 2001. An empirical study of regression test selection techniques. ACM Transactions on Software Engineering and Methodology. 10(2), 184-208.

[12] Gregg, R. and Mary Jean, H. 2000. A Safe, Efficient Regression Test Selection Technique.

[13] Gupta, R., Harrold, M.J., and Soffa, M.L. 1992. An approach to regression testing using slicing. In Conference on Software Maintenance 1992 (Cat.No.92CH3206-0). IEEE Comput. Soc. Press, 299-308.

[14] Gupta, R., Harrold, M.J., and Soffa, M.L. 1996. Program slicing-based regression testing techniques. Software Testing, Verification and Reliability. 6(2), 83-111.

[15] Haftmann, F., Kossmann, D., and Lo, E. 2007. A framework for efficient regression tests on database applications. VLDB Journal. 16(1), 145-64.

[16] Harrold, M.J., Jones, J.A., Tongyu, L., Donglin, L., Orso, A., Pennings, M., Sinha, S., Spoon, S.A., and Gujarathi, A. 2001. Regression test selection for Java software. In SIGPLAN Not. (USA). ACM, 312-26.

[17] Harrold, M.J. and Souffa, M.L. 1988. An incremental approach to unit testing during maintenance. In Proceedings of the Conference on Software Maintenance - 1988 (IEEE Cat. No.88CH2615-3). IEEE Comput. Soc. Press, 362-7.

[18] Hartmann, J. and Robson, D.J. 1988. Approaches to regression testing. In Proceedings of the Conference on Software Maintenance - 1988 (IEEE Cat. No.88CH2615-3). IEEE Comput. Soc. Press, 368-72.

[19] Hartmann, J. and Robson, D.J. 1990. Techniques for selective revalidation. IEEE Software. 7(1), 31-6.

[20] Hutchins, M., Foster, H., Goradia, T., and Ostrand, T. 1994. Experiments on the effectiveness of dataflow- and control-flow-based test adequacy criteria. In ICSE-16. 16th International Conference on Software Engineering (Cat. No.94CH3409-0). IEEE Comput. Soc. Press, 191-200.

[21] Hyunsook, D., Elbaum, S., and Rothermel, G. 2004. Infrastructure support for controlled experimentation with software testing and regression testing techniques. In 2004 International Symposium on Empirical Software Engineering. IEEE Comput. Soc, 60-70.

[22] Jiang, Z., Robinson, B., Williams, L., and Smiley, K. 2006. Applying regression test selection for COTS-based applications. In 28th International Conference on Software Engineering Proceedings. ACM, 512-21.

[23] Jiang, Z., Robinson, B., Williams, L., and Smiley, K. 2006. A lightweight process for change identification and regression test selection in using COTS components. In Fifth International Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems. IEEE Computer Society, 7 pp.

[24] Juristo, N., Moreno, A.M., Vegas, S., and Solari, M. 2006. In search of what we experimentally know about unit testing [software testing]. IEEE Software. 23(6), 72-80.

[25] Kampenes Vigdis, B., Dybå, T., Hannay Jo, E., and Sjöberg Dag, I.K. 2007. A systematic review of effect size in software engineering experiments. Information and Software Technology. 49(11-12), 1073-1073.

[26] Kim, J.-M., Porter, A., and Rothermel, G. 2005. An empirical study of regression test application frequency. Software Testing, Verification and Reliability. 15(4), 257-279.

[27] Kitchenham, B.A. 2007. Guidelines for performing Systematic Literature reviews in Software Engineering Version 2.3. Technical Report S.o.C.S.a.M. Software Engineering Group, Keele University and Department of Computer Science University of Durham.

[28] Kitchenham, B.A., Mendes, E., and Travassos, G.H. 2007. Cross versus within-company cost estimation studies: a systematic review. IEEE Transactions on Software Engineering. 33(5), 316-29.

[29] Klosch, R.R., Glaser, P.W., and Truschnegg, R.J. 2002. A testing approach for large system portfolios in industrial environments. Journal of Systems and Software. 62(1), 11-20.

[30] Koju, T., Takada, S., and Doi, N. 2003. Regression Test Selection based on Intermediate Code for Virtual Machines. In Conference on Software Maintenance. Institute of Electrical and Electronics Engineers Inc., 420-429.

[31] Leung, H.K.N. and White, L. 1989. Insights into regression testing. In Conference on Software Maintenance. Publ by IEEE, Piscataway, NJ, USA, 60-69.

[32] Leung, H.K.N. and White, L. 1990. A study of integration testing and software regression at the integration level. In Proceedings. Conference on Software Maintenance 1990 (Cat. No.90CH2921-5). IEEE Comput. Soc. Press, 290-301.

[33] Mansour, N., Bahsoon, R., and Baradhi, G. 2001. Empirical comparison of regression test selection algorithms. Journal of Systems and Software. 57(1), 79-90.

[34] Mansour, N. and El-Fakih, K. 1997. Natural optimization algorithms for optimal regression testing. In Proceedings - IEEE Computer Society's International Computer Software & Applications Conference. IEEE, Los Alamitos, CA, USA, 511-514.

[35] Memon, A.M. 2004. Using tasks to automate regression testing of GUIs. In IASTED International Conference on Artificial Intelligence and Applications - AIA 2004. ACTA Press, 477-82.

[36] Orso, A., Harrold, M.J., Rosenblum, D., Rothermel, G., Soffa, M.L., and Do, H. 2001. Using component metacontent to support the regression testing of component-based software. In Proceedings IEEE International Conference on Software Maintenance. ICSM 2001. IEEE Comput. Soc, 716-25.

[37] Orso, A., Nanjuan, S., and Harrold, M.J. 2004. Scaling regression testing to large software systems. In Softw. Eng. Notes (USA). ACM, 241-51.

[38] Pasala, A. and Bhowmick, A. 2005. An approach for test suite selection to validate applications on deployment of COTS upgrades. In Proceedings - Asia-Pacific Software Engineering Conference, APSEC. IEEE Computer Society, Los Alamitos, CA 90720-1314, United States, 401-407.

[39] Pei, H., Xiaolin, L., Kung, D.C., Chih-Tung, H., Liang, L., Toyoshima, Y., and Chen, C. 1997. A technique for the selective revalidation of OO software. Journal of Software Maintenance: Research and Practice. 9(4), 217-33.

[40] Ren, X., Shah, F., Tip, F., Ryder, B.G., and Chesley, O. 2004. Chianti: A tool for change impact analysis of java programs. In 19th Annual ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA'04. Association for Computing Machinery, New York, NY 10036-5701, United States, 432-448.

[41] Rothermel, G. and Harrold, M.J. 1994. A framework for evaluating regression test selection techniques. In ICSE-16. 16th International Conference on Software Engineering (Cat. No.94CH3409-0). IEEE Comput. Soc. Press, 201-10.

[42] Rothermel, G. and Harrold, M.J. 1996. Analyzing regression test selection techniques. IEEE Transactions on Software Engineering. 22(8), 529-51.

[43] Rothermel, G. and Harrold, M.J. 1997. A safe, efficient regression test selection technique. ACM Transactions on Software Engineering and Methodology. 6(2), 173-210.

[44] Rothermel, G., Harrold, M.J., and Dedhia, J. 2000. Regression test selection for C++ software. Journal of Software Testing Verification and Reliability. 10(2), 77-109.

[45] Sajeev, A.S.M. and Wibowo, B. 2003. Regression test selection based on version changes of components. In Tenth Asia-Pacific Software Engineering Conference. IEEE Comput. Soc, 78-85.

[46] Skoglund, M. and Runeson, P. 2005. A case study of the class firewall regression test selection technique on a large scale distributed software system. In 2005 International Symposium on Empirical Software Engineering (IEEE Cat. No. 05EX1213). IEEE, 10 pp.

[47] Staples, M. and Niazi, M. 2007. Experiences using systematic review guidelines. The Journal of Systems &amp; Software. 80(9), 1425-37.

[48] Toshihiko, K., Shingo, T., and Norihisa, D. 2003. Regression test selection based on intermediate code for virtual machines. In Proceedings International Conference on Software Maintenance ICSM 2003. IEEE Comput. Soc, 420-9.

[49] White, L. and Abdullah, K. 1997. A firewall approach for the regression testing of object-oriented software. Software Quality Week

[50] White, L., Jaber, K., and Robinson, B. 2005. Utilization of extended firewall for object-oriented regression testing. In IEEE International Conference on Software Maintenance, ICSM. IEEE Computer Society, Los Alamitos, CA 90720-1314, United States, 695-698.

[51] White, L.J. and Leung, H.K.N. 1992. A firewall concept for both control-flow and data-flow in regression integration testing. In Conference on Software Maintenance 1992 (Cat.No.92CH3206-0). IEEE Comput. Soc. Press, 262-71.

[52] Willmor, D. and Embury, S.M. 2005. A safe regression test selection technique for database-driven applications. In Proceedings of the 21st IEEE International Conference on Software Maintenance. IEEE Comput. Soc, 421-30.

[53] Vokolos, F.I. and Frankl, P.G. 1997. Pythia: a regression test selection tool based on textual differencing. In Reliability, Quality and Safety of Software-Intensive Systems. IFIP TC5 WG5.4 3rd International Conference. Chapman &amp; Hall, 3-21.

[54] Vokolos, F.I. and Frankl, P.G. 1998. Empirical evaluation of the textual differencing regression testing technique. In Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272). IEEE Comput. Soc, 44-53.

[55] Wong, W.E., Horgan, J.R., London, S., and Agrawal, H. 1997. A study of effective regression testing in practice. In Proceedings. The Eighth International Symposium on Software Reliability Engineering (Cat. No.97TB100170). IEEE Comput. Soc, 264-74.

[56] Wong, W.E., Horgan, J.R., London, S., and Agrawal, H. 1997. Study of effective regression testing in practice. In Proceedings of the International Symposium on Software Reliability Engineering, ISSRE. IEEE Comp Soc, Los Alamitos, CA, USA, 264-274.

[57] Wu, Y., Chen, M.-H., and Kao, H.M. 1999. Regression testing on object-oriented programs. In Proceedings 10th International Symposium on Software Reliability Engineering (Cat. No.PR00443). IEEE Comput. Soc, 270-9.

[58] Yanping, C., Robert, L.P., and Sims, D.P. 2002. Specification-based regression test selection with risk analysis. Proceedings of the 2002 conference of the Centre for Advanced Studies on Collaborative research. IBM Press.

[59] Yih-Farn, C., Rosenblum, D.S., and Kiem-Phong, V. 1994. TESTTUBE: a system for selective regression testing. In ICSE-16. 16th International Conference on Software Engineering (Cat. No.94CH3409-0). IEEE Comput. Soc. Press, 211-20.

[60] Zheng, J., Robinson, B., Williams, L., and Smiley, K. 2005. An initial study of a lightweight process for change identification and regression test selection when source code is not available. In Proceedings - International Symposium on Software Reliability Engineering, ISSRE. IEEE Computer Society, Los Alamitos, CA 90720-1314, United States, 225-234.

[61] Zheng, J., Robinson, B., Williams, L., and Smiley, K. 2006. A lightweight process for change identification and regression test selection in using COTS components. In Proceedings - Fifth International Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems. Institute of Electrical and Electronics Engineers Computer Society, Piscataway, NJ 08855-1331, United States, 137-143.