Master Thesis
Software Engineering
Thesis no: MSE-2007:26
October 2007

A Systematic Review of Automated Test Data Generation Techniques

Shahid Mahmood

School of Engineering
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

This thesis is submitted to the School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Author:
Shahid Mahmood
Address:
Folkparksvägen 14: 23, 37 240,
Ronneby, Sweden.
E-mail:
Shahidmahmood07@yahoo.com

University advisor(s):
Dr. Richard Torkar
Assistant Professor
Department of Systems and Software Engineering
Blekinge Institute of Technology
School of Engineering
PO Box 520, SE – 372 25 Ronneby
Sweden

School of Engineering
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

Internet : www.bth.se/tek
Phone : +46 457 38 50 00
Fax : + 46 457 271 25

ABSTRACT

Automated Test Data Generation (ATDG) is an activity that in the course of software testing automatically generates test data for the software under test (SUT). It usually makes the testing more efficient and cost effective. Test Data Generation (TDG) is crucial for software testing because test data is one of the key factors for determining the quality of any software test during its execution. The multi-phased activity of ATDG involves various techniques for each of its phases. This research field is not new by any means, albeit lately new techniques have been devised and a gradual increase in the level of maturity has brought some diversified trends into it. To this end several ATDG techniques are available, but emerging trends in computing have raised the necessity to summarize and assess the current status of this area particularly for practitioners, future researchers and students. Further, analysis of the ATDG techniques becomes even more important when Miller et al. [4] highlight the hardship in general acceptance of these techniques. Under this scenario only a systematic review can address the issues because systematic reviews provide evaluation and interpretation of all available research relevant to a particular research question, topic area, or phenomenon of interest.

This thesis, by using a trustworthy, rigorous, and auditable methodology, provides a systematic review that is aimed at presenting a fair evaluation of research concerning ATDG techniques of the period 1997-2006. Moreover it also aims at identifying probable gaps in research about ATDG techniques of defined period so as to suggest the scope for further research. This systematic review is basically presented on the pattern of [5 and 8] and follows the techniques suggested by [1].The articles published in journals and conference proceedings during the defined period are of concern in this review. The motive behind this selection is quite logical in the sense that the techniques that are discussed in literature of this period might reflect their suitability for the prevailing software environment of today and are believed to fulfill the needs of foreseeable future. Furthermore only automated and/or semi-automated ATDG techniques have been chosen for consideration while leaving the manual techniques as they are out of the scope.

As a result of the preliminary study the review identifies ATDG techniques and relevant articles of the defined period whereas the detailed study evaluates and interprets all available research relevant to ATDG techniques. For interpretation and elaboration of the discovered ATDG techniques a novel approach called ‘Natural Clustering’ is introduced.

To accomplish the task of systematic review a comprehensive research method has been developed. Then on the practical implications of this research method important results have been gained. These results have been presented in statistical/numeric, diagrammatic, and descriptive forms. Additionally the thesis also introduces various criterions for classification of the discovered ATDG techniques and presents a comprehensive analysis of the results of these techniques. Some interesting facts have also been highlighted during the course of discussion. Finally, the discussion culminates with inferences and recommendations which emanate from this analysis. As the research work produced in the thesis is based on a rich amount of trustworthy information, therefore, it could also serve the purpose of being an up-to-date guide about ATDG techniques.

Key words: Systematic review, Automated test data generation techniques

ACKNOWLEDGEMENTS

All that I produced in here is because that almighty God has provided me with the ability, the resources, the opportunity, and His kind support. My every word of thanks and praise ultimately refers to Him.

I would like to thank especially my supervisor Dr. Richard Torkar, who not only guided me in this work at each and every crucial step but also provided support in all respects. His worthy criticism and appreciation of work helped me in transforming my fluid concepts in my thoughts into concrete tasks. I would like to gratefully acknowledge his contributions in enhancing my knowledge on the subject.

I am grateful to my examiner Dr. Robert Feldt for helping me in choosing the topic for the thesis and successfully graduating me from Blekinge Institute of Technology, Sweden.

I would like to express my gratitude to Dr. Robert Feldt, Dr. Richard Torkar, and Dr. Tony Gorschek who statically validated my search method that was developed for the systematic review.

I am also grateful to all my teachers whose efforts and guidance resulted in me having the knowledge to successfully conclude this thesis.

I would like to thank all my friends who contributed in this task through positive discourse and guidance. I would also like to convey my gratitude to my fellow Muzafar Khan and my brother Khalid Mahmood who spared their precious time to proof read the thesis.

My special thanks to my parents, brothers and sisters who significantly contributed in my upbringing and brought me to a stage where I can make my humble contributions to human knowledge. At the same time I can never forget the contributions of my wife and children who suffered a lot and sacrificed their wonderful time by sending me abroad for the noble cause of acquiring higher levels of knowledge. I am also grateful to all my relatives who supported me by looking after my family during my absence. In the absence of this important support it would have been extremely hard for me to devotedly concentrate on my education and this thesis.

CONTENTS

TABLE OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Preamble

Reviews of a topic area or phenomenon of interest are always in demand, specifically by researchers for the purpose of receiving assistance in their research work. A systematic review however, brings additional benefits as compared to a simple review because it provides evaluation and interpretation of all available research work relevant to a particular research question, topic area, or phenomenon of interest [1]. During the course of research for this thesis, preliminary literature study indicated that to the best of the author's knowledge, no systematic review has been published so far that could report on Automated Test Data Generation (ATDG) techniques. Therefore, for assisting and accelerating the research concerning ATDG (which is, a specialized area of automated testing), executing a systematic review was seen as an appropriate contribution to this research field.

Automated Test Data Generation is an activity that in the course of software testing automatically generates test data for the software under test (SUT). It makes software testing more efficient and cost effective [6]. ATDG is crucial for testing because test data is one of the key factors that determine the quality of a software test during its execution [6]. ATDG reduces testing time by an order of magnitude and presses the rising cost of test data generation downwards. However, at an initial stage, purchasing and installing automated tools may incur some extra cost but it might be better to accept a short-term penalty in the hope of long-term rewards.

The multi-phased activity of ATDG involves various techniques for each of its phases. This research field is not new by any means, albeit lately new techniques have been devised for its different phases, and a gradual increase in the level of maturity has brought some diversified trends into it. To this end several ATDG techniques are available, however emerging trends in computing have raised the necessity of summarizing and assessing the current status of this area particularly for practitioners, future researchers and students. Furthermore, analysis of the ATDG techniques becomes even more important when Miller et al. [4] highlight the hardship in general acceptance of these techniques. Under this scenario only a sound systematic review can address the concerning issues.

This thesis, by using a trustworthy, rigorous, and auditable methodology, provides a systematic review, which is aimed at presenting a fair evaluation of research concerning ATDG techniques of the period 1997-2006. Moreover it also aims at identifying probable gaps in the research concerning ATDG techniques during the defined period so as to suggest scope and directions for further research. This systematic review is basically presented on the pattern of [5 and 8] and follows the techniques suggested by [1].The articles published in journals and conference proceedings of the defined period are of concern in this review. The motive behind this selection is quite logical in the sense that the techniques that are discussed in literature of this period might reflect their suitability for the prevailing software environment of today and are believed to fulfill the needs of the future in sight. Furthermore only Automated and/or semi-Automated ATDG techniques were chosen for consideration while leaving the manual techniques out of the scope.

As a result of the preliminary study the review identified ATDG techniques and relevant articles of the defined period whereas the detailed study evaluated and interpreted all available research relevant to ATDG techniques. To describe the discovered ATDG techniques a novel technique called 'Natural Clustering' was introduced.

The primary task of this thesis is to present a systematic review of ATDG techniques. Additionally it also introduces modes of classification of these techniques and presents a comprehensive discussion on the findings. The discussion highlights interesting facts, deduces fruitful results and on the basis of these deductions puts forward some valid recommendations. Obviously, in the end, this thesis could also serve as an up-to-date guide about ATDG techniques.

Following the guidelines provided by Kitchenham [1], the remaining parts of this chapter (and some part of chapter 3) represent the planning phase of a systematic review. Chapter 2 establishes the background of the topic, whereas Chapter 3 which mainly deals in execution phase discusses in detail the research method adopted for conducting the systematic review. Chapter 4 being the core presents detailed information of the findings of the research method. It mainly concerns with reporting phase of the systematic review. Chapter 5 presents the classification of the ATDG techniques discovered during the systematic review. Chapter 6 presents the results by critically analyzing and discussing the findings. Chapter 7 presents the epilogue comprising elements like: Summary, Conclusion, Recommendations, and Future work. Important terminologies, list of references and appendices are given at the end.

An important aspect of this thesis is that it gives quality factor its due importance. At each and every step quality has been maintained by adopting proven methods. In this regard it takes guidance from already established relevant work, validation from experts, and addresses risk factors or validity threats at sensitive spots.

1.2 Purpose

The purpose of this thesis is to provide guidance to the intended users, about ATDG techniques discussed in literature during the decade of 1997-2006. Research on the work carried out during these years is most likely to cover all the ATDG techniques that are either currently in use by software industry or that might provide a base for development of such techniques in future.

1.3 Aims and Objectives

This thesis, by using a trustworthy, rigorous, and auditable methodology, provides a systematic review which is aimed at presenting a fair evaluation of research concerning ATDG techniques of the period 1997-2006. Moreover it also aims at identifying probable gaps in research about ATDG techniques of the defined period so as to suggest the scope for further research.

For achieving the abovementioned aims, following objectives are set:

a) Perform a systematic review of ATDG techniques on the lines of already proven methods/framework for such a review such as [1, 5].
b) A classification of ATDG techniques.
c) An analysis of the findings concluded from the systematic review and classification.
d) Point out areas of research and future work.

1.4 Research Questions

Research questions that need to be answered to meet the set aims and objectives for this project are given below:

1. Which ATDG techniques are discussed in literature during the period 1997-2006?
2. How can ATDG techniques be classified?
3. From the systematic review and classification of ATDG techniques, what can be deduced about general guidance and future work?

1.5 Thesis Methodology

Given in Figure 1.1 is a two-entity dataflow diagram (DFD) of level-1, which depicts the methodology of the thesis. This methodology includes the *data objects* (to represent the *objectives)*, various *processes* (to represent the *research questions)*, and *databases* (to represent the *outcomes)*. Further expansions of these processes are produced as a DFD of level-2 in the following chapters.

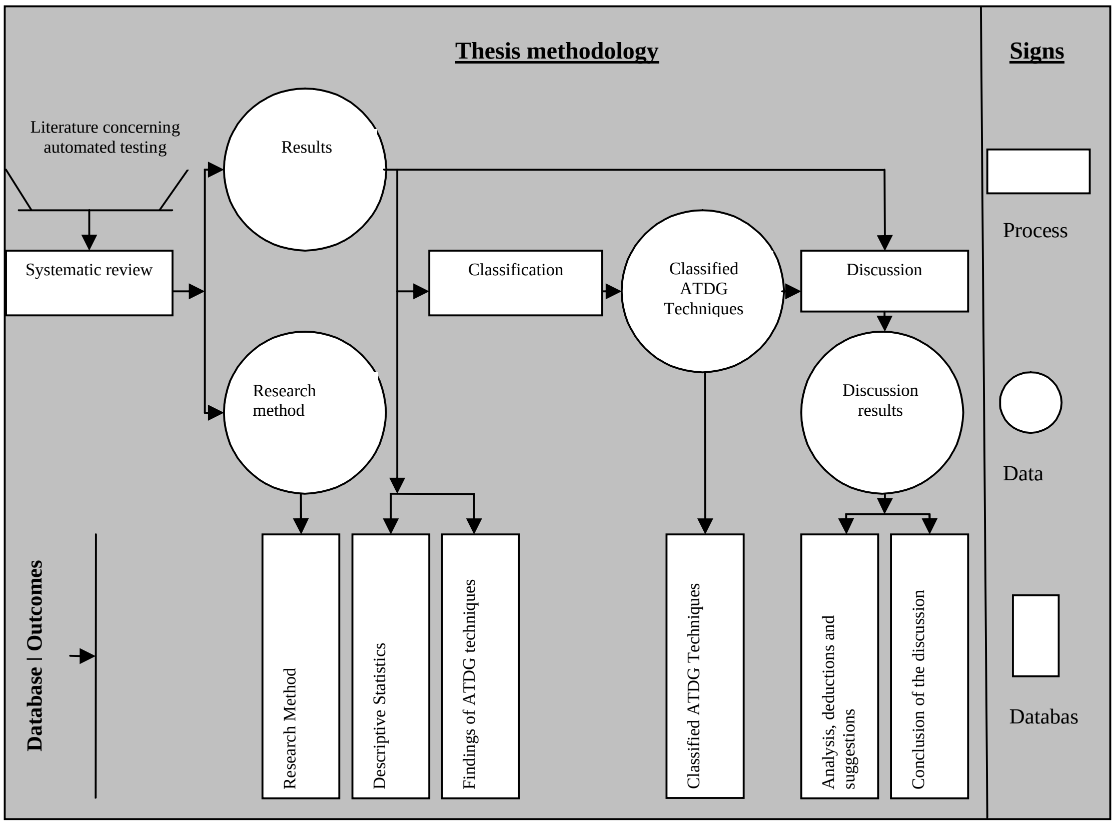

Thesis methodology
Signs
Literature concerning automated testing
Results
Systematic review
Classification
Classified ATDG Techniques
Discussion
Research method
Discussion results
Process
Data
Databas
Database | Outcomes
Research Method
Descriptive Statistics
Findings of ATDG techniques
Classified ATDG Techniques
Analysis, deductions and suggestions
Conclusion of the discussion

Figure 1.1 A two-entity Data Flow Diagram (DFD) of level-1, showing the thesis methodology

1.6 Intended Users

Generally this work provides knowledge and guidance to any person seeking in this field. However it particularly facilitates practitioners, future researchers and students in their research and investigation related to ‘automated test data generation’.

2 BACKGROUND

Unreliable software is considered to be prone to failures and hence, in general, carries little worth. Software testing which according to [7] is the "process of ensuring that a certain piece of software item fulfills its requirements" is one of the vital factors that can ensure reliability of the software. According to [12] assurance of software reliability partially depends on testing. However it is interesting to note that testing itself also needs to be reliable. Automating the testing process is a sound engineering approach, which can make the testing efficient, cost effective and reliable. Further mostly customers rely on software that is prepared according to some authenticated standards. Automization could promote standardization through standardised and patent test data generation tools.

Automated test data generation is one of the core factors that contribute towards automated testing. It is used for automatically generating test data for a SUT during software testing. Test data generation is a tedious, expensive and error prone process, if it is done manually [35]. Automation in the test data generation process could curtail testing expenses and at the same time, increase the reliability of testing as a whole. For these reasons automated test data generation has remained a topic of interest for the past four decades.

In [6]'s opinion ATDG is one of the most crucial activities of automated testing because test data is one of those key factors that determine the quality of a test during its execution. ATDG itself is a multi-phased process, which involves different techniques for every phase. For example program analysis, path generation and test data generation are the three basic phases of ATDG process that have been mentioned by [6] while explaining the architecture of a test data generator system. Similarly path-oriented, constraint-based, chaining, and assertion-based are some of the test data generation approaches that can generally be described as data generation techniques [7].

Developments in the field of 'automated test data generation' were initiated in early 70's. Articles like "Testing large software with automated software evaluation systems" by Ramamoorthy and Ho, in 1975 and "Automatic Generation of Floating-Point Test Data" by Miller and Spooner, in 1976, are a few examples of the early work in this field. Nevertheless, Clarke's paper of 1976 [10] is considered to be the first of its kind to produce a solid algorithm for ATDG. Hence this research field is not new by any means, albeit lately new techniques have been devised and a gradual increase in the level of maturity has brought some diversified trends into it. Therefore it seems necessary to summarize and assess the current status of this area for practitioners, future researchers and students. Additionally, analysis of the ATDG techniques becomes even more important when [4], while identifying limitations, highlights the hardship in general acceptance of these techniques.

Determining of an appropriate ATDG technique according to the SUT is yet another big issue for practitioners. Preliminary literature study indicated that so far no proper up-to-date guide about ATDG techniques is available that could provide some plausible guidance for this process. For providing a solution to all of the above stated issues a systematic review is likely to be an appropriate choice.

2.1 Basic Concepts

This thesis provides a systematic review that is aimed at presenting a fair evaluation of the research concerning ATDG techniques of the period 1997-2006. The review is based on a trustworthy, rigorous, and auditable methodology. Before conducting the systematic review and related tasks it seems appropriate to elaborate some of the very basic concepts of systematic review and test data generation.

2.1.1 Systematic Review

To quote Kitchenham [1], "A systematic review is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, topic area, or phenomenon of interest. Systematic reviews aim to present a fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology. Individual studies contributing to a systematic review are called ‘Primary’ studies; a systematic review is a form a Secondary study”. This definition by describing basic attributes of systematic reviews signifies their necessity during the process of research.

There could be at least three consecutive phases of a systematic review i.e., *planning*, *conducting*, and *reporting*. The planning phase identifies the need for a review and concerns the development of a review protocol. Conducting the review involves identification of research, selection of primary studies, study quality assessment, data extraction and monitoring, and finally data synthesis. Additionally, a systematic review could be reported in the form of formal reports or as a part of the thesis.

According to [1] systematic reviews require some extra efforts as compared to the conduct of simple reviews but, at the same time, they also offer several advantages over common reviews. Some of the distinct attributes of systematic reviews are:

- **a)** Systematic reviews start by defining a review protocol that specifies the research question being addressed and the methods that will be used to perform the review.
- **b)** Systematic reviews are based on a defined search strategy that aims at detecting as much of the relevant literature as possible.
- **c)** Systematic reviews document their research strategy so that readers can access its rigor and completeness.
- **d)** Systematic reviews require explicit inclusion and exclusion criteria to assess each potential primary study.
- **e)** Systematic reviews specify the information to be obtained from each primary study including quality criteria by which to evaluate each primary study.
- **f)** Systematic reviews are a pre-requisite for quantitative meta-analysis.

2.1.2 Test Data Generation

As a matter of normal practice, all new inventions are parsed through a variety of tests which are based on a well-established criterion developed over the years. Mostly this activity is generated to verify the correctness of the product against its specifications. Testing also determines the performance and quality of the product. This practice is also valid with respect to the development of software systems.

Generally software is tested for its structure or functions which are supposed to be performed by its components. Testing in the prior case is commonly known as structural or white box testing, while in the later case it is called functional or black box testing [46]. For either of the cases test data is required to “traverse” through the SUT. The outcome of this traversal determines correctness, performance, or in general, the quality of the SUT. In some cases the test data is already available or given but in most cases it is required to be generated. Test data generation is a complex activity because it involves so many steps and each step has several related issues. The architecture of a test data generator can better elaborate this notion of complexity. Figure 2.1 (initially drawn by [6]) shows the architecture of an automated test data generator using the path-oriented approach, in simple and easily understandable format.

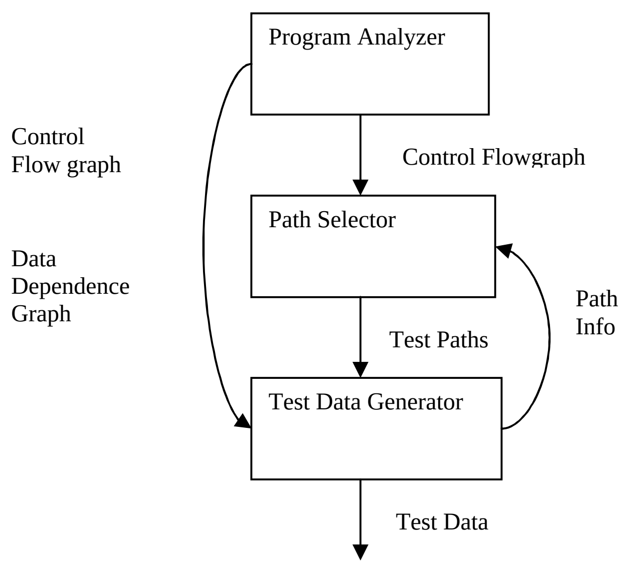


Figure 2.1 Architecture of a test data generator using the path-oriented approach [6]

As depicted in Figure 2.1 the program analyzer, through control flow graph or data dependence graph, analyses the SUT. At the next step the path selector identifies a set of paths to satisfy selected testing criterion and in the last step the test data generator generates test data accordingly. Important to note here is that test data generation is not a single step process rather it involves several steps. Adequate test data can never be created if performance at any of the steps is poor or the execution is incorrectly performed.

Under the two broad categories of testing there could be several types of testing. Accordingly there could be more than one possible test data generation criterion/approach or technique. Additionally, several different representations to distinguish ATDG techniques are possible, e.g. specification-based, code-based [18], random, static, and dynamic test data generation techniques [40]. To develop an initial level of understanding Figure 2.2 highlights the areas where an ATDG technique could belong. It is an interesting fact that functional testing and structural testing techniques complement each other, and are typically applied in different phases of the testing process [50].

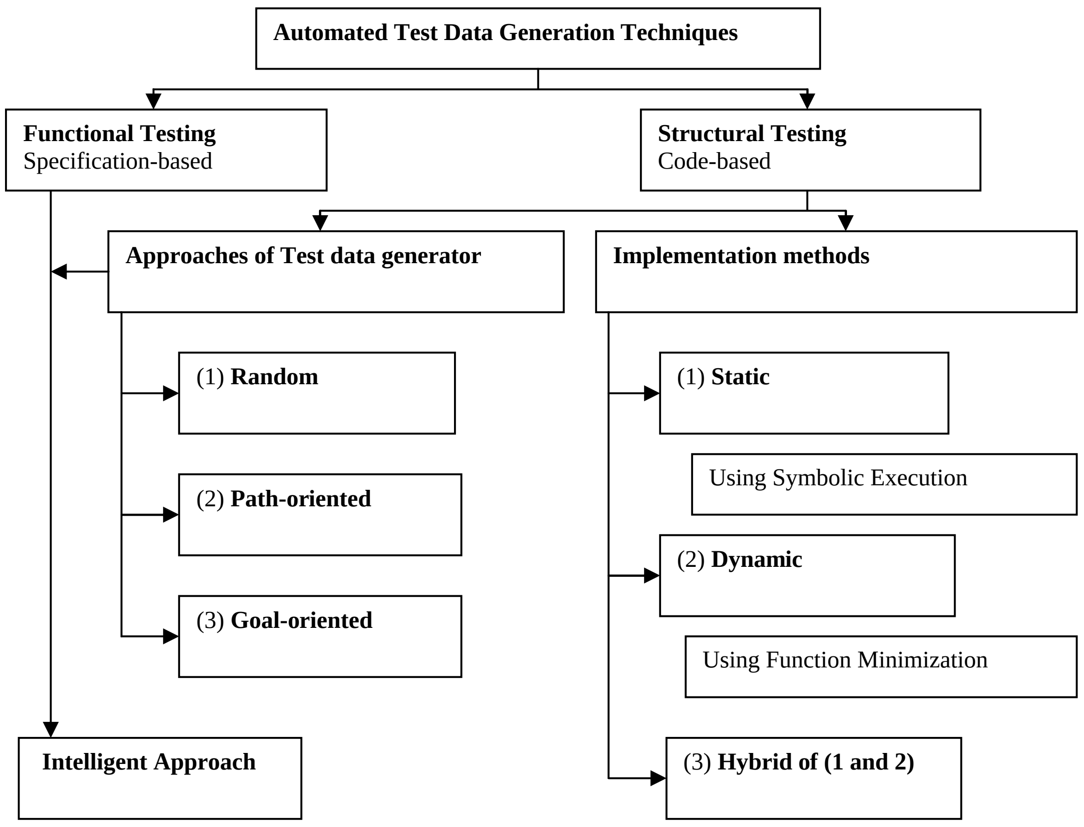


Figure 2.2 An overview of ATDG techniques

Figure 2.2 at an initial step shows the division of ATDG techniques into two parts, i.e. functional testing and structural testing. Further it highlights divisions on approaches and implementation methods. These divisions are further elaborated in the subsequent paragraphs.

Functional Testing Techniques These techniques are based on the functionality of the software, where test cases are derived from the program's specification. Examples of such techniques are equivalence partitioning, boundary value analysis, random testing and functional analysis-based testing [27].

Structural Testing techniques Structural ATDG techniques are based on the internal structure of a program, where test cases are selected so that structural components are covered (i.e. all paths are exercised in the SUT). Examples of such techniques are statement testing, branch testing, path testing, predicate testing, dataflow testing, structured testing, and domain testing [27]. Further structural techniques could be based on different approaches and implementation methods.

Random Approach of TDG According to [56] random test data generation simply consists of generating inputs at random until a useful input is found. This approach is quick and simple but might be a poor choice with complex programs and with complex adequacy criteria. The probability of selecting an adequate input by chance could be low in this case.

The biggest issue for random approach is that of adequate test data *selection*.

Path-oriented Approach of TDG In path-oriented test data generation the typical approach is generation of a control-flow graph. In this approach, at first a graph is generated first and subsequently, by using the graph a particular path is selected. With the help of a technique such as symbolic evaluation (in the *static* case otherwise it is called function minimization) test data is generated for that path in the end [13, 29]. In symbolic execution variables are used instead of actual values while traversing the path.

The path-oriented approach might face the problems when generating paths/graphs, traversing test data through branches and predicates (infeasible path problem), and while complexity of data types.

Goal-oriented Approach of TDG In the goal-oriented approach test-data is selected from the available pool of candidate test data to execute the selected goal, such as a statement, irrespective of the path taken [13]. This approach involves two basic steps: to identify a set of statements (respective branches) the covering of which implies covering the criterion; to generate input test data that execute every selected statement (respective branch) [30].

Two typical approaches, ‘Assertion-based’ and ‘Chaining approach’ are known as goal-oriented. In the first case assertions are inserted and then solved while in the second case data dependence analysis is carried out.

Generally the goal-oriented approach faces issues of goal selection and selection of adequate test data.

Intelligent Approach of TDG According to Pargas et al [13] the intelligent test-data generation approach often relies on sophisticated analysis of the code to guide the search for new test data. However in the author’s opinion this approach can be extended up to the intelligent analysis of program specification as well. With the proposed extended ability this approach will fall in between functional and structural testing. At this time this approach is quite limited in use (just reported by [13, 33, and 46]) therefore, its specialization in use or its pros and cons cannot be stated with any certainty.

Static Methods These are the testing methods adopted for analysis and checking of system representations such as the requirements documents, design diagrams and the software source code, either manually or automatically, without actually executing the software [25, 55]. In other words, the static methods do not require the software under test to be executed. They generally use symbolic execution to obtain constraints on input variables for the particular test criterion. Solutions to these constraints represent the test-data [46]. Executing a program symbolically means that instead of using actual values, ‘variable substitution’ is used.

Dynamic Methods Instead of using variable substitution, these methods execute the SUT with some, possibly randomly selected input [25]. By monitoring the program flow the system can determine whether the intended path was taken or not. In case of a negative answer the system backtracks to the node where the flow took the wrong direction. Using different kinds of search methods the flow can then be altered by manipulating the input in a way so that the intended branch is taken.

Hybrid Methods As the name suggests, a hybrid method is the mixture of static and dynamic methods. This mixture could be in the form of side-by-side use, probably solving some tasks by using static methods and some by using the dynamic methods, or it could be a synthesis of static and dynamic methods. Such an ATDG technique is reported by [14] where the symbolic execution is used for a dynamic approach.

2.2 Related Work

The preliminary study of literature indicates that no systematic review has been published concerning Automated Test Data Generation (ATDG) techniques. However, some work in the form of surveys about one particular aspect/area of ATDG techniques, or systematic reviews of a separate but related area is available. Comparison of the available work is possible with this systematic review on the bases of common factors. This section therefore, establishes relationship between these competing works. The purpose of this comparison is to analyze the quality of this systematic review and to make it prominent through contrast.

Juristo et al. [57] presented updates on the maturity level of different testing techniques. To analyze their studies they classified the derived testing techniques. During the process they were successful in obtaining a knowledge classification of testing techniques which was based on their factuality and objectivity according to four parameters.

This systematic review relates to the work of [57] in the sense that for classification of ATDG techniques it takes into consideration both the aspects of factuality and objectivity. The factuality based classification is performed according to the gathered data statistics, whereas objectivity based classification is achieved through segregation of parameters/properties.

In [6] Edvardsson presented a survey that is limited to program-based automated test data generation techniques. He particularly focused on the basic concepts and notions of test data generation and functioning of test data generator system. Additionally, he also highlighted other issues concerning the same topic.

While drawing a comparison with [6], it can be safely concluded that the current work presents quite similar basic concepts of test data generation. However, owing to the difference of goals between the two, the recent work only provides essential knowledge about basic concepts of test data generation while it mainly focuses on the ATDG techniques.

With a limited focus on search-based software test data generation McMinn in [53] presented a survey on the application of metaheuristic search techniques. His article mainly focuses on genetic algorithms and simulated annealing metaheuristic approaches while presenting their application to automate test data generation for structural, functional, and grey-box testing (Here the term ‘grey-box test data generation’ is used for those approaches which combine methods from the areas of structural and functional testing.).

This thesis is not really related to [53] with respect to the subject however details of metaheuristic search technique (as given in [53]) have been stated with same notion here, e.g. this thesis additionally describes the Tabu search.

Hannay et al. [5], does not seem to have any connection to this thesis however, similar pattern of presenting a systematic review has been adopted for this thesis. As a matter of fact [5] have adopted the research method of [8]; therefore it can safely be assumed that the roots of the idea of the research method adopted here are linked with that of [8].

2.3 Contribution of the Chapter

This chapter has contributed to the thesis by gradually building up the knowledge base of stakeholders. It has provided background of the topic thereby strengthened the base of the thesis. At first it has highlighted the concepts of testing, their necessity and the significance of ATDG in the overall testing process. After highlighting the historical background of the ATDG techniques it has built up a logical argument to emphasize upon the need for a systematic review of these techniques. For the sake of producing better understanding of the topic it presented a comprehensive discussion on the core concepts of systematic review and test data generation. Lastly to make this work prominent, related work has been presented by drawing its comprehensive comparison with the current work.

3 RESEARCH METHOD

Preamble: This chapter describes the research method adopted for identification of the articles reporting ATDG techniques. The chapter also describes extraction of ATDG techniques from those articles. With this work, the chapter denotes the primary critical step of a systematic review.

A research method provides foundation for a systematic review. Its quality directly influences the reliability of the systematic review. Therefore, for confirmation of this intuitive view, extra care and efforts have been put in to formulate the research method which makes it unlikely that a relevant paper or ATDG technique is overlooked. One of the main concerns for adopting a solid research method was the filtering from the huge pool of available information on test data generation techniques, only the automated and/or semi automated techniques.

The research method developed for this systematic review basically follows the pattern of [8] and the techniques suggested by [1]. Further as a proof of its soundness and comprehensiveness static validation of its components was received from Software Engineering researchers (whose names are mentioned in the acknowledgement section). It is believed that a research method that does not consider or mention the risk factor might not qualify to be truly authentic. Therefore, in this research method due consideration was given to the risk factor or validity threat at each and every step. The methodology of following already proven work and getting static validation have minimized the risk of mono track thinking or over looking any important source of information. Overall, in the author's opinion the design of the research method prevented any serious validity threats to the results.

This chapter has been organized in a way that initially it diagrammatically depicts the research method, which is followed by a description of the development mode and the research method itself. Subsequently, in Section 3.1 it discusses the steps and tasks relevant to the preparation for search under the title of pre-search criteria. Section 3.2 gives a detailed description of the research method. In that it discusses selection of search engines, journals and conference proceedings. It also describes the process of searching the required articles, and saving the search results. At the end it highlights about the contributions of this chapter in completing this thesis.

For better understanding, a schematic view of the development mode of the adopted research method is shown in Figure 3.1.

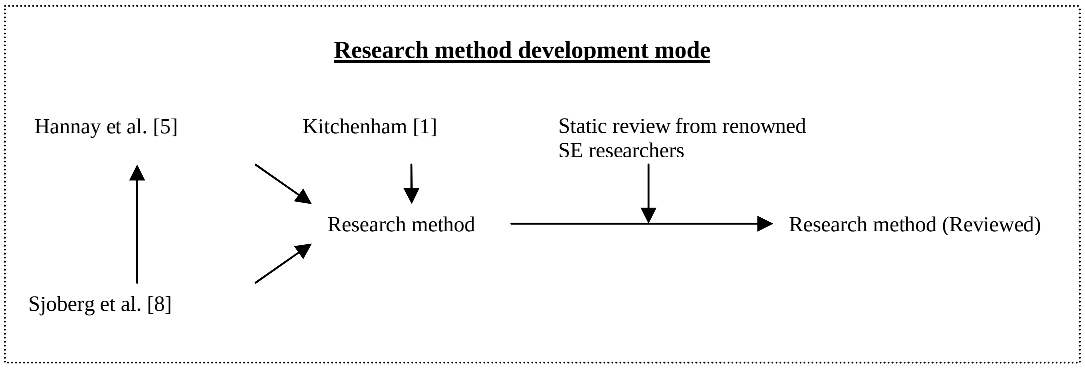
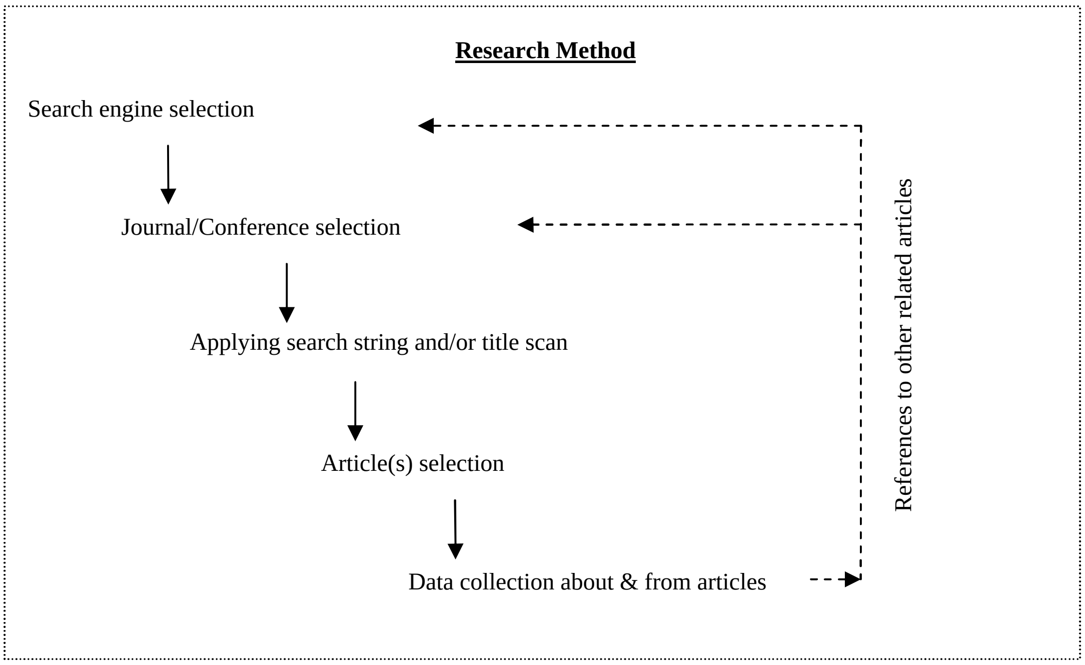

Research method development mode
Hannay et al. [5]
Kitchenham [1]
Static review from renowned SE researchers
Research method
Research method (Reviewed)
Sjoberg et al. [8]

Figure 3.1 Research method development mode

Graphical representation shows that the research method took a linear sequential form as depicted in Figure3.2. Detailed description of this research method is given in the subsequent section 3.2.



Research Method
Search engine selection
Journal/Conference selection
Applying search string and/or title scan
Article(s) selection
Data collection about & from articles
References to other related articles

Figure 3.2 Research method for the systematic review

3.1 Pre-search Criteria

For making a qualitative and comprehensive research on any topic, objectivity and scope of work should always be clearly defined and well understood. Therefore, before starting the search work, defining the terms "ATDG technique" and "test data" was considered essential. The two terms have been defined in a number of ways by different people, however, the following two definitions, available in the contemporary literature were considered to be most comprehensive and appropriate to define the fundamentals for the research work.

Test Data: Test data is the input needed for executing the test code, which has been written to test the 'software under test' (SUT).

Automated Test Data Generation (ATDG) Technique: It is a technique used for automatic test data generation for SUT.

The abovementioned two definitions are self-explanatory and believed to be comprehensive in nature. However for better understanding of the adopted selection criteria it may be noted that the "test data" is not the input that is required for testing software like UML based specification. Owing to the different and complex nature of their software architecture as compare to the conventional computer program, they require input e.g. high level specification/description of actual tests data is a case in point. Similarly, the term "ATDG technique" does not refer to a "fully automated" test data generation technique alone, but it also encompasses the "semi-automated" techniques.

To achieve maximum hit rate of the required articles it was decided that a broad quantitative base of well known and most likely journals and conference proceedings would be set up for search. Moreover only the relevant and scholarly search engines, prioritized on the bases of top most probability, were to be considered for the search. It was also decided that only the accomplished search strings/key words would be used for searching. For search string application 'From more specific to general' schema would be adopted because it was characterized to quickly highlight the relevant articles (see Appendix 2).

To minimize the risk of overlooking a relevant article/ATDG technique it was decided that journals and conference proceedings would be explored both ways i.e. by following the chronological sequence (from year-to-year) and also through the use of 'search string criteria'.

For eliminating the risk of loosing any searched data it was decided that a solid database along with data and count logs would be developed. Additionally, the searched/downloaded articles were also to be maintained systematically.

For extraction of ATDG techniques from the selected articles, a careful and thorough reading was to be carried out. During the process, die consideration was to be given to the context and references mentioned in the articles. It was also decided that a systematic way of noting the extracted techniques would be adopted that would subsequently help in classifying the techniques.

3.2 Detailed Description of the Research Method

This research method relies upon renowned and reliable search engines, databases, journals, and proceedings. Search in strong databases is expected to provide access to high quality of original work, research papers and authentic references. Therefore, exploration of the rich databases, checking of the lists of sources and ensuring that all relevant papers were covered was considered to be a good set of measures for validation of the research method.

This research method can be divided into three main parts. The first part pertains to the selection of search engines, journals and conference proceedings. The second part is about searching the required articles and there after bringing improvements in the search results whereas the third part elaborates storage criterion of the search results. Extraction of ATDG techniques is the final outcome of this research method which will be discussed in the next chapter.

3.2.1 Selection of Search Engines, Journals and Conference Proceedings

For starting the search and for achieving the required precise results in the minimum number of hits, choice of appropriate search engines was essentially required. With this objective, the above mentioned 'pre-search criterion for search engine selection' was applied and following a professional engineering approach the search engines were prioritized to set a search sequence. A prioritized list of all selected search engines with due motivation for each engine is given as Appendix 1. To support the selection criterion of search engines some general points of motivation are given below.

- These search engines are famous for providing scholarly search results therefore; they are expected to minimize the irrelevant search results.
- These search engines are focused and can lead straight to the required journals or conference proceedings and/or required articles.
- All these search engines provide advanced research facilities like purification of research up to several levels.
- Their selection is validated by comments from the renowned researcher.

For appropriate selection relevant journals and conference proceedings the above mentioned pre-search criterion was applied. For this purpose, knowledge gained from preliminary study in the specific domain of ATDG was applied and views of renowned SE researchers were also given due consideration. During the course of study it was observed that most of the references from articles consulted during preliminary search phase, were leading towards these journals and conference proceedings. Additionally, lists of top SE journals maintained by Professor Jeff Offutt of Software Engineering ISE department, George Mason University, and top SE conferences maintained by Sung Kim and Tao Xie of software engineering community were also considered for selection. Towards the end, cognitive perception was also utilized for choosing some of the names. Finally, ten renowned and most likely journals and six conferences were selected for conducting the search. As a ready reference, the final list of selected journals and conference proceedings is presented in Table 3.1.

Table 3.1 List of selected journals and conferences

Sr. No.	JOURNAL/CONFERENCE PROCEEDING
	JOURNALS
1	Software Testing, Verification and Reliability (STVR)
2	Journal of Systems and Software (JSS)
3	Automated Software Engineering Journal (ASEJ)
4	Information Sciences: An International Journal (ISJ)
5	Empirical Software Engineering Journal (EMSE)
6	ACM Transactions on Software Engineering Methodology (TOSEM)
7	IEEE Transactions on Reliability (TOR)
8	IEEE Transactions on Software Engineering (TSE)
9	Software Quality Journal (SQJ)
10	IEE/IEEE Software (ISW)
	CONFERENCE PROCEEDINGS
1	International Conference on Software Engineering (ICSE)
2	International Symposium on Software Testing and Analysis (ISSTA)
3	European Software Engineering Conference/ Foundation of Software Engineering (ESEC/FSE)
4	International Conference on Automated Software Engineering (ASE)
5	Fundamental Approaches to Software Engineering (FASE)
6	International Symposium on Software Reliability Engineering (ISSRE)

3.2.2 Searching the Required Articles

This sub-section describes the search criteria to identify and extract the required articles. Following a two step procedure, at first journals and conference proceedings were explored through year-to-year search, and also through the search string criteria. In the next step, the search results were improved on the basis of the information that had been gained during detailed scrutiny of the selected articles. As a result of the first step **48** articles were selected for detail scrutiny out of the total evaluations of **7848** articles. However, due to cogent reasons **8** of these articles were removed from the selected list. (A list of these articles along with detailed reasoning for their removal from the list is attached as Appendix-3). Finally, **40** articles were selected for further proceedings. These articles were only related to the ATDG techniques and pertained to the selected journals or conference proceedings for the period 1997-2006.

On the pattern of [8] editorials, prefaces, article summaries, interviews, news, reviews, correspondence, discussions, comments, reader's letters, summaries of tutorials, workshops, panels, poster sessions were excluded from the search. Those articles which only mentioned the names of some ATDG techniques without actually elaborating their operational capabilities or uses were also discarded. This measure is taken to ascertain the quality and reliability of the selected material thereby reducing the validity threat of creeping in any sub-standard data.

During the above mentioned course of action, scanning and scrutiny of the articles could not remained focused because the use of terminologies in this area was non-standard and confusing. For example, at places researchers used the terms 'test data generation' and 'test case generation', interchangeably. Similarly, at other places, 'test data' was denoted as 'test values' or 'test input' (these term are used for test cases as well). Likewise synonyms or parallels of the word 'technique' like: approach, method, procedure were also used. Therefore, to overcome this difficulty, the strategy of thorough reading to understand the semantics of confusing terms was adopted. The effort thus generated, promoted a cautious extraction of the required data about ATDG techniques from the evidences available in the articles.

The overall output of this research method is given in Table 3.2. In fact, the table gives the working view of the research method at a glance.

Table 3.2 Overall output of the research method

Purpose	**Scope**	**Journal/Proceedings**	**Sampling of articles**	**Investigated articles**
To search out articles of a specific time period, which reflect ATDG techniques	SE (ATDG tech.)	STVR, JSS, ASEJ, ISJ, EMSE, TOSEM, TOR, TSE, SQJ, ISW, ICSE, ISSTA, ESEC/FSE, ASE, FASE, ISSRE,	All articles in the period 1997-2006 pertaining to the selected journal and proceedings	7848articles scanned and 40 were considered for further proceeding

3.2.3 Extraction of ATDG Techniques

This section elaborates the approach chalked out for extraction of ATDG techniques. The findings of ATDG techniques, extracted by application of this method, are given in next chapter.

As explained earlier, extraction of ATDG techniques turned out to be an uphill because of the use of synonyms or parallel words for core terminologies by different writers. Mostly the ATDG techniques are described as a part of some big process like test case generation, test generation, or development of specific testing framework. In quite a few of the selected articles, a proper name was used for the described ATDG technique, where as in others only the name of ATDG technique with a short description of the technique was used. In some of the cases, a technique (possibly with different versions) was reported by several articles; conversely, there were several articles in which every single article reported many techniques.

To resolve the above mentioned complexities, a systematic approach of thorough reading with emphasis on semantics of the matter for extraction of the ATDG techniques was adopted. For identification of nameless ATDG techniques (given in these articles), a mention of only the references of the articles with category of technique was considered to be the more appropriate rather than giving pseudo names to these techniques.

3.2.4 Saving the Search Results

Since proper saving of the search results is as important as the search itself, a number of procedures were adopted to save the results according to their individual requirements and suitability of the procedure. These procedures were adopted while keeping in view the risks involved like: wrong storage of results, OR leaving some result without storage, OR duplication of result in the database.

A database was specifically designed for storing the search results. The data model of this database comprises of two related tables and a form. One table contains data about all the selected journals and conference proceedings, while the other contains data about articles picked up during the search. This database is patterned to store complete record of an article by giving it a storage number along with a cross reference. It prohibits duplicate entry of an article. The database also establishes preliminary categories of the articles and stores their brief descriptions of the portions which were discussed in selected articles. As a matter of rule, no article was saved without making an entry in the database. At the same time, the database was designed to facilitate the up coming tasks like: analysis, classification, and discussion.

Remembering exactly as to ‘what was discussed in which article’ was a cumbersome task. This issue was resolved by designing a detailed data log in such a manner that it could also serve the purpose of being a source for upcoming findings and analysis phases. Additionally, a comprehensive paper based ‘article count log’ was also maintained for counting the scanned articles. As a result of this effort, the article count log holds the complete ‘data count’ as a ready reference for any related query. Formats of these data logs and article count logs are given in Appendix 4.

3.3 Contribution of the Chapter

The adopted research method described in this chapter, contributed to the thesis in a number of ways. It sorted out the required material from a pool of miscellaneous information for further use and processing in the thesis. It also compiled all the work related to ATDG techniques of the decade 1997-2006 at a single place for ready reference. Further it also provided the knowledge and guidelines for conducting such a work in future. In this regard it gave a systematic way of description of a research method. In that at first it described the research method itself and presented a diagrammatic depiction of research method development mode. Later, in Section 3.1 it discussed the steps and preparatory tasks for the search as ‘pre-search criteria’. At the third step it provided a detail description of the research method in Section 3.2, where selection of search engines, journals and conference proceedings, searching of the required articles, issues of extraction of ATDG techniques along with criteria of extraction, and saving the search results were discussed. An important aspect of the contributions of this chapter is that it provides a foundation for the next chapter on ‘Findings of the systematic review’. In fact this chapter is an imperative and complementary part of the next chapter. Keeping in view the importance and sensitivity of the chapter consideration of risk factors or validity threats at each and every step is given due importance and proper measures are taken to encounter them.

4 RESULTS

Preamble: Information is commonly defined as "a collection of facts/data from which valid conclusions may be drawn". In line with this definition, this chapter provides information about the metadata and the matter inside the articles concerning ATDG techniques that has been collected by following a pre-defined research methodology. In general it offers quantitative meta-analysis of the research about ATDG techniques. In the thesis this chapter is of prime importance because it provides basic matter for the next two chapters which discuss the knowledge gained from the information provided here. The knowledge presented in this chapter is basically a blend of contextual information, framed experience, values, and intuition.

According to [1], a systematic review is a form a secondary study. This chapter is indeed based on the secondary study of the selected articles. Moreover this chapter evaluates and interprets all available research about ATDG techniques of the defined period, which according to [1] is one of the basic functions of systematic review. At an abstract level this chapter is divided into two main sections i.e. 'Descriptive Statistics' and 'Findings of ATDG techniques'. The concluding section at the end elaborates contribution of the chapter in overall work of the thesis.

For achieving a better understanding of the chapter and to prevent the risk of misinterpretation of a term, it is recommended that the readers, (especially the naïve users) should first read the terminology part of the thesis first and then acquaint themselves with rest of the technical matter produced in the thesis.

Validity Threats Associated with Results:

The points given below depict the probable threats and preventive measures associated with the results part of the systematic review.

a) Research results in a systematic review heavily depend on the quality of the research method that is adopted.

Preventive Measure: A properly defined research method based on quality measures (like the one adopted here) might prevent the threat to quality.

b) The depicted results might not be coming from a properly maintained and reliable database.

Preventive Measure: this threat could be prevented by maintaining a database that fulfills necessary quality measures, as in the current case.

c) On-spot analysis, discussion and on the basis of that, recommendations, might be baseless and misleading.

Preventive Measure: The analysis, discussion and recommendations based on statistically produced results that could be verified through cross check, might be a good preventive measure to such threats and therefore adopted here.

This chapter, along with the previous chapter (on research methodology); collectively contribute to the process denoted as 'systematic review' in Figure1.1 of chapter 1.Expansion of this process at level 2 of the DFD is depicted below.

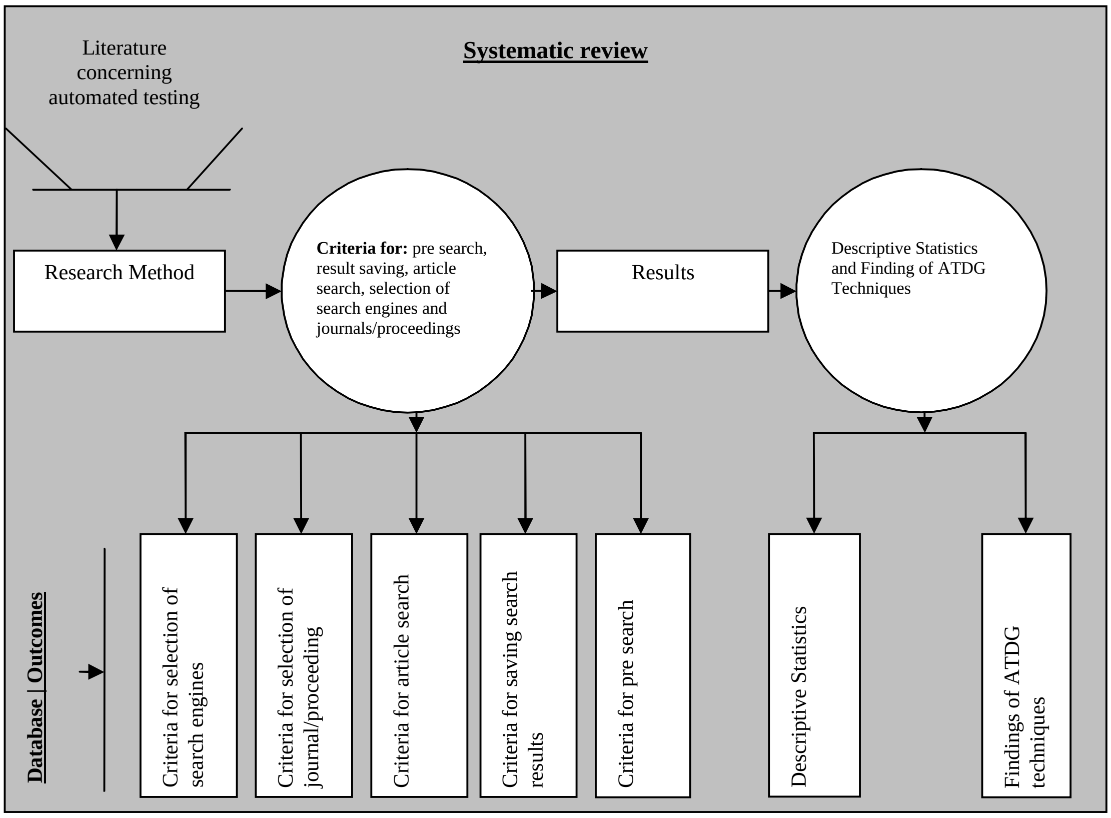

Literature concerning automated testing
Systematic review
Research Method
Criteria for: pre search, result saving, article search, selection of search engines and journals/proceedings
Results
Descriptive Statistics and Finding of ATDG Techniques
Database | Outcomes
Criteria for selection of search engines
Criteria for selection of journal/proceeding
Criteria for article search
Criteria for saving search results
Criteria for pre search
Descriptive Statistics
Findings of ATDG techniques

Figure 4.1 Two-Entity DFD of level-2, showing Systematic review process.

4.1 Descriptive Statistics

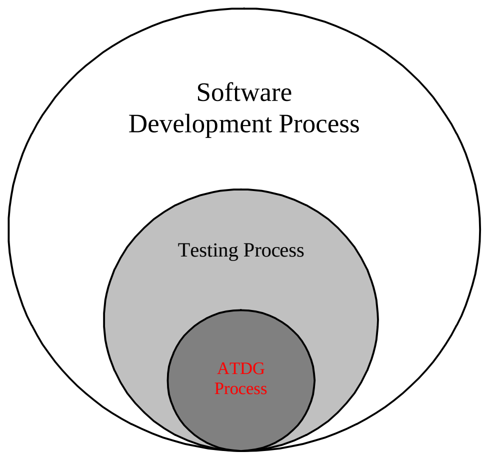
This section describes different aspects of quantitative results, achieved through thorough study of the selected articles, which could be of interest from various perspectives. All the data gathered during the study of the articles is produced here in different formats so that the research work is presentable in different perspectives. This measure might help in reducing the risk of depicting wrong results and improving the quality of the results produced through cross check of their different formats.

4.1.1 Statistics of Journals, Conference Proceedings, and Articles

It has already been mentioned in the previous chapter that out of the total of 7848 scanned articles, 40 were considered suitable for further proceeding. A statistical data of the selected journals and conference proceedings along with distribution of articles about ATDG techniques of the period 1997-2006 in different formats is presented in Table 4.1. Further to strengthen the understanding of these outcomes figure 4.2 shows approximate proportion of ATDG in the overall process of software development as well as association of the articles that are scanned or found directly related to ATDG techniques.

Table 4.1Distribution of Articles Describing ATDG Techniques of the period1997-2006.

Journal/Conference Proceeding	**Articles investigated**	**Articles reported ATDG tech.**	
		N	***%***
JOURNALS/CONFERENCE PROCEEDINGS			
ASE	467	13	32.5
STVR	97	8	20.0
ISSTA	115	7	17.5
SQJ	165	3	7.5
ESEC/FSE	117	3	7.5
JSS	1099	2	5.0
TSE	744	2	5.0
ISJ	1374	1	2.5
ISSRE	271	1	2.5
ASEJ	145	-	-
ICSE	1030	-	-
EMSE	191	-	-
TOSEM	125	-	-
TOR	629	-	-
ISW	1060	-	-
FASE	218	-	-
Total	**7848**	**40**	**100**



Total article scanned related to software development process = **7848**

Articles found related to ATDG techniques = **40**

Figure 4.2: Venn diagram showing approximate proportion of ATDG in the overall process of software development.

Table 4.1 indicates that the overall research volume in the field of ATDG techniques remained considerably low during the selected decade i.e. 1997-2006. The overall percentage of related articles that were found is just 0.5%, which indeed is a very low figure. Probable reason of this low percentage could be that:

a) ATDG is a highly specialized and specific area of testing (representing a very small proportion in the overall software development process); therefore, many of the researchers refrained from venturing in this domain. Figure 4.2 further clarifies and supports this argument.

b) There was probably less demand for research during the above mentioned period.

c) This field has little attraction for researchers.

d) Adequate knowledge had already been gained in this field which was sufficient for fulfilling the requirements of that period.

Clustering of the articles in some of the selected journals and conference proceedings may indicate appropriateness/specialization of that journal or conference proceeding in that area.

An analysis of the collected data indicates that the research volume or the trend of publishing the research work was not evenly distributed over the years of the defined period. In fact it showed a lot of variation, as shown in the line graph below. In this graph, year-wise publication of articles (on ATDG techniques) vis-à-vis calendar year is shown. It serves the purpose to analyze the publishing trends in different years.

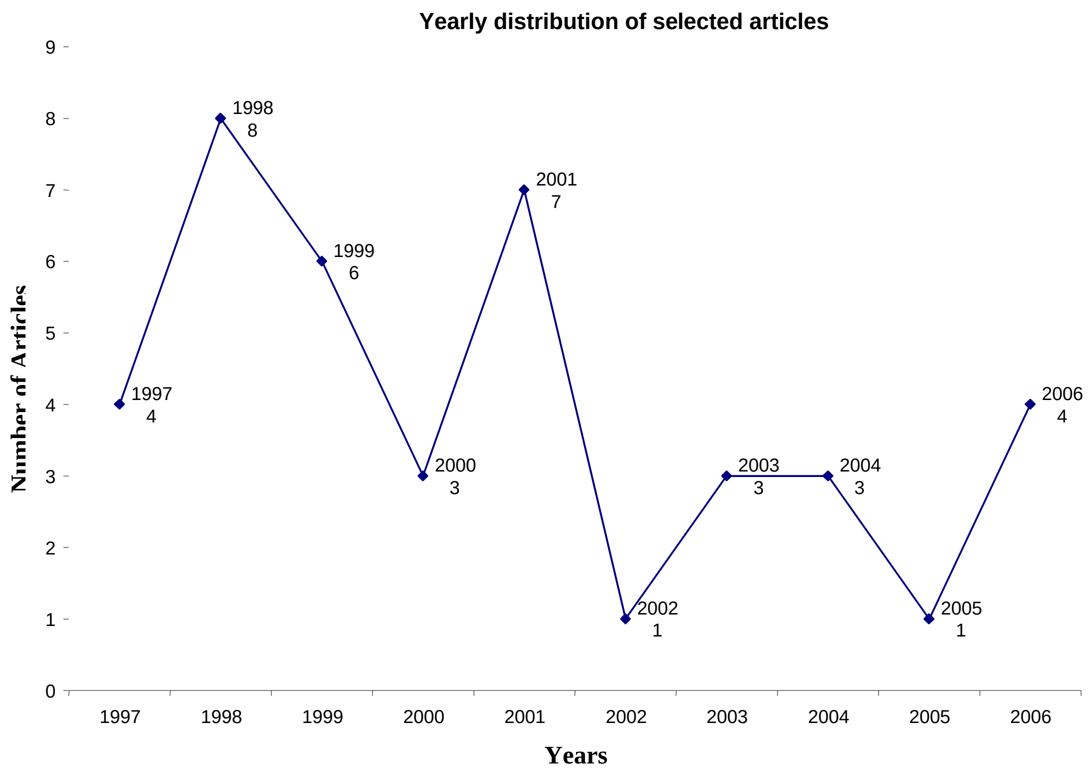


Figure 4.3 Line graph representation of distribution of selected articles on ATDG techniques published in selected journals and conference proceedings of the decade 1997-2006.

When analyzing the graph in Figure 4.2 it becomes evident that there was a considerable recession in the trend of publishing articles/research papers, during the years 2002 to 2005. During these years only 1 to 2 publications were made available. Similarly, only three publications were traceable in the 2003-04. However, year 2001 showed a considerable rise with 7 articles published. It is interesting to note that the year 2006, which is the last year of the chosen decade, showed an increasing trend in the number of publications. This rising trend is maintained in the current year (year-2007). The rising trend could be interpreted as an indication of the growing need of research in the field of ATDG and also as the rising interest of the researcher’s in this field.

4.1.2 Descriptive Statistics Regarding Areas of Research

Thorough study of the articles revealed several interesting facts about the research trends of the selected period and other aspects of ATDG techniques. A systematic arrangement of these facts and trends is presented in Table 4.2, from where some statistical results could be drawn.

Table 4.2 Research statistics and trends in articles on ATDG techniques of the decade 1997-2006.

Sr. No.	Article Ref. No.	Author (s)	Article Pub Year	Testing category			Approach			Implementation method		
				BBT	WBT	MixT	Random	Path Orient	Goal Orient	Static	Dynamic	Hybrid
1	11	Offutt and Pan	**1997**		x			x		x		
2	12	Hoffman et al.	**1999**	x			x				x	
3	13	Pargas et al.	**1999**		x				x		x	
4	14	Christophe	**2001**		x			x			x	
5	15	Botella et al.	**2006**		x			x		x	x	
6	16	Alshraideh et al.	**2006**		x			x			x	
7	17	Gallagher et al.	**2006**		x			x			x	
8	18	Offutt et al.	**1999**	x			x				x	
9	19	Jeng and Forgacs	**1999**	x			x					x
10	20	Lin and Yeh	**2001**		x				x		x	
11	21	Nquyen and Deville	**2001**		x			x			x	
12	22	Khor and Grogono	**2004**		x			x			x	
13	23	Gallagher and Narasimhan	**1997**	x				x			x	
14	24	Michael et al.	**2001**	x			x				x	
15	25	Chu et al.	**1997**	x			x					x
16	26	Claudia and Regina	**2003**		x			x			x	
17	27	Mansour et al.	**2004**		x			x			x	
18	28	Yang et al.	**1998**		x			x		x		
19	29	Hajnal and Forgacs	**1998**			x		x				x
20	30	Gotlieb et al.	**1998**		x				x			x
21	31	Nguyen and Deville	**2003**		x			x			x	
22	32	Michael et al.	**1997**	x			x				x	
23	33	Michael and McGraw	**1998**	x			x				x	
24	34	Tracey et al.	**1998**			x	x				x	
25	35	McMinn et al.	**2006**		x				x			x
26	36	Gupta et al.	**1999**		x			x			x	
27	37	Gupta et al.	**2000**		x			x			x	
28	38	Gouraud et al.	**2001**		x		x	x			x	
29	39	Visvanathan and Gupta	**2002**		x			x			x	
30	40	Díaz et al.	**2003**		x				x			x
31	41	Edwards	**2001**	x				x			x	
32	42	Gupta et al.	**1998**		x		x				x	
33	43	Edvardsson and Kamkar	**2001**		x			x				x
34	44	Baresel et al.	**2004**		x				x		x	
35	45	Korel and Al-Yami	**1998**		x				x		x	
36	46	Tracey et al.	**1998**	x	x			x		x	x	
37	47	Bueno and Jino	**2000**		x			x			x	
38	48	Liu et al.	**2005**			x	x				x	
39	49	Bousquet and Zuanon	**1999**	x			x				x	
40	54	Taylor and Cukic	**2000**		x			x			x	
				11	**27**	**3**	**13**	**22**	**7**	**4**	**31**	**7**

The statistics produced in the Table 4.2 are outcome of a thorough study of the selected articles in which a variety of evaluation criterion was used. Therefore, they provide very rich information about articles and the ATDG techniques involved there in. One can extract variety of facts from these statistics. Aggregates of each column, as are evident from the bottom row, indicate several other facts/trends in the research of the selected period. Other statistics of interest which Table 4.2 highlights could be elaborated as follows:

a) None of the year from selected decade is without a publication, which shows a continuous research trend in the area of ATDG techniques. However the volume of research or quantity of publications differs from year to year (as already been shown in Figure 4.2 and 4.3).

b) Most of the articles indicate the research as a collective effort of two or more researchers. This could be interpreted as an indication of the high degree of difficulty of the research area.

c) All, the Black box testing (BBT), the White box testing (WBT), and mixed (BBT + WBT) categories of testing were addressed by researchers, which shows versatility of research. However, the research trend towards these three categories varies i.e. highest towards WBT, followed by BBT and the least towards mixed testing approach. These different trends may indicate several aspects like: need for research in that period, market trend of ATDG tools, and researchers approach or interest.

d) Statistics indicate that ATDG techniques of selected decade were developed by adopting all the three testing approaches i.e. Random, Path oriented and Goal oriented however, highest research trend was towards the path oriented approach. The random and goal oriented approaches have shown almost the same research trends.

e) Out of the implementation methods the dynamic method was the most popular, the hybrid method occupied the second place in popularity, whereas static method was the least popular. This could be interpreted as the rising trend towards full automation of the testing process. The static method is mostly common in use in manual or semi-automated test data generation approaches.

f) The statistics have highlighted implicit categorization of articles reporting ATDG techniques. This categorization could be carried out on the basis of testing methods, the adopted approach, or the implementation method.

To fulfill further requirements, possibly much more information could have been drawn from the produced statistics. In this context it is a vital contribution for further research in this field.

4.2 Findings of ATDG Techniques

Techniques represent a partial or complete solution of certain issues. With this consideration, some questions regarding ATDG techniques must be addressed. In this regard some of these questions are:

What are the issues concerning automated test data generation?

Which techniques provide their solution and in what way?

How different techniques can be described?

Which articles provided what technique(s)?

Answers to these questions are provided in this section. In fact all these questions are implicit to one of the primary research questions of this thesis, which is: “Which ATDG techniques are discussed in the literature of the period 1997-2006”.

Answering the above mentioned questions was an arduous task but fortunately several feasible solutions were available for consideration and adoption.

Examples are:

a) Describing ATDG techniques by addressing respective articles in a serial; OR

b) Putting up various groups against types of automated test data generation; OR

c) Against the issues they addressed.

However the approach introduced and considered most appropriate to highlight the extracted ATDG techniques, was ‘Natural clustering’. (Appendix-5 provides some of the **l**imitations of approaches that could be used to describe ATDG techniques)

Natural Clustering: Natural clustering is an approach that highlights the clustering of techniques, which is formed due to common attributes among them. It is fully capable of depicting, discussing and comparing similar techniques with ease in an easily understandable format. Natural Clustering is flexible in defining and maintaining the clusters. The clusters and the respective sub-clusters are maintained here in hierarchical format. It carries good properties of other competing solutions as well.

During the course of in-depth study and subsequent collection of statistical data, certain behavioral trends of ATDG techniques were observed. Amongst these, an important trend, suitable for highlighting the techniques, was 'Natural clustering' (as the name is given to the approach representing this trend). Natural clustering indicates that ATDG techniques naturally tend to cluster against certain issue, approach, or method the objective being highlighting these clusters. Natural clustering achieves this objective by maintaining the abstraction level of the clusters according to requirement. This characteristic makes this approach flexible. Once an abstraction level for the clusters is set, the naturally formed sub-clusters within the set of clusters can be traced out easily. In case of ATDG techniques the abstraction level of a cluster can be set in various types of selections e.g., according to issue, testing type. But the flexibility property of Natural clustering allows sub-clusters on the basis of matching properties of ATDG techniques within a cluster. A typical example of such a phenomenon is a cluster formed on the bases of test data search issue that contains sub-clusters, which represent ATDG techniques adopting genetic algorithm approach, or simulated annealing. It is the glory of natural clustering that it allows an ATDG technique to show up in more than one cluster according to its properties, leaving almost no chance for a property to remains hidden. In natural clustering clusters and sub-cluster of ATDG techniques were maintained in hierarchy, such that one big cluster would contain sub-clusters up to several levels in depth. This hierarchy of clusters is shown in Figure 4.4.

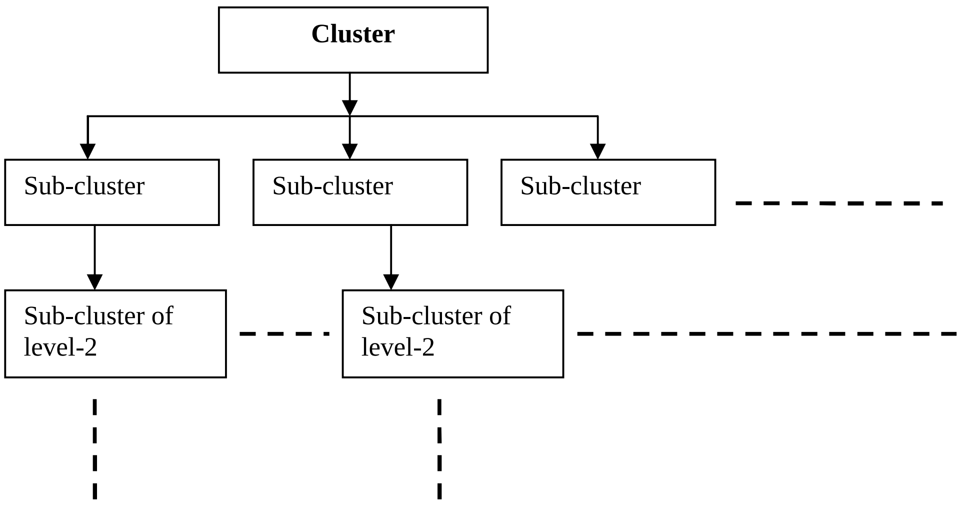


Figure 4.4 Hierarchy of Natural clustering of ATDG techniques

Figure 4.4 depicts that a cluster could contain many sub-clusters, further each sub-cluster could also contain sub-clusters of level-2 and so on. This hierarchy could go up to several levels in depth. The purpose of maintaining this hierarchy of clusters and sub-clusters is to depict, discuss, compare and establish links between the extracted ATDG techniques.

It is observed that ATDG techniques reflected by the selected literature are either novel or a complementary part or extension of some previously defined technique. Moreover some novel ATDG techniques were developed by combining two or more approaches or implementation methods or even testing criterion. These factors have made description of ATDG techniques even more difficult. However, 'natural clustering' is fully capable of accommodating such complex and difficult conditions; therefore, this section provides a systematic presentation of 'natural clustering'.

It would be worth mentioning here that the objective is not to classify the ATDG techniques (although it can be used for classification) but it is to highlighting or elaborate these techniques, therefore, clusters at the most abstract or prominent level are entertained. In the adopted criteria cluster at the most abstract level are denoted as 'cluster'.

According to [29 and 47] for structural testing cases, "Test data generation consists of two main steps: 1- select a set of paths that should be executed and 2- find test data to cover the above set of paths". This statement is further strengthened by another statement from [6] according to which "The selection of a path can largely affect the whole process of test data generation." In the light of these two statements techniques providing solution of any of these test data generation steps are considered for elaboration. ATDG techniques reported by the selected articles could be more specific, dealing just one or two parts of an issue (e.g., [48]), or they could be comprehensive encompassing the whole ATDG process (e.g., [17 and 38]). In either case, for highlighting purpose the techniques are not considered to be comprehensive or complete, until and unless the reporting article explicitly make a mention of it, like for example [46] has reported two or more separate techniques to do so.

Table 4.3 gives statistics about some of the naturally formed clusters of various sizes. Subsequently, these statistics have been used for elaboration of techniques. It may be interesting to note that some clusters are formed on the basis of strong natural bonding against certain common issues between sub-clusters.

Table 4.3 Statistics about some of the naturally formed clusters of ATDG techniques.

Sr. No	Property	Article ref	Total Articles
1	Path analysis, Branch, Statement, predicate coverage	11, 14, 15, 16, 17, 22, 28, 29, 35, 40, 44, 47	**12**
2	Graph-based approach	13, 17, 20, 27, 28, 30, 31, 36, 37, 38, 40, 41, 42	**13**
3	Path constraint solution	20, 21, 23, 27, 31, 36, 37, 38, 39, 44, 43, 46, 54	**13**
4	Function minimization	19, 24, 26, 29, 32, 33, 45	**7**
5	Domain testing, Boundary values analysis	12, 19, 23, 29, 34	**5**
6	Mathematical approach, Numerical optimization	12, 15, 17, 23, 36, 37, 42, 43	**8**
7	Heuristic/Evolutionary approach-based	11, 13, 16, 20, 22, 24, 26, 27, 32, 33, 34, 35, 38, 40, 46, 47, 48	**17**
8	Specification based, model based techniques	18, 46, 49	**3**
9	Statistics based testing	25, 38, 54	**3**
10	Combinatorial, Optimization-based	19, 29, 30, 31, 36, 41, 44	**7**
11	Component re-use testing	34, 41	**2**
12	Programming-based	14, 15, 17, 18, 23, 24, 28, 29, 32, 33	**10**

4.2.1 ATDG Techniques for Path Analysis/Selection, Path Constraint, Function Minimization, Branch and Statement Coverage

This cluster of ATDG techniques addresses the issue related to feasible path analysis or infeasible path detection, branch or statement coverage, and path constraints. This cluster particularly covers the two steps i.e. path drawing/selection for execution, and finding test data to cover the abovementioned set of paths. The cluster also mentions the techniques which use BBT but addresses path, constraint, or branch issues. Details of such techniques are given in the respective clusters. Articles [11, 14, 15, 16, 17, 19, 20, 21, 22, 23, 24, , 27, 28, 29, 30, 31, 32, 33, 35, 36, 37, 38, 39, 40, 42, 43, 44, 45, 47, and 54] have reported ATDG techniques belonging to this cluster. With the accumulation of 28 articles, this is one of the biggest clusters.

One of the **sub-clusters** here contains ATDG techniques related to the issues of path constraints. A total of 13 articles i.e. [20, 21, 23, 27, 31, 36, 37, 38, 39, 44, 43, 46, and 54] have reported ATDG techniques providing path constraint solutions. Generally in path constraint approach, constraints are defined on values of possible test data against certain path. These constraints are then solved one-by-one (in this case automatically) to detect and spare infeasible paths.

Articles [36, 37, 42, 43, and 54] have addressed issues of linear constraints thereby representing a **sub-cluster of level 2**. In their article [42] Gupta et al. have presented a novel program execution based approach to automatically generate test data. Their approach is based on traditional relaxation technique used in numerical analysis. They adopted an advance approach in another article [37], which can handle branch predicates that compute general nonlinear functions of the input. Their new algorithm identifies infeasible linear paths and likely infeasible paths and removes them from the set of paths being explored. In [36] Gupta et al. presented Unified Numerical Approach (UNA) iterative relaxation technique to generate test data. The importance of this technique is that it can be used to systematically compute a 'Real', 'Integer' and a mix of 'Real-and-integer' feasible solutions for a set of linear constraints comprising inequalities and equalities. In contrast to UNA Edvardsson and Kamkar [43] introduced the simplex method. Simplex method in combination with branch-and-bound and/or a cutting-plane algorithm solves constraints. In [54] Taylor and Cukic used framework approach based on statistical regression analysis for test data generation by solving linear constraints. Their approach is for software systems with closed-loop process controller that use data trajectories as inputs.

In article [20] Lin and Yeh have reported an approach where they apply genetic algorithm to path testing. In this approach, at first the target paths are clearly defined and then applied fitness function named, 'SIMILARITY' to achieve the goal of the selected path. The approach extends the Hamming distance from the first-order to the *n*th order (n>1) to measure the distance between two paths. These are the related issues for ATDG. Nquyen and Deville [21] have also presented a novel approach, that is based on consistency techniques for ATDG of imperative programs containing integer, Boolean and/or float variables. Before path generation this approach at first converts the original program into static single assignment (SSA) form. After dynamic generation of path and constraints, the technique solves these constraints by using interval-based constraint solving algorithm. For reducing the domain of variables without removing solutions the approach uses eBox consistency. Gallagher and Narasimhan in their article [23] adopted a unique BBT approach providing path constraint solution. They used program instrumentation to solve a set of path constraints without explicitly knowing their form hence avoiding problems associated with symbolic execution approach. Article [27] by Mansour et al. reported path testing through simulated annealing and genetic algorithm. In [30] Gotlieb et al. reported a program execution based approach for test data generation. It detects infeasible paths by constructing a set of global constraints for the program and uses constraint solving techniques. The global constraints are generated for the whole program even if a single node is to be exercised. In [31] Nguyen and Deville presented a novel consistency-based approach for automated test data generation of imperative programs containing integer, boolean and/or float variables. In this technique they introduced Interprocedural Control Flow Graph (ICFG) for Interprocedural test data generation. The ICFG transforms the specified path into a path constraint which is then solved. Article [38] by Gouraud et al. has reported ATDG technique that encompasses the whole process of test data generation i.e. from data selection and path generation to constraints solving. In this way it covers all the three sub- clusters. They extended the work of Fosse and Waeyslynck 1991 "An investigation of software statistical testing" with the main difference that they avoid the construction of a distribution on the input domain. In their approach the authors attempted to eradicate problems of random test data generation through automating statistical structural testing. Their approach is based on the combination of uniform generation of combinatorial structure and the randomized constraint resolution techniques. In this approach control flow graph is formalized as a combinatorial structure specification and then the path predicates are solved using powerful constraint solver. The constraints introduction and management processes are done through constraint logic programming where the constraints are awakened through heuristics. Test data is generated by resolution of the predicates characterizing the input causing the execution of a given control flow path. In the article [39] Visvanathan and Gupta have introduced an ATDG technique which operates for functions with pointer inputs in two domains i.e. the address domain and the value domain. This technique at its first step generates the shape of the input data structure by solving a set of pointer constraints derived in a single pass of the statements along the path. At the next step the technique generates the data values in the fields of the data structure to force execution through the test path. Article [46] by Tracey et al. has reported an approach for representing testing specification failure and exception condition generation in an optimization framework. These two problems are simply a special case of constraint solving by involving a transformation of the variable values i.e. the execution of the software under test. Thus the same objective function can be used to guide the search for general constraint solving.

ATDG techniques addressing issues pertaining to feasible path analysis or infeasible path detection are arranged in this **sub-cluster**. A total of 12 articles i.e. [11, 14, 15, 16, 17, 22, 28, 29, 35, 40, 44, and 47] have reported ATDG techniques belong to this cluster. In the sub-cluster for Offutt and Pan [11], have chosen the mathematical constraints based heuristic approach for detection of equivalent mutants (which are the instances of feasible path problem). Here a heuristic-based set of transformations is applied to mathematical constraints to look for infeasibility. The technique generates test data as a by-product. Articles [14 and 15] have reported almost similar approaches. In [14] Christophe presents an automatic test data generator (ATGen) based on constraint logic programming and symbolic execution. Basic feature of their techniques is that it uses dynamic approach with symbolic execution (which analysis the path). In [15] Botella et al. have enhanced a previous work where the peculiarities of symbolic execution of programs with floating-point numbers are addressed. They have implemented a symbolic execution tool for ANSI C floating-point computations, called the Floating Point Symbolic Execution (FPSE) system. Article [16] by Alshraideh and Leonardo has reported a novel ATDG technique for issues related to branch coverage. The available test data within this is intended to cover program branches which depend on string predicates such as string equality, string ordering and regular expression matching. It provided a new method for searching for string test data and empirically proved its efficiency. Article [17] by Gallagher et al. is highly comprehensive and rich in bringing research implication for ATDG. To achieve the goal of ATDG, they introduced several testing techniques at major and minor levels. Generally, in their approach they used object oriented programming base. For finding feasible and infeasible paths, they adopted computational approach using database compute engine of control flow graph information for deriving define-use (du) paths. For branch coverage, Khor and Grogono [22] have used a genetic algorithm called ‘genet’. Genet produces test data for branch coverage with simple instrumentation. Article [28] by Yang et al. describes work in automatic generation of all-du-paths for testing shared memory parallel programs. For ATDG they adopted the procedure to generated all-du paths statically and developed an algorithm to find all-du-paths for shared memory parallel programs. A prototype tool Della pasta was used in this regard. In their novel approach they used temporal testing in conjunction with path testing. Article [29] has introduced a novel approach which is based on algorithms that entertain path selection as well as domain testing in the same time. The approach adopted here is a mix of testing criterion and hybrid in implementation method. The authors Hajnal and Forgacs claim that their approach is the first one to provide a complete solution of automated test data generation problem. Their method is based on the gradual creation of new program paths. Their extended path selection method involves static principal slicing into our algorithm otherwise, it is dynamic. The article [35] by McMinn et al., presented a unique species per path ATDG technique which tackle path problems through goal oriented search-based approach. It showed as to how the paths to the target can be assigned to species statically on the basis of slicing and transformation. In this case, Branch coverage was the key aspect of the approach. The evolutionary searches were performed here with the publicly available Genetic and Evolutionary Algorithm Toolbox (GEATbx). Article [40], by Diaz et al., shows its presence in this cluster because it reports an ATDG technique for branch coverage. In article [44] Baresel et al. have addressed test data search issue through evolutionary approach. A testability transformation algorithm is introduced which can handle flags assigned in loops, thereby improving the performance of a previously chosen test data generation technique. Bueno and Jino in [47] have presented a tool along with some techniques for the automation of test data generation and infeasible path identification. The dynamic technique for input data evaluation and search is based on genetic algorithms. Their work introduced a new fitness function that combined control and data flow dynamic information to improve test data searching. The unfeasibility issue is addressed by monitoring the Genetic Algorithm’s search progress. This approach is applicable indistinctively, even to non linear paths.

In the functional minimization **sub-cluster**, Articles [19, 24, 26, 29, 32, 33, and 45] have reported ATDG techniques. Articles [26 and 29] reported for this sub-cluster, have already been mentioned above in a sub-cluster where they show more strong bonding. Discussion on the remaining articles is as follows:

Article [19] by Jeng and Forgacs, reported ATDG technique that is mainly related to domain testing but it also contains path finding task. It combines the static approach and dynamic search method. The static approach computes the ON and OFF points by a series of algebraic operations whereas the dynamic search method determines an ON point in a trial-and-error manner. Their Shooting Line Algorithm (SLA) uses constraint slicing for functional minimization, which can inspect a family of prefix paths to a target predicate during the search for a solution.

The ATDG technique reported in articles [24, 32, and 33] by Michael et al. basically pertains to cluster of heuristic search but it shows its presence over here because of functional minimization property. This technique, which is an extension of a previously defined technique, minimizes the function by using a genetic algorithm.

The ATDG technique reported by Korel and AI-Yami [45] was adopted for regression testing. They used TESTGEN system, which uses chaining approach of goal-oriented method for nodes and branch coverage for test data generation.

4.2.2 ATDG Techniques Providing Graph-based Solution

Articles [13, 17, 20, 27, 28, 30, 31, 36, 37, 38, 40, 41, and 42] have reported ATDG techniques providing graph-based solution.

Article [13] by Pargas et al., has reported genetic algorithm based technique where the algorithm evaluates the candidate test data, and guides the direction of the search, using the program’s control-dependence graph.

Article [17] (introduced in cluster 4.3.1) has described a process which translates state transition specification of an object oriented software system component into a component flow graph with nodes and edges labeled for control and variable definitions and uses. These graphs are then used for executable test construction.

Article [20] while applying genetic algorithm, generates control flow graph in its adopted approach.

The ATDG technique reported by [27] used flow graph approach in optimization-based testing.

In their work [28] used the criterion of making an essential parallel program flow graph.

The constraint solving techniques mentioned in [30] used procedure control flow graph. This at first statically transforms a procedure into a constraint system by using "Static Single Assignment" form and control-dependencies. Subsequently, the system is solved to check feasible control flow path for test data generation.

In ATDG technique reported by Nguyen and Deville [31], they introduced Interprocedural Control Flow Graph (ICFG) for Interprocedural test data generation. The ICFG transforms the specified path into a path constraint which is then solved.

In [36, 37, and 42] where Gupta et al. presented Unified Numerical Approach (UNA) iterative relaxation technique to generate test data, and handled non-linear functions, used flow graph approach was used. To derive a basic set of paths for a given program they construct the control flow graph of the program.

Article [38] has described ATDG techniques within statistical structural testing, based on the combination of uniform generation of combinatorial structures, and of randomized constraint solving techniques. Their multi-stage comprehensive ATDG technique formalizes control flow graph to provide a way of uniformly drawing execution paths. Test data is generated by resolution of the predicates characterizing the input thus causing the execution of a given control flow path.

The ATDG technique reported by [40] apply Tabu search for automatically generating test data have used program control flow graph for branch coverage.

For automatic generation of black-box test data from a component's behavioral description, Edwards [41] has adopted the flow graphs generation approach. This two ended directed flow graph represents creation of an object at one end, and its destruction at the other end. The 'object lifetime', which is composed of some legal sequence of operations applied to a given object, is represented as some path through the graph.

4.2.3 ATDG Techniques Providing Mathematical or Numeric Solutions

Articles [12, 15, 17, 23, 36, 37, 42, and 43] have reported ATDG techniques providing mathematical or numeric solutions.

To solve the very basic issue of boundary values, Hoffman et al. [12] presented a technique based on mixed approach, which was then generalized at that time. This technique provides a mathematical solution of the issue, which manipulates boundary value with the combination of perimeter.

The article [15] by Botella et al. covered not only arithmetic operators but also the comparison and format-conversion operators. It adopted arithmetic approach in ATDG technique that addresses issues of floating point computation.

In article [17] Gallagher et al. adopted mathematical approach in the database compute engine of component flow graph. In their presented approach every database table could be associated with a mathematical set, where 'the set' was defined to be a set of sequences comprising only of the primary key values of the table.

Through a software system ADTEST for Ada83, [23] handled the problem of test data generation entirely as a numerical optimization problem. They used penalty functions in the optimization search.

In three consecutive articles [36, 37, and 42], as explained in sub section 4.3.3, Gupta et al. adopted numeric solutions providing ATDG techniques. In their different techniques they used traditional relaxation technique used in numerical analysis.

In their defined ATDG technique Edvardsson and Kamkar [43] have used Simplex method in combination with branch-and-bound and/or a cutting-plane algorithm solving constraints. The simplex method involves optimization of a linear function constrained by a linear equation system, which is a field of mathematical optimization.

4.2.4 ATDG Techniques for Issues of Syntax-based Testing

Syntax-based testing involves Boundary value analysis and Partition/Splice analysis. This cluster also contains techniques that address issues relating to domain testing, equivalence partitioning, domain partitioning, and functional analysis.

Articles [12, 19, 23, 29, and 34] have reported ATDG techniques that belong to this cluster.

To solve the basic issue of boundary values Hoffman et al. [12] presented a technique based on mixed approach, which was then generalized. In that technique the relationship between a generalized approach to boundary values and statement coverage is explored. This technique provides a mathematical solution of the issue, which manipulates boundary value with the combination of perimeter.

Article [19] has reported ATDG technique that is mainly related to domain testing. It combined the static approach with the dynamic search method. The static approach computes the ON and OFF points by a series of algebraic operations whereas, the dynamic search method determines an ON point in a trial-and-error manner. They proposed a Shooting Line Algorithm (SLA) which combines the two approaches to determine a pair of ON/OFF points simultaneously for a path domain. They applied it on C++ language.

The test data generation system ADTEST reported by [23] is also capable of locating input domain boundaries intersections and generating ON/OFF test data points. They applied ADTEST for domain testing on Ada83.

In their algorithmic approach, [29] have introduced a border testing strategy that considers predicates with single relational operator and also the compound predicates containing boolean operators. Their dual phase algorithm at first covers a predefined program path and then applies domain testing for additional requirement of ON and OFF points which lie very close to each other.

Tracey et al. [34] have reported an ATDG technique that uses simulated annealing search for test data. They generated a generic tool set for several applications. ‘Boundary value analysis’ being one of them.

4.2.5 ATDG Techniques Providing Programming-based Solutions

Articles [14, 15, 17, 18, 23, 24, 28, 29, 32, and 33] have reported ATDG techniques providing programming-based solutions.

In [14] Christophe presents an automatic test data generator ATGen, based on constraint logic programming. The technique was implemented on SPARK Ada programs where ATGen automatically generates test data for total decision coverage.

Botella et al. in [15], by using programming-based approach, implemented a symbolic execution tool for ANSI C floating-point computations, called the Floating Point Symbolic Execution (FPSE) system.

In their article [18], Offutt and Liu described a technique that can be used for automated test data generation from SOFL specification. SOFL is a specification language and methodology that is intended to be useable and graphical, and it is also expected to combine the object oriented design methodology with the structured design methodology. The technique basically addresses the issue of developing formalizable and measurable criteria for generating test cases from specifications.

Articles [24, 32, and 33] by Michael C.C et al. which are mainly discussed in cluster 4.3.6 all report GEDGET system. This system works on large software programs whereas C/C++ programs are used for description of programming based issues. The ATDG technique reported here uses combinatorial optimization to obtain condition/decision coverage of C/C++ programs.

In their approach, [17] have used object oriented programming base. The components of a graph drawn from database are equivalent to objects and classes of an object oriented language, that communicate with each other, however their testing tool approach is independent of the language.

In article [23], Gallagher and Narasimhan have reported an ATDG technique that provides programming-based solution. They experimented on Ada95 to insert instrumentation statements to dynamically return information about the values of the constraints which must be satisfied before the input data is made to execute a selected test path.

The all-du path finding algorithm reported by [28] uses programming-based approach. The two phases that annotate the graph and generate the du-path coverage, of this algorithm are based on programming.

In their article, [29] have adopted the algorithmic approach which provides programming-based solution. This algorithm computes ON/OFF points along a program path.

4.2.6 ATDG Techniques Based on Heuristic Search Method or Evolutionary Computation

Articles [11, 13, 16, 20, 22, 24, 26, 27, 32, 33, 34, 35, 38, 40, 46, 47, and 48] have reported ATDG techniques that are based on heuristic search methods or evolutionary computation. Gradient descent, Tabu search, Simulated annealing, and Genetic algorithms are all the instances of heuristic search.

To look for infeasibility of paths, article [11] has reported ATDG technique based on heuristics that automatically detects many of the equivalent mutants. Specifically, a heuristic-based set of transformations is applied to mathematical constraints to look for the infeasibility.

Article [13] by Pargas et al., has reported a genetic algorithm based goal-oriented ATDG technique. The test data generated through genetic algorithm causes execution of a given statement, branch, path, or definition–use (du) pair in the program under test. For ATDG they used the combination of two tools, named TGen and Random.

Article [16] has reported empirical evaluation of its ATDG technique where a steady-state style genetic algorithm was used to measure the cost function.

In article [20] Lin and Yeh reported an approach where they have applied genetic algorithm in path testing.

For branch coverage Khor and Grogono [22] have applied heuristic approach with genetic algorithm called 'genet'.

In three consecutive articles [24, 32, and 33] (where article [32] is a preliminary of [24]), Michael et al. used genetic algorithms and simulated annealing as tools for function minimization issue of ATDG. The genetic algorithm data generation tool (GEDGET), which uses dynamic test data generation paradigm, and is suitable specifically for large programs is introduced here. They basically applied GEDGET on large C/C++ program but in fact in their articles they addressed a number of issues related with ATDG for complex software systems. For empirical comparison with other heuristic approaches and verification purposes random data generation was used as a baseline.

In article [26] Claudia and Regina have introduced two GP-based procedures for selection and evaluation of test data. They have adopted an approach similar to the mutation analysis criterion; however, instead of mutant operators the concepts of evolution are used.

Article [27] by Mansour et al. reported an optimization-based path testing through simulated annealing and genetic algorithm. The genetic algorithm in this ATDG technique generates input data for path testing (not simply the statements and branches), and it works for both integer- and real-value data.

Tracey et al. [34] have reported a heuristic-global optimization technique that uses simulated annealing search for test data. They generated a generic tool set for applications like boundary value analysis, assertion/run-time exception testing, and component re-use testing.

The article [35] presented a good implementation of the species per path ATDG technique, which was based on evolutionary search. These evolutionary searches were performed with the publicly available Genetic and Evolutionary Algorithm Toolbox (GEATbx).

As a part of their comprehensive ATDG technique [38] used heuristics with some added criteria to awaken the constraints within the specified constraint solving processes.

Diaz et al. in [40] reported an ATDG technique that apply Tabu search for automatically generating test data for branch coverage. In their testing approach they combined Tabu Search with Korel's chaining approach.

In their constraint solving procedure [46] used the technique of simulated annealing.

In [47] Bueno and Jino introduced a tool and a technique. This tool dynamically evaluates input data and uses a genetic algorithm for input data selection that can execute the intended path. They introduced a new fitness function that combines control and data flow dynamic information for improving the process of search for the test data.

Liu et al. in article [48] have reported introduction of flag cost function, which is the main component of fitness function in evolutionary testing. They used the multi agent genetic algorithm to find the test data for the statements along with flag problems, at a low cost.

4.2.7 ATDG Techniques for Issues of Specification-based Testing

This section elaborates ATDG techniques concerning issues of specification based testing, which is a kind of BBT.

Articles [18, 46, and 49] have reported ATDG techniques concerning specification based testing issues. In their article [18], Offutt and Liu describe a technique that can be used for automated test data generation from SOFL specification. The technique basically addresses the issue of developing formalizable and measurable criteria for generating test cases from specifications. By using model-based approach of specification testing, the techniques solve the issue by defining measurable criteria.

Article [46] by Tracey et al. has reported an approach for representing testing specification failure and exception condition generation, in an optimization framework.

Article [49] by Bousquet and Zuanon has reported a testing environment named Lutess, for synchronous reactive software. Lutess offers several specification-based testing methods. These methods aim at simulating more realistic environment behaviors that produce relevant data with respect to some properties or interesting situations. These methods produce test data by using different type of guides, which include conditional probabilities, properties, and behavioral patterns.

4.3 Contribution of the Chapter

This chapter contributed mostly novel work regarding systematic review of ATDG techniques of the selected period. It has presented quantitative meta-analysis of the research about ATDG techniques. Its contribution to the thesis is in the sense that at first it depicted descriptive statistics concerning findings. These statistics include statistics of journals, conference proceedings and articles; statistics about articles area(s) of research; graphical representation of year-wise contribution of the articles; and other statistical facts of interest. In the second stage, the chapter embodies a very important section in which it depicts the extraction and elaboration of ATDG techniques. For highlighting/description of the discovered ATDG techniques an effective approach called 'Natural Clustering' has been introduced and explained. Through Natural clustering the discovered ATDG techniques are arranged and discussed in the form of clusters and sub-cluster of different levels. Highlighting of some of the facts, which could be helpful for classification of ATDG techniques, is an important byproduct of this chapter. In this manner it has addressed one of the core research questions which had been defined for the thesis. Another important contribution of this chapter is that it lays foundation for the other components of the thesis in general, and the base for classification and discussion in the subsequent chapters in particular. With due consideration to its contents and properties, this chapter could be called as the most critical component of the whole thesis.

5 CLASSIFICATION OF ATDG TECHNIQUES

Preamble: Classification improves understanding of the subject matter, opens doors for further research, and provides ease in reviewing the previous research work. These attributes of classification signify the necessity of classification in a research work. Concurrence of the author's views with this concept provided the required motivation for writing this chapter and classifying the ATDG techniques that had been extracted during the systematic review.

Classification of discovered ATDG techniques is one of the core components of the process of writing the thesis. On expanding the classification processes, as given in Figure1.1 of the first chapter i.e. 'Introduction', the resultant DFD has adopted the shape shown in Figure 5.1.

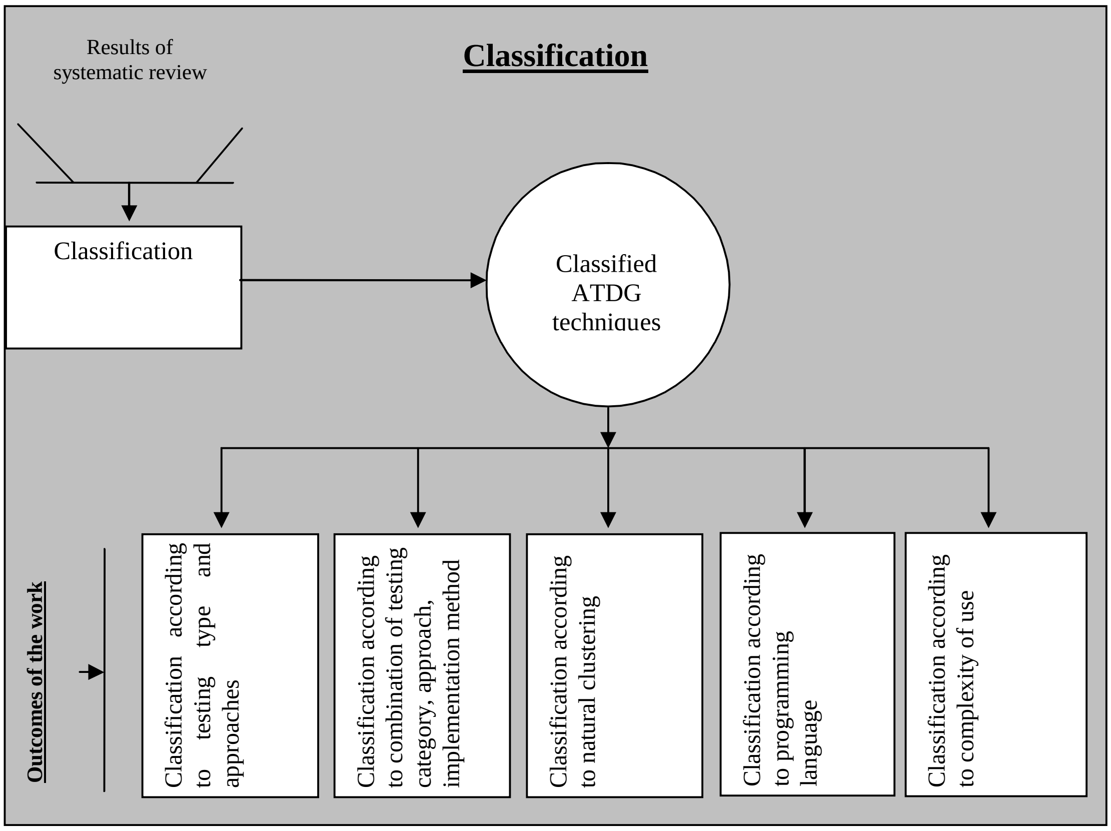

Results of systematic review
Classification
Classification
Classified ATDG techniques
Outcomes of the work
Classification according to testing type and approaches
Classification according to combination of testing category, approach, implementation method
Classification according to natural clustering
Classification according to programming language
Classification according to complexity of use

Figure 5.1 Two-Entity DFD of level-2, showing the classification process.

There could possibly be several criteria and ways of classifying the ATDG techniques. Each one of those methods and criteria has its own significance, pros and cons. Classification by 'types of testing' is commonly used criterion for classifying the ATDG techniques e.g. BBT, WBT, or Mixed. However, in author's point of view the classification should be based on objectivity, and it should be descriptive, meaningful, and purposeful.

The purpose of writing this chapter is to address one of the main research questions i.e. "How can ATDG techniques be classified?" With this objective, the initial sections of this chapter are dedicated to defining various criteria of classification.

Validity Threats Associated with Classification:

Prior to the classification of ATDG technique, some factors that could have affected the classification are worth pointing out along with the preventive measures:

a) Classification might not prove meaningful or purposeful.

Preventive measure: Mentioning of at least one purpose of every classification would prevent this threat.

b) Classification might not provide the required results. For example, misinterpretation in some specific condition might be the reason.

Preventive measure: Although this threat could not be eliminated completely because it refers to sudden specific conditions however, at least one example of every classification might reduce this threat.

5.1 Classification Based on Testing Types and Approaches

This mode of classification basically represents classification of ATDG techniques according to the adopted testing approaches. However, these approaches pertain to any one of the types of testing i.e., BBT, WBT, or Mixed, therefore, by default they get classification in this manner too.

A primary source of information for classification of discovered ATDG techniques according to testing types and approaches was article [46]. In that Tracey et al. have produced a highly relevant figure that could be used for such a classification. The figure basically describes some approaches to automating test-case data generation but additionally it also contains the fundamental information required for classifying the ATDG techniques. Fortunately all the discovered ATDG techniques got accommodated by the approaches shown in this figure. Realizing its importance the figure has been redrawn with reviewed titles according to their use.

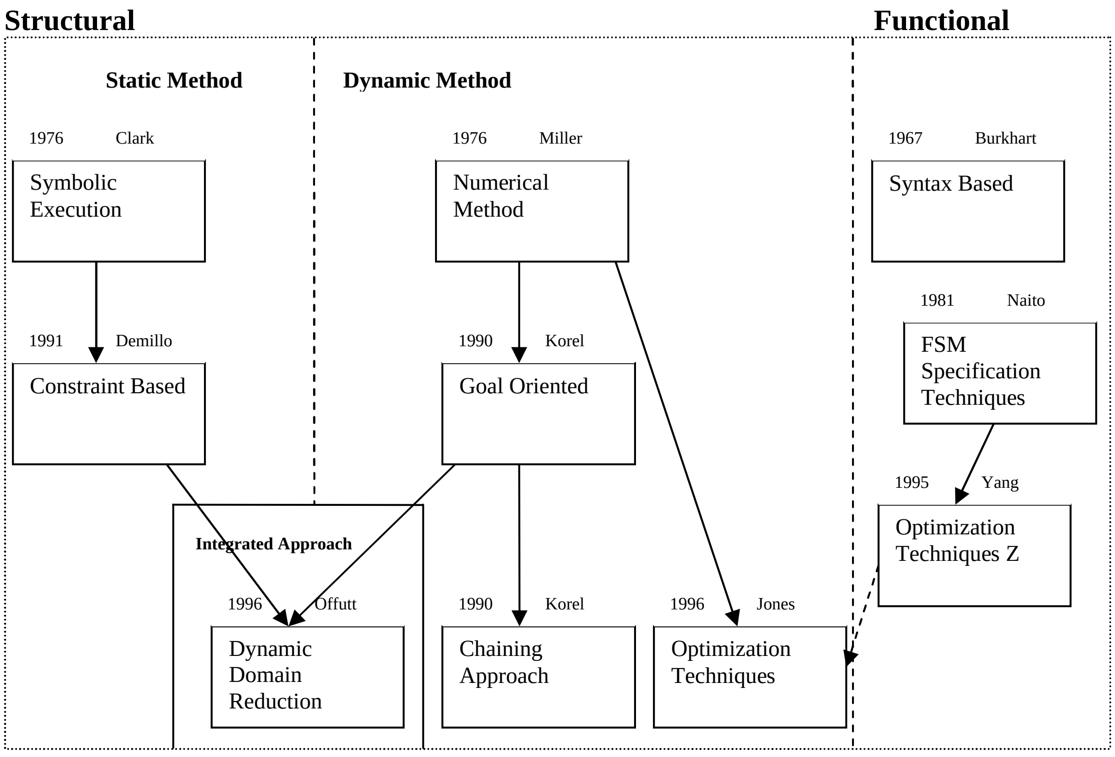


Figure 5.2 Classification of ATDG techniques according to testing type and approaches.

Figure 5.2 is self explanatory in the sense that each box represents an approach according to which the ATDG techniques could be classified. Titles above the figure show testing types and every discovered ATDG technique is associated with any one or more of these boxes.

It is to be noted here that techniques pertaining to structural testing might be either static or dynamic. However ATDG techniques related to functional testing could only be dynamic in nature. The same is evident from the statistics depicted in Table 4.2. This table also provides a solid evidence of the usefulness of this classification, which has further been strengthen through discussion on the statistics given in this.

5.2 Classification Based on Combination of Attributes of ATDG Techniques

Statistics presented in Table 4.2, have provided the opportunity for a new approach of classification of discovered ATDG techniques. According to this approach, attributes of ATDG techniques like testing category, approach, and implementation method, are taken in certain combination. These are not mere combinations but they rather represent the logic for provision of an optimal solution of some ATDG issue. Additionally these combinations highlight some important aspects of the representative ATDG techniques. Furthermore it provides help in linking and studying alike techniques with ease. Such classifications hold provision to draw a lot of new information. Table 5.1 shows a very clear picture of this classification along with real statistics of the discovered techniques.

Table 5.1 Classification of discovered ATDG techniques according to the combination of attributes of ATDG techniques.

Gr. No	Group (Testing category + Approach + Implementation method)	Relevant Article reference	Total Articles
G1	BBT + Random + Dynamic	12, 18, 24, 32, 33, 49	**6**
G2	BBT + Random + Hybrid	19, 25	**2**
G3	BBT + Path-orient + Dynamic	23, 41, 46	**3**
G4	WBT + Random + Dynamic	38, 42	**2**
G5	WBT + Path-orient + Static	11, 15, 28, 46	**4**
G6	WBT + Path-orient + Dynamic	14, 15, 16, 17, 21, 22, 26, 27, 31, 36, 37, 38, 39, 46, 47, 54	**16**
G7	WBT + Path-orient + Hybrid	43	**1**
G8	WBT + Goal-orient + Dynamic	13, 20, 44, 45	**4**
G9	WBT + Goal-orient + Hybrid	30, 35, 40	**3**
G10	Mixed + Random + Dynamic	34, 48	**2**
G11	Mixed + Path-orient+ Hybrid	29	**1**

Table 5.1 provides a unique way of identifying the discovered ATDG techniques. This classification relies upon associative approach. Based on the collected information, optimal combinations of attributes have been defined and then the techniques representing those particular combinations have been put against the associated groups.

5.3 Classification Based on ‘Natural Clustering’

The taxonomy adopted in Natural clustering represents the classification mode of commonality/similarity with respect to the nature of the extracted ATDG techniques. It represents a novel approach introduced in this thesis, called the ‘Natural clustering’. This approach indicates that ATDG techniques naturally tend to cluster against certain issue, approach, or method. In natural clustering these clusters of ATDG techniques are maintained in a hierarchical manner, such that one big cluster might contain sub-clusters up to several levels in depth.

Natural clustering is flexible in the sense that abstraction level of clusters can be maintained according to the necessity. Once an abstraction level is decided and a cluster is organized accordingly, the other probable sub-clusters could be detected for description.

Natural clustering is generic in the sense that it can accommodate any kind of grouping e.g., according to issue, testing type, or combination of attributes. In this way, it provides an effective method of classification. A detailed explanation of this technique is given in Chapter 4. Supplementing the information given in the last chapter, Table 5.2 depicts an example of natural clustering which is different from the one shown in Table 4.3. Here, the discovered techniques are shown forming clusters according to the implementation method of ATDG techniques.

Table 5.2 Classification of discovered ATDG techniques according to natural clusters for ATDG technique implementation method.

Sr. No	Implementation method	Article/ATDG technique ref	Total Articles
1	Static	11, 15, 28, 46	**4**
2	Dynamic	12, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 24, 26, 27, 31, 32, 33, 34, 36, 37, 38, 39, 41, 42, 44, 45, 46, 47, 48, 49, 54	**31**
3	Hybrid	19, 25, 29, 30, 35, 40, 43,	**7**

5.4 Classification Based on Programming Language

ATDG techniques could also be classified according to the programming language in which they are specified, or for which they are specified. In this kind of classification two different criteria could be set i.e. according to the language category or type. The language category might involve Sequential language, Object Oriented language, Intelligent language, or Specification based language. On the other hand the language type might involve languages like C, C++, and Java.

From the discovered ATDG techniques, only 10 are based on programming language, which can easily be classified according to the language category. Classification based on programming language might help in understanding and resolving the issues related to such specific techniques and that of the particular langue as well. Table 5.3 provides a categorical classification of the discovered ATDG techniques based on programming languages.

Table 5.3 Classification of discovered ATDG techniques according to programming language category.

Sr. No	Language type	Article/ATDG technique ref	Total Articles
1	Sequential language	15, 23, 24, 28, 29, 32, 33,	**7**
2	Object Oriented language	17, 24, 32, 33	**4**
3	Specification based language	18	**1**

5.5 Classification Based on Complexity of Use

Complexity of use could be another possible classification criterion that could be set for ATDG techniques. Determination of the complexity of use of an ATDG technique might involve several factors like: structure of the technique, its implementation criterion, testing issues it addressed, or it could also be determined on the basis of its empirical implication. To determine the complexity factor of all these cases, a methodology based on considerations like the author’s own experience, his intuition, and other’s experiences/comments seems to be viable. The same criterion has been adopted here and the levels have been classified as highly complex, complex, and less complex.

Considering the structure of the discovered ATDG techniques, they could be classified in descending order of complexity such as: Algorithmic, language based, and Graph based techniques.

The complexity of use regarding implementation of ATDG techniques deteriorates from Hybrid to Dynamic and then to Static.

Issues related to testing could be several in number and size. However the most commonly observed issues were Graph generation, path generation, path coverage, branch coverage, statement coverage, constraint generation, constraint solving, equivalent mutant generation and analysis, boundary value analysis, domain partitioning, flag issues, and data type issues. It has been observed that the ATDG techniques dealing with minute issues are more complex in use. Nevertheless the type of an approach used for TDG defines its level of complexity.

Empirical implication of an ATDG technique could be the best way to determine its level for complexity of use because probably it is the only possible way of highlighting all the factors that make a technique complex for use.

Taxonomy like complexity of use of ATDG techniques makes it easy to determine the time and cost factor involve in resolving a related issue. Pricing of associated ATDG tools could also be done on these bases. Moreover it helps in resolving technical issues on the bases of similarity.

5.6 Contribution of the Chapter

This chapter has encompassed both the quantitative and qualitative aspects of research. The main contribution of this chapter was in the form of the author’s own views about classification of ATDG techniques that were extracted during the systematic review. Out of several possible methods of classification, five have been mentioned with adequate explanation. Being unique in nature, most of the taxonomies adopted in the thesis have been presented with requisite logical justifications. Additionally, all these taxonomies are supported with real statistical information collected during the systematic review. As a quality measure the chapter not only considered validity threats to classifications but proposed and then applied preventive measures to them. To summarize, chapter successfully addressed the core research question about possible ways of classification of ATDG techniques.

6 DISCUSSION AND ANALYSIS

In general this chapter presents discussion on results of the systematic review and classification of ATDG techniques. Notwithstanding its outlook, this chapter focuses on the new knowledge that has been gained through the systematic review. It also offers a qualitative meta-analysis of the research about ATDG techniques. The fundamental research question of "What can be deduced regarding general guidance and future work, from the systematic review and classification of ATDG techniques?" has been answered in this chapter.

Discussion on results of the systematic review and classification of discovered ATDG techniques is one of the core components of the thesis process. Expansion of the discussion process of Figure1.1 is given as two-entity DFD of level-2 in Figure 6.1.

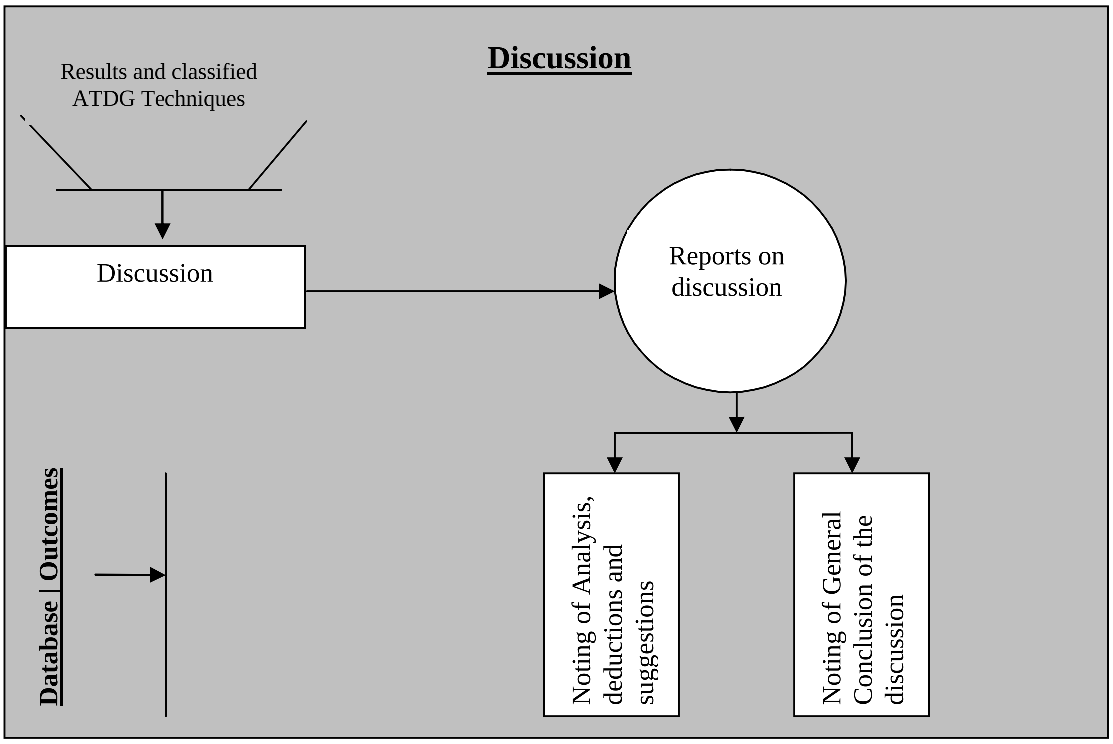

Discussion
Results and classified ATDG Techniques
Discussion
Reports on discussion
Database | Outcomes
Noting of Analysis, deductions and suggestions
Noting of General Conclusion of the discussion

Figure 6.1 Two-Entity DFD of level-2, showing the discussion process

Validity Threats Associated with Analysis and Discussion: The author believes that an analysis might be well motivated and based on a purpose or aim. Likewise the discussion must be purposeful, based on solid arguments and expected to bring some trust worthy results. Materialization of such aims could not be without threats, which might be in the form:

a) The analysis or discussion based on vague evidences.

Preventive measure: The analysis and discussion on logical, authentic, and proven evidences and arguments is the best preventive measure to such threats.

b) Discussion might go open ended or lead to wrong direction.

Preventive measure: A clear description of the results of the discussion like for example, whether the results are based just on solid facts or some intuitions from the author are involved, might prevent such threat. Furthermore a proper conclusion of the whole discussion might prevent misleading effect of the discussion.

6.1 Analysis, Deductions and Suggestions

This section presents discussions based on the knowledge gained during the systematic review, the deep analysis of the results and the classifications of ATDG techniques. On the basis of results gained through discussions it also provides on-spot recommendations.

6.1.1 Analysis of Descriptive Statistics

This sub-section analyses the results of systematic review achieved in the form of statistics and on the basis of this analyses, the results have been deduced. Most of the analysis results have already been presented within the scope of their respective chapters, the remaining analysis, however, have been presented in this chapter.

Chu et al. in their article [25] have proved with their empirical study and also with the support of other articles that BBT is more effective than WBT. However, statistical trend about ATDG techniques of the selected period indicates more inclination towards WBT. This might indicate that the research trend is somewhat distancing away from its more beneficial aspects.

McMinn [53] narrates "The use of metaheuristic search techniques for the automatic generation of test data has been a burgeoning interest for many researchers in recent years". This systematic review proves this narration. At least 19 ATDG techniques are either fully or partially dependent on metaheuristic search techniques. An analysis of the composition of this cluster reveals that 12 ATDG techniques depend on genetic algorithms, 4 techniques depend on simulated annealing, 3 reported algorithms are based on evolutionary search, and one on Tabu search, whereas some of the techniques have been adopted on a mixed approach. All these techniques are instances of heuristic search.

Statistics reveal that only three articles [18, 46, and 49] have reported the techniques directly related to specification-based testing. However in 1999 [18] highlighted vital benefits of this approach by describing "Specification-based testing is also currently immature, which means that there is a scarcity of formalizable criteria and automated tool support." Further research reveals that during the next seven years of publishing this statement not a single attempt was made to develop ATDG technique on this pattern. Considering the significant benefits of this approach it is recommended that more research may be done in this area.

6.1.2 Nature of the ATDG Techniques

Analysis of the results of the research revealed that on the basis of their nature, ATDG techniques could broadly be sub-divided in two categories. They could either be general in nature to tackle the whole process of ATDG or at least most of its associated issues (e.g., [23]), or they could be more specific, tackling the issues related to just small task of the ATDG process (e.g., [48]). Some of the techniques, however, are mere extensions of some previously described techniques e.g., [38].

6.1.3 Specific Observations, Analyses and Deductions

In the article [20] the word 'test case' has been used instead of 'test data'. But this can be justified by a statement from Alshraideh and Leonardo [16] where they state "The genetic algorithm maintains not one but a population of many candidate solutions. A candidate solution is an attempt to solve the problem. In the context of test data generation, a candidate is a potential test case."

Several techniques adopting a mixed approach have been discovered so far but only a few adopt mixing of contradictory approaches. An example in this regard is article [14] where the ATDG technique used symbolic execution for dynamic approach the results thus achieved are impressive. Since experimentation of this nature brings innovation therefore it should be encouraged.

Only a few techniques provide a complete solution of the whole problem of ATDG; most of the techniques however, concentrate only on a specific part of the whole problem. Article [29] has reported such a technique that provides complete algorithm for ATDG. It is believed that such techniques might bring optimal combination with complete compatibility of the related components; therefore they are likely to produce appreciable results.

6.1.4 General Observations, Analysis and Deductions

During the decade selected for research no work was found that could explicitly refer to an ATDG technique based on purely intelligent approach. Only [13] have mentioned a mixed approach with intelligent elements. Techniques reported in articles [31, 44, and 48] have adopted nearly the same approach but it is not purely intelligent. However it is believed that computing trend of the future is towards independent software modules, which would definitely involve intelligent testing. It could, therefore, be deduced that volume of research on this aspect is considerably low, which needs to be expanded.

Encouraging trends towards the use of metaheuristic approaches were observed. Use of these approaches seemed to produce better results in handling the issues of ATDG. In this regard, article [58] reported hybrid meta-heuristic algorithm named SAaGA. By combining the parallel search ability of the adaptive genetic algorithm (aGA) with the controllable jumping property of simulated annealing (SA) SAaGA produced quite encouraging results for ATDG. The article proposed further research probably with the combination of genetic algorithm and Tabu search. “The Genetic and Evolutionary Computation Conference” is also contributing to the search in this important aspect. The author strongly believes that research in this area and on the lines suggested by [58] could prove beneficial for ATDG.

6.1.5 Analysis of Classification

It has been observed that several different classifications of discovered ATDG techniques are possible. All carry worth according the use and situation. To classify the discovered ATDG techniques five distinct approaches were adopted in this thesis. These classifications are based on the author’s own point of view that holds the notion that classification should be descriptive, meaningful and purposeful. Descriptiveness and meaningfulness are quite clear from the respective chapters; however, the purpose of classification depends on the user’s particular necessity. For instance, classification according to testing type and approaches might serve two basic purposes i.e. the identification and comparison of strength of ATDG techniques or to produce the linkage between newly developed and old techniques.

6.2 Conclusion of the Discussion

To conclude the discussion it can be said that a big gap exists in research on some of the ATDG areas like intelligent approach, specification-based approach, and component-based approach. Additionally, the discussion highlighted some trends like ‘use of metaheuristic and intelligent approach’, in the research which did not get the due importance, but it could prove worthy for ATDG if it is adopted on a large scale. Likewise taxonomies of ATDG techniques could highlight certain aspects carrying worth for further research and produce new techniques as a result of that.

It was almost impossible for the author to discuss each and every aspect of the work therefore only the hot moot points have been highlighted in the discussion section. Valid conclusions from the ensuing discussions and on-spot recommendations have also been included in the research work.

6.3 Contribution of the Chapter

Scope of this chapter encompassed vital discussions on the results of systematic review, findings and classification of ATDG techniques. It carried important analysis and suggestions from the author on different aspects of findings. The most significant contribution of this chapter is that it has satisfactorily addressed one of the fundamental research questions about critical analysis of findings, deductions of results, and discussions on the work. The discussion process in the research method DFD of Figure 1.1 has been expanded in this chapter. Overall analysis and discussion produced in this chapter is done while keeping in view the probable validity threats.

7 EPILOGUE

This chapter comprises Summary of the thesis, the Conclusion, General recommendations on the basis of the study carried out for the thesis, and the identification of the areas where further research might be needed.

7.1 Summary

Today the cumbersome task of software testing is relying more and more on the support received from automated tools. The automation of software test data generation is of particular concern to the researchers. In this regard whatever had been done during the past ten years i.e. from 1997 to 2006 has been systematically reviewed in this thesis. During the course of research for this thesis, preliminary literature study indicated that to the best of the author's knowledge, no systematic review has been published so far that could report on Automated Test Data Generation (ATDG) techniques. Therefore to accomplish the task three comprehensive research questions were posed, which guided the design of this review process. The review was also depicted in the form of two-entity DFD. Later on each sub-process of this DFD was expended which represents the proceedings of the respective chapter.

To build up a strong knowledge base of the topic, an additional 'background' chapter has been included. This chapter did not only provide a motivational background of the topic but it also comprehensively elaborated the basic concepts about key areas and issues concerning the ATDG. Additionally, it also provided the related work that helped in analyzing the quality of this systematic review and to make it prominent through contrast.

For the systematic review a comprehensive and reliable research method has been designed. In addition to the author's own creative approach, this research method took inspiration from three key articles [1, 5, and 8]. The adopted research method encompassed all steps from research engine selection to the extraction of ATDG techniques. Each and every step that is involved in this research method is well explained and documented in the thesis there by leaving no stone unturned to make it fool hardy and credible. The chapter also describes the criteria of storing the search results in a comprehensive and strong database, specifically designed to accomplish this task. A parallel use of specifically designed 'Articles count log' and 'Data log' are making to strengthen the results/data storage criteria.

In this document, outcomes of the systematic review are documented as 'Results'. The obtained results are presented in the form of statistics and description of findings of the discovered ATDG techniques. For the systematic description of discovered ATDG techniques a novel approach called 'Natural clustering' has been introduced. This approach offers great benefits like easy description of ATDG techniques, identification of similar techniques, flexibility, and generality in use. Natural clustering was adopted on the basis of merit after drawing a comparison with the available approaches. Natural clustering facilitated elaboration/comparison and classification of the discovered ATDG techniques in an impressive and encouraging way.

The thesis also provided classification of the ATDG techniques discovered during the systematic review. Later the adopted taxonomy has been generalized for all ATDG techniques. As a result of this quantitative and qualitative effort, five types of classifications have been exposed that are based on:

1. Testing type and approaches;
2. Combination of testing category, approach, and implementation method;
3. Natural clustering;
4. Programming languages;
5. Complexity of use.

A comprehensive discussion in the end highlighted some important aspects about the findings. The discussion has provided vital results deduced on the bases of information gained during the systematic review from selected or exclusive resources. As part of the discussion on-spot recommendations provide guidance for future research.

It is worth mentioning here that validity threats or risk factors have been given due consideration throughout this work which have resulted in improvement of the overall quality of the work. Additionally it brought reliability and credibility for the work, which could attract new researchers in the field. In most of the cases, the validity threats are discussed in the beginning of a chapter, which has improved the readability aspect of the thesis.

The structure of the thesis is designed purposefully in the manner that every chapter starts with a preamble, then the main body, and ends up by mentions of the contributions it provided for the thesis.

The abovementioned provides multi-dimensional and vital contribution of this work in summarized form. The information provided in this section could also be used as ready reference for the next coming 'Conclusion' section.

7.2 Conclusions

Overall it can be concluded that the thesis by using proven methods and some novel approaches, successfully highlighted important facts about various aspects of ATDG techniques. The continuity in research during the selected period, current rising trend in research, researcher's experimentation for new techniques, and research gaps in different methods/approaches of ATDG techniques are some of the important aspects brought into light by the thesis. It has also been discovered that there could be several possible ways to classify and present ATDG techniques for discussion and then for further carrying forward the research. Furthermore the research disclosed that in the volatile world of computer, there exist a number of research opportunities in the field of ATDG techniques.

The thesis has contributed to the knowledge pool in several distinct manners. These contributions are specifically mentioned at the end of each chapter and in the summary part. However, the systematic review can be called the core contribution in the field of ATDG techniques. Within the defined structure the thesis fulfilled all the requirements of a standard systematic review as they are proposed by Kitchenham [1]. Efforts has been made to address the topic well, offer a meaningful presentation of a rich collection of the data, keep the deductions well supported by logical arguments and the logical sequence of presentation of the text. The expression has been kept lucid, clear, precise and concise such that it would become easily understandable even for naïve users. The structure has been designed on the patterns of proven works. Providing preamble in the beginning and contribution of the chapter in the end of each chapter has further improved the quality of the thesis structure. Quality factor has been given prime importance at each and every step during research and presentation of the whole thesis matter. In this regard risk factors and validity threats are specifically taken care of and addressed. The thesis has successfully addressed all the research questions posed to achieve the defined aims and objectives. Some Novel work for example 'Natural Clustering' approach and modes of classification of ATDG techniques has been produced. Furthermore through deep analysis and meaningful discussions it highlighted some gaps in the research and provided recommendations for them. For having elaboration of the concluding remarks at a glance the above given summary section is recommended to be consulted first. Otherwise rests of the chapters are sufficient to provide adequate elaborations. It has already been mentioned that the year 2006(last year of the selected period) and 2007 (the current year) have shown rising trends in publications related to the ATDG techniques. Therefore this effort of presenting a systematic review could be remarked as a timely support for research community.

7.3 Recommendations

The author believes that recommendations should always be comprehensive, all encompassing and given with a complete authority on the subject, therefore, they must be based upon and supported by logical reasoning, and derived from proven facts. With this motivation, mostly on-spot recommendations have been provided in this thesis, which have been put forward right after deduction of some results. The whole thesis in general and chapter 'discussion' in particular, contains many on-spot recommendations. Nevertheless, some general recommendations are still required to be forwarded.

It has been observed during the systematic review that most of research work is performed on experimental or toy software, which is short in size, simple, and based on subset of language features. Under these environments, real worth of these ATDG techniques in actual practice has not been established as yet. It is therefore recommended that results gained through empirical implementation of such techniques may be considered for further research on such ATDG techniques.

During the systematic review, it was found that on some of the important areas, issues, or methods like parallel programming [24], Tabu search [40], and equivalent mutants [11], only a single attempt was made. This observation is indicative of a gap in the research that might have to be filled.

Software languages are experiencing new trends, which are mostly towards the intelligent side. Need of the hour therefore, is to do more research on ATDG techniques related to such languages.

Computing trends are diverting more and more towards the independent development of components. It is, therefore, recommended that this trend should also be considered for the research on ATDG techniques.

Research by [25] has proven that BBT is more beneficial than WBT. It is therefore recommended that more research should be done while considering this aspect, it is expected that it would not only save the resources but would also improve upon the quality of ATDG.

7.4 Future Work

The work presented in this thesis is, hopefully, comprehensive within the defined scope but research never ends therefore, future research is expected to explore horizons beyond the scope of this thesis. It is hoped that the limitations of this work would be considered as the beginning for the research in future.

Containment of research to specified journals and conference proceedings might have incurred a publication bias (according to [1]). For limiting and compensating for this bias, studies based on large sample would be preferred in future.

It was speculated on the basis of primary study and other evidences that the articles of the selected decade might depict all the ATDG techniques that are being used or would be used in future. This speculation has contained the scope of this work within the defined era. However, it is hoped that further work would extend the scope of the search by removing such limitations.

Due to cogent reasons ATDG techniques based on UML diagrams were not entertained in this work (see section 3.1). Although preliminary search has shown very few articles in this respect but considering the significance of this area, it is expected that it would be included in the next attempt.

A case study on use of ATDG techniques in the software industry (which was left from performing mainly due to time constraint) could bring several interesting results for the systematic review.

This thesis has provided an up-to-date knowledge about the research trends, volume of research in specific areas, and newly introduced ATDG techniques and approaches. Hopefully, future work either in the form of survey of some specific area or a systematic review, would be based on the knowledge produced in this systematic review.

TERMINOLOGY

Systematic review:	A systematic review is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, topic area, or phenomenon of interest. Systematic reviews aim to present a fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology [1].
Software testing:	Software testing is a process of ensuring that a certain piece of software item fulfills its requirements [7].
Test data:	Test data is the input needed for executing the test code, which has been written to test the SUT.
Test data generation:	Test data generation in program testing is the process of identifying a set of test data that satisfies a selected testing criterion, such as, statement coverage or branch coverage [56].
Test Suite:	The collection of test data generated by the test data generator (A tool consists of a constraint generator and a constraint solver) is called a test suite [43].
ATDG technique:	It is a technique used for automatic test data generation for SUT.
Structural or White Box Testing (WBT):	Structural testing tests the program at the structural level. It includes (1) choice of a criterion (statement, branch or path), (2) identification of a set of nodes, branches or paths, and (3) generation of test data for each element of this set [31].
Functional or Black Box Testing (BBT):	functional testing is considered black-box, where the test inputs and expected outputs are derived solely from the functional specification [46].
Integration testing:	It ensures that messages from objects in one class or component are sent and received in the proper order and have the intended effect on the state of the objects that receive the messages [17].
Regression testing:	It involves testing the modified program with test cases in order to establish the confidence that the program will perform according to the (possibly modified) specification. In the development phase, regression testing may be used after the detection and correction of errors in a tested program [45].
Combinatorial Testing:	A testing situation that requires testing several input variables in combination [51].
Temporal testing:	Temporal testing alters the schedule of execution time of program segments in order to detect synchronization errors [28].
Domain testing:	It is usually observed in software testing that input data near the boundary of a domain partition are more sensitive to program faults and should be carefully checked. A formalization of this intuitive observation is known as domain testing, which was first proposed by White and Cohen (1980) [19].
Specification-based testing:	In it the specifications are used to produce test cases, as well as to produce the program [18].
Statistical testing:	Statistical software testing involves exercising a piece of software by supplying it with test data that are randomly drawn according to a defined probability distribution on its input

	domain. This provides a scientific basis for making inferences from testing about operational environment [25].
Static testing techniques:	Techniques that are concerned with the analysis and the checking of system representations such as the requirements documents, design diagrams and the software source code, either manually or automatically, without actually executing the software [25, 55]. OR Static techniques are those which do not require the software under test to be executed. They generally use symbolic execution to obtain constraints on input variables for the particular test criterion. Solutions to these constraints represent the test-data [46].
Dynamic testing techniques:	Techniques that examine the software with a view to generating test data for execution by the software [25, 55].
Evolutionary testing:	Evolutionary testing is a search–based software engineering technique, based upon evolutionary algorithms like Genetic algorithm, Simulated annealing, Gradient descent, and Tabu search [44].
Metaheuristic search techniques:	These are high-level frameworks which utilize heuristics in order to find solutions to combinatorial problems at a reasonable computational cost [53].
Fitness Function:	In evolutionary testing at the end of each generation, each solution is evaluated for its fitness, using a problem-specific fitness function. The fitness function is made up of two components – the approach level and the branch distance. The approach level measures how close an input vector was to executing the current structure of interest, on the basis of how near the execution path was to reaching it in the program's control flow graph. The branch distance reflects how 'close' the alternative branch from the last critical branching node (a branching node with an exit that, if taken, causes the target to be missed) was to being taken [42].
Optimization techniques:	Optimization techniques are simply directed search method which attempts to find minimal (or maximal) values of a particular objective function. Optimization techniques make very few assumptions about the underlying problem which they are attempting to solve. Thus to apply optimization techniques to a problem it is only necessary to devise an objective function to direct the search. Simulated annealing is one such optimization technique [46].
Formal specifications:	Formal specifications precisely describe what functions the software is supposed to provide in a form that can be easily manipulated [18].
Dynamic methods:	Dynamic methods require that the software under test is executed. Dynamic methods generally involve a directed search for test-data which meets a desired criterion [46].
Cost Function:	The guidance for searching program input in dynamic method is given by cost function. This guidance is given in the form of a cost- function which relates a program input to a measure of how good it is. The amount of guidance given by the cost-function is one of the key elements in determining the effectiveness of the test-data generation [34].
Symbolic execution:	The symbolic execution of computer programs is an automatic static analysis technique that allows the derivation of symbolic expressions encapsulating the entire semantics of programs. Symbolic execution extracts information from the source code

	of programs by abstracting inputs and sub-program parameters as symbols rather than by using actual values as during actual program execution [14]. Or Symbolic execution is a classical program testing technique which evaluates a selected control flow path with symbolic input data. A constraint solver can be used to enforce the satisfiability of the extracted path conditions as well as to derive test data [15].
Genetic algorithm:	A genetic algorithm is a heuristic that mimics the evolution of natural species in searching for the optimal solution to a problem. In the test-data generation application, the solution sought by the genetic algorithm is test data that causes execution of a given statement, branch, path, or definition–use pair in the program under test. The genetic algorithm conducts its search by constructing new test data from previously generated test data that are evaluated as good candidates. The algorithm evaluates the candidate test data, and hence guides the direction of the search, using the program's control-dependence graph [13].
Feasible path analysis problem:	It states that for certain structural testing criteria some of the test requirements are infeasible in the sense that the semantics of the program imply that no test case satisfies the test requirements. Equivalent mutants, unreachable statements in path testing techniques, and infeasible DU-pairs in data flow testing are all instances of the feasible path problem [11].
Define-Use (du) Pair:	In a control flow graph, edges are formed from the branching statements of the program. A *definition* (*def*) of a memory location *x* is a node in which *x* is given a value and a *use* is a node in which that value is accessed, either through the same name or a different name via aliasing. Hence a *DU-pair* is a definition and a use of the same location such that there is a def-clear sub-path from the def to the use [17].
Define-Use pair coverage:	The general procedure for finding a du-pair coverage begins with finding du-pairs in a program. For each du-pair, a path is then generated to cover the specific du-pair. Finally, test data for testing the path is produced [28].
Function minimization:	The function minimization for each branch predicate is carried out with respect to one input variable at a time and cycles through all the input variables in search for an input that forces the predicate to evaluate to the desired outcome. While trying to modify the input to force execute the selected branch, if a preceding branch evaluates to an undesired outcome, then the method backtracks to the preceding branch predicate and the process is repeated by minimizing the function associated with that predicate [36].
Control flow graph:	A graph where the nodes are either a decision node or a block of instructions without decision statement. The edges represent the possible control flow between the nodes [21].
Statement coverage:	It requires exercising a given set of program statements (a set of nodes). The problem is thus to find, for each program statement, a program input on which this statement is executed [21].
Branch coverage:	It is the dual version where an input must be found, such that the execution traverses a specified edge of the control flow graph associated to the program [21].
Path coverage:	It is a generalization of branch coverage where the input causes the execution of a specified path [21].

Constraint:	A constraint is a mathematical algebraic expression that restricts the input space of the program to be the portion of the input domain that satisfies a certain goal [11].
Random test-data generators:	These generators select random inputs for the test data from some distribution [13].
Structural or path-oriented test-data generators:	These generators typically use the program's control flow graph, select a particular path, and use a technique such as symbolic evaluation to generate test data for that path [13].
Goal oriented test-data generators:	These generators select inputs to execute the selected goal, such as a statement, irrespective of the path taken [13].
Intelligent test-data generators:	These generators often rely on sophisticated analyses of the code to guide the search for new test data [13].
Test trajectory:	A test trajectory is defined as a series of data points, with each (possibly multidimensional) point relying upon the value(s) of previous point(s) [54].

REFERENCES

[1] Barbara Kitchenham. Procedures for Performing Systematic Reviews. *Keele University Technical Report* TR/SE-0401 ISSN: 1353-7776, 2004 and *NICTA Technical Report* 0400011T.1, 2004

[2] C.W. Dawson. *The Essence of Computing Projects – a Student's Guide*. Prentice Hall, Harlow UK, 2000

[3] D. M. Andrews and J. P. Benson. An Automated Program Testing Methodology and its Implementation. In *Proceedings of the 5th International Conference on Software Engineering*, pages 254–261, IEEE Computer Society, 1981

[4] James Miller, Marek Reformat, and Howard Zhang. Automatic Test Data Generation Using Genetic Algorithm and Program Dependence Graphs. *Information and Software Technology*, vol. 48, pages 586–605, 2006

[5] Jo E. Hannay, Dag I.K. Sjoberg, and Tore Dyba. A Systematic Review of Theory Use in Software Engineering Experiments. *IEEE Transaction on Software Engineering*, vol. 33, No.2, February 2007

[6] Jon Edvardsson. A Survey on Automatic Test Data Generation. In *Proceedings of the Second Conference on Computer Science and Engineering in Linkoping*, pages 21-28. ECSEL, October 1999

[7] R. Torkar. Towards Automated Software Testing, Techniques, Classifications and Frameworks. *Karlskrona Blekinge Institute of Technology,* 2006:4. p. 235 ISBN: 91-7295-089-7, 2006

[8] D.I.K. Sjøberg, J.E. Hannay, O. Hansen, V.B Kampenes, A. Karahasanovic, N.K Liborg, and A.C. Rekdal. A Survey of Controlled Experiments in Software Engineering. *IEEE transactions on software engineering*, VOL. 31, NO 9, 2005

[9]C.C. Michael, G. McGraw, and M.A. Schatz. Generating Software Test Data by Evolution. *IEEE Transactions on Software Engineering*'01 27(12): 1085-1110, 2001

[10]L. Clarke. A system to generate test data and symbolically execute programs. *IEEE Transactions on Software Engineering*, SE–2(3), 215-222/76, Sept. 1976

[11]Offutt A. Jefferson and Pan Jie. Automatically detecting equivalent mutants and infeasible paths. *Software Testing, Verification and Reliability* VOL 7, 165–192/97, 1997

[12] Hoffman Daniel, Strooper Paul, White Lee. Boundary values and automated component testing. *Software Testing, Verification and* Reliability 9, 3–26/99, 1999

[13] Pargas Roy P., Harrold Mary Jean, Peck Robert R. Test-data generation using genetic algorithms. *Software Testing, Verification and Reliability* 9, 263–282, 1999

[14] Meudec Christophe. ATGen: automatic test data generation using constraint logic programming and symbolic execution. *Software Testing, Verification and Reliability* 11:81–96 (DOI: 10.1002/stvr.225), 2001

[15] Botella Bernard, Gotlieb Arnaud, Michel Claude. Symbolic execution of floating-point computations. *Software Testing, Verification and Reliability* 16; 97–121, 2006

[16] Alshraideh Mohammad, Bottaci Leonardo. Search-based software test data generation for string data using program-specific search operators. *Software Testing, Verification and Reliability* 16; 175–203, 2006

[17] Gallagher Leonard, Offutt A. Jefferson, Cincotta Anthony. Integration testing of object-oriented components using finite state machines. *Software Testing, Verification and Reliability* 16; 215–266, 2006

[18] Offutt A. Jefferson and Liu S. Generating test data from SOFL specifications. *The Journal of Systems and Software* 49; 49-62, 1999

[19] Jeng B, Forgacs I. An automatic approach of domain test data generation. *The Journal of Systems and Software* 49; 97-112, 1999

[20] Jin-Cherng Lin, Pu-Lin Yeh. Automatic test data generation for path testing using Gas. *Information Sciences* 131; 47-64, 2001

[21] Nguyen Tran Sy, Deville Y. Automatic test data generation for programs with integer and float variables. In *proceedings of the 16th International Conference on Automated Software Engineering16;* 3-21/01; 2001

[22] Khor S., Grogono P. Using a genetic algorithm and formal concept analysis to generate branch coverage test data automatically. In *Proceedings of the 19th International Conference on Automated Software Engineering04;* 1068-3062/04; 2004

[23] Gallagher M.J, Lakshmi Narasimhan V. ADTEST: a test data generation suite for Ada software systems. *IEEE Transactions on Software Engineering97;* Vol. 23, No. 8, August 1997

[24] Christoph C. Michael, McGraw G., Schatz M.A. Generating software test data by evolution. *IEEE Transactions on Software Engineering01;* Vol. 27, No. 12, December 2001

[25] Huey-Der Chu, John E. Dobson, and I-Chiang Liu. FAST: a framework for automating statistics-based testing. *Software Quality Journal* 6; 13–36, 1997

[26] Emar Maria Claudia F.P. and Vergiio Silvia Regina. Selection and Evaluation of Test Data Based on Genetic Programming. *Software Quality Journal03;* 11, 167–186, 2003

[27] Mansour, Nashat; Salame, Miran. Data Generation for Path Testing. *Software Quality Journal04;* 12, 121–136, 2004

[28] Cheer-Sun D. Yang, Amie L. Souter, Lori L. Pollock. All-du-path Coverage for Parallel Programs. In *proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis98;* Vol. 23, No. 2, 153-162, ISSTA 1998

[29] Akos Hajnal and Istvan Forgacs. An Applicable Test Data Generation Algorithm for Domain Errors. In *proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis ISSTA98;* Vol. 23, No. 2, 63-72/98, ISSTA 1998

[30] Arand Gotlieb, Bernard Botella, Michel Ruether. Automatic Test Data Generation using Constraint Solving Techniques. In *proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis98;* Vol. 23, No. 2, 53-62/98, ISSTA 1998

[31] Nguyen Tran Sy and Yves Deville. Consistency Techniques for Interprocedural Test Data Generation. In *proceeding of the 9th European software engineering conference held jointly with 11th ACM SIGSOFT international symposium on Foundations of software engineering03;* Vol. 28, No 5, 108-117/03, ESEC/FSE 2003

[32] Christoph C. Michael, Gary E. McGraw, Michael A. Schatz, and Curtis C. Walton. Genetic Algorithms for Dynamic Test Data Generation. In *proceedings of 12th International conference on Automated Software Engineering97;* 307-308/97, ASE 1997

[33] Christoph C. Michael and Gary McGraw. Automated Software Test Data Generation for Complex Programs. In *proceedings of 13th International conference on Automated Software Engineering98;* 136-146/98, ASE 1998

[34] Nigel Tracey, John Clark, Keith Mander, and John McDermid. An Automated Framework for Structural Test-Data Generation. In *proceedings of 13th International conference on Automated Software Engineering98;* 285-288/98, ASE 1998

[35] Phil McMinn, Mark Harman, David Binkley, and Paolo Tonella. The Species per Path Approach to Search Based Test Data Generation. In *Proceedings of the 6th International Symposium on Software Testing and Analysis06;* 13-24/06, ISSTA 2006

[36] Neelam Gupta, Aditya P. Mathur, and Mary Lou Soffa. UNA Based Iterative Test Data Generation and its Evaluation. In *proceedings of 14th International conference on Automated Software Engineering99;* 224-232/99, ASE 1999

[37] Neelam Gupta, Aditya P. Mathur, and Mary Lou Soffa. Generating Test Data for Branch Coverage. In *proceedings of 15th International conference on Automated Software Engineering;* 219-227, ASE 2000

[38] S.D. Gouraud, A.Denise, M.C. Gandel, and B. Marre. A New Way of Automating Statistical Testing Methods. In *proceedings of 15th International conference on Automated Software Engineering01;* 5-12/01, ASE 2001

[39] Srinivas Visvanathan and Neelam Gupta. Generating Test Data for Functions with Pointer Inputs. In *Proceedings of the 17th IEEE International Conference on Automated Software Engineering02; 149-160/02,* 2002

[40] Eugenia Diaz, Javier Tuya, and Raquel Blanco. Automated Software Testing Using a Metaheuristic Technique Based on Tabu Search. In *Proceedings of the 18th IEEE International Conference on Automated Software Engineering03;* 310-313/03, 2003

[41] Stephen H. Edwards. A framework for practical, automated black-box testing of component-based software. *Software Testing, Verification and Reliability;* 11:97–111, 2001

[42] Neelam Gupta, Aditya P. Mathur, and Mary Lou Sofia. Automated Test Data Generation Using An Iterative Relaxation Method. In *proceeding of the 6th ACM SIGSOFT international symposium on Foundations of software engineering;* 231-244/98, FSE 1998

[43] Jon Edvardsson and Mariam Kamkar. Analysis of the Constraint Solver in UNA Based Test Data Generation. In *proceeding of the 9th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering2003;* Vol. 26, No 5, 237-245/03, ESEC/FSE-9 2001

[44] Andre Baresel, Mark Harman, David Binkley, and Bogdan Korel. Evolutionary Testing in the Presence of Loop–Assigned Flags: A Testability Transformation Approach. In *Proceedings of the International Symposium on Software Testing and Analysis;* Vol. 29, No. 04, 108-118/04 ISSTA 2004

[45] Bogdan Korel and Ali M. AI-Yami. Automated regression test generation. In *proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis '98,* 143-152/98, Vol. 23, No 02, ISSTA. 1998

[46] Nigel Tracey, John Clark and Keith Mander. Automated Program Flaw Finding using Simulated Annealing. In *proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis '98,* 73-81/98, Vol.23, No 02, ISSTA 1998

[47] Paulo Marcos Siqueira Bueno and Mario Jino. Identification of potentially infeasible program paths by monitoring the search for test data. In *proceedings of 15th International conference on Automated Software Engineering*, 209-218, ASE 2000

[48] Xiyang Liu, Hehui Liu, Bin Wang, Ping Chen, and Xiyao Cai. A Unified Fitness Function Calculation Rule for Flag Conditions to Improve Evolutionary Testing. In *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering,* 337-341 ASE. 2005

[49] L. du Bousquet and N. Zuanon. An Overview of Lutess A Specification-based Tool for Testing Synchronous Software. In *proceedings of 14th International conference on Automated Software Engineering*, 208-215, ASE 1999

[50] R. M. Hierons. Avoiding Coincidental Correctness in Boundary Value Analysis. *ACM Transactions on Software Engineering and Methodology,* Vol. 15, No. 3, 2006.

[51] Patrick J. Schroeder, Pankaj Bolaki, and Vijayram Gopu. Comparing the Fault Detection Effectiveness of N-way and Random Test Suites. In *Proceedings of the International Symposium on Empirical Software Engineering (ISESE'04),* 2004

[52] Man Xiao, Mohamed El-Attar, Marek Reformat, and James Miller. Empirical evaluation of optimization algorithms when used in goal-oriented automated test data generation techniques. *Empirical Software Engineering* 12:183–239, 2007

[53] McMinn P. Search-based software test data generation: a survey. Software Testing, Verification and Reliability 14(2):105– 156, 2004

[54] Taylor B.J and Cukic B. Evaluation of regressive methods for automated generation of test trajectories. In *Proceedings of 11th International Symposium on Software Reliability Engineering,* 97-109/2000. ISSRE 2000

[55] I. Sommerville. *Software Engineering,* 5th edn (Addison-Wesley, Wokingham, 1996).

[56] Bogdan Korel. Automated Test Data Generation for Programs with Procedures. In *proceedings of the 1996 ACM SIGSOFT international symposium on Software testing and analysis '96, 209*-215/96, Vol.21, No 03, ISSTA 1996

[57] Natalia Juristo, Ana M. Moreno, and Sira Vegas. Reviewing 25 Years of Testing Technique Experiments. *Empirical Software Engineering* 9:1-2, Pages 7-44, 2004

[58] Haichang Gao, Boqin Feng, and Li Zhu. A kind of SAaGA Hybrid Meta-heuristic Algorithm for the Automatic Test Data Generation. In *proceedings of IEEE International Conference on Neural Networks and Brain, ICNN&B '05*. Vol. 1, 111-114, 2005.

APPENDIX 1. PRIORITIZED LIST OF SEARCH ENGINES AND MOTIVATIONS

1. **ACM Digital Library:** Association for Computing Machinery (ACM) is the first society in computing. It is one of the authorities in providing software engineering standards. ACM digital library is a component of ACM portal. It is an authenticated source of on-line articles from renowned journals and proceedings related to computing machinery. It is evaluated here as the most likely, to the point, and readily available source of information about ATDG techniques. It is the primary source of exploration for one of the chosen journals and three top most conferences.
2. **IEEE Xplore:** Institute of Electrical and Electronics Engineers, Inc. (IEEE) is the world's leading professional association for the advancement of technology. Although it deals in all famous engineering disciplines but at the same time it is one of the authorities in providing software engineering standards. It is a publisher of famous magazines and journals. The IEEE Xplore is a famous portal among software engineers that provides access to articles from famous magazines, journals and proceedings. It is the primary source of exploration for three of the chosen journals and two conferences.
3. **Springer Verlag:** It is a famous multi disciplinary publisher, a mammoth source of required literary information, and web portal. It is a source of exploration for three of the chosen journals and two conferences.
4. **Science Direct:** It is web portal and host of one of the largest online collection of published scientific research. It was considered as the fourth most likely, precise and to the point source of information about ATDG techniques. It is a source of exploration for two of the chosen journals.
5. **Compendex:** It is a big database and comprehensive web portal. Its collaboration with Inspec and NTIS databases allows for searching on a broad range of updated topics within the scientific, applied science, technical and engineering disciplines. Compendex Engineering Village is the premier web-based discovery platform meeting the information needs of the engineering community.
6. **Google Scholar:** It is one of the most popular public search engines that provide access to a variety of literary resources. It provides very strong general search criteria.

7. **CiteSeer:** It is a missionary public search engine and digital library for scientific and academic papers. Its goal is to improve the dissemination and access of academic and scientific literature. It freely provides Open Archives Initiative metadata of all indexed documents and links indexed documents when possible to other sources of metadata.

APPENDIX 2. PRIORITIZED LIST OF PROBABLE SEARCH STRINGS/KEY WORDS FOR DIFFERENT SEARCH LEVELS

a) **<u>Search engine, Journal/Conference level</u>**

1. software testing
2. Automated testing
3. Automatic test data generation
4. Testing technique
5. Testing method
6. verification and validation

b) **<u>Paper level</u>**

1. Automated test data generation technique
2. Automatic test data generation technique
3. Automatic test data generation
4. Automatic test data
5. Automated test data
6. Test data generation
7. Test data creation
8. Test data
9. Test values
10. Test input

Note: Group-a above represents prioritized list of probable search strings/key words required to select search engines and journal/conference proceedings that are more likely to produce some better output during article search. On the other hand Group-b above represents prioritized list of probable search strings/key words required to find out articles related to ATDG techniques, from the selected search engines, journals/conference proceedings. The search string application schema 'From more specific to general' has been adopted on the bases that it is characterized to quickly highlight the relevant articles.

APPENDIX 3. LIST OF DESELECTED ARTICLES AND REASONS

The appendix includes articles which were selected in the first phase of the search but deselected in the next against cogent reasons.

Deselected Articles and Reasons for deselecting them

1) **2:** "Declarative Paradigm of Test Coverage" (STVR1998).

Reasons: The article is about logical programming and formal specification test input generation. This is quite different from that of conventional programming and the input is also very different from that of test data. This paper presents a new approach extending previous work by the authors on test input (not test data) generation for declarative programs. *The logic paradigm enables one to specify the problem (Kowalski, 1979). A logic program consists of facts and properties about a problem, but it is the task of the system to find the solution to the problem. Programming languages relying on the logic paradigm are also called declarative.*

2) **11:** "Automatic generation of assumptions for modular verification of software specifications" (JSS2006).

Reasons: The article is not related to ATDG techniques; instead it talks about automated generation of assumptions.

3) **20:** "A theory of probabilistic functional testing" (ICSE 1997).

Reasons: The article is not about automatic test data generation rather it talks about test data set selection on the bases of a functional theory.

4) **21:** "Specification-based testing of reactive software: tools and experiments: experience report" (ICSE 97).

Reasons: The article does not discuss ATDG rather it talks about automating the testing process. It generally talks about test oracles, test harness and test enabled application.

5) **22:** "Automated Support for Development, Maintenance, and Testing in the Presence of Implicit Control Flow" (ICSE2004).

Reasons: The article discusses manual test data generation: "*Our approach guides testers in (1) exploring test requirements to select a suitable coverage level, and (2) generating test data to exercise the selected test requirements*".

6) **23:** "Testing database transactions with AGENDA" (ICSE2005).

Reasons: The paper introduces a technique for checking complex properties of the database state transition performed by the transaction under test, as well as an improved input generation heuristic. Results of using AGENDA to test three applications with seeded faults are presented.

7) **27:** "Black-Box Test Reduction Using Input-Output Analysis" (ISSTA2000).

Reasons: The article is important with respect to testing terminology definitions but unfortunately it doesn't describe any ATDG technique but rather it says: "*We create tests by first selecting test data values using black-box test design techniques.*" It is totally silent about these techniques. In facts this article is discussing generation of test sets: *In this paper, we introduce an approach to test reduction that uses automated input-output analysis to identify relationships between program inputs and outputs.*

8) **32:** “Test Input Generation with Java Path Finder”.
Reasons: The article talks about test input generation e.g. in shape of java data structure for generating paths. It does not discuss ATDG. *Our framework can be used to automatically generate Java data structures from a description of method preconditions.*

APPENDIX 4. FORMATS OF DATA LOG AND ARTICLES COUNT LOG

Data log

Sr. No.: **DB ref.:** **Thesis ref.:**

Article/Book Title:

Journal/Conference: **Year:**

Author(s):

Main Area:

Key words:

Data of interest at page(s): e.g.

a) Good for ATDG Tech Programming (2,3,6)
b) Describes………

Articles count log

Sr. No.	**Journal/con. Proceeding**	**Year**	**Volume/No**	**Issue**	**Articles seen**	**Articles collected**	**Remarks**

APPENDIX 5. LIMITATIONS OF APPROACHES THAT COULD BE USED TO DESCRIBE ATDG TECHNIQUES

a) Describing ATDG techniques by addressing respective articles in a serial
   a. It is hard to define sequence criteria.
   b. It is hard to show relationships among techniques.
   c. Text redundancy will incur in case of describing relationships among techniques.
   d. It is hard to discuss and deduce results from findings from a serial arrangement.

b) Putting up various groups against types of automated test data generation
   a. It is hard to accommodate opposite testing categories which a single method is expected to adopt e.g., both BBT and WBT techniques can adopt dynamic method.
   b. Controlling other factors of classification like testing category or approaches for description will become cumbersome.
   c. It is hard to show and describe ATDG techniques which adopt mixing of contradictory approaches as reported by [14].

c) Against the issues the ATDG techniques addressed.
   a. It is hard to define all the issues concerning ATDG.
   b. It is hard to segregate techniques that address the dependent or closely linked issues.
   c. It is must that a technique is developed to address an issue of ATDG. But it looks awkward to put for comparison, techniques which adopt opposite or quite different methodology for the same issue.
   d. It is hard to describe when a comprehensive technique addresses more than one issue at a time.