Systematic Mapping Study in Automatic Test Case Generation

Shayma Mustafa MOHII-ALDEEN\textsuperscript{a,b,1,}, Safaai DERIS\textsuperscript{a}, and Radziah MOHAMAD\textsuperscript{a}

\textsuperscript{a}Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia
\textsuperscript{b}University of Mosul, Mosul, Iraq

Abstract. Test case generation is a painstaking task in software testing and has a strong influence on the efficiency and the effectiveness of software tests. It is an important subject in software testing research that has led to the development of several tools and approaches over the decades. This paper presented a systematic mapping study to get an overview about the current studies of distinct techniques for generation of test cases automatically. The techniques presented in this paper are random-based methods, search-based methods and data mining-based methods. Each technique is explored briefly to give the basic idea behind it. In general, the paper’s objective is to give an up-to-date introduction and short review of the research in generation of test cases automatically. Systematic mapping study is the process of finding and collecting as much literature as possible, provides a structure of the type of research reports and the results that have been published by categorizing them depending on specific search questions to provide a background for further research. This study was based on a comprehensive set of 85 papers published in conference and journals between 2002 and 2013 obtained after using multistage selection criteria in the field of automatic test cases generation. The results from our systematic mapping study include information about the researches techniques used to generate test cases automatically and types of coverage within a specific period that can help researchers in this field through providing an overview of the current researches in this area. Furthermore, it may serve as a first step towards a great explanation of the topic with the help of systematic literature review.

Keywords. Automatic test case generation methods, Software testing, Test case generation, Test case generation methods.

Introduction

Software testing is a necessary step in software development and an essential branch of software engineering. Nevertheless, testing is costly and labor intensive, accounting for more than 50\% of total software development cost. Thus, it is important to decrease cost and improve the software testing efficiency by automating the testing process [1].

\textsuperscript{1}Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia (Phone: 006-01-07165982, E-mail: shaima.mustafa76@yahoo.com).
Among the several testing activities, the generation of test cases is one of the most important and the most critical, as it can have a strong effect on the efficiency and effectiveness of the testing process [2][3][4]. Not surprisingly, in previous decades a great deal of research effort has been expended in generating test cases automatically.

A good set of test cases has a high chance of discovering unknown errors and successful testing is that which uncovered these errors [5]. To discover all possible errors in a software, comprehensive testing is required to check all potential input and execution paths, but it is impossible and not economical [6]. Therefore, the essential aim of software testing is to increase the capability of discovering errors by using a limited number of test cases, expending the least time with the least effort [7].

Several metrics have been developed for measuring the test case quality that has been generated such as time, effort, cost, coverage criteria and generation complexity. The aim of some researchers can be optimization or improvement in test case quality [8][9]. Optimization can take different forms, such as minimizing the testing time or effort, minimizing the cost or complexity of the methods of generation, maximizing the coverage criteria, and decreasing the tests cases generated or test data generated [10].

A test adequacy criterion provides a measurement of test-suite quality and can be used to guide test generation. There are three main kinds of coverage criteria: mutation coverage (to evaluate the fault-revealing capability of a test suite), code coverage (specify the percentage of the source code of the software that has been tested), and specification based coverage (specify the percentage of testing requirements identified in specifications that have been exercised by the test suite) [11].

Existing test case generation methods can be categorized into white-box testing and black box testing. White-box test cases are derived from the internal structure of the program [12]. Black-box test cases are derived from the program specification [13]. However, in both cases, it is difficult to achieve complete automation of the test case design [14].

This paper provides a systematic mapping study to present a broad overview of the primary studies on generation of test cases automatically in both white and black box testing since 2002. The motivation is the identification of the evidence available on the topic and identification of the research gap in automatic test case generation techniques.

Following the introduction, the structure of this paper is as follows: In Section 1, a short overview of the current study has been presented and its need justified. Section 2 describes the systematic mapping method that includes the research questions, search strategy, the inclusion and exclusion criteria, classification scheme, and systematic map. The results are discussed in section 3 and conclusions are presented in Section 4.

1. Background and Motivation

In this section, a short background to automatic test case generation is presented as well as the motivation behind it and the methods used to automate test case generation. This section also presented the need for a systematic mapping study. The goal of this study is to get an overview of existing researches on automatic test case generation techniques.

1.1. Automatic Test Case Generation methods

There is no search-based algorithm that exceeds all other algorithms [15]. In this way,
various algorithms should be compared when new problems arise. Thus, more suitable search algorithms can be presented. In this paper, the techniques presented include random-based methods, search-based methods and data mining-based methods.

Random Search (RS) is the simplest among the search algorithms and while it may give good coverage, the problem is the need of determine the length of the sequences that will be generated through the random search. If no constraint on the length has been determined, long sequences can be present that are useless and require long computational time [15]. RS is commonly used in the literature as a base line for comparing and understanding the effectiveness of the other search algorithms.

The search-based methods are classified into local search and global search based on the search space that they cover. Local search algorithms operate using a single current node and generally move only to the neighbors of that node. These methods have advantage in using little memory and converge to a solution in large spaces [16]. In this paper, three methods are used: simulated annealing (SA), hill climbing (HC), and Tabu search (TS).

Global search algorithms aim to exceed the local optimum problem in the search space and to find new global optimal solutions. Local search may become stuck in local optima within the solution space, but it can be more efficient for simple search problems [9]. In software engineering, this proved a clear example of the classic comparison between effectiveness and efficiency; one may suppose that global search achieved the branch coverage better than local search, but at the cost of higher computation effort [9]. The global based methods used in this paper are Genetic Algorithms (GA), Differential Evolution (DE), Scatter Search (SS), and swarm optimization techniques: Particle Swarm Optimization (PSO), Artificial Colony Optimization (ACO) and Artificial Bee Colony (ABC).

Finally, Memetic Algorithm (MA) is a hybrid algorithm that merges local and global searches together that may lead to improved results [9].

In Data Mining based, the main goal is analyzing the I/O of SUT to reduce the number of test cases by eliminating unimportant and unfeasible test cases. Being more efficient due to their intelligence, decreasing the number of test cases has a direct effect on saving software testing resources [12]. This method is highly complex and the efficiency of this method depends on the number of available training samples.

1.2. Need for Systematic Mapping Study

The systematic mapping study process was used to collect, classify, conduct thematic analysis and identify publication of the available published papers. A systematic mapping study process provides a structure of the type of research reports and results that have been published by categorizing them dependent on specific search questions designed to provide a background for further research [17].

Due to the significance of automated test case generation process in software testing and as the methods to generate it mature, there has been a large increase in the number of researches available. However, there remains a lack in the current literature on classification methods.

Although several studies have presented a literature review of test case generation, no existing research has conducted a systematic mapping study of this field. Therefore, it becomes significantly importance to summarize these researches and provide an...
overview of the current research in the field of automatic test case generation. This study has been conducted to fulfill this gap.

This study was based on a comprehensive set of 82 articles (66 of them on white-box testing and the rest on black box testing) this multistage selection process focused on conferences and journal articles released between 2002 and 2013.

2. Research method

A systematic mapping study is a process of identifying and classifying research of as much existing literature as possible related to a topic of interest. When used in a specific field, classifying different types of research reports in several dimensions, it often yielded a summary and a map of its results. These studies have been recommended mostly for research areas where there is a little relevant evidence found during primary studies of the domain and for very broad topics. Conducted at a coarse-grained review, the objective is only to find and identify the evidence related to research questions, and to identify research gaps that may help in direct future research [17].

In this study, we conduct a systematic mapping study of automatic test case generation since it seems to be a broad topic with various researches focus fields. However, no existing research has conducted a systematic mapping study of the area of automatic test case generation. This section characterizes the review protocol.

Petersen et al. describe the process of systematic mapping study in software engineering. They considered that the mapping process consists of three essential activities: search for relevant publications, definition of a classification scheme, and mapping of publications [17]. Our mapping study performed the fundamental process steps of defining the questions of research, defining the strategy of search, selecting of primary studies, classification of the scheme and systematic map.

2.1. Research Questions

The goal of this study is to obtain a display of the current researches in the area of automatic test case generation by way of the following questions. These questions aim to give an overview about the techniques used in generated test cases automatically. The overall goal is defined in these research questions:

RQ1: What types of testing and coverage criteria have been done?
RQ2: What are several methods used for automatic test case generation?
RQ3: What are the benefits of using various methods?

2.2. Search Strategy

The search strategy was based on the identification of alternative words and synonyms of terms used in research questions to decrease the effect of the differences in terms. The research scope is to show a wide view of systematic study of the researches done in the generation test case field especially those related to search based automatic test case generation within the period 2002 to 2013.
In terms of search method, we used both manual and automatic search. Manual search means manually browsing journals and conference proceedings. Automatic search, on the other hand, means searching main electronic sources of data using a combination of previously determined search strings. Manual searching was performed by selecting the following venues because a number of studies of automatic test case generation were found there during the search.

- Journals:

- Conferences:

The automatic search has been done by using the search string that is given in Table 1 which represents the basic concepts related to test case generation and automatic test case generation methods. The search was done within the following digital sources: Google Scholar, IEEEXplor, Science Direct, ACM Digital Library, Springer, and Wiley journals.

2.3. Study Selection Criteria for Inclusion and Exclusion of Primary Studies

As explained previously, we used both manual and automatical search. We start the search by using the search string in Table 1 within digital libraries given previously to determine the initial set of publications in this area within the time span 2002-2013.

<table>
<thead>
<tr>
<th>Search string</th>
<th>Alternate words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case generation</td>
<td>(test case AND generation) OR (generate AND test cases)</td>
</tr>
<tr>
<td>Automatic test case generation methods</td>
<td>(automatic AND test case methods) OR (generated AND test cases methods AND automatically)</td>
</tr>
<tr>
<td>Search based test case generation</td>
<td>(search based test case generation methods) OR (Metaheuristic test case generation methods)</td>
</tr>
</tbody>
</table>

This initial set of publications can be used to determine which journals and conference proceedings are related to our study. These journals and conferences were used to find more publications in test case generation, and from these publications, we read introductions and related works which increased the number of publications related to our study. After performing automatic search, duplicate publications are excluded by comparing the results obtained automatically with those obtained manually.

A list of all inclusion and exclusion criteria of the studies is given below.

Inclusion:
- Studies that present an introduction of test case generation.
- Studies that present the test coverage criteria.
- Papers that present an automatic test case generation method or approach.
- Papers that present white box and black box test case generation.
- Papers that present search-based test case generation.
Exclusion:
- Duplicate papers of the same study in different resources.
- Papers that used random-based only to generate test cases because it is commonly used in literature as a base line for comparing with other search-based methods.
- Papers which stated test case generation in abstract only.
- Editorials, books, reports, tutorial summaries and other non-peer review publications.
- Non-English papers.

2.4. Classification scheme

In this study and after selecting the publications related to study we found 85 articles of the present works in automatic test case generation methods for white-box and black box testing. Of these, 66 are white box testing, while the rest are black box testing. White box and black box testing have different methods to automate the test case generation. These methods are classified into three categories including random-based methods, search-based methods and data mining-based methods as seen in Fig. 1. We excluded the random-based methods from the study as these methods are used as a basis of comparison with other methods. Table 2 contains the methods that are used in both white box and black box test case generation with the number of publications.

![Figure 1. Automatic Test Case Generation Methods.](image-url)
Table 2. Methods used in Automatic Test Case Generations

<table>
<thead>
<tr>
<th>Methods</th>
<th>White box</th>
<th>References</th>
<th>Black box</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genetic Algorithm</td>
<td>20</td>
<td>[18]-[37]</td>
<td>7</td>
<td>[38]-[44]</td>
</tr>
<tr>
<td>Particle Swarm Optimizat</td>
<td>9</td>
<td>[45]-[53]</td>
<td>2</td>
<td>[54]-[55]</td>
</tr>
<tr>
<td>Hybrid GA and PSO</td>
<td>4</td>
<td>[56]-[59]</td>
<td>2</td>
<td>[60]-[61]</td>
</tr>
<tr>
<td>Scatter Search</td>
<td>3</td>
<td>[7][14][62]</td>
<td>2</td>
<td>[63][64]</td>
</tr>
<tr>
<td>Memetic Algorithm</td>
<td>8</td>
<td>[6][53][65][70]</td>
<td>-</td>
<td>[66][67]</td>
</tr>
<tr>
<td>Ant Colony Optimization</td>
<td>6</td>
<td>[71]-[75]</td>
<td>1</td>
<td>[76]</td>
</tr>
<tr>
<td>Artificial Bee Colony</td>
<td>4</td>
<td>[77]-[80]</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Differential Evolution</td>
<td>1</td>
<td>[81]</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Tabu Search</td>
<td>4</td>
<td>[77][82][84]</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Simulated Annealing</td>
<td>3</td>
<td>[85][86]</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Hill Climbing</td>
<td>2</td>
<td>[16][27]</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Data Mining</td>
<td>2</td>
<td>[87][88]</td>
<td>5</td>
<td>[89][93]</td>
</tr>
<tr>
<td>Sum of studies</td>
<td>66</td>
<td></td>
<td>19</td>
<td></td>
</tr>
</tbody>
</table>

2.5. Systematic Map

The results analysis focuses on presenting the publication frequencies for each classification. This map made it possible to see which classification have been undertaken in past researches, and so to identify the gaps which may help to identify the possibility of future research. The results of mapping study are presented in section 3.

3. Discussion of research questions

This section presents the results of mapping study related to the research questions. 
RQ1: What type of testing and coverage criteria has been done?

There are two types of testing that will be covered in this systematic mapping study: white-box testing and black box testing, as well as the type of coverage criteria depending on the type of testing that it is classified into: code coverage, specification-based coverage and mutation coverage [10] as explained previous. The coverage types studied in literature are shown in Fig. 2. Most researchers focus on code coverage because it is more difficult and costly to generate test cases [1].

Many techniques have been used for many years to generate test cases automatically depending on the test adequacy criterion.

![Coverage Types](image)

**Figure 2.** Test Coverage Criteria Percentages.
Based on the literature, Fig. 3 shows the bubble plot of the systematic map for test coverage types which showing the frequencies of publications in each category, each bubble contains the number of articles that are in the pair of categories corresponding to the bubble coordinates. The right side of the figure consists of the number of researches with the specified method that is used in a determined coverage type and the left side contains the number of researches that used different coverage types in different years from 2002 to 2013.

This study focuses on code coverage that characterizes the test percentage of the source code program that has been tested. Various kinds of code coverage criteria have been proposed, such as control-flow coverage criteria and data-flow coverage criteria. Different from control-flow coverage criteria that are based on the control-flow graph, data-flow coverage criteria are based on the definitions and use of variables. The control-flow coverage criteria contain statement coverage, condition coverage, path coverage and branch coverage [5].

Fig. 4 shows the coverage criteria types that are used in literature for white box testing technique. Most researches focus on branch coverage, which is the most widely used coverage in structural testing as well as path coverage.
RQ2: What are the various methods used to generate test case automatically?

There are several methods used to generate test cases automatically including: Random based methods, Search based methods and Data mining based methods as shown in Fig. 1. As explained previously, random based is used to evaluate the other methods, while search based methods are classified into local search and global search depending on the search space.

In local search, the methods used in test case generation are: Simulated Annealing (SA) [27][85][86], Hill climbing (HC) [16][27] and Tabu Search (TS) [7][82]-[84]. For global search, the methods used are: Genetic Algorithms (GA) [18]-[37] for white box and for black box testing [38]-[44], Scatter Search (SS) is also used in generated test cases for white box testing [7][62] and for black box testing [63][64], Particle Swarm Optimization (PSO) is used to generate test cases for white box testing [45]-[53] and for black box [54][55]. Furthermore, Artificial Colony Optimization (ACO) is used for white box testing [71]-[75] and [76] for black box testing, while Artificial Bee Colony (ABC) is used for white box in [77]-[80] and Differential Evolution (DE) in [81]. The final method is Memetic Algorithm (MA), that combines the local search with global search to get better solutions for white box testing [9][14][65]-[70]. Data mining based method (DM) is more efficient than other methods but it is complex to execute. DM is also used for generating test cases for white and black box testing [86]-[92].

Fig. 5 and Fig. 6 show the relative volume of automatic test generation technique studies in reviewed literature for White-Box and Black Box testing. Genetic Algorithm (GA) had the most published studies making it the most studied method with about a third (33% and 35% respectively) of the total reviewed literature for White-Box and Black Box.

Particle Swarm Optimization (PSO) followed with a 15% share of total published studies for the White-Box while Black Box, Data mining (DM) accounted for 26% of total published studies to rank second. Memetic Algorithm (MA) and Scatter Search (SS) subsequently placed third with 12% and 11% of published studies for White-Box and Black Box respectively. Interest in hybrid studies using both the combined GA+PSO techniques shows greater interest (11%) in Black-Box studies than in similar studies (6%) for White-Box testing.

![Test case Generation Methods in White-Box Testing](image)

*Figure 5. Automatic Test Case Generation Methods for White Box Testing*
After describing the automatic test case generation methods in the previous literature on the optimization of test suite, which generates maximum coverage percentage to cover the efficient test cases with minimum size of test suite and minimum time and cost, a summary of the comparison between methods is shown in Table 3. The table contains the overall main idea of the methods that have been classified by the type of method in the previous studies, which are Random-Based, Search-Based and Data Mining-Based, and their advantages and disadvantages, as well as the level of their complexity and application, including the type of test adequacy criteria that can be covered.

Table 3. Comparison between Automatic Test Case Generation Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Random-Based</th>
<th>Search-Based</th>
<th>Data mining-Based</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main Idea</strong></td>
<td>Test cases and events sequences are generated randomly.</td>
<td>In these approaches the problems of generating the test cases are regarded as an optimization problem, and try to find optimize solution including best test set for the problem under test.</td>
<td>The main goal is analysing the I/O of SUT to reduce the number of test cases by eliminating unimportant and infeasible test cases.</td>
</tr>
<tr>
<td><strong>Advantages</strong></td>
<td>1. Easy to implement. 2. Low costs. 3. Works well for simple programs. 4. Good at exercising systems when the source code and the specifications of the program under test are not available or incomplete. 5. Bringing randomness to software testing process which reflects in best way the irregularity of the system environment. 6. Simply generating a huge number of test cases automatically.</td>
<td>1. The results show the efficiency of these methods. 2. Speed. 3. Reduces time. 4. Minimize cost. 5. Intelligent path search. 6. Optimize test cases generation.</td>
<td>1. Being more efficient, as a result of being more intelligent. 2. Decreasing the test case number that directly results in saving software testing resources. 3. Avoid redundancy and low time consumption.</td>
</tr>
</tbody>
</table>
There are many search-based methods used to generate test cases automatically with a main goal of maximizing the selected coverage with minimum test suite size and minimum time. GA is the most popular method in test case generation. Local search methods like HC, SA, TS are simple and offer better runtime than RS and are more efficient for simple problems, but they can be trapped in local optima. Although HC gives high coverage better than GA it required the generation of a larger number of test cases (long test suite) to provide high level coverage and more execution time. It is also highly dependent on the starting point which has an effect on efficiency. On the other hand, SA is less dependent on the starting point and easy to code even for complex programs. SA gives a good solution and provides better coverage than HC but it is very slow and cannot tell if the solution is optimal. TS is effective in coverage especially in branch coverage. TS decreased test suite size better than SS but required more long term memory to avoid local optima trap. Short term memory is required for remembering all test cases in the search. In addition, SA needs more execution time when compared with SS.

Global search methods like GA, DE, SS, MA, PSO, ACO, ABC outperform the problems of local search methods and find more global optimal solutions, but they are more difficult and complex. SS is more efficient than local search but TS is better in coverage percentage. In addition, it cannot cover the unfeasible branches. MA is more efficient than local or global search alone and is capable of providing the best overall performance in terms of coverage compared to local search but it failed to cover new branches, and consumes greater computational cost.

Although GA is the most popular and performs efficiently in terms of execution time, number of generations and coverage percentage, in many complex problems it may move to local optima instead of global optimum in terms of coverage which makes it slow. In addition the fitness function is an essential factor that effects the efficiency and speed of the algorithm. DE is simple and more efficient with coverage percentage than GA but the parameters need to be well set.
PSO convergence is fast and requires short computation time, which can effectively improve the deficiencies of GA which is slow convergence. However it suffers from premature which causes the convergence problem that effects the speed and direction. To outperform the problems with GA and PSO a hybrid algorithm GA+PSO has been proposed but this hybrid method needs more execution time than PSO. In addition, the value of fitness affects the efficiency of the algorithm and in many problems, it gives the same coverage percentage as GA and PSO alone.

ACO is a more modern approach and is seen to have a better success rate. However, although it can reduce the test suite size it has problems which include high length of test sequences and duplication of nodes during the same sequence without any benefit on test adequacy criteria. In addition, the need to update pheromone value affects efficiency and the time required. ABC is better than the other methods in optimization and it outperforms other methods in terms of reducing the number of generations and time, but it still needs to improve the coverage percentage. In addition, the value of colony size has an effect on the method’s performance.

RQ3: What are the benefits of using various methods?

The benefits of using these methods include reducing the manual testing cost and increasing the testing reliability. Researchers have attempted to automate it and automatic test case generation represents one such attempt. The metrics that were used for measuring the test case generation quality include reducing testing time or effort, reducing the generation algorithm complexity or cost, increasing the coverage and reducing the generated test cases. Fig. 7 shows the relative volume of research publications focused on several motivations using both white and black box testing techniques.

Research interest in reduction of test case generation tops the list as the dominant research concern with almost half (44%) of the total volume of published studies reviewed. Studies in reduced time, fault detection and reduced costs respectively complete the hierarchy of research interest using the two techniques. The figure shows that research studies in reduced costs takes up the rear implying less focus of current research on cost implications of testing.

![Figure 7. White and Black box Testing Evaluation Metrics](image-url)
Testing performances were evaluated by these metrics, so all researches dealing with automatic test case generation focus on optimization of these metrics and the objectives are to maximize the coverage percentage in minimize test suite size and to reduce time and cost, as well as the capability of fault detected. The systematic map of the testing performance metrics is illustrated using summary statistics which showing the frequencies of publications in each category. In our study we used a bubble plot to report the frequencies shown in Fig. 8, each bubble contains the number of articles that are in the pair of categories corresponding to the bubble coordinates. The right side of figure shows the number of researches that focus on these metrics with a specified method, and the left side shows the number of researches that have been focused on in different years. The bubble plot is more powerful in giving a quick overview of a field, and thus to provide a map.

![Figure 8. Visualization of Systematic Map for Testing Performance Metrics in the Form of a Bubble Plot](image)

### 4. Conclusions

This paper presented a systematic mapping study to obtain an overview of the existing researches in the field of automatic test case generation that is one of the most critical process of software testing as it can have a strong effect on the efficiency and effectiveness of the testing process and the motivation to automate it. We identified 85 papers published between 2002 and 2013.

Several methods have been used from time to time to generate test cases automatically in order to increase the coverage percentage of program testing and to reduce the time and cost required for testing process. These methods have been divided into random-based, search-based and data mining based. Description and comparisons are made between them in this paper.

This study presents initial findings related to automatic test case generation; the analysis of these studies have been presented in terms of type for test coverage, applied methods in generated test cases depending on type of testing, testing evaluation metrics, and the coverage criteria of white box testing. From the analysis, we have determined that most researches, 65% of studies have dealt with code coverage since it is more difficult and costly, providing the answer to our first research question. To respond to our second research question, we found that different approaches have been applied to
generate test cases automatically, however GA is the most frequently used and 30% of studies involve white box testing and 36% involve black box testing. As far as answering the third research question, a large percentage, 45% of the studies, deal with reducing test suite size among the testing evaluation metrics.

In future works we can extend the results of this systematic mapping study into a systematic literature review.

5. Acknowledgment

- We would like to thank Universiti Teknologi Malaysia for providing the facilities and support for the research.
- We would like to thank Ministry of Higher Education & Scientific Research of Iraq and University of Mosul for sponsoring the research.

References


A. S. Ghiduk, “A New Software Data Intelligence”, In M. Al-Fayoumi, P. Automation Engineering (CSAE), Shanghai, 10-12 June 2011, pp. 38-42.


