JCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-8, Aug 2018 E-ISSN: 2347-2693

Swarm Intelligence Based Automated Testing for MTAAS

R. Sharma[1*], S. Singh[2]

[1, 2]Computer Science Department, Deenbandhu Chhotu Ram University of Science and Technlology, Murthal, Sonipat, Haryana, India

*Corresponding Author: 2607rsharma@gmail.com

Available online at: www.ijcseonline.org

Accepted: 13/Aug/2018, Published: 31/Aug/2018

***Abstract*—** As Search in testing is a most time consuming task which takes approximately 60% work load of the total software development time. If the testing is performed using automated testing then it will lead to reduce in software development cost by a significant margin. Metaheuristic search based testing techniques have been extensively used to automate the process of generating test cases and thus providing solutions for a more cost-effective testing process. Mobile Testing as a Service (known as Mobile TaaS a.k.a MTAAS) provides on-demand testing services for mobile applications and/or SaaS to support software validation and quality engineering processes by leveraging a cloud-based scalable mobile testing environment to assure pre-defined given QoS requirements and service-level-agreements (SLAs) . Most MTAAS are managed in an ad-hoc way with very limited mobile test automation tools. This approach offers the benefits of in-the-wild testing without the need to invest in a lab or purchase or rent devices, but at the risk of low testing quality and an uncertain validation schedule. In this paper a methodology is discussed based on Clonal Selection Based Optimization Approach that utilizes Crowed Sourcing.

***Keywords*—** Software testing , Search based testing ,Crowed Sourcing, Swarm Optimization

I. INTRODUCTION

With the increase in the number of mobile users, more mobile devices are shipped daily and more mobile APPs and applications are deployed and launched to give diverse mobile application service need.
As mobile APP and mobile application vendors, they have met the following critical issues.

1. Testing mobile APPs on different mobile platforms and various devices becomes very costly and tedious due to the fast upgrading of mobile devices, and rapid updates of mobile platforms and technologies
2. Testing mobile web applications on diverse mobile browsers on different devices become very difficult and costly due to a) increasing mobile scalability; b) constantly updates of mobile devices and mobile technologies; c) diversity of involved mobile web technology; and d) fast upgrading mobile application services.

II. RELATED WORK

A lots of work has been done in MTAAS in the previous years. In this section, the related work done by the other researchers will be discussed.
Malden A Vouch 2008 [1] "Cloud" computing –"On decades of research in virtualization, distributed computing, utility computing and more recently networking, web and software services. It implies a service oriented architecture, reduced information technology overhead for the end-user, great flexibility, reduced total cost of ownership, on-demand services and many other things. This paper discusses the concept of "cloud" computing, some of the issues it tries to address, related research topics and a "cloud".

A.I. Avetisyan, R. Campbell, I. Gupta 2010 [2] Open Cirrus is a cloud computing testbed that, unlike existing alternatives, federates distributed data centers. It aims to spur innovation in systems and applications research and catalyze development of an open source service stack for the cloud.

"**Wei-Tec Tsai, Qigong Shao 2010 [3]** Software-as-as-Service (SaaS) is a new approach for developing software and it is characterized by its multi-tenancy architecture and its ability to provide flexible customization to individual tenant. However, the multi-tenancy architecture and customization requirements have brought up new issues in software, such as database design, database partition, scalability, recovery and continuous testing. This paper proposes a hybrid test database design to support SaaS customization with two-layer database partitioning.

K. Mao, L. Capra, M. Harman, and Y. Jian 2017 [4] the term 'crowdsourcing' was initially introduced in 2006 to describe an emerging distributed problem-solving model by online workers. Since then it has been widely studied and practiced to support software engineering. In this paper they provide a comprehensive survey of the use of crowdsourcing in software engineering, seeking to cover all literature on this topic. They first review the definitions of crowdsourcing and derive their definition of Crowdsourcing Software Engineering together with its taxonomy. Then they summaries industrial crowdsourcing practice in software engineering and corresponding case studies.

K. Mao, Y. Yang, Q. Wang 2015 [5] Crowdsourced software development utilizes an open call format to attract geographically distributed developers to accomplish various types of software development tasks. Although the open call format enables wide task accessibility, potential developers must choose from a dauntingly large set of task options (usually more than one hundred available tasks on Top Coder each day). Inappropriate developer-task matching may lower the quality of the software deliverables. In this paper, they employ content-based recommendation techniques to automatically match tasks and developers.

Dolata, R. Vliegendhart, and J. 2013 [6] Graphical user interfaces are difficult to test: automated tests are hard to create and maintain, while manual tests are time-consuming, expensive and hard to integrate in a continuous testing process. In this paper, they show that it is possible to crowdsource GUI tests, that is, to outsource them to individuals drawn from a large pool of workers on the Internet, by instantiating virtual machines (VMs) running the system under test and letting testers access the VMs through their web browsers.

R. Vliegendhart, E. Dolata, and J. Powles 2012 [7] Conducting a conventional experiment to test an application's user interface in a lab environment is a costly and time-consuming process. In this paper, they show that it is feasible to carry out A/B tests for a multimedia application through Amazon's crowdsourcing platform Mechanical Turk involving hundreds of workers at low costs.

Wei-Tec Tsai, Pride Zheng, J. Balasooriya 2011 [8] as cloud services proliferate, it becomes difficult to facilitate service composition and testing in clouds. In traditional service-oriented computing, service composition and testing are carried out independently. This paper proposes a new approach to manage services on the cloud so that it can facilitate service composition and testing. The paper uses service implementation selection to facilitate service composition similar to Google's Guise and spring tools, and apply the group testing technique to identify the oracle and use the established oracle to perform continuous testing for new services or compositions.

Stefan Baucus, Vlad Urschel, and Cristian 2011 [9]"This paper introduces Cloud, a platform for automated testing of real-world software. "Our main contribution is the scalable parallelization of symbolic execution on clusters of commodity hardware to help cope with path explosion. "Cloud provides a systematic interface for writing "symbolic tests" that concisely specify entire families of inputs and behaviors to be tested, thus improving testing productivity.

Xiaofei Zhang, Hui Liu, Bin Li, Xing 2010 [10] the emergence and application of cloud computing can help users access to various computing resources and services more conveniently. However, it also brings forth many security challenges. This paper proposes the application oriented remote verification trust model, which is capable of adjusting the user's trust authorization verification contents according to the specific security requirements of different applications.

III. METHODOLOGY

This section is related to the proposed methodology based on clonal selection algorithm.

Clonal selection Algorithm is stimulated by using the clonal choice idea of acquired immunity, formerly called CSA. Anew clonal choice inspired class set of rules is referred to as CSCA. CLONALG evokes some functions from clonal selection principle, which is mentioned above. The purpose here is to develop a reminiscence pool containing quality antigen matching antibodies that constitute a solution to engineering troubles.

The algorithm provides two searching mechanism for the desired final pool of memory antibodies. These two are as follows:

1. Local search, provided via affinity maturation of cloned antibodies. More clones are produced for better-matched antibodies.
2. A search that provides a global scope and involves the insertion of randomly generated antibodies to be inserted into the population to further increase the diversity and provide a means for potentially escaping local optima.

In Initialization step, an antibody pool is created based on randomly selected antigen which has size S. In loop phase, there may be a Selection and Pruning step that suggests and scoring the antibody pool to each antigen set, which can be both correct type score and misclassification score.
Proposed algorithm is shown below:

Algorithm Clonal Selection Algorithm for Crowd based MTASS

"**Input**: Population$_{Size,}$ Selection$_{Size}$, Problem Size, Random Pop$_{n}$, Clone rate, Mutation$_{rate}$

```
Output: Population
•  Population = CreateRandomPopulation(Selection_Size)
•  While (~Stop Condition())"
•  "For ( p_I ∈ Population)
      Affinity (p_I)
•  End"
•  Population_Select = Select(Population, Selection_Size)
•  Population_clones = Null
•  For ( p_I ∈ Population)
      Population_clones = Clone (p_I, Clone_rate)
•  End
•  For ( p_I ∈ Population)
      Hyper mutate (p_I, Mutation_ate)
      Affinity (p_I)
•  End
•  Population_Select = Select(Population, Population_clones , Selection_Size)
•  Population_rand = createRandomPolulation(Population, Random Pop_n , Selection_Size)
•  Select(Population, Population_clones , Population_rand)
•  Return (Population)
```

Steps of the methodology are as per below:

1. Data Collection: - One needs data for the testing process. This data could be generated or collected from some source. Here the data set is collected from google play store.

2. Pre-processing: - Data collected from google play store have so much details. Some of details about the apps are not need according to the requirement. So it is necessary to clean out that some details form data set and to prepare a clean data set.

3. Processing: - Here algorithm selected for testing process is applied on the clean data set.

4. Evaluation : - After applying the algo, one need to check the testing algo according to parameters selected for the testing. According to that parameters result are generated.

5. Stop criteria: - if the result generated is satisfying the stopping condition then the process is stop otherwise it is repeated in loops.

Fig below shows the process in details:

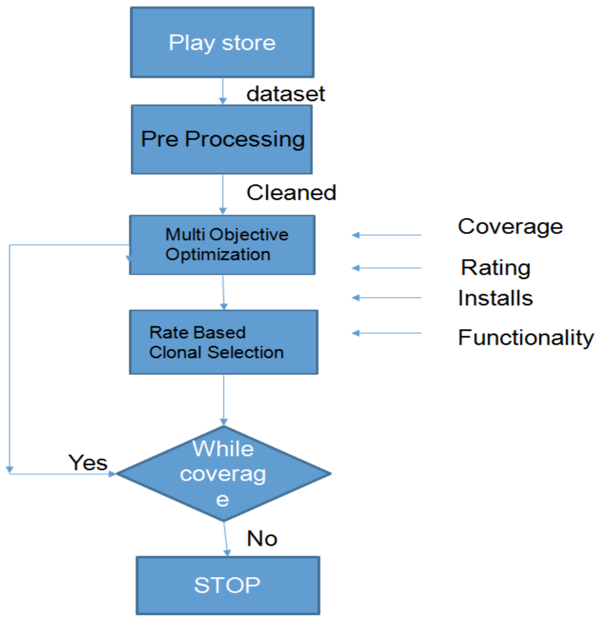


Figure 1: Flow chart (Proposed Algorithm)

IV. Results and Discussion

In this section, results of this paper is discussed. As day by day, development of mobile applications are increasing. In this paper, we focus on coverage and fitness values.
In the dataset first of all, active development in last 8 years is studied and a graph is plotted according to frequency of app development.
Fig 2 show this in details: -

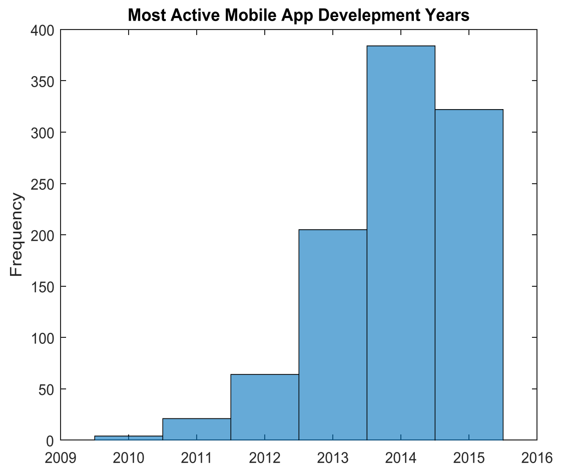


Figure 2: Most Active Mobile App Development Years identified in the dataset

As the use of technology is increasing, use of mobile apps is also increasing. Downloading of apps depends upon use of apps. Most usable app are downloaded frequently.

Fig 3 shows most downloaded apps in dataset.

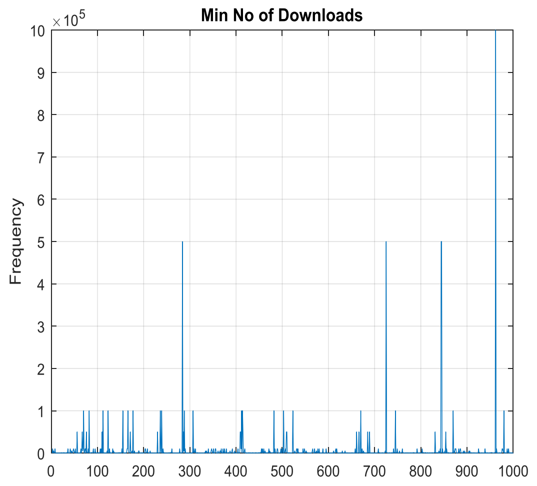


Figure 3: Most Downloaded Mobile Apps identified in the dataset, it can be seen easily that some apps are way more actively downloaded than others

Further in this paper, coverage criteria is studied. **Coverage** is defined as a technique which determines whether our **test** cases are actually covering the application code and how much code is exercised when we run those **test** cases.

Fig 4 show the coverage: -

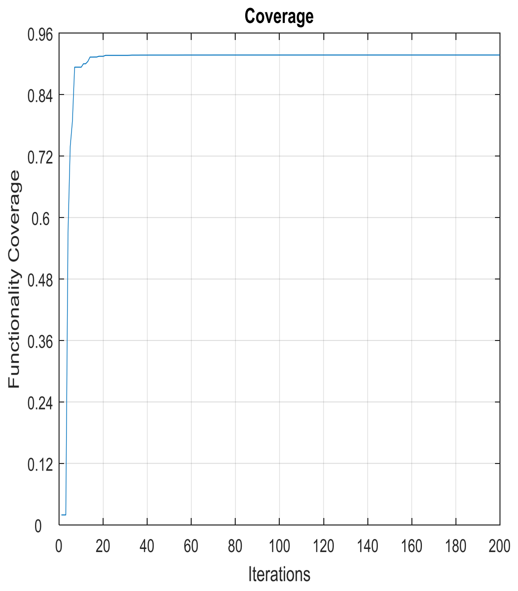


Figure 4: Coverage after applying proposed algorithm

Further fitness values are plotted in graph. Fig 5 shows the fitness values after iterations.

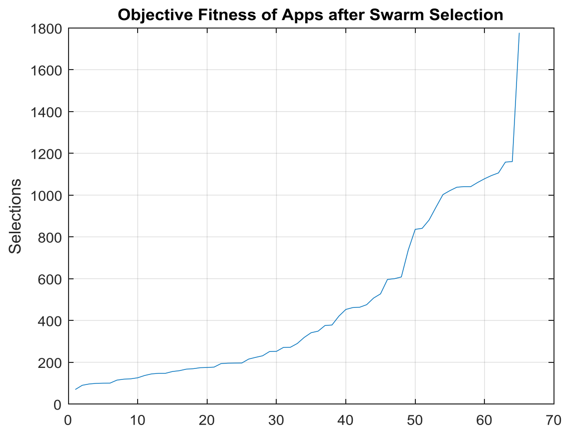


Figure 5:Objective fitness of apps after swarm selection

V. CONCLUSION AND FUTURE SCOPE

Crowd testing is a dynamic testing state of affairs wherein a crowd is concerned mostly with output from given specific inputs because they don't recognize or see the supply code . Crowd testing gives several possibilities for coping with today's software- testing challenges. Software pleasant is now not decided via functional accuracy; user experience, localization and lots of other dimensions have become even more important. The crowd-primarily based testing approach includes the use of freelance or shrunk checking out engineers or a network of end customers alongside a crowd-based checking out infrastructure and a provider control server to support diverse customers.
An avenue for future research issues the extraction of non-technical information from app shops, and extracting samples of apps (cognizant of the App Sampling Problem). Cross-keep studies are also a road for future research.

REFERENCES

[1] Malden A Vouch. Cloud computing: Issues, research and implementations. ITI 2008 30th International Conference on Information Technology Interfaces, 16(4):31–40, 2008.
[2] A.I. Avetisyan, R. Campbell, I. Gupta, M.T. Heath, S.Y. Koi, G.R. Ganger, M.A. Couch, D. O'Halloran, M. Kenzie, T.T. Kwan, K. Lai, M. Lyons, D.S. Milojicic, Hang Yan Lee, Yang Chai SoHo, Ng Kwan Ming, J-Y. Luke, and Han Neigong. Open cirrus: A global cloud computing testbed. Computer, 43(4):35 –43, April 2010.
[3] Wei-Tec Tsai, Qigong Shao, Yu Huang, and Xiaoping Bai. Towards a scalable and robust multi-tenancy SaaS. In Proc. of the Second Asia Pacific Symposium on Internet ware, pages 8:1–8:15, New York, NY, USA, 2010.
[4] K. Mao, L. Capra, M. Harman, and Y. Jian, "A survey of the use of crowdsourcing in software engineering," Journal of Systems and Software, vol. 126, pp. 57 – 84, 2017.
[5] K. Mao, Y. Yang, Q. Wang, Y. Jian, and M. Harman, "Developer recommendation for crowdsourced software development tasks," pinprick. Of SOSE'15, 2015, pp. 347–356.

[6] Dolata, R. Vliegendhart, and J. Powles, “Crowdsourcing GUI tests,” in Proc. of ISSTA’13, March 2013, pp. 332–341.

[7] R. Vliegendhart, E. Dolata, and J. Powles, “Crowdsourced user interface testing for multimedia applications,” in Proc. of CrowdMM’12, 2012, pp. 21–22.

[8] Wei-Tec Tsai, Pride Zheng, J. Balasooriya, Minong Chen, Xiaoping Bai, and J. Elton. An approach for service composition and testing for cloud computing. In Autonomous Decentralized Systems (ISADS), 2011 10th International Symposium on, pages 631 –636, March 2011.

[9] Stefan Baucus, Vlad Urschel, Cristian Zafar, and George Candia. Parallel symbolic execution for automated real-world software testing. InProc. Of The Sixth Conference on Computer Systems, pages 183–198, New York, NY, USA, 2011.

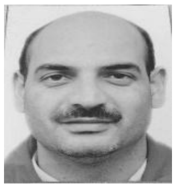
[10] Xiaofei Zhang, Hui Liu, Bin Li, Xing Wang, Haitian Chen, and Shushing Wu. Application-oriented remote verification trust model in cloud computing.2010 IEEE Second International Conference on Cloud Computing Technology and Science, pages 405–408, 2010.

[11] Sarang, R. P., & Bunkar, R. K. (2013). Study of Services and Privacy Usage in Cloud Computing. International Journal of Scientific Research in Computer Science and Engineering, 1(6), 7-12.

[12] Palve, A., Sonawane, R. D., & Potgantwar, A. D. (2017). Sentiment Analysis of Twitter Streaming Data for Recommendation using, Apache Spark. International Journal of Scientific Research in Network Security and Communication, 5(3), 99-103.

Authors Profile

R. Sharma pursed Bachelor of Technology from BPSMV,Khanpur Kalan,Sonipat in 2016. She is pursuing Master of technology in Computer Science and Engineering from Deenbandhu Chhotu Ram University, Murthal, Sonipat, Haryana, India. Her research are is Cloud computing ,mobile testing and software engineering.



Dr. Sukhdeep Singh pursed Bachelor of Engineering in Computer science & Engineering from Maharishi Dayanand University Rohtak, Haryana India in year 1999. He pursed Ph.D. from Maharishi Dayanand University Rohtak, Haryana India in year 2013. He is currently working as Associate Professor in department of Computer Science & Engineering at Deenbandhu Chhotu Ram University of Science and Technology, Murhtal, Sonipat, Haryana India since 2002. He is a life member of Indian Society of Technical Education (ISTE). He has published more than 30 Research papers in International Journals and Conferences of repute. His research areas includes Software testing, Software Quality and Computational Intelligence. He has 19 years of teaching experience and 5 year of research experience.