

GLOBAL JOURNAL

OF COMPUTER SCIENCE AND TECHNOLOGY: C

Software & Data Engineering

Circular Microstrip

Kanban based Scheduling

Highlights

Design and Analysis

Rhythm Sequence Technique

Discovering Thoughts, Inventing Future

VOLUME 17 ISSUE 1 VERSION 1.0



GLOBAL JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY: C
SOFTWARE & DATA ENGINEERING



GLOBAL JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY: C
SOFTWARE & DATA ENGINEERING

VOLUME 17 ISSUE 1 (VER. 1.0)

© Global Journal of Computer Science and Technology. 2017.

All rights reserved.

This is a special issue published in version 1.0 of "Global Journal of Computer Science and Technology" By Global Journals Inc.

All articles are open access articles distributed under "Global Journal of Computer Science and Technology"

Reading License, which permits restricted use. Entire contents are copyright by of "Global Journal of Computer Science and Technology" unless otherwise noted on specific articles.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without written permission.

The opinions and statements made in this book are those of the authors concerned. Ultraculture has not verified and neither confirms nor denies any of the foregoing and no warranty or fitness is implied.

Engage with the contents herein at your own risk.

The use of this journal, and the terms and conditions for our providing information, is governed by our Disclaimer, Terms and Conditions and Privacy Policy given on our website <http://globaljournals.us/terms-and-condition/menu-id-1463/>

By referring / using / reading / any type of association / referencing this journal, this signifies and you acknowledge that you have read them and that you accept and will be bound by the terms thereof.

All information, journals, this journal, activities undertaken, materials, services and our website, terms and conditions, privacy policy, and this journal is subject to change anytime without any prior notice.

Incorporation No.: 0423089
License No.: 42125/022010/1186
Registration No.: 430374
Import-Export Code: 1109007027
Employer Identification Number (EIN):
USA Tax ID: 98-0673427

Global Journals Inc.

(A Delaware USA Incorporation with "Good Standing"; Reg. Number: 0423089)

Sponsors: Open Association of Research Society
Open Scientific Standards

Publisher's Headquarters office

Global Journals® Headquarters
945th Concord Streets,
Framingham Massachusetts Pin: 01701,
United States of America

USA Toll Free: +001-888-839-7392
USA Toll Free Fax: +001-888-839-7392

Offset Typesetting

Global Journals Incorporated
2nd, Lansdowne, Lansdowne Rd., Croydon-Surrey,
Pin: CR9 2ER, United Kingdom

Packaging & Continental Dispatching

Global Journals
E-3130 Sudama Nagar, Near Gopur Square,
Indore, M.P., Pin: 452009, India

Find a correspondence nodal officer near you

To find nodal officer of your country, please
email us at local@globaljournals.org

eContacts

Press Inquiries: press@globaljournals.org
Investor Inquiries: investors@globaljournals.org
Technical Support: technology@globaljournals.org
Media & Releases: media@globaljournals.org

Pricing (Including by Air Parcel Charges):

For Authors:

22 USD (B/W) & 50 USD (Color)
Yearly Subscription (Personal & Institutional):
200 USD (B/W) & 250 USD (Color)

EDITORIAL BOARD

GLOBAL JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY

Dr. Corina Sas

School of Computing and Communication
Lancaster University Lancaster, UK

Dr. Kassim Mwitondi

M.Sc., PGCLT, Ph.D.
Senior Lecturer Applied Statistics/Data Mining,
Sheffield Hallam University, UK

Dr. Yogita Bajpai

M.Sc. (Computer Science), FICCT
U.S.A.
Email: yogita@computerresearch.org

Dr. Diego Gonzalez-Aguilera

Ph.D. in Photogrammetry and Computer Vision
Head of the Cartographic and
Land Engineering Department
University of Salamanca
Spain

Alessandra Lumini

Associate Researcher
Department of Computer Science
and Engineering
University of Bologna Italy

Dr. Osman Balci, Professor

Department of Computer Science
Virginia Tech, Virginia University
Ph.D. and M.S. Syracuse University, Syracuse, New York
M.S. and B.S. Bogazici University, Istanbul, Turkey
Web: manta.cs.vt.edu/balci

Dr. Kurt Maly

Ph.D. in Computer Networks, New York University,
Department of Computer Science
Old Dominion University, Norfolk, Virginia

Dr. Stefano Berretti

Ph.D. in Computer Engineering and Telecommunications,
University of Firenze
Professor Department of Information Engineering,
University of Firenze, Italy

Dr. Federico Tamarin

Ph.D., Computer Engineering and Networks Group,
Institute of Electronics, Italy
Department of Information Engineering of the
University of Padova, Italy

Dr. Aziz M. Barbar, Ph.D.

IEEE Senior Member
Chairperson, Department of Computer Science
AUST - American University of Science & Technology
Alfred Naccash Avenue – Ashrafieh

Dr. Anis Bey

Dept. of Comput. Sci.,
Badji Mokhtar-Annaba Univ.,
Annaba, Algeria

Er. Suyog Dixit

(M.Tech), BE (HONS. in CSE), FICCT
SAP Certified Consultant
CEO at IOSRD, Ph.DGAOR OSS
Technical Dean, Global Journals Inc.(US)
Website: www.suyogdixit.com
Email: suyog@suyogdixit.com,
deanind@globaljournals.org

Dr. Abdurrahman Arslanyilmaz

Computer Science & Information Systems Department
Youngstown State University
Ph.D., Texas A&M University
University of Missouri, Columbia
Gazi University, Turkey
Web: cis.yzu.edu/~aarslanyilmaz/professional_web

Er. Pritesh Rajvaidya

Computer Science Department
California State University
BE (Computer Science), FICCT
Technical Dean, US
Email: pritesh@computerresearch.org,
deanusa@globaljournals.org

Dr. Chutisant Kerdvibulvech

Dept. of Inf.& Commun. Technol.,
Rangsit University
Pathum Thani, Thailand
Chulalongkorn University Ph.D. Thailand
Keio University, Tokyo, Japan

Dr. Sotiris Kotsiantis

Ph.D. in Computer Science, University of Patras, Greece
Department of Mathematics, University of Patras, Greece

CONTENTS OF THE ISSUE

- i. Copyright Notice
 - ii. Editorial Board Members
 - iii. Chief Author and Dean
 - iv. Contents of the Issue
-
1. Quality Assurance in Requirement Engineering. *1-6*
 2. Kanban based Scheduling in A Multistage and Multiproduct System. *7-13*
 3. Discovery of Non-Persistent Motif Mixtures using MRST (Multivariate Rhythm Sequence Technique). *15-24*
 4. Classification of HRS using SVM. *25-29*
 5. Study of Effective Scheduling Algorithm for Application of Big Data. *31-34*
 6. A New View on Classification of Software Vulnerability Mitigation Methods. *35-55*
-
- v. Fellows
 - vi. Auxiliary Memberships
 - vii. Process of Submission of Research Paper
 - viii. Preferred Author Guidelines
 - ix. Index



Quality Assurance in Requirement Engineering

By Uzma Noorin & Mehreen Sirshar

Fatima Jinnah Women University

Abstract- Requirement engineering is the most important process in software development life cycle. Quality assurance in requirement engineering has a great impact on the product quality. It checks whether the requirements meet the desired quality attributes i.e. adequacy, completeness, consistency etc. Quality Assurance of the requirement is important because the cost of requirements failure is very high. The proposed research is based on the survey of the quality assurance in requirement engineering. The major focus of this research paper is to analyze the quality parameters which assure the overcome of the issues related to the requirements. The research papers include brief overview of those parameters.

Keywords: *requirement engineering, quality assurance, models, artifact -based requirements, TSLA, laquso.*

GJCST-C Classification: *B.8.1, D.2.5*



Strictly as per the compliance and regulations of:



Quality Assurance in Requirement Engineering

Uzma Noorin ^α & Mehreen Sirshar ^σ

Abstract- Requirement engineering is the most important process in software development life cycle. Quality assurance in requirement engineering has a great impact on the product quality. It checks whether the requirements meet the desired quality attributes i.e. adequacy, completeness, consistency etc. Quality Assurance of the requirement is important because the cost of requirements failure is very high. The proposed research is based on the survey of the quality assurance in requirement engineering. The major focus of this research paper is to analyze the quality parameters which assure the overcome of the issues related to the requirements. The research papers include brief overview of those parameters.

Keywords: requirement engineering, quality assurance, models, artifact-based requirements, TSLA, laquso.

I. INTRODUCTION

Quality Assurance ensures that the product manufactured is without defects. For this, the quality of the requirements should be fulfilled. Quality Assurance must be done in requirement engineering process, so that the most valuable requirements should be added. The Quality Assurance of Requirements is much important, as it is said by Boehm and Basili;

"Finding and fixing a software problem after delivery is often 100 times more expensive than finding and fixing it during the requirements and design phase."

This research work is based on a survey related to the quality assurance in requirement engineering. The analysis of different on the perspective of different research paper is done using different parameters like quality, feasibility, maintainability, understanding, reusability etc. Each research paper has its own perspective to verify the quality of requirements. The evaluation criteria are given in the TABLE 1, for comparing effects of different parameters that are discuss in analysis. Brief summaries of each survey papers are also mentioned below.

II. EXPERIENCES ON ANALYSIS OF REQUIREMENTS QUALITY [1]

Requirement engineering is the most important and critical process in the software development life cycle. The quality of the system depends on the quality of requirements that are difficult to recognize in requirements development phase but it has a great impact on

the product quality. This paper proposed a method known as LaQuSo Software Product Certification Model for certifying the quality of requirements specification. This method explains three different certification criteria for all product areas; Completeness of the required elements, Uniformity of the product area should be with respect to standards, Conformance of the elements should be according to the property that is subject of the certification. These all criteria focus on the verification of the requirements.

III. ASSESSING THE QUALITY OF SOFTWARE REQUIREMENT SPECIFICATION [2]

In the initial stage it is difficult to determine whether the SRS will provide the final product. So, a quality model is needed known as Goal-Question-Matric (GQM) method. This paper gives the outline of researchers plan related to GQM, their findings, and finally proves that their quality assessment helps in the project success.

IV. IMPROVEMENT OF QUALITY OF SOFTWARE REQUIREMENTS WITH REQUIREMENTS ONTOLOGY [3]

Requirements elicitation is the most difficult task in requirement engineering. The quality and the quantity of elicited requirements depend on both analysts ability and his domain knowledge of the target system. This paper helps in the improvement of quality and quantity of the elicited requirements using requirements ontology model. This paper checks the correctness, completeness and quality of the requirements.

V. APPLYING CASE-BASED REASONING TO SOFTWARE REQUIREMENTS SPECIFICATIONS QUALITY ANALYSIS SYSTEM [4]

This paper focuses on the quality of the prepared software requirement specification. The technique used is Software Quality Assurance audit to check whether the required standards and procedures within this phase are followed or not. This technique checks whether the requirements are complete, modifiable, consistent, ranked, correct, unambiguous, understandable and traceable. The case-based Reasoning technique is used to check the quality of requirements with respect to the previous quality based cases.

Author ^α ^σ: Software Engineering Department, Fatima Jinnah Women University, Rawalpindi. e-mails: Uzmakhattak63@yahoo.com, msirshar@gmail.com

VI. A CASE STUDY ON THE APPLICATION OF AN ARTEFACT-BASED REQUIREMENTS ENGINEERING APPROACH [5]

Requirement engineering process is volatile. This paper developed a model that support the new artifact based philosophy. This approach helps in the improvements in syntactic and the semantic quality of created artifacts. This approach defines the reference model of the artifacts for development process. This paper also showed the increase in the completeness and consistency of artifact and also their flexibility.

VII. WHAT YOU NEED IS WHAT YOU GET! THE VISION OF VIEW-BASED REQUIREMENTS SPECIFICATIONS [6]

This paper worked on the improvement of high quality SRS that fit the particular requests for progressive record stakeholders. For the quality of requirements, its data and the system function and interaction, viewpoints of architectural experts, goals description and technical requirements are most important artefacts.

VIII. DEFECTS IN NATURAL LANGUAGE REQUIREMENT SPECIFICATIONS AT MERCEDES-BENZ: AN INVESTIGATION USING A COMBINATION OF LEGACY DATA AND EXPERT OPINION [7]

This paper works on the study of natural language of Software Requirement Specification. With respect to the quality model, the results obtained are: defect in natural language of SRS in automotive domain; for defect extremity, the associations of quality parameters; signs on the handling quality parameters; time needed information for defect association on the basis of quality parameters. The results ensure that the important quality parameters in investigated NL parameters are completeness, consistency and correctness.

IX. FOREWORD QUALITY IN AGILE METHODS [8]

This paper worked on many different methods and views related to the quality of agile methods. The discoveries of these investigations assist developers, researchers and managers, in this agile method field. The discoveries are made to understand how to reduce the issues of quality while working on this method.

X. A FRAMEWORK OF SOFTWARE REQUIREMENTS QUALITY ANALYSIS SYSTEM USING CASE-BASED REASONING AND NEURAL NETWORK [9]

This paper worked in the improvement of quality analysis process of Software Requirement Specification.

It ensures that the Software Requirement Specification document must have some standards. Further, the analysis of SRS quality is done by using Neural Network (ANN) and CBR techniques. CBR works in the improvement of quality by the reference of past experiences. ANN also works with CBR.

XI. A PROCESS IMPROVEMENT IN REQUIREMENT VERIFICATION AND VALIDATION USING ONTOLOGY [10]

After developing the system, the verification and validation is always done but still there are issues of stakeholders in different domains. The problem arises in the verification and validation of requirement which are difficult activities in requirement engineering. This paper works on removing the conflicts of requirements, their consistencies and recognizing the failure due to requirements. These goals are achieved by ontology which is also used for the verification and validation of requirements.

XII. IT'S THE ACTIVITIES, STUPID! A NEW PERSPECTIVE ON RE QUALITY [11]

Requirement engineering is the important process in System Development. Its Quality must be ensured for testing, development and other software activities. This paper introduces a context-specific Requirement Engineering artifacts Quality and explain how the quality parameters of RE artefacts affect the activities of development.

XIII. SOFTWARE QUALITY CONTROL VIA EXIT CRITERIA METHODOLOGY: AN INDUSTRIAL EXPERIENCE REPORT [12]

This paper introduces the Exit Criteria Methodology. This methodology helps in vast financial systems to improve the quality of the system from the beginning of a SCRUM sprint to the end. As a whole-process control of quality, the methodology is executed from the quality plan to the quality review.

XIV. QUALITY ASSESSMENT METHOD FOR SOFTWARE REQUIREMENTS SPECIFICATIONS BASED ON DOCUMENT CHARACTERISTICS AND ITS STRUCTURE [13]

In this research it is presented that in software development process quality of SRS should be maintained. For this process different assessment methods are used by keeping in mind the three characteristics of SRS unambiguous variable and modifiable because it is necessary to satisfy the customers.

XV. SECURITY ASSURANCE REQUIREMENTS ENGINEERING (STARE) FOR TRUSTWORTHY SERVICE LEVEL AGREEMENTS [14]

In this research through two case studies it is discussed that how STARE is effective in real world by evaluating that STARE from different techniques and easement criteria and how TSLA can be used to achieve TSLA can be used to achieve the trust worthless service.

XVI. DOES QUALITY OF REQUIREMENTS SPECIFICATIONS MATTER? COMBINED RESULTS OF TWO EMPIRICAL STUDIES [15]

Quality of SRS depends on efficient and effectiveness of quality assurance and by two correlations it is proved that SRS is less used for communication because of it defects. In future it is necessary to document SRS on certain degree.

XVII. NAMING THE PAIN IN REQUIREMENTS ENGINEERING [16]

This paper explains that for better quality requirements, human interaction is important for elicitation regardless of project type and company size. For this standard process models and certifiable improvement standards are used. There is not necessary to use agile or non-agile methods for better quality. It is actually the way to solve the problems. The problem itself manifest in different ways.

XVIII. QUALITY ASSURANCE OF COMPONENT BASED SOFTWARE SYSTEMS [17]

On this latest technology our software's and web based applications are more complicated and with complex coding. Quality Assurance also maintains and reduces more work effort and also helps organization to develop application in the right path from the initial stage to the ending point.

XIX. ASSESSING THE QUALITY OF SOFTWARE REQUIREMENTS SPECIFICATIONS FOR AUTOMOTIVE SOFTWARE SYSTEMS [18]

In this paper we conclude that SRS can be depend on the software or application we need to develop. There some types of SRS std. IEEE 830 to 1998. Several standards organizations including the IEEE have identified some points during designing and writing an SRS; Interfaces, Functional Capabilities, Performance Levels, Data Structures/Elements, Safety, Reliability, Security/Privacy, Quality, Constraints and Limitations.

XX. A QUALITY ASSURANCE MODEL FOR ANALYSIS PHASE [19]

According to my opinion Q.A is most important part it plays as a back bone of any application. Accor

ding to experts Q. A reduce 50% effort in final stage. It also helps developers and also help to track any bug. By applying all Q.A methodology you can also improve your application efficiency.

XXI. HOW ARTIFACTS SUPPORT AND IMPEDE REQUIREMENTS COMMUNICATION [20]

In this we collect information from different recourses that artificial mapping is not suitable for all types of project because in some cases it will be costly according to other, it should be suitable for linking bases modules.

XXII. ANALYSIS OF QUALITY PARAMETERS FOR REQUIREMENTS

a) *Feasibility*

The requirements must be feasible in perspective of the financial plan, timetable, and innovation limitations

b) *Comprehensibility*

The definition of requirements must be intelligible by the general population who need to utilize them.

c) *Good Structuring*

The document of the requirements ought to be composed as it were that highlights the auxiliary connections among its components

d) *Modifiability*

It ought to be conceivable to change, adjust, develop, or contract the document of the requirements through adjustments that are as nearby as could reasonably be expected

e) *Traceability*

The setting in which a thing of the requirements archive was made, adjusted, or utilized ought to be anything but difficult to recover. Such setting ought to incorporate the method of reasoning for creation, adjustment, or utilize.

f) *Efficiency*

The requirements should be efficient that is they must fulfill the functionality and they also help in increasing the performance of the product.

g) *Correctness*

The requirements that are going to be implemented must be correct. They must be related to the functionality of the product.

h) *Completeness*

The requirements that are going to implemented must be complete.

XXIII. CONCLUSION

The paper discusses different factors to improve the quality of the product from the initial stage

that is requirement engineering. Different models like LaQuSo, ANN, CBR and many more are introduced for checking different quality parameters including correctness, completeness etc. The feasibility, comprehensibility, Modifiability, good structuring, completeness etc. are the parameters that effect quality of requirements and by analyzing them a better quality of requirements can be achieved. For better quality product, the requirements must be of better quality.

XXIV. ACKNOWLEDGEMENT

I thank to my department of Software Engineering and my teacher, Ma'am Mehreen Sirshar, who have contributed towards the development of this research paper.

REFERENCES RÉFÉRENCES REFERENCIAS

1. Petra Heck, Päivi Parviainen, Experiences on Analysis of Requirements Quality, 2008 IEEE The Third International Conference on Software Engineering Advances.
2. Eric Knauss, Christian El Boustani, Assessing the Quality of Software Requirements Specifications, 2008 16th IEEE International Requirements Engineering Conference
3. Dang Viet Dzung, Atsushi Ohnishi, Improvement of Quality of Software Requirements with Requirements Ontology, 2009 Ninth International Conference on Quality Software
4. Hajar Mat Jani, Applying Case-Based Reasoning to Software Requirements Specifications Quality Analysis System,
5. Daniel M'endez Fern'andez, Klaus Lochmann, Birgit Penzenstadler, Stefan Wagner, A Case Study on the Application of an Artefact-Based Requirements Engineering Approach, 2011
6. Anne Gross, Joerg Doerr, What You Need Is What You Get! The Vision of View-Based Requirements Specifications, 2012
7. Daniel Ott, Defects in Natural Language Requirement Specifications at Mercedes-Benz: An Investigation Using a Combination of Legacy Data and Expert Opinion, 2012
8. Panagiotis Sfetsos, Foreword Quality in Agile Methods,
9. Hajar Mat Jani, ABM Tariqul Islam, A Framework of Software Requirements Quality Analysis System using Case-Based Reasoning and Neural Network,
10. Sana Nazir, Yasir Hafeez Motla, Tahir Abbas, Asma Khatoon, Javaria Jabeen, Mehwish Iqbal, Khush Bakhat, A Process Improvement in Requirement Verification and Validation using Ontology,
11. Henning Femmer, Jakob Mund, Daniel M'endez Fern'andez, It's the Activities, Stupid! A New Perspective on RE Quality, 2015 IEEE/ACM 2nd International Workshop on Requirements Engineering and Testing
12. Xiaoqiong Zhao, Xiao Xuan, Aoyu Wangy, Dong Liuz, Lingyun Zheng, Software Quality Control via Exit Criteria Methodology: An Industrial Experience Report, 2014 21st Asia-Pacific Software Engineering Conference
13. Patra Thitisathienkul, Nakornthip Prompoon, Quality Assessment Method for Software Requirements Specifications based on Document Characteristics and its Structure, 2015 2nd International Conference on Trustworthy Systems and Their Applications
14. Yudhistira Nugraha, Security Assurance Requirements Engineering (STARE) for Trustworthy Service Level Agreements, 2015
15. Jakob Mund, Henning Femmer, Daniel M'endez Fern'andez, Jonas Eckhardt, Does Quality of Requirements Specifications matter? Combined Results of Two Empirical Studies, 2015
16. Daniel Méndez Fernández, Stefan Wagner, Marcos Kalinowski, André Schekelmann, Ahmet Tuzcu, Tayana Conte, Rodrigo Spinola, and Rafael Prikladnicki, Naming the Pain in Requirements Engineering, 2015
17. Ravi Kumar Sharma, Parul Gandhi, Quality Assurance of Component Based Software Systems, 2016
18. Akiyuki Takoshima, Mikio Aoyama, Assessing the Quality of Software Requirements Specifications for Automotive Software Systems, 2015
19. Reham Ejaz, Mubina Nazmeen, Maryam Zafar, A Quality Assurance Model for Analysis Phase, Oct 4th, 2010
20. Olga Liskin, How Artifacts Support and Impede Requirements Communication, January 2011.

Table 1: Evaluation Criteria for Copiler Optimization

No.	Parameters	Description	Value
1.	Feasibility	Requirements are easy to use	Yes, No
2.	Comprehensibility	Requirements are understandable to other people	Yes, No
3.	Good Structuring	Requirements are well structured and have links among its elements	Yes, No
4.	Modifiability	Requirements are revisable and adaptable.	Yes, No
5.	Traceability	Easy to trace the requirements	Yes, No
6.	Efficiency	Requirements fulfill the functionality.	Yes, No
7.	Correctness	Checks whether the requirements are correct.	Yes, No
8.	Completeness	Checks whether the requirements are complete.	Yes/No

Table 2: Evaluations Of Parameters For Compiler Optimization

S #	Reasearch Paper's Author	Feasibility	Comprehe nsibility	Good Structuring	Modifi-ability	Trace-ability	Efficiency	Correct -ness	Complete ness
1	P. Heck, P. Parviainen	Yes	Yes	No	No	Yes	Yes	Yes	Yes
2	E. Knauss, C. E.Boustani	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
3	D. V. Dzung, A. Ohnishi	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
4	H. M. Jani	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
5	D. Fern´andez, M. Lochmann, B. Penzenstadler, S. Wagner	Yes	No	No	No	No	No	Yes	Yes
6	A. Gross, J. Doerr	Yes	Yes	Yes	No	Yes	No	Yes	Yes
7	D. Ott	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
8	P. Sfetsos	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
9	H. M. Jani, A. T. Islam	No	Yes	Yes	No	Yes	No	Yes	No
10	S. Nazir, Y. H. Motla, T. Abbas, A. Khatoon, J. Jabeen, M. Iqr, K.Bakhat	Yes	No	Yes	No	Yes	No	Yes	Yes
11	H. Femmer, J.Mund, D. M. Fern´andez	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
12	X. Zhao, X. Xuan, A.Wangy, D. Liuz, L. Zheng	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
13	P.Thitisathienkul, N.Promptoon	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
14	Y. Nugraha	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
15	J. Mund, H. Femmer, D. M. Fern´andez, J. Eckhardt	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

16	D. M. Fernández, S. Wagner, M. Kalinowski, A. Schekelmann, A. Tuzcu, T. Conte, R. Spinola, R.Prikladnick	No	No	No	No	No	Yes	Yes	Yes
17	R. K. Sharma, P. Gandhi	Yes	No	Yes	No	Yes	Yes	Yes	Yes
18	A. Takoshima, M. Aoyama	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
19	R. Ejaz, M. Nazmeen, M. Zafar	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
20	O.Liskin	No	Yes	No	No	Yes	Yes	Yes	Yes





Kanban based Scheduling in A Multistage and Multiproduct System

By Chukwuedozie N. Ezema, Eric C. Okafor & Christiana C. Okezie

Enugu State University of Science & Technology

Abstract- In the highly competitive manufacturing environment, many companies around the world are searching for ways to improve manufacturing performance. This is in response to changes in the manufacturing environment reflected by shortened product life cycles, diverse customer needs and the rapid progress of manufacturing technology. A JIT tool otherwise referred to as kanban based scheduling is then considered as a suitable management concept for Juhel Pharmaceutical Nigeria Ltd to address the challenges by minimising all the components, particularly on the shop floor. The JIT system is not just related to Kanban implementation but it comprises an all-inclusive approach for improving batch size reduction, setup time reduction, quality, production planning, and human resources management. Mechanisms and operating procedures are required to provide detailed step-by-step instructions for the implementation of a pull system.

Keywords: *JIT system; kanban; MRP; MPS; drug process plant.*

GJCST-C Classification: *J.5, H.5.5, D.2.5*



Strictly as per the compliance and regulations of:



Kanban based Scheduling in A Multistage and Multiproduct System

Chukwuedozie N. Ezema ^α, Eric C. Okafor ^σ & Christiana C. Okezie ^ρ

Abstract In the highly competitive manufacturing environment, many companies around the world are searching for ways to improve manufacturing performance. This is in response to changes in the manufacturing environment reflected by shortened product life cycles, diverse customer needs and the rapid progress of manufacturing technology. A JIT tool otherwise referred to as kanban based scheduling is then considered as a suitable management concept for Juhel Pharmaceutical Nigeria Ltd to address the challenges by minimising all the components, particularly on the shop floor. The JIT system is not just related to Kanban implementation but it comprises an all-inclusive approach for improving batch size reduction, setup time reduction, quality, production planning, and human resources management. Mechanisms and operating procedures are required to provide detailed step-by-step instructions for the implementation of a pull system. Basically, the Drug Process Plant operations are mainly characterised by single flow line production processes, periodical and multi-items orders. Based on the evaluation of the implementation of the new system, there are some factors that must be considered for further improvement including inventory reduction, improving visibility, batch size reduction and matching with other systems.

Keywords: JIT system; kanban; MRP; MPS; drug process plant.

1. INTRODUCTION

Manufacturing firms are currently encountering problems because of changing environment, varying weather conditions, product design changes and rapidly changing customer demand. Thus, the Manufacturing Resource Planning (MRP) system and the Mass Production System (MPS) cannot respond quickly enough to the product design changes. This results in, amongst other things, high levels of obsolete stocks.

Also, manufacturing environment could be too turbulent to allow accurate forecasting. This results in excessive obsolescence. This can only be improved by reducing the lead time below what can be achieved by Manufacturing Resource Planning (MRP). However, the need to be more responsive to rapidly changing customer demand as a result of market competition remains a constant dominant challenge. In the highly competitive manufacturing environment, many companies around the world are searching for ways to improve manufactu-

ring performance. This is in response to changes in the manufacturing environment reflected by shortened product life cycles, diverse customer needs and the rapid progress of manufacturing technology (Uwakwe, 2015).

Amongst the available technology and systems is Just-In-Time (JIT), a Japanese manufacturing concept that puts emphasis on waste elimination. This is a suitable means for a company that wants to perform in a competitive market. Some potential benefits that can be achieved by applying JIT concepts include: significant reduction of setup time, reduced cost of quality (such as scrap/rework reduction), increased inventory turn-over, increased manufacturing flexibility and shorter lead time (Melitus, Kevin, & Collins, 2016). Companies operating in highly competitive environments are the most appropriate for implementing JIT concepts.

There are four motives for using JIT in industries (Vincent and Abdul-Karim, 2009). First, some industries are characterised by a short product life cycle, for that reason, lead time and inventory reduction must become main concerns. Secondly, a large percentage of the cost of goods sold is material cost so decreased inventory and scrap is very essential. Thirdly, the collective effects of short life cycle and high material costs lead to a high level of material obsolescence, thereby, decrease in lead time and inventory again becomes main concerns. Finally, rapid technological advancement causes shorter life cycle so the company must be able to reduce time required to meet up with customer needs.

In this paper, a serial multi-stage manufacturing system controlled by Kanbans is considered which acquires raw materials from outside suppliers and route them through multiple work-stages to produce a varying quantity of finished products to customers at a fixed-interval of time (Figure1). The raw materials are also replenished instantaneously to the manufacturing system to meet the JIT operation and time-varying finished product demand model, and the production capability of the system is flexible.

An ideal JIT manufacturing system produces only the right items in right quantities at right time. Traditional manufacturing facilities carry large inventories of finished goods to satisfy the demands of customers that adopt a JIT delivery system. In the newly proposed JIT system, lot sizes are compact as much as possible and deliveries of products are scheduled repeatedly. The express impact of the JIT system is decrease of inventory holding cost.

Author α ρ: Department of Electronic and Computer Engineering, Nnamdi Azikiwe University, Awka, Anambra State, Nigeria.

Author σ: Department of Computer Engineering, Enugu State University of Science & Technology, Enugu State, Nigeria.
e-mail: ecnaxel@gmail.com

As a result, the manufacturer should get exact knowledge of demands of finished products and sustain an optimum production schedule to match the supply chain manufacturing system. If production is synchronized with the customers' lumpy demands and the ordering of raw material with production schedules is

properly coordinated, all raw materials, WIP and finished goods inventories could be sustained at an economic level in a manufacturing firm to reduce the integrated inventory cost incurred due to raw materials, WIP, and finished products.

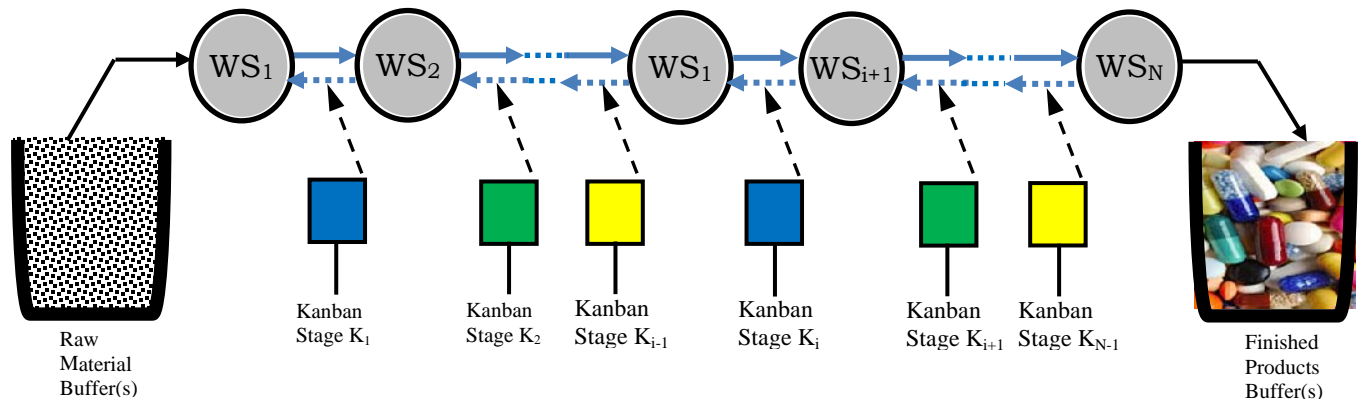


Figure 1: A serial multi-stage drug manufacturing system with Kanban operations

II. MATERIALS AND METHOD

One of several indicators that pharmaceutical companies are able to survive within the global marketplace is their ability to improve return on assets (ROA). ROA will improve if either turnover or return on sales (ROS) increases. Turn over that is obtained by dividing sales into assets can be increased if assets decrease. Since in a pharmaceutical company, inventory is a major part of assets, inventory reduction will improve turnover significantly. Similarly, ROS will increase if operating profit that is obtained by subtracting sales against total costs and expense increases. Since in such companies inventory is a major part of the total cost, inventory reduction will significantly improve ROS. Therefore, inventory reduction, is a key factor for improving ROA and eventually to survive global competition (Mathew, Bali & Edmund, 2014).

These considerations require the companies to find better ways for reducing various type of inventory such as raw materials, WIP and finished goods. JIT is then considered as a suitable management concept for Juhel Pharmaceutical Nigeria Ltd to address the challenges by minimising all the components, particularly on the shop floor.

a) An Overview about the Drug Process Plant

Basically, the Drug Process Plant operations are mainly characterised by single flow line production processes, periodical and multi-items orders. There are around 97 periodical items produced by the Drug Process Plant with Mechanisms and operating procedures are necessary to provide detailed step-by-step instructions for the implementation of a pull system. These management tools must be developed clearly so all people working with this the order quantity ranging

from one pallet to 700 pallets. With such characteristics, it is not surprising that MRP was then introduced to control the plant.

b) Products

Basically, items produced by the Drug Process Plant can be classified into three types as shown in figure 2: Product A otherwise referred to as tablets (55% of order volume), Product B otherwise known as capsules (35% of order volume) and Product C otherwise referred to as pills (10% of order volume).

c) Layout

Because of the type of manufacturing processes, the Drug Process Plant employs product flow layout as shown in the Figure 3. The benefit of this layout is that the process paths are clear so everyone understands what the next process is. Unfortunately, because of space constraint and the size of particular machines, most process paths are not straight lines so the processes necessitate extra time for transport as a result of extra distances. Moreover, these problems also lead to other problems such as unfixed locations of buffers so WIP and inventory are not observable.

d) Designing Mechanisms or Operating Procedures for Running the System

system understand how to accomplish the task. Diagrammatically, the mechanisms of the pull system at the Drug Process Plant are based on the design of the system as shown in Figure 4.

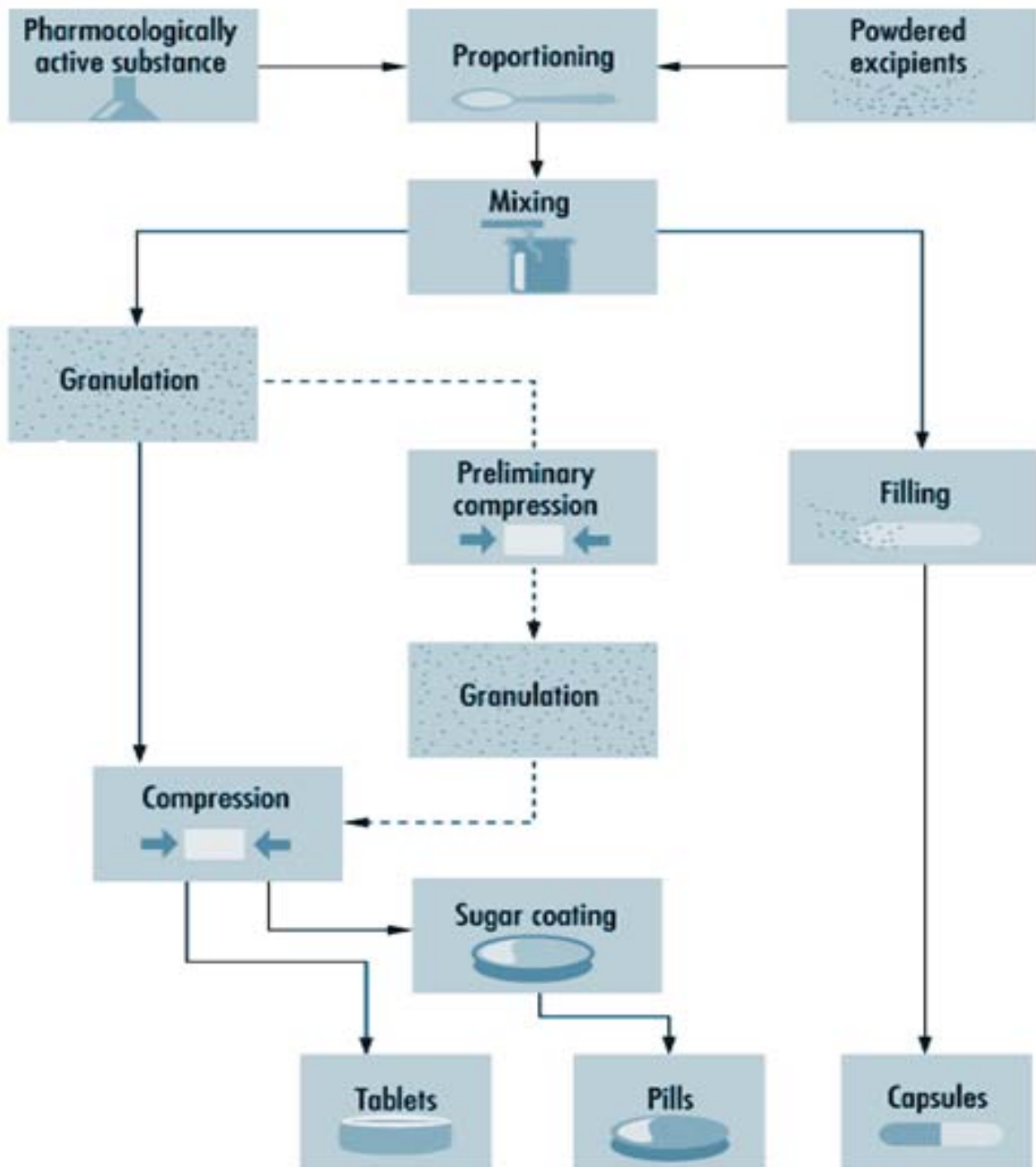


Figure 2: Items Produced by the Drug Process Plant

The mechanisms of the novel pull system can be illustrated in the following procedures. Customers pull in to pick up full containers of sub-pallets from the Operators in the blister packing/strip sealing section must check the board whether there are cards or not. If there are cards, they take the cards and start production by taking raw materials from buffer 2 and putting them into the unfilled containers. The quantity of raw materials taken is equal to the total Kanban quantity detached from the board. If buffer storage area (end buffer). When

taking the containers, they must put green Kanbans from the full containers on board 2.

2 reaches the trigger point, they place the yellow Kanban from buffer 2 onto board 1. Operators in the mixing/blending section must check this board. If there is a yellow Kanban, they must take this card and start the production. In this block, the production quantity is 360 units as printed on the yellow Kanban. The numbers in the flow chart refer to activities as shown in Figure 4.

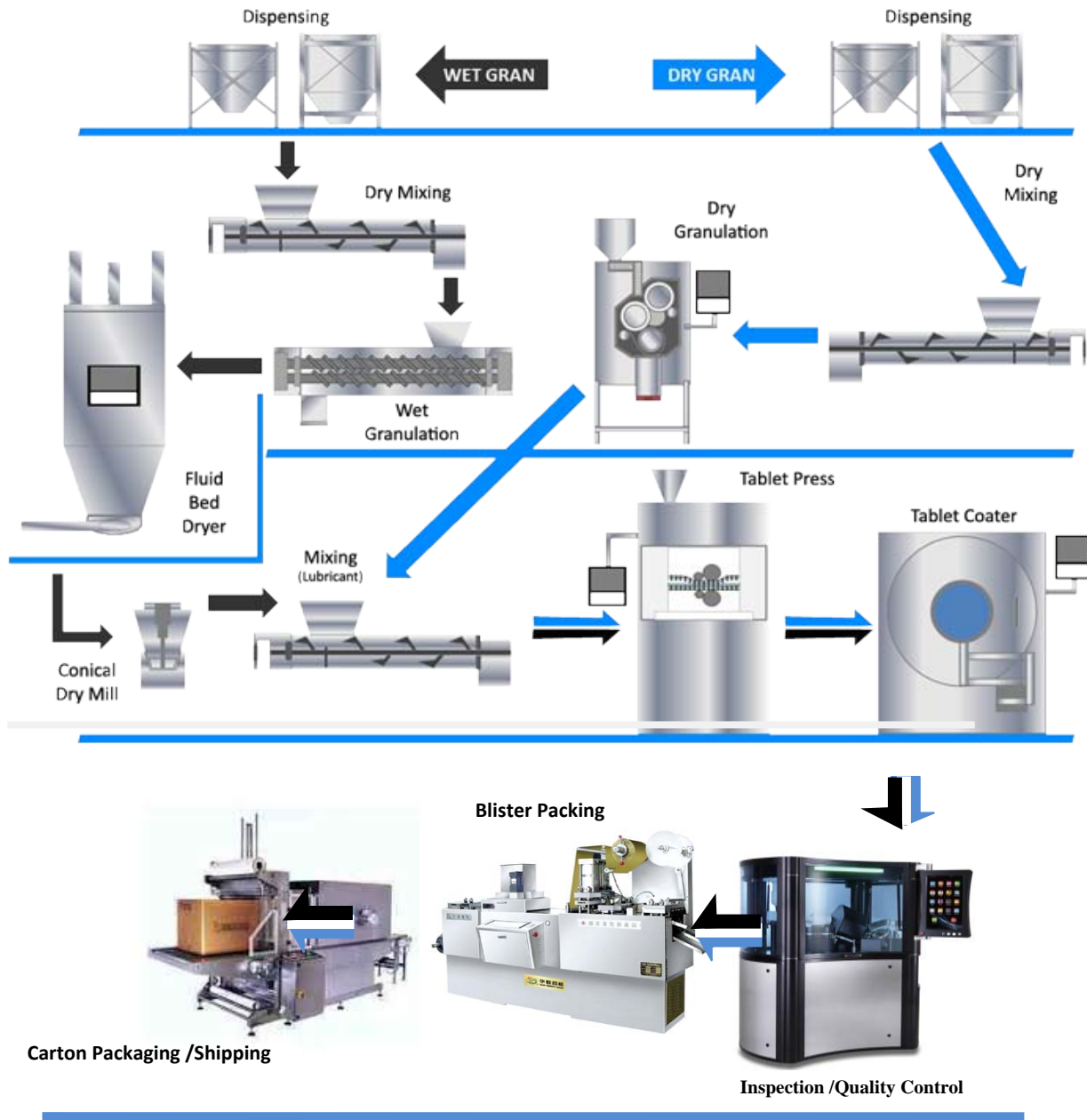


Figure 3: General Layout of the Drug Process Plant

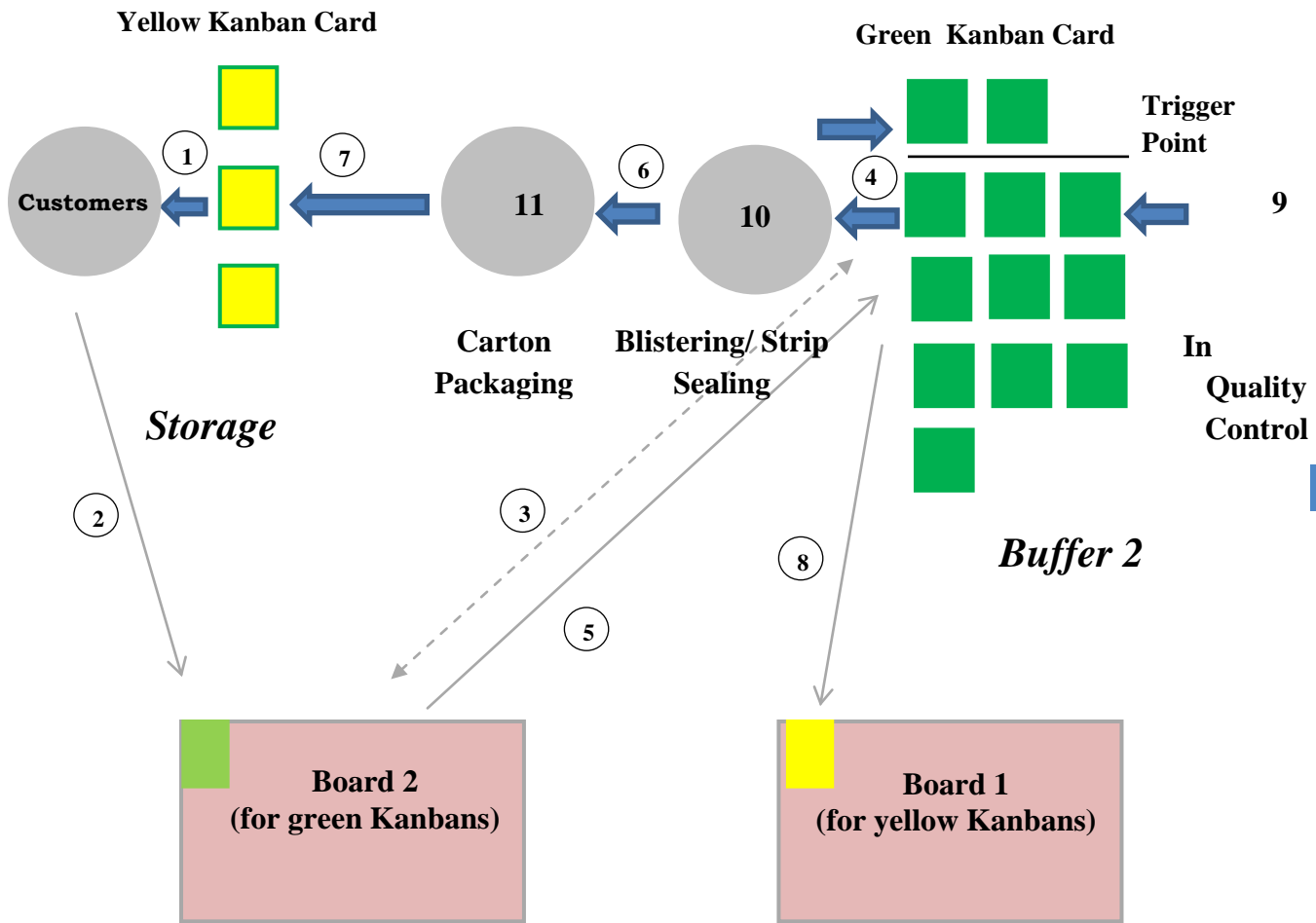


Figure 4: Mechanism of JIT System

III. RESULTS AND DISCUSSION

a) Raw Data and Descriptive Statistics

Because the data were generated through an ARENA simulation model and manually entered into an Excel spreadsheet for sorting and calculations uploading into SPSS, the possibility of researcher error in transferring the data exists. The product costs were first determined in the simulation model by using different manufacturing system alternatives: Mass Production System (MPS), Material Resource Planning System (MRP), and Just in Time Manufacturing System (JIT). The product cost data were then input into the Integer Linear Programming (ILP) model to determine the optimal product mix, which was then input into the simulation model for use in the product mix decision. Average performance data were collected for 60 replications of 30 days each for 27 experimental condition groups, representing three different Manufacturing Systems (MAS) (Mass Production System (MPS), Material Resource Planning System (MRP), and Just in Time Manufacturing System (JIT)), three levels of manufacturing overhead (low, medium,

high), and three levels of product mix complexity (low, medium, high) for a total of 1620 data points. Table 1 below shows the number of observations by experimental factor.

b) Practical Implications

Because the primary focus of this study is to examine the impact of kanban based scheduling on manufacturing performance in the context of a time-based competitive environment, it is necessary to take a more detailed look at this impact on each individual performance measure. The three performance measures were chosen because they represent both internal and external and financial and non-financial measures of performance. Table 1 below presents a summary of the results in performance measures by manufacturing system alternative.

Demand fulfillment rate measures an external (market) non-financial representation of manufacturing performance. It corresponds to the percentage of demand that is ultimately fulfilled by the production system. The maximum performance in terms of this measure was Material Resource Planning System (MRP) (MAS_2) with 86.6% of demand fulfillment rate and Just

in Time Manufacturing System (JIT) (MAS_3) with 85.4% of demand fulfillment rate. The most abysmal performance was Mass Production System (MPS) (MAS_1) with 69.8% of demand fulfillment rate. Even though the difference between MRP and JIT in terms of

demand fulfillment rate was statistically significant, from a practical perspective, this difference may not corroborate the high cost of implementing an MRP system.

Table 1: Multiple Comparisons by MAS

Dependent Variable	(I) MAS	(J) MAS	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
DFR_2	1	2	-.16799907*	.000245697	.000	-.16860104	-.16739710
		3	-.15545861*	.000245697	.000	-.15606058	-.15485664
	2	1	.16799907*	.000245697	.000	.16739710	.16860104
		3	.01254046*	.000245697	.000	.01193849	.01314244
	3	1	.15545861*	.000245697	.000	.15485664	.15606058
		2	-.01254046*	.000245697	.000	-.01314244	-.01193849
CT_2	1	2	-29.211410*	1.91319765	.000	-33.89883996	-24.52397938
		3	53.468745*	1.91319765	.000	48.78131487	58.15617545
	2	1	29.211410*	1.91319765	.000	24.52397938	33.89883996
		3	82.680155*	1.91319765	.000	77.99272454	87.36758512
	3	1	-53.468745*	1.91319765	.000	-58.15617545	-48.78131487
		2	-82.680155*	1.91319765	.000	-87.36758512	-77.99272454
NOI_2	1	2	-6.3592155*	.124743740	.000	-6.66484388	-6.05358703
		3	-.10501750	.124743740	.702	-.41064593	.20061092
	2	1	6.35921545*	.124743740	.000	6.05358703	6.66484388
		3	6.25419795*	.124743740	.000	5.94856952	6.55982637
	3	1	.10501750	.124743740	.702	-.20061092	.41064593
		2	-6.2541979*	.124743740	.000	-6.55982637	-5.94856952

Based on observed means.
The mean difference is significant *, at the .05 level.

IV. CONCLUSION

Based on this evaluation, the differences between using the previous system and the new system are recapitulated in the Table 2. The JIT system is not just associated to Kanban implementation but it involves

an all-inclusive approach for enhancing the performance of a system that covers batch size reduction, setup time reduction, quality improvement, production planning, and human resources management. Therefore, there will be more significant results if the enhancement also covers those areas using an integrated approach.

Table 2: The Results of the Implementation

OUTCOME	BEFORE JIT	AFTER JIT	IMPROVEMENT
Lead time	10 days	2 day	5 times
Inventory	1080 units (2 x360 +360)	540 units (90 + 360)	50%
Visual Control	None	Self-Driven	Better
Employee Motivation	Normal	Higher	Better

Based on the assessment of the implementation of the new system, there are some factors that must be put into consideration for further improvement including inventory reduction, improving visibility, batch size reduction and matching with other systems.

V. ACKNOWLEDGEMENT

The authors acknowledge that the manuscript submitted is their own original work. All authors

participated in the work in a substantive way. The submitted manuscript is reviewed and approved by all authors. The authors thank Juhel Nigeria Ltd for providing data for the problem. This study was conducted at Juhel Nigeria Ltd. The authors are grateful to all who provided various forms of assistance during this study.

REFERENCES RÉFÉRENCES REFERENCIAS

1. Uwakwe, N., (2015). "System Approach to ComputerIntegrated Design and Manufacturing," Precision Publishers Ltd, Enugu, 2015.
2. Melitus, J., Kevin, D., and Collins, R., "A comparison of strategies to dampen nervousness in MRP systems," *Management Science*, Vol. 32, No. 4, pp. 413–429, 2016.
3. Vincent, J. A., and Abdul-Karim, B., "Production and Quality Improvement in Electronics Assembly," Africap Publishng Co., Kenya, 2009.
4. Mathew, J., Bali, A.A., and Edmund, S., "Just-In-Time Information Sharing Architectures in Multiagent Systems," *International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Bologna, 2014.



This page is intentionally left blank





Discovery of Non-Persistent Motif Mixtures using MRST (Multivariate Rhythm Sequence Technique)

By R Kumar & Capt. Dr. S Santhosh Baboo

Manonmaniam Sundaranar University

Abstract- In this paper we present a prototype to discover the unsupervised repeating temporary perception in a time series. The purpose of this work is to control the case of random variable and to find out the measurements caused by the phenomena of simultaneous synchronization. The proposed model has used the non-parametric Bayesian technique to trace the motifs and their occurrences in the data documents. We introduce the Multivariate Rhythm Sequence Technique (MRST) method to find the rebound and repeated motifs and their instance in every document automatically and simultaneously. This model is used in wide range of applications and concentrates on datasets from different modalities.

Keywords: *motif mining, multivariate time series, unsuper-vised analysis, bayesian modelling, camera network.*

GJCST-C Classification: *H.5.5*



Strictly as per the compliance and regulations of:



Discovery of Non-Persistent Motif Mixtures using MRST (Multivariate Rhythm Sequence Technique)

R Kumar ^α & Capt. Dr. S Santhosh Baboo ^σ

Abstract- In this paper we present a prototype to discover the unsupervised repeating temporary perception in a time series. The purpose of this work is to control the case of random variable and to find out the measurements caused by the phenomena of simultaneous synchronization. The proposed model has used the non-parametric Bayesian technique to trace the motifs and their occurrences in the data documents. We introduce the Multivariate Rhythm Sequence Technique (MRST) method to find the rebound and repeated motifs and their instance in every document automatically and simultaneously. This model is used in wide range of applications and concentrates on datasets from different modalities.

The video footages from non-dynamic cameras and data location bounded to the motif-mining server. The high semantic internal representation of the method gives advantage in operation such as event counting or analyse the scène. We used the sample images and videos from New York City traffic data for experiments with and the results shows better performance than the existing motif mixtures analysis in the time series.

Keyword: motif mining, multivariate time series, unsupervised analysis, bayesian modelling, camera network.

I. INTRODUCTION

Motif Mining is one of the active research area in recurrent temporal pattern in time series. The ultimate aim is to find out the small amount of repeating temporal pattern with little possible supervision in multiple variant time series. The super position analysis using various phenomena in time series is without synchronization. This work is to extractarious operations and activities in different domains from the video footage. Normally we get the video sequence consists different movements by various people or various objects present in the video footage. In this case we are using the long term recording to study the independent activities and their presence in the video automatically.

Time series motifs are recurrent segments in a long time series that their presence implies the precise information about the underlying source of the time series. Motif discovery in time series data has received

significant attention in the data mining community since its inception, principally because, motif discovery is meaningful and more likely to succeed when the data is large.

Different types of the time series has the same characteristics of being unification of the multiple actions or motifs. Here we assume the time series pertaining to the electricity lining and consumption of the water in a particular building. In this experiment, we can find out the motif such as water consumption and short circuit in the building. It is also possible that we can find some other method to supply the water and electricity in the particular apartment. It is also possible to determine the consumption of water and electricity. This kind of multiple incidence of motifs happens at the same time without any synchronization.

To find out the specific activity patterns without supervision is our primary goal. The starting point of the task is to recap the scene, count or detect the specific scene to find with the unusual activity. Figure 1 explains the difference in the particular case without supervision video sequence.

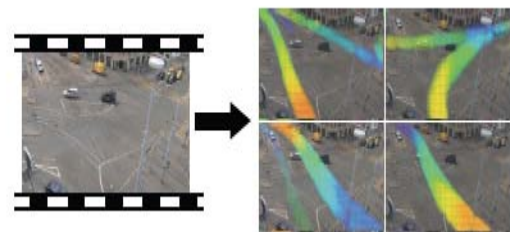


Figure 1(a)

Figure 1(b)

Figure 1 is the result of the experiment on a video footage. 1(a) without supervision from which we are going to extract the temporary activity methods. 1(b) motif represented as time denoted by the gradient colour.

II. RELATED WORK

Considering the non-parametric Bayesian technique, it is systematically investigating each of the implicit number of motifs and number of motif present in every document. This method validates the synthetic data. This method also used for other prediction efficient such as video sequence and other domains. [1]

Author ^α: Research Scholar, Manonmaniam Sundaranar University, Tirunelveli. e-mail: rkumarh2008@gmail.com

Author ^σ: Associate Professor, P.G and Research Dept. of Computer Science, D.G Vaishnav College, Chennai.

The video sequences are divided into simple clips in order to find the flow fields in that particular video sequences. Pixels has quantized based in the motion location and direction. This is a group of words denoted in the clips. Once we find the group of words, we can do the next state called as screening. This stage is to measure the words using the technique called "conditional entropy" after getting the result in full words, which are applied, to the diffusion map. Diffusion map is the framework, which has included the multiplex of the points into lower dimensional space while preserving the intrinsic local geometric pattern. [2]

Hierarchical Bayesian method combines three elements in a visual surveillance. Basic a) level visual aspect, b) uncomplicated atomic activities and c) communication. Atomic activities are modelled as distributions over low-level visual features, and interactions are modelled as distributions over atomic activities. This method uses unsupervised learning method. Taking a long video footage, movable points are clustered as various atomic activities and small video sequence shows the interactions. [3]

Unsupervised learning method relies on possibilities. Latent semantic analysis approach is used to set visual characteristics including the attributes like size and motion activities for finding same actions happening in the particular scenes. Then the patterns are found in the segments into regions is clear and activity content. [4]

We present two novel methods to automatically learn spatiotemporal dependencies of moving agents in complex dynamic scenes. They allow to discover temporal rules, such as the right of way between different lanes or typical traffic light sequences Systematically, the spatiotemporal dependencies of moving agents are observed in the in complex dynamic scenes. This scene allows to find out temporary protocols as the exact way between various lanes in the typical traffic areas. First model is based on the protocol based learning method. The next method uses Dependent Dirichlet Processes to learn an arbitrary number of infinite Hidden Markov Models. DDP-HMM. [5]. Different guessing based on kalman filter and non-argument regression are getting posterior inference in the topic. LDA is the Latent Dirichlet Algorithm method captures the data not only in the depressed concept data. We analysis how the architecture changes over the time. Here every method is related with the continuous distribution timestamps. Every document word generated by co-occurrence of the timestamp. [6][7]

A Markov clustering topic method is present to build in earlier method of dynamic Bayesian network method. Bayesian method has eliminated the draw backs about accuracy, strength and efficiency. Difficult dynamic clips by strength clustering visual activities correlates over the time. The Gibbs sampling is used for the offline and unlabelled data. [8]

LDA is used to find out the global correlation in the spread camera network LDA is used to divide the object action structure and local behaviour in every camera view first interference of the two local behaviours globally over the different camera views. The LDA is preparing to find out actions and temporal correlations. [9]

Latent Dirichlet Approach based method take the snap of the activities that changes over a period. The agglomerative based clustering on optical flow vectors in different angle and spatial information. Here every activity interrelates with not only in the distribution. It is interrelated with distribution over the timestamps. [10]

Normally every document has the combination of continuous motif activity and their starting appearance. Here they are using the three methods. First method has interrelated word at the particular time stamps and find out the word repeated in the document in the particular time series or temporal window. Second find the repeated action in the document. The third to find out the same occurrence and activity, which should be monitored. [11]

Bunch of data in each observation are grouped together in a mixture model. The number of colloid tools is known as unknown priori frequency from the data. This arrangement is known as the Dirichlet process the identified cluster property has provided the non-argument prior the number of composition of tools in each group.[12]

A structure of the non argument Bayesian method is known as the dual Hierarchical Dirichlet process. Unsupervised gravity discusses and semantic circle method in investigation settings. Heretrajectory is as the words in the document, which clusters into various activities. Defective words are identifying as sample with low probability. Semantic atmosphere sets way to get objects and relate the actions in the particular scene and creates the model of the scene. [13]

Global discourse method detects the abnormal activity that isolated appearance. Practical sector believe that kind of argument modulate is needed and real time is finagle. [14][15].

Using the activity trend has presence of instance to invoke similarity. Here no need of the inter camera registration or adjustment. However, apart from that system learn the camera network and possibilities of the path and instance in Parson Window at the time of training. When the training is finished related are assigned the maximum posterior the estimated structure. [16]

A cross-canonical correlation analysis structure detect and express the normal relationship in the two regional activities in the cross camera view. Find the spatial and network structure of the camera. Accurate and identify the person or object. Perform the activity segmentation collect the different camera views.[17]

Using the base level of the queue instance is finding in the each camera view independently and the landmark and speed of instance and trajectories are calculated as advantage. Generative method learns the actions and features in different camera sites. Grouping a same picture or image has convert into one cluster. Then all the cluster has find out from the different camera views and make it as a co- clustering has been published. [18]

III. PROPOSED MODEL

This proposed model MRST is to process the motif, number and occurrences. This section will explain three variations of the method called TAMM, VLTAMM and TSVLMM before going to the main concept in detail.

a) Background on Time Series Mrst Process

We introduce the Time Series of MRST Process, as a model, which handle the infinite mixture in the building, is our approach collusion components categorizes to the form of distributions. Here we are used the Gaussian mixture model to explain the concepts in the introduction part.

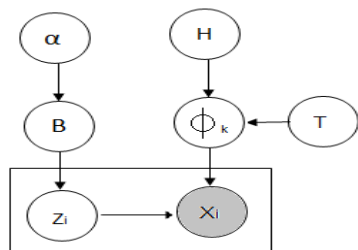


Figure 2 (a)

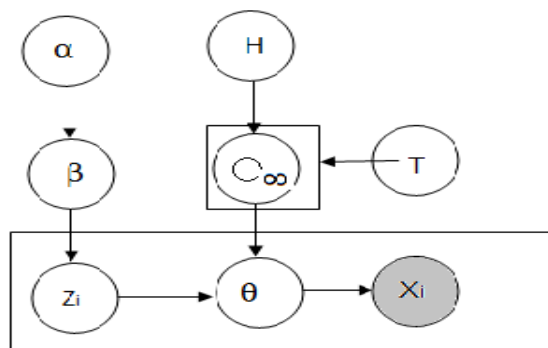


Figure 2 (b)

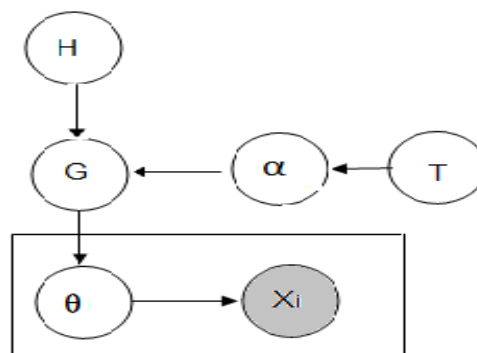


Figure 2 (c)

Figure 2 a) finite mixture with k elements with time t. b) Mixture representation with time t c) Compact representation of MRST.

The infinite number of mixture has an alternative delimited number with k and the time serial to accept this mixture nodes weight vector β as the length and α as the form. A α is the real number as argument and t is the time period while we are using the “stick breaking” method has given the infinite list of data that sum to 1. The weight vector $\beta_1 = \beta'_1$ is obtained from the beta distribution. The second weight vector in the same way $\beta_2 = (T - \beta_1) * \beta'_1$ and so on this way is called as the stick breaking method.

Algorithm 1

- getting the video sequence as INPUT
- find video sequence ϕ_k (1)
- find the time of the ϕ_k with time t
- find the x_i
- convert into compact representation (2)
- take it as ϕ_k in to ∞
- calculate the ∞ with a time period t (3)
- adding the θ value where $\theta = Z(i) * \infty$ (4)

A concentrated equal node has been used to denote the time series MRST process. The mixture presentation has well acceptable derive the Gibbs sampling method concentrated presentation is used in the broad manner to get a quick view of the model.

b) The Proposed Model

Our aim is to derive set of motifs a collection of record containing the index words. We define the record j is the group of experiment. $\{(w(ji) \text{ at}(ji))\}_{i=1 \dots n(j)}$ where $w(ji)$ words refer to the word book and the $\text{at}(ji)$ is the at mean time of the experiment occurred in the document

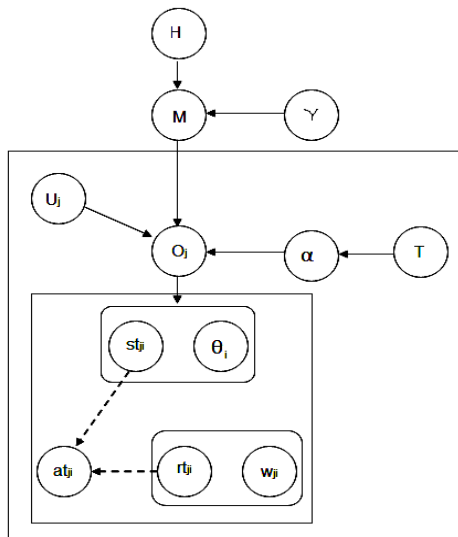


Figure 3(a)

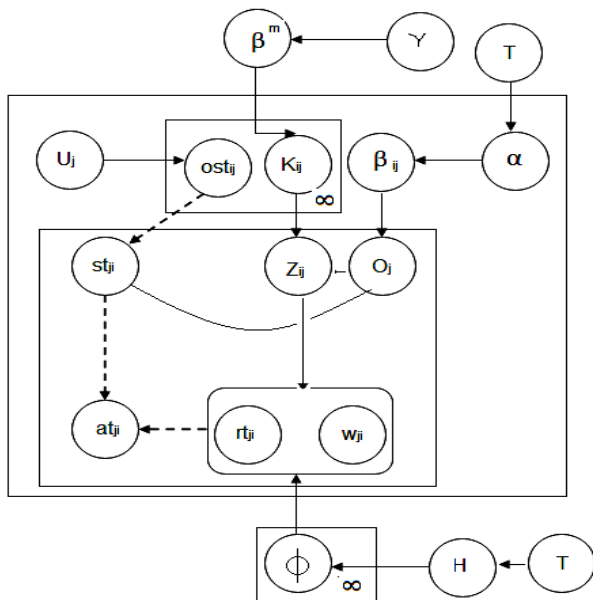


Figure 3(b)

Figure 3 represents the proposed models a) consolidate with MRST notation b) with develop time series MRST process and both levels.

Algorithm 2

get the video sequence with vector (5)

$B(m)$ send to K_i where K_i is number of finite motif

$B(i)$ from α when the time T (6)

find the ost multiple with $U(j)$ where $U(j)$ is possible of motifs (7)

$O(j) * st(ij)$ where $st(ij)$ is the founded possible motif

$k(i) * Z(i)$ where $Z(i) =$ convert the video sequence in to word

$$R(t) * W(t) = (\infty * H * T) \quad (8)$$

$$A(t) = R(t) * S(t) \quad (9)$$

We look up the time details when defining the motifs temporal possibilities map. More importantly ϕ_k point the motif table and $\phi_k(w, RT)$ defines the possibility the word w presence at the relevant time object when the motif occurs (RT) is stated out aim is conclude the set of motifs more than one document. As per the early discussion it is compulsorily added to the every document. It is very difficult to find the number of motifs in advance so here we are using the time series MRST process which allows to read the number of variable in the motifs in the document.

Our new approach of the MRST process has use the graphical representation method shown in the figure 5.1 and 5.2. Consider the two diagrams the time series process notation has indicated by the group of square in the above diagram and the further method has used to find the variables and number of motifs in the time series of the document.

Here the settled relation is referring as the first MRST level which prepare the number of motifs from the ultimate combination M . Every motif has derived from an MRST process of distribution with argument N . Basic combination model has been LDA or HDP, the set of combination of the section is not only mutual information or document but also across the MRST using the second level commonly the document specification displacement $O(j)$ is not consider the motif mixture.

Observations (w_{ji}, at_{ji}) are produced by the repeated sampling motif using the motif θ_j to sample the word and its relevant time and motif. Using the sampling start time at_{ji} absolute time st_{ji} can be reduced. The developed method given in fig 4bit helps to understand the creation process.

The important difference in this small scale model is the way of the repentance as created is represented explicitly. Occurrence of the table Chinese restaurant model document specific is used to create the examples.

c) Modeling Prior H And Motif Length In A Time

The earlier section shows the worldwide structure of the method specially simplified the explanation of the superior H and neglect the details

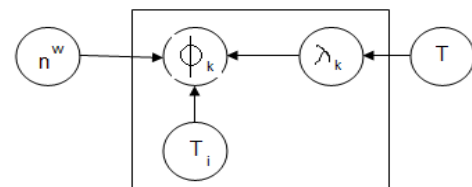
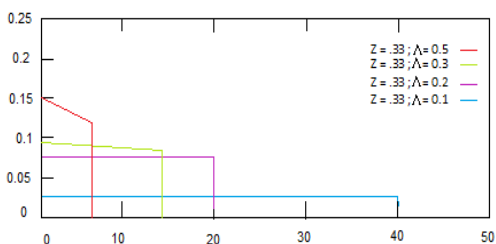


Figure 4

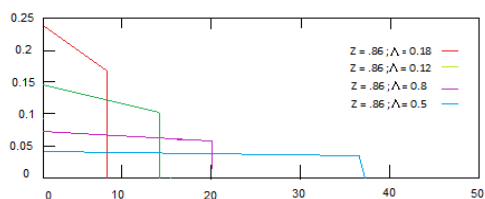
Figure: 4 different variable lengths have taken care of the various motif lengths according to time period.

The first method uses the finite motif length. The method of the paper created to allow the various motifs length of each is automatically has variable length temporal analysis of motifs that derived from the model 1 as the TMM temporal analysis of motif mixture which is the important setting of the hyper arguments.

TMM is the fixed duration so we are adding the variable time in $t(i)$ to above model diagram so that it can be easily find out the motifs in the various time sequence. The hyper argument has variation so that we can find out easily in the secular time stamps or time periods. Here the influences of the variability surround this variation expectation. A huge weight of the result is less variability. Define the possibility of the words uniformly given and eliminate the size of the shape is the main role. However reduce the shape acts the important role in the interference.



Graph 1



Graph 2

Graph 1 & 2 shows the weight truncated distribution with different values and exponential rate. This is used to control the size of the slope in the particular time period.

A Gama distribution with arguments $(T=T1,T2)$ is used as the earlier argument of every motif. It must to be corrected Gama conjugate with the weight in the truncated distribution. Argument lambda and Z is fixed. This conjugate condition in the face of the L (lambda and Z) is greater than analysed from the truncated exponential distribution. It is represented as the following expression

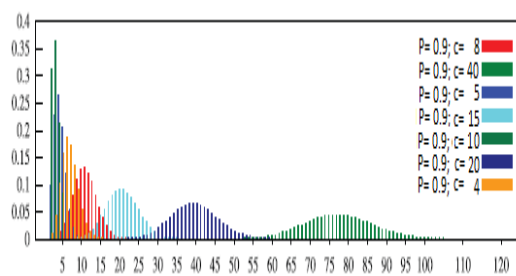
IV. INFERENCE

Here we are going to explain the stages about the inference. A pass over the stage executable for the VLTMM and TMM method will be performed publically. To perform the inferences uses the confused Gibbs sampling $\{o_{ji}, k_{jo}, ost_{jo}, \lambda k\}$. The balance variable has logical integrating sampling and can integrate the motifs

in their self used as H in the time series TRVSPT process distribution.

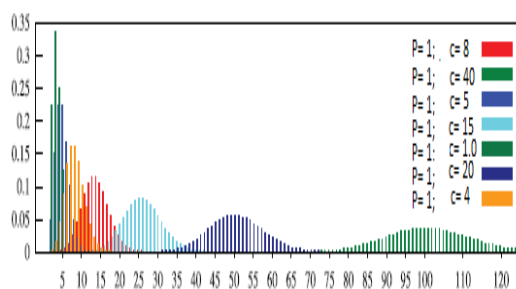
According to the sampling possibility organize and observer the mention earlier appearance is propositional to the two quantities. The first quantity is according to the MRST and CRP on the appearance and related to the number of repentance associated in the occurrence. The next step has to be comes in the likelihood of the particular observation then its virtual association and in the particular repentance.

The different method to crate the new repentance for this example is the Chinese restaurant process as its inverse to α . This possibility of the repentance is the inverse of the likelihood observation to the control of the hypothesis joined with random repentance. The linear process has all starting time control to integrate the initial time. Here we have a MRST and motif already as in the above example the possibility has inverse to the occurrence of the counting. Conjugate the MRST distribution H over motifs. We control to integrate the finding of motifs drawn to H as in Graph 3.



Graph 3

Distribution of the number of mixture elements sufficient to cover a proportion P of the total weight



Graph 4

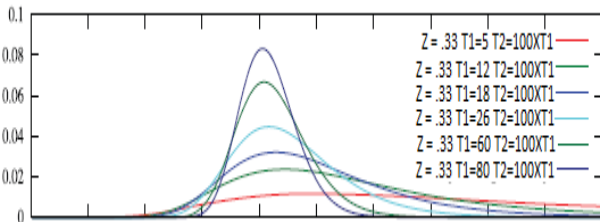
Distributions are shown for different concentration c and proportions P (90 and 95 percent)

V. MEANING AND SETTINGS OF PARAMETER

In our model we are using the different various arguments that can set to the influence of the inference. According to VLTMM method has taken the arguments in hierarchal manner of the TMM the argument has directly converted into the number of motifs. The

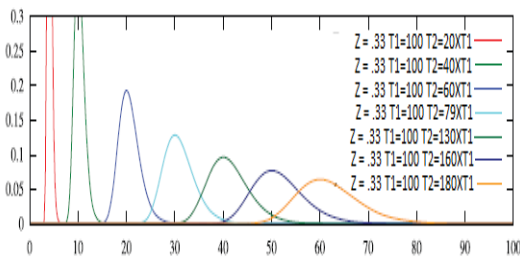
isolation of the parameter gets the more important. So that controls the number of repentance in a document that might depend on the documents duration. Examples taking from the Gibbs sampling method are to get the data and large amount of the data repeated or overlap in the particular time frame. The consequences are data set independent of the document and take the reference the small values.

The fixed weight truncation Z is the structure argument for weight truncation method. It manages the structure of the temporal in the motif. Weight exponential method is less support. To choose Z value considers the q ratio in between the values of the distribution.



Graph 5

Maximum motif length: Distribution of maximum motif length L_k ; Z when varying the prior $_$ and keeping the parameter Z fixed



Graph 6

T can be used to control the location and spread of the range of prior acceptable values for L_k ; Z .

VI. VIDEO DATA ANALYSIS

Our method explains the experiments about the video data. Here we show the temporal documents from the input video talking about the temporal timing duration and method aspects of our model. Also establishes the interest and result of model time in the motif by comparing the results with other methods.

a) Videos Convert To Temporal Document

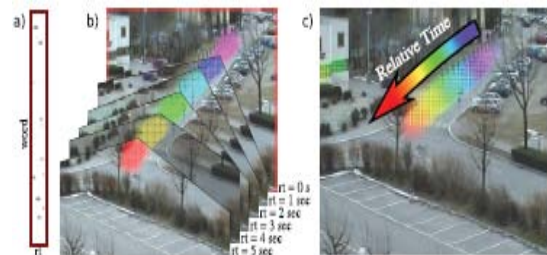
Created temporal documents has extracted the number of words at every time. Time step for the impermanent document use the pixel resolution for one second. One method will be our vocabulary directly using to low level. Result in a huge vocabulary with heavy temporal leads to a high disturbance computing

load we capture the data in the low level feature of co-occurrence and convert to high level word. To eliminate the confusion about the notation use the super script in the word from the low-level layer.

In the first stage of feature extraction we have to extract the flow feature of the optical dense image grid. We place or store the pixels where the motion could be happened or detected. We divide the quantize in to nine categories of one to eight commonly quantize the flow of direction to prepare the slow motion. The low level character has used to define the position and image motion category. The size of the low level word should be high but however we reduce the words in to nearly 25000 words. Because we are considering the words only we run the slid window for second five to forty frames to get depending on the data set without dead lock and collect every time stamp in the particular window.

Details on dimensionality reduction getting the set of document have to be applied the probabilistic latent semantic method. This method takes the input words and count every document learn the set of data words by the defined distribution on low level word to corresponding to the soft cluster words that repeated words in the document. Scene has fixed here because we are reduced the dimensionality reduction.

PLSA method is the entire processes of this stage learns the new video documents and give the decomposition of the every document mixture of the earlier method the topic weights given by the distribution. We are use this data reweighted by according to the activity use to build the documents constitute the input method.



Divided in to two five-second motifs.

Figure 5

Here the video is factorized to full length and it is divided into the possible motifs and to full duration of the time motif. At last the motif duration has been increased and recovers the motif properly.

Use of the motif duration arguments VLTAMM provides the approach to deal with the short coming of the fixed motif duration. Variable length with motif duration with VLTAMM shows the sum of the result.

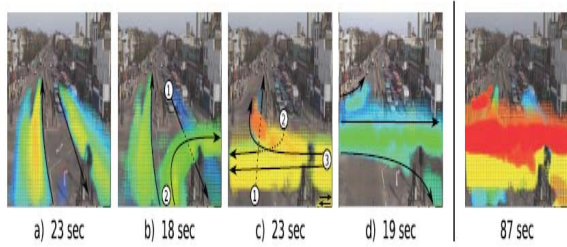


Figure 6: This figure shows how the junction data is analyzed by VLTAMM in a real time activity

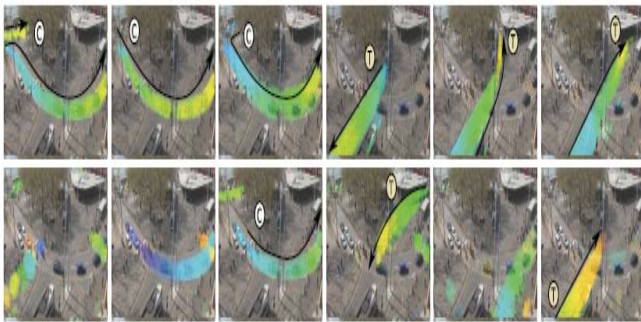


Figure 7: This shows motif is the 98 percentage of the observation one car coming from the left image and take the right direction of the road. 2 cars going the straightway and 3 cars taking the top turn, 4 car move from the traffic light it is the all motif taking at the different plane

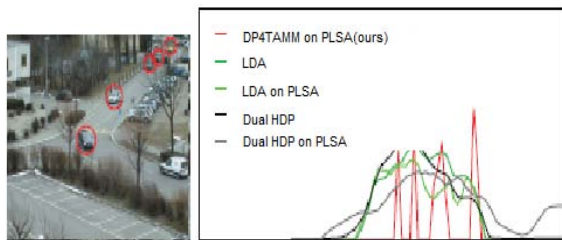


Figure 8

Figure 9 shows difference of the present and selected motif consider in the car activity. That image coming from the figure 8.

Activity diary and abnormality reporting is used to find the real motif when they occur at the real time. Here we take the example for the length of the video of 30 minutes and extract the motif and find out the motif time in the particular video scene. After getting the scene combine together and find out the motif occurrence.

VII. COMPARISON WITH OTHER TYPES

One disadvantage of other existing method is the difficulty to find out the motif and it's recurrent. Were as to find the temporal document ahs to be build the independent documents in the information is neglected here using LDA HDP-LDA. PLSA is used for the process method but using long temporal window it has

neglected the time ordering to recover the motif and do not carry the temporal information and temporal granularity noise of the video.

The second model is time order sensitive LDA, here same slide window use the modification Cartesian product the original vocabulary and relative time within the particular window. The issue of this model are documents considering only in the independent and there is no necessity to align the data with the original activities in the video. So that the real time activity assume happen at different places. Our approach doesn't have any disadvantage compare to the above methods. Our method has store the temporal data independent and starting of their time of the actual scene of the video.

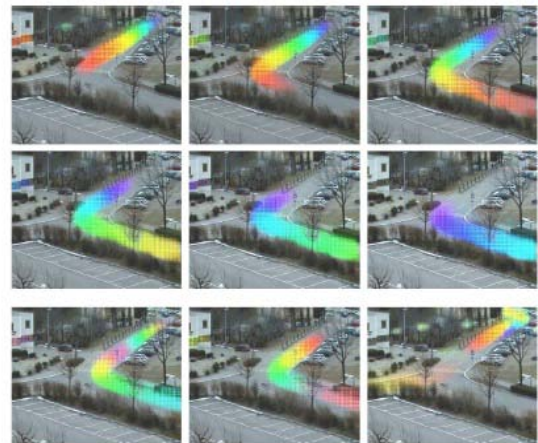


Figure 9: It shows the different activity and temporal activity at different time of the real scene

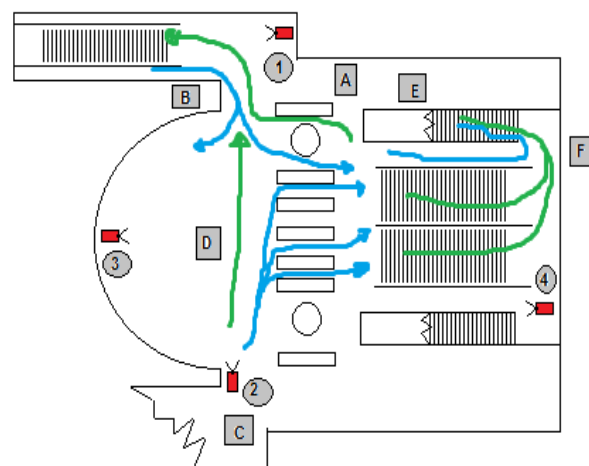


Figure 10: Architectural of the process to find the location to analyse the video samples and camera locations

a) Analysis of Multi camera calibration

To capture the images at the different camera position to analyse the temporal dependencies, the process use more number of cameras to integration method and join the all camera image and process the

low level features using the low level count matrix. Low level camera calibration has the possibility to take large amount space because of the different camera view, we are using the three hundred or high level motifs. Low level support in the normal camera view suppose the overhead as in the camera 2 and camera 2 low level method has span the camera and make it as the two view and choose the nosily view and random occurrence. This activity has solved if larger amount of training data has used in this example

Recovered motif in multi camera these motif has resent the 78 percentage of the observation has find the actions and find the activity capture automatically.

Evaluation of the timing information has arises at the time that matches the original timing of the real image. But it is the problem to find out the different camera our model has find out very easily it recover the people activity between the two camera views. The start and end projection has displayed in the motif background process. Then we can easily calculate the difference between the original image activity and the motif representation.

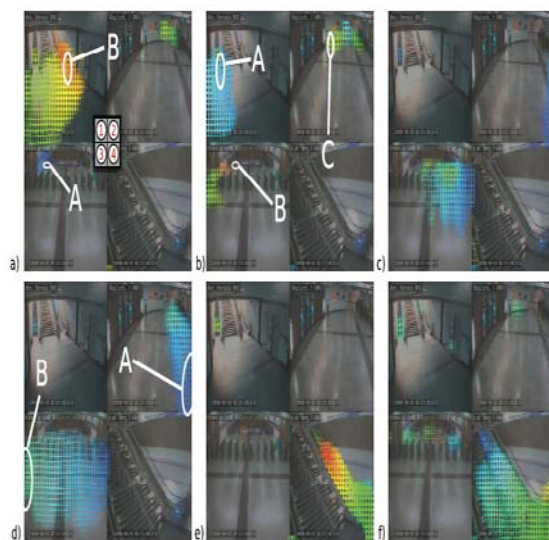


Figure 12

Figure 12 visual of the different camera views this map has taken from the metro activities. Here camera has capture the motif and who enter in the station and who are all comes out from the station has

taken and clipped by the different camera angle and it represent the motif at the time level duration.

For example the result we are taken from the station about nearly two hours of unlabeled scenes.

Table 1: shows the recover the motif from the typical path duration during the experiment

Path		Measured Duration					Duration Motif
Motif	Start end	average	std	min	max	median	motif
A	a-b	26	2.8	30	32	28	24
B	a-b	28.5	5.2	25	43	27	22
C	a-c	19.9	9.0	15	38	18	12
D	a-b	9.5	1	9	16	8.5	6

Recover motif has compared and the presence of the repentance is extract these data has used to remove the multiple sequence of the data motif and convert the low level into high level of the motif using in

the larger data. So that the repentance should be very high. The place of the track let information decreases the suspicious matches.

VIII. CONCLUSION

This model shown is new concept of finding the redundant temporal patterns in the particular time series. This method has used to find the joint and repeating motifs in the particular document. Identify how many common motifs present in the document and number of times motif occur in the document and also estimate the motif duration.

This method has validated the broad range of the data set and real time activity like traffic signals, micro phone pair, and video data's. Here we explained that the video data is activities in the traffic signals such as the movement of the car and typical person in the metro train station. Apply the simultaneous image on different camera view information without any calibration.

Audio signal coming from the two microphones in our model has used to recover the interest activities and yielded detection and precision. Using the artificial data assuming the robust model in the various hyperactive parameters has produced the meaning full information at the various situation applied in the variety of the data with success rate.

IX. FUTURE WORK

In future, this proposed model can be improved at the different levels to find the repeating values but it is not applied on the global activities. Suppose we are using the traffic signals we have to incorporate the heterogeneous methods to find out the recurrent motifs detection approaches. Then we will emphasise the different cycles of motifs and relation data for discover patterns. To scale up the input camera footage polynomial data will be concentrated in the upcoming approaches in time series.

REFERENCES RÉFÉRENCES REFERENCIAS

1. R. Emonet, J. Varadarajan, and J. Odobez, "Extracting and Locating Temporal Motifs in Video Scenes Using a Hierarchical Non Parametric Bayesian Model," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2011.
2. Y. Yang, J. Liu, and M. Shah, "Video Scene Understanding Using Multi-Scale Analysis," Proc. IEEE Int'l Conf. Computer Vision, 2009.
3. Wang, X. Ma, and E.L. Grimson, "Unsupervised Activity Perception in Crowded and Complicated Scenes Using Hierarchical Bayesian Models," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 31, no. 3, pp. 539-555, Mar. 2009.
4. J. Varadarajan and J. Odobez, "Topic Models for Scene Analysis and Abnormality Detection," Proc. 12th IEEE Int'l Conf. Computer Vision Workshop on Visual Surveillance, 2009.
5. D. Kuettel, M.D. Breitenstein, L.V. Gool, and V. Ferrari, "What's Going On? Discovering Spatio-Temporal Dependencies in Dynamic Scenes," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2010.
6. D. Blei and J. Lafferty, "Dynamic Topic Models," Proc. 23rd Int'l Conf. Machine Learning, 2006.
7. Wang and A. McCallum, "Topics over Time: A Non-Markov Continuous-Time Model of Topical Trends," Proc. Conf. Knowledge Discovery and Data Mining, 2006.
8. T. Hospedales, S. Gong, and T. Xiang, "A Markov Clustering Topic Model for Mining Behavior in Video," Proc. IEEE Int'l Conf. Computer Vision, 2009.
9. J. Li, S. Gong, and T. Xiang, "Discovering Multi-Camera Behaviour Correlations for On-the-Fly Global Activity Prediction and Anomaly Detection," Proc. IEEE 12th Int'l Conf. Computer Vision Workshop on Visual Surveillance, 2009.
10. T.A. Faruque, P.K. Kalra, and S. Banerjee, "Time Based Activity Inference Using Latent Dirichlet Allocation," Proc. British Machine Vision Conf., 2009.
11. J. Varadarajan, R. Emonet, and J. Odobez, "Probabilistic Latent Sequential Motifs: Discovering Temporal Activity Patterns in Video Scenes." Proc. British Machine Vision Conf., 2010.
12. Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei, "Hierarchical Dirichlet Processes," J. Am. Statistical Assoc., vol. 101, no. 476, pp. 1566-1581, 2006.
13. X. Wang, K. Ma, G. Ng, and W. Grimson, "Trajectory Analysis and Semantic Region Modeling Using a Nonparametric Bayesian Model," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2008.
14. T.S.F. Haines and T. Xiang, "Delta-Dual Hierarchical Dirichlet Processes: A Pragmatic Abnormal Behaviour Detector," Proc. IEEE Int'l Conf. Computer Vision, 2011.
15. E. Jouneau and C. Carincotte, "Particle-Based Tracking Model for Automatic Anomaly Detection," Proc. 18th IEEE Int'l Conf. Image Processing (ICIP), 2011.
16. O. Javed and M. Shah, "Tracking in Multiple Cameras with Disjoint Views," Automated Multi-Camera Surveillance: Algorithms and Practice, pp. 1-26, Springer, 2008.
17. C.C. Loy, T. Xiang, and S. Gong, "Multi-Camera Activity Correlation Analysis," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 1988-1995, 2009.
18. X. Wang, K. Tieu, and W. Grimson, "Correspondence-Free Multi-Camera Activity Analysis and Scene Modeling," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2008.
19. X. Wang, K. Tieu, and E. Grimson, "Correspondence-Free Activity Analysis and Scene Modeling in Multiple Camera Views," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 31, no. 1, pp. 893-908, Jan. 2009.

20. P. Marmaroli, J.-M. Odobez, X. Falourd, and H. Lissek, "A Bimodal Sound Source Model for Vehicle Tracking in Traffic Monitoring," Proc. 19th European Signal Processing Conf., 2011.
21. C. Wang and D. Blei, "Decoupling Sparsity and Smoothness in the Discrete Hierarchical Dirichlet Process," Proc. Advances in Neural Information Processing Systems, 2009.
22. R. Emonet, J. Varadarajan, and J.-M. Odobez, "Multi-Camera Open Space Human Activity Discovery for Anomaly Detection," Proc. Eighth IEEE Int'l Conf. Advanced Video and Signal Based Surveillance, p. 6, Aug. 2011.
23. B. Zhou, X. Wang, and X. Tang, "Random Field Topic Model for Semantic Region Analysis in Crowded Scenes from Tracklets," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2011.
24. J. Varadarajan, R. Emonet, and J.-M. Odobez, "Bridging the Past, Present and Future: Modeling Scene Activities from Event Relationships and Global Rules," Proc. IEEE Conf. Computer Vision and Pattern Recognition, June 2012.





Classification of HRS using SVM

By Astha Ameta & Kalpana Jain

College of Technology and Engineering

Abstract- The kidney diseases are one of the main causes of death around the world. Automatic detection and classification of kidney related diseases are important for diagnosis of kidney irregularities. Hepatorenal Syndrome (HRS) is a life-threatening medical condition when kidney fails due to liver failure. The treatment to such cases is liver transplant, or dialysis for temporary basis. This paper proposed to apply the Support Vector Machine (SVM) classification for diagnosis of HRS. The results were evaluated using realistic data from hospitals. RBF kernel function is used along with SVM. The results show a significant accuracy of 95%.

Keywords: *support vector machine, RBF kernel, cross validation, accuracy, ROC curve.*

GJCST-C Classification: *H.5.5, D.2.5*



Strictly as per the compliance and regulations of:



Classification of HRS using SVM

Astha Ameta^α & Kalpana Jain^σ

Abstract- The kidney diseases are one of the main causes of death around the world. Automatic detection and classification of kidney related diseases are important for diagnosis of kidney irregularities. Hepatorenal Syndrome (HRS) is a life-threatening medical condition when kidney fails due to liver failure. The treatment to such cases is liver transplant, or dialysis for temporary basis. This paper proposed to apply the Support Vector Machine (SVM) classification for diagnosis of HRS. The results were evaluated using realistic data from hospitals. RBF kernel function is used along with SVM. The results show a significant accuracy of 95%.

Keywords: support vector machine, RBF kernel, cross validation, accuracy, ROC curve.

I. INTRODUCTION

Hepatorenal Syndrome (HRS) is a major complication of Cirrhosis, where approximately 8% patients with ascites are annually incident. HRS starts developing at the latest phase of disease. It is now medically proven that it is a very important determinant for showing survival rate. A majority of reviews on HRS reflect the problems in the investigation of this syndrome. On the contrary, HRS has no experimental model. Hence, many of its aspects are still poorly understood.

A high degree of predictive accuracy is needed in the healthcare sector. The predictive accuracy of any data mining/Machine learning technique is based on the data, its quantity and quality. Techniques such as classification, clustering, time series, temporal analysis, association and correlation analysis are various data mining techniques taken into consideration. Classification techniques are used to analyze data and predict labels that describe important properties of data. Many classification techniques have been developed such as Naïve Bayes, k-NN, SVM, Decision Tree induction, Back propagation, and more. Here, we propose SVM technique to be used for diagnosis of HRS.

II. SUPPORT VECTOR MACHINE

SVM, abbreviated as Support Vector Machine, is a class of learning methods that can be used for the purpose of classification. Many classifiers have been proposed in the literature to study classification problems. In training SVMs, decision boundaries are

directly determined from training data thus maximizing its generalization ability. Hence, ability of SVM to generalize is somehow different than those of other classifiers, usually in the case of small number of training data. In its simplest or linear form, SVM is defined as a hyperplane which separates a set of negative examples from set of positive examples by using the concept of maximize the class margin. The form in which data points are provided is $\{(y_1, x_1), (y_2, x_2), \dots, (y_n, x_n)\}$, where x_i is a vector of n-dimensions and y_i can either be 1 or -1, which denotes the class to which point x_i belongs. For training SVM, set of x_i are pre-labeled with y_i components which denotes the correct classification which is required by SVM to search for a separating hyperplane.

For the case where data are linearly separable, two hyperplanes, $w \cdot x - b = -1$ and $w \cdot x + b = 1$ are generated which are parallel. Thus, no training sample lies in between and distance is maximized for the two planes. In the quadratic form, it can be formalized as:

$$\text{Min } \frac{1}{2} \|w\|^2$$

$$\text{Subject to } y_i(w \cdot x_i - b) \geq 1, 1 \leq i \leq l.$$

This is a convex problem. Its dual form is:

$$\text{min } \frac{1}{2} \alpha^T Q \alpha - e^T \alpha$$

$$\text{subject to } y^T \alpha = 0 \text{ and } \alpha \geq 0,$$

where Q is an $l \times l$ matrix with $Q_{ij} = y_i y_j x_i \cdot x_j$ and e is the vector of all ones. Let α be the solution to dual problem, then $w = \sum_{i=1}^l y_i \alpha_i x_i$ is a solution to the primal problem. Vectors x_i , which corresponds to $\alpha_i > 0$, lie on the margin. Such vectors are termed as support vectors (SV). Once the above equations are resolved, then new items can be classified with $w \cdot x$ where x is the new sample vector that is to be classified.

For the case of non-linear separable data, Cortes and Vapnik ([14]) proposed a modification to the QP formulation (namely soft margin) according to which, examples that fall on the wrong side of the decision boundary are allowed but with a penalty. Boser et al. ([15]) also proposed an extension to the non-linear classifiers. A generalized form of the QP problem having soft margin along with nonlinear classifier is shown below:

$$\text{min } \frac{1}{2} \|w\|^2 + C \sum \epsilon_i$$

$$\text{subject to } y_i(w \cdot \phi(x_i) - b) \geq 1 - \epsilon_i$$

$$\text{and } \epsilon_i \geq 0, 1 \leq i \leq l,$$

where ϵ_i shows the training error and the parameter C is used to adjust the training error and the regularization term $1/2 \|w\|^2$. The function ϕ maps \mathfrak{R}^n to a higher

Author α : Department of Computer Science and Engineering, College of Technology and Engineering, Udaipur.
e-mail: Ameta.astha@gmail.com

Author σ : Assistant Professor, Department of Computer Science and Engineering, College of Technology and Engineering, Udaipur.
e-mail: Kalpana_jain2@rediffmail.com

dimensional space. In practice, kernel functions are used to perform the process of mapping. The kernel functions are represented in the form of dot product as below:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j).$$

Some commonly used kernel functions include Linear:

$$k(x_i, x_j) = x_i \cdot x_j$$

Polynomial:

$$k(x_i, x_j) = (x_i \cdot x_j)^d$$

Radial Basis Function (RBF):

$$k(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2), \gamma > 0$$

Syndrome (HRS) based on clinical data. We have collected data for 100 patients from few hospitals. For each patient data, there are 14 features, including serum albumin, billirubine, creatinine, serum sodium, serum urea, urine output, urine microscopy, USG, ascites, cirrhosis, BP-systolic, diastolic, hemoglobin, urine protein. The data collected in medicine is generally collected because of patient care activity so as to benefit patients; hence data contained in medical databases is redundant, irrelevant, and inconsistent which can affect the results produced with the use of data mining techniques. Thus, data preprocessing and scaling are required so as to remove redundant as well as noisy data and to use normal forms of data. All of the data were transformed to real values with proper definition. For example, "Normal" converted to 1 and "Abnormal" to 0.

III. PROPOSED METHODOLOGY FOR CLASSIFICATION OF HRS USING SVM

In this paper, we propose to use Support Vector Machines (SVMs) for the diagnosis of Hepatorenal

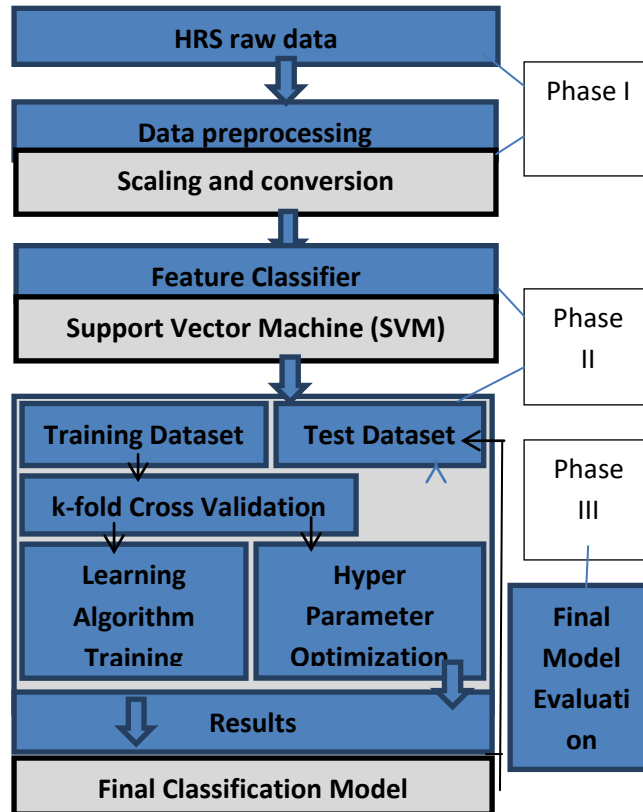


Figure 1: Architecture

The results obtained provide good classification accuracy. Figure 1 shows the architecture of our proposed work. Flowchart for proposed methodology can be described as the following phases:

Phase I:

- a) HRS clinical data is collected and preprocessed. Preprocessing of proposed work includes:

- Conversion of string data to numeric form:
 1. Data value "Normal" is converted to 1 and "Abnormal" to 0.
 2. Data value "Yes" is converted to 1 and "No" to 0.
- Scaling: Conversion of urine output in milliliter to liter.

Phase II: SVM is used as the classification technique.

- a) The preprocessed dataset is divided into training set(contains 80% of data) and testing set(contains 20% of data).
- b) Cross-Validation or CV is applied on training dataset. In the proposed work, k-fold cross validation is used where k=5, hence it is known as five-fold cross validation.
- c) In the proposed work, grid search method is employed to find the best parameters. Mesh grid is used to employ grid search.
- d) A final model is obtained which is ready to test for new or unseen data.

Phase III.

The final model obtained is tested on new or unseen data. This is known as final model evaluation. The accuracy hence obtained is considered as the accuracy of the model generated and it shows how much accurate and efficient model has been generated.

IV. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

We used Support Vector Machine as the classification technique using LIBSVM – Matlab interface for our experiment. LIBSVM is an SVM package provided by Matlab. The computations involved were implemented on intel core i5 processor. The kernel function used here is Radial Basis Function (RBF) kernel, also known as sigmoid kernel.

Accuracy is evaluated using k-fold cross validation test. K-fold Cross-validation process includes dividing a dataset into k pieces, and on each piece, testing the performance of a predictor build from the

remaining 90% of the data. In our work, k=5. The performance of the classification is evaluated for six parameters, namely, accuracy, sensitivity, specificity, precision, recall, f-measure. The definitions are as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{1}$$

$$Sensitivity = \frac{TP}{TP+FN} \tag{2}$$

$$Specificity = \frac{TN}{TN+FP} \tag{3}$$

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

$$F - Measure = \frac{2TP}{2TP+FP+FN} \tag{6}$$

where TP represents number of true positives (If the instance is positive and it is classified as positive), TN represents number of True negatives (If the instance is negative and it is classified as negative), FP represents number of False positives (If the instance is negative but it is classified as positive) and FN represents number of False negatives (If the instance is positive but it is classified as negative).

Figure 2 shows cross-validation accuracy of 95%. This is a curve between logarithm of two important parameters, cost function C and rbf sigma, also known as gamma, represented by γ . The best value of both these factors gives the best cross validation accuracy of 95%.

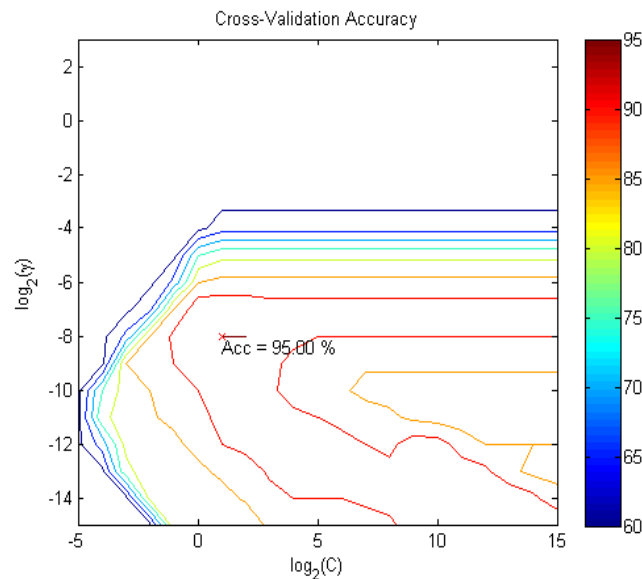


Figure 2: Accuracy Curve

The performance of the classifier can be visualized using Receiver Operating Characteristic (ROC) curve. The 2-D ROC curve is defined by the false

positive rate (FPR) on x-axis and true positive rate (TPR) on y-axis, where TPR determines a classifier performance on classifying positive instances correctly

among all positive samples and FPR, on the other hand, defines how many incorrect positive results occur among all negative samples. It is also known as graph between sensitivity and 1-specificity. Figure 3 shows

ROC curve obtained for proposed work. The area under the ROC curve (AUC) obtained is 0.95. This value of AUC proves that the performance of classifier is good.

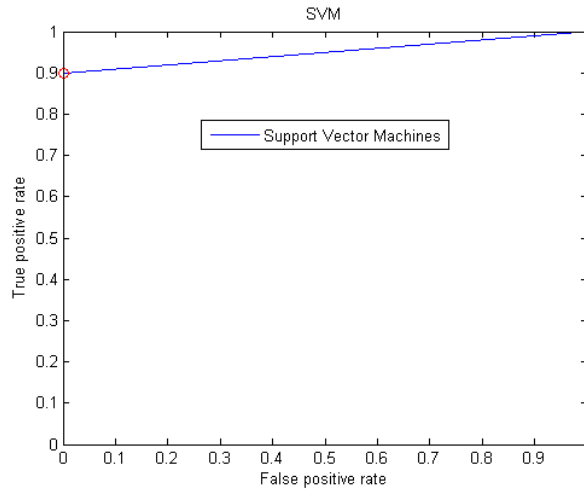


Figure 3: ROC Curve

Figure 4 shows various performance parameters in the form of a bar chart with their experimental values.

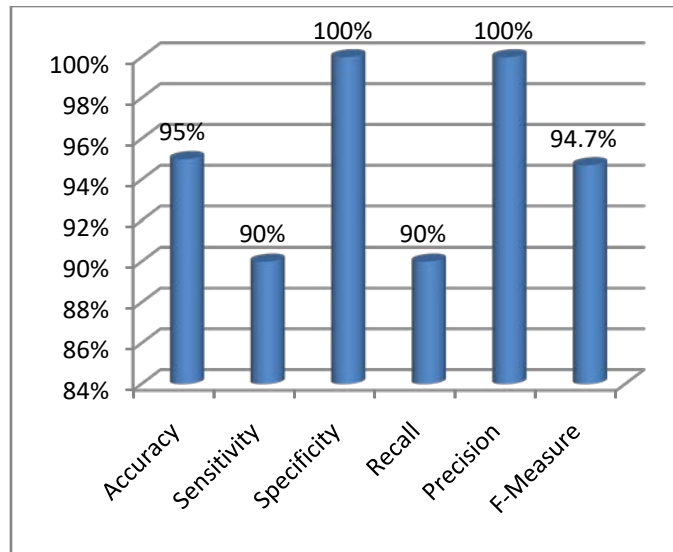


Figure 4: Performance parameters

V. CONCLUSION

In this research work, we propose to use SVM as the classification technique to diagnose HRS in patients of Cirrhosis. The performance is analyzed by comparing the predicted results with the manual results received along with data sets from hospitals. Our approach provides 95% classification accuracy and precision is recorded as 100%. It helps physician to diagnose the disease with more precision and accuracy. Sensitivity and Specificity are computed as 90% and 100% respectively. Recall and F-Measure are measured as 90% and 94.74% respectively. Thus, SVM is proven as a good classifier for the prediction of HRS.

The proposed work can be further extended using feature selection or optimization techniques. Another extension can be application of SVM for diagnosis of similar diseases.

REFERENCES RÉFÉRENCES REFERENCIAS

1. Chen, X., Ching, W. K., Aoki-Kinoshita, K. F., & Furuta, K. (2010, May). Support Vector Machine Methods for the Prediction of Cancer Growth. In *Computational Science and Optimization (CSO), 2010 Third International Joint Conference on* (Vol. 1, pp. 229-232).IEEE.
2. Balakrishnan, S., Narayanaswamy, R., Savarimuthu, N., & Samikannu, R. (2008, October). SVM ranking

- with backward search for feature selection in type II diabetes databases. In *Systems, Man and Cybernetics, 2008.SMC 2008. IEEE International Conference on* (pp. 2628-2633). IEEE.
3. Liu, J., Yuan, X., & Buckles, B. P. (2008, August). Breast cancer diagnosis using level-set statistics and support vector machines. In *Engineering in Medicine and Biology Society, 2008.EMBS 2008. 30th Annual International Conference of the IEEE* (pp. 3044-3047). IEEE.
 4. Ghumbre, S., Patil, C., & Ghatol, A. (2011, December). Heart disease diagnosis using support vector machine. In *International conference on computer science and information technology (ICCSIT) Pattaya*.
 5. Kousarrizi, M. N., Seiti, F., & Teshnehlab M., 2012. An experimental comparative study on thyroid disease diagnosis based on feature subset selection and classification, *International Journal of Electrical & Computer Sciences IJECS- IJENS*, **12**: 01.
 6. Hiesh, M. H., Andy, Y. Y. L., Shen, C. P., Chen, W., Lin, F. S., Sung, H. Y., Lin J.W., Chiu M. J. and Lai, F. (2013, July). Classification of schizophrenia using genetic algorithm - support vector machine (ga-svm). In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBC* pp. 6047-6050. *IJENS*, **12**: 01.
 7. Jiang H., Tang F., Zhang X., 2010. Liver cancer Identification based on PSO-SVM Model. *11th Int. Conf. Control, Automation, Robotics and Vision Singapore*.
 8. Harb H.M., Desuky A.S., 2014, Feature Selection on Classification of Medical Datasets based on Particle Swarm Optimization, *International Journal of Computer Applications (0975-8887)*, **104**.
 9. Tomar D. and Agarwal S., 2013. A survey on Data Mining approaches for Healthcare. *International Journal of Bio-Science and Bio-Technology*, **5**, 241-266.
 10. Kohli N., & Verma N.K., 2011. Arrhythmia classification using SVM with selected features, *International Journal of Engineering, Science and Technology*, **3**: 122-131.
 11. Han J. And Kamber M., 2000. data mining Concepts and Techniques, *Morgan Kaufmann Publishers*, pp. 337-342.
 12. D. Delen, G. Walker, A. Kadam, "Predicting breast cancer survivability: A comparison of three data mining methods," *Artificial Intelligence in Medicine*, vol. 34, 2005, pp. 113-127.
 13. C. Cortes and V. Vapnik, "Support Vector Networks," *Machine Learning*, 20, pp. 273-297, 1995.
 14. B. E. Boser, I.M. Guyon, and V. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Fifth Annual Workshop on Computational Learning Theory*, ACM, 1992.



This page is intentionally left blank



Study of Effective Scheduling Algorithm for Application of Big Data

By Tanmay Paul

Adamas Institute of Technology

Abstract- In this new era with the advancement in the technological world the data storage, analysis becomes a major problem. Although the availability of different data storage component like electronic storage such as hard drive or virtual storage such as cloud still the problems remains. The major issue is processing the data because usually the data is in several format and size. Usually processing such huge amount of data with several formats can be time consuming. Using of application such as Hadoop can be beneficial but using of scheduling algorithm can be the best way to for data set analysis to make the process time efficient and analysis the requirement of different scheduling algorithm for the specific data set. In this paper we analysis different data set to explain the most effective scheduling algorithm for that specific data set and then store and execute data set after processing.

Keywords: *big data, hadoop, scheduling algorithm, data analysis, HDFS, FCFS.*

GJCST-C Classification: *H.2.8*



Strictly as per the compliance and regulations of:



Study of Effective Scheduling Algorithm for Application of Big Data

Tanmay Paul

Abstract- In this new era with the advancement in the technological world the data storage, analysis becomes a major problem. Although the availability of different data storage component like electronic storage such as hard drive or virtual storage such as cloud still the problems remains. The major issue is processing the data because usually the data is in several format and size. Usually processing such huge amount of data with several formats can be time consuming. Using of application such as Hadoop can be beneficial but using of scheduling algorithm can be the best way to for data set analysis to make the process time efficient and analysis the requirement of different scheduling algorithm for the specific data set. In this paper we analysis different data set to explain the most effective scheduling algorithm for that specific data set and then store and execute data set after processing.

Keywords: big data, hadoop, scheduling algorithm, data analysis, HDFS, FCFS.

I. INTRODUCTION

In the data analysis the efficiency plays the most important factor and the development in the data storage, analysis efficiency in the stipulated time and the endeavor for the output of data analysis in the executional environment and storage of that data is defined as Hadoop distributed file system (HDFS) [1]. The application comprises of certain sub system application which reshape data in terms of times which are analyzed using scheduling algorithm MinMin [2], minimum completion time (MCT) [3]. In HDFS huge amount of data can be stored which provides cost effective and also reliability. In first come first serve (FCFS) [4] the big data changes dynamically for the application access which consists of different speed and size. In order to execute in an executional environment HDFS is implemented. HDFS allow large storage and data analysis but the problem is to process a large amount of data. In the computing environment HDFS gives efficient data analyzing, storage, execution. Scheduling algorithm administer data work flow within time constraints. Scheduling algorithm FCFS, Distributed heterogeneous earliest finish time(DHEFT) works unsurpassable for given set of data in cloud environment. Performance and data analysis is done by scheduling algorithm. For checking the performance of the specific data set the algorithm must be known to priori for ease in implementation and time effective manner. The task scheduling is executed single task at

a time so that performance of the entire scheduling algorithm executed can be manifest. VM can execute single task at single time.

II. SYSTEM ARCHITECTURE

a) System Workflow

The data set is given as input for execution. Each data set is converted in smaller subtask and the entire subtasks are dependent on each other. The subtask is executed in sequential manner as every subtask execution is completed only after the execution of the previous subtask. The new subtask waits until the execution of the previous sub task gets completed. Below in fig 1 data execution work flow is given.

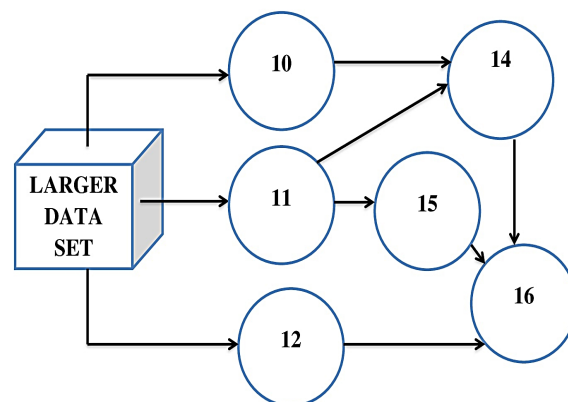


Fig.1: Data execution work flow

b) Cloud Server Model

Cloud server [4] is the primary module which is the resource provider for performing the processing activity. Virtual machine (VM) constructed can be accessed only by the registered user. Once the file is received in the VM it is divided into the subtasks. VM executes single task and the remaining task is shared between VM through round [5] robin algorithm.

c) Scheduling Algorithm

Scheduling algorithm is implemented to VM to utilize the resources effectively so that no VM is ideal mode of operation. Initially during task assigning we have to assign proper VM for the specific task and also the resource mapping for execution. There are various scheduling algorithm which can be implemented for large data set to be examined their performance. The

scheduling algorithms are MinMin algorithm, Data aware scheduling algorithm, MaxMin scheduling algorithm, first come first serve (FCFS) scheduling algorithm, MCT algorithm, and heterogeneous earliest finish time (HEFT) algorithm.

- **MinMin Algorithm**

In MinMin algorithm [2] task is arranged in ascending order with least or minimum time of completion and the resources or VM are allocated to the fastest job and this process is looped until all the jobs are scheduled to the VM.

- **Data Aware Scheduling Algorithm**

In data aware scheduling algorithm [6] the data is stored in the VM which is vacant to the resources which are closest to be executed by the VM. It eliminate over utilization of time in scheduling the task one by one.

- **MaxMin Scheduling Algorithm**

In the MaxMin [2] algorithm task is arranged in descending order with maximum time of completion for task allocation. It is in actually the opposite of the MinMin algorithm.

- **First Come First Serve (FCFS) Algorithm**

In the first come first serve [7] algorithm the task scheduled in a queue and allocated according to first come first serve basis not according to the VM efficiency or maximum or minimum time completion. The only disadvantage of these is if the task which is longer executed in the VM then the smaller task has to wait for the longer task to be executed.

- **MCT Algorithm**

In MCT algorithm [3] the task assigned to the resources or the available VM get executed with minimum time.

III. EXPERIMENTAL ANALYSIS

In this paper the performance of various scheduling algorithm is analyzed on big data. Cloudera [8] has been used as a platform for analyzing in eclipse [9] environment. Three VM has been created for user registration. The task available is allocated to VM by the server and all tasks which are in queue are allocated simultaneously to all available VM at same for the execution purpose. Dynamic data set has been used to performance evaluation analysis which composed of data of different size and set using various scheduling algorithm. In the evaluation of data set two parameters has been considered firstly the delay and secondly the task span. Delay may occur due to two major causes firstly system failure secondly due to low system memory in comparison to the task allotted because every time there is input in the data set there is change in size due to big data which can be sometimes non compatible. We have considered three cases 12Kb, 22Kb and 55Kb of data set. In the first case 12Kb data set we implemented all the scheduling algorithm where

x-axis defines the scheduling algorithm and y-axis defines the time. The entire scheduling algorithm differs with each other. Make span comprises of addition of data processing time, time taken for data transfer from storage to execution, waiting time and time of computation.

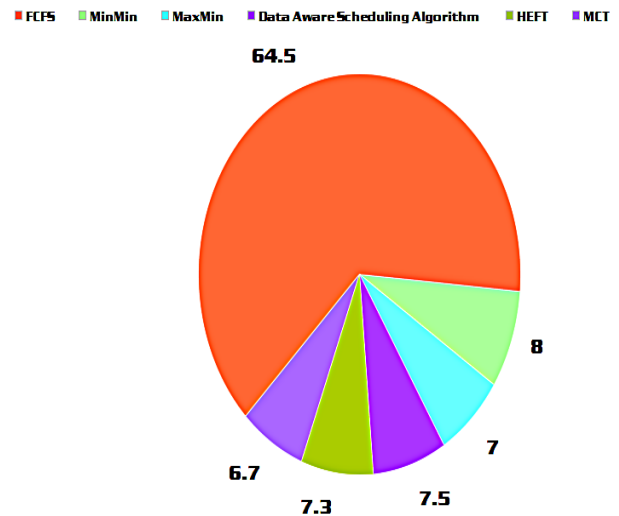


Fig.2: Data set of 12Kb by scheduling algorithm

The MinMin algorithm specify the task with minimum completion time to available VM and the MaxMin specify the task with maximum completion time available to VM so the task is assigned to the specific VM which has maximum computation time. Firstly we have a 12Kb data set which analyzed by different scheduling algorithm. Above a graph has been shown with data set of 12Kb by all the six scheduling algorithm where the FCFS performs the best. Secondly the data set chosen is 25kb here also MinMin and MaxMin performs same as above stated in Fig2. The performances changes for 25Kb size of data set. The change in task size challenges the computational capacity as the increase in data size. HEFT performs best for the sample data set. DASA performs least for the taken data set. The processing time, delay time makes DASA performance least. Below in Fig3 the 25Kb data set is shown by all the six scheduling algorithm.

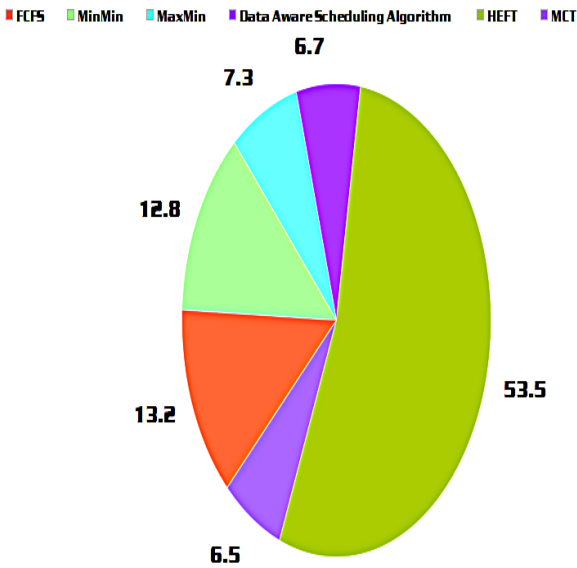


Fig.3: Data set of 25Kb by scheduling algorithm

Thirdly we have taken 55Kb size of data set and MCT performs better. The task is assigned to the available VM for execution but the execution time is taken a little more as the task is increased in terms of data size. MaxMin performance is constantly vice versa in compared to the MinMin and the remaining algorithm performs least in compared to the MCT. Below in Fig4 the 55Kb data set is shown by all the six scheduling algorithm.

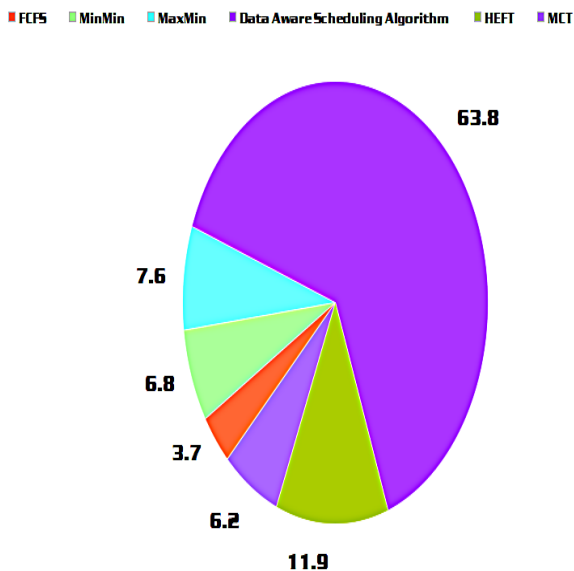


Fig4: Data set of 55Kb by scheduling algorithm

IV. CONCLUSION

The scheduling algorithm on data set of big data comprising FCFS algorithm, MCT algorithm, MinMin algorithm, DAS algorithm, HEFT algorithm is performed for analyzing. The result of performance

analysis varies differently with dynamic dataset. After the data analysis the data is stored in the form of HDFS in encrypted. For the future work various data set of different data size can be used for performance analyzing and assessment.

REFERENCES RÉFÉRENCES REFERENCIAS

1. Youngho Song; Young-Sung Shin; Miyoung Jang; Jae-Woo Chang, "Design and implementation of HDFS data encryption scheme using ARIA algorithm on Hadoop":2017 IEEE International Conference on Big Data and Smart Computing (BigComp).
2. E. Kartal Tabak; B. Barla Cambazoglu; Cevdet Aykanat, "Improving the Performance of Independent Task Assignment Heuristics MinMin, MaxMin and Sufferage" : IEEE Transactions on Parallel and Distributed Systems Year: 2014, Volume: 25, Issue: 5.
3. S. S. Guia; A. Espírito-Santo; V. Paciello; F. Abate; A. Pietrosanto "A comparison between FFT and MCT for period measurement with an ARM microcontroller" 2015; IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings.
4. Xueyan Tang, Yusen Li, Runtian Ren, Wentong Cai "On First Fit Bin Packing for Online Cloud Server Allocation":2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS).
5. Saptarshi Bhowmik, Sudipa Biswas, Karan Vishwakarma, Subhankar Chattoraj "An Efficient Load Balancing Approach in a Cloud Computing Platform": IOSR Journal of Computer Engineering (IOSR-JCE) Volume 18, Issue 6, Ver. VI (Nov.-Dec. 2016).
6. Jianhua Jiang; Gaochao Xu; Xiaohui Wei, "An Enhanced Data-aware Scheduling Algorithm for Batch-mode Data intensive Jobs on Data Grid":2006 International Conference on Hybrid Information Technology.
7. David Taylor-Fuller; Susan J. Lincke "A QoS comparison of 4G first-come-first-serve load sharing algorithms involving speech & packet data" :2007 IEEE International Conference on Electro/Information Technology.
8. Swati Yadav' SantoshVishwakarma, Ashok Verma "Efficient & Accurate Scheduling Algorithm for Cloudera Hadoop":2015 International Conference on Computational Intelligence and Communication Networks (CICN).
9. Jeroen Laverman, Dennis Grewe, Olaf Weinmann, Marco Wagner, Sebastian Schildt "Integrating Vehicular Data into Smart Home IoT Systems Using Eclipse Vorto" :2016 IEEE 84th Vehicular Technology Conference (VTC-Fall).

10. J. D. Monte K. R. Pattipati, "Scheduling parallelizable tasks to minimize make-span and weighted response time": IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans Year: 2002, Volume: 32, Issue: 3.





A New View on Classification of Software Vulnerability Mitigation Methods

By Maryam Mouzarani & Babak Sadeghiyan

Amirkabir University of Technology

Abstract- Software vulnerability mitigation is a well-known research area, and many methods have been proposed for it. Some papers try to classify these methods from different specific points of views. In this paper, we aggregate all proposed classifications and present a comprehensive classification of vulnerability mitigation methods. We define software vulnerability as a kind of software fault, and correspond the classes of software vulnerability mitigation methods accordingly. In this paper, the software vulnerability mitigation methods are classified into vulnerability prevention, vulnerability tolerance, vulnerability removal and vulnerability forecasting. We define each vulnerability mitigation method in our new point of view and indicate some methods for each class. Our general point of view helps to consider all of the proposed methods in this review. We also identify the fault mitigation methods that might be effective in mitigating the software vulnerabilities but are not yet applied in this area. Based on that, new directions are suggested for the future research.

GJCST-C Classification: H.3.4



Strictly as per the compliance and regulations of:



A New View on Classification of Software Vulnerability Mitigation Methods

Maryam Mouzarani ^α & Babak Sadeghiyan ^σ

Abstract- Software vulnerability mitigation is a well-known research area, and many methods have been proposed for it. Some papers try to classify these methods from different specific points of views. In this paper, we aggregate all proposed classifications and present a comprehensive classification of vulnerability mitigation methods. We define software vulnerability as a kind of software fault, and correspond the classes of software vulnerability mitigation methods accordingly. In this paper, the software vulnerability mitigation methods are classified into vulnerability prevention, vulnerability tolerance, vulnerability removal and vulnerability forecasting. We define each vulnerability mitigation method in our new point of view and indicate some methods for each class. Our general point of view helps to consider all of the proposed methods in this review. We also identify the fault mitigation methods that might be effective in mitigating the software vulnerabilities but are not yet applied in this area. Based on that, new directions are suggested for the future research.

I. INTRODUCTION

Software is an important part of a computer system. Being complex or created by incompetent developers, faults might be introduced to the software. There are faults that cause violating the system security. These faults are called vulnerability. There has been much research on preventing, detecting and analyzing software vulnerabilities.

By the time of writing this paper there is a number of surveys on the methods of mitigating vulnerabilities, i.e. [1], [2], [3] and [4]. Among them, [4] surveys the static analysis vulnerability detection methods that are applied in three areas that are associated with sources of vulnerabilities, i.e., access-control, information-flow and application-programming-conformance. It reviews around 88 papers. The studied methods, however, do not cover all the software vulnerability classes. Static analysis methods are also surveyed in [3]. It reviews 23 papers and classifies their methods with a different point of view. In [1] static and dynamic analysis methods are classified and 18 papers are briefly reviewed. The classification for static analysis methods presented in that paper is similar to the one in [3]. The most comprehensive survey is presented in [2]

by Shahriar et al. in 2012. They review 173 papers and classify their methods in four classes, i.e., static analysis, dynamic analysis, monitoring and hybrid analysis.

In this paper, we present a new definition for software vulnerability. Based on this definition, vulnerability mitigation methods are classified and reviewed with a new point of view. We use the general classification of fault mitigation methods as a base and extend it to a detailed classification of software vulnerability mitigation methods.

Our comprehensive classification aggregates many of the classification presented in the previous surveys, i.e., [1], [3], [2] and [5]. Also, the general perspective applied in our survey helps to identify the fault mitigation methods that are not yet used in mitigating software vulnerabilities. Since we consider the software vulnerability as a type of fault, these methods may be helpful in mitigating software vulnerabilities. We suggest new directions for the future researches based on our analysis during the review of the proposed vulnerability mitigation methods.

In this paper, our definition of software vulnerability is presented in section II. Based on this definition, software vulnerability mitigation methods are classified in section III. In this section, each class is described in details and some examples are reviewed. Section IV concludes the paper and presents some future directions.

II. DEFINING SOFTWARE VULNERABILITY

To review vulnerability mitigation methods, a precise definition of software vulnerability is required. Different researchers have suggested definitions for this term which are nearly analogous but have differences. Matt Bishop et al. define software vulnerability by modeling the software as a state machine in [6], [7], [8] and [9]. In this model, a vulnerable state is the state that let unauthorized reads, changes or accessibility modifications to a source. They define vulnerability as a property in the system that let it enter into a vulnerable state. In [8] Bishop defines vulnerability as a weakness that makes it possible for a threat to occur, where a threat is a potential violation of security policy. Amoroso defines vulnerability as an unfortunate characteristic that allows a threat to potentially occur [10]. There are other definitions of software vulnerability in relation with

Author α: Dept. of Computer Engineering & Information Technology, Amirkabir University of Technology, Tehran, Iran.
e-mail: mouzarani@aut.ac.ir

Author σ: Dept. of Computer Engineering & Information Technology, Amirkabir University of Technology, Tehran, Iran.
e-mail: basadegh@aut.ac.ir

security policy, e.g. [11] and [1]. Most of them define it as a property, characteristic or weakness that may cause compromising the security policy.

In order to clarify the terms property and security policy compromise, we redefine "software vulnerability". We use the precise definitions for the concepts in software security and reliability that are presented in [12] and construct our definition of software vulnerability. The taxonomy in [12] is presented in 2004 for the concepts of software security and reliability, such as fault, error, failure, vulnerability and attack. The authors define fault as the cause of error, while error is a state of the system that is probable to failure. Failure -or service failure is an event in which the delivered service is deviated from the correct service. In fact, a fault may become active and produce an error. Also the error may propagate inside the system and produce more errors. If the propagated error reaches system boundaries and affects the services, it becomes a failure.

A service is defined in [12] as the behavior perceived by users in system boundaries. Correct services are determined by the system specification. Some parts of the system behavior are specified by the security policy, which is a partial system specification. Thus when a system deviates from the security policy, a security failure occurs. This means that compromising security policy causes a security failure.

Faults are classified in [12] based on eight criteria, such as the phase of creation or occurrence, the objective, the phenomenological cause, the system boundary and the dimension. All combinations of the eight elementary fault classes would result in 256 different combined classes. The authors, however, believe that not all combinations are possible. For example, there is no malicious non-deliberate faults, or all the natural faults are non-malicious.

An attack is defined in [12] as a malicious external fault. An attack may be either an external hardware malicious fault, such as heating the RAM with a hairdryer to cause memory errors, or an external software malicious fault, such as a Trojan horse [12]. The term vulnerability is also defined in [12] as an internal fault that enables an external fault to harm the computer system, although harming the computer system is not clearly defined.

According to the previous definitions, we consider software vulnerabilities as:

Definition 2.1 Software vulnerabilities are internal faults that may cause a security failure.

We have concluded this definition, out of the definitions in [12], [8], [10], [11] and [1], since looking a vulnerability as a fault, instead of a property, better clarifies the concept of vulnerability by considering its relation to error, security failure and thus security policy. Like faults, a vulnerability may be dormant and never be

activated. It also may be activated and propagated in the system. The activated vulnerability might never reach the boundaries. As an example, suppose that a buffer overflow occurs and the value of a return address in the stack changes as a result. But using a monitoring procedure, the unauthorized change is detected and the program halts. Thus, the security policy is not violated. Monitoring the program, as a vulnerability detection method, is explained in section III-B. When an active vulnerability reaches the system boundaries, it causes a security failure. For example, an attacker may activate the format string vulnerability in a program and make it print some confidential data from the memory [13]. Since the active vulnerability has reached the system boundaries, it has made a security failure.

III. VULNERABILITY MITIGATION METHODS

Since vulnerability mitigation is a well-known research area, a structured approach is required to review the previous related works. In this paper, we review vulnerability mitigation methods using a new point of view. We classify and review these methods based on how we define software vulnerability. In the previous section, software vulnerability is defined as an internal software fault. Since we considered vulnerability as a type of fault, the classifications of fault mitigation methods can be used as a base for classifying vulnerability mitigation methods. Avizienis et al. present a classification for the means of mitigating the faults to achieve a secure and dependable system in [12]. We use this general classification as a base and extend it into a detailed classification of vulnerability mitigation methods. Our classification is illustrated in figure 1. The vulnerability mitigation classes that are shown in figure 1 are described in more details in the following sections. This figure presents a comprehensive view of the previous efforts in mitigating software vulnerabilities. Our classification also aggregates the classifications presented in the previous surveys, such as the ones presented in [1], [3], [2]. Moreover, this classification helps to identify the fault mitigation methods that can be applied to improve current software vulnerability mitigation methods. This helps to suggest new directions for the future research.

a) *Vulnerability prevention*

Generally, fault prevention means avoiding the fault introduction and occurrence in the application during the development. A fault may be introduced during any of the development phases: requirement analysis, design and implementation. To prevent the occurrence of software vulnerabilities during these phases, software security is emerged. Software security is the process of designing, building and testing software for security [14]. It aims at designing and implementing a secure software and educating developers, architects and users to build security in the

software [14]. There are various secure software development methods presented by now, such as Microsoft Security Development Lifecycle (SDL) [15], Security Quality Requirement Engineering (SQUARE) [16] and McGraw's secure development method [14]. Also, there are secure coding best practices that are suggested for different programming languages. These best practices educate the programmers to prevent introduction of well-known vulnerabilities during the coding phase, such as [17] for .NET framework, [18] for C/C++ and [19] for Java.

The programmers' lack of security knowledge is an important reason for the introduction of vulnerabilities. Transferring the related information to the developers is an issue in vulnerability prevention. The SHIELDS project was an example of the attempts in this area [20]. The goal in this project was to create a database of security related information for programmers that can be used automatically. A unified modeling language was proposed in SHIELDS for representing this information [21]. Using this language, it is possible to specify a vulnerability class and its relations to the well-known attacks. It also helps to

define the methods of preventing a vulnerability class. Thus, it helps the developers to learn how to prevent vulnerabilities in order to achieve the security goals of the application. Some tools were also developed based on this language in that project, such as GOAT [20] and TestInv-Code [22].

b) *Vulnerability Tolerance*

In spite of vulnerability prevention efforts, vulnerabilities are created. Thus, vulnerability tolerance is required. Generally, fault tolerance methods accept the existence of faults and focus on preventing the activated faults from reaching the system boundaries and causing a failure. Fault tolerance is usually performed in two steps: error detection and recovery [12]. Therefore, we study monitoring methods based on three aspects: the applied error detection, error handling and fault handling techniques. Please note that since we look a vulnerability as a fault, we consider error as an active vulnerability. Thus, the mentioned three aspects are also named as active vulnerability detection, active vulnerability handling and vulnerability handling techniques respectively.

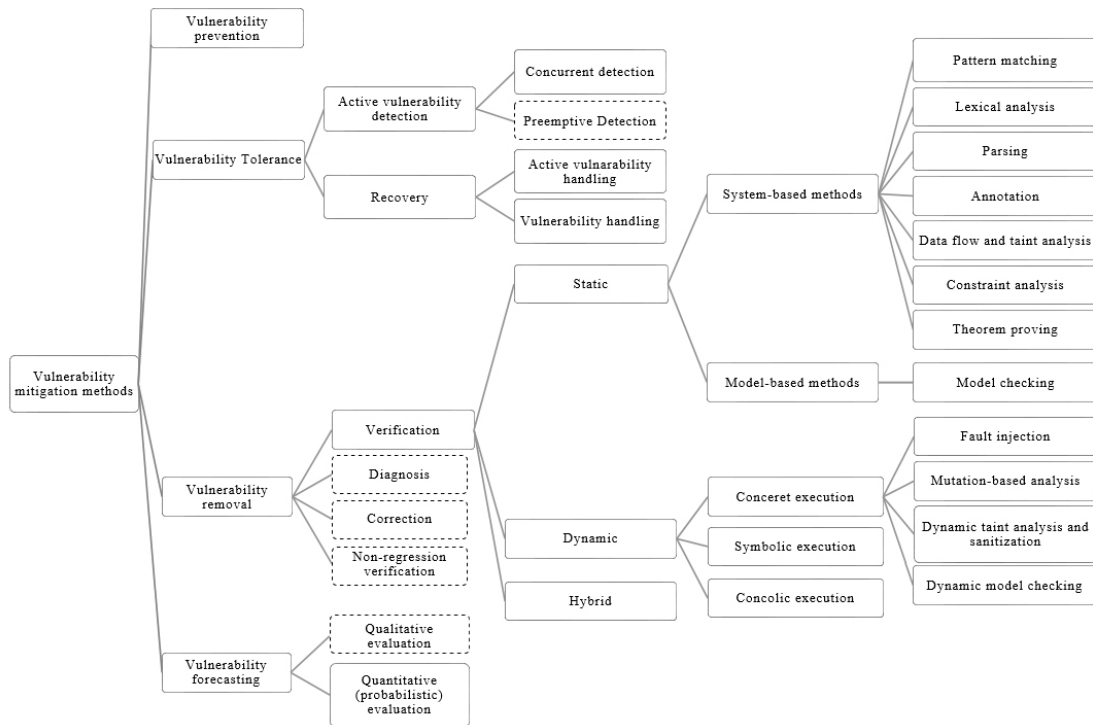


Fig. 1: Our classification of vulnerability mitigation methods according to the classification of fault mitigation methods in [12]. The boxes with dashed borders show the methods that have not been used in mitigating software vulnerabilities yet.

Error detection (active vulnerability detection)

There are vulnerability mitigation methods that control the execution of a program and detects active vulnerabilities at run-time. These methods are also called monitoring methods [2]. Various active vulnerability detection techniques have been used in the proposed monitoring methods. Some examples are monitoring the memory and validating its integrity [23], [24], [25], controlling the flow of user provided data (taint analysis) [26], [27], [28], [29] and validating the arguments of specific functions [30], [31], [32].

For example, the return addresses of functions in the stack memory of the program are monitored in [23], [24] and [25] to detect stack overflows at run-time. If any unauthorized changes of the return addresses is detected, it is concluded that a buffer overflow vulnerability has become active in the program.

Some monitoring methods track the flow of user provided un-trusted data at run-time and react appropriately if the untrusted data reach sensitive statements in the program, such as [26], [27], [28], [29]. This method is used to tolerate various vulnerabilities, such as DOM-based XSS [29], SQL injection [26], [27], [28], buffer overflow [26], [27], [28] and format string [26], [27], [28]. Some monitoring methods locate specific functions in the program and control their arguments during the program execution, such as [30], [31] and [32].

For example, in [31] the program code is analyzed statically and the query strings, that are used as the arguments of SQL functions, are parsed to extract the ASTs of legitimate queries. In this method, the code is instrumented to control the values of SQL queries before executing the relevant functions. Before executing a query with un-trusted data, the monitoring procedure extracts the AST of the query. It then compares the extracted AST with the AST of the legitimate queries. Any inconsistency between the two ASTs might reveal a malicious query. Thus, an appropriate reaction is taken by the monitoring procedure to prevent security failures.

Detecting the errors may be performed during the normal service delivery (concurrent detection). Also, it may be performed in specific times in which the application does not deliver services (preemptive detection). The latter is usually applied to eliminate the negative effects of software aging. All the studied monitoring methods detect active vulnerabilities during the normal service delivery. However, preemptive error detection can be used to detect the activation of vulnerabilities that makes the program overuse the system resources, like the memory leakage vulnerability.

Error handling (active vulnerability handling)

After an error is detected, it is handled in one of three ways: rollback, roll-forward and compensation.

Using the rollback method, the system is restored to a previously stored error-free state. Then, the program continues normal execution from the restored state. In some applications, such as real-time applications, there is no time to rollback. Thus, roll forwarding is performed to change the system state into a degraded new state that contains no errors. Then, the program executes normally from the degraded state. Roll-forwarding is applicable for predictable errors. Another error handling method is compensation. In this method, the redundancy in the current state is used to mask the error and let the program continues the execution. Many of the monitoring methods halt the program and generate an error message when they detect an active vulnerability, e.g. [32], [27], [33], [24]. In other words, many of the monitoring methods do not perform error handling. Some monitoring methods call an exception handler and take the program to a pre-defined state [34], [26], [28]. Most of the monitoring methods that are used for web applications ignore the requests that result in errors and continue normal execution [30], [31], [35], [29]. Calling exception handlers and ignoring the malicious requests can be considered as simple roll-forwarding actions, since the erroneous state is changed into an error-free state and the program continues normal execution. However, more intelligent reactions can be performed after detecting active vulnerabilities. For example, in [36] the stack content and return addresses are stored to compensate for buffer overflow errors. When a buffer overflow error is detected, the monitoring procedure uses the stored data to help the program continue execution securely.

Many of the presented monitoring methods focus on detecting active vulnerabilities, but less attention is paid to handling the active vulnerabilities. It seems that more effort is required on designing appropriate handling methods for active vulnerabilities. Although halting the program and throwing an exception prevents a successful attack, they violate the availability of the software to the legitimate users. Thus, it may result in deniable of service. Therefore, more intelligent active vulnerability handling techniques should be designed for the monitoring methods. Since the rollback technique is usually used for the transient faults and software vulnerability is a permanent fault, this technique cannot be applied in the monitoring methods. Thus, the roll-forward and compensation techniques can be used to design more intelligent active vulnerability handling methods.

Fault handling (vulnerability handling)

After handling the error, sometimes fault handling is performed to remove the fault and prevent the similar errors in the future. Of course, sometimes the fault is handled immediately after error detection. Fault handling is performed by first recognizing causes of the

error. Usually, the faulty component is isolated to prevent the future activation of the fault. A spare fault-free component is then replaced by the faulty one. The system is reconfigured based on the new structure. We are not aware of any monitoring method that consists of a vulnerability handling procedure. However, there are some specific methods for automatically patching the software vulnerabilities, such as [37], [38], [39], [40] and [41]. These methods might be usable in the proposed vulnerability tolerance methods to handle the vulnerabilities. The automatic patching methods analyze the malicious data that is used in an attack and modify the program to filter similar data in the future. These methods can be combined with preemptive active vulnerability detection techniques to generate a complete vulnerability tolerance solution.

Table I summarizes the presented vulnerability tolerance methods so that the reader can review them easier. To sum up, there are various monitoring methods with enhanced error detection mechanisms presented by now. These methods pay more attention to detecting the errors. This might be due to the difference between software vulnerability and the other faults. Usually, software vulnerability is activated by malicious external faults. Therefore, detecting an active vulnerability reveals an ongoing attack. The software should resist the attack as soon as possible to prevent further damages. Thus, the quick detection of the active vulnerability is very important. Halting the program is the fastest low-risk response to the attack. However, it makes the program unavailable to the legitimate users as well. Thus, more intelligent error handling and vulnerability handling techniques should be added to the monitoring methods. To do so, a good starting point is inspiring by the current fault handling and error handling techniques and designing software vulnerability handling techniques.

c) *Vulnerability removal*

Vulnerability removal is performed to detect and remove the vulnerabilities that are created in software despite the vulnerability prevention efforts. Based on figure 1, the fault removal process consists of four steps: verification, diagnosis, correction and non-regression verification. During the verification step, it is verified if the system adheres to the specification. If not, the reason (fault) is diagnosed and corrected. After removing the fault, the verification is repeated to check if the removal was effective. The verification at this step is called non-regression verification.

Most of the vulnerability removal methods focus on the verification step and don't suggest any diagnosis or correction methods for the detected vulnerabilities. There are, however, special vulnerability diagnosis methods that diagnose the vulnerabilities that are exploited by malicious users. For example, in [42] exploitation of memory corruption vulnerabilities is

detected and then the exploited vulnerability is automatically diagnosed. The result of diagnosis consists of the instruction that are exploited by an attacker to corrupt critical program data, the stack trace at the time of memory corruption and the history that the corrupted data are propagated after the initial corruption. This information helps the developers to remove the diagnosed vulnerabilities. We could not find any vulnerability diagnosis or correction procedure that is used after the verification step of a vulnerability removal method. We need vulnerability diagnosis and correction procedures that can be used after the verification step, not after detecting an attack. In other words, these procedures should not be based on the attack information, but based on the information achieved during the verification step.

Some vulnerability detection methods perform the verification step by checking if the software adheres to the security specification, while some of them verify if specific vulnerabilities exist in the software. Figure 1 illustrated our classification of vulnerability verification methods. We divide the verification methods into three main classes: static, dynamic and hybrid methods.

i. *Static analysis*

Static analysis methods do not execute the program. Instead, they examine the program code and study its possible behaviors. Therefore, the result of static analysis is true regardless of the input data and static methods are usually sound and conservative [43]. A sound method is able to detect any specified vulnerability in the program. In other words, if a vulnerability is defined for the static analyzer and exists in a program, the analyzer will surely find it. In order to be sound, the analyzer produces conservative results that are weaker than the actual ones and may not be very useful [43]. In fact, static analysis is appropriate in proving the absence of a specific vulnerability. Usually static analyzers create many false alarms, hence they cannot be very useful in proving the existence of a specific vulnerability. Static analysis may be performed on the program or on the behavior model of the program [12]. Thus, static analysis methods are divided into two main classes: program-based and model-based methods.

Table 1: Examples of vulnerability tolerance methods.

Reviewed Paper	Active vulnerability detection	Active vulnerability handling	Vulnerability handling	Vulnerability
[23]	Detect unauthorized changes of return addresses. (preemptive)	Halts or restarts the program. (rollback)	None	Buffer overflow
[24]	Detect unauthorized changes of return addresses. (preemptive)	Halts the program.	None	Buffer overflow
[25]	Detect unauthorized changes of return addresses. (preemptive)	Halts the program.	None	Buffer overflow
[26]	Monitor the flow of un-trusted data. (preemptive)	Invokes exception handlers. (roll-forward)	None	Any vulnerability that is exploitable by malicious input data.
[27]	Monitor the flow of un-trusted data. (preemptive)	Halts the program.	None	Any vulnerability that is exploitable by malicious input data.
[30]	Monitor the argument of SQL-related functions. (preemptive)	Ignores the request. (roll-forward)	None	SQL injection
[31]	Monitor the argument of SQL-related functions. (preemptive)	Ignores the request. (roll-forward)	None	SQL injection
[32]	Monitor the format argument of printing functions. (preemptive)	Halts the program.	None	Format string
[28]	Monitor the flow of un-trusted data. (preemptive)	Invokes exception handlers. (roll-forward)	None	SQL injection Buffer overflow Format string
[36]	Detect unauthorized changes of return addresses. (preemptive)	Recovers the stack. (compensation)	None	Buffer overflow
[29]	Monitor the flow of un-trusted data. (preemptive)	Ignores the request. (roll-forward)	None	DOM-based XSS

ii. Program-based methods

As figure 1 shows, these methods are classified into seven subclasses. Each class is explained as follows.

Pattern Matching

The most basic static analysis method is pattern matching. A pattern matcher considers the program as a text file. It may not even distinguish between the code and the comments. The pattern matcher searches for vulnerable functions or patterns in the text of the program code. Thus, this method can be implemented using any pattern matching utility, such as grep. Such a tool needs a database of the vulnerability patterns. As an example, Flawfinder [44] scans C/C++ programs to detect buffer overflow or format string in them. This tool ignores the text inside the comments and strings. However, it does not recognize the type of function parameters and control flow or data flow of the program. This lack of knowledge results in many false decisions. Thus, it makes many false positive and false negative alarms.

Lexical analysis

In this method, source code of the program is tokenized in order to recognize the variables and function arguments. Thus, the results of a lexical analyzer can be more accurate than the results of a pattern matcher. As an example, the tool ITS4 applies lexical analysis to detect buffer overflow, format string

and race condition vulnerabilities in C or C++ programs [45]. ITS4 scans the source code statically and breaks it into series of lexical tokens. These tokens are compared with the token streams that are defined in a vulnerability database. The vulnerability database contains several handlers for well-known vulnerable functions in C/C++.

Parsing

In this method, source code of the program is parsed and represented in Abstract Syntax Trees (AST). The ASTs are then used to analyze the program syntactically and semantically. For example, Lint uses this method to detect vulnerabilities in programs written in C [46]. As another example, in [47] the ASTs of the source code are extracted and compared to the ASTs of different vulnerable codes. The main idea in [47] is that different vulnerabilities in software may be related to the same flawed programming pattern. Thus, the suggested method uses the ASTs of known vulnerable codes and searches for similar patterns in the target program. When a similar pattern is found in the program, it may reveal an unknown vulnerability.

Data flow and taint analysis

In this method, the flow of data among the instructions is analyzed to determine possible values that a variable holds during the run time. Two well-known program representations are used in this method: control flow and data flow graphs. In a control flow graph, each node represents an instruction and a

directed edge between two nodes represents their execution dependency. In other words, a directed edge from node *A* to node *B* means that node *A* is executed after the execution of node *B*. The data flow graph is a modified version of the control flow graph in which new directed edges are added to show the data dependency among the instructions. In this graph, the node *A* has data dependency to the node *B*, if the data that is used in node *A* is already manipulated in node *B*. As an example, the vulnerability detection method presented in [48] extracts the control flow and data flow graphs from the source code. It then compares extracted graphs with some patterns of known vulnerabilities. In this method, known vulnerabilities are specified as simple patterns of vulnerable functions or more complex flow-based rules.

A subclass of data flow analysis is called taint analysis. A taint analyzer only tracks the flow of data that come from un-trusted resources. The un-trusted resources include the network protocols, keyboard, touchpad, webcam, files, etc. Since most of the vulnerabilities are exploited by un-trusted input data, this method pays attention to the flow of un-trusted input data in the program. If such data reach sensitive statements in the program, a vulnerability may be reported by the taint analyzer. The sensitive statements, called sinks, are defined according to the specified vulnerabilities. For example, the functions that execute SQL queries are usually defined as the sinks for SQL injection vulnerability. The propagation of tainted data among the instructions is determined based on some predefined rules. For example, if the data in a tainted variable is assigned to an un-tainted variable, the un-tainted variable will get tainted too.

Taint analysis is used in many of the proposed vulnerability detection solutions, e.g. [49], [50], [51], [52] and [53], to detect various vulnerability classes. Since this method focuses on the flow of tainted data, it does not consider the execution paths in the program that are not affected by malicious data. This feature reduces the time of analysis and number of produced false positives. However, there are vulnerability classes that cannot be specified in such a source-sink structure, e.g. logic vulnerabilities. Although an attacker exploits logic vulnerabilities with malicious data, the sinks cannot be easily specified for this class of vulnerability. For example, the sinks for SQL injection vulnerability are the query execution statements. But a sink for logic vulnerabilities may be any statement that manipulates the input data.

Annotation-based methods

Annotation is a comment that the programmer makes in the code about the desired behavior of a function or an instruction. It may be defined as a set of pre- and post-conditions or as simple pre-execution conditions. An annotation-based analysis algorithm

reads the annotations, analyzes the code statically and verifies if the conditions are met in the program. There are plenty of annotation languages presented so far, such as SPLINT [54], MECA [55], Sparse [56], SAL [57] and a Comment [58].

Since there is a huge number of statements and functions in the programs, manual annotation is usually very time consuming and fault prone [58]. There are annotation languages that provide some facilities to annotate the program more easily, such as MECA [55] and aComment [58]. Among them, aComment is designed to help in detecting concurrency faults in the operating systems and allows the programmers to define the pre- and post-conditions that are related to the interrupts in each function. It also infers the annotation of some functions automatically to reduce the programmers' workload. In this way, the programmers are not supposed to annotate all the functions manually.

Although some of these languages help in reducing the required time and effort for annotating the programs, they usually have a different syntax and semantics than the applied programming languages. Therefore, the programmers and verifiers have to make extra efforts to learn another language in order to use this method. Also, the programmers should be familiar with the security requirements of the programs and the vulnerability classes to annotate the program appropriately. Therefore, the success of this method depends on the programmers' knowledge of software security. Moreover, this method is not helpful in analyzing the COTS¹ software and third party components since their source code is not available.

Constraint analysis

In this method, the program is analyzed statically and some constraints are calculated for specific objects in it. The constraints are defined according to specific vulnerabilities and are solved to verify if the program suffers from those vulnerabilities. Constraint analysis was first proposed by Wagner et al. in [50]. The resulted tool, called BOON, considers the strings in a C program as an abstract data type. There are also predefined functions that manipulate this data type, such as strcpy(), strcat(), etc. BOON summarizes the state of each string by two integer values: the allocated size for the string and its current length. For each string in the buffer, it analyzes the string manipulating statements in the program to verify if the length of the string exceeds its allocated size. If such condition is inferred, the program might contains buffer overflow vulnerabilities.

It is important to note that the constraints are determined by the analyzer in this method, not by the programmer. This makes the constraint analysis method different from the annotation-based analysis method.

¹Commercial off-the-shelf

Moreover, constraint analysis does not increase the programmer's workload since generation of the constraints is performed automatically and does not involve the programmer. Of course, it cannot profit the programmers' knowledge of the code to do a more efficient analysis.

Theorem proving

In this method, the software and its specification are expressed as some formulas of logics or algebraic systems. Also, the security requirements of software are expressed as some theorems. Proving these theorems demonstrates the satisfaction of the security requirements. Otherwise, there is a fault (vulnerability) in the program. As an example, in [59] the source code of

target program is statically analyzed and some first-order formulas are generated that assert the absence of certain faults and vulnerabilities, such as out-of-bounds array access. If the generated asserts are proved, the program does not contain such faults and vulnerabilities.

Although the results of analysis are accurate in the theorem proving methods, they demand expertise and enough experience. In fact, theorem proving is difficult to be achieved automatically and requires high-quality staff to apply this method, which is very time-consuming. So it is generally used to verify correct design rather than the actual code [60].

Table 2: Static analysis methods: a comparison

Analysis Method	Description	Advantages	shortcomings	Examples
Pattern Matching	Considers the program as a text file and searches for vulnerability patterns in the text.	Simple, fast.	Does not have any idea about the types of function parameters and control or data flow of the program and so generates many false alarms.	[44].
Lexical Analysis	Tokenizes the code to recognize variables and function arguments.	Variables and function arguments are recognized. More accurate than the pattern matching method.	Lack of knowledge about the syntax and semantics of the code causes false alarms. Requires the high level source code.	[45].
Parsing	Parses the code and represents it in Abstract Syntax Trees (AST) to be analyzed syntactically and semantically.	Understands the code syntactically and semantically, less false alarms in comparison with the above two methods.	Requires the high level source code.	[47], [46], [30], [31], [66].
Annotation-based methods	Comments that the programmer makes about the desired behavior of the code. The code is then analyzed statically to verify if the conditions are met.	Profits the programmers' knowledge to do a focused analysis.	The programmer must learn an additional language to do the annotation.	[54], [55], [56], [57], [58].
Theorem proving	The security requirements of software are expressed as some theorems. Proving these theorems, demonstrates the satisfaction of the security requirements or existence of vulnerabilities.	Accuracy.	Difficult to be achieved automatically and requires high-quality staff to apply this method	[59].
Data flow analysis (Taint analysis)	Tracks the flow of the data that comes from un-trusted resources and warns if the data reaches sensitive program points.	Reduces the analysis time and number of false positives by not considering the execution paths in the program that are not affected by un-trusted data.	Cannot detect vulnerabilities that are not defined specifically in a source-sink structure.	[51], [52], [53], [67], [50], [68], [69], [70].
Constraint analysis	Analyzes the program, associates constraints with some objects in the code and solves them to verify if the program is vulnerable.	Constraints are generated automatically and do not increase the programmer's workload.	Does not profit the programmer's knowledge of the code (in comparison with the annotation method).	[50], [51], [52], [53].
Model checking	Models the program and then checks the model to verify if it satisfies specified requirements.	Only the modeling and requirement specification is performed manually by the human analyzer, rest of the analysis is done automatically.	Modeling the program and specifying its security requirements- if done manually- is time consuming and fault prone. State-explosion problem when the number of program states is large.	[61], [62], [63], [71], [64].

iii. *Model-based methods (model checking)*

In this method, the program is modeled and then analyzed to verify if it complies with its specifications, e.g. [61], [62], [63] and [64]. If a specific requirement is not satisfied in the software, this method provides some counter examples. Model checking helps the human analyzers by automating a noticeable part of the analysis. Although modeling and specifying the requirements may be done manually, analyzing all

possible states of the program and verifying the requirements are done automatically. This is a great help in analyzing large programs. A well-known example of using this method for detecting vulnerabilities is MOPS [63]. Using MOPS, the program is modeled as a push-down automaton². Also, the requirements are

² A push down automaton is a type of computational model. It is similar to NFAs except that it uses an additional component called a stack. In this model, state transitions are chosen based on three

defined through safety properties. A safety property is represented as a finite state automaton. It defines the ordering constraints on security related operations. After modeling the program and defining its constraints, MOPS searches exhaustively through possible program states to check if a reachable state violates the safety properties.

Abstracting the program in a model is a challenging task in this method. The model should be expressive enough to have a precise analysis. Thus, some model checking methods push down automaton is a type of computational model. It is similar to NFAs except that it uses an additional component called a stack. In this model, state transitions are chosen based on three components; input signal, current state and what is at the top of the stack. Thus, the stack plays the role of an additional memory for it. also help the analyzer to model the target program. For example, MOPS uses the control flow of the program to build its automata. Also, in [64] the GCC compiler is used to automatically model and verify the programs that are written in any language supported by this compiler, i.e. C, C++, Java, etc. This is done by employing an intermediate language of GCC, called GIMPIL, that is common to all the supported languages. The model is extracted from the intermediate representation of the program and is checked against the defined specification by the use of Moped. Moped is a model checking tool for push down systems³⁶⁵].

There are, however, some shortcomings in the model checking method. Although the modeling phase is performed automatically in some model checking methods, the analyzer should manually specify the security requirements in this method. This is again time consuming and may cause errors in the results. Also, the method suffers from state-explosion problem for large programs.

Table II summarizes the reviewed static analysis methods. Note that all these methods inherit the general advantages and shortcomings of static analysis.

IV. DYNAMIC ANALYSIS

By executing the program with actual data, dynamic analysis studies the exact run-time behavior of the program. Dynamic analysis can be as fast as the execution of the program, whereas static analysis generally requires more computation time to obtain accurate results [43]. The main challenge in dynamic analysis methods is executing all the possible execution paths in the program and activating all vulnerabilities in those paths. In fact, acquiring an appropriate test data set, that make the program behave more diversely, is an issue in these methods. The most important shortcoming of dynamic analysis methods is that they

components; input signal, current state and what is at the top of the stack. Thus, the stack plays the role of an additional memory for it.

are unable to guarantee the analysis of all feasible execution paths. Therefore, the dynamic analysis is not sound and is mostly used to prove the existence of specific vulnerabilities in the programs. The dynamic methods are classified into two main classes in [12]: methods that use symbolic input values and methods that use actual (concrete) input values to test the program. Based on the recent advances in dynamic analysis methods, we classify these methods in three classes based on the type of applied input values: concrete execution, symbolic execution and concolic (concrete + symbolic) execution methods. The following subsections describe each class in more details.

a) Concrete execution

In this method, the program is executed with actual data and its behavior is analyzed to detect vulnerabilities. There are four dynamic analysis methods that use actual data to execute the program during the analysis: fault injection, mutation-based analysis, dynamic taint analysis and dynamic model checking.

i. Fault injection

In this method, the external faults are injected to the program to examine its behavior. According to our definition in section II, the external faults abuse the internal faults and cause unauthorized behaviors in the program. In other words, internal faults are activated by the external fault and are propagated to reach the program boundaries. Therefore, inability to handle external faults may reveal a vulnerability in the program.

The external faults may be injected by corrupting input data to verify if the program is able to handle them. Most of the blackbox vulnerability scanners corrupt input data and analyze the reaction of the program, such as [72] and [73]. The black-box scanners have access to the inputs and outputs of the program. They might also have very little knowledge about the program internal structure [74]. They usually create the corrupted data based on known attack patterns to study if the program can resist these attacks or suffers from the relevant vulnerabilities. Another group of dynamic vulnerability detectors that inject corrupted input data to the programs are fuzzers. Takanen et al. introduced fuzzing for detecting vulnerabilities for the first time. They suggested injecting unexpected random input data to the program and studying its behavior [74]. The difference between fuzzers and black-box vulnerability scanners is that fuzzers don't corrupt input data exactly based on a list of attack patterns. In fact, they generate numerous random faulty data hoping that some data make the program crash. The main advantages of this method were simplicity and independence from the analyzed program. Thus, the method could be used easily to detect vulnerabilities in different programs. However, fuzzers were not intelligent enough to corrupt input data effectively and cover most of the execution paths. In order to have better program

coverage, new fuzzers focus on producing well-formed corrupted data [75], satisfying data validation checks in the program like checksums [76], being aware of the state of the program during the fuzzing [77] and producing consistent input data with the path conditions to make the program execute all the branches [78], [79], [80], [81]. All these enhancements made fuzzers play an effective role in detecting vulnerabilities during the recent years [82]. Injecting faults into the program can be done randomly or intelligently. By the word random, we mean that faulty data are generated semi-randomly based on predefined patterns. For example, in order to detect buffer overflow, random input data with different lengths are generated. Here the predefined pattern determines the length of input data and the other properties are set randomly. Takanen et al. consider random fuzzers as the ones that make small random changes into the valid data. For example, a FTP fuzzer may randomly add valid/invalid commands to the test data or chose the arguments of the commands randomly [74]. Random fuzzers sometimes use evolutionary algorithms to guide random choices and extend the program coverage, e.g. [83], [84]. Random corruption of data is simple and independent from the logic and structure of the programs. Moreover, randomness helps to reveal a wide range of behaviors of the programs while the designed testcases by the human analyzer may not. This is because the designed test-cases are prepared by a human analyzer who may not think of all possible behaviors of the program.

Corrupting the data intelligently is performed based on a previous analysis of the program. Although it requires more analysis efforts, it helps in extending the program coverage. For example, imagine a program that compares one of the input values with an integer value and exits if they are not equal. Using the random method, the possibility of passing this constraint is one out of 232. By analyzing the code before injecting faulty data, the analyzer is able to extract the constraint and generate the data in a way that complies with the constraint. This helps the intelligent corruption method have more reliable program coverage [74].

ii. *Mutation-based analysis*

As mentioned before, acquiring appropriate test data is an issue in dynamic analysis. When the program behaves normally during the test process, it means that either there is no vulnerability in the program or the test data don't reveal the vulnerabilities in the program. In the latter case, the data set is not diverse enough to activate the vulnerabilities. Mutation is a method that is concerned with enhancing the data set during the dynamic analysis. In this method, specific vulnerabilities are injected into the program code intentionally. If the current data set does not detect the injected vulnerability, it will not detect similar vulnerabilities in the original version of the program. Thus, the analyzer

augments the data set so that it can detect the vulnerability. A version of a program in which a specific vulnerability is created, is called a mutant. For example, in a mutant the function *strncpy()* is replaced with *strcpy()* to make it buffer overflow vulnerable. A good test data set distinguishes the mutants from the original version of the program and kills them. If no test-case kills the mutants, the data set must be augmented [85].

This method is effective in detecting software vulnerabilities [85], though it requires considerable amount of time and effort. If the changed statements in a mutant are executed by the test data, the mutant would be effective. Otherwise, the result of analysis does not reveal the difference between the mutant and the original version of the program. Therefore, some computations are required to generate appropriate testcases that make the program execute the intended path which contains the vulnerability.

Also, automatic creation of mutants for complex vulnerabilities is a challenge. As an example, the *strncpy()* functions are automatically changed to *strcpy()* for creating mutants to detect buffer overflow in [85]. There are, however, more complicated buffer overflow scenarios like copying an array in a loop that causes overflow. Moreover, creating mutants for logic vulnerabilities requires a deep understanding of the logic of the program. Thus, automatic generation of mutants may not be feasible.

iii. *Dynamic model checking*

This method, which is also called execution-based model checking [86], [87], is a model checking method that executes the program exhaustively and checks if it satisfies the specifications. For example, the tools VeriSoft [88], JavaPathFinder [89], CMC [90], Bogor [91] and DART [92] apply this method in their analysis. Random execution in dynamic model checking is mostly the result of two factors: program inputs and scheduling choices of a scheduler [87]. For each random input and schedule choice, the resulted behavior of the program is analyzed by monitoring the process and its environment, e.g. registers and the stack. Here, each state consists of the entire machine state. When the execution reaches a state, in which the specification is compromised, the related input value and schedule choice are presented as a counter-example.

An advantage of dynamic model checking is that by executing the program, the machine handles the semantics of the instructions. In other words, there is no need to formally represent the semantics of the programming language and the machine instructions [87]. However, there is a time-state-soundness tradeoff in this method. Since the states represent the entire machine state, they contain many details and require more storage space. Thus, storing all the states might be infeasible for large programs. At the same time,

exploring the states without a history of visited ones may cause visiting similar states again and again. When no state is recorded, the model checker spends too much time to make sure it has traversed all possible states. Storing the states reduces the verification time by making sure that no state is revisited. Yet, it requires too much space [87].

iv. *Dynamic taint analysis*

This method is similar to static taint analysis as it tracks the flow of information from un-trusted sources to the sinks. However, it tracks the flow of tainted data during the execution of the program, some examples are [93], [94], [95] and [96]. Schwartz et al. describe this method precisely in [93]. They introduce a language, named SIMPIL, that formally defines the algorithms of dynamic taint analysis. Before the execution, all the variables are considered untainted. While executing the program, variables may get tainted according to a predefined policy. This policy defines how the taint data propagate from a variable to other variables. For example, when tainted data are used in an argument of an arithmetic operation, the policy defines that the result of this operation should be considered tainted. If a tainted value reaches a sink, the analyzer reports a vulnerability.

The basic taint analysis methods limit taint propagation to the direct assignments. This might make the results of the analysis inaccurate [97]. Sarwar et al. present some scenarios in [97] to show how basic taint analysis can be ineffective. An example scenario is that the tainted data are used in a conditional statement (without any direct assignment to other variables) and affect on the control flow of the program. Also, tainted data might be used to define the number of an iterative action or as the index of an un-tainted array. The taint analysis method should pay attention to these indirect effects of the tainted data in calculating the taint propagation. Considering such effects is not always easy. For example, the tainted variable might cause information leakage through a side channel. To detect such vulnerability, the analyzer should taint a large amount of variables that results in many false alarms [97].

Table III summarizes and compares the advantages and disadvantages of the concrete execution methods. Each method inherits the advantages and shortcomings of dynamic analysis.

b) *Symbolic execution*

Using the symbolic execution method, the program is executed with symbolic input values instead of concrete data values [98], [99]. Thus, the values of program variables are represented as symbolic expressions over the symbolic input. During the symbolic execution, the state of the program and the conditions of the current path are calculated

symbolically. The path conditions are updated any time a branch instruction is executed. At the end of an executed path, the path conditions are solved using a constraint solver. There are various constraint solvers presented by now, such as STP [100] and Z3 [101] that solve the constraints on binary vectors and Hampi [102] and S3 [103] that solve the constraints on string variables. If the constraint solver solves the path conditions, it generates some concrete input data that are used to execute the intended path in the program.

There are several challenges with the symbolic execution method. For example path explosion, the overhead of constraint solving for complicated paths, non-determinism of concurrent programs and the trade-off between precision and scalability of modeling the memory are some of the challenges in applying symbolic execution [104]. Cadar and Sen present the challenges of this method and mention some solutions for them [104].

To overcome these challenges, a solution is combining symbolic execution with concrete execution. The result is a new method that is called concolic execution. This method is described in the next section.

c) *Concolic execution*

A problem with pure symbolic execution is that the constraints of complex loops and recursive functions may get very complicated and cannot be resolved in an acceptable time [105]. Concrete execution applies real data to execute the program. There is a little chance to traverse all the feasible paths in this method. Using the combined method, concolic + symbolic execution, the concrete data is used to simplify the complex constraints that are generated by the symbolic execution. This method was first presented by Godefroid et al. in [92]. Concolic execution is performed by changing some symbols in the complex constraints into the concrete values. This helps to achieve better program coverage with much less computation overhead.

Concolic execution is used in many of the recent fuzzers to extend their knowledge about the program, such as KLEE [78], EXE [80], Simfuzz [75], CUTE [106], SAGE [79], Taintscope [105] and [107]. For example, CUTE combines symbolic execution with concrete execution to create input data traverse deeper paths in the program. It first executes the program with concrete input data. During the execution, it calculates symbolically the constraints of the executed path. The calculated constraints are then negated one by one, from the last to the first. After each negation, the resulted constraints are queried from a constraint solver. If the constraint solver solves the new constraints, the result is used to generate new test data that traverse other execution paths in the program.

Table 3: Concrete execution methods: a comparison

Analysis Method	Description	Advantages	shortcomings	Examples
Fault injection	Faults are generated semi-randomly. (random corruption) Fault injection is based on some previous analysis of the program. (intelligent corruption)	Simplicity and independence in random corruption of inputs. (random corruption) More reliable code coverage, less false negatives. (intelligent corruption)	Cannot detect logic vulnerability. Less reliable code coverage. (random corruption) More effort is required for testing each single program. (intelligent corruption)	[83],[84], [75], [76], [78],[79], [80].
Mutation-based Analysis	Injects vulnerability into the program code. If the current data set does not reflect the injected vulnerability, it would not detect similar vulnerabilities in the original version of the program.	Reduces false negatives by enriching test data.	Expensive in time and computation. Automatic mutation of complicated vulnerabilities is a challenge.	[85].
Dynamic taint analysis and sanitization	Tracks the flow of information from input sources to the sinks during the run-time.	Reduces the analysis time and number of false positives by not considering the paths in the program that are not affected by malicious data.	Cannot detect vulnerabilities that are not defined in specific source-sink structure.	[93],[96].
Dynamic Model checking	A model checking method in which the program is executed with concrete input values exhaustively.	No need to formally represent the semantics of the programming languages and machine instructions.	Time-state-soundness trade-off.	[88],[89], [90],[91], [92].

Concolic execution is also used in other dynamic vulnerability detection methods. For example in [108] a dynamic model checking method is applied that uses concolic execution for state-space exploration of the analyzed application. In [108], concolic execution helps to model the application as a finitestate automata and to guide further state-space exploration.

d) Hybrid analysis

The previous sections described static and dynamic analysis methods and their advantages and shortcomings. The idea of combining static and dynamic analysis was first proposed by Ernst in [43]. He suggested that hybrid analysis can combine the static and dynamic analysis methods to generate a new analysis method that profits a great amount of soundness and accuracy advantages of each method with little sacrifices.

From then, many researchers have combined these methods, in different manners, to make up for each other's shortcomings. For example, Monga et al. combine static and dynamic analysis to detect XSS and SQL injection vulnerabilities in PHP applications in [109]. The suggested method first analyzes the code statically and extracts the control flow graph of the functions in it. These graphs are then connected together to obtain an inter-procedural control flow graph (iCFG). The iCFG is analyzed to extract the possible paths from the tainted sources to the sinks in it. For each sink, backward slicing is used to detect the statements that affect the tainted argument. These statements are monitored at run time. When a tainted value is used in a sink, the monitoring procedure passes it to an oracle to verify if it can exploit a vulnerability. The oracle have a database of well-known attack patterns that are used to exploit different vulnerabilities. For example, the implemented

oracle for mysql query() performs a limited syntactically analysis on the SQL queries and searches for the tainted characters in unsafe positions. In this method, the sanitizing procedures are assumed to be perfect.

Monitoring and static analysis methods are also combined in [110] to detect SQL injection errors and prevent the successful attacks. In the static analysis phase the hotspots, that are statements in the program that execute a SQL query, are identified. Also the control flow of the program is extracted. Then, the query strings in the hotspots are parsed. Considering the control flow of the program, a FSA for each hotspot is created to model the legitimate queries. During the monitoring phase, the queries are checked against the relative FSA to prevent execution of malicious queries.

As the last example, hybrid analysis is used in [111] to detect logic vulnerabilities in web applications. The logic vulnerabilities are usually related to the intended functionality of an application. Thus, there is no general specification for them that can be used in different applications. For example, consider an online store that allows the users to use coupons for having discount on specific items. It has a policy which determines that each coupon should be used only once. A logic vulnerability, however, allows the users to reuse a coupon and reduce the cost to zero. Since logic vulnerabilities are created based on the functionality of the application, the vulnerability detection method requires the specification of the program. The proposed method in [111] consists of two steps. First, the web application is executed with normal input data. The executed traces are then analyzed to infer the specification of the application. This is based on the intuition that normal behavior of the program reflects the properties that are intended by the programmer. In fact, because the specification of the program is not always

available, this method uses dynamic analysis to obtain it. The inferred specification is presented in the form of likely invariants. In the second step, model checking is used to analyze the web application based on the inferred specification.

e) *Vulnerability forecasting*

Generally, fault forecasting is used to predict the quality or quantity of the faults that are left in the system and will be activated in the future. It is mainly concerned with estimating the current reliability of the system and predicting its future reliability. This prediction may be qualitative or quantitative (usually probabilistic). The qualitative forecasting identifies and ranks the future failure modes. Also, the event combinations that lead to the failures are identified.

The quantitative forecasting is performed by modeling or operational testing. These methods are complementary, since the results of operational tests are usually used to model the system more accurately. Software Reliability Growth Models (SRGM) are used generally for fault forecasting. In fact, SRGMs model the testing process [112]. In most of these models, the rate of fault detection gradually reduces and the cumulative number of faults eventually approaches a fixed value. These models help to predict the number of left faults in the software and determine when the software is ready to be released. They are also used to estimate the required efforts for future maintenance.

There are probabilistic models for predicting the rate of vulnerability detection, named Vulnerability Detection Models (VDM). Alhazmi and Malaiya proposed a specific model, named AM⁴ for vulnerability detection in [113]. In this model the rate of vulnerability detection depends on two factors: one of these factors reduces as the number of remaining undetected vulnerabilities declines. The second factor increases with the time. In this way, the rate of vulnerability detection is modeled in a S-shaped form. In fact, AML is created based on the observation that the detectors⁵ pay little attention to the newly published software. Gradually people become familiar with the software and the detectors pay more attention to it. Thus, the rate of vulnerability detection increases by time and peaks at some period. By the introduction of newer versions of the program, the detectors' interest becomes lower and the rate of vulnerability detection decreases. Alhazmi and Malaiya examined the applicability of this model to various operating systems in [113] and [114]. The results demonstrated that AML fits the data of several operating systems.

All the mentioned models are time-based. It means that they determine the detection rate based on the calendar time. An effort-based model, named AME²

⁴Alhazmi-Malaiya Logistic model

⁵People that analyze applications in order to detect vulnerabilities in the mand report them to the vendors of applications or use them maliciously

²Alhazmi-Malaiya Effort-based model

is proposed by Alhazmi and Malaiya in [113]. They believe that time-based models do not consider the changes that occur in the environment during the lifetime of the system. Thus, they consider the number of installations as an important environmental factor that affects the rate of vulnerability detection. It is based on the observation that the detectors are more interested in the software that is installed in many computers. Therefore, the rate of vulnerability detection is modeled in AME based on the number of installations. Sungwhan Woo et al. explore the applicability of AML and AME to some HTTP servers, i.e., IIS and Apache. The results indicate that these models are applicable to the HTTP servers in addition to the operating systems [112].

A problem with the studied VDMs is that they are parametric models that should be fitted to real vulnerability data [115]. To model the vulnerability detection rate in a specific application, a large amount of historical vulnerability data is required. Therefore, it is necessary that many of the vulnerabilities be discovered already. Hence, these models cannot be applied to predict the detection rate for newly released software. Also, it is shown in [116] that the precision of VDMs depend on the number of known vulnerabilities. The precision of the VDMs are usually very low at the early stages in the lifecycle of the program. It seems that the problem is because the models don't consider the features of each application in their predictions. Thus, they need a history of detected vulnerabilities to estimate the security level of the program. There are many features in each application and its environment that affect the rate of vulnerability detection. Rahimi and Zargham present a VDM in [115] based on two effective factors: code complexity and code quality. The code complexity is defined based on the cyclomatic complexity. Also, the code quality determines its compliance with secure coding practices. They believe that more vulnerabilities are detected in the applications with lower code quality. Also, the possibility of detecting vulnerabilities is less in the applications with complicated codes. Thus, the source code of the application is statically analyzed to compute the two factors. The computed data are then used to model the vulnerability detection rate. Since this model does not need a database of detected vulnerabilities, it can be used for newly released applications. The authors analyze four applications to study the impact of these factors on the vulnerability detection trend. The analysis results show that the proposed method can predict vulnerabilities even in early stages of the application's lifecycle.

Of course, this method does not consider many of the effective factors on the detection trend. For example, it only calculates the cyclomatic complexity to estimate the code complexity. There are other complexity metrics that can be considered in addition to

this one. The environmental parameters can also be considered for a good prediction. As an example, even the seasonal changes affect the rate of vulnerability discovery. It is shown in [117] that more vulnerabilities are reported during the mid-end and year-end months. Also, the presented method in [115] is based on analyzing the source code of the application. So it cannot be helpful when the source code is not available. There are some qualitative methods for estimating the current security level of the application. For example OWASP ASVS consists of several check lists that helps to determine the security level of a web application [118]. It classifies the check lists in thirteen classes, such as authentication, access control, session management, etc. In each class the check lists are grouped into three security levels. If an application passes all the check lists of a group, it achieves the respective security level. These methods only estimate the current security level. Based on the current level, it is possible to predict the future failure modes. However, we could not find any qualitative method that predicts and ranks the future security failures based on the current state.

V. CONCLUSIONS

During the past decades various methods have been presented for mitigating software vulnerabilities. A comprehensive classification of the proposed methods helps to achieve a general understanding of this research area. In this paper, we defined software vulnerability as an internal fault. By considering software vulnerability as a type of fault, we classified the vulnerability mitigation methods based on the general classification of the fault mitigation methods. We extended the general classification of fault mitigation methods, represented it in the context of software vulnerability and added more detailed subclasses into it. We divided vulnerability mitigation methods into four main classes: vulnerability prevention, vulnerability tolerance, vulnerability removal and vulnerability forecasting. The vulnerability prevention methods attempt to prevent the occurrence of software vulnerability. Software security and the secure coding best practices are examples of these efforts. The question is why, despite the vulnerability prevention efforts, vulnerabilities are still created. Oliveira et al. believe that educating the developers is not enough for preventing the vulnerabilities [119], because security is not an issue for the developers. They believe that the human's memory is limited and can only keep a limited number of mental elements available at a time. The programmers are also supposed to create applications with correct functionality and acceptable performance. Under the time pressure, an ordinary situation in software programming, the programmers usually chose

the simplest solutions for developing the software and pay little attention to the security concerns. Oliveira et al. suggest developing assistant tools that remind the educated programmers the security concerns during the development.

Besides educating the programmers, intelligent assistant tools are required to notify the security concerns at specific statements or functions. Thus, in the future we should work on designing and implementing intelligent assistant tools that help the programmers to avoid generating vulnerabilities during the design and implementation of the applications. These tools should be intelligent enough not to bother the developers with many false alarms. They may use the enhanced static analysis methods to analyze the code during the coding phase and warn the programmers at sensitive situations. This will help the programmers to use their security knowledge more effectively in preventing the vulnerabilities.

Vulnerability tolerance methods accept the existence of vulnerabilities in the programs and prevent the active vulnerabilities from making security failures. In this paper, the vulnerability tolerance methods were studied based on three aspects: the applied active vulnerability detection technique, active vulnerability handling technique and vulnerability handling technique. All the reviewed vulnerability mitigation methods detect active vulnerabilities during the normal execution of the program (concurrently). However, there are active vulnerabilities that overuse system resources and make the resources unavailable to legitimate users after a period of time, such as the memory leakage vulnerability. The security failure as a result of these vulnerabilities can be prevented by checking the system resources periodically. Thus, preemptive error detection can be applied to detect if such vulnerabilities are active.

Most of the vulnerability tolerance methods focus on detecting the active vulnerabilities. However, less attention is paid to handling the (active) vulnerability. In the proposed methods, active vulnerabilities are handled by halting the program, restarting the program or invoking an exception handler. Although these mechanisms limit the negative effects of the active vulnerability, they violate the availability of the application to the legitimate users. Thus, more intelligent vulnerability handling techniques are required for the current vulnerability tolerance methods. A good starting point is inspiring by the current fault tolerance methods. As an example, software diversity is a fault tolerance method that is used to make the programs reliable. In this method, various versions of software with the same specification but different design or implementation-process the same request and the correct result is achieved by voting the results of the different versions. The result of such system is more reliable, since it is less

possible that all the versions of software suffer from the same fault and so a request does not cause errors in all versions. This method can be used in software security to tolerate malicious requests. For example, recently software diversity has been used in [120] to tolerate active vulnerabilities in web browsers. In the proposed method, different browsers are used to process the user's requests. Since the browsers are designed and implemented differently, it is less probable that all the applied browsers contain similar vulnerabilities. Thus, malicious data cannot compromise all the browsers. The correct response to the client's request is achieved by voting the responses of the browsers. Also, some protection mechanisms, such as ASLR that protects system against memory corruption vulnerabilities [121], are inspired by the idea of using diversity to make the program unpredictable for the attackers. A new direction for the future research could be adopting the current fault tolerance methods to handle different software vulnerabilities. As there is no vulnerability handling mechanism in the current proposed methods, we should work on designing complete vulnerability tolerance methods that contain appropriate active vulnerability handling and vulnerability handling mechanisms.

Vulnerability removal is performed to detect and remove the vulnerabilities in the software. The focus of most of the current vulnerability removal methods is on verifying the vulnerabilities. In fact, less attention is paid on designing appropriate methods for diagnosis, correction and regression verification of software vulnerabilities. There are some vulnerability diagnosis and correction methods that are used after detecting the exploitation of a vulnerability. But specific methods are required for diagnosis and correction of the vulnerabilities that are detected during the verification of the program. Currently, automatic patching methods analyze an attack and generate software patches based on the pattern of malicious data that are used in the attack. These methods can be modified to automatically generate patches based on the results of analyzing the program and according to the mechanism of detected vulnerabilities.

There are numerous vulnerability detection methods presented by now. Most of the recent vulnerability detection methods tend to combine the previous methods in order to profit their advantages at the same time. For example, the concolic execution method combines the concrete and symbolic execution methods to reduce the complexity of pure symbolic execution and increase the program coverage. As another example, static taint analysis is used with the constraint analysis method to limit the overhead of program analysis and compute the constraints only on the tainted data [51]. Also, the control and data flow of the program are extracted in [122], [62] and [63] to model the program automatically and detect

vulnerabilities by performing the model checking. There are more possible combinations that are not applied yet and might be effective in detecting the vulnerabilities more accurately. For example, the annotation can be used in model checking to profit the programmers' knowledge for modeling the program.

Also, most of the vulnerability removal and vulnerability tolerance methods consider a specific vulnerability class based on their own definition of the relevant software vulnerability. Therefore, an imprecise definition of the intended vulnerability would cause inaccurate results in the proposed method. In addition, some methods only consider a limited number of vulnerability classes. To handle new vulnerability classes, the algorithm of these methods has to be changed. Many of the presented methods or tools are not able to detect all of the vulnerability classes [123], [124]. By now, the researchers' focus was mainly on designing more accurate methods.

A new research trend is making the vulnerability detection methods extendable. In this way, an accurate method for detecting a specific vulnerability can also be used to detect other vulnerabilities. To make a vulnerability detection method extendable, we suggest designing vulnerability detection algorithms that are independent from the sought vulnerability classes. Such algorithms are able to detect any specified vulnerabilities in the program. Designing such methods requires a general model for specifying the vulnerabilities that encompasses any vulnerability classes, even the future ones. Based on this model, various vulnerabilities are specified for the detection algorithms to be detected automatically in the programs.

There are a few extendable vulnerability removal methods, such as [125], [126] and [124]. However, these methods are limited to specific programming languages. Also, some of them are not expressive enough to specify any vulnerability classes. For example, in [127] an extendable vulnerability method is presented for detecting the vulnerabilities in web applications that are written in Java. The specification method of [127] does not support some data types, e.g. integer, float and character. Therefore, it is not possible to define certain operations, such as mathematical operations or comparing the characters, in specifying a vulnerability. This is not a limitation for specifying vulnerabilities in object-oriented programs, such as Java. Because they encapsulate these operations in certain methods for each data type. It is, however, a limitation for specifying vulnerabilities in other languages, such as C. For example, it is not possible to specify integer overflow vulnerabilities in C programs with this method.

The vulnerability forecasting methods predict the number of left vulnerabilities in the software and determine when the software is ready to be released.

They are also used to estimate the required efforts for the future maintenance. Some vulnerability forecasting methods use the vulnerability detection models to predict the rate of vulnerability detection during the lifecycle of the software. The first vulnerability detection models were in fact software reliability growth models that were applied for predicting the vulnerability detection rate. The next models were designed especially for the vulnerability detection rate. These models consider effective parameters on the detection of vulnerabilities, such as time and the number of installations. Since these models do not consider characteristics of the software in their predictions, they need a history of the detected vulnerabilities to predict the vulnerability detection rate in the future. These models are not accurate especially at the early stages in the lifecycle of programs. New vulnerability detection models consider the characteristics of the software to achieve more accurate predictions. In this paper, we reviewed a vulnerability detection model that is based on two characteristics of the program: cyclomatic complexity of the source code and the level of compliance with secure coding practices. There are, however, other characteristics that affect on the rate of vulnerability detection, such as software support, version of the program, availability of the source code or usage of third-party components. For example, the rate of vulnerability detection decreases in time for a program with effective support that periodically presents patches and resolves problems in the program. Also, it might be more difficult to detect vulnerabilities in the higher versions of a program than in its lower versions. The future models can use other characteristics of software to model the vulnerability prediction rate more accurately. Also, they can combine software characteristics with the effective environmental factors, such as time and number of installations, to generate more accurate models.

The possibility of analyzing the program and the program analysis method become important when the vulnerability detection models consider the characteristics of software. For example, a model may be based on some characteristics in the source code of the program. Thus, it is not possible to use such model when the source code of the program is not available. Also, vulnerability forecasts based on inaccurate software analysis are not reliable. In the future, static and dynamic analysis methods that are proposed for detecting the vulnerabilities can be used to better analyze the current characteristics of a program and predict the future rate of vulnerability detection accurately.

REFERENCES RÉFÉRENCES REFERENCIAS

1. W. Jimenez, A. Mammari, A. Cavalli, R. Fourier, Software vulnerabilities, prevention and detection methods: a review, in: Proceeding of the first International Workshop on Security in Model Driven Architecture (SEC-MDA), 2009.
2. H. Shahriar, M. Zulkernine, Mitigating program security vulnerabilities: Approaches and challenges, *ACM Computing Surveys (CSUR)* 44 (3) (2012) 11.
3. K. Zafar, A. Ali, Static techniques for vulnerability detection, Linköping University, Sweden.
4. M. Pistoia, S. Chandra, S. J. Fink, E. Yahav, A survey of static analysis methods for identifying security vulnerabilities in software systems, *IBM Systems Journal* 46 (2) (2007) 265–288.
5. B. Liu, L. Shi, Z. Cai, M. Li, Software vulnerability discovery techniques: A survey, in: Proceeding of the Fourth International Conference on Multimedia Information Networking and Security (MINES), IEEE, 2012, pp. 152–156.
6. M. Bishop, A taxonomy of unix system and network vulnerabilities, Tech. rep., Technical Report CSE-95-10, Department of Computer Science, University of California at Davis (1995).
7. M. Bishop, D. Bailey, A critical analysis of vulnerability taxonomies, Tech. rep., Technical Report CSE-96-11, Department of Computer Science, University of California at Davis. (1996).
8. M. Bishop, *Computer security: art and science*, Addison-Wesley, 2002.
9. H. R. Shahriari, R. Jalili, M. Bishop, A general framework for categorizing vulnerabilities regarding their impact on security policy, *Computers and Security*.
10. I. V. Krsul, *Software vulnerability analysis*, Ph.D. thesis, Purdue University (1998).
11. R. C. Seacord, A. D. Householder, A structured approach to classifying security vulnerabilities, Tech. rep., Technical report CMU/SEI-2005- TN-003. Carnegie-mellon univ pittsburgh pa software engineering inst. (2005).
12. A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr, Basic concepts and taxonomy of dependable and secure computing, *Dependable and Secure Computing*, *IEEE Transactions on* 1 (1) (2004) 11–33.
13. M. F. Ringenburg, D. Grossman, Preventing format-string attacks via automatic and efficient dynamic checking, in: *Proceedings of the 12th ACM conference on Computer and communications security*, ACM, 2005, pp. 354–363.
14. G. McGraw, *Software security: building security in*, Vol. 1, Addison- Wesley Professional, 2006.
15. M. Howard, How do they do it? a look inside the security development lifecycle at microsoft, *MSDN Magazine* (2005) 107–114.
16. N. R. Mead, T. Stehney, *Security quality requirements engineering (SQUARE) methodology*, Vol. 30, ACM, 2005.

17. Secure Coding Guidelines, <https://msdn.microsoft.com/en-us/library/d55zzx87%28v=vs.90%29.aspx>, [Online; accessed 2016-10-14].
18. R. C. Seacord, *Secure Coding in C and C++*, Pearson Education, 2005.
19. F. Long, D. Mohindra, R. C. Seacord, D. F. Sutherland, D. Svoboda, *The CERT Oracle Secure Coding Standard for Java*, Addison-Wesley Professional, 2011.
20. The Shields Project, (2012), <http://www.shields-project.eu>, [Online; accessed 2016-10-23].
21. D. Byers, N. Shahmehri, Unified modeling of attacks, vulnerabilities and security activities, in: *Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems*, ACM, 2010, pp. 36–42.
22. N. Shahmehri, A. Mammar, E. Montes de Oca, D. Byers, A. Cavalli, S. Ardi, W. Jimenez, An advanced approach for modeling and detecting software vulnerabilities, *Information and Software Technology* 54 (9) (2012) 997–1013.
23. C. Cowan, C. Pu, D. Maier, J. Walpole, P. Bakke, S. Beattie, A. Grier, P. Wagle, Q. Zhang, H. Hinton, Stackguard: Automatic adaptive detection and prevention of buffer-overflow attacks., in: *Usenix Security*, Vol. 98, 1998, pp. 63–78.
24. T.-c. Chiueh, F.-H. Hsu, Rad: A compile-time solution to buffer overflow attacks, in: *Proceeding of the 21st International Conference on Distributed Computing Systems*, 2001., IEEE, 2001, pp. 409–417.
25. B. B. Madan, S. Phoha, K. S. Trivedi, Stackoffence: a technique for defending against buffer overflow attacks, in: *Proceeding of International Conference on Information Technology: Coding and Computing. ITCC 2005.*, Vol. 1, IEEE, 2005, pp. 656–661.
26. M. Dalton, H. Kannan, C. Kozyrakis, Raksha: a flexible information flow architecture for software security, in: *ACM SIGARCH Computer Architecture News*, Vol. 35, ACM, 2007, pp. 482–493.
27. G. E. Suh, J. W. Lee, D. Zhang, S. Devadas, Secure program execution via dynamic information flow tracking, in: *ACM SIGOPS Operating Systems Review*, Vol. 38, ACM, 2004, pp. 85–96.
28. J. Clause, W. Li, A. Orso, Dytan: a generic dynamic taint analysis framework, in: *Proceedings of the 2007 international symposium on Software testing and analysis*, ACM, 2007, pp. 196–206.
29. B. Stock, S. Lekies, T. Mueller, P. Spiegel, M. Johns, Precise clientside protection against dom-based cross-site scripting, in: *Proceedings of the 23rd USENIX security symposium*, 2014, pp. 655–670.
30. S. Manmadhan, T. Manesh, A method of detecting sql injection attack to secure web applications, *International Journal of Distributed and Parallel Systems* 3 (6) (2012) 1.
31. Z. Su, G. Wassermann, The essence of command injection attacks in web applications, in: *ACM SIGPLAN Notices*, Vol. 41, ACM, 2006, pp. 372–382.
32. C. Cowan, M. Barringer, S. Beattie, G. Kroah-Hartman, M. Frantzen, J. Lokier, Formatguard: Automatic protection from printf format string vulnerabilities., in: *USENIX Security Symposium*, Vol. 91, Washington, DC, 2001.
33. B. Salamat, A. Gal, T. Jackson, K. Manivannan, G. Wagner, M. Franz, Multi-variant program execution: Using multi-core systems to defuse buffer-overflow vulnerabilities, in: *proceedings of International Conference on Complex, Intelligent and Software Intensive Systems. CISIS 2008.*, IEEE, 2008, pp. 843–848.
34. G. S. Kc, A. D. Keromytis, V. Prevelakis, Countering code-injection attacks with instruction-set randomization, in: *Proceedings of the 10th ACM conference on Computer and communications security*, ACM, 2003, pp. 272–280.
35. G. Iha, H. Doi, An implementation of the binding mechanism in the web browser for preventing xss attacks: introducing the bind-value headers, in: *Availability, Reliability and Security, 2009. ARES'09. International Conference on*, IEEE, 2009, pp. 966–971.
36. S. Gupta, P. Pratap, H. Saran, S. Arun-Kumar, Dynamic code instrumentation to detect and recover from return address corruption, in: *Proceedings of the 2006 international workshop on Dynamic systems analysis*, ACM, 2006, pp. 65–72.
37. M. Zhang, H. Yin, Appsealer: Automatic generation of vulnerability specific patches for preventing component hijacking attacks in android applications, in: *Proceedings of the 21th Annual Network and Distributed System Security Symposium (NDSS 2014)*, 2014.
38. W. Cui, M. Peinado, H. J. Wang, M. E. Locasto, Shieldgen: Automatic data patch generation for unknown vulnerabilities with informed probing, in: *Security and Privacy, 2007. SP'07. IEEE Symposium on*, IEEE, 2007, pp. 252–266.
39. T. Wang, C. Song, W. Lee, Diagnosis and emergency patch generation for integer overflow exploits, in: *Detection of Intrusions and Malware, and Vulnerability Assessment*, Springer, 2014, pp. 255–275.
40. A. Smirnov, T.-c. Chiueh, Automatic patch generation for buffer overflow attacks, in: *Information Assurance and Security, 2007. IAS 2007. Third International Symposium on*, IEEE, 2007, pp. 165–170.
41. Z. Liang, R. Sekar, D. C. DuVarney, Automatic synthesis of filters to discard buffer overflow attacks: A step towards realizing self-healing systems., in:

- USENIX Annual Technical Conference, General Track, 2005, pp. 375–378.
42. J. Xu, P. Ning, C. Kil, Y. Zhai, C. Bookholt, Automatic diagnosis and response to memory corruption vulnerabilities, in: Proceedings of the 12th ACM conference on Computer and communications security, ACM, 2005, pp. 223–234.
 43. M. D. Ernst, Static and dynamic analysis: Synergy and duality, in: WODA 2003: ICSE Workshop on Dynamic Analysis, 2003, pp. 24–27.
 44. D. A. Wheeler, Flawfinder, <http://www.dwheeler.com/flawfinder>, [Online; accessed 2016-10-23] (2001).
 45. J. Viega, J.-T. Bloch, Y. Kohno, G. McGraw, Its4: A static vulnerability scanner for c and c++ code, in: Proceedings of the 16th Annual Computer Security Applications Conference, ACSAC'00, IEEE, 2000, pp. 257–267.
 46. S. C. Johnson, Lint, a C program checker, Bell Telephone Laboratories, 1977.
 47. F. Yamaguchi, M. Lottmann, K. Rieck, Generalized vulnerability extrapolation using abstract syntax trees, in: Proceedings of the 28th Annual Computer Security Applications Conference, ACM, 2012, pp. 359–368.
 48. H. Kim, T.-H. Choi, S.-C. Jung, H.-C. Kim, O. Lee, K.-G. Doh, Applying dataflow analysis to detecting software vulnerability, in: Proceeding of the 10th International Conference on Advanced Communication Technology. ICACT 2008., Vol. 1, IEEE, 2008, pp. 255–258.
 49. N. Jovanovic, C. Kruegel, E. Kirda, Pixy: A static analysis tool for detecting web application vulnerabilities, in: Proceeding of the 2006 IEEE Symposium on Security and Privacy, IEEE, 2006, pp. 6–pp.
 50. D. Wagner, J. S. Foster, E. A. Brewer, A. Aiken, A first step towards automated detection of buffer overrun vulnerabilities., in: NDSS, 2000, pp. 2000–02.
 51. V Ganapathy, S. Jha, D. Chandler, D. Melski, D. Vitek, Buffer overrun detection using linear programming and static analysis, in: Proceedings of the 10th ACM conference on Computer and communications security, ACM, 2003, pp. 345–354.
 52. Y. Xia, J. Luo, M. Zhang, Detecting memory access errors with flow-sensitive conditional range analysis, in: Embedded Software and Systems, Springer, L 2005, pp. 320–331.
 53. F. Yu, T. Bultan, O. H. Ibarra, Symbolic string verification: Combining string analysis and size analysis, in: Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2009, pp. 322–336.
 54. D. Evans, D. Larochele, Improving security using extensible lightweight static analysis, software, IEEE 19 (1) (2002) 42–51.
 55. J. Yang, T. Kremenek, Y. Xie, D. Engler, Meca: an extensible, expressive system and language for statically checking security properties, in: Proceedings of the 10th ACM conference on Computer and communications security, ACM, 2003, pp. 321–334.
 56. Y. Tsuruoka, J. Tsujii, S. Ananiadou, Accelerating the annotation of sparse named entities by dynamic sentence selection, BMC bioinformatics 9 (Suppl 11) (2008) S8.
 57. M. Howard, A brief introduction to the standard annotation language (SAL), http://blogs.msdn.com/b/michael_howard/archive/2006/05/19/602077.aspx, [Online; accessed 2016-10-22] (2006).
 58. L. Tan, Y. Zhou, Y. Padioleau, acomment: mining annotations from comments and code to detect interrupt related concurrency bugs, in: Proceedings of the 33rd international conference on software engineering, ACM, 2011, pp. 11–20.
 59. D. Detlefs, G. Nelson, J. B. Saxe, Simplify: a theorem prover for program checking, Journal of the ACM (JACM) 52 (3) (2005) 365– 473.
 60. G. Tian-yang, S. Yin-sheng, F. You-yuan, Research on software security testing, World Academy of science, engineering and Technology 70 (2010) 647–651.
 61. S. Chaki, S. Hissam, Precise buffer overflow detection via model checking (2005).
 62. W.-S. Rödiger, Merging static analysis and model checking for improved security vulnerability detection, Ph.D. thesis, Master thesis, Dept. of Com. Sc. Augsburg University (2011).
 63. H. Chen, D. Wagner, Mops: an infrastructure for examining security properties of software, in: Proceedings of the 9th ACM conference on Computer and communications security, ACM, 2002, pp. 235–244.
 64. R. Hadjidj, X. Yang, S. Tlili, M. Debbabi, Model-checking for software vulnerabilities detection with multi-language support, in: Proceeding of the sixth Annual Conference on Privacy, Security and Trust, PST'08., IEEE, 2008, pp. 133–142.
 65. J. Esparza, D. Hansel, P. Rossmanith, S. Schwoon, Efficient algorithms for model checking pushdown systems, in: Computer Aided Verification, Springer, 2000, pp. 232–247.
 66. J. Ren, B. Cai, H. HE, C. HU, A method for detecting software vulnerabilities based on clustering and model analyzing, Journal of Computational Information Systems 7 (4) (2011) 1065–1073.

67. B. Livshits, M. S. Lam, Finding security vulnerabilities in java applications with static analysis, in: Proceedings of the 14th conference on USENIX Security Symposium, Vol. 14, 2005.
68. G. Wassermann, Z. Su, Static detection of cross-site scripting vulnerabilities, in: Proceeding of the ACM/IEEE 30th International Conference on Software Engineering. ICSE'08., IEEE, 2008, pp. 171–180.
69. J. Newsome, D. Song, Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software, in: Proceedings of the 12th Network and Distributed System Security Symposium (NDSS05), 2005.
70. P. Hooimeijer, B. Livshits, D. Molnar, P. Saxena, M. Veanes, Fast and precise sanitizer analysis with bek, in: Proceedings of the 20th USENIX conference on Security, USENIX Association, 2011, pp. 1–1.
71. L. Wang, Q. Zhang, P. Zhao, Automated detection of code vulnerabilities based on program analysis and model checking, in: Proceeding of the Eighth IEEE International Working Conference on Source Code Analysis and Manipulation, IEEE, 2008, pp. 165–173.
72. G. Pellegrino, D. Balzarotti, Toward black-box detection of logic flaws in web applications, in: Proceedings of the Network and Distributed System Security (NDSS) Symposium, 2014.
73. S. Kals, E. Kirda, C. Kruegel, N. Jovanovic, Secubat: a web vulnerability scanner, in: Proceedings of the 15th international conference on World Wide Web, ACM, 2006, pp. 247–256.
74. A. Takanen, J. D. Demott, C. Miller, Fuzzing for software security testing and quality assurance, Artech House, 2008.
75. D. Zhang, D. Liu, Y. Lei, D. Kung, C. Csallner, N. Nystrom, W. Wang, Simfuzz: Test case similarity directed deep fuzzing, *Journal of Systems and Software* 85 (1) (2012) 102–111.
76. T. Wang, T. Wei, G. Gu, W. Zou, Taintscope: A checksum-aware directed fuzzing tool for automatic software vulnerability detection, in: Proceeding of 2010 IEEE Symposium on Security and Privacy (SP), IEEE, 2010, pp. 497–512.
77. A. Doupé, L. Cavedon, C. Kruegel, G. Vigna, Enemy of the state: A state-aware black-box web vulnerability scanner., in: USENIX Security Symposium, 2012, pp. 523–538.
78. A. Cadar, D. Dunbar, D. R. Engler, Klee: Unassisted and automatic generation of high-coverage tests for complex systems programs., in: OSDI, Vol. 8, 2008, pp. 209–224.
79. P. Godefroid, M. Y. Levin, D. A. Molnar, Automated whitebox fuzz testing., in: NDSS, Vol. 8, 2008, pp. 151–166.
80. C. Cadar, V. Ganesh, P. M. Pawlowski, D. L. Dill, D. R. Engler, Exe: automatically generating inputs of death, *ACM Transactions on Information and System Security (TISSEC)* 12 (2) (2008) 10.
81. I. Haller, A. Slowinska, M. Neugschwandtner, H. Bos, Dowsing for overflows: A guided fuzzer to find buffer boundary violations., in: *Userenix Security*, 2013, pp. 49–64.
82. S. Heelan, Vulnerability detection systems: Think cyborg, not robot, *IEEE Security and Privacy* 9 (3) (2011) 74–77.
83. S. Sparks, S. Embleton, R. Cunningham, C. Zou, Automated vulnerability analysis: Leveraging control flow for evolutionary input crafting, in: *Proceeding of Twenty-Third Annual Computer Security Applications Conference. ACSAC 2007.*, IEEE, 2007, pp. 477–486.
84. J. DeMott, R. Enbody, W. F. Punch, Revolutionizing the field of greybox attack surface testing with evolutionary fuzzing, *BlackHat and Defcon*.
85. H. Shahriar, M. Zulkernine, Music: Mutation-based sql injection vulnerability checking, in: *Proceeding of the Eighth International Conference on Quality Software. QSIC'08.*, IEEE, 2008, pp. 77–86.
86. A. Groce, R. Joshi, Extending model checking with dynamic analysis, in: *Verification, Model Checking, and Abstract Interpretation*, Springer, 2008, pp. 142–156.
87. R. Jhala, R. Majumdar, Software model checking, *ACM Computing Surveys (CSUR)* 41 (4) (2009) 21.
88. P. Godefroid, Model checking for programming languages using verisoft, in: *Proceedings of the 24th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, ACM, 1997, pp. 174–186.
89. K. Havelund, T. Pressburger, Model checking java programs using java pathfinder, *International Journal on Software Tools for Technology Transfer* 2 (4) (2000) 366–381.
90. M. Musuvathi, D. Y. Park, A. Chou, D. R. Engler, D. L. Dill, Cmc: A pragmatic approach to model checking real code, *ACM SIGOPS Operating Systems Review* 36 (SI) (2002) 75–88.
91. M. B. Dwyer, J. Hatcliff, Bogor: an extensible and highly-modular software model checking framework, in: *ACM SIGSOFT Software Engineering Notes*, Vol. 28, ACM, 2003, pp. 267–276.
92. P. Godefroid, N. Klarlund, K. Sen, Dart: directed automated random testing, in: *ACM Sigplan Notices*, Vol. 40, ACM, 2005, pp. 213–223.
93. E. J. Schwartz, T. Avgerinos, D. Brumley, All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask), in: *Proceeding of the 2010 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2010, pp. 317–331.

94. S. Lekies, B. Stock, M. Johns, 25 million flows later: large-scale detection of dom-based xss, in: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, ACM, 2013, pp.1193–1204.
95. M. G. Kang, S. McCamant, P. Poosankam, D. Song, Dta++: Dynamic taint analysis with targeted control-flow propagation., in: NDSS, 2011.
96. D. Balzarotti, M. Cova, V. Felmetzger, N. Jovanovic, E. Kirda, C. Kruegel, G. Vigna, Saner: Composing static and dynamic analysis to validate sanitization in web applications, in: Proceeding of IEEE Symposium on Security and Privacy. SP 2008., IEEE, 2008, pp. 387–401.
97. G. Sarwar, O. Mehani, R. Boreli, D. Kaafar, On the effectiveness of dynamic taint analysis for protecting against private information leaks on android-based devices, in: Proceeding of the 10th International Conference on Security and Cryptography (SECRYPT), 2013.
98. L. A. Clarke, A program testing system, in: Proceedings of the 1976 annual conference, ACM, 1976, pp. 488–491.
99. D. Davidson, B. Moench, T. Ristenpart, S. Jha, Fie on firmware: Finding vulnerabilities in embedded systems using symbolic execution., in: USENIX Security, 2013, pp. 463–478.
100. V. Ganesh, D. L. Dill, A decision procedure for bit-vectors and arrays, in: Computer Aided Verification, Springer, 2007, pp. 519–531.
101. L. De Moura, N. Bjørner, Z3: An efficient smt solver, in: Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2008, pp. 337–340.
102. A. Kiezun, V. Ganesh, P. J. Guo, P. Hooimeijer, M. D. Ernst, Hampi: a solver for string constraints, in: Proceedings of the eighteenth international symposium on Software testing and analysis, ACM, 2009, pp. 105–116.
103. M.-T. Trinh, D.-H. Chu, J. Jaffar, S3: A symbolic string solver for vulnerability detection in web applications, in: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2014, pp. 1232–1243.
104. C. Cadar, K. Sen, Symbolic execution for software testing: three decades later, Communications of the ACM 56 (2) (2013) 82–90.
105. Z. Wang, J. Ming, C. Jia, D. Gao, Linear obfuscation to combat symbolic execution, in: Computer Security–ESORICS 2011, Springer, 2011, pp. 210–226.
106. K. Sen, D. Marinov, G. Agha, CUTE: a concolic unit testing engine for C, Vol. 30, ACM, 2005.
107. D. Molnar, X. C. Li, D. A. Wagner, Dynamic test generation to find integer bugs in x86 binary linux programs, in: Proceedings of the 18th conference on USENIX security symposium, USENIX Association, 2009, pp. 67–82.
108. C. Y. Cho, D. Babic, P. Poosankam, K. Z. Chen, E. X. Wu, D. Song, Mace: Model-inference-assisted concolic exploration for protocol and vulnerability discovery., in: USENIX Security Symposium, 2011, pp. 139–154.
109. M. Monga, R. Paleari, E. Passerini, A hybrid analysis framework for detecting web application vulnerabilities, in: Proceedings of the 2009 ICSE Workshop on Software Engineering for Secure Systems, IEEE Computer Society, 2009, pp. 25–32.
110. W. G. Halfond, A. Orso, Combining static analysis and runtime monitoring to counter sql-injection attacks, in: ACM SIGSOFT Software Engineering Notes, Vol. 30, ACM, 2005, pp. 1–7.
111. Felmetzger, L. Cavedon, C. Kruegel, G. Vigna, Toward automated detection of logic vulnerabilities in web applications, in: USENIX Security Symposium, 2010, pp. 143–160.
112. S.-W. Woo, H. Joh, O. H. Alhazmi, Y. K. Malaiya, Modeling vulnerability discovery process in apache and iis http servers, Computers & Security 30 (1) (2011) 50–62.
113. O. H. Alhazmi, Y. K. Malaiya, Quantitative vulnerability assessment of systems software, in: Proceedings of annual reliability and maintainability symposium, 2005, pp. 615–620.
114. O. Alhazmi, Y. Malaiya, I. Ray, Security vulnerabilities in software systems: A quantitative perspective, in: Data and Applications Security XIX, Springer, 2005, pp. 281–294.
115. S. Rahimi, M. Zargham, Vulnerability scrying method for software vulnerability discovery prediction without a vulnerability database, Reliability, IEEE Transactions on 62 (2) (2013) 395–407.
116. O. H. Alhazmi, Y. K. Malaiya, Application of vulnerability discovery models to major operating systems, Reliability, IEEE Transactions on 57 (1) (2008) 14–22.
117. H. Joh, Y. K. Malaiya, Seasonal variation in the vulnerability discovery process, in: Proceedings of ICST'09 International Conference on Software Testing Verification and Validation, 2009., IEEE, 2009, pp. 191–200.
118. Application Security Verification Standard (ASVS), https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf, [Online; accessed 2016-10-8].
119. D. Oliveira, M. Rosenthal, N. Morin, K.-C. Yeh, J. Cappos, Y. Zhuang, It's the psychology stupid: how heuristics explain software vulnerabilities and how

- priming can illuminate developer's blind spots, in: Proceedings of the 30th Annual Computer Security Applications Conference, ACM, 2014, pp. 296–305.
120. H. Xue, N. Dautenhahn, S. T. King, Using replicated execution for a more secure and reliable web browser., in: NDSS, 2012.
 121. L. Szekeres, M. Payer, T. Wei, D. Song, Eternal war in memory, in: IEEE Symposium on Security and Privacy, 2013.
 122. D. Balzarotti, M. Cova, V. V. Felmetzger, G. Vigna, Multi-module vulnerability analysis of web-based applications, in: Proceedings of the 14th ACM conference on Computer and communications security, ACM, 2007, pp. 25–35.
 123. J. Bau, E. Bursztein, D. Gupta, J. Mitchell, State of the art: Automated black-box web application vulnerability testing, in: Proceeding of the 2010 IEEE Symposium on Security and Privacy (SP), IEEE, 2010, pp. 332–345.
 124. M. Almorsy, J. Grundy, A. S. Ibrahim, Supporting automated vulnerability analysis using formalized vulnerability signatures, in: Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, ACM, 2012, pp. 100–109.
 125. F. Yamaguchi, N. Golde, D. Arp, K. Rieck, Modeling and discovering vulnerabilities with code property graphs, in: Proceedings of 2014 IEEE Symposium on Security and Privacy (SP), IEEE, 2014, pp. 590–604.
 126. W. Mallouli, A. Mammar, A. Cavalli, W. Jimenez, Vdc-based dynamic code analysis: Application to c programs, *Journal of Internet Services and Information Security* 1 (2/3) (2011) 4–20.
 127. B. Livshits, Improving software security with precise static and runtime analysis, Ph.D. thesis, Stanford University (2006).
 128. U. Shankar, K. Talwar, J. S. Foster, D. Wagner, Detecting format string vulnerabilities with type qualifiers., in: USENIX Security Symposium, 2001, pp. 201–220.
 129. L. C. Paulson, Proving properties of security protocols by induction, in: Proceedings of the 10th workshop on Computer Security Foundations, IEEE, 1997, pp. 70–83.
 130. M. Burrows, M. Abadi, R. M. Needham, A logic of authentication, in: Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, Vol. 426, The Royal Society, 1989, pp. 233–271.
 131. T. Nipkow, L. C. Paulson, M. Wenzel, Isabelle/HOL: a proof assistant for higher-order logic, Vol. 2283, Springer Science & Business Media, 2002.
 132. D. Larochelle, D. Evans, Statically detecting likely buffer overflow vulnerabilities., in: USENIX Security Symposium, Vol. 32, Washington DC, 2001.
 133. C. E. Landwehr, A. R. Bull, J. P. McDermott, W. S. Choi, A taxonomy of computer program security flaws, *ACM Computing Surveys (CSUR)* 26 (3) (1994) 211–254.
 134. R. P. Abbott, J. S. Chin, J. E. Donnelley, W. L. Konigsford, S. Tokubo, D. A. Webb, Security analysis and enhancements of computer operating systems, Tech. rep., Technical report NBSIR-76-1041. National bureau of standards Washington inst for computer sciences and technology. (1976).
 135. R. Bisbey, D. Hollingworth, Protection analysis: Final report (1978).
 136. P. Anderson, Codesurfer/path inspector, Proceeding of the 20th IEEE International Conference on Software Maintenance, 2004. Proceedings, 2004.
 137. T. A. Henzinger, R. Jhala, R. Majumdar, G. Sutre, Software verification with blast, in: Model Checking Software, Springer, 2003, pp. 235–239.
 138. S. Rawat, D. Ceara, L. Mounier, M.-L. Potet, Combining static and dynamic analysis for vulnerability detection, arXiv preprint arXiv: 1305.3883.
 139. R. Anderson, Security in open versus closed system the dance of boltzmann, coase and moore, *Open Source Software Economics*.
 140. E. Rescorla, Is finding security holes a good idea?, *Security & Privacy, IEEE* 3 (1) (2005) 14–19.
 141. J. D. Musa, K. Okumoto, A logarithmic poisson execution time model for software reliability measurement, in: Proceedings of the 7th international conference on Software engineering, IEEE Press, 1984, pp. 230–238.
 142. J. Dahse, T. Holz, Simulation of built-in php features for precise static code analysis, in: Symposium on Network and Distributed System Security (NDSS), 2014.

GLOBAL JOURNALS INC. (US) GUIDELINES HANDBOOK 2017

WWW.GLOBALJOURNALS.ORG

FELLOWS

FELLOW OF ASSOCIATION OF RESEARCH SOCIETY IN COMPUTING (FARSC)

Global Journals Incorporate (USA) is accredited by Open Association of Research Society (OARS), U.S.A and in turn, awards “FARSC” title to individuals. The 'FARSC' title is accorded to a selected professional after the approval of the Editor-in-Chief/Editorial Board Members/Dean.



- The “FARSC” is a dignified title which is accorded to a person’s name viz. Dr. John E. Hall, Ph.D., FARSC or William Walldroff, M.S., FARSC.

FARSC accrediting is an honor. It authenticates your research activities. After recognition as FARSC, you can add 'FARSC' title with your name as you use this recognition as additional suffix to your status. This will definitely enhance and add more value and repute to your name. You may use it on your professional Counseling Materials such as CV, Resume, and Visiting Card etc.

The following benefits can be availed by you only for next three years from the date of certification:



FARSC designated members are entitled to avail a 40% discount while publishing their research papers (of a single author) with Global Journals Incorporation (USA), if the same is accepted by Editorial Board/Peer Reviewers. If you are a main author or co-author in case of multiple authors, you will be entitled to avail discount of 10%.

Once FARSC title is accorded, the Fellow is authorized to organize a symposium/seminar/conference on behalf of Global Journal Incorporation (USA). The Fellow can also participate in conference/seminar/symposium organized by another institution as representative of Global Journal. In both the cases, it is mandatory for him to discuss with us and obtain our consent.



You may join as member of the Editorial Board of Global Journals Incorporation (USA) after successful completion of three years as Fellow and as Peer Reviewer. In addition, it is also desirable that you should organize seminar/symposium/conference at least once.

We shall provide you intimation regarding launching of e-version of journal of your stream time to time. This may be utilized in your library for the enrichment of knowledge of your students as well as it can also be helpful for the concerned faculty members.

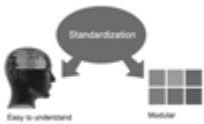




Journals Research
inducing researches

The FARSC can go through standards of OARS. You can also play vital role if you have any suggestions so that proper amendment can take place to improve the same for the benefit of entire research community.

As FARSC, you will be given a renowned, secure and free professional email address with 100 GB of space e.g. johnhall@globaljournals.org. This will include Webmail, Spam Assassin, Email Forwarders, Auto-Responders, Email Delivery Route tracing, etc.



The FARSC will be eligible for a free application of standardization of their researches. Standardization of research will be subject to acceptability within stipulated norms as the next step after publishing in a journal. We shall depute a team of specialized research professionals who will render their services for elevating your researches to next higher level, which is worldwide open standardization.

The FARSC member can apply for grading and certification of standards of their educational and Institutional Degrees to Open Association of Research, Society U.S.A. Once you are designated as FARSC, you may send us a scanned copy of all of your credentials. OARS will verify, grade and certify them. This will be based on your academic records, quality of research papers published by you, and some more criteria. After certification of all your credentials by OARS, they will be published on your Fellow Profile link on website <https://associationofresearch.org> which will be helpful to upgrade the dignity.



The FARSC members can avail the benefits of free research podcasting in Global Research Radio with their research documents. After publishing the work, (including published elsewhere worldwide with proper authorization) you can upload your research paper with your recorded voice or you can utilize chargeable services of our professional RJs to record your paper in their voice on request.

The FARSC member also entitled to get the benefits of free research podcasting of their research documents through video clips. We can also streamline your conference videos and display your slides/ online slides and online research video clips at reasonable charges, on request.





The FARSC is eligible to earn from sales proceeds of his/her researches/reference/review Books or literature, while publishing with Global Journals. The FARSC can decide whether he/she would like to publish his/her research in a closed manner. In this case, whenever readers purchase that individual research paper for reading, maximum 60% of its profit earned as royalty by Global Journals, will be credited to his/her bank account. The entire entitled amount will be credited to his/her bank account exceeding limit of minimum fixed balance. There is no minimum time limit for collection. The FARSC member can decide its price and we can help in making the right decision.

The FARSC member is eligible to join as a paid peer reviewer at Global Journals Incorporation (USA) and can get remuneration of 15% of author fees, taken from the author of a respective paper. After reviewing 5 or more papers you can request to transfer the amount to your bank account.



MEMBER OF ASSOCIATION OF RESEARCH SOCIETY IN COMPUTING (MARSC)

The ' MARSC ' title is accorded to a selected professional after the approval of the Editor-in-Chief / Editorial Board Members/Dean.

The "MARSC" is a dignified ornament which is accorded to a person's name viz. Dr. John E. Hall, Ph.D., MARSC or William Walldroff, M.S., MARSC.



MARSC accrediting is an honor. It authenticates your research activities. After becoming MARSC, you can add 'MARSC' title with your name as you use this recognition as additional suffix to your status. This will definitely enhance and add more value and repute to your name. You may use it on your professional Counseling Materials such as CV, Resume, Visiting Card and Name Plate etc.

The following benefits can be availed by you only for next three years from the date of certification.



MARSC designated members are entitled to avail a 25% discount while publishing their research papers (of a single author) in Global Journals Inc., if the same is accepted by our Editorial Board and Peer Reviewers. If you are a main author or co-author of a group of authors, you will get discount of 10%.

As MARSC, you will be given a renowned, secure and free professional email address with 30 GB of space e.g. johnhall@globaljournals.org. This will include Webmail, Spam Assassin, Email Forwarders, Auto-Responders, Email Delivery Route tracing, etc.





We shall provide you intimation regarding launching of e-version of journal of your stream time to time. This may be utilized in your library for the enrichment of knowledge of your students as well as it can also be helpful for the concerned faculty members.

The MARSC member can apply for approval, grading and certification of standards of their educational and Institutional Degrees to Open Association of Research, Society U.S.A.



Once you are designated as MARSC, you may send us a scanned copy of all of your credentials. OARS will verify, grade and certify them. This will be based on your academic records, quality of research papers published by you, and some more criteria.

It is mandatory to read all terms and conditions carefully.



AUXILIARY MEMBERSHIPS

Institutional Fellow of Open Association of Research Society (USA)-OARS (USA)

Global Journals Incorporation (USA) is accredited by Open Association of Research Society, U.S.A (OARS) and in turn, affiliates research institutions as “Institutional Fellow of Open Association of Research Society” (IFOARS).



The “FARSC” is a dignified title which is accorded to a person’s name viz. Dr. John E. Hall, Ph.D., FARSC or William Walldroff, M.S., FARSC.

The IFOARS institution is entitled to form a Board comprised of one Chairperson and three to five board members preferably from different streams. The Board will be recognized as “Institutional Board of Open Association of Research Society”-(IBOARS).

The Institute will be entitled to following benefits:



The IBOARS can initially review research papers of their institute and recommend them to publish with respective journal of Global Journals. It can also review the papers of other institutions after obtaining our consent. The second review will be done by peer reviewer of Global Journals Incorporation (USA) The Board is at liberty to appoint a peer reviewer with the approval of chairperson after consulting us.

The author fees of such paper may be waived off up to 40%.

The Global Journals Incorporation (USA) at its discretion can also refer double blind peer reviewed paper at their end to the board for the verification and to get recommendation for final stage of acceptance of publication.



The IBOARS can organize symposium/seminar/conference in their country on behalf of Global Journals Incorporation (USA)-OARS (USA). The terms and conditions can be discussed separately.

The Board can also play vital role by exploring and giving valuable suggestions regarding the Standards of “Open Association of Research Society, U.S.A (OARS)” so that proper amendment can take place for the benefit of entire research community. We shall provide details of particular standard only on receipt of request from the Board.



Journals Research
inducing researches

The board members can also join us as Individual Fellow with 40% discount on total fees applicable to Individual Fellow. They will be entitled to avail all the benefits as declared. Please visit Individual Fellow-sub menu of GlobalJournals.org to have more relevant details.



We shall provide you intimation regarding launching of e-version of journal of your stream time to time. This may be utilized in your library for the enrichment of knowledge of your students as well as it can also be helpful for the concerned faculty members.



After nomination of your institution as “Institutional Fellow” and constantly functioning successfully for one year, we can consider giving recognition to your institute to function as Regional/Zonal office on our behalf.

The board can also take up the additional allied activities for betterment after our consultation.

The following entitlements are applicable to individual Fellows:

Open Association of Research Society, U.S.A (OARS) By-laws states that an individual Fellow may use the designations as applicable, or the corresponding initials. The Credentials of individual Fellow and Associate designations signify that the individual has gained knowledge of the fundamental concepts. One is magnanimous and proficient in an expertise course covering the professional code of conduct, and follows recognized standards of practice.



Open Association of Research Society (US)/ Global Journals Incorporation (USA), as described in Corporate Statements, are educational, research publishing and professional membership organizations. Achieving our individual Fellow or Associate status is based mainly on meeting stated educational research requirements.

Disbursement of 40% Royalty earned through Global Journals : Researcher = 50%, Peer Reviewer = 37.50%, Institution = 12.50% E.g. Out of 40%, the 20% benefit should be passed on to researcher, 15 % benefit towards remuneration should be given to a reviewer and remaining 5% is to be retained by the institution.



We shall provide print version of 12 issues of any three journals [as per your requirement] out of our 38 journals worth \$ 2376 USD.

Other:

The individual Fellow and Associate designations accredited by Open Association of Research Society (US) credentials signify guarantees following achievements:

- The professional accredited with Fellow honor, is entitled to various benefits viz. name, fame, honor, regular flow of income, secured bright future, social status etc.



- In addition to above, if one is single author, then entitled to 40% discount on publishing research paper and can get 10% discount if one is co-author or main author among group of authors.
- The Fellow can organize symposium/seminar/conference on behalf of Global Journals Incorporation (USA) and he/she can also attend the same organized by other institutes on behalf of Global Journals.
- The Fellow can become member of Editorial Board Member after completing 3yrs.
- The Fellow can earn 60% of sales proceeds from the sale of reference/review books/literature/publishing of research paper.
- Fellow can also join as paid peer reviewer and earn 15% remuneration of author charges and can also get an opportunity to join as member of the Editorial Board of Global Journals Incorporation (USA)
- • This individual has learned the basic methods of applying those concepts and techniques to common challenging situations. This individual has further demonstrated an in-depth understanding of the application of suitable techniques to a particular area of research practice.

Note :

“

- In future, if the board feels the necessity to change any board member, the same can be done with the consent of the chairperson along with anyone board member without our approval.
- In case, the chairperson needs to be replaced then consent of 2/3rd board members are required and they are also required to jointly pass the resolution copy of which should be sent to us. In such case, it will be compulsory to obtain our approval before replacement.
- In case of “Difference of Opinion [if any]” among the Board members, our decision will be final and binding to everyone.

”



PROCESS OF SUBMISSION OF RESEARCH PAPER

The Area or field of specialization may or may not be of any category as mentioned in 'Scope of Journal' menu of the GlobalJournals.org website. There are 37 Research Journal categorized with Six parental Journals GJCST, GJMR, GJRE, GJMBR, GJSFR, GJHSS. For Authors should prefer the mentioned categories. There are three widely used systems UDC, DDC and LCC. The details are available as 'Knowledge Abstract' at Home page. The major advantage of this coding is that, the research work will be exposed to and shared with all over the world as we are being abstracted and indexed worldwide.

The paper should be in proper format. The format can be downloaded from first page of 'Author Guideline' Menu. The Author is expected to follow the general rules as mentioned in this menu. The paper should be written in MS-Word Format (*.DOC,*.DOCX).

The Author can submit the paper either online or offline. The authors should prefer online submission.Online Submission: There are three ways to submit your paper:

(A) (I) First, register yourself using top right corner of Home page then Login. If you are already registered, then login using your username and password.

(II) Choose corresponding Journal.

(III) Click 'Submit Manuscript'. Fill required information and Upload the paper.

(B) If you are using Internet Explorer, then Direct Submission through Homepage is also available.

(C) If these two are not convenient, and then email the paper directly to dean@globaljournals.org.

Offline Submission: Author can send the typed form of paper by Post. However, online submission should be preferred.

PREFERRED AUTHOR GUIDELINES

MANUSCRIPT STYLE INSTRUCTION (Must be strictly followed)

Page Size: 8.27" X 11"

- Left Margin: 0.65
- Right Margin: 0.65
- Top Margin: 0.75
- Bottom Margin: 0.75
- Font type of all text should be Swis 721 Lt BT.
- Paper Title should be of Font Size 24 with one Column section.
- Author Name in Font Size of 11 with one column as of Title.
- Abstract Font size of 9 Bold, "Abstract" word in Italic Bold.
- Main Text: Font size 10 with justified two columns section
- Two Column with Equal Column with of 3.38 and Gaping of .2
- First Character must be three lines Drop capped.
- Paragraph before Spacing of 1 pt and After of 0 pt.
- Line Spacing of 1 pt
- Large Images must be in One Column
- Numbering of First Main Headings (Heading 1) must be in Roman Letters, Capital Letter, and Font Size of 10.
- Numbering of Second Main Headings (Heading 2) must be in Alphabets, Italic, and Font Size of 10.

You can use your own standard format also.

Author Guidelines:

1. General,
2. Ethical Guidelines,
3. Submission of Manuscripts,
4. Manuscript's Category,
5. Structure and Format of Manuscript,
6. After Acceptance.

1. GENERAL

Before submitting your research paper, one is advised to go through the details as mentioned in following heads. It will be beneficial, while peer reviewer justify your paper for publication.

Scope

The Global Journals Inc. (US) welcome the submission of original paper, review paper, survey article relevant to the all the streams of Philosophy and knowledge. The Global Journals Inc. (US) is parental platform for Global Journal of Computer Science and Technology, Researches in Engineering, Medical Research, Science Frontier Research, Human Social Science, Management, and Business organization. The choice of specific field can be done otherwise as following in Abstracting and Indexing Page on this Website. As the all Global

Journals Inc. (US) are being abstracted and indexed (in process) by most of the reputed organizations. Topics of only narrow interest will not be accepted unless they have wider potential or consequences.

2. ETHICAL GUIDELINES

Authors should follow the ethical guidelines as mentioned below for publication of research paper and research activities.

Papers are accepted on strict understanding that the material in whole or in part has not been, nor is being, considered for publication elsewhere. If the paper once accepted by Global Journals Inc. (US) and Editorial Board, will become the copyright of the Global Journals Inc. (US).

Authorship: The authors and coauthors should have active contribution to conception design, analysis and interpretation of findings. They should critically review the contents and drafting of the paper. All should approve the final version of the paper before submission

The Global Journals Inc. (US) follows the definition of authorship set up by the Global Academy of Research and Development. According to the Global Academy of R&D authorship, criteria must be based on:

- 1) Substantial contributions to conception and acquisition of data, analysis and interpretation of the findings.
- 2) Drafting the paper and revising it critically regarding important academic content.
- 3) Final approval of the version of the paper to be published.

All authors should have been credited according to their appropriate contribution in research activity and preparing paper. Contributors who do not match the criteria as authors may be mentioned under Acknowledgement.

Acknowledgements: Contributors to the research other than authors credited should be mentioned under acknowledgement. The specifications of the source of funding for the research if appropriate can be included. Suppliers of resources may be mentioned along with address.

Appeal of Decision: The Editorial Board's decision on publication of the paper is final and cannot be appealed elsewhere.

Permissions: It is the author's responsibility to have prior permission if all or parts of earlier published illustrations are used in this paper.

Please mention proper reference and appropriate acknowledgements wherever expected.

If all or parts of previously published illustrations are used, permission must be taken from the copyright holder concerned. It is the author's responsibility to take these in writing.

Approval for reproduction/modification of any information (including figures and tables) published elsewhere must be obtained by the authors/copyright holders before submission of the manuscript. Contributors (Authors) are responsible for any copyright fee involved.

3. SUBMISSION OF MANUSCRIPTS

Manuscripts should be uploaded via this online submission page. The online submission is most efficient method for submission of papers, as it enables rapid distribution of manuscripts and consequently speeds up the review procedure. It also enables authors to know the status of their own manuscripts by emailing us. Complete instructions for submitting a paper is available below.

Manuscript submission is a systematic procedure and little preparation is required beyond having all parts of your manuscript in a given format and a computer with an Internet connection and a Web browser. Full help and instructions are provided on-screen. As an author, you will be prompted for login and manuscript details as Field of Paper and then to upload your manuscript file(s) according to the instructions.



To avoid postal delays, all transaction is preferred by e-mail. A finished manuscript submission is confirmed by e-mail immediately and your paper enters the editorial process with no postal delays. When a conclusion is made about the publication of your paper by our Editorial Board, revisions can be submitted online with the same procedure, with an occasion to view and respond to all comments.

Complete support for both authors and co-author is provided.

4. MANUSCRIPT'S CATEGORY

Based on potential and nature, the manuscript can be categorized under the following heads:

Original research paper: Such papers are reports of high-level significant original research work.

Review papers: These are concise, significant but helpful and decisive topics for young researchers.

Research articles: These are handled with small investigation and applications.

Research letters: The letters are small and concise comments on previously published matters.

5. STRUCTURE AND FORMAT OF MANUSCRIPT

The recommended size of original research paper is less than seven thousand words, review papers fewer than seven thousands words also. Preparation of research paper or how to write research paper, are major hurdle, while writing manuscript. The research articles and research letters should be fewer than three thousand words, the structure original research paper; sometime review paper should be as follows:

Papers: These are reports of significant research (typically less than 7000 words equivalent, including tables, figures, references), and comprise:

- (a) Title should be relevant and commensurate with the theme of the paper.
- (b) A brief Summary, "Abstract" (less than 150 words) containing the major results and conclusions.
- (c) Up to ten keywords, that precisely identifies the paper's subject, purpose, and focus.
- (d) An Introduction, giving necessary background excluding subheadings; objectives must be clearly declared.
- (e) Resources and techniques with sufficient complete experimental details (wherever possible by reference) to permit repetition; sources of information must be given and numerical methods must be specified by reference, unless non-standard.
- (f) Results should be presented concisely, by well-designed tables and/or figures; the same data may not be used in both; suitable statistical data should be given. All data must be obtained with attention to numerical detail in the planning stage. As reproduced design has been recognized to be important to experiments for a considerable time, the Editor has decided that any paper that appears not to have adequate numerical treatments of the data will be returned un-refereed;
- (g) Discussion should cover the implications and consequences, not just recapitulating the results; conclusions should be summarizing.
- (h) Brief Acknowledgements.
- (i) References in the proper form.

Authors should very cautiously consider the preparation of papers to ensure that they communicate efficiently. Papers are much more likely to be accepted, if they are cautiously designed and laid out, contain few or no errors, are summarizing, and be conventional to the approach and instructions. They will in addition, be published with much less delays than those that require much technical and editorial correction.



The Editorial Board reserves the right to make literary corrections and to make suggestions to improve brevity.

It is vital, that authors take care in submitting a manuscript that is written in simple language and adheres to published guidelines.

Format

Language: The language of publication is UK English. Authors, for whom English is a second language, must have their manuscript efficiently edited by an English-speaking person before submission to make sure that, the English is of high excellence. It is preferable, that manuscripts should be professionally edited.

Standard Usage, Abbreviations, and Units: Spelling and hyphenation should be conventional to The Concise Oxford English Dictionary. Statistics and measurements should at all times be given in figures, e.g. 16 min, except for when the number begins a sentence. When the number does not refer to a unit of measurement it should be spelt in full unless, it is 160 or greater.

Abbreviations supposed to be used carefully. The abbreviated name or expression is supposed to be cited in full at first usage, followed by the conventional abbreviation in parentheses.

Metric SI units are supposed to generally be used excluding where they conflict with current practice or are confusing. For illustration, 1.4 l rather than $1.4 \times 10^{-3} \text{ m}^3$, or 4 mm somewhat than $4 \times 10^{-3} \text{ m}$. Chemical formula and solutions must identify the form used, e.g. anhydrous or hydrated, and the concentration must be in clearly defined units. Common species names should be followed by underlines at the first mention. For following use the generic name should be constricted to a single letter, if it is clear.

Structure

All manuscripts submitted to Global Journals Inc. (US), ought to include:

Title: The title page must carry an instructive title that reflects the content, a running title (less than 45 characters together with spaces), names of the authors and co-authors, and the place(s) wherever the work was carried out. The full postal address in addition with the e-mail address of related author must be given. Up to eleven keywords or very brief phrases have to be given to help data retrieval, mining and indexing.

Abstract, used in Original Papers and Reviews:

Optimizing Abstract for Search Engines

Many researchers searching for information online will use search engines such as Google, Yahoo or similar. By optimizing your paper for search engines, you will amplify the chance of someone finding it. This in turn will make it more likely to be viewed and/or cited in a further work. Global Journals Inc. (US) have compiled these guidelines to facilitate you to maximize the web-friendliness of the most public part of your paper.

Key Words

A major linchpin in research work for the writing research paper is the keyword search, which one will employ to find both library and Internet resources.

One must be persistent and creative in using keywords. An effective keyword search requires a strategy and planning a list of possible keywords and phrases to try.

Search engines for most searches, use Boolean searching, which is somewhat different from Internet searches. The Boolean search uses "operators," words (and, or, not, and near) that enable you to expand or narrow your affords. Tips for research paper while preparing research paper are very helpful guideline of research paper.

Choice of key words is first tool of tips to write research paper. Research paper writing is an art. A few tips for deciding as strategically as possible about keyword search:



- One should start brainstorming lists of possible keywords before even begin searching. Think about the most important concepts related to research work. Ask, "What words would a source have to include to be truly valuable in research paper?" Then consider synonyms for the important words.
- It may take the discovery of only one relevant paper to let steer in the right keyword direction because in most databases, the keywords under which a research paper is abstracted are listed with the paper.
- One should avoid outdated words.

Keywords are the key that opens a door to research work sources. Keyword searching is an art in which researcher's skills are bound to improve with experience and time.

Numerical Methods: Numerical methods used should be clear and, where appropriate, supported by references.

Acknowledgements: Please make these as concise as possible.

References

References follow the Harvard scheme of referencing. References in the text should cite the authors' names followed by the time of their publication, unless there are three or more authors when simply the first author's name is quoted followed by et al. unpublished work has to only be cited where necessary, and only in the text. Copies of references in press in other journals have to be supplied with submitted typescripts. It is necessary that all citations and references be carefully checked before submission, as mistakes or omissions will cause delays.

References to information on the World Wide Web can be given, but only if the information is available without charge to readers on an official site. Wikipedia and Similar websites are not allowed where anyone can change the information. Authors will be asked to make available electronic copies of the cited information for inclusion on the Global Journals Inc. (US) homepage at the judgment of the Editorial Board.

The Editorial Board and Global Journals Inc. (US) recommend that, citation of online-published papers and other material should be done via a DOI (digital object identifier). If an author cites anything, which does not have a DOI, they run the risk of the cited material not being noticeable.

The Editorial Board and Global Journals Inc. (US) recommend the use of a tool such as Reference Manager for reference management and formatting.

Tables, Figures and Figure Legends

Tables: Tables should be few in number, cautiously designed, uncrowned, and include only essential data. Each must have an Arabic number, e.g. Table 4, a self-explanatory caption and be on a separate sheet. Vertical lines should not be used.

Figures: Figures are supposed to be submitted as separate files. Always take in a citation in the text for each figure using Arabic numbers, e.g. Fig. 4. Artwork must be submitted online in electronic form by e-mailing them.

Preparation of Electronic Figures for Publication

Even though low quality images are sufficient for review purposes, print publication requires high quality images to prevent the final product being blurred or fuzzy. Submit (or e-mail) EPS (line art) or TIFF (halftone/photographs) files only. MS PowerPoint and Word Graphics are unsuitable for printed pictures. Do not use pixel-oriented software. Scans (TIFF only) should have a resolution of at least 350 dpi (halftone) or 700 to 1100 dpi (line drawings) in relation to the imitation size. Please give the data for figures in black and white or submit a Color Work Agreement Form. EPS files must be saved with fonts embedded (and with a TIFF preview, if possible).

For scanned images, the scanning resolution (at final image size) ought to be as follows to ensure good reproduction: line art: >650 dpi; halftones (including gel photographs) : >350 dpi; figures containing both halftone and line images: >650 dpi.

Color Charges: It is the rule of the Global Journals Inc. (US) for authors to pay the full cost for the reproduction of their color artwork. Hence, please note that, if there is color artwork in your manuscript when it is accepted for publication, we would require you to complete and return a color work agreement form before your paper can be published.



Figure Legends: Self-explanatory legends of all figures should be incorporated separately under the heading 'Legends to Figures'. In the full-text online edition of the journal, figure legends may possibly be truncated in abbreviated links to the full screen version. Therefore, the first 100 characters of any legend should notify the reader, about the key aspects of the figure.

6. AFTER ACCEPTANCE

Upon approval of a paper for publication, the manuscript will be forwarded to the dean, who is responsible for the publication of the Global Journals Inc. (US).

6.1 Proof Corrections

The corresponding author will receive an e-mail alert containing a link to a website or will be attached. A working e-mail address must therefore be provided for the related author.

Acrobat Reader will be required in order to read this file. This software can be downloaded

(Free of charge) from the following website:

www.adobe.com/products/acrobat/readstep2.html. This will facilitate the file to be opened, read on screen, and printed out in order for any corrections to be added. Further instructions will be sent with the proof.

Proofs must be returned to the dean at dean@globaljournals.org within three days of receipt.

As changes to proofs are costly, we inquire that you only correct typesetting errors. All illustrations are retained by the publisher. Please note that the authors are responsible for all statements made in their work, including changes made by the copy editor.

6.2 Early View of Global Journals Inc. (US) (Publication Prior to Print)

The Global Journals Inc. (US) are enclosed by our publishing's Early View service. Early View articles are complete full-text articles sent in advance of their publication. Early View articles are absolute and final. They have been completely reviewed, revised and edited for publication, and the authors' final corrections have been incorporated. Because they are in final form, no changes can be made after sending them. The nature of Early View articles means that they do not yet have volume, issue or page numbers, so Early View articles cannot be cited in the conventional way.

6.3 Author Services

Online production tracking is available for your article through Author Services. Author Services enables authors to track their article - once it has been accepted - through the production process to publication online and in print. Authors can check the status of their articles online and choose to receive automated e-mails at key stages of production. The authors will receive an e-mail with a unique link that enables them to register and have their article automatically added to the system. Please ensure that a complete e-mail address is provided when submitting the manuscript.

6.4 Author Material Archive Policy

Please note that if not specifically requested, publisher will dispose off hardcopy & electronic information submitted, after the two months of publication. If you require the return of any information submitted, please inform the Editorial Board or dean as soon as possible.

6.5 Offprint and Extra Copies

A PDF offprint of the online-published article will be provided free of charge to the related author, and may be distributed according to the Publisher's terms and conditions. Additional paper offprint may be ordered by emailing us at: editor@globaljournals.org.

You must strictly follow above Author Guidelines before submitting your paper or else we will not at all be responsible for any corrections in future in any of the way.



Before start writing a good quality Computer Science Research Paper, let us first understand what is Computer Science Research Paper? So, Computer Science Research Paper is the paper which is written by professionals or scientists who are associated to Computer Science and Information Technology, or doing research study in these areas. If you are novel to this field then you can consult about this field from your supervisor or guide.

TECHNIQUES FOR WRITING A GOOD QUALITY RESEARCH PAPER:

1. Choosing the topic: In most cases, the topic is searched by the interest of author but it can be also suggested by the guides. You can have several topics and then you can judge that in which topic or subject you are finding yourself most comfortable. This can be done by asking several questions to yourself, like Will I be able to carry our search in this area? Will I find all necessary recourses to accomplish the search? Will I be able to find all information in this field area? If the answer of these types of questions will be "Yes" then you can choose that topic. In most of the cases, you may have to conduct the surveys and have to visit several places because this field is related to Computer Science and Information Technology. Also, you may have to do a lot of work to find all rise and falls regarding the various data of that subject. Sometimes, detailed information plays a vital role, instead of short information.

2. Evaluators are human: First thing to remember that evaluators are also human being. They are not only meant for rejecting a paper. They are here to evaluate your paper. So, present your Best.

3. Think Like Evaluators: If you are in a confusion or getting demotivated that your paper will be accepted by evaluators or not, then think and try to evaluate your paper like an Evaluator. Try to understand that what an evaluator wants in your research paper and automatically you will have your answer.

4. Make blueprints of paper: The outline is the plan or framework that will help you to arrange your thoughts. It will make your paper logical. But remember that all points of your outline must be related to the topic you have chosen.

5. Ask your Guides: If you are having any difficulty in your research, then do not hesitate to share your difficulty to your guide (if you have any). They will surely help you out and resolve your doubts. If you can't clarify what exactly you require for your work then ask the supervisor to help you with the alternative. He might also provide you the list of essential readings.

6. Use of computer is recommended: As you are doing research in the field of Computer Science, then this point is quite obvious.

7. Use right software: Always use good quality software packages. If you are not capable to judge good software then you can lose quality of your paper unknowingly. There are various software programs available to help you, which you can get through Internet.

8. Use the Internet for help: An excellent start for your paper can be by using the Google. It is an excellent search engine, where you can have your doubts resolved. You may also read some answers for the frequent question how to write my research paper or find model research paper. From the internet library you can download books. If you have all required books make important reading selecting and analyzing the specified information. Then put together research paper sketch out.

9. Use and get big pictures: Always use encyclopedias, Wikipedia to get pictures so that you can go into the depth.

10. Bookmarks are useful: When you read any book or magazine, you generally use bookmarks, right! It is a good habit, which helps to not to lose your continuity. You should always use bookmarks while searching on Internet also, which will make your search easier.

11. Revise what you wrote: When you write anything, always read it, summarize it and then finalize it.



12. Make all efforts: Make all efforts to mention what you are going to write in your paper. That means always have a good start. Try to mention everything in introduction, that what is the need of a particular research paper. Polish your work by good skill of writing and always give an evaluator, what he wants.

13. Have backups: When you are going to do any important thing like making research paper, you should always have backup copies of it either in your computer or in paper. This will help you to not to lose any of your important.

14. Produce good diagrams of your own: Always try to include good charts or diagrams in your paper to improve quality. Using several and unnecessary diagrams will degrade the quality of your paper by creating "hotchpotch." So always, try to make and include those diagrams, which are made by your own to improve readability and understandability of your paper.

15. Use of direct quotes: When you do research relevant to literature, history or current affairs then use of quotes become essential but if study is relevant to science then use of quotes is not preferable.

16. Use proper verb tense: Use proper verb tenses in your paper. Use past tense, to present those events that happened. Use present tense to indicate events that are going on. Use future tense to indicate future happening events. Use of improper and wrong tenses will confuse the evaluator. Avoid the sentences that are incomplete.

17. Never use online paper: If you are getting any paper on Internet, then never use it as your research paper because it might be possible that evaluator has already seen it or maybe it is outdated version.

18. Pick a good study spot: To do your research studies always try to pick a spot, which is quiet. Every spot is not for studies. Spot that suits you choose it and proceed further.

19. Know what you know: Always try to know, what you know by making objectives. Else, you will be confused and cannot achieve your target.

20. Use good quality grammar: Always use a good quality grammar and use words that will throw positive impact on evaluator. Use of good quality grammar does not mean to use tough words, that for each word the evaluator has to go through dictionary. Do not start sentence with a conjunction. Do not fragment sentences. Eliminate one-word sentences. Ignore passive voice. Do not ever use a big word when a diminutive one would suffice. Verbs have to be in agreement with their subjects. Prepositions are not expressions to finish sentences with. It is incorrect to ever divide an infinitive. Avoid clichés like the disease. Also, always shun irritating alliteration. Use language that is simple and straight forward. put together a neat summary.

21. Arrangement of information: Each section of the main body should start with an opening sentence and there should be a changeover at the end of the section. Give only valid and powerful arguments to your topic. You may also maintain your arguments with records.

22. Never start in last minute: Always start at right time and give enough time to research work. Leaving everything to the last minute will degrade your paper and spoil your work.

23. Multitasking in research is not good: Doing several things at the same time proves bad habit in case of research activity. Research is an area, where everything has a particular time slot. Divide your research work in parts and do particular part in particular time slot.

24. Never copy others' work: Never copy others' work and give it your name because if evaluator has seen it anywhere you will be in trouble.

25. Take proper rest and food: No matter how many hours you spend for your research activity, if you are not taking care of your health then all your efforts will be in vain. For a quality research, study is must, and this can be done by taking proper rest and food.

26. Go for seminars: Attend seminars if the topic is relevant to your research area. Utilize all your resources.



27. Refresh your mind after intervals: Try to give rest to your mind by listening to soft music or by sleeping in intervals. This will also improve your memory.

28. Make colleagues: Always try to make colleagues. No matter how sharper or intelligent you are, if you make colleagues you can have several ideas, which will be helpful for your research.

29. Think technically: Always think technically. If anything happens, then search its reasons, its benefits, and demerits.

30. Think and then print: When you will go to print your paper, notice that tables are not be split, headings are not detached from their descriptions, and page sequence is maintained.

31. Adding unnecessary information: Do not add unnecessary information, like, I have used MS Excel to draw graph. Do not add irrelevant and inappropriate material. These all will create superfluous. Foreign terminology and phrases are not apropos. One should NEVER take a broad view. Analogy in script is like feathers on a snake. Not at all use a large word when a very small one would be sufficient. Use words properly, regardless of how others use them. Remove quotations. Puns are for kids, not grunt readers. Amplification is a billion times of inferior quality than sarcasm.

32. Never oversimplify everything: To add material in your research paper, never go for oversimplification. This will definitely irritate the evaluator. Be more or less specific. Also too, by no means, ever use rhythmic redundancies. Contractions aren't essential and shouldn't be there used. Comparisons are as terrible as clichés. Give up ampersands and abbreviations, and so on. Remove commas, that are, not necessary. Parenthetical words however should be together with this in commas. Understatement is all the time the complete best way to put onward earth-shaking thoughts. Give a detailed literary review.

33. Report concluded results: Use concluded results. From raw data, filter the results and then conclude your studies based on measurements and observations taken. Significant figures and appropriate number of decimal places should be used. Parenthetical remarks are prohibitive. Proofread carefully at final stage. In the end give outline to your arguments. Spot out perspectives of further study of this subject. Justify your conclusion by at the bottom of them with sufficient justifications and examples.

34. After conclusion: Once you have concluded your research, the next most important step is to present your findings. Presentation is extremely important as it is the definite medium through which your research is going to be in print to the rest of the crowd. Care should be taken to categorize your thoughts well and present them in a logical and neat manner. A good quality research paper format is essential because it serves to highlight your research paper and bring to light all necessary aspects in your research.

INFORMAL GUIDELINES OF RESEARCH PAPER WRITING

Key points to remember:

- Submit all work in its final form.
- Write your paper in the form, which is presented in the guidelines using the template.
- Please note the criterion for grading the final paper by peer-reviewers.

Final Points:

A purpose of organizing a research paper is to let people to interpret your effort selectively. The journal requires the following sections, submitted in the order listed, each section to start on a new page.

The introduction will be compiled from reference matter and will reflect the design processes or outline of basis that direct you to make study. As you will carry out the process of study, the method and process section will be constructed as like that. The result segment will show related statistics in nearly sequential order and will direct the reviewers next to the similar intellectual paths throughout the data that you took to carry out your study. The discussion section will provide understanding of the data and projections as to the implication of the results. The use of good quality references all through the paper will give the effort trustworthiness by representing an alertness of prior workings.



Writing a research paper is not an easy job no matter how trouble-free the actual research or concept. Practice, excellent preparation, and controlled record keeping are the only means to make straightforward the progression.

General style:

Specific editorial column necessities for compliance of a manuscript will always take over from directions in these general guidelines.

To make a paper clear

- Adhere to recommended page limits

Mistakes to evade

- Insertion a title at the foot of a page with the subsequent text on the next page
- Separating a table/chart or figure - impound each figure/table to a single page
- Submitting a manuscript with pages out of sequence

In every sections of your document

- Use standard writing style including articles ("a", "the," etc.)
- Keep on paying attention on the research topic of the paper
- Use paragraphs to split each significant point (excluding for the abstract)
- Align the primary line of each section
- Present your points in sound order
- Use present tense to report well accepted
- Use past tense to describe specific results
- Shun familiar wording, don't address the reviewer directly, and don't use slang, slang language, or superlatives
- Shun use of extra pictures - include only those figures essential to presenting results

Title Page:

Choose a revealing title. It should be short. It should not have non-standard acronyms or abbreviations. It should not exceed two printed lines. It should include the name(s) and address (es) of all authors.



Abstract:

The summary should be two hundred words or less. It should briefly and clearly explain the key findings reported in the manuscript-- must have precise statistics. It should not have abnormal acronyms or abbreviations. It should be logical in itself. Shun citing references at this point.

An abstract is a brief distinct paragraph summary of finished work or work in development. In a minute or less a reviewer can be taught the foundation behind the study, common approach to the problem, relevant results, and significant conclusions or new questions.

Write your summary when your paper is completed because how can you write the summary of anything which is not yet written? Wealth of terminology is very essential in abstract. Yet, use comprehensive sentences and do not let go readability for briefness. You can maintain it succinct by phrasing sentences so that they provide more than lone rationale. The author can at this moment go straight to shortening the outcome. Sum up the study, with the subsequent elements in any summary. Try to maintain the initial two items to no more than one ruling each.

- Reason of the study - theory, overall issue, purpose
- Fundamental goal
- To the point depiction of the research
- Consequences, including definite statistics - if the consequences are quantitative in nature, account quantitative data; results of any numerical analysis should be reported
- Significant conclusions or questions that track from the research(es)

Approach:

- Single section, and succinct
- As a outline of job done, it is always written in past tense
- A conceptual should situate on its own, and not submit to any other part of the paper such as a form or table
- Center on shortening results - bound background information to a verdict or two, if completely necessary
- What you account in an conceptual must be regular with what you reported in the manuscript
- Exact spelling, clearness of sentences and phrases, and appropriate reporting of quantities (proper units, important statistics) are just as significant in an abstract as they are anywhere else

Introduction:

The **Introduction** should "introduce" the manuscript. The reviewer should be presented with sufficient background information to be capable to comprehend and calculate the purpose of your study without having to submit to other works. The basis for the study should be offered. Give most important references but shun difficult to make a comprehensive appraisal of the topic. In the introduction, describe the problem visibly. If the problem is not acknowledged in a logical, reasonable way, the reviewer will have no attention in your result. Speak in common terms about techniques used to explain the problem, if needed, but do not present any particulars about the protocols here. Following approach can create a valuable beginning:

- Explain the value (significance) of the study
- Shield the model - why did you employ this particular system or method? What is its compensation? You strength remark on its appropriateness from a abstract point of vision as well as point out sensible reasons for using it.
- Present a justification. Status your particular theory (es) or aim(s), and describe the logic that led you to choose them.
- Very for a short time explain the tentative propose and how it skilled the declared objectives.

Approach:

- Use past tense except for when referring to recognized facts. After all, the manuscript will be submitted after the entire job is done.
- Sort out your thoughts; manufacture one key point with every section. If you make the four points listed above, you will need a least of four paragraphs.



- Present surroundings information only as desirable in order hold up a situation. The reviewer does not desire to read the whole thing you know about a topic.
- Shape the theory/purpose specifically - do not take a broad view.
- As always, give awareness to spelling, simplicity and correctness of sentences and phrases.

Procedures (Methods and Materials):

This part is supposed to be the easiest to carve if you have good skills. A sound written Procedures segment allows a capable scientist to replacement your results. Present precise information about your supplies. The suppliers and clarity of reagents can be helpful bits of information. Present methods in sequential order but linked methodologies can be grouped as a segment. Be concise when relating the protocols. Attempt for the least amount of information that would permit another capable scientist to spare your outcome but be cautious that vital information is integrated. The use of subheadings is suggested and ought to be synchronized with the results section. When a technique is used that has been well described in another object, mention the specific item describing a way but draw the basic principle while stating the situation. The purpose is to text all particular resources and broad procedures, so that another person may use some or all of the methods in one more study or referee the scientific value of your work. It is not to be a step by step report of the whole thing you did, nor is a methods section a set of orders.

Materials:

- Explain materials individually only if the study is so complex that it saves liberty this way.
- Embrace particular materials, and any tools or provisions that are not frequently found in laboratories.
- Do not take in frequently found.
- If use of a definite type of tools.
- Materials may be reported in a part section or else they may be recognized along with your measures.

Methods:

- Report the method (not particulars of each process that engaged the same methodology)
- Describe the method entirely
- To be succinct, present methods under headings dedicated to specific dealings or groups of measures
- Simplify - details how procedures were completed not how they were exclusively performed on a particular day.
- If well known procedures were used, account the procedure by name, possibly with reference, and that's all.

Approach:

- It is embarrassed or not possible to use vigorous voice when documenting methods with no using first person, which would focus the reviewer's interest on the researcher rather than the job. As a result when script up the methods most authors use third person passive voice.
- Use standard style in this and in every other part of the paper - avoid familiar lists, and use full sentences.

What to keep away from

- Resources and methods are not a set of information.
- Skip all descriptive information and surroundings - save it for the argument.
- Leave out information that is immaterial to a third party.

Results:

The principle of a results segment is to present and demonstrate your conclusion. Create this part a entirely objective details of the outcome, and save all understanding for the discussion.

The page length of this segment is set by the sum and types of data to be reported. Carry on to be to the point, by means of statistics and tables, if suitable, to present consequences most efficiently. You must obviously differentiate material that would usually be incorporated in a study editorial from any unprocessed data or additional appendix matter that would not be available. In fact, such matter should not be submitted at all except requested by the instructor.



Content

- Sum up your conclusion in text and demonstrate them, if suitable, with figures and tables.
- In manuscript, explain each of your consequences, point the reader to remarks that are most appropriate.
- Present a background, such as by describing the question that was addressed by creation an exacting study.
- Explain results of control experiments and comprise remarks that are not accessible in a prescribed figure or table, if appropriate.
- Examine your data, then prepare the analyzed (transformed) data in the form of a figure (graph), table, or in manuscript form.

What to stay away from

- Do not discuss or infer your outcome, report surroundings information, or try to explain anything.
- Not at all, take in raw data or intermediate calculations in a research manuscript.
- Do not present the similar data more than once.
- Manuscript should complement any figures or tables, not duplicate the identical information.
- Never confuse figures with tables - there is a difference.

Approach

- As forever, use past tense when you submit to your results, and put the whole thing in a reasonable order.
- Put figures and tables, appropriately numbered, in order at the end of the report
- If you desire, you may place your figures and tables properly within the text of your results part.

Figures and tables

- If you put figures and tables at the end of the details, make certain that they are visibly distinguished from any attach appendix materials, such as raw facts
- Despite of position, each figure must be numbered one after the other and complete with subtitle
- In spite of position, each table must be titled, numbered one after the other and complete with heading
- All figure and table must be adequately complete that it could situate on its own, divide from text

Discussion:

The Discussion is expected the trickiest segment to write and describe. A lot of papers submitted for journal are discarded based on problems with the Discussion. There is no head of state for how long a argument should be. Position your understanding of the outcome visibly to lead the reviewer through your conclusions, and then finish the paper with a summing up of the implication of the study. The purpose here is to offer an understanding of your results and hold up for all of your conclusions, using facts from your research and generally accepted information, if suitable. The implication of result should be visibly described. Infer your data in the conversation in suitable depth. This means that when you clarify an observable fact you must explain mechanisms that may account for the observation. If your results vary from your prospect, make clear why that may have happened. If your results agree, then explain the theory that the proof supported. It is never suitable to just state that the data approved with prospect, and let it drop at that.

- Make a decision if each premise is supported, discarded, or if you cannot make a conclusion with assurance. Do not just dismiss a study or part of a study as "uncertain."
- Research papers are not acknowledged if the work is imperfect. Draw what conclusions you can based upon the results that you have, and take care of the study as a finished work
- You may propose future guidelines, such as how the experiment might be personalized to accomplish a new idea.
- Give details all of your remarks as much as possible, focus on mechanisms.
- Make a decision if the tentative design sufficiently addressed the theory, and whether or not it was correctly restricted.
- Try to present substitute explanations if sensible alternatives be present.
- One research will not counter an overall question, so maintain the large picture in mind, where do you go next? The best studies unlock new avenues of study. What questions remain?
- Recommendations for detailed papers will offer supplementary suggestions.

Approach:

- When you refer to information, differentiate data generated by your own studies from available information
- Submit to work done by specific persons (including you) in past tense.
- Submit to generally acknowledged facts and main beliefs in present tense.



THE ADMINISTRATION RULES

Please carefully note down following rules and regulation before submitting your Research Paper to Global Journals Inc. (US):

Segment Draft and Final Research Paper: You have to strictly follow the template of research paper. If it is not done your paper may get rejected.

- The **major constraint** is that you must independently make all content, tables, graphs, and facts that are offered in the paper. You must write each part of the paper wholly on your own. The Peer-reviewers need to identify your own perceptives of the concepts in your own terms. NEVER extract straight from any foundation, and never rephrase someone else's analysis.
- Do not give permission to anyone else to "PROOFREAD" your manuscript.
- **Methods to avoid Plagiarism is applied by us on every paper, if found guilty, you will be blacklisted by all of our collaborated research groups, your institution will be informed for this and strict legal actions will be taken immediately.)**
- To guard yourself and others from possible illegal use please do not permit anyone right to use to your paper and files.



CRITERION FOR GRADING A RESEARCH PAPER (COMPILATION)
BY GLOBAL JOURNALS INC. (US)

Please note that following table is only a Grading of "Paper Compilation" and not on "Performed/Stated Research" whose grading solely depends on Individual Assigned Peer Reviewer and Editorial Board Member. These can be available only on request and after decision of Paper. This report will be the property of Global Journals Inc. (US).

Topics	Grades		
	A-B	C-D	E-F
<i>Abstract</i>	Clear and concise with appropriate content, Correct format. 200 words or below	Unclear summary and no specific data, Incorrect form Above 200 words	No specific data with ambiguous information Above 250 words
<i>Introduction</i>	Containing all background details with clear goal and appropriate details, flow specification, no grammar and spelling mistake, well organized sentence and paragraph, reference cited	Unclear and confusing data, appropriate format, grammar and spelling errors with unorganized matter	Out of place depth and content, hazy format
<i>Methods and Procedures</i>	Clear and to the point with well arranged paragraph, precision and accuracy of facts and figures, well organized subheads	Difficult to comprehend with embarrassed text, too much explanation but completed	Incorrect and unorganized structure with hazy meaning
<i>Result</i>	Well organized, Clear and specific, Correct units with precision, correct data, well structuring of paragraph, no grammar and spelling mistake	Complete and embarrassed text, difficult to comprehend	Irregular format with wrong facts and figures
<i>Discussion</i>	Well organized, meaningful specification, sound conclusion, logical and concise explanation, highly structured paragraph reference cited	Wordy, unclear conclusion, spurious	Conclusion is not cited, unorganized, difficult to comprehend
<i>References</i>	Complete and correct format, well organized	Beside the point, Incomplete	Wrong format and structuring



INDEX

A

Abysmal · 14
Arbitrary · 17

B

Bayesian · 16
Billirubine · 27

C

Cirrhosis · 26, 30
Cybernetics · 30

D

Dirichlet · 17, 24, 25

H

Hepatorenal · 26, 27
Heretrajectory · 17
Hierarchical · 16, 17, 24, 25

L

Laquso · 1, 4

R

Redundant · 24, 27

S

Schizophrenia · 30

U

Unambiguous · 1, 2



save our planet



Global Journal of Computer Science and Technology

Visit us on the Web at www.GlobalJournals.org | www.ComputerResearch.org
or email us at helpdesk@globaljournals.org



ISSN 9754350