Atul Negi · Raj Bhatnagar
Laxmi Parida (Eds.)

LNCS 10722

Distributed Computing and Internet Technology

14th International Conference, ICDCIT 2018
Bhubaneswar, India, January 11–13, 2018
Proceedings

Springer

Lecture Notes in Computer Science 10722

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

More information about this series at http://www.springer.com/series/7409

Atul Negi · Raj Bhatnagar
Laxmi Parida (Eds.)

Distributed Computing and Internet Technology

14th International Conference, ICDCIT 2018
Bhubaneswar, India, January 11–13, 2018
Proceedings

Springer

Editors
Atul Negi
University of Hyderabad
Hyderabad
India

Laxmi Parida
IBM Thomas J. Watson Research Center
Yorktown Heights, NY
USA

Raj Bhatnagar
University of Cincinnati
Cincinnati, OH
USA

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-319-72343-3 ISBN 978-3-319-72344-0 (eBook)
https://doi.org/10.1007/978-3-319-72344-0

Library of Congress Control Number: 2017960870

LNCS Sublibrary: SL3 – Information Systems and Applications, incl. Internet/Web, and HCI

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

It is our pleasure to welcome you to the proceedings of the 14th International Conference on Distributed Computing and Internet Technology, ICDCIT-2018, held in Bhubaneswar, India, during January 11–13, 2018. The conference was sponsored by the Kalinga Institute of Industrial Technology (KIIT) University and hosted on their campus. The ICDCIT conference series focuses on foundations and applications of distributed computing and Internet technologies. Year-on-year the conference topics have kept abreast with current advances in the subject. The conference aim is to enable academics, researchers, including students, practitioners, and developers to present their research findings and also to have an exchange of ideas on various relevant topics. Since the inception of the ICDCIT series, the conference proceedings have been published by Springer as *Lecture Notes in Computer Science*, and the 2018 volume is number 10722 in the series.

The call for papers attracted 154 abstracts and finally 120 full submissions. The full versions were reviewed by members of the Program Committee (PC) and other reviewers. On average each paper received three reviews and some had more than that. After receiving the reviews, the PC had an electronic discussion to finalize the acceptance of the submissions. After a robust discussion that took into account technical merit, presentation style, and relevance to the conference, a total of 32 papers (about 27%) were accepted. Of these, nine were accepted as full papers, 11 as short papers, and 12 as poster papers. Five papers were withdrawn or failed to register.

We wish to thank all the authors for their contributions, and also thank all the 38 PC members and 24 additional reviewers for their diligent reviews, which enabled us to have a quality program. The past PC chairs Paddy Krishnan and P. Radhakrishna were generous with their support in the reviews and transfer of learning to the new PC chairs.

The program included invited speakers who also contributed with their extended invited papers. We are grateful to Prof. Lenore Zuck (UIC, USA), Prof. Andre Rossi (University of Angers, France) and Prof. P. Srinivasa Kumar (IIT Madras, India). We are very thankful to all the invited speakers for taking time out of their busy schedule and for sharing their expertise.

We extend our special thanks to KIIT for their generous support and express our gratitude to Dr. Achyuta Samanta (Founder of KIIT University) for his patronage and to the vice-chancellor and administration of the university for providing us with the infrastructure and logistical arrangements. We thankfully acknowledge the support KIIT provides in hosting ICDCIT conferences since the inception of the series.

We are grateful to the Advisory Committee members for their guidance on all matters pertaining to the conference. We also greatly appreciate the invaluable support and tireless efforts of the organizing chair, finance chair, publicity chair, registration chair, session management chair, the publications chair, and all members of various committees who made a great contribution to the conference's success. We would like to thank Chittaranjan Pradhan in particular, for his help in communicating all matters related to registration and submissions. Hrushikesha Mohanty and N. Raja deserve special mention for their unwavering support, guidance, and timely advice and for propping up the program chairs when in need. In particular, we thank N. Raja for the able guidance from the Steering Committee for specific tricky issues. It is our pleasure to acknowledge EasyChair for enabling efficient and smooth handling of all activities starting from paper submissions to preparation of the proceedings.

We sincerely thank Alfred Hofmann and Anna Kramer from Springer for their cooperation and constant support throughout the publication process of this LNCS volume. We wish to specifically acknowledge the financial support offer received from Springer. Finally, we thank all the participants, without whom there would have been no conference. We hope you find the proceedings to be valuable for your professional development.

Bhubaneswar
January 2018

Atul Negi
Raj Bhatnagar
Laxmi Parida

Organization

Program Committee

Rafah Almuttairi	University of Babylon, Iraq
Kavitha Ammayappan	Robert Bosch Engineering and Business Solutions Limited, Bangalore, India
Kavitha Athota	JNT University Hyderabad, India
Gowtham Atluri	University of Cincinnati, USA
Raj Bhatnagar	University of Cincinnati, USA
Hung Dang Van	UET, Vietnam National University, Hanoi, Vietnam
Manik Lal Das	DA-IICT, Gandhinagar, India
Elsa Estevez	Universidad Nacional del Sur, Argentina
Günter Fahrnberger	University of Hagen, North Rhine-Westphalia, Germany
Pablo Fillottrani	Universidad Nacional del Sur, Argentina
G. R. Gangadharan	Institute for Development and Research in Banking Technology, Hyderabad, India
P. Jagadeesh	Kohls Departmental Stores Inc.
Vineet Joshi	University of Cincinnati, USA
Aneesh Krishna	Curtin University, Australia
Paddy Krishnan	Oracle, Australia
Pradeep Kumar	Indian Institute of Management Lucknow, India
Markus Lumpe	Swinburne University of Technology, Melbourne, Australia
Kamesh Madduri	The Pennsylvania State University, USA
Hrushikesha Mohanty	University of Hyderabad and KIIT Bhubaneswar
Dmitry Namiot	MSU, Russia
Raja Natarajan	TIFR, Mumbai, India
Atul Negi	University of Hyderabad, Telangana, India
N. Parimala	JNU, New Delhi, India
Syam Kumar Pasupuleti	Institute for Development and Research in Banking Technology, Hyderabad, India
Manas Ranjan Patra	Berhampur University, Odisha, India
Dana Petcu	West University of Timisoara, Romania
Radha Krishna Pisipati	SET Labs, Infosys Technologies Limited, Hyderabad, India
Deepak Poola	The University of Melbourne, Australia
Anupama Potluri	University of Hyderabad, Telangana, India
Jitendra Kumar Rai	ANURAG, Hyderabad, India
R. Ramanujam	Institute of Mathematical Sciences, Chennai, India
Divya Sardana	University of Cincinnati, USA

Akella Sastry	Xilinx, USA
Nagesh Bhattu Sristy	Institute for Development and Research in Banking Technology, Hyderabad, India
Ibrahim Tabash	Palestine University
Rajeev Wankar	University of Hyderabad, India
Shomir Wilson	Carnegie Mellon University, USA
Ennan Zhai	Yale University, USA

Additional Reviewers

Atluri, Vani Vathsala
Busi Reddy, Vijender
Cenci, Karina
Das, Madhabananda
Ghosh, Sujata
Goriparthi, Thulasi
Kadappa, Vijayakumar
Majumdar, Diptapriyo
Maruthi, Padmaja
Mohanty, Sachi
Mohanty, Jnyanaranjan
Mohapatra, Debasis
P. S. V. S., Sai Prasad
Pattnaik, Prasant K.
Prasad, M. V. N. K.
Reddy, V. Dinesh
Rendla, Chandrashekar
Roy, Monideepa
S., Sheerazuddin
Shankar Prasad Mishra, Bhabani
Subba Rao, Y. V.
Sundararajan, Vaishnavi
Suresh, S. P.
Vaddi, Supriya

Contents

Invited Papers

Security and Privacy

Distributed and Multiprocessing Approaches

Networks Protocols and Applications

Databases, Algorithms, Data Processing and Applications

Invited Papers

Properties and Exact Solution Approaches for the Minimum Cost Dominating Tree Problem

André Rossi[1](✉), Alok Singh[2], and Shyam Sundar[3]

[1] LERIA, Université d'Angers, Angers, France
andre.rossi@univ-angers.fr
[2] School of Computer and Information Sciences, University of Hyderabad, Hyderabad, India
[3] National Institute of Technology, Raipur, India

1 Introduction

The problem under consideration is called the minimum cost dominating tree problem [6]. It arises in the context of wireless mobile communication, when building a *virtual backbone* is required. A virtual backbone is a set of vertices (mobile devices) that will root the communication in the network. Let $G = (V, E)$ be an edge weighted, simple, connected and undirected graph that models the communication network, with $n = |V|$ and $m = |E|$. The problem is to build a tree of minimum cost such that each vertex is either covered by the tree, or is adjacent to a vertex that is covered by the tree. The cost of edge e is denoted by w_e for all $e \in E$, and it is assumed to be strictly positive. It represents the total energy cost of establishing communication between the endpoints of the corresponding edge.

Figure 1 displays a graph G on $n = 11$ vertices and $m = 13$ edges. A minimum cost dominating tree is also shown, its vertex set is in grey, and its edge set is in thick lines. The total cost of this solution is 4. It can be observed that the dominating tree made of the vertices 5, 6, 7 has only two edges, but its cost is 16, hence minimizing the total cost is not equivalent to minimizing the number of edges.

Results on the minimum cost dominating tree problem approximability can be found on [6]. Heuristic approaches are available in [7,9]. Exact approaches are proposed in [1]. Before presenting two new integer linear programming-based exact approaches in Sect. 3, and a Bender's decomposition algorithm in Sect. 4, the present paper proposes a theoretical study of the minimum cost dominating tree problem in the next section.

2 General Properties on the Minimum Cost Dominating Tree Problem

2.1 Theoretical Results

Definitions and notations
Let $G = (V, E)$ be an undirected, simple, edge weighted graph. The open neighborhood of $v \in V$ is denoted by $N_G(v)$, it is the set of all vertices adjacent to v. The closed neighborhood of v is defined by $N_G[v] = N_G(v) \cup \{v\}$.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 3–26, 2018.
https://doi.org/10.1007/978-3-319-72344-0_1

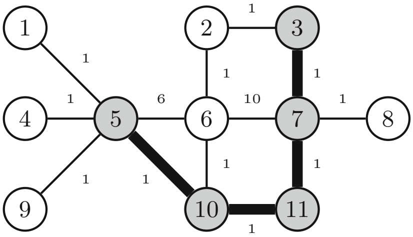


Fig. 1. Example of a dominating tree

A dominating tree $T = (V_T, E_T)$ of G is defined by a set of vertices $V_T \subset V$ and a set of edges $E_T \subset E$ such that

- Each vertex in V is either in V_T, or is adjacent to at least one vertex in V_T,
- E_T is a minimum spanning tree of the subgraph of G induced by V_T.

We define the following notations, for all $S \subseteq V$

- $\delta(S)$ is the set of all the edges that have exactly one endpoint in S
- $E(S)$ is the set of all the edges that have both their endpoints in S
- $G[S]$ is the subgraph of G induced by S, *i.e.*, $G[S] = (S, E(S))$
- $N_G[S]$ is the closed neighborhood of S, *i.e.*, all the vertices that are in S or are adjacent to a vertex in S. We have $S \subset N_G[S]$
- $N_G(S)$ is the open neighborhood of S, *i.e.*, all the vertices that are adjacent to a vertex in S but are not in S. Consequently $S \cap N_G(S)$ is empty, and $N_G[S] = S \cup N_G(S)$.

Lemma 1. *If $\ell \in V$ is a leaf of G that is adjacent to v, then $v \in V_T$ for all dominating tree $T = (V_T, E_T)$ of G.*

Proof. If a dominating tree includes edge (ℓ, v), then the property holds. If a dominating tree does not include edge (ℓ, v), then ℓ must be adjacent to a vertex in V_T. Vertex v is the only one that satisfies this property, hence v has to be in V_T.

As an illustration of Lemma 1, it can be observed in Fig. 1 that vertex 1 is a leaf connected to vertex 5. Consequently, vertex 5 has to be part of any dominating tree.

Lemma 2. *Let $e_i = (u_1, u_2)$ be an isthmus of G that partitions V in V_1 and V_2 with $u_1 \in V_1$ and $u_2 \in V_2$, where both $E(V_1)$ and $E(V_1)$ are nonempty. The following two implications hold:*

- *Edge e_i is part of any dominating tree of G*
- *A minimum cost dominating tree of G can be found by the following three-step procedure:*
  1. *Find a minimum dominating tree on the graph induced by $V_1 \cup \{u_2\}$*
  2. *Find a minimum dominating tree on the graph induced by $V_2 \cup \{u_1\}$*
  3. *Add edge e_i to the edges found in the previous steps*

Proof. Let e_i be an isthmus that satisfies the conditions of the lemma. By hypothesis, there exists $v_1 \in N_G(u_1)\backslash\{u_2\}$ such that $N_G[v_1] \subset V_1$, and $v_2 \in N_G(u_2)\backslash\{u_1\}$ such that $N_G[v_2] \subset V_2$. As e_i is the only edge for connecting $N_G[v_1]$ and $N_G[v_2]$, it has to be part of any dominating tree of G.

Second, we show that the three-step procedure of the lemma leads to a feasible dominating tree of G. Since u_2 is a leaf of $G[V_1 \cup \{u_2\}]$, u_1 is part of the dominating tree of $G[V_1 \cup \{u_2\}]$ by Lemma 1. Similarly, u_2 is part of the dominating tree of $G[V_2 \cup \{u_1\}]$. As edge e_1 is added to the two dominating trees, the result is a feasible dominating tree of G.

Now, we show by contradiction that the cost of the dominating tree returned by the three-step procedure of the lemma is minimum. We assume that there exists a minimum cost dominating tree of G, denoted by $T' = (V'_T, E'_T)$, whose cost is strictly less than the cost of the tree returned by the three-step procedure. Then, $T'[V'_T \cap (V_1 \cup \{u_2\})]$ (or $T'[V'_T \cap (V_2 \cup \{u_1\})]$) must be a dominating tree whose cost is strictly less than the cost of the minimum cost dominating tree of $G[V_1 \cup \{u_2\}]$ (or of $G[V_2 \cup \{u_1\}]$), which is a contradiction.

Lemma 2 can be used to decompose the problem of finding a minimum cost dominating tree on a graph into a collection of independent minimum cost dominating tree problems on isthmus-free subgraphs, as illustrated with the graph G of Fig. 2. Edge $(4,5)$ is an isthmus of G, so $V = \{1,\ldots,8\}$ can be partitioned into $V_1 = \{1,\ldots,4\}$ and $V_2 = \{5,\ldots,8\}$, so finding the minimum cost dominating tree of G can be achieved by finding the minimum dominating tree of $G_1 = G[V_1 \cup \{5\}]$ and of $G_2 = G[V_2 \cup \{4\}]$. These graphs are shown in Fig. 3.

By Lemma 1, vertices 1 and 5 are leaves of G_1, vertices 3 and 4 must be part of any dominating tree of G_1. It can also be deduced that vertex 5 is part of any dominating tree of G_2, and by enumeration, the optimal dominating tree of G_2 is to select edge $(5,6)$. Then, the minimum cost dominating tree of G is made of the three following edges $(3,4)$, $(5,6)$, and $(4,5)$, its cost is $1+4+2=7$.

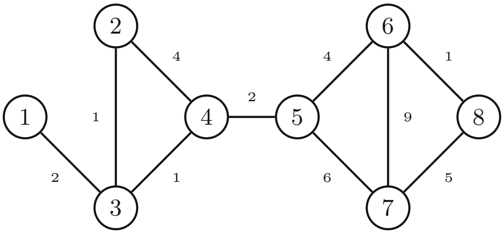


Fig. 2. Example of a dominating tree on a graph with an isthmus

Theorem 1. *Let $G = (V,E)$ be a simple, undirected graph. Finding a minimum cost dominating tree of G is an $\mathcal{NP}$-complete problem.*

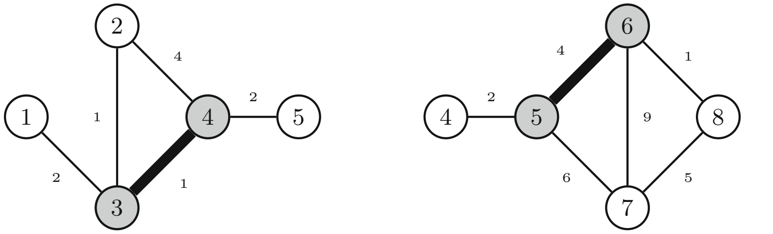


Fig. 3. The minimum cost dominating tree of graphs G_1 (left) and G_2 (right)

Proof. Let $G' = (V', E')$ the simple, undirected graph built from G as follows:

- For all $v \in V$, we create a new vertex $u_v \in T_e$, so $V' = V \cup T_e$, hence $|V'| = 2|V|$
- For all $e = (v, w) \in E$, we create two new edges in E' both of them with a zero cost: (u_v, w) and (v, u_w), so $E' = E' \cup E$, hence $|E'| = 3|E|$

It can be observed that any Steiner tree of G' (whose terminal nodes are those in T_e) is associated a dominating tree of G of identical cost, by removing all the edges in $\delta(V)$. The vertices in V that are part of the dominating tree are the Steiner nodes. Conversely, any dominating of G with at least one edge is associated a Steiner tree of G' (whose terminal nodes are those in T_e) of identical cost, by adding n edges in $\delta(V)$ as follows. For all $v \in V$, there exists at least one vertex adjacent to v in the dominating tree. So one edge joining such a vertex to u_v is added for all $v \in V$, which results in a Steiner tree of G' whose cost is the same as the cost of the dominating tree of G since the n new edges have a zero cost.

Hence, fining a minimum cost dominating tree of G with at least one edge is equivalent to finding the minimum cost Steiner tree of G', where the terminal nodes are those in T_e. As the Steiner tree problem is $\mathcal{NP}$-complete [4], then so is the minimum cost dominating tree problem.

This suggests that the minimum cost dominating tree problem can be solved by using approximation algorithms for the Steiner tree problem. The authors in [6] have exploited that idea. Computational results show that customized approaches yield better results.

Lemma 3. *There exists a zero-cost dominating tree if and only if there exists a degree $n-1$ vertex in V.*

Proof. A zero-cost dominating tree is a single vertex dominating tree, with no edge. If there exists a degree $n-1$ vertex $v \in V$, then v is adjacent to all the vertices in $V \backslash \{v\}$, so $V_T = \{v\}$ and $E_T = \emptyset$ is a zero-cost dominating tree of G. Conversely, if there exists a zero-cost dominating tree on G, then there exists a vertex $v \in V$ that is adjacent to all the vertices in $V \backslash \{v\}$, hence $deg(v) = n-1$.

Lemma 4. *Let G be a connected graph on at most three vertices. Any minimum cost dominating tree of G has at most $n-3$ edges.*

Proof. First, it should be observed that if $n \leq 2$, G is either a single vertex graph or a complete graph on two vertices (because G should be connected). In both cases there exists a zero-cost dominating tree. Now we assume that $n \geq 3$. Any spanning tree of G has $n-1$ edges and at least two leaves. If the edge incident to each leaf of the spanning tree is removed, the result is a tree on at most $n-3$ vertices, and each leaf of the original spanning tree is adjacent to at least one vertex in that tree. Hence that tree is a dominating tree of G.

Lemma 5. *Let $G = (V, E)$ be a simple, undirected graph. Let $S \subset V$, with $S \neq V$ and $S \neq \emptyset$. The vertices that are incident to at least one edge in $\delta(S)$ are denoted by $V_{\delta(S)}$.*

If $S \not\subset V_{\delta(S)}$ and $V \backslash S \not\subset V_{\delta(S)}$, then at least one edge in $\delta(S)$ is part of any dominating tree of G.

Proof. Assume that $S \not\subset V_{\delta(S)}$ and $V \backslash S \not\subset V_{\delta(S)}$. Then there exists $v_1 \in S \backslash V_{\delta(S)}$, and at least one of its neighboring vertex must be in the dominating tree (this vertex is in S). Identically, there exists $v_2 \in (V \backslash S) \backslash V_{\delta(S)}$ and at least one of its neighboring vertex must be in the dominating tree (this vertex is in $V \backslash S$). Hence, at least one edge in $\delta(S)$ must be part of the dominating tree for the vertices in $E_T \cap S$ and $E_T \cap V \backslash S$ to be connected.

For example in the graph of Fig. 1, let $S = \{1, 4, 5, 9\}$. This implies that $\delta(S) = \{(5, 6), (5, 10)\}$, hence $V_{\delta(S)} = \{5, 6, 10\}$. Since $S \not\subset V_{\delta(S)}$ and $V \backslash S \not\subset V_{\delta(S)}$, then we have that at least one of the edges of $\delta(S)$ has to be part of any dominating tree of G.

An articulation point is a vertex $a \in V$ such that removing that vertex from G creates $\kappa \geq 2$ connected components denoted by S^k for all $k \in \{1, \ldots, \kappa\}$.

Theorem 2. *If there exists an articulation point $a \in V$, then the problem of finding a minimum dominating tree of G can be split up into κ independent minimum dominating tree problems.*

Proof. First it can be observed that a is part of any dominating tree of G, for connectivity reasons. For all $k \in \{1, \ldots, \kappa\}$, S^k is a connected component of $G[V \backslash \{a\}]$. Naturally, $E(S^k) \cap E(S^{k'})$ is empty for all $k \neq k'$ in $\{1, \ldots, \kappa\}$.

For all $k \in \{1, \ldots, \kappa\}$, problem P_k is to find a minimum cost dominating tree on $G[S^k \cup \{a\} \cup \{a'\}]$, where a' is a new dummy vertex ($a' \notin V$) connected to a only with a new dummy edge having any strictly positive cost (a' is a leaf). Doing so implies that a is part of the minimum cost dominating tree in problem P^k for all k, whereas edge (a, a') is never selected.

Then, performing the union of all the edges used in the solution of problem P^k for all k yields a feasible dominating tree of G. This solution can be proved optimal by contradiction: the existence of a dominating tree of G having a strictly lower cost that this one would imply that the solution to problem P^k is not optimal for some $k \in \{1, \ldots, \kappa\}$, which is a contradiction.

A straightforward application of this theorem is to observe that any vertex that is adjacent to a leaf is an articulation point of G (for example, vertices

5 and 7 in the graph of Fig. 1). Hence, these vertices should be part of any dominating tree, which is exactly what is stated in Lemma 1. Theorem 2 is an extension of Lemma 1, as it allows to prove that all the non-leaves in a line graph must be part of any dominating tree. In the graph of Fig. 2, Theorem 2 allows to show that vertices 3, 4 and 5 are also part of any dominating tree, since they are articulation points.

Lemma 6. *Let $E^1 \subset E$ be the set of all edges e such that e is a dominating tree of G. No minimum cost dominating tree of G having at least two edges includes an edge from E^1.*

Proof. This lemma can easily be proven as follows. Let us assume that E^1 is not empty, and let $e \in E^1$ be a single-edge dominating tree. Since all edges have a strictly positive cost, there does not exist any dominating tree having two or more edges that contain e, since the corresponding cost would necessarily be strictly larger than w_e.

2.2 Classes of Graphs for Which the Minimum Cost Dominating Tree Problem Is Polynomial

Lemmas 7 and 9 show that the minimum cost dominating tree can be solved to optimality in polynomial time if G is a tree, or a cycle graph. Lemmas 8 and 10 generalize Lemmas 7 and 9 to a more constrained dominating tree problem version, for which a given subset of vertices has to be part of any dominating tree. These extensions, as well as Lemma 11 on branch vertices, are useful for showing that the minimum cost dominating tree problem can be solved to optimality in polynomial time if G is a cactus, *i.e.*, a connected graph in which any pair of cycles has not more than one vertex in common in Theorem 3.

Lemma 7. *Let $G = (V, E)$ be a tree. The set of its leaves is denoted by $L \subset V$. The minimum cost dominating tree of G, denoted by $T = (V_T, E_T)$ is defined by*

- $V_T = V \backslash L$
- $E_T = (u, v) \in E, (u, v) \in V_T \times V_T$

Proof. Since G is a tree, any dominating tree of G is also a tree. By Lemma 5 any edge joining two non-leaves of G must be part of any dominating tree. Hence, the dominating tree resulting from the removal of all edges incident to a leaf in G is the unique minimum cost dominating tree of G. Thus, building the minimum cost dominating tree of G requires computing the degree of all vertices, and removing all leaves. The complexity of this algorithm is $\mathcal{O}(n^2)$ operations.

Lemma 8. *Let $G = (V, E)$ be a tree, and let $V^+ \subseteq V$ be a given subset of V such that all the vertices in V^+ have to be part of any dominating tree. In that case the minimum cost dominating tree of G, denoted by $T = (V_T, E_T)$ is defined by*

- $V_T = (V \backslash L) \cup V^+$
- $E_T = (u, v) \in E, (u, v) \in V_T \times V_T$

Proof. Any non-leaf of G has to be part of the dominating tree, so if $V^+ \cap L$ is empty, the dominating tree is the same as without the constraint on V^+. If $V^+ \cap L$ is non-empty, then these vertices (that are leaves of G) and their incident edge must be added to the minimum cost-dominating tree. Consequently, building the dominating tree of G requires computing the degree of all vertices, and removing the leaves which are not in V^+. The complexity of this algorithm is $\mathcal{O}(n^2)$ operations.

Lemma 9. *Let $G = (V, E)$ be a cycle graph, with $E = \{e_0, e_1, \dots, e_{n-1}\}$ and where e_i and $e_{(i+1) \bmod n}$ are adjacent for all $i \in \{0, \dots, n-1\}$.*

Let j be the integer defined by

$$e_j = \underset{i \in \{0,\dots,n-1\}}{\operatorname{argmax}} \left(w_{e_{(i-1) \bmod n}} + w_{e_i} + w_{e_{(i+1) \bmod n}} \right)$$

The minimum cost dominating tree of G, denoted by $T = (V_T, E_T)$ is defined by

- $V_T = V \backslash \{u, v\}$ *where* $(u, v) = e_j$
- $E_T = E \backslash \{e_{(j-1) \bmod n}, e_j, e_{(j+1) \bmod n}\}$

Proof. Since G is a cycle graph, any dominating tree of G is a path. Consequently, the edges in $E \backslash E_T$ also form a path. Thus, if the path formed by $E \backslash E_T$ is made of four edges or more, then there exists a vertex which is neither in V_T, nor is adjacent to any vertex in V_T. Consequently, the minimum cost dominating tree has $n - 3$ edges.

More precisely, there exist n dominating trees on G, obtained by removing a triplet of 'consecutive' edges of the form $\{e_{(i-1) \bmod n}, e_i, e_{(i+1) \bmod n}\}$. The minimum cost dominating tree is the one resulting from the removal of the costliest triplet of edges. This triplet can be found by enumeration, which requires $\mathcal{O}(n)$ operations.

As an illustration of Lemma 9, consider the cycle graph on $n = 5$ vertices shown in Fig. 4. Each edge appear with its name and its cost (*e.g.*, edge $(1, 2)$ is referred to as e_0, and its cost is 3).

Finding a minimum cost dominating tree on the cycle graph of Fig. 4 is simply enumerating the following five feasible solutions. The first one considers the dominating tree made of edges e_0 and e_1, whose cost is 10. The second one is made of edges e_1 and e_2, with cost 8, and the last solution is made of edges e_4 and e_0, with cost 6. In the present case, the last solution is the optimal one, since its cost is minimum.

Lemma 10. *Let $G = (V, E)$ be a cycle graph, with $E = \{e_0, e_1, \dots, e_{n-1}\}$ and where e_i and $e_{(i+1) \bmod n}$ are adjacent for all $i \in \{0, \dots, n-1\}$. Let $V^+ \subset V$ be a given subset of V, such that all the vertices in V^+ have to be part of any dominating tree. A dominating tree of G has either $n-1$, $n-2$ or $n-3$ edges:*

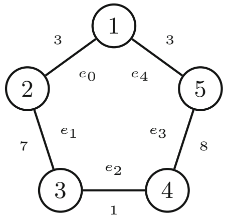


Fig. 4. A cycle graph

with n edges, the dominating tree would have a cycle, and it cannot have $n-4$ edges or less as shown in the proof of Lemma 9.

We show the four following properties:

- If T is a minimum cost dominating tree of G having $n-1$ edges, then there exists $i \in \{0, \dots, n-1\}$ such that $E_T = E \backslash \{e_i\}$ where $e_i \in E(V^+)$.

Proof. Assume that T is a $n-1$ edges minimum cost dominating tree with $E_T = E \backslash \{e_i\}$, but where $e_i \notin E(V^+)$. Then e_i has at least one endpoint in $V \backslash V^+$. Suppose that $e_i = (u, v)$ with $u \in V \backslash V^+$. Then, there exists a unique edge $e_{i'} = (w, u) \in E$ that can be removed from T, leading to a feasible dominating tree T' with $u \notin V_{T'}$, but whose cost is strictly less than the cost of T since all edges have a strictly positive cost. This is a contradiction.

- If T is a minimum cost dominating tree of G having $n-3$ edges, then there exists $i \in \{0, \dots, n-1\}$ such that $E_T = E \backslash \{e_{(i-1) \bmod n}, e_i, e_{(i+1) \bmod n}\}$ where $e_i \in (V \backslash V^+)$.

Proof. Assume that T is a $n-3$ edges minimum cost dominating tree such that there does not exist $i \in \{0, \dots, n-1\}$ such that $E_T = E \backslash \{e_{(i-1) \bmod n}, e_i, e_{(i+1) \bmod n}\}$ with $e_i \in E(V \backslash V^+)$. Then, $e_i = (u, v)$ has at least one endpoint in V^+ (say u). In that case, removing $e_{(i-1) \bmod n}$, e_i and $e_{(i+1) \bmod n}$ from E_T leads to isolate vertex u from the other vertices in V_T, hence V_T is not connected. This is a contradiction.

- If T is a minimum cost dominating tree of G having $n-2$ edges, then $E_T = E \backslash \{e_i, e_{(i+1) \bmod n}\}$ where e_i and $e_{(i+1) \bmod n}$ are adjacent to the same vertex $u \in V \backslash V^+$ and such that they both have one endpoint in V^+ (*i.e.*, they both belong to $\delta(V^+)$).

Proof. First, e_i and $e_{(i+1) \bmod n}$ must be adjacent, otherwise $E \backslash \{e_i, e_{(i+1) \bmod n}\}$ is not connected. These two edges cannot be incident to a vertex in $u \in V^+$ because their removal would lead to isolate u from the rest of the cycle. Now, assume that e_i and $e_{(i+1) \bmod n}$ are adjacent to the same vertex $u \in V \backslash V^+$, but that one of these edges is also incident to $v \in V \backslash V^+$ (with $v \neq u$). Then there exists a unique edge $e_{i'} \in E$ incident to v and different from e_i and $e_{(i+1) \bmod n}$, that can be removed from T, leading to a valid dominating tree whose cost is strictly less than the cost of T. This is a contradiction.

– The total number of candidate minimum cost dominating trees of G is at most n.

Proof. Each edge in $E(V^+)$ is associated to a candidate minimum cost dominating tree having $n-1$ edges, and each edge in $E(V \backslash V^+)$ is associated to a candidate minimum cost dominating tree having $n-3$ edges. Each couple of adjacent edges in $\delta(V^+)$ is associated to a candidate minimum cost dominating tree having $n-2$ edges. Note that two such couples of adjacent edges cannot overlap as their common vertex is in $V \backslash V^+$: if they would overlap, one edge in $\delta(V^+)$ would have to be in $E(V \backslash V^+)$. Consequently, there exist at most $\frac{|\delta(V^+)|}{2}$ candidate minimum cost dominating trees having $n-2$ edges. Since $|E(V^+)|+|E(V \backslash V^+)|+|\delta(V^+)| = n$, we have $|E(V^+)|+|E(V \backslash V^+)|+|\frac{\delta(V^+)}{2}| \leq n$, so the number of candidate minimum cost dominating trees is at most n (it is equal to n only if $V^+ = V$ or if $V^+ = \emptyset$). Consequently, the minimum cost dominating tree of G can be computed in polynomial time as the maximum number of candidate dominating trees to consider is at most n.

Note that Lemma 9 is a particular case of Lemma 10, where $V^+ = \emptyset$. In that case, the minimum dominating tree has always $n-3$ edges.

As an illustration of Lemma 10, we consider again the cycle graph of Fig. 4, and assume that $V^+ = \{3\}$. Finding an optimal solution to the constrained minimum dominating tree in that case is enumerating $n = 5$ feasible solutions, based on an edge that should be absent from the solution, as follows. We consider that edge e_0 should be absent, and try to remove its left and right neighbors. Both of them can be removed, hence the first solution found contains $\{e_2, e_3\}$ and has cost 9. The second solution is based on the fact that e_1 should be absent, but since vertex 3 should be in the solution, edge e_2 cannot be removed, hence the second solution is the same as before. The third and fourth solutions are identical: $\{e_0, e_1\}$ with cost 10, and the fifth on is $\{e_1, e_2\}$ with cost 8. It is the optimal one, as its cost is minimum (Fig. 4).

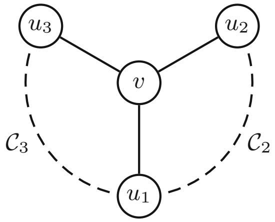


Fig. 5. Illustration for the proof of Lemma 11

Lemma 11. *Let $G = (V, E)$ be a connected, undirected graph such that any pair of cycles has at most one vertex in common. All branch vertices of G are part of any dominating tree, where a branch vertex is a vertex whose degree is at least three.*

Proof. Let $v \in V$ be a branch vertex of G such that $v \notin V_T$. Vertex v is adjacent to at least three vertices u_1, u_2 and u_3, so at least one of them must be part the dominating tree, say u_1. As u_2 must be adjacent to a vertex in the dominating tree, there must exists a path in T from u_1 to a neighboring vertex of u_2. Since $v \notin V_T$, this implies that u_1 and u_2 are part of a cycle in G, denoted by $\mathcal{C}_2$. Note that in this cycle, edges (u_1, v) and (v, u_2) are adjacent. However, u_3 must also be adjacent to a vertex in the dominating tree, so by the same argument, we are led to the conclusion that u_1 and u_3 are also part of a cycle denoted by $\mathcal{C}_3$, that is such that edges (u_1, v) and (v, u_3) are adjacent. There cannot be three edges incident to the same vertex in a cycle, so $\mathcal{C}_2$ and $\mathcal{C}_3$ must be two distinct cycles. But in that case, $\mathcal{C}_2$ and $\mathcal{C}_3$ have one edge in common, and so they have two vertices in common (u_1 and v) which is a contradiction.

Theorem 3. *Let $G = (V, E)$ be a cactus. A minimum cost dominating tree of G can be computed in polynomial time.*

Proof. Let $E_C \subseteq E$ be the set of all edges that are part of a cycle (E_C is supposed to be given), $V_C \subseteq V$ be the endpoints of the edges in E_C, and V_B be the set of branch vertices of G. By Lemma 11, $V_B \subseteq V_T$. A minimum cost dominating tree of G can be found as follows:

Step one: Any edge $e \in E \backslash E_C$ that is not incident to a leaf of G is part of any dominating tree of G. Indeed, if such an edge is not in E_T, then the dominating tree is not connected by Lemma 8.

Step two: The vertices that are part of every cycle in G are connected to the dominating tree by solving the constrained dominating tree problem on each cycle.

The result is a minimum dominating tree of G, because the edges added at step one are mandatory for connectivity reasons, and because the edges added at step two allow for connecting the vertices in V_C to the rest of the tree at minimum cost by Lemma 10.

Finally, finding all branch vertices in G requires $\mathcal{O}(n^2)$ operations for computing the degree of all vertices, step one requires $\mathcal{O}(n^2)$ operations for checking the degree of both the endpoints of each edge in $E \backslash E_C$, and step two consists in solving the constrained dominating tree problem on at most $\lfloor \frac{n}{2} \rfloor$ cycles, each of them requiring at most n operations. Consequently, the computational complexity of this algorithm is $\mathcal{O}(n^2)$.

As an example, we consider the cactus of Fig. 6, left. Its branch vertices (namely vertices 2, 7 and 10) are shown in gray, as by Lemma 11, they are all part of any dominating tree for connectivity reasons. We now consider the cycles one by one, and solve the constrained minimum cost dominating tree on them, the constraints being induced by the branch vertices. The first cycle is $(1, 2, 7, 11, 6)$, where $V^+ = \{2, 7\}$. The optimal solution is to select edges $\{(1, 2), (2, 7)\}$. The second cycle is $(7, 8, 4, 8, 10, 15, 14, 13)$, with $V^+ = \{7, 10\}$. The optimal solution is to select edges $\{(7, 13), (7, 8), (4, 8), (4, 5), (5, 10)\}$. Finally, the optimal solution of the minimum cost dominating tree of that cactus is shown in Fig. 6, right, and its cost is 22.

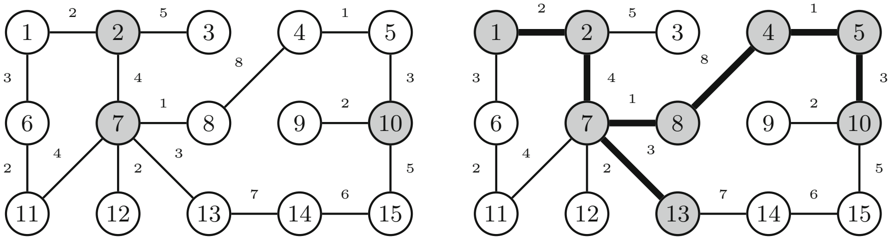


Fig. 6. A cactus graph (left) and a minimum cost dominating tree on it (right)

3 Two ILP-Based Exact Approaches for the Minimum Cost Dominating Tree Problem

3.1 A Flow-Formulation for the Minimum Cost Dominating Tree Problem

In this section, a flow-formulation is proposed for the minimum cost dominating tree problem. A dummy vertex 0 is also added to the graph, it corresponds to the source of the flow that should reach each vertex. The total flow is equal to the number of vertices in the minimum cost dominating tree, and each such vertex is assumed to consume exactly one unit of flow, for enforcing that each vertex is adjacent to a vertex in the dominating tree. There are five sets of decision variables:

- x_e is a binary variable that is set to one if and only if edge $e \in E$ is part of the minimum cost dominating tree
- y_v is a binary variable that is set to one if and only if vertex $v \in V$ is part of the dominating tree
- The continuous decision variables f_{uv} and f_{vu} represent the flow in the edge $(u, v) \in E$, from u to v, and from v to u, respectively
- The continuous decision variables f_{0v} are the amount of flow sent from the dummy vertex 0 to the vertices $v \in N_G[1]$, with the restriction that vertex 0 sends flow to a single vertex
- The binary variables z_v are used for that purpose: z_v is set to one if ans only if vertex $u \in N_G(1)$ receives all the flow from vertex 0.

The corresponding ILP formulation is as follows:

$$
\begin{aligned}
&\text{Minimize } \sum_{e\in E} w_e x_e \\
&\sum_{u\in N_G[v]} y_u \geq 1 \quad \forall v \in V \\
&\sum_{e\in E} x_e = \sum_{v\in V} y_v - 1 \\
&x_e \leq y_u \quad \forall e=(u,v)\in E \\
&x_e \leq y_v \quad \forall e=(u,v)\in E \\
&\sum_{u\in N_G(v)} f_{uv} + f_{0v} = y_v + \sum_{u\in N_G(v)} f_{vu} \quad \forall v \in N_G(1) \\
&\sum_{u\in N_G(v)} f_{uv} = y_v + \sum_{u\in N_G(v)} f_{vu} \quad \forall v \in V \backslash N_G(1) \\
&f_{uv} + f_{vu} \leq (n-3)x_e \quad \forall e=(u,v)\in E \\
&\sum_{u\in N_G[1]} f_{0u} = \sum_{u\in V} y_u \\
&f_{0u} \leq (n-2) z_v \quad \forall v \in N_G[1] \\
&\sum_{u\in N_G[1]} z_u = 1 \\
&f_{uv} \geq 0 \quad \forall (u,v)\in E \\
&x_e \in \{0,1\} \quad \forall e \in E \\
&y_v \in \{0,1\} \quad \forall v \in V \\
&z_v \in \{0,1\} \quad \forall v \in N_G[1]
\end{aligned}
$$

The first constraint enforces that a dominating set is built. The second constraint is a necessary condition for the solution to be a tree. The third and fourth ones ensure that an edge cannot be selected if one of its endpoint is not part of the dominating tree. The fifth and sixth ones are the classical flow balancing constraint, that ensure that each vertex of the dominating tree (except 0) consumes exactly one unit of flow. The seventh constraint enforces that an edge is selected if the flow in that edge is strictly positive. The eighth constraint states that the total amount of flow sent by the dummy source vertex is equal to the number of vertices in the minimum cost dominating tree. The ninth and tenth constraints enforce that the dummy vertex sends all its flow to a unique vertex in $N_G[1]$.

As an illustration, Fig. 7 shows the value of the f_{uv} and f_{0v} variables on the optimal solution of the example given in Fig. 1 (dashed edges indicate that there is no flow in the corresponding edge). Furthermore, this model is likely to have a poor linear programming relaxation in general because of numerous *big M* constraints, like the ones that link x_e and z_v to f_{uv} and f_{0v} variables. For example, the optimal objective value of the example given in Fig. 1 is 4, whereas the linear programming objective value is only 2.5. Finally, this model has quite a large number of integer variables, which makes it not very efficient in practice.

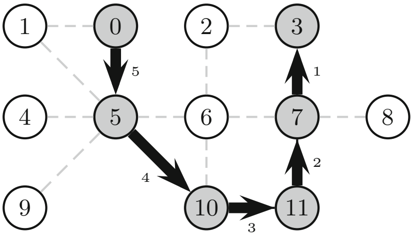


Fig. 7. Illustration of the flow variables on the optimal solution

3.2 A Cutting Plane Algorithm for the Minimum Cost Dominating Tree Problem

We assume that G has no vertex with degree $n-1$ (*i.e.*, there does not exists a single vertex dominating tree, by Lemma 3). Let $E^* \subset E$ be the set of all edges that are such that for all $e \in E^*$, any vertex in V is adjacent to at least one endpoint of e. Then, w_e is an upper bound on the cost of the minimum cost dominating tree of G. Moreover, no minimum cost dominating tree with at least two edges can include e, as such a dominating tree would have a cost strictly greater that w_e, since edge costs are all strictly positive.

Lemma 12. *If $G=(V,E)$ has no vertex with degree $n-1$, but has a minimum cost edge $e=(u,v) \in E$ such that $N_G(u) \cup N_G(v) = V$, then $T=(V_T, E_T)$ with $V_T = \{u,v\}$ and $E_T = \{e\}$ is a minimum cost dominating tree of G.*

Proof. Since there does not exist any vertex with degree $n-1$, then there is no zero-cost dominating tree, hence E_T has at least one edge. If all the vertices are adjacent to at least one endpoint of a minimum cost edge, then this edge is a minimum cost dominating tree.

Algorithm 1 shows an overview of the proposed solution process for finding the minimum cost dominating tree of a simple, connected, strictly positive edge-weighted graph $G=(V,E)$. We first search for a single vertex dominating tree, if no such vertex exists we turn to the enumeration of all single edge dominating trees. If no such dominating tree exists, or if the optimality of the minimum cost single edge dominating tree cannot be proven, then we address the problem of finding a minimum cost dominating tree with at least two edges. An integer linear programming formulation [8] is provided for this problem in the next subsection.

It can be observed that the search for a single vertex dominating tree and the enumeration of all single edge dominating trees requires $\mathcal{O}(mn^2)$ operations: in the worst case, we have to compute m times the union of $N_G(u)$ and $N_G(v)$ that can have up to $n-2$ elements each.

```
input  : $G = (V, E)$
output: $V_T, E_T$
// Initialization
$UB \leftarrow +\infty$;
$deg(v) \leftarrow 0 \quad \forall v \in V$;
$N_G(v) \leftarrow \emptyset \quad \forall v \in V$;
for $e = 1$ to $m$ do
    // $u$ and $v$ are the endpoints of $e$
    $deg(u) \leftarrow deg(u) + 1$;
    $deg(v) \leftarrow deg(v) + 1$;
    $N_G(u) \leftarrow N_G(u) \cup \{v\}$;
    $N_G(v) \leftarrow N_G(v) \cup \{u\}$;
end
// Searching for a single vertex dominating tree
$v \leftarrow 1$;
while $v \leq n$ do
    if $deg(v) = n - 1$ then
        $V_T \leftarrow \{v\}$;
        $E_T \leftarrow \emptyset$;
        return $V_T, E_T$;
    end
    $v \leftarrow v + 1$ ;
end
// Searching for a single edge dominating tree
$E^* \leftarrow \emptyset$;
for $e = 1$ to $m$ do
    // $u$ and $v$ are the endpoints of $e$
    if $|N_G(u) \cup N_G(v)| = n$ then
        $E^* \leftarrow E^* \cup \{e\}$;
    end
end
$UB \leftarrow \min_{e \in E^*} w_e$;
if $UB = \min_{e \in E} w_e$ then
    $V_T \leftarrow \{u, v\}$;
    $E_T \leftarrow \{e\}$;
    return $V_T, E_T$;
end
// Searching for a minimum cost dominating tree with at least two
   edges
Solve the $ILP$;
$E_T \leftarrow \{e \in E | x_e = 1\}$;
$V_T \leftarrow \{v \in V |$there exists $e \in E$ with $x_e = 1$, such that $v$ is incident to $e\}$;
return $V_T, E_T$;
```

Algorithm 1. Overview of the solution process

3.3 Second ILP Formulation to the Dominating Tree Problem with at Least Two Edges

For all edge $e \in E$, the Boolean variable x_e is set to one if and only if edge $e \in E_T$. No variable is needed for determining the vertices in V_T, as these vertices can be determined from x_e as shown in Algorithm 1.

By definition, any vertex v is either in V_T or is adjacent to a vertex in V_T, which implies that any dominating tree has at least one edge incident to $N_G(v)$. This requirement can be stated as

$$\sum_{e \in E(N_G(v)) \cup \delta(N_G(v))} x_e \geq 1 \quad \forall v \in V \tag{1}$$

If $N_G(v)$ is not adjacent to all the vertices in V, then there exists $u \in V$ such that $N_G(u) \cap N_G(v)$ is empty. Consequently, any dominating tree should have at least one edge in $\delta(N_G(v))$ for connecting $N_G(v)$ and $N_G(u)$. As $\delta(N_G(v)) \subset E(N_G(v)) \cup \delta(N_G(v))$, Eq. (1) can then be strengthened to

$$\sum_{e \in \delta(N_G(v))} x_e \geq 1 \quad \forall v \in V \tag{2}$$

In the sequel, the edge set D_v is defined by $D_v = E(N_G(v)) \cup \delta(N_G(v))$ if $N_G(v)$ is adjacent to all the vertices in V, and by $D_v = \delta(N_G(v))$ otherwise. Thus, Eqs. (1) and (2) can be generalized to

$$\sum_{e \in D_v} x_e \geq 1 \quad \forall v \in V \tag{3}$$

Definition. Let Ω be the set of all $S \subset V$ that satisfy the following conditions:

- $\exists v \in S$ such that $N_G[v] \subset S$
- $E(V \backslash S)$ is non empty

Lemma 13. *Let E_T be any collection of edges in E with $|E_T| \geq 2$, and let V_T be the endpoints of these edges. We assume that E_T is such that every vertex in V is either in V_T or is adjacent to V_T.*

E_T is connected or is a collection of dominating sets of edges of G if and only if

$$x_{e_1} \leq \sum_{e \in \delta(S)} x_e \; \forall S \in \Omega \forall e_1 \in E(V \backslash S) \tag{4}$$

Proof. We use contraposition for proving both implications.

First, we show that if E_T is not connected and is not a collection of disjoint dominating sets of edges of G, then there exists $S \in \Omega$ and $e_1 \in E(V \backslash S)$ such that $x_{e_1} > \sum_{e \in \delta(S)} x_e$.

If E_T is not connected and is not a collection of disjoint dominating sets of edges of G, then $G[V_T]$ (the subgraph of G induced by V_T) can be partitioned into $\kappa \geq 2$ connected components denoted by $V_T^1, \ldots, V_T^\kappa$. Notice that $\delta(V_T^g) \cap E_T$ is empty for all $g \in \{1, \ldots, \kappa\}$, otherwise V_T^g would not be a connected component of $G[V_T]$. Then, E_T can also be partitioned as follows $E_T = \bigcup_{g=1}^{\kappa} E_T^g$, where $E_T^g \subset E(V_T^g)$. By hypothesis, there exists $i \in \{1, \ldots, \kappa\}$ such that E_T^i is not a dominating set of edges of G. This implies that if $S = V \backslash V_T^i$, then there exists $v \in S$ such that $N_G[v] \subset S$. Moreover, $V \backslash S = V_T^i$ is non-empty and so is $E(V \backslash S)$ because it contains E_T^i. Consequently, $S \in \Omega$, for any edge $e_1 \in E_T^i \subset E(V \backslash S)$ we have $x_{e_1} = 1$, and $\sum_{e \in \delta(S)} x_e = 0$ as $\delta(S) = \delta(V_T^i)$, and $\delta(V_T^i) \cap E_T$ is empty.

Second, we show that if there exists $S \in \Omega$ and $e_1 \in E(V \backslash S)$ such that $x_{e_1} > \sum_{e \in \delta(S)} x_e$, then E_T is not connected, and is not a collection of disjoint dominating sets of edges of G.

If there exists $S \in \Omega$ and $e_1 \in E(V \backslash S)$ such that $x_{e_1} > \sum_{e \in \delta(S)} x_e$, then the inequality implies that $e_1 \in E_T$, and that no edge $e \in \delta(S)$ is in E_T. In addition, since $S \in \Omega$, there exists $v \in S$ such that $N_G[v] \subset S$, and no edge in $\delta(S)$ is incident to v. This proves that the edges in $E_T \cap E(V \backslash S)$ are not a dominating set of edges of G, as none of these edges is incident to $N_G[v]$. Consequently, E_T is not a collection of disjoint dominating sets of edges of G. Since there is at least one edge of E_T (say e_2) in $(\delta(N_G(v)) \cup E(N_G(v))) \backslash \delta(S)$, there is no path for joining the endpoints of e_1 to the endpoints of e_2, so E_T is not connected.

The fact that E_T may be a collection of disjoint dominating sets of edges may sound like an inconvenient. However, if such a case arises, it can be immediately discarded, as any dominating tree in that set has a cost which is strictly less than the cost of E_T.

The problem of finding a minimum cost dominating tree, which is referred to as ILP_{DT} can be stated as follows

$$(ILP_{DT}) : \begin{cases} \text{Minimize } \sum_{e \in E} w_e x_e & \\ \sum_{e \in D_v} x_e \geq 1 & \forall v \in V \\ \sum_{f \in I_e} x_f \geq x_e & \forall e \in E \\ x_{e_1} \leq \sum_{e \in \delta(S)} x_e & \forall S \in \Omega, \forall e_1 \in E(V \backslash S) \\ x_e \in \{0,1\} & \forall e \in E \end{cases}$$

The objective function is to find a dominating tree of minimum cost. The first constraint enforces that any vertex is adjacent to at least one endpoint of the dominating tree. The second one states that if edge e is part of the solution, then, at least one other edge incident to it should also be selected, since we search for solutions having at least two edges. In this constraint, I_e is the set of all the edges that have one endpoint in common with edge e. The second constraint ensures that E_T (*i.e.*, the set of edges $e \in E$ such that $x_e = 1$) is connected, or is a collection of disjoint dominating sets of edges as shown in Lemma 13. The last set of constraints enforces integrality requirements.

Since the number of constraints (4) grows exponentially with the problem size, they cannot be enumerated. Hence a cutting-plane algorithm is used to address the problem of finding a minimum cost dominating tree having at least two edges.

First, we address a master problem MP defined by

$$(MP) : \begin{cases} \text{Minimize } \sum_{e \in E} w_e x_e & \\ \sum_{e \in D_v} x_e \geq 1 & \forall v \in V \\ \sum_{f \in I_e} x_f \geq x_e & \forall e \in E \\ x_e \in \{0, 1\} & \forall e \in E \end{cases}$$

It can be seen that MP is a relaxation of ILP_{DT}, where the constraints (4) have been dropped. As a result, MP is easier to solve (it has a linear number of constraints), but its solution may not be connected.

Valid inequalities are added to MP in an attempt to reduce the number of iterations of the cutting-plane algorithm. For all u and v in V such that u and v are distant of three or more edges, we add an inequality that states that there should be a chain from at least one vertex in $N_G[u]$ to at least one vertex of $N_G[v]$. This is done by computing the minimum cut between $N_G[u]$ and $N_G[v]$ in G, in terms of number of edges. In the graph of Fig. 1, the vertices 1 and 8 are at distance 4, hence such a valid inequality can be computed. The minimum cut is made of the edges $(5, 6)$ and $(5, 10)$, so the valid inequality is $x_{5,6} + x_{5,10} \geq 1$.

However, these valid inequalities do not prevent the optimal solution of MP to be disconnected in general. When this happens, we have to select at least one constraint in (4), to be added to MP. Then, MP is re-solved, and the current iteration completes. The process stops when the optimal solution of MP is connected.

For the sake of efficiency, Ω can be partitioned into two disjoint sets Ω_1 and Ω_2, where Ω_1 is the set of all $S \subset \Omega$ such that there is no $N_G[u] \subset V \backslash S$, whereas Ω_2 is the set of all $S \subset \Omega$ such that there exists $N_G[u] \subset V \backslash S$. More formally

- $\Omega_1 = \{S \in \Omega | \nexists u \in V \backslash S, N_G[u] \subset V \backslash S\}$
- $\Omega_2 = \{S \in \Omega | \exists u \in V \backslash S, N_G[u] \subset V \backslash S\}$

Lemma 14. *For all $S \in \Omega_2$, inequality (4) can be written as*

$$1 \leq \sum_{e \in \delta(S)} x_e \ \forall S \in \Omega_2 \tag{5}$$

In the case where the solution of ILP_{DT} is not connected, V_T can be partitioned into $\eta \geq 2$ connected components $C_1, \ldots, C_\eta$. If there exists $v \in C_k$ such that $(N_G[v] \cup C_k) \in \Omega_2$, a cut having the form of Eq. (5) can be generated. By definition of Ω_2, there exists $u \notin S$ such that $N_G[u] \cap S$ is empty. Then, solving the minimum cut problem on G (in terms of number of edges), where the source is v and the sink is u, while enforcing that all the vertices in $N_G[v]$ belong to the first partition of V and $N_G[v]$ belong to the second partition, we avoid dense cuts, that are well known to be less efficient [3].

4 Benders Formulation

It is recalled that we search for a solution having at least 2 edges (so the dominating tree has at least 3 vertices), because we have a preprocess that enumerate all no-edge solutions and all one-edge solutions. The starting point of the Benders decomposition is the first ILP model:

$$\begin{array}{lll}
\text{Minimize} \displaystyle\sum_{e \in E} w_e x_e & & \\
\displaystyle\sum_{e \in E} x_e = \sum_{v \in V} y_v - 1 & & \\
\displaystyle\sum_{e \in E(S)} x_e \leq |S| - 1 & \forall S \subseteq V \\
\displaystyle\sum_{e \in \delta(\{v\})} x_e \leq d_v y_v & \forall v \in V \\
x_e = 0 & \forall e \in E^1 \\
\displaystyle\sum_{u \in N_G(v)} y_u \geq 1 & \forall v \in V \\
\displaystyle\sum_{v \in V} y_v \geq 3 & \\
y_v = 1 & \forall v \in V^* \\
y_v \in \{0,1\} & \forall v \in V \\
x_e \in \{0,1\} & \forall e \in E
\end{array}$$

The objective function is to minimize the cost of the dominating tree, the first constraint states that the solution is a tree on the dominating vertices, then are the packing inequalities (or cycle breaking inequalities). An edge cannot be selected in the solution if one of its endpoints is not a dominating vertex, and single edge dominating trees are excluded from the search. The neighborhood of any vertex must contain at least one dominating vertex, those vertices that are adjacent to a leaf (if there exist some) must be part of the dominating tree. All variables are Boolean.

The master problem is then

$$\begin{aligned}
&\text{Minimize } q(y)\\
&\sum_{u\in N_G(v)} y_u \geq 1 \ \forall v \in V\\
&\sum_{v\in V} y_v \geq 3\\
&y_v = 1 \qquad \forall v \in V^*\\
&y_v \in \{0,1\} \qquad \forall v \in V
\end{aligned}$$

where $q(y)$ is the subproblem objective value. The subproblem is to compute the minimum spanning tree on the vertex set $V_T = \{v \in V : y_v = 1\}$. The subproblem is then

$$\begin{aligned}
&\text{Minimize } q(y) = \sum_{e\in E(V_T)} w_e x_e\\
&\sum_{e\in E(V_T)} x_e \geq \sum_{v\in V} y_v - 1\\
&\sum_{e\in E(S)} -x_e \geq -|S| + 1 \quad \forall S \subseteq V_T\\
&x_e = 0 \qquad \forall e \in E^1\\
&x_e \geq 0 \qquad \forall e \in E(V_T)
\end{aligned}$$

Whose dual is

$$\begin{aligned}
&\text{Maximize } q(y) = \left(\sum_{v\in V} y_v - 1\right)\pi_1 - \sum_{S\subseteq V_T} (|S|-1)\,\pi_S\\
&\pi_1 - \left(\sum_{S\subseteq V_T : e\in E(S)} \pi_S\right) \leq w_e \qquad \forall e \in E(V_T)\backslash E^1\\
&\pi_1 - \left(\sum_{S\subseteq V_T : e\in E(S)} \pi_S\right) - \mu_e \leq w_e \ \forall e \in E^1\\
&\pi_1 \geq 0\\
&\pi_S \geq 0 \qquad \forall S \subseteq V_T\\
&\mu_e \geq 0 \qquad \forall e \in E^1
\end{aligned}$$

The minimum spanning tree problem is such that its linear programming relaxation can be considered. Moreover, using the Dual Greedy Algorithm (based on Kruskal algorithm) [2], dual variables π_1, π_S for all $S \subset V_T$ and μ_e for all $e \in E^1$ can be computed without actually solving the LP formulation of the subproblem that would require the enumeration of packing inequalities.

The feasible set of the dual of the subproblem can be described using extreme rays and extreme points. Let I and J be the number of extreme points and extreme rays of the feasible set, respectively. Then, vector $\begin{bmatrix} \pi_1^j \\ \pi_S^j \end{bmatrix}$ is an extreme ray for all $j \in \{1, \dots, J\}$, and $\begin{bmatrix} \pi_1^i \\ \pi_S^i \end{bmatrix}$ is an extreme point for all $i \in \{1, \dots, I\}$.

$$\begin{aligned}
&\text{Minimize } q\\
&\left(\sum_{v \in V} y_v - 1\right) \pi_1^j - \left(\sum_{S \subseteq V_T} |S| - 1\right) \pi_S^j \leq 0 \; \forall j \in \{1, \dots, J\}\\
&\left(\sum_{v \in V} y_v - 1\right) \pi_1^i - \left(\sum_{S \subseteq V_T} |S| - 1\right) \pi_S^i \leq q \; \forall i \in \{1, \dots, I\}\\
&q \in \mathbb{R}
\end{aligned}$$

Hence, the minimum dominating tree problem can be formulated as

$$\begin{aligned}
&\text{Minimize } q\\
&\left(\sum_{v \in V} y_v - 1\right) \pi_1^j - \left(\sum_{S \subseteq V_T} |S| - 1\right) \pi_S^j \leq 0 \; \forall j \in \{1, \dots, J\}\\
&\left(\sum_{v \in V} y_v - 1\right) \pi_1^i - \left(\sum_{S \subseteq V_T} |S| - 1\right) \pi_S^i \leq q \; \forall i \in \{1, \dots, I\}\\
&\sum_{u \in N_G(v)} y_u \geq 1 && \forall v \in V\\
&\sum_{v \in V} y_v \geq 3\\
&y_v = 1 && \forall v \in V^*\\
&y_v \in \{0, 1\} && \forall v \in V\\
&q \in \mathbb{R}
\end{aligned}$$

Since the number of extreme rays and extreme points may be huge (as the number of columns in column generation), we might generate them by solving the dual of the subproblem using Kruskal algorithm. We would then have an easy subproblem, and a hard master problem, which is the opposite of what happens with column generation.

4.1 Strategy for Solving the Subproblem

The strategy is to solve the primal of the subproblem (having x_e as decision variables) using the Dual Greedy Algorithm of Kruskal. Doing so allows to compute dual variables π_1 and π_S for all $S \subseteq V$.

The (primal) subproblem is infeasible when the set of dominating vertices is not connected (it is always dominant however). In that case, we should be able to build an extreme ray for the dual subproblem, that should be unbounded.

4.2 Dealing with Primal Subproblem Infeasibility

We want to generate “feasibility cuts” in the master problem whenever the dominating set it just returned is not connected. In order to enforce these cuts, we show the following lemmas.

Preliminary note: Let $S \subset V$, a non-dominated set of G. This means that there exists a vertex in $v \in V$ whose distance to S is at least two edges. Equivalently, if S is not a dominated set, then $N_G(N_G[S])$ is non-empty. And equivalently, $V \backslash N_G[S]$ is non-empty.

Theorem 4. *Let $G = (V, E)$ be a connected, undirected edge weighted graph. The vertex set $V_T = \{v \in V : y_v = 1\}$ is a connected dominating tree of G if and only if*

$$\sum_{v \in F_S} y_v \geq 1 \qquad \forall S \subset V : S \neq \emptyset \wedge N_G(N_G[S]) \neq \emptyset \tag{6}$$

where $F_S = N_G(S) \cap N_G(V \backslash N_G[S])$ is the set of vertices adjacent to S, and also adjacent to at least one vertex whose distance to S is two. In other words, F_S is a minimum cardinality separator set of S with respect to $V \backslash N_G[S]$.

Remark. The dominance requirements are enforced by the sets S having cardinality one.

Proof. We first show by contraposition that if V_T is a dominating tree of G, then (6) holds. Suppose that there exists a non-empty and non-dominating set $S \subset V$ such that $\sum_{v \in F_S} y_v = 0$. For the dominating requirements to be satisfied for the vertices in S, there must exist $u_1 \in N_G[S] \backslash F_S$ such that $y_{u_1} = 1$. For the dominating requirements to be satisfied for the vertices in $N_G(N_G[S])$, there must exist $u_2 \in V \backslash N_G[S]$ such that $y_{u_2} = 1$. However, since all the vertices $v \in F_S$ are such that $y_v = 0$, there is no path from u_1 to u_2 as such a path would have to pass through at least one vertex in F_S. Consequently, V_T is not a connected dominated set.

Second, we show by contraposition that if (6) holds, then V_T is a dominating tree. If V_T is not a dominating set, then there exists $v \in V$ such that $\sum_{u \in N_G(v)} y_u = 0$. Hence, the constraint of (6) associated with $S = \{v\}$ is violated.

If V_T is a dominated set, but is not connected, then it has at least two connected components. If one of these connected components is a dominating set, then V_T can be reduced to that sole connected component, leading to a solution at lower cost since arcs have a nonnegative cost. If none of these connected components is a dominating set, then let S be any of them. It satisfies $\sum_{v \in F_S} y_v = 0$, hence the constraint in (6) associated with S is violated.

5 Computational Results

The proposed three exact approaches to address the minimum cost dominating tree problem are compared on a subset of instance coming from [7]. These instances are called ins-R-N-D where R is in the set $\{100, 125, 150\}$, N is the number of vertices in the set $\{50, 100\}$, and D is an instance identifier in $\{1, 2, 3\}$. These instances have been generated randomly: the vertices lie in a 500 by 500 square, and two vertices are connected whenever their distance is less than or equal to R. The cost of an edge is the Euclidean distance between its endpoints.

The proposed algorithms have been implemented in C and compiled using gcc 4.8.4 under a Ubuntu system, using the -O3 option. All linear and integer programs are solved using IBM CPLEX 12.7, called through the C API. The computer used to run the experiments has an Intel Core i7 processor at 3.4 GHz and 16 GBytes RAM. CPLEX is set to use at most two cores of the processor.

As a preliminary test, the three proposed approaches are compared on three instances having 50 vertices, but different densities. The results are given in Table 1. The first two columns show the instance name and density in percent. The third one is the optimal objective value returned by the three algorithm. The fourth, fifth and sixth columns provide the CPU time in seconds of the flow-formulation (see Sect. 3.1), the cutting plane algorithm (see Sect. 3.2) and the Benders decomposition approach (see Sect. 4).

Table 1. Computational times and solution values for the three algorithms

Instance	Density	Opt. val.	Flow-model	Cutting plane	Benders
ins-100-050-1	10.0	1204.41	39.86	1.91	362.64
ins-125-050-1	15.8	802.95	127.04	2.14	554.65
ins-150-050-1	22.5	647.75	2638.45	1.78	1194.01

It can be seen that the cutting-plane approach is by far the most efficient one. The flow-based formulation is hampered by its large number of variables and constraints, the situation becoming worse when density increases. It has been observed that this algorithm often finds the optimal solution, but proving optimality can take a lot of time and effort, because of the poor linear programming bound. The modest performances of the Benders decomposition algorithm is probably due to the fact that in the partition used to separate the *complicating variables*, the y_v variables do not make a direct contribution to the objective function, which causes the approach to generate a very large number of cuts to converge.

The best of the three proposed approaches, *i.e.*, the cutting plane algorithm, is compared to the results of [1], in Table 2. The instances can be found in the first column of Table 2. The second column is the graph density in percent, the third column is the objective value reached by the cutting plane algorithm with the corresponding CPU time (in seconds). The last column shows the improvement

Table 2. Computational times and solution values for the cutting plane algorithm, compared with [1]

Instance	Density	Opt. val.	CPU time (s)	Improvement
ins-100-050-1	10.0	1204.41	1.91	51.4%
ins-100-050-2	9.6	1340.44	1.46	98.9%
ins-100-050-3	10.3	1316.39	1.48	99.4%
ins-100-100-1	10.8	1217.47	3897.85	84.5%
ins-125-050-1	15.8	802.95	2.14	60.4%
ins-125-050-2	15.7	1055.10	8.74	93.6%
ins-125-050-3	15.3	877.77	2.40	89.8%
ins-125-100-1	16.1	943.01	4434.32	−307.6%
ins-150-050-1	22.5	647.75	1.78	24.6%
ins-150-050-2	21.4	863.69	7.57	−3.4%
ins-150-050-3	22.0	743.94	2.46	13.4%
ins-150-100-1	21.0	876.69	6021.43	−188.7%

of the CPU time compared to [1]. A negative number indicates that the cutting plane algorithm is slower.

Table 2 shows the the proposed approach can be significantly faster for small instances, and when density is low. The reason why it does not perform well for dense and large instance may be due to the nature of the cuts that are produced. Indeed, for the largest instances, the solutions of the master problem tend to have a large number of connected components (up to 11). This suggests that instead of computing minimum cuts between a pair of connected component, minimum multi-cuts would be more appropriate, as these inequality are tighter than the ones that can be produced based on a pair of connected component. But since finding minimum multi-cut is generally difficult, heuristics may be used for this purpose [5]. In addition, the cutting-plane algorithm may be upgraded to a branch-and-cut algorithm, which would probably be more efficient.

6 Conclusion

The minimum cost dominating tree problem is now a well-studied problem. Approximability results, heuristics and exact algorithms have become available during the last decade. The main contribution of this paper is to study the structural properties of the problem, and to identify the numerous cases for which a polynomial algorithm can be used. These properties can also lead to new proposal for exact and hybrid algorithms. Indeed, enhancing a cutting plane approach would require separating multi-cuts, which may be achieved by a heuristic. Since connectivity is the major challenge of the cutting plane algorithm, another idea would be to pre-compute a pool of valid inequalities for avoiding some disconnected solution to appear.

References

1. Adasme, P., Andrade, R., Lisser, A.: Minimum cost dominating tree sensor networks under probabilistic constraints. Comput. Netw. **112**, 208–222 (2017)
2. Cheriyan, J., Ravi, R.: Lecture notes on approximation algorithms for network problems (1998). http://www.math.uwaterloo.ca/jcheriya/lecnotes.html
3. Méndez-Díaz, I., Zabala, P.: A cutting plane algorithm for graph coloring. Discret. Appl. Math. **156**, 159–179 (2008)
4. Magnanti, T.L., Wolsey, L.A.: Optimal trees. In: Ball, M., Magnanti, M.L., Monma, C.L., Nemhauser, G.L. (eds.) Network Models. Handbooks in Operations Research and Management Science, vol. 7, pp. 503–615. North-Holland, Amsterdam (1995)
5. Puchinger, J., Raidl, G.R., Pirkwieser, S.: MetaBoosting: enhancing integer programming techniques by metaheuristics. In: Maniezzo, V., Stützle, T., Voß, S. (eds.) Matheuristics: Hybridizing Metaheuristics and Mathematical Programming. AOIS, vol. 10, pp. 71–102. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-1306-7_3
6. Schin, I., Shen, Y., Thai, M.: On approximation of dominating tree in wireless sensor networks. Optim. Lett. **4**, 393–403 (2010)
7. Sundar, S., Singh, A.: New heuristic approaches for the dominating tree problem. Appl. Soft Comput. **13**, 4695–4703 (2013)
8. Wolsey, L.: Integer Programming. Wiley, New York (1998)
9. Chaurasia, S.N., Singh, A.: A hybrid heuristic for dominating tree problem. Soft Comput. **20**, 377–397 (2016)

Review of Approaches for Linked Data Ontology Enrichment

S. Subhashree, Rajeev Irny, and P. Sreenivasa Kumar(✉)

Department of Computer Science and Engineering,
Indian Institute of Technology - Madras, Chennai, India
{ssshree,rajeeviv,psk}@cse.iitm.ac.in

Abstract. Semantic Web technology has established a framework for creating a "web of data" where the nodes correspond to resources of interest in a domain and the edges correspond to logical statements that link these resources using binary relations of interest in the domain. The framework provides a standardized way of describing a domain of interest so that the description is machine-processable. This enables applications to share data and knowledge about entities in an unambiguous manner. Also, as all resources are represented using IRIs, a massive distributed network of datasets gets created. Applications can dynamically discover these datasets, access most recent data, interpret it using the associated meta-data (ontologies) and integrate them into their operations. While the Linked Open Data (LOD) initiative, based on the Semantic Web standards, has resulted in a huge web corpus of domain datasets, it is well-known that the majority of the statements in a dataset are of the type that link specific individuals to specific individuals (e.g. Paris is the capital of France) and there is major need to augment the datasets with statements that link higher-level entities (e.g. A statement about Countries and Cities such as "Every country has a city as its capital"). Adding statements of this kind is part of the task of enrichment of the LOD datasets called "ontology enrichment". In this paper, we review various recent research efforts that address this task. We investigate different types of ontology enrichments that are possible and summarize the research efforts in each category. We observe that while the initial rapid growth of LOD was contributed by techniques that converted structured data into the LOD space, the ontology enrichment is more involved and requires several techniques from natural language processing, machine learning and also methods that cleverly make use of the existing ontology statements to obtain new statements.

Keywords: Linked data · Knowledge enrichment · LOD enrichment
T-Box enrichment · Schema enrichment

1 Introduction

The Semantic Web (aka Web of data or Web 3.0) enables data from one source to be linked to any other source and to be "understood" by machines so that

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 27–49, 2018.
https://doi.org/10.1007/978-3-319-72344-0_2

they can perform increasingly sophisticated tasks without human supervision. Semantic Web is often perceived as complementary to the World Wide Web, while it is actually an extension to the World Wide Web. It provides a framework to add new data and metadata to augment the existing web of documents. The Semantic Web technologies bring forth a new "web of data" paving the way for software agents to integrate data from diverse sources in a meaningful manner. RDF (Resource Description Framework) is the technology used to represent the nodes and edges of this new web of data. Linked Data is the particular realization of the web of data and it has now become a major constituent of the Semantic Web [1].

Linked Data refers to a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF. The LOD community project[1] works with the main objective of publishing open datasets as RDF triples and establishing RDF links between entities from different datasets. LOD complements the World Wide Web with a data space of entities connected to one another with labelled edges, which represent the relations among entity pairs (or entities and literal values).

With over 1014 interlinked datasets[2] across diverse domains such as life science, geography, politics, etc., the Linked Data initiative now supports a variety of applications ranging from semantic search to open domain question answering. For example, the Google's Knowledge Graph which is powered (partly) by the Freebase linked dataset is now being used by Google to enhance its search results with semantic-search information gathered from a wide variety of sources[3]. While many prominent organizations have started realising and exploiting the potential of linked datasets, these linked datasets are far from being complete [54]. More domains need to be covered, and more entities, concepts and links between them are required to be represented as RDF to enable improved and more intelligent usage of Linked Data. Sophisticated question answering systems like Watson which have linked datasets as part of their knowledge sources make use of the enriched linked datasets to answer more number and also a wider range of questions. The Linked Data community has realised the importance of enriching the linked datasets and hence the number of efforts towards enriching linked datasets in LOD have increased immensely in the past few years. A comprehensive study of the works done on Linked Data enrichment so far will help the community to understand the impact of LOD enrichment and its future scope.

1.1 Preliminaries

In this sub-section, we describe the important terms involved in the context of Semantic Web.

A *resource* is a real-world object we want to describe, and it is represented using an URI. A *class* (aka *concept* or *type*) is a group of resources, which is

[1] http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData.

[2] http://lod-cloud.net/state/state_2014/.

[3] http://neilpatel.com/blog/the-beginners-guide-to-the-googles-knowledge-graph/.

also a resource by itself. A *property* (aka *role* or *relation* or binary *predicate*) is a relation between resources, and is also a resource by itself. A *statement* (aka an RDF *triple*) is composed of three parts - (*subject*, *predicate*, *object*) where the subject is a resource, predicate is a property and object is a resource or a *literal*. A literal is a constant value such as a string or a date. Given below is an example of an RDF triple:

(<http://dbpedia.org/resource/Barack_Obama>,
<http://dbpedia.org/ontology/birthPlace>,
<http://dbpedia.org/resource/Honolulu>).

A statement can be represented as a directed edge of a graph or as a triple or in XML (Fig. 1, Listings 1.1 and 1.2 respectively[4]).

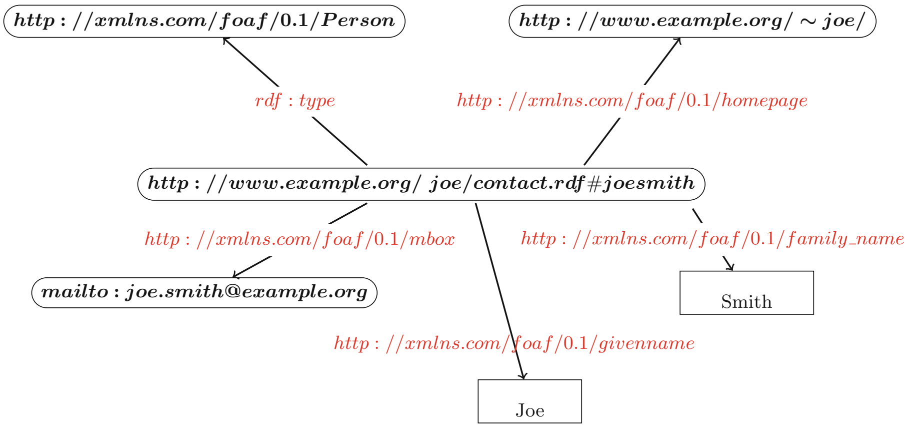

http://xmlns.com/foaf/0.1/Person
http://www.example.org/~joe/
rdf:type
http://xmlns.com/foaf/0.1/homepage
http://www.example.org/ joe/contact.rdf#joesmith
http://xmlns.com/foaf/0.1/mbox
http://xmlns.com/foaf/0.1/family_name
mailto:joe.smith@example.org
Smith
http://xmlns.com/foaf/0.1/givenname
Joe

Fig. 1. RDF graph representation

```
@prefix      : <http://www.example.org/~joe/contact.rdf#> .
@prefix foaf : <http://www.xmlns.com/foaf/0.1> .
@prefix rdf  : <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

:joesmith a foaf:Person .
       foaf:givenname"Joe";
       foaf:last_name"Smith";
       foaf:homepage <http://www.example.org/~joe/>;
       foaf:mbox <mailto:joe.smith@example.org> .
```

Listing 1.1. Triple Representation

[4] http://www.obitko.com/tutorials/ontologies-semantic-web/rdf-graph-and-syntax.html.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:foaf="http://www.xmlns.com/foaf/0.1"
    xmlns="http://www.example.org/~joe/contact.rdf#">

  <foaf:Person rdf:about=
    "http://www.example.org/~joe/contact.rdf#joesmith">
   <foaf:mbox rdf:resource="mailto:joe.smith@example.org"/>
   <foaf:homepage rdf:resource="http://www.example.org/~joe/"/>
   <foaf:family_name>Smith</foaf:family_name>
   <foaf:givenname>Joe</foaf:givenname>
  </foaf:Person>
</rdf:RDF>
```

Listing 1.2. RDF/XML Representation

There are two types of properties. An *object property* is a property between two resources while a *datatype property* is a property between a resource and a literal. The *domain* of a property is an assertion about the type of the resources that occur as subject of the property. Similarly, the *range* of a property is an assertion about the type of the resources that occur as object of the property. Domain (range) statements are also sometimes considered as domain (range) restrictions as they impose certain restrictions on the individuals that can be in the subject (object) position of a statement. For example, regarding the object property birthPlace mentioned above, one can state that its domain is a concept called Person and its range is Place. This allows us to infer that Barack_Obama is of type Person and Honolulu is of type Place.

An *ontology* is an explicit and formal representation of knowledge about a domain. It consists of classes, properties, axioms relating the classes and properties, and individuals of the domain. Statements in an ontology are divided into *T-Box* and *A-Box*. The T-Box is the terminological component of the ontology. It consists of class descriptions, properties and axioms involving them. The A-Box forms the assertion component of the ontology. Statements about the individuals (instances) fall into the A-Box.

Class expressions are used to give a detailed description about a class. There are two different types of class expressions:

- Atomic concept - denoted by a concept name Eg.: Person.
- Compound concept - denoted as a class expression involving one or more of the following operators (where A and B are class expressions, R is a property or role):
  1. Union of classes - $A \sqcup B$
     For example, the expression $Father \sqcup Mother$ can be used to describe the class $Parent$.

  2. Intersection of classes - $A \sqcap B$
     For example, the expression $Male \sqcap Parent$ can be used to denote the class $Father$.

  3. Complement of a class - $\neg A$
     For example, the expression $\neg Male$ can be used to describe the class of individuals who are not in the $Male$ class.

4. Existential restriction - $\exists R.A$
   This expression denotes the class of all individuals that are related to some individual of type A through a relation R. For example, the expression $\exists hasChild.Female$ can be used to describe the class of individuals who have daughters.

5. Universal restriction - $\forall R.A$
   This expression denotes the set of all those individuals whose all R-successors belong to the class A (if (x,R,y) is a triple, y is called an R-successor of x). For example, the expression $\forall hasChild.Female$ can be used to describe the class of individuals who have only daughters as children.

6. Cardinality restriction - $\leq nR.A$
   This expression denotes the set of all individuals that have at most n R-successors. Similarly, $\geq nR.A$ can be used to place a lower bound on the R-successors. For example, the expression $\geq 2hasChild.Female$ can be used to describe the class of individuals who have at least 2 daughters.

Different Description Logics are formed from subsets of these operators, more details of which can be found in [3]. The above mentioned description logic (DL) notation is used occasionally in the paper to give class descriptions.

Important ontology frameworks and languages are listed below:

- RDF - Resource Description Framework - defines constructs which are the building blocks of the Semantic Web such as classes and properties. E.g.: rdf:type
- RDFS - Resource Description Framework Schema - defines properties and classes of RDF resources. E.g.: rdfs:subClassOf
- OWL - Web Ontology Language - defines richer ontology constructs. E.g.: owl:disjointWith, owl:sameAs
- SPARQL - SPARQL Protocol and RDF Query Language - a query language similar to SQL in syntax. It is used to query the triples in linked datasets.

1.2 Prominent Linked Data Projects

DBpedia: DBpedia [32] is one of the most popular linked datasets and has been developed based on a crowd-sourced community effort. DBpedia is composed of the structured information extracted from Wikipedia articles and is represented in triple format. Currently, DBpedia is available in 125 languages. The English version of the DBpedia Knowledge Base (KB) currently describes 6.6 million entities, out of which, 5.5 million are described in a consistent ontology[5] including

[5] http://wiki.dbpedia.org/downloads-2016-10#dbpedia-ontology.

1.5 million persons, 840 K places, 496 K works, 286 K organizations, 306 K species, 58 K plants and 6 K diseases[6].

YAGO: YAGO is a huge linked dataset constructed through automatic extraction from sources such as Wikipedia and WordNet. The current version of YAGO, namely, YAGO3 [34] has around 10 million entities (of types persons, organizations, cities, etc.) and contains more than 120 million facts about these entities[7].

LinkedMDB: LinkedMDB is the first open Semantic Web dataset for movies. It contains links to other datasets such as DBpedia, Geonames, etc. and to websites such as IMDb. A few important classes of LinkedMDB include films, actors, movie characters, directors, producers, editors, writers, music composers, soundtracks, and movie ratings[8].

1.3 Categories of LOD Enrichment

This section categorizes and describes the different ways in which the linked datasets in LOD can be enriched. We can classify LOD enrichment works broadly into two types: T-Box enrichment and A-Box enrichment. T-Box enrichment includes the following: discovering property axioms, discovering class axioms, discovering new properties, and discovering new classes. A-Box enrichment involves the following: discovering owl:sameAs links[9], discovering instances of a class (type assertions), discovering instances of existing relations, detecting erroneous type assertions, detecting erroneous relations, detecting erroneous literal values, and detecting erroneous owl:sameAs links.

The focus of this survey is to provide a comprehensive overview of the works proposed for T-Box enrichment. A recent study on knowledge graph[10] refinement approaches [41] can be referred to for works on A-Box enrichment. It should be noted that a knowledge graph mainly consists of individual members (of classes) and relations among them [41] - i.e. a knowledge graph focusses on its A-Box while its T-Box plays a minimal role. However, in the context of Linked Data, the goal is to add more semantics to the dataset which is possible only when we enrich the T-Box (schema) of the linked dataset. As this paper is written from the perspective of LOD enrichment rather than Knowledge Graph enrichment, we mainly focus on T-Box enrichment techniques. However, if there are Knowledge

[6] http://wiki.dbpedia.org/datasets/dbpedia-version-2016-10.

[7] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/.

[8] http://www.linkedmdb.org/.

[9] owl:sameAs is a built-in OWL property which links an individual to another individual denoting that the two resources represent the same real-world entity.

[10] The term Knowledge Graph was coined by Google in 2012, referring to their use of semantic knowledge in Web Search. The term is recently being used in a broader sense: any graph-based representation of some knowledge could be considered a knowledge graph.

Graph enrichment techniques which also focus on T-Box enrichment, we include them in this survey. Papers which discuss ontology building (from the ground-up) are not dealt with in this survey as they are not enrichment works i.e. in such papers, a partially built ontology does not exist.

The rest of the paper is organized as follows: Sect. 2 gives an account of the techniques proposed in the literature for discovering property axioms. A brief summary of approaches proposed in the literature for discovering class axioms is given in Sect. 3. Sections 4 and 5 describe the systems proposed for discovering new properties and new classes respectively. Conclusions drawn from the survey are given in Sect. 6.

2 Discovering Property Axioms

Properties in Linked Data provide semantic associations between instances in Linked Data and thus are indispensable in representing information in the semantic web. Property Axioms give additional information about predicates and the various Property Axioms that can be used are listed in Table 1. Additional details on the semantics of these axioms can be found in [3]. Most linked datasets in the LOD are deficient in Property Axioms. Thus, a considerable effort has been directed towards enriching schemas associated with datasets in LOD by discovering Property Axioms. We categorize these methods to discover axioms as : (1) Instance based and (2) Schema based methods. The instance based methods rely upon triples in the linked dataset while the schema based methods utilize the schema information like Domain or Range restrictions, type statements to enrich the linked datasets with axioms. We discuss these methods in detail below:

Table 1. Semantics of Property Axioms: r_i, r_j are properties in a linked dataset KB and x, y, z are distinct instances in the KB. Here, $r_i(x, y)$ denotes a triple $< x\, r_i\, y >$ in the KB.

Axiom	Semantics
Subsumption (r_i, r_j)	$r_i(x,y) \in \text{KB} \implies r_j(x,y) \in \text{KB}$
Equivalence (r_i, r_j)	$r_i(x,y) \in \text{KB} \implies r_j(x,y) \in \text{KB} \wedge r_j(x,y) \in \text{KB} \implies r_i(x,y) \in \text{KB}$
Symmetry (r_i)	$r_i(x,y) \in \text{KB} \implies r_i(y,x) \in \text{KB}$
Inverse (r_i, r_j)	$r_i(x,y) \in \text{KB} \implies r_j(y,x) \in \text{KB} \wedge r_j(x,y) \in \text{KB} \implies r_i(y,x) \in \text{KB}$
Asymmetry (r_i)	$r_i(x,y) \in \text{KB} \implies r_i(y,x) \notin \text{KB}$
Transitivity (r_i)	$r_i(x,y) \in \text{KB} \wedge r_i(y,z) \in \text{KB} \implies r_i(x,z) \in \text{KB}$
Disjoint (r_i, r_j)	$r_i(x,y) \in \text{KB} \implies r_j(x,y) \notin \text{KB}$
Functionality (r_i)	$r_i(x,y) \in \text{KB} \wedge r_i(x,z) \in \text{KB} \implies y = z$
Inverse functionality (r_i)	$r_i(x,y) \in \text{KB} \wedge r_i(z,y) \in \text{KB} \implies x = z$

2.1 Instance Based Methods

Methods under this category leverage the large number of triples present in the linked datasets and use statistical techniques like classification, clustering and association rule mining to discover axioms. Fleischhacker et al. [19] proposed a method to discover all the property axioms shown in Table 1. The method involved the use of an off-the-shelf association rule miner [9] to mine association rules. The inputs to this rule miner were transaction tables which are created from the linked datasets. Each row in the transaction tables represents a pair of instances in the linked data and all the predicates that hold between them. The rules mined by the rule miner are checked against a set of predefined patterns. Rules matching these predefined patterns are selected to be converted to property axioms. Most other works in the literature concentrate on discovering subsumption and equivalence axioms.

Galárraga et al. [22] proposed Association Mining under Incomplete Evidence (AMIE) for mining closed Horn rules under incomplete evidence. The rules generated by AMIE are of the form shown in Eq. (1). Here r is a predicate in the linked dataset and B_i is a triple of the form $<?x \ p_i \ ?y>$ and x, y are placeholder variables for instances in the linked dataset. As such $\overrightarrow{B}$ is called the *Body* of a rule and $r(x, y)$ is called the head of the rule. A rule produced by AMIE is closed, i.e. every variable is in the rule occurs in multiples of two and always in pairs.

$$\begin{aligned} \overrightarrow{B} &\Rightarrow r(x, y) \\ \overrightarrow{B} &= B_1 \wedge B_2 \wedge .... \wedge B_n \end{aligned} \tag{1}$$

The Horn rules in Eq. (1) represent the correlations between properties in the dataset. To ensure efficient computation of the Horn rules, Galárraga et al. proposed several logical constraints in the form of refinement operators. Galárraga et al. [22] also proposed the notion of PCA (Partial Completeness Assumption) which makes concessions in selecting negative assertions for a rule. A negative assertion for the Horn rule shown in Eq. (1) is a subject object pair (x, y) such that it is a valid instantiation of the *Body* of the rule but is an invalid instantiation of the *Head* of the rule. PCA states that for a rule shown in Eq. (1), given a predicate r and its subject x, if we know one corresponding object y then we may assume that we know all objects y- that is the objects associated with x in the data are the only ones that x can get associated with.

In [21], the Horn rules generated by AMIE are interpreted as subsumption or equivalence axioms. The interpretation of rules is based on a set of patterns called ROSA (Rule for Ontology Schema Alignment) rules, shown in Fig. 2. Each rule that fits the patterns shown in Fig. 2 is also associated with a PCA confidence score. The same technique can also be used to align equivalent predicates across heterogeneous datasets. This involves first aligning the instances in the two datasets using the owl:sameAs links between instances in the two datasets prior to mining the rules. Once we have all the rules generated by AMIE, all it takes to identify equivalent predicates across datasets is to compare the rules against

$$r(x, y) \Rightarrow r'(x, y) \qquad \text{(Property Subsumption)}$$
$$r(x, y) \iff r'(x, y) \qquad \text{(Property Equivalence)}$$

Fig. 2. ROSA rules for Property Subsumption and Equivalence

the ROSA rule for Property Equivalence. A owl:sameAs link between two entities in Linked Data signifies that they are synonymous. For instance, the instances yago:Barack_Obama and dbr[11]:Barack_Obama are resources to identify the same person and hence will be linked by a owl:sameAs link. For datasets where the owl:sameAs links are not mentioned, Galárraga et al. in [20] introduce a technique to canonicalize the instances and predicates across heterogeneous linked datasets. They do this by first clustering synonymous instances using a technique called Token Blocking [40]. Under such a technique synonymous instances in the two datasets like President_Obama and Barack_Obama will be placed in the same cluster implying that they refer to the same entity. Post this clustering, we obtain a normalized dataset where instances have been aligned. Using these aligned instances, we can now align equivalent predicates across the two datasets. To obtain equivalent predicates a similar procedure involving use of AMIE and ROSA rules can be employed. Each of the equivalent predicate pairs discovered in the above methods can be added to the ontology as Equivalent Property Axioms. The same holds true for the Subsumption Property Axioms.

On similar lines, in our recent work [25], we discover latent Inverse and Symmetric axioms in linked datasets. In this work we outline the challenges involved in discovering latent property axioms by an instance based method and then propose measures to overcome these challenges. One such challenge is the presence of synonymous predicates in the Linked Data and a higher preference to use one of them. For instance, we have dbo:infuenced and dbo:influencedBy as predicates in DBpedia, these predicates convey similar meaning but are inverse of each other. However, dbo:influencedBy is more frequently used among the two to make an assertion. This points to a preference of one predicate over the other which makes the discovery of inverse axioms a challenging task. To this end, we introduced predicate-preference factor (*ppf*) to account for the difference in frequency of use of synonymous (but inverse) predicates. Also, to remedy the lack of reliable and useful domain and range information in linked datasets, we introduced a novel semantic-similarity measure which uses the rdf:type information of instances in the subject and object of a predicate to suggest the reliable axioms. Through experiments we show that the proposed method discovers twice as many axioms, at improved accuracy.

The triples in a linked dataset can also be visualized as a graph with the instances in the subject/object of a triple as nodes and the predicates as edges. Based on this view of linked dataset as a graph, many works apply graph mining techniques to extract meaningful semantic associations between the nodes or edges in the graph. However, most of these approaches consider just the instance-level

[11] http://dbpedia.org/resource.

information (i.e. triples) to suggest rdf:type statements [8,27], to summarize graph entities [47] or query re-writing [55].

2.2 Schema Based Methods

Methods in this section use the schema-level knowledge in addition to using the instance-level information. Axioms discovered by instance based methods do not use the schema information associated with the linked datasets. Work by Barati et al. [5] describe how the lack of schema could negatively impact the induction of axioms. To this end, they propose SWARM (Semantic Web Association Rule Mining), which generalizes Association Rule mining for the semantic web setting. SWARM adds semantics to the association rules by using schema-level knowledge such as rdf:type and rdfs:subClassOf statements. Augmenting the association rules with semantics allows us to interpret them as Behavioral Patterns. For instance, consider a rule mined by SWARM as shown below:

$$\{Person\} : (livesIn, Delhi) \Rightarrow (Speaks, Hindi)$$

The rule above means that the dataset contains many instances to support the pattern that a Person who is a resident of Delhi, speaks Hindi and SWARM uses such rules to identify behavioral patterns from the linked datasets.

Ontology Matching and Alignment techniques [43] involve finding correspondences among the properties either in the same ontology or across different ontologies. Recent advances in this field have given emphasis on the use of large linked datasets to align ontologies or match equivalent properties across ontologies based on the evidence in the linked datasets. For instance, Suchanek et al. propose PARIS [45], which automates the matching of instances, classes and properties across ontologies. PARIS presents a probabilistic approach to estimate the degree of overlap between the instances of two properties in the datasets under consideration. It processes the instances in the linked dataset as well as the ontologies associated with them to align equivalent predicates across datasets. To work with heterogeneous datasets, Suchanek et al. begin by finding equivalent instances across these datasets. They propose a probabilistic model to find equivalent instance pairs. For example, two instances $x \equiv x'$ holds if there is a common predicate r such that triples $r(x, y)$ and $r(x', y')$ exist in the datasets, $y \equiv y'$ and r is inverse-functional. Here x, y belong to one dataset while x', y' belong to another dataset. Observe that to align equivalent instances using the above method, a common predicate r must exist in the two datasets. In addition to finding equivalent instances, PARIS also attempts to discover equivalent predicates (r, r') across the two datasets and does so by using the instances aligned in the method mentioned above. To discover equivalent predicate pairs it checks for the existence of subsumption relation between them, i.e. $r \equiv r'$ if $r \sqsubseteq r'$ and $r' \sqsubseteq r$. Here, $r \equiv r'$ implies that r, r' are equivalent predicates and $r' \sqsubseteq r$ implies that r' is a sub-property of r. PARIS determines the probability that r' is sub-property of r i.e. $Pr(r' \sqsubseteq r)$ as the ratio of number of instance-pairs x, y in r' that are also in r. Note that with the discovery of new equivalent predicate pairs, we can update the probability of equivalence of two instances in the dataset which in turn updates the probability of equivalence of predicate pairs. Thus, the two steps of finding equivalent instances and equivalent predicates are iterated repeatedly until convergence, i.e. when the probabilities do not change any more. It is found experimentally that the convergence is reached after a few iterations. Details about how the probability values were calculated are explained in [45].

On similar lines, Koutraki et al. [28] propose *SORAL* (Supervised Ontology Relation Alignment), a supervised approach to learn the subsumption and equivalence property axioms. They propose the use of several ILP (Inductive Logic Programming) and frequency based features to model a binary classifier to determine if a pair of predicates form an equivalence or subsumption axiom. Some of these features are discussed below:

1. **ILP based features**: This set of features include the confidence measure calculated normally and confidence calculated under the partial completeness assumption [22]. PCA works best when predicates are functional or quasi-functional (The authors in [22] quantize the functionality of a predicate as a value between $0, 1$ where a value of 1 implies the predicate is functional and 0 otherwise. Quasi-functional predicates are those which have a functionality values close to 1). To overcome this drawback, Koutaki et al. introduce PIA (Partial Incompleteness Assumption) which can be considered as a weighted PCA for less functional predicates.
2. **Frequency based features**: These features consider statistics of entities in the dataset like cardinality of relations, type distributions of predicates etc. The features under this category include the functionality of predicates, Jaccard Similarity between the type distributions of 2 predicates. These features also include joint probabilities of confidence score calculated normally and under PCA.

It is worth noting that the training data used in the learning algorithm was created by the authors. Thus, being a supervised technique to align predicates, it is dependent on existence of a training resource. Koutraki et al. [28] also suggest a method to alleviate the challenges of handling large linked datasets by using some sampling techniques. They present experimental results for sample size 100, 500 and 1000. Through experiments Koutraki et al. show that a supervised method to learn subsumption and equivalence axioms based on degree of overlap of instance between two relations is effective in matching properties across ontologies.

However, a major drawback of techniques described above is assuming the existence of common instances between ontologies. While it is a reasonable assumption to make, the methods that depend on common instances fail when the ontologies being aligned share very few or no common instances. To overcome the lack of common instances in an ontology, Wijiya et al. [52] propose PIDGIN, a system that supplements the lack of common overlapping instances between ontologies with the information present in large natural language corpus. They use the corpus to ground the relations and instances in the ontology to verbs and instances in the corpus respectively. This makes up for the lack of common instances between ontologies being matched.

Domain and Range Restrictions. Often overlooked albeit important part of ontologies are the Domain and Range restrictions related to properties. These restrictions ensure that the instances in the subject or object of a property are of the correct class-type. For instance dbo:manager has class dbo:SportsTeam as Domain and class dbo:Person as Range, which means that the instances in the subject of dbo:manager should belong to the class dbo:SportsTeam. However, Tonon et al. [48] show that in most linked datasets, the domain and range restrictions are violated. Thus, we can enrich the corresponding ontology by updating the domain and range restrictions based on the evidence in the linked datasets. For example, consider the property dbo:manager above. Even though the DBpedia ontology mentions dbo:SportsTeam, the instances in the linked data suggests that the domain should be dbo:SportsSeason.

Work by Tonon et al. [48] explores determining the domain and range of properties based on the instances in the linked dataset. They propose LeXt and ReXt to suggest the instance based domain and range of properties. The LeXt performs a depth-first search on the class-type hierarchy for each instance in the subject of a property to statistically determine the most specific class of instances occurring as the domain of the properties. Similarly ReXt determines the instance-based range of a predicate. Töpper et al. [49] also propose a frequency based method to suggest the domain and range of properties in linked dataset based on the class-types of the instances in the subject and object of a property.

3 Discovering Concept Axioms

In ontologies, Concept Axioms play an important role in expressing the relationships that hold between the different Concepts. The semantics of Concept axioms are shown in Table 2 where we see that compared to property axioms, class axioms are less diverse.

Töpper et al. [49] motivated the need for disjoint axioms as a means to find inconsistencies in a linked dataset. They propose to find similarity between two concepts in the ontology, thus, those concept pairs that have similarity scores below a fixed threshold are considered disjoint. To this end, they represent a concept (C) in the vector space. The length of the vector is equal to the number of properties

Table 2. Semantics of Concept Axioms: C_i, C_j are concepts in a linked dataset KB and x, y, z are distinct instances in the KB. Here, $C_i(x)$ denotes that x is of class-type C_i in the KB.

Axiom	Semantics
Subsumption (C_i, C_j)	$C_i(x) \in \text{KB} \implies C_j(x) \in \text{KB}$
Equivalence (C_i, C_j)	$C_i(x) \in \text{KB} \implies C_j(x) \in \text{KB} \wedge C_j(x) \in \text{KB} \implies C_i(x) \in \text{KB}$
Disjoint (C_i, C_j)	$C_i(x) \in \text{KB} \implies C_j(x) \notin \text{KB}$

in the dataset. The weight of each property is modeled after *tf, idf* in Information Retrieval where the *tf* part denotes frequency of occurrence of the property with class C in the dataset and the *idf* part denotes the general relevance of the property in the dataset. Additionally, Fleischhacker et al. [18] propose a method to inductively learn disjointness axioms. They discuss multiple strategies like learning correlation between two concepts based on the count of common instantiations between them. Thus, concepts that have very low or negative correlation are considered to be disjoint with each other. Another technique they suggest is similar to [19] where the difference lies in the representation of rows (discussed in Sect. 2.1). In this case, a row in the transaction table represents the set of concepts that an instance belongs to.

Similar to property axioms, a large portion of the work in literature discusses the discovery of concept subsumption axioms. The set of all subsumption axioms in an ontology aid in creating the class hierarchy or the taxonomy while equivalence axioms are mostly used to align two different ontologies. Volker et al. [51] propose a framework which is a precursor to [19], explained in Sect. 2.1. As explained in the discussion about the disjointness axioms above, the difference between [19,51] is in the representation of transaction tables and in the patterns that are matched to interpret association rules as axioms. Li et al. [33] suggest an improvement over [51] by proposing a method to mine axioms more efficiently. It involves dividing the linked dataset into several blocks (based on disjoint properties) to facilitate the application of mining axioms in parallel. Also note that the methods [21,45] mentioned in Sect. 2 can also be used to find equivalent and subsumption class axioms.

In addition to the axioms shown in Table 2, [33,51] also discover class expressions like $C_i \sqsubseteq \exists r.C_j$ or $\exists r.C_j \sqsubseteq C_i$. Here C_i, C_j are concepts and r is a property in an ontology. The class expression above can be considered as a specialized form of subsumption axioms where the latter expression suggests that whenever we have a triple $< x \ r \ y >$ in KB and $C_j(y)$, then we have $C_i(x)$. Such class expression are useful in describing class definitions. For instance for the expression $C_i \sqsubseteq \exists r.C_j$, if r is *authorOf*, C_j is *Journal_Article* and C_i is *Doctoral_Advisor* then, it means that every *Doctoral_Advisor* besides other things has authored a *Journal_Article*.

The DL-Learner framework [12] encompasses various algorithms for inductive learning of concept axioms and class expressions. The procedure followed by the framework to detect axioms is as follows [11]: Frequent axiom patterns in various ontologies are discovered and converted into corresponding SPARQL query patterns. The query patterns are then applied to other datasets to enrich them with new axioms. For example, in the experiments conducted by [11], patterns have been mined from more than one thousand ontologies and then applied on the DBpedia dataset. A few patterns which were obtained among the top 15 patterns are given below:

A **SubClassOf** p **some** (q **some** B), or equivalently $A \sqsubseteq \exists p.(\exists q.B)$ in DL
A **equivalentTo** B **and** p **some** C, or equivalently $A \equiv B \sqcap \exists p.C$
A **SubClassOf** p **value** A, or equivalently $A \sqsubseteq \exists p.\{A\}$

A few axioms which were obtained by applying the above patterns on DBpedia are:

> Song **equivalentTo** MusicalWork **and** (artist **some** Agent) **and** (writer **some** Artist), or equivalently $Song \equiv MusicalWork \sqcap (\exists artist.Agent) \sqcap (\exists writer.Artist)$
> Conifer **SubClassOf** order **value** Pinales, or equivalently $Conifer \sqsubseteq \exists order.\{Pinales\}$

The algorithms proposed under the DL-Learner framework for learning of class expressions are described in Sect. 5.

3.1 Discussion

It is worth noting that most of the methods we discussed in this and in the previous section focus on the discovery of subsumption and equivalence axioms, be it property or concept axioms. These axioms, while crucial to formation of a class/property hierarchy, limit the diversity of the axioms in the ontology. We believe that expanding the scope of these methods to discover additional axioms namely, Functionality, Inverse functionality, Inverse, Transitivity will enhance the understanding of the underlying domain and also help in keeping the dataset consistent with the world-knowledge. Thus, the discovery of axioms that add value to the ontology is one of the promising areas of research. Additionally, with the use of PCA (and PIA), several works described in the Sections above compensate for the incomplete nature of data in the semantic web. While this is a step in the correct direction, a technique that is not restricted by the functionality of the predicates (like PCA) will surely provide a more versatile method to overcome the incompleteness in semantic web and thus is a potential future extension.

4 Discovering New Properties

Most of the linked datasets are deficient in the number of object properties they have. For example, the linked dataset YAGO has 488,469 classes [34]. Among such a huge number of classes, surprisingly there are only 32 object properties[12] and hence looking for more object properties to connect these classes becomes a necessary step towards enriching linked datasets. Details of the methods proposed in literature to add new object properties are given below:

Several works have been proposed to discover new object properties in the context of enriching the NELL (Never Ending Language Learner) Knowledge Base. NELL [37] is a part of the "Read the Web" project[13] which is an initiative

[12] http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/statistics/ - totally there are 60 object properties, but 28 of them connect the domain class to the class http://dbpedia.org/class/yago/YagoLiteral.

[13] http://rtw.ml.cmu.edu/rtw/.

to create a machine that learns to read the entire web. NELL has been running continuously since January 2010 and it performs two main tasks - extract facts from web pages, and improve its learning techniques to extract more accurate facts in future. NELL has a few helper systems which aid it in extending its T-Box so that more instances of the newly discovered relations and concepts can be fetched by NELL from the web. Mohamed et al. proposed OntExt (Ontology Extension system) [38] which discovers new relations given two categories from the NELL ontology (classes are called as categories in the NELL KB). OntExt does this by extracting text patterns from the web corpus and clustering them based on co-occurrence values. For example, if the phrases "Ganges flows through Allahabad" and "Ganges in the heart of Allahabad" occur in the web corpus with a very high frequency then this is taken as an indicator that the patterns, "flows through" and "in the heart of" are similar to each other. When such an evidence is shown by many number of subject-object pairs, OntExt gives a very high similarity score between the two patterns. In general, OntExt works in the following manner: given a pair of categories and a set of sentences-each containing a pair of instances known to belong to the given categories, OntExt collects the words in between the instances from each sentence and calls these words a "context-pattern". Then it builds a co-occurrence matrix (context-pattern X context-pattern) which is based on the frequencies of occurrence of the context-patterns with the same subject-object instance pairs. For example, in the above case of finding relations between Rivers and Cities, if the pair "Ganges" and "Allahabad" occurs with the context-pattern "flows through" with a frequency f_1 and the pair occurs with the pattern "in the heart of" with a frequency f_2, then the matrix entry corresponding to these two context-patterns will be given a value of $(f_1 + f_2)$. In case there is another subject-object pair (for example-Thames, London) occurring with both these context-patterns with frequencies f_3 and f_4 respectively, then the matrix cell value becomes $(f_1 + f_2 + f_3 + f_4)$. K-means clustering is applied on the normalized matrix to group the related context-patterns together. The centroid of each cluster is proposed as a new relation. OntExt also generates the instances (subject-object pairs) of these new relations based on how often each subject-object pair co-occurs with a new relation in the web corpus. OntExt was followed by newOntExt [6,7] whose primary goal was to overcome certain challenges faced by OntExt and to make the ontology extension process scalable and feasible so that it can be effectively utilised on the NELL Knowledge Base. The authors incorporated the following changes in newOntExt: Instead of considering all the words in between the two input instances as a pattern, newOntExt used ReVerb [17] for extracting the patterns in order to reduce the number of noisy patterns obtained. In order to reduce the computational cost, a more elegant file structure was used for searching through the sentences. Instead of considering every pair of categories as input to this system, reduced category groups of interest were formed to pick the input category pairs (for example, the categories related to the domain of sports: sports league, sport, athlete and sports team). Later, Cergani et al. [13] identified the following issues in the clustering phase of newOntExt: An entity pair cannot be connected by multiple relations and even the most obvious outliers (noisy relations) cannot be removed by their clustering phase. Also, the number of clusters had to be specified before-hand. Hence they have proposed a minor improvement to overcome these issues by replacing the clustering phase of newOntExt with matrix factorization techniques such as Non-negative Matrix Factorization (NMF) and Boolean Matrix Factorization (BMF). The authors of [44] made the following observations w.r.t. the working of newOntExt: The number of incorrect relations produced by newOntExt is high mostly because the new relations are not filtered based on any contextual check. Even semantically dissimilar but meaningful relations are placed in the same cluster and hence important relations get dropped by newOntExt. Also, the relations produced are not grounded to the knowledge base (by grounding, we mean mapping of discovered relations to existing LOD object properties). In [44], the authors propose a system called DART (**D**etecting **A**rbitrary **R**elations for enriching **T**-Boxes of Linked Data) to enrich linked datasets with new object properties between two given classes by means of contextual similarity detection and paraphrase detection tools. DART performs grounding of relations and is also shown to be better than newOntExt in terms of both precision and recall.

Nimishakavi et al. [39] have explored the idea of using tensor factorization models for inducing new relations and their schemas from OpenIE triples into an ontology. By a relation schema, they mean the type signature of the relation. For example, the type signature for the relation $cityLocatedInCountry$ is cityLocatedInCountry(City, Country). The OpenIE triples are represented as a tensor. An element x_{ijk} of the tensor refers to triple formed by i^{th} noun phrase, j^{th} noun phrase and k^{th} verb phrase. The possible hypernyms of the noun phrases are collected from the text corpus using Hearst patterns [23] (for example, "a <hypernym> such as a <noun phrase>") and stored in a matrix. Another matrix is used to store the similarity between the verb phrases (relations). The intuition behind using this similarity matrix is that if two relations are found to be similar in meaning, then their type signatures should also be same/similar. By similarity, the authors mean the cosine similarity of the Word2Vec vectors of the verbs [36]. Coupled factorization of the tensor and the two input matrices is performed to obtain a core tensor which contains the relation schemas such as $suffer\_from(patient, disease)$, $have\_undergo(patient, treatment)$ etc. and a matrix containing the assignment of the noun phrases to the classes.

SOFIE, the system proposed in [46], has been primarily designed for adding more instances of existing relations i.e. for A-Box enrichment. However, the authors have conducted experiments and demonstrated its application for adding a new property and its instances. The authors introduce seed instances manually for a new property and thus adopt the same system to add more instances of the new property. SOFIE works in the following manner: First, facts are collected in two ways - ontological facts collected from the dataset under consideration (includes the manual seed instances for the relation) and textual facts collected from the corpus. These existing facts are given a truth value of 1. Hypotheses for new facts are formed using the known facts. Truth value of these hypotheses are said to be unknown. In order to determine which hypotheses should be accepted as true facts, a set of manually written logical rules are employed. Now the problem is recast as finding the hypotheses that are likely to be true, such that maximal number of rules are satisfied. This can be seen as a maximum satisfiability problem (MAX SAT problem) with all facts, hypotheses and rules rewritten as logical clauses in a uniform manner. A lower set of weights are assigned to clauses which can be violated and a very large weight is assigned for those clauses which are derived from existing facts. A new approximation algorithm (as MAX SAT problem is NP-Hard) called the Functional Max Sat (FMS) algorithm has been implemented to solve this Weighted Max Sat problem. It should be noted that SOFIE is different from the other systems described in this Section in the following aspect: SOFIE needs to know what property should be added to the linked dataset, while the other systems do not take this input.

5 Discovering New Classes

There are quite a few works in the literature which focus on learning class expressions to enrich ontologies. Petrucci et al. [42] solve the problem of class expression learning from natural language text with a learn-by-examples approach. They formulate the problem as a machine transduction task. In this case, a sequence of words in natural language has to be converted into a sequence of logical symbols - a formula. The system operates in two parallel phases, namely, sentence transduction and sentence tagging. The sentence transduction phase identifies the logical structure of the formula corresponding to the natural language input given. The output of this phase is a formula template. The sentence tagging phase tags each word of the input sentence into one of the following types: a concept, a role, a number, or a generic word. Then these tagged words are fit into the formula template to generate the final class expression. For example, let (2) be given as input to both the phases.

$$\textit{A bee is an insect that has 6 legs and produces honey.} \tag{2}$$

Sentence transduction phase outputs the template (3) while the sentence tagging phase tags the sentence and outputs (4).

$$C_0 \sqsubseteq C_1 \sqcap (= N_0 R_0.C_2) \sqcap (\exists R_1.C_3) \tag{3}$$

$$\text{A } [bee]_{C_0} \text{ is an } [insect]_{C_1} \text{ that } [has]_{R_0}[6]_{N_0}[legs]_{C_2} \text{ and } [produces]_{R_1}[honey]_{C_3} \tag{4}$$

The outputs of both the phases are combined to produce the class expression given in (5).

$$Bee \sqsubseteq Insect \sqcap (= 6 have.Leg) \sqcap (\exists produce.Honey) \tag{5}$$

Both the phases employ Recurrent Neural Networks (RNNs) to accomplish their goals. The training data for sentence transduction phase would ideally consist of huge number of pairs of natural language sentences and their corresponding

DL axioms. Since such a dataset was not available, the authors have created such a training dataset. The authors have first verbalized a set of OWL class definitions using Attempto Controlled English (ACE) [26] to get definitions such as the one given in Eq. (2). Then natural language variations of the verbalization were added manually and finally a generalized grammar was built to generate huge number of such training instances.

The authors of [2] handle the problem of class expression learning through syntactic transformation of English sentences to OWL axioms. Syntactic transformation is implemented through various rules of transformation of the parse tree of a sentence. The paper proposes a new controlled natural language called TEDEI (TExtual DEscription Identifier) to define the scope of the input sentences that can be handled by their system. They employ an existing controlled natural language, namely ACE, as an intermediate language and in this way, address some of the limitations of ACE in the context of ontology authoring. They also investigate the impact of two types of ambiguity in natural language sentences, namely lexical ambiguity and semantic ambiguity. Instead of producing one axiom from a given sentence, their system generates all possible axioms that can be generated from the sentence, which are then presented to the user.

As mentioned in Sect. 3, the DL-Learner framework [12] encompasses a set of algorithms for learning class expressions by means of refinement operators i.e. a refinement operator is used to traverse an ordered search space in order to determine the correct concept definition. Informally, a refinement operator can be defined as follows: a downward refinement operator is one which gives rise to a set of more specific concepts and an upward refinement operator returns a set of more general concepts for the given input concept. The general goal of these algorithms is to devise refinement operators that have the following properties [31] while still being able to efficiently traverse through the search space in search of good candidate class expressions:

Let ρ be a downward refinement operator.

Finite: ρ is finite iff $\rho(C)$ is finite for any concept C.
Non-redundant: ρ is redundant iff there exists a refinement chain from a concept C to a concept D, which does not go through some concept E and a refinement chain from C to a concept approximately equal to D, which does go through E.
Proper: ρ is proper iff for all concepts C and D, $D \in \rho(C)$ implies $C \not\equiv D$.
Complete: ρ is complete iff for all concepts C and D with $C \sqsubset D$ we can reach a concept E with $E \equiv C$ from D by ρ
Ideal: ρ is ideal iff ρ is finite, complete, and proper.

However, no refinement operator is ideal and hence the algorithms in the framework work towards handling the missing properties. The major refinement-operator based algorithms are OCEL, CELOE, ELTL and ISLE [12]. OCEL (OWL Class Expression Learner) was the first algorithm defined specifically for the Description Logic ALC. It was designed to cope with redundancy and lack of finiteness property of the refinement operator. CELOE (Class Expression Learning for Ontology Engineering) [29] which is an evolved form of OCEL contains changes specific for learning shorter class expressions as long concept expressions are difficult to maintain and understand in the context of ontology creation. ELTL (EL Tree Learner) [30] is an algorithm for class expression learning specifically designed to suit the OWL EL profile. ISLE (Inductive Statistical Learning of Expressions) [10] which is an extension of the ELTL, also took textual evidence from external corpus into account. Information from the corpus has been used to modify the search heuristic and has been proven to give more accurate expressions on manual evaluation.

Another set of algorithms proposed within the DL-Learner framework for class expression learning which are not based on refinement operators are PARCEL and Fuzzy-DLL. PARCEL (Parallel Class Expression Learning) [50] is suitable for situations which are better solved by parallelization. PARCEL computes partial definitions of a learning problem, which are then aggregated to give complete solutions. Fuzzy-DLL [24] was proposed to handle class expression learning in vague and imprecise domains.

While the above described systems learn class expressions, the system proposed in [39] (see Sect. 4) finds and adds new classes (atomic class names) to the ontology in the process of finding new properties. The coupled tensor factorization process results in a core tensor and a matrix. The core tensor consists of the relation schemas generated and the matrix contains noun phrases assigned to new classes.

5.1 Discussion

The task of inducing new properties and classes from within the linked dataset itself is very difficult to accomplish and hence it becomes imperative to make use of external sources. In this context, data generated through web-scale information extraction systems [17] (which include OpenIE systems such as TextRunner [4], WOE [53], ReVerb [17], SRLIE [14], OLLIE [35] and systems such as NELL, ClausIE [15]) serve as a good starting point for enriching Linked Data. Mapping triples from the former kind of systems (let us call them web triples) to Linked Data's RDF triples can be beneficial in two ways: Linked Data can give more structure and accuracy to the web triples and Linked Data can be enriched (both A-box as well as T-Box) through the web triples. We have seen this trend in [7,38] and also in [39] where the web triples form one of the main inputs for the proposed system. Another set of works following this direction are [16,56]. [56] proposes a framework to give RDF representation to NELL triples. [16] gives RDF representation to NELL KB by linking it to DBpedia and also enriches DBpedia in the process. However, these works are confined mostly to the NELL KB while the opportunities of exploiting the outcomes of the other web-scale IE projects remain largely unexplored.

6 Conclusion

In order to realize the full potential of Linked Data in various applications, it is important to enrich LOD with as many appropriate ontological axioms and assertions as possible. This paper acquaints the readers with the recent advancements in the field of T-Box enrichment of LOD datasets. Techniques for discovery of property and class axioms are mostly based on the RDF triples from within the linked datasets itself while discovery of new properties and classes rely on external sources of data such as OpenIE triples. These enrichment techniques move the datasets towards completeness, all the while making sure that the datasets remain consistent and the manual effort for verifying the correctness of the newly added properties, classes and axioms is reduced. However, as discussed in Sects. 3.1 and 5.1, there are many directions to be explored that might enable further enrichment of LOD.

References

1. Linked Data - Connect Distributed Data across the Web. http://linkeddata.org/
2. Alex Mathews, K., Sreenivasa Kumar, P.: Extracting ontological knowledge from textual descriptions through grammar-based transformation. In: Proceedings of the Ninth International Conference on Knowledge Capture (K-CAP), 4–6 December, Austin, Texas, USA (2017)
3. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, New York (2003)
4. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. In: IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, 6–12 January 2007, pp. 2670–2676 (2007)
5. Barati, M., Bai, Q., Liu, Q.: Mining semantic association rules from RDF data. Knowl. Based Syst. **133**, 183–196 (2017)
6. Barchi, P.H., Hruschka, E.R.: Never-ending ontology extension through machine reading. In: 2014 14th International Conference on Hybrid Intelligent Systems, pp. 266–272, December 2014
7. Barchi, P.H., Hruschka, E.R.: Two different approaches to ontology extension through machine reading. J. Netw. Innov. Comput. **3**(1), 78–87 (2015)
8. Basse, A., Gandon, F., Mirbel, I., Lo, M.: DFS-based frequent graph pattern extraction to characterize the content of RDF triple stores. In: Web Science Conference 2010 (WebSci 2010) (2010)
9. Borgelt, C., Kruse, R.: Induction of association rules: apriori implementation. In: Härdle, W., Rönz, B. (eds.) Compstat, pp. 395–400. Springer, Heidelberg (2002). https://doi.org/10.1007/978-3-642-57489-4_59
10. Bühmann, L., Fleischhacker, D., Lehmann, J., Melo, A., Völker, J.: Inductive lexical learning of class expressions. In: Janowicz, K., Schlobach, S., Lambrix, P., Hyvönen, E. (eds.) EKAW 2014. LNCS (LNAI), vol. 8876, pp. 42–53. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13704-9_4
11. Bühmann, L., Lehmann, J.: Pattern based knowledge base enrichment. In: Alani, H., et al. (eds.) ISWC 2013. LNCS, vol. 8218, pp. 33–48. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41335-3_3

12. Bühmann, L., Lehmann, J., Westphal, P.: DL-Learner - a framework for inductive learning on the semantic web. Web Semant. Sci. Serv. Agents WWW **39**, 15–24 (2016)
13. Cergani, E., Miettinen, P.: Discovering relations using matrix factorization methods. In: 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, San Francisco, CA, USA, 27 October-1 November, 2013, pp. 1549–1552 (2013)
14. Christensen, J., Mausam, Soderland, S., Etzioni, O.: An analysis of open information extraction based on semantic role labeling. In: Proceedings of the 6th International Conference on Knowledge Capture (K-CAP 2011), 26–29 June, 2011, Banff, Alberta, Canada, pp. 113–120 (2011)
15. Del Corro, L., Gemulla, R.: ClausIE: clause-based open information extraction. In: Proceedings of the 22nd International Conference on World Wide Web, WWW 2013, pp. 355–366 (2013)
16. Dutta, A., Meilicke, C., Stuckenschmidt, H.: Semantifying triples from open information extraction systems. In: STAIRS 2014 - Proceedings of the 7th European Starting AI Researcher Symposium, Prague, Czech Republic, 18–22 August 2014, pp. 111–120 (2014)
17. Etzioni, O., Fader, A., Christensen, J., Soderland, S., Mausam, M.: Open information extraction: the second generation. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - IJCAI 2011, vol. 1, pp. 3–10. AAAI Press (2011)
18. Fleischhacker, D., Völker, J.: Inductive learning of disjointness axioms. In: Meersman, R., et al. (eds.) OTM 2011. LNCS, vol. 7045, pp. 680–697. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25106-1_20
19. Fleischhacker, D., Völker, J., Stuckenschmidt, H.: Mining RDF data for property axioms. In: Meersman, R., et al. (eds.) OTM 2012. LNCS, vol. 7566, pp. 718–735. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33615-7_18
20. Galárraga, L., Heitz, G., Murphy, K., Suchanek, F.M.: Canonicalizing open knowledge bases. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pp. 1679–1688. ACM (2014)
21. Galárraga, L.A., Preda, N., Suchanek, F.M.: Mining rules to align knowledge bases. In: Proceedings of the 2013 Workshop on Automated Knowledge Base Construction, pp. 43–48. ACM (2013)
22. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: AMIE: Association rule Mining under Incomplete Evidence in ontological knowledge bases. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 413–422. ACM (2013)
23. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: 14th International Conference on Computational Linguistics, COLING 1992, Nantes, France, 23–28 August 1992, pp. 539–545 (1992)
24. Iglesias, J., Lehmann, J.: Towards integrating fuzzy logic capabilities into an ontology-based inductive logic programming framework. In: 2011 11th International Conference on Intelligent Systems Design and Applications, pp. 1323–1328, November 2011
25. Irny, R., Kumar, S.P.: Mining inverse and symmetric axioms in Linked Data. In: Proceedings of the Seventh Joint International Semantic Technologies Conference, Gold Coast, Australia, 10–12 November (2017)
26. Kaljurand, K., Fuchs, N.E.: Verbalizing OWL in Attempto Controlled English. In: Proceedings of the OWLED 2007 Workshop on OWL: Experiences and Directions, Innsbruck, Austria, 6–7 June 2007 (2007)

27. Kasneci, G., Elbassuoni, S., Weikum, G.: MING: mining informative entity relationship subgraphs. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 1653–1656. ACM (2009)
28. Koutraki, M., Preda, N., Vodislav, D.: Online relation alignment for linked datasets. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10249, pp. 152–168. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58068-5_10
29. Lehmann, J., Auer, S., Bühmann, L., Tramp, S.: Class expression learning for ontology engineering. J. Web Semant. **9**(1), 71–81 (2011)
30. Lehmann, J., Haase, C.: Ideal downward refinement in the $\mathcal{EL}$ description logic. In: De Raedt, L. (ed.) ILP 2009. LNCS (LNAI), vol. 5989, pp. 73–87. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13840-9_8
31. Lehmann, J., Hitzler, P.: Foundations of refinement operators for description logics. In: Blockeel, H., Ramon, J., Shavlik, J., Tadepalli, P. (eds.) ILP 2007. LNCS (LNAI), vol. 4894, pp. 161–174. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78469-2_18
32. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. Semant. Web **6**, 167–195 (2015)
33. Li, H., Sima, Q.: Parallel mining of OWL 2 EL ontology from large linked datasets. Knowl. Based Syst. **84**, 10–17 (2015)
34. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: a knowledge base from multilingual Wikipedias. In: CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, 4–7 January 2015, Online Proceedings (2015)
35. Mausam, M.S., Bart, R., Soderland, S., Etzioni, O.: Open language learning for information extraction. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, pp. 523–534 (2012)
36. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems, vol. 26, pp. 3111–3119 (2013)
37. Mitchell, T., Cohen, W., Hruschka, E., Talukdar, P., Betteridge, J., Carlson, A., Dalvi, B., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., Mohamed, T., Nakashole, N., Platanios, E., Ritter, A., Samadi, M., Settles, B., Wang, R., Wijaya, D., Gupta, A., Chen, X., Saparov, A., Greaves, M., Welling, J.: Never-ending learning. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2015) (2015)
38. Mohamed, T.P., Hruschka Jr., E.R., Mitchell, T.M.: Discovering relations between noun categories. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, pp. 1447–1455 (2011)
39. Nimishakavi, M., Saini, U.S., Talukdar, P.P.: Relation schema induction using tensor factorization with side information. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, 1–4 November 2016, pp. 414–423 (2016)
40. Papadakis, G., Ioannou, E., Niederée, C., Fankhauser, P.: Efficient entity resolution for large heterogeneous information spaces. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, pp. 535–544. ACM (2011)

41. Paulheim, H.: Knowledge graph refinement: a survey of approaches and evaluation methods. Semant. Web **8**(3), 489–508 (2017)
42. Petrucci, G., Ghidini, C., Rospocher, M.: Ontology learning in the deep. In: Blomqvist, E., Ciancarini, P., Poggi, F., Vitali, F. (eds.) EKAW 2016. LNCS (LNAI), vol. 10024, pp. 480–495. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49004-5_31
43. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Trans. Knowl. Data Eng. **25**(1), 158–176 (2013)
44. Subhashree, S., Kumar, P.S.: Enriching linked datasets with new object properties. CoRR abs/1606.07572 (2016). http://arxiv.org/abs/1606.07572
45. Suchanek, F.M., Abiteboul, S., Senellart, P.: PARIS: probabilistic alignment of relations, instances, and schema. Proc. VLDB Endow. **5**(3), 157–168 (2011)
46. Suchanek, F.M., Sozio, M., Weikum, G.: SOFIE: a self-organizing framework for information extraction. In: Proceedings of the 18th International Conference on World Wide Web, WWW 2009, New York, pp. 631–640. ACM (2009)
47. Thor, A., Anderson, P., Raschid, L., Navlakha, S., Saha, B., Khuller, S., Zhang, X.-N.: Link prediction for annotation graphs using graph summarization. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011. LNCS, vol. 7031, pp. 714–729. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25073-6_45
48. Tonon, A., Catasta, M., Demartini, G., Cudré-Mauroux, P.: Fixing the domain and range of properties in Linked Data by context disambiguation. In: LDOW@ WWW (2015)
49. Töpper, G., Knuth, M., Sack, H.: DBpedia ontology enrichment for inconsistency detection. In: Proceedings of the 8th International Conference on Semantic Systems, pp. 33–40. ACM (2012)
50. Tran, A.C., Dietrich, J., Guesgen, H.W., Marsland, S.: An approach to parallel class expression learning. In: Bikakis, A., Giurca, A. (eds.) RuleML 2012. LNCS, vol. 7438, pp. 302–316. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32689-9_25
51. Völker, J., Niepert, M.: Statistical schema induction. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011. LNCS, vol. 6643, pp. 124–138. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21034-1_9
52. Wijaya, D., Talukdar, P.P., Mitchell, T.: PIDGIN: ontology alignment using web text as interlingua. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 589–598. ACM (2013)
53. Wu, F., Weld, D.S.: Open information extraction using Wikipedia. In: ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 11–16 July 2010, Uppsala, Sweden, pp. 118–127 (2010)
54. Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auer, S.: Quality assessment for Linked Data: a survey. Semant. Web **7**(1), 63–93 (2016)
55. Zheng, W., Zou, L., Peng, W., Yan, X., Song, S., Zhao, D.: Semantic SPARQL similarity search over RDF knowledge graphs. Proc. VLDB Endow. **9**(11), 840–851 (2016)
56. Zimmermann, A., Gravier, C., Subercaze, J., Cruzille, Q.: Nell2rdf: read the web, and turn it into RDF. In: Proceedings of the Second International Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data, Montpellier, France, 26 May 2013, pp. 2–8 (2013)

Formal Verification of Optimizing Compilers

Yiji Zhang and Lenore D. Zuck(✉)

Department of Computer Science, University of Illinois at Chicago, Chicago, USA
yzhan79@uic.edu, lenore@cs.uic.edu

Abstract. Formally verifying that a compiler, especially an optimizing one, maintains the semantics of its input has been a challenging problem. This paper surveys several of the main efforts in the area and describes recent efforts that target the LLVM compiler infrastructure while taking a novel viewpoint on the problem.

1 Introduction

Formal verification attempts to formally verify a formal system against its formal specifications. It is well known that formal verification is most effective at high levels of abstraction. The verified high-level models, however, have to be transformed into executable code. Hence, A "good" verification effort should include all phases of the translation.

Today's compilers are often *optimizing* and perform many modifications to their input. Consequently, it is virtually impossible to create a mapping between the high level system and the code they produce. Yet, it is often the case, as in mission critical code, that it is vital that every step is formally proven.

Active research into the formal verification of optimizing compilers has been going on almost since the introduction of high-level languages and compilation. Generally speaking, a compiler receives a high level code, translates it into some intermediate language (which we call IR for Intermediate Representation), and performs a sequence of IR into IR modifications, say IR_1 to IR_n. The final IR, IR_n, is then translated into a low level code, often machine language. Here we focus on verification of the IR sequence, namely, formally establishing that IR_n satisfies the specifications described by IR_1. We thus ignore efforts to show that IR_1 implements the high level code, and that the machine code implements IR_n. These, while extremely important, are of a different nature since they deal with two different languages.

We survey the main efforts of accomplishing such a proof. Roughly speaking, there are two directions that have been pursued. One can view the compiler as a *translator* from one IR_i into the next. The first approach is to directly verify the translator itself, that is, to formally verify that for every input of the compiler, its output preserves the semantics of the input. While this seems daunting (and it is),

This work was partially funded by DARPA CSFV project and NSF award CCF-1563393.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 50–65, 2018.
https://doi.org/10.1007/978-3-319-72344-0_3

many modern compilers are modular, so rather than verifying a formidable monolithic code base, one can verify each module separately, making the task more manageable. Of course, if any of the modules is modified, it should be re-verified.

The second approach was first proposed in [24] in the context of equivalence of LISP code and its translation into an assembly language, and then in [21] in the context of translating Signal code into ADA. The approach suggests that rather than verifying the "translator" (the compiler itself), one verifies that the code produced by each run—"translation"—implements the input code. That is, each run of the compiler is verified separately. In the notation above, for every $i = 2, \ldots, n$, one shows that IR_i implements IR_{i-1}. This approach was termed *Translation Validation* (TV). At first glance this seems to be impossible, after all, equivalence of even "just" context-free languages is undecidable. At a second glance this seems to be inefficient, that the overhead incurred will not be worth the freedom of not having to verify the whole compiler. Both these points turn out to be easily addressed. For the first, one should note that the transformations of the code performed by a compiler are simple and one needs only to establish some trivial properties, such as "if $x = 4$ then after $x := x + 3$ is performed, $x = 7$." For the second, it turned out the overhead is minimal and well justifies the effort.

The TV approach has a couple of other advantages. Not only does it alleviate the need to verify a frequently modified moving-target compiler, it also can accommodate compilers that are closed-source as those whose code is proprietary. Moreover, it generates verification conditions (VCs) that can be independently checked by numerous theorem provers. This, in turn, allows for *certification* of the compiled code.

After the survey we describe our current efforts in applying TV to LLVM. LLVM is a relatively new compiler platform that is being adopted by many operating systems. It is open source, and, unlike many other compilers, each of its "passes" (the code that moves from IR_i into IR_{i+1}) is independent from the other. This allows to simplify TV and to apply scant knowledge and understanding of the code to create VCs.

As stated above, the transformations that a compiler performs are rather simple. This is partially due to the known "rule" that the analysis the compiler performs has to be extremely efficient, at most linear (the GCC wiki page, e.g., has as rule 1: "Do not add algorithms with quadratic or worse behavior, ever[1].") However, if one cares more about runtime than about compile-time, then this no longer holds. After all, one may be willing to wait hours or days to optimize a program that is to run frequently (for example, a Domain Name Server). For such, we experimented with using external, possibly slow, more precise static analyzers. Our results are that we can push optimizations even further and get better runtime results when combining TV with such external tools.

[1] https://gcc.gnu.org/wiki/Speedup_areas.

2 A Survey

Compilers are rather buggy, and, consequently, so are optimizing compilers. The work in [26] describes a randomized test-case generator that produces C programs to trigger deep compiler bugs, and, as expected, many (325) bugs were found in numerous (11) compilers. Proteus [12] uses [26] to perform randomized link-time optimization testing and uncovered 37 bugs in GCC and LLVM. More than 75% produce mis-compiled code. Those bugs were all reported and fixed.

One approach to verification of optimizing compilers is to verify the "translator" itself, that is, the compiler. This can be done by generating a machine-checkable manual proof. We describe several examples of this approach, some targeted at LLVM. The other approach is TV, and we describe some of the efforts in this direction.

2.1 Compiler Verification

Perhaps the earliest work in compiler verification dates to 1967, when John McCarthy and James Painter proved the correctness of a compiler that translates arithmetic expressions into machine language [17]. Another early work (starting late 1980s) is described in [11], where the entire chain from high level code to machine code was verified using the theorem prover ACL2. Based on [5], optimizing compilers have become a target for research several decades later, in 2000, and the earlier work employed TV.

Cobalt [14]: Cobalt is a domain-specific language for implementing optimizations as guarded rewrite rules, with its generator of proof obligations that is based on temporal logic to reason about data flow analysis (this type of reasoning was proposed in [25].) The idea behind Cobalt is to create verification of simple transformations that can be later used as building blocks when attempting to establish more complex ones.

Consider, for example, the transformation used for *constant propagation*. Roughly speaking, when a variable, say x, is a constant, then constant propagation is an optimization that replaces every reference to x by that constant. This allows to save the number of allocated registers and detect unreachable code. This transformation can be expressed by the temporal expression whose meaning is explained below:

$$\mathit{Stmt}(x := C);\ \neg \mathit{MayDef}(x)\ \mathbf{Until}\ (y := \mathit{expr}\ \Rightarrow y := \mathit{expr}[x \leftarrow C])$$

Generally, in an assignment of the type $y := \mathit{expr}(x_1, \ldots, x_n)$ the variable on the left-hand-side (y) is *defined*, and the variables on the righ-hand-side ($x_1, \ldots, x_n$) are *used*. The expression above, which is written in a Cobalt-like syntax, states that if x is defined as the constant C, and x is not re-defined until it is used in an expression that defines y as an *expr* which depends on x, then in the definition of y, every reference to x can be safely replaced by the constant C.

```
1 x := 3;
2 y := 5;
3 z := x + 4;
```

Before

```
1 x := 3;
2 y := 5;
3 z := 7;
```

After

Fig. 1. Code before and after constant propagation

Consider the example in Fig. 1 where x is defined in statement 1, is not re-defined in statement 2, and is used in statement 3.

Using the expression above, at statement 3 we have:

$$\{\texttt{stmt1} :(x := 3)\};\ \neg \mathit{MayDef}(x))\ \textbf{Until}\ (y := x + 4\ \ \Rightarrow y := 7)$$

which justifies the correctness of the transformation in the figure.

CompCert [15]: CompCert is a verified compiler developed by Xavier Leroy and his colleagues. The goal of CompCert is to formally verify an optimizaing compiler whose input is in Clight (a subset of C) and whose output is PowerPC assembly code. Each optimization is verified by Coq [1]. In the terminology above, the code that translates each IR_{i+1} into IR_i is verified.

Most optimizations in CompCert are verified by defining a simulation relation, `match_state`, between the (symbolic) states of the code before and after the optimization. The online repository of CompCert[2] provides with numerous examples of verified optimizations, including that of constant propagation.

LLVM[3] (Low-Level Virtual Machine) is a relatively new open-source compiler infrastructure that is widely used. There has been considerable effort in verifying its optimizations. Similarly to GCC, LLVM's IR language uses Single Static Assignment (SSA) [2]. Unlike GCC, the static analysis of LLVM is not "centralized" but rather each optimization is in charge of performing the static analysis it needs. We now review several projects whose goal is to verify LLVM optimizations.

Vellvm [28]: Vellvm (verified LLVM) is a framework that is specific to LLVM's IR. It includes a formal semantics for the IR language, and a framework to reason about IR to IR transformation, all in Coq. Vellvm covers a wide spectrum of LLVM's IR, including heap operations and procedure calls. Vellvm was used in [29] to verify a variant of the `mem2reg` LLVM transformation that translate into the initial SSA and performs some register allocation. (It is interesting to note that Vellvm failed to cover the original `mem2reg`. We'll return to this point when discussing witnesses.)

Project Vellvm verifies LLVM optimizations in a method that is similar to CompCert. It handles optimizations in a way the is reminiscent of Cobalt: Each

[2] https://github.com/AbsInt/CompCert.

[3] https://llvm.org/.

optimization is divided into several micro-steps, that include, for example, a single instruction removal. Using program refinement, each the micro step is proved to be well-formed and to preserve the semantics of the source program. One then composes the proofs of the micro steps in order to obtain a proof for the correctness of the full transformation. (The next section contains formal definitions of refinement relation and compositionality.) Using a clever pipelining mechanism, Vellvm allows to re-cycle proofs, yet, these still to be manually generated in Coq.

Alive [16]: Alive is a domain-specific language that is suitable for writing program optimizations. Alive automatically verifies the transformation. Based on prior work ([26]), the authors of Alive found that the LLVM pass that combines instruction, InstCombine, has numerous bugs (this is a rather tricky pass that has many sub-cases and corner cases.) This InstCombine passed into Alive, and the tool detected numerous bugs, which were then shown to be true bugs in InstCombine (as opposed to modeling mistakes.) Alive creates VCs which is sends to Z3 SMT solver[4].

Figure 2 shows an example of an InstCombine optimization in the Alive syntax. The first two lines is the source code, and the last line its optimized target code. There, 32-bit integer x is shifted left, then right, 29 positions. The target replaces the two shifts by a logical `and` with the decimal constant 7.

```
%1 = shl i32 %x, 29
%2 = lshr i32 %1, 29
          ⟹
%2 = and i32 %x, 7
```

Fig. 2. An InstrCombine transformation written in Alive

Figure 3 shows the resulting query to Z3, which attempts to find an assignment for x such that when shifted left, then right, 29 positions is not equal to its bitwise `and` with the decimal 7. If Z3 cannot find such an x it reports failure, else it returns some x that is a counterexample to the correctness of the optimization.

```
(declare-const X (_BitVec 32))
(assert (not (=
  (bvlshr (bvshl X 29) 29)
  (bvand X 7)
)))
```

Fig. 3. Z3 query generated by Alive for the example optimization

[4] https://github.com/Z3Prover/z3/wiki.

2.2 Translation Validation

Recall that a compiler receives a *source program* written in some high-level language, translates it into an *Intermediate Representation (IR)*, and then applies a series of optimizations to the program – starting with classical architecture-independent *global* optimizations, and then architecture-dependent ones such as instruction scheduling. Typically, these optimizations are performed in several passes where each pass applies a certain type of optimization.

In order to prove that one code translates the other, we introduce of formal model for a system defined by a code so to give a common semantics to both. Here we follow the terminology in [31] and use the formalism of *Transition Systems* (TS's). The notion of a target code T being a correct implementation of a source code S is then defined in terms of *refinement*, stating that every computation of T corresponds to some computation of S with matching values of the corresponding variables.

The intermediate code is a three-address code, most often (in modern compilers) given in SSA . It is described by a *flow graph*, which is a graph representation of the three-address code. Each node in the flow graph represents a *basic block*, that is, a sequence of statements that is executed in its entirety and contains no branches. The edges of the graph represent the flow of control.

Transition Systems: In order to present the formal semantics of source and intermediate code we introduce *transition systems*, TS's, a variant of the *transition systems* of [21]. A *Transition System* $S = \langle V, \Omega, \Theta, \rho \rangle$ is a state machine consisting of:

- V a set of *state variables*,
- $\Omega \subseteq V$ a set of *observable variables*,
- Θ an *initial condition* characterizing the initial states of the system, and
- ρ a *transition relation*, relating a state to its possible successors.

The variables are typed, and a *state* of a TS is a type-consistent interpretation of the variables. For a state s and a variable $x \in V$, we denote by $s[x]$ the value that s assigns to x. The transition relation refers to both unprimed and primed versions of the variables, where the primed versions refer to the values of the variables in the successor states, while unprimed versions of variables refer to their value in the pre-transition state. Thus, e.g., the transition relation may include "$y' = y + 1$" to denote that the value of the variable y in the successor state is greater by one than its value in the old (pre-transition) state.

The observable variables are the variables we care about. When comparing two systems, we will require that the observable variables in the two systems match. We require that all variables whose values are printed by the program be identified as an observable variables. If desired, we can also include among the observables the history of external procedure calls for a selected set of procedures.

A computation of a TS is a maximal finite or infinite sequence of states $\sigma\colon s_0, s_1, \ldots$ starting with a state that satisfies the initial condition such that every two consecutive states are related by the transition relation.

A transition system is *deterministic* when the observable part of the initial condition uniquely determines the rest of the computation. We restrict our attention to deterministic transition systems and the programs that generate such systems. Thus, to simplify the presentation, we do not consider here programs whose behavior may depend on additional inputs that the program reads throughout the computation. It is straightforward to extend the theory and methods to such intermediate input-driven programs.

Let $P_S = \langle V_S, \Omega_S, \Theta_S, \rho_S \rangle$ and $P_T = \langle V_T, \Omega_T, \Theta_T, \rho_T \rangle$ be two TS's, to which we refer as the *source* and *target* TS's, respectively. Such two systems are called *comparable* if there exists a one-to-one correspondence between the observables of P_S and those of P_T. To simplify the notation, we denote by $X \in \Omega_S$ and $x \in \Omega_T$ the corresponding observables in the two systems. A source state s is defined to be *compatible* with the target state t, if s and t agree on their observable parts. That is, $s[X] = t[x]$ for every $x \in \Omega_T$. We say that P_T is a *correct translation* (*refinement*) of P_S if they are comparable and, for every $\sigma_T : t_0, t_1, \ldots$ a computation of P_T and every $\sigma_S : s_0, s_1, \ldots$ a computation of P_S such that s_0 is compatible with t_0, then σ_T is terminating (finite) iff σ_S is and, in the case of termination, their final states are compatible. Note that here the notion of compatible states implies agreeing on values of all observable variables.

The definition above seems to imply that observable variables should only match at the end of a computation. In fact, we want the output (assuming same input) and at times procedure calls to also match whether or not computations are terminating. We can therefore define a "stopping point" to be the prefix of any computation that ends with either a true termination, an output (write call), or even possibly a procedure call (for the latter, see discussion below). For all such, we consider the true termination or output (or even procedure call) as output-ing the values of all observables, and require that the target is compatible with the source upon those output point.

As for procedure calls, we have some latitude. If a procedure all in target appears in source, then all observable (which include the parameters) must match. However, at time (e.g., when inlining a procedure) the target may not include a procedure call. In such cases, one has to choose the observables and create artificial "termination points" so to be able to check the equality between values of observables. While this may seem tricky to do, in practice it is not, since one usually cares about the final values of variables rather than intermediate ones. To see this, consider, for example, a case of an observable variable X that stores some counter whose final value is N^2 for some input N computed by a loop that adds successive $2 \cdot i + 1$ to X, $i = 0, \ldots, N$. If X is output-ed after every iteration, then the only target that matches the source has to do same, and is therefore similar to the source (but for replacing $2 \cdot i + 1$ by adding 2 to the last incremented value.) If we only care about the final value of X, then there are many optimizations that may occur, and the only thing that matters is the *last* value of X. So, while X is observable, we only care about its value once set to (presumably) N^2, rather that about all its intermediate values before the final one is defined.

TVI [20]: Translation Validation Infrastructure (TVI) is the first project that implements translation validation for optimizing compilers. TVI is provided with a simulation relation of the form $(\mathsf{PC}_S, \mathsf{pc}_T, \alpha)$, where PC_S is a source location (basic block), pc_T is a target location, and α is a conjunctions of equalities that describe relations between variables in V_T and V_S.

Going back to the example of Fig. 1, we may view all the statements as if in the same basic block, say B1. There is a single exit from the block, assume it is to B2. There, the data mapping α may include

$$\Big(\mathtt{B1}, \mathtt{B1}, \bigwedge_{v \in \{x,y,z\}} v = V\Big)$$

(where the lower case variables are target ones and upper case are source ones) as well as

$$\Big(\mathtt{B2}, \mathtt{B2}, \bigwedge_{v \in \{x,y,z\}} v = V \ \wedge \ x = 3 \ \wedge \ y = 5 \ \wedge \ z = 7\Big)$$

TVI checks that if the simulation relation holds at the beginning of some simple path, it holds at its end.

TVI was implemented on GCC and was successful in validating numerous programs. It was the first true implementation of TV to a real-life optimizing compiler. It has two apparent weaknesses: For one, the simulation relations are manually generated. For another, the lack of invariants (which we'll see in TVOC) restricts TVI's power to optimizations that are completely order preserving. In particular, it cannot handle any code motion, including LICM.

TVOC [30]: TVOC, Translation Validation of Optimizing Compilers is a project that originated in NYU in the early 2000's and headed by Benjamin Goldberg, Amir Pnueli, Lenore Zuck, and later Clark Barrett joined the team and facilitated a direct connection from the VCs (Verification Conditions) produced by the tool to the theorem prover CVC [3]. Yi Fang was the chief architect of the project, and many other students contributed (including, Ying Hu, Ittai Balaban, and Ganna Zaks).

TVOC's history followed open-source compilers that were eventually closed-sourced, the last in the chain was Intel ORC. The philosophy of TVOC (well justified by the history of eventual closed-sourcing of initially open-source compilers) was that the tool doesn't have access to the compiler. This allowed to initially depend on information from the compiler (static analysis, information about optimizations performed) and eventually removing this dependence.

The main part of TVOC is that of global optimizations that are, more or less, structure preserving. Roughly speaking, these are optimizations that do not drastically change the ordering of statements. It does allow, for example, for a statement to move in the code (as in LICM), but so that its execution is moved back, or forth, more than a constant number of steps that is independent of the values of variables. The latter occurs when, for example, loops are interchanged or reversed.

Let $P_{_S} = \langle V_{_S}, \Omega_{_S}, \Theta_{_S}, \rho_{_S} \rangle$ and $P_{_T} = \langle V_{_T}, \Omega_{_T}, \Theta_{_T}, \rho_{_T} \rangle$ be comparable TS's, where $P_{_S}$ is the *source* and $P_{_T}$ is the *target*. In order to establish that $P_{_T}$ is a correct translation of $P_{_S}$ for the cases that the structure of $P_{_T}$ does not radically differ from the structure of $P_{_S}$, a proof rule, VALIDATE is applied [31]. The proof rule VALIDATE is inspired by the computational induction approach ([7]), originally introduced for proving properties of a single program, Rule VALIDATE provides a proof methodology by which one can prove that one program *refines* another. This is achieved by establishing a *control mapping* from target to source locations, a *data abstraction* mapping from source to target variables, and proving that these abstractions are maintained along basic execution paths of the target program.

The proof rule assumes each TS has a *cut-point set* CP. This is a set of blocks that includes the initial and terminal block, as well as at least one block from each of the cycles in the programs' control flow graph. A *simple path* is a path connecting two cut-points, and containing no other cut-point as an intermediate node. We assume that there is at most one simple path between every two cut-points. For each simple path leading from `Bi` to `Bj`, ρ_{ij} describes the transition relation between blocks `Bi` and `Bj`. Typically, such a transition relation contains the condition which enables this path to be traversed, and the data transformation effected by the path. Note that when the path from `Bi` to `Bj` passes through blocks that are not in the cut-point set, ρ_{ij} is a compressed transition relation that can be computed by the composition of the intermediate transition relation on the path from `Bi` to `Bj`.

The main proof rule of TVOC (which can be found in [31] with a soundness proof) calls for:

1. Control abstraction κ that maps target's control points to source ones, such that the initial and terminal blocks of target map into corresponding ones of source;
2. An invariant over target variables for each basic block of target;
3. A data abstraction α which is a conjunction of (1) statement stating that source location is at the κ-corresponding location of the target, (2) guarded expressions of the form $p \rightarrow V = e$ where p is a condition, V is an source variable, and e is an expression over target variables. It is required that for every initial target block `Bi`, $\Theta_{_T} \wedge \Theta_{_S} \rightarrow \alpha \wedge \varphi_i$, that is, that the conjunction of the initial conditions of the source and target implies α as well as the invariant at `Bi`, and, similarly, that for every observable variable $V \in \Omega_{_S}$ whose target counterpart is v and every terminal target block `B`, α implies that $V = v$;
4. For each pair of target basic blocks `Bi` and `Bj` such that there is a simple target path from `Bi` into `Bj` (that has no other cutpoint on but for its endpoints), construct a *verification condition* C_{ij} that asserts if the assertion φ_i and the data abstraction α hold before the transition, and the transition takes place, then after the transition there exist new source variables that reflect the corresponding transition in the source, and the data abstraction and the assertion φ_j hold in the new state. Hence, φ_i is used as a hypothesis at

the antecedent of the implication C_{ij}. In return, the validator also has to establish that φ_j holds after the transition. Thus, as part of the verification effort, TVOC confirms that the proposed assertions are indeed inductive and hold whenever the corresponding block is visited.

Following the generation of the verification conditions whose validity implies that the target T is a correct translation of the source program S, it only remains to check that these implications are indeed valid.

Using the example of Fig. 1, using `B1` and `B2` as before. The control abstraction then maps, for each $i = 1, 2$, the target `Bi` into the source `Bi`. There are no invariants at the entry to `B1`, that is, $\varphi_1 = \mathsf{true}$. (There will be, however, an invariant φ_2, namely $(x = 3) \ \wedge \ (y = 5) \ \wedge \ (z = 7)$.) If we follow the TVOC literature and denote source variables by upper cases and target ones by lower cases, we obtain the data mapping

$$\alpha : \ (\mathsf{PC} = \kappa(\mathsf{pc}) \ \wedge \ (\mathsf{pc} = 2 \ \rightarrow \ (X = x \ \wedge \ Y = y \ \wedge \ Z = z))$$

where PC (resp. pc) is the source (resp. target) program counter. The verification condition for the path from `B1` to `B2` is

$$\begin{aligned} C_{12} : \quad & (\mathsf{pc} = \mathsf{PC} = 1 \ \wedge \ \mathsf{pc}' = \mathsf{PC}' = 2 \wedge x' = 3 \ \wedge \ y' = 5 \ \wedge \ z' = 7 \ \wedge \\ & \qquad\qquad\qquad\qquad\qquad\qquad X' = 3 \ \wedge \ Y' = 5 \ \wedge \ Z' = X' + 3) \\ & \rightarrow \quad (\mathsf{PC}' = \kappa(\mathsf{pc}') \ \wedge \ X' = x' \ \wedge \ Y' = y' \ \wedge \ Z' = z') \end{aligned}$$

which is trivially true.

The approach makes sense only if this validation (as well as the preceding steps of the conditions' generation) can be done in a fully automatic manner with no user intervention. Indeed, as shown in [6], by performing its own static analysis, TVOC can often compute all that is needed (control mapping, invariants, data mapping, and verification conditions). At times there are several candidates for κ and α. Then the tool uses some heuristics to choose one, and it if fails, it may try others.

TVOC has a separate part to validate loop optimizations such as loop interchange, loop reversal, and tiling. Initially they were constructed using a file (`*.l`) that seemed to have been kept for ORC debugging purposes. Later, the dependence on this file was replaced by heuristics that guessed which loop optimizations were applied [9], using only the fact that (in ORC) loop optimizations followed global optimizations.

One should note the invariants of TVOC that gave it an additional power then previous methods. These invariants allowed TVOC to deal with what referred to above as "minor reordering" such as LICM. In fact, these invariants play a major role in the LLVM project which is the topic of the next section. In essence, they allow to carry information in between basic blocks. The more precise the invariants are, the more precise the static analysis is, which, in turn, allow for more aggressive optimizations.

TVOC, and tools similar to it that were developed at the time, did not deal with either pointer analysis (in particular, aliasing) or inter-procedural optimizations (such as tail recursion, inter-procedural constant propagation, or inlining).

Later [22], a framework for dealing with a certain type of inter-procedural optimization was developed by the creators of TVOC. Yet, the implementation [27] was not performed on the ORC (that, ironically, was no longer open source at the time) bur rather on LLVM.

3 TV for LLVM: Witnessing

As before, a program is described by a transition system $S = \langle V, \Omega, \Theta, \rho \rangle$. We assume that the CFG has a unique B basic block with no incoming edges such that $\Theta \rightarrow \mathsf{B}$, and a unique E basic block that has no outdoing edges. (Note that even while a code may have several termination nodes, one can connect them all to a single a E basic block so we lose no generality in assuming that there is a single E basic block.) All other basic blocks are *intermediate*. We assume that a program has no direct transition from B to E.

As in TVOC, one associates, with each basic block, a *generalized* transition relation, describing the effect of executing the block. Here it is assumed that the transition relation of a program is *complete*; that is, for every non-final state s, there is a state s' such that $\rho(s, s')$ holds. We also assume that the transition relation is *location-deterministic*, in that there is a at most one transition between any two locations. Formally, $[(\rho(s,s') \ \wedge \ \rho(s,s'') \ \wedge \ s'[\mathsf{pc}] = s''[\mathsf{pc}]) \ \Rightarrow \ s' = s'']$ (where pc is the location variable). This allows non-determinism in the sense of Dijkstra's **if-fi** and **do-od** constructs where multiple guards may be true at a state, since the successor states have different locations.

The notion of correct implementation ("program T (target) implements program S (source)") is just like before, only expressed directly as a simulation relation. More formally, fix a program $S = \langle V_S, \Omega_S, \Theta_S, \rho_S \rangle$ and $T = \langle V_T \Omega_T, \Theta_T, \rho_T \rangle$ and a relation $\preceq$ between T's and S's states. A T-state t *matches* an S-state s if $t \preceq s$. The definitions of path matching and system matching follow. As before, we require non-terminating computations of T to be matched to non-terminating computations of S so rules out pathological "implementations" where T does not terminate on any input.

One nice feature of the implementation notion is that it is *compositional*, that is, If T implements S and U implements T, then U implements S. This allows to seamlessly compose a sequence of transformation.

In practice, just like in TVOC, the matching relation is often a conjunction of equalities of the form $v_S = \mathcal{E}(V_T)$ where v_S is a source variable and $\mathcal{E}(V_T)$ is an expression of the target variables. When T is derived from S by a set of global optimizations, it often suffices to define $\preceq$ only for program counters that are at the beginning of a basic block and to reason only on simple paths (that include no cycles.) This is often insufficient for dealing with other transformations (for example, inter-procedural optimizations or loop optimizations) and other methods have to be used.

We refer to a "good" $\preceq$—one that allows to prove an implementation relation—as a *witness*. We outfit LLVM so thateach optimization pass with a source S and a target T, the pass produces its own witness to the correctness (implementation relation) of the optimization. Based on the compositionality property, if each pass has a witness, then so does that whole compilation.

3.1 Examples of Witnesses

Consider our example of constant propagation as described in Fig. 1, with B1 and B2 being the basic blocks as for the TVOC example. There, the transitions relations for source and programs are the same: $\mathsf{pc} = 1 \ \wedge \ \mathsf{pc}' = 2 \ \wedge \ x' = 3 \ \wedge \ y' = 5 \ \wedge \ z' = 7$ and the witness is the trivial $x = X \ \wedge \ y = Y \ \wedge \ z = Z$ (where we follow the convention that upper case denote source variables and lower case denote target variables.)

A slightly less trivial example is for the program described in Fig. 4. We show each program with its CFG denoting its B and E blocks.

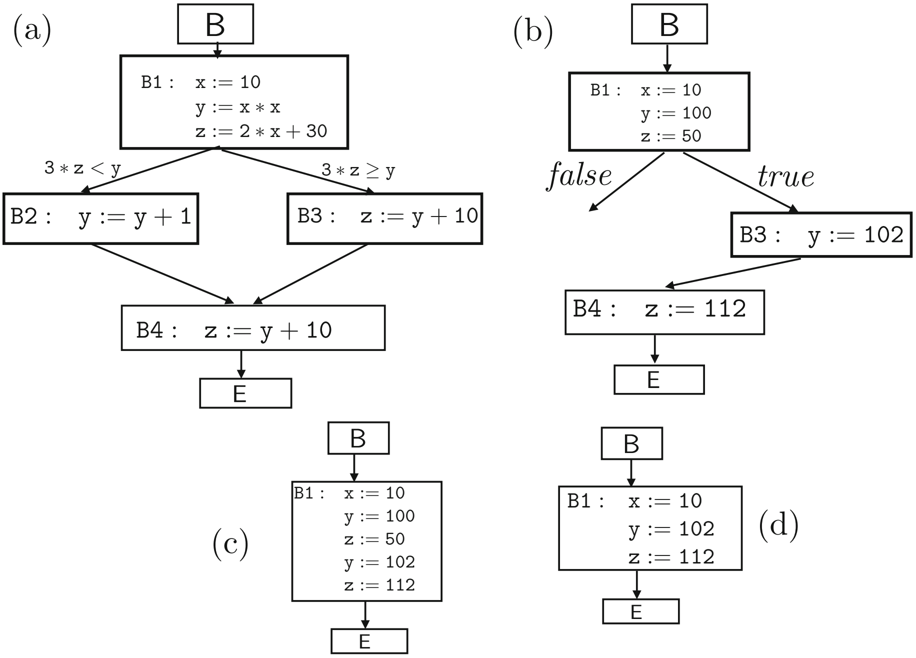


Fig. 4. A sequence of transformation

The first, (a), is the source program. The second (b) is the result of constant propagation and elimination of the resulting dead branch: since $z = 50$ and $y = 100$, $(150 =)3 \cdot z > y(= 100)$, and the condition on the left branch evaluates to false while the condition on the right branch evaluates to true, hence the left branch is never taken and can be eliminated (unreachable code followed by dead code elimination). Since basic blocks are only constrained by being single-entry single-exit, the basic blocks B1, B3, and B4 can now be merged into a single basic block (block merge), as shown in (c). Finally, the first assignments to y and z are never used, which renders them *dead* and they can be removed (dead store elimination.)

Each sub-step ((a) to (b), (b) to (c), and (c) to (d)) can be assigned a witness for each target location (basic block) in the obvious way. When composed we get a witness for the (a) to (d) transformation. E.g., at location E the witness is:

$$\mathsf{pc} = \mathsf{PC} = \mathsf{E} \ \wedge\ X = x \ \wedge\ Y = y \ \wedge\ Z = z \ \wedge\ x = 10 \ \wedge\ y = 102 \ \wedge\ z = 112$$

3.2 Witnesses vs. TVOC

The original goal of TV is to determine the equivalence of S and T. The methodology of doing that, which relies on heuristics, has become sufficiently complex so to merit its own verification. In fact, all known implementations of the methodology require much ingenuity and skill. The witness approach requires instrumenting each optimization pass. This instrumentation is rather simple. Ideally, it would be obtained from the designer (c is extraneous) of the optimization. In some cases it is possible to craft the instrumentation without deep understanding of the optimization. This instrumentation is in the form of small "footprints," from which a witness can be constructed fully automatically. All the global optimizations as well as the "simplify-CFG" ones were successfully performed by Master's level students with little experience in compilers. A notable example (not performed by a student) is the instrumentation of `mem2reg`. This optimization pass performs both translation to SSA and some register allocation. A validation of a variant of this transformation took over 18 man months, 15K Coq-lines ([4]), while using the witness theory, the creation of a witness for the original transformation took three man month (most of which spent on understanding the code) and about 300-500 LOC in OCaml and C++ [18]. It is well beyond the capabilities of TVOC.

Of course, TVOC and similar tools avoid instrumenting the compiler. Judging from the history at the time, when compilers were rarely left open source, this was probably the right approach in the early 2000s. Currently, however, compilers are often open source, and instrumenting it so to ease validation is no more a faux pas.

3.3 Implementation

The witness checking infrastructure for LLVM consists of two parts: *witness generation* and *refinement checking*. Once an optimization pass is instrumented (hopefully by its author, but usually by graduate students), a witness can be generated for the optimization. The current implementation assumes that both source and target codes are deterministic, that is, every state has a unique successor. Suppose a source S (some IR_i) and a target T (IR_{i+1}) with a witness relation $\preceq$. With the determinism assumption, the verification condition implied by $\preceq$ is:

$$t \preceq s \wedge \rho_T(t, t') \ \wedge\ \rho_S(s, s') \implies \quad t' \preceq s'$$

There are several tools that verify LLVM code against high-level specifications (see, e.g., [10,23]), as far as we know there are not tools that can check an LLVM IR program refines another.

We chain two existing tools to accomplish this check: Smack and Boogie. Smack [23] is ongoing project whose goal is to verify LLVM IR, and part of it is a translation into Boogie. Boogie [13] is verification language using Z3 at the backend. We input the source and target code into Smack, and obtain Boogie programs. These Boogie program, together with the witness relation $\preceq$, are then composed with proper variable renaming to guarantee mutually exclusive memory space. The composition is such that executions of matching simple paths of source and target are interleaved. Boogie then generates and checks the verification condition implied by the witness.

4 Conclusion

There is a growing awareness, both in industry and academia, of the crucial role of formally proving the correctness of safety-critical portions of systems. Most verification methods deal with the high-level specification of the system. However, if one is to prove that the high-level specification is correctly implemented at the lower level, one needs to verify the compiler which performs the translations. Verifying the correctness of modern optimizing compilers is challenging due to the complexity and reconfigurability of the target architectures and the sophisticated analysis and optimization algorithms used in the compilers.

The paper surveys some of the work of the recent two decades in verifying optimizing compilers. The first direction is to verify the compiler (translator). The most successful effort in this direction is CompCert that combines the development of the compiler with its verification.

Most compilers, however, are a given and not developed from scratch. Formally verifying a full-fledged optimizing compiler, as one would verify any other large program, is often infeasible, due to its size and evolution over time. *Translation validation* offers an alternative to the verification of translators in general and of compilers in particular. According to the *translation validation* approach, rather than verifying the compiler itself, one constructs a *validating tool* which, after every run of the compiler, formally confirms that the target code produced is a correct translation of the source program. In addition to providing a proof that the target code of the compiler implements the source code, the translation validation approach also offers means to *certify* that the code produced is true to its source. All TV methodologies output VCs, these can be verified by independent theorem provers (or, as is often the case, SMT solvers), which allows an additional degree of confidence. As an anecdote, SNECMA (currently Safran Aircraft Engines) used to employ hundreds of highly skilled people whose sole job was to manually check that optimized code correctly translated the source code. With TV there is no need for such manual verification.

The paper surveys some of the past work in translation validation and describes a current effort in providing with translation validation to LLVM. While we focused only on the global optimizations fragment of this effort, it was also applied to loop optimization [19]. We are currently applying a novel technique, that combines TV with re-writing rules, to validate inter-procedural optimizations. Yet another part of the witness theory that this paper omits for space reasons is that of *witness propagation*. This is pretty similar to the invariants of TVOC, only that the propagation mechanism allows to carry, from one transformation to another, any information that is known, as well as to constantly update this information as optimizations passes are executed. The idea of propagation can be used in numerous ways. To date, we augmented LLVM with external program analysis tools (whose runtime is far from linear!) and propagated the resulting witnesses as to accomplish more efficient runtime checks such as ones for buffer and integer overflows. The results are very promising. In fact, they allow for what used to be "unscalable" runtime check to be highly scalable [8].

It should be noted that all methodologies described here, in spite of presumably attempting to decide an undecidable problem, accomplish their task in practice and incur a very small overhead.

Acknowledgements. We thank DARPA and NSF for funding this project. Thanks are also due to our numerous collaborators on this project throughout the years, and especially for Amir Pnueli who introduced Lenore Zuck to the area, as well as Kedar Namjoshi and Venkat Venkatakrishnan who have been close collaborators of hers on the LLVM project.

References

1. Coq development team. The Coq proof assistant. https://coq.inria.fr/
2. Alpern, B., Wegman, M.N., Zadeck, F.K.: Detecting equality of variables in programs. In: POPL 1988, pp. 1–11. ACM, New York (1988)
3. Barrett, C., Berezin, S.: CVC lite: a new implementation of the cooperating validity checker. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 515–518. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27813-9_49
4. Barthe, G., Demange, D., Pichardie, D.: Formal verification of an SSA-based middle-end for CompCert. TOPLAS **36**(1), 4:1–4:35 (2014)
5. Dave, M.A.: Compiler verification: a bibliography. SIGSOFT SEN **28**(6), 2 (2003)
6. Fang, Y., Zuck, L.D.: Improved invariant generation for TVOC. ENTCS **176**(3), 21–35 (2007)
7. Floyd, R.: Assigning meanings to programs. Proc. Symp. Appl. Math. **19**, 19–32 (1967)
8. Gjomemo, R., Namjoshi, K.S., Phung, P.H., Venkatakrishnan, V.N., Zuck, L.D.: From verification to optimizations. In: DSouza, D., Lal, A., Larsen, K.G. (eds.) VMCAI 2015. LNCS, vol. 8931, pp. 300–317. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46081-8_17
9. Goldberg, B., Zuck, L., Barrett, C.: Into the loops: practical issues in translation validation for optimizing compilers. ENTCS **132**(1), 53–71 (2005)
10. Gurfinkel, A., Kahsai, T., Komuravelli, A., Navas, J.A.: The seahorn verification framework. In: CAV, pp. 343–361 (2015)

11. Hunt Jr., W.A., Kaufmann, M., Moore, J.S., Slobodova, A.: Industrial hardware and software verification with ACL2. Philos. Trans. R. Soc. **375**, 40 (2017). (Article Number 20150399)
12. Le, V., Sun, C., Su, Z.: Randomized stress-testing of link-time optimizers. In: ISSTA, pp. 327–337. ACM(2015)
13. Leino, K.R.M.: This is boogie 2. Manuscript KRML **178**, 131 (2008)
14. Lerner, S., Millstein, T., Chambers, C.: Automatically proving the correctness of compiler optimizations. ACM SIGPLAN Not. **38**(5), 220–231 (2003)
15. Leroy, X.: Formal verification of a realistic compiler. Commun. ACM **52**(7), 107–115 (2009)
16. Lopes, N.P., Menendez, D., Nagarakatte, S., Regehr, J.: Provably correct peephole optimizations with alive. ACM SIGPLAN Not. **50**(6), 22–32 (2015)
17. McCarthy, J., Painter, J.: Correctness of a compiler for arithmetic expressions. Math. Aspects Comput. Sci. **1**, 219–222 (1967)
18. Namjoshi, K.S.: Witnessing an SSA transformation. In: VeriSure Workshop and Personal Communication, CAV 2014 (2014). http://ect.bell-labs.com/who/knamjoshi/papers/Namjoshi-VeriSure-CAV-2014.pdf
19. Namjoshi, K.S., Singhania, N.: Loopy: programmable and formally verified loop transformations. In: Rival, X. (ed.) SAS 2016. LNCS, vol. 9837, pp. 383–402. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53413-7_19
20. Necula, G.C.: Translation validation for an optimizing compiler. ACM Sigplan Not. **35**(5), 83–94 (2000)
21. Pnueli, A., Siegel, M., Singerman, E.: Translation validation. In: Steffen, B. (ed.) TACAS 1998. LNCS, vol. 1384, pp. 151–166. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054170
22. Pnueli, A., Zaks, A.: Translation validation of interprocedural optimizations. In: International Workshop on Software Verification and Validation (2006)
23. Rakamarić, Z., Emmi, M.: SMACK: decoupling source language details from verifier implementations. In: Biere, A., Bloem, R. (eds.) CAV 2014. LNCS, vol. 8559, pp. 106–113. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08867-9_7
24. Samet, H.: Automatically proving the correctness of translations involving optimized code. PhD thesis, Stanford University (1975)
25. Schmidt, D.A.: Data flow analysis is model checking of abstract interpretations. In: POPL (1998), pp. 38–48. ACM (1998)
26. Yang, X., Chen, Y., Eide, E., Regehr, J.: Finding and understanding bugs in C compilers. ACM SIGPLAN Not. **46**(6), 283–294 (2011)
27. Zaks, G.: Ensuring correctness of compiled code. Ph.D. thesis, New York University (2009)
28. Zhao, J., Nagarakatte, S., Martin, M.M.K., Zdancewic, S.: Formalizing the LLVM intermediate representation for verified program transformations. In: ACM SIGPLAN Notices, pp. 427–440. ACM (2012)
29. Zhao, J., Nagarakatte, S., Martin, M.M.K., Zdancewic, S.: Formal verification of SSA-based optimizations for LLVM. ACM SIGPLAN Not. **48**(6), 175–186 (2013)
30. Zuck, L., Pnueli, A., Goldberg, B., Barrett, C., Fang, Y., Hu, Y.: Translation and run-time validation of loop transformations. FMSD **27**(3), 335–360 (2005)
31. Zuck, L.D., Pnueli, A., Goldberg, B.: VOC: a methodology for the translation validation of optimizing compilers. J. UCS **9**(3), 223–247 (2003)

Security and Privacy

Security Analysis of EMV Protocol and Approaches for Strengthening It

Khedkar Shrikrishna[1], N. V. Narendra Kumar[2](✉), and R. K. Shyamasundar[1]

[1] Department of Computer Science and Engineering,
Indian Institute of Technology Bombay, Mumbai, India
shrikrishnakhedkar1@gmail.com, shyamasundar@gmail.com
[2] Centre for Payment Systems, IDRBT, Hyderabad, India
naren.nelabhotla@gmail.com

Abstract. Reliance on smart cards for our daily lives makes their security essential. Credit card fraud has been a major hassle for electronic commerce over the past few years. A worldwide standard for payment has been introduced by Europay, Mastercard, and Visa (EMV) with the objective of limiting the card payment frauds. The EMV standard has two main pillars, card authentication (chip) - counters skimming and counterfeiting frauds, and cardholder verification (PIN) - counters stolen or lost cards fraud. Today EMV (aka Chip-and-PIN) is the leading system for the card payments worldwide with more than 4.8 billion cards. Although EMV cards are widely adopted around the world, it is still amenable to attacks as our analysis reveals.

In this paper, we present an approach for analyzing the security of the EMV protocol using a novel information security model called the Readers-Writers Flow Model (RWFM) that explicitly captures the intentions of the protocol designer. An assessment of security of the EMV protocol by the approach automatically reveals several attacks on the EMV protocol presented in the literature, and provides implementation guidelines for realizing a secure EMV protocol w.r.t different threat models. It is experimentally illustrated that most of these attacks are overcome by using a RWFM wrapper in a prototype implementation following the guidelines. Efficacy of the approach is demonstrated by successfully preventing the software simulation of the "No-PIN" attack.

Keywords: EMV Chip-and-PIN cards · Secure information-flow
Payment protocols

1 Introduction

For quite some time, we had been using the magnetic stripe card. Whenever the magnetic stripe cards is swiped in the terminal, it captures all the data contained in the card for processing. However, by inserting simple hardware between the card and the terminal, one can easily clone the magnetic stripe card by copying its data. Thus, magnetic stripe cards are unsafe. To overcome these problems,

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 69–85, 2018.
https://doi.org/10.1007/978-3-319-72344-0_4

cards containing microprocessors capable of performing simple computations have been introduced.

EMV[1] or chip-and-PIN is a standard for communication between chip supported cards and terminals. EMV contact cards have been introduced to reduce the damages due to exponential growth in skimming attacks on the magnetic stripe cards. Chip cards use fixed commands to communicate with the terminal. In EMV, chip prevents the attacker from card counterfeiting, and PIN prevents him from using lost or stolen cards. EMV card contains a chip as well as a magnetic stripe for backward compatibility.

EMV chip cards are considered more secure than the magnetic stripe cards. However, the chip cards are proven to be vulnerable to some attacks, such as transaction can be performed without the knowledge of the PIN [14], cloning the card with pre-play attack [5], the relay attack [8] etc. Further, in the EMV protocol, most of the data exchanged between the card and the terminal is in plaintext, which enables an attacker to easily collect data for performing online-usage attacks, or cloning magnetic stripe cards. Recently in October 2015, an organised hacker group from France realized the no-PIN attack with almost invisible hardware [13].

Several researchers have formally analyzed the EMV protocol [1,5,8,14,20,21] and demonstrated that it is prone to attacks. Some of these researchers have also suggested modifications to the protocol for overcoming the specific attacks they have identified. Our goal has been to assess the possibility of protecting the chip card based transactions without deviating from the protocol structure very much. Towards this end, we labelled (using RWFM model [15,17]) the initial data involved in the transaction, and used the RWFM model for automatically deriving the labels of the messages being exchanged between the various parties in the transaction.

This labelled protocol clearly identifies the owner, permissible readers and influencers of information, thus, enabling us to analyze its security w.r.t different threat models by tracking the confidentiality, integrity as well as subject authorities. For example, the label of the PDOL data sent by card to the terminal is $(C, \{C, B, T\}, \{C, B, T\})$, clearly suggesting that the card has created it, and only the card, bank and the terminal can read it. Our analysis immediately identifies that exchanging data in plain text (this is the case as per the EMV protocol) ensures security only in the case where the terminal is trusted to be free of malware (software and/or hardware like skimmers). From another perspective, the terminal also needs to authenticate the card involved in the transaction. Thus, our approach presents a practical model which provides implementations tuned to different threat models. Further, the labelled transaction thus derived also adheres to several principles for good protocol design like the principle of full information [22] and canonical intensional specification [19].

[1] The EMV standard is designed by Europay, MasterCard, and Visa.

Main contributions of the paper include:

- a novel general approach to analyze the security of the EMV protocol,
- implementation of EMV protocol through our approach that prevents the "No-PIN" attack, and
- general recommendations to be followed for realizing secure EMV transactions w.r.t different threat models, since the EMV protocol is here to stay. Note that our approach is general and applies to other protocols also equally well.

The rest of the paper is organized as follows: Sect. 2 presents an overview of the EMV and the RWFM model. In Sects. 3 and 4, we present our approach, its advantages and an implementation. Section 5 discusses the comparison of our approach with some relevant literature, and Sect. 6 provides concluding remarks.

2 Background

In this section, we present an overview of the EMV transaction[2] and the information security model Readers-Writers Flow Model (RWFM).

2.1 Flow of EMV Transaction

Each successful transaction of EMV contact card has four phases: initialization, card authentication, cardholder verification and transaction authorization.

Initialization phase. The goal of this phase is that the right application is selected and that all mandatory information is exchanged from card to terminal to determine how to proceed with the further steps.

The protocol starts by selecting the payment applications supported by cards such as credit or debit or ATM. The card provides the supported payment applications to the terminal and optionally provides Processing Options Data Object list (PDOL), that specifies the data which card wants from the terminal for subsequent phases.

The card then also provides its Application Interchange Profile (AIP) and the Application File Locator (AFL) [21]. AIP indicates the card supported features for card authentication and cardholder verification. AFL is a list of files which are used for currently selected application.

Card Authentication Methods (CAM). The goal of this phase is that the card authenticates its genuineness and which bank issued the card to the terminal. EMV has three different card authentication methods: Static Data Authentication (SDA), Dynamic Data Authentication (DDA), and Combined Data Authentication (CDA). CDA always has the highest priority, followed DDA and finally SDA. The method that is supported by both the terminal and the card with the highest priority is selected. In this phase, only the card is authenticated to the terminal but the terminal is not authenticated to the card.

[2] http://www.emvco.com.

Cardholder Verification Methods (CVM). The goal of this phase is to verify the identity of the cardholder, so that stolen or lost cards cannot be easily used. The terminal chooses which CVM to use based on the CVM rules provided by the card. Cardholder verification methods supported by EMV are:

1. Online PIN: bank checks the PIN.
2. Offline plaintext PIN: chip card checks the PIN transmitted in clear.
3. Offline encrypted PIN: chip card checks the PIN sent in encrypted form.
4. Handwritten signature.
5. None.

Transaction Authorization. For a transaction, the card generates one or two cryptograms, one in the case of an offline transaction and two in the case of an online transaction.

1. In an offline transaction the card provides a proof (Transaction Certificate (TC)) to the terminal that a transaction took place, which the terminal sends to the issuer later.
2. In an online transaction the card first provides an Authorisation Request Cryptogram (ARQC) which the terminal forwards to the issuer for approval. If the card receives approval, the card then provides a Transaction Certificate (TC) as a proof that the transaction has been completed.

Complete details on the EMV standard and its transaction flow can be found in [9–12].

2.2 Readers-Writers Flow Model (RWFM)

The Readers-Writers Flow Model (RWFM) proposed in [15,16] is a novel model for information flow control. RWFM is obtained by recasting the Denning's label model [7], and has a label structure that: (i) explicitly captures the readers and writers of information, (ii) makes the semantics of labels explicit, and (iii) immediately provides an intuition for its position in the lattice flow policy.

Recasting Procedure. Given a Denning's lattice model $DFM = (S, O, SC, \oplus, \leqslant)$ with flow policy $\lambda : S \cup O \rightarrow SC$, we recast the labels in terms of the readers and writers to obtain an equivalent flow policy defined by $DFM_1 = (S, O, SC_1, \oplus_1, \leqslant_1)$ and $\lambda_1 : S \cup O \rightarrow SC_1$, where: (i) $SC_1 = 2^S \times 2^S$, (ii) $\oplus_1 = (\cap, \cup)$, (iii) $\leqslant_1 = (\supseteq, \subseteq)$, and (iv) $\lambda_1(e) = (\{s \in S \mid \lambda(e) \leqslant \lambda(s)\}, \{s \in S \mid \lambda(s) \leqslant \lambda(e)\})$, where e is a subject or object.

RWFM is obtained by generalizing the above recasting procedure, and is defined as follows:

Definition 1 (Readers-Writers Flow Model (RWFM)). *Readers-writers flow model is defined as the eight tuple* $(S, O, SC, \leqslant, \oplus, \otimes, \top, \bot)$, *where*

S and O are the set of subjects and objects in the system,
$SC = S \times 2^S \times 2^S$ is the set of labels,

$\leqslant = (-, \supseteq, \subseteq)$ *is the permissible flows ordering,*
$\oplus = (-, \cap, \cup)$ *and* $\otimes = (-, \cup, \cap)$ *are the join and meet operators respectively, and*
$\top = (-, \emptyset, S)$ *and* $\bot = (-, S, \emptyset)$ *are respectively the maximum and minimum elements in the lattice.*

The first component of a security label in RWFM is to be interpreted as the owner of information, the second component as the set of readers, and the third component as the set of influencers. Note that RWFM is fully defined in terms of S, the set of subjects in the information system.

Note that the first component in the label is introduced only to facilitate limited discretionary flows (downgrades), and has no impact on the permissible information flows, or joins and meets. Therefore, we have abused notation in the above definition for simplicity, by uniformly blanking out the first component of the label.

Note that in RWFM information flows upwards in the lattice as readers decrease and writers increase.

Property of the recasting procedure: *RWFM is a complete model, w.r.t Denning's lattice model, for studying information flows in a system.*

The recasting procedure presented at the beginning of this section actually constructs such an equivalent RWFM policy for a given Denning's policy.

RWFM Semantics of Secure Information Flow. RWFM provides a state transition semantics of secure information flow, which presents significant advantages and preserves useful invariants that aid in establishing that the system is secure or not misusing information. In the following, we present the RWFM semantics.

Let S denote the set of all the subjects in the system. RWFM follows a floating-label approach for subjects, with labels $(s, S, \{s\})$ and $(s, \{s\}, S)$ denoting the "**default label**" - the label below which a subject cannot write, and "**clearance**" - the label above which a subject cannot read, for a subject s respectively. Object labels are fixed and are provided by the desired policy at the time of their creation.

Definition 2 (State of Information System). *State of an information system is defined as the set of current subjects and objects in the system together with their current labels.*

Next, we describe the permissible state transitions of an information system, considering the primitive operations that cause information flows. The operations that are of interest are: (i) subject reads an object, (ii) subject writes an object, (iii) subject downgrades an object, and (iv) subject creates a new object. We believe that these operations are complete for studying information flows in a system. Note that we consider the set of subjects as fixed, and hence no operations for creation of new subjects.

For each of the above operations, we describe the conditions under which it is safe (causes only permissible information flows) and hence can be permitted.

Note that when a subject s requests a new session, system assigns $(s, S, \{s\})$ as its label.

READ Rule *Subject s with label (s_1, R_1, W_1) requests read access to an object o with label (s_2, R_2, W_2).*

```
If ($s \in R_2$) then
    change the label of $s$ to $(s_1, R_1 \cap R_2, W_1 \cup W_2)$
    ALLOW
Else
    DENY
```

WRITE Rule *Subject s with label (s_1, R_1, W_1) requests write access to an object o with label (s_2, R_2, W_2).*

```
If ($s \in W_2 \wedge R_1 \supseteq R_2 \wedge W_1 \subseteq W_2$) then
    ALLOW
Else
    DENY
```

DOWNGRADE Rule *Subject s with label (s_1, R_1, W_1) requests to downgrade an object o from its current label (s_2, R_2, W_2) to (s_3, R_3, W_3).*

```
If ($s \in R_2 \wedge s_1 = s_2 = s_3 \wedge R_1 = R_2 \wedge W_1 = W_2 = W_3 \wedge R_2 \subseteq R_3 \wedge$
($W_1 = \{s_1\} \vee (R_3 - R_2 \subseteq W_2)$)) then
    ALLOW
Else
    DENY
```

Intuitively, downgrading is allowed only by the owner at the same label as the information being downgraded, and (i) unrestricted addition of readers if he is the only influencer of information, or (ii) additional readers restricted to the set of stakeholders that contributed to the computation.

CREATE Rule *Subject s labelled (s_1, R_1, W_1) requests to create an object o.*
Create a new object o, label it as (s_1, R_1, W_1) and add it to the set of objects O.

Given an initial set of objects on a lattice, the above transition system accurately computes the labels for the newly created information at various stages of the transaction/workflow.

The transition system above satisfies the following invariants that are handy to establish flow security:

1. subject and object labels float upwards only,
2. for a subject s, $A(s) = s$, $s \in R(s)$, and $s \in W(s)$,
3. the set of writers of information is always accurately maintained (exactly the set of subjects that influenced the information content), this plays a vital role in forensics and audit,
4. label of newly created objects precisely reflects the circumstances under which they are created, and
5. downgrade rule is within the boundaries of the flows permissible under a given transaction.

3 RWFM Dynamic Labelling for EMV Transactions

In this section, we describe the EMV protocol for a specific choice of parameters, label the initial data in the protocol, derive the complete labelled protocol using the initial labels, and discuss the security properties implied by the labels.

3.1 EMV Protocol

We describe the EMV protocol where static data authentication (SDA), offline plaintext PIN verification, and online transaction authorization are the chosen methods. Note that, in SDA the terminal authenticates the card using the data received in the initialization phase. Therefore, in the protocol given below, there will not be any messages in the card authentication phase.

Initialization
M_1 T → C : Select application
M_2 C → T : PDOL
M_3 T → C : Get Processing Options
M_4 C → T : AIP, AFL
Cardholder verification
M_5 T → U : Get PIN
M_6 U → T : Enters PIN
M_7 T → C : PIN verify
M_8 C → T : PIN correct
Transaction authorization
M_9 T → C : N_t
M_{10} C → T : ARQC=(ATC,IAD,MAC(N_t,ATC,IAD))
M_{11} T → B : N_t, ARQC
M_{12} B → T : ARC, ARPC
M_{13} T → C : ARC, ARPC
M_{14} C → T : TC=(ATC,IAD,MAC(ARC,N_t,ATC,IAD))
M_{15} T → B : TC

For simplicity, we assume that all the records needed by the terminal are included in M_4 itself. In particular, the CVM list is a part of M_4.

3.2 Labelled EMV Protocol

Generally, in any protocol, when a step of the form "$X \rightarrow Y : M$" is specified, it intends to capture the following:

- Initial data and their intended usage i.e., which principals can access/modify which data.
- Principal X constructed the message M from his current knowledge, for the purpose of sharing it with Y.
- It is intended that M can be read by Y only.

- If a component of M appears in a subsequent step in the protocol, it is intended that it is also readable by the receiver in the step.
- When Y receives the message, he must be able to authenticate that the message is indeed constructed and sent by the intended party (X), and is in the form specified by the protocol.
- Y should not use the message M or its components, in the current session or in any other session, in ways other than those specified by the protocol.

Based on the discussion above, intentions for the EMV protocol can be derived systematically. For example, M_4 is to be interpreted as saying: when the card sends processing options to the terminal, it is for the use of the terminal only and it is to be used in this session only. The terminal is not supposed to retain these details and use them later.

Now, we derive labels for EMV protocol described in the previous section, to capture the protocol designers' intentions automatically from the initial set of subjects and objects together with their respective labels. In the EMV protocol, there are 4 subjects, namely, the bank (B), user (U), card (C) and the terminal (T). Initial objects in the EMV protocol are: PDOL, AIP, AFL, PIN, ATC, IAD, and the shared key (SK) between the card and the bank.

Initial state of the system
$\mathrm{PDOL}^{(B,\{B,C\},\{B\})}$, $\mathrm{AIP}^{(B,\{B,C\},\{B\})}$, $\mathrm{AFL}^{(B,\{B,C\},\{B\})}$, $\mathrm{PIN}^{(C,\{B,C\},\{B,C,U\})}$, $\mathrm{ATC}^{(C,\{C\},\{C,U\})}$, $\mathrm{IAD}^{(C,\{C\},\{C\})}$, $\mathrm{SK}^{(B,\{B,C\},\{B\})}$, $\mathrm{C}^{(C,\{B,C,T,U\},\{C\})}$, $\mathrm{B}^{(B,\{B,C,T,U\},\{B\})}$, $\mathrm{T}^{(T,\{B,C,T,U\},\{T\})}$, $\mathrm{U}^{(U,\{B,C,T,U\},\{U\})}$.

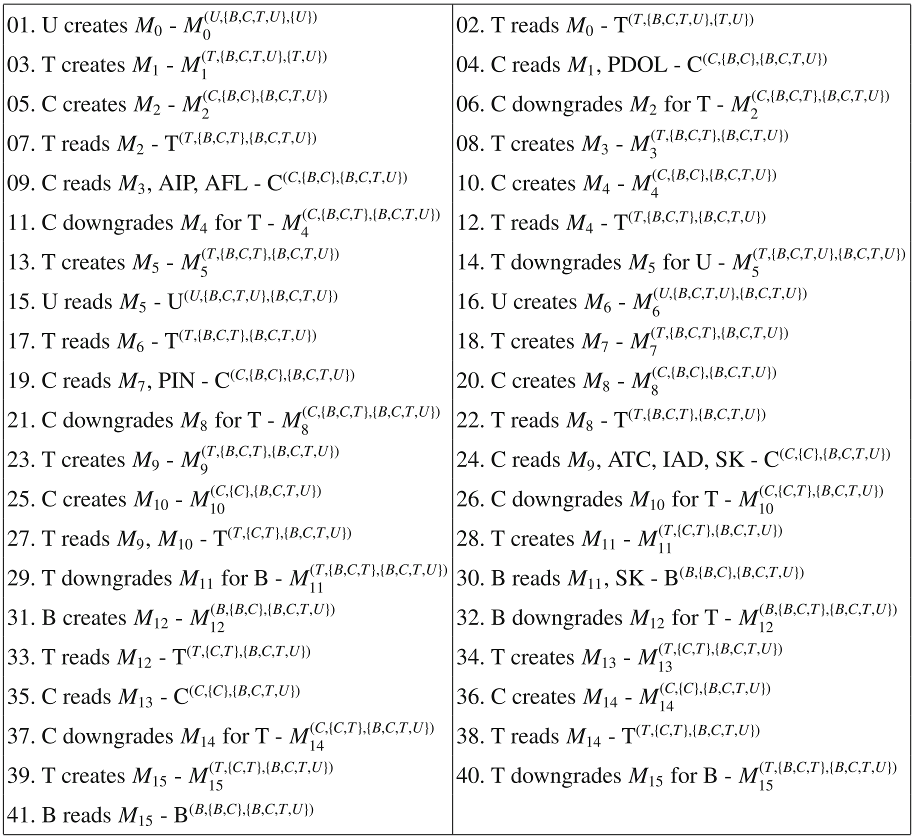
PDOL, AIP, AFL, and SK are created by the bank and stored on the card at the time of its' issue. Therefore, they have the label $(B, \{B, C\}, \{B\})$ indicating that they have been created by the bank, readable only by the bank and the card, and influenced only by the bank. PIN is readable only by the bank and the card, and has been influenced by bank, card and the user. Note that this is so because when a user changes his PIN, he influences the PIN, also because the bank has to authenticate the user before a PIN change, it would have influenced, and finally the card stores the PIN in an encrypted form, so it also influences the PIN. Similarly, the labels for the other initial objects can be understood.

In addition to the EMV protocol described above, we consider a message M_0 at the very beginning, where the user inserts the card into the terminal. Labelled EMV protocol is depicted in Fig. 1.

Information-flow diagrams (IFD) provide a visual representation of the state transitions in the labelled protocol, and serve as simple yet fully formal models of the labelled protocol. IFD for a portion of the EMV transaction where the terminal gets the PIN verified by the card is depicted in Fig. 2.

In Fig. 2, we have only represented those subjects and objects in the initial state that are of relevance to the portion of the transaction for which the IFD is provided. Changes to the state after each transition are highlighted in bold. In the last state, T is highlighted in the readers set of M_8 to depict that the object has been successfully downgraded.

The labelled EMV protocol and the IFD derived above explicitly specify the security requirements of each of the messages exchanged in the protocol.



01. U creates M_0 - $M_0^{(U,\{B,C,T,U\},\{U\})}$	02. T reads M_0 - $T^{(T,\{B,C,T,U\},\{T,U\})}$
03. T creates M_1 - $M_1^{(T,\{B,C,T,U\},\{T,U\})}$	04. C reads M_1, PDOL - $C^{(C,\{B,C\},\{B,C,T,U\})}$
05. C creates M_2 - $M_2^{(C,\{B,C\},\{B,C,T,U\})}$	06. C downgrades M_2 for T - $M_2^{(C,\{B,C,T\},\{B,C,T,U\})}$
07. T reads M_2 - $T^{(T,\{B,C,T\},\{B,C,T,U\})}$	08. T creates M_3 - $M_3^{(T,\{B,C,T\},\{B,C,T,U\})}$
09. C reads M_3, AIP, AFL - $C^{(C,\{B,C\},\{B,C,T,U\})}$	10. C creates M_4 - $M_4^{(C,\{B,C\},\{B,C,T,U\})}$
11. C downgrades M_4 for T - $M_4^{(C,\{B,C,T\},\{B,C,T,U\})}$	12. T reads M_4 - $T^{(T,\{B,C,T\},\{B,C,T,U\})}$
13. T creates M_5 - $M_5^{(T,\{B,C,T\},\{B,C,T,U\})}$	14. T downgrades M_5 for U - $M_5^{(T,\{B,C,T,U\},\{B,C,T,U\})}$
15. U reads M_5 - $U^{(U,\{B,C,T,U\},\{B,C,T,U\})}$	16. U creates M_6 - $M_6^{(U,\{B,C,T,U\},\{B,C,T,U\})}$
17. T reads M_6 - $T^{(T,\{B,C,T\},\{B,C,T,U\})}$	18. T creates M_7 - $M_7^{(T,\{B,C,T\},\{B,C,T,U\})}$
19. C reads M_7, PIN - $C^{(C,\{B,C\},\{B,C,T,U\})}$	20. C creates M_8 - $M_8^{(C,\{B,C\},\{B,C,T,U\})}$
21. C downgrades M_8 for T - $M_8^{(C,\{B,C,T\},\{B,C,T,U\})}$	22. T reads M_8 - $T^{(T,\{B,C,T\},\{B,C,T,U\})}$
23. T creates M_9 - $M_9^{(T,\{B,C,T\},\{B,C,T,U\})}$	24. C reads M_9, ATC, IAD, SK - $C^{(C,\{C\},\{B,C,T,U\})}$
25. C creates M_{10} - $M_{10}^{(C,\{C\},\{B,C,T,U\})}$	26. C downgrades M_{10} for T - $M_{10}^{(C,\{C,T\},\{B,C,T,U\})}$
27. T reads M_9, M_{10} - $T^{(T,\{C,T\},\{B,C,T,U\})}$	28. T creates M_{11} - $M_{11}^{(T,\{C,T\},\{B,C,T,U\})}$
29. T downgrades M_{11} for B - $M_{11}^{(T,\{B,C,T\},\{B,C,T,U\})}$	30. B reads M_{11}, SK - $B^{(B,\{B,C\},\{B,C,T,U\})}$
31. B creates M_{12} - $M_{12}^{(B,\{B,C\},\{B,C,T,U\})}$	32. B downgrades M_{12} for T - $M_{12}^{(B,\{B,C,T\},\{B,C,T,U\})}$
33. T reads M_{12} - $T^{(T,\{C,T\},\{B,C,T,U\})}$	34. T creates M_{13} - $M_{13}^{(T,\{C,T\},\{B,C,T,U\})}$
35. C reads M_{13} - $C^{(C,\{C\},\{B,C,T,U\})}$	36. C creates M_{14} - $M_{14}^{(C,\{C\},\{B,C,T,U\})}$
37. C downgrades M_{14} for T - $M_{14}^{(C,\{C,T\},\{B,C,T,U\})}$	38. T reads M_{14} - $T^{(T,\{C,T\},\{B,C,T,U\})}$
39. T creates M_{15} - $M_{15}^{(T,\{C,T\},\{B,C,T,U\})}$	40. T downgrades M_{15} for B - $M_{15}^{(T,\{B,C,T\},\{B,C,T,U\})}$
41. B reads M_{15} - $B^{(B,\{B,C\},\{B,C,T,U\})}$	

Fig. 1. Labelled EMV protocol

Security Properties Implied by the Labelling. Consider the message M_4 in which the card provides its preferences to the terminal. Label of M_4 specifies the following security requirements: (i) authenticity: it is generated by the card, (ii) confidentiality: it can only be read by the bank, and the card, and (iii) integrity: it has been influenced only by the stakeholders of the computation. However, note that the terminal should also be able to read this message for the proper working of the protocol. Therefore the card downgrades the message and adds T as a reader - this is allowed in RWFM because T has influenced the message.

Similar interpretations also hold for the other messages of the protocol like cardholder verification (M_8) and transaction authorization (M_{10}). A security assessment of the EMV protocol based on the security properties specified by the labels reveals several potential security issues in it.

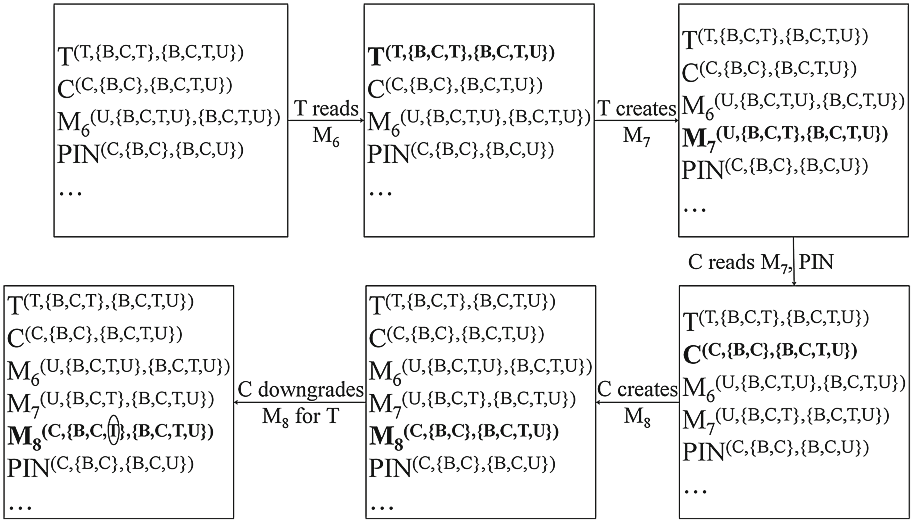


Fig. 2. Information-flow diagram for a portion of the EMV protocol

Security Interpretation/Assessment of EMV Protocol based on the Labelling

- [Cloning] In the EMV protocol most of the messages are exchanged in plain text - no constraint on readers. Therefore, any subject in the system can read these messages, not only the intended recipient. This leads to the possibility of cloning the card if an attacker captures the necessary information. This attack has been demonstrated in [2].
- [Replay/Modifications] Since the messages exchanged are in plaintext, the recipient has no guarantee that the message has been generated and sent by the intended sender. This leads to the possibility of an attacker either modifying certain contents of a message or replaying an old message that he had captured. Replay attack has been demonstrated in [6]. CVM downgrade attack [2] where the terminal is forced to choose offline PIN verification, the Chip-and-PIN is broken attack [14] where the PIN correct command is sent by the attacker, and the relay attack [8] where the attacker is relaying messages from a fake terminal are examples of attacks realized by modifying the transmitted message. The Chip-and-skim attack [5] in which an attacker captures a transaction certificate (TC) for a pre-selected random number and uses it for authorizing another transaction involves both replaying the TC, and modifying the random number in the message.

This information in combination with the threat model and trust specifications aids in identifying suitable mechanisms for realizing the desired security.

Implementation Guidelines

- If the terminal cannot be fully trusted (could be infected with malware and/or contains skimmers), then the labels immediately suggest that messages need to be encrypted for preserving security.
- When the messages are decrypted by the terminal for processing, temporary data stored on its RAM also needs to be protected from unwanted access and modifications. This can be easily ensured by implementing an RWFM monitor because the intermediate data gets labelled automatically.
- The decision of whether (and how) to respond to a message received is very much dependant on the current state of the receiver. Labels provide a coarse-grained notion of state, which becomes important in deciding further actions.

Thus, our approach provides a general unified technique to specify, analyze/assess, and derive implementations tuned to different threat models assuring truly end-to-end security for the EMV protocol.

4 Implementation of EMV via RWFM and Assessment of Its' Security

To validate our approach, we have implemented a RWFM wrapper on the software implementation[3] of EMV terminal. In this section, we discuss the software simulation of the Chip-and-PIN is broken attack [14], and its prevention.

4.1 Simulation of the "Chip-and-PIN Is Definitely Broken" Attack

Murdoch et al. [14] have shown that it is possible to use a stolen or lost smart card without the knowledge of its PIN (https://www.youtube.com/watch?v=Wh7W8Dn2PsA). The attack assumes that the card is configured with offline PIN verification method as its highest priority. Subsequently, most issuers have changed the default to online PIN verification. However, a man-in-the-middle can still force the terminal to choose offline PIN verification by modifying the CVM list sent by the card as it is in plaintext. This attack has also been practically demonstrated by Barisani et al. [2].

Since the main objective of this paper is to demonstrate how our approach prevents attacks on the EMV protocol by labelling it, we have worked with a software simulation instead of on the actual hardware. We illustrate this with a simulation of the Chip-and-PIN is definitely broken attack [2] which combines the Chip-and-PIN is broken attack [14] with CVM downgrade attack [2].

Experimental Setup. The following main hardware and software components are needed to perform this attack:

- **Hardware:** Chip card, Smart card reader, and Laptop/PC.
- **Software:** Terminal, and man-in-the-middle (MitM).

[3] https://sites.uclouvain.be/EMV-CAP/resources/Data/EMVCAP-1.4.tar.gz.

For the terminal, we have used the software implementation by Jean-Pierre Szikora and Philippe Teuwen (available from https://sites.uclouvain.be/EMV-CAP/), and we have implemented our own program for MitM. In line with the practical considerations, our experimental setup is such that all the communication between the card and the terminal goes via the MitM. The MitM just acts as a proxy for all the messages exchanged other than CVM list and PIN verify.

Experiment steps

1. Insert the chip card into the smart card reader and connect it to the PC through a USB port.
2. [CMV downgrading] When the card sends the CVM list to the terminal, the MitM changes the CVM list and forces the terminal to perform offline PIN verification.
3. [PIN spoofing] When the terminal sends the PIN verify command to the card, MitM suppresses it and replies with the PIN correct command.

Since these attacks are widely discussed in the literature, and on various forums on the web, we do not provide much more details and refer the interested reader to [2,14].

4.2 Experimental Evaluation of Our Approach

To realize the labelled protocol derived in Sect. 3.2, we have implemented the following:

- a library for RWFM functionality including managing and manipulating labels, and providing access decisions,
- a program for simulating the actions of the card - this is needed because we do not have control over the actual card for labelling it, and
- modified terminal and MitM programs integrated with the appropriate RWFM function calls.

In this setup, RWFM library monitors the actions of all the stakeholders in the system and permits data accesses only as per the labels. Initially, RWFM library is loaded with the list of initial objects and their labels. Subsequently, as each operation is performed, RWFM library automatically labels the subjects and the messages being exchanged.

When we retried the simulation of the Chip-and-PIN is definitely broken attack on the labelled protocol, it provides several layers of defence. For exploring all the possible defences provided by our approach, the implementation only displays error messages whenever an unauthorized access is about to happen, but does not prevent it.

1. The CVM list (M_4) sent from the card to the terminal is labelled $(C, \{B, C, T\}, \{B, C, T, U\})$. When the MitM tries to read this message, RWFM library logs an error because MitM is not a valid reader of the message as per its label.

2. MitM modifies the CVM list for sending to the terminal, which will be labelled $(M, \{B, C, T\}, \{B, C, M, T, U\})$. When the terminal receives this message, RWFM library identifies that the message is created by MitM (M) and is also influenced by it, and since this is not the expected label for this message, RWFM logs an error message. Notice that in this step we actually have two defences - one based on the owner and the other based on the set of influencers.
3. PIN verify command (M_7) sent from the terminal to the card is labelled $(T, \{B, C, T\}, \{B, C, T, U\})$. When the MitM tries to read this message, RWFM library logs an error because MitM is not a valid reader of the message as per its label.
4. MitM sends PIN correct command to the terminal, which will be labelled $(M, \{B, C, T\}, \{B, C, M, T, U\})$. When the terminal receives this message, RWFM library identifies that the message is created by MitM (M) and is also influenced by it, and since this is not the expected label for this message, RWFM logs an error message. In this step we actually have two defences - one based on the owner and the other based on the set of influencers.

Having demonstrated how our approach prevents the Chip-and-PIN is definitely broken attack, we now discuss how it overcomes several other attacks discovered in the literature.

1. Skimming attacks [2] and the Chip-and-skim attack [5] are prevented for the same reasons as discussed in the above example.
2. PoS RAM scraper attacks [18] are prevented because RWFM also labels the temporary data in the RAM and provides access controls to it.
3. Relay attacks [8] can be prevented by forcing the stakeholders to interact only with others that are RWFM enabled (can be verified by a simple certificate - provides a root of trust).
4. Replay attacks are prevented to an extent by RWFM because the messages captured can only be replayed in another session between the same parties.

The above discussion is summarized in the table below:

Attack	RWFM[a]	Suggestion given in the literature
Chip-and-PIN is broken [14]	✓	
CVM Downgrade [2]	✓	
Replay [6]		Nonce
Skimming [2]	✓	
Chip-and-skim [5]	✓	Random number generator algorithm
PoS RAM Scraper [18]	✓	
Relay [8]	✓	Distance bounding

[a] In the table above, by RWFM we mean an apt implementation of RWFM labels that realizes the security implied by the labels.

We have demonstrated that our approach based on the dynamic labelling using RWFM provides a uniform solution to many of the attacks discovered on the EMV protocol. Further, the labelled protocol also keeps track of the transaction history (coarse-grained), thus conforming to well-established principles for secure protocol design such as:

- The principle of full information [22]: in every outgoing message, participants must include all of the information they have gathered so far in the exchange,
- Canonical intensional specification [19]: no participant can believe a protocol run has completed unless a correct series of messages has occurred up to and including the last message the given participant communicates.

5 Related Work

Literature on the EMV protocol has three types of research: (i) attacks on EMV, (ii) analysis of the EMV protocol (high-level), and (iii) analysis of the cryptographic primitives of EMV protocol. We present a brief overview of only the first two types of research since they are directly connected to the paper.

No-PIN attack [14]**:** A man in the middle device which intercepts and modifies the communications between the card and the terminal, tricks the terminal into believing that PIN verification succeeded. The card believes that the terminal has either skipped cardholder verification or used a signature instead.

Chip and Skim [5]**:** The attacker records an ARQC for a transaction with nonce N, and presents it to a terminal that actually generated the nonce N1. The terminal sends the ARQC along with N1 (plain text) to the bank. The attacker changes N1 to N and the transaction succeeds.

CVM downgrade attack [2]**:** CVM list is sent by the smart card to the terminal which chooses the highest priority method that is supported by both of them. But the CVM list is sent in plaintext, thereby allowing an attacker to modify it to force the terminal to choose offline PIN verification method. Further, most of the data exchanged between the smart card and the terminal is in plaintext including the card number, expiry date, cardholder name etc. This data is enough to perform online attacks. It has been established that after the introduction of EMV, attackers moved towards the card not present transactions.

Some of the other attacks presented in literature include [6,8,18].

A formal analysis of the EMV protocol is presented in [20,21]. The authors modelled the EMV protocol in F#, and presented its analysis using ProVerif [4] and FS2PV [3]. The formalisation covers all the major options of the EMV protocol suite. The following are the verification results using their formal model: (i) the private asymmetric keys and the shared symmetric keys remain confidential, (ii) if the terminal successfully performs a card authentication, it should be the highest card authentication method supported by both the card and the terminal, (iii) in the case of DDA, if a terminal completes an authentication, the corresponding card is in fact involved, (iv) in the case of SDA, a terminal may complete an authentication without the corresponding card being involved, (v) if the customer is authenticated using his PIN, the terminal and card need not agree on whether the PIN was accepted, (vi) using SDA or DDA, if a transaction is successfully completed by the terminal, the corresponding card need not agree on having the transaction completed successfully, and (vii) using DDA, the card and terminal should agree on the result of the transaction.

Our observations on the related work are summarized as follows:

Realistically speaking, very few protocols, get formally verified w.r.t their specifications. Even assuming that the protocol is verified, it is essentially a verification of the model and not that of the actual protocol. In the context of financial protocols such as EMV, the added disadvantage is that the threat models and attackers evolve rapidly. Thus, it is imperative to derive an analysis technique parameterized on threat models relative to the designers' intentions. We can say that our approach realizes the following characteristics that would benefit a security analysis of the EMV protocol:

- [Attacks] The attacks are usually discovered accidentally; or possibly during the testing phase by extensive trial and error with deep intuitions for a possible issue (that arises in similar cases). Thus, it is desirable to provide an analysis based on the protocol designers' intentions that can be effectively used under different resource constraints or underlying threat models. Our analysis using the RWFM does exactly this by generating the underlying IFD that highlights the constraints that need to be observed for a secure realization. In a sense, our analysis is incremental, and allows different implementations under different threat models.
- [Formal Modelling and Analysis] The IFD generated by our approach is quite small compared to the analysis reported in [20, 21] – that consists of five pages of F# model. Furthermore, how one integrates the security requirements with F# model is not clear. Our approach essentially provides a complete transition system clearly highlighting the capabilities of the subjects and objects and hence, any formal model can be used to check for the properties in the transition system.

6 Conclusions

In this paper, we have analyzed the EMV protocol from a security perspective, keeping in view the intentions of the designer. Our analysis not only brings to light the attacks that have been found in the literature, but also demonstrates that while implementing the protocol if one keeps track of the functional requirements of read and write by the stakeholders, it brings out the constraints that should be satisfied for realization in a given threat model. Thus, if the architecture of implementation does not satisfy these constraints then the implementation will not guarantee the expected security. Further, it was also demonstrated how the design can be realized using the RWFM security model to avoid "No-PIN" attack. In summary, the work brings to light the need of analysis techniques that keep track of the constraints for satisfying the functionality of the protocol, and the constraints that needs to be satisfied for realizing security policy. Such an explicit bookkeeping aids in possible realizations for different threat models and architectures. In particular, our approach leads to a general implementation that works even if the regular card was replaced by another payment mechanism such as a mobile or e-wallet. We further demonstrated how the RWFM security model leads to analysis techniques that satisfy these criteria. Further, work on linking the system with automatic tools for establishing higher level properties is in progress.

Acknowledgement. The work was done as part of Information Security Research and Development Centre (ISRDC) at IIT Bombay, funded by MEITY, Government of India.

References

1. Adida, B., Bond, M., Clulow, J., Lin, A., Murdoch, S., Anderson, R., Rivest, R.: Phish and chips. In: Christianson, B., Crispo, B., Malcolm, J.A., Roe, M. (eds.) Security Protocols 2006. LNCS, vol. 5087, pp. 40–48. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04904-0_7
2. Barisani, A., Bianco, D.: Practical EMV PIN interception and fraud detection. In: 31th Chaos Communication Congress [31c3] of the Chaos Computer Club [CCC] (2014)
3. Bhargavan, K., Fournet, C., Gordon, A.D., Tse, S.: Verified interoperable implementations of security protocols. In: 19th IEEE CSFW, pp. 139–152 (2006)
4. Blanchet, B.: An efficient cryptographic protocol verifier based on prolog rules. In: 14th IEEE CSFW, pp. 82–96 (2001)
5. Bond, M., Choudary, O., Murdoch, S.J., Skorobogatov, S.P., Anderson, R.J.: Chip and skim: cloning EMV cards with the pre-play attack. CoRR abs/1209.2531 (2012)
6. Degabriele, J.P., Lehmann, A., Paterson, K.G., Smart, N.P., Strefler, M.: On the joint security of encryption and signature in EMV. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 116–135. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_8
7. Denning, D.E.: A lattice model of secure information flow. Commun. ACM **19**(5), 236–243 (1976). http://doi.acm.org/10.1145/360051.360056
8. Drimer, S., Murdoch, S.J.: Keep your enemies close: Distance bounding against smartcard relay attacks. In: Provos, N. (ed.) 16th USENIX Security Symposium. USENIX Association (2007)
9. EMVCo: Book 1: Application independent ICC to terminal interface requirements v4.3 (2011). http://www.emvco.com
10. EMVCo: Book 2: Security and key management v4.3 (2011). http://www.emvco.com
11. EMVCo: Book 3: Application specification v4.3 (2011). http://www.emvco.com
12. EMVCo: Book 4: Cardholder, attendant, and acquirer interface requirements v4.3 (2011). http://www.emvco.com
13. Ferradi, H., Géraud, R., Naccache, D., Tria, A.: When organized crime applies academic results: a forensic analysis of an in-card listening device. J. Crypt. Eng. **6**(1), 49–59 (2016)
14. Murdoch, S.J., Drimer, S., Anderson, R.J., Bond, M.: Chip and PIN is broken. In: 31st IEEE S&P, pp. 433–446. IEEE Computer Society (2010)

15. Narendra Kumar, N.V., Shyamasundar, R.K.: Realizing purpose-based privacy policies succinctly via information-flow labels. In: 4th IEEE BDCloud, pp. 753–760. IEEE (2014)
16. Narendra Kumar, N.V., Shyamasundar, R.K.: POSTER: dynamic labelling for analyzing security protocols. In: 22nd ACM CCS, pp. 1665–1667 (2015)
17. Narendra Kumar, N.V., Shyamasundar, R.K.: Analyzing protocol security through information-flow control. In: Krishnan, P., Radha Krishna, P., Parida, L. (eds.) ICDCIT 2017. LNCS, vol. 10109, pp. 159–171. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-50472-8_13
18. Rodríguez, R.J.: Evolution and characterization of point-of-sale RAM scraping malware. J. Comput. Virol. Hacking Tech. **13**(3), 179–192 (2017). https://doi.org/10.1007/s11416-016-0280-4
19. Roscoe, A.W.: Intensional specifications of security protocols. In: 9th IEEE CSF, pp. 28–38 (1996)
20. de Ruiter, J.: Lessons learned in the analysis of the EMV and TLS security protocols. Ph.D. thesis, Radboud University Nijmegen, August 2015
21. de Ruiter, J., Poll, E.: Formal analysis of the EMV protocol suite. In: Mödersheim, S., Palamidessi, C. (eds.) TOSCA 2011. LNCS, vol. 6993, pp. 113–129. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27375-9_7
22. Woo, T.Y.C., Lam, S.S.: A lesson on authentication protocol design. SIGOPS Oper. Syst. Rev. **28**(3), 24–37 (1994)

Secure Synthesis of IoT via Readers-Writers Flow Model

Shashank Khobragade[1], N. V. Narendra Kumar[2(✉)], and R. K. Shyamasundar[1]

[1] Department of Computer Science and Engineering,
Indian Institute of Technology Bombay, Mumbai, India
kshashank17@gmail.com, shyamasundar@gmail.com
[2] Centre for Payment Systems, IDRBT, Hyderabad, India
naren.nelabhotla@gmail.com

Abstract. Internet of Things (IoT) is a game changer for the connected society. Safe and reliable operation of IoT connected devices is of paramount importance and thus, security and privacy is a foundational enabler for IoT. In this paper, we arrive at a synthesis methodology for the IoT and demonstrate how information flow among the connected devices using a three tier architecture enables us to assess the required security and privacy of the IoT based on the given security and privacy capabilities of the components. Our methodology uses a recent information security model called RWFM (Readers-Writers Flow Model) and shows how flexible approaches of synthesis of IoT through frameworks like Django can be integrated to realize the security/privacy requirements of the IoT. We demonstrate how the methodology concretely enables us to derive the constraints to be satisfied by the underlying components and enabled communications. A case study of a healthcare IoT implementation is discussed to illustrate the advantages of the methodology.

Keywords: Secure IoT · Information-flow control · Privacy

1 Introduction

Internet of things (IoT) is transforming the world of remote access, with the tremendous control it provides. The potential to access and control anything from anywhere has attracted researchers and investors alike. IoT can be implemented not just to reduce human labour but to provide some life saving solutions. For example, IoT for healthcare can alert a doctor if a diabetic patient goes into severe hypoglycemia, or diabetic shock, enabling the doctor to provide treatment without any delay, as negligible time is spent on conveying the critical status of the patient. Technology has always made human life easier, but seldom comes a concept so colossal that possesses the potential to impact the daily human life, but this revolutionizing technology has also raised eyebrows in the security community.

Since IoT is essentially a synthesis of components to realize a desired goal, it is important that we understand the causal relationship of the components

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 86–104, 2018.
https://doi.org/10.1007/978-3-319-72344-0_5

with respect to one another. Thus, there is a need to assess the security of each individual component and realize the desired structure of their composition to preserve the security and privacy of IoT. Some of the important security challenges in IoT that can be addressed by a correct structure of composition are: scalability, device authentication, wireless security, access controls, data confidentiality, integrity, and privacy [10]. In this paper, our endeavour is to develop a unified model for a secure IoT that addresses most of the fundamental security requirements mentioned above.

In this paper, we propose an approach to synthesize a secure IoT from the given set of components and their intended interactions. Our approach to the security of IoT is based on tracking information flows in the system, and enforcing that these are valid with respect to the intended security of the system. In particular, we use the Readers-Writers Flow Model [13,15] (RWFM) to dynamically label the transactions of the IoT, and derive the constraints to be satisfied by the various components and their interactions to preserve the desired security and privacy requirements. Further, we demonstrate how flexible approaches to synthesis of secure IoT can be achieved through frameworks like Django for realizing the security/privacy requirements. We also demonstrate a prototype implementation of the approach, considering a healthcare IoT system as an example.

The rest of the paper is organised as follows: Sect. 2 provides a background on IoT and the security model RWFM. Sections 3, 4 and 5 explain the proposed architecture, the security analysis of illustrative examples, and a prototype implementation of the architecture. Section 6 provides comparison of our approach with some prominent approaches in the literature, and Sect. 7 concludes the paper.

2 Background

In this section, we present an overview of the Internet of Things and the Readers-Writers Flow Model.

2.1 Internet of Things (IoT)

Internet-of-Things (IoT) is generally a system of devices with firmware or software and network connectivity. Essentially the “things”, can be anything that can be connected to internet - it can be a toaster to a pacemaker. The internet connectivity enables the ability to control the devices remotely.

The essence of the IoT lies within its architectural capability to provide remote access to all kinds of devices. These devices can be anything that can be assigned an IP, with networking functionality. If the entity itself does not have a networking capability, then it can be achieved by means of a plug-and-play device, RFID tags, bio-chips, etc. Thus, IoT is a network of devices with sensing, computing, and networking capabilities.

Experts estimate [8] that the IoT will consist of almost 50 billion objects by 2020. However, for IoT to achieve its full impact many issues need to be resolved like: scalability, unique identification address, security, privacy etc.

IoT has invaded the society in a variety of ways some of which are: *Environment* [7], *Energy* [26], *Human life* [1,22], *Manufacturing* [2,4], and *Healthcare* [19].

Some of the notable implementations that have turned out to be commercially successful are: smart grid [6], wearable devices [11], smart thermostats like Nest [21], smart meters, and smart light[1]. While IoT has become ubiquitous in a welcoming way, one worrying question is: "what will happen to the privacy of our data"?

2.2 Readers-Writers Flow Model (RWFM)

In this section, we provide a brief overview of the Readers-Writers Flow Model (RWFM) [13,14] which is a novel model for information flow control. RWFM is obtained by recasting the Denning's label model, and has a label structure that: (i) explicitly captures the readers and writers of information, (ii) makes the semantics of labels explicit, and (iii) immediately provides an intuition for its position in the lattice flow policy.

Recasting Procedure. Given a Denning's lattice model $DFM = (S, O, SC, \oplus, \leqslant)$ with flow policy $\lambda : S \cup O \rightarrow SC$, we recast the labels in terms of the readers and writers to obtain an equivalent flow policy defined by $DFM_1 = (S, O, SC_1, \oplus_1, \leqslant_1)$ and $\lambda_1 : S \cup O \rightarrow SC_1$, where:

(i) $SC_1 = 2^S \times 2^S$,
(ii) $\oplus_1 = (\cap, \cup)$,
(iii) $\leqslant_1 = (\supseteq, \subseteq)$, and
(iv) $\lambda_1(e) = (\{s \in S \mid \lambda(e) \leqslant \lambda(s)\}, \{s \in S \mid \lambda(s) \leqslant \lambda(e)\})$, where e is a subject or object.

In the new label system, we use $R(e)$ and $W(e)$ to denote the first (Readers) and second (Writers) components of the label assigned to an entity e respectively.

Next, we illustrate the recasting of Denning's policy with the help of an example.

Example 1. *Consider the two-point lattice $\{l_1, l_2\}$, with $l_1 < l_2$. Let s_1 and s_2 be the only subjects in the system i.e., $S = \{s_1, s_2\}$. Similarly, let $O = \{o_1, o_2\}$. Consider the policy λ_1: $\lambda_1(s_1) = \lambda_1(o_1) = l_1$ and $\lambda_1(s_2) = \lambda_1(o_2) = l_2$. $s_1 \in R(o_1)$ because $\lambda_1(o_1) \leqslant \lambda_1(s_1)$ reduces to $l_1 \leqslant l_1$ which is true. $s_2 \in R(o_1)$ because $\lambda_1(o_1) \leqslant \lambda_1(s_2)$ reduces to $l_1 \leqslant l_2$ which is also true. Therefore $R(o_1) = \{s_1, s_2\}$. Similarly, we can derive the following labels on objects: $R(o_2) = \{s_2\}$, $W(o_1) = \{s_1\}$ and $W(o_2) = \{s_1, s_2\}$. The labels for subjects are as below: $R(s_1) = \{s_1, s_2\}$, $R(s_2) = \{s_2\}$, $W(s_1) = \{s_1\}$ and $W(s_2) = \{s_1, s_2\}$.*

The original and the inferred policies are depicted in Fig. 1. □

[1] http://www2.meethue.com/en-in/.

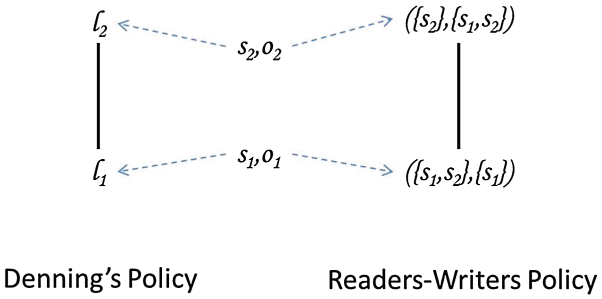


Fig. 1. Denning's policy and corresponding readers-writers policy inferred in Example 1

RWFM is obtained by generalizing the recasting procedure as follows:

Definition 1 (Readers-Writers Flow Model (RWFM)). *Readers-writers flow model is defined as the eight tuple* $(S, O, SC, \leqslant, \oplus, \otimes, \top, \bot)$*, where*
S *and* O *are the set of subjects and objects in the system,*
$SC = S \times 2^S \times 2^S$ *is the set of labels,*
$\leqslant = (-, \supseteq, \subseteq)$ *is the permissible flows ordering,*
$\oplus = (-, \cap, \cup)$ *and* $\otimes = (-, \cup, \cap)$ *are the join and meet operators respectively, and*
$\top = (-, \emptyset, S)$ *and* $\bot = (-, S, \emptyset)$ *are respectively the maximum and minimum elements in the lattice.*

The first component of a security label in RWFM is to be interpreted as the owner of information, the second component as the set of readers, and the third component as the set of influencers. Note that RWFM is fully defined in terms of S, the set of subjects in the system.

Note that the first component in the label is introduced only to facilitate additional flows (downgrades), and has no impact on the permissible information flows, or joins and meets. Therefore, we have abused notation in the above definition by uniformly blanking out the first component of the label.

Note That in RWFM Information Flows Upwards in the Lattice as Readers Decrease and Writers Increase

Theorem 1 (Completeness). *RWFM is a complete model, w.r.to Denning's lattice model, for studying information flows in an information system.*

The recasting procedure presented at the beginning of this section actually constructs such an equivalent RWFM policy for a given Denning's policy.

RWFM Semantics of Secure Information Flow. RWFM provides a state transition semantics of secure information flow, which presents significant advantages and preserves useful invariants that aid in establishing that the system is secure or not misusing information. In the following, we present the RWFM semantics.

Let S denote the set of all the subjects in the system. RWFM follows a floating-label approach for subjects, with labels $(s, S, \{s\})$ and $(s, \{s\}, S)$ denoting the "**default label**" - the label below which a subject cannot write, and "**clearance**" - the label above which a subject cannot read, for a subject s respectively. Object labels are fixed and are initially provided by the desired policy.

Definition 2 (State of Information System). *State of an information system is defined as the set of current subjects and objects in the system together with their current labels.*

Next, we describe the permissible state transitions of an information system, considering the primitive operations that cause information flows. The operations that are of interest are: (i) subject reads an object, (ii) subject writes an object, (iii) subject downgrades an object, and (iv) subject creates a new object. We believe that these operations are complete for studying information flows in a system. Note that we consider the set of subjects as fixed, and hence no operations for creation of new subjects.

For each of the above operations, we describe the conditions under which it is safe (causes only permissible information flows) and hence can be permitted. Note that when a subject s requests a new session, its label is set to $(s, S, \{s\})$.

READ Rule *Subject s with label (s_1, R_1, W_1) requests read access to an object o with label (s_2, R_2, W_2).*

```
If ($s \in R_2$) then
    change the label of $s$ to $(s_1, R_1 \cap R_2, W_1 \cup W_2)$
    ALLOW
Else
    DENY
```

WRITE Rule *Subject s with label (s_1, R_1, W_1) requests write access to an object o with label (s_2, R_2, W_2).*

```
If ($s \in W_2 \wedge R_1 \supseteq R_2 \wedge W_1 \subseteq W_2$) then
    ALLOW
Else
    DENY
```

DOWNGRADE Rule *Subject s with label (s_1, R_1, W_1) requests to downgrade an object o from its current label (s_2, R_2, W_2) to (s_3, R_3, W_3).*

```
If ($s \in R_2 \wedge s_1 = s_2 = s_3 \wedge R_1 = R_2 \wedge W_1 = W_2 = W_3 \wedge R_2 \subseteq R_3 \wedge$
($W_1 = \{s_1\} \vee (R_3 - R_2 \subseteq W_2)$)) then
    ALLOW
Else
    DENY
```

Intuitively, downgrading is allowed only by the owner at the same label as the information being downgraded, and (i) unrestricted addition of readers if he is the only influencer of information, or (ii) additional readers restricted to the set of stakeholders contributed to the computation.

CREATE Rule. *Subject s labelled (s_1, R_1, W_1) requests to create an object o.* Create a new object o, label it as (s_1, R_1, W_1) and add it to the set of objects O.

Given an initial set of objects on a lattice, the above transition system accurately computes the labels for the newly created information at various stages of the transaction.

The transition system above satisfies the following invariants that are handy to establish flow security:

1. subject labels float upwards only,
2. for a subject s, $A(s) = s$, $s \in R(s)$, and $s \in W(s)$,
3. the set of writers of information is always accurately maintained (exactly the set of subjects that influenced the information content), this plays a vital role in forensics and audit,
4. label of newly created objects precisely reflects the circumstances under which it is created, and
5. downgrade rule is within the boundaries of the flows permissible under a given transaction.

3 Securing IoT via RWFM

In this section, we describe our approach to securing IoT through RWFM.

3.1 System Architecture

Some of the main steps in realizing an IoT are:

1. Identify the various components like sensors, monitors, actuators, etc., that carry out the basic functionalities.
2. Realize the computing goal by composing various components by networking them – possibly hierarchically if needed. This can be refined to clearly define the interaction among the various components.
3. The interaction among the components clearly defines the reading/writing capability of the components and thus can be used to control the flow of information in the system. These explicit interactions define clearly the functionality of the system and also enables us to assess the confidentiality of the underlying interactions and computations. For instance, in a healthcare monitoring system, it is extremely important to be sure of the source of information before prescribing a medical procedure.

In summary, for a smart IoT system, realizing its functionality as well as security (confidentiality/privacy) issues of interactions are very interrelated. In the following we propose a three tier architecture and a method of specifying the IoT transitions.

Three-Tier Architecture. The "things" of the IoT are quite often sensors running on low energy and possessing low computational power. A control and desired functionality can not be achieved if it requires high computation cost. A strong cryptographic suite can not be employed on the sensor due to the same shortcoming. In order to tackle this problem, our architecture uses a gateway that can act as a communicating agent between the sensors in a Private Area Network (PAN) and the computational entity. The gateway thus has control over sensor data, can employ lightweight computing suite, detect impersonated sensors etc. The middleware on the computing entity may be used to employ any computation on the data so collected from sensors.

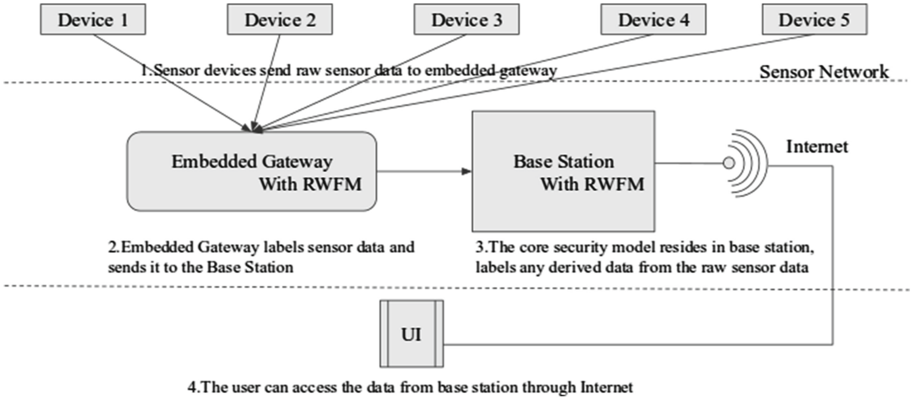

Device 1
Device 2
Device 3
Device 4
Device 5
1.Sensor devices send raw sensor data to embedded gateway
Sensor Network
Embedded Gateway With RWFM
Base Station With RWFM
Internet
2.Embedded Gateway labels sensor data and sends it to the Base Station
3.The core security model resides in base station, labels any derived data from the raw sensor data
UI
4.The user can access the data from base station through Internet

Fig. 2. System architecture for secure IoT

We propose a three tier architecture depicted in Fig. 2:

1. the first tier comprises of sensor devices in a private area network,
2. the second tier consisting of middleware - gateway and base station,
3. the third-tier has the user interface from which the user senses and controls the smart system as desired,
4. provides the needed secure communication between gateway and devices,
5. gateway consolidates the sensor data and sends it to the base station.

Note that the base station with higher computing power can be used to implement the computation on the data and also provides a gateway for users to access the data based upon the requirements of the application.

The crux of the approach to realize security lies in

1. Tracking the flow of information in the system and analysing any deviation from the intended usage.
2. Tracking the readers and writers for each of the object flowing between entities or principals.

3. Given the initial labels of the "things", all the labels will be generated for all the interactions in the IoT automatically by RWFM. In other words, we would have a complete labelled protocol diagram of the IoT (becomes clear in the sequel).

With above labelled protocol diagram of the IoT, the appropriate authentication, access control, encryption of the messages can be designed to achieve the required confidentiality, privacy and integrity. Informally, using interactions from point i to point j, we can assess:

- what data becomes available to which principal. From the flow of information, we can clearly assess the confidentiality or privacy of the data involved in the interaction.
- requirements of the system for adaptation to different architectures of "things" and also different threat models; for instance, given some assumptions about the perceived threat, it is possible that communication can be in plain text. However, if the same data is subjected to attacks by man-in-the middle, we would need to secure appropriate interactions through different mechanisms.

Our approach given above is summarized below.

3.2 Methodology

The broad steps are described below:

1. The embedded gateway and the base station are enriched with RWFM that controls the access and the flow of information in the system.
2. The device identifier or RFID tag of a sensor is used to identify the device/sensor; the initial specification (label) defines who its owner, readers and writers are in the system. Thus, the initial lattice of information is formulated.
3. The embedded gateway receives data from a sensor/device, authenticates through (2), and automatically labels it using RWFM model and then sends it to the base station.
4. The access of the data at various points by the stakeholders (users, sensors, devices and middleman) is dictated by the labels of the data being accessed.

These aspects will become clear in the case studies given below.

4 Case Studies on Security of IoT

In this section, we describe two examples - healthcare and Philips Hue - and illustrate how our approach given above becomes an enabler for security of IoT.

4.1 Healthcare

Healthcare is one of the major IoT implementation areas, which is already in making with all the wearable devices and sensors that can be used to monitor patients extensively, enabling doctors to diagnose the disease more accurately. The system architecture explained in Sect. 3 can be modeled for a Healthcare scenario.

The patient - doctor mapping is specified by an administrator such as a supervisor, which determines which doctor is supposed to monitor which patient and thereby allowed to access that patient's sensor data. Let us consider an example scenario of a hospital comprising of three wards in a hospital namely - Cardiology, Neurology, and Diabetes unit. Three patients suffering from critical illness are being monitored in the respective wards by respective specialists. Sensors for ECG (ElectroCardioGram), EEG (ElectroEncephaloGram), and CGM (Continuous Glucose Monitoring) are monitoring the patients P_1, P_2, and P_3. Adaption of our architecture for the example is depicted in Fig. 3.

The continuous monitoring by sensors sends the sensor data to the gateway, this sensor data is then stored in database residing at base station. A statistical computation may be performed so as to help the doctor for a better diagnosis. The doctor accesses the sensor data as well as the statistical computation result of a patient assigned to him through a computer or a handheld device by accessing the base station interface through internet. He can then create a report stating the diagnosis and prescribed tests and medicines. This report can then be accessed by a registered user - doctors or patients depending upon the access control. Note that this further allows us to trigger automated events - like notifying the doctor - based on the thresholds set for the sensor data readings.

The security requirements for the healthcare model described above are:

- Confidentiality and privacy: only an authorized doctor assigned to treat a patient should be able to access the sensor data of the patient. Neither the other patients nor the other doctors be provided any access to a patient's data, as any information leak about the patient's health could have an adverse impact on his social life.
- Integrity: the suggested treatment plan for a patient must be prepared by the assigned doctor only, and only under the influence of the patient's sensor readings.

Security Analysis of Healthcare Example. Consider a health care system having two doctors - "sk", "test1", and three patients - "p1", "p2", "p3" are the users in the system. Further, let us assume that patient p1 is being monitored by the doctor sk, p2 is being monitored by test1, and p3 is being monitored by both sk and test1. Each patient has a sensor device attached that monitors his health status. Assume that the sensor devices "sd1", "sd2" and "sd3" are the sensors attached to patients "p1", "p2", and "p3" respectively. We abuse notation by denoting the data of sensor device sd also by sd. There are three objects sd1, sd2

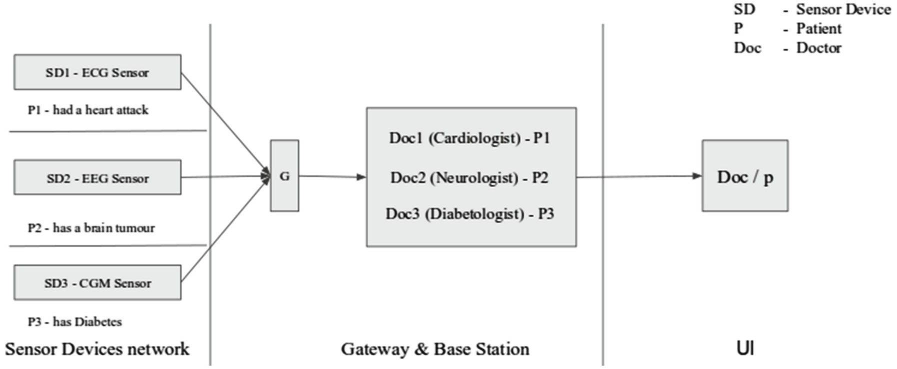


Fig. 3. Architecture for healthcare IoT

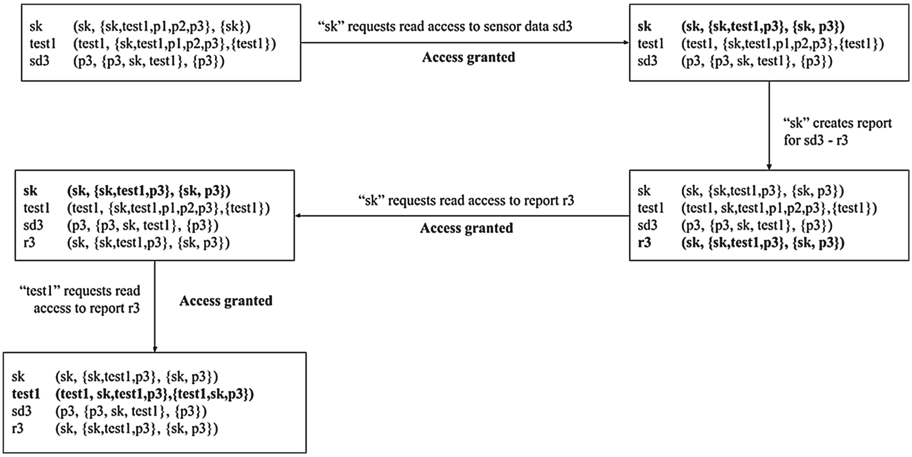

sk (sk, {sk,test1,p1,p2,p3}, {sk})
test1 (test1, {sk,test1,p1,p2,p3},{test1})
sd3 (p3, {p3, sk, test1}, {p3})
"sk" requests read access to sensor data sd3
Access granted
sk (sk, {sk,test1,p3}, {sk, p3})
test1 (test1, {sk,test1,p1,p2,p3},{test1})
sd3 (p3, {p3, sk, test1}, {p3})
"sk" creates report for sd3 - r3
sk (sk, {sk,test1,p3}, {sk, p3})
test1 (test1, sk,test1,p1,p2,p3},{test1})
sd3 (p3, {p3, sk, test1}, {p3})
r3 (sk, {sk,test1,p3}, {sk, p3})
"sk" requests read access to report r3
Access granted
sk (sk, {sk,test1,p3}, {sk, p3})
test1 (test1, {sk,test1,p1,p2,p3},{test1})
sd3 (p3, {p3, sk, test1}, {p3})
r3 (sk, {sk,test1,p3}, {sk, p3})
"test1" requests read access to report r3
Access granted
sk (sk, {sk,test1,p3}, {sk, p3})
test1 (test1, sk,test1,p3},{test1,sk,p3})
sd3 (p3, {p3, sk, test1}, {p3})
r3 (sk, {sk,test1,p3}, {sk, p3})

Fig. 4. IFD for example scenario 1

and sd3 in the system each representing the database of the respective sensor's data. From the given assignment of doctors to patients, we derive the following labels for these objects: (i) sd1 is labelled (p1, {p1,sk}, {p1}), (ii) sd2 is labelled (p2, {p2,test1}, {p2}), and (iii) sd3 is labelled (p3, {p3,sk,test1}, {p3}).

The above labels intuitively capture the following security requirements, for i = 1 to 3:

- sd_i is owned by patient p_i.
- sd_i is accessible only to patient p_i and the doctors assigned to treat him.
- sd_i has been influenced by p_i only.

The initial label of the subjects is as follows: (i) sk is labelled (sk, {sk, test1, p1, p2, p3}, {sk}), (ii) test1 is labelled (test1, {sk,test1,p1,p2,p3}, {test1}), and (iii) for i = 1 to 3, patient p_i is labelled (p_i, {sk,test1,p1,p2,p3}, {p_i}).

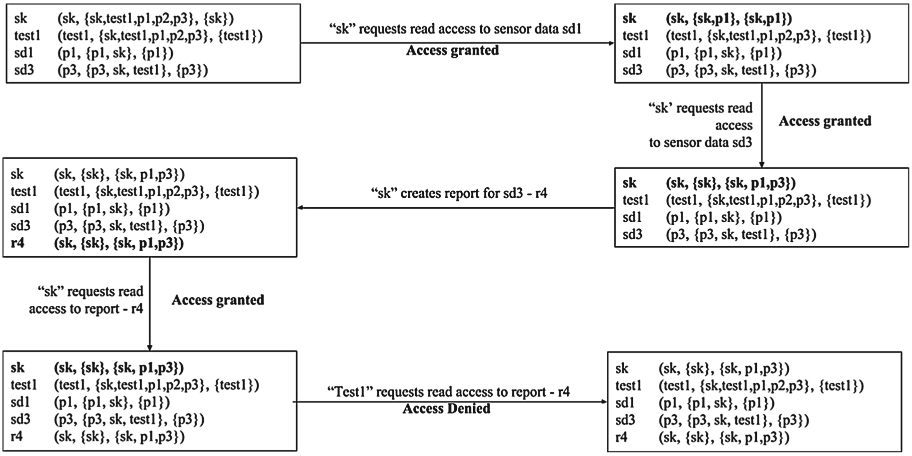

sk (sk, {sk,test1,p1,p2,p3}, {sk})
test1 (test1, {sk,test1,p1,p2,p3}, {test1})
sd1 (p1, {p1, sk}, {p1})
sd3 (p3, {p3, sk, test1}, {p3})
"sk" requests read access to sensor data sd1
Access granted
sk (sk, {sk,p1}, {sk,p1})
test1 (test1, {sk,test1,p1,p2,p3}, {test1})
sd1 (p1, {p1, sk}, {p1})
sd3 (p3, {p3, sk, test1}, {p3})
"sk' requests read access to sensor data sd3
Access granted
sk (sk, {sk}, {sk, p1,p3})
test1 (test1, {sk,test1,p1,p2,p3}, {test1})
sd1 (p1, {p1, sk}, {p1})
sd3 (p3, {p3, sk, test1}, {p3})
"sk" creates report for sd3 - r4
sk (sk, {sk}, {sk, p1,p3})
test1 (test1, {sk,test1,p1,p2,p3}, {test1})
sd1 (p1, {p1, sk}, {p1})
sd3 (p3, {p3, sk, test1}, {p3})
r4 (sk, {sk}, {sk, p1,p3})
"sk" requests read access to report - r4
Access granted
sk (sk, {sk}, {sk, p1,p3})
test1 (test1, {sk,test1,p1,p2,p3}, {test1})
sd1 (p1, {p1, sk}, {p1})
sd3 (p3, {p3, sk, test1}, {p3})
r4 (sk, {sk}, {sk, p1,p3})
"Test1" requests read access to report - r4
Access Denied
sk (sk, {sk}, {sk, p1,p3})
test1 (test1, {sk,test1,p1,p2,p3}, {test1})
sd1 (p1, {p1, sk}, {p1})
sd3 (p3, {p3, sk, test1}, {p3})
r4 (sk, {sk}, {sk, p1,p3})

Fig. 5. IFD for example scenario 2

We now analyze the security provided by our approach by considering some practical usage scenarios for this example.

- **Scenario 1**: doctors sk and test1 try to read the report r3 generated by sk for patient p3.
- **Scenario 2**: doctors sk and test1 try to read the report r4 generated by sk after accessing patients p1 and p3's sensor data.

IFDs for the above described scenarios is presented below.

IFD for Scenario 1 is depicted in Fig. 4. Note that in this scenario all the access requests are granted because they do not violate the policy. In particular, note that, the report r3 generated by sk can be accessed by test1 because he is also assigned to treat patient p3. Further note that, sk has not used any information in preparing r3 that test1 cannot access. The label of r3 clearly indicates its integrity as it has been created by sk, and has been influenced only by sk and p3.

IFD for Scenario 2 is depicted in Fig. 5. Note that in this scenario, report r4 is generated by sk after accessing the sensor data of both p1 and p3 i.e. sd1 and sd3. Only sk is eligible to access r4. None of the patients are allowed to access r4 because it may have information about other patients as well, thus protecting their privacy. Test is not allowed to access r4 because it may have information about p1 which he is not supposed to access.

The example scenarios discussed in this section clearly demonstrate that our model enforces the required security and privacy requirements, by controlling the flow of information in IoT.

4.2 Philips Hue

Philips Hue[2], are the smart light bulbs manufactured by Philips, and can be instructed to change the light color, intensity, and also to respond to certain triggers for e.g., a different light is emitted to notify an email. The Philips hue system has the following four components described in detail below: light, bridge, app, and portal.

The lights are LED light bulbs, which produce the output of the system - the light. The ZigBee mesh network used to connect the light bulbs use ZigBee Light Link protocol to carry the commands. The mesh network provides robustness and resilience to single point of failure.

Bridge is a small gateway device that manages the light bulbs network and the communication of app or the portal with the light bulbs. Since the light bulbs are connected with mesh network, even if a light is not within the network range of the bridge, it can receive the packets sent by the bridge through other light bulbs.

The app is a user interface that lets the user control the light - set a specific color, triggers to change the colors, alarms, notifications for emails, tweets etc. In order to control the light bulbs through the app, the device with the app must be on the same wireless network as that of bridge.

The portal is a web interface provided by Philips, so that the lights can be controlled remotely. The Philips portal requires the user to be on the same wireless network as the bridge when registering for the first time, so that it can discover the bridge and map the user to the bridge. The Portal is what makes the Philips Hue, an Internet of Things.

Adaption of our architecture for this example is depicted in Fig. 6.

For the Philips Hue system to work securely, the authenticity of the commands play a vital role. In the following sub-section, we shall demonstrate how our approach ensures the required security properties.

Security Analysis of Philips Hue Example. Consider a Philips Hue system with the following assumptions: (i) star topology for lights with bridge as the hub, (ii) light bulbs are objects whose value (state) is directly changed by the bridge. The RWFM module resides in the portal and the bridge.

Consider the instance where the system has three light bulbs denoted by $L1$, $L2$ and $L3$. In terms of the information flow, there are three subjects - user (u), portal (p), and bridge (b), and three objects - $L1$, $L2$, $L3$ - the light bulbs. Labels of the light bulbs are same and equal to $(u, \{b\}, \{b, p, u\})$.

Consider the following usage scenario, where the user tries to remotely command the lights:

1. User logs into the system and creates a request ‘Rq1’ to know the status of light ‘L1’ and sends this request to the portal.
2. The portal processes the request ‘Rq1’ and sends a request ‘Rq2’ to the bridge.

[2] http://www2.meethue.com/en-in/, http://www.developers.meethue.com/.

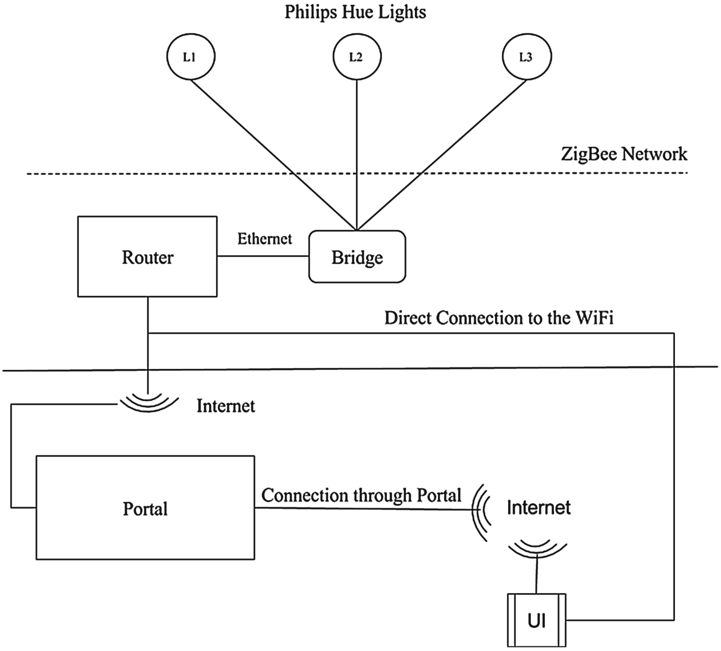


Fig. 6. Architecture for Philips Hue IoT

3. The bridge processes the request ‘Rq2’ and responds ‘Rs1’ to the portal with the status of the light bulb ‘L1’.
4. The portal receives the response sent by the bridge, and sends the response ‘Rs2’ to the user.
5. The user then reads the response ‘Rs2’ and knows the status of the Light ‘L1’. He wants to change the color of the light. So, he sends a request ‘Rq3’ for the same to the portal.
6. The portal reads the request ‘Rq3’ and sends the request ‘Rq4’ to the bridge.
7. The bridge reads the request ‘Rq4’ and changes the color of light ‘L1’.

Labels of the various objects and subjects in the above scenario are depicted in Fig. 7. In the figure, the following convention are follows:

- numbers on the arrow denote the ordering of events,
- dashed arrows represent downgrading,
- arrows from subject to object denote object creation/modification, while arrows from object to subject denote reading
- subjects appear at multiple points on the lattice because as they gain information their label raises, and
- label of the object and subject is depicted in the left-hand column.

From Fig. 7 it is easy to deduce that the request to change the color of the light bulb has been influenced by the user and the portal, and is readable

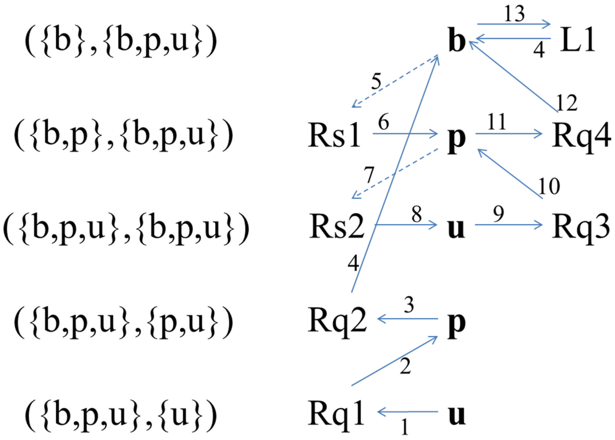


Fig. 7. IFD for the example scenario of Philips Hue

only by the bridge and the portal. If the user was trying to control the light from the same wireless network as the bridge, then the request would have been influenced by the user alone. Thus using our approach it becomes easy to deduce the authenticity of the request.

5 Implementation and Evaluation

In this section, we briefly discuss the implementation of our prototype secure IoT system.

5.1 Implementation Architecture

We have implemented a prototype of the system architecture presented in Sect. 5. In our prototype system, the base station has been developed as a Django[3] based web server. The functionality of the illustrative example presented in Sect. 4 has been implemented as a web application. For managing the security aspects, we have developed a python package for RWFM using MongoDB[4] - a non relational database to store the labels of objects, while the subject labels are stored in the Django session itself. The RWFM package implements the read, write and create rules discussed in Sect. 5. The registered users are managed with a third party app - “django registration redux”. Registration redux provides a simple to use UI with a powerful user authentication and easy user management. Redux provides email support for registration confirmation. Once the registration is done, the registered user can be considered as subjects in the system and the data from sensors as the objects.

[3] https://www.djangoproject.com.

[4] https://www.mongodb.com.

Modules. We have used python modules to create different functions which will manage labels, check for access, and perform various set operations. These modules will be invoked whenever there is a change in state of the information system.

Label Manager. The label manager class has functions that perform basic operation of label assignment, retrieval, updation and deletion. Initial labels are stored along with objects in database as per the policy specification.

The `saveLabel` function saves the given label as the label of the object, the object instance of the class is used to invoke the function. Similarly the `getLabel` function takes the object id and returns the corresponding label. The `updateLabel` function takes the instance, object primary key, and a label as inputs, and replaces the existing label with the given label.

RWFM Operations. The RWFM class has functions that perform operations such as check for read or write, change labels, etc. The functions here, implement the access policies and returns the information state of system with respect to subject and object labels. The `checkRead` method call checks if a given subject can read the given object, and returns a Boolean value True or False and changed label of subject after a successful read. So, we save the subject label after receiving the return value.

Similarly the `checkWrite` function returns whether a given subject can write to an existing object. The `LUB` function returns the least upper bound of two given labels. Function `createObject` simply creates object with a unique objectid used as primary key in relations, and adds the invoking subject's label as the label of the new object in the database.

Using our prototype implementation, we have successfully verified the example scenario 2 discussed in Sect. 4. This is a clear demonstration that our prototype implementation is working as expected in theory and is able to control not only direct but also indirect (through derived objects) misuses of information.

6 Related Work

Singh et al. [24] propose the use of IFC by using security tags to achieve confidentiality, integrity, node identification etc. in IoT. The approach specifies the use of security and integrity tags following classic information flow control model of Bell and LaPadula [3], and Biba [12] of "no read up, no write down" for secrecy and "no read down, no write up" for integrity. The approach further proposes the integration of security tags with digital certificates. With the security and integrity tags, the Certification Authority issued certificates can be used to achieve confidentiality, integrity, anonymization etc. As RWFM provides a nice integration of MAC and DAC, our approach overcomes the difficencies of the above suggestions.

Rghioui et al. [19] suggest the use of symmetric or asymmetric key cryptography. Symmetric key cryptography seems to be a better choice because of it's low power consumption. Generation of symmetric key and its usage as a session key - performing node identification by encrypting the identity of the node and getting verified by gateway is also suggested. Although, cryptography provides basic security features, it cannot capture the information flows, and in particular fails at automatically assigning a policy to the derived data.

Margarida et al. [25] propose the use of REMOA - a healthcare oriented tele-monitoring system with Shibboleth - an authentication middleware. They suggest using the gateway to implement a proxy agent that can provide a functionality to block, permit or filter data. Thus, the said paper achieves authentication and access control with the use of a middleware and transparent proxy. In contrast, in our approach, we use the reference monitor in the base station to achieve the authentication and access control.

Pang et al. [16] provide a detailed discussion on the ecosystem analysis for IoT, by making use of existing infrastructure. Authentication relies on a trusted public third party organisation. The security models of existing channels and service providers are proposed to strengthen the security. The said paper uses trusted authentication, repository based credential management, and SE-based cryptography. The secure element (SE) is used by the system to achieve computationally infeasible to break cryptography. In comparison with our work, we embrace the use of existing resources and infrastructure. We propose to integrate the runtime monitor with the appropriate entities in the system so that security can be realized through IFC. We also use a light cryptographic suite, depending upon the system and implementation specification for IoT as our model is not implementation specific any Private Area Network security suite will work.

The basic security requirement of authenticity, privacy, and access control are the most discussed security issues for IoT. Roman et al. [20] consider the security risks in distributed IoT. The said paper suggests use of Role-Based Access Control (RBAC) [9] policies that use attribute certificates, and further states the need of an infrastructure that allows validating such certificates in a cross-domain environment.

Bohli et al. [5] discuss the SMARTIE Project[5] that makes use of DCapBAC [18] - an authorization scheme that takes access control decisions before the actual service is accessed, it makes use of signed authorization tokens, encryption and digital signature for authentication, and lightweight secure CoAP[6] [23] for wireless security. Radomirovic [17] discusses the issues related to device fingerprinting and profiling and how existing countermeasures like filtering and scanning techniques such as firewalls and malware scanners can be used to overcome the same. Most of these basic security concerns can be tackled by RWFM.

7 Conclusions

In this paper we have described an architecture for synthesising IoT from the underlying set of devices that are connected and their communications. Our approach provides a unified way to realize an IoT with the required security/privacy

[5] http://www.smartie-project.eu/.

[6] http://coap.technology/.

in a flexible way and adaptable to different threat models. The methodology enables us to derive the constraints to be satisfied by the components and the communications of the connected devices for complying with the required security and privacy of the IoT. The methodology has been implemented using the Django framework without diluting the flexibility of the Django framework. Our case studies demonstrate the advantages of the RWFM model in assuring the security and privacy of the IoT. Where there is less consensus is how best to implement security in IoT at the device, network, and system levels. As highlighted in [10], while network firewalls and protocols can manage the high-level traffic coursing through the Internet, protecting deeply embedded endpoint devices that usually have a very specific, defined mission with limited resources available to accomplish it is quite a challenge. Our methodology of adapting the RWFM model in a flexible framework like Django shows that RWFM model is a very viable flexible security model for the IoT framework that takes into account explicit and implicit flows among the various components of the IoT. Furthermore, as highlighted in [10], the security of the IoT also depends on the underlying OS for which RWFM provides a sound information flow model.

Acknowledgement. The work was done as part of Information Security Research and Development Centre (ISRDC) at IIT Bombay, funded by MEITY, Government of India.

References

1. Arias, O., Wurm, J., Hoang, K., Jin, Y.: Privacy and security in internet of things and wearable devices. IEEE Trans. Multi-Scale Comput. Syst. **1**(2), 99–109 (2015)
2. Atzori, L., Iera, A., Morabito, G.: The internet of things: a survey. Comput. Netw. **54**(15), 2787–2805 (2010). http://www.sciencedirect.com/science/article/pii/S1389128610001568
3. Bell, D., La Padula, L.: Secure computer systems: unified exposition and multics interpretation. Technical report ESD-TR-75-306, MTR-2997, MITRE, Bedford, Mass (1975)
4. Bi, Z., Xu, L.D., Wang, C.: Internet of things for enterprise systems of modern manufacturing. IEEE Trans. Industr. Inf. **10**(2), 1537–1546 (2014)
5. Bohli, J.M., Skarmeta, A., Moreno, M.V., Garca, D., Langendrfer, P.: Smartie project: secure IoT data management for smart cities. In: 2015 International Conference on Recent Advances in Internet of Things (RIoT), pp. 1–6, April 2015
6. Collier, S.E.: The emerging enernet: convergence of the smart grid with the internet of things. In: 2015 IEEE Rural Electric Power Conference (REPC), pp. 65–68, April 2015
7. Dlodlo, N.: Adopting the internet of things technologies in environmental management in South Africa. In: 2nd International Conference on Environment Science and Engineering (ICESE 2012), pp. 45–55. IACSIT Press (2012)
8. Evans, D.: The internet of things: how the next evolution of the internet is changing everything. CISCO White Pap. **1**, 1–11 (2011)

9. Ferraiolo, D., Kuhn, R.: Role-based access controls. In: 15th NIST-NCSC National Computer Security Conference, pp. 554–563 (1992)
10. Intel: Security in the internet of things, January 2015
11. Jara, A.J., Bocchi, Y., Genoud, D.: Determining human dynamics through the internet of things. In: 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 3, pp. 109–113, November 2013
12. Biba, K.: Integrity considerations for secure computer systems. Technical report ESD-TR-76-372, MITRE, Bedford, Mass (1976)
13. Narendra Kumar, N.V., Shyamasundar, R.K.: Realizing purpose-based privacy policies succinctly via information-flow labels. In: 2014 IEEE Fourth International Conference on Big Data and Cloud Computing, BDCloud 2014, Sydney, Australia, December 3–5, 2014, pp. 753–760. IEEE (2014). https://doi.org/10.1109/BDCloud.2014.89
14. Narendra Kumar, N.V., Shyamasundar, R.K.: POSTER: dynamic labelling for analyzing security protocols. In: Ray, I., Li, N., Kruegel, C. (eds.) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12–6, 2015, pp. 1665–1667. ACM (2015). http://doi.acm.org/10.1145/2810103.2810113
15. Narendra Kumar, N.V., Shyamasundar, R.K.: Analyzing protocol security through information-flow control. In: Krishnan, P., Radha Krishna, P., Parida, L. (eds.) ICDCIT 2017. LNCS, vol. 10109, pp. 159–171. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-50472-8_13
16. Pang, Z., Chen, Q., Tian, J., Zheng, L., Dubrova, E.: Ecosystem analysis in the design of open platform-based in-home healthcare terminals towards the internet-of-things. In: 2013 15th International Conference on Advanced Communication Technology (ICACT), pp. 529–534, January 2013
17. Radomirovic, S.: Towards a model for security and privacy in the internet of things. In: 1st International Workshop on the Security of the Internet of Things (SecIoT 2010), Tokyo, Japan, December 2010
18. Ramos, J.L.H., Jara, A.J., Marin, L., Skarmeta-Gómez, A.F.: DCapBAC: embedding authorization logic into smart things through ECC optimizations. Int. J. Comput. Math. **93**(2), 345–366 (2016). https://doi.org/10.1080/00207160.2014.915316
19. Rghioui, A., L'aarje, A., Elouaai, F., Bouhorma, M.: The internet of things for healthcare monitoring: security review and proposed solution. In: 2014 Third IEEE International Colloquium in Information Science and Technology (CIST), pp. 384–389, October 2014
20. Roman, R., Zhou, J., Lopez, J.: On the features and challenges of security and privacy in distributed internet of things. Comput. Netw. **57**(10), 2266–2279 (2013). https://doi.org/10.1016/j.comnet.2012.12.018
21. Sangani, K.: The heat is on. Eng. Technol. **9**(7), 49–51 (2014)
22. Schmid, S., Bourchas, T., Mangold, S., Gross, T.R.: Linux light bulbs: enabling internet protocol connectivity for light bulb networks. In: Proceedings of the 2nd International Workshop on Visible Light Communications Systems, VLCS 2015, pp. 3–8. ACM, New York (2015). http://doi.acm.org/10.1145/2801073.2801074
23. Shelby, Z., Hartke, K., Bormann, C.: RFC 7252: The Constrained Application Protocol (CoAP). IETF RFC Publication (2014)
24. Singh, J., Pasquier, T.F.J.M., Bacon, J.: Securing tags to control information flows within the internet of things. In: 2015 International Conference on Recent Advances in Internet of Things (RIoT), pp. 1–6, April 2015

25. Tarouco, L.M.R., Bertholdo, L.M., Granville, L.Z., Arbiza, L.M.R., Carbone, F., Marotta, M., de Santanna, J.J.C.: Internet of things in healthcare: interoperatibility and security issues. In: 2012 IEEE International Conference on Communications (ICC), pp. 6121–6125, June 2012
26. Yun, M., Yuxin, B.: Research on the architecture and key technology of internet of things (IoT) applied on smart grid. In: 2010 International Conference on Advances in Energy Engineering (ICAEE), pp. 69–72, June 2010

Auditing Access to Private Data on Android Platform

Vishal Maral, Nachiket Trivedi, and Manik Lal Das(✉)

DA-IICT, Gandhinagar, India
{vishal_maral,nachiket_trivedi,maniklal_das}@daiict.ac.in

Abstract. App-based utility service on mobile phone has found enormous success in modern digital society. While App-based services on mobile platform make life easy, security and privacy concern of App installed on mobile phone poses a potential threat to user of mobile phone. Users typically do not pay much attention at the time of App installation before accepting the privacy terms display on his/her mobile phone. In this paper, we present a security monitor, a user level tool to detect the events of sensitive data access by mobile Apps and alert user for any suspicious data access. The security monitor does not require the Android root permission to run on mobile platform, instead, it relies on adding hooks to the application package at the bytecode level. The experimental results show that the proposed security monitor can effectively detect private or sensitive data access of Apps with almost no overhead on power consumption of mobile phone and App performance.

Keywords: Security monitor · Security vulnerability · Android Apps Privacy

1 Introduction

Modern information and communication technology makes mobile phone resource rich, equipped with smart sensing technology and usable for varying applications such as banking, shopping, utility services, video conferencing and so on. Furthermore, along with the utility applications, android application market is also swamped by the phishing and privacy intrusive applications. This makes the mobile phone users a potential target for cyber criminals, as user's mobile phone stores user's personals information such as financial data, medical data and behavioral data (phone logs, browsing history, etc.). Many Android Apps ask for permissions, access to resources from user's device and thereby, collect data for marketing and advertising purposes. There has been 50% increase in the number of unique malwares, with the addition of new malware techniques such as SilverPush [1]. Along with the mobile device information like IMEI (International Mobile Equipment Identity), Operating System version, location, and identity of the user, SilverPush listens to the audio from the microphone for near-ultrasonic sounds placed in TV, radio and Web advertisements in order to learn types of ads user watches or listens to.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 105–111, 2018.
https://doi.org/10.1007/978-3-319-72344-0_6

Android uses permission based model to regulate access to privileged data and resources. Applications ask for the permissions in order to access any of those and user typically tends to skip through the list. Even if the user accepts privacy terms and allows resource access, he/she is very unlikely to get any information about how the resource is being used once the access is granted. However, flagging on application as suspicious, based on the privacy terms accepted by user, can not be reliable and complete approach. From user's point of view, whether accessing any particular information is reasonable, in terms of privacy, depends on the context in which information is relied upon. For example, a social networking application to provide a facility to a user to share the places he visits, location information of the user is likely to be needed, but when user is just viewing the posts, it is very unlikely to be needed. Therefore 'when' the sensitive data is accessed, is important from the end users perspective.

In this paper, we present a security monitor, a user level tool, which helps in detecting sensitive data access events within Android applications. The proposed security monitor does not require the Android root permission to run on mobile platform, instead, it relies on adding hooks to the application package at the bytecode level. Our approach consists of (i) a reporting code patch is added to the application binary to track the data access, and (ii) an alert message to make user aware of the event in a non-intrusive fashion. We have experimented the proposed security monitor and found it effective for detecting sensitive data access of Apps without adding any overhead on Apps' performance.

The remainder of the paper is organized as follows. Section 2 provides some background and the work related to the security analysis of Android applications. Section 3 presents the proposed security monitor. Section 4 provides the experimental results of the proposed work. We conclude the paper in Sect. 5.

2 Background and Related Work

2.1 Android Security Internals

Android mainly relies on permission mechanism for its security enforcement. If an application wants to use a sensitive data resource, the developer needs mentioning of it in the application's manifest file (AndroidManifest.xml). The protected system resources are accessed through the APIs. Each of these sensitive APIs is assigned with a permission label, which is a unique security label. The manifest file which is part of the android package is parsed by the Android OS during the installation and the list of permissions is presented to the user.

Android's software architecture consists of four layers: Linux Kernel, Native Userspace, Application framework and Applications. Linux Kernel with several custom changes is at the heart of the Android. Native Userspace and Application Framework layers are together referred as Android Middleware. Hardware Abstraction Layer provides the implementation to the API's used by the upper layer to interact with hardware layer. Applications are executed by Android OS with the help of an Android Runtime, which is the Android's register based virtual machine that runs dex format and supports Dalvik's bytecode specifications. Some of the components and services of Android such as media framework, SQLite are built with native code and forms the Native Libraries section of Android. Android framework refers to the Java API, which is used to access the feature set provided by entire software stack.

2.2 Related Work

In 2012, Google revealed the service, named Bouncer, that they have deployed, which automatically scans the Apps submitted in Google play market for malicious behavior. Oberheide [2] proved with experiments that the dynamic runtime analysis is performed on the submitted applications by the Bouncer for a specific time in an emulated Android environment and the App is flagged if it performs activities of attacking nature such as scan the system for passwords, execute frequent system calls etc. An investigation carried out by Oulehla [3] also indicated that the security tests performed by Bouncer are focused on inspection of AndroidManifest.xml and dynamic runtime analysis and can be easily bypassed. This indicates that Bouncer's main intention was not to detect privacy intrusive applications but to detect malicious applications.

Static Analysis. Leonid et al. [4] proposed a method in which static analysis is performed on decompiled applications and the detection results are used to generate a report in a user-readable and comprehensible form. Siyuan et al. [6] extended the static analysis to make use of inter-procedure analysis by building the call flow graph. Chen et al. [7] suggested converting Dalvik bytecode to Java bytecode before building a call flow graph. Suleiman et al. [8] used Baysian classification for its low computational overhead and its ability to model both an trained and under training system with relative ease. Sahs and Khan [9] used a single class SVM model that is trained from benign samples alone. Zhao et al. [10] built SVM based classifier using both benign and malicious application dataset for signature based malware detection.

Dynamic Analysis. Enck et al. [11] modified the DVM interpreter of Android for variable level tracking. Roshandel et al. [12] modified the ContentObserver class of Androd OS to monitor the data, which is accessed through content URI such as contacts, bookmarks, media, etc. Jia et al. [13] suggested modifying the native layer for intercepting Binder IPC calls to monitor application's privacy data access behavior. Berthome et al. [14] proposed to repackage the application injecting a small patch of code which will work as an audit reporter. Quan et al. [5] combined the static and dynamic analysis approach to propose a hybrid model. First, potential risk applications are identified using the permission combination matrix. Dynamic monitoring module then tracks the runtime calls to sensitive APIs of those applications which are found suspicious by the permission analysis.

Location Privacy. Montjoye et al. [15] showed that up to 95% of the users can be identified from just four randomly chosen location points in a location data of 1.5 million people. Fu et al. [16] suggested a heuristic approach to detect a location access by Android application, based on a consideration that the return value of `getLastKnownLocation()` will change only if location update has been received by any application has. Fawaz et al. [17] proposed LP-Doctor, which allows users to mitigate profiling threat when Apps sporadically access users location data while maintaining the required App's functionality. The difficulty with using LP-Doctor is that it is not really a user level tool. It requires the developer permission to use dumpsys tool, where the permission can only be granted through Android development tools.

Network Monitoring. Intercepting a network traffic to analyze what data is being sent over the network is another major approach used to identify the leakage. Arora et al. [18] suggested analysing network traffic for the malware detection, where malicious traffic is distinguished from the legitimate one with the help of the decision tree classifier. Song and Hengartner [19] proposed PrivacyGuard for intercepting the network traffic of android applications, which makes use of VPNService API provided by Android. PrivacyGuard uses string matching to detect leakage, so it works only if the data is being sent in the plain text format.

3 Proposed Security Monitor

The working principle of the proposed Security Monitor is divided into two phases. In the first phase (Active phase), whenever a new application is installed, Monitor takes application package, decompiles it, adds a reporting section into the bytecode and repackages the application. The second phase (Listening phase) comprises of informing the user about the sensitive data access when the introduced patch generates a notification.

Android broadcasts `android.intent.action.PACKAGE_ADDED` intent (`Intent` [20] is an object that provides runtime binding in response to an event, which contains the name of the action or event happened and related data) whenever a new application is installed. Security Monitor listens to these intents and start functioning. First, the apk of an installed application is copied with the help of a `PackageManager`. Once the apk is retrieved, it is decompiled with the help of Apktool. Then, Dex bytecode contained in the apk is converted into the readable Smali code in the decompilation step. After decompilation, `Reporter` class is added to the application (in Smali format), which contains a `report` method, which generates an intent with location access action with extra data containing the name of the application and time of access. This method is used to raise notification at runtime. In order to get the context, initContext method is placed in the launcher activity which saves the instance of the application context.

Algorithm 1. Repackaging process of Security Monitor

```
Decompile apk
Determine Launcher activity from Manifest file
Add Reporter class to the package
Add call to initContext() method of reporter class in Launcher activity to save
    ApplicationContext
for Every Smali file in the package do
    Scan Smali file to detect sensitive data access
    Add call to report() method of reporter class wherever sensitive data is accessed
Compile into apk
Sign apk
Install newly created apk
```

Sensitive user data targeting operations are detected by scanning the Smali code. Available location co-ordinates can then be retrieved with the LocationManager by passing it as an argument to the method `getLastKnownLocation`. If application wishes to receive continuous location updates, it can do so by calling method `requestLocationUpdates`. Smali parser of the Monitor searches for such calls in the smali files obtained from decompilation. When such call is found, it adds a call to `report` method of a `Reporter` at that place. Modified Smali code base is then assembled and the resultant apk is digitally signed with our own certificate. In run time, whenever location is accessed, `report` method is called which eventually generates an Intent. Algorithm 1 reflects the Monitor's functioning logic. For the second phase of operation, Monitor listens to the Intents generated by the Reporter and logs the access attempts. A Toast message is generated to inform the user about the application that has accessed location.

4 Results and Analysis

We have experimented Security Monitor on Lenovo A6000 device, running Android 5.1, having 2 GB RAM, 1.2 GHz quad core CPU and 2300 mAh battery. We have tested 25 Google Play market Apps, which request the location access permission claiming to facilitate user with various location functionalities. Our proposed methodology is succeeded in repackaging all the applications except the 'mcdeliveryonline' App and the security monitor was able to receive access notifications. In the case of 'mcdeliveryonline' App, the injection failed because the monitor could not locate the entry method. We executed the injected Apps for 24 h. The observations shown in Fig. 1 indicate that Apps can access the location in background as long as 1.5 h and can access as many times as 432. The battery consumption is about 2% for apk of size 5–10 MB, which indicates that CPU is not heavily loaded due to Monitor's operation. To evaluate the overhead introduced in the operation of the repackaged Apps due to the injection, we determined the delay incurred by noting timestamp before and after

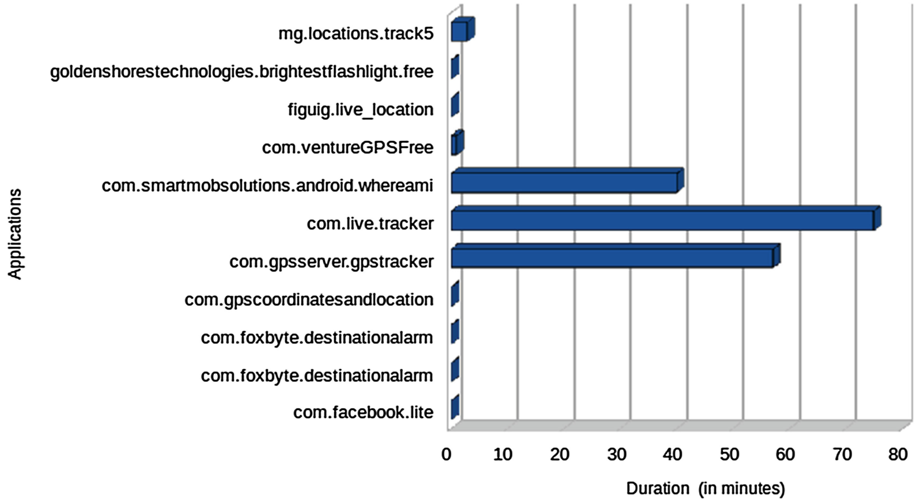


Fig. 1. Duration for tracking location

the call to `report` function. The observed values indicate that injection incurs an unnoticeable delay of 2 to 5 ms.

5 Conclusion

We proposed a user level tool to detect Apps' sensitive data access and alert user for any suspicious data access. To audit the private data access events by android applications, we used repackaging approach. The Security Monitor does not require the Android root permission to run on mobile platform, instead, it relies on adding hooks to the application package at the bytecode level. With experimental results on 26 applications, the proposed Security Monitor is able to detect sensitive data access of 88% applications with almost no overhead on power consumption and App performance.

References

1. SilverPush Android Apps. https://public.addonsdetector.com/silverpush-android-apps/
2. Oberheide, J.: Disecting the Android Bouncer. http://jon.oberheide.org/files/summercon12-bouncer.pdf
3. Oulehla, M.: Investigation into Google Play security mechanisms via experimental botnet. In: Proceedings of IEEE International Symposium on Signal Processing and Information Technology, pp. 591–596 (2015)
4. Batyuk, L., Herpich, M., Camtepe, S.A., Raddatz, K., Schmidt, A., Albayrak, S.: Using static analysis for automatic assessment and mitigation of unwanted and malicious activities within Android applications. In Proceedings of International Conference on Malicious and Unwanted Software, pp. 66–72 (2011)

5. Qian, Q., Cai, J., Xie, M., Zhang, R.: Malicious behavior analysis for android applications. Int. J. Netw. Secur. **18**(1), 182–192 (2016)
6. Ma, S., Tang, Z., Xiao, Q., Liu, J., Duong, T.T., Lin, X., Zhu, H.: Detecting GPS information leakage in Android applications. In: Proceedings of Global Communications Conference, pp. 826–831 (2013)
7. Chen, C., Lin, J., Lai, G.: Detecting mobile application malicious behaviors based on data flow of source code. In: Proceedings of International Conference on Trustworthy Systems and their Applications, pp. 1–6 (2014)
8. Yerima, S.Y., Sezer, S., McWilliams, G., Muttik, I.: A new android malware detection approach using Bayesian classification. In: Proceedings of International Conference on Advanced Information Networking and Applications, pp. 121–128 (2013)
9. Sahs, J., Khan, L.: A machine learning approach to android malware detection. In: Proceedings of Intelligence and Security Informatics, pp. 141–147 (2012)
10. Zhao, M., Zhang, T., Ge, F., Yuan, Z.: RobotDroid: a lightweight malware detection framework on smartphones. J. Netw. **7**(4), 715–722 (2012)
11. Enck, W., Gilbert, P., Han, S., Tendulkar, V., Chun, B., Cox, L.P., Jung, J., McDaniel, P., Sheth, A.N.: TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. ACM Trans. Comput. Syst. **32**(2), 5 (2014)
12. Roshandel, R., Tyler, R.: User-centric monitoring of sensitive information access in Android applications. In: Proceedings of International Conference on Mobile Software Engineering and Systems, pp. 144–145 (2015)
13. Jia, P., He, X., Liu, L., Gu, B., Fang, Y.: A framework for privacy information protection on Android. In: Proceedings of International Conference on Computing, Networking and Communications, pp. 1127–1131 (2015)
14. Berthome, P., Fecherolle, T., Guilloteau, N., Lalande, J.: Repackaging android applications for auditing access to private data. In: Proceedings of International Conference on Availability, Reliability and Security, pp. 388–396 (2012)
15. De Montjoye, Y., Hidalgo, C.A., Verleysen, M., Blondel, V.D.: Unique in the crowd: the privacy bounds of human mobility, vol. 3, p. 1376. Nature Publishing Group (2013)
16. Fu, H., Yang, Y., Shingte, N., Lindqvist, J., Gruteser, M.: A field study of run-time location access disclosures on android smartphones. In: Proceedings of Workshop on Usable Security 2014 (2014)
17. Fawaz, K., Feng, H., Shin, K.G.: Anatomization and protection of mobile apps location privacy threats. In: Proceedings of USENIX Security Symposium, pp. 753–768 (2015)
18. Arora, A., Garg, S., Peddoju, S.K.: Malware detection using network traffic analysis in android based mobile devices. In: Proceedings of International Conference on Next Generation Mobile Apps, Services and Technologies, pp. 66–71 (2014)
19. Song, Y., Hengartner, U.: PrivacyGuard: a VPN-based platform to detect information leakage on android devices. In: Proceedings of the ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices, pp. 15–26 (2015)
20. Android Developer Preview. https://developer.android.com

Privacy Preserving Data Utility Mining Using Perturbation

Joseph Jisna(✉) and A. Salim

College of Engineering Trivandrum, Thiruvananthapuram, India
jisnajoseph.k@gmail.com

Abstract. Data Mining is a field of research dealing with the automatic discovery of knowledge within databases. Recent advances in data mining has increased the disclosure risks that one may encounter when releasing data to outside parties. Privacy preserving data mining (PPDM) deals with protecting the privacy of individual data or sensitive knowledge without sacrificing the utility of the data. Privacy Preserving Utility Mining (PPUM) is an extension of PPDM where the quantity as well as the utility are taken care of. Perturbation is a technique which modifies the contents of database with constraints and satisfies the privacy policies of the data holder. A Fast Perturbation using Frequency Count (FPUFC) algorithm is proposed to hide all sensitive high utility itemsets. The performance of proposed algorithm were compared with that of the existing algorithm, Fast Perturbation using Tree and Table Structures (FPUTT). FPUFC shows better performance by taking lesser execution time compared to FPUTT.

Keywords: Data mining · Utility mining · Perturbation · PPDM PPUM

1 Introduction

Data mining is the process of uncovering hidden valuable knowledge by analyzing large amounts of data. Techniques such as machine learning, artificial intelligence etc. are applied for mining of data stored in databases or data warehouse [1]. Frequent pattern mining is a data mining process to identify relevant patterns or itemsets in a database. Relevant patterns are most frequently occurring itemsets in a database. Most renowned algorithms in this context are Apriori, FP-growth and ECLAT algorithm etc. [5]. However, the practical usefulness of the frequent itemset mining is limited by the significance of the discovered itemsets [3].

The High-Utility Itemset Mining (HUIM) is a utility mining method which produces itemsets having a high profit in transaction databases. It is more practical than frequent itemset mining in real-life situations. A key problem in the area of HUIM is that of confidentiality and privacy preservation. When the results of HUIM are to be made public, privacy threats may arise. To address these issues, Privacy Preserving Data Mining techniques have been proposed which comprises

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 112–120, 2018.
https://doi.org/10.1007/978-3-319-72344-0_7

of perturbing a database to sanitize it by modifying contents of database. Privacy preserving utility mining (PPUM) is an extension of privacy preserving data mining (PPDM). PPUM deals with hiding Sensitive High Utility Itemsets (SHUI).

In this work, we focuses on how to develop a model which can solve issues in PPUM. The SHUIs according to company privacy policy needs to be hidden. This is achieved by modifying the utility value of the sensitive high utility itemsets and making them as non–high utility itemsets. This modification is made by a method, which performs database modification with only fewer number of database scans compared to previous algorithms [8]. The resulting database will not contain sensitive high utility itemsets, and can be used for publishing data which ensures privacy preservation.

2 Related Works

Data mining is a process used by organizations to transform raw data into useful information. Agrawal et al. [13] proposed Apriori algorithm for frequent itemset mining. It uses a bottom up approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The FP-Growth Algorithm, proposed by Han, is an efficient and scalable method for mining the entire set of frequent patterns by pattern fragment growth, using an extended prefix-tree structure for storing compressed and crucial information about frequent patterns named frequent-pattern tree (FP-tree). FP-growth has less execution time compared to Apriori algorithm.

For some applications, it is required that private or confidential information in a database is hidden before the data is publicly published or shared with collaborators. Privacy-Preserving Data Mining (PPDM) has been used for this purpose with the goal to hide sensitive itemsets with minimal side effects [2]. The measurements of these side effects namely, $hiding\ failure$, $missing\ cost$ and $artificial\ cost$ are commonly used as criteria to evaluate the effectiveness and efficiency of PPDM algorithms.

The term, $hiding\ \ failure$ denoted as α shows the set of sensitive itemsets that the data sanitization process failed to hide. Ideally, the set α should be empty when the PPDM procedure finishes its execution. The $missing\ cost$ is denoted as β is the set of non-sensitive frequent itemsets appearing in the original database that cannot be seen in the sanitized database. The $artificial\ cost$ in PPDM is denoted γ represents the set of frequent itemsets appearing in the sanitized database that are infrequent in the original database. The objective of PPDM algorithms is to diminish the side effects as minimal as possible.

The unit profits and quantities of items purchased are not taken into consideration together in frequent itemset mining. But, the utility mining finds all itemsets whose utility values are equal to or greater than a user specified threshold in a transaction database. The main challenge of utility mining is in limiting the size of the candidate sets and simplifying the computation for calculating utility [4].

The first proposal in utility mining was from Yao et al., a unified framework for incorporating several utility-based measures and defining the unified utility function [11]. Liu Y et al. proposed a two-phase algorithm to effectively mine high utility itemsets (HUIs) [12]. It utilizes the transaction-weighted downward closure property to speed up the mining procedure of HUIs. Lin C W et al. used a tree-based structure to find out HUIs [8]. Liu et al. developed the HUI-Miner algorithm to directly mine HUIs without candidate generation [10]. The data structures used to mine HUIs includes an utility-list structure and an enumeration tree. Tseng et al. proposed UP–Growth algorithm which utilizes a tree data structure named UP–Tree for discovering HUIs [7]. The structure of tree used in this algorithm is nearly similar to that used in FP–Growth algorithm, but utility is also included in this.

High Utility Itemset Mining (HUIM) has various real-life applications. Privacy-Preserving Utility Mining (PPUM) which makes use of HUIM, has also become a critical issue in recent years. PPUM manages privacy concerns by combining PPDM and utility pattern mining methods. Mainly two approaches are used, namely *input privacy* and *output privacy.* Input privacy changes the contents of database before conducting mining operations (perturbation, k-anonymity etc.). In the case of output privacy, contents of database are not changed, but rather data is made accessible to only intended people (Secure multi–party computation).

Yeh et al. first designed the Hiding High Utility Itemsets First (HHUIF) and Maximum Sensitive Itemsets Conflict First (MSICF) algorithms to hide sensitive high utility itemsets [2]. The HHUIF algorithm takes each sensitive itemset and identifies the item having maximal utility by scanning database and by projecting database with respect to a sensitive itemset. Then the item is either deleted or its quantity is decreased in transactions. Lin et al. [9] developed a genetic algorithm based method for finding sensitive high-utility itemsets (SHUIs) which modifies database by adding appropriate dummy transactions to decrease the support of sensitive high utility itemsets. Yun and Kim [6] proposed Fast Perturbation using Tree and Table Structures (FPUTT) algorithm for perturbation. It constructs a FPUTT-tree similar to FP-growth tree to aid for faster perturbation, and gives modified database by only 3 database scans. FPUTT perturbs the database by finding item for modification, from the constructed tree. The tree stores only sensitive itemsets and aids to find items faster compared to previous methods. The modification done on items in tree are either decreasing or delete operations. These modified data in tree along with details in IIT table is combined to update the database and thus produce a perturbed database.

Privacy preserving utility mining algorithms mainly use perturbation techniques for modifying databases, to produce a sanitized database. But the modifications done on input database leads to side effects such as hiding failure, misses cost and artificial cost. Also, the time required for perturbation needs to be reduced, by decreasing costly database scans. A better solution for privacy preserving utility mining with lesser execution time as well as reduced side effects is needed.

3 Fast Perturbation Using Frequency Count (FPUFC) Algorithm

Figure 1 shows overall architectural view of the proposed method. The input is transaction database where each row contains transaction Id, item and count. The external utility information is also given as input which indicates the utility value of each item in database. Utility threshold is used as a utility cut off measure. All itemsets having utility value above this threshold are considered high utility itemsets and these items will appear in the results when utility mining is performed.

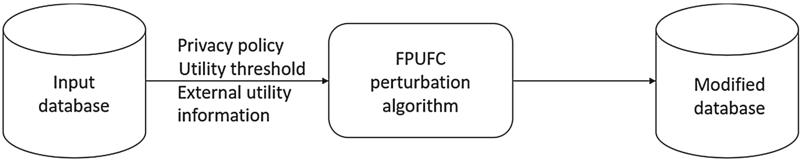


Fig. 1. Design of the process

The privacy policy is determined by the data owner, is also given as input. Privacy policy contains all the sensitive itemsets to be hidden before data is made public.

The main goal of the Fast Perturbation using Frequency Count (FPUFC) algorithm is to decrease the utility value of each sensitive itemset SI, by modifying the quantity values of each item present in it. For each sensitive itemset in SI, FPUFC has to identify the item having highest utility in the transactional database and to decrease its count with least information loss. The process repeats until the utility values of all sensitive itemsets are below the minimum utility threshold. The overall process of the algorithm is given in Algorithm 1.

Input provided to algorithm are the original database DB, the minimum utility threshold δ, sensitive itemsets SI list and the internal utility associated with each item present in database. The output produced is perturbed database, PDB which satisfies privacy policy. In FPUFC, the data structures, PDB and SIT are initialized with null values. SIT is the sensitive itemset table that stores the TIDs corresponding to each of the sensitive itemset. Then, the count of each item present in the sensitive itemsets are calculated (line 2-4) and these items are sorted on the basis of frequency count and stored as sorted_s_items (line 5). Input database consisting of transactions scanned for sensitive itemset and added to SIT. Each entry of SIT consists of TID, item and count.

Algorithm 1. Fast Perturbation using Frequency Count FPUFC

Input : The original database: DB, The minimum utility threshold: δ
Sensitive itemset: SI, Internal utility u associated with each item
Output : The perturbed database PDB

```
procedure –FPUFC
    for each itemset in SI do
        find count of each item
    end for
    Sort item based on frequency count S, sorted_s_items
    for each transaction Ti in DB do
        for each sensitive itemset, sr $\epsilon$ SI do
            if (sr is included in Ti) then
                SIT $\leftarrow$ SIT $\cup$ {TID, (item, count) pair of sr}
            end if
        end for
    end for
    for each item I in sorted_s_items do
        Find itemset, sr from SI that contains I
        utility_diff = $Utilty\_V\_Difference(sr, \delta)$
        while (utility_diff > 0) do
            Find each (item,count) from SIT indexed by TID, $\forall$ item $\epsilon$ sr
            c_utility $\leftarrow$ utility of (item,count)
            if utility_diff > c_utility then
                update count = 0
                update utility_diff = utility_diff - c_utility
            else $count = count - \frac{utility\_diff}{externalutility(item)}$
                update utility_diff = 0
                Calculate Utilty Value Difference
            end if
        end while
    end for
end procedure
```

Algorithm 2. Utility_V_Difference(sr, δ)

```
for tid, tuple in SIT do
    for i, count in tuple do
        total = total + (count * utility[i])
    end for
end for
return utility_diff = total - $\delta$;
```

For each item i in sorted_s_item, an itemset (sr) containing i as a part of it is retrieved from the SI list. The function utility_value_difference in Algorithm 2 is invoked to calculate the utility value difference of sensitive itemset (sr) with that of utility threshold. If the utility_diff value is positive, SIT table is searched to find all transactions that includes this sensitive itemset (sr). The (item, count) pair having maximum utility in these transactions is selected and its utility, c_utility is calculated (line 18). The modification in the quantity of item selected is decided based on the utility_diff and c_utility values. If utility_diff for sensitive itemset (sr) is greater than the c_utility of item selected, then the count of the item gets updated to zero. These changes are effected to SIT and the transaction database. On the other hand, if utility difference of sensitive itemset (sr) is not greater than the utility value of (item, count) selected, the count is updated by making use of external utility value of item (line 22). This process is repeated for all items in sorted_s_items. The updated database is known as perturbed database (PDB).

4 Experimental Evaluation

To measure the effectiveness of the Fast Perturbation Using Frequency Count (FPUFC) algorithm, it is compared with the state-of-the-art algorithm Fast Perturbation Using Tree Structure and Tables (FPUTT). A *Complexity Analysis* has been conducted on the basis of total execution time required by each of the algorithm to perturb a given database. The parameters used in the analysis are as follows. Let N_s be the number of sensitive itemsets, $R_{N(Sp)}$ be the repetition number required for perturbing a sensitive itemset, Sp. Assume DB_{scan} represent the number of item accesses during one database scan, $C(K, SI)$ is the constant time cost for updating the utility information of SI in k^{th} scanning operation, SIT_{scan} is the number of item accesses in SIT for finding required item, N_{SI} is the number of all different items included in a set of sensitive itemsets and $TREE_{scan}$ is the number of node accesses during one FPUTT-tree traversal.

Based on the analysis of algorithms, the time required for FPUTT can be computed as

$$T(FPUTT) \rightarrow 3 * DB_{scan} + \sum_{p=1}^{N_s} \sum_{k=1}^{RN_{Sp}} TREE_{scan} + C(K, S_p) \tag{1}$$

Similarly, the time required for FPUFC can be computed as

$$T(FPUFC) \rightarrow 2 * DB_{scan} + \sum_{r=1}^{N_{SI}} \sum_{k=1}^{N_S} SIT_{scan} + C(K, S_r) \tag{2}$$

Initially algorithms perform database scan to read whole input. Then both algorithms find out the transactions containing sensitive itemset present. In case of FPUTT, it constructs FPUTT-tree containing sensitive itemsets by 2 database scans along with sensitive itemset table and insensitive itemset table. Then modification is done on tree structure and update the original database. In case of FPUFC, it calculates the count of sensitive itemsets present in sensitive itemsets and sort in decreasing order. Then a dictionary based on sensitive itemset is made and item for modification is found out making use of this structure. Updations are done on database itself.

The time complexity of the total time-cost formulas can be denoted as follows.

$$FPUTT = O(DB_{scan} + N_s * P_n * TREE_{scan}) \tag{3}$$

$$FPUFC = O(DB_{scan} + N_s * R_n * SIT_{scan}) \tag{4}$$

The reason why FPUFC outperforms FPUTT is that the time to construct FPUTT tree and associated structures is higher than the time to construct the dictionary and list structures used in FPUFC. Also after values are modified in FPUTT-tree, all those values in nodes of tree need to be combined with those in II table to update database. But in FPUFC, there is no such process.

A *Runtime analysis* is also conducted on the basis of total execution time necessary to perturb a given database. Both algorithms were implemented in Python programming language. The experimental results are tabulated in Table 1. As the number of transactions increases, time for perturbation increases for both algorithms. But, the rate of increase is less for FPUFC when compared to FPUTT.

Table 1. Comparison of time for perturbation

No. of transactions	Time in seconds	
	FPUTT	FPUFC
8	0.1610	0.0039
16	0.3040	0.0060
32	0.5789	0.1600
64	1.2349	0.0239
100	1.8340	0.1089
200	3.6819	0.4430
250	4.6119	0.9980
500	9.4299	3.7820
1000	18.2160	15.5220

Another important consideration is that of side effects while perturbation. Hiding failure, missing cost and artificial cost are three well defined parameters for the performance evaluation of privacy preserving utility mining algorithms. Hiding failure shows the set of sensitive items that the sanitization process failed to hide. Both of the algorithms are very much effective in hiding the privacy policy by bringing down the utility value of sensitive itemsets below the threshold value. In the proposed FPUFC algorithm, hiding failure is 0. Also the only kind of modification made to database is reducing the count of items and hence no new rules are being generated. Thus artificial cost of both algorithms are also

0. Artificial cost represents the set of high utility itemsets appearing in the sanitized database that are non-high utility itemsets in the original database. But by decreasing the count of items in the FPUFC algorithm, missing cost can occur and full elimination of misses cost is not possible in this scenario.

5 Conclusion

In this paper, a fast algorithm for privacy preservation, Fast Perturbation Using Frequency Count (FPUFC) is proposed. It uses sorted list of sensitive items, separate structures for itemset tables, and novel perturbation method to sanitize the transaction database. Efficiency was demonstrated by comparing FPUFC with state-of-art Fast Perturbation Using Tree Structure and Tables (FPUTT) algorithm under various criteria. It turns out that the runtime by FPUFC outperforms FPUTT and it provides a simple and elegant solution for perturbing a database to protect privacy requirements by satisfying privacy policy of the data holder. FPUFC makes use of table and multi-list structure for fast perturbation. The contributions of this work can be applied to other areas like anonymization, data stream mining, and so on.

References

1. Lin, J.C.W., Wu, T.Y., Fournier - Viger, P., Lin, G., Zhan, J., Voznak, M.: Fast algorithms for hiding sensitive high-utility itemsets in privacy-preserving utility mining. Eng. Appl. Artif. Intell. **55**(C), 269–284 (2016)
2. Yeh, J.S., Hsu, P.C.: HHUIF and MSICF: novel algorithms for privacy preserving utility mining. Expert Syst. Appl. **37**(7), 4779–4786 (2010)
3. Verykios, V.S., Elmagarmid, A.K., Bertino, E., Saygin, Y., Dasseni, E.: Association rule hiding. IEEE Trans. Knowl. Data Eng. **16**(4), 434–447 (2004)
4. Dehkordi, M.N., Badie, K., Zadeh, A.K.: A novel method for privacy preservation in association rule mining based on genetic algorithms. J. Softw. **4**, 555–562 (2009)
5. Oliveira, S.R., Zaiane, O.R.: Privacy preserving frequent itemset mining. In: Proceedings of IEEE ICDM Workshop On Privacy, Security and Data Mining, pp. 45–54 (2002)
6. Yun, U., Kim, J.: Fast perturbation algorithm using tree structure for privacy preserving utility mining. Expert Syst. Appl. **42**, 1149–1165 (2015). ELSEVIER
7. Tseng, V.S., Wu, C.W., Shie, B.E., Yu, P.S.: UP–growth: an efficient algorithm for high utility itemset mining. In: Proceedings of 16th ACM SIGKDD, pp. 253–262 (2010)
8. Lin, C.W., Hong, T.P., Lu, W.H.: An effective tree structure for mining high utility itemsets. Expert Syst. Appl. **38**(6), 7419–7424 (2011). ELSEVIER
9. Lin, C.W., Zhang, B., Yang, K.T., Hong, T.P.: Efficiently hiding sensitive itemsets with transaction deletion based on genetic algorithms. Sci. World J. **2014**, 1–13 (2014)
10. Liu, M., Qu, J.: Mining high utility itemsets without candidate generation. In: ACM International Conference on Information and Knowledge Management, pp. 55–64 (2012)

11. Yao, H., Hailton, H.J.: Mining itemset utilities from transaction databases. Data Knowl. Eng. **59**(3), 603–626 (2006)
12. Liu, Y., Liao, W., Choudhary, A.: A two-phase algorithm for fast discovery of high utility itemsets. In: Ho, T.B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 689–695. Springer, Heidelberg (2005). https://doi.org/10.1007/11430919_79
13. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 547–559 (1994)

BDI Based Performance Enhancement in DNSSEC

Kollapalli Ramesh Babu(✉) and Vineet Padmanabhan

School of Computer and Information Sciences,
University of Hyderabad, Hyderabad 500046, India
krubabu@gmail.com, vineetcs@uohyd.ernet.in

Abstract. The DNSSEC (Domain Name System Security Extensions) protocol was basically designed to protect the DNS (Domain Name System) system. In order to perform the intended tsk, the DNSSEC protocol primarily uses a technique called *chain of trust*. During chain of trust construction, the DNSSEC protocol actually ensures the identity/credentials of each node that come across with its parent node. Finally, when it reaches to the point of trust anchor (i.e. root), it stops the process.

The problem that we addressed in this paper is - effective utilization of chain of trust results to improve performance of the DNSSEC protocol. We proposed temporal BDI (Belief Desire Intention) based approach to enhance the performance of DNSSEC.

Keywords: DNS · DNSSEC · BDI · Chain-of-trust

1 Introduction

The DNSSEC protocol was designed to protect DNS [3] from various attacks [2], and the major technique used to accomplish this task is *chain of trust*. The DNSSEC protocol [1] tries to build a chain of trust from the domain under consideration to the point of trust, also called *trust anchor* (in our case it is *root* node). The actual task carried during the construction of chain of trust is verification of the credentials of child nodes with its respective parent node. If we observe chain of trust construction process closely, it is very clear that additional requests/queries are generated to ensure the *authentication* and *integrity* of the received response. Obviously this will take additional time to construct chain of trust.

If we can effectively utilize the previous results that were obtained while constructing the chain of trust, probably upto some extent we can save the time as well as avoid the generation of additional requests to construct chain of trust, which intern reduce the *Internet traffic*. Therefore, the effective utilization of results obtained from previous queries would definitely improve the performance of the DNSSEC protocol.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 121–126, 2018.
https://doi.org/10.1007/978-3-319-72344-0_8

The rest of the paper is organized as follows: In Sect. 2, we have discussed analysis of DNSSEC within BDI system. In Sect. 3, we have illustrated our idea with an example. In Sect. 4, we have discussed conclusion.

2 Analysis of DNSSEC Using Temporal BDI

The basic idea to find solution for a given problem and/or analyzing a given security protocol by making use of BDI model [4,5] is, first we have to comprehend the given problem/protocol and then develop *belief-base*, *action-base* corresponding to it. Once we get the belief-base, action-base then we can apply appropriate algorithms to achieve our goal.

The following steps are applied to analyze DNS and DNSSEC within BDI.

1. The IP address for a given domain name is checked in belief-base, if it is found, then use the IP address and exit the process. Otherwise continue with step 2.
2. The DNS query resolution process is used to find the IP address for a given domain name. If the IP address is detected successfully, then continue, otherwise stop the process and exit.
3. If belief-base contain belief-axiom for a given IP address then believe IP address. Otherwise continue with step 4.
4. If the chain of trust construction is successful, then add the belief-axiom correspond to the given IP address and also add belief-axioms corresponding to all the all servers that come across in chain of trust construction into belief-base. Otherwise simply reject the IP address.

The fourth step in the above procedure is crucial in our analysis because it improves the overall performance of the system by avoiding the reconstruction of the chain of trust for the domains to which it has already constructed.

3 An Illustrative Example

Problem definition: Let us assume that there is a system called `D` under `com` domain which wants to find an IP address of `scis` under `uohyd` domain and also construct the chain of trust between the server from which it has received an IP address of `scis` to the root node, before believing and using the IP address of `scis`.

Solution

The problem said above can be visualized as shown in Fig. 1. For a given problem we can construct the following *belief-base* and *action-base* to proceed further (Tables 1 and 2).

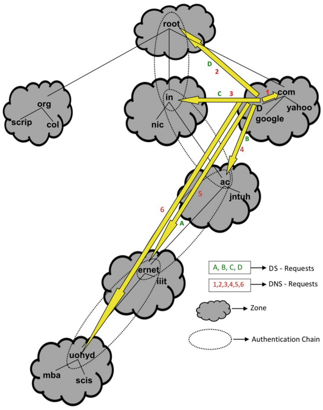


Fig. 1. DNSSEC within BDI

Table 1. Belief-base of DNSSEC.

1. believe(root)

Table 2. Action-base of DNSSEC.

1. send_DNS_request(sent_message, src_address, dst_address)
2. receive_DNS_request(received_message, dst_address, src_address)
3. send_DNS_response(sent_message, src_address, dst_address)
4. receive_DNS_response(received_message, dst_address, src_address)
5. send_DS_request(sent_message, src_address, dst_address)
6. receive_DS_request(received_message, dst_address, src_address)
7. send_DS_response(sent_message, src_address, dst_address)
8. receive_DS_response(received_message, dst_address, src_address)

In order to keep the solution simple, the detailed formats of queries and responses that the actual DNS and DNSSEC systems use, are not used here. The client and server algorithms are designed in such a way that they resolve the given DNS query and construct chain of trust efficiently. The part-A of client and part-A of server algorithms will resolve DNS query, whereas part-B of client and part-B of server will verify *authentication* and *integrity* of the received data. The plan generated to resolve DNS-query is as shown in Table 3. The plan generated to authenticate received data is shown in Table 4. The plan in Table 3 is represented as [1, 2, 3, 4, 5, 6] in Fig. 1. The plan in Table 4 is represented as [A, B, C, D] in Fig. 1. As the plan shown Table 4 gets executed, the belief-base get updated as shown in Table 5. As we can easily observe in Table 5, the four axioms *believe(in)*, *believe(ernet)*, *believe(ac)* and *believe(uohyd)* get appended into belief-base after successful construction of chain of trust. We can keep them in the belief-base for a specified time period. If we get a DNS request within a short time, for any system which is under *in* domain or *ac* domain or *ernet* domain or *uohyd* domain, then we need not repeat the whole process of chain of trust construction to believe the response. Avoiding of unnecessary chain of trust constructions will definitely improve the performance of DNSSEC protocol. This will also reduce *response time* and *Internet traffic*.

Algorithm 1. DNSSEC enabled DNS-Client with BDI

```
Require: belief base, action base, and FQDN (Fully Qualified Domain Name)
Ensure: IP address, chain of trust
    ************* part-A ********************
 1: if is_IP_address_exist(FQDN, belief_base) then
 2:    return IP_address
 3: end if
 4: sent_message ← FQDN
 5: src_address ← sender
 6: dst_address ← parent()
 7: send_DNS_request(sent_message, src_address, dst_address)
 8: receive_DNS_response(received_message, dst_address, src_address)
 9: if is_received_referral(received_message) then
10:    dst_address ← retrieve_referral(received_message)
11:    goto step 7
12: else
13:    IP_address ← retrieve_IP_address(received_message)
14: end if
    ************* part-B ********************
15: if is_source_exist(dst_address, belief_base) then
16:    update_belief_base(IP_address)
17:    Accept and return IP_address
18: else
19:    sent_message ← dst_address
20:    dst_address ← parent(dst_address)
21:    send_DS_request(sent_message, src_address, dst_address)
22:    receive_DS_response(received_message, dst_address, src_address)
23:    if received_message = yes & dst_address ≠ root_address then
24:       append_to_list(believed_source_list, dst_address)
25:       goto step 19
26:    else
27:       if received_message = yes & dst_address = root_address then
28:          chain of trust construction successful
29:          update_belief_base(IP_address)
30:          update_belief_base(believed_source_list)
31:          Accept and return IP_address
32:       else
33:          chain of trust construction failed
34:          Reject received IP_address
35:          clear_source_believed_list(believed_source_list)
36:       end if
37:    end if
38: end if
```

Algorithm 2. DNSSEC enabled DNS-Server with BDI

```
Require: domain name, belief base, action base
Ensure: IP address, referral, child registration
    ************** part-A ********************
receive_DNS_response(received_message, dst_address, src_address)
if is_in_child_list(received_message) then
    sent_message ← retrieve_IP_address(received_message)
else
    sent_message ← retrieve_referral(received_message)
end if
send_DNS_response(sent_message, src_address, dst_address)
    ************** part-B ********************
receive_DS_request(received_message, dst_address, src_address)
if is_child_registered(received_message) then
    sent_message ← yes
else
    sent_message ← no
end if
send_DS_response(sent_message, src_address, dst_address)
```

Table 3. Plan generated by BDI to resolve DNS query

1. send_DNS_request(scis, D, com)
2. receive_DNS_request(scis, D, com)
3. send_DNS_response(root, com, D)
4. receive_DNS_response(root, com, D)
5. send_DNS_request(scis, D, root)
6. receive_DNS_response(scis, D, root)
7. send_DNS_response(in, root, D)
8. receive_DNS_response(in, root, D)
9. send_DNS_request(scis, D, in)
10. receive_DNS_request(scis, D, in)
11. send_DNS_response(ac, in, D)
12. receive_DNS_response(ac, in, D)
13. send_DNS_request(scis, D, ac)
14. receive_DNS_request(scis, D, ac)
15. send_DNS_response(ernet, ac, D)
16. receive_DNS_response(ernet, ac, D)
17. send_DNS_request(scis, D, ernet)
18. receive_DNS_request(scis, D, ernet)
19. send_DNS_response(uohyd, ernet, D)
20. receive_DNS_response(uohyd, ernet, D)
21. send_DNS_request(scis, D, uohyd)
22. receive_DNS_request(scis, D, uohyd)
23. send_DNS_response(IPadd, uohyd, D)
24. receive_DNS_response(IPadd, uohyd, D)

Table 4. Plan generated by BDI to construct chain of trust

1. send_DS_request(uohyd, D, ernet)
2. receive_DS_request(uohyd, D, ernet)
3. send_DS_response(yes, ernet, D)
4. receive_DS_response(yes, ernet, D)
5. send_DS_request(ernet, D, ac)
6. receive_DS_request(ernet, D, ac)
7. send_DS_response(yes, ac, D)
8. receive_DS_response(yes, ac, D)
9. send_DS_request(ac, D, in)
10. receive_DS_request(ac, D, in)
11. send_DS_response(yes, in, D)
12. receive_DS_response(yes, in, D)
13. send_DS_request(in, D, root)
14. receive_DS_request(in, D, root)
15. send_DS_response(yes, root, D)
16. receive_DS_response(yes, root, D)

Table 5. Belief-base after execution of chain of trust plan

1. believe(root)
2. parent(com, root)
3. parent(in, root)
4. parent(ac, in)
5. parent(ernet, ac)
6. parent(uohyd, ernet)
7. IP_address(scis, ”xx.xx.xx.xx”)
8. believe(in)
9. believe(ac)
10. believe (ernet)
11. believe(uohyd)
12. believe IP_address(scis, ”xx.xx.xx.xx”)

4 Conclusion and Future Work

In this paper we have discussed incorporation of DNSSEC protocol into BDI system to improve the performance. To analyze DNSSEC with BDI, first, we have developed belief-base, action-base. Then we have designed algorithms to accomplish name-address resolution and chain of trust construction tasks. The algorithms were designed in such a way that they take the strategic advantage from the results obtained from previous protocol runs, so that the overall performance of the system is improved. In future work we would like to compare the results of our work with others work in terms of performance enhancement in DNSSEC protocol.

References

1. Arends, R., Austein, R., Larson, M., Massey, D., Rose, S.W.: DNS security introduction and requirements, RFC 4033. IETF (2005)
2. Ariyapperuma, S., Mitchell, C.J.: Security vulnerabilities in DNS and DNSSEC. In: ARES, pp. 335–342. IEEE (2007)
3. Mockapetris, P., Dunlap, K.J.: Development of the domain name system, vol. 18. ACM (1988)
4. Padmanabhan, V., Sattar, A., Governatori, G., Babu, K.R.: Incorporating temporal planning within a BDI architecture. In: IICAI, pp. 1618–1636. Springer (2011)
5. Rao, A.S.: Agentspeak(L): BDI agents speak out in a logical computable language. In: Van de Velde, W., Perram, J.W. (eds.) MAAMAW 1996. LNCS, vol. 1038, pp. 42–55. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0031845

A Layered Approach to Fraud Analytics for NFC-Enabled Mobile Payment System

Pinki Prakash Vishwakarma[1](✉), Amiya Kumar Tripathy[2,3], and Srikanth Vemuru[1]

[1] Department of Computer Science and Engineering,
K L University, Guntur, Andhra Pradesh, India
vishwakarmapp@gmail.com, vsrikanth@kluniversity.in
[2] Department of Computer Engineering,
Don Bosco Institute of Technology, Mumbai, India
amiya@dbit.in
[3] School of Science, Edith Cowan University, Perth, Australia

Abstract. Near Field Communication is the technology that will remain widespread in continual with the growth of smart phone influx [1, 10, 12, 15]. Moreover, people use smart phone's to imperforate their mobile banking activities which in turn results in fraudulent activities. The fast growing use of electronic payments has increased the demand for emphatic, decisive and real time based method for fraud detection and prevention. To prevent fraudulent transaction a layered approach for NFC-enabled mobile payment system is proposed. The layered approach for fraud analytic will provide a solution based on transaction risk-modeling, business rule-based, and cross-field referencing.

Keywords: Near Field Communication · Mobile payment · Fraud analytics

1 Introduction

Mobile payment is the means of exchanging financial value between two parties using mobile devices. The amalgamation of the mobile device with the Near Field Communication technology makes payment process possible. NFC-enabled mobile payment is an emanate industry [7]; mobile payments have leading-edge. In the fast emerging modern technologies and global communications, increase in fraud has foisted huge loss to the financial businesses [14]. Therefore, it is an essential affair to identify fraud. Near Field Communication is the technology that will remain widespread in continual with the growth of smart phone influx [10, 12, 15]. Nevertheless, how secure is the mobile payment system still, there will be fraud attacks hence fraud detection measures have to be enforced. Moreover, fraud prevention measures should be associated with fraud detection.

The fast growing use of electronic payments has increased the demand for emphatic, decisive and real time based method for fraud detection and prevention [13, 14]. A NFC mobile phone can communicate with the backend server with secure financial transaction service. In fraud detection location information [4] is vital to detect and prevent frauds. However, it is substantial to fathom that fraudsters have no scope and they can

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 127–131, 2018.
https://doi.org/10.1007/978-3-319-72344-0_9

attack the mobile payment system from any angle. Alluding the consumer transaction history and the spending pattern will curtail the risk of fraudulent transaction [3].

1.1 Our Contribution

The intention is to impart a layered approach for an NFC payment system that identifies fraudulent transactions. The layered approach to fraud analytic provides mastery in each layer. Each layer provides utility to the next higher layer. In this article, following research question has been addressed:

What are the facial characteristics to abate transaction fraud in mobile payment system?

There is a need to identify and position the solution plan for fraud in mobile payment system. The primary countenance to abate transaction fraud in our proposal is:

Real-time transaction monitoring - Monitoring payment transaction across the mobile banking channel, processing payment transaction in real-time. The post-facto monitoring comprehends real time transaction monitoring and step-up alert if any suspicious transaction identified.

Consumer behavioral patterns - The consumer behavior analysis is performed to identify normal and abnormal patterns. The authentication of a transaction using mobile device is based on the consumer behavior pattern.

Multifactor authentication - The countermeasure of the mobile payment process to fraud analytic is multifactor authentication. In fraud prevention, it is required to integrate the prevention system with two factor authentication system.

The remainder of this paper is organized as follows: Sect. 2 is the Motivation and Related Work description. Then the proposed system is described in Sect. 3 which comprises Layered Approach for NFC-enabled mobile payment system and finally conclusion is concluded in Sect. 4.

2 Motivation and Related Work

The layered approach scheme is proposed to abate transaction fraud in NFC-enabled mobile payment system. Nonetheless, with a NFC-enabled mobile payment fraud analytic ecosystem, it provides an opportunity to identify fraudulent pattern and endorse prevention actions. As a result of growing ramification in mobile payment solutions and increase in fraudulent patterns, using only rule-based method for identifying fraud is not competent [8]. Therefore, a solution required which constitutes fraud analytic system with has real-time transaction monitoring, consumer behavior patterns and multifactor authentication for processing payment transactions. Also, there is necessary to entrust the consumers performing mobile payments by proving the consumer rest on their behavior. However, ensuing behavioral patterns enables you to imbibe who the real consumer is in the mobile payment process [12].

The behavior and impingement of feature selection techniques for fraud detection in web payment systems was evaluated [2], the work limited to fraudulent behavior in web transaction scenario. Moreover, using multifactor authentication in NFC-enabled mobile payments is a padding security bestowed in the payment system [11]. Fraud detection and prevention technique which works in backend avails data mining techniques adamantine to secure the facts. The online banking fraud detection framework comprise contrast pattern mining, cost-sensitive neural network and decision forest all these models are combined to generate risk score of an online transaction [5]. The numbers of fraudulent transactions should be less bringing together to number of genuine transactions [6].

3 Description of the Proposed System

In this section the layered approach for NFC-enabled mobile payment system; fraud detection and prevention is bestowed in Sect. 3.1 in detail.

3.1 Layered Approach for NFC-Enabled Mobile Payment System

To attenuate financial losses in mobile payment system institutions ought to take layered approach to fraud analytic.

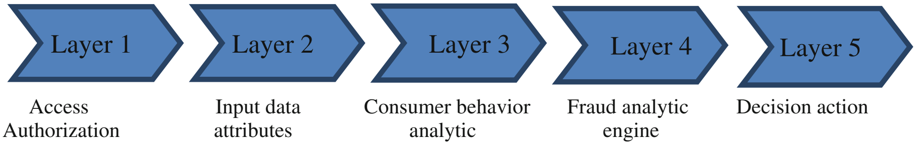


Fig. 1. Layered approach for NFC-enabled mobile payment system

Layer 1 - Access authorization - It encompasses user authentication and device authentication. Layer 1 necessitates access authorization for the consumer. To assist the progress of user authentication a personal identification number (PIN), user ID and password is required. To facilitate device authentication, consumer device is registered using IMEI and device ID [9].

Layer 2 - Input data attributes - Layer 2 includes the attributes for data analysis, which is used to build a consumer behavior profile, that determines normal or abnormal pattern. The attributes for data analysis are velocity, geolocation, IP address, device fingerprint and transaction details which is real-time and dynamic acquisition of consumer information.

Layer 3 - Consumer behavior analytic - Layer 3 presents an image of consumer behavior profile. Based on layer 2 data analysis it identifies normal or abnormal pattern. The consumer behavior profile is a prosperous knowledge and a base for making real-time decisions. The behavior profile acquires real-time data from the mobile device application while performing payment process.

Layer 4 - Fraud analytic engine - Layer 4 provides solution to the payment transaction request which is transaction risk-modeling, business rule-based, cross-field referencing, transaction monitoring and transaction scoring. Layer 4 does thorough transaction monitoring and cross-field referencing to expose sophisticated fraud faster. Originally the transactions that look impeccable may appear fraudulent when attributes are correlated using cross-field referencing. The business rules are defined to percolate fraudulent pattern and suspicious behavior and transaction scoring is done based on the cross-field referencing. Fraud analytic presents an exquisite opportunity to identify fraudulent pattern and endorse prevention actions.

Layer 5 - Decision action - Layer 5 gives the output of the transaction execution. The decision action can be legitimate or fraudulent or suspicious. When the decision is suspicious it enforces second factor authentication like SMS, Email or OTP send to the mobile combined with a transaction pin.

The growth in the modern technology, complexity in fraud management requires a booming approach with mastery in the layers of fraud detection and prevention. Start of an event in NFC-enabled mobile payment system is the user and device authentication which facilitates the user and the device for payment process (Fig. 1). Ensuing user and device authorization the input data attributes are captured for data analysis in real-time and further processes it to build consumer behavior profile. The consumer behavior profile determines the user performing payment process in real-time is a normal or abnormal user. Forthwith the fraud analytic engine impels solution for the payment request from the consumer. The fraud analytic engine identifies fraudulent pattern based on business rules, cross-field referencing, real-time transaction monitoring and transaction scoring thereby recommending fraud prevention actions. Transaction scoring method would determine whether the transaction payment request is from a legitimate user, or fraudulent user. Conclusively the decision action gives the output of the payment transaction execution.

4 Conclusion

The fortuity in mobile payment industry along with growth in smart phones commutes the finance industry towards mobility. The primary countenance to abate transaction fraud in the proposed system is addressed. The proposed and the ongoing work target the user and device authentication, performs fraud analytic thereby maneuvering secure mobile payment. Whither and howbeit the transaction is initiated, the real time transaction monitoring identifies the fraudulent or suspicious payments. However, the multifactor authentication in fraud analytic lead to better accuracy in mobile payment system. It is critical to identify the fraudulent transaction more precisely than the legitimate transactions.

Acknowledgments. The authors thank our colleagues from K L University who provided knowledge and encouragement that eminently assisted the research carried by us.

References

1. Bangdao, C., Roscoe, A.W.: Mobile electronic identity: securing payment on mobile phones. In: Ardagna, C.A., Zhou, J. (eds.) WISTP 2011. LNCS, vol. 6633, pp. 22–37. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21040-2_2
2. Lima, R.F., Pereira, A.C.M.: A fraud detection model based on feature selection and undersampling applied to web payment systems. In: 2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), Singapore, 6–9 December 2015, pp. 219–222 (2015)
3. Almuairf, S., Veeraraghavan, P., Chilamkurti, N., Park, D.-S.: Anonymous proximity mobile payment (APMP). Peer-to-Peer Netw. Appl. **7**(4), 620–627 (2014)
4. Demiriz, A., Ekizoğlu, B.: Using location aware business rules for preventing retail banking frauds. In: 2015 First International Conference on Anti-Cybercrime (ICACC), Riyadh, Saudi Arabia, 10–12 November 2015, pp. 1–6 (2015)
5. Wei, W., Li, J., Cao, L., Yuming, O., Chen, J.: Effective detection of sophisticated online banking fraud on extremely imbalanced data. World Wide Web **16**(4), 449–475 (2013)
6. Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., Bontempi, G.: Learned lessons in credit card fraud detection from a practitioner perspective. Expert Syst. Appl. **41**(10), 4915–4928 (2014)
7. Mehrnezhad, M., Hao, F., Shahandashti, S.F.: Tap-Tap and Pay (TTP): preventing the mafia attack in NFC payment. In: Chen, L., Matsuo, S. (eds.) SSR 2015. LNCS, vol. 9497, pp. 21–39. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27152-1_2
8. Preuveneers, D., Goosens, B., Joosen, W.: Enhanced fraud detection as a service supporting merchant-specific runtime customization. In: Proceedings of the Symposium on Applied Computing, Marrakech, Morocco, 03–07 April 2017, pp. 72–76. ACM (2017)
9. Vishwakarma, P., Tripathy, A.K., Vemuru, S.: A hybrid security framework for near field communication driven mobile payment model. Int. J. Comput. Sci. Inf. Secur. **14**(12), 337–348 (2016)
10. Coskun, V., Ozdenizci, B., Ok, K.: A survey on near field communication (NFC) technology. Wirel. Pers. Commun. **71**(3), 2259–2294 (2013)
11. Wang, Y., Hahn, C., Sutrave, K.: Mobile payment security, threats, and challenges. In: 2016 Second International Conference on Mobile and Secure Services (MobiSecServ), Gainesville, FL, USA, 26–27 February 2016, pp. 1–5 (2016)
12. Cai, C., Weng, J., Liu, J.: Mobile authentication system based on national regulation and NFC technology. In: 2016 IEEE First International Conference on Data Science in Cyberspace (DSC), Changsha, 2016, pp. 590–595 (2016)
13. Van Damme, G., Wouters, K.M., Karahan, H., Preneel, B.: Offline NFC payments with electronic vouchers. In: MobiHeld 2009, Proceedings of the 1st ACM Workshop on Networking, Systems, and Applications for Mobile Handhelds, Barcelona, Spain, 17 August 2009, pp. 25–30 (2009)
14. Edge, M.E., Sampaio, P.R.F., Choudhary, M.: Towards a proactive fraud management framework for financial data streams. In: Third IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC 2007), Columbia, MD, pp. 55–64 (2007)
15. Htat, K.K., Williams, P.A.H., McCauley, V.: Security of ePrescriptions: data in transit comparison using existing and mobile device services. In: ACSW 2017, Proceedings of the Australasian Computer Science Week Multiconference, Article no. 56, Geelong, Australia, 30 January–3 February 03 (2017)

Distributed and Multiprocessing Approaches

Time-Triggered Scheduling for Multiprocessor Mixed-Criticality Systems

Lalatendu Behera(✉) and Purandar Bhaduri

Indian Institute of Technology, Guwahati 781039, India
{lalatendu,pbhaduri}@iitg.ernet.in

Abstract. Real-time safety-critical systems are getting more complex by integrating multiple applications with different criticality levels on a single platform. The increasing complexity in the design of mixed-criticality real-time systems has motivated researchers to move from uniprocessor to multiprocessor platforms. In this paper, we focus on the time-triggered scheduling of both independent and dependent mixed-criticality jobs on an identical multiprocessor platform. We show that our algorithm is more efficient than the Mixed criticality Priority Improvement (MCPI) algorithm, the only existing such algorithm for a multiprocessor platform.

1 Introduction

A *mixed-criticality real-time system* (MCRTS) [1,2] has two or more distinct levels of criticality, such as, safety-critical, mission-critical, non-critical, etc. For example in the domain of unmanned aerial vehicles (UAV's) [2,3] the functionalities are classified into two levels of criticality, viz., *mission-critical* (e.g., capturing and transmitting images) and *flight-critical* (e.g., safe operation of the UAV). The flight-critical functionality, due to its safety critical nature, is subject to certification by a certification authority (CA). The CAs are very conservative, using tools and techniques that estimate more pessimistic worst-case execution times (WCET) than that of the system designers. On the other hand, the CAs are not concerned with the mission-critical functionalities. The system designers are interested in both flight-critical and mission-critical functionalities but their tools are less conservative in estimating the WCETs.

The challenge in scheduling such mixed critical systems is to find a single scheduling policy so that the requirements of both the system designers and the CAs are met. This means that in a scenario where all the jobs complete their executions by their LO-criticality WCETs, they must all be scheduled correctly. On the other hand, in a scenario where the execution time of any one HI-criticality job exceeds its LO-criticality WCET, then all the HI-criticality jobs need to meet their deadlines assuming their HI-criticality WCET to satisfy the CAs.

In this paper, we describe an approach to find a preemptive, global, time-triggered schedule of mixed-criticality, non-recurrent task systems on identical

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 135–151, 2018.
https://doi.org/10.1007/978-3-319-72344-0_10

multiprocessor platforms that can satisfy the assumption of both the CAs and SDs. We show that the worst-case time complexity of our proposed algorithm is better than the existing algorithm in [4], the only existing time-triggered algorithm for such systems.

2 System Model

A mixed-criticality system consists of n jobs $\{j_1, j_2, \ldots, j_n\}$, each with a criticality level. Here we focus on dual-criticality jobs, i.e., LO-criticality and HI-criticality. A job j_i is characterized by a 5-tuple of parameters: $j_i = (a_i, d_i, \chi_i, C_i(\text{LO}), C_i(\text{HI}))$, where

- $a_i \in \mathbb{N}$ denotes the *arrival time*, $a_i \geq 0$.
- $d_i \in \mathbb{N}^+$ denotes the *absolute deadline*, $d_i \geq a_i$.
- $\chi_i \in \{LO, HI\}$ denotes the *criticality* level.
- $C_i(\text{LO}) \in \mathbb{N}^+$ denotes the LO-criticality *worst-case execution time.*
- $C_i(\text{HI}) \in \mathbb{N}^+$ denotes the HI-criticality *worst-case execution time.*

We assume that the system is *preemptive* and $C_i(\text{LO}) \leq C_i(\text{HI})$ for $1 \leq i \leq n$. Note that in this paper, we consider arbitrary arrival times of jobs. An instance of mixed-criticality job set can be defined as a finite collection of mixed-criticality jobs, i.e., $I = \{j_1, j_2, \ldots, j_n\}$. Generally, a job in the instance I is available for execution at time a_i and should finish its execution before d_i. The job j_i must execute for c_i amount of time which is the actual execution time between a_i and d_i, but this can be known only at the time of execution. The collection of actual execution times (c_i) of the jobs in an instance I at run time is called a ***scenario***. Scenarios in our model can be of two types, i.e., *LO-criticality scenarios* and *HI-criticality scenarios.* When each job j_i in instance I executes c_i units of time and signals completion before its $C_i(\text{LO})$ execution time, it is called a LO-criticality scenario. If any job j_i in instance I executes c_i units of time and doesn't signal its completion after it completes the $C_i(\text{LO})$ execution time, then this is called a HI-criticality scenario. Now we define a schedulability condition for a mixed-criticality instance I.

Definition 1. A scheduling strategy is *feasible or correct* if and only if the following conditions are true:

1. If all the jobs finish their $C_i(\text{LO})$ units of execution time on or before their deadlines.
2. If any job doesn't declare its completion after executing its $C_i(\text{LO})$ units of execution time, then all the HI-criticality jobs must finish their $C_i(\text{HI})$ units of execution time on or before their deadlines.

Here we focus on the **time-triggered schedule** [5] of MC instances on a multiprocessor system with identical processors. We will construct two tables S_{HI} and S_{LO} for each processor for a given instance I for use at run time. The length of the tables is the length of the interval $[\min_{j_i \in I}\{a_i\}, \max_{j_i \in I}\{d_i\}]$. The rules to use the tables S_{HI} and S_{LO} at run time, (i.e., the *scheduler*) are as follows:

- The criticality level indicator Γ is initialized to LO.
- While $(\Gamma = LO)$, at each time instant t the job available at time t in the table S_{LO} for processor P_i will execute on P_i.
- If a job executes for more than its LO-criticality WCET without signaling completion in any processor P_i, then Γ is changed to HI.
- While $(\Gamma = HI)$, at each time instant t the job available at time t in the table S_{HI} for processor P_i will execute on P_i.

3 Related Work

Most research on mixed-criticality systems focuses on the uniprocessor case (see for example, [2,6]). The increasing functionalities in mixed-criticality systems motivate researchers to turn to multiprocessor systems (see [4,7–10]). Among the above cited work only [5,6] focus on a time-triggered scheduling algorithm for uniprocessor systems and [4] introduces a time-triggered scheduling algorithm for multiprocessor systems. To the best of our knowledge, there has not been *any other work* studying time-triggered mixed-criticality scheduling for multiprocessor systems.

Socci et al. [4] proposed the Mixed criticality Priority Improvement (MCPI) algorithm to schedule jobs with precedence constraints. In this algorithm, they construct a priority order of jobs from the support algorithm (i.e., a multiprocessor algorithm for non-critical jobs) which is used to find a table for the LO-scenario and the support algorithm is used to schedule the HI-criticality jobs in HI-scenarios. They showed the worst-case time complexity of the algorithm to be $O(n^2 + mn^3 log(n))$, where n is the number of jobs in the instance I and m is the number of processors.

4 The Proposed Algorithm

In this section, we propose an algorithm for mixed-criticality jobs on multiprocessor systems which not only schedules the same set of instances as the existing algorithm [4] but also has a better worst-case time complexity.

The time-triggered scheduling approach to mixed-criticality jobs [4] constructs two scheduling tables S_{LO} and S_{HI} to schedule a dual-criticality instance. Since we consider mixed-criticality jobs for a multiprocessor system, we need two separate scheduling tables for each processor. The schedule constructed by our algorithm is a global one, i.e., a job can be preempted in one processor and resume its execution in another processor. Here we assume that the system is a closely coupled synchronous homogeneous multiprocessor system with shared last level cache and the job context switch time is negligible. We also assume that the cache miss penalty is negligible.

Algorithm 1. LoCBP (LO-criticality based Priority)

Notation:
$I = \{j_1, j_2, ..., j_n\}$, where $j_i =< a_i,\ d_i,\ \chi_i,\ C_i(\text{LO}),\ C_i(\text{HI}) >$.
Input : I
Output : Priority Order (Ψ) of Instance I
Assume the earliest arrival time is 0.

```
 1: Compute the LO-scenario deadline ($d_i{}^{\Delta}$) of each job $j_i$ as $d_i{}^{\Delta} = d_i - (C_i(\text{HI}) - C_i(\text{LO}))$;
 2: while $I$ is not empty do
 3:     Assign a LO-criticality latest deadline[a] job $j_i$ as the lowest priority job if $j_i$ can finish
            its execution in the interval $[a_i, d_i{}^{\Delta}]$ after all other jobs finish their execution in
            LO-scenario under the global EDF scheme;
 4:     If any LO-criticality job cannot be given a lowest priority then a HI-criticality latest
            deadline[a] job $j_i$ is assigned as the lowest priority job if $j_i$ can finish its execution
            in the interval $[a_i, d_i{}^{\Delta}]$ after all other jobs finish their execution in LO-scenario
            under the global EDF scheme;
 5:     if No job is assigned a lowest priority then
 6:         Declare FAIL and EXIT;
 7:     else
 8:         Add job $j_i$ to the priority order $\Psi$;
 9:         Remove job $j_i$ from the instance and continue;
10:     end if
11: end while
12: Construct table $S_{\text{LO}}$ for each processor using the priority order;
13: if $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ then
14:     return FAIL and EXIT;
15: end if
16: The same order as $S_{\text{LO}}$ is followed to allocate the jobs in $S_{\text{HI}}$;
17: After a HI-criticality job $j_i$ is allocated its $C_i(\text{LO})$ execution time in $S_{\text{HI}}$, $C_i(\text{HI}) - C_i(\text{LO})$
        units of execution time of job $j_i$ is allocated after the rightmost segment of job $j_i$ in
        $S_{\text{LO}}$ without disturbing the priority order $\Psi$ and overwriting LO-criticality jobs in the
        process, if any;
```

[a] The original deadline and not the LO-scenario one.

Algorithm 1 determines a priority order, which is used to construct the scheduling tables for all the processors, in steps 1 to 11. First, our algorithm finds the LO-scenario deadline ($d_i{}^{\Delta}$) of each job. For the LO-criticality jobs $d_i{}^{\Delta} = d_i$, but for HI-criticality ones $d_i{}^{\Delta} \leq d_i$. Then the algorithm starts to assign the lowest priority jobs from the instance I. It always selects the latest deadline job to be assigned as the lowest priority job, but LO-criticality jobs are considered before the HI-criticality jobs. A job j_i can be assigned the lowest priority if and only if all other jobs j_k finish their executions when run according to the global EDF algorithm and there remains sufficient time for j_i to complete its $C_i(\text{LO})$ units of execution time before $d_i{}^{\Delta}$. After job j_i is assigned the lowest priority, it is removed from the instance, and the remaining jobs are considered for priority assignment. If at any step a job cannot be assigned a priority, the algorithm declares failure. In step 10, the algorithm constructs table S_{LO}. In steps 11 to 13, it checks for any possible HI-criticality scenario failure. The subroutine $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ checks if at least one job runs at its $C_i(\text{HI})$ execution time, then all HI-criticality jobs must complete their HI-criticality execution before their deadline. If it doesn't find a HI-criticality scenario failure from the subroutine $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$, then the priority order constructed by Algorithm 1 can successfully schedule the instance I. Algorithm 1 constructs table S_{LO} for each processor. Then Table S_{HI} is constructed for each processor by allocating the remaining $C_i(\text{HI}) - C_i(\text{LO})$ units of execution time of each HI-criticality job after its $C_i(\text{LO})$ units of execution time in S_{HI} using the same priority order and also a HI-criticality job is given higher priority over LO-criticality jobs. This means a HI-criticality job can overwrite a LO-criticality job in the process of allocating its $C_i(\text{HI}) - C_i(\text{LO})$ units of execution time.

We illustrate the operation of this algorithm by an example.

Example 1. Consider the mixed-criticality instance given in Table 1 to be scheduled on a multiprocessor system having two identical processors P_0 and P_1.

Table 1. The instance for Example 1

Job	Arrival time	Deadline	Criticality	C_i(LO)	C_i(HI)
j_1	1	5	LO	3	3
j_2	0	8	LO	4	4
j_3	0	7	HI	3	5
j_4	0	4	HI	2	2

Now we construct a priority order using our algorithm. The LO-scenario deadlines d_i^{Δ} of jobs j_1, j_2, j_3, j_4 are $5, 8, 5, 4$ respectively. Now we start assigning priorities to each job.

- The job j_2 is the latest LO-criticality deadline job. If j_2 is assigned the lowest priority, then j_3 and j_4 can run simultaneously in P_0 and P_1 over $[0, 3]$ and $[0, 2]$ respectively. Then j_1 will run over $[2, 5]$ in P_1. So j_2 can execute its 4 units of execution time in P_0 over $[3, 7]$ to finish by its deadline. Now we can assign job j_2 the lowest priority. We remove job j_2 and consider $\{j_1, j_3, j_4\}$ to find the next lowest priority job.
- If j_1 is assigned the lowest priority, then j_3 and j_4 can run simultaneously on P_0 and P_1 over $[0, 3]$ and $[0, 2]$ respectively. Then j_1 will run over $[2, 5]$ in P_1. So j_1 can execute its 3 units of execution time in P_1 over $[2, 5]$ to finish by its deadline. Now we can assign job j_1 the lowest priority. Next, we remove the job j_1 and consider $\{j_3, j_4\}$ to assign the next lowest priority.
- Since there are two jobs and two processors, any job can be given lower priority among the two. But our algorithm assigns the latest deadline job as the lowest priority job. So job j_3 is given the lowest priority.

Finally, the priority order of the jobs in instance I is $j_4 \triangleright j_3 \triangleright j_1 \triangleright j_2$. Now Algorithm 1 constructs the table S_{LO} for each processor using the above priority order. The table S_{LO} for each processor is given in Fig. 1.

Then the $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ subroutine checks for all possible HI-criticality scenarios. We can check that all HI-criticality scenarios are schedulable using the priority order $\{j_4, j_3, j_1, j_2\}$ of I. Finally, table S_{HI} is constructed for each processor by allocating the remaining $C_i(\text{HI}) - C_i(\text{LO})$ units

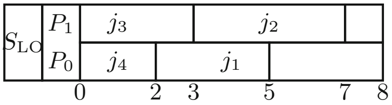


Fig. 1. Table S_{LO} for processor P_0 and P_1

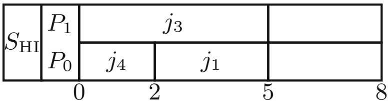


Fig. 2. Table S_{HI} for processor P_0 and P_1

of execution time of each HI-criticality job after its $C_i(\mathrm{LO})$ units of execution time in S_{HI} using the same priority order, where a HI-criticality job is given higher priority over LO-criticality jobs. The table S_{HI} for each processor is given in Fig. 2. □

4.1 Correctness Proof

For correctness, we have to show that if our algorithm finds a priority order for instance I and the $anyHIscenarioFailure(S_{\mathrm{LO}}, I)$ subroutine doesn't fail, then the scheduling tables S_{LO} and S_{HI} will give a correct scheduling strategy. We start with the proof of some properties of the schedule.

Lemma 1. If Algorithm 1 doesn't declare failure and finds a priority order, then each job j_i receives $C_i(\mathrm{LO})$ units of execution time in S_{LO} and each HI-criticality job j_k receives $C_k(\mathrm{HI})$ units of execution time in S_{HI}.

Proof. First, we show that any job j_i receives $C_i(\mathrm{LO})$ units of execution time in S_{LO}. This follows directly from the algorithm as each job j_i must finish its $C_i(\mathrm{LO})$ units of execution time before ${d_i}^{\Delta} \leq d_i$ to be assigned the lowest priority job.

Next we show that any HI-criticality job j_k receives $C_k(\mathrm{HI})$ units of execution time in S_{HI}. We construct the table S_{HI} according to the same priority order. Since $anyHIscenarioFailure(S_{\mathrm{LO}}, I, \Psi)$ subroutine doesn't find any HI-criticality scenario failure, so all the HI-criticality jobs have received their $C_i(\mathrm{HI})$ units of execution time.

Lemma 2. At any time t, if a job j_i is present in S_{HI} but not in S_{LO}, then the job j_i has finished its execution in S_{LO}.

Proof. We use the same order of jobs in S_{LO} to construct S_{HI}. Whenever a job j_i has executed for time $c_i \leq C_i(\mathrm{LO})$ at time t, then it is present in both the tables S_{LO} and S_{HI}. We know the HI-criticality jobs are allocated their $C_i(\mathrm{HI}) - C_i(\mathrm{LO})$ units of execution time after the allocation of $C_i(\mathrm{LO})$ units of execution time in both S_{HI} and S_{LO}. In S_{HI}, the HI-criticality jobs are higher priority job than LO-criticality jobs. When a job j_i is present in S_{HI} and not in S_{LO} at time t, it means this has already completed its execution in S_{LO}. □

Lemma 3. At any time t, when a mode change occurs, each HI-criticality job still has $C_i(\text{HI}) - c_i$ units of execution time in S_{HI} after time t to complete its execution, where c_i is the execution time already completed by job j_i before time t in S_{LO}.

Proof. Let a mode change occur at time t. This means that the following statements hold: (i) all the HI-criticality jobs other than the current job, or none of them has completed their $C_i(\text{LO})$ units of execution time at time t, (ii) the current HI-criticality job is the first one to complete its $C_i(\text{LO})$ units of execution time without signaling its completion. We know that all the HI-criticality jobs are allocated their $C_i(\text{HI}) - C_i(\text{LO})$ units of execution time in S_{HI} after the completion of their $C_i(\text{LO})$ units of execution time in both S_{LO} and S_{HI}. If a job j_i has already executed its $C_i(\text{LO})$ units of execution time in S_{LO}, then it requires $C_i(\text{HI}) - C_i(\text{LO})$ units of time to be completed in S_{HI}. When job j_i initiates the mode change, this is the first job which doesn't signal its completion after completing its $C_i(\text{LO})$ units of execution time. Before time t, the scheduler uses the table S_{LO} to schedule the jobs, while subsequently the scheduler uses table S_{HI} due to the mode change. If a job j_i has already executed its c_i units of execution time in S_{LO}, then it requires $C_i(\text{HI}) - c_i$ units of time to be completed its execution in S_{HI}. We know that the tables S_{HI} and S_{LO} have the same order and according to Lemmas 1 and 2, each job will get sufficient time to complete its $C_i(\text{HI})$ units of execution time. Hence, each HI-criticality job will get $C_i(\text{HI}) - c_i$ units of time in S_{HI} to complete its execution after the mode change at time t. □

Theorem 1. If the scheduler dispatches the jobs according to S_{LO} and S_{HI}, then it will be a correct scheduling strategy.

Proof. For the LO-criticality scenarios, all the jobs can be correctly scheduled by the table S_{LO} as proved in Lemma 1. Now, we need to prove that in a HI-criticality scenario, all the HI-criticality jobs can be correctly scheduled by the table S_{HI}. In Lemma 1, we have already proved that all the HI-criticality jobs get sufficient units of time to complete their execution in S_{HI}. In Lemma 3, we have proved that when the mode change occurs at time t, all the HI-criticality jobs can be scheduled without missing their deadline. So from Lemmas 1 and 3, it is clear that if the scheduler uses tables S_{LO} and S_{HI} to dispatch the jobs then it will be a correct scheduling strategy. □

4.2 Comparison with MCPI Algorithm

Theorem 2. An instance I is schedulable by the MCPI algorithm [4] if and only if it is schedulable by our algorithm.

Proof. ($\Rightarrow$) The MCPI algorithm generates a priority order for an instance I which is used to find table S_{LO}. When a mode change occurs, it uses a support algorithm to schedule the HI-criticality jobs of instance I. We need to show that if MCPI generates a priority order for an instance I, then our algorithm will always find a priority order for instance I and the $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ subroutine will not fail.

Suppose the MCPI algorithm finds a priority order for an instance I. Now the least priority job of the priority order (according to the MCPI algorithm) can be either a LO-criticality or HI-criticality job. First, we consider the case where a job is of LO-criticality. Let j_i be the lowest priority job and its criticality be low. So at the time of construction of the table S_{LO}, every higher priority job j_k finishes its $C_k(\text{LO})$ units of execution time and there remains sufficient time for the lowest priority job j_i to finish its $C_i(\text{LO})$ units of execution time in the interval $[a_i, d_i]$. So this condition is the same as our proposed algorithm.

Let job j_i be the lowest priority job and its criticality be high. Since MCPI successfully finds the priority order, it must have checked all the scenarios and didn't find any failure. Now after every higher priority job j_k finishes its $C_k(\text{LO})$ units of execution time, there remains sufficient time for the lowest priority job j_i to finish its $C_i(\text{LO})$ units of execution time in the interval $[a_i, d_i^{\Delta}]$. Unlike the LO-criticality job, the HI-criticality jobs need to finish their LO-criticality execution on or before d_i^{Δ}. So this condition is the same as our proposed algorithm.

Then j_i is removed from the instance and the next priority can be assigned from the remaining jobs. We can argue in the same way for the remaining jobs. From the above argument, it is proved that our proposed algorithm finds the same priority order for instance I as the MCPI algorithm. Since the priority order is the same and the MCPI algorithm doesn't find any HI-scenario or LO-scenario failure, the $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ subroutine in our algorithm will not fail as well. Thus, for a MCPI schedulable instance, our algorithm can also construct priority tables S_{LO} and S_{HI}.

($\Leftarrow$) Our algorithm generates a priority order for an instance I which is used to find the table S_{LO}. When a mode change occurs, our algorithm uses the table S_{HI} to schedule the HI-criticality jobs which is constructed from the job ordering in S_{LO}. We need to show that if our algorithm generates a priority order for an instance I, then the MCPI algorithm will always find a priority order and the $anyHIScenarioFailure(PT, T)$ subroutine will not fail.

Suppose our algorithm finds a priority order for an instance I. The least priority job assigned by our algorithm can be either a HI-criticality or a LO-criticality job. First, we consider the case where the lowest priority job is LO-criticality. Let j_i be the lowest priority job and its criticality be LO which means the job j_i finishes its execution between its arrival time and deadline after all other jobs finish their execution. So according to the priority table (SPT) of MCPI, job j_i can be given the lowest priority among the LO-criticality jobs. Since the job can meet its deadline after all other jobs finish their execution, the $PullUp()$ subroutine [4] will pull up the HI-criticality jobs upward in the priority tree. So according to the MCPI algorithm the job j_i is the lowest priority job among the HI-criticality jobs as well. This shows that the job j_i is the lowest priority job according to the MCPI algorithm.

Now assume j_i is the lowest priority job and its criticality is HI which means the job j_i can finish its execution between its arrival time and deadline after all other jobs finish their execution. Since our algorithm prefers LO-criticality jobs to assign the lowest priority over HI-criticality jobs, there are no LO-criticality jobs available which can be assigned the lower priority. As in the previous case, job j_i is the lowest priority job in the SPT priority table of the MCPI algorithm. Since no LO-criticality job can finish its execution after the execution of job j_i, the $PullUp()$ subroutine will not be able to pull up the HI-criticality job upward in the priority tree. So job j_i is the lowest priority job according to the MCPI algorithm.

So both the algorithms generate the same priority order for instance I. Since our algorithm doesn't find any HI-scenario failure in the $anyHIscenario$ $Failure(S_{\mathrm{LO}}, I, \Psi)$ subroutine, the MCPI algorithm also doesn't find any HI-scenario failure in its $anyHIscenarioFailure()$ subroutine. □

Theorem 3. The computational complexity of LoBCP on page 4 is $O(mn^2)$, where n is the number of jobs in an instance I and m is the number of processors.

Proof. Line 1 requires $O(n)$ time. Lines 3–4 take $O(n)$ time and each line is simulated on m processors resulting in $O(mn)$ time. Since the outer for loop in line 2 runs n times, the overall complexity is $O(mn^2)$ time. □

This is in contrast to MCPI [4], the only existing time-triggered scheduling algorithm for mixed-criticality systems on multiprocessors, whose complexity is $O(n^2 + mn^3 log(n))$, where n is the number of jobs in instance I and m is the number of processors.

5 Extension for Dependent Jobs

In previous sections, we have discussed instances with independent jobs. Now, we discuss the case of the dual-criticality instances with dependent jobs. In this section, we modify the algorithm given in Sect. 4 to find the scheduling tables such that if the scheduler discussed in Sect. 2 dispatches the jobs according to these scheduling tables then it will be a correct online scheduling strategy without disturbing the dependencies between them. There exists an algorithm [4] which can schedule the jobs of an instance I with dependencies with worst-case time complexity $O(En^2 + mn^3 log(n))$, where n is the number of jobs, E the number of edges in the DAG and m the number of processors. We claim that our algorithm has a better worst-case time complexity than the existing algorithm.

5.1 Model

We use the same model as discussed in Sect. 2. Additionally, an instance of a mixed-criticality system containing dependent jobs can be defined as a *directed acyclic graph* (DAG). An instance I is represented in the form of $I(V, E)$, where V represents the set of jobs, i.e., $\{j_1, j_2, \ldots, j_n\}$ and E represents the edges which depict dependencies between jobs. We assume that a HI-criticality job can depend on a LO-criticality job only if the HI-criticality job depends upon another

HI-criticality job. This means, there are some instances where an outward edge from a LO-criticality job j_l becomes an inward edge to a HI-criticality job j_{h1} with another inward edge from a HI-criticality job j_h to job j_{h1}.

Definition 2. A dual-criticality MC instance I with job dependencies is said to be **time-triggered schedulable** on a multiprocessor system if it is possible to construct the two scheduling tables S_{LO} and S_{HI} for each processor of instance I without violating the dependencies, such that the run-time algorithm described in Sect. 2 schedules I correctly.

5.2 The Algorithm

Here we propose an algorithm which can construct two scheduling tables S_{LO} and S_{HI} for a dual-criticality instance with dependent jobs. A DAG of mixed-criticality jobs is *MC-schedulable* if there exists a correct online scheduling policy for it. Our algorithm finds a LO-criticality priority order for the jobs of instance I which is used to construct the table S_{LO}. Then the same job allocation order of S_{LO} is used to construct the table S_{HI}, where HI-criticality jobs have greater priority than LO-criticality jobs, and the HI-criticality jobs are allocated their C_iHI units of execution time in S_{HI} without violating the dependency constraints. The priority between two jobs j_i and j_k is denoted by $j_i \triangleright j_k$, where j_i is higher priority than j_k. This priority ordering must satisfy two properties:

- If a node j_i is assigned higher priority than node j_k (i.e., $j_i \triangleright j_k$), then there should not be a path in the DAG from node j_k to node j_i.
- If the DAG is scheduled according to this priority ordering then each job j_i of the DAG must finish its $C_i(\text{LO})$ units of execution time before d_i^{Δ}.

Now we present the algorithm DP_LoCBP which finds a priority order for mixed-criticality dependent jobs.

Algorithm 2 finds a priority order which is used to construct the scheduling tables for all the processors in steps 1 to 11. First, our algorithm finds the LO-scenario deadline (${d_i}^{\Delta}$) of each job. For the LO-criticality jobs ${d_i}^{\Delta} = d_i$, but ${d_i}^{\Delta} \leq d_i$ for the HI-criticality jobs. Then the algorithm starts to assign the lowest priority jobs from the instance I. It always selects the latest deadline job which doesn't have an outward edge as the lowest priority job, but LO-criticality jobs are considered before the HI-criticality jobs. A job j_i can be assigned the lowest priority if and only if all other jobs j_k finish their execution and there remains sufficient time for j_i to complete its $C_i(\text{LO})$ units of execution time before ${d_i}^{\Delta}$. After a job j_i is assigned the lowest priority, it is removed from the instance and added to the priority order Ψ. Then the remaining jobs are considered for priority assignment. If at any step a job cannot be assigned a priority, the algorithm declares failure. In step 12, the algorithm constructs the table S_{LO}. In steps 13 to 15, it checks for any possible HI-criticality scenario failure. If it doesn't find a HI-criticality scenario failure, then the priority order constructed by Algorithm 2 can successfully schedule the instance I. Then the table S_{HI} is constructed for each processor by allocating $C_i(\text{HI})$ units of execution

Algorithm 2. DP_LoCBP

Notation:
$I = \{j_1, j_2, ..., j_n\}$, where $j_i =< a_i,\ d_i,\ \chi_i,\ C_i(\mathrm{LO}),\ C_i(\mathrm{HI}) >$.
Input : I
Output : Tables S_{LO} and S_{HI}
Assume earliest arrival time is 0.

```
Compute the LO-scenario deadline ($d_i{}^{\Delta}$) of each job $j_i$ as $d_i{}^{\Delta} = d_i - (C_i(\mathrm{HI}) - C_i(\mathrm{LO}))$;
while $I$ is not empty do
    Assign a LO-criticality latest deadline job $j_i$ which doesn't have an outward edge as
        the lowest priority job if $j_i$ can finish its execution in the interval $[a_i, d_i{}^{\Delta}]$ after
        all other jobs finish their execution in LO-scenario under the global EDF scheme;
    If any LO-criticality job with no outward edge cannot be given the lowest priority
        then a HI-criticality latest deadline job $j_i$ which doesn't have an outward edge
        is assigned as the lowest priority job if $j_i$ can finish its execution in the interval
        $[a_i, d_i{}^{\Delta}]$ after all other jobs finishes their execution in LO-scenario under the global
        EDF scheme;
    if No job is assigned a lowest priority then
        Declare FAIL and EXIT;
    else
        Add the job $j_i$ to the priority order $\Psi$.
        Remove job $j_i$ from the instance and continue;
    end if
end while
Construct table $S_{\mathrm{LO}}$ for each processor $P_i$ using the priority order $\Psi$;
if $anyHIscenarioFailure(S_{\mathrm{LO}},\ I,\ \Psi)$ then
    return FAIL and EXIT;
else
    Construct table $S_{\mathrm{HI}}$ for each processor $P_i$ using the same order of allocated jobs in
        $S_{\mathrm{LO}}$.
    The same order as $S_{\mathrm{LO}}$ is followed to allocate the jobs in $S_{\mathrm{HI}}$;
    After a HI-criticality job $j_i$ is allocated its $C_i(\mathrm{LO})$ execution time in $S_{\mathrm{HI}}$, $C_i(\mathrm{HI}) -$
        $C_i(\mathrm{LO})$ units of execution time of job $j_i$ is allocated after the rightmost segment of
        job $j_i$ in $S_{\mathrm{LO}}$ without violating the dependency constraints and without disturbing
        the priority order $\Psi$;
end if
```

time of each HI-criticality job using the same order of allocated jobs as S_{LO} where a HI-criticality job is given higher priority over LO-criticality jobs. In S_{HI} each HI-criticality job is allocated its $C_i(\mathrm{LO})$ units of execution time without violating the dependency constraints. Once the $C_i(\mathrm{LO})$ units of execution time are allocated for HI-criticality jobs in S_{HI}, the remaining $C_i(\mathrm{HI}){-}C_i(\mathrm{LO})$ units of execution time are allocated immediately without disturbing the priority order Ψ and without violating the dependency constraints. At each instant, the allocation is done without violating the dependency constraints.

We illustrate the operation of this algorithm by an example.

Example 2. Consider the mixed-criticality instance given in Fig. 3 which is going to be scheduled on a multiprocessor system having two homogeneous processors, i.e., P_0 and P_1. The corresponding DAG is given in Fig. 4.

Now we construct a priority order using our proposed algorithm. The LO-criticality scenario d_i^{Δ} of the jobs $j_1, j_2, j_3, j_4.j_5$ are $3, 3, 3, 2, 4$ respectively. Next we start assigning priorities to each job.

- We start with a node having no outward edges from it. The only such node is job j_5. So Algorithm 2 assigns job j_5 the lowest priority. If j_5 is assigned

Job	Arrival time	Deadline	Criticality	C_i(LO)	C_i(HI)
j_1	0	3	LO	1	1
j_2	0	3	LO	1	1
j_3	0	3	LO	1	1
j_4	0	4	HI	1	3
j_5	1	6	HI	1	3

Fig. 3. Instance for Example 2

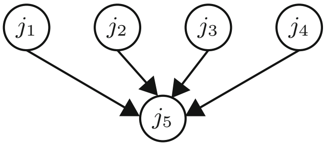


Fig. 4. A DAG showing job dependency among the jobs given in Fig. 3

the lowest priority, then j_1 and j_2 can run simultaneously in P_0 and P_1 over $[0, 1]$ and $[0, 1]$ respectively. Then j_3 and j_4 can run over $[1, 2]$ in P_0 and P_1 respectively. Then j_5 can easily execute its 1 unit of execution on either P_0 or P_1 over $[2, 3]$ to finish by its LO-scenario deadline (d_i^{Δ}). Now we can assign job j_5 the lowest priority job.

We remove job j_5 and consider $\{j_1, j_2, j_3, j_4\}$ to find the next lowest priority job.

- Since the LO-criticality jobs are given the lowest priority by the proposed algorithm, it is easy to verify that the successive lowest priority jobs will be j_1, j_2 and j_3 respectively. Finally, j_4 is the highest priority job.

So the final priority order of jobs in instance I is $j_4 \triangleright j_3 \triangleright j_2 \triangleright j_1 \triangleright j_5$. The table S_{LO} for each processor is given in Fig. 5.

Now the $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ subroutine checks for all possible HI-criticality scenarios. We can check that all HI-criticality scenarios are schedulable using the priority order $j_4 \triangleright j_3 \triangleright j_2 \triangleright j_1 \triangleright j_5$ of the instance I. Finally, the table S_{HI} is constructed for each processor by allocating C_i(HI) units of execution time of each HI-criticality job using the same order of allocated jobs in S_{LO} where a HI-criticality job is given higher priority over a LO-criticality job. On processor P_0, the job order of S_{HI} remains the same as in S_{LO}. Job j_4 is a HI-criticality job and doesn't depend on any other job, so it is allocated its C_i(LO) units of execution time over $[0, 1]$ and the remaining $C_i(\text{HI}) - C_i(\text{LO})$ units of execution time are allocated in the interval $[1, 3]$. Job j_5 is allocated in the interval $[2, 3]$ in table S_{LO} of P_0. But j_5 is allocated in the interval $[3, 6]$ due to dependency constraints which doesn't affect the scheduling after a mode change. On processor P_1, job j_3 and j_2 (LO-criticality) which don't depend on any other jobs, are allocated their one unit of execution time in the intervals $[0, 1]$ and $[1, 2]$ respectively. The table S_{HI} for each processor is given in Fig. 6. □

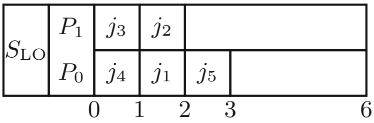


Fig. 5. Table S_{LO} for processor P_0 and P_1

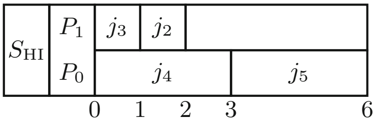


Fig. 6. Table S_{HI} for processor P_0 and P_1

5.3 Comparison with MCPI Algorithm

Theorem 4. An instance I is schedulable by the MCPI algorithm [4] if and only if it is schedulable by our algorithm.

Proof. ⇒ We need to show that if MCPI generates a priority order for an instance I, then our algorithm will always find a priority order for instance I and the $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ subroutine will not fail.

Suppose the MCPI algorithm finds a priority order for instance I. Now the lowest priority job of the priority order (according to the MCPI algorithm) can be either a LO-criticality or HI-criticality job. First, we prove the case where a job is LO-criticality and then HI-criticality. Let j_i be the lowest priority job and its criticality be LO which means no other job depends on j_i. So at the time of construction of table S_{LO}, every higher priority job j_k finishes its $C_k(\text{LO})$ units of execution time without violating the dependency constraints and there remains sufficient time for the lowest priority job j_i to finish its $C_i(\text{LO})$ units of execution time in the interval $[a_i, (d_i^{\Delta})]$. So this condition is the same as our proposed algorithm.

Let job j_i be the lowest priority, and its criticality be HI which means no other job depends on j_i. Since MCPI successfully finds the priority order, it must have checked all the scenarios and doesn't find any failure in the HI-scenario situations. After every higher priority job j_k finishes its $C_k(\text{LO})$ units of execution time, there remains sufficient time for the lowest priority job j_i to finish its $C_i(\text{LO})$ units of execution time in the interval $[a_i, d_i^{\Delta}]$ without violating the dependency constraints. The HI-criticality jobs need to finish their LO-criticality execution on or before d_i^{Δ} in LO-scenario, so that they have sufficient time to finish their remaining $C_i(\text{HI}) - C_i(\text{LO})$ units of execution time before their deadline d_i. This condition doesn't violate the dependency constraints as it is the job which doesn't have an outward edge from it. So this condition is the same as our proposed algorithm.

Then j_i is removed from the instance I and the next priority can be assigned from the remaining jobs. We can argue in the same way for the remaining jobs. From the above argument, it is proved that our proposed algorithm finds the same priority order, for instance I as the MCPI algorithm. Since the priority order is the same and MCPI doesn't find any HI-scenario or LO-scenario failure, $anyHIscenarioFailure(S_{\text{LO}}, I, \Psi)$ subroutine in our algorithm will not fail as well. Thus, for a MCPI schedulable instance, our algorithm can also construct priority tables S_{LO} and S_{HI}.

($\Leftarrow$) Our algorithm generates a priority order for instance I which is used to find the table S_{LO}. When a mode change occurs, our algorithm uses the table S_{HI} which is constructed from the job ordering in S_{LO} to schedule the HI-criticality jobs. We need to show that if our algorithm generates a priority order for instance I, then the MCPI algorithm will always find a priority order and the $anyHIScenarioFailure(PT, T)$ subroutine will not fail.

Suppose our algorithm finds a priority order, for instance I. The lowest priority job assigned by our algorithm can be either a HI-criticality or a LO-criticality job. First, we consider the case where a job is LO-criticality. Let j_i be the lowest priority job, and its criticality be LO which means the job j_i can finish its execution between its arrival time and deadline after all other job finishes their execution without violating the dependency constraints. So according to the priority table (SPT) of MCPI, job j_i can be given the lowest priority among the LO-criticality jobs. Since the job can meet its deadline after all other jobs finished their execution, the $PullUp()$ subroutine will pull up the HI-criticality jobs upward in the priority tree. So according to the MCPI algorithm, the job j_i is the lowest priority job among the HI-criticality jobs as well. This shows that the job j_i is the lowest priority job according to the MCPI algorithm.

Let j_i be the lowest priority job, and its criticality be HI which means the job j_i can finish its execution between its arrival time and deadline after all other job finishes their execution without violating the dependency constraints. Since our algorithm prefers LO-criticality jobs to assign the lowest priority over HI-criticality jobs, there are no LO-criticality jobs available which can be assigned lower priority than job j_i. Our algorithm chooses the job with no outward edges which means no job depends on the lowest priority job. So due to the dependency constraints, all the LO-criticality jobs finish before job j_i. Since no LO-criticality job can finish its execution after the execution of job j_i, the $PullUp()$ subroutine will not be able to pull up the HI-criticality jobs upward in the priority tree. So job j_i is the lowest priority job according to the MCPI algorithm.

In the same way, we argue for the next priority assignment of jobs of instance I. □

Theorem 5. The computational complexity of DP_LoCBP on page 11 is $O(mn^2)$, where n is the number of jobs in the instance I.

Proof. Line 1 requires $O(1)$ time. Lines 3–4 take $O(n)$ time and each line is simulated on m processors resulting in $O(mn)$ times. Since the outer for loop in line 2 runs n time, they require an overall $O(mn^2)$ time. Lines 17–18 takes $O(1)$ time each. So the overall time complexity of our algorithm is $O(mn^2)$. □

This is in contrast to the MCPI algorithm [4], the only existing time-triggered scheduling algorithm for the dependent jobs of mixed-criticality systems on multiprocessors is $O(En^2+mn^3log(n))$, where n is the number of jobs, E the number of edges in the DAG and m the number of processors.

6 Results and Discussion

In this section, we present the experiments conducted to evaluate the LoCBP algorithm for the dual-criticality case. The experiments compare the running times of LoCBP and MCPI. The comparison is done over numerous instances with randomly generated parameters.

The job generation policy may have significant effect on the experiments. The details of the job generation policy are given below.

- The utilization (u_i) of the jobs of instance I are generated according to the Staffords randfixedsum algorithm [11].
- We use the exponential distribution proposed by Davis *et al.* [12] to generate the deadline (d_i) of the jobs of instance I.
- The $C_i(\mathrm{LO})$ units of execution of the jobs are calculated by $u_i \times d_i$.
- The $C_i(\mathrm{HI})$ units of execution of the jobs are calculated as $C_i(\mathrm{HI}) = \mathrm{CF} \times C_i(\mathrm{LO})$ where CF is the criticality factor which varies between 2 and 6 for each HI-criticality job j_i in our experiments.
- Each instance I contains at least one HI-criticality job and one LO-criticality job. We have generated random instances for 2, 4, 8 and 16 processors, where each instance has atleast $m+1$ number of jobs. Each instance is LO-scenario schedulable. We have used an intel core 2 duo processor machine with speed of 2.3 Ghz to conduct the experiments.

In the first experiment, we fix the number of processors to 2 and let the deadline of the jobs vary between 1 and 2000. The graph in Fig. 7 shows the time consumption by each schedulable instances from different numbers of randomly generated instances.

From the graph in Fig. 7, it is clear that our algorithm consumes significantly less time than the MCPI algorithm. As can be seen from Fig. 7, for a multiprocessor with two processors the time consumption by MCPI is much higher than our

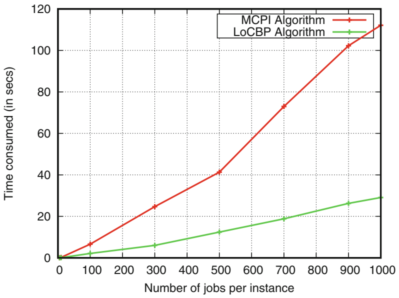


Fig. 7. Comparison of time consumption of MC-schedulable instances for $m = 2$

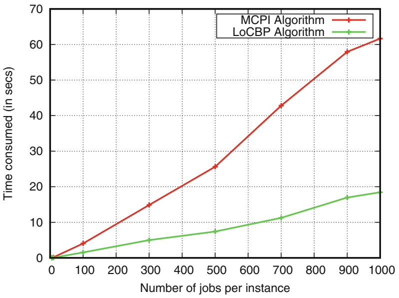


Fig. 8. Comparison of time consumption of MC-schedulable instances for $m = 4$

algorithm. The ratio of time consumed also increases with the increase of number of jobs per instance and is close to five for 1000 jobs. In another experiment, we have shown that the time consumption decreases for $m = 4$, but the ratio of time consumed by our algorithm in comparison to the MCPI algorithm is very much similar to the case $m = 2$, as can be seen in Fig. 8.

7 Conclusion

In this paper, we proposed a new algorithm for time-triggered scheduling of mixed-criticality jobs for multiprocessor systems. We proved that our algorithm has a better worst-case time complexity than the previous algorithm (MCPI). We also proved the correctness of our algorithm. Then we extended our algorithm for dependent jobs and compared the worst-case time complexity with the existing algorithm. We examined the theoretical result by comparing the actual time consumption between LoCBP and MCPI.

References

1. Vestal, S.: Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In: 28th IEEE International Real-Time Systems Symposium (RTSS 2007), pp. 239–243, December 2007
2. Baruah, S., Bonifaci, V., D'Angelo, G., Li, H., Marchetti-Spaccamela, A., Megow, N., Stougie, L.: Scheduling real-time mixed-criticality jobs. IEEE Trans. Comput. **61**(8), 1140–1152 (2012)
3. Valavanis, K.P.: Advances in Unmanned Aerial Vehicles: State of the Art and the Road to Autonomy. Intelligent Systems, Control and Automation: Science and Engineering, vol. 33. Springer, Dordrecht (2007). https://doi.org/10.1007/978-1-4020-6114-1
4. Socci, D., Poplavko, P., Bensalem, S., Bozga, M.: Time-triggered mixed-critical scheduler on single and multi-processor platforms. In: HPCC/CSS/ICESS, pp. 684–687, August 2015
5. Baruah, S., Fohler, G.: Certification-cognizant time-triggered scheduling of mixed-criticality systems. In: 32nd IEEE Real-Time Systems Symposium (RTSS), pp. 3–12. IEEE (2011)
6. Socci, D., Poplavko, P., Bensalem, S., Bozga, M.: Mixed critical earliest deadline first. In: 2013 25th Euromicro Conference on Real-Time Systems, pp. 93–102, July 2013
7. Baruah, S., Chattopadhyay, B., Li, H., Shin, I.: Mixed-criticality scheduling on multiprocessors. Real Time Syst. **50**(1), 142–177 (2014)
8. Giannopoulou, G., Stoimenov, N., Huang, P., Thiele, L.: Mapping mixed-criticality applications on multi-core architectures. In Design, Automation Test in Europe Conference Exhibition (DATE), pp. 1–6, March 2014
9. Giannopoulou, G., Stoimenov, N., Huang, P., Thiele, L.: Scheduling of mixed-criticality applications on resource-sharing multicore systems. In: Proceedings of the Eleventh ACM International Conference on Embedded Software (EMSOFT 2013), pp. 17:1–17:15. IEEE Press (2013)

10. Pathan, R.M.: Schedulability analysis of mixed-criticality systems on multiprocessors. In: 24th Euromicro Conference on Real-Time Systems, pp. 309–320, July 2012
11. Emberson, P., Stafford, R., Davis, R.I.: Techniques for the synthesis of multiprocessor tasksets. In: Proceedings 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS 2010), pp. 6–11 (2010)
12. Davis, R.I., Zabos, A., Burns, A.: Efficient exact schedulability tests for fixed priority real-time systems. IEEE Trans. Comput. **57**(9), 1261–1276 (2008)

Effect of Live Migration on Virtual Hadoop Cluster

Garima Singh(✉) and Anil Kumar Singh

Motilal Nehru National Institute of Technology Allahabad, Allahabad, India
singhgarima4688@gmail.com, ak@mnnit.ac.in

Abstract. Emerging computational requirement for large scale data analysis has resulted in the importance of big data processing. Meanwhile, with virtualization it is now feasible to deploy Hadoop in private or public cloud environment which offers unique benefits like scalability, high availability etc. Live migration is an important feature provided by virtualization that migrate a running VM from one physical host to another to facilitate load balancing, maintenance, server consolidation and avoid SLA violation of VM. However, live migration adds overhead and degrades the performance of the application running inside the VM. This paper discusses the performance of Hadoop when VMs are migrated from one host to another. Experiment shows that job completion time, average downtime as well as average migration time gets increased with increase in the number of VMs that are migrated.

Keywords: Virtualization · SAN · Live migration · Hadoop MapReduce · Pre-copy

1 Introduction

Cloud has appeared as a powerful computing technology in the world of distributed computing. With the advancement in cloud and virtualization technology more and more computation is assumed to be done on the cloud. Virtualization is the core driving technology behind cloud that allows running multiple OS instances on the same physical machine concurrently and facilitate load balancing, server consolidation, live migration etc.

Meanwhile, to process large and distributed dataset Google proposed a scalable programing model known as MapReduce [1]. Hadoop [2], is an open source implementation of MapReduce. It has computational power of MapReduce with HDFS(Hadoop Distributed File System). So, with the advancement in virtualization technology and big data analysis it is now feasible to use virtual machine for big data computing [3,4].

One of the feature provided by virtualization is live VM migration that allows allocation and reallocation of server resources by moving VM from one physical host to another. However, existing migration techniques focused on migration of

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 152–157, 2018.
https://doi.org/10.1007/978-3-319-72344-0_11

a single VM only. When a group of VM [5] or cluster need to be migrated simultaneously these techniques may prove inefficient. This paper analyzes the performance of Hadoop cluster when different number of VMs are migrated within the same subnet. The rest of the paper is organized as follows. Section 2 briefly describes the background work. Section 3 presents the related work. Section 4 presents performance analysis of Hadoop on the cloud. Finally, Sect. 5 conclude our work.

2 Background

MapReduce frees user from the complicated task of parallelization, failure handling and data distribution by requiring user to specify map and reduce function only. Map function processes key/value pairs to generate intermediate data which is another set of key/value pairs. This intermediate data is further processed by reduce function which merges values associated with same key to generate the final result.

Live migration primarily transfers VM's memory, CPU and I/O state from source to destination host. It can be further classified as pre-copy, post-copy and hybrid migration. With pre-copy [6] migration the dirty pages are iteratively transfered across the network followed by a short suspend and copy phase. This iterative phase ends when some terminating condition is reached. This technique is the default migration technique for many hypervisors including Xen which is used in this paper. On the other side, post-copy VM migration first captures and transfers the minimum state to the destination server where the VM is resumed and the remaining memory pages are fetched from the source depending on the read/write request. Lastly, hybrid migration scheme follows bounded number of pre-copy round to transfer a subset of pages followed by a short suspend & resume phase. Next, the remaining pages are pulled from the source.

3 Related Work

Virtualization is useful but it adds overhead to run VM. Hwang et al. in [7] have performed the analysis of the overheads offered by different hypervisor under hardware assisted virtualization. The author has concluded that the overhead incurred by different hypervisor vary depending upon the application hosted on it.

Jhonson and Chiu in [8] have discussed migration of MapReduce task across hosts using pause, migrate and resume strategy. It checkpoint intermediate state and metadata information to facilitate migration. However, this technique transfers only Hadoop MapReduce task, not the entire VM and incurs significant overhead.

Ibrahim et al. in [3] have compared Hadoop performance on physical and virtual machine. The author has discussed that the performance of Hadoop on VM is degraded due to the overhead added due to virtualization. To reduce the cost and improve the performance Kambatla et al. in [9] have optimized the

Table 1. Job completion time(in sec) with different cluster and file size(in MB)

CS:Cluster Size				
CS	File Size			
	128	256	512	1024
1	60	61	104	150
2	60	64	118	154
4	68	70	120	179
6	63	65	173	259
8	63	64	222	600

Table 2. Reduce task completion time(in msec) with different cluster and file size(in MB)

CS:Cluster Size				
CS	File Size			
	128	256	512	1024
1	4287	4351	47171	66349
2	4330	4813	58137	67581
4	4533	7787	55502	105367
6	5283	5923	55089	106305
8	4759	4690	115218	155218

Table 3. Map task completion time(in msec) with different cluster and file size(in MB)

CS:Cluster Size				
CS	File Size			
	128	256	512	1024
1	81664	160186	531106	1500951
2	77861	161892	632904	1486645
4	91543	174938	604315	2277456
6	80069	170504	939676	2476982
8	80686	167409	1069634	2557333

Hadoop provisioning on cloud. However, these approaches does not address the performance of any application when multiple VMs are migrated. This paper studies the performance of Hadoop cluster with live VM migration.

4 Performance Analysis

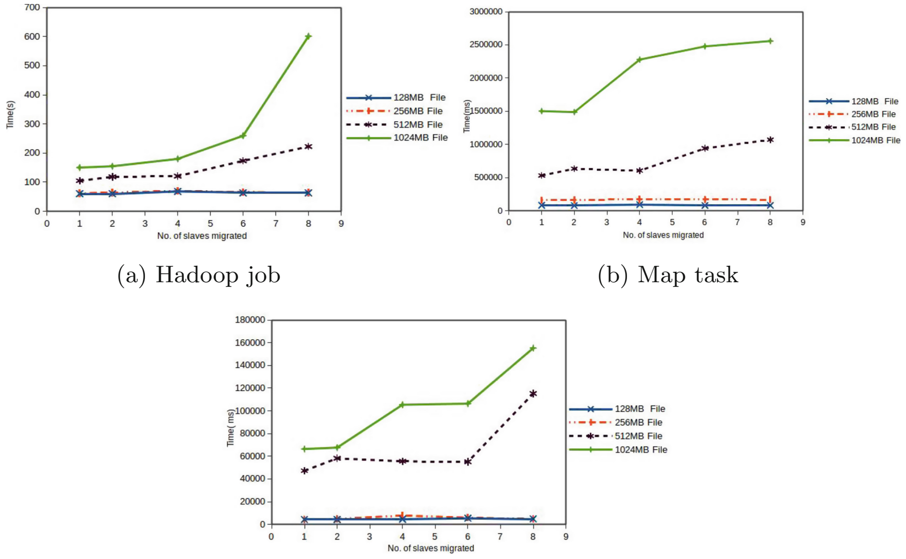
In this section, experimental setup and performance of Hadoop cluster on cloud is discussed.

4.1 Experimental Configuration

The private cloud setup has 6 blade servers each with 2x4 core Intel Xeon CPU E5-2665 @ 2.40GHz processor, 270 GB local disk space and 2 physical NIC with 1 Gbps in both direction. Each blade has hyper threading enabled with two threads running on each CPU core. Each server has Xen 6.5 hypervisor and Linux with kernel version 3.10.0 running in dom0. Ubuntu 10.04 is configured with Hadoop 2.6.1 on each VM. The disk image of each VM is stored on the shared storage. So, when VM migrate disk image is not transfered across hosts. The performance of Hadoop on cloud is analyzed with different cluster size of 2, 4, 6 and 8 nodes. Four different files of 128, 256, 512 and 1024 MB are used as an input to Wordcount program running on Hadoop. Next to study the impact of migration on Hadoop, experiments are conducted on a cluster with 8 slave and 1 master node.

4.2 Performance of Hadoop on Cloud

This section discusses the performance of Hadoop with different cluster sizes. As shown in Table 1, job completion time(in sec) decreases with increase in cluster size. However with small file size, this decrease is small as Hadoop split the file into chunks(known as splits) depending upon the split size(default 64 MB) and for each split it usually has one map task running on it. With small file size the number of splits and accordingly the number of map tasks are less. So, even if the number of VMs in the cluster are increased the job completion time is not reduced much as number of map tasks are not sufficient to utilize the increased computational power. Further, Tables 2 and 3 shows the time(in msec) taken by map and reduce task to process different input file on different cluster size.



(a) Hadoop job

(b) Map task

(c) Reduce task

Fig. 1. Effect of live migration on Hadoop cluster

4.3 Performance of Hadoop with VM Migration

This section evaluates the performance of Hadoop cluster when different number of VMs are migrated within the same subnet. Three performance parameters job completion time, migration time and downtime are used to estimate the impact of migration on Hadoop.

As shown in Fig. 1(a), job completion increases with increase in number of VM migration. This is due to the network contention between Hadoop traffic and migration traffic at source and destination. Also, for same number of VMs that are migrated the performance is more degraded when input file size is large as the amount of dirty memory generated with large file size is more. Figures 1(b) and (c) shows that time taken by map and reduce task increases with increase in the number of VM migration which is again due to the network contention of map-reduce traffic with migration traffic.

Next, the effect of migration on average downtime and average migration time is analyzed. The performance of Hadoop cluster running MapReduce job on 1024 MB input file is compared with an idle cluster. As shown in Table 4 the average downtime increases with increase in number of migrations. It is because of sharing of bandwidth reserved for migration among the VMs. So, with more VMs the bandwidth available per VM is reduced resulting in increased migration time. Further, Table 4 shows that average migration time increases with increase in number of VM migrations which is due to increase in migration traffic and reduction in per VM available bandwidth for migration.

Lastly, the effect of migrating master and slave node on job completion time is evaluated. As shown in Table 5 migrating the master increases the job completion time for a give file size in comparison to the slave. This is because the master

Table 4. Effect of migration on average downtime and migration time

ADT:Avg. Downtime, AMT:Avg. Migration time				
Node	ADT/AMT			
	ADT(idle) (msec)	ADT(running) (msec)	AMT(idle) (sec)	AMT(running) (sec)
2	219	247	21	22
4	384	408	34	42
6	411	430	48	68
8	424	613	60	81

Table 5. Effect of migrating master and slave VM on job completion time(in sec) with different file size

Node	File size			
	128 MB	256 MB	512 MB	1024 MB
Master	65	64	129	182
Slave	61	61	104	150

is in frequent communication with the slaves so migrating the master has more impact on performance and job completion time.

5 Conclusion

In this paper a series of experiments are conducted to analyze the performance of Hadoop with VM migration. Experiment shows that job completion time, average downtime as well as average migration time gets increased with increase in the number of VMs that are migrated. Further, for the same number of VM migration, job completion time is higher with large file size. So, in case multiple Hadoop clusters of same size are running and decision has to be made to select appropriate VM to migrate, Hadoop VM running on small file size should be selected to reduce the impact of migration on performance. Further, among master or slave, slave VM should be selected for migration as the job completion time is higher when master is migrated. In future, experiment will be conducted to analyze the performance of Hadoop cluster with VM migration across WAN where network latency is high and no shared storage is available.

References

1. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2008)
2. Borthakur, D.: The hadoop distributed file system: architecture and design. Hadoop Proj. Website **11**(2007), 21 (2007)
3. Ibrahim, S., Jin, H., Lu, L., Qi, L., Wu, S., Shi, X.: Evaluating MapReduce on virtual machines: the Hadoop case. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) CloudCom 2009. LNCS, vol. 5931, pp. 519–528. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10665-1_47
4. Xu, G., Xu, F., Ma, H.: Deploying and researching hadoop in virtual machines. In: International Conference on Automation and Logistics (ICAL), 2012 IEEE, pp. 395–399. IEEE (2012)
5. Ye, K., Jiang, X., Chen, S., Huang, D., Wang, B.: Analyzing and modeling the performance in xen-based virtual cluster environment. In: 12th IEEE International Conference on High Performance Computing and Communications (HPCC), 2010, pp. 273–280. IEEE (2010)
6. Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live migration of virtual machines. In: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation, vol. 2, pp. 273–286. USENIX Association (2005)
7. Hwang, J., Zeng, S., Wu, F.Y., Wood, T.: A component-based performance comparison of four hypervisors. In: IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), 2013, pp. 269–276. IEEE (2013)
8. Johnson, C., Chiu, D.: Hadoop in flight: Migrating live mapreduce jobs for power-shifting data centers. In: IEEE 9th International Conference on Cloud Computing (CLOUD), 2016, pp. 92–99. IEEE (2016)
9. Kambatla, K., Pathak, A., Pucha, H.: Towards optimizing hadoop provisioning in the cloud. HotCloud **9**, 12 (2009)

Performance Analysis of Parallel K-Means with Optimization Algorithms for Clustering on Spark

V. Santhi(✉) and Rini Jose

PSG College of Technology, Anna University, Coimbatore, Tamil Nadu, India
vsr@cse.psgtech.ac.in

Abstract. Clustering divides data into meaningful, useful groups known as clusters without any prior knowledge about the data. One of the drawbacks of K-Means clustering is the estimation of initial centroids which influence the performance of the algorithm. To overcome this issue, optimization algorithms like Bat and Firefly are executed as pre-processing step. These algorithms return optimal centroids which is given as input to the K-Means algorithm. Clustering is carried out on large data sets, therefore Apache Spark, an open source software framework is used. The performance of the optimization algorithms is evaluated and the best algorithm is determined.

Keywords: Clustering · K-Means · Bat algorithm · Firefly algorithm
Big data · Spark

1 Introduction

In today's world, increasing amount of data is being generated and collected. Every traditional algorithm which exists for handling significant amount of data proves to be inefficient when data sets get bigger and the iterations for fixing certain initial parameters also increase. In K-Means clustering the numbers of iterations depend on the initial centroids. The accuracy of the initial centroids taken has a major impact on the time taken for completion of clustering. This project is an attempt to mitigate this problem by modifying K-Means algorithm [1] using optimization algorithms like Bat [2] and Firefly algorithm [3] in Apache Spark framework.

K-Means clustering has always been one of the most commonly used machine learning techniques for the data clustering problems. For being such a commonly used algorithm it still has two major drawbacks that lead to unnecessary increase in the execution time. First drawback is the number of clusters which is to be given by the end user; which is the k value. The next is the initial k centroids taken, for which there are certain practices followed, such as the factor of taking the ordering into account, where whichever data point comes in first k tuples is taken, and when the order changes the data points taken initially also change, therefore bringing a difference in the execution time by increasing the number of iterations. Unfortunately when the initial data points tend to be outliers, it would lead to unnecessary iterations to locate the actual centroid points in the first place.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 158–162, 2018.
https://doi.org/10.1007/978-3-319-72344-0_12

Moreover, when handling huge datasets, running an algorithm on a piled up data set is difficult and execution time is high which is to be minimised. Our objective is to optimize the K-Means clustering technique by using optimization algorithms and evaluate their performance. These optimization algorithms acts as the pre-processing step to identify the initial centroid points and the implementation is carried out on a large dataset in the Apache spark framework. The optimization is depicted by comparing the various measures of the K-Means algorithm with chosen optimization techniques.

2 Related Work

In [4], discussed the survey of clustering algorithms for data sets appearing in statistics, computer science, and machine learning, and illustrated their applications in some benchmark data sets. In [5], proposed an optimized K-Means clustering technique using Bat algorithm (KMBA), KMBA algorithm does not require the user to give the number of clusters in advance. In [6], proposed clustering using Firefly algorithm. In [7], proposed parallelizing K-Means-based clustering on Spark. Exploring the parallel implementations of two-phase iterative procedure on Spark is not only universal to a wealth of clustering algorithms but also meets the practical needs addressed by big data. In [8], proposed the K-Means data clustering on Spark, for improving sales performance in sales data of a super market. In [9], proposed scalable parallel clustering approach using parallel K-Means and Firefly algorithm for up to thousand kilo bytes of data. In [10], proposed the intelligent K-Means algorithm on Spark for big data clustering.

3 Proposed System

The proposed system aims to optimize data clustering in a distributed environment. Bat algorithm and Firefly algorithm have been used to optimize the performance of K-Means clustering algorithm in Apache Spark environment. The optimization algorithms have individually been used as a pre-processing step to give the initial centroids as input to the K-Means clustering algorithm. The data clustering is performed in Apache Spark framework.

The following steps are taken in the proposed system:

1. Initially the data set is available in HDFS.
2. The HDFS data is converted in to number of RDDs.
3. Initial Optimal Centroid K is chosen from Optimization algorithm.
4. This K value is feed into K-Mean algorithm
5. Then calculate the distance of each object from centroids using Euclidean distance.
6. Place the object in the nearest cluster.
7. Find the new centroid for each cluster.

8. The process is continued in an iterative manner and steps 5, 6 and 7 are repeated until when the stopping condition are met.

The following steps show how the proposed system is run on the spark environment:

1. In the Spark environment, 1 driver node and n worker nodes are created.
2. The data set is divided into number of worker nodes.
3. Initially the K value is chosen from optimization algorithm. This value is broadcasted into all worker nodes.
4. The worker node allocates each data object to one cluster according to the cluster centroid it receives.
5. Each and every worker node's data objects are clustered together; finally it finds the local clustering centroid. This new local centroid along with the group of data objects are transferred to the driver node.
6. When the driver node receives all local centroid from all worker nodes, then it finds the global new centroid with these values.
7. This process is repeated and step 4 to 6 is done until when the distance between new centroid and the older centroid are equal to less than the threshold value.
8. The finalized centroid value is taken.

Here all the processes are done on different RDDs simultaneously. This will reduce the complexity of the proposed algorithm and at the same time the clusters are well refined with optimization algorithm. The disadvantage of K-Means algorithm is overcome by merging of optimization algorithm with K-Means algorithm.

4 Experiments and Results

KDD Cup 99 data set [11] from UCI repository is used here. This data set is based on evaluation of anomaly detection methods. It has approximately 4,900,000 rows and each row contains 41 features with 23 classes. From this dataset, only 20% sample of the data set is taken. The dataset is taken based on scalable K-means clustering [12]. One driver node and 2 worker nodes are created for Spark in virtual machine environment and HDFS file system is used for storing the original data set. The initial value of Bat and Firefly algorithm are set randomly for finding optimal centroid. The experiments are done using Scala without machine learning libraries. The performance analysis of proposed system is done based on three metrics such as purity, NMI and execution time.

4.1 Purity

Purity value is range from 0 to 1. Both of the optimization algorithms provide the purity values nearly 1. Figure 1 shows the purity values.

4.2 NMI

NMI is normalized mutual information. It is the mutual information of the clustering results and number of classes. Purity alone is not suitable for measuring cluster since the number of cluster goes high, Purity always gives nearly 1.Unlike purity, when the number of clusters goes high, NMI does not necessarily grow. This value is also ranges from 0 to 1 and higher values indicate better clustering results. Figure 2 shows the NMI values.

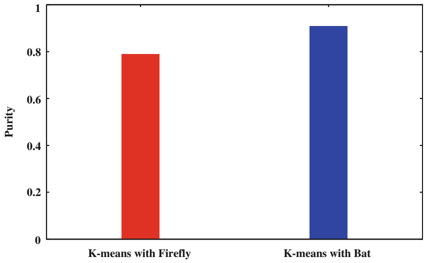


Fig. 1. Purity

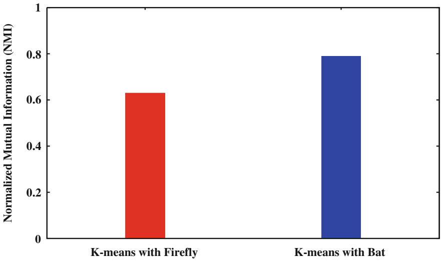


Fig. 2. Normalized mutual information

4.3 Execution Time

This is the time to be involved for completion of K-Means with optimization algorithms under Spark environment. The execution is measured based on without machine learning libraries. Table 1 is shown below depicts the execution time of K-Means clustering algorithm with the optimization algorithms as pre-processing step. K-Means with Firefly algorithm requires more number of iteration to converge than K-Means with Bat optimization.

Table 1. Comparison of execution time

S.no	Algorithm	Execution time (in sec)
1.	K-Means	1815.64
2.	K-Means with Bat Optimization	1109.23
3.	K-Means with Firefly Optimization	1317.42

5 Conclusion

The initial centroids obtained from Bat and Firefly optimization algorithms are given as input to the K-Means clustering algorithm. The modified K-Means is executed in Apache Spark environment and hence it is fast even when the dataset is large. The performance is evaluated by calculating the execution time of these algorithms. The results are compared and the best algorithm is determined. In future, the complete KDD Cup 99 data set is taken for perfomance evalution.

References

1. Kanungo, T., Netanyahu, N.S., Wu, A.Y.: An efficient k-means clustering algorithm: analysis and implementation. IEEE Trans. Pattern Anal. Mach. Intell. **24**, 881–892 (2002)
2. Yang, X.S.: Bat algorithm: literature review and applications. Int. J. Bio-Inspir. Com. **5**, 9–141 (2013)
3. Yang, X.S.: Firefly algorithms for multimodal optimization. In: Watanabe, O., Zeugmann, T. (eds.) SAGA 2009. LNCS, vol. 5792, pp. 169–178. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04944-6_14
4. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Trans. Neural Netw. **16**, 645–678 (2005)
5. Komarasamy, G., Wahi, A.: An optimized k-means clustering technique using bat algorithm. Eur. J. Sci. Res. **84**, 263–273 (2012)
6. Senthilnath, J., Omkar, S.N., Mani, V.: Clustering using firefly algorithm: performance study. Swarm Evol. Comput. **1**, 164–171 (2011)
7. Wang, B., Yin, J., Hua, Q., Wu, Z., Cao, J.: Parallelizing k-means-based clustering on spark. In: IEEE International Conference on Advanced Cloud and Big Data, Chengdu, China (2016). https://doi.org/10.1109/CBD.2016.016
8. Huang, Q., Zhou, F.: Research on retailer data clustering algorithm based on spark. In: AIP Conference Proceedings (2017). https://doi.org/10.1063/1.4977378
9. Mathew, J., Vijayakumar, R.: Scalable parallel clustering approach for large data using parallel k means and firefly algorithms. In: IEEE International Conference on High Performance Computing and Applications, Bhubaneswar, India (2014). https://doi.org/10.1109/ICHPCA.2014.7045322
10. Kusuma, I., Ma'sum, M.A., Habibie, N., Jatmiko, W., Suhartanto, H.: In design of intelligent k-means based on spark for big data clustering. In: IEEE International Workshop on Big Data and Information Security, Jakarta, Indonesia (2016). https://doi.org/10.1109/IWBIS.2016.7872895
11. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
12. Bahmani, B., Moseley, B., Vattani, A., Kumar, R., Vassilvitskii, S.: Scalable k-means ++. In: Proceedings of the VLDB Endowment, pp. 622–633 (2012)

Hashing Supported Iterative MapReduce Based Scalable SBE Reduct Computation

U. Venkata Divya[1(✉)] and P. S. V. S. Sai Prasad[2]

[1] Quadratic Insights Pvt Ltd., Hyderabad, India
uvdivya.iiit@gmail.com

[2] School of Computer and Information Sciences,
University of Hyderabad, Hyderabad, India
saics@uohyd.ernet.in

Abstract. Feature Selection plays a major role in preprocessing stage of Data mining and helps in model construction by recognizing relevant features. Rough Sets has emerged in recent years as an important paradigm for feature selection i.e. finding Reduct of conditional attributes in given data set. Two control strategies for Reduct Computation are Sequential Forward Selection (SFS), Sequential Backward Elimination(SBE). With the objective of scalable feature seletion, several MapReduce based approaches were proposed in literature. All these approaches are SFS based and results in super set of reduct i.e. with redundant attributes. Even though SBE approaches results in exact Reduct, it requires lot of data movement in shuffle and sort phase of MapReduce. To overcome this problem and to optimize the network bandwidth utilization, a novel hashing supported SBE Reduct algorithm(MRSBER_Hash) is proposed in this work and implemented using Iterative MapReduce framework of Apache Spark. Experiments conducted on large benchmark decision systems have empirically established the relevance of proposed approach for decision systems with large cardinality of conditional attributes.

Keywords: Rough Sets · Reduct · Iterative MapReduce
Apache Spark · Scalable feature selection

1 Introduction

The field of Rough Sets [5] was introduced in 1980's by Prof. Pawlak as a soft computing paradigm for data analysis amidst vagueness and uncertainty. Reduct computation(feature subset selection) is an important application of Rough Sets in Data Mining. Two primary control strategies for Reduct Computation are Sequential Forward Selection(SFS) and Sequential Backward Elimination(SBE). The SFS approaches, though computationally efficient, have disadvantage in resulting in super set of reduct. SBE algorithm results always in exact reduct without any redundancy.

Standalone Reduct computation approaches suffer from scalability issues with large decision systems. For scalable reduct computation, several MapReduce

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 163–170, 2018.
https://doi.org/10.1007/978-3-319-72344-0_13

based distributed/parallel approaches [6,8,9] for SFS based reduct computation were developed in literature in frameworks such as Hadoop [10], Twister [4], Apache Spark [11].

In our literature review we have not found any MapReduce based SBE implementations. In this work we identify the problems in MapReduce based SBE Reduct computation and design and develop an approach called MRSBER_HASH for overcoming these problems. The developed approach facilitates exact Reduct computation for very large scale decision systems. The proposed approach also can be utilized as a post processing optimization step in SFS based MapReduce algorithms for generation of exact Reduct out of super Reduct obtained.

2 SBE Based Reduct Computation

The basics of Rough Sets and approaches for reduct computation are given in [6]. Classical Rough Sets are applied to complete symbolic decision system DT defined as [7]

$$DT = (U, C \cup \{d\}, \{V_a, f_a\}_{a \in C \bigcup \{d\}}) \tag{1}$$

where U: Set of Objects, d: Decision attribute, C: Set of Conditional Attributes, for each a $\in C \cup \{d\}$, V_a is domain of a and $f_a : U \rightarrow V_a$ is value mapping for attribute a.

Reduct is a subset of a features that are individually necessary and jointly sufficient in order to maintain a heuristic dependency measure of a decision system. If M denotes heuristic dependency measure Reduct R is a minimal subset of conditional attribute set C such that $M(R) = M(C)$. The structure for SBE reduct computation is given in Algorithm 1.

Algorithm 1. Sequential Backward elimination Algorithm

```
Input Decision table $DT = (U, C \cup \{d\}, \{V_a, f_a\}_{a \in C \cup \{d\}})$
Dependency Measure ($M$)
Output Reduct ($R$)
Ranking of attributes based on Dependency Measure $M$
R ← C
for each attribute a in Ranking Order do
  if M(R−{a})== M(C) then
    R← R −{a}
  end if
end for
return R
```

SBE algorithm start with initialization of Reduct R to all conditional attributes, and involves $|C|$ iterations. In each iteration, a conditional attribute

a is tested for redundancy based on given dependency measure M. An attribute $'a'$ is said to be redundant, if $M(R - \{a\})$ is equal to $M(R)$. If an attribute is found to be redundant, then it is dropped from R, otherwise it is retained. After $|C|$ iterations R contains irreducible set of attributes satisfying $M(R) = M(C)$. Hence SBE approach always result in exact Reduct. Sorting the attributes to be used for redundancy check from least significant(individually) to highest, helps in retaining more potential attributes in the obtained Reduct.

In the proposed approach conditional information entropy(CIE) is used as dependency measure. The CIE of $B \subseteq C$ with respect to decision attributes {d} is defined as

$$E(\{d\}/B) = \sum_{g \epsilon IND(B)} P(g) \sum_{g^1 \epsilon g/IND(\{d\})} P(g^1)\log_2(P(g)) \tag{2}$$

where $p(g) = \frac{|g|}{|U|}$ and $p(g^1) = \frac{|g^1|}{|g|}$. Here $IND(B)$ denotes rough set based indiscernability relation defined as

$$IND(B) = \{(x, y) \in U^2/f_a(x) = f_a(y), \forall a \in B\} \tag{3}$$

$IND(B)$ is an equivalence relation and partition of U induced by $IND(B)$ is denoted by $U/IND(B)$ which is a collection of distinct equivalence classes or granules. An equivalence class of $x \in U$ is denoted by $[x]_b$. An equivalence class is said to be a consistent granule if all objects of equivalence class belongs to the same decision class, otherwise is said to be inconsistent.

3 Proposed MRSBER_Hash Algorithm

The proposed approach for iterative MapReduce based SBE Reduct algorithm is arrived with an objective of preserving scalability. The MapReduce based approach for $M(B)$ computation is usually done by a common pattern [8,9]. Computing $M(B)$ requires computation of summary information for granules of quotient set $U/IND(B)$ and arriving at $M(B)$ using the summary information of granules. In MapReduce approach $< key, value >$ for each object is formed by setting *key* to granule signature and *value* to required information for M computation. Granule signature is the domain value combination for B satisfied by all the objects of granule. In SBE Reduct algorithm, especially in the beginning iterations, $|B|$ is nearly equal to $|C|$. Owing to Curse of Dimensionality principle, for decision systems of large cardinality of attributes, number of granules $|U/IND(B)|$ is in order of $|U|$. In decision systems with $|U/IND(C)|$ is near to $|U|$, the size of data communicated from mappers to reducers is in the order of original dataset. This can become a bottleneck and hamper the scalability of MapReduce based SBE Reduct algorithm.

Hence, the proposed approach is evolved with the objective of reducing the amount of data transferred in shuffle and sort phase of MapReduce. Algorithm MRSBER_Hash is the distributed/parallel approach for SBE Reduct computation using iterative MapReduce Framework. Initially ranking of attributes in the decreasing order of CIE is obtained using a single MapReduce job. The procedure for computing CIE for each attribute using MapReduce is adopted from [8].

Datasets of large cardinality of attributes have the least amount of inconsistent granules. Removal of them will make the decision system to be consistent without significant impact on resulting reduct obtained. Hence in MRSBER_Hash algorithm another MapReduce job is invoked for extracting objects in inconsistent granules and removing them for formation of consistent decision system(CDS).

SBE Reduct computation in CDS is simplified, as the granules of $IND(C)$ are all consistent, redundancy check of an attribute a does not require exact computation of $M(R - \{a\})$(refer to Algorithm 1) but only requires any inconsistent granule is present or not in $U/IND(R - \{a\})$. This facilitate optimization in amount of data transferred as part of *value* portion in MapReduce. Rest of the section explains how a two stage process is developed for inconsistency verification. The SBE reduct is obtained by following this two stage process for inconsistency verification for each attribute in the rank order of CIE.

Let the given decision system $DT = (U, C \cup \{d\}, \{V_a, f_a\}_{a \in C \cup \{d\}})$ be horizontally partitioned into $DT_1, DT_2, ..., DT_n$. Here DT_i= $(U_i, C \cup \{d\}, \{V_a, f_a\}_{a \in C \cup \{d\}})$ for 1≤i≤n and $\{U_i\}$ forms a partition of U. n mappers working parallel on DT_i's produces $< key, value >$ pairs for each data object of their portions. At mapper level a local optimization(as in reduceByKey of Spark's MapReduce) helps in arriving at $< key, value >$ pairs for partial granules $U_i/IND(B)$. Each reduce invocation works with list of values from all mappers for the same key(granule signature) to arrive at required computation for granule of $U/IND(B)$. In the normal approach for inconsistency verification, at mapper level the generated $< key, value >$ pair represents $< key >$ as granule signature and value as $f_d(o)$. At reducer level, a granule is found to be inconsistent, if multiple decision values are associated with the same granule signature.

In order to avoid the communication of keys as granule signatures, the above process is divided into two stages. A hash function(HF) is employed and $< key >$ is set to hashed value of granule signature instead of granule signature. This helps in passing a single number, instead of $|B|$ numbers to reducers. Across all mappers the reduction in memory for keys in shuffle and sort phase is from the order of $|U * B|$ to $|U|$(in situation of $|U/IND(B)| \approx |U|$). In practice, one can assume the mapping to obey many-one property. This results in a reduce invocation working with coalesced granule g^* representing possibly multiple granules whose granule signature, is mapped to the same hash value. If g^* is found to be consistent then, all the granules coalesced into g^* will also be consistent. But the inconsistency of g^* will not automatically imply inconsistency of comprised granules.

In stage-2, the keys(Hashed values) of inconsistent granules resulting from stage-1 are broadcasted to mappers. Each mapper generates $< key, value >$ pairs for only those objects whose hashed values is present in inconsistent hash values. Here *key* is set to granule signature and *value* is set to decision value. Similar to normal procedure the reduce invocation determines the occurrence of inconsistency. The objective of two stage process is realized when the cardinality of the objects participating in $< key, value >$ generation in stage-2 is much lesser than $|U|$. In our exploration of different hash functions, deepHashCode function available in (java.util.Arrays class) is found to be a suitable choice and employed in our experiments.

3.1 Relevance of Two Stage Hash Based Approach

To illustrate the obtained benefits with two stage approach Table 1 summarizes the size of key space(cardinality of granules*size of key) in stage-1 and stage-2 for an iteration for datasets used in our experiments. Assuming that two stage process is not followed and granule signature is taken as key value, the resulting size of key space is also reported under normal SBE column in Table 1. The results are significant as number of keys under shuffle and sort phase in stage-2 of MRSBER_Hash are very few and most of the data transfer is in stage-1 involving keys consisting of singleton hash numbers instead of granule signatures of order $|C|$ as in Normal SBE.

Table 1. Size of key space

Dataset	Normal SBE	MRSBER_Hash stage1	MRSBER_Hash stage2
KDD	57822*40	54575*1	6*40
Gisette	6000*5000	5999*1	2*5000

4 Experiments and Analysis of Results

The details of the datasets used in experimentation are described in Table 2. Gisette is from UCI repository [3] and KDDCUP99 [2] is from UCI KDD Archive. Experiments were conducted on Baadal [1] cloud computing infrastructure, an initiative of Ministry of Human Resource Development, Government of India. Baadal is developed and supported by IIT-Delhi. A five node cluster environment is obtained on Baadal, where each node is with the following hardware and software configuration. Each node has 8 cpu cores and 8 GB of RAM. Each node is installed with Ubuntu 14.04 Desktop amd64, Java 1.7.0.131, Scala 2.10.4, sbt.13.8, Apache spark 1.6.2. In these 5 nodes one is set as master and the other 4 nodes are set as slaves.

Table 2. Description of dataset

Dataset	Objects	Features	Classes	Consistency
Gisette	6000	5001	2	Yes
kddcup99	4898431	41	23	No

4.1 Comparative Experiment with SFS MapReduce Approaches

Proposed algorithm(MRSBER_Hash) is compared with several SFS Reduct Computation approaches which are implemented using iterative MapReduce paradigm available in literature [8–10,12]. Algorithms PLAR_PR, PLAR_LCE, PLAR_SCE, PLAR_CCE [12] are Iterative MapReduce based SFS Reduct computation algorithms using dependency measures PR(gamma measure), various conditional information entropy measures LCE, SCE, CCE. These algorithms were proposed by Junbo Zhang et al. in 2016 incorporating features of granular computing based initialization, model parallelism and data parallelism. The implementation was done in Apache Spark.

Algorithm IN_MRQRA_IG was given by Praveen Kumar Singh et al. [9] in 2015, is an iterative MapReduce based distributed algorithm for QRA_IG [6] in 2015. IN_MRQRA_IG is implemented in Indiana University's Twister environment. Balu et al. in 2016 have given MRIQRA_IG [8] as an improvement to IN_MRQRA_IG as a distributed implementation of IQRA_IG algorithm using Twister's framework. PAR algorithm was given by [10] Yong Yang et al. in 2010 using Hadoop MapReduce framework. The cluster configuration involved in each of these implementation are different and details are summarized in Table 3. The results reported with respect to these algorithms are as given in the respective publications.

Table 3. Cluster configuration

	PLAR	IN_MRQRA_IG	MRIQRA_IG	PAR
Cluster Size	19	4	6	11
RAM SIZE	Atleast 8 GB	4 GB	4 GB	*
Cores	Atleast 8	4	4	*
OS	Cent OS 6.5	OpenSuse-12.2	OpenSuse	*
Software	Spark	Twister	Twister	Hadoop

Comparative Experiments with KDDCUP99: The results of Reduct length obtained, computation time in seconds are provided in Table 4. The results of KDDCUP99 establish the need for MapReduce based SBE Reduct algorithm. In contrast to SFS approaches giving a super Reduct of 31 attributes MRSBER_Hash has resulted in exact Reduct of 25 attributes. The only way SFS approaches can result in exact Reduct is by augmenting SBE approach on obtained super Reduct. Computational time of MRSBER_hash is significantly higher than MRIQRA_IG and PLAR based algorithms. The computation efficiency of PLAR based algorithms is primarily due to high configuration cluster of 19 nodes and capability to utilize model parallelism on supporting infrastructure. By model parallelism, in SFS approach, authors meant initiation of separate jobs for obtaining candidate attribute sets significance. The proposed algorithms, being SBE based lack the advantages of granular computations as in PLAR

Table 4. Comparison with KDDCUP99 dataset

	Computation Time(sec)	Reduct Length
MRSBER_Hash	184	25
PLAR-PR	8	*
PLAR-LCE	8	*
PLAR-SCE	8	*
PLAR-CCE	8	*
MRIQRA_IG	68.84	31
IN_MRQRA_IG	1947.338	31
PAR	5050	*

Note: * not reported in original publication.

approaches, positive region elimination as in MRIQRA_IG algorithm. It is to be noted that, proposed algorithms achieved much better performance than SFS approaches IN_MRQRA_IG, PAR which does not posses these optimization's. As a first attempt towards MapReduce based SBE Reduct computation, our proposed algorithm achieved comparable performance with leading approaches in SFS.

Comparative Experiments with Gisette Dataset: Gisette dataset is contrasting one from KDDCUP99 having much larger cardinality in attributes and smaller cardinality in objects. The results for Gisette dataset in [12] only reported for first 5 iterations(Selection of five attributes into Reduct Set) with varying model parallelism level. PLAR_SCE has incurred 30806 s without model parallelism, 15293 s with model parallelism level of 2(supporting two parallel MapReduce jobs at any instance) and 1856 s with level of 64. With much simpler computational infrastructure, MRSBER_Hash could complete the Reduct computation and resulted in a Reduct of size 23 in 4929 s.

Assuming that PLAR_SCE in [12] completed the Reduct computation with 23 attributes, the estimated computational time will be 8537 s. Hence, it can be deduced that in datasets of higher cardinality of attributes MRSBER_Hash can obtain better computational performance than SFS approach. In our opinion, this may be due to fewer number of MapReduce jobs initiated in SBE approach. To be specific, an SBE approach requires $|C|$ MapReduce jobs and an SFS approach followed in PLAR methods with model parallelism require $|C| + (|C| - 1|) + (|C| - 2|) + ... + (|C| - |R| + 1)$ MapReduce jobs.

5 Conclusion

In this work MRSBER_Hash algorithm is developed as a MapReduce based SBE Reduct computation approach. MRSBER_Hash was designed with an objective of minimizing data transfer in shuffle and sort phase of MapReduce. In extensive comparative analysis with leading MapReduce based SFS Reduct computation approaches mixed results are obtained. MRSBER_Hash is found to be achieving better computational performance with datasets of large cardinality of conditional attributes. In future, MRSBER_Hash will be further improved to achieve similar or better computational performance than SFS approaches such as MRIQRA_IG. Irrespective of performance in computational time aspect, MRSBER_Hash provides an approach for exact Reduct computation in large scale decision systems.

References

1. Baadal: the iitd computing cloud (2011). http://www.cc.iitd.ernet.in
2. Dataset used for experiments (1999). http://kdd.ics.uci.edu/databases/kddcup99/
3. Uci machine learning repository. https://archive.ics.uci.edu/ml/datasets (2013)
4. Ekanayake, J., Li, H., Zhang, B., Gunarathne, T., Bae, S.-H., Qiu, J., Fox, G.C.: Twister: a runtime for iterative mapreduce. In: HPDC, pp. 810–818. ACM (2010)
5. Pawlak, Z.: Rough sets. Int. J. Parallel Program. **11**(5), 341–356 (1982)
6. P.S.V.S., S.P., Raghavendra Rao, C.: Extensions to IQuickReduct. In: Sombattheera, C., Agarwal, A., Udgata, S.K., Lavangnananda, K. (eds.) MIWAI 2011. LNCS (LNAI), vol. 7080, pp. 351–362. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25725-4_31
7. Shen, Q., Jensen, R.: Rough set-based feature selection: a review. In: Rough Computing: Theories, Technologies and Applications, pp. 70–107. IGI Global (2008)
8. Sai Prasad, P.S.V.S., Bala Subrahmanyam, H., Singh, P.K.: Scalable IQRA_IG algorithm: an iterative MapReduce approach for reduct computation. In: Krishnan, P., Radha Krishna, P., Parida, L. (eds.) ICDCIT 2017. LNCS, vol. 10109, pp. 58–69. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-50472-8_5
9. Singh, P.K., Sai Prasad, P.S.V.S.: Scalable quick reduct algorithm: iterative mapreduce approach. In: CODS, pp. 25:1–25:2. ACM (2016)
10. Yang, Y., Chen, Z., Liang, Z., Wang, G.: Attribute reduction for massive data based on rough set theory and MapReduce. In: Yu, J., Greco, S., Lingras, P., Wang, G., Skowron, A. (eds.) RSKT 2010. LNCS (LNAI), vol. 6401, pp. 672–678. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16248-0_91
11. Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I.: Spark: Cluster computing with working sets. In: HotCloud. USENIX Association (2010)
12. Zhang, J., Li, T., Pan, Y.: Parallel large-scale attribute reduction on cloud systems. CoRR, abs/1610.01807 (2016)

Networks Protocols and Applications

Reliable Condition Monitoring of Telecommunication Services with Time-Varying Load Characteristic

Günter Fahrnberger(✉)

University of Hagen, Hagen, North Rhine-Westphalia, Germany
guenter.fahrnberger@studium.fernuni-hagen.de

Abstract. In general, SLAs (Service-Level Agreements) between TSPs (Telecommunication Service Providers) and their computer system vendors contain grants of penalty demands on the vendors in case of SLA violations. Occasionally, TSPs also cede such rights to their customers. In this case, TSPs behave wisely if they install CMSs (Condition Monitoring Systems) that nonstop supervise all significant KPIs (Key Performance Indicators) of their services and red-flag noticeable service problems. Scientists have researched a variety of concepts for CMSs with machined dynamic thresholds, for instance, to take the material aging of rotary machines into account. Nary such a concept deftly copes with time-based volatility, e.g. telecommunication services that show time-varying load characteristic. This disquisition fills this gap by presenting the requirements, the architecture, and the reliability analysis for an applicable CMS (Condition Monitoring System).

Keywords: CM · CMS · Condition Monitoring
Condition Monitoring System · Icinga · Measurement · Monitoring
Nagios · Prediction · Supervision · Surveillance

1 Introduction

Nowadays, many service companies commit themselves to an SLA (Service-Level Agreement) for their proffered services [11]. This does not happen due to pure altruism. Rather, they want to emphasize the virtues of their products with a prudent SLA to obtain competitive advantages.

TSPs as a subset of CSPs (Communication Service Providers) also keep pace with the pursuit of SLAs. Their consumer customers mostly can merely retrieve the personal performance statistics for their calls, short messages, and data sessions from credentials-secured web portals. In contrast, their business customers may rather opt for SLAs with KPIs and accordant thresholds whose exceeding might even award business customers the receipt of monetary penalties. The availability oftentimes excels as the most important KPI of a service. Table 1 gives the idea of a portfolio with availability classes that could be stipulated in an SLA.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 173–188, 2018.
https://doi.org/10.1007/978-3-319-72344-0_14

Table 1. Availability of system classes [7]

System type	Unavailability (Minutes/Year)	Availability (Percent)	Availability class
Unmanaged	50,000	90	1
Managed	5,000	99	2
Well-managed	500	99.9	3
Fault-tolerant	50	99.99	4
High-availability	5	99.999	5
Very-high-availability	0.5	99.9999	6
Ultra-availability	0.05	99.99999	7

It goes without saying that TSPs leave nothing undone to avert SLA violations and resultant penalty demands on them. Their business customers could overtly or clandestinely verify the adherence to the negotiated KPIs by dint of random or permanent inspection with their own supervising system. Therefore, TSPs usually deploy CMSs that continuously record crucial KPIs and alert appropriate personnel in the event of threshold exceedances to take remedial measures. The evaluation of KPIs with a CMS belongs to the functional areas of fault management and performance management [8]. A CMS further conduces to accounting management, configuration management, and security management, which do not directly contribute to the evaluation of KPIs.

TSPs are strongly advised to define more rigorous thresholds in their CMSs than in the SLAs with their customers. Apart from the compliance with agreed KPIs, the prior objective of TSPs must be the ascertainment and reparation of perturbations prior to their customers' cognition to achieve customer satisfaction [2].

How can TSPs determine optimal thresholds for KPIs? The determination process allegorizes a tightrope walk between missing and false alarms [9]. While missing alerts because of too permissive thresholds pose the risk of unrecognizable disturbances, false alarms as a consequence of too strict thresholds annoy and desensitize the alerted staff. Desensitization entails the employees' negligence or ignorance of all alarms, i.e. also of genuine ones. To make a long story short, the eventual implications of missing and false alerts resemble.

Especially in the telecommunications industry, services with time-varying intensities of use make threshold adjustments harder. While the same threshold value perfectly fits for a certain period, it could be completely inappropriate for other times of day. Such services necessitate the renunciation of fix thresholds in favor of time-based ones. It ought to be comprehensible for everyone that merely machines can efficiently and steadily derive such time-based thresholds and put them into effect.

A thorough literature scrutiny has made clear that many approaches about automatic threshold computation for machines with rotary constituents exist, but none for telecommunication services with time-varying load characteristic. Hence, this disquisition expunges this deficit with a novel proposal how reliable CM (Condition Monitoring) can be established for the latter.

Therefor, Sect. 2 specifies all demands on a prudential CMS for telecommunication services with time-varying load characteristic. Section 3 reasons why nary an existent scholarly piece meets these requirements and, thence, the literature benefits from the merits of this treatise. Section 4 as the centerpiece of this paper presents the detailed idea of a CMS architecture that ultimately complies with the contrived demands. The reliability analysis in Sect. 5 visualizes the accuracy of different threshold calculation parameters for real KPIs computed in a productive CMS. Section 6 comes up with a recapitulatory conclusion and suggestions for valuable future work.

2 Requirements

The needs of a sophisticated CMS for telecommunication services with time-varying load characteristic definitely differ from those of machines with rotary components. Thus, this section explicates the requirements that the proposed system architecture in Sect. 4 must fulfill.

2.1 Availability

The introduction already heralded that the desired CMS must be able to sense the KPIs of services irrespective of their availability class. On the account of this, the CMS itself also has to be highly available for two main purposes. First, surveillance intermissions, which occur during KPI deteriorations, delay triggering the requisite alarm notifications until supervision continues. Second, discontinuous histograms with missing or even wrong data values impair future threshold calculations and, on account of that, take hazards of absent or unwanted notifications.

If a cluster of replicated nodes cares for the required high availability of the CMS, also an unintended failover or a deliberate switchover between nodes must not cause the aforementioned voids in the series of historical data points.

2.2 Accuracy

An ideal CMS would facilitate perfect accuracy and, on account of this, no missing and no false alerts. Such a consummate system remains wishful thinking due to irrepressible influencing factors on KPIs (e.g. commercial or societal implications), but the minimization of absent and unnecessary alerting though is a valid objective.

Reasonably predicted thresholds for a KPI derived from its history shape up as the veritable key to success.

2.3 Robustness

In spite of the request for perfect accuracy, irregular spikes in a KPI history can significantly affect resulting thresholds. For that cause, a CMS should recognize both explainable and unexplainable statistical outliers with appreciably deviating measurements in the considered reference period and omit them for KPI evaluations. Notably, public holidays can entail such erratic events.

In addition, it turned out to be good practice for a CMS to retain an alert until a KPI consecutively exceeds a threshold too often.

2.4 Automation

The projected CMS must automatically compute at least one threshold per KPI gauging, which takes place at regular intervals, and compare the calculated threshold with the KPI in order to detect the condition of the KPI. Irrespective of such an automatism, the CMS shall incorporate manually configured, absolute or proportional offset values in threshold computations.

Doubtlessly, the fewer offsets need to be defined, the better because of less necessary human labor for their maintenance.

2.5 Topicality

The heed of the recent trend of a KPI makes younger data more relevant than older ones.

That is why thresholds made out of recent historic data values predict topical KPIs rather than entire histograms.

2.6 Efficiency

A single CMS is ordinarily obliged minute-by-minute to update the thresholds for hundreds or even thousands of KPIs including the subsequent comparisons. On this account, all computationally intensive proposals, which will be introduced in Sect. 3, do not come into consideration for the intended CMS.

It rather has to rely on *lightweight* plug-ins that comprise efficient program code for threshold determination and contrasting juxtaposition with the respective KPI.

Prior to the presentation of such an efficacious algorithm, the following section comes along with the chronological findings about automatical threshold derivation for CMSs.

3 Related Work

In 2004, Brooks, Thorpe, and Wilson published a simple visual method for alarm rationalization that quickly delivers large sets of consistent alarm limits for a CDU (Crude Distillation Unit) [3]. They only depicted the operating envelope for a couple of variables by measuring them for a specific period rather than explicitly evaluated any thresholds.

Jabłoński et al. stated that the discipline and endeavor for proper, manually adjusted thresholds descends in the course of time [9]. On that account, they offered a model that selects the best match out of five various probability distributions for automatic threshold setting in CMSs. A CMS for services with time-varying load characteristic relying on this model would have to elaborate a threshold adapted from the most fitting probability distribution before each comparison with a newly measured KPI. Assuming that a CMS for services with time-varying load characteristic draws a new value for each KPI every minute, the number of tests for the most suitable probability distribution per minute equals the KPI quantity. Hundreds or even thousands of such probability distribution tests would contradict the demand for efficiency in Subsect. 2.6.

Marhadi and Skrimpas abstained from such probability distribution checks and exclusively advocated the Johnson distribution family for automatical threshold adjustment in wind turbine CMSs [13,14]. In their opinion, the choice of a distribution function out of a distribution family appears more practical than testing various distribution functions. They justified this decision with the certainty of not choosing incorrect probability distributions with suboptimal thresholds. Their approach lets a CMS (re)ascertain new Johnson distribution parameters and thresholds just in case of false alarms to minimize computational efforts. Such a saving strategy may work for wind turbines and other machineries, but the distribution of the used data history for the threshold ascertaining in the case of a service with time-varying load characteristic might change with every fresh reading of a KPI. For that reason, a separately ascertained threshold for every sensed KPI value rather does the trick.

Kocare, Juričić, and Boškoski called attention to tedious commissioning phases in which skilled persons need to heuristically tune a plurality of threshold values [12]. For this reason, they suggested threshold selection algorithms for CBM (Condition Based Maintenance). CBM denotes a predictive maintenance strategy that relies on CM. Their algorithms use the PFA (Probability of False Alarm) as a design parameter and then calculate the thresholds associated with the relative changes in condition indicators instead of separately tuning many thresholds by hand. If the variance of a condition indicator rises, the condition of the concerned system normally deteriorates. However, such an assumption cannot be made for services with time-varying load characteristic.

The same authoring group propounded to contrast the empirical distribution of reference data with the distribution of current data [6,10]. Particularly, those services with time-varying load characteristic do not profit from this approach whose KPIs deteriorate slightly, because a CMS would trigger an alert too late or, even worse, not at all.

Just as well, Chen et al. did not address convenient CM for services with time-varying load characteristic [5]. They rather concentrated on CM of rolling mill drivetrains by signal processing of non-stationary vibration data with customized maximal-overlap multiwavelet denoising.

Agarwal, Kishor, and Raghuvanshi took a promising direction by monitoring wave heights and sea-surface temperatures in the vicinity of offshore wind farms with adaptive thresholds for different times of the day [1]. They admitted that their article just assumed four quantization levels for fault detection and demanded improvement by defining adequate ranges for better fault prediction. This recommendation deserves adoption for CMSs that monitor services with time-varying load characteristic.

Straczkiewicz, Barszcz, and Jabłoński presented a methodology to make an overwhelming flood of alerts manageable by ranking them according to their importance and, thereby, prioritizing their handling [16]. For this cause, they described the calculation of a so-called VPC (Violation Priority Coefficient) by incorporating the violation attributes *severity*, *time of occurrence*, *type of failure*, and *type of element*. In particular, the common differentiation of anomalies into the severities *warning* and *alarm* sounds reasonable to be taken over by CMSs for services with time-varying load characteristic.

Yan, Zhou, and Pang emphasized the significance of warning areas that allow to take preventive maintenance action just in time before a breakdown would emerge and, concomitantly, optimize inspection costs [18].

4 Condition Monitoring System Architecture

Predominately, the call for high availability in Subsect. 2.1 necessitates the explication of a whole CMS architecture rather than a mere plug-in design. Figure 1 illustrates a CMS for services with time-varying load characteristic and the linked proximity of interest. Subsection 4.1 commences with a description of the illustrated components. Focused on the CMS itself, Subsect. 4.2 represents the centerpiece of this document with the details of the elaborated monitoring plug-in.

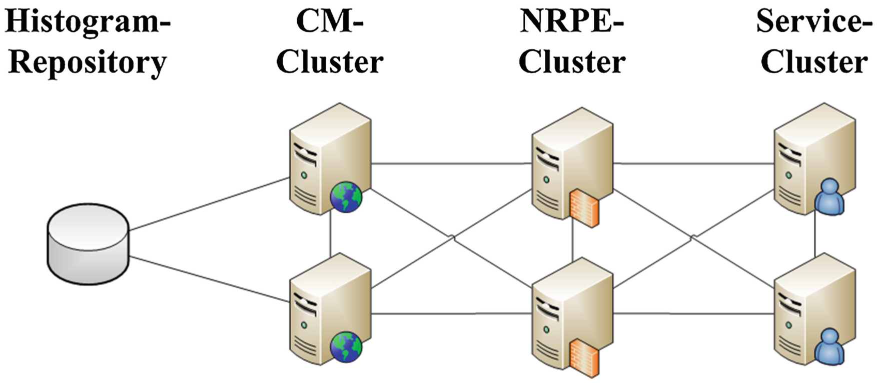


Fig. 1. Condition Monitoring System architecture

4.1 Components

Each of the three explicated clusters in this subsection stands for a buildup of at least two homogeneous computers to sustain high availability. The nodes of the NRPE (Nagios Remote Plugin Executor) and of the service cluster(s) do not necessarily have to be controlled by cluster software. Alternatively, loadbalancers or their clients can act as arbiters for stand-alone cluster nodes.

Histogram-Repository. The histogram-repository operates as the sole memory for historical KPI data. It only appears as singular data container in Fig. 1 to plainly visualize its unconditional responsibility for data consistency. Nonetheless, it must ensure the same level of high availability as the other CMS units do.

It can be realized in diverse ways. On the one hand, the CM-cluster in the next subsubsection can mount an external file system (such as an NFS (Network File System) share or another apt HA (High Availability) storage resource) that contains the histogram files. Also any kind of database can be connected as histogram-repository, provided that the CM-cluster supports such an interface. On the other hand, an exterior data store can be entirely omitted if the CM-cluster also enfolds an internal DFS (Distributed File System). While the latter goes without external data media, it requires protection against outages of the inter-cluster-communication and, based on this, against cluster-splits that induce data inconsistencies.

Condition Monitoring Cluster. The CM-cluster obligatorily depends on control by cluster software that enables the coordinated execution of plug-ins. Commonly, solely one cluster node is active and performs all condition checks. All remaining hosts serve as operational reserve by replicating the active master node. An outage of the master causes the residual hosts to immediately elect a new one. This operating mode with merely one active host undoubtedly wastes computing power for the sake of high availability. At the same time, it protects from overload, which might arise if (an) already active computer(s) must compensate the computing capacity of (a) broken-down peer(s).

Nagios Remote Plugin Executor Cluster. An NRPE-server acts as proxy for the above-mentioned CM-cluster, i.e. it executes health checks on behalf of the CM-cluster. The employment of an optional NRPE-cluster can be envisaged for two major matters. Firstly, in case the (master of the) CM-cluster becomes overstrained, distributed monitoring with the involvement of an NRPE-cluster mitigates the situation. Secondly, if any intentional firewall- or routing-rules deter (plug-ins of the) CM-cluster from accessing the below-mentioned service-cluster, then an NRPE-cluster with the correspondent access rules offers an acceptable loophole.

Service Cluster(s). The last part of the architectural chain consists of one or multiple service-clusters. A service-cluster allegorizes a highly available entity that either provides at least one telecommunication service with time-varying load characteristic and the corresponding KPIs, or only the latter. It can be stipulated by design that the prementioned CM-cluster needs to collect and aggregate the KPIs from several service-cluster elements.

4.2 Plug-in

This subsection proposes the usage of a plug-in that adheres to the novel Algorithm 1. The initiation of a condition test with a plug-in lies in the responsibility of (the active node of) the CM-cluster. This means that (the active node of) the CM-cluster either runs the plug-in itself or it assigns the execution order to an NRPE-cluster.

Algorithm 1 suits for CM of all services that provide every cumulative KPI with an OID (Object Identifier), for example, via SNMP (Simple Network Management Protocol) [4]. On these grounds, Algorithm 1 has been denominated as *check_ snmp*. A cumulative KPI quantifies all events since a definite moment (such as an application restart), i.e. it merely grows rather than shows a metric per time unit. Prior to a closer observation of the pseudo code in Algorithm 1, a reader is warmly recommended to catch a glimpse of Table 2 for a better understanding of the occurring notations.

The first five lines of Algorithm 1 revolve around a while-loop that tries to successively poll the current cumulative KPI value $csum$ of a service up to ret times. The reduced danger of missing previous KPI values vindicate the computational expenditure of such a retry mechanism. Nevertheless, if all ret retrievals fail, the branch between line 6 and 8 ends the plug-in with the unknown condition by returning exit code 3.

On the contrary, a successful polling of $csum$ instantaneously exits the loop and leads to the query of the most recent cumulative KPI value $psum$ in line 10.

Line 11 covers the if-condition with the usual circumstance that $csum$ exceeds the most recent $psum$. On those grounds, line 12 calculates the current real KPI value $cval$ as the (positive) difference between $csum$ as the minuend and the most recent $psum$ as the subtrahend.

The opposite in line 13 with a $csum$ less than or equal to the most recent $psum$ results in the zeroing of $cval$ in line 14. This else-branch customarily happens when an application restart has zeroed $csum$.

Line 16 initializes an empty ordered set $PVAL$ as a data structure intended for n previous real KPI values. The for-loop from line 17 to line 20 fills $PVAL$ with those n recent real KPI values that emerged a multiple of 10,080 min in the past, i.e. that one exactly a week ago, that one exactly two weeks ago, . . . , and that one exactly n weeks ago.

If a zeroed ign as the if-condition in line 21 comes true, then line 22 computes the critical threshold $crit$ by subtracting the mul-fold standard deviation of $PVAL$ from the arithmetic mean of $PVAL$. Line 23 formally indicates the disclaimer of a dedicated warning threshold $warn$ by only copying $warn$ from $crit$.

The branch for the preventive ignorance of ign outliers begins with the else-statement in line 24 and ranges to line 28. In detail, the statement in line 25 numerically sorts $PVAL$, followed by the creation of its subset $PVAL'$ in line 26, which lacks in the $\lfloor\frac{ign}{2}\rfloor$ highest and in the $\lfloor\frac{ign}{2}\rfloor$ lowest elements of $PVAL$. Line 27 evaluates the critical threshold $crit$ by deducting the mul-fold standard deviation of $PVAL$ from the arithmetic mean of $PVAL'$. Similarly, line 28

Table 2. Notations and their explanations

Notation	Explanation
$\boldsymbol{\Sigma}$	Alphabet
$\boldsymbol{\sigma}(\Gamma) : \underbrace{\mathbb{R} \times \cdots \times \mathbb{R}}_{\lvert\Gamma\rvert \text{ times}} \to \mathbb{R}, \Gamma \mapsto \sigma(\Gamma) =$ $\lvert\sqrt{\frac{\sum_{\gamma\in\Gamma}(\gamma - E(\Gamma))^2}{\lvert\Gamma\rvert}}\rvert$	Unary function that outputs the standard deviation of Γ
$\boldsymbol{E}(\Gamma) : \underbrace{\mathbb{R} \times \cdots \times \mathbb{R}}_{\lvert\Gamma\rvert \text{ times}} \to \mathbb{R}, \Gamma \mapsto E(\Gamma) =$ $\frac{\sum_{\gamma\in\Gamma}\gamma}{\lvert\Gamma\rvert}$	Unary function that outputs the arithmetic mean of Γ
$\boldsymbol{PVAL} \subsetneq \mathbb{R}^{\geq 0}$	Chronologically ordered set with n previous real KPI values, each pair of neighbored elements with an interval of one week between them
$\boldsymbol{PVAL'} \subset PVAL$	Chronologically ordered set with $n - ign$ previous real KPI values, each pair of neighbored elements with an interval of one week between them
$\boldsymbol{REPO} \subsetneq \mathbb{R}^{\geq 0} \times \mathbb{R}^{\geq 0}$	Histogram-repository with real and cumulative KPI values
$\boldsymbol{cond} \in \{0, 1, 2, 3\}$	Service condition: 0 = okay, 1 = warning, 2 = critical, 3 = unknown
$\boldsymbol{crit} \in \mathbb{R}^{\geq 0}$	Critical threshold
$\boldsymbol{csum} \in \Sigma^*$	Current cumulative KPI value
$\boldsymbol{cval} \in \mathbb{R}^{\geq 0}$	Current real KPI value
$\boldsymbol{hist}(\alpha, \beta)$: $\underbrace{\mathbb{R}^{\geq 0} \times \mathbb{R}^{\geq 0}, \cdots, \mathbb{R}^{\geq 0} \times \mathbb{R}^{\geq 0}}_{\lvert\alpha\rvert \text{ times}} \times \mathbb{N} \to \mathbb{R} \times \mathbb{R}$	Bivariate function that takes the Cartesian product of a histogram-repository α and a positive integer β as input and outputs the Cartesian product of *pval* and *psum*
$\boldsymbol{host} \in \Sigma^*$	Host name or IP (Internet Protocol) address of Service Cluster
$\boldsymbol{i} \in \mathbb{N}$	Loop-variable
$\boldsymbol{ign} \in \mathbb{N} \mid ign < n$	Number of ignored previous real KPI values: the highest $\lfloor \frac{ign}{2} \rfloor$ and the lowest $\lfloor \frac{ign}{2} \rfloor$ historical KPI values become ignored
$\boldsymbol{mul} \in \mathbb{N}$	Multiple of standard deviation between mean and threshold
$\boldsymbol{n} \in \mathbb{N}$	Sample size
$\boldsymbol{oid} \in \Sigma^*$	SNMP (Simple Network Management Protocol) OID (Object Identifier)
$\boldsymbol{psum} \in \mathbb{R}^{\geq 0}$	Previous cumulative KPI value
$\boldsymbol{pval} \in \mathbb{R}^{\geq 0}$	Previous real KPI value
$\boldsymbol{ret} \in \mathbb{N}$	Maximum number of retrieval attempts for current cumulative KPI value *csum*
$\boldsymbol{sort}(\Gamma) : \underbrace{\mathbb{R} \times \cdots \times \mathbb{R}}_{\lvert\Gamma\rvert \text{ times}} \to \underbrace{\mathbb{R} \times \cdots \times \mathbb{R}}_{\lvert\Gamma\rvert \text{ times}}, \Gamma \mapsto$ $\pi(\Gamma) =$ $\{\gamma_1, \cdots, \gamma_{\lvert\Gamma\rvert}\} \mid (\, forall \alpha \in \ \mathbb{N}^{<\lvert\Gamma\rvert})\gamma_\alpha \ \leq \gamma_{\alpha+1}$	Numerical sorting of the elements in set Γ
$\boldsymbol{warn} \in \mathbb{R}^{\geq 0}$	Warning threshold

Algorithm 1. check_snmp

```
Require: $REPO, host, ign, mul, n, oid, ret$
Ensure: $cond, cval, warn, crit, csum$
$i = 1$ {Initialization of $i$}
while $csum \notin \mathbb{R}^{\geq 0} \wedge i \leq ret$ do {$csum$ is invalid and $i$ is not larger than $ret$}
    $csum \leftarrow$ current value for $oid$ from $host$ {Retrieval of $csum$}
    $i = i + 1$ {Incrementation of $i$}
end while
if $csum \notin \mathbb{R}^{\geq 0}$ then {$csum$ is still not valid}
    $cond = 3$ {Condition is unknown}
    return $cond$ {Mere output of $cond$, i.e. omission of $cval, warn, crit, csum$}
else {$csum$ is valid}
    $(pval, psum) \leftarrow hist(REPO, 1)$ {Retrieval of most recent $psum$}
    if $csum > psum$ then {$csum$ is larger than $psum$}
        $cval = csum - psum$ {Calculation of $cval$}
    else {$csum$ is not larger than $psum$}
        $cval = 0$ {Zeroing of $cval$}
    end if
    $PVAL = \{\}$ {Initialization of data structure for $n$ previous real KPI values}
    for $i = 1$ to $n$ do {Iteration loop}
        $(pval, psum) \leftarrow hist(REPO, i * 10080)$ {Retrieval of $n$ previous real KPI
        values, each pair of neighbored elements with an interval of one week between
        them}
        $PVAL = PVAL \cup \{pval\}$ {Attachment of a previous real KPI value to data
        structure}
    end for
    if $ign = 0$ then {No previous real KPI values are ignored}
        $crit = E(PVAL) - mul * \sigma(PVAL)$ {Calculation of $crit$}
        $warn = crit$ {Calculation of $warn$}
    else {$ign$ previous real KPI values are ignored}
        $PVAL = sort(PVAL)$ {Numerical sorting of $n$ previous real KPI values}
        $PVAL' = \{pval_\alpha \in PVAL | 1 + \lfloor \frac{ign}{2} \rfloor \leq \alpha \leq |PVAL| - \lfloor \frac{ign}{2} \rfloor\}$ {Removal of
        $ign$ previous real KPI values}
        $crit = E(PVAL') - mul * \sigma(PVAL)$ {Calculation of $crit$}
        $warn = E(PVAL') - mul * \sigma(PVAL')$ {Calculation of $warn$}
    end if
    if $cval \leq crit$ then {$cval$ is not larger than $crit$}
        $cond = 2$ {Condition is critical}
    else if $cval \leq warn$ then {$cval$ is not larger than $warn$}
        $cond = 1$ {Condition is warning}
    else {$cval$ is larger than $warn$}
        $cond = 0$ {Condition is okay}
    end if
    return $cond, cval, warn, crit, csum$ {Output of $cond, cval, warn, crit, csum$}
end if
```

reckons the warning threshold *warn* by subducting the *mul*-fold standard deviation of $PVAL'$ from the arithmetic mean of $PVAL'$.

The lines numbered from 30 to 36 dedicate themselves to the assessment of the KPI-condition. Algorithm 1 assumes that the fall of *cval* below a threshold aggravates its condition. The worst case occurs if the if-condition in line 30 becomes true and, consequently, line 31 declares the condition *cond* to be critical. Line 33 assesses the less severe warning condition for *cond* in case *cval* lies between *crit* and *warn* and, as a consequence of this, fulfills the else-if-condition in line 32. At best, neither the if-condition in line 30 nor the else-if-condition in line 32 applies, whereby the else-branch in line 34 takes effect that attests *cond* being healthy in line 35.

The plug-in process terminates in line 37 with the output of *cond*, *cval*, *warn*, *crit*, and *csum*.

Just now, after the exposure of Algorithm 1, the time is ripe to demonstrate its operational capability.

5 Reliability Analysis

A plug-in based on Algorithm 1 can be certified as being reliable for CM of telecommunication services with time-varying load characteristics if it conforms to all compiled requirements in Sect. 2. For this purpose, the empiric approach by establishing and watching a bunch of experimental KPIs with the aid of a suiting prototype seems to be the inevitable way of evidence.

For that purpose, the author of this scientific paper implemented all components of the CMS architecture in Fig. 1 as follows.

– **Condition Monitoring Cluster**
    - **Hardware:** 2 HP ProLiant DL360 G6, each with 32 2.4 GHz CPU-cores and 36 GB main memory
    - **Operating System:** Fedora Core Linux release 25 64-bit
    - **Cluster Engine:** Corosync with Pacemaker
    - **Condition Monitoring System:** Nagios 4.0.8
    - **Plug-in Programming Language:** PHP 7.0.16

– **Histogram-Repository**
    - **Storage:** RRD (Round Robin Database) files in local file system
    - **Replication:** Asynchronously with LSyncD (Live Syncing Daemon)

– **Nagios Remote Plugin Executor Cluster**
    - **Hardware:** 2 Dell PowerEdge R630, each with 48 2.2 GHz CPU-cores and 256 GB main memory
    - **Operating System:** CentOS Linux release 7.2 64-bit
    - **Cluster Engine:** none
    - **NRPE Version:** 2.15
    - **Plug-in Programming Language:** PHP 7.0.16

– **Service Cluster**
    - **Hardware:** 4 HP ProLiant DL120 G9, each with 8 2.4 GHz CPU-cores and 16 GB main memory

- **Operating System:** Ubuntu Linux release 16.04 64-bit
- **Cluster Engine:** none
- **KPI Generator:** BIND (Berkeley Internet Name Domain) Server [17] 9.9.5
- **OID Generator:** SNMPD (Simple Network Management Protocol Daemon) 5.7.3

The service cluster stems from an Austrian TSP, which has made the OID with the cumulative amount of sent responses of each of its four DNS-servers via SNMP available for this experiment. Aside from an individual KPI for each DNS-server obtained through the NRPE-cluster, an additional plug-in in the CM-cluster added them together to a sum KPI with the help of queries towards the four pertinent RRD-files. This extra plug-in also includes the program code between line 16 and 37 of Algorithm 1 in order to generate both thresholds *warn* and *crit* and, eventually, the condition *cond* for the current sum KPI value *cval*.

The CM-cluster had collected all sum KPI values of an at least four months long preparatory phase before the effective one-month experimental stage with enabled notifications began. Four short periods with serious volatility due to network incidents apparently happened within this month. Accordingly, notifications were solely wanted during these four phases, respectively their clearings afterward. Figure 2 exemplarily depicts one of these four occurred volatile time spans.

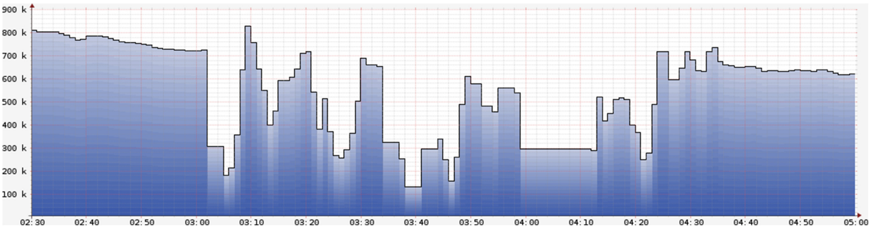


Fig. 2. KPI volatility

To gain no missing or false alerts, the assignment of useful values for the amount of preventively ignored outliers *ign*, for the standard deviation multiplier *mul*, and for the sample size *n* had to be solved. Due to the minimum retention time of four months, the experiment could and, indeed, combinatorially ran for $2 \leq n \leq 16$ and $0 \leq ign \leq n-2 | n \ mod \ 2 = 0$. The author of this treatise simply supposed Gaussian distribution of $PVAL'$ instead of a number-crunching probability distribution test on the tide of every plug-in execution. Furthermore, he doubled the aimed $\sum_{n=2}^{16} \lfloor \frac{n}{2} \rfloor = 64$ KPIs to 128 KPIs by tying each possible n-ign-combination with both standard deviation multipliers $mul = 2$ and $mul = 3$. While one-sided warning and critical thresholds with a distance of $2 * \sigma(PVAL')$

respectively $2 * \sigma(PVAL)$ to $E(PVAL)$ assume a theoretical notification likelihood of 2.275%, those with a margin of $3 * \sigma(PVAL')$ respectively $3 * \sigma(PVAL)$ to $E(PVAL)$ let expect a theoretic notification likeliness of 0.135% [15].

Each of the four surface diagrams in Fig. 3 displays the ground view of a spatial chart with the standard deviation multiplier $mul = 2$, the sample size n on the x-axis, the ignored outliers ign on the y-axis, and the passel of events on the z-axis. The top left graph adumbrates between 705 ($n = 16$, $ign = 2$) and 4,152 ($n = 8$, $ign = 6$) soft warnings events, i.e. undershootings of the 43,200 induced warning thresholds in the observed month (one threshold per minute). There did no warnings bechance for $ign = 0$ because of $warn = crit$, i.e. every KPI lower than $warn$ implicitly fell below $crit$ and, in lieu thereof, caused the graver critical counterpart. Each soft warning event with a duration longer than five minutes became a hard warning event. The top right illustration showcases between 130 ($n = 16$, $ign = 2$) and 1,078 ($n = 6$, $ign = 4$) hard warning notifications. The bottom left drawing portrays between 226 ($n = 16$, $ign = 14$) and 3,971 ($n = 2$, $ign = 0$) soft critical events, i.e. breaches of the 43,200 reckoned critical thresholds. Identically to warnings, five minutes lasting soft critical events mutate into hard ones. The bottom right depiction testifies between 61 ($n = 16$, $ign = 0$) and 668 ($n = 4$, $ign = 2$) hard critical notifications. 61 critical notifications in one month signifies two per day on average. If TSPs might be kept informed even about minor KPI volatilities, then they are well advised to set $mul = 2$.

Otherwise, the TSPs' preference could be a higher standard deviation multiplier of $mul = 3$ in expectation of a diminished event count (as shown in Fig. 4). The top left picture exhibits between 303 ($n = 16$, $ign = 2$) and 4,591 ($n = 6$, $ign = 4$) soft warning events respectively the top right one between 22 ($n = 16$, $ign = 2$) and 863 ($n = 6$, $ign = 4$) hard warning notifications. As expected, the extent of warnings as a result of $mul = 3$ overall abates compared to the raised ones by $mul = 2$. Likewise, the soft critical events in the bottom left image wane to the range between 76 ($n = 16$, $ign = 14$) and 3,449 ($n = 2$, $ign = 0$) respectively the hard critical notifications in the bottom right one to incidences between 9 ($n = 16$, $ign = 0$) and 399 ($n = 2$, $ign = 0$).

Interestingly, solely the setup with $ign = 0$, $mul = 3$, and $n = 16$ achieved that the CMS triggered its nine hard critical notifications merely within the aforesaid four volatile stages. This parameter composition palpably crystallizes as optimum pick to produce neither undesired nor absent alarms. Generally, the conducted trial obviously proved that the maximization of mul and n as well as the minimization of ign minimizes the expectable alert count. Howsoever, a very low ign (specifically 0) bears the latent risk of objectionable alerts if past long-lasting outliers (e.g. total outages) sorely influence successive thresholds. Moreover, the retention time of soft events (i.e. the maximal count of consecutive threshold exceedings before a soft condition turns into a hard one) connotes a subsidiary parametrization possibility whose prolongation decreases hard notifications.

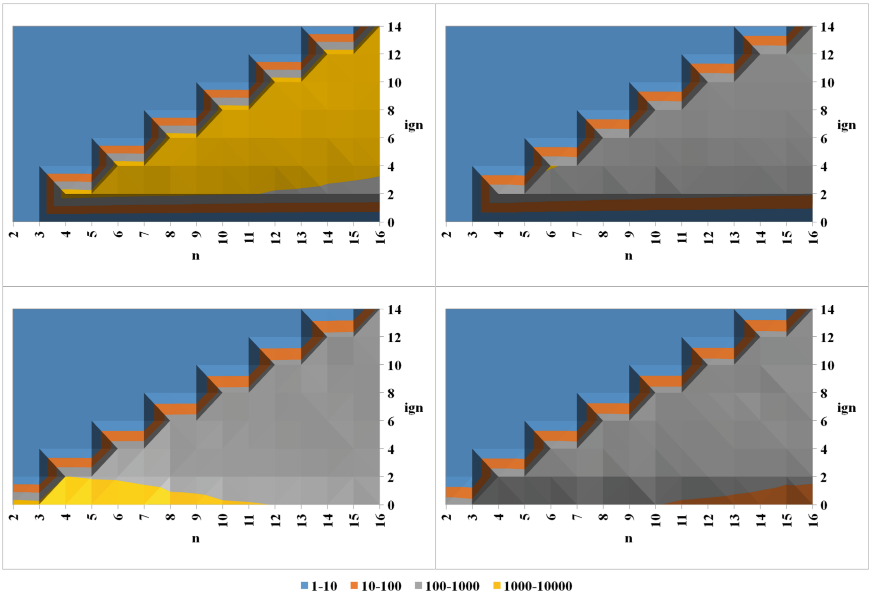


Fig. 3. Notification frequency with standard deviation multiplier $mul = 2$ (Top left: Soft warning notifications, Top right: Hard warning notifications, Bottom left: Soft critical notifications, Bottom right: Hard critical notifications)

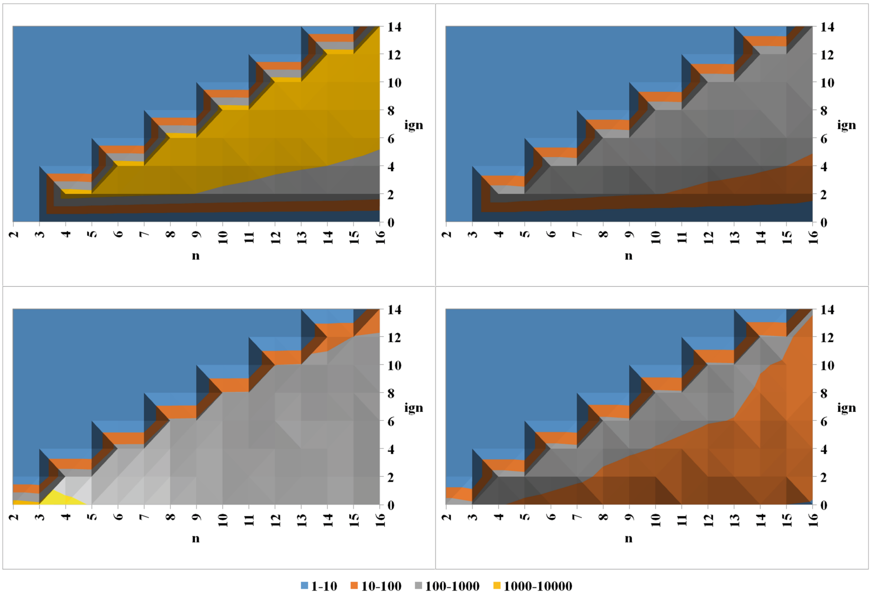


Fig. 4. Notification frequency with standard deviation multiplier $mul = 3$ (Top left: Soft warning notifications, Top right: Hard warning notifications, Bottom left: Soft critical notifications, Bottom right: Hard critical notifications)

6 Conclusion

The reliability analysis in Sect. 5 already evidences the feasibility of Algorithm 1 for reliable CM of telecommunication services with time-varying load characteristic. Anyway, the very last session here lends itself to a self-critical reflection whether the limned CM architecture in Sect. 4 satisfies all developed requirements of Sect. 2.

The plenary replication of all CMS parts achieves high *availability*. The integrated revaluation of a warning and a critical threshold for every fetched KPI value warrants a high degree of alert *accuracy*. The option of prophylactic outlier eradication with $ign > 0$ proffers *robustness* against unwanted alarms. Further, a configurable retention time also makes a CMS *robust* by limiting the conversions of soft into hard events. Supplementarily to KPI collection, the plug-in based on Algorithm 1 guarantees the implicit recalculation of warning and critical thresholds to accomplish the requested *automation*. These revaluations also offer *topicality* because they only process the last n KPI values. The plug-in guarantees *efficiency* with its *lightweight* implementation in PHP 7 with its JIT (Just In Time) compiler. Also, the supposition of Gaussian distribution instead of any periodical probability distribution tests redounds to improved *efficiency*.

Despite that, worthwhile future work should explore further expedient statistical distributions for lightweight threshold computation within the scope of reliable CM of telecommunication services with time-varying load characteristic. In the same context, the development of a lightweight algorithmic panacea, which inherently chooses the optimal probability distribution and its parameters, turns out to be another interesting challenge.

Acknowledgments. Many thanks to Bettina Baumgartner from the University of Vienna for proofreading this paper!

References

1. Agarwal, D., Kishor, N., Raghuvanshi, A.S.: Flexible threshold selection and fault prediction method for health monitoring of offshore wind farm. IET Wirel. Sens. Syst. **5**, 183–192 (2015). https://doi.org/10.1049/iet-wss.2014.0008
2. Beyaz, S.: Konzeption, Einführung und Integration eines Monitoringsystems in bestehende Netzwerkdienste in einer Krankenhausumgebung. Ph.D. thesis, University of Erlangen-Nuremberg, Erlangen, Bavaria, Germany, June 2010. https://opus4.kobv.de/opus4-fau/files/1260/Dissertation_Beyaz.pdf
3. Brooks, R., Thorpe, R., Wilson, J.: A new method for defining and managing process alarms and for correcting process operation when an alarm occurs. J. Hazard. Mater. **115**(1–3), 169–174 (2004). https://doi.org/10.1016/j.jhazmat.2004.05.040
4. Case, J.D., Fedor, M., Schoffstall, M.L., Davin, J.R.: A simple network management protocol (SNMP). RFC 1157 (Historic), May 1990. https://www.ietf.org/rfc/rfc1157.txt
5. Chen, J., Wan, Z., Pan, J., Zi, Y., Wang, Y., Chen, B., Sun, H., Yuan, J., He, Z.: Customized maximal-overlap multiwavelet denoising with data-driven group threshold for condition monitoring of rolling mill drivetrain. Mech. Syst. Sign. Process. **6869**, 44–67 (2016). https://doi.org/10.1016/j.ymssp.2015.07.022

6. Dolenc, B., Boškoski, P., Juričić, Đ.: Robust information indices for diagnosing mechanical drives under non-stationary operating conditions. In: Chaari, F., Zimroz, R., Bartelmus, W., Haddar, M. (eds.) Advances in Condition Monitoring of Machinery in Non-stationary Operations. ACM, vol. 4, pp. 139–149. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-20463-5_11
7. Gray, J., Siewiorek, D.P.: High-availability computer systems. Computer **24**(9), 39–48 (1991). https://doi.org/10.1109/2.84898
8. International Telecommunication Union: X.700: Management framework for open systems interconnection (OSI) for CCITT applications, September 1992. https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.700-199209-I!!PDF-E
9. Jabłoński, A., Barszcz, T., Bielecka, M., Breuhaus, P.: Modeling of probability distribution functions for automatic threshold calculation in condition monitoring systems. Measurement **46**(1), 727–738 (2013). https://doi.org/10.1016/j.measurement.2012.09.011
10. Juričić, Đ., Kocare, N., Boškoski, P.: On optimal threshold selection for condition monitoring. In: Chaari, F., Zimroz, R., Bartelmus, W., Haddar, M. (eds.) Advances in Condition Monitoring of Machinery in Non-stationary Operations. ACM, vol. 4, pp. 237–249. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-20463-5_18
11. Kearney, K.T., Torelli, F.: The SLA model. In: Wieder, P., Butler, J., Theilmann, W., Yahyapour, R. (eds.) Service Level Agreements for Cloud Computing, pp. 43–67. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-1614-2_4
12. Kocare, N., Juričić, D., Boškoski, P.: Optimal threshold selection in condition monitoring based on probability of false alarm. In: 23rd International Electrotechnical and Computer Science Conference ERK 2014, pp. 175–178, September 2014. http://erk.fe.uni-lj.si/2014/kocare(optimal)p.pdf
13. Marhadi, K.S., Skrimpas, G.A.: Using Johnson distribution for automatic threshold setting in wind turbine condition monitoring system. In: Annual Conference of the PHM Society (PHM), vol. 5, pp. 1–13, October 2014. https://www.phmsociety.org/node/1395
14. Marhadi, K.S., Skrimpas, G.A.: Automatic threshold setting and its uncertainty quantification in wind turbine condition monitoring system. Int. J. Prognostics Health Manage. **6**(Special Issue Uncertainty in PHM), 1–15 (2015). https://www.phmsociety.org/node/1571
15. Pukelsheim, F.: The three sigma rule. Am. Stat. **48**(2), 88–91 (1994). https://doi.org/10.2307/2684253
16. Straczkiewicz, M., Barszcz, T., Jabłoński, A.: Detection and classification of alarm threshold violations in condition monitoring systems working in highly varying operational conditions. J. Phys: Conf. Ser. **628**(1), 1–8 (2015). https://doi.org/10.1088/1742-6596/628/1/012087
17. Terry, D.B., Painter, M., Riggle, D.W., Zhou, S.: The Berkeley internet name domain server. Technical report, EECS Department, University of California, Berkeley, CA, USA, May 1984. https://www2.eecs.berkeley.edu/Pubs/TechRpts/1984/CSD-84-182.pdf
18. Yan, H.C., Zhou, J.H., Pang, C.K.: Cost optimization on warning threshold and non-fixed periodic inspection intervals for machine degradation monitoring. In: IECON 2015–41st Annual Conference of the IEEE Industrial Electronics Society, pp. 1079–1084, November 2015. https://doi.org/10.1109/IECON.2015.7392243

Fast and Secure Handoffs for V2I Communication in Smart City Wi-Fi Deployment

Pranav Kumar Singh(✉), Subhredu Chattopadhyay, Pradeepkumar Bhale, and Sukumar Nandi

Department of Computer Science and Engineering, Indian Institute of Technology,
Guwahati 781039, India
{pranav.singh,subhrendu,pradeepkumar,sukumar}@iitg.ernet.in

Abstract. The Intelligent Transport System (ITS) is a vital part of smart city developments. Due to densely deployed access points and vehicular mobility in a smart city, the number of handovers also increases proportionately. Minimization of the handoff latency is crucial to provide a better quality of service for vehicles to have access different ITS services and applications. Increased handover latency can cause an interruption in vehicle-to-infrastructure (V2I) communication. In this paper, we propose a fast and secure handoff mechanism for smart cities that have acceptable handoff latency for delay-sensitive ITS applications and services. Our proposal considers mobility and communication overhead to provide lower handoff latency. We compare our proposed mobility aware background scanning mechanism (AdBack) with standard Active Scanning mechanism in an emulated test bed. Our test results reveal that the proposed AdBack mechanism significantly outperforms the existing mechanisms in terms of handover latency, packet drop rates, and throughput. Experimental results show that amalgamation of AdBack and existing fast re-authentication (IEEE 802.11r) can improve connectivity for V2I communication in a smart city. We provide rigorous emulation results to justify the performance of our proposed scheme.

1 Introduction

Vehicular communication that emerged from wireless communication has gained much interest from academia, industries, and governments to improve road safety, fuel efficiency, and convenience of travel. Vehicular communication is one of the leading research areas because of its applications and its specific characteristics. In the wake of the Information and communication technology (ICT) revolutions, road transportation has entered into a new era, and today we have technologies and services such as connected vehicles, driverless cars, smart cars, VANET, Internet of vehicles, vehicle telematics, and intelligent transportation systems (ITS). Based on the data collected from various studies, surveys, polls and driver's experiences hundreds of applications can be suggested. Most of

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 189–204, 2018.
https://doi.org/10.1007/978-3-319-72344-0_15

these vehicular applications fall into three main categories: safety, non-safety, and infotainment.

- **Safety Applications:** The road safety can be provided by vehicle-to-vehicle (V2V), vehicle-to-pedestrian (V2P) and vehicle-to-infrastructure (V2I) communication. However, safety-critical applications such as post-crash warning, pedestrian crossing, lane change warning, emergency brake, and do-not-pass warning are provided by V2V and V2P mode.
- **Non-Safety Applications:** Vehicles with a set of sensors can provide a wide variety of sensed information related to the vehicle, traffic, environment, parking and driving conditions. Collected sensor information can be uploaded from vehicles to a smart city road administration database using the roadside infrastructures (access points) in V2I mode. Vehicles traveling on the same route can obtain information for their applications.
- **Infotainment Applications:** Vehicle-to-Network (V2N) communications enable infotainment applications such as Voice over Internet Protocol (VoIP), video download, streaming, live TV, software update of the in-vehicle unit, map update, instant messaging, cloud-based services and Internet access. We consider V2I for this category as well. These applications can improve driving comfort and keep vehicle occupants informed. The maximum allowable latency for the VoIP application can be up to 50 ms. The allowed latency and the range of communication required for these application categories can be found in [1].

With the increasing number of vehicles and development of the smart cities, new issues related to traffic jam, road accidents, and carbon emission have emerged. Vehicular communication can help solve these problems and bring innovations in these contexts. Many countries worldwide, have already deployed early stages of vehicular networks to make transport safer and comfortable.

With urbanization and the development of smart cities, there is rapid growth in Wi-Fi deployments that make Wi-Fi a complementary, low-cost solution for V2I/I2V connectivity. Providing Quality of Service (QoS), seamless connectivity and security in densely deployed scenarios of a smart city Wi-Fi are some of the biggest challenges. The communications range of the road-side access points is a maximum of 300–400 m. The handover delay at Layer 2 due to the scanning and reauthentication delay can disrupt ongoing communication (such as VoIP) of vehicle occupants in a smart city. The Wi-Fi protocol stack (IEEE 802.11a/b/g/n/ac) is not designed for the vehicular mobility context. However, various amendments to the IEEE 802.11 standard is bringing advancements in Wi-Fi. Such as, IEEE 802.11r provides fast re-authentication [2], IEEE 802.11i [3] for enhanced security, IEEE 802.11w provides protection to management frames, IEEE 802.11e for QoS, etc. These amendments does not solve the vehicular mobility problem. The wireless network architecture must provide fast, secure, seamless, and highly available connectivity to its users, regardless of whether they are static or moving. In this paper, we demonstrate that our proposed mobility aware scanning with IEEE 802.11r (rather than full IEEE 802.11i scanning) can provide seamless connectivity to the V2I services.

The rest of the paper is organized as follows: Sect. 2 describes the background to our topic. We analyze the work related to Level 2 handover latency minimization in Sect. 3. We present our proposal for scanning mechanism in Sect. 4. Section 5 covers the details of the emulation setup used to demonstrate our proposed mechanism. In Sect. 6, we analyze the performance in a Wi-Fi deployment of a smart city in terms of handover latency, packet loss, and average throughput. Finally, we conclude our work in Sect. 7.

2 Background

This section presents a summary based on comprehensive analysis of IEEE 802.11 management frames that are responsible for maintaining communication between access points (APs) and wireless clients during the static as well as handover procedure. We present the detailed background of handover management which includes discovery and reauthentication in wireless network protected with Wi-Fi Protected Access II (WPA2: 802.1X/Extensible Authentication Protocol (EAP)) mechanism.

2.1 Discovery Mechanism

Handoff due to vehicular mobility is the process of reassociation to the new AP when a vehicle moves away from the currently associated AP, and the received signal strength decreases to a certain threshold. The discovery process of the new AP involves initiation and scanning. As the mobile node moves away from the connected AP, the Received Signal Strength Indicator (RSSI) begin to drop and force the mobile node to discover new accessible APs. Of all scanned APs, the mobile node selects one for its association based on some specific criteria. There are two types of the mechanism that allows the mobile node to discover target AP: Passive Scanning and Active Scanning.

1. **Passive Scanning:** In passive scanning mode, the mobile nodes listen for the beacon management frames broadcasted by the APs. Beacon frames are transmitted periodically by the AP to announce its presence. The default broadcasting interval is usually configured as 100 ms and is known as the Beacon Interval. Thus, it may take 100 ms for a mobile node to hear a beacon frame. Passive scanning usually takes more time, since the mobile node has to wait long enough on a channel for a beacon frame. Because it is a time-consuming process to hear a beacon frame, most mobile nodes prefer an active scan.
2. **Active Scanning:** In active scanning mode, the mobile node switches to a new channel and broadcasts Probe Request frames on it and waits for the Probe Response frames from APs operating on that channel. If no response received on that channel, it is assumed empty, and the mobile node switches to a new channel. This process repeats for all operating channels. The set of the channel depends on mode and country. Finally, received Probe Response frames are processed by the mobile node to obtain information about candidate access point.

The research studies [4,5] have already measured the scanning delay, which suggests that it varies between 600 ms to 700 ms. They observed that discovery delay is the dominating component of the handoff delay. It accounts more than 90% of the overall handoff delay. Thus probing is the bottleneck for fast handoff and should be reduced to provide seamless connectivity.

In case of an wireless infrastructure based WPA2 Enterprise network, every handover mechanism must be followed by a reauthentication procedure after the scanning. The reauthentication phase that includes key management is equally time-consuming. It varies between few milliseconds to second [4], depends on which authentication mode (Pre Shared Key (PSK) or 802.1X/EAP) and protocol used. Authors [6] in their performance study have shown that if 802.1X/EAP authentication (baseline 802.11i authentication) is used then the average roaming time is 525 ms and maximum consecutive lost datagrams (Average) is 53.

The handover due to mobility can severely affect QoS and QoE for real time applications and ITS services in the 802.11i Enterprise based security framework. Thus, to minimize latency during reauthentication and key management the IEEE Task Group r (TGr) was formed.

2.2 IEEE 802.11r Fast BSS Transition

The 802.11r standard amendment specifies a Fast Basic Service Set Transition (FT-BSS) mechanism ratified in 2008.

This section describes the IEEE 802.11r security framework and FT-BSS transition process.

Fast BSS Transition Security Framework. The handover process based on the 802.1X/EAP security framework consists of 6 phases: initiation, discovery, 802.11 open authentications, reassociation, reauthentication, and the key-handshake. The (re)authentication phase of WPA2 Enterprise based on 802.1X/EAP uses an external server (e.g., Remote Authentication Dial-In User Service (RADIUS)) to provide Authentication, Authorization, and Accounting (AAA). Without FT enabled, the mobile node needs to go through a complete reauthentication (including key management) after reassociation in each handover. In the FT framework of IEEE 802.11r, reauthentication is performed efficiently before reassociation.

As per the specification of the current draft of IEEE 802.11r, 802.1X/EAP based authentication is done once when the mobile node initially joins the network and generates the Pairwise Master Key (PMK). The generated PMK is distributed to all APs belonging to the same mobility domain. Thus, this presence of PMK at all APs helps to reduce reauthentication delay that incurs in communication to an external server (RADIUS) for authentication.

For a mobile node that is 802.11r compatible, the 4-way handshake followed by the QoS request over WLAN using IEEE 802.11e is completed during the reassociation phase, further reducing the overall handover latency. In contrast, an 802.11i-based mobile node needs to repeat full 802.1X authentication and

4-way handshake during every handover. If QoS is enabled, then there will be more frame exchange in IEEE 802.11i, which will contribute to an additional delay to the overall latency.

FT BSS Transition. FT BSS transition is the process of disassociating from one and re-associating to new AP, and all the APs belong to the same mobility domain (same Extended Service Set (ESS)). The set of frame exchanges in reauthentication and key-handshake takes a considerable amount of time in a secure WLAN based on 802.1X/EAP. Thus, the number of the frame exchange between a mobile node and an AP must be reduced during the transition. It will help minimizing interruption to delay-sensitive services such as voice and video during the handover from one AP to another. There are two underlying FT protocols used for subsequent re-associations to APs within the same mobility domain. These two are described as follows:

FT Protocol: FT protocol is for a simple transition of the mobile node that does not require resource request before its transition.

FT Resource Request Protocol: In this protocol, the mobile node requires a resource request before its transition. In this paper, we consider FT Protocol only (without resource request) in this work. There are two methods of Fast BSS transition: Over-the-Air and Over-the-Distribution System (DS) Fast BSS Transition. A mobile node can opt one of these for it's handover to a target AP (selected after scanning) from the currently associated AP.

Over-the-Air Fast BSS Transition: In this Fast BSS Transition, mobile node directly communicates with the target AP over the air. Only four frames are exchanged between the mobile node and the target AP during reauthentication. They contain appropriate information for PTK generation at the both the end. Now, time-critical phases 802.1X/EAP including 4-way key-handshake are not required to unblock the uncontrolled port.

Over-the-DS Fast BSS Transition: In Fast BSS Transition, mobile node communicates with the target AP via the current AP. Communication between a mobile node and the target AP takes place using FT Action frames.

Over-the-Air vs Over-the-DS: In case of Over-the-Air (OTA), the mobile node needs to leave its active channel to negotiate on another channel during scanning. The mobile node sends a frame to its currently associated AP and tells it to go into sleep mode. When the negotiation completes, then it returns to the active channel to flush its and AP's buffer. The OTA can interrupt communication in a place where the mobile node is already at the edge of the AP range, suffering from poor performance.

In the Over-the-DS mode, the mobile node does not leave the channel. The mobile node stays on its current channel and asks the current AP to negotiate with the next AP. However, mobile node still needs to discover the target AP first by using some scanning mechanism. Over-the-DS mechanism improves performance in terms of lower BSS transition time than the OTA mechanism.

3 Related Work

The early research work, [5,7], tried to solve the problem related to the high probe delay observed in [4] using selective scanning, caching and neighbor graph. However, these mechanisms not tested in the context of vehicular mobility. The method proposed in [7] requires changes inside the presently deployed 802.11 APs. Authors of [8] have focused on the same problem using interleaved scanning in a random mobility. The public HotSpots region is selected covered by several APs with more than 20% of the overlapping area.

Studies in [9,10] proposed multiple wireless cards for AP and the mobile device, respectively. The mechanisms proposed in these studies are not practical to the same technology access and could be expensive as well.

The research works in [11–14] target to reduce Layer-2 handoff latency by adopting synchronization and pre-scan mechanism. However, these works used the passive approach, and the complexity of implementation is quite high.

The researchers of [15–17] have used a prediction of node mobility to improve performance and provide a better connectivity. The prediction mechanisms requires information such as position and movement direction, geolocation, and mobility history, respectively. Since determining correct position of the vehicle is not that easy, forecasting a better connectivity in a highly dynamic vehicular environment accurately is a difficult task. The navigation driven algorithms proposed in [18] may not be suitable for the vehicular context because vehicles have to move on the defined road topology and cannot change their route immediately depending on the handoff decision. The handoff strategy used in this work is a lazy type, where the handoff initiation occurs only when a mobile node disconnected with currently associated AP.

Finally, in [19–22] work is done related to the vehicular context. The researchers of [19] have used a directional antenna and beam steering techniques to collect information on a particular route, which in practice not feasible. The handover protocol proposed in [20] is complex in its implementation and [21] again used a prediction mechanism based on historical information.

In [22], the authors have analyzed the security properties and performance of IEEE 802.11i, IEEE 802.11r, HandOver KEY (HOKEY) and Control and Provision of Wireless Access Points (CAPWAP) for handoff in V2I communication. Studies in [6,23,24] tested and analyzed Layer-2 handover delay due to re-authentication only. Authors in [25] have compared the performance of IEEE 802.11r with Legacy IEEE 802.11 but the details of the discovery phase based on the location mechanism are not provided.

Most of the research work on Layer-2 handover scheme contributes only on minimization of the discovery delay (search or finding target AP) and does not consider the re-authentication delay part. In this paper, we are proposing a simple and fast scanning mechanism and test it with IEEE 802.11r-2008 as well.

4 Proposed Mechanism

From the discussion in the earlier section, primary contributing factors of handover latency are discovery delay and re-authenticating delay. In this paper, we provide a novel scheme for minimizing discovery delay of target roadside unit (RSU). Notations used in our proposed algorithm are listed in Table 1.

Table 1. Notations used in Algorithm 1

Symbol	Description
$RSSI_{RSU}$	RSSI of the RSU
$RSSI_{th}$	RSSI Threshold specified in bgscan modes
T_s	Short-interval for scanning
T_l	Long-interval for scanning
$Channels_{RSU}$	Database for scanned AP information in Learn Mode
STA_{speed}	Current speed of the Vehicle
$Speed_{th}$	Vehicle's maximum speed
BGScanLearn()	Learn Mode of the bgscan
BGScanSimple()	Simple Mode of the bgscan

4.1 Adaptive Background Scanning

As mentioned earlier (in Sect. 3) that the scanning phase is the bottleneck for fast handoff. Thus, ProbeDelay has to be reduced to provide seamless handover.

Handover strategies for vehicular communication need to be mobility aware. The reason to consider the mobility is that it severely affects performance in wireless networks. Therefore, we propose our Adaptive Background Scanning scheme (AdBack) to support fast and seamless roaming in densely deployed APs. The proposed AdBack scheme is mobility aware that relies on bgscan [26]. The On-board Unit (OBU) of vehicles usually equipped with a set of sensors including Gyro sensor, a processing unit, memory, and storage. Today, even our smartphones come with a set of sensors like Proximity, Gyro, light, accelerometer, digital compass, and magnetometer as well. Sensors can provide three crucial information: movement, direction, and speed. We are using speed information to improve connectivity and overall performance. Our mechanism is Adaptive because it adapts to different speeds which is detected by sensor. In Proposed AdBack algorithm, we are using movement information, which in a real scenario, can be provided by the accelerometer sensor. The handover decision can be improved by the use of accelerometer sensor data.

Periodic Background Scanning. In background scanning (bgscan), the mobile node scans channels to roam within an ESS (i.e., within a single network block). Other criteria of this mechanism are that all the APs in the ESS should have same Service Set Identifier (SSID). The bgscan provides three different modes. In None mode, the background scanning is disabled. The Simple mode enables periodic background scanning based on $RSSI_{th}$. When $RSSI_{RSU}$ is greater than or equals to the $RSSI_{th}$ perform background scanning after every T_l and when the $RSSI_{RSU}$ is less the $RSSI_{th}$ perform scanning after the T_s. In Learn mode of bgscan, the mobile node learns channels used by the network and try to avoid bgscans on other channels which reduces the effect on the data connection. A mobile node in Learn mode maintains a $Channels_{RSU}$.

We describe our proposed AdBack scanning scheme in Algorithm 1. AdBack relies on STA_{speed} in addition to $RSSI_{th}$ for the handover decision. When the vehicle is static, proposed AdBack does not perform scanning unless it reaches to $RSSI_{th}$, as it might interrupt some of the ongoing communication unnecessarily. If AdBack detects vehicle is moving at slow speed, it switches to the Simple mode and performs periodic background scanning. We assign fixed values to T_s and T_l, which can be derived from simple calculation on STA_{speed} and communication range of the RSU. Finally, if the vehicle is moving at high speed, daemon switches to Learn mode, where it tries to associate RSU learned previously (maintained as $Channels_{RSU}$) and avoids any interruption in communication due to scanning. The running daemon on OBU switches to Simple mode only if the vehicle is not able to associate RSUs present in $Channels_{RSU}$.

We claim that the proposed AdBack scheme provides better performance in terms of handover latency, packet loss and average throughput. We provide an emulation experiment to ascertain our claim.

5 Experimental Setup

To justify our claim for our proposed fast and secure handoff mechanism, we use Mininet-WiFi emulator [28]. This section covers implementation in Mininet-WiFi, selected reference scenario and parameters used for performance analysis in our experiment.

5.1 Implementation in Mininet-WiFi

We implemented our proposal AdBack scanning mechanism and existing standard IEEE 802.11r in Mininet-WiFi. A detailed description related to our implementation given at Mininet-WiFi discussion forum [29]. We installed freeradius server on Ubuntu and integrated with Mininet-WiFi. We implemented IEEE 802.11r and AdBack scanning (modified bgscan) in userspace that makes it more flexible and enables faster implementation than kernel-space. The association control is implemented for proper execution of handover due to mobility. The traffic simulator Simulation of Urban Mobility (SUMO) [27] is used to model mobility. For handover latency and packet loss analysis, the vehicle speed is fixed to 14 m/s. For the average throughput analysis, we have assigned random speed to the vehicles, which includes stoppage and slowdown. The varying mobility

```
Function BGScanSimple($RSSI_{RSU}, RSSI_{th}, T_s, T_l$):
    while true do
        if $RSSI_{RSU} \geq RSSI_{th}$ then
            Wait for $T_l$;
            Scan;
        else
            Wait for $T_s$;
            Scan;
        end
    end
return
Function BGScanLearn($RSSI_{RSU}, RSSI_{th}, T_s, T_l$):
    BGScanSimple($RSSI_{RSU}, RSSI_{th}, \infty, \infty$);
    $Channels_{RSU}$ ← Store Channels with active RSUs;
    while $RSSI_{RSU} \leq RSSI_{th}$ do
        Scan channels $ch | ch \in Channels_{RSU}$;
        if RSU Available then
            Associate with RSU;
        else
            BGScanSimple();
        end
    end
return
Input: $RSSI_{RSU}, RSSI_{th}, T_s, T_l, Speed_{th}$
BGScanSimple();
if $STA_{speed} == 0m/s$ then
    if $RSSI_{RSU} \leq RSSI_{th}$ then
        BGScanSimple();
    else
        Do not scan;
    end
else
    if $0 < STA_{speed} < Speed_{th}$ then
        BGScanSimple();
    else
        BGScanLearn();
    end
end
```

Algorithm 1. Proposed AdBack Scheme

helps us to model parking, braking and stoppage at the traffic light, fuel station and service center. We created the network topology for selected reference scenario.

5.2 Reference Scenario

As depicted in Fig. 1, we have taken smart city Wi-Fi setup as our reference scenario. Our scenario consists of set of RSUs deployed alongside the road. All RSUs are connected through a DS and belong to the same ESS. In the ESS, the vehicle performs intra-domain handover when it moves across RSUs. We exported Pune city road segment from an open street map (OSM) to model real traffic scenario using SUMO. The Corresponding Node represents the server installed at smart-city road authority to maintain real-time traffic information. The vehicle driver tries to fetch that data from the server while it is moving across those RSUs in the given road segment. An AAA server is maintained by the smart citys road administration to allow an authorized vehicle to use applications and services in a secured manner.

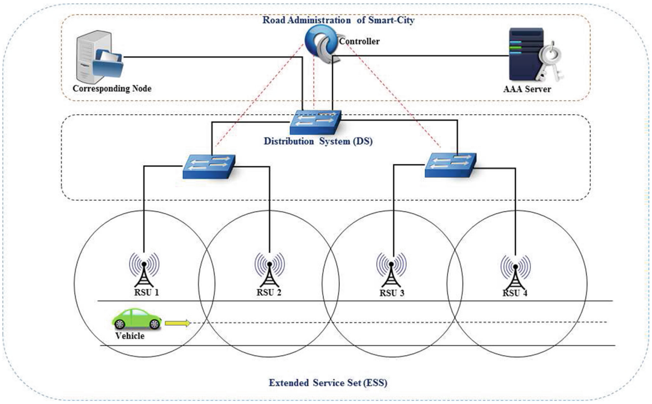


Fig. 1. Reference scenario of a smart city

5.3 Simulation Parameters

We run our simulation for 200 s on a smart city road segment of length approximately 3000 m. For realistic urban modeling, we have used Log Distance propagation loss model. We use SUMO for traffic modeling with varying speed, and maximum limit is set 50 Kmph (approx. 14 m/s). We are using 802.11g that operates in 2.4 GHz band and provide speed up to 54 Mbps. The RSUs have an overlapping coverage area of 20% of its radio range. Overlapping RSUs operate in non-overlapping channels. We have a set of 3 non-overlapping channels, i.e., a combination of 1, 6 and 11. Ten RSUs cover our target region. The vehicle is initially associated with RSU1 in a smart city network; the vehicle and the corresponding node communicates over IP in WLAN. Vehicular mobility ($Speed_{th}$) and signal strength threshold ($RSSI_{th}$) triggers handover when it moves across different RSUs deployed within the same ESS.

All our simulation parameters and modeling are close to the realistic scenario and as per the smart city Wi-Fi requirements. Table 2 shows details of simulation parameters used in our experiment.

Table 2. Simulation parameters

Parameters	Values
Operating system	Ubuntu 14.04-LTS
AAA server	FreeRADIUS Version 2.1.12
Traffic simulator	SUMO
Wi-Fi emulator	Mininet-WiFi
Wired link parameters	Bandwidth: 100 Mbps, Propagation Delay: 5 ms
RSU antenna type	Omnidirectional
Propagation model	Log Distance Propagation Loss Model
Path loss exponent	3.5 (Urban)
Simulation area	Approx 3 Km
Number of RSUs	10
Maximum velocity	14 m/s
Radio range of RSU	300 m
Wireless mode	IEEE 802.11g, Data rate: 54 Mpbs, RTS/CTS enabled
Authentication mode	WPA-Enterprise: 802.1X/EAP and FT-EAP
Authentication protocol	EAP-TLS
Traffic type	VoIP
Protocols used	ICMP, TCP
WLAN security framework tested	IEEE 802.11i, IEEE 802.11r FT over-DS
Scanning mechanism	Active, and AdBack
Simulation duration	200 s
Performance metrics	Packet Loss, Handoff delay and Avg. Throughput

6 Performance Evaluation

In WLAN based V2I communication, the QoS metrics can be Layer-2 handoff latency, packet loss, and throughput. The analysis of these parameters is helpful to evaluate the performance of V2I communication in a smart-city Wi-Fi deployment. In this section, the performance of our mechanism as well as the Legacy approach is assessed on Mininet-WiFi and compared based on these metrics.

6.1 Handoff Latency and Packet Loss with VoIP Like Traffic

The handoff latency is the time when the handover decision was made by the vehicle to join new RSU, and when successfully associated with the new RSU. In our experiment, VoIP like packets transmitted and received from the mobile node to the corresponding node. Packets are Internet Control Message Protocol (ICMP) packets since we are creating similar traffic using the ping utility. Packets of size 80 bytes are generated at every 20 ms to model VoIP like traffic. The objective is to identify handover latency and packet loss for VoIP communication during the handover process. In our experiment, nine handoffs performed, and we recorded the results of handover latency and packet loss. During a passage from one RSU to another, a mobile node continuously communicates with the corresponding node on the network using ICMP packets generated via ping utility at the rate of one per 20 ms.

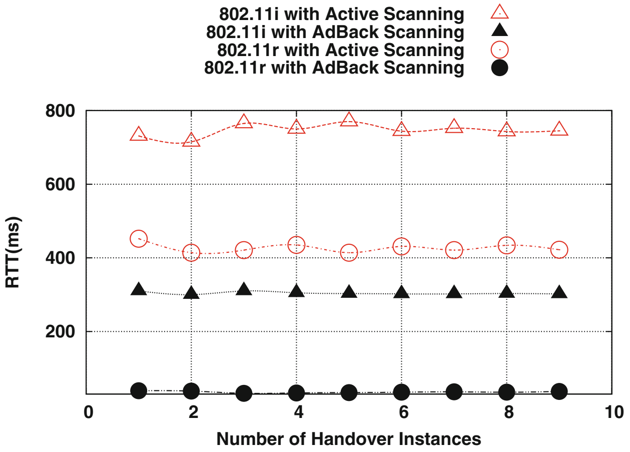


Fig. 2. Handover latency in milliseconds

In Fig. 2, we provide the simulation results for AdBack scanning and compare it with standard Active Scanning mechanism when used with IEEE 802.11i and IEEE 802.11r. We use round trip time (RTT) in milliseconds (ms) to measure the handover latency of the combinations mentioned above. We can see that the AdBack scanning with the IEEE 802.11r security framework is one of the fastest that takes approx. 35 ms in handover while other combinations take more than 300 ms. The proposed combination of scanning and reauthentication reduces the handover latency to the extent required for delay-sensitive applications such as VoIP.

Figure 3, shows that the AdBack scanning with IEEE 802.11r (FT-BSS Transition) has lower packet loss (no more than 5%) compared to other combinations. The mobile node in our proposed combination takes less time to reassociate with

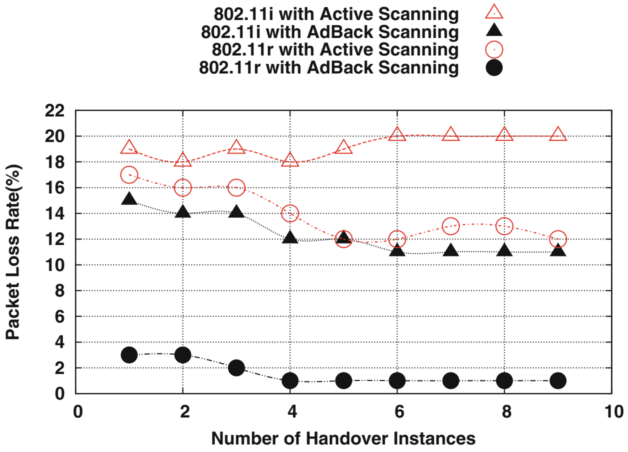


Fig. 3. Packet loss in percentage

the new RSU (approximately 35 ms) which reduces the number of packet losses. The packet loss is directly related to the handoff latency. Thus, our proposed mechanism outperforms. It has handover delay and packet loss of an acceptable QoS level for delay-sensitive applications.

6.2 Throughput Measurement Using Iperf Tool

To evaluate the performance in terms of throughput, we are using Iperf, network performance monitoring tool. The corresponding node works as a server and vehicular nodes as a client. The throughput measured through Transmission

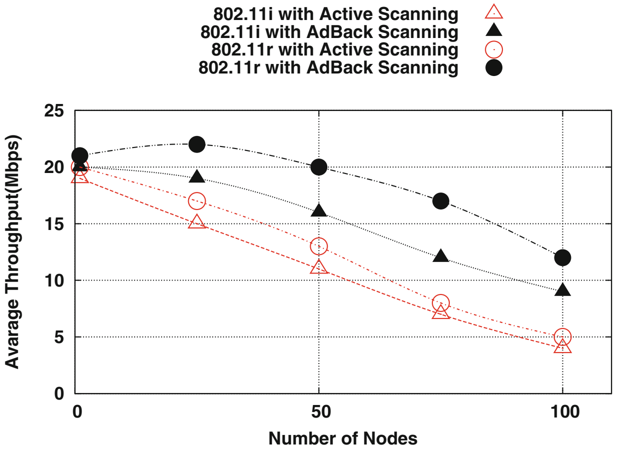


Fig. 4. Average throughput in Mbps

Control Protocol (TCP) tests in varying vehicular node density. A graph of the average throughput in Megabit per second (Mbps) versus vehicular node density shown in Fig. 4. We use the same test combinations and it can be seen that the disruption time due to these handover mechanisms affects performance. Although 802.11g mode is used, that supports a transmission speed of 54 Mbps, the throughput decreases significantly in the first method when the vehicular node density increases. The throughput in case of AdBack scanning is higher than the Active scanning. In the proposed combination of IEEE 802.11r with AdBack scanning, the handover execution is faster. It performs much better than the rest and throughput does not decrease rapidly in varying density. We can observe that throughput decreases when the number of vehicles increases because the wireless channel is shared among a large number of nodes.

7 Conclusion

In this work, we have proposed a new mechanism to maximize the quality of service for ITS applications and services in smart city Wi-Fi setup. In our proposed scheme, the collected information about the discovered APs during the periodic scan is cached. If RSSI drops to the defined threshold, it does not need to scan again, instead select potential AP from a cached neighbor list. This approach reduces the discovery delay drastically. Moreover, our scanning mechanism adapts to different mobility modes and does not require any modifications at the AP. In a smart-city highly secure Wi-Fi with WPA Enterprise (802.1X\EAP), reauthentication delay (inclusive of key-management) during each handover can cause a significant interruption to many services. The IEEE 802.11r FT over-the-DS reduces handover delay (due to reauthentication) by over 50% because the mobile node is already pre-authenticated in its network domain.

The WPA2 Enterprise (WPA2 802.1X/EAP) security has been used to provide authentication, privacy, integrity, and availability. This security standard is still considered the gold standard for wireless network security. The combination of our proposed AdBack scanning and IEEE 802.11r based fast reauthentication mechanism maximizes network throughput, minimizes handover latency and packet loss and complies with the QoS requirements for V2I applications mentioned in Sect. 1. This approach can be helpful for delay-sensitive applications such as VoIP and real-time services.

In this study, we did not compare the energy consumption, because vehicles do not suffer from power constraints like handheld devices. If a vehicle engine is running, it can power itself and always have sufficient energy. Most of the time, vehicles are either in the parking lot or driveway, during this period the proposed daemon running on OBU is energy efficient and will not trigger the scanning if it receives a better signal.

Our approach is simple in its implementation and does not require any change in AP side or installation of any additional server. As a future work, we are focusing on dynamic handoff management, threshold selection along with load balancing in a high mobility scenario of vehicular communication.

Acknowledgment. We are thankful to R. R. Fontes, one of the authors of [32] and developer of Mininet-WiFi who provided us help in solving the problems related to our experimental setup on Mininet-WiFi.

References

1. Consortium, C.V.S.C., et al.: Vehicle safety communications project: task 3 final report: identify intelligent vehicle safety applications enabled by DSRC. National Highway Traffic Safety Administration, US Department of Transportation, Washington DC (2005)
2. IEEE Std 802.11r/D01.0: Draft Amendment to Standard for Information Technology Telecommunications and Information Exchange Between Systems LAN/MAN Specific Requirements Part 11: Wireless Medium Access Control (MAC) and Physical Layer Specifications: Amendment 8: Fast BSS Transition
3. IEEE Std 802.11i: IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications: Amendment 6: Medium Access Control Security Enhancements
4. Mishra, A., Shin, M., Arbaugh, W.: An empirical analysis of the IEEE 802.11 MAC layer handoff process. ACM SIGCOMM Comput. Commun. Rev. **33**(2), 93–102 (2003)
5. Shin, S., Forte, A.G., Rawat, A.S., Schulzrinne, H.: Reducing MAC layer handoff latency in IEEE 802.11 wireless LANs. In: Proceedings of the Second International Workshop on Mobility Management and Wireless Access Protocols, pp. 19–26. ACM (2004)
6. Bangolae, S., Bell, C., Qi, E.: Performance study of fast BSS transition using IEEE 802.11r. In: Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing, pp. 737–742. ACM (2006)
7. Park, S.-H., Kim, H.-S., Park, C.-S., Kim, J.-W., Ko, S.-J.: Selective channel scanning for fast handoff in wireless LAN using neighbor graph. In: Niemegeers, I., de Groot, S.H. (eds.) PWC 2004. LNCS, vol. 3260, pp. 194–203. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30199-8_16
8. Sarma, A., Chakraborty, S., Nandi, S., Choubey, A.: Context aware inter-bss handoff in IEEE 802.11 networks: efficient resource utilization and performance improvement. Wireless Pers. Commun. **77**(4), 2587–2614 (2014)
9. Brik, V., Mishra, A., Banerjee, S.: Eliminating handoff latencies in 802.11 WLANs using multiple radios: applications, experience, and evaluation. In: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement, IMC 2005, p. 27 (2005)
10. Jin, S., Choi, S.: A seamless handoff with multiple radios in IEEE 802.11 WLANs. IEEE Trans. Veh. Technol. **63**(3), 1408–1418 (2014)
11. Ramani, I., Savage, S.: SyncScan: practical fast handoff for 802.11 infrastructure networks. In: INFOCOM 2005, 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Proceedings IEEE. vol. 1, pp. 675–684. IEEE (2005)
12. Chen, Y.S., Chuang, M.C., Chen, C.K.: DeuceScan: deuce-based fast handoff scheme in IEEE 802.11 wireless networks. IEEE Trans. Veh. Technol. **57**(2), 1126–1141 (2008)

13. Yoon, M., Cho, K., Li, J., Yun, J., Yoo, M., Kim, Y., Shu, Q., Yun, J., Han, K.: Adaptivescan: the fast layer-2 handoff for WLAN. In: 2011 Eighth International Conference on Information Technology: New Generations (ITNG), pp. 106–111. IEEE (2011)
14. Wu, T.Y., Obaidat, M.S., Chan, H.L.: Qualityscan scheme for load balancing efficiency in vehicular ad hoc networks (VANETs). J. Syst. Softw. **104**, 60–68 (2015)
15. Lee, J., Cho, S.-P., Kim, H.: Position based handover control method. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganà, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3481, pp. 781–788. Springer, Heidelberg (2005). https://doi.org/10.1007/11424826_83
16. Montavont, J., Noel, T.: IEEE 802.11 handovers assisted by GPS information. In: 2006 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2006), pp. 166–172. IEEE (2006)
17. Nicholson, A.J., Noble, B.D.: Breadcrumbs: forecasting mobile connectivity. In: Proceedings of the 14th ACM International Conference on Mobile Computing and Networking, pp. 46–57. ACM (2008)
18. Zhao, Y., Li, W., Lu, S.: Navigation-driven handoff minimization in wireless networks. J. Netw. Comput. Appl. **74**, 11–20 (2016)
19. Navda, V., Subramanian, A.P., Dhanasekaran, K., Timm-Giel, A., Das, S.: Mobisteer: using steerable beam directional antenna for vehicular network access. In: Proceedings of the 5th International Conference on Mobile Systems, Applications and Services, pp. 192–205. ACM (2007)
20. Balasubramanian, A., Mahajan, R., Venkataramani, A., Levine, B.N., Zahorjan, J.: Interactive wifi connectivity for moving vehicles. ACM SIGCOMM Comput. Commun. Rev. **38**(4), 427–438 (2008)
21. Deshpande, P., Kashyap, A., Sung, C., Das, S.R.: Predictive methods for improved vehicular wifi access. In: Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services, pp. 263–276. ACM (2009)
22. Gañán, C.H., Reñé, S., Muñoz-Tapia, J.L., Esparza, O., Mata-Díaz, J., Alins, J.: Secure handoffs for V2I communications in 802.11 networks. In: Proceedings of the 10th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks, pp. 49–56. ACM (2013)
23. Tabassam, A.A., Trsek, H., Heiss, S., Jasperneite, J.: Fast and seamless handover for secure mobile industrial applications with 802.11r. In: 2009 IEEE 34th Conference on Local Computer Networks, LCN 2009, pp. 750–757. IEEE (2009)
24. Martinovic, I., Zdarsky, F.A., Bachorek, A., Schmitt, J.B.: Measurement and analysis of handover latencies in IEEE 802.11i secured networks. In: Proceedings of the 13th European Wireless Conference (EW2007), Paris, France (2007)
25. Machań, P., Wozniak, J.: On the fast BSS transition algorithms in the IEEE 802.11r local area wireless networks. Telecommun. Syst. **52**(4), 2713–2720 (2013)
26. WPASupplicant: Bgscan. https://w1.fi/cgit/hostap/plain/wpasupplicant/wpa supplicant.conf
27. Behrisch, M., Bieker, L., Erdmann, J., Krajzewicz, D.: Sumo-simulation of Urban mobility: an overview. In: Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation, ThinkMind (2011)
28. Fontes, R.R., Afzal, S., Brito, S.H., Santos, M.A., Rothenberg, C.E.: Mininet-wifi: emulating software-defined wireless networks. In: 2015 11th International Conference on Network and Service Management (CNSM), pp. 384–389. IEEE (2015)
29. MininetWiFi: Discussion group. https://groups.google.com/forum/#!topic/mininet-wifi-discuss/jez7TA98eJQ

Ego Based Community Detection in Online Social Network

Paramita Dey[1](✉), Sarbani Roy[2], and Sanjit Roy[1]

[1] Department of Information Technology, GCECT, Kolkata, India
dey.paramita77@gmail.com

[2] Department of Computer Science and Engineering, Jadavpur University, Kolkata, India
sarbani.roy@jadavpuruniversity.in

Abstract. Social network can be represented by a graph, where individual users are represented as nodes/vertices and connections between them are represented as edges of the graph. The classification of people based on their tastes, choices, likes or dislikes are associated with each other, forms a virtual cluster or community. The basis of a better community detection algorithm refers to within the community the interaction will be maximized and with other community the interaction will be minimized. In this paper, we are proposing an ego based community detection algorithm and compared with three most popular hierarchical community detection algorithms, namely edge betweenness, label propagation and walktrap and compare them in terms of modularity, transitivity, average path length and time complexity. A network is formed based on the data collected from a Twitter account, using Node-XL and I-graph and data are processed in R based Hadoop framework.

Keywords: Community detection · Social network · Twitter
Ego based community detection · Edge betweenness
Label propagation · Walktrap · Modularity · Transitivity

1 Introduction

Social network analysis is the new emerging, but quickly extended inter-discipline area which become most ubiquitous topic from both industrial and research viewpoint. Twitter is a social network, where interactions between the users are made up of in the form of a Tweet, Retweet, like etc. From the social graph, structural properties like significantly coherent patterns, influence propagation, community structure [1] and power-law distribution can be derived. Community structure, or clustering, denotes the set of vertices in community, where edges connecting nodes of same cluster are significantly much higher than the edges connecting nodes of different clusters. These types of communities can be treated as an independent module of the graph.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 205–210, 2018.
https://doi.org/10.1007/978-3-319-72344-0_16

Though several algorithms existed for community detection, all of them have their advantages and disadvantages. In this paper, we propose ego based community detection algorithm and compare with three hierarchical community detection algorithms edge betweenness, label propagation and walktrap. Data are collected from the Twitter network using I-Graph and Node-XL. Usually online social network consists of large scale of data and it is represented as the scale free network. The time complexity of our proposed algorithm is linear, that is in the order of n. We choose two most important quality parameters of community detection to show the effectiveness of our algorithm: modularity and transitivity. Modularity denotes how well the different communities are segregated whereas transitivity denotes the higher clustering within the community. Moreover, we calculate the average path length of each module after community detection and compare it with an average path length of the whole network.

Remaining of the paper is organized in the following manner. We proposed and demonstrate ego based community detection algorithm in Sect. 2. Three state of the art hierarchical community detection algorithms are discussed in Sect. 3. The results of the comparative study are analyzed in Sect. 4. In Sect. 5, we have concluded our paper with future scope.

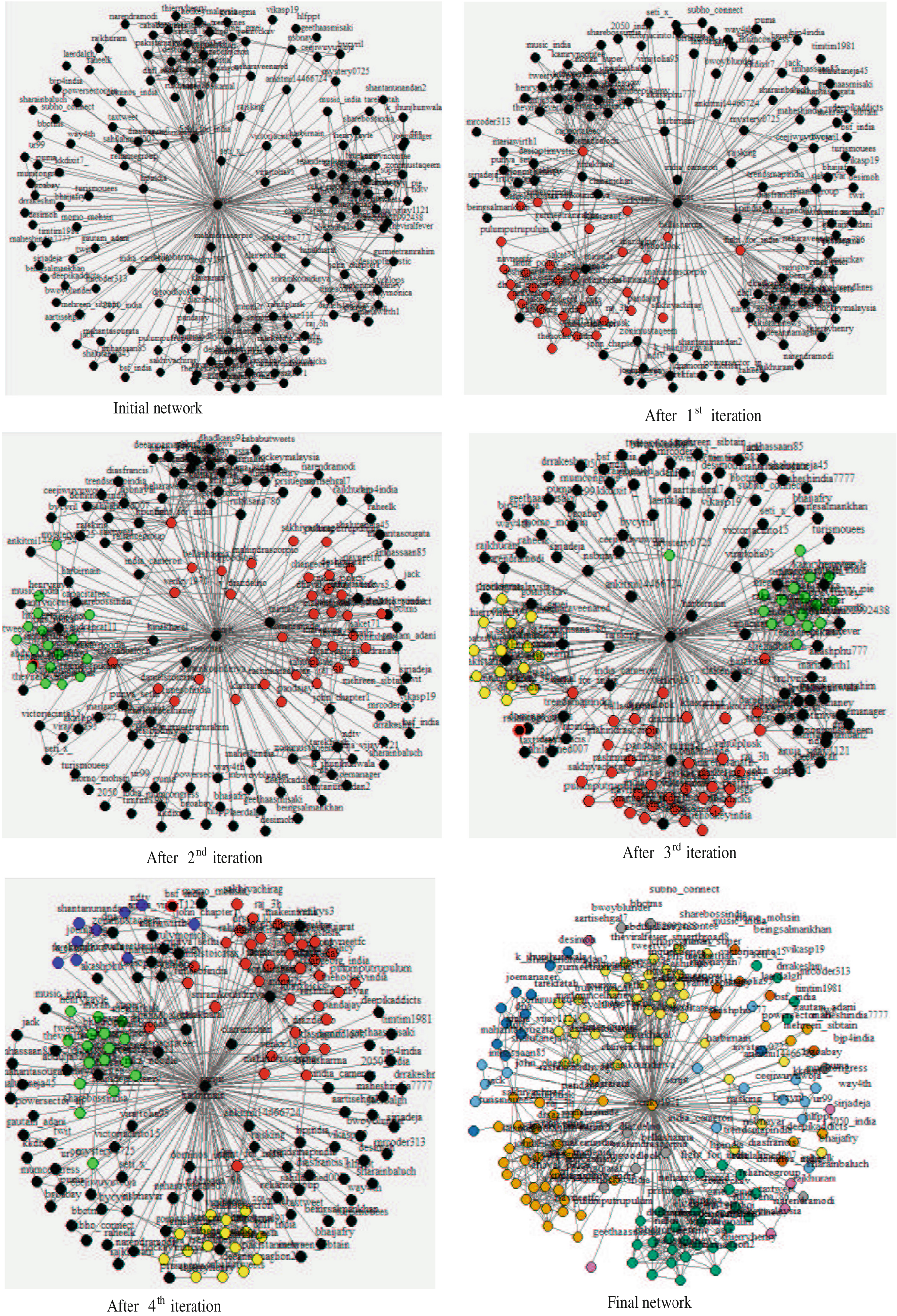
2 Ego Based Community Detection

We used real world network by collecting Twitter data from an existing Twitter account using NodeXL. Then Twitter data is extracted and analysed in Hadoop based R platform and output is visualised through I-Graph.

This section presents the proposed algorithm EBCD(Ego based community detection) in detail. The EBCD employs a generic algorithm to detect ego of the network and then forms the communities. Ego is the focal node within a network or graph. Ego denotes the node which is connected to the maximum number of nodes within a graph.

Initially the *ego* (node with maximum degree) is derived for the whole network. Let $G = (V, E)$ denote a network, where $V = v_1, v_2, ..., v_n$ and $E = e_1, e_2, ..., e_m$ denotes the node set and edge set respectively. Using function $findNodeDegree(v)$, the node degree value of all vertices $V = v_1, v_2, ..., v_n$ is derived from the network. A node with a maximum node degree is derived as the ego. Neighbors (adjacent vertices) of that ego node are derived from the network. Assign neighbors, including the ego node into single community C. A sub-graph is generated by removing all edges and nodes belongs to that community. Removal of this set of edges and nodes from the network segregates the network in partitioned network. This process continues until there exists at least one node with $degree \geq 3$ in the network. As a result there exists no open triplets in the network. Stepwise community formation using ego based community are shown in Fig. 1.

In our dataset, we get 18 communities using Ego based community detection algorithm with modularity value 0.4973. The average transitivity value is derived as 0.576 and average path Length is 1.447493.



Initial network

After 1st iteration

After 2nd iteration

After 3rd iteration

After 4th iteration

Final network

Fig. 1. Community formation in Ego based community detection

Algorithm 1. Ego based community detection

```
/* Input $G(V, E)$ */
$Repeat$
{
$\forall v \in V$
{
$findNodeDegree(v)$;
/*calculate node degree for all nodes*/
$Ego \leftarrow maxNodeDegree(v)$;
/*Ego is defined as the node with highest node degree*/
$value \leftarrow NodeDegreeEgo(e)$;
/*value defines the node degree value of ego i.e. the edged connected to ego
node*/
}
$\forall v \in V$ and connected to Ego && $\forall e \in E$ and connected to Ego
{ $v, e \in C$; /* Ego and all nodes connected with ego are considered as single
community*/
}
/* Remove all edges and vertices belongs to that community from the original
graph.*/
$\forall v \in C$ { $V \leftarrow V - v$ ; }
$\forall e \in C$ { $E \leftarrow E - e$ ; }
} $until(value \leq 3)$
/* iteration completed after there is no open triplate within community.*/
```

3 Benchmark Community Detection Algorithms

Three benchmark hierarchical algorithms are discussed in this section. Algorithms are applied to the same data set derived from the Twitter network as mentioned earlier.

3.1 Edge Betweenness

Edge betweenness of an edge can be defined as the number of shortest paths involved this edge in their paths. Edge betweenness algorithm proposes that if two nodes have more than one shortest path, each edge will assign with an equal weight, subject to total weight will be 1. The edge with the highest betweenness will be removed and thus the community will be separated [2].

3.2 Label Propagation

In label propagation algorithm [4], each label is assigned to a different community value. The node is updated to the value which is more frequent among the neighbour nodes of that node. If more than one label is frequent among the neighbours, it randomly chooses one of them. Thus the label updating for node x can be expressed as:

$$l_x^{\text{new}} = \arg_l max(\sum_{u=1}^{x} A_{ux}\delta(l_x, l))$$

where l_x^{new} indicates a new label for node x. The iteration is done till each node assigned with label which is most frequent label among the neighbours.

3.3 Walktrap

Walktrap algorithm [3] is based on Random walk within short distance among the network. It is a hierarchical bottom up approach which consider agglomeration of the nodes of the network.

4 Result and Discussion

We have analyzed the Twitter network based on two aspects of quality functions, modularity and transitivity. In our sample Twitter network, global transitivity is calculated as 0.2297137. After community detection transitivity of each module are calculated and the average value of each module is derived as the transitive value of that algorithm. A comparative chart showing modularity, average transitivity value and average path length after clustering is represented in Table 1.

Table 1. Comparative chart of community detection algorithms with respect to modularity, transitivity, average path length and time complexity

Algorithm	Modularity	Transitivity	Average path length (after clustering)	Time complexity
Ego based community detection	0.4973	0.576	1.447493	$O(n)$
Edge betweenness	0.4895973	0.76054315	1.239548	$O(n^3)$
Label propagation	0.5259585	0.66849175	1.154545	$O(n^2)$
Walktrap	0.6402915	0.527455062	1.428795	$O(n^2)$

Comparing these four algorithms, it is evident that modularity values for Walktrap algorithm exhibits highest modularity value. Modularity denotes how well the different communities are segregated i.e. for walktrap algorithm, the overlapping is minimized. Our proposed algorithm exhibits modularity as 0.4973, which is better than the edge betweenness algorithm. Transitivity refers to the extent to which the relation that relates two vertices, connected to a network through an edge. Edge betweenness show maximum average transitivity. It can be noted that for each algorithm the average module transitivity is always much higher than the global transitivity of the network (which is 0.2297137 for our dataset) and this is ideal for a good community detection algorithm as it reflects that each module's connectivity is more dense than the total network's connectivity i.e. nodes within modules are more densely connected. From the extracted dataset average path length is derived as 2.947 for the whole network without clustering. Edge betweenness show minimum average path length among all community detection algorithm. For all the algorithms, average path length within a community is much less than the average path length of the total network, which is desirable after communities are formed as shown in Table 1.

The main advantage of our proposed algorithm is the time complexity of this algorithm over other algorithm as shown in Table 1. The proposed ego based community detection algorithm is in the order of $O(n)$ as there is only linear searching for the ego of the graph. For the derivation of edge betweenness, all possible shortest paths between all vertices has to calculate first which required time complexity in the order of n^2. Community detection using edge betweenness algorithm is in the order of $O(n^3)$. Similarly for rest two algorithms' time complexities are in the order of $O(n^2)$.

5 Conclusion

The proposed ego based community detection approach is comparable to the other state of the art community detection algorithms in terms of all the quality functions as demonstrated in the test case presented here. As social networks exhibits the property of scale free networks, the time complexity is very important of community detection algorithm. In many social network applications, it requires quick and correct identification of the vertices most connected in some aspects. As there are no calculations of the distances in this community detection algorithm, it is must faster than other algorithms. But, one drawback of the proposed algorithm is that here only first hop distances from the ego is considered. In our future work, we will consider other factors like the degree of adjacent nodes, overlapping communities and make modifications on that.

References

1. He, L., Lu, C.T., Ma, J., Cao, J., Shen, L., Yu, P.S.: Joint community and structural hole spanner detection via harmonic modularity. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, pp. 875–884, New York, NY, USA (2016). http://doi.acm.org/10.1145/2939672.2939807
2. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**, 026113 (2004)
3. Pons, P., Latapy, M.: Computing communities in large networks using random walks. J. Graph Algorithm Appl. **10**(2), 191–218 (2006)
4. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E **76**, 036106 (2007). https://link.aps.org/doi/10.1103/PhysRevE.76.036106

A Gateway Virtual Network Function for InfiniBand and Ethernet Networks

Utkal Sinha[1(✉)], Ratnakar Dash[1], and Nandish Jayaram Kopri[2]

[1] National Institute of Technology Rourkela, Rourkela, Odisha 769008, India
utkalsinha.nits@gmail.com
[2] Unisys Global Services India, Bangalore 560025, India

Abstract. The InfiniBand Architecture is an industry standard interconnect technology developed by the InfiniBand Trade Association in 1999 that provides higher reliability, higher availability, better performance, higher scalability than can be achieved using traditional interconnect technologies like the Ethernet. Due to its extremely lower latencies, InfiniBand is used to connect computer nodes in datacenters whereas Ethernet is used for management, storage area networks (SANs), etc. Therefore, it is imperative to have a mechanism to translate the InfiniBand frames to the Ethernet frames to make the two types of networks seamlessly communicate with each other. This paper introduces a gateway model which can be implemented as a virtual network function that translates Ethernet and InfiniBand frames using IPoIB protocol. Experimental results show that performance of the proposed solution is at par with that of Ethernet. Ethernet is considered for performance benchmark comparison since in the testing environment it had lesser bandwidth as compared to InfiniBand.

Keywords: IPoIB gateway · Ethernet - InfiniBand routing · Packet translation
Network function virtualization

1 Introduction

Today's business world requires a reliable, secure and efficient access to information to achieve a competitive advantage. The traditional file cabinets and a pile of papers have been replaced by the computers that store and manage information electronically. Computer networking technologies join these elements together. The public Internet allows businesses around the world to share information with each other and their customers. All these computers are connected to each other using various interconnect technologies. Some of the known interconnect technologies are – InfiniBand, Ethernet, Fiber Channel, 10 Gigabit Ethernet, Cray Interconnect.

Network functions virtualization (NFV) is an emerging technology to virtualize the network services that are now being carried out by proprietary, dedicated hardware. The incorporation of NFV will decrease the amount of proprietary hardware that is needed to launch and operate network services.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 211–216, 2018.
https://doi.org/10.1007/978-3-319-72344-0_17

This work comprises designing an efficient algorithm to translate InfiniBand and Ethernet frames that will run in a virtual gateway which in turn may run in a virtual machine (VM) to support NFV.

2 Background

Ethernet is the most widely installed local area network (LAN) technology. It describes the data format to be used by network devices for transmission on the same network segment and how to put that formatted data out on the network connection [1, 2]. Ethernet uses MAC addresses to identify and deliver frames among computers that are on the same physical network segment. The InfiniBand Architecture (IBA) has its own computer networking protocol stack as Upper Layers, Transport Layer, Network Layer, Link Layer, and Physical Layer [3]. The IBA [3] defines various data packet structure depending on IBA packet types. InfiniBand uses a non-IP based addressing mechanism and does not support sockets. Therefore, inherently, it does not support TCP/IP based applications as well. To support IP based applications on top of IB fabric, the IETF IPoIB working group [4] has specified the IPoIB protocol [5, 6]. It is important to note that since this protocol is designed to support IP based applications or protocols which use the IP address, it does not support VLAN. The IPoIB implementation is done at the layer 2 of OSI protocol stack.

3 Motivation

The InfiniBand (IB) is comparatively a new interconnect technology. Hence, some of the data centers have both of the interconnect technologies as they are migrating from the Ethernet to InfiniBand. So, it is imperative to have a gateway which will seamlessly translate the Ethernet frames to the IB frames and vice versa. Existing hardware gateway solutions are expensive. Building a software gateway on a Network Functions Virtualization (NFV) platform to translate these two types of frames would not only reduce the cost but also increase manageability and flexibility along with reliable performance. For example, a higher number of virtual network ports could be bonded to a particular gateway VNF either statically or dynamically.

4 Literature

There have been some attempts to translate or bridge InfiniBand and Ethernet networks. They are either implemented in a hardware switch or a software switch. But, one thing is common in most of the approaches is that they either encapsulate IP packets inside an IPoIB frame [7] or the entire Ethernet frame inside an IB Frame [8]. Both of the approaches have their benefits and limitations.

For instance, the approach in [7] proposes a software implementation of an IPoIB gateway with Network Address Translation (NAT). Here, the IPoIB gateway receives an Ethernet Frame via its Network Interface Card (NIC), inspects and modifies the IP destination address and then re-injects the frame in the IPoIB interface to deliver it to the IB destination node. The destination IPoIB node then retrieves the IP packet from the received IPoIB frame. Though this approach is straightforward from the implementation perspective, this method has limitations as follows: inefficient because of IP fragmentation and high CPU utilization, high latency (IP layer solution), poor bandwidth utilization and no load balancing which decreases the performance, and does not support VLAN-PKey mapping since IPoIB does not support VLAN in itself.

Again, approach [8] suggests encapsulating the entire Ethernet frame inside an IB frame at the gateway node which is then injected to the HCA by the IB driver. Although this approach has shown significant performance improvement over [7], it also has some limitations as follows: Periodic exchange of gateway control information may lead to poor bandwidth utilization, creation and maintenance of mapping tables, do not support PKey—VLAN mapping.

Also, there has been an attempt to translate these two types of frames using a software switch [9]. This approach requires a MAC—LID mapping table in the IB SA [3] to which the translator queries whenever it wishes to translate Ethernet and InfiniBand frames. But still it does not handle the PKey—VLAN mapping to create separate broadcast domains and secure fabrics across the two networks.

5 Proposed Solution

The IPoIB driver is registered to the IB SM with a QPN (queue pair number) and a LID (Local identifier) and to the OS kernel with a virtual MAC address. The IPoIB driver handles the backend works within the IB physical segment like querying the SM for destination LID details, destination path information, etc. So, in the proposed method we try to leverage this feature to make the Ethernet node and the InfiniBand node seamlessly communicate with each other. A typical network topology for the proposed solution is shown in Fig. 1.

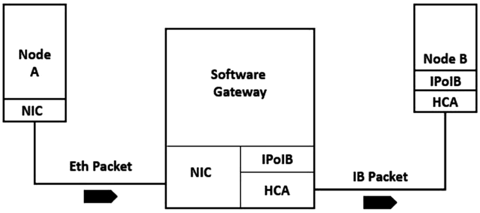
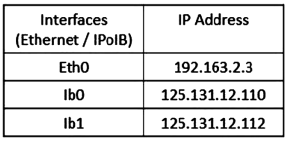


Fig. 1. Proposed solution network topology and architecture.



Interfaces (Ethernet / IPoIB)	IP Address
Eth0	192.163.2.3
Ib0	125.131.12.110
Ib1	125.131.12.112

Fig. 2. Gateway system interface list along with their IP addresses.

Here, the gateway node receives the Ethernet Packet via its Ethernet NIC interface which is then transferred to the gateway (GW) software (GWS) running in the application layer at GW node. The GWS inspects the received packet, makes changes to it if needed and then forwards it via the IPoIB interface. The IPoIB driver encapsulates the

IP packet inside the IPoIB frame and delivers it to the destination IB node. Similar is the case for InfiniBand to Ethernet communication only difference in that the gateway node receives the IPoIB packet via the IPoIB network interface and forwards the received packet via the Ethernet NIC. The GWS also maintains a table of all the interfaces available on the GW system along with their corresponding IP addresses as shown in Fig. 2. The algorithm for the GWS is shown in Algorithm 1.

Algorithm 1 IPoIB Gateway Algorithm

```
procedure GATEWAYMODULE(pkt)
/* Gateway module receives the network packet */
   destIP: = getDestIP (pkt)
   srcIP: = getSrcIP (pkt)
   if IncomingPacketFromEthernetInterface(pkt) then
      if CheckInInterfaceList (destIP, srcIP) then
      /* If there is a network interface with bind IP
      same as destination IP but not same as source IP,
      then forward received packet to that interface */
         intf: = CheckInInterfaceList (destIP, srcIP)
         ForwardReceivedPacketViaIPoIB(intf)
      else DiscardReceivedPacket(pkt)
      end if
   else
   /* Received packet is an IPoIB packet. Received pack-
et via the IPoIB network interface */
      if IncomingPacketFromEthernetInterface(pkt) then
         intf: = CheckInInterfaceList (destIP, srcIP)
         ForwardReceivedPacketViaEth(intf)
      else DiscardReceivedPacket(pkt)
      end if
   end if
end procedure
```

The *CheckInInterfaceList* () method in Algorithm 1, checks if there is any network interface in the interface list maintained having bind IP address same as that of received packet destination address. If there is an entry, then it returns the network interface corresponding to the mapped IP address and received packet is then forwarded to that particular interface (achieved via functions *ForwardReceivedPacketViaIPoIB* () and *ForwardReceivedPacketViaEth* ()). Unlike existing approaches which deal with modifying the destination address at the application layer, the introduction of network interface list in our proposed gateway reduces the processing overheads by performing all the translation operation at the network layer itself (which leads to fewer buffer copies of the received packet).

6 Implementation and Results

For the implementation of the proposed solution three guest partitions – two Windows Server 2012 and one SLES11, on a PEPP platform [10] is considered as shown in Fig. 3.

SUSE Linux (SLES11) system is the Gateway system. SLES11 has been configured with IPTables [11] to forward the received packet to the application layer. Also routing entries have been made to route packets to the windows machines. PassMark PerformanceTest 8.0 [12] for the bandwidth test between Windows System (NIC) SLES11 (Gateway, NIC IPoIB) Windows System (IPoIB). The data block size varied from 32 bytes to 16384 bytes. Figures 5 and 6 shows the bandwidth tests for

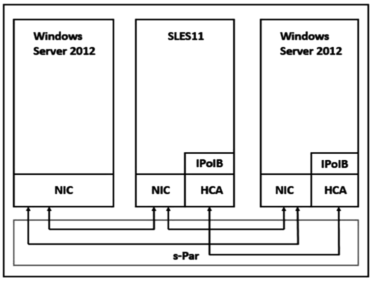


Fig. 3. Implementation environment

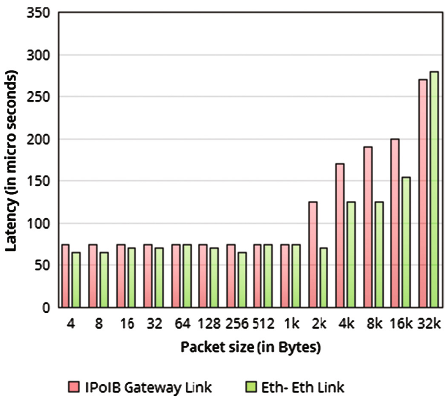
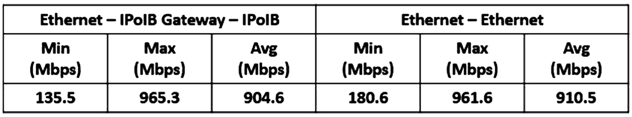


Fig. 4. UDP connection latencies for the two links

Ethernet IPoIB gateway IPoIB and Ethernet-Ethernet links for both TCP and UDP connections. PsPing [13] has been used to measure the average latencies of the two links. The formula equation for calculating the one-way average latency is:

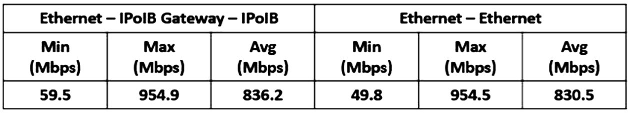
$$\mathit{Average\ one}-\text{way latency} = \mathrm{RTT}/2 \tag{1}$$

(RTT is the Round Trip Time). The variance of the UDP connection latencies on these links for small sized packets is shown in Fig. 4. Also, JPerf 2.0.2 [14] showed the packet loss ratio to be 0%.



Ethernet – IPoIB Gateway – IPoIB			Ethernet – Ethernet		
Min (Mbps)	Max (Mbps)	Avg (Mbps)	Min (Mbps)	Max (Mbps)	Avg (Mbps)
135.5	965.3	904.6	180.6	961.6	910.5

Fig. 5. Bandwidth test for TCP connections



Ethernet – IPoIB Gateway – IPoIB			Ethernet – Ethernet		
Min (Mbps)	Max (Mbps)	Avg (Mbps)	Min (Mbps)	Max (Mbps)	Avg (Mbps)
59.5	954.9	836.2	49.8	954.5	830.5

Fig. 6. Bandwidth test for UDP connections

7 Conclusion and Future Work

The proposed method for seamless communication between the Ethernet and InfiniBand networks has been found to be at par with the performance of Ethernet with 1Gbps connection speed, regarding bandwidth and latencies. Also, JPerf showed that the method is reliable. Although, there are some limitations to it which may lead to future work. The proposed method uses the IP address along with IPoIB interface to translate the Ethernet and InfiniBand packets which leads to lack of data link layer management regarding creating and managing VLAN across the heterogeneous fabric. Instead, a data link layer method to translate the two types of frames at link layer would not only improve the performance but also increase efficient management of the physical fabric.

References

1. ieee802.org. Ieee 802.3 ethernet working group. http://www.ieee802.org/3/
2. The ethernet, a local area network data link layer and physical layer specifications. http://research.microsoft.com/enus/um/people/gbell/Digital/Ethernet%20Blue%20Book.pdf
3. InfiniBand Trade Association: InfiniBand architecture volume 1 and volume 2. http://www.infinibandta.org/content/pages.php?pg=technology%20public%20specification
4. Internet engineering task force. https://www.ietf.org/
5. Kashyap, V., Chu, J.: Transmission of IP over infiniband. https://tools.ietf.org/html/rfc4391
6. Kashyap, V.: IP over InfiniBand (IPoIB) architecture. https://tools.ietf.org/html/rfc4392
7. Zifeng, X., Hongwei, Z., Yonghao, Z., Jizhong, H.: Design and performance evaluation of IPoIB gateway. In: 2006 International Workshop on Networking, Architecture, and Storages (IWNAS 2006), pp. 3–8. IEEE, Shenyang (2006)
8. Lin, Y., Li, N., Lv, G., Sun, Z.: Research and implementation of IB and ethernet stateless conversion technology. In: Conference Anthology, pp. 1–4. IEEE, China, January 2013
9. Michael T.: Systems and methods for ethernet frame translation to internet protocol over InfiniBand. US Patent App. 13/766,340., 14 August 2014. https://www.google.co.in/patents/US20140226659
10. Unisys Corporation Forward! information center. https://public.support.unisys.com/forwardic2.0/index.jsp?topic=%2Fforwardinformationcenter%2Fhtml%2Fsection-000001529.htm
11. Harald W.: IPTables. http://ipset.netfilter.org/iptables.man.html
12. PassMark® Software Pty Ltd. Passmark performancetest. http://www.passmark.com
13. Mark R: Psping. https://technet.microsoft.com/en-us/sysinternals/jj729731.aspx
14. Nicolas R.: JPerf. https://code.google.com/p/xjperf/

GMP2P: Mobile P2P over GSM for Efficient File Sharing

Sumit Kumar Tetarave[1(✉)], Somanath Tripathy[1], and R. K. Ghosh[2]

[1] Department of Computer Science and Engineering,
Indian Institute of Technology Patna, Patna, India
{sktetarave,som}@iitp.ac.in

[2] Department of Computer Science and Engineering,
Indian Institute of Technology Kanpur, Kanpur, India
rkg@iitk.ac.in

Abstract. Implementing Peer to Peer (P2P) system on a cellular network is an interesting idea to provide distributed storage, that has caught attention of many researchers. Leaving aside the legal issues, overcoming technical challenges are key to the development of cellular based P2P business applications. In this paper, we propose a mobile P2P file sharing system called GMP2P on GSM-GPRS network. GMP2P integrates distributed hash table (DHT) mechanism into GSM mobile stations and base transceiver stations. The proposed solution addresses the issues related to efficient P2P file sharing over GSM-GPRS network without requiring a centralised server. The communication cost involved in searching and downloading of the shared files is also analysed and the results are compared with existing mobile P2P schemes. GMP2P is found to be more efficient and scalable.

1 Introduction

The rapid growth of the mobile devices (smart phones, PDAs) having capabilities comparable to desktop computers has raised the demand for increasing levels of content sharing in mobile environment. According to Ericsson mobility report [3], most of the digital data now a days are shared on wireless environment. In the near future, this trend likely to increase due to higher data rates offered by 3G and 4G networks.

File sharing is a technique through which contents (text, image, audio, video) can be stored, searched and accessed by different user groups. Many P2P applications like Napster, Gnutella, eDonkey, BitTorrent, Freenet, are used for content sharing over the wired Internet [11]. For content distribution on a mobile system, P2P overlay would be an eminently suitable mechanism due to, high reliability, ease of communication between the peers, and extensible distribution of the resources [4].

Three possible options are available for the underlay physical network which can support implementation of a P2P overlay, namely, (i) Bluetooth, (ii) Mobile

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 217–231, 2018.
https://doi.org/10.1007/978-3-319-72344-0_18

Ad hoc Network and (iii) GSM-GPRS network. Bluetooth offers file exchange facility only if both the peers are within a maximum permissible transmission range of 100 m (class 1). Alternatively, mobile P2P (MP2P) can be implemented over Mobile Ad hoc network (MANET) underlay to support communication between peers within a maximum transmission range of 300 m. Thus, neither Bluetooth nor MANET can provide a satisfactory underlay platform for implementation of a wide area mobile P2P system content distribution and sharing application. This motivated us to explore GSM infrastructure for underlay network support to provide wider area communication.

Existing MP2P applications rely on wired infrastructure underlay integrated with a centralised index server. Each MS pre-stores the address of the Index server and the Index server provides index of the shared contents along with the corresponding mobile station (MS) ID. Mobile stations retrieve index information from the centralised Index server by sending explicit requests. After receiving the index information, MS communicates directly with target MSs to obtain the shared contents [1]. Target MS usually uploads the shared content to a Content server in order to minimize the communication overhead [12]. This solution has the inherent limitations of centralised mechanism like single point of failure, and lack of scalability.

A decentralised mobile P2P offers several advantages over a centralised overlay. However, it struggles for efficient solutions from bootstrapping to file distribution and search in mobile environment [5]. Retrieving a large file from a single source is inefficient over GSM-GPRS network. Not only does it suffer from long latency, but also is less reliable. An alternative approach would be to divide a large file into smaller chunks and distributed over many peers.

This paper proposes a mobile P2P over GSM architecture called GMP2P for efficient file distribution and sharing. It integrates the concept of Distributed Hash Table (DHT) with MSs and BTSs in the GSM-GPRS infrastructure. An efficient bootstrapping and a key search mechanism for GMP2P tailored to GSM-GPRS underlay are also proposed.

Rest of the paper is organized as follows. Section 2 describes the related work on mobile P2P networks. The proposed GMP2P architecture for decentralised mobile P2P file sharing is explained in Sect. 3. The analysis and simulation results related to communication costs of GSM and existing mobile P2P over GSM with GMP2P architectures are compared in Sect. 4. Section 5 concludes the work and provides some future directions.

2 Related Work

Mobile P2P has accelerated the growth of traditional P2P overlays. To accommodate mobility of P2P nodes, the overlay network depends on the underlay physical network. Several mobile P2P techniques have been proposed in the past. MadPastry [14] and CMP2P [6], for example, rely on MANET (Mobile Ad-hoc NETwork) as underlay network. P2P decentralized file sharing mechanism through Bluetooth using GSM network have been explored in [9,13].

The main idea, behind all three mechanisms, is to form a closed local P2P group via Bluetooth connections.

WP2P [1] proposes a P2P application over wide area through SMS service. This mechanism covers a large number of mobile users. It uses a centralized web server called mobile server. In response to a search request, the server provides the ID of a target mobile which stores the required contents. The interested mobile user can download the contents by sending an SMS request to the owner of the file. Subsequently, the owner shares the content over a Cloud storage.

P2P-SIP [7] presented a Session Initiation Protocol (SIP) as the underlying signalling protocol. It does not provide the actual transport of information between mobile stations. It just controls the delivery of the shared file information and assists in overlay control messaging. The file downloading requests are sent through SMS as in WP2P [1]. On completion of the session, the downloader sends an acknowledgement to the content provider about a successful downloading.

P2P-Content Distribution [12] mechanism deploys three servers, namely, an Index server, a Control server, and a Content server in the GSM core network. In order to avail the content service, a mobile station (MS) should register itself with the Control server. Contents of all registered MSs are stored in the Content server, and indexed at the Index server. To initiate a search request for a shared file, MS collects the updated real-time state of the Index server, and then downloads the indexes for its own search. A Control server guides the mobile station to upload and download the resources from the Content server. In other words, this mechanism deploys the servers to publish, index and retrieve shared files instead of downloading via expensive GSM service, like SMS.

To the best of our knowledge, all the existing MP2P schemes that operate on the wireless infrastructure network make use of a central server to locate and retrieve the shared files on mobile P2P overlay. This limits the benefits of P2P overlays. Authors in [8] proposed a framework for distributed computing aiming to reduce wireless communications cost while retrieving the file chunks. In this work, we propose an efficient P2P overlay on GSM-GPRS architecture in which no centralised server is used and the chunks of a file are distributed over different MSs to increase storage efficiency as well as reliability.

3 GSM Based Mobile P2P (GMP2P)

The proposed GMP2P overlay integrates DHT (Distributed Hash Table) with GSM underlay infrastructure. The underlay communication could be MS to BTS, BTS to MSC (including BSS) or MSC to MSC (including backbone network) in the GSM core network. However, the proposal is only limited to the availability of a few tables related to DHT information at BTSs.

3.1 Architecture

The GSM infrastructure is expected to contain both DHT and non-DHT components (BTSs and MSs). For convenience in description, we refer to the set of

Table 1. Notations used for analysis and communication cost.

Description	Notation (Max val)
Number of Mobile Stations (MSs) under each BTS	N_{MS} (=1K)
Number of DHT Mobile Stations under each BTS	N_{dMS} (=5%)
Number of BTSs in each region	N_{BTS} (=1K)
Number of dBTSs in each region	N_{dBTS} (=10%)
Total number of regions (each maintained by a different MSC)	N_r (=40)
Total number of file	(=100K)
Average number of chunks per file	(=10)
Total number of file chunks	N_c (=1000K)
Communication cost between pair of BTSs or between BTS and its MSC	C_f
Communication cost one MSC to another	$k.C_f$ ($k = 5$)
Communication cost between MS to BTS	C_w (=$k.C_f$)

BTSs participating in DHT as dBTS and the set of MS that participate in a DHT as dMS. Further, we also use "dBTS" to refer to a nominal member of dBTS set, and similarly "dMS" to refer to a nominal member of dMS set.

Each $dBTS_i \in dBTS$ is assigned a fixed 16-bit gray-code ID according to their relative physical positions. Therefore, it is possible to assign a maximum 65536 members in a dBTS. The neighbours of $dBTS_i$ are assigned gray codes at a hamming distance (HD) one from its own gray code ID. Each $dBTS_i \in dBTS$ maintains a table of 16 neighbouring dBTS ids in its local routing table. Thus, the members of dBTS together form a hypercube overlay topology through gray code ids.

Each $dMS_i \in dMS$ is assigned with a unique 128-bit DHT ID that includes a 16-bit $dBTS_i$ ID as the prefix, which is shown in Fig. 2. The notations of our proposed model are summarised in Table 1.

The rationale behind ID assignments is explained below:

1. An associated dBTS ID forms a prefix of a 128-bit MS ID. Therefore, it is possible to extract the ID of the associated dBTS of an MS from its own ID.
2. The collection of dBTSs are organized in the form of a hypercube. Therefore, by using overlay DHT, it is possible to reach the closest dBTS serving a target MS within at most 16 hops in the underlay network.

3.2 Caching Mechanism

The overlay DHT connectivity is used to guide local as well as global searches of shared files from different dMS_i. In the proposed model, files are stored at different members of a dMS group in a distributed fashion. Each dMS_i holds the chunks of files with key ids which have closest prefix match to the ID of dMS_i.

The corresponding $dBTS_i$ caches the meta data relevant those chunks. The meta data include respective file chunk ID, dMS ID and dBTS ID of each chunk of a file. The cache entries can be updated on insertion of a newly shared file and joining of a new member to the dMS group. A similar update of cache may be required at the time of deletion of a file or when a dMS leaves the group.

Each $dBTS_i$ maintains information about its all associated set of mobile stations. A non-DHT BTS temporarily caches ids of bootstrapping dMS_i, its associated DHT-BTS $dBTS_i$ and its corresponding underlay BTS-ID during the execution of bootstrapping step. In order to track the mobility of a dMS_i, the corresponding BTS or dBTS member updates its cache as dMS_i hops from one cell to another cell. For example, whenever a dMS_i moves away from the cell of its associated $dBTS_i$ to a cell under a new $dBTS_k$, $dBTS_i$ caches the information of $dBTS_k$. Thus, the proposed scheme prevents re-assignment of overlay ids during inter-region movements unlike it is done in either Cluster MP2P [6] or in MadPastry [14].

Apart from dMS cache information, the members of $dBTS$ also cache their closest subset of $dBTS$ for the overlay routing. Each $dBTS_i$ prefers to maintain a set of $dBTS$ member ids at unit hamming distances. In other words, the overlay connectivity of a $dBTS$ group defines a hypercube structure as explained earlier.

3.3 Bootstrapping

When a new node, say MS_{new} wishes to join an overlay, it first downloads GMP2P bootstrap file either from a member of dMS or from a web server. The application code pre-stores an active list of DHT BTSs for bootstrapping like a torrent file. If MS_{new} is fortunate enough to have its associated BTS (BTS_a) in this active list, then MS_{new} selects BTS_a as bootstrapping BTS. Otherwise, MS_{new} selects one dBTS ($dBTS_d$) from the active list and sends a join request (`JReq`) through the BTS it is associated with (say BTS_a). The message exchanges for assignment of overlay ID of a new node is shown with help of a sequence diagram in Fig. 1.

MS_{new} sends the `JReq` message to the current BTS_a with three important fields: (i) m the ID of MS_{new}, (ii) q the overlay ID of BTS_r, i.e., $dBTS_k$, and (iii) underlay ID of BTS r, i.e., BTS_r. If BTS_a is not a DHT BTS, it forwards the request to a $dBTS_k$ (if available in the cache) or broadcast it. In response to the request, $dBTS_k$ may either sends its own overlay ID or the ID of a nearby dBTS (say $dBTS_i$). The response message (m, i, v) mentions mobile ID (m), overlay ID (i) and underlay ID (v) corresponding to the overlay ID i. After receiving the response, MS_{new} computes hash $H(m)$ for its ID, then sends the ID to $dBTS_i$ via BTS_a. BTS_a caches relevant information, before forwarding the message to BTS_v ($dBTS_i$), which would be used later.

3.4 Key Insertion

Since mobile stations are resource poor, large files cannot be stored locally by a single MS. To handle this problem, GMP2P divides each file into small chunks

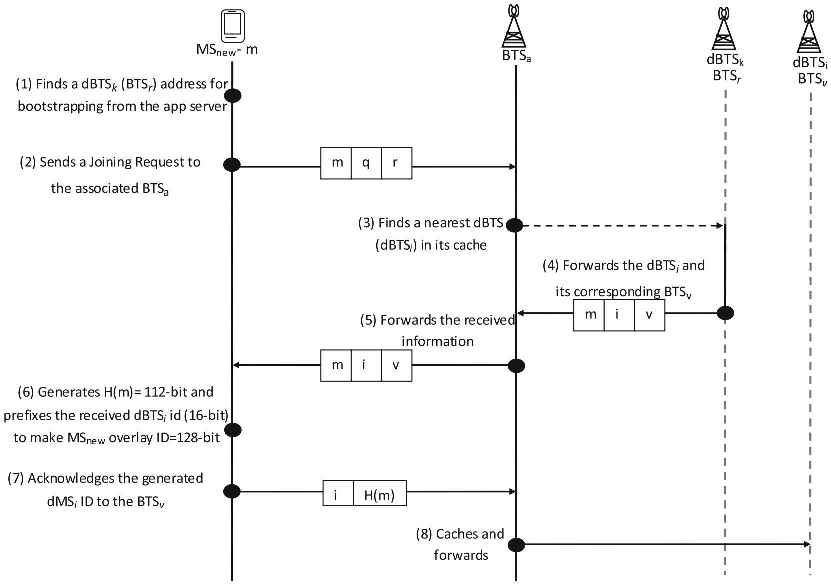

MS$_{new}$- m
BTS$_a$
dBTS$_k$
BTS$_r$
dBTS$_i$
BTS$_v$
(1) Finds a dBTS$_k$ (BTS$_r$) address for bootstrapping from the app server
(2) Sends a Joining Request to the associated BTS$_a$
m q r
(3) Finds a nearest dBTS (dBTS$_i$) in its cache
(4) Forwards the dBTS$_i$ and its corresponding BTS$_v$
m i v
(5) Forwards the received information
m i v
(6) Generates H(m)= 112-bit and prefixes the received dBTS$_i$ id (16-bit) to make MS$_{new}$ overlay ID=128-bit
(7) Acknowledges the generated dMS$_i$ ID to the BTS$_v$
i H(m)
(8) Caches and forwards

Fig. 1. Sequence diagram for joining of new MS.

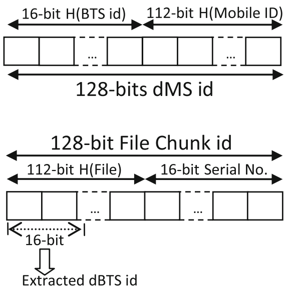


Fig. 2. DHT ID for MS and BTS under GMP2P.

of equal sizes. Each chunk of a file is assigned a unique key comprising of three fields: (i) a 112-bit hash value of the file (ii) an 8-bit file chunk serial number and (iii) an 8-bit total number of chunks comprising the file. Thus, each chunk of a file has a 128-bit random key ID that includes a 112-bit prefix representing the hashed value of the file name.

Each key is inserted to the dMS member of the $dBTS$ having an overlay ID that has the closest prefix match with the key. The insertion process begins with extraction of two left most octets (16-bit) from the key ID. These two octets refers to $dBTS_i$ ID for a target dMS_i where the key gets inserted. The procedure for forming overlay ids is illustrated by Fig. 2. The insertion process ensures that all the chunks of a file are inserted into DHT-MS members clustered under a same $dBTS_i$. As we will see later, this approach to insertion process reduces search cost for downloading files.

3.5 Key Search

Keys are searched through DHT based prefix matching technique. When dMS_i wishes to download a file, it hashes the desired file to obtain the 112-bit digest and sends the search request with the (128-bit) key as (112-bit digest padded right with 16 zero bits).

The left most 16-bits determine the $dBTS$ (say $dBTS_i$), which is responsible for storing that file. dMS_i forwards the search request to $dBTS_i$ through its associated BTS (BTS_a). Then $dBTS_i$ forwards the query to the mobile station dMS_j which has the closest prefix match with the key being searched for. This mobile station is responsible for that key.

If BTS_a is not under the same MSC as that of $dBTS_i$, BTS_a forwards the search request to the appropriate MSC under which $dBTS_i$ is located. The MSC performs an underlay search to find the underlay BTS ID of $dBTS_i$. From its routing table $dBTS_i$ selects another member of $dBTS$ which has a prefix closer to the target $dBTS$, and sends the request to find the selected $dBTS$ through corresponding MSC. The forwarding process is repeated at a new $dBTS$ member, until the $dBTS$ with the closest prefix is either successful in locating the key or becomes unsuccessful. Finally, the closest $dBTS$ member, say, $dBTS_j$ sends reply to the route request after fetching the object from associated dMS_j. Algorithm 1 describes the search process of the proposed GMP2P file sharing model. This search mechanism uses the closest prefix match based routing to

ALGORITHM 1. (Key Searching in GMP2P.)

```
dMS$_a$ requests a key to associate BTS$_a$.
if ($BTS_a \in dBTS$) then
    Finds a closest prefix of dBTS ($\approx dBTS_i$) in its table
    if (found) then
        Replies the key.
    end
    else
        repeat
            Forwards the request to dBTS ($\approx dBTS_i$)
        until ($TargetdBTS_j \not\approx dBTS_i$);
        Replies the key.
    end
end
else
    // for Non DHT-BTS
    if ($BTS_a.tempCache == empty$) then
        Broadcasts to find a nearest $dBTS_i$
    end
    repeat
        Forwards the request to dBTS ($\approx dBTS_i$)
    until ($TargetdBTS_j \not\approx dBTS_i$);
    Replies the key.
end
```

find the $\approx dBTS_i$, where $\approx$ denotes the closest prefix ID corresponding to the searching key. This mechanism is similar that one used in Pastry [10].

4 Analysis and Simulation Results

To analyse the communication cost in retrieving shared files over the proposed GMP2P, the communication structure in the design of GMP2P system model is described precisely.

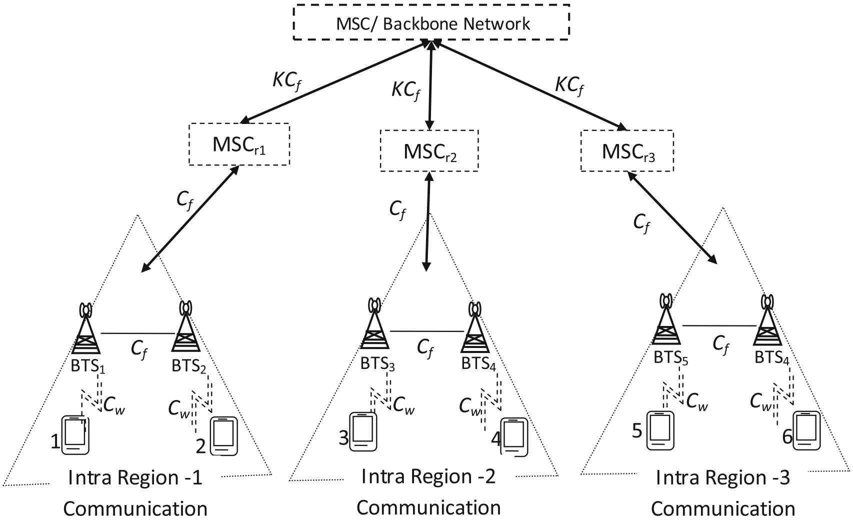


Fig. 3. Communication cost in GSM network.

4.1 Communication Structure

Underlay communication is performed point to point through wireless and fixed stations based on GSM. In GSM, the cost of sending a message between a wireless station its associated BTS is significantly higher than that of sending a message between two hosts belonging to BSS, MSC or backbone network [2]. The communication cost of routing between different stations of the underlay GSM is depicted in Fig. 3. The communication between stations located in the same region is referred as intra-region communication in our model while the communication among MSCs (including backbone network) is inter-region communication. It is known that wireless communication cost (C_w) from MS to BTS or vice versa is higher than fixed communication cost (C_f) from BTS to BTS (or MSC) of the same region. GMP2P assumes that the communication cost between MSC to MSC over wired network (included backbone communication) is k times higher than C_f regardless of their geographical distance, where k is a constant value. The notations used for GSM network specifications are depicted in Table 1.

Communication Cost. For searching a file key, GSM network has to broadcast the search request for the key. Hence, total communication (TC) cost for searching a key within GSM can be analysed in two parts: intra-region and inter-region communication costs. The intra-region communication cost is $2C_w(N_{MS}-1)$ and inter-region communication cost is $2C_f+2KC_f(N_r-1)$ for downloading a single file. Assuming a file has N_c number of chunks, we need to add the communication cost for all N_c keys searching. Hence, the total cost in GSM would be:

$$TC_{gsm} = N_c(2C_w(N_{MS}-1) + 2C_f + 2KC_f(N_r-1)) \tag{1}$$

where the total number of file chunks and the number of regions are denoted as N_c and N_r respectively.

In case of multiple files with multiple chunks, GMP2P divides communication cost for searching and downloading into intra region (x% of files chunks) and inter region (remaining [100-x]%). Hence, the total intra region (x% of N_c) communication cost for proposed model is:

$$\begin{aligned} Intra\_region\_cost = 4C_w + C_f log(N_{dbts}-1) \\ + 7C_f + 2N_c(x/100)C_w \end{aligned} \tag{2}$$

The total inter region ([100-x]% of N_c) communication cost for proposed model is:

$$\begin{aligned} Inter\_region\_cost = 5C_w + 4N_c((1-x)/100))C_w \\ + (N_r-1)C_f log(N_{dbts}-1) + (N_r+4)kC_f + 7C_f, \end{aligned} \tag{3}$$

4.2 Simulation Result

Our analysis is based on intra and inter region communication costs for searching the distributed file chunks. We assumed 50 K chunks of different files are distributed over different regions. The simulation is carried out using numerical values from Table 1 over MATLAB. The first result (shown in Fig. 4) confirms that file sharing application is not suitable for traditional GSM networks. To find a file inside an MS, GSM has to broadcast the search request to all. Therefore, the searching cost rises exponentially as shown in the plot. On the other hand, after integrating DHT with GSM-GPRS network as proposed in GMP2P approach, the performance improves significantly. For the comparison of efficiency of the proposed model, we selected three other existing mobile P2P mechanisms. These are P2P-CD [12], WP2P [1] and P2P-SIP [7]. In P2P-SIP, an acknowledgement scheme has also been implemented. Therefore, we also added acknowledgement to GMP2P with extra overhead of sending acknowledgements after successful downloading of meta data, we call this implementation as GMP2P(Ack). Initially, the comparisons are performed on the basis of Eqs. 2 and 3 for file distribution (as shown in Fig. 5). It is observed that the proposed mechanism performs better than all three existing mechanism. We also observed the scalability effects by increasing number of regions. Figure 6 shows that our system is scalable and performance does not degrade significantly with increase in the number of network regions. Existing mechanisms are not able to sustain good performance in search and download costs in the case when file chunks are scattered over a large number of regions. Some interesting but obvious outcomes are presented in Figs. 7 and 8 where we analysed the only communication overheads for searching of a single file. As Fig. 7 shows, for a big file with all the chunks located in one region, the communication cost in our model is the least. Furthermore, Fig. 8 shows that GMP2P substantially outperforms when file chunks are distributed over different regions.

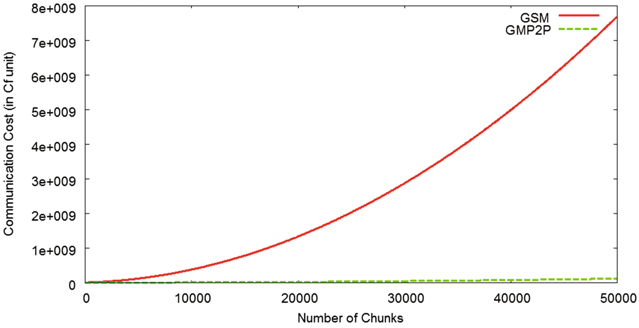


Fig. 4. Cost comparisons of searching files over plain GSM and GMP2P.

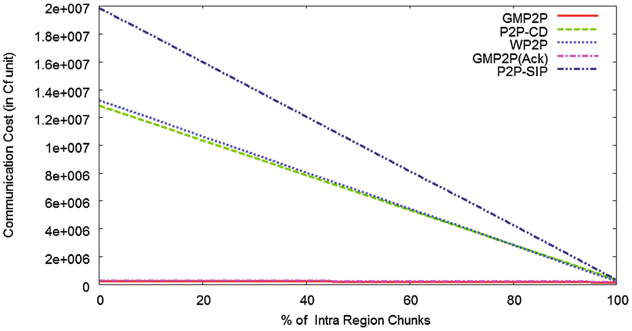


Fig. 5. Cost comparisons of search & download of meta data with different chunk distribution patterns.

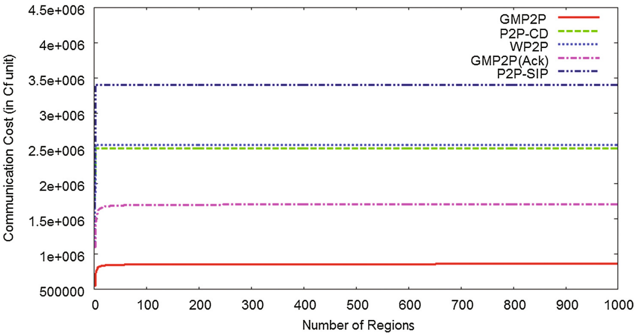


Fig. 6. Cost comparisons with increase in spread of chunk distribution.

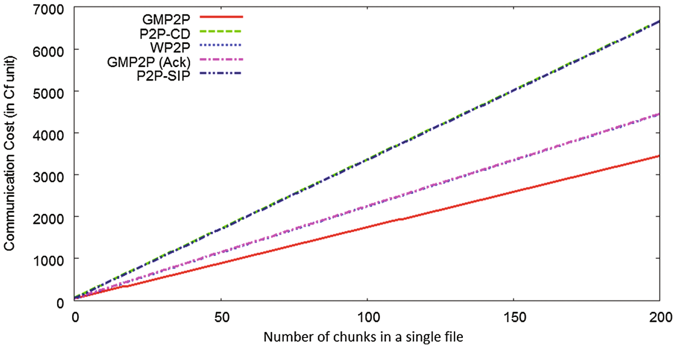


Fig. 7. Cost comparisons for searching chunks of a file located in the same network region.

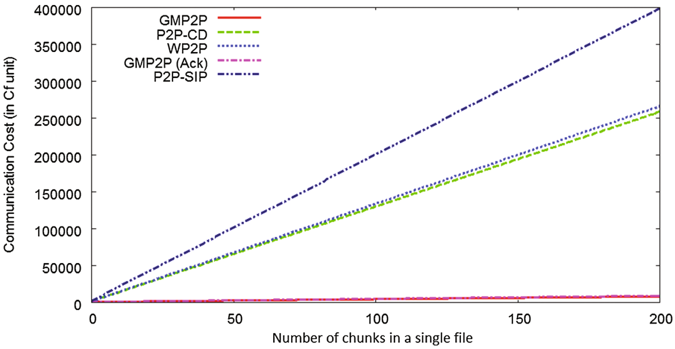


Fig. 8. Cost comparisons for searching chunks of a file scattered over different network region.

4.3 Implementation

We used Java 2 Micro Edition (J2ME) platform using Mobile Information Device Profile (MIDlet) to implement GMP2P Framework as showed in Fig. 9. The implementation performs with P2PGSM framework which includes four packages namely, overlay, underlay, domain and util. The architecture of this framework comprises with the following four packages.

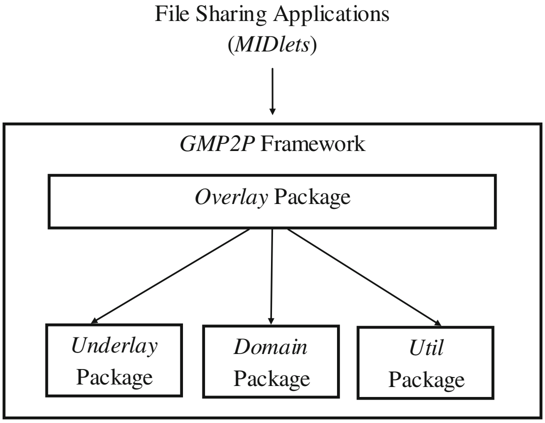


Fig. 9. Overall interaction of GMP2P framework and MIDlets in J2ME.

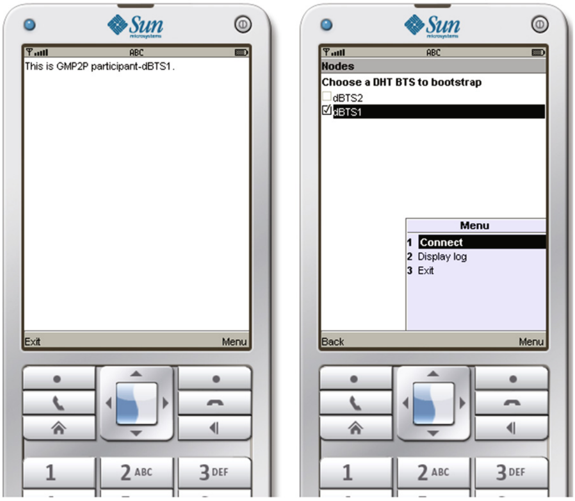
Overlay. This package contains the OverlayNetwork class which is the core of GMP2P overlay construction and maintenance. All communication between the overlay application (such as, MIDletGMP2P) and the overlay network are performed using this class.

Underlay. The underlay package keeps the classes related to all GSM-GPRS network communication. This communication uses datagram service of J2ME framework to establish a connection between MS to BTS. In this underlay network, an MS can communicate through BTS only.

Domain. This package maintains different classes, which help to create conceptual domain of GMP2P application. These conceptual domain represent a real time objects like mobile nodes, intra region and inter region peer group, etc.

Util. This package maintains different useful log information regarding a GMP2P MIDlet application. These information helps to other classes of the GMP2P framework to perform certain activities regarding the same GMP2P application.

This framework is able to create and connect different *dMS* of GMP2P. Figure 10 shows an initial set up of our GMP2P file sharing application. We assume that some selected mobile stations under P2PGSM framework are working as *dBTS*. In this framework, each *dBTS* MIDlet is enabled to cache the information of the associated mobile stations.


This is GMP2P participant-dBTS1.

Nodes

Choose a DHT BTS to bootstrap

□ dBTS2

☑ dBTS1

Fig. 10. Bootstrapping in GMP2P Framework.

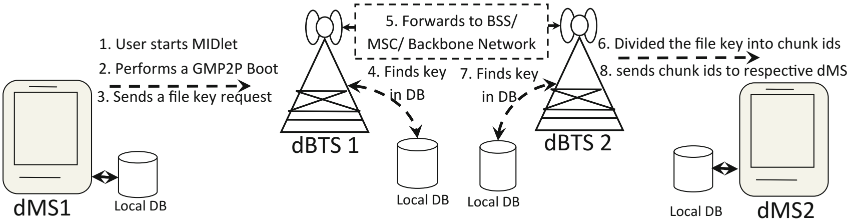


Fig. 11. Information Flow between dMS and dBTS in GMP2P scheme.

Information flow between dMS and dBTS through GMP2P MIDlet is shown in Fig. 11. GMP2P MIDlet uses wireless communication channel for MS to BTS and fixed communication channel for BTS to BTS (or backbone network). Each dBTS maintains a local database of shared files' meta data. These meta data are maintained through neighbouring *dBTS* periodically.

5 Conclusion

Implementing file sharing in wireless network is interesting, but has many challenges. In this paper, we proposed a mobile P2P architecture called GMP2P by integrating DHT based P2P overlay with the infrastructure wireless network like GSM. We also evolved Bootstrapping, Inserting, and Searching mechanisms for GMP2P. Analytical results confirms that file sharing through the proposed scheme is more efficient than file sharing through other existing mobile P2P systems. The large file can be fragmented and stored at the peer members unlike other mobile P2P system where files are stored at centralised web servers. So, many features like security, privacy and fault tolerance, can be integrated at user levels. Accessing large files is faster in our proposed scheme because each peer has to share only a small chunk of file. Furthermore, a file uploading is shared distributively by many peers. The Bootstrapping mechanism of GMP2P is implemented in J2ME platform while implementation of other components are in progress.

Acknowledgement. This work is funded by the E-security Division, Ministry of Electronics and Information Technology, Government of India, through project grant number 12(7)/2015-ESD.

References

1. Abiona, O.O., Oluwaranti, A.I., Anjali, T., Onime, C.E., Popoola, E., Aderounmu, G.A., Oluwatope, A.O., Kehinde, L.O.: Architectural model for wireless peer-to-peer (wp2p) file sharing for ubiquitous mobile devices. In: IEEE International Conference on Electro/Information Technology, 2009, eit 2009, pp. 35–39. IEEE (2009)
2. Aggelou, G.N., Tafazolli, R.: On the relaying capability of next-generation gsm cellular networks. IEEE Pers. Commun. **8**(1), 40–47 (2001)
3. Cerwall, P., Jonsson, P., Möller, R., Bävertoft, S., Carson, S., Godor, I., Kersch, P., Kälvemark, A., Lemne, G., Lindberg, P.: Ericsson mobility report. Ericsson (2015). www.ericsson.com/res/docs/2015/ericsson-mobility-report-june-2015.pdf [June 2015]
4. Chang, R.S., Chang, J.S.: Adaptable replica consistency service for data grids. In: Third International Conference on Information Technology: New Generations (ITNG 2006), pp. 646–651. IEEE (2006)
5. Dıaz, A., Merino, P., Panizo, L., Recio, A.: A survey on mobile peer-to-peer technology. In: Proceedings XV Conference on Concurrency and Distributed Systems (JCSD07), pp. 59–68 (2007)
6. Gottron, C., Konig, A., Steinmetz, R.: A cluster-based locality-aware mobile peer-to-peer architecture. In: IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012, pp. 643–648, March 2012
7. Li, L., Wang, X.: P2p file-sharing application on mobile phones based on sip. In: 4th International Conference on Innovations in Information Technology, 2007, IIT 2007, pp. 601–605. IEEE (2007)
8. Li, S., Yu, Q., Maddah-Ali, M.A., Avestimehr, A.S.: Edge-facilitated wireless distributed computing. In: IEEE GLOBECOM, December 2016
9. Paiva, S.: An intuitive information share system for mobile devices. In: IEEE 18th International Conference on Computational Science and Engineering (CSE), 2015, pp. 314–318. IEEE (2015)
10. Rowstron, A., Druschel, P.: Pastry-scalable, decentralized object location and routing for large-scale peer-to-peer systems. In: Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), pp. 329–350 (2001)
11. Shen, X.S., Yu, H., Buford, J., Akon, M.: Handbook of Peer-to-peer Networking, vol. 34. Springer, New York (2010)

12. Sun, Y., Li, Y., Wen, X., Zhao, Z.: Mobile p2p content distribution in wireless networks environment. In: International Conference on E-Business and E-Government (ICEE), 2010, pp. 1656–1659. IEEE (2010)
13. Wu, R., Cao, Y., Liu, C.H., Hui, P., Li, L., Liu, E.: Exploring passenger dynamics and connectivities in beijing underground via bluetooth networks. In: 2012 IEEE, Wireless Communications and Networking Conference Workshops (WCNCW), pp. 208–213. IEEE (2012)
14. Zahn, T., Schiller, J.: MADPastry: A DHT substrate for practicably sized MANETs. In: Proceedings of 5th Workshop on Applications and Services in Wireless Networks (ASWN2005), Paris, France, June 2005

Simulation of Network Growth Using Community Discovery in Biological Networks

Y. Divya Brahmani, T. Sobha Rani(✉), and S. Durga Bhavani

School of Computer and Information Sciences, University of Hyderabad,
Hyderabad, India
tsrcs@uohyd.ernet.in

Abstract. Interactions between proteins in a cell can be modeled as a graphical network. The problem addressed in this paper is to model the network evolution in biological networks in order to understand the underlying mechanism that morphs a normal cell into a disease (cancer) cell. In this paper, concepts from social networks are utilized for this purpose. Though many models for network evolution exist in the literature, they have not been applied in the context of evolution of normal cell into a disease state. In this work, target network is evolved in two ways: (i) starting from common subgraph of the normal and cancer networks and (ii) using a divide and conquer approach, the network is grown from communities using preferential attachment models. Triadic model yields good performance with respect to the global characteristics, but actual edge prediction performance is very low when applied on the entire network. In the case of community approach, the results of edge prediction for two dense communities are satisfactory with precision of 62% and recall 62%. Since edge prediction is a challenging problem, the approach needs to be refined further so that it works for small and sparse communities as well before it can become a full-fledged algorithm.

1 Introduction

A biological network is a representation of interactions within a biological system. When a normal person is affected by cancer, abnormal cell growth occurs in his/her body. Network analysis with respect to human diseases helps in understanding basic mechanisms responsible for the cellular processes, and disease pathologies. In biological networks, the process of the growth of network is not known. To analyze the growth of biological networks, network evolution models applied in social networks can be used.

Barabasi and Albert [2] proposed the preferential attachment model which generates networks with power law degree distribution that is observed in many social and biological networks. This model is extended to include different kinds of Preferential attachement rules [16]. More recently, Barabasi et al. laid down hypotheses connecting network properties like hubs, modules, shortest paths to human disease [5]. Currently, most network analysis work is being carried out for Alzheimer's disease (AD). In [9], it is shown that attack of disease disturbs

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 232–239, 2018.
https://doi.org/10.1007/978-3-319-72344-0_19

the hub like structures in the normal network. These topological changes are shown to be siginificant as compared to random null models. Rahman et al. [14] worked on cancer networks of different tissues and made a comparative study of these networks with respect to differential expression. This group extended the study further [1,6] by obtaining communities that show differential modularity between normal and cancer networks. Sahoo et al. [15] consider the common subgraph between normal and cancer networks, and through bipartite graph analysis show a significant difference in connectivity between normal and cancer networks. Can we understand how the network changes when transitioning from normal to disease state? In this work, the study is extended to model the network evolution and possible changes that occur in cancer network during growth from the normal network.

The main contributions of the paper are the following. Differentiating cancer from normal using community discovery algorithms has not been attempted so far. In this work, popular network growth models using preferential attachment like triadic closure and Price model [13,16] are applied to differentiate growth in normal network to cancer network. This growth has been modeled in two ways: by growing the communities locally and constructing the whole network combining these communities and constructing the whole network using global growth models.

1.1 Data Set

Rahman et al. [14] construct a protein-protein interaction (PPI) network with protein molecules involved in ten major cancer signal transduction pathways. The expression values of the nodes are given along with the presence/absence of the edge information. We consider weighted graphs by defining the weight as sum of the expression values of the end points. Only the pair of proteins having assigned value 1 for both the proteins are considered to have a valid interaction. Rahman et al. construct networks for 5 types of cells: Bone, Breast, Liver, Colon and Kidney of which the experimentation is carried out on Bone cancer data. Bone cell consists of 192 nodes and 619 interactions in the normal state and 351 nodes and 1783 interactions in the cancerous state.

2 Differentiating the Normal from Cancer States

2.1 Community Discovery

A community is a large or small group of units which have some common property [10]. To discover communities, a few community discovery algorithms are applied on this biological network. They are Fast greedy [4] and Multi Level [3] community discovery algorithms implemented in R package [8].

3 Approach to Network Evolution

Network evolution methods like Preferential attachement methods [16] are used to grow the biological networks. Community discovery algorithms [10] used in social networks are used here to discover communities in biological networks. Instead of growing entire network, communities are grown individually to find which process models the evolution better.

3.1 Network Evolution in Communities

Preferential attachment models are used to generate the bilological networks [2,16]. Global network growth has been carried out using Triadic attachment model from the common subgraph of cancer and normal networks. Eventhough the scale-free property expressed as a power law exponent obtained is satisfactory, precision and recall values are not upto the mark.

A divide and conquer approach is proposed based on community discovery to improve the edge prediction performance. As a first step, members of the communities are discovered and then the local interactions among members are predicted using preferential attachment algorithms. Communities discovered in PPI networks using Fast greedy algorithm and Multi level community algorithms are taken as ground truth. Here the network is considered as undirected.

LEMON (Local Expansion via Minimum One Norm) [7] algorithm is applied to discover the communities in cancer network by taking a seed set from the ground truth communities. LEMON algorithm produces overlapping communities. This algorithm does not provide the network per se. Output of LEMON algorithm is the set of nodes in a community grown from the given seed set, and F1 score between ground truth community and resultant community grown from seed set. Correct seed set has to be selected to get good F1 score. Network induced by 50% of the nodes in the community generated by the LEMON algorithm is taken as the initial network. Entire community is grown from the network (initial network) formed from these initial nodes. In order to connect these nodes to predict the underlying network, Triadic model [16] is used.

In Triadic model, new nodes get attached to m previous nodes chosen at random with a probability proportional to the degree of the previous nodes and also to m_{sec} secondary contacts (neighbors) of the m previous nodes. This is a connection mechanism based on the characteristic "friend of a friend is a friend". Triadic model has the time complexity of O($m{\cdot}m_{sec}$) for each new node added. Number of connections a node receives could increase with the number of connections it already has.

3.2 Implementation

Since the ground truth is not known, one of the popular community discovery algorithms, namely, Fast Greedy algorithm is used to discover the communities. Total cancer network is divided into 9 communities using Fast greedy algorithm. Out of them only 6 communities are big. LEMON algorithm is applied on those

6 communities. This algorithm returns the communities with highest F1 score. Here, the seed set selected is in such a way that it contains high degree node, low degree node and average degree nodes to obtain a good F1 score. Table 1 gives the details about the communities discovered by LEMON algorithm. Network formed with first nodes in list is taken as initial network to apply the Triadic model.

Table 1. Number of nodes in LEMON produced communities

Community number	Number of nodes in fast greedy	Number of nodes in LEMON	Percentage of true positives in LEMON communities
Community 1	21	21	80.95
Community 2	82	81	76.07
Community 3	76	81	63.69
Community 4	75	81	60.25
Community 5	37	81	54.62
Community 6	45	81	57.14

3.3 Network Growth Within a Community

Triadic model is applied on each of these 6 communities taking initial nodes (approximately 50% of nodes generated by the LEMON algorithm) as the nodes in initial network. list of nodes in the community and F1 score between ground truth community and resultant community grow from seed set. Community with good F1 score is taken to apply Triadic model. Using Triadic algorithm entire community will be grown.

Community 2: LEMON produced community 2 having 81 nodes and 611 edges. Initial network is considered with 41 nodes with 325 edges. Triadic model starts from initial network. Let the number of high degree nodes to be selected be m, the number of secondary contacts to be selected be m_{sec}. Degree is normalized so that we can easily select nodes with high degree, let this normalized degree be t. Triadic model is applied with different parameter settings for m, t and m_{sec}. Parameters for which results showing good output are presented in Table 2. As the number of false positives is more in the resultant network, a standard procedure of random deletion of edges is carried out on the resultant network to improve Precision and Recall values. Time complexity of random edge deletion is O(n), where n is number of edges to be deleted.

Community 4: LEMON produced community 4 having 81 nodes and 354 edges. Initial network has 40 nodes with 168 edges. Results obtained when Triadic model is applied with different parameter settings m, t and m_{sec} are given in

Table 2. Growth of community 2 network using triadic model. Power law exponent and RMSE values.

Experiments with different parameter settings	Exponent	RMSE
Original	3.74	-
m=2 t=0.55 msec=2	3.57	3.13
m=2 t=0.5 msec=2	3.56	2.58
m=2 t=0.65 msec=2	3.60	3.10
m=2 t=0.65 msec=3	3.94	2.53
m=2 t=0.65 msec=4	3.67	2.6
m=2 t=0.6 msec=2	3.89	3.19
m=2 t=0.6 msec=3	3.73	2.09
m=2 t=0.6 msec=3	3.77	2.27
m=3 t=0.55 msec=2	3.99	2.24
m=3 t=0.55 msec=3	4.11	2.15
m=3 t=0.65 msec=2	3.51	2.53

Table 3. Number of predicted edges, Precison, Recall and F-measure, Exponent and RMSE for resultant community 2 network after random edge deletion

Experiments with different parameter settings	Predicted edges	Precision	Recall	F-measure	Exponent	RMSE
m=2 t=0.5 msec=2	357	0.58	0.58	0.58	3.46	2.87
m=2 t=0.6 msec=2	378	**0.62**	**0.628**	**0.618**	3.89	3.14
m=2 t=0.65 msec=2	368	0.60	0.602	0.602	3.46	2.87
m=2 t=0.7 msec=2	365	0.597	0.597	0.59	3.66	2.35
m=2 t=0.8 msec=2	368	0.602	0.602	0.602	3.44	2.21
m=2 t=0.7 msec=3	352	0.576	0.576	0.576	3.41	2.21

the Table 4. To reduce the number of false positives in resultant network, random edge deletion is done on resultant network to increase Precision and Recall values.

Construction of the Whole Network Using Local Approach: The clusters are grown separately using LEMON and Triadic model in cancer network. Combining the clusters with highest precision value whole network is formed. But there are only 264 nodes in this network. So, there are 87(351-264) nodes remaining outside the 6 clusters. These nodes are added to the network using Triadic model. These are the results for total network using Lemon and Triadic model (Table 6).

It could be seen from the results that out of all the six communities, good Precision, Recall and F-measure values are obtained for communities 2 and 4.

Table 4. Growth to community 4 network using triadic model. Power law exponent and RMSE values.

Experiments with different parameter settings	Exponent	RMSE
Original	3.086889	-
m=2 t=0.55 msec=2	3.29	3.31
m=2 t=0.55 msec=3	3.14	4.08
m=2 t=0.55 msec=4	3.07	3.28
m=2 t=0.5 msec=2	3.18	4.02
m=2 t=0.6 msec=2	3.28	3.71
m=2 t=0.6 msec=3	3.54	3.90
m=2 t=0.6 msec=4	3.37	3.28
m=2 t=0.75 msec=2	3.08	3.94
m=3 t=0.55 msec=2	3.28	3.33
m=3 t=0.5 msec=2	3.21	3.59
m=3 t=0.5 msec=3	3.141	3.95

Table 5. Number of predicted edges, Precison, Recall and F-measure, Exponent and RMSE for resultant community 4 network after random edge deletion

Experiments with different parameter settings	Predicted edges	Precision	Recall	F-measure	Exponent	RMSE
m=2 t=0.5 msec=2	163	0.445	0.446	0.445	3.33	4.18
m=2 t=0.55 msec=2	158	0.43	0.432	0.432	3.85	3.68
m=2 t=0.65 msec=2	169	0.462	0.463	0.462	3.06	3.26
m=2 t=0.75 msec=2	174	0.477	0.476	0.4762	3.6	3.64

Table 6. Construction of the whole network.

	Parameters	Exponent	RMSE	Number of predicted edges	Precision	Recall	F-measure
Before deletion of edges	m=1, $msec$=1, t=0.5	2.014	9.55	835	0.39	0.47	0.42
After deletion of edges	m = 1, $msec$=1, t=0.2	1.910	7	714	0.4	0.40	0.40

A good power law exponent and lesser RMSE values have been obtained in every community after random removal of edges. These two communities are dense communities. Community 2 is more dense than all the communities, so good results for Precision, Recall and F-measure values for m=2, t=0.6 and msec=2 have been obtained. So, this method seems to work for dense rather than sparse communities.

4 Conclusions

Biological network analysis is necessary, since it can provide insights into the underlying processes responsible for various biological manifestations. Especially in the case of transition from a normal state to a disease state, the topological analysis together with social network concepts can reveal important modules that are instrumental in causing the changes in the network topology. Evolving a specific network, is a challenging problem since predicting a connection between a particular pair of proteins also requires domain knowledge. This problem specifically known as "link prediction" in social network analysis is tackled using machine learning approach with complex features [11]. In this work, no domain specific features for proteins have been used and the network is evolved using the Triadic model. The global characteristics of the network have been predicted satisfactorily. The actual link prediction within one community is obtained with 60% accuracy. The algorithm needs to be refined so that link prediction can be improved over all communities.

Acknowledgement. The authors would like to acknowledge the support provided by UPE2, University of Hyderabad to carry out this work.

References

1. Banik, R.S., Rahman, M.D., Rahman, K.M.T., Islam, M.F., Enayetul, S.E.: Comparison of molecular signatures in large-scale protein interaction networks in normal and cancer conditions of brain, cervix, lung, ovary and prostate. Biomed. Res. Ther. **3**(4), 605–615 (2016)
2. Barabási, A.L., Gulbahce, N., Loscalzo, J.: Network medicine: a network-based approach to human disease. Nat. Rev. Genet. **12**(1), 56–68 (2011)
3. Blondel, V., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. **10**, 10008 (2008)
4. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70**, 066111 (2004)
5. Goh, K., Cusick, M.E., Valle, D., Childs, B., Vidal, M., Barabási, A.L.: The human disease network. PNAS **104**(21), 8685–8690 (2007). https://doi.org/10.1073/pnas.0701361104
6. Islam, M.F., Hoque, M.M., Banik, R.S., Roy, S., Sumi, S.S., Nazmul Hassan, F.M., Tomal, M.T.S., Ullah, A., Rahman, K.M.T.: Comparative analysis of differential network modularity in tissue specific normal and cancer protein interaction networks. J. Clin. Bioinform. **3**, 19 (2013)
7. Li, Y., He, K., Bindel, D., Hopcroft, J.: Uncovering the small community structure in large networks: a local spectral approach(2015)
8. https://www.rstudio.com/
9. Li, W., Wang, M., Zhu, W., Qin, Y., Huang, Y., Chena, X.: Simulating the evolution of functional brain networks in alzheimer's disease: exploring disease dynamics from the perspective of global activity. Sci. Rep. **6**, 34156 (2016)
10. Newman, M.E.J.: Networks: An Introduction. Oxford University Press Inc., New York (2010)

11. Davis, D.A., Lichtenwalter, R., Chawla, N.V.: Supervised methods for multi-relational link prediction. Soc. Netw. Anal. Min. **3**(2), 127–141 (2013)
12. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: Proceedings of 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146 (2003)
13. de Price, D.J.S.: A general theory of bibliometric and other cumulative advantage processes. J. Am. Soc. Inform. Sci. **27**(5), 292–306 (1976). https://doi.org/10.1002/asi.4630270505
14. Rahman, K.M.T., Islam, T., Banik, M.F., Honi, R.S., Diba, U., Sumi, F.S., Kabir, S.S., Tamim, S.: Changes in protein interaction networks between normal and cancer conditions: total chaos or ordered disorder? Netw. Biol. **3**(1), 15–28 (2013)
15. Sahoo, R., Sobha Rani, T., Durga Bhavani, S.: A network analysis perspective to differentiate cancer and normal networks, chap. 17. In: Tran, Q.-N., Arabnia, H.R. (eds.) Emerging Trends in Computational Biology, Bioinformatics, and Systems Biology - Systems & Applications. Morgan Kauffman (2016)
16. Toivonen, R., Kovanena, L., Kivela, M., Onnela, J.P: A comparative study of social network models: network evolution models and nodal attribute models, socnet (2009)
17. http://www.genecards.org/

A Style Sheets Based Approach for Semantic Transformation of Web Pages

Gollapudi V. R. J. Sai Prasad(✉), Venkatesh Choppella,
and Sridhar Chimalakonda

Department of Computer Science & Engineering,
Indian Institute of Information Technology Tirupati,
Tirupati, Andhra Pradesh, India
saigollapudi1@gmail.com, venkatesh.choppella@iiit.ac.in, ch@iittp.ac.in

Abstract. The goal of this paper is to propose a style sheet based approach for enabling semantic transformations of existing, already published web pages. Traditionally, web page transformations were largely driven by approaches such as XSLT that focuses on XML documents, and CSS that transforms the style of HTML content. However, despite their wide usage, XSLT is considered as too complex and rigid while CSS only focuses on form and the aesthetics of display. To address this major concern, we propose a new type of style sheet that is (1) applicable on existing, published web content, (2) able to perform semantic transformations, and (3) able to do some client-side processing of published web content. We present the design of the prototype and demonstrate the idea of using semantic style sheets by delivering a set of multiple transformations of a random web page from NASA website.

Keywords: Style sheet · Semantic transformation
Web page transformation · Web accessibility · Client side modification

1 Introduction

The goal of this paper is to propose a style sheet based approach for enabling semantic transformations of existing, already published web pages. This is motivated by our earlier work [19–21] in Renarration of web content. The approach of using style sheets for enabling semantic transformation of web pages is novel and it contributes to the larger work of Web Accessibility.

1.1 Background and Motivation

While the word *renarration* is uncommon, the concept itself is fairly prevalent and straightforward. For instance, as humans, whenever we are socially communicating an idea to somebody, we ensure that it is expressed and delivered in a way that is suitable to our audience. The idea in our head may be one, but its articulations may be many. For instance, a technical idea may be presented

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 240–255, 2018.
https://doi.org/10.1007/978-3-319-72344-0_20

in a scholarly way to a researcher, but the same idea may be re-narrated in a more simplistic manner to a layperson. This variation in the content, delivery and expression ensures that we make our idea more sensible to our audience.

Examples of renarration of written content may also be found in such day-to-day things as commercial documents and academic literature. In teaching and learning of mathematics and sciences the variant versions of the original are labeled as MERs or Multiple External Representations [27]. We frequently use a diagram, a graph or a table to better illustrate a point that we may have already made elsewhere in text [22].

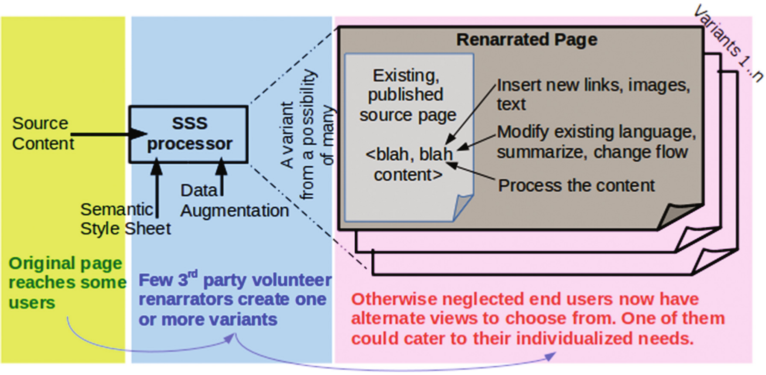

Source Content
SSS processor
Semantic Style Sheet
Data Augmentation
A variant from a possibility of many
Renarrated Page
Existing, published source page
<blah, blah content>
Insert new links, images, text
Modify existing language, summarize, change flow
Process the content
Variants 1..n
Original page reaches some users
Few 3rd party volunteer renarrators create one or more variants
Otherwise neglected end users now have alternate views to choose from. One of them could cater to their individualized needs.

Fig. 1. A web page may be said to have a scope and reach which may exclude some end-users with different needs. 3rd party volunteer renarrators can include them by creating alternative views.

1.2 Renarrating the Web

We are interested in taking this concept of renarration and applying it to existing, published web page content. See Fig. 1. Current material on the web, though prolific with lot of good information, it is not in itself equally accessible by all. Few segments of the user population, for instance, those with poor English skills or those with different or maybe even lower-order thinking skills, or those coming from a different socio-economic or cultural context may find some of the existing material foreign and incomprehensible [19]. For these minority groups, renarrating the original into a more simplified variant, or renarrating the original in a different language, or renarrating the source to have more diagrams and references may prove useful.

Simple renarrations of web pages could potentially include one or more of the following:

- **Augmenting:** adding more diagrams, links, videos; adding locally relevant examples, adding local information etc.
- **Modifying:** summarizing a complex piece of text, translating something into vernacular, removing some clutter, changing language etc.

- **Processing:** changing the numbering system from one representation (e.g. Arabic) to another (e.g. British), computing forex, changing the formats of dates, changing the representation of units from Kilograms to Pounds, or from Kilometers to Miles etc.

Dinesh et al. [7] talk about the difficulties that laborers may face in interpreting the online contents of a government labor law document. Renarrating it into some non-legalese language may help even some common users better interpret it. They also cite the example of a fire safety website being renarrated to include local fire safety standards and contacts.

1.3 Web Accessibility Problem

According to internetlivestats.com[1], a site dedicated to web analytics, there are over 1.2 billion websites and nearly 3.7 billion Internet users in the world today. Of this user-base, 48.4% are from Asia and 9.8% are from Africa. According to another web analytics site – W3Tech.com[2], English is the most dominant language on the web. That is, there are over 51.3% English websites out there on the web today. This is despite the fact that 1.5 billion users are from a non-English background.

In [21] we had already highlighted the Web Accessibility issue of these non-native English speakers. In a study of N = 372 college students [23] we observe that, just by adding instructions in local vernacular would help in improving accessibility. That is, just by adding guidance, by adding additional links to more locally relevant information, by giving tips & tricks and by giving more examples in vernacular to an already published web page, one could help these non-native English speakers make more sense out of that existing English content.

In addition to the language challenge, localization (i.e. changing information from one representational system to another), personalization (i.e. matching to the preferences of an individual user) and translation from one mode to another (i.e. moving text dominant content to visual information or even braille) could all help in improving Web Accessibility.

The bottom line is that to improve the accessibility of an already published web content, one needs to have a mechanism to alter the content of a web page at the semantic level. Renarration of a web page, is thus this ability to semantically transform web pages. In this paper, we propose the notion of *Semantic Style Sheets* as an approach to enable this semantic transformation (or renarration) of web pages. We believe that by developing a style sheet based approach will not only be resolving our challenge, but also be contributing to the larger goals of Translations, Personalization, Localization, and Customization as well.

[1] http://www.internetlivestats.com/internet-users/.

[2] https://w3techs.com/technologies/overview/content_language/all.

1.4 Notion of Style Sheets

History: The notion of style sheet is not new. It comes to us from the publishing industry where editors, designers and typesetters physically marked the author's printed manuscript with a blue pencil. The designers and typesetters came in after the editing phase. They specified things like the location and size of margins, the layout of the chapters, the fonts to be used in printing the text etc. The whole concept was based on the principle of Separation of Concerns (SoC) between content and its presentation, which continues even today.

The electronic publishing (or e-publishing) industry took the concept of markup and presentation and used it digitally [8]. They used the notion of style sheets to change the way a particular document appeared on screen, from how it appeared on print, and differentiated it from how it was recorded on a CD-ROM. SGML, the predecessor to XML (and HTML) used this in their definition of Document Type Definitions (DTD) [15]. Overtime, with the advent of HTML, the general variation in DTD was fixed to HTML versions and the styling aspect slowly moved out to CSS.

Popular Style Sheets on the Web: SGML initially started out with Document Style Semantics and Specification Language (DSSSL) but it was deemed too complex, and later on was not directly applied to web standards [25].

For the HTML users of the web, CSS (Cascading Style Sheets) has now become the defacto standard [4]. It was first proposed by Hakon W Lie [14], and it later became a CSS1 standard that was promoted by W3C[3]. Now with CSS3, there are even some preprocessors like *Less*, *Stylus* and *Sass* that are available for handling CSS content [16].

For the XML users, XSLT with XSL-FO has been offered as an option [3]. Currently, XSLT is more associated with data-rich, XML-marked, database-interacting web pages. While XLST does not enjoy as much success as its HTML cousin CSS, XSLT does provide a very wide range of transformation possibilities. But, when it comes to already published but not so "well formed" HTML pages, XSLT is less tolerant and less useful for semantic transformations.

Usage of Web Style Sheets: It was already mentioned that style sheets have been popularly used to deal with aesthetics and presentation. In particular they can be used to adjust color, size and style of font, layout, margins, spacing etc.

While XSL styling has the ability to re-assemble and transform documents, like XSL-FO, CSS is chiefly limited to look-and-feel only. Since CSS is the most popular of the style sheets, we restrict our attention to it only.

CSS for HTML: The directives of a CSS Style Sheet are articulated as rule-sets. Their syntax is: *selector* {*property:value;... property:value;*}. The selector

[3] World Wide Web Consortium; A standards body for the Web; https://www.w3.org/TR/html4/present/styles.html.

is the HTML element tag. The property value pairs are unique and defined by the CSS specification released by W3C. Rule-sets are default and apply to the entire document, unless overridden by another applicable rule-set given elsewhere. Rule-sets can be defined inline or imported from another *.css* file.

Here is a snippet of HTML source code from a real NPTEL web page[4]. It shows how styles are integrated into a real web page.

```
<!DOCTYPE html>
<html>
<head><title>NPTEL</title>
   <meta charset="UTF-8">
   <link rel="stylesheet" href="...css">
   ...
   <link rel="stylesheet" href="...footer.css">
   <link rel="stylesheet" href="..css">
   <link href="http://fonts...Cookie" rel="stylesheet">
   ...
</head>
<body>
  ...
  <link rel="stylesheet" href="...lightslider.css"/>
  <link rel="stylesheet" href="...testimonial.css"/>
  ...
  h2 { text-align:top;
       font-weight:bold;
       color:#fff; }
  ...
</body>
</html>
```

Execution: The execution of the style declarations happens in the browser, in rendering engine, as part of the painting work[5]. A CSS processor interprets the rule-sets and produces a CSS Object Model called CSSOM and links it to the Document Object Model (DOM) for rendering. Currently the more popular browsers like Microsoft's Internet Explorer, Mozilla's Firefox and Apple's Safari mostly support CSS and only partially support XSLT[6]. This is yet another reason for our detailed focus on CSS.

1.5 Layout of the Paper

Thus far, in the **Introduction** section, we have already established (1) Web Accessibility and Renarration as the motivation and problem space for our work;

[4] http://nptel.ac.in/.

[5] http://taligarsiel.com/Projects/howbrowserswork1.htm#The_browser_main_functionality.

[6] http://greenbytes.de/tech/tc/xslt/.

(2) we have discussed the background and context of style sheets, both as a notion as well as in practice. Also, we established the need for us to be able to do semantic transformation of web pages.

Going forward, we start with our **Design Consideration** for our Semantic Style Sheets (SSS). This will set the tone for what we are trying to accomplish with our style sheets. We do a literature survey to discuss the gaps with the current approaches and why we are motivated to select style sheets as an approach. We then proceed to discuss the actual **Design** where we give a indicative grammar and the structure of the SSS. Later in **Implementation** we discuss how the SSS has been actualized in code. In **Validation** section we apply the prototype we developed for demonstration to a NASA web page and show how it can be renarrated to suit the various needs of a few minority user communities. Finally, we finish with some reflections and insights in our **Discussion and Conclusion** section.

2 Design Considerations for SSS

At a high level we want to semantically transform a web page to make the renarrated web pages more accessible to a wider group of users. But, more specifically, here are our requirements:

1. It should allow for making changes to the semantics of the page. That is, user should be able to add new content or replace some existing information present in a given published page. We call this **Augmentation** and **Modification** respectively.
2. It should allow for changes to be made to the flow of material in the given page as well. That is, order of nodes should be reconfigurable. This is also part of our **Modification** requirement.
3. It should facilitate the processing or the computation of some existing content to create new values. We call this **Processing**.

3 Literature Survey of Techniques to Modify Web Content

Network and Proxy Solutions: Many options exist for manipulating already published web content. For instance, proxy assisted network based transcoding options exist for changing web page content to fit into the display requirements of a smaller-screen device [13,28]. Network side solutions use separate app servers and backend development. Proxy based solutions intercept and modify content before it reaches the browser [10]. However, these servers require to be configured into the browser flow. That is, the IP address of the proxy needs to be input into the browser. Such imposition may pose some concerns: For instance,

(1) users may perceive configuration as a 'technical' task; or, (2) the users may have some other mandatory institutional proxy that they are compelled to use which forbids them from using ours; or, (3) the end users may have security concerns. Such conditions have been known to dissuade users similar to our target users from using a proxy based solution [12]. It is for these reasons that we did not opt for a network side, proxy based solution.

Web Augmentation Solutions: Web Augmentation techniques exists to allow for the modification of content on the client side [6]. Client-side scripting tools like *GreaseMonkey* [18] or testing tools like *Selinium* [2] exist to enable modifications of published content at the browser level. While the browser is now-a-days quite powerful and capable of running complex *JavaScript*, it is still a programming option. Also, now-a-days security concerns are forcing people not to opt for enabling *JavaScripts* [1]. So, this discouraged us from going with this choice.

Style Sheet Based Solutions: As already indicated, style sheets have also been used to make modification to (the style of) a page. And, when it comes to exploring the choices available to the user, there are mainly two prominent options, namely CSS and XSL.

In our case, for renarration we are opting for a style sheet based choice (over the above listed options) because we wish to empower the end-user and her agents. From a Web Accessibility point of view, we notice that it is often necessary to modify the content for a diverse set of (and also an individualized set of) needs of only a minority of end-users [6,19]. For such low volume users, a complex, coding-intensive, back-end solution may not be appropriate. What they may prefer are simple environments that can be run on the client side, by themselves, or by 3rd party volunteer supporters. Having such a nimble client side environment would assure them more solutions, and also quicker development time. We opted for a style sheets based solution because they offer us this promise. Also [12] suggest that for web accessibility their users preferred a style sheet based approach.

Challenges with the Style Sheet Option: The style sheet based approach for making semantic transformation essentially presents three choices: Option one, go with XSLT; Option two, go with CSS based approach; Or option three, create own style sheet.

Challenges with option one is that XSLT is not supported by all browsers. It is also quite complex to work with[7]. Tool support is also dwindling. Moreover, its popularity has been restricted to XML or data-rich content only. Also, there has been lot of criticism on XSLT, claiming it to be too rigid and cumbersome. This discouraged us from going with this option.

[7] Criticism on XSLT can be found here on Stack Overflow. Refer: https://stackoverflow.com/questions/78716/is-xslt-worth-it.

CSS based option two has its own challenges as well [26]. While CSS is quite popular, it appears to be predominantly focused on styling and aesthetics only. While it is popular and tightly integrated with HTML, the syntax orientation – i.e. it works with selectors and not semantics of the content – makes it difficult for us to do higher order semantic operations. To meet our requirements, we need control at the node level or the concept level, not selector level. Moreover, it lacks the power to compute or process content information. Due to these limitations, we opted out of this as well.

Option three has to do with developing our own style sheet. This task can indeed be quite complex and large. However, selecting this option allows for us to custom design a semantics based "style" declaration that allows for Augmentation, Modification and Processing, which are now missing in CSS. By creating a new style sheet we are not intending to replace either XSL or CSS. Instead, this new proposed style sheet will only add to that portfolio.

4 Design of Semantic Style Sheet

Design Goals: There are two areas in which our proposed Semantic Style Sheet (SSS) differs from the popular CSS. One is in concept – ours is focused on semantics and not just style – and the other is in its application – we empower the end-user or her agent instead of the author or the publisher. By empowerment we mean that the creation of a new style sheet, the potential co-existing of multiple style sheets for the end user to select from, the definition of criteria of when to apply which style sheet etc. are all done without the involvement of either the author or the publisher of the original source page. We summarize our conceptual goals as:

1. **Augment** (or add) new content (like text, links, images etc.)
2. **Modify** existing content (i.e. change or replace content form an existing page)
3. **Process** content (e.g. compute over number, currency, units of measure; carry intelligent processing over text etc.)

Our application or usage goals include:

1. **Meta data:** SSS must have information on who created it, when etc.; there should also be information on the SSS, its label, ID, description etc. And, finally, the web page URL for which the SSS applies.
2. **Selection Criteria:** There should be some guidance on when this particular SSS must be applied when there are other choices available and co-existing for a given web page (i.e. URL).

We present indicative grammar to articulate our conceptual goals. And, we have couched the semantic directives which augment, modify and process, in a larger structure to ensure we meet the application goals.

4.1 Indicative Grammar

We articulate our design for the SSS directives by using a simple grammar. We use the following EBNF notation (as defined in [24]):

= for definition,
, for concatenation,
; for termination,
— for alternation,
[...] for optional,
{...} for repetition, and
e for null

```
SSS = {Directives}
Directives = (SeqNo, Operation)
SeqNo = <integer>
Operation =
    (Add (src-location:<xpath>, add-loc:(above | below),
       (new-url:url | new-nod:<xpath> | new-text:<string> |
        new-tool-tip-text:<string> | new-link:<url>) )) |
    (Repl (src-location:<xpath>,
       (new-nod:<xpath> | new-text:<string> |
        new-tool-tip-text:<string> | new-link:<url>) )) |
    (Process [xform-crncy-inr, xform-unit-british, xform-temp-celsius])
```

4.2 The SSS Document

The SSS Document is the larger structure containing the prior described SSS directives. This document consists of three segments: (1) some meta data (2) a list of semantic transformation directives, and (3) a selection criteria.

Segment 1 of the SSS document (Meta-data) is to contain the URL of the web page which is being transformed, the creator's information, and, label, description and ID for this SSS. Segment 2 of the SSS document (SSS Directives) is to contain the declaratives having a sequence number, a operation and some attributes as indicated by the indicative grammar. Segment 3 (Selection Criteria) is about identifying the criteria that is to be used for selecting this particular SSS from a choice of many. It consists of rules for selection.

Here is a snippet of a basic SSS document structure which we are using to renarrate a web page.

```
{"seg1":{  "sssName":"S1",
           "description":"Created by R1 for U1 community",
           "wp-url":"https://www.grc.nasa.gov/...html" },
 "seg2":[ { "seqNo":"1",
           "operation":"Add",
           "new-text":"<div><img
               src='http://...steam-engine-works.png'></div>",
          "src-location": "//",
          "add_loc": "above" },
        { "seqNo":"2",
          "operation":"Repl",
          "src-location":"",
          "new-text":"<p>Thermodynamics is the ...energy.</p>", } ],
"seg3":{ "communityId":"U1" } }
```

5 Implementation of Prototype

5.1 Architecture of Prototype

We build a simple web application to demonstrate our notion of SSS. The front-end (FE) of the application was developed using *AngularJS* [5]. The back-end (BE) runs *Flask* on *Python 2.7* in a *Virtual Environment* [9]. The processing of the SSS happens at the server level, in the BE, in Python. See Fig. 2. To demonstrate the application of an SSS on a given web page, the FE supplies three things to the BE:

1. **URL** - the location of the web page which is to be renarrated (or semantically transformed)
2. **SSS** - the choice of SSS which is to be applied to this web page
3. **UserID** - the identity of the person accessing the web page

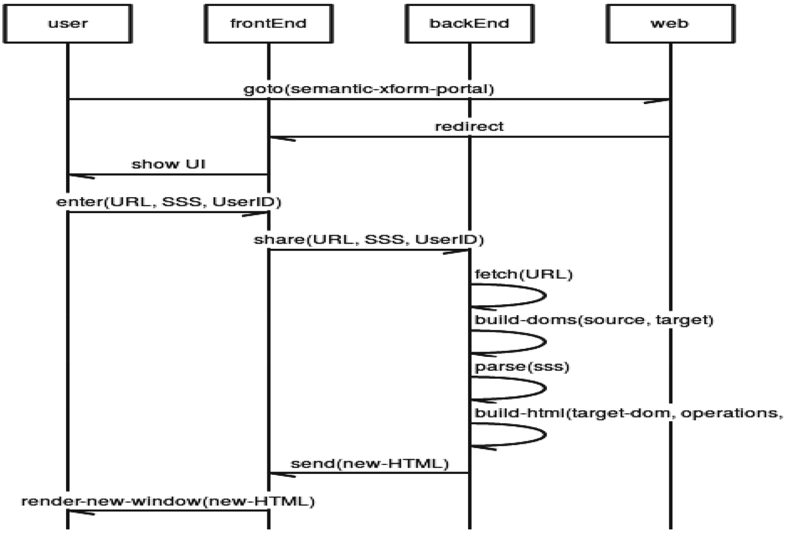
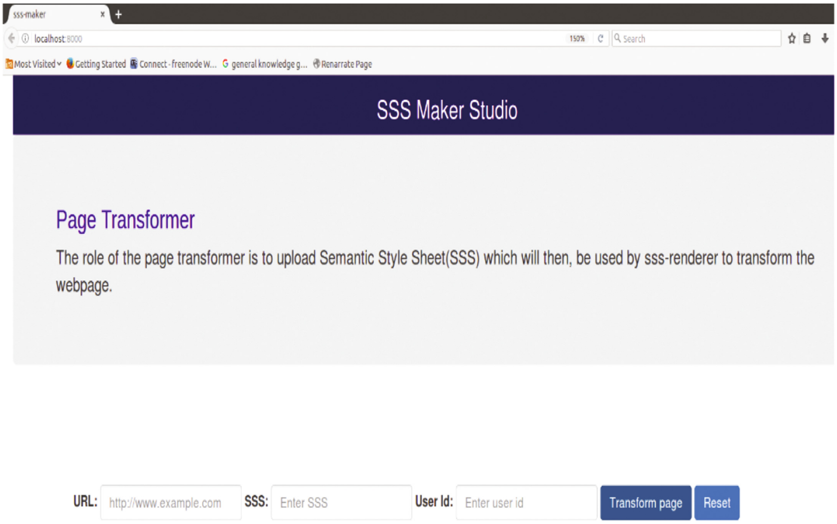


Fig. 2. The sequence chart for the processor application of our SSS.



SSS Maker Studio

Page Transformer

The role of the page transformer is to upload Semantic Style Sheet(SSS) which will then, be used by sss-renderer to transform the webpage.

URL: http://www.example.com SSS: Enter SSS User Id: Enter user id

Fig. 3. The front end of the web app prototype which implements SSS.

See Fig. 3. The BE takes the first parameter, the URL, and uses it to fetch the web page which is to be semantically transformed. It uses the second parameter - SSS - to process the change. And, finally, it uses the UserID to see how the SSS applies to the user. Here is the basic algorithm behind the SSS processing in the BE.

```
fetch-page (URL)
source-DOM = construct-DOM(fetched-page)
target-DOM = source-DOM
sss = parse(JSONobject-sss)
REPEAT
   d = get (next directive)
   a = read d.operation.attributes
   SWITCH (d.operation)
      /* AUGMENTATION handling */
      CASE (add):
          target-DOM = add(a.location, a.new-node, source-DOM)
      /* MODIFY handling */
      CASE (replace):
          target-DOM = replace(a.location, a.new-node, source-DOM)
      /* PROCESS handling */
      CASE (process):
          p = d.operation.attribute [1]
          SWITCH (p)
              CASE (xform-crncy-inr):
                 target-DOM = xform-crncy(p, inr)
              CASE (xform-unit-british):
                target-DOM = xform-units(p, british)
             CASE (xform-temp-celsius):
                target-DOM = xform-temp(p, celsius)
             DEFAULT: Log (operation, unknown)
             Print ("Error: unknown Process operation attempted", p)
      DEFAULT:
         Log (operation, unknown)
         print("Error: unknown operation attempted", operation)
UNTIL all directives done
new-page = construct-HTML(target-DOM)
front-end = send(new-page)
```

5.2 Implementation of SSS

The SSS document has been implemented as a JSON object. This data structure has the three segments that were previously identified.

Processing of SSS: Typically a style sheet has a processor associated with it. For web documents, the browser contains this processor. In our prototype implementation, we have positioned the code in the BE, in the network side. That is, we transform the source web page with SSS before it comes to the browser.

In the server side, the processor could parse and interpret the SSS directives as if it were processing a Domain Specific Language (DSL) [17]. However, in the prototype, we do not do parsing and interpretation, which could easily be added later. Instead, we focus on the transformation algorithm. The pseudo-code for this algorithm has already been shared. This algorithm simply transforms source HTML into a target HTML by way of the SSS directives.

Accessing DOMs and XPaths on the Server: To augment or modify a portion of an HTML, we use DOMs and/or XPaths. But, in current web implementation, the DOMs and XPaths for a given web page are constructed within a browser. Since the processing for our implementation is now in the server side (and not the browser), we lack access to the DOM of both the source and the target pages. To overcome this problem, we use a headless browser, which in our case is *WebKit* based *PhantomJS* [11]. Using the headless browser, on the server side, we create two instances of it to access the source and target DOMs. In this way we gain access to the XPath on the server side.

Additions, replacements or just extractions of node elements from a page (or to a page) are now done in this headless browser using DOM API. Once a target DOM is created, we have our HTML to push to the front-end. See Fig. 2. Completing the target DOM requires processing all the directives in the SSS. Which, in turn, is equivalent to completing the semantic transformation of the source web page.

Implementation of Selection Criteria: Selection criteria can consist of rules which are then addressed by a rule engine like *npm*[8]. However, for our initial implementation, we did not go this route. As we were only trying to demonstrate the feasibility of this aspect, we simply carry out a regular expression match on one of two variables: Community ID or a User ID.

The idea is that one single page may have multiple, co-existing, alternative views. Each of these is a renarration of the original. And, each requiring its own SSS document. The selection of which view to present turns into a selection of which SSS to implement. This selection criteria section enables that choice.

A document meant for the blind users my have their community ID as its selection criteria. Similarly, the same document could also be meant for people of a different language. Their version may have a different SSS, with its own selection criteria indicating the different language speakers. During rendering, the selection criteria checks to see who is viewing the SSS. If it is a registered blind user, then blind SSS is processed. Else, if the user is of a different language, then the diff-language SSS is processed. The community ID and/or user ID are obtained through login and registration processes.

Selection criteria and algorithm can indeed be made more sophisticated to handle complex cases. For example, a rule engine may be established to optimize this facet of SSS processing. But this is seen as future work and is considered out of scope for this paper.

6 Validation of SSS

We validate our idea of SSS by applying our prototype to a real web page and renarrating it. For our test, we arbitrarily chose a page that was focused on a 'complex' topic of Thermodynamics from NASA's website[9].

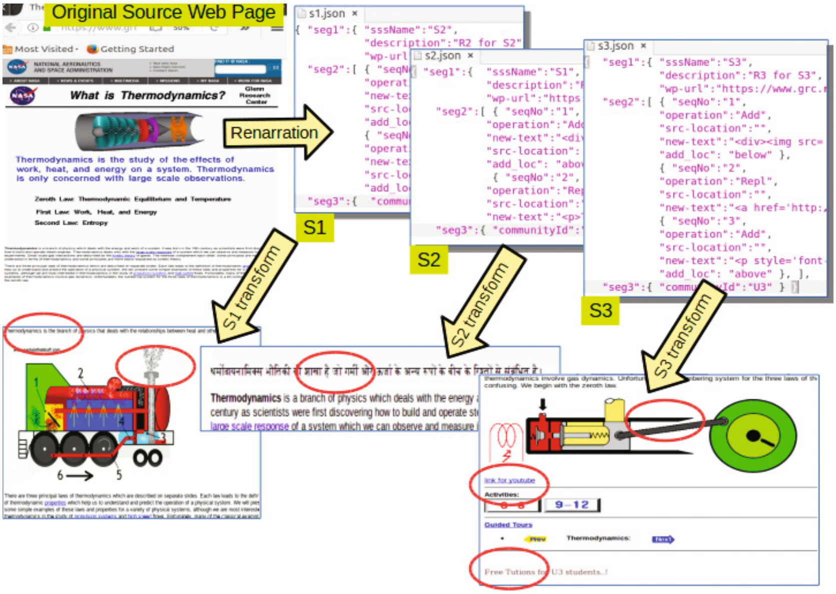
For our demonstration, we made the assumption that the NASA site is confusing and inaccessible to certain non-native English speaking users. We took the roles of three independent, unrelated renarrator volunteers (R1-3), having to develop 3 different SSS (S1-3) for our respective target audience (U1-3).

The assumed objective of R1 was to reduce the complexity of NASA's page for U1. She thus creates an S1 for it. Similarly, R2's objective in creating S2 was to meet the vernacular needs of U2. That is, U2 is Hindi speaking audience, and they need guidance in it. R3's objective was to insert advertisements into a page for all U3. The general idea here is that renarrations can be many, and they can co-exist, but they have to be targeted to a community or a user. For instance, when U1 logs in, they should only see the S1 renarrated NASA site. Similarly U2 should get the S2 renarration, and U3 should get S3.

See earlier given SSS doc snippet for S1. See Fig. 4 for the process of converting original source web page (top left), to S1–S3 (top right), finally yielding three different outputs (bottom). The bottom portion showcases the actual transformed output of our web application. The S1-3 are also original. They were

[8] https://www.npmjs.com/package/json-rules-engine.

[9] https://www.grc.nasa.gov/www/K-12/airplane/thermo.html.



```
{ "seg1":{ "sssName":"S2",
           "description":"R2 for S2"
           "wp-url
  "seg2":[ { "seqN
           "operat
           "new-te
           "src-lo
           "add_lo
           { "seqN
           "operat
           "new-te
           "src-lo
           "add_lo
  "seg3":{  "commu
```

```
{ "seg1":{  "sssName":"S1",
            "description":"R
            "wp-url":"https
   "seg2":[ { "seqNo":"1",
            "operation":"Add
            "new-text":"<div
            "src-location":
            "add_loc": "abov
             { "seqNo":"2",
            "operation":"Rep
            "src-location":"
            "new-text":"<p>"
   "seg3":{ "communityId":"
```

```
{ "seg1":{ "sssName":"S3",
           "description":"R3 for S3",
           "wp-url":"https://www.grc.n
  "seg2":[ { "seqNo":"1",
           "operation":"Add",
           "src-location":"",
           "new-text":"<div><img src=
           "add_loc": "below" },
           { "seqNo":"2",
           "operation":"Repl",
           "src-location":"",
           "new-text":"<a href='http:/
           { "seqNo":"3",
           "operation":"Add",
           "src-location":"",
           "new-text":"<p style='font
           "add_loc": "above" }, ],
  "seg3":{ "communityId":"U3" } }
```

Fig. 4. Three renarrated views being constructed out of three SSS. Real output of a working prototype.

designed to semantically transform the content by adding new content (augmenting), altering existing content (modify), and computing some numbers (processing). The output clearly shows how the content is now semantically transformed to meet the individualized needs of U1-3.

7 Discussion and Conclusions

The prototype demonstrates our proposed idea for our SSS. The intent of SSS is not to replace an existing style sheet like CSS or XSLT. Instead, our intent here is to complement the power of styling the CSS provides with an additional control on semantics.

In developing our renarration idea for addressing the Web Accessibility problem, we discovered that semantic transformation of existing, already published web sites is necessary. Amongst the various approaches out there, we opted to work with a style sheet based approach because it offered non-technical involvement of the end-user. And, it empowered the end-user. Investigating the needs we discovered that our need to Augment, Modify and Process content was not being sufficiently addressed by the current defacto standard - CSS. To facilitate semantic level transformation of web page content we proposed a new style sheet called Semantic Style Sheet. We proposed a grammar for it and developed a simple processor for it. Its utility was finally demonstrated by way of a prototype application.

Through this exercise we realize that

1. style sheets can indeed be developed for doing more than just style adjustments
2. the concept of a semantic oriented style sheet shows promise and can be further developed; perhaps it can be turned into a DSL and be linked with a rule engine
3. Web Accessibility needs of the non-body-disabled user can, to some degree, be met by renarrating existing, already published web content.

Upon reflecting, we also realize that due to the sheer volume of web content that is out there, as a next step, we need to move from a manual renarration process to an automated one. This requires a set of standard semantic structures in specific domains, a set of renarration needs and techniques, which can then be analyzed by developing a set of tools that can automatically process and apply renarration techniques. We leave the task of automation of renarration as a logical next step or a future activity. Finally, we see the work in this paper is a first step towards a major research direction spinning off multiple research areas in the space of semantic web, renarration and web accessibility.

Acknowledgements. The authors are grateful to the effort by Ms. Sadhana Virupaksha, and Mr. Raghav Mittal for developing the prototype for SSS.

References

1. Bandhakavi, S., Tiku, N., Pittman, W., King, S.T., Madhusudan, P., Winslett, M.: Vetting browser extensions for security vulnerabilities with vex. Commun. ACM **54**(9), 91–99 (2011)
2. Bruns, A., Kornstadt, A., Wichmann, D.: Web application tests with selenium. IEEE Softw. **26**(5), 88–91 (2009)
3. World Wide Web Consortium, et al.: XSL transformations (XSLT) version 2.0 (2007)
4. World Wide Web Consortium, et al.: Cascading style sheets level 2 revision 1 (CSS 2.1) specification (2011)
5. Darwin, P.B., Kozlowski, P.: AngularJS Web Application Development. Packt Publishing, Birmingham (2013)
6. Díaz, O.: Understanding web augmentation. In: Grossniklaus, M., Wimmer, M. (eds.) ICWE 2012. LNCS, vol. 7703, pp. 79–80. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35623-0_8
7. Dinesh, T., Uskudarli, S., Sastry, S., Aggarwal, D., Choppella, V.: Alipi: a framework for re-narrating web pages. In: Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, p. 22. ACM (2012)
8. Goldfarb, C.F.: SGML: the reason why and the first published hint. J. Am. Soc. Inf. Sci. (1986–1998) **48**(7), 656 (1997)
9. Grinberg, M.: Flask Web Development: Developing Web Applications with Python. O'Reilly Media, Inc., Sebastopol (2014)
10. Gupta, S., Kaiser, G., Neistadt, D., Grimm, P.: DOM-based content extraction of HTML documents. In: Proceedings of the 12th International Conference on World Wide Web, pp. 207–214. ACM (2003)

11. Hidayat, A.: PhantomJS: headless webkit with Javascript API. WSEAS Trans. Commun. (2013)
12. Kurniawan, S.H., King, A., Evans, D.G., Blenkhorn, P.: Personalising web page presentation for older people. Interact. Comput. **18**(3), 457–477 (2006)
13. Laakko, T., Hiltunen, T.: Adapting web content to mobile user agents. IEEE Internet Comput. **9**(2), 46–53 (2005)
14. Lie, H.W.: Cascading HTML style sheets-a proposal. World Wide Web Consortium (W3C) (1994)
15. Maler, E., Andaloussi, J.E.: Developing SGML DTDs: From Text to Model to Markup. Prentice Hall PTR, Upper Saddle River (1995)
16. Mazinanian, D., Tsantalis, N.: An empirical study on the use of CSS preprocessors. In: 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), vol. 1, pp. 168–178. IEEE (2016)
17. Mernik, M., Heering, J., Sloane, A.M.: When and how to develop domain-specific languages. ACM Comput. Surv. (CSUR) **37**(4), 316–344 (2005)
18. Pilgrim, M.: Greasemonkey Hacks: Tips & Tools for Remixing the Web with Firefox. O'Reilly Media Inc., Sebastapol (2005)
19. Prasad, G.V.S.: Renarrating web content to increase web accessibility. In: Proceedings of the 10th International Conference on Theory and Practice of Electronic Governance, pp. 598–601. ACM (2017)
20. Prasad, G.V.S., Chimalakonda, S., Choppella, V., Reddy, Y.R.: An aspect oriented approach for renarrating web content. In: Proceedings of the 10th Innovations in Software Engineering Conference, pp. 56–65. ACM (2017)
21. Prasad, G.V.S., Dinesh, T., Choppella, V.: Overcoming the new accessibility challenges using the sweet framework. In: Proceedings of the 11th Web for All Conference, p. 22. ACM (2014)
22. Prasad, G.V.S., Ojha, A.: Text, table and graph-which is faster and more accurate to understand? In: 2012 IEEE Fourth International Conference on Technology for Education (T4E), pp. 126–131. IEEE (2012)
23. Prasad, V.G.S., Choppella, V.: Descriptive study of college bound rural youth of AP, India. In: 2013 IEEE Fifth International Conference on Technology for Education (T4E), pp. 76–79. IEEE (2013)
24. Scowen, R.: Extended BNF - a Generic Base Standard. Technical report 14977 (1998)
25. Sperberg-McQueen, C., Goldstein, R.F.: HTML to the max: a manifesto for adding SGML intelligence to the world-wide web. Comput. Netw. ISDN Syst. **28**(1–2), 3–11 (1995)
26. Tidwell, D.: XSLT. O'Reilly Media Inc., Sebastopol (2008)
27. Wu, H.K., Puntambekar, S.: Pedagogical affordances of multiple external representations in scientific processes. J. Sci. Educ. Technol. **21**(6), 754–767 (2012)
28. Zhang, D.: Web content adaptation for mobile handheld devices. Commun. ACM **50**(2), 75–79 (2007)

Efficient Anomaly Detection Methodology for Power Saving in Massive IoT Architecture

Palani Kumar[1](✉), Meenakshi D'Souza[2], and Debabrata Das[2]

[1] Samsung Semiconductor India Research, Bangalore, India
palanikumar@samsung.com
[2] International Institute of Information Technology, Bangalore, India
{meenakshi,ddas}@iiitb.ac.in

Abstract. Energy saving is the paramount factor in the evolving Internet of Things (IoT) due to limited battery energy in the devices. An IoT device in anomaly condition will lead to transmission failure and power drain. It is imperative to detect the anomaly, whose occurrence is random in nature over time. The randomness in failure needs a statistical model of an IoT device to predict and stop the occurrence of anomaly. We propose a novel approach of modeling IoT devices as a finite state automaton, which is irreducible, a priori, has well determined emissions (refers to received signal strength) and finite hidden state space. We have designed a Hidden Markov Model (HMM) based approach to efficiently predict anomaly using which we orchestrate the time interval between successive transmissions from an IoT device. Experimental results reveal that our approach can determine anomaly of IoT device with accuracy as high as 98%. The higher anomaly detection rate results in saving around 14% of IoT device battery power by avoiding redundant transmissions.

1 Introduction

Devices in IoT eco-system are battery powered, heterogeneous, and autonomous in nature. Hence self-management in IoT is inevitable to monitor the health of the devices. The status of the device is indicated to the centralized monitoring server through the radio signal. 802.11 devices consume around 200 mA for processing the sequence of operations of modulation, processing, encoding, and decoding for every wake-up and sending the status. Hence transmitting an optimum number of radio signals between the device and server is important. Device anomalies happen in a random and unpredictable manner. The randomness in failure demands non-enumerative method to predict the anomaly. Hence statistical model is the best choice [10] to predict the possible anomalies. The anomalies are communicated to the centralized server only by means of received signal strength. The transmission needs to be processed on a non-intermittent interval; hence the status of the signal being sent to the server plays a critical role. When the transition from a normal state to abnormal state is slow, it is

D. Das—Senior Member, IEEE.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 256–262, 2018.
https://doi.org/10.1007/978-3-319-72344-0_21

even more difficult to trace out the existence of devices. Also in densely distributed networks, the cluster of progressive events could collide. Collision can lead to dropping of events reaching from the devices to the centralized monitor system, which may be considered as malfunctioning of the device. In this paper, we have proposed a novel statistical model based approach for power saving. The transmission intervals to be followed by a node are determined based on training, through which the state of devices can be studied using Hidden Markov Model (HMM) [8] based framework. HMM predicts the hidden transitions, thereby proposed approach can identify and determine the device anomaly as high as 98% which results in nodal power saving of around 14% by intelligently transmitting the status. Hence our approach guarantees maximum life span of an IoT device and the average battery power spent by each device is optimized to a good extent without compromising on reliability, performance, and efficiency.

Power saving techniques employed at hardware, software, and network area are *Gating* (Clock Gating and Power Gating) [9], *DRx* (Discontinuous Reception) [1], and *Power Save - Polling* [2] respectively. To the best of our knowledge, none of the above techniques, study the random occurrence of anomaly prediction in a massive IoT deployment architecture as well as its impact on power saving. Our proposed approach reduces the average *Power per Bit* in an efficient manner by detecting an anomaly in IoT. The rest of the paper is organized as follows. Section 2 explains HMM modeling of hidden states in IoT device and anomaly prediction. The analytical model of proposed HMM framework for power saving is discussed in Sect. 3. Experimental results are captured through simulation and the conclusion is described in Sects. 4 and 5 respectively.

2 Anomaly Detection and Power Saving in IoT

The invariant in an IoT system is Received Signal Strength Indicator (RSSI) and it is considered as the emission symbol. Since the IoT device satisfies the Markovian properties of being a priori, irreducible, and having a finite set of states, we propose that the entire prediction process can be modeled using Hidden Markov Model. Hence we consider each device as an automaton, where S is set of states $S = S_{(Green)}, S_{(Yellow)}, S_{(Red)}$ which is finite but hidden from the observer. Only the emissions from the states such as normal, degrading, and anomaly are inferred based on the signal strength [4] measured from the devices. The emitted symbol from each state is random and independent of each other. However, evaluating the networked entities that require coordinated interactions among several autonomic devices

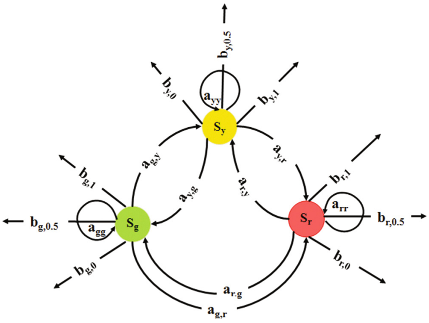


Fig. 1. Three state automaton of an IoT node (Color figure online)

will be even more difficult. Figure 1, depicts the automaton corresponding to an IoT device. We define three states: *Green* (normal state-healthy), *Yellow* (degrading state), and *Red* (anomaly state-nonfunctional). The states are hidden by RSSI observations, which is the signal strength emitted by the model. In Fig. 1, transitions a_{ij} are from normal to degrading and vice-versa or normal to an anomaly and vice-versa or degrading to an anomaly and vice-versa hidden to receiving node. The emission probabilities are represented as b_{jk}, the probability of emitting a symbol k from the state j. For our experiment, we considered the emission probability K with values 0, 0.5, and 1 from each state.

3 Analytical Model of Power Saving HMM Framework

The functional flow of HMM-based approach in an IoT test setup is depicted in Fig. 2. We created one setup integrated with HMM and another setup without HMM-based approach. For our simulation, we primarily used the Contiki operating system based open source wireless sensor networking simulator called "COOJA" [3] to validate our approach. In Cooja simulator, we have designed and implemented three different modules for the server class node. They are HMM algorithm computation, analytic mechanism, and tuning algorithm.

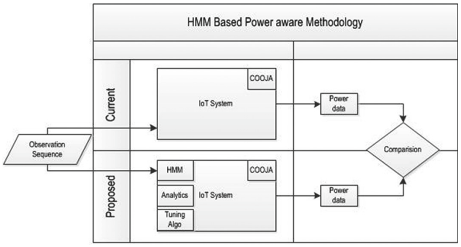


Fig. 2. HMM based approach

The HMM block performs the forward algorithm [6] (only the forward algorithm is considered for implementation because IoT node in Cooja simulator has less memory foot print) to identify the maximum likelihood of the given sequence. The analytic mechanism compares the computed maximum likelihood with assumed maximum value. The duration of the transmission interval is calculated based on the analytic result. Tuning algorithm block identifies the transmission interval value which needs to be designated for the given sequence of inputs from the device and communicates the interval duration back to the device. The final block gets the power factor from two mechanisms for the given sequence and analyzes the difference between the output of HMM and Non HMM-based approach [7].

Signal strength, RSSI, is continuously measured from the device is used to infer the condition of device. To reduce the high dimension data, we introduced *slopping* by means of sequence length. We applied algorithm on the observation sequence recorded at the receiver. Then, we determine the best possible notification interval at which the devices need to communicate to the receiver. The empirical data is derived from the Cooja simulator. Our method identifies the instants where the probability of failure of IoT device is predictable. Using this probability, the number of transmissions between the device and centralized receiver is greatly reduced. Hence by reducing the redundant transmissions between the device and receiver, the power consumption of the IoT device is reduced to increase the battery life time of the device. Based on the above procedure, we avoid storing data for post-processing in the server. The estimated failure probabilities are fed to a tuning algorithm, which computes appropriate transmission interval to be used between the device and the receiver. The equations to implement the forward algorithm are given below for initialization, induction and termination respectively [5].

$$\alpha_t(i) = \pi_1 b_1 O_1; \quad \alpha_{t+1}(j) = \left[\sum_{i=1}^{N} a_{ij}\right] b_j O_{t+1}; \quad P\{O \mid \lambda\} = \sum_{i=1}^{N} \alpha_t(i) \tag{1}$$

In Eq. (1) P{O| λ} needs to be calculated in an iterative manner. Based on the probability value, we determine the transmission interval, such that, higher the probability P{O|λ}, yields higher interval. Hence, based on the newly identified transmission interval, the device will efficiently and optimally do the transmission which needs to be communicated between the IoT device and receiving node.

We conducted the following experiments to validate our estimation.

1. Spreadsheet mechanism to estimate the likelihood of anomaly in IoT setup [7].
2. Cooja simulator based anomaly detection and power saving by optimizing the interval of successive transmissions between the IoT device and receiver.

4 Experimental Results

4.1 Method 1: Experimental Likelihood Estimation

We tested our proposed HMM-based approach on live data feed obtained from an IoT system consisting of six devices that communicate using 802.11n. RSSI signal is collected over a time period of 3600 s and the corresponding HMM models are defined, with one automaton for each device such as p$\{O_1 \mid \lambda_1\}$ ... p$\{O_n \mid \lambda_n\}$ where n = 6. Forward algorithm on this data is analyzed using our proposed approach. Figure 3, depicts two results. The graph on the left (Fig. 3(a)) represents different RSSI data feed from six 802.11n devices. X-axis shows the time (secs) and Y-axis shows the signal strength (dBm) measured from all the six devices differentiated by different markers. The graph on the right (Fig. 3(b)) represents the corresponding anomaly likelihood of the device being in the functional or non-functional state. X-axis shows the sliced data sequence derived from the RSSI value. Y-axis shows the corresponding likelihood of anomaly happening when the signal strength drops significantly low. The result shows that the prediction of the likelihood of occurrence to be as high as 98%.

4.2 Method 2: Simulation in COOJA

In this method, Cooja simulator is used to prove the power saving of the proposed approach. This simulator runs based on a tiny operating system called

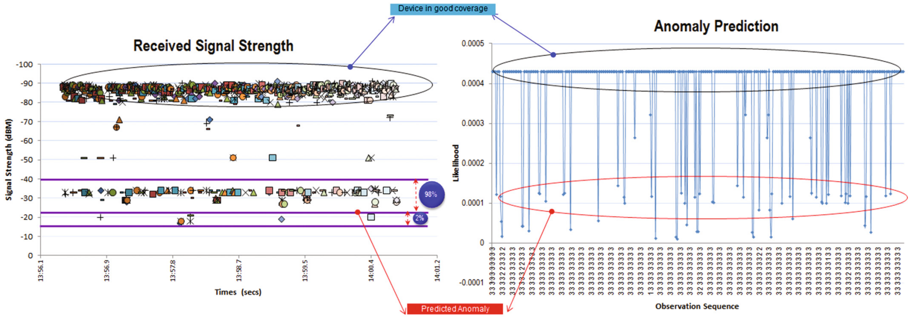


Fig. 3. RSSI data and the corresponding anomaly detection (a): Measured signal strength data from six devices, (b): Prediction of anomaly based on the sliced input sequence categorized up on signal strength

Contiki. We configured two motes with one as a server and other as a client mote. Server mote is kept stationary and the client mote is displaced to different coordinates to simulate the variations in RSSI at the server. We observed that the power consumption level varies when the motes move closer or far away from the coverage area of the server. The event transactions between the server and motes are atomic in nature, which means the success and failure of the transactions are independent of power-aware algorithm. To prove the power saving, the simulation is conducted on dynamic spatial distribution in the same coverage area as one node using ZigBee based mote CC2420. The power factor has been calculated based on CC2420 data sheet (Operating voltage: 2.1 V to 3.6 V, Rx mode: 18.1 mA, Tx mode: 17.4 mA). In Cooja simulator, we prepared one setup with HMM based approach and other without HMM. In the first experiment, we placed the client node far from the server to ensure that the coverage is greatly reduced with the server. We retained the default transmission interval time for the transmissions. It was observed that the number of transmissions were more than 5 packets in an interval of 1 s. The calculated likelihood is in the range of '*42896*'. The RSSI for the corresponding likelihood was around −105 dBm. In the second experiment, we placed the client node closer enough to the server to make sure that the coverage area is within the server. During this experiment, based on the RSSI value received from the node, which was around −75 dBm, the HMM based forward algorithm monitored the RSSI and computed the analytic and tuning process. The outcome clearly showed a reduction in the number of packets transacted between the node and server. The notable likelihood estimation for this observation is '*469873*'. This validates our claim that *when the device is retained in the normal condition and the probability of being in the normal state is high, the number of transactions can be reduced by four fold.*

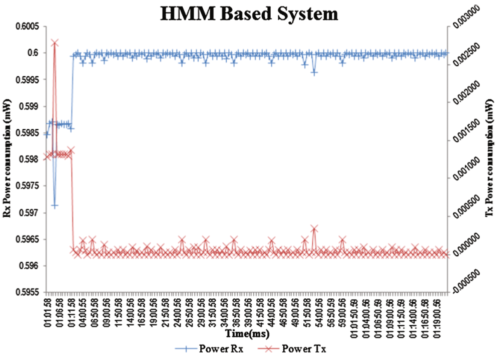


Fig. 4. Power saving: Optimized transmission

To prove our result, we extracted the data from the Cooja. The graph in Fig. 4, depicts the power consumption calculated with HMM implementation. The X-axis shows the time in the milliseconds, primary and secondary Y-axis shows the Rx (marker **X** in the graph) and Tx (marker + in the graph) power consumption respectively for the experimental period. We collected data from both test setups and derived the power consumption of CC2420 mote based on, the amount of time, the device's radio was on Rx mode (rxon), Tx mode (txon), device's cpu in active mode (cpu) and low power mode (lpm). As depicted in Fig. 5, the total number of transmissions is reduced by 13.1% in HMM-based approach. The amount of power consumed in receiving mode (Rx) is reduced to approximately 13.12% and in transmission mode (Tx) is reduced by approximately 1.03%.

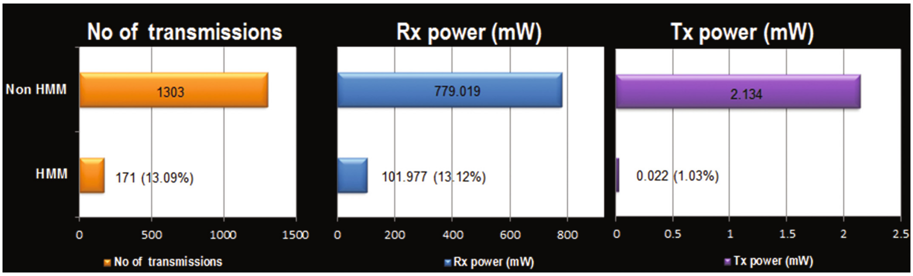


Fig. 5. Result: Reduction in transmissions, Rx power and Tx power

5 Conclusion

The experimental result proves that our approach could bring an average total power saving of approximately 14.12% in the simulation environment. We plan to further optimize this model using a learning-based approach. Our experience in this study suggests that, the optimization of transmission in IoT system is possible through stochastic modeling, thereby resulting in optimal power saving in an IoT devices without compromising on reliability.

References

1. User Equipment procedures in Idle mode, 3GPP 25.304 v1.0.0 (1994–2004)
2. 802.11 Wireless Lan Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 29 March 2012
3. Bagula, B.A., Zenville, E.: IoT emulation with COOJA, pp. 1–44, March 2015
4. Das, D., Das, D., Saha, S.: Evaluation of mobile handset recovery from radio link failure in a multi-rats environment, pp. 1–6, January 2009. https://doi.org/10.1109/IMSAA.2008.4753932
5. Dymarski, P.: Hidden Markov Models, Theory and Applications, 1st edn. InTech Publisher, Vienna (2011)
6. Rabiner, L.R., Juang, B.: An introduction to hidden Markov models. IEEE ASSP Mag. **3**(1), 5–18 (1986). 0740–746/86/0100-0004
7. Kumar, P., D'Souza, M.: Design a power aware methodology in IoT based on hidden markov model. In: 2017 9th International Conference on Communication Systems and Networks (COMSNETS), pp. 580–581 (2017)
8. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE **77**(8825949), 257–286 (1989)
9. Sangiovanni-Vincentelli, A.: A practical guide to Low-Power Design
10. Stefan, M.: Project deliverable d2.5 - adaptive, fault tolerant orchestration of distributed IoT service interactions. In: Internet of Things - Architecture IoT-A, pp. 1–90, November 2011. 257521

Databases, Algorithms, Data Processing and Applications

Approximation Algorithms for Permanent Dominating Set Problem on Dynamic Networks

Subhrangsu Mandal(✉) and Arobinda Gupta

Department of Computer Science and Engineering,
Indian Institute of Technology Kharagpur, Kharagpur, India
{subhrangsum,agupta}@cse.iitkgp.ernet.in

Abstract. A temporal graph is a graph whose node and/or edge set changes with time. Many dynamic networks in practice can be modeled as temporal graphs with different properties. Finding different types of dominating sets in such graphs is an important problem for efficient routing, broadcasting, or information dissemination in the network. In this paper, we address the problems of finding the minimum permanent dominating set and maximum k-dominant node set in temporal graphs modeled as evolving graphs. The problems are first shown to be NP-hard. A $\ln(n\tau)$-approximation algorithm is then presented for finding a minimum permanent dominating set, where n is the number of nodes, and τ is the lifetime of the given temporal graph. Detailed simulation results on some real life data sets representing different networks are also presented to evaluate the performance of the proposed algorithm. Finally, a $(1 - \frac{1}{e})$-approximation algorithm is presented for finding a maximum k-dominant node set.

Keywords: Permanent dominating set
Maximum k-dominant node set · Approximation algorithm
Temporal graph

1 Introduction

Increased use of various mobile devices has introduced communication networks where the network topology changes frequently. Vehicular networks, delay/disruption tolerant networks, low earth orbiting systems etc. are some examples of such dynamic networks. Traditional graph models are not always sufficient to model and analyse such highly dynamic networks, and several models have been proposed recently to represent them such as temporal graphs [11], dynamic graphs [17] etc. In the rest of this paper, we will informally refer to all such graphs as temporal graphs, as all of them represent graphs with time-dependent topologies.

For network with predictable mobility of the mobile entities, Ferreira et al. [4] proposed *evolving graphs* model to represent these types of networks. In this model, the temporal graph is represented as a finite sequence of static graphs,

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 265–279, 2018.
https://doi.org/10.1007/978-3-319-72344-0_22

each static graph being the graph at a discrete timestep. The nodes in each static graph represent the nodes in the network, and an edge in the static graph signifies that a link exists between the corresponding nodes at that timestep. The dynamicity of the topology is captured by the changing node/edge sets of the static graphs in the sequence. The total number of timesteps is called the *lifetime* of the temporal graph. The node at a single timestep is called an *instance* of that node.

Finding different types of dominating sets in a graph has important applications in routing, broadcasting, and information dissemination in static as well as dynamic networks [10,16,19] which may be centralized or distributed in nature. A dominating set (a set of nodes such that every node in the graph is either in the set or has at least one neighbor in the set) provides a subset of nodes such that if those nodes possess some information, they can disseminate that information to all other nodes in the next round of information dissemination. Using similar strategy, dominating set provides an efficient method for broadcasting, routing etc. in a distributed environment. Various definitions of dominating set for dynamic networks are available in the literature. Casteigts et al. [1] have proposed three variations of the dominating set problem for temporal graphs, *temporal dominating set*, *evolving dominating set* and *permanent dominating set.* A *temporal dominating set* is the set of nodes which dominates every other node of the given temporal graph in at least one timestep in the sequence of static graphs defining the evolving graph. An *evolving dominating set* is the set of dominating sets for the static graphs at each timestep. A *permanent dominating set* is the set of nodes which is a dominating set for every static graph at each timestep.

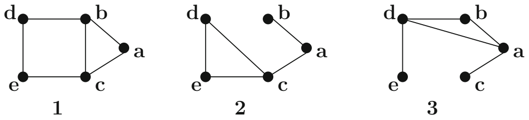


Fig. 1. A temporal graph at different timesteps

In Fig. 1, for the temporal graph $G = \{1, 2, 3\}$ a *temporal dominating set* is $\{c\}$, an *evolving dominating set* is $\{\{b, e\}, \{b, c\}, \{c, d\}\}$ where $\{b, e\}$, $\{b, c\}$ and $\{c, d\}$ are dominating sets for static graphs at timesteps 1, 2 and 3 respectively, and a *permanent dominating set* is $\{a, e\}$. The problem of finding an *evolving dominating set* has been addressed in [8,20]. To the best of our knowledge the problem of finding a *permanent dominating set* has not been addressed till now.

In this paper, we address the problem of finding a *permanent dominating set* for a given temporal graph modeled using the *evolving graphs* model. A solution for the problem of finding the *minimum permanent dominating set* (permanent dominating set of minimum cardinality) for a given temporal graph is proposed first. Then the problem of finding the *maximum k-dominant node set* for a given temporal graph is addressed. The *maximum k-dominant node set* is the set of nodes with a given cardinality k, which covers the maximum number of node instances of the temporal graph among all such sets of size k. It has been assumed for both the problems that only the edge set of the temporal graph changes with time. We show that the first problem is NP-Complete, the second problem is known to be NP-Hard [14]. We first propose a $\ln(n\tau)$-approximation algorithm for finding a minimum permanent dominating set, where n is the number of nodes, and τ is the lifetime of the given temporal graph. Detailed simulation results on some real life data sets for different networks are presented to evaluate the performance of the proposed algorithm. A $(1 - \frac{1}{e})$-approximation algorithm is then presented for finding the maximum k-dominant node set.

The rest of the paper is organised as follows. Section 2 discusses some related works. Section 3 presents the system model used in this paper. Section 4 first shows that the permanent dominating set problem is NP-Complete. It then describes a greedy approximation algorithm to solve the permanent dominating set problem. It also presents detailed simulation results on some real life data sets for different networks to evaluate the performance of the proposed algorithm. In Sect. 5, a greedy approximation algorithm to find a maximum k-dominant node set is given. Finally, Sect. 6 concludes the paper.

2 Related Works

Finding minimum dominating set for static graphs has been a well researched problem because of its applications in various problems of communication networks. Extension of this problem from static graphs to temporal graphs is not straightforward, and various versions of this problem are solved using different approaches.

One approach is the maintenance of a minimum dominating set for a given temporal graph when changes in underlying graph topology occur. Whitbeck et al. [20] solve this problem in an on-demand basis. They compute the dominating set for a graph first and after every edge addition or deletion, recompute it, if the previous one is no longer a dominating set for the changed graph. In [8], Guibas et al. solve the problem of finding a minimum connected dominating set for geometric graphs under node insertion and deletion in a similar fashion. There are other works by Gao et al. [5], and Harshberger [9] which address the problem of finding a dominating set for temporal graphs with a similar approach where nodes follow some continuous trajectory to enter or leave the graph.

Another version of this problem is finding the *temporal dominating set* [1] for a given temporal graph. This version of the problem basically maps to the problem of finding a minimum dominating set for the *underlying graph* which is the union of all graphs at every timestep of the lifetime of the given temporal graph. Hence prior knowledge of changes in the graph topology is required to solve this problem. In [16] Ros et al. have used a graph structure similar to *connected temporal dominating set* (a temporal dominating set which is connected) to propose a backbone structure to address the problem of information dissemination in the context of vehicular networks. In [3], Dubois et al. have solved a slightly altered version of this problem. They have computed the dominating set for the underlying graph after eliminating edges which do not occur infinitely often in the temporal graph.

In this paper, we have proposed an approximation algorithm to solve the *minimum permanent dominating set* problem for a temporal graph. To the best of our knowledge, there is no other existing work which addresses the problem.

3 System Model and Assumptions

In this paper, we represent a temporal graph by the *evolving graphs* [4] model. In this model the node set of the temporal graph remains same, the edge set changes with time and all information about the changes of the edge set is available for a certain time period which is called the lifetime of the given temporal graph. All the changes in the graph topology are known apriori. The node set of the temporal graph is denoted by V and the lifetime is denoted by τ. The time interval for which the temporal graph is available is $(0, \tau]$. Throughout this paper all time intervals are in discrete time system. The edge set is denoted by E where every element $e \in E$ is represented by $e(u, v, (s_e^1, d_e^1), (s_e^2, d_e^2), \cdots)$, where $u, v \in V$ and edge e connects nodes u and v. Each pair of (s_e^i, d_e^i) denotes a half open time interval $(s_e^i, s_e^i + d_e^i]$ for which edge e is connecting u and v in the given temporal graph, s_e^i, $0 \leq s_e^i < \tau$, denotes the starting time of that time interval and d_e^i, $0 < d_e^i \leq \tau$ is the corresponding duration for which e is available, $(s_e^i + d_e^i) \leq \tau$. There may be multiple time intervals for an edge. In such a condition, for any two time intervals say (s_e^i, d_e^i) and (s_e^j, d_e^j), $s_e^i \neq s_e^j$ and if $s_e^i < s_e^j$ then $s_e^i + d_e^i < s_e^j$. Thus the maximum number of time intervals for an edge can be $\lfloor \frac{\tau}{2} \rfloor$. An edge $e \in E$ connecting nodes $u, v \in V$ can be denoted by e_{uv}. As G is an undirected graph the ordering of u, v does not matter. For a node $u \in V$ the sum of all durations of all edges incident on it is called *temporal degree* of u, denoted by TD_u. The temporal graph is denoted by $G(V, E)$.

An instance of a node u at time t $(0 < t \leq \tau)$ is denoted by u^t. All nodes are present for the whole lifetime of G. So each node has τ instances and in total there are $n\tau$ node instances in G, where $|V| = n$.

The set of *neighbouring node instances*, N_u, of a node u in G is defined as the set of node instances in G which has an instance of u as its neighbour. Instances of node u are also included in N_u. Set of neighbouring node instances, N_S, of a set of nodes $S \subseteq V$ is given by $N_S = \bigcup_{v \in S} N_v$. If any instance of a node u does not have any neighbour then u is called an *isolated* node in G.

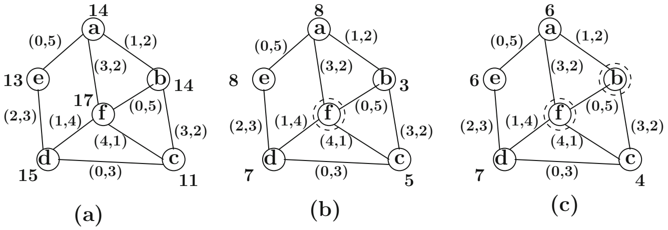
4 Finding Permanent Dominating Set for Temporal Graphs

The *dominating set* problem [6] for static graphs is a well known NP-Complete problem. We know that every static graph can be represented as a temporal graph of lifetime 1. Using this it is quite straightforward to prove that the *dominating set* problem is polynomially reducible to the *permanent dominating set* problem. From this we get the following theorem:

Theorem 1. *Permanent dominating set problem is NP-Complete.*

We next present a greedy approximation algorithm to find a minimum permanent dominating set for a given temporal graph. The algorithm will add nodes one by one to the permanent dominating set. Every node selection will be done greedily based on some parameter. Once a node is added to the permanent dominating set, it will not be removed thereafter. We first define the following.

Definition 1. ***Dominance of node*** u ***at time*** t (CD_u^t)**:** *Let* D^t *be the already constructed partial permanent dominating set till timestep* t, $0 \leq t \leq \tau$. *The dominance of node* u *at time* t CD_u^t, *is equal to* $|N_u \setminus N_{D^t}|$.



(a) (b) (c)

Fig. 2. A temporal graph with dominance information

Figure 2 shows a temporal graph with lifetime $\tau = 5$ and set of nodes $\{a, b, c, d, e, f\}$. Figure 2(a) shows dominance when no node is member of permanent dominating set. Figure 2(b) and (c) show dominance when node f and nodes $\{f, b\}$ are members of permanent dominating set respectively.

Let there be an edge $e(u, v, s_e^1, d_e^1)$ connecting two nodes u and v in the given temporal graph. If u gets added to the permanent dominating set, then we say that node u dominates node v for the duration $(s_e^1, s_e^1 + d_e^1]$. We also say that d_e^1 instances of node v are dominated or covered by node u.

The proposed greedy algorithm works as follows. At any point of execution of the algorithm, all the nodes of the given temporal graph are coloured with any one of the colours *white, black,* or *grey.* A node is coloured *black* if it is a member of the permanent dominating set. A node is coloured *grey* if all instances of it are dominated by at least one node in the permanent dominating set formed so far. A node is coloured *white* if at least one instance of it is not dominated by any node in the permanent dominating set. Initially all nodes are *white.* If a node becomes *grey* or *black* it never becomes *white* again. This implies that the dominance of a node either decreases or remains the same with each round of node selection as a member of the permanent dominating set. Higher dominance of a node at a certain point of time means very few neighbours of this node are *black* and it will dominate more uncovered node instances if it is selected as a member of the permanent dominating set. Thus the node with the highest value of dominance becomes a good choice for selection as a member of the permanent dominating set. The proposed algorithm *GreedyPDS* uses this greedy strategy to construct a permanent dominating set. At any step, the algorithm will choose the node with the highest dominance value, add it to the permanent dominating set, recompute the dominance of all other nodes and then proceed to the next step. The pseudo code shown in Algorithm 1 describes the basic steps of the algorithm. In Algorithm 1, I is the set of isolated nodes in G, $colour_u^t$ is the colour of a node u at time t, CV_u^t denotes the total number of instances of u with black neighbours till time t, TD_u is the temporal degree of node u. In Algorithm 1, initializations are done in Lines 1–3. Then Lines 4 and 5 add all isolated nodes to the permanent dominating set, D. Finally Lines 7–9 add nodes with maximum dominance at the time of node selection to D. It stops when all nodes in G are coloured with black or grey.

Algorithm 1. GreedyPDS(G)

```
Input: A temporal graph $G$ with node set $V$ edge set $E$, $|V| = n$, lifetime $\tau$.
Output: A permanent dominating set $D$.
 1: $D := \emptyset$
 2: $\forall u \in V$ $colour_u^0 := white$,
 3: $\forall u \in V$ $CV_u^0 := 0$, $CD_u^0 := TD_u + \tau$
 4: for all $i, u_i \in I$ do
 5:     AddToDomSet($u_i$)
 6: end for
 7: while $\exists\ v \in V$ such that $colour_v^t = white$ do
 8:     $u :=$ max $\{CD_u^t | u \in V\}$
 9:     AddToDomSet($u$)
10: end while
11: return ($D$)
```

The algorithm *GreedyPDS* calls the subroutine *AddToDomSet* to add each node to D and update the dominance information of the other nodes in G. To recalculate the dominance information of each node after each node selection, we maintain a list of time intervals, *cvg* list, sorted by the starting time of time interval, per node in G. After adding a node u to D, the subroutine *AddToDomSet* colours it with black and inserts the associated time intervals of edges incident on u to the *cvg* lists of the nodes which are connected with u by those edges. During insertion, it merges all the overlapping time intervals into a single time interval. Then, if any neighbour of u has neighbours in D throughout the lifetime, it colours that node with grey. As u is added to D the neighbouring nodes of u will no longer dominate any node instances of u, *AddToDomSet* subtracts that value from the dominance information of the neighbours of u.

As some instances of neighbouring nodes of u get dominated by u, some changes may happen in the dominance information of two hop neighbours of u. Let node v and node w be one hop and two hop neighbours of u respectively, such that e_{uv}, $e_{vw} \in E$ and $|(s_{e_{uv}}, s_{e_{uv}} + d_{e_{uv}}] \cap (s_{e_{uv}}, s_{e_{vw}} + d_{e_{vw}}]| = \lambda \neq 0$. Since u is a black node and it dominates v for the duration $(s_{e_{uv}}, s_{e_{uv}} + d_{e_{uv}}]$, then w

is no longer going to dominate node v for this λ duration. So *AddToDomSet* subroutine subtracts λ from the dominance of w.

Figure 3 shows the execution of the proposed greedy algorithm *GreedyPDS* on a given temporal graph with lifetime 5. Figure 3(a) shows the given temporal graph with dominance information at the beginning of the execution. At first, as node f has the highest dominance of 17, it gets added to the permanent dominating set and it is coloured black. As all instances of node b are dominated by f, it is coloured grey. Figure 3(b) depicts this scenario. After that, by similar greedy strategy, node a and node c are coloured black, then rest of the nodes are coloured grey, and the algorithm terminates. Figure 3(c) and (d) show the steps of selection of node a and node c as members of the permanent dominating set. $\{f, a, c\}$ is the resultant permanent dominating set.

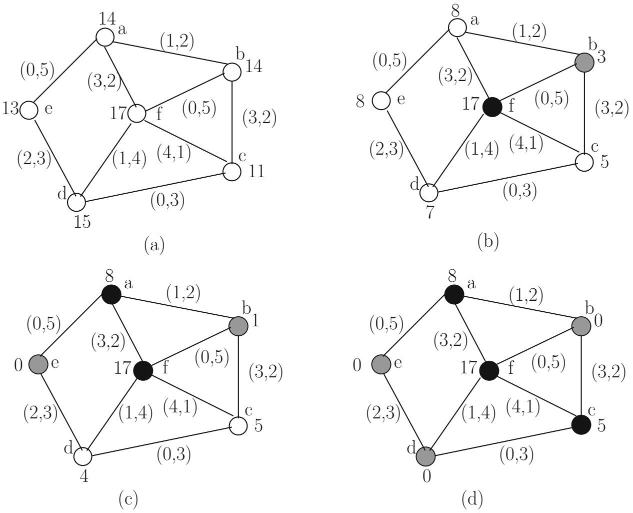


Fig. 3. Execution of *GreedyPDS* on a temporal graph with lifetime 5

Theorem 2. *GreedyPDS correctly computes a permanent dominating set D for a temporal graph G in $O(m\tau^2 n^2)$ time.*

Proof. At first *GreedyPDS* selects all isolated nodes as members of D. Then it adds a node with the highest dominance at that point of time to D. This procedure continues until all nodes are coloured with black or grey. This means that *GreedyPDS* terminates only when all nodes of the given temporal graph are either a member of D or have neighbours in D throughout the lifetime of G. This implies that D is a permanent dominating set for G.

For the dominance information, one sorted list per node is maintained and the maximum number of time intervals per edge is $\lfloor \frac{\tau}{2} \rfloor$. Insertion of a time interval to the list of a node takes $O(\frac{\tau}{2})$ time. So after each node selection, update of the dominance information of its one and two hop neighbours takes $O(\frac{\tau^2 n}{4})$ and $O(\frac{\tau^2 n^2}{4})$ time respectively, as any node can have maximum n neighbours. Hence the algorithm takes $O(m\tau^2 n^2)$ time to find D for G, where $m = |D|$. □

To prove the approximation ratio of *GreedyPDS* we need following lemmas.

Lemma 1. *Let G be a temporal graph with lifetime τ, P be the set of uncovered node instances at time t, $0 \leq t < \tau$. Let D be any permanent dominating set for P. Then there must be at least one node u in P, such that $CD_u^t \geq \frac{|P|}{|D|}$.*

Proof. We prove the lemma by contradiction. Suppose that there are no such node in G with dominance at time t greater than or equal to $\frac{|P|}{|D|}$. So every node has dominance less than $\frac{|P|}{|D|}$ at time t. All $|D|$ nodes of the permanent dominating set are chosen from P. As dominance of any node does not increase with time so the total number of node instances dominated by D is less than $|P|$ which shows that D does not dominate all $|P|$ node instances. This is a contradiction. Hence the lemma holds. □

Lemma 2. *Let G be a temporal graph, OPT be the optimal permanent dominating set for G. We run algorithm GreedyPDS on G and let N_k be the uncovered node instances remaining after $(k-1)$ rounds of GreedyPDS. Let v_k be the selected node at the k^{th} round then $CD_{v_k}^{t_k} \geq \frac{|N_k|}{|OPT|}$.*

Proof. We prove this lemma by induction.

- ***Base Case:*** Base case of this lemma is the first round of node selection. At first round of node selection by *GreedyPDS*, as all node instances of G are uncovered, from Lemma 1, we know that there exists at least one node which has dominance greater than or equal to $\frac{|N_1|}{|OPT|}$. As *GreedyPDS* selects the node with maximum dominance for any $k = 1, 2, \cdots, l$ where l is the size of permanent dominating set, so $CD_{v_1}^{t_1} \geq \frac{|N_1|}{|OPT|}$.
- ***Inductive Step:*** Let this result hold upto m rounds. We have to show that this lemma holds for $(m+1)^{th}$ round as well. At $(m+1)^{th}$ round we have the set $N_{m+1} \subset N_1$ of uncovered node instances and m nodes which are already selected by *GreedyPDS* as member of the permanent dominating set. According to the algorithm these m nodes do not dominate any instances from N_{m+1}. Now as $N_{m+1} \subset N_1$ and OPT is a dominating set for N_1, it is also a dominating set for N_{m+1}. There are three possible cases:

  Case 1: *No node from the m nodes are in OPT*: For this case all nodes of OPT are selected from unselected nodes and it is currently dominating N_{m+1}. So from Lemma 1 it holds that $CD_{v_{m+1}}^{t_{m+1}} \geq \frac{|N_{m+1}|}{|OPT|}$.

  Case 2: *Some nodes of OPT are from m selected nodes and some from unselected nodes:* For this case let there be p nodes of OPT which are from the m selected nodes. So $|OPT| - p$ nodes of OPT dominate all N_{m+1} uncovered node instances as m selected nodes do not dominate any instances of N_{m+1}. So from Lemma 1, $CD_{v_{m+1}}^{t_{m+1}} \geq \frac{|N_{m+1}|}{|OPT|-p}$. This shows that $CD_{v_{m+1}}^{t_{m+1}} \geq \frac{|N_{m+1}|}{|OPT|}$ as $p > 0$.

Case 3: *All nodes of OPT are from m selected nodes:* This case is not possible because, then it will not be able to dominate N_{m+1} uncovered node instances which is a contradiction.

In all cases the stated lemma holds. □

Theorem 3. *GreedyPDS is a* $\ln(n\tau)$*-approximation algorithm.*

Proof. Let there be n nodes in the given temporal graph G and its lifetime be τ. So after the construction of the permanent dominating set D, for all $n\tau$ node instances of G, either those node instances or their neighbours should be in D. Let OPT be the optimal permanent dominating set and $GPDS$ be the permanent dominating set returned by *GreedyPDS* for G and $|GPDS| = l$.

GreedyPDS selects a node at each round and this process continues upto l rounds. Let the i^{th} round of node selection happens at time t_i. Let before the selection at k^{th} round the number of uncovered node instances be n_k. So $n_1 = n\tau$ and $n_{l+1} = 0$. At the first iteration which occurs at t_1, according to our greedy strategy, the node with the highest dominance is added to $GPDS$. As OPT optimally dominates $n\tau$ node instances, the size of the optimal permanent dominating set for uncovered node instances at t_1 is $|OPT|$. Then the average number of node instances dominated by each node of OPT is $\frac{n\tau}{|OPT|}$. So from Lemma 1 there must be at least one node with dominance at that point of time greater than or equal to $\frac{n\tau}{|OPT|}$. So if the selected node with the maximum dominance is v_1 and dominance of the node v_1 at time t_1 is $CD_{v_1}^{t_1}$, then:

$$CD_{v_1}^{t_1} \geq \frac{n\tau}{|OPT|} \implies \frac{1}{CD_{v_1}^{t_1}} \leq \frac{|OPT|}{n\tau}$$

Since OPT is also a permanent dominating set for n_k number of uncovered node instances therefore, from Lemmas 1 and 2, there is at least one node u such that CD_u greater than or equal to $\frac{n_k}{|OPT|}$. So for the k^{th} round of node selection if the selected node is v_k, it can be written that:

$$\begin{aligned}\frac{1}{CD_{v_k}^{t_k}} &\leq \frac{|OPT|}{n_k}\\ 1 &\leq \frac{CD_{v_k}^{t_k}}{n_k}.|OPT|\\ &\leq \frac{n_k - n_{k+1}}{n_k}.|OPT|\end{aligned}$$

Taking summation from $k = 1$ to l on both sides of the inequality.

$$\sum_{k=1}^{l} 1 \leq \sum_{k=1}^{l} \frac{n_k - n_{k+1}}{n_k}.|OPT| \tag{1}$$

$$l \leq |OPT|.\sum_{k=1}^{l}\left(\frac{1}{n_k} + \frac{1}{n_k - 1} + \cdots + \frac{1}{n_{k+1}+1}\right) \tag{2}$$

$$= |OPT|.\sum_{i=1}^{n\tau}\frac{1}{i} = \ln(n\tau).|OPT|$$

Equation 2 comes from Eq. 1 from the fact that $\frac{1}{n_k} \leq \frac{1}{n_k - i}$ for each $0 \leq i < n_k$. The final result shows that *GreedyPDS* is a $\ln(n\tau)$-approximation algorithm. □

4.1 Experimental Results

We have evaluated the proposed algorithm by running it on some real life data sets representing different types of networks that capture the mobility patterns of mobile devices [2], contacts between people [7,18], and communications between different autonomous systems [12]. The work in [2] reports experiments done to find the communication opportunities between mobile devices distributed between the participants of INFOCOM'06. A permanent dominating set on this network will identify the devices that can be used to disseminate information to all other devices. Similarly, the works in [7,18] represent networks to model contacts between students and [12] reports data transmission between autonomous systems in the internet.

Each data set contains contact information for a total of T time, where T varies between the data sets. For each such data set, we have considered several values of $\rho < T$, and divided the total time into $\tau = \frac{T}{\rho}$ subintervals, each of which is taken as a single timestep for the temporal graph to be constructed. The static graph for each timestep is constructed with edges added between two nodes if the nodes come into contact at least once in that timestep (i.e., in the subinterval ρ). This gives a temporal graph with lifetime τ. The permanent dominating set of the resultant temporal graph is then computed. Note that an obvious lower bound on the size of the permanent dominating set of any temporal graph is the number of isolated nodes in the graph, as such nodes must be included in the set. A lower value of ρ increases the number of isolated nodes, and hence the values of ρ are chosen such that the number of isolated nodes are not very high.

The first data set contains $T = 342915$ s of contact information between 98 imotes distributed among the participants of INFOCOM'06 [2]. The values of ρ considered for this data set are 25000, 30000, 35000 and 40000 s. The second data set from SocioPatterns contains $T = 61960$ s of contact information between students in a primary school [7,18]. The values of ρ considered for this data set are 5400, 7200, 9000, 10800 s. The third and fourth data sets are collected from SNAP [13] autonomous systems graph, as-733 [12] and as-caida [12]. The as-733 data set has 733 daily instances of communication data between different nodes of different autonomous systems on the internet. We have taken $T = 50$ days of communication data, with $\rho = 1$, 2, 3, and 4 days (T was taken to be 51 and 48 days for $\rho = 3$ and 4 respectively to get an integer number of timesteps). The as-caida data set has 122 instances of communication data between different nodes of different autonomous systems on the internet. Each communication instance contains the communication data on a particular day over the years 2004 to 2007. The values of ρ considered for this data set are 1, 2, 3, and 4 months. We have merged all available data in a single month to create communication data for a single month. Thus we have got $T = 47$. For $\rho = 2$, 3 and 4, the last timestep contains data of 1, 2 and 3 months respectively. Thus, for each data set, four

Table 1. Results of running *GreedyPDS* on three different data sets

Data sets	Interval length (ρ)	No. of nodes	No. of edges	Lifetime (τ)	No. of isolated nodes	Size of PDS		$\ln(n\tau)$
						Naive approach	GreedyPDS	
INFOCOM'06	25000 s	98	4398	14	53	75	60	7.224
	30000 s	98	4398	12	48	72	56	7.069
	35000 s	98	4398	10	36	59	47	6.887
	40000 s	98	4398	9	37	62	47	6.782
SocioPatterns	5400 s	242	8317	12	91	203	128	7.973
	7200 s	242	8317	10	87	189	123	7.791
	9000 s	242	8317	7	121	216	139	7.434
	10800 s	242	8317	6	62	135	92	7.281
as-733, SNAP	1 day	3328	7167	50	626	1530	1422	12.022
	2 days	3328	7167	25	407	1315	1238	11.329
	3 days	3328	7167	17	383	1283	1211	10.943
	4 days	3328	7167	12	352	1244	1174	10.595
as-caida, SNAP	1 month	31379	101945	47	20512	23235	21946	14.204
	2 months	31379	101945	24	18364	21257	19949	13.531
	3 months	31379	101945	16	17874	20716	19460	13.126
	4 months	31379	101945	12	17463	20274	19040	12.838

different temporal graphs are constructed with different lifetime (τ) values. The first six columns of Table 1 shows the details of the temporal graphs formed.

The size of the permanent dominating set obtained by running the *GreedyPDS* algorithm on these temporal graphs is compared with two other values, the lower bound obtained from the number of isolated nodes, and the size of a permanent dominating set obtained by using a greedy approach for computing dominating set for a static graph based on the most number of uncovered neighbours of a node [15]. In this approach, the dominating set for each static graph at every timestep of the lifetime of the given temporal graph is computed first. Then the union of all these static dominating sets is taken to construct the permanent dominating set of the temporal graph. Note that the lower bound using isolated nodes is a loose bound; however, it can still serve as a worst case reference for comparing the size of the sets obtained.

Table 1 shows the results of our experiments. The results show that for the first two and last data sets, the size of the permanent dominating set obtained by *GreedyPDS* is within twice the lower bound, while for the third data set it is within four times the lower bound. The size of the permanent dominating set obtained for all cases is much better than the worst case size obtained from the theoretical bounds proved. The results also clearly show that the *GreedyPDS* algorithm outperforms the naive algorithm in all cases.

The reason for the relatively higher size of the permanent dominating set for the third data set is as follows. As can be seen from the table, the edge to node ratio for the third data set is 2.153, while for the first two data sets, this ratio

44.877 and 34.367 respectively. This shows that the graphs generated from the first two data sets are more densely connected than those generated from the third data set. The relatively sparse graphs of the third data set result in the relatively larger size of the permanent dominating set. Though the temporal graphs generated from the last data set are also relatively sparse in nature with edge to node ratio 3.249, but for this data set, in all cases, more than half of the nodes are isolated and all are included in the permanent dominating set. This results in a permanent dominating set of relatively smaller size for the last data set.

5 Finding Approximate Maximum *k*-Dominant Node Set for Temporal Graphs

In this section, the problem of finding maximum k-dominant node set for a temporal graph $G(V, E)$ has been addressed. It can be seen that if we select a single node, $v \in V$ as a member of the maximum k-dominant node set it will dominate set of its neighbouring node instances N_v, we refer to it as *dominance set* (DS_v) of v and $|DS_v|$ is the total dominance of v. Similarly the dominance set of a set $S \subseteq V$ is $DS_S := \bigcup_{v \in S} DS_v$ and total dominance of S is $|DS_S|$.

Total dominance can also be expressed as a function $f : \{A : A \subseteq V\} \rightarrow \mathbb{Z}_+$, $f(A) = |\bigcup_{v \in A} DS_v|$. Then the problem reduces to finding a set A with cardinality of a given positive integer $k \leq |V|$, for which $f(A)$ is maximum in the given temporal graph G. This problem is a NP-Hard problem [14].

The proposed greedy approximation algorithm uses the same steps as algorithm of *GreedyPDS* with the only difference that this time the algorithm stops either when the number of selected nodes becomes k or all node instances of G are dominated. We refer to this algorithm as *GreedyKDS*.

Lemma 3. *If $A \subset B \subseteq V$ for a given temporal graph $G(V, E)$ then, $DS_A \subseteq DS_B$.*

Proof. We prove this lemma by contradiction. Given that $A \subset B \subseteq V$, let $DS_A \nsubseteq DS_B$. This means that there is atleast one node instance u^t, where $0 < t \leq \tau$, such that $u^t \in DS_A$ and $u^t \notin DS_B$. From the definition of dominance set we know that either $u^t \in A$ or a neighbour of u^t belongs to A. This implies that there is atleast one instance of a node belonging to set A that does not belong to B. This contradicts the given statement $A \subset B$ because when one node is included in a set, all instances of that node are included in that set. Hence $DS_A \subseteq DS_B$. □

Lemma 4. *f is a nondecreasing submodular function.*

Proof. It is straight forward to show from Lemma 3 that f is a nondecreasing function. Next we prove that f is a submodular function.

Let there be two sets A and B such that $A \subset B \subset V$ and u be a node, such that $u \in V$ and $u \notin B$. We need to prove that $f(A \cup \{u\}) - f(A) \geq f(B \cup \{u\}) - f(B)$. The following cases are possible:

- ***Case 1:*** $(DS_u \cap DS_B) = \emptyset$ ***:***
  For this case,

$$DS_u \cap DS_B = \emptyset \Rightarrow DS_u \cap DS_A = \emptyset$$

So,

$$\begin{aligned} f(B \cup \{u\}) - f(B) &= |DS_u \cup DS_B| - |DS_B| \\ &= |DS_u| + |DS_B| - |DS_B| \\ &= |DS_u| \end{aligned}$$

$$\begin{aligned} f(A \cup \{u\}) - f(A) &= |DS_u \cup DS_A| - |DS_A| \\ &= |DS_u| + |DS_A| - |DS_A| \\ &= |DS_u| \end{aligned}$$

So for this case,

$$f(A \cup \{u\}) - f(A) = f(B \cup \{u\}) - f(B) \tag{3}$$

- ***Case 2:*** $(DS_u \cap DS_B) \neq \emptyset$ ***:***
  For this case, the following subcases are possible:
  - ***Case 2a:*** $(DS_u \cap DS_B) = (DS_u \cap DS_A)$ ***:***
    For this scenario,

$$\begin{aligned} f(A \cup \{u\}) - f(A) &= |DS_u \cup DS_A| - |DS_A| \\ &= |DS_u| + |DS_A| - |DS_u \cap DS_A| - |DS_A| \\ &= |DS_u| - |DS_u \cap DS_A| \end{aligned}$$

$$\begin{aligned} f(B \cup \{u\}) - f(B) &= |DS_u \cup DS_B| - |DS_B| \\ &= |DS_u| + |DS_B| - |DS_u \cap DS_B| - |DS_B| \\ &= |DS_u| - |DS_u \cap DS_B| \\ &= |DS_u| - |DS_u \cap DS_A| \end{aligned}$$

So for this case,

$$f(A \cup \{u\}) - f(A) = f(B \cup \{u\}) - f(B) \tag{4}$$

  - ***Case 2b:*** $(DS_u \cap DS_B) \supset (DS_u \cap DS_A)$ ***:***
    For this scenario,

$$\begin{aligned} f(A \cup \{u\}) - f(A) &= |DS_u \cup DS_A| - |DS_A| \\ &= |DS_u| + |DS_A| - |DS_u \cap DS_A| - |DS_A| \\ &= |DS_u| - |DS_u \cap DS_A| \end{aligned}$$

$$\begin{aligned} f(B \cup \{u\}) - f(B) &= |DS_u \cup DS_B| - |DS_B| \\ &= |DS_u| + |DS_B| - |DS_u \cap DS_B| - |DS_B| \\ &= |DS_u| - |DS_u \cap DS_B| \end{aligned}$$

As $(DS_u \cap DS_B) \supset (DS_u \cap DS_A) \Rightarrow |DS_u \cap DS_B| > |DS_u \cap DS_A|$, then,

$$f(A \cup \{u\}) - f(A) > f(B \cup \{u\}) - f(B) \tag{5}$$

- ***Case 3:*** $(DS_u \cap DS_B) \subset (DS_u \cap DS_A)$ ***:***
  As $A \subset B$, from Lemma 3 this scenario is not possible.
- ***Case 4:*** $(DS_u \cap DS_B) \not\supseteq (DS_u \cap DS_A)$ ***:***
  As $A \subset B$, from Lemma 3 this is also an impossible scenario.

So from (3), (4) and (5), it can be said that that,

$$f(A \cup \{u\}) - f(A) \geq f(B \cup \{u\}) - f(B) \tag{6}$$

Hence f is a nondecreasing submodular function. □

Theorem 4. *Algorithm GreedyKDS is a* $(1-\frac{1}{e})$*-approximation algorithm, where* e *is the base of natural logarithm.*

Proof. According to Lemma 4, f is a nondecreasing submodular function. Algorithm *GreedyKDS* maximizes the function f with the constraint that the cardinality of the resultant set is bounded by k. It has been proved (Proposition 4.1, 4.2 and 4.3, Lemma 4.1 and Theorem 4.1 and 4.2) in [14] that if a nondecreasing submodular function maximization problem with cardinality constraint is solved with greedy approach, then the approximation ratio of the greedy algorithm is $(1 - \frac{1}{e})$. Hence, using the results in [14] we can say that *GreedyKDS* is a $(1 - \frac{1}{e})$-approximation algorithm. If the algorithm terminates before k rounds the resultant solution is the optimal one. □

6 Conclusion

In this paper, we have investigated permanent dominating sets and its applications in highly dynamic networks. In particular, we have presented a $\ln(n\tau)$-approximation algorithm for finding a minimum permanent dominating set and a $(1 - \frac{1}{e})$-approximation algorithm for finding the maximum k-dominant node set for a given temporal graph. The first algorithm has also been simulated on four real life data sets for performance evaluation which supports the theoretical results.

References

1. Casteigts, A., Flocchini, P.: Deterministic algorithms in dynamic networks: Problems, analysis, and algorithmic tools, Technical report, DRDC 2013-020 (2013)
2. Chaintreau, A., Hui, P., Scott, J., Gass, R., Crowcroft, J., Diot, C.: Impact of human mobility on opportunistic forwarding algorithms. IEEE Trans. Mobile Comput. **6**, 606–620 (2007)
3. Dubois, S., Kaaouachi, M., Petit, F.: Enabling minimal dominating set in highly dynamic distributed systems. In: Stabilization, Safety, and Security of Distributed Systems - 17th International Symposium, SSS, 18–21 August, pp. 51–66 (2015)
4. Ferreira, A.: On models and algorithms for dynamic communication networks: the case for evolving graphs. In: 4e Rencontres Francophones sur les Aspects Algorithmiques des Telecommunications (ALGOTEL), pp. 155–161 (2002)
5. Gao, J., Guibas, L.J., Hershberger, J., Zhang, L., Zhu, A.: Discrete mobile centers. In: 17th Annual Symposium on Computational Geometry, 3–5 June, pp. 188–196 (2001)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, New York (1979)
7. Gemmetto, V., Barrat, A., Cattuto, C.: Mitigation of infectious disease at school: targeted class closure vs school closure. BMC Infect. Dis. **14**, 695 (2014)
8. Guibas, L.J., Milosavljevic, N., Motskin, A.: Connected dominating sets on dynamic geometric graphs. Comput. Geom. **46**, 160–172 (2013)
9. Hershberger, J.: Smooth kinetic maintenance of clusters. In: 19th ACM Symposium on Computational Geometry, 8–10 June, pp. 48–57 (2003)
10. Jain, S., Fall, K.R., Patra, R.K.: Routing in a delay tolerant network. In: ACM SIGCOMM 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, pp. 145–158 (2004)
11. Kostakos, V.: Temporal graphs. Phys. A **388**, 1007–1023 (2009)
12. Leskovec, J., Kleinberg, J.M., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 21–24 August, pp. 177–187 (2005)
13. Leskovec, J., Krevl, A.: SNAP datasets: stanford large network dataset collection, June 2014. http://snap.stanford.edu/data
14. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions-i. Math. Program. **14**, 265–294 (1978)
15. Parekh, A.K.: Analysis of a greedy heuristic for finding small dominating sets in graphs. Inf. Process. Lett. **39**, 237–240 (1991)
16. Ros, F.J., Ruiz, P.M.: Minimum broadcasting structure for optimal data dissemination in vehicular networks. IEEE Trans. Veh. Technol. **62**, 3964–3973 (2013)
17. Siljak, D.: Dynamic graphs. Nonlinear Anal. Hybrid Syst. **2**, 544–567 (2008)
18. Stehl, J., Voirin, N., Barrat, A., Cattuto, C., Isella, L., Pinton, J., Quaggiotto, M., Van den Broeck, W., Rgis, C., Lina, B., Vanhems, P.: High-resolution measurements of face-to-face contact patterns in a primary school. PLOS ONE **6**, e23176 (2011)
19. Stojmenovic, I., Seddigh, M., Zunic, J.: Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. IEEE Trans. Parallel Distrib. Syst. **13**, 14–25 (2002)
20. Whitbeck, J., de Amorim, M.D., Conan, V., Guillaume, J.: Temporal reachability graphs. In: 18th Annual International Conference on Mobile Computing and Networking, Mobicom 2012, 22–26 August, pp. 377–388 (2012)

Lightweight Verifiable Auditing for Outsourced Database in Cloud Computing

Mayank Kumar[1](✉) and Syam Kumar Pasupuleti[2]

[1] IIT (BHU) Varanasi, Varanasi 221005, Uttar Pradesh, India
mayank.kumar.cse15@iitbhu.ac.in
[2] IDRBT, Hyderabad 500057, India
psyamkumar@idrbt.ac.in

Abstract. Database outsourcing in Cloud Computing enables the data owner to store the data on cloud and assign the management to a cloud service provider (CSP), which provides various cloud services to the users. However, outsourcing database to cloud poses many challenges. One of the major concerns is confidentiality of the data. The general approach to tackle this issue is by encrypting the database before outsourcing, this helps in protecting confidentiality but poses a new problem of verifying the search results. We propose a lightweight verifiable auditing scheme for secure outsourcing of database, which encrypts the database and verifies search results with both parameters of correctness and completeness. We design our scheme based on a lightweight homomorphic encryption scheme (LHE) and efficient authenticated data structures to ensure the confidentiality and integrity of the database respectively.

Keywords: Cloud computing · Database
Lightweight homomorhic encryption · Embedded Merkle B tree

1 Introduction

The outsourced database (ODB) paradigm has numerous benefits but also suffers from security challenges. The outsourced database may contain sensitive information so, confidentiality of outsourced data is a big challenge. The practical solution for confidentiality of data is to encrypt the data using traditional encryption techniques but it is difficult to search over the encrypted data. Secondly, verifying of search results is a challenging problem, because semi-honest provider may return wrong or incomplete results to users i.e. when a user sends a search request to the CSP, he might return an empty result without even computing it or return a partial result instead of a complete one and use the computational resources elsewhere. The important characteristics of a good ODB paradigm also include integrity along with confidentiality. Thus, verifying outsourced database efficiently and effectively by satisfying completeness and correctness properties is a matter of concern.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 280–287, 2018.
https://doi.org/10.1007/978-3-319-72344-0_23

Recently, several researchers have studied on verifying outsourced databases. Most of the of the schemes [1–6] focus on generic solutions. An efficient method is required for verifiable auditing scheme for outsourced databases with full support for completeness and correctness properties. The biggest obstacle is to securely outsource the expensive computations. As for storage, any globally accepted encryption scheme will work well enough to prevent the curious CSP to peek in the database, but making it compute on that data without revealing any of it to the CSP was a tough target to shoot. We propose a lightweight verifiable auditing scheme for outsourced database based on homomorpic encryption [7], bloom filter and Merkle embedded B-tree. Our main contributions are as follows:

- A lightweight verifiable auditing scheme for outsourced database in cloud computing environment which achieves confidentiality and integrity of data.
- Our scheme employs lightweight homomorphic encryption to encrypt database that allows performing SQL operations efficiently.
- Further, we have used Embedded Merkle B tree (EMBT) and Bloom Filter (BF) to authenticate and verify the search results with correctness and completeness properties. The scheme verifies even the empty result set sent by the CSP.

2 Problem Formulation

2.1 System Model

We work on a system model which is used by the recent schemes [1]. There are four entities namely, the data owner, users, cloud service provider (CSP) and arbitrament center (AC). Their relation has been described in Fig. 1 and their roles are described below:

- **Data Owner:** This entity who owns the data and wants to outsource the database to the cloud.
- **User:** The user who needs to access the database and its services.

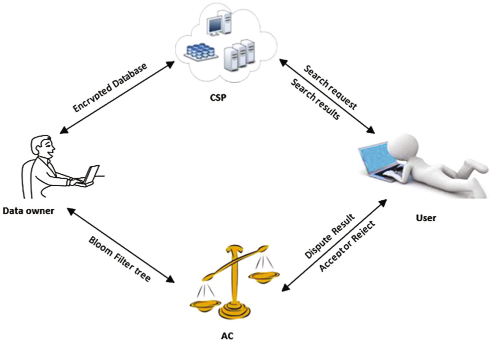


Fig. 1. The data outsourcing model

- **CSP:** This entity which is responsible for storing and processing the data.
- **AC:** The trusted third party which comes into play only when there is a dispute between data owner and CSP.

2.2 Threat Model

During this work, at every point the CSP is always considered a honest-but-curious server. The CSP is always curious to look into our data and hence it must be encrypted. The semi-honest part means the CSP can violate the protocols set by us in order to save its computational and storage resources and hence return fraction of the result set [8]. The communication channels between the CSP, data owner and the users can be insecure too. Hence broadly two types of attacks are possible: (i) Internal attacks which are generally executed by the CSP in search of benefits. (ii) External attacks which are mainly caused by unauthorized or revoked users through public channels in search of sensitive data.

3 Proposed Scheme

The proposed scheme consists of five phases: system setup, data outsourcing, data retrieving, verification and decryption.

3.1 System Setup

In this phase, we setup the necessary parameters for later use as follows:

The data owner generates single key (K(m)) for encryption, a tuple of three components, K(m) = $(\varGamma, \varTheta, \varPhi)$. The first component $\varGamma$ is a list of $[(k_1, s_1, t_1)..,(k_m, s_m, t_m)]$ where k_i, s_i and t_i are random real numbers. To ensure that the scheme works every time correctly we have to make sure $m \geq 4$, $k_i \neq 0$ for $1 \leq i \leq m-1$, $k_m + s_m + t_m \neq 0$ and $t_i \neq 0$ for $1 \leq i \leq m-1$. The component $\varTheta$ is a pair of two random real numbers θ_1 and θ_2. The component $\varPhi$ is also defined as a list, $\varPhi = (\varPhi_1, \varPhi_2, \varPhi_2)$ where $\varPhi_j$ is a m dimensional vector $(\phi_{\mathrm{j1}}, ..., \phi_{\mathrm{j}m})$ for $1 \leq j \leq 3$.

- Uniformly sample m random real numbers $r_{j1}, ..., r_{jm}$. The numbers can be arbitrarily large.
- Compute $\phi_{\mathrm{j1}} = k_1 * t_1 * \theta_j + s_1 * r_{\mathrm{jm}} + k_1 * (r_{\mathrm{j1}} - r_{\mathrm{j(m-1)}})$
- Compute $\phi_{\mathrm{ji}} = k_i * t_i * \theta_j + s_i * r_{\mathrm{jm}} + k_i * (r_{\mathrm{ji}} - r_{\mathrm{j(i-1)}})$ for $2 \leq i \leq m-1$
- Compute $\phi_{\mathrm{jm}} = (k_m + s_m + t_m) * r_{\mathrm{jm}}$

After key generation we initialize a bloom filter (BF) having k hash functions, followed by all the attribute values getting inserted into the bloom filter. Once this is complete the data owner generates a bloom filter tree to store all the BFs in a schematic order.

3.2 Data Outsourcing Phase

The data owner outsources the database by encrypting the data and uploading all the tuples along with an index for referencing it in the future by the users. We generate index, I_i for total number tuples, denoted by n_i, in which the value of the attribute column A_i is equal to a_i. For each element a_i in tuple compute the corresponding hash value as $h(a_i||n_i)$. Then construct the MBT based on hash values for the whole database. The data owner then encrypts every element a_i of the data using lightweight homomorpic encryption scheme as $c_i = $Enc(K(m), a_i) and uploads the encrypted value with their index for every tuple r as

$$r_e = (I_1, c_1, h(a_1||n_1), ..., (I_n, c_n, h(a_n||n_n)$$

Along with the data the data owner then uploads the signature S on the root, $h(r)$ of the EMB tree and the corresponding BF. The BF is also made valid by adding a signature to it. The BFT is uploaded to the Arbitration Center, which is used only in case of solving disputes.

3.3 Data Retrieving Phase

We define this section with a simple example of SELECT query.

SELECT * from T where c = v

where the value v is in encrypted form. The user generates the query with all the data records in the encrypted form as shown in the SELECT example above where the data field "v" is encrypted. The CSP receives the query and the CSP checks each tuple. It returns the corresponding result along with the index and authentication data structures. As for the example above it will simply check all the entries which match the value v. The user receives the result and checks for it validity using the signatures and EMB tree.

3.4 Verification Phase

The performance of the scheme on the metric of integrity is evaluated in terms of both, correctness and completeness. After the verification, the user sends accept or reject, that decides whether the CSP was honest or malicious in the transaction.

- **Correctness**: It must be ensured that the result is not tampered by the CSP at any point. There are two cases which needed to be checked in this category. When the result set is empty i.e. CSP claims that the outsourced database contains no matched tuples, the CSP is supposed to return the bloom filter to user. Using the bloom filter, the user can easily verify the correctness of search result. When the result set contains some tuples, the user performs the correctness check using the EMB tree. The user checks if any tuple in the search result has been tampered with. If all the tuples of the result are genuine then the user proceeds forward for checking the completeness of the result.

- **Completeness**: Once the user verifies that the result set returned by the CSP is correct then he has to ensure that it is also complete. The proposed solution can achieve the completeness of the solution using the EMB tree i.e. we can check the neighbouring values and hence verify whether the search result returned by the CSP is complete or not.

3.5 Decryption Phase

After successful verification,if search results are valid then user decrypts the data as follows:

The decryption algorithm denoted by Dec(K(m), $(e_1, ..., e_m)$) gives the plaint text 'v' for the given cipher-text $(e_1, ..., e_m)$ in the following steps.

- T = $\sum_{i=1}^{m-1} t_i$
- S = $e_m/(k_m + s_m + t_m)$
- v = $(\sum_{i=1}^{m-1}(e_i - S * s_i))/T$

The conditions $m \geq 4$, $k_i \neq 0$ for $1 \leq i \leq m-1$, $k_m + s_m + t_m \neq 0$ and $t_i \neq 0$ for $1 \leq i \leq m-1$ ensure the validity of the above decryption scheme.

4 Security Analysis

4.1 Confidentiality

One of the key parameters in any outsourced database architecture is confidentiality, it must always be ensured. Majority of the databases around the globe are filled with personal data of organizations and individuals and if leaked can easily lead to identity theft on a large scale. Hence it is the one the most important metric to evaluate any data outsourcing paradigm.

Theorem 1. *The proposed scheme is secure on parameters of data confidentiality.*

Proof. To search tuples satisfying the value of attribute A_q = a the user sends a simple SQL query to the CSP, to return all the tuples with $A_q = (Enc(K(m), a))$. The CSP neither needs any part of the plain-text values nor the encryption key to do its part of computations and return the search results and can never know what is inside the database.

The scheme we propose uses homomorphic encryption and is secure from cipher text-only attacks. Also, the pseudo-random number generator function $f_k()$ has negligible probability of collision and the linear combination $f_k(i)$ & $f_k(j)$ such as $f_k(i) + f_k(j)$ and $f_k(i) - f_k(j)$ are also pseudo-random and cannot be guessed by the CSP or anyone else. Hence we can say that proposed scheme is secure in terms of data privacy and can achieve the corresponding level of confidentiality if the encryption key is strictly limited to the data owner and authorized users only.

4.2 Integrity

Integrity of the data as a parameter ensures that changes to the data (adding or deleting) are done only by the authorized agents such as data owner or users, in an authorized way. From the cloud computing scenario, we have to prevent the CSP to make any changes to the database. A dishonest CSP, to save its storage, might keep on deleting the database in small fragments without the knowledge of the data-owner.

Theorem 2. *The proposed scheme achieves the integrity of search results.*

Proof. This proof involves verifying the completeness and correctness of the search results as follows:

For evaluating the correctness of the result, we consider two possibilities. First, if the result set returned by the CSP is empty, the CSP must return the corresponding bloom filter as a *proof* to the user. Then the user checks if BF(a)=1 to verify the validity of the CSP claim. For additional security, the data owner signs the bloom filter using a secure signature algorithm such as RSA or ElGamal.

Second, if there are some tuples in the result, then user can check the correctness of the cipher-text $(e_1, ..., e_m)$ in result tuples. Since the encryption scheme is secure from ciphertext only attacks, so an attacker cannot forge a valid ciphertext result without the knowledge of the encryption key. The correctness of the search result tuple is based on the assumption of the collision resistance in the hash functions used while creating the EMB tree.

Once the user has checked the result for correctness, the completeness can be easily verified using the fact that the number of tuples n_i is involved in the attribute hash value $h(a_i||n_i)$. The user with the help of EMB tree can prove that the every tuple in the results satisfies the query condition.

5 Performance Analysis

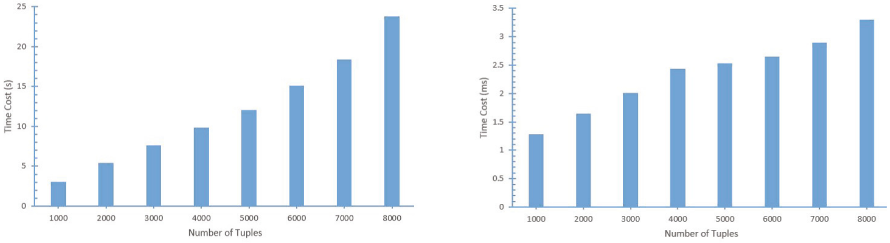
This section includes the experimental evaluation of our proposed scheme. The experiments are performed and verified both on a local machine and in openstack cloud environment. The cloud server equipped with Intel core-i7, 16 GB of memory and the local machine has python 3.6 running on Intel Core-i5 5200U 2.20 GHz CPU with 8 GB of memory. In order to find computational overheads which didn't include the CSP and client communication time, we present the results of local machine. In order to make the result easy to understand, we divide it into sections similar to the proposed scheme viz. System Setup, data outsourcing and retrieving, and data verifying.

5.1 System Setup

This phase primarily consists of key generation and bloom filter setup. The time to calculate all the hash functions is the significant part of the computation overhead in this phase. In our experiments we have fixed the value of $k = 14$ and vary the size of the bit-array such that its ratio to the number of tuples is always 32. We have maintained the false probability $(P_f \leq 2^{-20})$ to always be negligible. We have varied the number of tuples from 1000 to 8000 for results. We observed that the relation between computation cost and the number of tuples is almost linear. The results indicated that the setup is acceptable, since for as large as 8000 tuples, we make it under 350 ms.

5.2 Data Outsourcing and Retrieving

In this section, we discuss computational cost of data outsourcing by the data owner and then data retrieving by the users.



(a) Time cost for data outsourcing

(b) Time cost for data retrieving

Fig. 2. Performance results for data outsourcing and retrieving

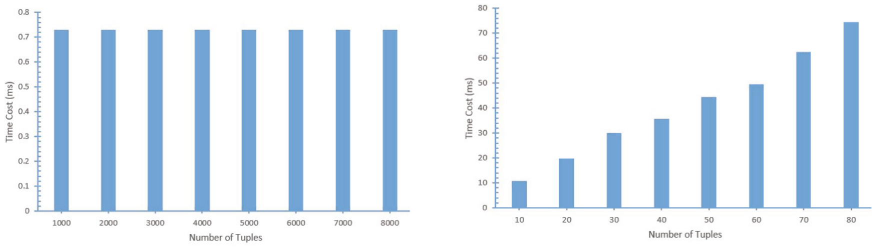
For outsourcing the data, owner has to encrypt every tuple and then create the EMB trees for them. We can minimize the number of hashes using the EMB tree but the signatures on each verification object (VO) still needs to be done, accounting to n signatures. This obviously makes the data outsourcing phase relatively slower to the data retrieving phase, but its a one time process and hence can be accepted. Moreover, from the graph in Fig. 2(a) one can see that we have managed to keep the cost less than 25 s for 8000 tuples, which is pretty decent.

For retrieving data, the user has to encrypt its plain-text value of the query and then upon receiving the request the CSP just searches for any matching tuples and returns the results. The user then just has to decrypt the results to get the required values. This way a lot of time is saved on the CSP side, resulting in better computational time results as shown in Fig. 2(b).

5.3 Verifying

For the purpose of verification, we have considered two cases throughout, and hence we will evaluate them separately.

Case 1: When the CSP claims that no result was found then it is bound to return the corresponding bloom filter. The user has to verify that whether bloom filter is genuine or not, using signatures. The task is independent of the total number of tuples and hence gives constant result as in Fig. 3(a).



(a) Time cost in some result case

(b) Time cost in some result case

Fig. 3. Time cost in no result case

Case 2: When the user gets some result sets then she can check the integrity of the result by verifying through the EMB tree returned for every tuple. Also the signature on the VO has to be verified. The time cost varies linearly as shown in Fig. 3(b).

References

1. Wang, J., Chen, X., Huang, X., You, I., Xiang, Y.: Verifiable auditing for outsourced database in cloud computing. IEEE Trans. Comput. **64**(11), 3293–3303 (2015)
2. Narasimha, M., Tsudik, G.: Authentication of outsourced databases using signature aggregation and chaining. In: Li Lee, M., Tan, K.-L., Wuwongse, V. (eds.) DASFAA 2006. LNCS, vol. 3882, pp. 420–436. Springer, Heidelberg (2006). https://doi.org/10.1007/11733836_30
3. Hacigümüş, H., Iyer, B., Mehrotra, S.: Executing SQL over encrypted data in the Database-Service-Provider model. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2002, pp. 216–227 (2002)
4. Pang, H., Zhang, J., Mouratidis, K.: Scalable verification for outsourced dynamic databases. Proc. VLDB Endowment **2**(1), 802–813 (2009)
5. Li, F., Hadjieleftheriou, M., Kollio, G., Reyzin, L.: Dynamic authenticated index structures for outsourced databases. In: SIGMOD 2006, 27–29 June 2006, Chicago, Illinois, USA (2006)
6. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and integrity in outsourced databases. ACM Trans. Storage **2**(2), 107–138 (2006)
7. Liu, D., Wang, S., Zic, J.: Privacy and integrity of outsourced data storage and processing. In: Big Data: Storage, Sharing, and Security (3S)
8. Chai, Q., Gong, G.: Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers. In: Proceedings of the IEEE International Conference on Communications, ICC 2012, pp. 917–922 (2012)

A One Phase Priority Inheritance Commit Protocol

Sarvesh Pandey(✉) and Udai Shanker

Computer Science & Engineering Department,
Madan Mohan Malaviya University of Technology, Gorakhpur, India
pandeysarvesh100@gmail.com, udaigkp@gmail.com

Abstract. High priority two phase locking (HP2PL) concurrency control algorithm can be used for accessing of data items to resolve conflicts amongst the concurrently executing transactions. Inclusion of priority inversion and data inaccessibility are the most undesirable problems in transactions' execution which seriously affect the system performance. Previously developed protocols for resolving such issues put a lot of messages and time overhead which is not desirable. In distributed real-time database system (DRTDBS), basic aim is to minimize the number of transactions missing their deadline. This can be achieved by minimizing commit time. In this paper, A One Phase Priority Inheritance Commit Protocol (OPPIC) has been proposed specifically reducing one round of message transfer among coordinator and participating cohorts' sites in case of priority inversion problem at any of the cohort of low priority transaction. Focus of this protocol is to minimize the priority inversion duration that in turn minimizes commit time. A distributed database system is simulated for measuring the performance of this protocol with 2PC and PIC protocols. The results confirm the significant improvement in system performance with the OPPIC protocol.

Keywords: Concurrency control · Commit protocol · Priority inheritance
Priority inversion · Distributed real-time database systems

1 Introduction

Today, database systems (DBS) become an indivisible part of our daily life [1]. It is a logical collection of data items shared over several sites [2–4]. We can classify DBS as Centralized DBS and Distributed DBS. A centralized DBS consists of single site, while a distributed DBS (DDBS) consists of multiple sites connected through a network to facilitate communication between them [5]. Various DDBS issues like transaction commit and concurrency control protocols, caching, replication [6] etc. have been widely discussed and protocols proposed to address them. However, these protocols are not appropriate in the real-time environment due to associated time constraint with transactions. In Real Time Systems (RTS), the correctness is determined by the logical as well as temporal properties of the result [7, 8]. Henceforward, actions performed in RTS must be temporally valid; otherwise it may lead to the catastrophic results. A join of DDBS and RTS gives birth to the new area of research, named as DRTDBS.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 288–294, 2018.
https://doi.org/10.1007/978-3-319-72344-0_24

DRTDBS is a collection of numerous logically interrelated database systems connected via a communication network specifically designed to serve the purpose of real time systems [9, 10]. They support transactions having explicit time constraints. Transaction timing constraint is represented as deadline, which shows that it must complete before that specific time in future [11, 12]. Distributed real-time transactions (DRTT) can be categorized as hard, firm and soft. See [4, 13] for details.

Although, the deadline of transaction is difficult to meet because of several reasons such as site failure, data conflicts, communication delays etc., data conflicts between transactions become a major factor responsible for degradation in system performance. Data conflicts can occur in two ways; first between two transactions in execution period (executing-executing conflict) and second between one transaction in the execution period and other in the committing period (executing-committing conflict) [14]. In literature, the issue of handling executing-executing data conflicts is discussed in detailed; however, the issues of handling executing-committing data conflicts between the transactions have got very little attention. Executing-committing conflict between the transactions is one of the most undesirable situations in firm DRTDBS. It sometimes gives birth to the problem of priority inversion. Priority inversion problem occur in the system, if low priority distributed transaction blocks a high priority distributed transaction as a result of resource conflict (data and/or CPU) [15, 16].

PIC protocol [17] is virtually identical to 2PC protocol. It has many problems. The first problem is that although the priority inversion is bounded, it can be extreme depended on the number of transactions and locks in a specified system. The second problem with PIC protocol is that it may suffer from deadlock. PIC protocol also has a drawback of making the database inconsistent in case of multiple read access followed by write access in conflicting mode. One more problem with PIC protocol is that when the number of priority inversions increase in DRTDBS, number of transactions with inherited priority will also get increased. Increase in the priority of transaction affects other concurrently executing transactions. Now, a transaction with its inherited higher priority will further compete to access system resources such as data items, CPU, I/O etc. with other already existing higher priority transactions [12]. This in turn reduces the higher priority transactions' performance with the increase of every new transaction undergoing priority inheritance because of priority inversion. With increase in number of priority inversions, competition to access data items would also get increased. As a result, data conflict percentage will increase which negatively affects system's performance.

PIC protocol [17] requires two rounds of message transfer for processing of priority inheritance information which is an extra overhead. One more problem with PIC protocol is the delay in priority inheritance information dissemination to brotherly cohorts at remote site. Proposed OPPIC protocol reduces the delay up to half by requiring a single round of message transfer. Section 2 discusses proposed OPPIC protocol in detail. The performance study presented in Sect. 3 shows significant performance improvement in OPPIC protocol over PIC protocol. Section 4 concludes the paper.

2 A One Phase Priority Inheritance Commit Protocol

OPPIC protocol is an extension to the PIC protocol which is an alternative of the PROMPT protocol [17]. However, in PROMPT it is said that PIC protocol is not performing well in distributed real-time environment; OPPIC protocol based on it, gives significantly better result.

In PIC Protocol [17], if prepared cohort C_a of low priority transaction (T_L) blocks the data item required by a requesting cohort C_b of a newly arrived high priority transaction (T_H), then priority of T_L is upgraded to that of T_H. Note here that C_a and C_b are the conflicting cohorts of T_L and T_H respectively. For inheriting the priority, cohort C_a of low priority transaction sends PRIORITY-INHERIT message to its coordinator. The coordinator, in sequence, sends this PRIORITY-INHERIT message to all the other participating cohorts, and after that, all further processing associated with T_L takes place at inherited priority. It is claimed that performance of PIC protocol is virtually identical to that of 2PC protocol [17]. In PIC protocol, dissemination of the priority inheritance information to the brotherly cohorts requires two round of message delays, which is an extra overhead. This is the major reason behind no any significant performance benefits of PIC protocol over 2PC protocol. Specially, delay in the priority inheritance information dissemination to brotherly cohorts at remote sites negatively affects PIC protocol.

As opposite to PIC protocol, proposed OPPIC protocol says that when priority inversion occurs at cohort C_a of T_L then it directly sends PRIORITY-INHERIT message to all the participating sites including coordinator in parallel fashion. It is helpful in dealing with priority inversion problem in following aspects:

I. There is no need of two rounds of message transfer between participants (Coordinator and Cohorts). Only single round message transfer is required.
II. All Cohorts receive the PRIORITY-INHERIT message in about half-time duration compared to the base PIC protocol.
III. It significantly minimizes the overall distributed firm real-time transaction completion time, a most critical resource in DRTDBS.

OPPIC protocol eliminate one phase of message transfer, and thereby improves system's performance. The pseudo code for OPPIC protocol is,

```
If (Priority ($C_a$) < Priority ($C_b$))
{
 Send PRIORITY-INHERIT message to all participants (including Coordinator) of
 $T_L$ in parallel fashion.
}
Else Cohort $C_a$ of $T_L$ commit as usual.
```

To gain performance benefits, OPPIC protocol allow communication among cohorts of same transaction as opposed to the base PIC protocol.

3 Performance Study

In DRTDBS research community, there is no hands-on benchmark available to assess the performance of the proposed protocol [19]. Therefore, a DRTDBS including N sites [13, 15, 20, 21] was simulated in accordance to the environment assumed in earlier studies [18, 13]. Table 1 presents different parameters used in simulation study with their default values.

Table 1.

Parameter	Meaning	Default setting
DB_{Size}	Size of Database (No of Pages in databases)	200 data objects/site
Ndb	No of database Sites	6
AR	Transaction arrival rate per site	0–4 transactions/sec (Uniform Distribution)
Tcom	Communication delay among transactions	Either 0 ms or 100 ms
Nop	No of operations in transaction	4–20 (Uniform Distribution)
SF	Transaction Slack Factor	1–4 (Uniform Distribution)
P(w)	Probability of write operation	0.60
CPU_{page}	Processing time required for accessing CPU page	5 ms
$Disk\text{-}_{page}$	Processing time required for accessing Disk page	20 ms

We ensured significant level of resource and data contention during performance study. Earliest Deadline First (EDF) is used as the cohort's priority assignment policy for performance study of OPPIC protocol. As per EDF, a transaction with closest deadline is assigned a highest priority in system. In case of tie, we assign priority to the transaction using FCFS scheme. The performance of OPPIC protocol is measured based on the number of transactions missing their deadline as given below.

$$\boldsymbol{Kill\ percent} = \frac{\textbf{Number of transaction aborted}}{\textbf{Total number of transactions in system}}$$

Our work is an extension to PIC protocol. Therefore, we compared the OPPIC protocol with following protocols: 2PC protocol and PIC Protocol. Figures 1, 2, 3 and 4 show the transaction kill percent at communication delay of either 100 ms or 0 ms in disk resident databases with different transaction arrival rates. Results shows that proposed protocol performs better than 2PC and PIC protocol under all load conditions. Performance improvements are mainly because of reduction of one phase in disseminating priority inheritance message to all other participating cohorts of a transaction in case of priority inversion.

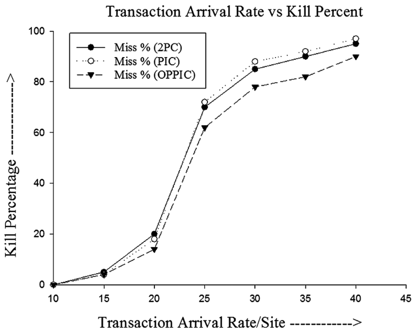


Fig. 1. Transaction Kill Percentage with Resource and Data Contention at 0 ms communication delay under normal and heavy load

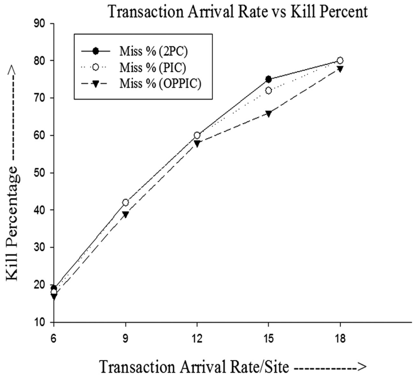


Fig. 3. Transaction Kill Percentage with Resource and Data Contention at 0 ms communication delay under heavy load

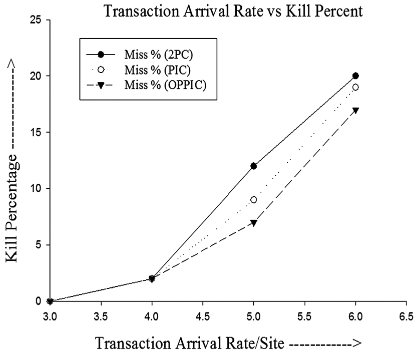


Fig. 2. Transaction Kill Percentage with Resource and Data Contention at 0 ms communication delay under normal load

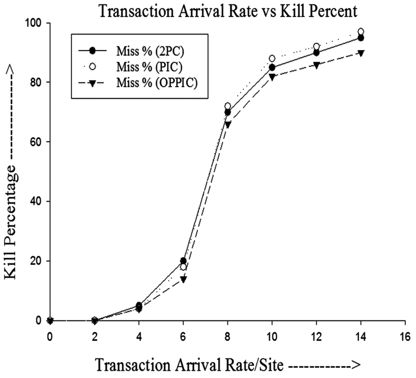


Fig. 4. Transaction Kill Percentage with Resource and Data Contention at 100 ms communication delay under normal and heavy load

4 Conclusions

Proposed OPPIC protocol is based on the priority inheritance concept. It overcomes the problem of execute-commit conflict up to some extent. In this protocol, there is no need of two rounds of message transfer between participants (Coordinator and Cohorts) in case of priority inversion; only a single round message transfer between participants is required. Here, all cohorts receive PRIORITY-INHERIT message in about half-time interval compared to the base PIC protocol. In other words, in case of priority inversion, it significantly minimizes the overall firm distributed real-time transaction completion time which is the most critical resource in distributed real-time environment.

Further, this protocol significantly reduces the commit processing time, and thereby minimizes the number of transactions missing their deadline.

Acknowledgments. We acknowledge the financial support provided by the Council of Scientific and Industrial Research (CSIR), New Delhi, India under grant no 1061461137 during this research work.

References

1. Pandey, S., Shanker, U.: Transaction execution in distributed real-time database systems. In: Proceedings of International Conference on Innovations in information Embedded and Communication Systems, pp. 96–100 (2016)
2. Elmasri, R., Navathe, S.B.: Fundamentals of Database Systems. Pearson, Boston (2015)
3. Ramakrishnan, R., Gehrke, J.: Database Management Systems. McGraw Hill, Berkeley (2000)
4. Shanker, U., Misra, M., Sarje, A.K.: Hard real-time distributed database systems: future directions. In: Proceedings of All India Seminar on Recent Trends in Computer Communication Networks. Department of ECE, IIT Roorkee, India, 7–8 November, pp. 172–177 (2001)
5. Garcia-Molina, H., Lindsay, B.: Research directions for distributed databases. ACM SIGMOD Rec. **19**(4), 98–103 (1990)
6. Pandey, S.: Changing trends in replicated database systems. Int. J. Sci. Res. Dev. **5**(4), 463–465 (2017)
7. Aldarmi, S.A.: Scheduling soft-deadline real-time transactions. Ph.D. thesis, University of York (1999)
8. Ellis, C.A., Muth, P., Rakow, T.C., Neuhold, E.J.: Engineering, vol. 14, no. 1 (1991)
9. Shanker, U., Misra, M., Sarje, A.K.: Distributed real time database systems: background and literature review. Int. J. Distrib. Parallel Databases **23**(2), 127–149 (2008). Springer Verlag
10. Gruenwald, L., Chen, Y.: Research issues for a real-time nested transaction model. In: Proceedings of the IEEE Workshop on Real-Time Applications. IEEE Computer Society, Washington, D.C. (1994)
11. Agrawal, D., El Abbadi, A., Jeffers, R., Lin, L.: Ordered shared locks for real-time databases. VLDB J. Int J. Very Large Data Bases **126**, 87–126 (1995)
12. Kao, B., Garcia-Molina, H.: An overview of real-time database systems. Real Time Comput. **127**, 261–282 (1993). Springer, Berlin, Heidelberg
13. Lam, K.Y.: Concurrency control in distributed real time database systems, Ph.D. thesis, City University of Hong Kong (1994)
14. Shanker, U., Misra, M., Sarje, A.K.: SWIFT - a new real time commit protocol. Distrib. Parallel Databases **20**(1), 29–56 (2006)
15. Ulusoy, O.: A study of two transaction-processing architectures for distributed real-time data base systems. J. Syst. Softw. **31**(2), 97–108 (1995)
16. Levine, G.: Priority inversion with fungible resources. ACM SIGAda Ada Lett. **31**(2), 9–14 (2012)
17. Haritsa, J.R., Ramamritham, K., Gupta, R.: The PROMPT real-time commit protocol. IEEE Trans. Parallel Distrib. Syst. **11**(2), 160–181 (2000)
18. Qin, B., Liu, Y.: High performance distributed real-time commit protocol. J. Syst. Softw. **68**(2), 145–152 (2003)

19. Lee, V.C.S., Lam, K.-Y., Kao, B.: Priority scheduling of transactions in distributed real-time databases. Real-Time Syst. **16**(1), 31–62 (1999)
20. Lee, V.C.S., Lam, K.W., Hung, S.L.: Concurrency control for mixed transactions in real-time databases. IEEE Trans. Comput. **51**(7), 821–834 (2002)
21. Ulusoy, Ö., Buchmann, A.: A real-time concurrency control protocol for main-memory database systems. Inf. Syst. **23**(2), 109–125 (1998)

Issues and Challenges in Big Data: A Survey

Ripon Patgiri(✉)

Department of Computer Science and Engineering,
National Institute of Technology Silchar, Assam 788010, India
ripon@cse.nits.ac.in
http://cse.nits.ac.in/rp/

Abstract. Undoubtedly, the Big Data is the most promising technology to serve an organization in a better way. It provides an organized way to think about data, whatever the data size is, and whatever the data type is. Moreover, the Big Data provides a platform to make decisions, and to analyze future possibilities using the past and present data. The Big Data technology eases the large dataset to store, process and manage. The Big Data is the most fashionable trendsetter in the world of computing i.e., the most popular buzzword around the globe upon which the future of the most of IT industries depends on it. This paper presents a study report on numerous research issues and challenges of Big Data which is employed in very large dataset. This paper uncovers the nuts and bolts of Big Data. This study report provides rich insight on the Big Data.

Keywords: Big Data · Big Data survey · Big Data paradigm
Issues and challenges · Big Data analytics · Big Data security
Healthcare

1 Introduction

The Big Data is not only the most promising technology, but also a social necessity to reclaim their lifestyle. Facebook, for instance. Therefore, data are increasing at an exponential pace. The Big Data is a high volume of data, and Seife quotes this data-ism as, "Data-ism is very much a conventional business book, full of anecdotes, mini-profiles and aphorisms that grow ever less compelling [24]". The Big Data concerns about assigning worth to those high volume of data. The Big Data comprises of unstructured, semi-structured and structured data. The Big Data is proven as a game changer in many data-intensive field. The Big Data enhance the decision making process automatically [16]. *A wrong decision can destroy an organization.* That's why, Big Data Analytics evolves to assist and guide the decision-making process. The data-driven decision-making process is the most crucial and critical part of an organization to make a move [26]. However, the Big Data face the "curse of dimensionality" issue. Many new services evolve based on Big Data, namely, Big Data as a Service, Big Data Security as a Service, Big Health Data as a Service, and Big Data Analytics as a Service.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 295–300, 2018.
https://doi.org/10.1007/978-3-319-72344-0_25

Besides, the Big Data has nothing untouched area, namely, engineering, science, government, economy, and environment.

Interestingly, Einav et al. [11] describes the role of Big Data in economic growth. McAfee et al. [21] emphasize more on the generation of business revenue using Big Data. Therefore, there are numerous field which smoothens by Big Data technology, namely, eHealth, Wearable technology, customization of Medicine, Internet of Things (IoT), customer analytics solution, surveillance system, transport system, Digital experience solution and Power/Energy. Bourne et al. [5] quote as, “the research community must find more efficient models for storing, organizing and accessing biomedical data”. The biocuration requires access to Big Data by both biomedical and biological discoveries [14]. Therefore, big interdisciplinary data are also growing. These technologies produce an enormous volume of data. The well-known data-intensive fields are, namely, GIS, weather, earthquake, gnome, ocean, soil, oil, and drone. Therefore, computer scientists, physicists, economists, mathematicians, bio-informaticists, and sociologists use Big Data for various purposes. The government and private sectors are the key areas to use Big Data analytics. Thus, a massive amount of data is spawned excessively with the course of time, and therefore, these data are cumbersome to store, process, visualize and analyze the data in conventional ways. Most of these data are from different fields, and these are unstructured. Therefore, interdisciplinary research on Big Data is prominent challenge and opportunity nowadays.

2 Issues and Challenges in Big Data Analytics

Big Data analytics is a method of logical analysis on a very large dataset. There are numerous purposes of practicing Big Data analytics (BDA) and results enormous possibilities in research. For example, BDA leads to better healthcare [22] which is the most prominent research challenge nowadays. BDA is used in decision support system in healthcare for a better outcome, prevention, low-cost solution and early detection of an event. The BDA dig into the data for new facts or insight. The first key challenge of BDA is the monitoring real-time events with high volume, and to explore the available high volume of data. The second key challenge of BDA is the prediction of future based on Big Data using machine learning algorithms. The BDA is deployed in the business organization to know the behavior of their business and to know the future using their own data set. The third key challenge is the decision making process using a very large data set which requires BDA. The key challenges of the BDA is to learn the decisions and make a right decision based on huge volume of the dataset. The BDA suggests a good solution among many solutions in large dataset. The fourth key challenge is the diagnostic analytics of BDA to discover about past performance/events. Fifth, uncover the hidden patterns of the past events is also a challenge. The BDA recognizes the behavior of users and data to detect anomalies. Finally, the most promising challenge is the anomaly detection, intrusion detection, and fraud detection over a very large dataset. This is a tough challenge to achieve.

3 Issues in Big Data Security

Protecting and securing data is the top priority of the Big Data paradigm. Moreover, Big Data is used to detect security threats. On the contrary, the Big Data is also used to detect the breach of a system to attack/test. The Big Data eases the capability of securing data, protecting data and ensuring the privacy of data. The Big Data analytics address data security, technology to keep customers data private, provenance, data transparency, performance benchmarking, data and system interoperability. There are two aspects of Big Data security (BDS), namely, Big Data for security and Security for Big Data. Moreover, secure infrastructure, secure data management, data privacy, and real-time security are some example of BDS [12]. The requirement of BDS is confidentiality, authenticity, integrity and availability (-service should not down due to DDOS) [3]. Moreover, BDS reduces the breach of risk sensitive data. Cloud Security Alliance (CSA) categorizes BDS as infrastructure security, data protection, data management and reactive security [2,13]. However, CSA enlisted top ten challenges in Big Data Security [13], namely, (a) secure computations in distributed programming frameworks, (b) security best practices for non-relational data stores, (c) secure data storage and transaction logs, (d) endpoint input validation/filtering, (e) real-time security/compliance monitoring, (f) Scalable and composable privacy-preserving data mining and analytics, (g) cryptographically enforced access control and secure communication, (h) granular access control, (i) granular audits, and (j) data provenance.

The BDA is used to detect of anomaly, intrusion, fraud, and advanced persistent threats (APT) [6]. It is impossible to spell-check the large size security analysis in a conventional way. The security of Big Data data has been achieved by deploying BDA on the large sized logs, system events, network traffic, website traffic, security information and event management (SIEM) alert, cyber attack patterns, business processes and other information sources. Besides, access control of the billion users is the perplex job [3]. It is an open challenge to protect data from malicious attackers. The diversity of data sources, data formats, streaming of data and infrastructures can lead to security vulnerabilities.

4 Open Challenges

The challenges of Big Data are outlined below-

(a) The Big Data is really big enough to transmit data from one source to another. (b) A large dataset is difficult to visualize, very tough to mine a meaningful information, and perplex to manage these data. (c) These huge sets of data consist of structured data, unstructured data, and semi-structured data. The key issue is the various sources of data, which forms various kinds of data. Storing these data heterogeneity itself a great challenge. (d) The real-time Big Data processing is a big challenge. (e) It is a challenge to make a correct decision using the large dataset. (f) The efficient visualization of data is an open challenge [15]. (g) The "pay-as-you-go" model helps in decreasing the costs of users. The Big Data as a Service and Big Data Analytics as a Service significantly reduces the costs. However, it is still a challenge in the Big Data paradigm for lowering the cost. (h) The most prominent issues in load balancing are heterogeneity, scalability, consistency, and adaptability. (i) The fault-tolerance system is the most cumbersome for administrator and fault cannot be obviated easily. The fault-tolerance model is implemented by RAID, replication, erasure coding, de-duplication, and journaling. (j) The Big Data technology requires auto-scalability with dynamic data size. The scalability is the big issue in the Big Data. The designing infinite scalability is the biggest challenge. (k) Another research challenge is the achieving high performance using the low-cost commodity hardware. (l) The key issue is dynamic volume and the technology requires to adjust itself to cope up with the ever changing environment. (m) The prominent issue is to design an automatic failover mechanism to ensure high availability. (n) The bandwidth is not unlimited to transfer data in a real scenario, thus, reducing bandwidth consumption a challenge. (o) Another issue is the ameliorating the throughput significantly. (p) The performance of a file system depends on the how minimal network traffic has generated. A fine tuning of network traffic is required to excel in performance in data-intensive computing. (q) The disaster recovery and management are the big issue and the big challenge for all time. The Disaster Recovery is the most cardinal part of data storage system. Disaster Recovery as a Service or Recovery as a Service is the most prominent emerging cloud model in disaster recovery. (r) The data acquisition is an issue of Big Data. Data does not come automatically, but user makes bigger database size. Either data is collected explicitly or implicitly, database size grows continuously. (s) The data curation of Big Data concerns with data reuse, data discovery, and data preservation, such that the value of the data is maintained over time [1,7]. Especially, the data curation in Big Data becomes more complex due to high volume.

5 Issues and Challenges in Big Data Applications

The Big Data is very complex to deploy in real system due to mammoth sized data, and moreover, it is continuously monitored, processed, and visualized. However, the data-intensive fields use Big Data technology to enhance their revenue and performance, like Biomedical engineering. Undoubtedly, the Big Data is a good choice for Biomedical engineering due to the massive amount of data to be analyzed [8]. The Big Biomedical Data Engineering (BBDE) requires huge storage spaces, processing capacities, visualization and analysis. The article [4] ask a question- “why do we write?”. The assumed environment may differ, but the answer is similar. However, the biomedical engineering requires the data to write, so that someone will use in future to study the diseases in the curing process. The answer converges with article [4]. Big Data Analytics (BDA) is a merger of Big Data and Analytics [9]. The analytics means the logical method of analysis. BDA provides a platform to discover the hidden jewels from data.

6 Discussion, and Future Direction

Big Data technology is developed to serve the billions of clients for the purposes of the generating revenue. The future of Big Data targets more on Interdisciplinary computing [23]. Lynch [18] quote as, "If data cannot survive in the short term, it is pointless to talk about long-term use". Bourne et al. [5] call for a more efficient way of storing, managing and processing the Medical data. Landhuis [17] reports the Neuroscience is another emerging field for Big Data because neuron size of any species is very big to store and process. Marx et al. [19] report that the Big Data is required in the cancer study. Nature [10] editorial quote as, "Health professionals will confront more data than do those in finance". Topol [25] quote as, "a massive, open, online medicine resource would help to quickly identify the genetic cause of the disorder". The Big Data is used to enhance the healthcare process [27]. Moreover, the NASA process petabytes of Climate data [20]. Another future agenda is the real-time processing of the monster size data [8]. The real-time Big Data processing is a very complex process. A strong programming paradigm is required to process real-time Big Data efficiently in the scale of infinite (Exabyte or beyond). Moreover, the Big Data span from pernicious project to constructive project. All people on the earth will engage with Big Data from 2020 and onward either directly or indirectly.

7 Conclusions

We have exposed the issues and challenges of Big Data. The key issues of Big Data are volume, velocity and variety. Moreover, security, privacy, adaptability, fault-tolerance, consistency, data curation, data acquisition, network traffic, bandwidth and latency, performance, scalability, load balancing are also some prominent issues of Big Data technology. The BDS and BDA also play vital role which is the prominent issues for many organizations for many years. We have also discussed that the direction of Big Data is moving towards "Interdisciplinary Big Data Computing" and "Very Big Data". The scope of Big Data is not limited to engineering, Science, environment, economic, biology, and agriculture. For example, the Big Data can be used in the medical domain, like cancer treatment, and brain analysis.

References

1. Abe, A.: Curating and mining (big) data. In: 2013 IEEE 13th International Conference on Data Mining Workshops, pp. 664–671 (2013)
2. Alguliyev, R., Imamverdiyev, Y.: Big data: big promises for information security. In: IEEE 8th International Conference on Application of Information and Communication Technologies (AICT 2014), pp. 1–4 (2014)
3. Bertino, E.: Big data - security and privacy. In: 2015 IEEE International Congress on Big Data, pp. 757–761 (2015)

4. Bonenfant, M., Desai, B.C., Desai, D., Fung, B.C.M., Ozsu, M.T., Ullman, J.D.: Panel: the state of data: invited paper from panelists. In: Proceedings of the 20th International Database Engineering & Applications Symposium, pp. 2–11 (2016)
5. Bourne, P.E., Lorsch, J.R., Green, E.D.: Perspective: sustaining the big-data ecosystem. Nature **527**(7576), S16–S17 (2015)
6. Cardenas, A.A., Manadhata, P.K., Rajan, S.P.: Big data analytics for security. IEEE Secur. Priv. **11**(6), 74–76 (2013)
7. Chen, C.P., Zhang, C.-Y.: Data-intensive applications, challenges, techniques and technologies: a survey on big data. Inf. Sci. **275**(2014), 314–347 (2014)
8. Cuzzocrea, A., Sacca, D., Ullman, J.D.: Big data: a research agenda. In: Proceedings of the 17th International Database Engineering & Applications Symposium, pp. 198–203 (2013)
9. Desai, B.C.: Technological singularities. In: Proceedings of the 19th International Database Engineering & Applications Symposium, pp. 10–22 (2015)
10. Editorial: The power of big data must be harnessed for medical progress. Nature, **539**(7630), 467468 (2016)
11. Einav, L., Levin, J.: Economics in the age of big data. Science **346**(6210), 1243089 (2014)
12. Fang, W., Wen, X.Z., Zheng, Y., Zhou, M.: A survey of big data security and privacy preserving. IETE Technical Review, pp. 1–17 (2016)
13. Big Data Working Group: Expanded top ten big data security and privacy challenges. Cloud Security Alliance, pp. 1–39, April 2013
14. Howe, D., Costanzo, M., Fey, P., Gojobori, T., Hannick, L., Hide, W., Hill, D.P., Kania, R., Schaeffer, M., St Pierre, S., et al.: Big data: the future of biocuration. Nature **455**(7209), 47–50 (2008)
15. Jagadish, H.V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., Shahabi, C.: Big data and its technical challenges. Commun. ACM **57**(7), 86–94 (2014)
16. Labrinidis, A., Jagadish, H.V.: Challenges and opportunities with big data. Proc. VLDB Endowment **5**(12), 2032–2033 (2012)
17. Landhuis, E.: Neuroscience: big brain, big data. Nature **541**(7638), 559–561 (2017)
18. Lynch, C.: Big data: How do your data grow? Nature **455**(7209), 28–29 (2008)
19. Marx, V.: Biology: the big challenges of big data. Nature **498**(7453), 255260 (2013)
20. Mattmann, C.A.: Computing: a vision for data science. Nature **493**(7433), 473475 (2013)
21. McAfee, A., Brynjolfsson, E., Davenport, T.H., Patil, D., Barton, D.: Big data: the management revolution. Harvard Bus. Rev. **90**(10), 61–67 (2012)
22. Schadt, E.E.: The changing privacy landscape in the era of big data. Mol. Syst. Biol. **8**(612), 1–3 (2012)
23. Schadt, E.E., Linderman, M.D., Sorenson, J., Lee, L., Nolan, G.P.: Computational solutions to large-scale data management and analysis. Nat. Rev. Genet. **11**(9), 647657 (2010)
24. Seife, C.: Big data: the revolution is digitized. Nature **518**(7540), 480–481 (2015)
25. Topol, E.J.: The big medical data miss: challenges in establishing an open medical resource. Nat. Rev. Genet. **16**(5), 253254 (2015)
26. Wang, H., Xu, Z., Fujita, H., Liu, S.: Towards felicitous decision making: an overview on challenges and trends of big data. Inf. Sci. **367–368**(2016), 747–765 (2016)
27. Wang, Y., Hajli, N.: Exploring the path to big data analytics success in healthcare. J. Bus. Res. **70**(2017), 287–299 (2017)

Generate Optimal Distributed Query Plans Using Clonal Selection Process

Ruby Rani(✉)

School of Computer and Systems Sciences, Jawaharlal Nehru University,
New Delhi 110067, India
ruby73_scs@jnu.ac.in

Abstract. In this paper we have proposed a bio-inspired approach which selects optimal Top-k query plans of minimal cost from large search space. We have also shown that proposed approach is better in computing Top-k Query plans in terms of shipping cost for Distributed Query Plans Generation (DQPG).

Keywords: Clonal selection · Artificial immune system · Genetic algorithm

1 Introduction

Due to replication property of DDBMS, responses are collected from different nodes. Responses are collected as an output of query plans which increases exponentially with the rise in number of relations in distributed SQL query [1]. This is how DQPG results into an NP hard problem and most efficient execution plans needs to be computed for the query. As per the miscellaneous analysis done over DQPG problem, optimal execution strategy with least computational means is selected from exponential search space where execution cost is the sum of I/O cost, CPU cost and Communication cost [2]. Amongst all costs, communication cost is considered as the most dominant factor. Earlier DQPG problem has been tried to solve using evolutionary algorithms to get the sub-optimal solutions [3, 6] and closeness property was used by GA to generate optimal execution plans [4].

Paper is organized in such a way that problem formulation is discussed in Sect. 2 followed by proposed approach in Sect. 3. Section 4 shows graphs based comparison of proposed approach and GA followed by conclusion in Sect. 5.

2 Problem Formulation

To understand DQPG, let us consider a user query is formed with six relations e.g. T1, T2, T3, T4, T5 and T6 which are present on various sites for e.g. N1, N2, N3, N4, N5 and N6 over DDBS. Relations and corresponding accommodating sites are depicted in the Table 1. Entry‘1’ in cell represents presence of relation at its respective node and ‘0’ otherwise. For example: E1 execution plan is represented with the set of relations as {T1, T2,T3,T4,T5,T6} and corresponding nodes carrying these relations are {2, 1, 1, 4, 3, 6} as shown in Table 2.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 301–305, 2018.
https://doi.org/10.1007/978-3-319-72344-0_26

Table 1. Relation Vs Nodes Matrix, where 1,2,…6 in columns denotes N1, N2,.. N6 and 1,2,…6 in rows denotes T1, T2,.. T6 respectively.

T	N					
	1	2	3	4	5	6
1	1	0	1	0	1	1
2	0	0	1	0	1	0
4	0	0	1	1	0	0
3	1	0	1	0	1	1
5	0	1	0	1	0	0
6	1	1	0	0	0	1

Table 2. Two succeeding generation query plan.

QPN	Query plan	QPC	Next gen query plan	QPC
E1	{2,1,1,4,3,6}	0.780	{6,5,5,5,5,6}	0.278
E2	{3,2,3,4,4,5}	0.860	{4,4,1,4,4,5}	0.389
E3	{4,4,5,4,4,5}	0.444	{4,4,5,4,4,5}	0.444
E4	{6,5,5,5,5,6}	0.444	{6,5,5,5,5,6}	0.444
E5	{2,1,3,4,5,6}	0.830	{4,4,5,4,4,5}	0.444
E6	{2,2,5,5,2,5}	0.500	{4,4,5,5,4,5}	0.500
E7	{6,2,3,4,5,6}	0.780	{6,2,3,4,5,6}	0.500
E8	{3,4,1,5,2,6}	0.830	{4,5,3,5,3,5}	0.611
E9	{4,5,3,5,3,5}	0.611	{4,5,3,5,3,5}	0.611
E10	{6,1,5,4,5,6}	0.720	{6,2,5,5,5,6}	0.611

Heuristics: Two heuristics used here are the number of relations taking part in user query and the concentration of nodes containing these relations. The closeness property depends on the quantity of nodes participating in a query plan such that lesser the number of nodes in query plan lesser is the communication cost. Here, our goal is to compute optimal cost query plan for SQL query while communication cost is equivalent to computing the Query Proximity Cost (QPC). Where,

$$QPC = \sum_{i=1}^{M} \frac{Ni}{R}\left(1 - \frac{Ni}{R}\right)$$

3 Proposed Approach: M-DQPG$_{\text{CLNG}}$

A clonal selection based M-DQPG$_{\text{CLNG}}$ method to solve DQPG is explained here.

```
Algorithm 1. Clonal Selection approach for DQPG (M-DQPG_CLNG)
1: For given RXN matrix and P, Compute random valid execution plans.
2: Compute communication cost of each antibody execution plan($E_i$)
3: Choose top 'n' antibody Query plans of minimal QPC.
4: Calculate total copies to be generated top 'n' execution plans using $C_n = C_n + \left(\frac{(\beta * P)}{i}\right)$
5: Use Roulette Wheel selection to distribute total copies among Top 'n' execution plans $C_i = \frac{(Z-QPC_i)}{Z1}$
      Where, $Z_1 = Z_1 + (Z - QPC_i)$;  $Z = \frac{(R-1)}{R}$;
6: Mutated Plans = Mutation (P, Pm, n, $C_n$, QPC, Cloned execution Plans);
      Compute fitness of mutated copies using step 2.
      Compute Cumulative population using
      Total_ Pop= P+ Mutated Clones;
7: Select higher fitness 'P' execution plans for next generation.
Until Iterations=$I_p$;
```

Notations: RxN: Relation-Node matrix*; P:* Population size*; GP:* Total number of generation of query (execution) plans*; R:* number of relations in query plan*; N:*number of nodes carrying relations; β:clone rate and *n:* Top-n query plans with high fitness (lower QPC). Where C_n, $Clone_i$ and *QPC* notate the total number of clones, i^{th} top query plan clone and Query Proximity Cost respectively.

First step initializes the population of antibodies execution plans from *RxN* relations and P as shown in Table 2. Fitness (communication cost) of these execution plans is computed using QPC function as given in Eq. 1 of [5]. In step 3 Top 'n' antibody execution plans of lower QPC are picked. Step 4 calculates the total number of copies to be generated of Top 'n' execution plans. Where *β* is employed as normalization constant. Step 5 distributes the total C_n clones among Top 'n' chosen execution plans using the *Roulette Wheel Selection Operator* [7] which randomly selects i^{th} execution plan with a probability directly proportional to the fitness value of query plan and generates clones for it. In step 6 Fitness proportionate Selection [7] is applied to mutate clones execution plan computed in step 5, with a constant rate. Mutation probability of higher fitted execution plan is less and vice-versa. In the same step, again compute the fitness of mutated clones of execution plans using step 2 function. In step 7, all the mutated clones are added to the same generation population of antibody execution plans and Top '*P*' execution plans are selected as new population for succeeding generation as depicted in Table 2. Every function except initial two algorithms, would be duplicated over generations until we get a population of fittest query plans based on different parameters explained in experimental section.

4 Experimental Results and Analysis

Simulation study has been done for GA and M-DQPG$_{\text{CLNG}}$ through graphs between number of Generations and commutation cost (QPC) for different set of relations based on various parameters such as mutation rate for GA and M-DQPG$_{\text{CLNG}}$ are {0.05} and {0.05,0.01} respectively. Clone rate {0.99, 0.98} is considered for M-DQPG$_{\text{CLNG}}$ while Crossover for GA is {0.8}. Top-k query plan {0.4, 0.6} for both GA and M-DQPG$_{\text{CLNG}}$ and the simulation was performed for 400 generations. MATLAB 7.9 (R2009a) has been used to implement the proposed approach and graphs for the outcomes are drawn in MS-access. In Fig. 1, readers can see that as the number of generations increases QPC value goes down. Blue line in graph indicates GA while other denotes the M-DQPG$_{\text{CLNG}}$ for different parameters. Graphs shows us that proposed approach has nearly 5–15% less shipping cost than GA. The speculation we get from these graphical results is that this cost will become constant at some of time in future. Figure 2 represents the graphs between Average *QPC* and *Top-k* query plans of relation set {12, 16, 20, 24} both for M-DQPG$_{\text{CLNG}}$ and GA. One can observe from the graphs that with the increase in number of relations in user query, QPC of Top-k query plans also increases and this increasing behavior depicts that query plan with the lowest QPC is the most fittest query plan. Ultimately, Fig. 2 graphs also shows that M-DQPG$_{\text{CLNG}}$ is more competent to collect more cost-effective Top-k execution plans than GA for smaller set of relations in distributed SQL query.

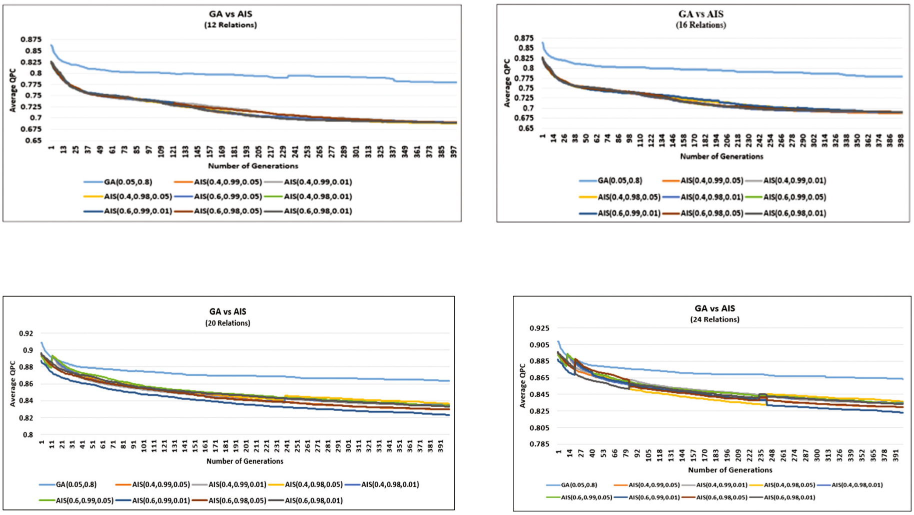


Fig. 1. (a) GA vs M-DQPG$_{CLNG}$ for 12 Relations (b) GA vs M-DQPG$_{CLNG}$ for 16 Relations (c) GA vs M-DQPG$_{CLNG}$ for 20 Relations (d) GA vs M-DQPG$_{CLNG}$ for 24 Relations. (Color figure online)

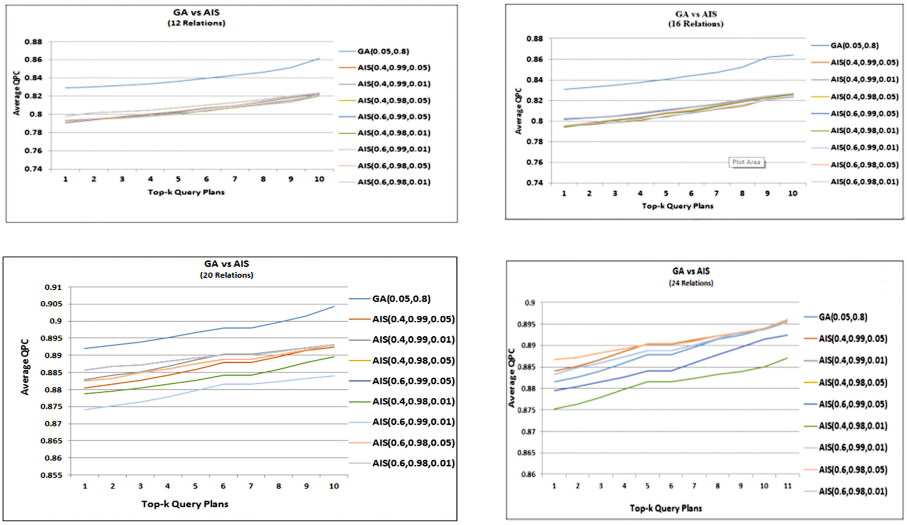


Fig. 2. (a) GA vs M-DQPG$_{CLNG}$Top-k Query Plans for 12 Relations (b) GA vs M-DQPG$_{CLNG}$ Top-k Query Plans for 16 Relations (c) GA vs M-DQPG$_{CLNG}$ Top-k Query Plans for 20 Relations (d) GA vs M-DQPG$_{CLNG}$ Top-k Query Plans for 24 Relations.

5 Conclusion

The objective of the proposed approach is to generate efficient query plans to answer user query. Experiments are performed to calculate the minimum average QPC through GA and M-DQPG$_{\mathrm{CLNG}}$ and later finds better closeness than GA. Further, We show M-DQPG$_{\mathrm{CLNG}}$ is more efficient in collecting top-k most fit query plans as compared to GA.

Acknowledgments. This research work is partially supported by the UPE-II grant received from JNU. Author sincerely thanks to the anonymous reviewers for their kind comments.

References

1. Zhu, Q., Larson, P.-A.: Global query processing and optimization in the CORDS multidatabase system. In: Proceedings of the 9th PDCS Conference, pp. 640–646 (1996)
2. Özsu, M.T., Valduriez, P.: Principles of Distributed Database Systems. Springer, Cham (2011)
3. Jarke, M., Koch, J.: Query optimization in database systems. ACM Comput. Surv. **16**(2), 111–152 (1984)
4. Kumar, T.V.V., Singh, V., Verma, A.K.: Distributed Query Processing Plans Generationusing Genetic Algorithm. Int. J. Comput. Theory Eng. **3**(1), 38 (2011)
5. de Castro, L.N., Timmis, J.: Artificial immune systems: a novel paradigm to pattern recognition. Artif. Neural Networks Pattern Recogn. **1**, 67–84 (2002)
6. Rani, R.: An efficient bio-inspired approach to generate distributed query plans. In: IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), pp. 1–5 (2016)
7. Abdoun, O., Abouchabaka, J., Tajani, C.: Analyzing the performance of mutation operators to solve the travelling salesman problem. arXiv Prepr. arXiv:1203.3099 (2012)

CapAct: A Wordnet-Based Summarizer for Real-World Events from Microblogs

Surender Singh Samant(✉), N. L. Bhanu Murthy, and Aruna Malapati

Department of Computer Science and Information Systems,
Birla Institute of Technology and Science, Hyderabad Campus,
Pilani, Telangana, India
{surender.samant,bhanu,arunam}@hyderabad.bits-pilani.ac.in

Abstract. Short messages from microblog streams often contain information about real-world events. Streams of related messages can be clustered and classified as events or non-events. Summarizing events from clusters of event related messages is a challenging task as the summary needs to be concise yet informational. We present a novel method of summarization of events from short messages. We also propose a method of creating a set of extensive reference summaries from manually created summaries for effective evaluation. We used standard ROUGE based metrics to compare the proposed summarizer with many existing baselines including a strong Hybrid-tfidf method. Our summarizer consistently outperformed others in F1-score with a margin of 11% in ROUGE-1 and 5% in ROUGE-2 over Hybrid-tfidf.

Keywords: Summarization · Event summarization · Twitter

1 Introduction

Short messages from microblogs such as Twitter are often informational and contain discussions on real-world events. Similar messages from a microblog stream can be clustered and classified as event or non-event using different approaches [1–3]. The task of summarizing an event from a cluster of messages is challenging as the summary has to be concise yet provide most information about the event. One application of summarization is to provide automated news updates or alerts to users.

Summarization is either extractive where phrases from the messages are used to get summary or abstractive where new phrases may be constructed from the messages. We use the more popular extractive summarization approach in this work as it works well for event summarization task from short messages where each message mostly focus on a certain fixed topic.

Problem Statement: Given a set of related messages discussing an event, extract the most representative message for the event. The message provides the most information about the event, and contains minimal extra unrelated words.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 306–314, 2018.
https://doi.org/10.1007/978-3-319-72344-0_27

The main contributions of this paper are:

1. We present a novel method of summarizing short event-related messages. We use language specific and statistical properties of short written messages along with a publicly available corpus to extract the most representative summary (Sect. 3.2).
2. We propose a method to create a set of extensive reference summaries from a single manual summary (Sect. 4.1). Reference summaries include various combinations of words that could be substituted for existing ones. Since out of vocabulary (OOV) words can be used by humans for the same event, a set of reference summaries is more effective than a single manual summary. Wordnet [4] is used to enhance the manual summary dataset. We discuss results of our experiments in Sect. 4.2.

2 Related Work

The work in [5] used an abstractive and unsupervised method to generate concise phrases from opinions as an optimization problem with the help of web-based corpus. The research in [6] extracted representative tweets by exploiting the temporal correlation to create topic models. [7] summarized sporting event by extracting key moments during the event using temporal cues such as spikes in the number of messages during such moments. Sporting event summarization was also topic of [8] that used a Hidden Markov Model based algorithm to summarize a recurring event in terms of the main states during that event. A supervised approach to twitter context summarization using user popularity was studied in [9] where the goal was to summarize a context of twitter messages into a few informative tweets. In contrast to these works, our method uses only texts of the messages to extract the most concise and informational message.

The research in [10] had similar goal to ours. It introduced a Hybrid-tfidf based method to summarize twitter messages (tweets). Our work is different as we used properties of written English, a publicly available corpus and statistical properties of messages. Also, we introduce a method to generate extensive reference summaries to better evaluate the system summaries.

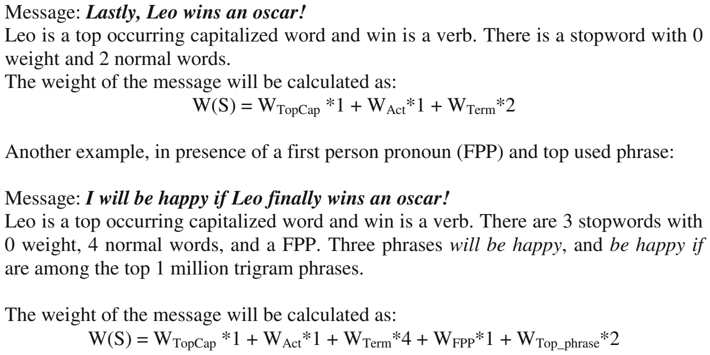
3 Summarization Methods

We used a few naive and a strong summarizer called Hybrid-tfidf as baselines that are explained next, followed by our proposed summarizer.

3.1 Existing Summarizers

Random summarizer (R): Selects a random message as summary.

Centroid-based summarizer (C): In this method, each message is converted to a bag of words and the centroid is computed. The nearest message to the centroid is selected as summary. This is better than random summarizer.



Message: ***Lastly, Leo wins an oscar!***
Leo is a top occurring capitalized word and win is a verb. There is a stopword with 0 weight and 2 normal words.
The weight of the message will be calculated as:

$$W(S) = W_{TopCap}*1 + W_{Act}*1 + W_{Term}*2$$

Another example, in presence of a first person pronoun (FPP) and top used phrase:

Message: ***I will be happy if Leo finally wins an oscar!***
Leo is a top occurring capitalized word and win is a verb. There are 3 stopwords with 0 weight, 4 normal words, and a FPP. Three phrases *will be happy*, and *be happy if* are among the top 1 million trigram phrases.

The weight of the message will be calculated as:

$$W(S) = W_{TopCap}*1 + W_{Act}*1 + W_{Term}*4 + W_{FPP}*1 + W_{Top\_phrase}*2$$

Fig. 1. Examples of weight calculation of short messages.

Hybrid-tfidf summarizer (HT): This summarizer uses a Hybrid-tfidf based approach as explained in [10] to assign weights to messages and extracts the highest weighted message as the representative summary. We selected the best performing parameters of this summarizer in our dataset.

3.2 CapAct: The Proposed Summarizer (CA)

If S is the set of all messages, V is the set of vocabulary (features) of all the messages in a cluster, $V_{i,j}$ is the value corresponding to i^{th} feature in j^{th} message, the centroid C is calculated by (1) that averages the features of all messages. Cosine similarity was used as the distance metric.

$$C = \frac{\sum_{i=1}^{|V|} \sum_{j=1}^{|S|} V_{i,j}}{|S|} \tag{1}$$

The following key observations of Twitter messages form the basis of the proposed summarization system: Capitalized words, if used by multiple messages, are very likely to be important words related to the event e.g. named entities. Verbs in messages can signify an important action being performed in the event. Publicly available corpus of most common n-grams can be utilized to help in weighing well-formed messages. After discarding messages with high proportion of capitalized words, the most commonly used capitalized words and verbs were identified. Each message was given a weight that is the sum total of weights of the constituting words. The following intuitive rules were used to assign weights to the words of a message:

- Top N (=10) most common capitalized words in the messages were given the more weight W_{TopCap} as they are likely to be significant words such as named entities. Other capitalized words were given less weight. Similarly, the top verbs get higher weight W_{Act} for the reason that they are likely to be action terms performed on or by the named entities.

- Messages with first person pronoun (FPP) such as ‘I’, ‘me’, ‘my’, etc. were penalized by adding a negative weight for each such occurrence. This penalizes noisy messages with personal opinions or other non-informational messages.
- We used the Corpus of Contemporary American English (COCA) [11] to give more weight to common trigram phrases, if present in a message. The corpus has publicly available top 1 million most commonly used phrases created from multiple sources on the web. Presence of common phrases in a message were assumed to indicate well formed-ness of the message.
- Remaining words of length greater than two characters were given the least weight $\mathrm{W}_{\mathrm{term}}$.

An example of weight calculation before normalization is shown in Fig. 1. The best values for the weights were computed empirically by experimenting with different weights for each type of word. It was found that assigning highest weight to most common capitalized words, followed by action words, other capitalized words and other words in decreasing order results in best performance. The following sets of weights were finalized after multiple experiments with different datasets. $\mathrm{W}_{\mathrm{TopCap}} = 5$, $\mathrm{W}_{\mathrm{Act}} = 4.5$, $\mathrm{W}_{\mathrm{Cap}} = 3$, $\mathrm{W}_{\mathrm{term}} = 1$. Since the top phrases are common and messages are likely to contain many of them, the weight for each such occurrence was selected as: $\mathrm{W}_{\mathrm{TopPhrase}} = (1 + \log)$ * (number of top phrases in the message).

$\mathrm{W}_{\mathrm{FPP}}$ were penalized by giving them a negative weight of -2.

The message nearest to the centroid of tweets with highest weights was extracted as system summary for the event. The summarizer is summarized in Algorithm 1, where W_t is weight of the type of word t present in the message, |t| is the number of words of a particular type t, |m| is the number of the non-stop words of length greater than two of corresponding type.

Algorithm 1. CapAct

```
procedure SUMMARIZE EVENTS(S)
    M ← REMOVE OUTLIERS(S)
    FIND TOP CAPITALIZED WORDS(M)
    T ← COMPUTE_WEIGHTS(M)
    EXTRACT SUMMARY(T)
procedure COMPUTE_WEIGHTS(M)
    for each $m \in M$, if len(m) $> 2$ do
        $W_m \leftarrow \frac{\sum |t|.W_t}{|m|}$, t ∈ m, t ∉ {stopwords} and
          $W_t \in \{W_{TopCap}, W_{Act}, W_{TopPhrase}, W_{Cap}, W_{FPP}, W_{term}, 0\}$
        $W_m \leftarrow 0$, if $\frac{\#Capitalized(m)}{|m|} > 0.8$
    return(top_weighted_messages)
```

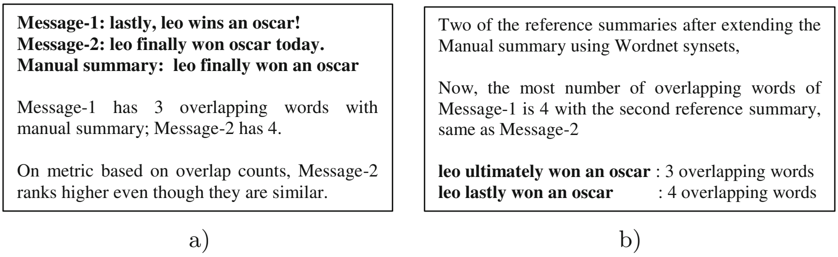
4 Experiment

Since there is no publicly available benchmark twitter dataset suitable for our work, we created our own event clusters in an automated manner. We used a message clustering and event classification approach as described in [3] that can be explained briefly as: Real time tweets were tracked for generic words such as ‘is’, ‘the’, ‘if’, etc. Since these words are present in almost every sentence, the method is expected to download all tweets that are available. About 100 million tweets across different times were collected and preprocessed. Preprocessing steps included discarding near duplicates, retweets, very short or non-English tweets, removing hyperlinks, etc. This brought down the number to 17 million tweets which were then divided into different sets of overlapping hourly bins of tweets. Top bigrams in each hourly bin were used to create clusters of high frequency tweets that contained the top bigrams during that hour. Various revealing features of the clusters containing event tweets were identified and multiple classifiers were trained to classify event clusters. Full details about the method can be found in [3].

For the purpose of summarization, the clusters classified as events were then manually inspected and only the ones verified to contain event related messages were selected. A total of 41 such clusters were used in this experiment with a total of 3963 messages, with minimum and maximum cluster size of 22 and 770 messages, respectively. The events were from domains such as sports, entertainment, awards, social, political, terrorism, etc. These clusters provide a common dataset to compare the relative performance of various summarizers. Furthermore, we empirically computed the best parameters of the Hybrid-tfidf before comparing it against our algorithm.

4.1 Creation of Reference Summary Dataset

A total of three sets of manual summary was created by different humans who were given general instructions to write in less than 140 characters a concise and informational summary for each of the clusters of messages provided. There is an



Message-1: lastly, leo wins an oscar!
Message-2: leo finally won oscar today.
Manual summary: leo finally won an oscar

Message-1 has 3 overlapping words with manual summary; Message-2 has 4.

On metric based on overlap counts, Message-2 ranks higher even though they are similar.

a)

Two of the reference summaries after extending the Manual summary using Wordnet synsets,

Now, the most number of overlapping words of Message-1 is 4 with the second reference summary, same as Message-2

leo ultimately won an oscar : 3 overlapping words
leo lastly won an oscar : 4 overlapping words

b)

Fig. 2. Overlapping words of system summary with (a) a single manual summary, and (b) multiple reference summaries. All messages have been lowercased. Reference summaries cover the word overlap based message similarity better.

issue common in using a manual dataset directly that the same message can be written by using different words with similar meanings. This would incorrectly give different result for messages with different words but similar meanings in metrics that rely on counting overlapping n-grams directly. Figure 2(a) is an example where two messages with same meaning would result in different scores in n-gram based metrics. In this example, stopwords have not been removed to keep it simple. To handle this, we created a set of extensive reference summaries from each manual summary by extending it using Wordnet synsets. Wordnet is a lexical database that makes different meanings of a word available into different contexts called synsets. Synsets were used to find all synonyms of a word that could replace a word in the manual summary. All words with similar meanings belong to a common synset. Capitalized words, stop words, and words with less than two characters were not processed. Since the context (called 'sense' in Wordnet) in which the word has been used in a manual summary is unknown, all possible contexts of the word were assumed equally possible, the first 10 synsets for each word, and for each synset the first 5 synonyms were used to extend the manual summary. This cross-product of sets of words increased the size of reference summaries significantly. An example of a manual summary and a few of the reference summaries extended using synsets is shown in Fig. 2(b). Figure 3 shows the improvement in performance of summarizers when a set of reference summaries were used in comparison to a manual summary. We used ROUGE-N ($\mathrm{N} = 1, 2$) metric to evaluate all our datasets as it is a widely used standard metric introduced for very short summary data in Document Understanding Conferences (DUC) [12]. In ROUGE-N, overlapping n-grams between automated system summary (called peer summary or system summary) and manual summary (called model summary) are compared. F1-score (F-score) is harmonic mean of Precision (P) and Recall (R) that are defined as:

$$\mathrm{P} = \frac{\#overlapping\ n\text{-}grams}{\#n\text{-}grams\ in\ peer\ summary},\ \mathrm{R} = \frac{\#overlapping\ n\text{-}grams}{\#n\text{-}grams\ in\ model\ summary}$$

Before running any summarizer, the farthest 20% messages from the cluster centroid were removed as outliers. Then, the messages were tokenized into individual words and cleaned by removing commonly occurring punctuations from the word boundaries.

4.2 Results and Analysis

We compared the set of manual summaries against each other to get realistic limits on the performance of our system. For each comparison, one of the sets was used as system summary, and the other's extended summary sets were used as model summary. To evaluate system summaries, we selected Manual-1 and Manual-2 and their reference summaries as their mutual ROUGE scores were the best. The summary extracted by our system was evaluated against all the corresponding reference summaries and the maximum F-score value was taken as the final score. All messages were lowercased and both the system summary and

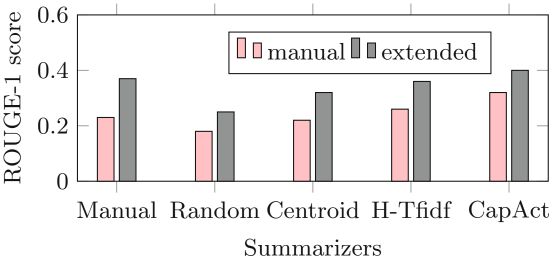


Fig. 3. F-scores of summarizers on manual vs. extended reference summaries.

Table 1. F-score, precision and recall of the summarizers on reference set using ROUGE-1 and ROUGE-2.

	ROUGE-1			ROUGE-2		
	F-score	Precision	Recall	F-score	Precision	Recall
Manual	0.37	0.39	0.39	0.17	0.18	0.18
Random	0.25	0.22	0.32	0.09	0.08	0.11
Centroid	0.32	0.34	0.30	0.16	0.17	0.15
H-Tfidf	0.36	0.40	0.34	0.20	0.22	0.19
CapAct	0.40	0.43	0.39	0.21	0.23	0.22

reference summaries were extended by performing stemming and morphing. A stemmer converts a word to its stem whereas morphing searches for a word form that is not in Wordnet. The F-score and the corresponding precision and recall of system summaries for all methods are given in Table 1. Except row labeled Manual, these are the average of best F-score and corresponding Precision and Recall achieved by the summarizers using the two reference summaries as model summaries. For Manual, the reported results are the average scores of the two manual summaries. For summarizers, the best F-scores of system summaries for each set of corresponding reference summaries was computed over all the 41 event clusters and averaged, as given by (2). Since there are two sets of reference summaries, the F$_{avg}$ results for the two sets were averaged and reported.

$$F_{avg} = \frac{\sum_{i=1}^{|Summary_{sys}|} F_i}{|Summary_{sys}|}, \text{ where } F_i = \arg\max_i (F\text{-}score), i \in \{Summary_{ref}\} \tag{2}$$

We measured the performance of all summarizers in terms of F-score that provides a fine balance between precision and recall. CapAct consistently performed well giving better results than the hybrid tfidf (H-Tfidf), especially for short manual summaries. CapAct had a margin of 11% in ROUGE-1 and 5% in ROUGE-2 above H-Tfidf in F-score. While the performance of our summarizer is likely to vary with different datasets, its overall performance was impressive in our experiments.

5 Conclusion

We presented a summarizer (CapAct) that extracts a representative message from a cluster of related messages. We compared the performances of the proposed summarizer on standard metrics with many baseline methods. We proposed a method to generate reference summaries from manual summary dataset using Wordnet. CapAct consistently outperformed the others summarizers in our experiments. As a future work, the dataset would be further increased and more comparison metrics would be used to compare the performances of various summarizers. It would be interesting to apply this summarizer to other sources of short messages. More linguistic properties could be added to refine the weights further. The approach could also be extended for n-grams with n > 1 and different parts of speech combinations. Another interesting work would be extract important information such as time or duration, key entities, and action involved in an event, and present them to a user.

References

1. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: Proceedings of the International conference on World Wide Web, Raleigh, North Carolina, USA, 26–30 April 2010
2. Becker, H., Naaman, M., Gravano, L.: Beyond trending topics: real-world event identification on Twitter. In: Proceedings of the International AAAI Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, 17–21 July 2011
3. Samant, S.S., Bhanu Murthy, N.L., Malapati, A.: Bigram-based features for real-world event identification from microblogs. In: Proceedings of the International Conference on Computing, Communication and Networking Technologies (ICCCNT), New Delhi, India, 3–5 July 2017
4. Miller, G.A.: Wordnet: a lexical database for english. Commun. ACM **38**(11), 39–41 (1995)
5. Ganesan, K., Zhai, C., Viegas, E.: Micropinion generation: an unsupervised approach to generating ultra-concise summaries of opinions. In: Proceedings of the International World Wide Web Conference, Lyon, France, 16–20 April 2012
6. Chua, F.C.T., Asur, S.: Automatic summarization of events from social media. In: Seventh International AAAI Conference on Weblogs and Social Media, Cambridge, Massachusetts, USA, 8–11 July 2013
7. Nichols, J., Mahmud, J., Drews, C.: Summarizing sporting events using Twitter. In: Proceedings of the International conference on Intelligent User Interfaces, Lisbon, Portugal, 14–17 February 2012
8. Chakrabarti, D., Punera, K.: Event summarization using tweets. In: Proceedings of the International Conference on Web and Social Media, Barcelona, Spain, 17–21 July 2011
9. Chang, Y., Wang, X., Mei, Q., Liu, Y.: Towards Twitter context summarization with user influence models. In: Proceedings of the International Conference on Web Search and Data Mining (WSDM), Rome, Italy, 4–8 February 2013

10. Sharifi, B., Hutton, M.A., Kalita, J.: Experiments in microblog summarization. In: Proceedings of the International Conference on Social Computing (SocialCom), Minneapolis, MN, USA, 20–22 August 2010
11. Davies, M.: The 385+ million word corpus of contemporary American english (1990-present). Int. J. Corpus Linguist. **14**(2), 159–190 (2009)
12. Lin, L.Y.: Looking for a few good metrics: rouge and its evaluation. In: Working Notes of NTCIR-4, Tokyo, Japan, 2–4 June 2004

ARMM: Adaptive Resource Management Model for Workflow Execution in Clouds

Harshpreet Singh[1,2](✉) and Rajneesh Randhawa[2]

[1] School of CSE, Lovely Professional University, Phagwara, Punjab, India
harshpreet.17478@lpu.co.in

[2] Department of Computer Science, Punjabi University, Patiala, India

Abstract. Cloud offers computational resources as a utility to execute dependent tasks ensemble as an application workflow, where each task has a different resource requirement. Resource management frameworks are required to dynamically provision resources to enable scalability and seamless execution of workflows. In this paper, an adaptive resource management model is presented, which allocates and reschedule the resources based on their usage history and performance metrics. It further makes decisions to adapt workflow tasks to optimize deadline, budget and resource performance. A case study using different workflows is used to describe the model in a simulated environment considering various run time scenarios.

1 Introduction

Cloud provides varied computational resources as a utility which enables workflow development, deployment and execution. Dynamically changing environment of cloud is a major concern among providers in providing seamless and scalable access to wide-ranging heterogeneous resources as per the user demand which affects the overall application performance. The ability of cloud to uphold the service demands and its allegiance to the timely response without affecting workflow execution is a challenge well rehearsed within cloud community.

Autonomic elements in clouds have limited operation both on the level of individual servers as well as on the level of clusters and virtual organizations. To enable adaptability, the run-time system should actively understand the user requirements and analyse the runtime performance to operate and fulfil application requirements. The static resource assignment may lead to uneven performance of cloud resources over time. In the computational clouds the major challenge is to adapt workflows to dynamically changing run time environment with an objective of increasing the performance and maintaining the required quality of service.

This paper proposes a Adaptive Resource Management (ARM) model which builds mechanism to self-adjust configuration parameters into the system which provides run time adaptability to seamlessly access the heterogeneous resources

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 315–329, 2018.
https://doi.org/10.1007/978-3-319-72344-0_28

and an ease in executing workflows within defined constraints without struggling with the application or resource complexity. The main aim is to maximize provider revenue and user experience with efficient management of cloud resources. Adaptations in the model are based on current resource performance and utilization along with application status in terms of makespan and cost. Objective of the model is to optimize execution budget and time along with improving utilization and performance of resources. A feedback controller is employed which collects data from various components and resource level metrics and dynamically adjust the system state based on the measured output and determine whether task or system level adaptations are required.

The paper is divided into following sections: Sect. 1 disused the overview of the problem to dynamically manage the resource configuration, Sect. 2 presents the related work in the area of adaptive resource management. Section 3 proposes the adaptive resource management model with its architecture and components. Section 4 presents the results and finally Sect. 6 provides summary and conclusion.

2 Related Work

Cloud system are diverse and often composed of heterogeneous set of resource but differ in computing configurations. In this section some of the proposed techniques are surveyed that are capable to adapt cloud services in accordance with the objective and service parameters.

Jung et al. [1] developed a scalable and multi-level hierarchical controller with an ability to balance steady state performance and power. The model predicts the response time and power consumption for various system configurations to manages application adaptations. Controller has the ability to transform the system state to a point where the overall utility can be improved.

PRESS (PRedictive Elastic reSource Scaling) is proposed by [2] that extracts fine-grained patterns from resource demand data and adjust the future allocations automatically. Signal processing technique and statistical state driven approach is used to identify resource pattern. A discrete time markov chain is used to predict resource demand for future. The model strives to allocate just enough resources to avoid over estimation and service level objective violations.

PDRS (Prediction based Dynamic Resource Scheduling) [3] is a technique proposed to auto scale the resources for cloud systems based on virtualization. The predictions are done to place VMs on PMs to ensure maximum utilization with minimum resource wastage.

An online temporal data mining system called ASAP (A Self-Adaptive Prediction System for Instant Cloud Resource Demand Provisioning) is developed in [4] to make adaptation according to varying user resource demands and trains itself to predict resource consumption for the future.

Ahmed et al. in [5] pro posed an adaptive controller model based on queueing theory. The controller estimates the future load on a service and horizontally scale up and scale down cloud resources. The proactive elastic controller is coupled with a reactive elastic controller to improve utilization and prevent resource oscillations.

Buyya et al. [6] presented autonomic resource provisioning and management techniques to support SaaS application on clouds. The resources are provisioned in accordance with the QoS requirement of the user with an aim to maximize efficiency and minimize operational cost.

Imai et al. [7] proposed Workload-tailored Elastic Compute Units (WECU) model for computing and allocating cloud resources. IT considers application-level migration to support dynamic workload scalability.

Jamshidi et al. [8] proposed a control theory mechanism for automatically adding or removing cloud resources based on type-2 fuzzy logic. The technique enables qualitative specification of threshold values for making decisions of auto configurations.

Sedaghat et al. in [9] proposed a peer to peer based resource management framework for maximizing the data center utilization and minimizing power consumption. The resource allocation mechanism is build over an agent model to accomplish goal oriented tasks. The local agents perceive their local view which is selecting the best PM to host the incoming request for a VM. They communicate the system state with other agents using gossip protocol. The system proceeds towards the global objective of high utilization of PM while enabling energy saving mode.

Joseph et al. in [10] proposed a service rate allocation mechanism based on time to completion and proportional fairness. The prediction of compute units required for completion of workload is carried out using kalman filter estimators. Additive Increase Multiplicative Decrease Algorithm is used for allocation and termination of compute units to improve resource utilization.

3 Adaptive Resource Management Model

The proposed Adaptive Resource Management Model (ARMM) has the ability to self optimize and adjust the mapping of tasks resources. The basic functionality of the ARMM is the execution of workflow tasks, identification of optimal cloud resources and adhering to users demand.

3.1 ARMM Architecture

In clouds the resources are acquired by the user for a definite time stamp depending on the execution time of the applications and the defined budget, which limits the amount and type of resources to be employed. Adhering to the set of constraints, which guarantee the cloud resources to function correctly under given conditions ensures scalability and integrity of the cloud. Let $R = r_1, r_2, \ldots, r_n$ be the set of cloud resources to deployed workflow tasks $T = t_1, t_2, \ldots, t_m$. The mapping of workflow tasks and cloud resources is governed by a set of agreements, namely Service Level Agreement (SLA) and Resource Level Agreement (RLA) between users and cloud providers.

The ARMM coordinates the action of making the changes using the components as specified in Fig. 1. The components are defined as follows:

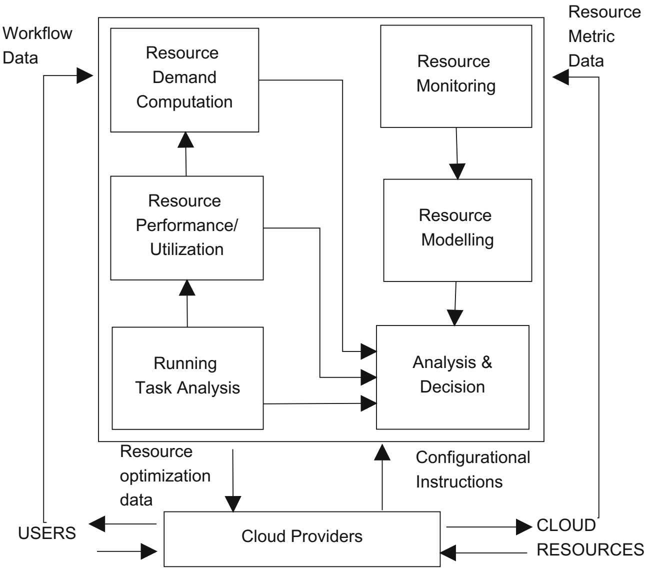


Fig. 1. Key architecture components of ARMM

Resource Demand Computation (RDC): Resource selection is based on the reinforcement learning where the component performs actions based on the past experiences and selects the optimal resources based on the rewards [11]. The component using Q-learning approach to optimize the resource demand and scheduling strategy as illustrated in Algorithm 1. It maintains resource performance and utilization rate of the resource while keeping a *Q-value* which indicates it's efficiency.

For each workflow task, the component selects a resource according to the ϵ-greedy rule: with probability $(1 - \epsilon)$ it chooses the resource with the highest Q-value (ties are broken randomly), while with (small) probability ϵ the component randomly and uniformly chooses among the other resources. After each completed workflow task, the component gets a reinforcement signal (containing the start-time and the end time for that task), calculates the metric E_i, and translates it into a reward e for resource r that can be chosen as follows:

$$r = sign([\rho r] - \rho r) \tag{1}$$

where $[\rho r]$ is the utility averaged over all the submitted tasks. Finally, the component updates the Q values as

$$Q_{r,t+1} \leftarrow Q_{r,t} + \alpha(eQ_{r,t}) \tag{2}$$

Algorithm 1. Resource Demand Computation

```
Input: $Q$ table
Initialize $Q$ table, state $s_t$
$s_o \leftarrow$ current system state; Select $a_o$ from $E$
repeat
    for each state $s$ do
        Take action $a_t$ using $\epsilon - greedy$ policy
        for each resource $r_j inR$ do
            Take action $a_t$, observe $e$ and $S_{t+1}$
            $Q_t = Q_t + \alpha[e - Q_t]$
            $s_t = s_{t+1}, a_t = a_{t+1}$
        end for
    end for
until All task are allocated resources
```

where α is the learning rate. Initial allocation of the task onto the resources is performed with the aspiration that the requirements for each workflow task are met and the overall resource utilization is optimized.

Resource Monitoring Component (RMC): The component is responsible for monitoring the execution of workflow and task performance in periodic intervals to ensure its execution within defined deadline $\mathcal{D}$.

Algorithm 2. Resource Monitoring Component

```
Input: Workflow tasks $T$ and available resources $R$
Output: Estimated completion time $M_{est}$ and Adapted task list
Initialize Set of available resource $R = r_1, r_2, \ldots, r_n$
Initialize Set of workflow tasks $T = t_1, t_2, \ldots, t_m$
Add Resource to resource pool $R_{add} = r_1, r_2, \ldots, r_a$ where $a \leq n$
Allocate resources of task using Algorithm 1
for Processing workflow, Compute resource requirement do
    if $r_{scheduled} < r_{avilable}$ then
        Start Execution of workflow tasks
    else
        Generate Warning
for All resource $R_{add}$ do,
    Calculate $M_{est}$
    if $M_{est} \leq M_{ex}$ then
        Continue Execution of Tasks
    else
        Perform adaptations using Algorithm 5
```

For example, analysis component computes the estimated completion time M_{est} of the tasks and checks for the condition $M_{est} \leq M_{ex}$ is satisfied. If it is not satisfied, it takes one of the following measures.

Algorithm 3. Resource Modelling Component

```
Input: Workflow tasks $T$ runtime and available resources $R$
Output: Predicted Makespan $M_{Pe}$
procedure PREDICTED MAKESPAN($runtime_l, slots$)
    for each level $l$ do
        for each task in level $t_i^l$ do                ▷ For each task in level $l$
            Run $t_i^l$ on $r_j$                          ▷ Execute each task on available resources
            Compare $Rt_{r_j}^{t_i^l} \leq Rt_{r_{j+1}}^{t_i^l}$      ▷ Compare runtime $Rt$ of each task
        $Rt_l = \sum_{t \forall l} Rt_t$                  ▷ Runtime for each level
        $maxSlots_l = taskBylevel_l$                      ▷ Maximum slots for each level
    $M_{Pe} = \sum_l Rt_l + Delay_l$        ▷ Predicted makespan by adding all level runtime
    return $M_{Pe}$                         ▷ Return makespan for given number of resources
```

- Run-time environment of the task on the same resource may be changed, so that its performance is improved.
- If the current resource is overloaded with some other computational jobs or resource with higher computational capabilities is added, then the monitoring component suggests rescheduling it to some new service provider.

Resource Modelling Component (RMOC): The component automatically constructs the resource forecasts using linear regression method based on historical dataset. The remaining makespan is computed using Eq. 3 and assessments for completion within deadline can be compared with the scheduled remaining makespan M_{Re}. Even when the initial prediction M_{Pe} is not 100% accurate as given in Algorithm 3, runtime prediction can determine with some accuracy that the execution will take significantly longer than anticipated.

$$M_{Re} = \frac{M100\% - \sum_{i=0}^{n-1} \varDelta M_i F_i}{F_{estimate}} \tag{3}$$

M_i is the elapsed time since the start of the execution. The fractional load F_i available to the workflow is measured over n samples at time M_i. M100% is the time the workflow would take to execute given no shortage in resource throughout the workflow execution. M100% can be generated using the initial prediction technique. $F_{estimate}$ is the mean CPU load available to the application over n samples as given using Eq. 4.

$$F_{estimate} = \frac{1}{n} \sum_{i=0}^{n} F_i \tag{4}$$

M_{Pe} is the predicted remaining makespan if $F_{estimate}$ is maintained for the remainder of the execution. In the event that $F_{estimate}$ continually drops in successive samples the estimated value will be invalid. This can be countered with a worst case value for $F_{estimate}$. After migration a scaling factor Y is used to adjust for the change in CPU potential. The value of M_{Pe} prior to migration is scaled using Eq. 5. M_{Pe}^{m} is the remaining time on the new resource given its potential. Overheads due to migration, such as file transfers and time in pending queue of new resource are assumed to be negligible.

$$M_{Pe}^{m} = YM_{Pe} \tag{5}$$

Up to date values of M_{Pe}^{m} are calculated using Algorithm 3. Then it decides the resource mapping, with the goal of achieving optimal performance for entire workflow, and submits all the results to the analysis and decision component.

Resource Performance and Utilization Component (RPUC): The component uses modelling techniques to compute the resource performance and model them for future availability. The initial allocation of resources is done on the basis of trust driven strategy and reinforcement learning. Initial allocation may not meet the performance guarantee, for which firstly, resources with enhanced capabilities are selected in order to process the task if execution time is less than M_{ex}. Secondly, the tasks (data computation) may run on additional resources to improve the computation time and the performance. Lastly, the task can be migrated to another resource if the resource is overloaded or it reaches the minimum acceptable levels of requirements.

Algorithm 4. Resource Performance and Utilization Component

Input: Workflow tasks T runtime and available resources R
Output: Task and Resource Pair ($T \rightarrow R$).

```
procedure RESOURCE UTILIZATION($R$)
    if $R_{provisioned} < R_{required}$ then
        Add new resources $R_{add}$ for allocation using Algorithm 1
    else
        Generate Warning
    if $M_{ex} > M_{Pe}$ then
        Reallocate resources using Algorithm 5
    else
        Generate Warning
```

Analysis of Running Services Component (ARC): The component deals with the changing execution profiles and adaptations. For example, workflow requirement is subjected to change when resource capabilities are added (memory, disk etc.) or different resources (software availability) are required, or when a lack of capacity (e.g. disk space) is detected.

The workflows task runs in two modes which depends on the workflow task list. The first mode is executed before any adaption are made and using the resources as scheduled. The second mode is executed after making the adaption. Makespan of the workflow M_{ex} is given by:

$$M_{ex} = ((M_{ne}/N) \times N_j)) + ((M_a/N) \times (N - N_j)) + O_V \tag{6}$$

where, M_{ne} is the execution time of tasks without adaptations, M_a be the execution time of the workflow tasks after adaptations, and N is the total number of task. Considering, $((M_{ne}/N) \times N_j)$ to be the execution time of the tasks without adaption and $((M_a/N) \times (N - N_j))$ is the execution time of the tasks after adaption, where N_j is the number of tasks executed without adaption.

$$M_{ex} = M_{Nt} + M_{Rt} + O_V \tag{7}$$

M_{Nt} is the time taken to execute complete workflow. M_{Rt} is the time spent in executing tasks after rescheduling and O_V is the overhead of adaptation which is considered to be negligible. Speedup can be computed by:

$$M_{ne}/M_{ex} = M_{ne}/(M_{Nt} + O_v + M_{Rt}) \tag{8}$$

The expected completion time of the workflow under constraints can be computed by confirming if the condition $M_{Nt} + O_v + M_{Rt} \leq M_{ex}$ is true.

Analysis and Decision Component (ADC): It comprises of the run time environment of workflow applications and supports advance reservation of resources and also gets the job net ready. On receiving a workflow it reserves the resources as per the schedule and if the allocation is a result of rescheduling, it revokes resource reservation for replaced schedule before making new reservations. The decision component uses an inference mechanism and rule base to determine if adaption is likely to occur. A series of rules based on those specified in the SLA and control variables are used to describe the workflow state during run-time.

Algorithm 5. Analysis and Decision component-1a

```
Input: Workflow tasks $T$ runtime and available resources $R$
Output: control_action
procedure ADAPTION1($M_{Re}, M_{Pe}, M_i$)
    if $M_{Re} > M_{Pe}$ or $M_i > 0.75M_{Pe}$ and ($R_{scheduled}[s] < R_{added}[s]$) then
        control_action = migrate($R_added[s]$)
    else if $M_{Re} > M_{Pe}$ and $0.5M_{Re} < M_i \leq 0.75M_{Pe}$ then
        control_action = checkpoint($W_{ex}[T_i]$)
    else
        control_action = None
    return control_action
```

Firstly, in case of the adaptations in the workflow the decision component controls adaption by Algorithm 5. The control variables used are: (1) the scheduled remaining makespan $M_{Re} \Rightarrow (M_{Re} = M_{ex} - M_{ne})$, and (2) the predicted remaining makespan $M_{Pe} \rightarrow (M_{Pe} = M_c - M_{ex})$. The component takes the scheduled remaining makespan M_{Re}, the predicted remaining makespan M_{Pe}. and M_i the elapsed time since the start of the execution as inputs.

The algorithm describes the situations when the predicted remaining makespan M_{Pe} is very much greater-than the scheduled remaining makespan M_{Re} and the application is more than 75% complete, or if the resources with higher services are available. For this situation an attempt is made to migrate the task onto a faster resource to reduce the execution time and to increase the performance. Also, the situation when the predicted remaining makespan M_{Pe} is much-greater-than the scheduled remaining makespan M_{Re} and the workflow is between 50% and 75%. For this situation information about the checkpoint is send if available for the executing workflow job/task instance. The Algorithm 6 describes the situation when a new workflow task t^i_{nw} arrives and the resources required ($t^i_{nw}[r_{rq}]$) by the new task are being used by the executing workflow t^i_{ex} then the resources are released for the new task and the current executing task is migrated to the minimum acceptable resources ($t^i_{ex}[R_{lw}]$) or to the original scheduled resources ($t^i_{ex}[r_s h]$).

Algorithm 6. Analysis and Decision component-1b

Input: Workflow tasks T runtime and available resources R
Output: control_action

```
procedure ADAPTION2($W_{ex}M_i, W_{ex}M_{Re}, R_{adapted}, t_{nw}, t_{nw}[R_{req}]$)
    List of new task $t_{nw}$ to run in parallel
    if $t_{nw}$ = high priority and $t_{nw}[R_{req}] = t_{ex}[R]$ then
        control_action = migrate ($t_{ex}[R] = t_{ex}[R_{low}]$)
    else if $t_{ex}M_i < t_{ex}M_{Re}$ and $t_{nw}[R_{req}] = t_{ex}[R_{adapted}]$ then
        control_action = checkpoint ($t_{ex}[r_{added}]$)
        control_action = migrate ($t_{ex}[R_c urrent] = t_{ex}[R_o riginal]$)
    else
        control_action = none
    return control_action
```

4 Experimental Setup

4.1 Workflow Configuration

The proposed ARM model is compared with a heuristic model with decisions to provision and allocate resources is based on best-fit approach. The resources are modelled as that of Amazon EC2 with varying hardware configuration and pricing as per computation speed. The workflows are generated with varying width, regularity, density, jumps [12] and Communication to Computation Ratio (CCR) [13] using workflow generator and are classified as Type-1, 2 and 3. The summary of the workflow is given in Table 1.

Table 1. Workflow summary

Parameter	Value		
	Workflow 1	Workflow 2	Workflow 3
Nos. of task	100	500	1000
Task length ($\times 10^9$ MI)	50–200	200–500	500–1500
Data dependency (GB's)	0.100–1	1–10	10–50
Width	5	10	20
CCR	0.5	0.5–1.0	0.5–2.0
Regularity	0.2–0.8		
Density	0.2–0.8		

4.2 Infrastructure Configuration

The different configuration of virtual machines are illustrated in Table 2. The VMs are divided into clusters where xsmall, small are general purpose compute instances, medium and large instance provide a balance of compute, memory and storage and xlarge and 2xlarge are compute-optimized instances for high computational tasks.

Table 2. Resource summary

VM type	VCores	Memory (GB)	Disk space (GB)	Price (per hour)
xsmall	1	0.75	20	$0.015
small	1	1.75	40	$0.03
medium	2	3.75	80	$0.059
large	2	6.5	32	$0.19
xlarge	4	15	80	$0.379
2xlarge	8	30	160	$0.458

The resources are can be dynamically provisioned as per the demand prediction. For the ARM model the number of resources are assumed to be large as the system has pre-configured copies of the above mentioned instances.

4.3 ARM Model Configuration

Initially the workflow are executed on cloud resources for performance modelling which provides an insight to the completion time and cost for executing the workflows. For this workflow tasks are executed first on VMs with high computational speed which returns minimum completion time and maximum cost incurred in executing each task and collectively the whole workflow. Similarly, the workflow tasks are executed on virtual machines with lowest configuration which results in maximum completion time and minimum cost. The initial executions are used for deciding the scaling factor which helps in recalculating the predicted makespan when remaining makespan is greater than the predicted makespan.

It also provides an insight to the maximum number of slots which are required for executing the workflow tasks, which helps in evaluating the ARM model by reducing the number of available resources to check the adaptability.

5 Results

The proposed ARM model is compared with a heuristic model with decisions to provision and allocate resources is based on best-fit approach. The resources are added if enough resources are not available to either accommodate the number of tasks or acceptable requirements. The competence of the heuristic model is increased as resources are allocated in order to meet the user constraints for deadline and cost.

Figure 2 presents the makespan to deadline ratio for all types of workflow using the ARM model architecture. Ratio value greater than one indicate a makespan larger than the deadline, value equal to one indicating makespan equal to the deadline and value less than one indicated makespan less than deadline. The deadlines are increased marginally and is based on the makespan of the critical path. The first deadline is too strict for the workflow to complete under the constrained makespan, but the marginal difference of 0.03% is unlikely to have a significant impact on the overall cost. As the deadlines are relaxed the workflows complete within constraints and hence a significant improvement in performance and utilization of resources is depicted.

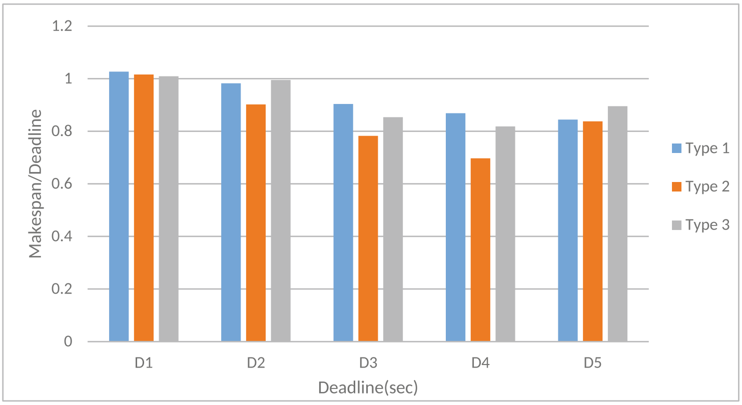


Fig. 2. Makespan to deadline ratio obtained for three types of workflows

Figure 3 presents the type and average amount of resources utilized to execute workflow tasks. It is clear from the graphical representation that resource acquired for deadline one are of the most powerful VM type, which shows the urgency in completing the workflow as fast as possible. As the deadline gets nominal, resources with lower frequency are used. ARM model acquires resource which are sufficient to complete the workflow within user defined constraints.

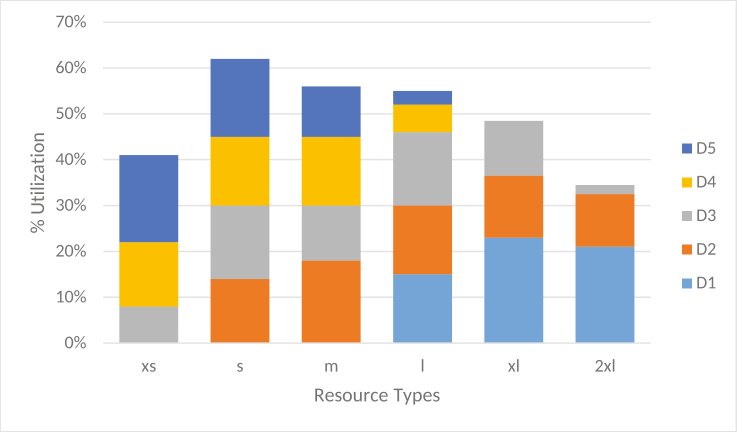


Fig. 3. Average utilization of resource type for workflow types with different deadline

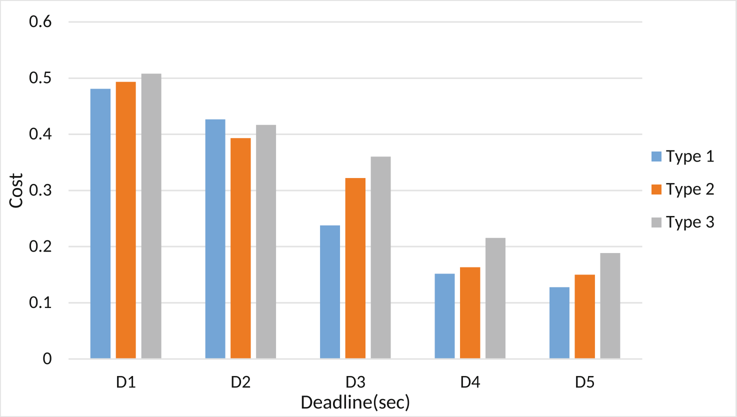


Fig. 4. Cost of executing workflow type with different deadline

Figure 4 shows the cost of executing workflow types with different deadlines. The cost is higher when the deadline is the tightest. This is due to the decision of executing the workflow with most powerful and expensive VM in order to finish the execution in time. In regards to the first deadline, the infrastructure cost for subsequent deadlines decreases as they are more relaxed in terms of execution time. The factor of adapting tasks from over-utilized to less-utilized resources also boosts the execution time and cost as it improves the performance of resources.

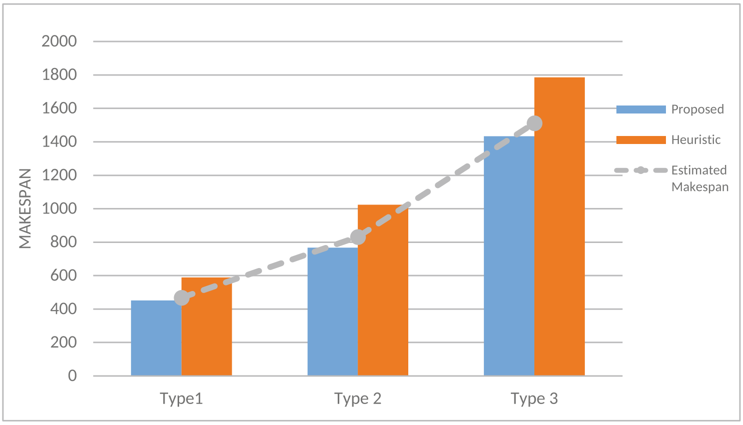


Fig. 5. Makespan for ARM model and heuristic model along with the estimated makespan

Figure 5 illustrates the makespan for different types of workflows as listed in the Sect. 4.1. It also presents the estimated makespan of the workflow, which indicates the efficiency of the ARM model to that of the heuristic model. The ARM model decreases the makespan for type 1 workflow by 23.26%, for type 2 workflows by 25.10% and by 19.76% for type 3 workflows. As depicted in the Fig. 5, the proposed model is proximately 94.5% accurate in achieving the estimated makespan. Continuous monitoring of resources and workflow tasks enabled the system to make decisions for adapting the tasks resulting in decreased workflow makespan.

ARM model employed monitoring and decision components for adapting workflow tasks from over-utilized resources to either under-utilized resources or newly provisioned resources. The adaptations notably increased the performance of resources and efficiency in tasks execution. Figure 6 presents the performance metric speedup which increases by migrating tasks to accommodate variable utilization of resources which further reduces makespan and cost of execution. The model led to a speedup of 15% for type 1 workflow with 9% task adaptations, 23% with 19% adaptations in type 2 workflow and 41% for type 3 workflow with nearly 37% adaptations.

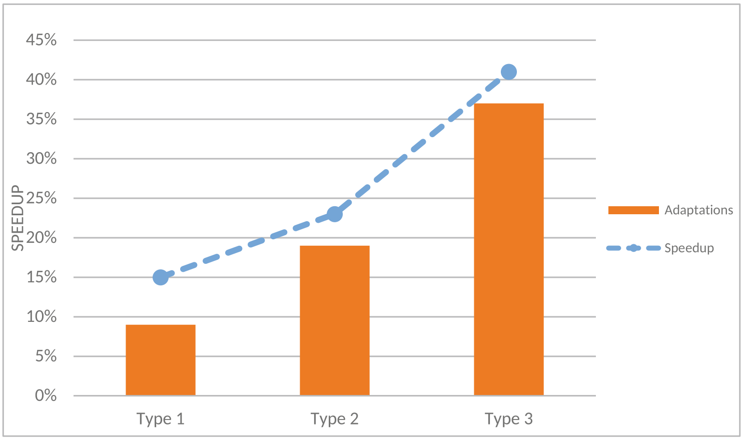


Fig. 6. Average Speedup with number of adaptations while executing different workflow types

6 Summary

This paper presented a reference model which is able to efficiently manage resources while executing workflow in clouds. The model monitors resources and compute the makespan of workflow tasks which assist in making decisions to improve performance and utilization of cloud resources. Adaptations of tasks from over-utilized to less-utilized resources improves the overall speedup of the workflow. The proposed model provisions and allocate right amount of resources by analysing the current execution time of the workflow. The model is evaluated using three workflow types with ranging parameter and is compared to a heuristic model. The experimental results presents that task adaptations can be leveraged to meet workflow QoS requirement by increasing utilization and performance of cloud resources thus benefiting cloud provider. In addition to this the model can be made more predictive using learning algorithms to increase resource performances and workflow makespan by reducing provisioning delays.

References

1. Jung, G., Hiltunen, M.A., Joshi, K.R., Schlichting, R.D., Pu, C.: Mistral: dynamically managing power, performance, and adaptation cost in cloud infrastructures. In: Proceedings - International Conference on Distributed Computing Systems, pp. 62–73 (2010)
2. Gong, Z., Gu, X., Wilkes, J.: Press: predictive elastic resource scaling for cloud systems. In: 2010 International Conference on Network and Service Management, pp. 9–16. IEEE (2010)
3. Huang, Q., Shuang, K., Xu, P., Li, J., Liu, X., Su, S.: Prediction-based dynamic resource scheduling for virtualized cloud systems. J. Netw. **9**(2), 375–383 (2014)

4. Jiang, Y., Perng, C.S., Li, T., Chang, R.: ASAP: a self-adaptive prediction system for instant cloud resource demand provisioning. In: Proceedings - IEEE International Conference on Data Mining, ICDM, pp. 1104–1109 (2011)
5. Ali-Eldin, A., Tordsson, J., Elmroth, E.: An adaptive hybrid elasticity controller for cloud infrastructures. In: Proceedings of the 2012 IEEE Network Operations and Management Symposium, NOMS 2012, vol. 978, pp. 204–212 (2012)
6. Buyya, R., Calheiros, R.N., Li, X.: Autonomic cloud computing: open challenges and architectural elements. In: Proceedings - 2012 3rd International Conference on Emerging Applications of Information Technology, EAIT 2012, pp. 3–10 (2012)
7. Imai, S., Chestna, T., Varela, C.A.: Accurate resource prediction for hybrid IaaS clouds using workload-tailored elastic compute units. In: Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, pp. 171–178. IEEE Computer Society (2013)
8. Jamshidi, P., Ahmad, A., Pahl, C.: Autonomic resource provisioning for cloud-based software. In: Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2014, pp. 95–104. ACM, New York (2014)
9. Sedaghat, M., Hernandez-Rodriguez, F., Elmroth, E.: Autonomic resource allocation for cloud data centers: a Peer to Peer approach. In: Proceedings - 2014 International Conference on Cloud and Autonomic Computing, ICCAC 2014, pp. 131–140 (2015)
10. Doyle, J., Giotsas, V., Anam, M.A., Andreopoulos, Y., Member, S.: Cloud instance management and resource prediction for computation-as-a-service platforms. In: Proceedings of IEEE International Conference on Cloud Engineering (IC2E), vol. 131983, pp. 1–11 (2016)
11. Perez, J., Germain-Renaud, C., Kégl, B., Loomis, C.: Multi-objective reinforcement learning for responsive Grids. J. Grid Comput. **8**(3), 473–492 (2010)
12. Arabnejad, H., Barbosa, J.G., Prodan, R.: Low-time complexity budget deadline constrained workflow scheduling on heterogeneous resources. Future Gener. Comput. Syst. **55**, 29–40 (2016)
13. Bittencourt, L.F., Madeira, E.R.M.: HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds. J. Internet Serv. Appl. **2**(3), 207–227 (2011)

A New Priority Heuristic Suitable in Mobile Distributed Real Time Database System

Prakash Kumar Singh(✉) and Udai Shanker

Department of Computer Science and Engineering,
MMM University of Technology, Gorakhpur, UP, India
pks.cse13@gmail.com, udaigkp@gmail.com

Abstract. Priority heuristic policies have been developed for centralized and distributed real time database systems where cohorts or sub transaction executed in sequential manner, however, these heuristics may not fit well for the mobile distributed real time database systems (MDRTDBS) where sub transactions are performing parallel execution and faces a lot of wireless challenges. In this paper, a MDRTDBS model has been introduced where sub-transaction executed parallel on different mobile sites and proposed a heuristic based on number of write locks. Proposed heuristic improves overall system performance by favoring sub transaction which demands lesser number of write locks. Further, a study has been done to evaluate impact of proposed heuristics with earliest deadline first and heuristic based on number of locks required using distributed high priority two phase locking protocol.

Keywords: Real time · Transaction · Priority heuristic · Mobile database

1 Introduction

Now-a-days, use of portable mobile devices and real time applications are becoming essential part of daily life. Researchers are focusing on these real time supporting systems [1–4]. In the past few decades, research in distributed real time database system (DRTDBS) [5–7] received a great attention. However, recent advances in mobile technology introduced a new era of research challenges in the field of MDRTDBS [8, 9, 18]. MDRTDBS are a collection of mobile and fixed devices (or participants), which are connected through wired or wireless channels and share and store the available resources [8, 9, 13–16]. They perform multiple concurrent transactions which are integrated with a time constraint. To minimize transaction miss rate, various priority heuristic (PH) approaches are integrated with CC protocols which decide the sequence of transaction's execution. Researchers have introduced heuristic based transaction scheduling [4, 10].

To the best of our knowledge, only two CCs policies are developed in MDRTDBS [8, 9]. PH may not fit well for the MDRTDBS when sub transactions are performing parallel execution [9, 11]. Shaker et al. [11] have developed a heuristic method to support parallel sub transaction execution on different sites for DRTDBS. Singh et al. [12] have also developed a PH using optimistic concurrency control in mobile environment. However, it is not much appropriate for pessimistic concurrency control

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 330–335, 2018.
https://doi.org/10.1007/978-3-319-72344-0_29

policies such as distributed high priority two phase-locking (DH2PL) [8]. Instead of past heuristic, where a sub transaction has been inherited the priority from its parent transaction, we have assigned priority on the basis of the number of write locks required by a sub transaction at a particular site. In this paper, Sect. 2 introduces MDRTDBS transaction model. Section 3 introduces the DH2PL policy for MDRTDBS. In Sect. 4, we have discussed our proposed PH. Section 5 presents performance evaluation and simulation results. Section 6 concludes the paper.

2 MDRTDBS Model

The structure of our MDRTDBS model consist some fixed hosts (FHs), participant mobile hosts (MHs), fixed mobile support stations (MSSs) and different database servers, as given in Fig. 1 [9, 12]. Server and mobile site maintains a transaction generator, data manger, concurrency controller, transaction manager, a local database such as main memory, data manager, ready and wait queue [8, 9, 12, 17]. Server site and mobile site maintain data broadcasting component and data receiver respectively [15].

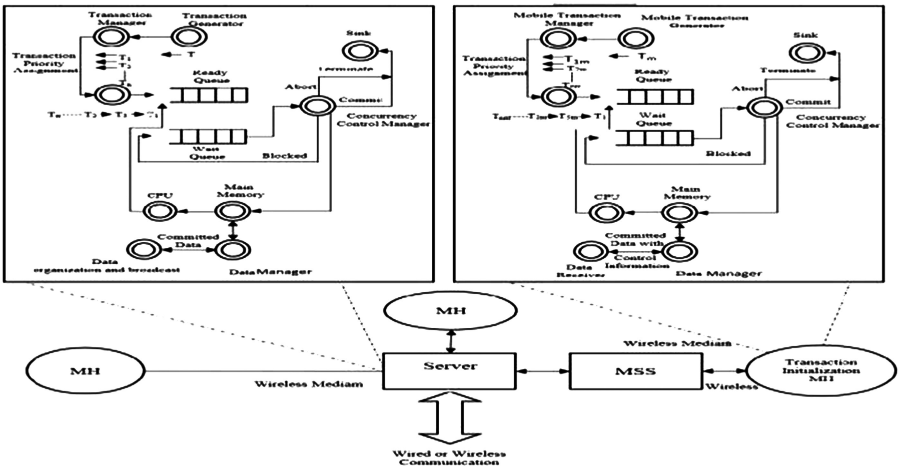


Fig. 1. Mobile distributed real time database model

In this model the transaction is initialized at mobile clients within its cell or from others cell. Mobile data manager manages the locally committed data items of read only transaction (ROT) [14, 16]. On the basis of transaction priority, the mobile cohorts are queued in ready queue and it has to pass through concurrency control mechanism to obtain a lock on the particular data item. The MDRTDBS model uses two phase commit (2PC) [13] to perform its commit procedure. Coordinator is responsible for the processing and committing of the transaction. The transaction operations will be processed one after one. If the required data item of an operation is located at another base station, the transaction will be forwarded to the base station. After the completion of a transaction, the mobile client will generate another transaction after a think time.

3 Distributed High Priority Two Phase Locking

Lam et al. [8] proposed a pessimistic protocol DH2PL which is first ever pessimistic CC protocol for MDRTDBS. It uses transaction restart and priority inheritance method [8] to reduce the problem of priority inversion. When priority inheritance is used in mobile wireless medium, the deadlock may be possible. To resolve this problem, DH2PL allows any (low or high priority) committing transaction to run unhindered and hold the data locks until it finishes its commit. Instead of restart, a priority inheritance concept is used to raise the priority of low priority committing transaction such that its priority will become slightly higher than (a) all blocked requesting high priority transactions and (b) all other executing transactions.

4 Proposed Heuristic Approach

Instead of both the number of read and write locks [11], the new heuristic approach is restricted only on number of write locks required by the cohorts. If N_w is number of write locks required by the cohort, initial priority (Init_P) of each cohort T_i is computed as Init_P = $1/N_{w,}$ (where, $N_w \geq 1$). Priority of cohort is inversely proportional to number of write locks required. The focus is on only write operations for deciding deadline of a transaction because of two reasons: (a) the percentage of ROT is much more than update transactions [9, 14, 16] and (b) write locks have a much more risk of data conflict with other transactions and it takes much more communication and processing time [8]. An intermediate priority assignment policy (I_Pr) is added with the proposed heuristic to minimize the transaction miss rate. Cohort's life time is based on two phases: Execution and Commitment phase [11]. In execution phase, the cohorts' lock data items, perform computation and, send WORKDONE message to their coordinators. Further, in case of dependency, WORKDONE message send is affected till removal of dependency [7]. Arrival of any intermediate higher priority cohort may abort the locks holding low priority cohort, if it has not sent PREPARED message.

I_Pr is used with the proposed heuristic to minimize the restart rate. In this policy an intermediate priority is assigned to newly arrived lock requesting cohort. This scheme is primarily based on the total remaining execution time Remain (T_{lh}) required for lock holding low priority cohort T_{lh} and the available slack time ST (T_{lr}) of the newly arrived higher priority cohort T_{lr}. Even if the requesting arrived cohort T_{lr} is higher priority than T_{lh}, the low abort of T_{lh} is possible only when the slack time of T_{lr} is lower than the total remaining execution time of T_{lh}. In this procedure this policy does not affect the initial priorities assigned to the two cohorts. Assume that Remain (T_i), Elapse (T_i), are the remaining time left and elapsed time of transaction execution respectively, whereas R_i, ST (T_i) are the remaining time needed of transaction and slack time of T_i respectively then Remain (T_i) of the lock holding cohort (T_i) is computed as: Remain $(T_i) = R_i - Elapse(T_i)$. In case of Remain ($T_{lh}$) is less than ST ($T_{lr}$), T_{lr} waits for the lock which is hold by executing T_{lh}, otherwise it force to T_{lh} for performing the abort procedure. The following is the algorithm developed for the above proposed heuristic with I_Pr. Assume that Pr (T_i) is the priority of cohort T_i. D_i is used for data item.

Algorithm:

```
Begin
 For each Arrived $T_{lr}$ {Assign Init_P to $T_{lr}$}
  For each data item
    If(! Lock_conflict) Allocate data item to $T_{lr}$
    Elseif(Pr ($T_{lr}$) > Pr ($T_{lh}$) and $T_{lh}$ is not committing)
                  For Each data item do\\$T_{lh}$ a Local or global Transaction
                    If( ST($T_{lr}$)< Remain ($T_{lh}$))Abort $T_{lh}$ and Allocate data item to $T_{lr}$
                    Else Insert $T_{lr}$ in wait queue
                    End_if
                  End_For
    Elseif(Pr($T_{lr}$)> Pr($T_{lh}$) and $T_{lh}$ is committing)
      Wait $T_{lr}$ until $T_{lh}$ unlock the locks and Pr ($T_{lh}$) = Pr ($T_{lr}$) + Thershold value
    ElseIf(Pr ($T_{lr}$) <= Pr ($T_{lh}$))Wait $T_{lr}$ until $T_{lh}$ releases the lock
    End_If    End_For End_For End
```

5 Performance Evaluation

5.1 Performance Parameters and Measures

The MDRTDBS with N_{sites} different sites is simulated on baseline setting and parameters of previous works [4, 8, 9, 11] using C language. Server distributes the cohorts among different sites. After completion of the transaction on different sites the next transaction will generate using think time. DH2PL is used to evaluate the impact of EDF, Heuristic approach [11], and proposed heuristic approach with I_Pr policy. In this paper, it has been assumed that, m is number of cohorts of transaction T, N is number of operations, T_l is time required to lock or unlock a data item, T_{pr} is the time to process a read or write data item, T_{comm} is communication delay between server and mobile node, N_{comm} is number of messages communicate between server and mobile host, T_{TPT} is total processing time of local or global transaction and SF is the slack factor. Than estimated deadline of a transaction Y is calculated as, $DL(Y) = AT(Y) + (T_{TPT} \times N + T_{comm} \times N_{comm}) \times (1 + SF)$. Further for local transaction, T_{TPT} is calculated as $T_{TPT} = 2T_l + T_{pr}$ and $T_{comm} = 0$.

However, for global transaction, for m cohorts, the value of T_{TPT} is computed as:

$$T_{TPT} = 2T_l + T_{pr} \times m.$$

The deadline of a mobile transaction is assigned as (C_Time + SF × P_Execution_Time), where C_Time is the current system time and P_Execution_Time is the predicted execution time and a function of transaction length. The results are based on ten independent runs. Every independent run are initialized with 2000 transaction. Miss rate and restart rate are two performance metrics of our simulation [4, 8, 9, 11].

5.2 Simulation Result

A comparison has been performed among our heuristic policy (HP_NWL), basic EDF based policy and a policy proposed by shanker et al. [11] for firm real time transactions. In Figs. 2 and 3, notation PH_EDF and PH_NL is used for EDF based policy and policy proposed by shanker et al. respectively. Figure 2 shows the miss rate of DH2PL with these policies. The values of miss rate against think time clearly indicate that the proposed heuristic is most suitable in mobile environment. It can be seen that proposed heuristic approach with DH2PL improves the overall MDRTDBS performance in terms of Miss rate and Restart rate. In our simulation, different transaction execution time has been considered. To calculate cohort deadline it is assumed that the all operations of the cohort are local operations. In Fig. 2, the transaction miss rate corresponding to the think time shows that an increase in think time decreases the miss rate. The cause behind this is the system decrease lower the workload value when think time value is increases. Similarly, in the Fig. 3, the graph shows the restart rate decreases with respect to increase of think time. In our simulation, we have study the performance of the proposed heuristic with high data contention, a uniform distribution of operations (N_{oper} = 7 to 14) in global as well as local transactions are introduced. The range of think time is varied from 1 to 7 s.

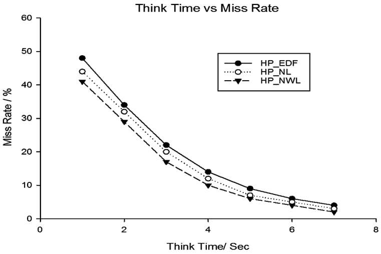


Fig. 2. Miss rate vs Think time

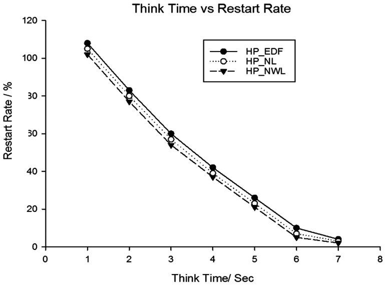


Fig. 3. Restart rate vs Think time

In Fig. 2, its look clearly that the performance of the proposed approach is consistently better than the other two policies. In Fig. 3, using restart rate metric, it has been found that the proposed approach HP_NWL perform better than PH_NL and PH_EDF.

6 Conclusion

This paper details MDRTDBS model and proposes a heuristic policy much suitable in mobile environment. The proposed heuristic policy, HP_NWL, has been introduced using intermediate priority assignment policy. The proposed heuristic is compared with HP_EDF and HP_NL via simulation. The proposed heuristic performs well than other two methods. It can be further extended for hard real time transactions.

References

1. Abbott, R.K., Molina, H.G.: Scheduling real time transactions: a performance evaluation. ACM Trans. Database Syst. **17**(3), 513–560 (1992)
2. Haritsa, J.R., Carey, M.J., Livny, M.: Data access scheduling in firm real-time database systems. J. Real Time Syst. **4**(3), 203–242 (1992)
3. Lam, K.Y., Lee, V.C.S., Hung, S.L., Kao, B.C.M.: Priority assignment in distributed real-time databases using optimistic concurrency control. IEE Proc. Comput. Digital Tech. **144**(5), 324–330 (1997)
4. Lee, V.C.S., Lam, K.Y., Kao, B.C.M., Lam, K.W., Hung, S.L.: Priority assignment for sub-transaction in distributed real-time databases. In: International Workshop on RTDBS (1996)
5. Lam, K.Y.: Concurrency control in distributed real-time database systems. Ph.D. Thesis, Department of Computer Science, City University of Hong Kong (1994)
6. Shanker, U., Misra, M., Sarje, A.K.: SWIFT: a new real time commit protocol. Distrib. Parallel Databases **20**(1), 29–56 (2006)
7. Shanker, U., Misra, M., Sarje, A.K.: Distributed real time database systems: background and literature review. Int. J. Distrib. Parallel Databases **23**(2), 127–149 (2008). Springer-Verlag
8. Lam, K.Y., Kuo, T.-W., Tsang, W.-H., Law, G.C.K.: Concurrency control in mobile distributed real-time database. J. Inf. Syst. **25**(4), 261–286 (2000)
9. Lei, X., Zhao, Y., Chen, S., Yuan, X.: Concurrency control in mobile distributed real-time database systems. J. Parallel Distrib. Comput. **69**, 866–876 (2009)
10. Kao, B., Molina, H.G.: Deadline assignment in a distributed soft real-time system. In: Proceedings of 13th International Conference on Distributed Computing Systems, pp. 428–437 (1993)
11. Shanker, U., Misra, M., Sarje, A.K.: Priority assignment heuristic to cohorts executing in parallel. In: Proceedings of the 9th WSEAS International Conference on Computers, World Scientific and Engineering Academy and Society (WSEAS), pp. 1–6 (2005)
12. Singh, P.K., Shanker, U.: Priority heuristic in mobile distributed real time database using optimistic concurrency control. In: International Conference on Advanced Computing and Communications (ADCOM 2017), Bangalore, 8–10 September 2017 (2017)
13. Gray, J.N.: Notes on database operating systems. In: Operating Systems: An Advanced Course, vol. 60, pp. 397–405 (1991)
14. Lee, V.C.S., Lam, K.W., Son, S.H.: Real-time transaction processing with partial validation at mobile clients. In: Proceedings of Seventh International Conference on Real-Time Computing Systems and Applications, pp. 473–477. IEEE (2000)
15. Lee, V.C.S., Lam, K.W., Son, S.H., Chan, E.Y.M.: On transaction processing with partial validation and timestamp ordering in mobile broadcast environments. J. IEEE Trans. Comput. **51**(10), 1196–1211 (2002)
16. Lee, V.C.S., Lam, K.W., Kuo, T.W.: Efficient validation of mobile transactions in wireless environments. J. Syst. Softw. **69**(1), 183–193 (2004)
17. Herman, G., Lee, K.C., Weinrib, A.: The datacycle architecture for very high throughput database systems. Proc. ACM SIGMOD Rec. **16**(3), 97–103 (1987)
18. Swaroop, V., Shanker, U.: Mobile distributed real time database systems: research challenges. In: Proceedings of International Conference on Computer and Communication Technology (ICCCT 2010), MNNIT, Allahabad, India, 17–19 September 2010 (2010)

Efficient Algorithms for Local Density Based Anomaly Detection

Ankita Sinha(✉) and Prasanta K. Jana

Department of Computer Science and Engineering,
Indian Institute of Technology (ISM), Dhanbad, India
ankitasinha051@gmail.com, prasantajana@yahoo.com
http://www.iitism.ac.in

Abstract. Anomaly detection is a crucial problem in the field of data mining. However, prevailing anomaly detection algorithms are serial in nature which fail to handle huge volume of data. In this paper, we propose two parallel local density based algorithms namely, MapReduce based Local Outlier Factor (MRLOF) and Spark based Local Outlier Factor (SLOF). The proposed algorithms have time complexity of O(N) for each. This is an improvement over the Simplified LOF (Local Outlier Factor) which has time complexity of $O(N^2)$, where N is the data size. We conducted extensive experiments with MRLOF and SLOF on various real life and synthetic datasets. The proposed algorithms are shown to outperform the serial Simplified LOF.

Keywords: Anomaly detection · Local Outlier Factor
Apache Hadoop · Apache Spark · Big data

1 Introduction

An abundance of data is generated from a plethora of sources, which needs to be processed and analyzed to infer knowledge. Increase in size has led to growth in noise and abnormality in data [1]. The data need to be tested for the instances which deviates considerably from the general trend. These elements are called anomalies or outliers [2]. Identification of anomalies in data is crucial in many fields such as fraud detection, network intrusion, medical diagnosis, ecosystem disturbance [3], programming defects etc.

Anomalies can be detected by applying various techniques such as proximity based, cluster based, density based etc. [2,3]. First local outlier method called LOF (Local Outlier Factor) was proposed by Breunig et al. [4]. It estimates the outlier score (LOF) of an object based on its neighborhood, which is determined by its k nearest neighbors. A higher value of LOF indicates outlierness of an object. Many more algorithms like LDOF [6], Simplified LOF [7], LoOP [5], KDEOS [5] etc. have tried to improve upon the basic LOF. Although the algorithms did provide some improvements, they were serial in nature with high time complexity.

A. Negi et al. (Eds.): ICDCIT 2018, LNCS 10722, pp. 336–342, 2018.
https://doi.org/10.1007/978-3-319-72344-0_30

In this paper, we propose two parallel algorithms for Simplified LOF. We call the algorithms MRLOF (MapReduce based Local Outlier Factor) and SLOF (Spark based Local Outlier Factor) as they are based on MapReduce [8–10] and Spark [11,12] respectively. In the proposed techniques, the pair wise distance computation is accomplished on different nodes in cluster, thereby reducing time complexity by a factor of N. Therefore, time complexity of our proposed algorithms is $O(N)$. Moreover, the algorithms are based on relative density and hence provide a qualitative measure of level of outlierness of an object even when data is not uniformly distributed. Organisation of remainder of the paper is as follows. We introduce local outlier detection using relative density in Sect. 2 followed by our proposed work in Sect. 3. Experiments and results is provided in Sect. 4. Finally, we conclude our paper in Sect. 5.

2 Outlier Detection Using Relative Density

In density based outlier detection schemes, outlier score of an object is inversely proportional to the distance of its k nearest neighbors. Density of an object d_i for k nearest neighbors is computed by using Eq. 1.

$$density(d_i, k) = \left(\frac{\sum_{y \in Near(d_i,k)} distance(d_i, y)}{|Near(d_i, k)|} \right)^{-1} \tag{1}$$

Here, $|Near(d_i, k)|$ is the set containing the k nearest neighbors of x, y is one of the nearest neighbor and $distance(x, y)$ is Euclidean distance between the points$x$ and y. Density around any data element is calculated by the number of elements around a predefined global parameter d, which might lead to ambiguities in regions of varying densities [3]. Local approach where density is defined in terms of neighborhood is more efficient. Average relative density of an object d_i w.r.t its k nearest neighbors average density is defined in Eq. 2.

$$avg\ relative\ density\ (ad_i) = \frac{density(d_i, k))}{\sum_{y \in Near(d_i,k)} density(y, k) / |Near(d_i, k)|} \tag{2}$$

The outlier score of an object d_i is calculated based on its average relative density. It provides a measure to identify whether a point d_i lies in dense or sparse region.

3 Proposed Work

3.1 MapReduce Based Local Outlier Factor Algorithm (MRLOF)

MapReduce parallel programming paradigm reads data from HDFS one line at a time. It mainly consists of two tasks, map task and reduce task, which works exclusively on $<key, value>$ pair [8]. An illustrative example of transition of

<*key*, *value*> pair for MRLOF algorithm is shown in Table 1. Here, a dataset $D = \{a, b, c, d, e\}$ is considered, which is stored in HDFS as two data blocks, $\{a, b, c\}$ and $\{d, e\}$. The initial *key* input to map task is byte offset and *value* is string record in one line of the data block. Map task reads one element and sends it to the reduce tasks for distance computation by assigning appropriate *key* and corresponding *value*. A *tag* is attached to distinguish between elements going to one reduce task. Point from which distance is to be calculated is given *tag* '0' and remaining elements have *tag* '1'. Map task assigns *key* ranging from 1 to N, such that each object is sent to all the reducers. Framework collects the

Table 1. Transition of <*key*, *value*> pair in MRLOF

Mapper			Reducer			
Input	Output		Input		Output	
Value (Text)	Key (Text)	Value (Text)	Key (Text)	Value (Text)	Key (Text)	Value (Text)
a	1	a;1	1	a;1, b;0, c;0, d;0, e;0	a	$Near_a$ & $Density_a$
	2	a;0				
	3	a;0				
	4	a;0				
	5	a;0				
b	1	b;0	2	a;0, b;1, c;0, d;0, e;0	b	$Near_b$ & $Density_b$
	2	b;1				
	3	b;0				
	4	b;0				
	5	b;0				
c	1	c;0	3	a;0, b;0, c;1, d;0, e;0	c	$Near_c$ & $Density_c$
	2	c;0				
	3	c;1				
	4	c;0				
	5	c;0				
d	1	d;0	4	a;0, b;0, c;0, d;1, e;0	d	$Near_d$ & $Density_d$
	2	d;0				
	3	d;0				
	4	d;1				
	5	d;0				
e	1	e;0	5	a;0, b;0, c;0, d;0, e;1	e	$Near_e$ & $Density_e$
	2	e;0				
	3	e;0				
	4	e;0				
	5	e;1				

values with same *key* and sends them to reduce task. Input to reducer is of the form <*key, list* <*values*>>. Reduce task then finds k nearest neighbors of each element, density and outlier score using Eq. 2. Pseudo code for MRLOF is given in Algorithm 1.

Algorithm 1. MRLOF Algorithm

Require: number of nearest neighbors k, data object d_i, data size N
Ensure: $Near_{key}$ and $Density_{key}$

```
Start map ($key$, $value$)
$count$=0
$x = d_i$+';'+1 $context.write(count, x)$
for ($i = 1$ $to$ $N$) do
    x =$d_i$+';'+0
    if $i$=count then continue
    end if
    $context.write(count, x)$
end for
Stop $map()$
Start $reduce < key, list < values >>$
A shared variable Density is initialized to 0 for each key.
while ($v'.hasNext()$) do
    $temp$=0
    if ($tag = 1$) then
        point=v'.value
    else
        $data_i$=+$v'.value$
    end if
end while
for ($i = 1$ $to$ $N$) do
    $distance_i$=$dis(point, data_i)$
end for
for ($i = 1$ $to$ $k$) do
    $near_{key}$=$k$ nearest neighbors of point
end for
Find $Density_{key}$ using Eq. 1
$context.write(key, near_{key}$ $and$ $Density_{key})$
Stop $reduce()$
```

3.2 Spark Based Local Outlier Factor Algorithm (SLOF)

SLOF works on the same principle as MRLOF. However, unlike MapReduce Spark used RDDs as data structure [11]. The Lineage graph which depicts the transformations of RDD, in SLOF algorithm is given in Fig. 1. Input data is attached the *tag* by using $flatMap()$ method and generates RDD RevisedData.

After that <$key, value$> pairs are generated by using $map()$ and saved in RDD PairedData, which is then grouped by key in GroupedData RDD. Pair wise distance between points is calculated, which is then used to calculate k nearest neighbors and are stored in RDD pointWithDistance and KNearestNeighbors respectively. Finally, density of each object is calculated and saved back in the HDFS along with the k nearest neighbors of each point.

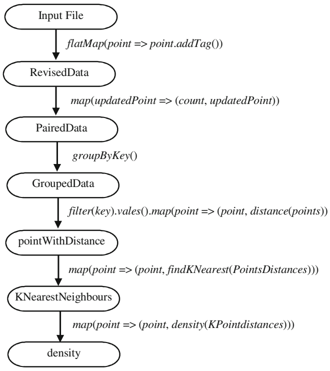

Input File
flatMap(point => point.addTag())
RevisedData
map(updatedPoint => (count, updatedPoint))
PairedData
groupByKey()
GroupedData
filter(key).vales().map(point => (point, distance(points))
pointWithDistance
map(point => (point, findKNearest(PointsDistances)))
KNearestNeighbours
map(point => (point, density(KPointdistances)))
density

Fig. 1. Lineage graph of RDD's in SLOF

4 Experiments and Results

A fully distributed heterogenous cluster of 5 nodes was set up. Three nodes were configured on a Server machine, ML 350E Gen 8 with Intel Xeon E5-24070 @2.20 GHz CPU and 24 GB RAM. One VM was made to work as master was allotted 8 GB RAM and remaining two VMs worked as slave and were allocated 6 GB RAM. Other two slaves in the cluster were workstations with Intel i7 gen2 and Intel i3 gen 2 processors and 8 GB RAM. Hadoop version 2.7.1 [10] and Spark version 2.1.1 [12] was installed on each machine. MapReduce codes were written Java version 1.7.0 and Spark programs were written in Scala programing language, version 2.11. We evaluated the performance of our proposed methodologies on four real datasets glass, spambase, Shuttle and pima diabetes in Indian patients [13]. Description of the datasets is provided in Table 2. In order to check scalability of MRLOF and SLOF algorithms, we generated synthetic datasets of varying size. The number of data points was increased from one thousand through one million, incrementing by a factor of 10.

4.1 Performance Evaluations

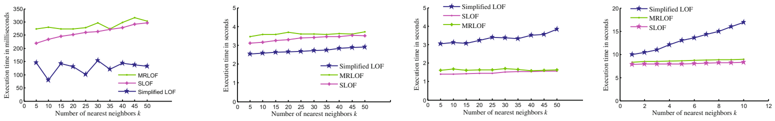
First, we evaluate the performance of our proposed techniques with respect to the Simplified LOF for the real life datasets. For all four datasets the number of nearest neighbors k was varied from 5 to 50 and the execution time was

Table 2. Description of the datasets

Dataset	Number of observation	Number of attributes
Glass	205	7
Spambase	4601	57
Shuttle	58000	9
Pima Indians Diabetes	768	8

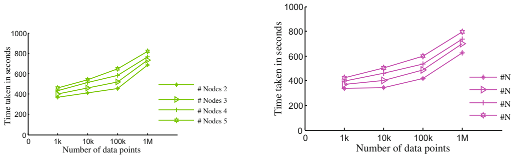
recorded. Figure 2 shows the execution time of MRLOF and SLOF in comparison to Simplified LOF. For small sized data like glass, pima Indian diabetes, the execution time of Simplified LOF is lesser in comparison to parallel version because of the overhead in parallelization. However, with the increase in size of data the efficiency of parallel algorithms increases. To compare the Simplified LOF with parallel algorithms, standalone cluster was used.

To evaluate the scalability of MRLOF and SLOF, we increased the data size by a factor of 10 and implemented both MRLOF and SLOF by varying the number of nodes in the cluster from 2 to 5. The scalability performance of MRLOF and SLOF is depicted in Fig. 3-a and 3-b respectively. It can be inferred from the graphs that with the increase in number of nodes in cluster the execution time decreases. It is also worth noting that the execution time of MRLOF and SLOF are approximately same, as MRLOF is a non iterative algorithms and hence does not gain much in Spark implementation i.e. SLOF. From the results it can be derived that the proposed algorithms are highly scalable.



(a) Glass (b) Pima Indians diabetes (c) Spam base (d) Shuttle

Fig. 2. Execution time for different real life datasets



(a) MRLOF algorithm (b) SLOF algorithm

Fig. 3. Scalability of proposed algorithms

5 Conclusion

In this paper, we presented two parallel local density based algorithms to detect anomalies in big data, MRLOF and SLOF. The algorithms works efficiently to identify outliers for various real life datasets. The Experimental results also demonstrated the algorithms are highly scalable and almost showed a linear gradient with the increase in size of dataset and addition of nodes in cluster.

Acknowledgements. The authors would like to thank Council of Scientific and Industrial Research (CSIR), New Delhi, India for the financial support for this research work (File No. 09/085(0111)/2014.EMR.1).

References

1. Hayes, M.A., Capretz, M.A.: Contextual anomaly detection framework for big sensor data. J. Big Data **2**(1), 2 (2015)
2. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. (CSUR) **41**(3), 15 (2009)
3. Tan, P.N., Kumar, V., Steinbach, M.: Introduction to Data Mining. Pearson Education, India (2011)
4. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. ACM SIGMOD Rec. **29**(2), 93–104 (2000)
5. Schubert, E., Zimek, A., Kriegel, H.P.: Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. Data Min. Knowl. Disc. **28**(1), 190–237 (2014)
6. Zhang, K., Hutter, M., Jin, H.: A new local distance-based outlier detection approach for scattered real-world data. In: Advances in Knowledge Discovery and Data Mining, pp. 813–822 (2009)
7. Schubert, E., Zimek, A., Kriegel, H.P.: Generalized outlier detection with flexible kernel density estimates. In: Proceedings of the 2014 SIAM International Conference on Data Mining, pp. 542–550. Society for Industrial and Applied Mathematics, April 2014
8. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2008)
9. Sinha, A., Jana, P.K.: A novel K-means based clustering algorithm for big data. In: 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1875–1879. IEEE, September 2016
10. Apache Hadoop. http://hadoop.apache.org/
11. Karau, H., Konwinski, A., Wendell, P., Zaharia, M.: Learning Spark: Lightning-Fast Big Data Analysis. O'Reilly Media, Inc., USA (2015)
12. https://spark.apache.org. Accessed 9 Aug 2017
13. http://archive.ics.uci.edu/ml/index.php. Accessed 14 Aug 2017

Author Index