Volume 41 Number 3 September 2017 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

1977

Editorial Boards

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Higher Education, Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

Bipartivity Index based Link Selection Strategy to Determine Stable and Energy-Efficient Data Gathering Trees for Mobile Sensor Networks

Natarajan Meghanathan
Professor, Department of Computer Science, Jackson State University
Jackson, MS, USA
E-mail: natarajan.meghanathan@jsums.edu

Keywords: link stability, mobile sensor networks, bipartivity index, data gathering trees, simulations

Received: August 12, 2016

Bipartivity Index (BPI) has been used in complex network analysis to quantify the extent of partitioning of the vertices of a network graph into two disjoint partitions; the edges between vertices within the same partition are called frustrated edges. The BPI values for a network graph ranges from 0 to 1 (the BPI of a network graph that is truly bipartite and has no frustrated edges is 1). Our hypothesis in this research is that the end nodes of a short distance link (the distance between the end nodes is significantly smaller than the transmission range per node) in a mobile sensor network (MSN) are more likely to share a significant fraction of their neighbors and such links are more likely to be stable. We introduce a notion called the egocentric network of an edge (adapted from egocentric network for a node) comprising of the end nodes of the edge and their neighbors (as vertices) and the edges incident on the end nodes (as edges). Our claim is that an edge whose egocentric network has a lower BPI score is more likely to be a stable short distance link, with a relatively larger fraction of shared neighborhood, and could be preferred for inclusion while determining stable data gathering trees for MSNs. Through extensive simulations, we show that the BPI-based DG trees are significantly more stable and energy-efficient compared to the DG trees determined using the predicted link expiration time (LET), currently the best known strategy.

Povzetek: Prispevek s pomočjo BPI indeksa ugotavlja stabilna in energijsko učinkovita drevesa za mobilne senzorske mreže.

1 Introduction

Mobile Sensor Networks (MSNs) are an emerging category of wireless sensor networks in which the sensor nodes are considered to move independent of each other. MSNs could be used for applications in which an entire region (that is being monitored) could be effectively covered by letting the sensor nodes to move rather than be static. For example [9], the pollutant concentration in an area (like the downtown of a city) could be effectively measured by fixing the sensor nodes in mobile vehicles (like cars) that move through the area. For most of the applications of wireless sensor networks (including those of the MSNs), the data recorded by the sensor nodes is forwarded to a control center (called the sink) through one of several network-wide communication topologies (like chains [11], clusters [7], trees [18], connected dominating sets [16], etc). Among these communication topologies, the data gathering trees (DG trees) have been observed to be energy-efficient [18] as they comprise of the minimum number of links needed to span all the sensor nodes and there are no redundant transmissions. In the case of DG trees, the leaf nodes merely sense the data and transmit them to an upstream intermediate node that would in turn aggregate its own data with data received from all of its child nodes and forward the aggregated data to an upstream node that is on the path to the root node of the DG tree. For the rest of the paper, the terms 'node' and 'vertex', 'link' and 'edge', 'network' and 'graph', 'data gathering' and 'data aggregation', 'construction' and 'configuration' mean the same. These terms are used interchangeably unless stated.

MSNs inherit all the constraints of their static counterpart (like energy and memory-constrained sensor nodes as well as limited network bandwidth); mobility of the nodes is an additional constraint that needs to be handled. Due to node mobility, the network topology changes dynamically with time and any communication topology (like DG trees) that is setup among the sensor nodes needs to be frequently reconfigured. Significant amount of energy might be lost if network-wide broadcasts are frequently initiated for reconfiguring the communication topology in use. This motivates the need to determine stable communication topologies that could exist for a longer time.

In [19], the authors took the first step towards using DG trees for MSNs and proposed a distributed algorithm for determining stable DG trees in MSNs using the concept of predicted link expiration time (LET) [31] that has been earlier successfully used for mobile ad hoc networks [20, 22]. In [24], the authors proposed a generic algorithm to determine maximum bottleneck link weight (MaxBLW)-based DG trees for static sensor networks: the bottleneck link weight for a path from a node to the root node of the DG tree is the minimum of the weights of the constituent links on the path and the MaxBLW-DG algorithm determines a DG tree in which the path from any node to the root node of the tree is the path with maximum value for the bottleneck link weight. In this paper, we explain a distributed version of the MaxBLW-DG algorithm to determine ALGC-based DG trees wherein the link weight is the link stability score (LSS) computed based on this strategy. For performance comparison purposes, we use the distributed version of the MaxBLW-DG tree algorithm to also determine the LET-based DG trees [19] wherein the weight of a link is its predicted LET.

The LET-based strategy is the only available link selection strategy that has been successfully demonstrated so far [19] for determining stable DG trees in MSNs. However, the LET formulation [19, 31] does not consider the distance between the constituent end nodes of a link and is prone to choosing links that could incur a larger transmission energy and ultimately contributing towards larger energy consumption per round. We opine that links whose constituent end nodes are closer to each other (i.e., the distance between the end nodes of the link is appreciably lower than the transmission range per node) are more likely to be stable (and vice-versa) as it would take a while for such end nodes to move out of the transmission range of each other. We refer to such links as "short distance" links. Also, as the energy lost per transmission is directly proportional to the square of the distance [28] over which the transmission is made, we claim that the DG trees comprising of short distance links are more likely to be both stable (and vice-versa) as well as incur lower energy consumption per round. Moreover, the LET approach [31] requires a sensor node to be aware of its own location and mobility as well as that of its neighbors. This would require the sensor nodes to be equipped with energy-draining hardware/software systems (like GPS [8]) that would make them location and mobility aware. All of the above observations form the motivation for the research conducted in this paper.

The high-level contribution of this paper is that we show the use of a spectral graph-theoretic metric called Bipartivity Index (BPI) [6] to quantify the extent of shared neighborhood between the end vertices of an edge and thereby model the link stability score (LSS) for the edge. The BPI has been widely used in complex network analysis [6] to quantify the extent of partitioning of a network graph into two disjoint partitions of vertices; the edges between vertices within the same partition are referred to as frustrated edges. BPI values range from 0 to 1 [6]. A network graph is said to be truly bipartite (such a partitioning also has no frustrated edges) if its BPI is 1 [6]. We propose to use a notion called the "egocentric network of an edge" (adapted from the notion of egocentric network of a node [13]) to quantitatively evaluate the extent of shared neighborhood between the end vertices of a link. The egocentric network of an edge *u*-*v* (denoted $EG_{u\text{-}v}$) comprises as vertices - the end nodes of the edge and their neighbors, and edges - the links incident on the end nodes of the edge. We claim that an edge *u*-*v* is more likely to be a stable short distance link with a larger fraction of shared neighborhood if the egocentric network $EG_{u\text{-}v}$ of the edge has a lower BPI. Accordingly, we model for an edge *u*-*v*: the LSS(*u*-*v*) as 1 - BPI($EG_{u\text{-}v}$).

We provide a high-level justification for the above modeling as follows (more details are presented in Section 4). If the end nodes of an edge *u*-*v* do not have any shared neighbors, then the egocentric network of the edge *u*-*v* would comprise of node *u* and the neighbors of node *v* in one of the two partitions, and node *v* and the neighbors of node *u* in the other partition; all the edges would connect the vertices in one partition to the other partition and there would be no frustrated edges within either partition (a frustrated edge is an edge involving vertices that are in the same partition [6]). Such an egocentric network is truly bipartite and will have a BPI of 1. Whereas, if the end nodes of an edge *u*-*v* have one or more shared neighbors, the egocentric network of the edge (when analyzed for bipartivity) would comprise of one or more frustrated edges contributing to a BPI less than 1. We anticipate the BPI for the egocentric network of an edge *u*-*v* to reduce with increase in the number of shared neighbors for the end nodes *u* and *v*.

The rest of the paper is organized as follows: Section 2 outlines the maximum bottleneck link weight-based algorithm for determining data gathering trees in sensor networks. Section 3 reviews related work, including the strategy of using the predicted link expiration time (LET) to determine stable data gathering trees for MSNs. Section 4 introduces the notions of short distance links, egocentric network for an edge and bipartivity index (BPI) as well as illustrates their use to quantify the extent of shared neighborhood and stability of links. Section 5 presents results of exhaustive simulations conducted to showcase the effectiveness of the BPI-based strategy to determine data gathering trees that are both stable as well as energy-efficient compared to the LET-based DG trees. Section 6 concludes the paper.

2 Distributed algorithm to construct a maximum bottleneck link weight-based data gathering tree

In this section, we describe a distributed version of the algorithm to construct maximum bottleneck link weight-based data gathering (MaxBLW-DG) trees for mobile sensor networks. A centralized version of the MaxBLW-DG algorithm has been earlier proposed in [24] and a distributed implementation of the algorithm to determine LET-based stable data gathering trees has been discussed in [19]. The distributed version of the MaxBLW-DG algorithm discussed here could be applied for any measure of link weight. For this section, we assume the link weights are randomly generated in the range [0...1].

In sections 4 and 5, the weight of a link depends on the link selection strategy (BPI or LET) employed.

2.1 Assumptions and definitions

We assume the sensor nodes to operate in a fixed transmission range, *R*. We assume the underlying network is modeled as a unit-disk graph wherein there exists a link between any two nodes if the Euclidean distance between them is within the transmission range, *R*. We assume the network to be homogeneous (i.e., all the nodes have an equal transmission range). We define the *fraction of link distance* (*fld*) as the ratio of the Euclidean distance between the end nodes of the link and the transmission range per node. In the case of heterogeneous networks (each node operating with a different transmission range), the fraction of link distance could be measured as the ratio of the Euclidean distance between the end nodes of the link and the maximum of the transmission ranges of the two end nodes. The data gathering algorithms (discussed in Sections 2 and 3) and the BPI strategy discussed in Section 4 could be used for both homogeneous and heterogeneous networks. For a directed edge $u \rightarrow v$, we refer to node *u* as the upstream node and node *v* as the downstream node. In the context of link weights, we assume the links/edges are undirected (bidirectional): i.e., the weight of a directed edge $u \rightarrow v$ is the same as the weight of the directed edge $v \rightarrow u$.

We define a round of data gathering to comprise of steps in which the sensor nodes individually sense the data within their sensing range (typically the sensing range of a sensor node is at most half its transmission range [36]), aggregate and forward only a representative version of the data (like the average temperature in a region) to the sink through a network-wide communication topology (like a data gathering tree) spanning all the sensor nodes. The size of the aggregated data is assumed to be the same as the size of the data collected at the individual sensor nodes. The root node of a DG tree is called the LEADER node and is chosen by the sink at the time of tree construction.

We assume the sensor nodes to be both TDMA (Time Division Multiple Access) and CDMA (Code Division Multiple Access)-enabled [35]. An upstream node communicates with its own immediate downstream child nodes using a TDMA schedule (one time slot per downstream node); such communication between every upstream node with their own downstream nodes can occur in parallel (using unique CDMA codes). The above assumptions and definitions hold good for both the BPI and LET-based MaxBLW-DG trees studied in this paper.

When used for constructing the LET-based DG trees, we assume a sensor node to be aware of its current location, velocity and direction of movement at any time instant and mentions the same in a location update vector (LUV) [19] included in the control messages broadcast as part of tree discovery. Such an assumption is not required for the MaxBLW-DG algorithm that makes use of BPI (discussed in Section 4) as the BPI scores could be computed without a priori knowledge about the location and mobility of the nodes. Each node maintains a *Link Weight Table* comprising of the estimates of the bottleneck link weights to the neighbor nodes that sent it the TREE-CONSTRUCT message (see Section 2.3 for more details about the message).

2.2 Initialization of state information at the sensor nodes

Each sensor node locally maintains state information about the data gathering tree that is currently being used or newly configured. The state information comprises of the following fields (with their initial values indicated in parenthesis): *estimated bottleneck link weight* (-∞), *upstream node id* (NULL), *tree level* (0), *LEADER node id* and *sequence number* (the latest sequence number in the TREE-INITIATE message broadcast by the sink). The *estimated bottleneck link weight* is the value for the currently known maximum weight for a link on the path to the LEADER node of the DG tree. The *upstream node id* corresponds to the neighbor node that lies on the currently estimated maximum bottleneck link weight path to the LEADER node. The *tree level* corresponds to the number of hops on the maximum bottleneck link weight path to the LEADER node. The *sequence number* corresponds to the latest sequence number for a TREE-CONSTRUCT message received by the node.

2.3 Initiation of the Tree-construct message

Whenever the sink fails to receive the aggregated data from the LEADER node of the DG tree used in the previous round of data aggregation, the sink queries all the sensor nodes to send it their estimates of the weight of the links from their neighbor nodes. The sink calculates the estimated weight of a node as the sum of the estimated weights of the directed edges originating from the node (as reported by its neighbors); the sink selects the node with the largest estimated weight to be the LEADER node (root node) of the new DG tree that is to be setup and sends a TREE-INITIATE message (including a sequence number) to the chosen root node to begin the construction of the new DG tree. The sequence number for the tree construction process is a monotonically increasing value maintained at the sink and the sink sends the latest value of the sequence number to the LEADER node to facilitate the sensor nodes to uniquely identify the control messages that are exchanged with regards to the new DG tree being constructed.

The LEADER node broadcasts a TREE-CONSTRUCT message to its neighbors; the message has a 5-element tuple: <sequence number, LEADER node id, upstream node id, sender's estimated bottleneck link weight, tree level>. The sequence number is the one that is sent by the sink to the LEADER node. For the TREE-CONSTRUCT message broadcast by the LEADER node, the values for the upstream node id, sender's estimated bottleneck link weight and tree level are respectively the LEADER node id, +∞ and 0. For the TREE-CONSTRCT message broadcast by the other nodes: the upstream node id is the id of the node that the sender of the message considers to be the best node that would connect it to the LEADER node through a path estimated to be the one with the maximum bottleneck link weight (the value of which is also indicated in the message). The tree level field indicates the number of hops on the estimated maximum bottleneck link weight path from the sender node to the LEADER node of the DG tree.

2.4 Propagation of the Tree-construct message

When a node receives the TREE-CONSTRCT message (from a neighbor node) with a higher sequence number, it assumes that the DG tree that had been used until then no longer exists and resets its state information to the values listed in Section 2.2. The receiving node (say, node *v*) decides to further process the TREE-CONSTRUCT message from a neighbor node (say, node *u*) if all the following conditions are met: (i) The *upstream node id* in the message is different from the id of the receiving node itself. (ii) The *tree level* value in the message is less than or equal to the *tree level* value maintained as part of the state information at the receiving node. (3) The value for the *sender's estimated bottleneck link weight* in the message is larger than the value for the receiver's estimated bottleneck link weight. (4) The weight of the directed edge from the sender node to the receiver node is larger than the latter's estimated bottleneck link weight for the path to the LEADER node. If all the above four conditions are met, the receiver node (node *v*) makes the following updates to its state information: (i) The receiver node updates its *estimated bottleneck link weight* for the path to the LEADER node to the minimum of the sender's (node *u*'s) estimated bottleneck link weight value in the TREE-CONSTRUCT message and the weight of the directed edge $u \rightarrow v$. (ii) The receiver node updates its *upstream node id* to that of the sender's node id. (iii) The value for the *tree level* is set to one more than the value for the tree level in the TREE-CONSTRUCT message. After making the above updates, the receiver node also rebroadcasts the TREE-CONSTRUCT message in its neighborhood by changing the values for the *upstream node id*, *sender's estimated bottleneck link weight* and *tree level* fields in the message to the most recently updated values for these fields in its state information.

Overall, a node receiving the TREE-CONSTRUCT message decides to further rebroadcast the message only if it can increase (through the sender node that sent it the message) its estimate for the bottleneck link weight path to the LEADER node (thus minimizing unnecessary retransmissions). Each node (other than the LEADER node) will be able to do so at least once because its initial value for the estimated bottleneck link weight is $-\infty$ and all edge weights are positive as well as the value for the sender's estimated bottleneck link weight in the TREE-CONSTRUCT message broadcast by the LEADER node is $+\infty$. At the end of the tree construction process, each node (other than the LEADER node) would have joined the DG tree through an upstream node that is on the maximum bottleneck link weight path to the LEADER node.

2.5 Propagation of the Tree-link-failure message

Whenever an upstream node fails to receive an aggregated data packet from one of its downstream child nodes, the upstream node decides that the link to the child node has broken and initiates a TREE-LINK-FAILURE message (included with a sequence number corresponding to the value sent by the LEADER node in the TREE-CONSTRUCT message) with the number of hops the message can get propagated equal to the tree level value for the initiating upstream node. The TREE-LINK-FAILURE message is essentially reverse broadcast higher up the currently used DG tree so that the LEADER node can receive the failure message and initiate the construction of a new DG tree. Nodes that lie downstream of the failed link get to learn about the tree failure when a TREE-CONSTRUCT message with a higher sequence number (larger than the current value for the sequence number known) is received.

3 Related work

In this section, we first discuss related work data gathering in mobile sensor networks and then focus our discussion specifically on related work on determining stable data gathering trees in mobile sensor networks.

3.1 Related work on data gathering in mobile sensor networks

To the best of our knowledge, other than the work presented in Section 3.2 and the related works discussed below, the existing works (e.g., [10, 14, 32, 33]) in the literature on mobile sensor networks take the following hybrid approach: The regular data sensing nodes are considered static and there exists one or more mobile data collecting nodes that move around the static sensor nodes; a data gathering topology involving the data collecting nodes is constructed and maintained, if needed. Since all the sensor nodes are not considered mobile (the type of mobile sensor networks considered in our research) and the data gathering topology constructed is not network-wide (i.e., spanning all the sensor nodes), we do not delve further on related works based on the above approach.

Among the very few network-wide spanning topology-based data gathering algorithms available in the literature for mobile sensor networks, most of the work focused on extending the classical LEACH (Low Energy Adaptive Clustering Hierarchy) [7] algorithm for static sensor networks to adapt to mobile environments. Variants of LEACH that have been proposed for MSNs focus on choosing the cluster heads by taking into account the residual energy available at the sensor nodes [2], mobility of the sensor nodes [29], stability of the links incident on a node [5] or proximity of the sensor nodes to certain landmarks [12]. Another work [30] related to cluster head selection proposed to set up a panel of cluster heads (some of which serve as backup) to facilitate cluster reconfiguration due to node mobility.

In [34], the authors proposed a cluster independent data collection tree (CIDT) protocol for mobile sensor networks that first partitions the entire network into clusters with a cluster head plus member nodes for each cluster and then chooses certain sensor nodes as data collection nodes (DCNs) that have better connection with the cluster heads. A data gathering tree of the DCNs is constructed and reconfigured over time when broken due to node mobility. The DCNs are selected in such a way that the links to the cluster heads and the links to the adjacent DCNs in the data gathering tree are stable. We opine that the CIDT protocol would incur a lot of control overhead (with respect to bandwidth and energy consumption) as two topologies (a cluster topology comprising of cluster heads plus their links to the member nodes and a tree topology of DCNs) have to be maintained in the network at any time. Though the two topologies have been formulated to be independent of each other, (due to node mobility) the identification of cluster heads and the DCNs has to be often initiated to maintain connectivity of the cluster heads to one or more near by DCNs.

In [15], the authors propose a directed acyclic graph (DAG)-based topology for determining data gathering trees in mobile sensor networks. Whenever a data gathering tree is required, the sink constructs a DAG of the underlying network and runs a maximum bottleneck node weight-based data gathering (MaxBNW-DG) algorithm on the DAG. In this pursuit, the sink initiates data collection from all the nodes in the network on one or more multi-hop paths; the paths traversed by the data in cycle-free manner constitute a DAG of the network. The weight of a sensor node is determined based on the theory of thermal fields applied on the utility of the data sensed by the node as well as that of its neighbors. The sink then initiates a distributed version of the MaxBNW-DG algorithm on the DAG such that each sensor node is located on a maximum bottleneck node weight path to the sink node. The bottleneck node weight for a path in [15] is calculated as the minimum of the weights of the intermediate node on the path; ties are broken in favor of paths of lower hop count. Due to node mobility, there may not be paths from one or more nodes to the sink node on the DAG. Similar to [34], we opine that a significant control overhead (in a mobile sensor network) would be encountered to first construct a DAG and then run a distributed version of the MaxBNW-DG algorithm on the DAG.

3.2 Related work on stable data gathering trees in mobile sensor networks

In [21], the authors had proposed a benchmarking algorithm to determine a sequence of stable data gathering trees that would exist for the longest time such that the number of tree transitions is the bare minimum. When a DG tree is required at a time instant *t*, the idea is to determine an intersection of the network graphs existing at time instants *t*, *t*+1, *t*+2, ... *t*+*k* such that the intersection graph is connected from time instants *t* ... *t*+*k* and not connected from time instants *t* ... *t*+*k*+1. That is, the inclusion of the graph at time instant *t*+*k*+1 to the intersection graph of time instants *t* ... *t*+*k* would disconnect the intersection graph from time instants *t* ... *t*+*k*+1. However, the algorithm is centralized in nature and would require the topology changes to be known a priori from the beginning to the end of the simulation session. On the other hand, the focus of research in this paper is to employ a distributed algorithm for determining stable data gathering trees using the BPI approach from complex network analysis - this approach does not require any a priori knowledge about the network topology changes as well as about the location and mobility of the nodes; we would just need the one-hop neighborhood information at every node.

In [19], the authors proposed distributed algorithms to determine the predicted link expiration time (LET)-based data gathering trees for longer tree lifetime and the minimum distance spanning tree (MST)-based data gathering trees for longer node lifetime (time of first node failure due to exhaustion of energy) and longer network lifetime (time at which the network gets disconnected due to the failure of one or more nodes). The predicted link expiration time (LET) of a link $i - j$ between two nodes *i* and *j*, currently at (X_i, Y_i) and (X_j, Y_j), and moving with velocities v_i and v_j in directions θ_i and θ_j (with respect to the positive X-axis) is computed using the formula proposed in [31]:

$$LET(i,j) = \frac{-(ab+cd)+\sqrt{(a^2+c^2)R^2-(ad-bc)^2}}{a^2+c^2} \quad (1)$$

where $a = v_i{*}\cos\theta_i - v_j{*}\cos\theta_j$; $b = X_i - X_j$;
$c = v_i{*}\sin\theta_i - v_j{*}\sin\theta_j$; $d = Y_i - Y_j$

The MST-based DG trees aim to minimize the largest Euclidean distance between the end nodes of a link in the DG tree, but are not as stable as the LET-DG trees [19]. Due to repeated tree reconfigurations, the gain obtained in the node lifetime (85-150% more than that of the LET-DG trees) does not equally get transferred to the gain obtained in network lifetime (only 15-130% more than that of the LET-DG trees) [19]. The LET-DG trees fit within the criteria of finding maximum bottleneck link weight-based DG trees (i.e., the objective is to maximize the minimum LET for a link on the path from any node to the LEADER node); whereas, the MST-DG trees fit within the criteria of finding minimum bottleneck link weight-based DG trees (i.e., the objective is to minimize the maximum value for the distance between the end nodes of the link on the path from any node to the LEADER node). In this paper, we model the short distance links as links with larger BPI/link stability score (measure of link weight; for further details, see Section 4) and run the maximum bottleneck link weight-based algorithm to maximize the minimum link weight on the path from any node to the LEADER node of the DG tree; we show that by doing so, we can simultaneously incur a larger tree lifetime as well as a lower energy consumption per round (see Section 5 for the simulation results).

In [24], the authors proposed a generic algorithm to determine maximum bottleneck node weight (MaxBNW)-based data gathering trees wherein the root node is the node with the largest weight. In [24], the weight of a node has been modeled as the sum of the weights of the links incident on it. The bottleneck node weight for a path from a node to the root node of the DG tree is the minimum of the weights of the nodes on the path. The MaxBNW-DG algorithm aims to determine a DG tree in which the path from any node to the root node is the path with the maximum bottleneck node weight. In [24], it has been observed that the MaxBNW-DG trees have different characteristics compared to the MaxBLW-DG trees. The focus of research in this paper is to determine MaxBLW-DG trees by modeling the link weight as a measure of the stability of the link using the algebraic connectivity approach from complex network analysis that does not need the location and mobility information of the nodes.

4 Bipartivity index (BPI)-based link selection strategy

In this section, we describe the Bipartivity Index (BPI) [6]-based link selection strategy adapted from complex network analysis to quantify the stability of links (i.e., the link weights) in a mobile sensor network. We compute the link stability score (LSS) for an edge by analyzing the bipartivity of the "egocentric network for the edge" that is adapted from the notion of egocentric network of a node [13]. The *egocentric network of a node* [13] in a graph is a sub graph comprising of: *vertices* - the nodes and its neighbors and *edges* - the links involving the node and/or its neighbors. We define the *egocentric network of an edge* in a graph to be a sub graph comprising of: *vertices* - the end nodes of the edge and their neighbors and *edges* - the links incident on the end nodes of the edge.

Our hypothesis for this research is based on the observation that links (we refer to as *short distance* links) whose end nodes are close enough to each other (vis-a-vis the transmission range per node) are more likely to be stable (and vice-versa) compared to links for which the distance between the end nodes is closer to the transmission range per node. We define the *fraction of link distance* (*fld*) for an edge as the ratio of the Euclidean distance between the end nodes of the edge and the transmission range per node. For a short distance link, *fld* is expected to be appreciably less than 1. Our hypothesis is that the end nodes of a short distance link are more likely to share a significant fraction of their neighbors (and vice-versa) and we could compute the BPI for the egocentric network of the link to quantify the extent of this shared neighborhood that can be in turn used as the link stability score (LSS). Note that the egocentric network of an edge could be independently (and identically) constructed by each of the two end nodes of the edge based on the one-hop neighborhood information received from the other node (as part of periodic beacon exchange).

A graph is said to be truly bipartite [4, 6] if we could partition the vertices of the graph into two disjoint sets such that all the edges in the graph are those that connect the vertices in one partition to vertices in the other partition and that there are no edges (called frustrated edges [6]) between vertices within the same partition. However, all network graphs cannot be expected to be truly bipartite. Hence, Estrada and Rodriguez-Velazquez [6] proposed the notion of bipartivity index (BPI) to measure the extent of bipartivity in a graph. The bipartivity index of a graph ranges from 0 to 1. If a graph is truly bipartite, then the bipartivity index is 1 and there are no frustrated edges between vertices within the same partition [6]. If a graph is not truly bipartite, then the bipartivity index will be less than 1. Estrada and Rodriguez-Velazquez [6] proposed a mechanism that will allow us to identify a partitioning of the vertices into two disjoint partitions as well as identify the frustrated edges (if the graph is not truly bipartite) involving vertices within the same partition. The mechanism proposed by Estrada Rodriguez-Velazquez [6] is to determine the eigenvalues of the adjacency matrix of the graph (to determine the bipartivity index, as shown in formulation 2) and use the signs (positive or negative) of the entries in the eigenvector corresponding to the smallest eigenvalue of the adjacency matrix to determine the partitioning of the vertices. If λ_1, λ_2, λ_3, ..., λ_n are the eigenvalues of the adjacency matrix of a graph G of n vertices, then the bipartivity index (BPI) of G is given by the formulation below [6]:

$$BPI(G) = \frac{\sum_{j=1}^{n} \cosh(\lambda_j)}{\sum_{j=1}^{n} \cosh(\lambda_j) + \sum_{j=1}^{n} \sinh(\lambda_j)} \tag{2}$$

To measure the extent of shared neighborhood of the end vertices of an edge in a graph, we propose to compute the bipartivity index on the egocentric network of the edge and use the complement of the bipartivity index (1 - BPI) as the link stability score (LSS) for the edge. That is: LSS(u-v) = 1 - BPI($EG_{u\text{-}v}$), where $EG_{u\text{-}v}$ is the egocentric network graph of the edge u-v. We justify the above proposal as follows (also illustrated in Figures 1-3: for an edge u-v where $u1$-$u4$ are four neighbors, other than vertex v, for a vertex u; and $v1$-$v4$ are four neighbors, other than vertex u, for a vertex v):

If the end vertices of an edge u-v do not have any shared neighbors (i.e., $u1$-$u4$ and $v1$-$v4$ are all distinct vertices: as shown in Figure 1), then we could partition the vertices in the egocentric network of the edge to two disjoint partitions such that vertex u and the neighbors of vertex v are in one partition (referred to as *partition-u*) and vertex v and the neighbors of vertex u are in the other partition (referred to as *partition-v*). The egocentric network of the edge u-v with no common neighbors for the end vertices would be a truly bipartite graph (as in

Figure 1) as the only edges in the graph would be edges connecting vertices in *partition-u* to vertices in *partition-v*. The bipartivity index of such an egocentric network graph would be 1.0 and as per our hypothesis, the link stability score for the edge would be 0.0.

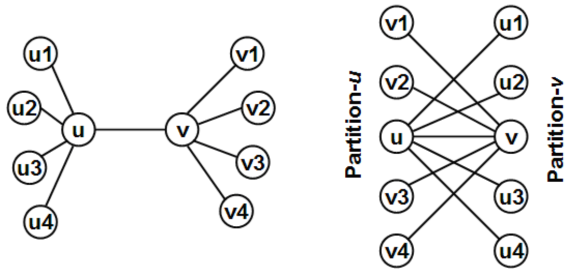
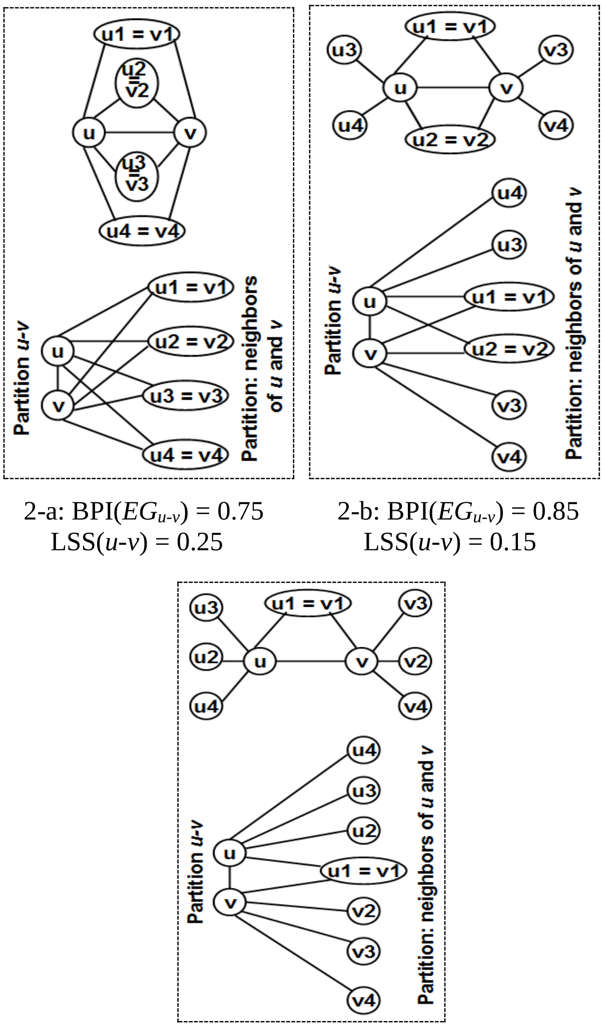


Figure 1: Example for a Truly Bipartite Egocentric Network of an Edge *u-v*.

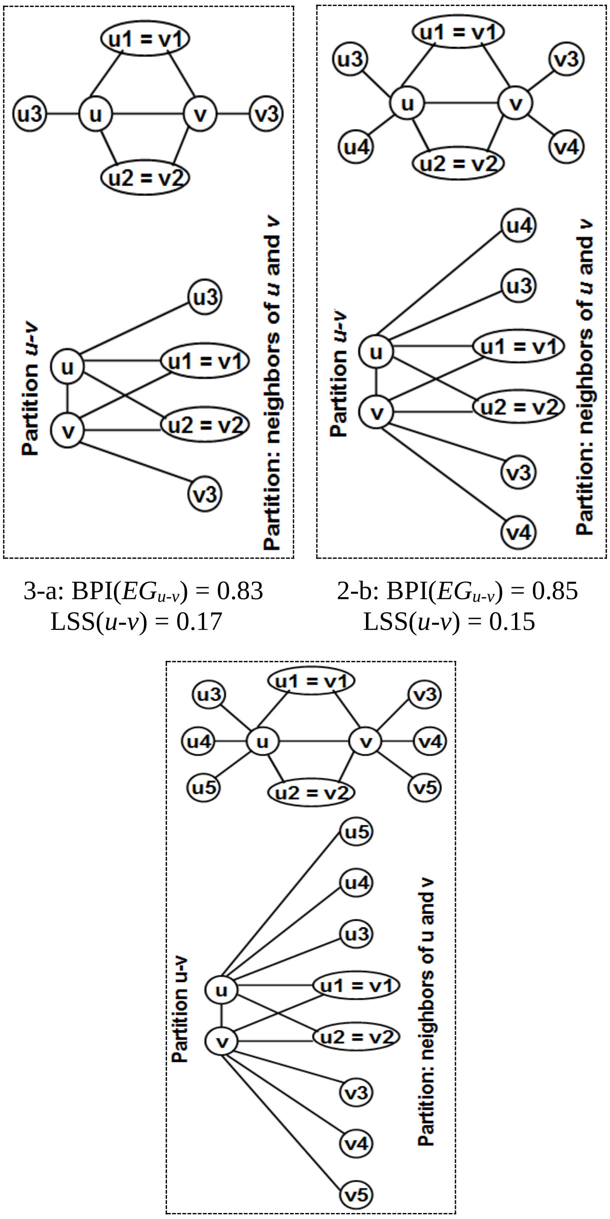


2-a: BPI($EG_{u\text{-}v}$) = 0.75
LSS(*u-v*) = 0.25

2-b: BPI($EG_{u\text{-}v}$) = 0.85
LSS(*u-v*) = 0.15

2-c: BPI($EG_{u\text{-}v}$) = 0.92
LSS(*u-v*) = 0.08

Figure 2: Bipartivity Index of the Egocentric Network Graph of an Edge and its Link Stability Score (Varying the Number of Shared Neighbors for a Fixed Number of Edges in the Egocentric Network).


u1 = v1
u2 = v2
u
v
u3
u4
u5
v3
v4
v5
Partition *u-v*
Partition: neighbors of *u* and *v*
Partition u-v
Partition: neighbors of u and v

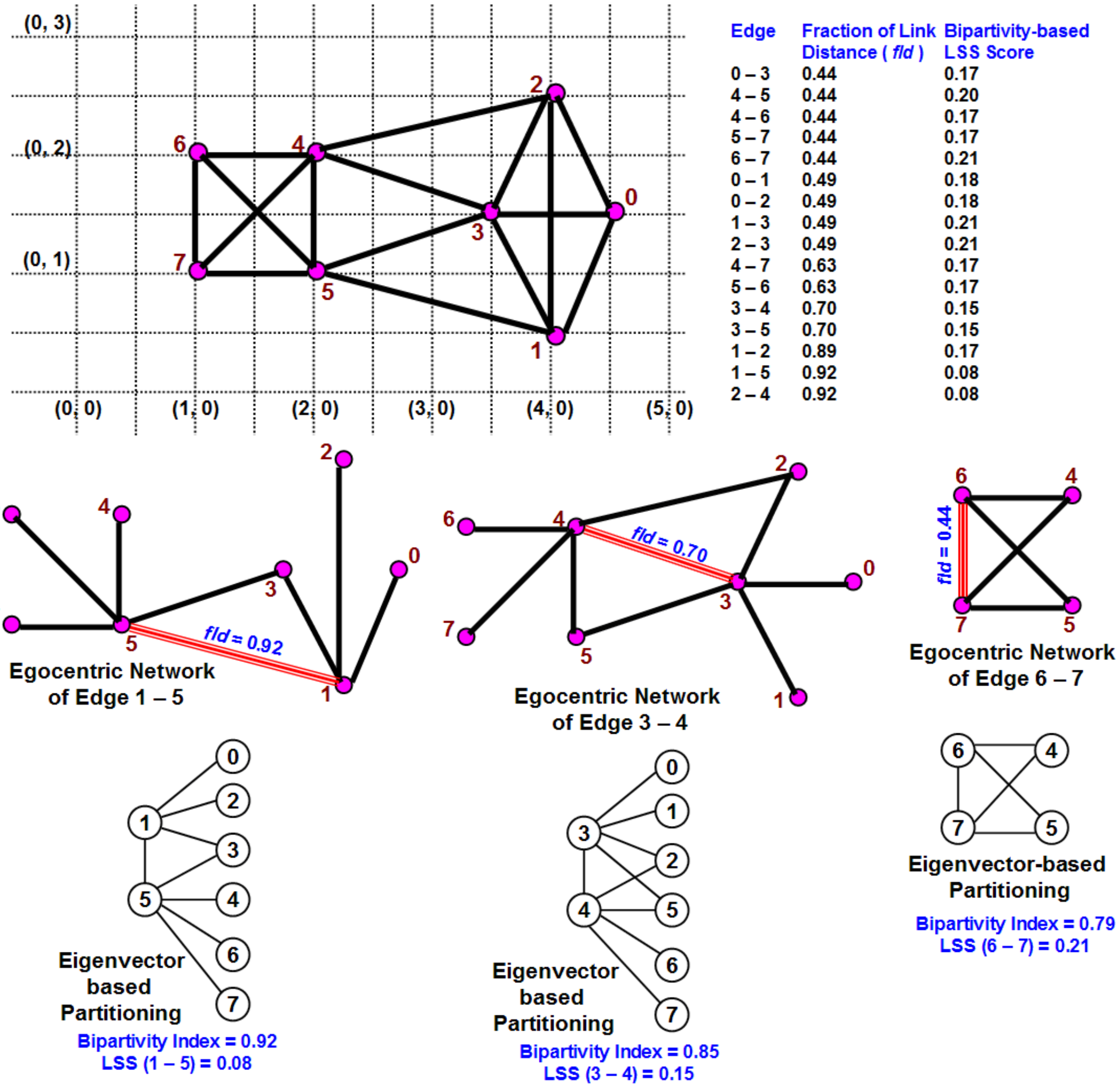
3-a: BPI($EG_{u\text{-}v}$) = 0.83
LSS(*u-v*) = 0.17

2-b: BPI($EG_{u\text{-}v}$) = 0.85
LSS(*u-v*) = 0.15

3-c: BPI($EG_{u\text{-}v}$) = 0.87
LSS(*u-v*) = 0.13

Figure 3: Bipartivity Index of the Egocentric Network Graph of an Edge and its Link Stability Score (Varying the Number of Edges in the Egocentric Network for a Fixed Number of Shared Neighbors).

On the other hand, if the end vertices of an edge *u-v* share one or more of their neighbors: then, the eigenvector-based decomposition for bipartivity [6] applied on the egocentric network of the edge would group the two end vertices *u* and *v* together in one partition (referred to as *partition u-v*) and the neighbors of *u* and *v* together in the other partition (referred to as *partition: neighbors of u and v*). The BPI of such egocentric network graphs would be less than 1 (due to the presence of the frustrated edge *u-v* in the same partition) and the actual magnitude of the BPI would depend on the actual number of neighbors for the two end vertices (i.e., on the number of vertices in the other



Edge	Fraction of Link Distance (*fld*)	Bipartivity-based LSS Score
0 – 3	0.44	0.17
4 – 5	0.44	0.20
4 – 6	0.44	0.17
5 – 7	0.44	0.17
6 – 7	0.44	0.21
0 – 1	0.49	0.18
0 – 2	0.49	0.18
1 – 3	0.49	0.21
2 – 3	0.49	0.21
4 – 7	0.63	0.17
5 – 6	0.63	0.17
3 – 4	0.70	0.15
3 – 5	0.70	0.15
1 – 2	0.89	0.17
1 – 5	0.92	0.08
2 – 4	0.92	0.08

Figure 4: Illustration of the Eigenvector-based Partitioning of the Egocentric Networks of the Edges and the Bipartivity Index and Link Stability Scores of the Edges in an Example Graph.

partition: neighbors of u and v) as well as on the number of shared neighbors (i.e., on the number of edges connecting the vertices in *partition u-v* to the vertices in the *partition: neighbors of u and v*). For a given egocentric network graph EG_{u-v} with a certain number of edges and is not truly bipartite (as shown in Figures 2-a, 2-b and 2-c): the larger the number of shared neighbors (i.e., fewer the number of vertices in the *partition: neighbors of u and v*), the lower the BPI (and larger will be the LSS score for the edge *u-v*). Likewise, for a given egocentric network graph EG_{u-v} with a certain number of shared neighbors and is not truly bipartite (as shown in Figures 3-a, 3-b and 3-c): the larger the number of vertices in the *partition: neighbors of u and v* (i.e., the larger the number of edges connecting the vertices in *partition u-v* to the vertices in the *partition: neighbors of u and v*), the larger the BPI (and lower will be the LSS score for the edge *u-v*).

In Figure 4, we illustrate the computation of the bipartivity index of the egocentric networks of the edges (with coordinates of the vertices as indicated in a grid) in an example graph. The egocentric networks for none of the edges have been observed to be truly bipartite. We illustrate the eigenvector-based partitioning of the egocentric networks of three edges: 6-7, 3-4 and 1-5 that have different values for the fraction of link distance (*fld*). We observe the bipartivity-based LSS values (1 - bipartivity index) for these three edges to increase with decrease in the fraction of link distance (*fld*) values. Overall, we see the expected trend between *fld* and the bipartivity-based LSS values for the edges: the LSS values are more likely to be higher for edges with lower *fld* values and vice-versa.

5 Simulations

In this section, we first present the simulation environment and the notion of normalized comprehensive relative performance (NCRP) score to identify the link selection strategy that effectively balances the tradeoffs with respect to the performance metrics, and then discuss in detail the results of the simulations obtained by running the distributed version of the MaxBLW-DG algorithm incorporated with BPI as well as the LET-based link selection strategies. The simulations were conducted in a discrete-event simulator implemented in Java for mobile sensor networks. The simulator was earlier successfully used for other related studies (e.g., [19][21][23]) for mobile sensor networks. The medium access control (MAC) layer is assumed to be ideal to extract the best possible performance from the data gathering algorithm and the link selection strategies.

5.1 Simulation environment

In this sub section, we present the simulation parameters (network density, maximum node velocity, data size and the number of rounds as well as the frequency of LSS updates), the mobility model, the energy consumption model, DG tree update policy and channel access policies as well as define the structural metrics and performance metrics.

Simulation Parameters: The network dimensions is 100m x 100m (Area A = 10,000 m2) and the sink is assumed to be outside the network: at (50, 300). The number of nodes (N) in the network is set to be 50 and 100, and the transmission range (R) per node values used are 25m and 35m. The average number of neighbors per node is computed using the formula: πR2N/A. Accordingly, we have the following scenarios of network density: low density (N = 50, R = 25m, Avg. # neighbors per node = 9.8), low-moderate density (N = 50, R = 35m, Avg. # neighbors per node = 19.2), moderate-high density (N = 100, R = 25m, Avg. # neighbors per node = 19.6) and high density (N = 100, R = 35m, Avg. # neighbors per node = 38.5). The maximum velocity of a node (vmax) is set to be: 1 m/s (low mobility), 3 m/s (low-moderate mobility), 5 m/s (moderate-high mobility) and 10 m/s (high mobility). Thus, we have a total of sixteen scenarios of various combinations of network density and node mobility. We generated 100 instances of node mobility profiles for each of the above sixteen scenarios of network density and node mobility and averaged the results (with respect to the performance metrics and structural metrics discussed below) obtained for the MaxBLW-DG algorithm incorporated with the BPI and LET-based link selection strategies run on these 100 instances.

Mobility Model: To start with, the nodes are uniform-randomly distributed throughout the network. Mobility of the nodes is modeled according to the Random Waypoint model [3] with the nodes moving continuously (zero pause time) and independent of each other. A node decides to move from its current location to a randomly chosen location within the network with a velocity uniform-randomly chosen from [0...vmax]; after reaching the chosen location, the node continues its movement by randomly choosing another location with a different randomly chosen velocity from the above range. A node continues its movement like this throughout the simulation. We record the instances of direction change and the corresponding location and velocity to construct (offline) a mobility profile for each node and feed in this mobility profile to the MaxBLW-DG tree algorithm.

Energy Consumption Model: Nodes are assumed to be of sufficient energy so that there are no node failures due to exhaustion of energy. The energy consumed at a node for data aggregation is the sum of the energy lost in receiving the aggregated data from each of its child nodes, fusing its own data with that of the aggregated data and transmitting the final aggregated data to its upstream node in the DG tree. The energy consumed at a node for broadcast tree discovery is the sum of the energy lost to receive the broadcast control message from each of its neighbors and to the transmit the control message in its neighborhood, if the conditions for rebroadcast are met. The energy consumption model used is a first-order radio model [28] that has been used in several of the previous work [7, 11] in the literature. According to this model: (i) the energy consumed at a sensor node to transmit a k-bit message over a distance d is given by: ETX(k, d) = Eelec*k + $\in_{amp}$ *k*d2, where Eelec = 50 nJ/bit is the energy lost to run the radio transmitter or receiver circuitry and $\in_{amp}$ = 100 pJ/bit/m2 is the energy lost to run the transmitter amplifier; (ii) the energy lost at a sensor node to broadcast a k-bit message to all its neighbors within the transmission range R is simply given by ETX(k, R); the energy consumed at a sensor node to receive a k-bit message is ERX(k) = Eelec *k. The total energy consumed at a sensor node to receive k-bit broadcast messages transmitted by all of its n-neighbors is simply given by n * ERX(k). We do not take into consideration the energy lost due to periodic beacon exchange as both the LET and BPI-based link selection strategies considered in this research use it to determine the link weights.

Data Size and Frequency of LSS Updates: We conduct the simulations for 2000 rounds (one round for every 0.25 seconds: a total of 500 seconds). The LSS scores of the links are estimated in the neighborhood of the nodes for every second. For each round: data gets aggregated across the network, starting from the leaf nodes and proceeding all the way to the LEADER node of the DG tree; the LEADER node forwards the final aggregated data to the sink. The data size is assumed to remain the same during network-wide aggregation. That is, the size of the aggregated data is assumed to be the same as the size of the data collected at the individual sensor nodes. The data size is 2000 bits and the size of the control messages used for tree configuration and maintenance is assumed to be 400 bits (sufficiently large enough to accommodate the various fields in the control messages).

DG Tree Update Policy: Every time a DG tree is needed, the sink collects the weights of the links of the sensor nodes using a network-wide broadcast. The node with the largest sum of the link weights is considered as the root node (a.k.a. LEADER node) and the sink node sends a control message to the LEADER node to initiate tree discovery (a process also called tree reconfiguration). A DG tree is used as long as it exists: this is referred to as the Least Overhead Routing Approach (LORA) [1] in the literature of mobile ad hoc networks.

Channel Access Policy: Note that in a particular timeslot, an intermediate node could collect data from only one of its child nodes (using Time Division Multiple Access, TDMA [35]) if the latter has its aggregated data available, and an intermediate node could transmit upstream its aggregated data only after receiving the same from each of its child nodes and aggregating with its own. An intermediate node could collect data from one of its child nodes at the same time (using Code Division Multiple Access, CDMA [35]) as any other intermediate node collects data from any of its child nodes. We assume that sufficient number of CDMA and TDMA codes are available at the sensor nodes (as needed) to facilitate data aggregation in the minimum number of time slots.

Structural Metrics: We evaluated the following three structural metrics: (S-i) Tree Height, TH: The tree height is the maximum of the level numbers of the vertices (i.e., the number of hops) from the root node of the DG tree (with the root node considered to be at level 0). (S-ii) Fraction of Leaf Nodes, FLN: The fraction of leaf nodes is the ratio of the number of leaf nodes to the total number of nodes in the network graph. (S-iii) Average Number of Child Nodes per Intermediate Node, CNI: The average number of child nodes per intermediate node is the weighted average of the number of child nodes per intermediate node considered across all intermediate nodes.

Performance Metrics: We evaluated the following three performance metrics: (P-i) Tree Lifetime, TL: The tree lifetime is the number of rounds a DG tree exists before one or more of its links fail due to node mobility, averaged over the duration of a simulation session. (P-ii) Aggregation Delay per Round, ADR: The aggregation delay per round is the minimum number of timeslots (computed as per algorithm [25]) it takes for data to get aggregated along the edges of the DG tree and reach the root node, averaged across all the rounds. (P-iii) Energy Consumption per Round, ECR: The energy consumed per round is the sum of the energy consumed at each of the nodes for data aggregation in the network plus the energy lost due to broadcast tree discoveries if the DG tree was reconfigured at the beginning of the round. We average the energy consumed across all the rounds of a simulation session.

5.2 Normalized comprehensive relative performance (NCRP) score

As described in Section 5.4, we observe a complex tradeoff between the three performance metrics: tree lifetime, energy consumption per round and aggregation delay per round. Since the performance metrics incur different levels of magnitude, we propose to bring the values incurred for these metrics on a common scale of 0 to 1 using the method of normalization and propose to prefer the link selection strategy that incurs the largest value for the normalized score (or the complement of the normalized score, as appropriate) with respect to the individual metrics and/or with respect to the normalized comprehensive relative performance (NCRP) score (introduced below). In other words, the idea is to normalize the values incurred for each of the performance metrics incurred for the BPI and LET link selection strategies for a particular simulation scenario and compute a normalized comprehensive relative performance (NCRP) score with respect to the performance metrics (as shown below).

As we seek for a larger tree lifetime (see Section 5.4), lower energy consumption per round (see Section 5.5) and lower aggregation delay per round (see Section 5.6), we use the normalized values for the tree lifetime (*TL*), but complement of the normalized values for the energy consumption per round (*ECR*) and aggregation delay per round (*ADR*) to compute the NCRP score as a weighted average (weight = 1/3 for each metric) of these three values.

Complement of Norm. *ECR* = 1 - Normalized *ECR*

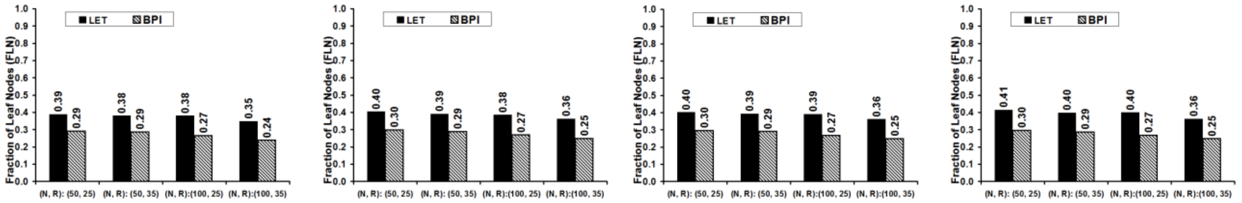
Complement of Norm. *ADR* = 1 - Normalized *ADR*

NCRP = {Normalized *TL* + Complement of Norm. *ECR* + Complement of Norm. *ADR*}/3 ....(3)

5.3 Structural metrics

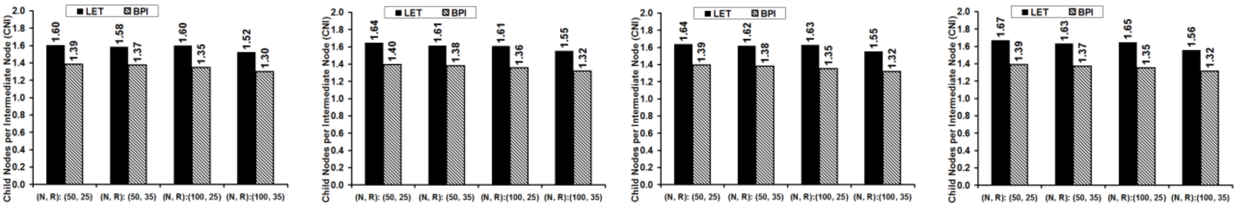
In this section, we illustrate the results obtained with respect to the structural metrics for the DG trees determined based on the BPI and LET strategies. For lower energy consumption and lower aggregation delay per round, we would desire to have DG trees with a lower number of child nodes per intermediate node (so that an intermediate node can spend less energy in receiving data from each of its child nodes as well as aggregate data from its child nodes in fewer time slots) and at the same time a larger fraction of leaf nodes (so that the energy lost due to receptions could be lower and the number of nodes that readily have the data to transmit could be larger). However, we observe that it would not be possible to simultaneously maximize the fraction of leaf nodes as well as minimize the number of child nodes per intermediate node. As the fraction of leaf nodes in a DG tree increases, the fraction of intermediate nodes in the DG tree is bound to decrease and hence the number of child nodes per intermediate node is bound to only increase. We also observe a similar trend in the results (see Figures 5-6) for the structural metrics obtained for the DG trees based the LET and BPI strategies.

The LET-based DG trees incur a larger fraction of leaf nodes (desirable for lower energy consumption and lower aggregation delay per round) and lower height (desirable for lower aggregation delay per round), but also simultaneously incur a larger number of child nodes per intermediate node. The BPI-based DG trees incur a relatively lower number of child nodes per intermediate node (desirable for lower energy consumption and lower aggregation delay per round), but also incur a lower fraction of leaf nodes. Thus, as envisioned previously, we observe a tradeoff between the fraction of leaf nodes and the number of child nodes per intermediate node.



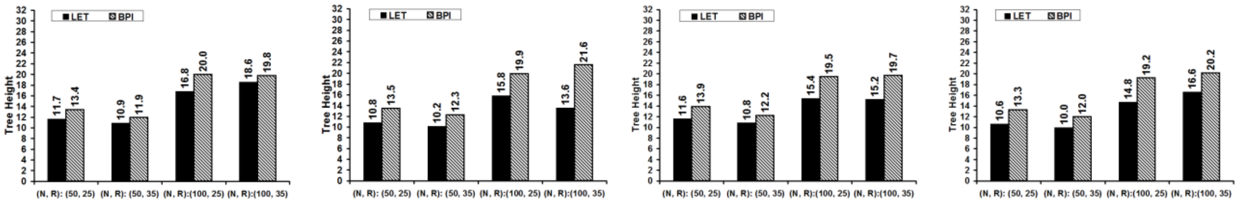
(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s (c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 5: Average Fraction of Leaf Nodes.



(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s (c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 6: Average Number of Child Nodes per Intermediate Node.



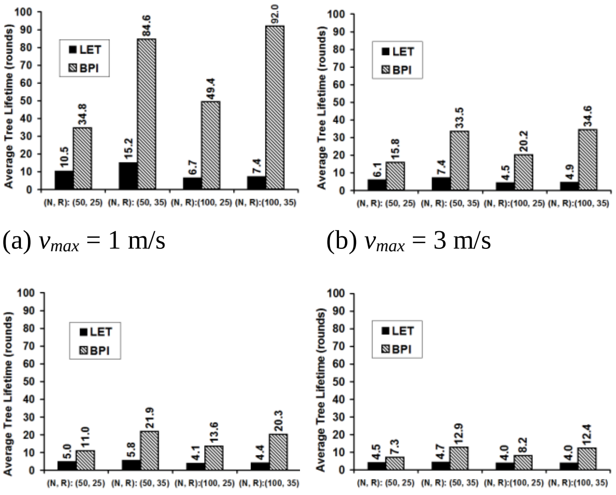
(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s (c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 7: Average Tree Height.

Since the structural metrics are not dependent on node mobility, for a given network density: we observe the values incurred for the structural metrics to be independent of node mobility. For a given level of node mobility, we observe the fraction of leaf nodes as well as the average number of child nodes per intermediate node (incurred for both the LET and BPI-based DG trees) to decrease with increase in network density. On the other hand, for a given level of node mobility, we observe the tree height to increase with increase in network density (especially, as we increase from 50 to 100 nodes).

5.4 Tree lifetime

The BPI-DG trees incur significantly larger values for the tree lifetime (see Figure 8) compared to that of the LET-DG trees. The lifetime of the BPI-DG trees could be as large as 12 times the lifetime of the LET-DG trees (especially in scenarios of high network density and low node mobility). Even in the worst case (scenarios of low network density and high node mobility), the lifetime of the BPI-DG trees is at least 60% larger than the lifetime of the LET-DG trees. When considered across all the 16 scenarios of network density and node mobility (refer Figure 9), the normalized values (with respect to tree lifetime) for the BPI-DG trees is at least 0.85; whereas, the normalized values for the LET-DG trees is at most 0.52.

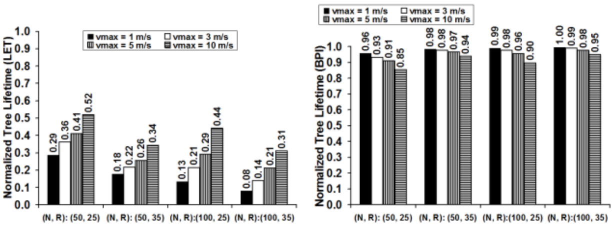


(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s

(c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 8: Absolute Value of Average Tree Lifetime.

From Figure 8, we could observe that for a fixed level of node mobility: the average lifetimes for the BPI-DG trees relatively increase with increase in network density, whereas the average lifetimes for the LET-DG trees relatively decrease with increase in network density. From Figure 9, for a given level of network density: we could observe that the relative performance of the LET-DG trees with respect to tree lifetime improves with increase in node mobility. On the other hand, the relative performance of the BPI-DG trees with respect to tree lifetime remains almost the same or only marginally degrades with increase in node mobility. Thus, with respect to tree lifetime, the BPI-DG trees are relatively more scalable (i.e., are robust to increase in network density for a given level of node mobility) and remains relatively about the same (with increase in node mobility for a given network density). Such observations on the relative performance of the link selection strategies cannot be easily assessed by simply looking at the actual values incurred for tree lifetime in Figure 8 (or for that matter any other performance metric).



(a) LET (b) BPI

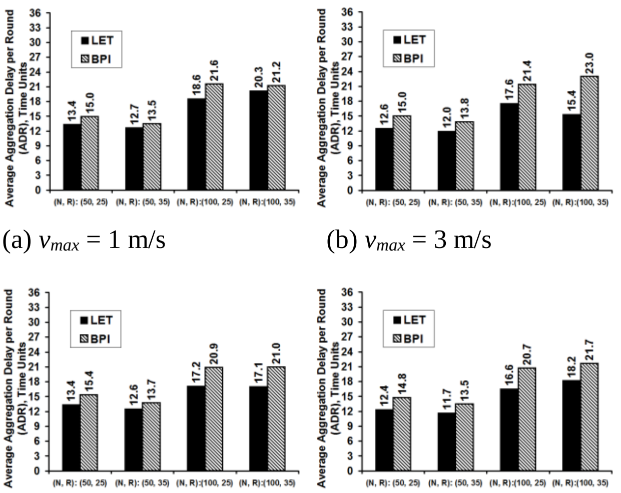
Figure 9: Normalized Value of Average Tree Lifetime.

5.5 Aggregation delay per round

From Figures 10-11, we observe the LET-DG trees to incur lower *ADR* values for all conditions of network density and node mobility. For a given level of node mobility, the difference in the magnitude of the *ADR* values between the BPI-DG trees and the LET-DG trees increases with increase in network density. For a given network density, the *ADR* values incurred for the DG trees based on a particular link selection strategy remain about the same (there is no particular or a significant trend of variation) at different levels of node mobility.

As we prefer a link selection strategy to yield lower aggregation delay per round for the DG trees, we plot the complement of the normalized *ADR* values (instead of just the normalized *ADR* values) of Figure 10 in Figure 11. The *ADR* values (shown in Figure 10) incurred for both the LET-DG and BPI-DG trees appear to be directly proportional and positively correlated with the height of the DG trees (shown in Figure 7). Though the absolute *ADR* values (Figure 10) are observed to increase with increase in network density for a given level of node mobility, there is no change in the trend of the normalized *ADR* values (a measure of the relative performance) incurred for the DG trees based on both LET and BPI (for a given level of node mobility, the complement of the normalized *ADR* values for either LET or BPI almost remains the same with increase in network density).

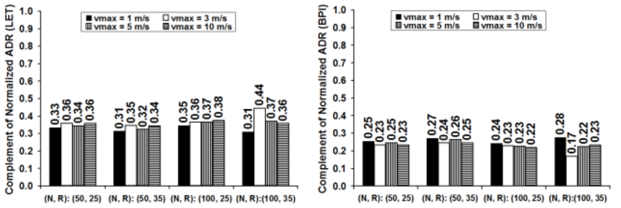
Unlike the exceptionally high values for the tree lifetime incurred with the BPI-DG trees and relatively very poor tree lifetime (see Figures 8-9) observed for the LET-DG trees, (the complement of) the normalized *ADR* values incurred for the DG trees determined based on LET and BPI are not far different. In the case of tree lifetime, the difference in the normalized values for the lifetime of the LET and BPI- based DG trees is at least 0.33 and is as large as 0.92 (with a high median of 0.69). On the other hand, in the case of aggregation delay per round, the difference in the complement of the normalized ADR values of the LET and BPI-based DG trees is at most 0.27 (with a low median of 0.11).



(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s

(c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 10: Absolute Value of Average Aggregation Delay per Round (in time units).

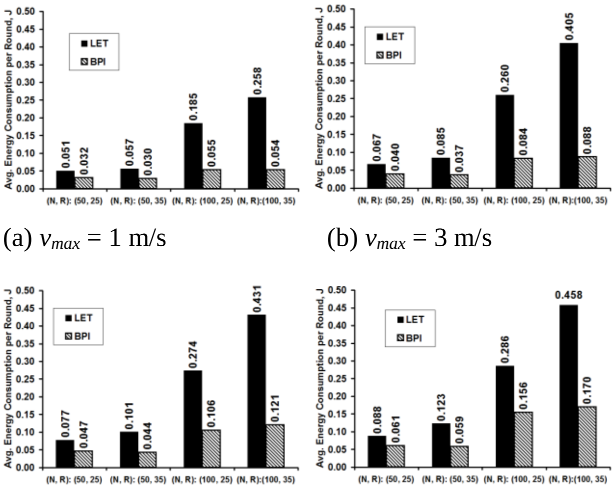


(a) LET (b) BPI

Figure 11: Complement of the Normalized Value of Average Aggregation Delay per Round.

5.6 Energy consumption per round

The BPI-DG trees incur lower values for energy consumption per round (*ECR*) for all scenarios of network density and node mobility (see Figures 12-13), and the LET-DG trees incur larger *ECR* values for all scenarios. The relatively better performance of the BPI-based DG trees with respect to *ECR* could be primarily attributed to the less frequent network-wide broadcasts (due to a larger tree lifetime) and the short distance nature of the links that are part of the transmissions during data aggregation. For a given level of node mobility: the difference in the *ECR* values between the BPI and LET-based DG trees increases with increase in network density (attributed to the relatively unstable LET-based DG trees with increase in network density). Of course, for a fixed network density, the difference in the *ECR* values increase with increase in node mobility. Though both LET and BPI incur an increase in the magnitude for the *ECR* values with increase in network density and/or node mobility, the *ECR* values for the LET-DG trees are significantly larger than those incurred for the BPI-DG trees.

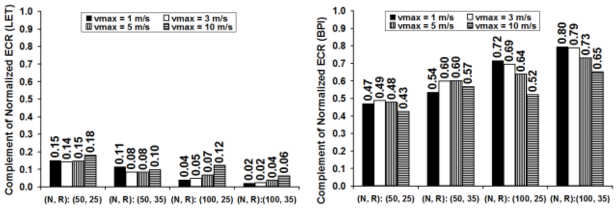


(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s

(c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 12: Absolute Value of Average Energy Consumption per Round (in Joules).

For a given level of node mobility (see Figure 13): the complement of the normalized *ECR* values for the BPI-DG trees increases with increase in network density; whereas, the complement of the normalized *ECR* values for the LET-DG trees decreases with increase in network density. Thus, for a given level of node mobility: the relative performance (with respect to *ECR*) of the BPI-DG trees vis-a-vis the LET-DG trees improves with increase in network density. On the other hand, (see Figure 13), for a given network density: the complement of the normalized *ECR* values for the LET-DG trees slightly increase with increase in node mobility; whereas, the complement of the normalized *ECR* values for the BPI-DG trees slightly decrease with increase in node mobility, more visibly in networks of high density.

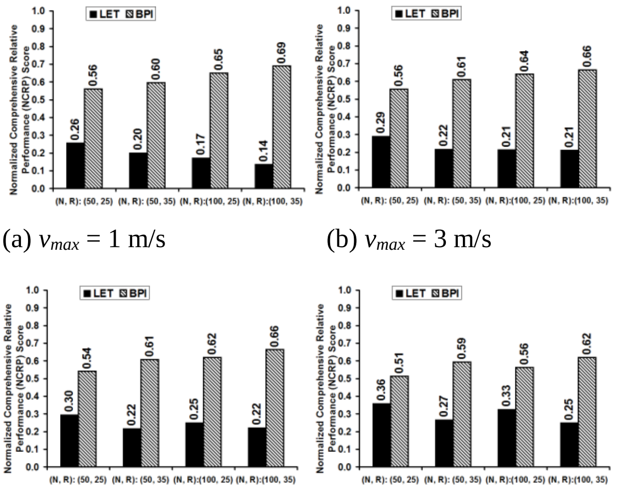


(a) LET (b) BPI

Figure 13: Complement of the Normalized Value of Average Energy Consumption per Round.

5.7 Analysis of the relative performance tradeoff based on the NCRP scores

We observe the lifetime incurred for the BPI-DG trees to be significantly larger than that of the LET-DG trees. Likewise, the average energy consumption per round incurred for the BPI-DG trees is lower than that incurred for the LET-DG trees. On the other hand, the aggregation delay per round incurred for the LET-DG trees is lower than the aggregation delay per round values incurred for the BPI-DG trees. This illustrates a complex {tree lifetime, energy consumption per round} vs. {aggregation delay per round} tradeoff. We use the normalization approach (introduced in Section 5.2) to analyze this tradeoff with respect to the above three performance metrics between LET and BPI. We observe (from Figure 14) the BPI-based link selection strategy to effectively balance this tradeoff and incur larger values for the NCRP score under all the 16 different scenarios of network density and node mobility. A closer look at the actual values (from Figures 8, 10 and 12) and normalized values (from Figures 9, 11 and 13) for the three individual performance metrics illustrates the same.



(a) v_{max} = 1 m/s (b) v_{max} = 3 m/s

(c) v_{max} = 5 m/s (d) v_{max} = 10 m/s

Figure 14: Normalized Comprehensive Relative Performance Score for the Link Selection Strategies.

With respect to the impact of network density and node mobility: the NCRP scores for the LET-DG trees decrease with increase in network density (for a fixed level of node mobility) and increase with increase in node mobility (for a fixed network density). On the other hand, the NCRP scores for the BPI-DG trees increase with increase in network density (for a fixed level of node mobility) and remain about the same with increase in node mobility (for a fixed network density). Thus, we observe the overall relative performance of the BPI-DG trees to only improve (or remain the same) with increase in network density and/or node mobility.

Table 1 provides a comprehensive overview of the simulation results, identifying the link selection strategy that yields the most desirable values for the structural metrics and performance metrics. With respect to the structural metrics, we desire to have larger values for the fraction of leaf nodes and lower values for the number of child nodes per intermediate node and tree height. With respect to the performance metrics, we desire to have larger values for tree lifetime and lower values for the energy consumption per round and aggregation delay per round. With respect to the NCRP score (see equation 3 for the formulation), we desire to have values closer to 1. On these lines, the LET strategy returns the most desirable values for two of the three structural metrics as

Table 1: Link Selection Strategies (LET vs. BPI) that Incur the most Desirable Values for the Structural Metrics and Performance Metrics as well as the NCRP Score.

# nodes	Tr. range	v_{max}	*FLN*	*CNI*	*TH*	*TL*	*ECR*	*ADR*	*NCRP*
50	25m	1 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		3 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		5 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		10 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
50	35m	1 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		3 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		5 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		10 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
100	25m	1 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		3 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		5 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		10 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
100	35m	1 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		3 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		5 m/s	LET	BPI	LET	BPI	BPI	LET	BPI
		10 m/s	LET	BPI	LET	BPI	BPI	LET	BPI

well as the aggregation delay per round (all of which are not dependent on node mobility); however, the BPI strategy is useful to discover stable DG trees (larger tree lifetime) that also incur a lower energy consumption per round (attributed to the less frequent network-wide broadcasts and short distance nature of the links). The relatively better performance of the BPI-DG trees with respect to the tree lifetime and energy consumption per round and competitive values for the aggregation delay per round lift the normalized comprehensive relative performance (NCRP) scores to be above that of the LET strategy. The minimum and maximum difference in the NCRP scores incurred for the BPI-DG trees vis-a-vis the LET-DG trees are respectively 0.15 (observed in scenario of low network density and high node mobility) and 0.55 (observed in scenario of high network density and low node mobility). The median difference in the NCRP scores is 0.38.

6 Conclusions

The high-level contribution of this paper is a proposal to use the Bipartivity Index (BPI) metric (a spectral graph-theoretic metric used in complex network analysis) to determine stable data gathering (DG) trees for mobile sensor networks (MSNs). Our hypothesis in this research is that the end nodes of short distance links (the Euclidean distance between the end nodes of the link is far less than the transmission range of the nodes) are more likely to share a significant fraction of their neighborhood (and vice-versa). As short distance links are more likely to be stable too (and vice-versa), we propose to use the BPI strategy to evaluate and quantify the extent of shared neighborhood of the end vertices of the edges for determining stable DG trees in mobile sensor networks.

We model the neighborhood of the end vertices of an edge *u-v* as an *egocentric network* $EG_{u\text{-}v}$ comprising of the end vertices and their neighbors as *nodes* and the edges incident on the end vertices as *links*. We have shown (through detailed theoretical analysis and illustrative examples) that edges whose egocentric networks have smaller values for the BPI are more likely to be short distance links. The egocentric network for an edge and its BPI score could be independently determined by the two end vertices of the edge based on just the one-hop neighborhood information and without knowledge about the location and mobility of the nodes. We quantify the link stability score (LSS) for an edge *u-v* as 1 - BPI($EG_{u\text{-}v}$). We define the bottleneck link weight of a path as the minimum of the weights of the constituent links on the path. Whenever a DG tree is required, we determine the maximum bottleneck link weight-based DG tree for which the bottleneck link weight of the path from any node to the root node is the maximum (the root node is the node with the largest sum of the LSS scores of its incident links).

We have compared the performance of the BPI-based DG trees with that of the DG trees determined based on the predicted link expiration time (LET) - the only well-known strategy so far [19] to determine stable DG trees for MSNs. We observe the BPI-DG trees to be significantly more stable as well as incur a lower energy consumption per round compared to that of the LET-DG trees. On the other hand, we observe the aggregation delay per round incurred for the LET-DG trees to be lower than the aggregation delay per round incurred for the BPI-DG trees. We thus observe a complex {tree lifetime, energy consumption per round} vs. {aggregation delay per round} tradeoff. We attribute this tradeoff to the unstable nature of the LET-DG trees (leading to more energy-intensive network-wide broadcast tree discoveries) and lower height as well as a larger fraction of leaf nodes (contributing to a lower aggregation delay per round).

Finally, we propose the use of a normalization-based approach to evaluate the relative performance of the link selection strategies in a scale of 0...1 and thereby overcome the difficulty arising in analyzing the tradeoffs among the performance metrics whose values fall under different levels of magnitude (as is the case for the three performance metrics studied in this paper). We illustrate the use of the normalization-based approach to identify the link selection strategy that best balances the performance tradeoff as well as whose relative performance is more scalable (with increase in network density and/or node mobility). We observe the BPI-based link selection strategy to yield DG trees that incur the largest values for the normalized comprehensive relative performance (NCRP) scores under all the 16 scenarios of network density and node mobility. To vindicate the larger NCRP scores, we observe the BPI-based DG trees to simultaneously incur larger values for tree lifetime and lower values for energy consumption per round and not so relatively high values for aggregation delay per round. We observe the comprehensive relative performance of the BPI-DG trees to be more scalable with increase in network density and not much affected with increase in node mobility.

7 Acknowledgment

The work leading to this paper is funded through the Minority Leaders Program, “A Stable Trustworthy Neighborhood Scheme for Secure Mobile Sensor Networks,” funded by the Clarkson Aerospace/US Air Force Research Lab. Subcontract Number: JACK 15-S7700-02-C2. The AFRL public clearance approval number for this article is 88ABW-2017-0475. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the funding agency.

8 References

[1] M. Abolhasan, T. Wysocki, E. Dutkiewicz, "A Review of Routing Protocols for Mobile Ad hoc Networks," *Ad hoc Networks*, vol. 2, no. 1, pp. 1-22, 2004.

[2] T. Banerjee, B. Xie, J. H. Jun and D. P. Agarwal, "LIMOC: Enhancing the Lifetime of a Sensor Network with Mobile Clusterheads," *Proceedings of the Vehicular Technology Conference Fall*, pp. 133-137, Baltimore, MD, USA, September 30 - October 3, 2007.

[3] C. Bettstetter, H. Hartenstein and X. Perez-Costa, “Stochastic Properties of the Random-Way Point Mobility Model,” *Wireless Networks*, vol. 10, no. 5, pp. 555-567, September 2004.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, 3rd Edition, MIT Press, July 2009.

[5] S. Deng, J. Li and L. Shen, "Mobility-based Clustering Protocol for Wireless Sensor Networks with Mobile Nodes," *IET Wireless Sensor Systems*, vol. 1, no. 1, pp. 39-47, March 2011.

[6] E. Estrada and J. A. Rodriguez-Velazquez, "Spectral Measures of Bipartivity in Complex Networks," *Physical Review E* 72, 046105, pp. 1-6, 2005.

[7] W. Heinzelman, A. Chandrakasan and H. Balakarishnan, “Energy-Efficient Communication Protocols for Wireless Microsensor Networks,” *Proceedings of the Hawaaian International Conference on Systems Science*, Maui, HI, USA, January 2000.

[8] B. Hofmann-Wellenhof, H. Lichtenegger and J. Collins, *Global Positioning System*: *Theory and Practice*, 5th Edition, Springer, October 2013.

[9] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan and S. Madden, “CarTel: A Distributed Mobile Sensor Computing System,” *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pp. 125-138, Boulder, CO, USA, November 2006.

[10] Y. Lai, J. Xie, Z. Lin, T. Wang and M. Liao, "Adaptive Data Gathering in Mobile Sensor Networks using Speedy Mobile Elements," *Sensors*, vol. 15, no. 9, pp. 23218-23248, 2015.

[11] S. Lindsey, C. Raghavendra and K. M. Sivalingam, "Data Gathering Algorithms in Sensor Networks using Energy Metrics," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 9, pp. 924-935, September 2002.

[12] C-M. Liu, C-H. Lee and L-C. Wang, "Distributed Clustering Algorithms for Data Gathering in Wireless Mobile Sensor Networks," *Journal of Parallel and Distributed Computing*, vol. 67, no. 11, pp. 1187-1200, November 2007.

[13] P. V. Marsden, "Egocentric and Sociocentric Measures of Network Centrality," vol. 24, no. 4, pp. 407-422, October 2002.

[14] M. Ma and Y. Yang, "SenCar: An Energy-Efficient Data Gathering Mechanism for Large-Scale Multihop Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 10, pp. 1476-1488, October 2007.

[15] M. Macuha, M. Tariq and T. Sato, "Data Collection Method for Mobile Sensor Networks Based on the Theory of Thermal Fields," *Sensors*, vol. 11, no. 7, pp. 7188-7203, July 2011.

[16] N. Meghanathan, “A Data Gathering Algorithm based on Energy-aware Connected Dominating Sets to Minimize Energy Consumption and Maximize Node Lifetime in Wireless Sensor Networks,” *International Journal of Interdisciplinary Telecommunications and Networking*, vol. 2, no. 3, pp. 1-17, July-September 2010.

[17] N. Meghanathan, "Exploring the Performance Tradeoffs among Stability-Oriented Routing Protocols for Mobile Ad hoc Networks," *Network Protocols and Algorithms – Special Issue on Data Dissemination for Large scale Complex Critical Infrastructures*, vol. 2, no. 3, pp. 18-36, November 2010.

[18] N. Meghanathan, "A Comprehensive Review and Performance Analysis of Data Gathering Algorithms for Wireless Sensor Networks,"

International Journal of Interdisciplinary Telecommunications and Networking, vol. 4, no. 2, pp. 1-29, April-June 2012.

[19] N. Meghanathan, "Link Expiration Time and Minimum Distance Spanning Trees based Distributed Data Gathering Algorithms for Wireless Mobile Sensor Networks," *International Journal of Communication Networks and Information Security*, vol. 4, no. 3, pp. 196-206, December 2012.

[20] N. Meghanathan, "Routing Protocols to Determine Stable Paths and Trees using the Inverse of Predicted Link Expiration times for Mobile Ad hoc Networks," *International Journal of Mobile Network Design and Innovation*, vol. 4, no. 4, pp. 214-234, June 2012.

[21] N. Meghanathan and P. Mumford, "A Benchmarking Algorithm to Determine the Sequence of Stable Data Gathering Trees for Wireless Mobile Sensor Networks," *Informatica – An International Journal of Computing and Informatics*, vol. 37, no. 3, pp. 315-338, October 2013.

[22] N. Meghanathan, *Recent Advances in Ad Hoc Networks Research*, Nova Science Publishers, August 2014.

[23] N. Meghanathan, "Stability-based and Energy-Efficient Distributed Data Gathering Algorithms for Mobile Sensor Networks," *Ad hoc Networks*, vol. 19, pp. 111-131, August 2014.

[24] N. Meghanathan, "A Generic Algorithm to Determine Maximum Bottleneck Node Weight-based Data Gathering Trees for Wireless Sensor Networks," *Network Protocols and Algorithms*, vol. 7, no. 3, pp. 18-51, November 2015.

[25] N. Meghanathan, "A Benchmarking Algorithm to Determine Minimum Aggregation Delay for Data Gathering Trees and an Analysis of the Diameter-Aggregation Delay Tradeoff," *Algorithms*, vol. 8, no. 3, pp. 435-458, July 2015.

[26] N. Meghanathan, "A Greedy Algorithm for Neighborhood Overlap-based Community Detection," *Algorithms*, vol. 9, no. 1, p. 8: 1-26, 2016.

[27] P. De Meo, E. Ferrara, G. Fiumara and A. Provetti, "On Facebook, Most Ties are Weak," *Communications of the ACM*, vol. 57, no. 11, pp. 78-84, November 2014.

[28] T. S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd edition, Prentice Hall, January 2002.

[29] G. Santhosh Kumar, M. V. Vinu Paul and K. Jacob Poulose, "Mobility Metric based LEACH-Mobile Protocol," *Proceedings of the 16th International Conference on Advanced Computing and Communications*, pp. 248-253, Chennai, India, December 2008.

[30] H. K. D. Sarma, R. Mall and A. Kar, "E^2R^2: Energy-Efficient and Reliable Routing for Mobile Wireless Sensor Networks," *IEEE Systems Journal*, vol. 10, no. 2, pp. 604-616, April 2015.

[31] W. Su and M. Gerla, "IPv6 Flow Handoff in Ad hoc Wireless Networks using Mobility Prediction," *Proceedings of the IEEE Global Telecommunications Conference*, pp. 271-275, December 1999.

[32] D. Tao, S. Tang and H. Ma, "Low Cost Data Gathering using Mobile Hybrid Sensor Networks," *Lecture Notes in Computer Science*, vol. 7363, pp. 193-206, July 2012.

[33] Y-C. Tseng, F-J. Wu and W-T. Lai, "Opportunistic Data Collection for Disconnected Wireless Sensor Networks by Mobile Mules," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1150-1164, May 2013.

[34] R. Velmani and B. Kaarthick, "An Energy Efficient Data Gathering in Dense Mobile Wireless Sensor Networks," *International Scholarly Research Notices Sensor Networks*, vol. 2014, Article ID: 518268, 10 pages, 2014.

[35] A. J. Viterbi, *CDMA*: *Principles of Spread Spectrum Communication*, 1st Edition, Prentice Hall, April 1995.

[36] H. Zhang and J. C. Hou, "Maintaining Sensing Coverage and Connectivity in Large Sensor Networks," *Wireless Ad hoc and Sensor Networks*: *An International Journal*, vol. 1, no. 1-2, pp. 89-123, January 2005.

Power and Limitations of Formal Methods for Software Fabrication: Thirty Years Later

Edgar Serna M. and Alexei Serna A.
Facultad de Ciencias Básicas e Ingeniería, Corporación Universitaria Remington. Medellín, Antioquia, Columbia
E-mail: edgar.serna@uniremington.edu.co, alexei.serna@uninremington.edu.co

Keywords: formalization, automation, formal languages, software lifecycle, software quality

Received: February 16, 2017

In 1987, Michael Jackson presented his work "Power and Limitations of Formal methods for software fabrication" at the AIT Conference, which analyzed the advantages and limitations of formal methods up to that time. His conclusion was that formal methods had undoubted capabilities and advantages, but they also had serious limitations that prevented their widespread acceptance and adoption. The aim of this paper is to present the current context of formal methods compared with what Jackson described three decades ago. A tour of the strengths and limitations of formal methods is taken through a review of literature in the timeline of the past thirty years. The conclusion is that little progress has been made on this issue in relation to the situation presented by Jackson, and formal methods still need more work from academia, industry and the community.

Povzetek: Prispevek analizira napredek formalnih metod s primerjavo z Jacksonovo metodo izpred trideset let.

1 Introduction

The idea of making mathematics an area of increased use and applicability in different disciplines and contexts can be traced to ancient Greece, where Pythagoras, Plato, Aristotle and Euclid tried to make its study and use accessible to a wide audience [1]. Beyond incipient astronomy, the development of physics, public works and the little there was of mechanics, however, its context continued to be limited to accounting and commercial calculations [2].

For a long time, initiatives were developed with similar objectives, and although some have been relatively successful, especially with the emergence of the engineering disciplines and scientific specializations, the situation appears to remain in other areas [3]. Despite these achievements, the general idea is that mathematics is a field of knowledge that is extremely complicated and difficult to learn and apply to social realities. This attitude has created many myths that have taken hold in the formative process, where students manifest a fear of taking mathematics courses whose content is higher than that in other courses [4, 5). Beyond the fact that these myths may or may not be true, the reality is that there is still no generalized context in which mathematics is appreciated for what it is and not what it seems. For example, one can mention that without the contributions of mathematics, major scientific and engineering developments would not have materialized in areas such as astronomy, physics, chemistry, natural sciences and, more recently, computer science [6]. In the latter, it was adopted as formal methods with the idea of mathematizing processes to develop software and design hardware [7].

Although many moments of the appearance of formal methods can be found in the history of these sciences and many authors have submitted contributions in this regard, it was not until the 1960s that the concept was taken seriously, after the enactment of the so-called software crisis [8]. The community then directed its gaze to mathematics as a lifeline that had helped other disciplines, with the aim of integrating it into software development to solve this crisis and ones that might appear later.

Since that time, various researchers, scientists, authors and organizations have been given the task of mathematizing software development and automating their tests in search of better-quality products. At this time there has been progress, but at the same time, many problems have been found due to the limitations diagnosed in formal methods [9]. In 1987, Michael Jackson [10], a British computer scientist and professor, made a presentation on what he considered the advantages and limitations of formal methods at that time. Three decades after his presentation, it is time to review whether mathematics in computer sciences has overcome those weaknesses and built upon its strengths, if it continues on the same path, or if it needs more time to achieve what was proposed as a lifesaver for software problems.

Building on the work of Jackson, the aim of this article is to take a tour through three decades of publications on the advantages and limitations of formal methods and determine how far we have advanced in their potentiation and/or improvement. This work presents what authors have proposed, innovated and applied regarding the formalization of software development, and the results and conclusions of Jackson are contrasted with the current reality. In addition, current and future challenges for industry, academia and the community regarding the acceptance and widespread use of formal methods are described.

2 Method

To develop this review, it was applied the methodology proposed by Serna [47] to perform reviews of the literature. The search was performed in the following databases: ScienceDirect, ACM Digital Library, Scopus, and Web of Science. It was searched the term Formal Method(s) combined with the terms definition, description, power, limitations, best practices, effective development, and development, first in the title or abstract. By applying this method were selected 78 works including articles, books, works in events and websites. To this population, the sample was applied the inclusion/exclusion criteria (thematic pertinence, author's relevance, focus, quality of results, practical application, and others) in order to determine if the content contributes to the achievement of the goals set in this review. After this procedure, we get 48 works, and after performing a quick reading and applying the concepts of quality in order to determine the value of each of them for this research the final sample was constituted by 40 works.

3 Jackson's conclusions

The work of Jackson [10] had the goal of presenting an analysis of the state of formal methods for the decade of the 1980s. In his presentation, he argued that it would be tedious and boring to describe the advantages and limitations alone, and for this reason, he dedicated almost all of the content to analyzing the fact that the problem was not so much with the formal methods but rather with the body of knowledge itself or the practice of developing software at that time. He asserted that these limitations could not be overcome solely by improving formal methods because they had been imposed by the inherent informality of that practice. To this end, it was one thing to describe the real world with mathematics and a very different thing to do so with natural language because the ambiguity of the latter creates complications for translation.

He also argued that formal methods offered a range of formalization but did not indicate which option to select nor how to apply it in the development of software, a task that corresponded to the developer based on experience and skills. For him, at that time, it was thought that software development was like a manufacturing process, in which descriptions were written in some language and assumed to be analogous to the parts of a mechanical product, where language was the raw material to manufacture them. It was also assumed at that time that software development was primarily a task of composition, not of decomposition, which duplicated the thinking in the forties of engineers in the construction of rockets, who felt that the process consisted of five descriptions: a guidance system, a propulsion system, fuel, a structure and aerodynamic principles, when, to satisfy them all, only the structure, fuel and streamlining were needed. According to Jackson, that process is what defines composition, but it demands creativity and invention that cannot always be automated.

He concluded that formal methods have undeniable advantages and potential, but for the practice of software development at the time, they also had serious limitations that hampered their widespread acceptance. Although most of his work was devoted to demonstrating that most of the blame belonged to the practice of software development, he listed some advantages and limitations, which are presented in Table 1.

Power
Their descriptions are accurate and non-ambiguous Their descriptions can be manipulated by symbols Mathematics provides a high degree of reliability The math is based on a large body of knowledge
Limitations
They restrict the developer to a single language They are focused on transformations between descriptions They tend not to be methods Not all software projects can be formalized Formalisms tend to be isolated from each other Research focuses on individual formalisms A broad integration of formalisms is required

Table 1: Power and limitations of formal methods [10].

Three decades have passed since these claims, and there remains in the environment the feeling that formal methods cannot become an alternative for developing reliable, secure and quality software. In the next section, the development of formal methods over the 30 years after the work of Michael Jackson is described.

4 The last thirty years of formal methods

In Seven myths of formal methods, Anthony Hall [11] presents his analysis of seven myths that existed at that time: 1) they ensure that the software is perfect, 2) they prove that the program is correct, 3) they are only useful in critical systems, 4) they involve complex mathematics, 5) they increase the cost of development, 6) they are incomprehensible to customers, and 7) nobody uses them in real projects. The author claims that many of the things said for or against formal methods were generated from the experiences of developers when applying them but that, although there might be some uncertainty, the reality was that they had more advantages than disadvantages. In his own experience, said Hall, these myths had to be reformulated and established as a type of process because, by then, the transfer from academia to industry was working consistently. To Gaudel [12], the advantages and problems of formal methods were limited to specification and design, as shown in Table 2.

According to Young [13], positive or negative opinions on formal methods had generated controversy for years, while the formal methods community fell short in explaining what they are and what their advantages are due to using descriptions and language that only community members understood and recognized. However, likewise, he asserted that there was a lack of will in the software engineering community to recognize the true value of formal methods. For him, if the goal was to improve the quality of software, it was necessary for both communities to work together to achieve it. For their part, Barroca and McDermid [14] felt that formal methods could be used in two different ways: to produce specifications for conventional systems development and, from that point, to generate formal specifications to verify the accuracy of the program. For these authors and at that time, the benefits of formal methods were as follows: 1) they ensure a consistent interpretation of the specification, 2) they allow verification of the application, and 3) they remove the ambiguity of language. They also listed weaknesses: 1) their development status is low, 2) the specification cannot be validated, 3) they only have mathematical interpretations, and 4) non-functional requirements cannot be adequately articulated in the context.

	Power	**Limitations**
Specification	They make it possible to analyze and encourage it It is structured and reusable It is testable	Correctness cannot be formalized To express the properties, it is necessary to formulate certain aspects of the application domain They only allow external verification
Design	It is the best way to prevent human errors It is a good approximation to zero failures Correctness tests are improved	They are difficult to implement It is still necessary to check the design with a tool The mathematical rigor does not completely eliminate errors The tests do not provide security

Table 2: Power and limitations of formal methods [12].

For Robert Vienneau [15], changes in computing in the 1970s and 1980s generated revolutionary ideas that materialized in formal methods, but there was not yet a unified philosophy about them. Although formal methods were promulgated as a technology applicable to the entire software life cycle, the author wondered why they were not more widely known. He asserted that part of the problem was educational and that many of its limitations would never be overcome, but he was convinced that some restrictions would be addressed through research and practice. Table 3 shows the limitations and advantages that this author described.

Power
They can be used to verify a system They complement the natural language descriptions and give them accuracy They can show that an implementation satisfies a specification More precise specifications are achieved Internal communication is improved They provide the ability to verify designs before running them during the test They offer higher quality and productivity
Limitations
They cannot be used to validate a system They can never replace the knowledge that the engineer has of the system They can never fully replace tests Their applicability is doubtful in systems with many lines of code

Table 3: Power and limitations of formal methods [15].

Liu and Adams [17] concluded that developers utilized formal methods with the hope of refining processes and improving specifications but often did not achieve their goals due to the limitations of this technology: the refinement rules are not sufficient to guarantee that a refined specification satisfies the requirements, and in addition, these rules cannot be reutilized and are difficult to apply in practice. For this reason, they recommended modifying the existing refinement rules, if the objective is to make formal methods widespread. According to Craigen et al. [18], at that time, formal methods were a developing technology, and therefore, they exhibited limitations, like any other such technology. Moreover, it was necessary to determine two key aspects: 1) what were the boundaries between the real world and the world of mathematics and 2) what were the internal limitations of the mathematics. They felt it was difficult to address these issues because, at that time, research placed the real world in doubt; therefore, mathematizing the needs of the client became an informal process. This limitation hindered the widespread growth and recognition of formal methods among software professionals.

Bowen and Hinchey [19] asserted that, for some reason, in the 1990s, formal methods had become one of the most controversial techniques in software engineering. They took as a foundation the work of Hall [12], and they added perspectives that they considered to be new myths: they delay the development process, tools do not support them, they are not really methods, they only apply to software, they are not necessary, they do not have support, and the formal-methods community always uses formal methods. For them, the problem was that more real relationships between academia and industry were required, it was necessary to spread experiences (positive and negative) by using them more widely, more research was needed, and it was necessary to demystify mathematics. Rushby [20] said that, in that decade, formal methods had progressed from an academic curiosity to an industrial reality, and he presented an analysis of their past, present and future. He concluded that to achieve widespread adoption, it was necessary to improve the tools and the scale of their applications, that theoretical research should provide better characterization, and that the software industry must have an open mind regarding this technology because the hardware industry was already enjoying its benefits.

Steve Easterbrook [21] described three case studies in which they applied formal methods to model requirements. They argued that, in contrast to other projects in which Requirements Engineering was used very early on to validate needs, in their experience, they were able to improve the specification. They concluded that the benefits of formal modeling are that it reduces process costs, it enables more effective verification and validation, and maintenance is structured better. Meanwhile, for Kneuper [22], formal methods could help improve the reliability of software development but did not solve all problems. The author described the limitations that make universal solution by formal methods impossible: complete formalization is not possible, there is no guarantee that the informal user requirements are correct and complete, it is difficult (almost impossible) to ensure that the program is correct, they do not determine correct tests, abstraction does not accurately reflect the application, they are applied on a small scale, the technical development is insufficient, and developers do not have the proper mathematical training.

According to John Knight and his team [23], by that time, formal methods had proven benefits, but there were several reasons they lacked broader acceptance: they extend the development cycle, they require complicated mathematics, and the existing tools are inadequate and incompatible with other software packages. However, after applying formal methods to critical application specifications, they concluded that, although several of those reasons could be valid, the main issue was to try to build a comprehensive evaluation framework for the specification. Given that until then, it had not been achieved, it became a stumbling block that the industry could not solve, but given the orientation of the subsequent research, it could be solved in future work. For Jeannette Wing [24], formal methods had limitations in ensuring the security of systems, but they delineated the boundaries of the systems and characterized their behavior more accurately, defining their desired properties with precision and providing a specification in terms of time. She explained that their limitations were because the system operates in an environment, and therefore, formal methods cannot provide total security. She also said that the future of formal methods was promising because, in that decade, many research initiatives were conducted.

Jones et al. [25] conducted research on the contributions of formal methods to requirements engineering and found that some challenges still remained to be overcome: how to couple informality to the formality of requirements, better manage changes, allow traceability, improve accuracy in the validation of the specification, offer better alternatives for non-functional, reconcile some inconsistencies in notations, allow multiple notations and create a body of knowledge that includes all parts involved in this phase. In the same vein, Heylighen [26] stated that the validation of knowledge requires formal expressions of the same and that mathematical determinism requires greater co-pagination with operational determinations. He believed that any formalization process has advantages: it removes ambiguity, it defines the extension of terms, it is independent of time, mathematical language is universal, and it is reusable and testable. However, it also has limitations: it is generally isolated from the context, it has intrinsic limitations, and it assumes normal conditions as implicit contexts, whereas causal factors determine context dependence. He predicted that it was necessary to overcome the limitations and potentiate the advantages to popularize the formalization of software.

Power
They provide units of measure They facilitate the detection of errors They ensure proper operation They reduce errors They help improve abstraction They perform rigorous analysis They are reliable They allow effective test cases
Limitations
They require informality to guarantee the specification It is not easy to see that the implementation satisfies the specification It is not possible to guarantee that the tests are correct The language features are complex Technical environments do not always recognize a formal specification

Table 4**:** Power and limitations of formal methods [37].

Wordsworth [27] summarized the benefits of formal methods as follows: 1) successful cases have been sufficiently reported, and 2) although the basis is mathematical, it is not always necessary. His caveats, however were that formal methods demand some degree of mathematical sophistication, they are not taken seriously in programs of study, users are satisfied with traditional methods, the requirements should be specified more precisely, and developers prefer to code without complete specification. According to [28], formal methods had not yet achieved greater penetration; a wide gap persisted between research in academia and industry application. They maintained that the fact that industry still did not believe in formal methods was due to the loss of scalability, limited access to specialists and the immaturity of tools and techniques.

Peter Amey [29] presented what he called the reality of formal methods and described a series of cases of successful software development. He inquired why, despite its utility, the approach still was not widely used in industry, and he concluded that it was because, typically, they were trying to use development at inopportune times. That is, the error was not how but when. He suggested it would be advisable to start with specification and then reach verification and validation. Edmonds and Bryson [30] argued that the idea of formal methods offered two advantages: 1) the specification is unambiguous, and 2) it can be self-handled syntactically. However, they felt that formal language presented difficulties when trying to translate to or from other languages, which generated two problems: 1) natural language translation is slow, and 2) coding is delayed. Martin Gogolla [31] summarized the benefits and problems of formal methods through a literature review and classified his findings based on indicators: domain of application, persons, properties, tools, understanding, development and general criticism. He concluded that the success or failure of formal methods was not determined by their mathematical properties but by the low usability of existing tools.

For Glass [32], formal methods had existed for a long time but still did not achieve the impact they should have, even though the specification is considerably more understandable, the confidence of the analyst increases because they identify the key problem to solve, errors in the final product are reduced, and maintenance costs are reduced due to the knowledge acquired. Hall [33] attempted to demonstrate the benefits of formal methods and asserted that achieving them was not automatic because there was no better way nor better method to do so. He thought that they were only part of the solution to the problems of software development and that their success depended largely on a clear integration: that intelligent use is required, that researchers should dedicate more time to them, that practical issues of integration and access are as important as the theoretical issues, and that developers must let go of the fear of formalisms. Sommerville [34] wrote that since the 1980s, many engineers and researchers had proposed the use of formal methods as the best way to improve the quality of software products, but that dream still had not come true. He concluded that there were four reasons: 1) the emergence of new methods and management proposals that have helped improve the quality and success of Software Engineering; 2) a new market where quality seems to be of secondary importance; 3) the limited reach of formal methods, which still do not adequately exceed specifications; and 4) limited scalability because large projects are still not satisfied. Still, for him, formalization was an excellent way to discover errors in specification.

David Parnas [35] argued that in the last 40 years, three alarming gaps had appeared in the software field: 1) between research and practice, 2) between software development and traditional engineering disciplines, and 3) between computer sciences and classical mathematics. He argued that advocates of formal methods proposed them as the solution to any of those gaps, even though, up until that time, they could not be verified. He concluded that formal methods had been left with only that perspective, and it was time to rethink them. To achieve this rethinking, he proposed the following: 1) software has problems, but some formal methods with problems will not solve them, 2) more research is needed as is fewer defensive efforts, 3) movement should be slow, not all at once, 4) abstractions should be simple, but true, and 5) our role in the model must be as engineers, not as philosophers and logicians. For the IET [36], formal methods offered the following advantages: they allow a consistent and reasonably complete specification, they reduce the likelihood of error and the cost of detection, and they permit identifying ambiguity in the specification and verification of security requirements. Nonetheless, they were not widely used because the industry did not give them real opportunities and because academia did not view them seriously.

In the article by Batra et al. [37], it became clear that by then, the demand for incorporating formalization in Information Systems had increased because the specification represented actual requirements, and the formal methods could ensure that the implementation met the specifications and demands of security, reliability and quality, although they also had weaknesses. Table 4 describes the advantages and limitations of formal methods for these authors.

In the results of a survey conducted by Fitzgerald [38] on the impact of formal methods on the cost, time and quality of the software, the opinions were divided: 25% reported a decrease and 20% an increase in time; in terms of cost, 33% said there was a reduction and 8% said it increased; and in quality, 88% asserted that it improved and 8% were not sure. In [9] identified eight obstacles to the research, teaching and practice of formal methods: 1) there is insufficient research and teaching, 2) support tools are not sufficient for use on a large scale, 3) students are not taught Computer Science or Software Engineering in mathematical terms, 4) no foundations are strengthened and no attempt is made to present new functions, 5) there are not enough graduates with mathematical knowledge to serve the industry, 6) professors of computer science and Software Engineering do not receive the same training they did 30 years ago, 7) formal methods lack tools for managing versions and configuration control, and 8) professionals who are trained in formal methods do not find support among their industry fellows, so they tend to abandon the practice.

Ishikawa et al. [39] stated that formal methods were increasingly attracting more attention as a solution to the high demand for efficient and reliable software, but a gap had developed between knowledge and teaching with which software engineers were educated and what was required to implement these methods. Meanwhile, for Mayo et al. [40], the limitations of formal methods lay in semantics and traceability, as software and hardware are formally tested during the development process only for explicit statements made ahead of time and as the requirements are validated only in the semantics in which they were tested. According to Gross et al. [41], the exhaustive testing of software systems is intractable and expensive, but if formal methods are incorporated throughout the design process, errors can be identified as they are introduced and the total cost of development dramatically reduced.

5 Analysis of results

After presenting a review of the timeline of formal methods in the past 30 years, looking for the advantages and limitations published by various authors, it is difficult to determine whether the advantages of formal methods have been improved or new ones found. Regarding the limitations, it cannot be stated clearly whether they have been overcome or if instead they have become more acute or others have appeared. The conclusions and presentations in these decades are divided between positivism and negativism toward formal methods, even predicting its possible disappearance from the software development stage. In any case, after analyzing these conclusions, the map of reality of formal methods in these three decades can be seen from three dimensions: from academia, from the community and from the industry. Next, we consider each from the perspective of the results.

In the academic dimension, many works report that formal methods do not achieve the expected penetration because the curriculum in Computer Science and Software Engineering still does not pay adequate attention to them. Thus, these professionals are not educated in applied mathematics, and the few who are have not found peers in industry interested in sharing their knowledge. In this sense, academia should provide an opportunity for formal methods and include them in its content, and in addition, professors need more training to avoid improvising in the classroom. While software problems will not be solved in this way from one moment to another, if progress has already been made, it is better to exploit formal methods to see if they can improve software quality [42].

Furthermore, education systems should take responsibility for the fact that students have mythicized mathematics because they are structured to educate everyone equally and in all areas. With this approach, the skills that each individual may have for one or another discipline are wasted because they do not receive a vocational orientation that tells them how to orient their educational needs. The reality is that to understand mathematics, one must first develop logical and abstract reasoning, but education systems have not contemplated this foundation. Hence, the student prefers more theoretical or less logical areas because they require less effort. This reality is combined with the fact that software development is highly abstract because it is a non-tangible engineering product, and only the outputs and not the processes can be observed. This characteristic differentiates it from other products, such as civil engineering, in which the manufacturing process is constantly evident. The result is that professionals are not adequately trained in mathematics, and thus, formal methods are not practiced nor experienced to exploit their advantages and overcome their limitations.

In the community dimension, the opinion is reiterated in the literature review that formal methods have demonstrated their power in the formalization of specification. For many authors, Requirements Engineering is where formal methods have had the greatest acceptability and where the most success stories are found. The effect has been that the community has devoted less effort to further strengthening procedures and tools to elicit and specify requirements, dedicating itself instead to possibilities in other phases of the life cycle. Some authors criticize this trend in that the community believes that formal methods have already exceeded their goal and that thus another goal should be developed, when the reality is that there are still many problems in the formalization of specification. One recommendation is that what has been achieved so far with requirements should first be strengthened and standardized, and other possibilities can then be considered.

Another issue with the formal methods community is that it is perceived as closed to the participation of other stakeholders because language has limited its communication to the strictly mathematical [43] and because transdisciplinary work is not considered as an alternative. With the development of different disciplines, many researchers interested in contributing from their specialty to the development of other specialties have appeared, as in the case of Neurocomputation for understanding the brain and how people learn. If the formal methods community were more open to contributions from areas such as philosophy, psychology or didactics, it could achieve better results than it has so far.

On the other side is the software industry, a crucial factor in the current map of the reality of formal methods. For years, software was developed in laboratories and with military support because its potential was considered only from that perspective. Over time, society realized that software could be expanded as a solution to other needs, commercialization began, and the software industry appeared with the aim of development for sale. However, software was nonetheless adopted and adapted with the methodology that military scientists had built in their laboratories and applied to develop the products offered on the market. This approach to software development triggered what NATO deemed a crisis in the late 1960s. It is understandable that industry has intended to mainly produce to sell and make a profit, but software is a product that is not manufactured but rather is created (developed); therefore, different procedures are needed from the ones used, for example, to manufacture an aircraft or turbine.

With respect to formal methods, the industry still does not assimilate them at their full potential because it believes that they delay processes and reduce usefulness. The issue is that, if not in industry, where can the proposals of the community and of academia be verified and validated? The three dimensions must work in unison so that a software-dependent society can enjoy better-quality software products. This need does not mean that formal methods are the immediate and magical solution to the software crisis, but as an alternative, it is worth the effort to give them an opportunity while there is no other alternative.

6 Conclusions

The increasing complexity of systems in this century is a challenge for research in Computer Science. The hardware and software that make up these systems have gone in a few years from a few components and lines of code to hundreds of thousands. One need only compare the reality that Jackson described in his work, three decades ago, with the one in which we currently live, in the midst of a software-dependent society with high demands for quality, reliability and product security. However, the software component costs half or more of the total value of the development of a system, and its applications impact economies around the world. For all these reasons, it is necessary to innovate with regard to development processes because, although many think otherwise, we still have not overcome the so-called software crisis of the 60s.

Two key issues can be identified from this analysis of the reality of formal methods in the past three decades. 1) The processes for developing software are still performed as they were more than 50 years ago. Very little innovation has occurred in this sense, and interest seems geared more to proposing and selling new languages and methodologies than to positioning and strengthening the ones that exist and have been proven to work. New approaches are not always the solution, and often what is achieved is to increase the range of options but hinder the work of developers. 2) Formal methods are mathematical and therefore are not yet widespread. In this sense, we must understand that mathematics is based on the understanding and application of logic and pure abstractions, and although computers exist in physical reality, software is basically responsible for representing and manipulating non-physical data. That is, mathematics represents the physical reality, while software models and simulates it.

Formal methods were developed over decades and have introduced principles, paradigms and influential conceptual innovations into computer science for the development software. This is reflected in the fact that a quarter of the Turing Awards between 1966 and 2013 recognize work with a significant component in formal methods. Nonetheless, they still seem to be at a crossroads: as an advantage, they seem well developed and are supported by a large number of applications, users and important critical developments; however, as a limitation, they have ceased to be a major component in computer science and engineering training, few professors are working on them, course offerings at the undergraduate and graduate levels are scarce, they are difficult to apply in important projects, and only a small number of graduates welcome them as a source of work. Moreover, education systems do not adequately develop the logical interpretation and abstracting ability of students [44, 45], which are necessary for logical reasoning. This area of study even seems to have decreased in recent years, which has made new generations increasingly prejudiced regarding mathematics [46].

Thirty years after the work of Michael Jackson, the outlook for formal methods seems not to have changed much. The advantages and limitations that this author described remain, and others seem to have become more acute because variables entered the scene that at the time were unknown: the abandonment of mathematics as the center of the universe of educational systems, the social demand for new and innovative products in a very short time frame, the increasing complexity of problems and the emergence of tools that attempt to displace developers, among others. The facts that the community of formal methods remains isolated from the reality of Computer Science and that the industry does not provide the necessary space to popularize its application are also of little help.

We can thus conclude that, as a solution to the problems of software development, formal methods still have a long way to go. Work over the past thirty years has been slow and achievements few. It has not been possible to form a suitable environment to establish formal methods as an area of academic and industrial interest, and professors have lacked the training and experience to include them in the curriculum. In addition, students still perceive mathematics as an obstacle to be overcome to graduate, rather than as an important part of the learning process. If the formal methods community is integrated and works with other knowledge disciplines, if the industry works a little more and if academia grounds its theories in an attempt to overcome these limitations, formal methods could constitute an alternative way for software to achieve the security and quality expected by society.

7 References

[1] Cohen, B. (1995). A brief history of ‘Formal Methods’. *Formal aspects of computing,* 1(3), 1-10.

[2] Neugebauer, O. (1969). *The Exact Sciences in Antiquity*. USA: Dover Publications.

[3] Serna, M.E. (Ed.) (2015). *Avances en ingeniería*. Medellín: Editorial Instituto Antioqueño de Investigación.

[4] Dani, S. (1993). 'Vedic Mathematics': Myth and Reality. *Economic and Political Weekly,* 28(31), 1577-1580.

[5] Dowling, P. (1998). The Sociology of Mathematics Education: Mathematical Myths/Pedagogic Texts. London: Falmer Press.

[6] Holloway, C. (1997). Why Engineers Should Consider Formal Methods. *16th Digital Avionics Systems Conference*. Irvine, USA, 16-22.

[7] Butler, R. (2001). What is Formal Methods? NASA. Online [Aug 2016].

[8] Naur, P. & Randell, B. (1968). Software Engineering. *Scientific Affairs Division NATO*. Germany, Garmisch.

[9] Bjørner, D. & Havelund, K. (2014). 40 years of formal methods- Some obstacles and some possibilities? *Lecture Notes in Computer Science*, 8442, 42-61.

[10] Jackson, M. (1987). Power and limitations of formal methods for software fabrication. *Journal of Information Technology,* 2(2), 1-6.

[11] Hall, A. (1991). Seven Myths of formal methods. *IEEE Software*, 7(5), 11-19.

[12] Gaudel, M. (1991). Advantages and limits of formal approaches for ultra-high dependability. *Proceedings of the Sixth International Workshop on Software Specification and Design*. Como, Italy, 237-241.

[13] Young, W. (1991). Formal Methods versus Software Engineering - Is there a conflict? *Fourth Testing,*

Analysis, and Verification Symposium. Victoria, Canada, 188-899.

[14] Barroca, L. & McDermid, J. (1992). Formal Methods: Use and relevance for the development of safety critical systems. *The Computer Journal*, 35(6), 579-599.

[15] Vienneau, R. (1993). *A review of Formal Methods*. Technical Report, Kaman Science Corporation.

[16] Gerhart, S., Craigen, D. & Ralston, T. (1994). Experience with Formal Methods in Critical Systems. *IEEE* Software, 11(1), 21-28.

[17] Liu, S. & Adams, R. (1995). Limitations of formal methods and an approach to improvement. *Proceedings Asia Pacific Software Engineering Conference*. Brisbane, Australia, 498-507.

[18] Craigen, D., Gerhart, S. & Ralston, T. (1995). Industrial applications of formal methods to model, design and analyze computer systems - An international survey. New Jersey: Noyes Data.

[19] Bowen, J. & Hinchey, M. (1995). Seven more myths of formal methods - Dispelling industrial prejudices. *Lecture Notes in Computer Science,* 873, 105-117.

[20] Rushby, J. (1996). Mechanized Formal Methods: Progress and prospects. *Lecture Notes in Computer Science*, 1180, 43-51.

[21] Easterbrook, S. et al. (1996). *Experiences using formal methods for requirements modeling*. Technical Report NASA-CR-203085.

[22] Kneuper, R. (1997). Limits of Formal Methods. *Formal Aspects of Computing*, 3(1), 1-16.

[23] Knight, J., Dejong, C., Gibble, M. & Nakano, L. (1997). Why are formal methods not used more widely? *Fourth NASA Langley Formal Methods Workshop*. Virginia, USA, 1-12.

[24] Wing, J. (1998). A symbiotic relationship between formal methods and security. *Proceedings Conf. on Computer Security, Dep. and Assu: From Needs to Solutions*. Williamsburg, USA, 26-38

[25] Jones, S., Till, D. & Wrightson, A. (1998). Formal Methods and Requirements Engineering - Challenges and Synergies. *Journal of Systems and Software*, 40(3), 263-273.

[26] Heylighen, F. (1999). Advantages and limitations of formal expression. *Foundations of Science*, 4(1), 25-56.

[27] Wordsworth, J. (1999). Getting the best from formal methods. *Information and Software Technology*, 41, 1027-1032.

[28] Broadfoot, G. & Broadfoot, P. (2003). Academia and industry meet - Some experiences of formal methods in practice. *Tenth Asia-Pacific Software Engine. Conference*. Chiang Mai, China, 49-58.

[29] Amey, P. (2004). Dear Sir, yours faithfully - An everyday story of formality. In Redmill, F. & Anderson, T. (Eds.), *Practical Elements of Safety*. UK: Springer, 3-15.

[30] Edmonds, B. and Bryson, J. (2004). The insufficiency of Formal Design Methods. *Proceedings Third International Joint Conference on Autonomous Agents and Multiagent Systems*. New York, USA, 938-945.

[31] Gogolla, M. (2004). Benefits and problems of Formal Methods. *LNCS*, 3063, 1-15.

[32] Glass, R. (2004). The mystery of Formal Methods Disuse. *Communications of the ACM*, 47(8), 15-17.

[33] Hall, A. (2005). Realising the benefits of Formal Methods. *LNCS*, 3785, 1-4.

[34] Sommerville, I. (2009). *Software Engineering*. New York: Pearson.

[35] Parnas, D. (2010). Really rethinking 'formal methods'. *Computer*, 34(1), 28-34.

[36] IET. (2011). *Formal Methods*. The IET.

[37] Batra, M., Malik, A. & DAVE, M. (2013). Formal methods - Benefits, challenges and future direction. *Journal of Global Research in Comp. Scien.*, 45, 21-25.

[38] Fitzgerald, J. (2013). Industrial deployment of formal methods: Trends and challenges. In Romanovsky, A. and Thomas, T. (Eds.), *Industrial Deployment of System Engineering Method*s. Berlin: Springer, 123-143.

[39] Ishikawa, F., Yoshioka, N. & Tanabe, Y. (2015). Keys and roles of formal methods education for industry: 10 Year experience with Top SE Program. *Proceedings First Workshop on Formal Methods in SEE & Training*. Oslo, Norway, 35-42.

[40] Mayo, J., Armstrong, R. & Hulette, G. (2015). Digital System Robustness via Design Constraints: The Lesson of Formal Methods. *In 9th IEEE International Systems Conference*. Vancouver, Canada, 109-114.

[41] Gross, K., Fifarek, A. & Hoffman, J. (2016). Incremental Formal Methods Based Design Approach Demonstrated on a Coupled Tanks Control System. *Proceedings 17th International Symposium on High Assurance Systems Engineering*. Orlando, USA, 181-188.

[42] Alvear, A. & Quintero, G. (2015). Integrating software development techniques, usability, and agile methodologies. *Actas de Ingeniería* 1, 94-103.

[43] Polansky, J. & Sinclair, M. (2014). The importance of training in formal methods in Software Engineering. *Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software (RACCIS)*, 4(2), 52-56.

[44] Serna, M.E. (2013). *Prueba funcional del software - Un proceso de Verificación constante*. Medellín: Editorial Instituto Antioqueño de Investigación.

[45] Serna, M.E. & Serna, A.A. (2013). Is it in crisis engineering in the world? A literature review. *Revista Facultad de Ingeniería*, 66, 197-206.

[46] Tucker, A., Kelemen, C. & Bruce, K. (2001). Our curriculum has become Math-Phobic! *Proceedings 32th Technical Symposium on Computer Science Education*. Charlotte, USA, 243-247.

[47] Serna, M.E. (2016). Methodology for perform reliable literature reviews. *Revista Investigación Económica,* in press.

Improved Lane Departure Warning Method Based on Hough Transformation and Kalman Filter

Minjian Liang[1,2], Zhou Zhou[1] and Qingsong Song[1]
[1] School of Information Engineering, Chang'an University, Nan Er Huan Zhong Duan, Xi'an, 710064, China
Email: qssong@chd.edu.cn
[2] Guangdong Special Equipment Inspection and Research Institute: Branch of Zhuhai,
West Renmin Street, Zhuhai, 519002, China

Keywords: automobile active safety, lane departure warning, Hough transform, Kalman filter

Received: June 24, 2016

Abstract: Lane departure warning is the key issues of automobile active safety problems. In this paper, an improved lane departure warning method is proposed based on Hough transformation and Kalman filter, noted as HK-LDWS. At first, the captured colour lane videos are decomposed into frames which are transformed and truncated into binary images subsequently. Secondly, Hough transformation is explored to detect lane lines in the truncated binary images, and Kalman filter is used to predict and track the detected lines. Finally, lane departure warnings are delivered out regarding as the predetermined safe distance based on the lateral distances. The actual road test results show that HK-LDWS can track lanes and make all departure warning correctly besides that it costs less than 35 milliseconds for per frame processing. HK-LDWS is an efficient solution for the lane departure warning problem.

Povzetek: S pomočjo Houghove transformacije in Kalmanovega filtra je razvita nova metoda za povečano avtomobilsko varnost.

1 Introduction

Lane departure is actually a kind of response distortion of sensors, and lane departure warning system, as one kind of key technologies for intelligent vehicles, can at least reduce 24% of traffic accidents which happened due to lane departure [1]. It has an important practical significance to develop lane departure warning system for driver safety alert, traffic accident avoidance, and even for ambient intelligence [2]. Lane departure warning system mainly consists of a HUD (head up display), camera, controller and sensor; when the lane departure system is running, camera (usually placed in the side of the body or the mirror position) will always collect driving lane marking line, through image processing parameters obtained in the current position of the car in the driveway. When the detected vehicle deviation lane, the sensor will collect data and the operating state of vehicle driver, and then the controller sends out the alarm signal; the whole process is about 0.5 seconds that will provide more time for the driver; and if the driver turn on the lights, the normal line change, then lane departure warning system will not make any tips. road and vehicle state perception, lane departure evaluation algorithm and signal display interface are composed he three basic modules of lane departure warning system.

Chen et al. propose a hyperbolic road model based lane recognition and departure warning algorithm, which at first searches out edge points of lane marks, and then identifies marks' parameters by exploring least squares fitting method, tracks identified lanes by particle filter algorithm, and finally judges the departure via spatial-temporal model [3]. Liu et al. propose another lane recognition algorithm based on deformable template and genetic algorithm, which to obtain the best assignments of the template parameters use a genetic algorithm to search the maximum value of likelihood function [4]. Wang et al. propose a B-Snake model based lane detecting and tracking method which use CHEVP (Canny/ Hough Estimation of Vanishing Point) to determine the initial position of lines, and then use minimum mean square error to update the control points of lines [5]. Lin et al propose a fuzzy algorithm for lane detection and tracking [6]. Spline curve model is also verified for lane departure warning system [7].

Although most of the algorithms are demonstrated to be with good alarm performance, there are still improvement requirements as far as noise-robustness and time-consuming are concerned, however. In view of those two concerned, here a lane departure warning method based on Kalman filter-based matching and tracking is proposed, which is noted as HK-LDWS. Firstly, the captured lane video is by frame transformed into binary images and cut by an assigned proportion coefficient. Secondly Hough transformation is explored to detect lines in the images, and HK-LDWS is designed to match and track the detected lines. And finally, a lane departure warning algorithm based on lateral distance is realized to deliver one alarms. The experiments show that HK-LDWS is can make early-warning accurately and efficiently.

2 HK-LDWS

HK-LDWS consists of four modules, i.e., video image pre-processing, lane detection, tracking and departure warning, as shown in Figure 1. The pre-processing module carries out lane image binarization and truncation. The detection module use Hough transformation to extract lines in the images. The following tracking module explores Kalman filter to track the extracted lines based on the assumption that the lines are moving linearly and continuously. The last module provides departure alarms based on the calculated deviated distances from the right lanes.

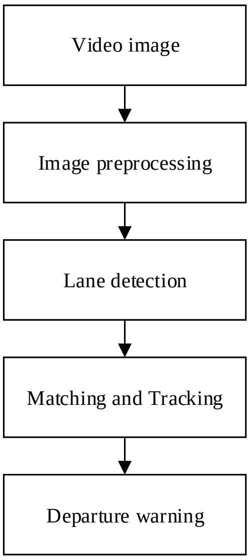


Figure 1: HK-LDWS modules.

2.1 Image pre-processing

This module executes two calculations--binarization and truncation. In order to improve the noise robustness of HK-LDWS, the input video images are at first transformed by frame into grey level images, and then converted into 0-1 binary images. Although lots of pre-processing methods have been proposed for image segmentation such as edge or corner detection, fuzzy logic [8-9], since being as one classic non-parametric and unsupervised adaptive threshold selection method here the Otsu method is exploited, where "0" points correspond to the pixels not or maybe not on the lines, "1" to the pixels on or maybe on the lines within the images [10]. The Otsu method is a global dynamic binarization method, also known as Otsu method, is a kind of grey image value of two commonly used algorithms. The basic idea of the algorithm is: use a grey image per the grey target size, divided into two parts part and the background, in this two within class variance minimum and maximum variance when the threshold value is the optimal binarization threshold.

Further the binary images are truncated by an assigned proportion coefficient to reduce the detection area and avoid noise interferences ahead of vehicle. Notes the original size of the images as $m \times n$, m, n represents image height and width respectively. Notes the assigned proportion coefficient as a, $a \in [0,1]$. The images are truncated from top to bottom with the proportion a, i.e., the top a are cut off, the bottom $(1-a)$ are remained. The remaining $(1-a)$ is taken as the region of interest (ROI), which is to be explored in the following modules.

2.2 Lane detection

In this module, Hough transformation is used to extract lines contained in the ROIs. The lanes are regarded as straight lines since almost all the expressways are designed with the least curvatures [11].

Note one line is described as the Equation $\rho = x*\cos(\theta) + y*\sin(\theta)$. It can be said that those collinear (x, y) pixels in the Cartesian coordinate are mapped to the same one (ρ, θ) point in the polar coordinate.

Being one of the most typical line detection algorithms Hough transformation executes accumulation calculation in the (ρ, θ) coordinate space where the maximum accumulation values correspond to the assignments to the extracted lines' parameters. The detailed lane detection process is as follows and shown in Fig. 2.

(i) Uniformly divide the (ρ, θ) space into small districts. Each district has an accumulator $acc(\rho, \theta)$, of which the initial value is set zero. And in addition, all the "1" pixels within ROI makes up of the initial point set;

(ii) If the point set is empty, the detection process ends; otherwise randomly pick up a pixel from the set to calculate out value of θ corresponding to all ρ, and then add up by one to the $acc(\rho, \theta)$, to which the district corresponds having the largest θ;

(iii) Delete the picked-up point from the set;

(iv) Judge whether there is an accumulator of which the value exceeds to some threshold thr or not.

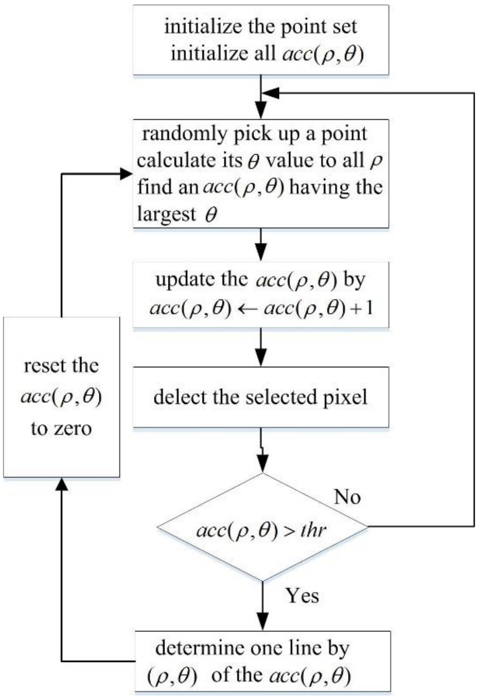

initialize the point set
initialize all $acc(\rho,\theta)$
randomly pick up a point
calculate its θ value to all ρ
find an $acc(\rho,\theta)$ having the largest θ
update the $acc(\rho,\theta)$ by
$acc(\rho,\theta) \leftarrow acc(\rho,\theta)+1$
delect the selected pixel
$acc(\rho,\theta) > thr$
No
Yes
determine one line by
(ρ,θ) of the $acc(\rho,\theta)$
reset the
$acc(\rho,\theta)$
to zero

Figure 2: Lane detection process based on Hough transformation.

If no, back to step(ii); if yes, determine a line based on the accumulator value, and then reset the accumulator to be zero, and delete all the pixels on this line from the set;

(v) Back to step(ii).

2.3 Matching and tracking

Since most expressways are designed to be structured, and the consideration that the lane model should be with cheap computational consumption, it is assumed that the lanes within the two consecutive frames are straight and continuous, and the vehicle is moving at the same one speed. Therefore, the motion model of the lanes relative to vehicles can be supposed to be uniform and linear [12]. Based on this supposition, Kalman filter is explored here for line matching and tracking.

Assume that in the ROIs of each frames it can be extracted out at most 20 lines, all of which make up of a line repository, i.e., $\{(\rho_{ik},\theta_{ik})\}$, $i=1,2,...,20$, $k=0$. Note $X_k=[\rho_k,\theta_k,\mu_k,\omega_k]^T$ as the detected line status, $\mu_k=d\rho_k/dk$, $\omega_k=d\theta_k/dk$ represent radial velocity and angular velocity respectively, k is time step.

Lane update equation is supposed,

$$X_k=AX_{k-1}+W_{k-1} \tag{1}$$

Observation equation,

$$Z_k=HX_k+V_k \tag{2}$$

Where A is the state transition matrix (STM), $A=\begin{bmatrix}1&0&1&0\\0&1&0&1\\0&0&1&0\\0&0&0&1\end{bmatrix}$. STM is used to find the solution of linear system which represented by state-space and in the time-variant case, there are many different functions satisfying these requirements that related to the structure of system. STM need to be determined before analysis on the time-varying solution can continue. $H=\begin{bmatrix}1&0&0&0\\0&1&0&0\end{bmatrix}$ is the measurement matrix, W_k and V_k are system noise and observation noise respectively.

Kalman filter is executed via two phases--time update and measurement update. Time update calculates the estimations of the state vectors through the prediction of current state vectors and their covariance matrix. Measurement update obtains the corrections of the state vectors through residuals computation based on the state vectors' estimations and the latest measurements. It is just via the prediction-correction cycle for Kalman filter to complete the recursive estimation of the state vectors.

Time update equation,

$$\hat{X}_k^-=A\hat{X}_{k-1}^- \tag{3}$$

$$P_k^-=AP_{k-1}A^T+Q \tag{4}$$

Measurement update equation,

$$K_k=P_k^-H^T[HP_k^-H^T+R]^{-1} \tag{5}$$

$$\hat{X}_k=\hat{X}_k^-+K_k\left(Z_k-H\hat{X}_k^-\right) \tag{6}$$

$$P_k=(I-K_kH)P_k^- \tag{7}$$

where $\hat{X}_k^-$, K_k, $\hat{X}_k$, Z_k are the prediction value, the gain, the estimation value, and the observation value at time step k respectively. The covariance matrix for the system noise and the observation noise is noted as Q and R respectively, which are initialized as follows,

$$Q=\begin{bmatrix}0.05&0&0&0\\0&0.05&0&0\\0&0&0.05&0\\0&0&0&0.05\end{bmatrix},\ R=\begin{bmatrix}1&0\\0&1\end{bmatrix}$$

Specifically, the proposed HK-LDWS is as follows, the procedure is shown with Fig.3.

(i) Initialize the line repository to be zero, i.e. $\{(\rho_{ik},\theta_{ik})\}$ all elements are zero, $i=1,2,...,20$, $k=0$, and initialize another 20 matching counters N_i, $i=1,2,...,20$, to be zeros also. One line in the repository has a counter.

(ii) Input the extracted lines which being outputted from the lane detection module at time step k.

(iii) Match the extracted lines with the already existing lines in the repository at time step $(k-1)$ one by one, calculate the distance D_i between each of them, i.e., $D_i=|\rho_{ik}-\rho_{j(k-1)}|+|\theta_{ik}-\theta_{j(k-1)}|*Size_{image}$, $Size_{image}$ is the width of the ROIs, $i,j=1,2,...,20$;

(iv) Update the values of those matching counters N_i, $i=1,2,...,20$. if $D_i\le D_{\min}^*$, the match is successful, the extracted line $\{(\rho_{ik},\theta_{ik})\}$ is replaced by the estimated line $(\rho_{j(k-1)},\theta_{j(k-1)})$ which is generated by Kalman filter, and in addition, the corresponding counter N_i is updated as $N_i=N_i+1$; If the match fails, the repository remains as it was at time step $(k-1)$, but N_i is updated $N_i=N_i-1$. Here $D_{\min}^*$ is noted as a predetermined threshold for the matching distances.

(v) Track the lines in the repository by Kalman filter formulated by Equation (3) ~ (7). Only those lines of which the value N_i exceed to some threshold are being tracked. Under the consideration of noise-robustness and computational efficiency, the threshold is set to be ten, i.e., only those lines which have been uninterruptedly detected in no less than ten consecutive ROIs are to be tracked by Kalman filter; those lines ($N_i < 10$) are not to be tracked at all.

(vi) Update the line repository with the estimated lines by Kalman filter.

(vii) Iterate until the end.

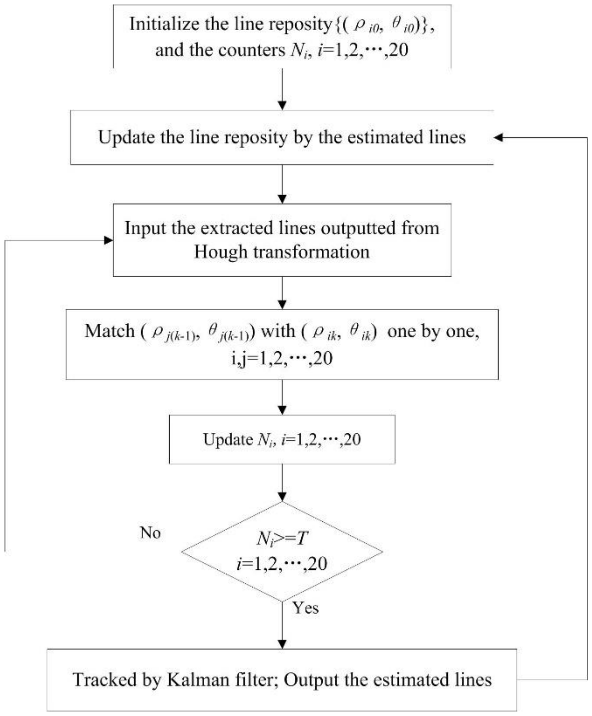

Initialize the line reposity{(ρ i0, θ i0)}, and the counters Ni, i=1,2,⋯,20
Update the line reposity by the estimated lines
Input the extracted lines outputted from Hough transformation
Match (ρ j(k-1), θ j(k-1)) with (ρ ik, θ ik) one by one, i,j=1,2,⋯,20
Update Ni, i=1,2,⋯,20
No
Ni>=T
i=1,2,⋯,20
Yes
Tracked by Kalman filter; Output the estimated lines

Figure 3: Kalman filter-based matching and tracking procedure.

2.4 Departure warning

The lane departure warning algorithm based on lateral distance is realized here [13]. At first, the polar coordinates of the lines obtained from the matching and tracking module are transformed into Cartesian coordinates. And further, the intersection positions that the tracked lines pass through the bottom boundary of the ROIs are calculated out. Therefore, the minimum distance among the intersections and the central point at the bottom boundary of the ROI can be obtained. If the minimum is less than some predetermined safe distance threshold, the module delivers sound alarms.

3 Experiments

The performances of HK-LDWS especially computational efficiency and warning accuracy are evaluated with the situations that the proportion coefficient a is assigned different values. It is defined three performance indices, i.e., effective detection rate (r_d^e), time consumption per frame (t_f), and number of missing alarms (n_a).

It is calculated per frame for the distances (d) between the extracted lines from Hough transformation and the estimated lines generated by Kalman filter as the same way as the distance D_i during matching are calculated. If the value of d is more than five pixels, the extracted lines are considered invalid, otherwise the extracted are effective. Note the number of frames within which the extracted are effective as N_f^e , the total number of frames in the test video as N_f^v . There is $r_d^e = N_f^e / N_f^v$. Note the total consumption time for one whole video as t_v. There is $t_f = t_v / N_f^v$. Missing alarm happens once, i.e., $n_a = n_a + 1$, when the departure occurs it should be delivered out an alarm but no alarm.

The value of the truncated proportion coefficient a is assigned 0.3, 0.4, 0.5, 0.6, and 0.7 respectively. The test result is shown in Table 1. According to Table 1, while a is assigned 0.3, 0.4, 0.5, the means of the distance d are usually less than five pixels, and it has no missing alarms. However, by contrast while a is chosen 0.6 and 0.7, d reach 7.8635 and 13.6823 respectively, and in addition the corresponding number of missing alarms n_a gets to 12 and 27 respectively.

When a is set 0.5, the time consumption per frame t_v is just 33.92ms. When a are set 0.3 and 0.4, t_v rise up to 44.39ms and 45.13ms respectively. By contrast while a increases up to 0.6, t_v decreases to 30.86ms. Compared with the situation that $a = 0.5$, it is reduced by 9% in time consumption, however, the number of effective extracted frames is only 262, the effective detection rate (r_d^e) is only 77.74%, and in addition the number of missing alarms reaches to 12. The computational efficiency for the situation that $a = 0.7$ is also improved where the time consumption per frame is less than 30ms, the warning accuracy gets even worse, the effective detection rate is just a bit more than 50%, the number of missing alarms is up to 20 times, however.

It can be said that the situation that $a = 0.5$ corresponds to the best performances as far as computational efficiency and warning accuracy are

a	d		N_f^v	N_f^e	effective detection rate (r_d^e)	time consumption per frame (t_f) /ms	number of missing alarms (n_a)
	mean	standard deviation					
0.3	3.2537	2.0784	337	327	97.03%	45.13	0
0.4	3.4178	2.1891	337	322	95.55%	44.39	0
0.5	3.2701	2.3588	337	324	96.14%	33.92	0
0.6	7.8635	11.5493	337	262	77.74%	30.86	12
0.7	13.6823	50.2165	337	187	55.49%	29.05	27

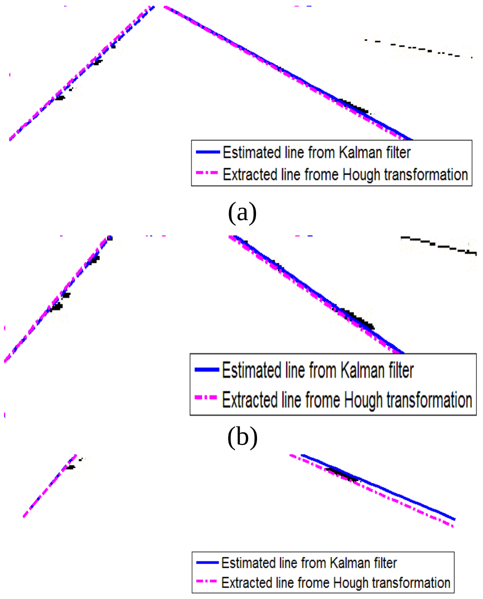
Table 1: Test results for different truncated proportion coefficient assignments.

concerned, where the effective detection rate reaches up to 96.14%, the time consumption per frame is about 33.92ms, and there are no missing alarms. The computational efficiency for the situations that $a > 0.5$ is better than that for the situation $a = 0.5$, the warning accuracy is worse, however. The warning accuracy for the situations that $a < 0.5$ is as the same good as that for the situation $a = 0.5$, but their computational efficiency is slightly worse. The performances corresponding to the situation $a = 0.5$ are fully fit for practical requirements, therefore it is chosen for the HK-LDWS.

It is shown in Fig.4 for the extracted lines from Hough transformation and the estimated lines generated by Kalman filter under three different situations (left $a = 0.4$, middle $a = 0.5$, right $a = 0.6$) where the extracted lines are denoted blue solid, the estimated are red dashed. For the situations a are 0.4 and 0.5, the extracted are almost coincided with the estimated, for $a = 0.6$, there are obvious deviations between the extracted and the estimated, however.

The calculated concreted values of ρ and θ with the situation $a = 0.5$ are demonstrated with Fig.5, where the curves obtained from Kalman filter are more smoothly than that from Hough transformation. Therefore, it can be said that HK-LDWS is more noise-robust than the method purely based on Hough transformation.

During tracking the residuals of Kalman filter are recorded based on Equation (6). Note the residual of ρ and θ as $\Delta\rho$ and $\Delta\theta$ respectively, which are illustrated with Fig.6. As being marked via the rectangular frames there are four regions where obvious abrupt changes happen, and correspondingly the lane departure is being occurred indeed, the hypothesis that the motion model of the lanes is uniform and linear does not hold at this point. After delivering out an alarm the HK-LDWS reset itself.

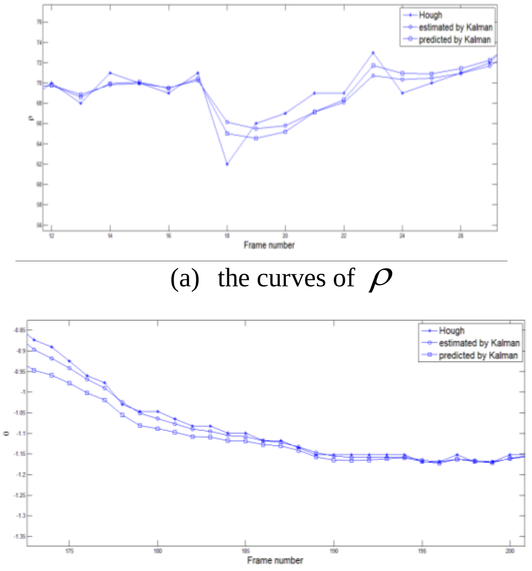

Estimated line from Kalman filter
Extracted line frome Hough transformation
Estimated line from Kalman filter
Extracted line frome Hough transformation
Estimated line from Kalman filter
Extracted line frome Hough transformation

(a)

(b)

(c)

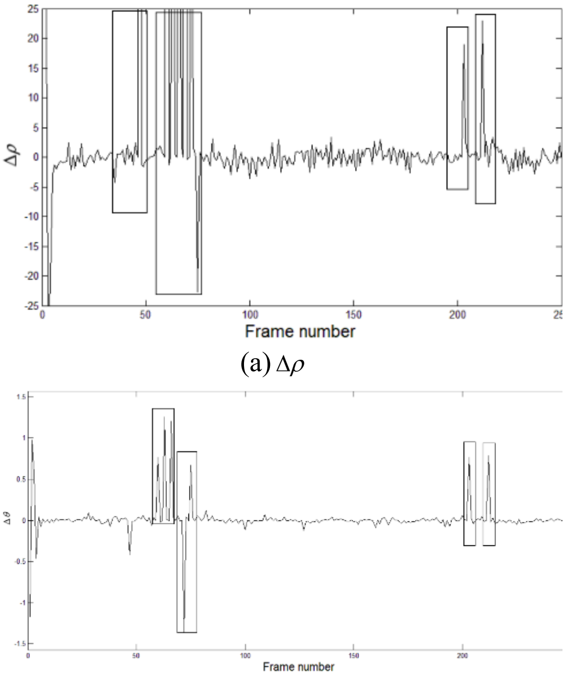
Figure 4: The extracted lines and the estimated lines with different a assignments.



(a) the curves of ρ

(b) the curves of θ

Figure 5: The curve of lines parameter.



(a) $\Delta\rho$

(b) $\Delta\theta$

Figure 6: The residuals of Kalman filter during tracking with the situation $a = 0.5$.

4 Conclusion

HK-LDWS is proposed as a method of lane departure warning based on Kalman filter-based matching and tracking. The pre-processing module gives out the truncated binary images for the subsequent detection module, which extract the lane lines by Hough transformation. And then Kalman filter is explored to track the extracted lines based on the assumption that the lines are moving linearly and continuously. Finally, the lane departure warning algorithm based on lateral distance is realized to deliver out deviation alarms. The actual road test results show that HK-LDWS takes less than 35 milliseconds for per frame processing, can accurately and quickly track lanes and make all departure warning correctly. Its performance of time consumption and tracking accuracy fit for the actual requirements. It’s concluded that the proposed HK-LDWS is an efficient solution for the lane departure warning problems. Time series based image analysis using different pre-processing algorithms need to be studied in future work.

5 Acknowledgement

This work was supported in part by the Fundamental Research Funds for the Central Universities (Grant No. 310824162022) and National Natural Science Foundation of China (Grant No. 61201406).

6 References

[1] J. M. Clanton, D. M. Bevly and A. S. Hodel. A Low-Cost Solution for an Integrated Multisensor Lane Departure Warning System, IEEE Transactions on Intelligent Transportation Systems, 10(1): 47 – 59, 2009.

[2] Marcello Cinque, Antonio Coronato, Alessandro Testa. On Dependability Issues in Ambient Intelligence Systems, International Journal of Ambient Computing and Intelligence, 3(3): 18-27, 2011.

[3] B. Chen, Lane recognition and departure warning based on hyperbolic model, Journal of Computer Applications, 33(9): 2562-2565, 2013.

[4] T. Liu, N. Zheng and H. Cheng. A novel approach of road recognition based on deformable template and genetic algorithm, in Proceedings Of the 2003 IEEE International Conference on Intelligent Transportation Systems, Piscataway, NJ, USA, Vol.2, pp. 1251-1256, 2003.

[5] Y. Wang, E. K. Teoh and D. Shen. Lane detection and tracking using B-Snake, Image & Vision Computing, 22(4): 269–280, 2004.

[6] J. Wang, C. Lin and S. Chen. Applying fuzzy method to vision-based lane detection and departure warning system, Expert systems with applications, 37(7): 113-126, 2010.

[7] S. Mammar, S. Glaser and M. Netto. Time to line crossing for lane departure avoidance: a theoretical study and an experimental setting, IEEE Transactions on Intelligent Transportation Systems, 7(7): 226-241, 2006.

[8] Sudipta Ghosh, Debasish Kundu, and Gopal Paul. A Fuzzy Logic Approach in Emotion Detection and Recognition and Formulation of an Odor-Based Emotional Fitness Assistive System, International Journal of Synthetic Emotions (IJSE), 6(2): 14-34, 2015.

[9] Nilanjan Dey, Moumita Pal, Achintya Das. A Session Based Blind Watermarking Technique Within the NROI of Retinal Fundus Images for Authencation Using DWT, Spread Spectrum and Harris Corner Detection, International Journal of Modern Engineering Research (IJMER), 3(3): 749-757, 2012.

[10] H. Tian, S. K. Lam and T. Srikanthan. Implementing Otsu's thresholding process using area-time efficient logarithmic approximation unit, in Proceedings of 2003 International Symposium on Circuits & Systems, vol.4, IV-21-IV-24, 2003.

[11] S. K. Lee, W. Kwon and J. W. Lee. A vision based lane departure warning system, Proc. Of the 1999 IEEE/RSJ International Conference on Intelligent Robots & Systems, pp.160-165, 1999.

[12] C. Mei, J. Todd and P. Dean. AURORA: A Vision Based Roadway Departure Warning System, Carnegie Mellon University technical report, 1997.

[13] P. Hsiao, K. Hung, S. Huang, W. Kao, C. Hsu and Y. Yu. An embedded lane Departure Warning System, Proc. IEEE 15th International Symposium on Consumer Electronics (ISCE), Singapore, pp.162-165, 2011.

Decision Tree Based Data Reconstruction for Privacy Preserving Classification Rule Mining

G. Kalyani
Research Scholar, Acharya Nagarjuna University, India
E-mail: kalyanichandrak@gmail.com

M.V.P. Chandra Sekhara Rao
Professor, Dept of CSE, RVR & JC College of Engineering, Guntur, India

B. Janakiramaiah
Professor, Dept of CSE, DVR & Dr.HS MIC College of Technology, Vijayawada, India

Keywords: privacy, sensitive patterns, data reconstruction, decision tree, classification rules

Received: June 8, 2017

Data sharing among the organizations is a general activity in several areas like business promotion and marketing. Useful and interesting patterns can be identified with data collaboration. But, some of the sensitive patterns that are supposed to be kept private may be disclosed and such disclosure of sensitive patterns may effects the profits of the organizations that own the data. Hence the rules which are sensitive must be concealed prior to sharing the data. Concealing of sensitive patterns can be handled by modifying or reconstructing the database before sharing with others. However, to make the reconstructed database usable for data analysts the utility or usability of the database is to be maximized. Hence, both privacy and usability are to be balanced. A novel method is proposed to conceal the classification rules which are sensitive by reconstructing a new database. Initially, classification rules identified from the database are made accessible to the owner of the data to spot out the sensitive rules that are to be concealed. In the next, from the non-sensitive rules of the database, a decision tree will be constructed based on the classifying capability of the rules, from which a new database will be reconstructed. Finally, the released reconstructed database to the analysts reveals only non-sensitive classification rules. Empirical studies proved that the proposed algorithm preserves the privacy effectively. In addition to that utility of the classification model on the reconstructed database was also be preserved.

Povzetek: Predstavljena je metoda strojnega učenja, ki skrbi za privatnost podatkov.

1 Introduction

Significant improvements in data storage have led to rise in inexpensive data storage techniques for databases. Improvements in storing and analyzing enormous amounts of data present a challenge to people and organizations for transforming this data into valuable knowledge. Data mining, which involves extorting the patterns that are novel and valuable from mass repositories of data, is efficient in transforming the data into knowledge.

Various data mining algorithms are in usage for mining interesting patterns from the collected data. Patterns like classification rules, association rules and clusters can be discovered with mining techniques. On the other side, in order to get the mutual benefits data will be shared among the collaborated organizations. But, some sensitive information or patterns may exist with in the data which is to be maintained as private, since the revelation of sensitive information or pattern may affect the business deals of the data owner and violates the privacy issues of the data owner as an end user. Hence, along with the need of sharing and collaborative mining, the importance of protecting the information or patterns against disclosure is one of the most important point in the security issues of data mining [1, 2]. To preserve the sensitive information or patterns from unwanted disclosure, privacy preserving data mining (PPDM) has emerged as a security area in data mining and database field [1, 12].

1.1 Classification of approaches in PPDM

PPDM is an interesting research area in the data mining community. It concentrates on the privacy issues of individuals or organizations which are violated due to the disclosure of sensitive information or patterns. PPDM converts the original database into a transformed database in such way that no sensitive data or pattern can be mined from the transformed database. Various methodologies exists in the literature, for this transformation to protect sensitive information or knowledge. A taxonomy for the PPDM techniques based on a set of parameters is discussed and the taxonomy is shown in Figure 1.

Based on the parameter whether the data owner requires privacy for the data or knowledge, PPDM techniques were classified as:

- **Data Hiding Techniques (Protecting Sensitive Data)**
  Data hiding approaches[3, 6, 11] investigate about maintaining the privacy of data or information before applying the data mining techniques on the database. These approaches concentrate on the exclusion of private information from the database before sharing the data with others. Perturbation, sampling, suppression, transformation[17], etc. are the general techniques used to create a transformed database. The final aim of data hiding is, after sharing the transformed database receiver has to get valid data mining results without disclosing the private data of the data owner.

- **Knowledge Hiding Techniques (Protecting Sensitive Knowledge)**
  Knowledge hiding approaches [4, 13] investigate on the protection of sensitive knowledge inferred from the data(instead of the data), by applying the mining tools on the original database. The ultimate goal of knowledge hiding techniques is no sensitive knowledge is to be mined by applying the data mining techniques on the transformed database. Knowledge hiding approaches mainly deals with the following techniques.

  - **Data Distortion Technique**: This technique tries to protect the knowledge by changing the parameters associated with the sensitive knowledge. These techniques works by altering 0s to 1s or vice versa in the specified transactions of the database, which may generate unwanted side effects in the new database[16].

  - **Data Blocking Technique**: In this technique, 0's and 1's related to the data of the sensitive knowledge will be replaced by "?"(Unknown) in selected transactions instead of doing insertion and deletion of items [15].

  - **Reconstruction Based Technique**: This technique reconstructs a database from the sanitized knowledge, extracted from the original database. When compared to the heuristic methods side effects will be reduced in reconstructed database [8, 19, 21].

The paper concentrates on protecting the sensitive knowledge by reconstructing the database from the non-sensitive knowledge mined from the original database i.e. knowledge hiding based on reconstruction based technique.

1.2 Problem motivation

In business organizations, classification techniques reveals a set of classification rules.Among the rules mined, some are crucial for decision making and there by to increase their profits. In order to get some mutual benefits, organizations share their data with others also. By getting their data, others also can identify all the classification rules. In some cases the person who owns the data does not want to reveal some of the rules to others even though the data was shared with them. The set of rules which are crucial and important for gaining the profits must be kept confidential i.e. they ,must not be revealed to others even they have applied classification techniques on the shared data. The set of rules which are to be hidden from disclosure to others are called as sensitive classification rules.

The focus of this paper is on the privacy of classification rules mined from the databases. The need of privacy in classification rule mining was explored with an example scenario [21]. A credit card company agreed to share their credit card approvals to a new home loan company. When people have applied for the credit card, their data will be maintained as a separate record in the database of Credit Card Company. The attributes financial status, experience, gender, salary, age and address are maintained for every person. The class label is maintained as the approval result of their credit card application. After getting the data from the credit card company, the home loan company constructs a classification model to categorize the applicants of home loan. Based on the classification model and predicted results, the home loan company can decide the approval of the home loan to the applicants. The home loan company gets benefited by avoiding the approvals to the wrong applicants based on the data taken from the credit card company. The home loan company can also make use of the credit card company database in another manner to improve their business. By changing the class label to the address attribute, the home loan company can identify the appropriate group or individual customers to send advertising mails about their offers. Hence, to avoid such type of advertising to their customers, the credit card company should modify their database before sharing with the home loan company in such a way that classification rules which are useful for identifying a group of valued customers must not be revealed to the home loan company. The above scenario clearly indicates the need of preserving the sensitive classification rules before sharing the data with the others.

2 Literature review

In the perspective of privacy in classification rule mining, the major part of the work in research concentrates on the privacy of individual data. In [5], privacy of individual data can be achieved by data reduction. In the data reduction method, the effect of non-sensitive knowledge on the sensitive knowledge was analyzed. For preserving the privacy of individual data, a decision tree can be constructed by col-

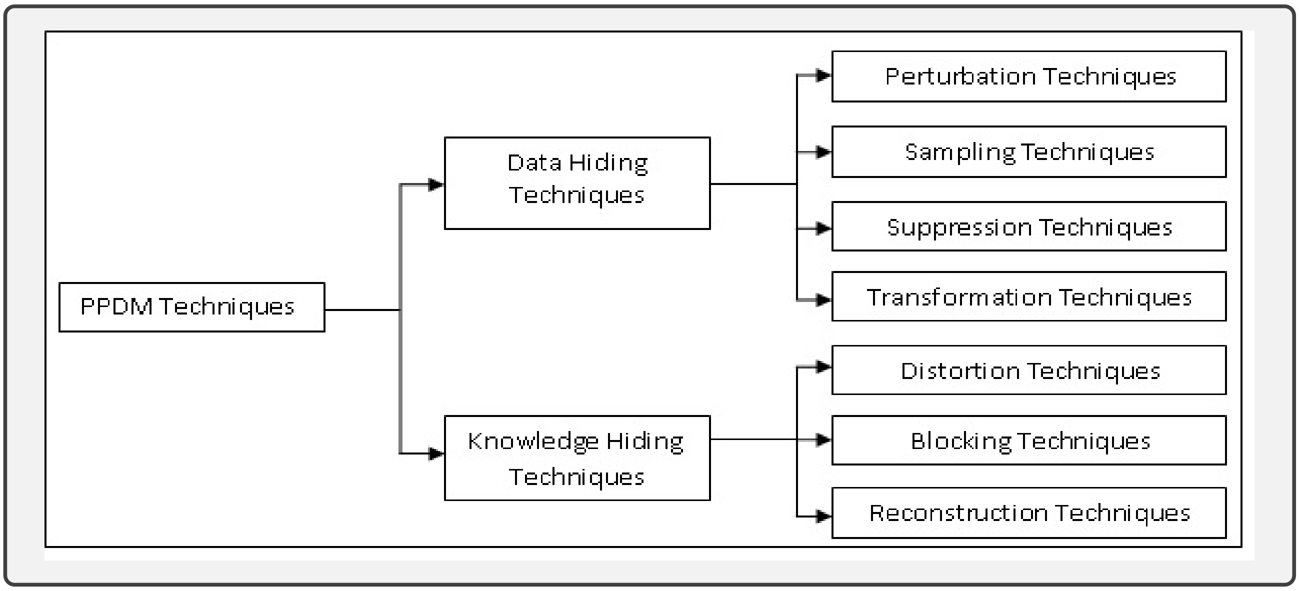


Figure 1: The Taxonomy of the PPDM Techniques.

lecting data from multiple parties without revealing others to their data was proposed in [7, 9].

In [18], the authors projected a classification rule hiding method based on reconstruction of categorical datasets. The methodology modifies the tuples of the original database which contains the values related to both sensitive and non-sensitive classification rules and then makes use of the tuples related to the non-sensitive rules to create its transformed database.

This paper [3] projected a novel method to defend the sensitive classification rules based on the reconstruction process for categorical datasets. Initially, the owner of the data will identify a set of sensitive rules that are to be concealed among the rules revealed from the original database. Later, the set of non-sensitive rules along with the characteristics extracted from the database are used to construct a decision tree. Finally, the new database is reconstructed which reveals only non-sensitive classification rules.

In [20], the authors proposed a template-based technique to protect against the threats caused by data mining functionality. The technique focuses on two points: preserving the privacy of knowledge and increase the usefulness of non-sensitive knowledge that can be derived from the data. Sensitive rules are indicated by a set of "privacy templates". Template includes the sensitive information which is to be concealed, a set of corresponding attributes, and the relationship between the two. Authors proved that suppressing the attribute values is an efficient approach to protect sensitive rules. For a large dataset, identifying an optimal possibility for suppression may be hard, because it needs to do optimization over all suppression's.

In [14], Verykios et al. projected a method for hiding the classification rules which are considered as private. Hiding is achieved before publishing the data on the web through data perturbation approach in categorical databases. The method used the characteristics of sequential covering classification algorithms. Modification will be done to the tuples of sensitive rules in such a way that the alterations are spread to the tuples of the significant non-sensitive rules. The spreading will be proportional to the rank in the rule set. So that, the method guarantees that the sensitive rules are hidden and maintains the current structure of the rule set, thereby the usefulness of the new database is maximized. Authors have proposed another distribution method with a modification to the basic method. Authors have proved that both the methods are effective in terms of privacy and usefulness of the new database.

3 Proposed method

3.1 Problem statement

Consider a database (D) consists of n tuples comprises of m dimensions along with associated labels known as class with number of distinct classes as C. By applying a classification rule mining algorithm on D, number of classification rules (CR) can be discovered. Given a set of classification rules among CR which are treated as sensitive classification rules (SCR $\subset$ CR) by domain expert (the data owner), the process of classification rule hiding is to appropriately reconstruct a database with the intention of mining the reconstructed database (D^1) by using any classification rule mining algorithms, reveals all the non-sensitive classification rules (NSCR=CR - SCR) that are revealed from the original database, whereas all the SCR are shielded from revelation and new rules (originally non-existent rules) cannot be mined.

3.2 Framework

The framework shown in Figure 2 addresses the problem statement. From the original database, a number of classification rules are discovered by applying any classification algorithm, which are useful to the data owner for forecasting purpose. The data owner or domain expert identifies the sensitive classification rules which must be preserved from revelation when classification rule mining algorithms are applied on the database before sharing the data with the others. The proposed method for classification rule hiding reconstructs a sanitized database by considering the original database, set of classification rules generated and a set of identified sensitive classification rules as input. By applying the classification rule mining algorithm on the reconstructed database, only non-sensitive classification rules which are discovered from the original database, are only be discovered and all the sensitive rules will be hidden from disclosure.

3.3 RCRH (Reconstruction based Classification Rule Hiding)Method

The proposed algorithm for classification rule hiding was reconstruction based algorithm, i.e. the transformed database will be reconstructed from the set of NSCR. The required input for the classification rule hiding is, the database D, classification rules CR mined from D and a set of sensitive classification rules SCR among CR which were decided by data owner depending on to whom they wish to share the database. The result of the algorithm is a reconstructed database D^1.

The proposed algorithm first eliminates the SCR from CR which are the possible classification rules from D (step 2 to 4 of Algorithm 1). Then for every rule in CR, calculate a measure called as capability of the rule. The Capability of the rule indicates the number of the tuples that are correctly classified by that rule (step 5 to 6 of Algorithm 1). The process of calculating the capability for a rule was shown in Algorithm 2. Then arrange the rules in the decreasing order of their capability values because high capability indicates the maximum ability of classifying the data in the database D. Now consider the rules in order and construct a decision tree with the non-sensitive classification rules only.

The construction of the decision tree will be as follows: Consider the rules in decreasing order of their capability values. Calculate the information gain of all the attributes of the database with respect to D. Information gain of an attribute is the measure of the difference in entropy before and after the tuples are divided into groups based on that attribute (step 7 to 9 of Algorithm 1). The information gain of an attribute is calculated as: Gain(A)= Entropy(D)- Entropy(D,A). Entropy(D) and Entropy(D,A) can be calculated by using the equations (1) and (2). The process of calculating the info-gain of an attribute was shown in Algorithm 4.

$$E\left(D\right)=\sum_{i=1}^{c}-P_i\log_2 P_i \tag{1}$$

Where D is the database, c is number of distinct class labels, P_i is the probability of the i^{th} class label.

$$E\left(D,A\right)=\sum_{V\in A}P\left(V\right)*E\left(V\right) \tag{2}$$

Where D is a Database, A is an attribute for which entropy is calculated, V is value of an attribute, P (V) probability of value V, E (V) is entropy of value V.

Consider the rule in CR in the decreasing order of capability values. The attributes of that rule are considered in decreasing order of their info-gain values. By considering the attributes in the order of info-gain, construct a path in the decision tree with the attribute having the highest info-gain at the root node. The possible values of that attributes in database D are considered as possible branches from that node. The path will be extended in the similar manner by considering all the attributes in considered rule. The capability of a rule will be considered as a measure for the branch created in the decision tree. The class label of that rule is given as a leaf node in the branch. For the next rules, based on the order of the attributes path will be checked in the decision tree. If the path matches with the existing path it continues and whenever the match fails, the new path will be constructed from that point. The same process will be repeated to all the non-sensitive rules of D (step 10 to 16 of Algorithm 1).

After the decision tree has constructed, then the transformed database will be reconstructed from the decision tree. The process of reconstructing the database will be applied to all the paths of the decision tree by considering only one path at a time. Hence, consider a single path in the decision tree. A path in the decision tree is associated with capability which indicates the influence of that rule on the database D. Insert number of tuples in the transformed database D^1 equal to the capability of that path in the decision tree.

The path in the decision tree may not contain all the attributes of the database D. Hence, if tuples are added in the database for a path in the decision tree, the tuples in the constructed database may contain some missing values related to the attributes which were not existed in the path of the decision tree (step 17 to 23 of Algorithm 1).

The missing values in the reconstructed database are to be filled by using methods to fill the missing values efficiently. The process of filling the missing values is shown in Algorithm 5. Consider all the attributes of the D^1 as TA (step 3 of Algorithm 5). Select an attributes of the D^1 which are having not null values, i.e. the set of the attributes which are having some data values as SA (step 4 of Algorithm 5). Identify the combination of the distinct values in the set of attributes SA, as a string which is indicated by C (step 5 of Algorithm 5). Scan the database D to retrieve the set of tuples which matches

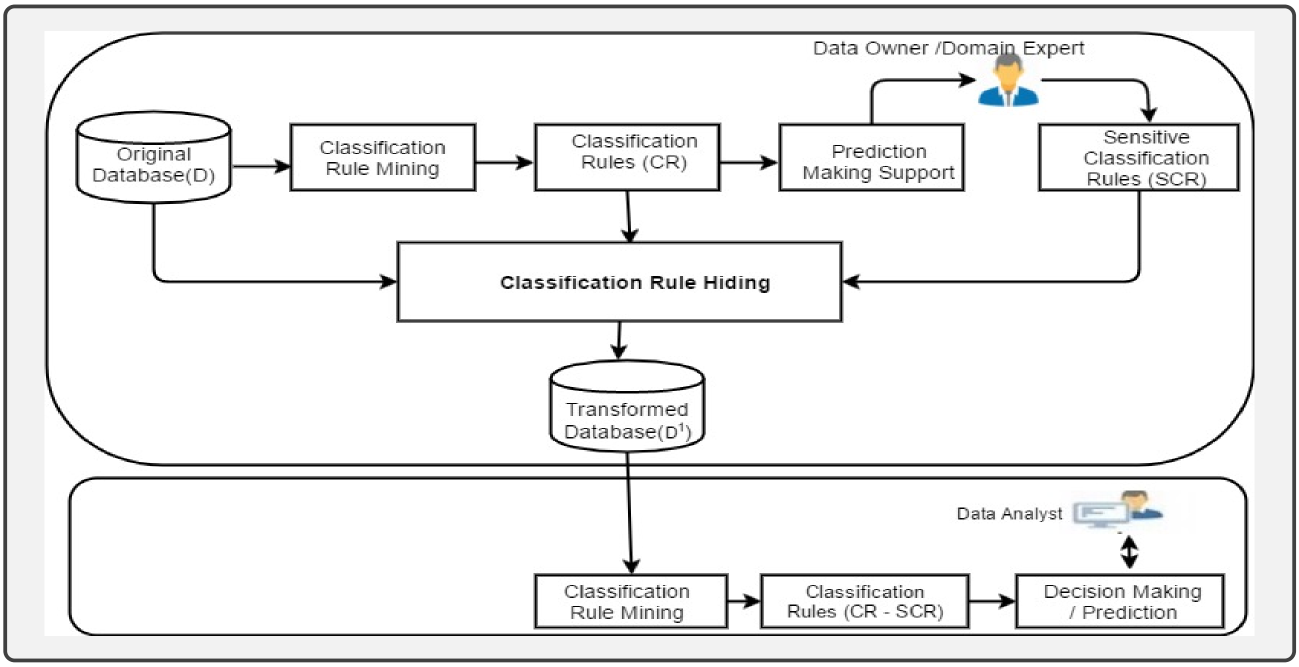

Data Owner /Domain Expert
Original Database(D)
Classification Rule Mining
Classification Rules (CR)
Prediction Making Support
Sensitive Classification Rules (SCR)
Classification Rule Hiding
Transformed Database(D^1)
Data Analyst
Classification Rule Mining
Classification Rules (CR - SCR)
Decision Making / Prediction

Figure 2: The Proposed Framework for Classification Rule Hiding.

Algorithm 1: RCRH(**R**econstruction based **C**lassification **R**ule **H**iding).

```
Data: Original Database D, Classification rules CR, Sensitive Classification Rules SCR.
Result: Transformed Database $D^1$
begin
    for every rule $r \in CR$ do
        if $r \in SCR$ then
            $CR = \{CR - r\}$;                          /* discard the sensitive rules */
    for every rule i = 1 to $|CR|$ do
        capb_i = Capability(R) ;   /* calculating the classifying ability of the rule */
    Ent_D = Entropy (D);                                /* entropy of database D */
    for every attribute $A \in D$ do
        Ig_A = Info-gain(A,D);              /* calculating the gain of attributes in D */
    while ($|CR| > 0$ ) do
        RL=$\{r/r, \forall k \in CR, capb\_r \geq capb\_k\}$ ;   /* select the rule with max capability */
        while (RL is not empty ) do
            attri=$\{x/x \in i \ and \ \forall y \in i, Ig\_x \geq Ig\_y\}$ ;
            create attri as non-terminal node of DT;      /* creating a path in the tree */
            Discard attri from RL ;
        Assign class label of RL as terminal node;           /* adding of terminal node */
    for every path $P \in DT$ do
        count=0 ;
        repeat
            Generate a tuple in $D^1$ with the attributes in P;     /* adding of tuples in $D^1$ */
            count++;
        until (count==capb_P));
        Fill_Missing_Values($D^1$)                 /* to fill the missing values in $D^1$ */
    Return $D^1$;
```

Algorithm 2: Function Capability(R).

```
Data: Original Database D,Classification rule R.
Result: Capability of Rule R.
begin
    Count=0;
    for each tuple $T \in D$ do
        if $T \in R$ then
                                        /* if tuple is classified by rule R */
            Count++;
    Return Count;
```

Algorithm 3: Function Entropy(D).

```
Data: Original Database D, Number of distinct class labels C.
Result: Entropy of D.
begin
    Ent_D = 0;
    for i= 1 to C do
                                 /* getting no.of tuples with $i^{th}$ class label */
        Tc = Select count(*) from D where class=$C_i$;
        $L = log(\frac{Tc}{|D|})$;
        $Ent\_D = Ent\_D + (\frac{Tc}{|D|}) * L$;
    Return ( - Ent_D);
```

Algorithm 4: Function Info_gain(A,D).

```
Data: Database D, No.of distinct values V in A, Ent_D, No.of distinct class labels C.
Result: Information gain of A
begin
    Ent_A = 0 ;
    for i= 1 to V do
        Tv  = select * from D where A=$V_i$;      /* getting tuples with $i^{th}$ value of A */
        E_Tv = 0;
        for j = 1 to C do
                             /* getting the no.of tuples with $j^{th}$ class label */
            Tvc = select count(*) from Tv where class=$C_j$;
            $L = log(\frac{Tvc}{|Tv|)} * L$;
            $E\_Tv = E\_Tv + (\frac{Tvc}{|Tv|}) * L$;
        $Ent\_A = Ent\_A + (\frac{|Tv|}{|D|}) * (-E\_Tv)$;      /* calculating entropy of A */
    Ig_A = (Ent_D) - (Ent_A);                              /* Gain of attribute A */
    Return Ig_A;
```

Algorithm 5: Fill_Missing_Values(D^1).

```
   Data: Original Database D,Reconstructed Database $D^1$.
   Result: Reconstructed Database $D^1$
1  begin
2      repeat
3          TA[ ]= attributes in $D^1$ ;
                                        /* get all the attributes of $D^1$ */
4          SA[ ] = attributes which are not empty in $D^1$ ;
5          Let C be the combination of values in the attributes of SA ;
6          for every tuple $t \in D$ do
7              if (values of SA[] in t == C) then
                                        /* values of the SA[]attributes in tuple */
8                  temp = temp $\cup$ t
                                        /* add the tuple to temp buffer */
9          for each tuple $t \in temp$ do
10             Count the occurrences of each distinct value in the attributes other than SA[] ;
11         Select the attribute A in which more number of occurrence are related to the same distinct value V ;
12         Insert value 'V' in attribute 'A' in $D^1$;
13     until (SA[ ]==TA[ ]);
```

to the combination C in the set of the attributes SA. Let the retrieved tuples be in the buffer temp (step 6 to 8 of Algorithm 5). By scanning the tuples in the temp buffer, count the number of occurrences of each distinct value of the attribute which does not belong to the set SA (step 9 to 10 of Algorithm 5). Then, select the attribute which has the major importance i.e. occurrence of a particular value in the attribute is more than the other values (step 11 of Algorithm 5). The selected value is filled with the value which has the maximum number of occurrences (step 12 of Algorithm 5). Repeat the process of filling the missing values by considering the new set of selected attributes SA, which are filled with the values until the selected attributes are equal to the total set of attributes in the database i.e. all the attributes are filled completely.

Let us consider a small example to demonstrate the working of the proposed method. Table 1 shows the sample database considered for the demonstration. The database contains 30 tuples with 6 attributes A0 to A5 which are binary-valued attributes with two possible values True and False. A class label which has two distinct classes C0 and C1 is associated with each tuple.

Table 2 includes the 12 classification rules which are identified by applying a classification rule mining on the database of Table 1. Rule number 7 of Table 2 is considered as the sensitive classification rule which requires protection from the disclosure.

Consider the non-sensitive rules among the rules mined from the database to construct a decision tree from which the database was reconstructed in classification rule hiding. Hence, among the 12 rules discovered from the database, we are considering 11 rules (other than the rule 7 which is sensitive). For every non-sensitive rule calculate the capability which indicates the classification ability of that rule on the database (Steps 5 to 6 of Algorithm 1). The rules and their capability values are shown in Table 2.

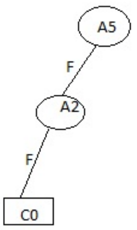


Figure 3: The Decision Tree Path for Rule 1 in Table 2.

Calculate the measure info-gain for every attribute A0 to A5. The info-gain specifies how much information we gained by doing the split using that particular attribute. The attribute which will have maximum info-gain will be better for splitting the database (Steps 7 to 8 of Algorithm 1). The info-gain values of the attributes A0 to A5 are shown in Table 3.

Construction of the decision tree is as follows: consider the non-sensitive rules in the decreasing order of their capability. Hence, consider the rule A2=False & A5=False ⇒ C0 which has highest capability 8. The rule contains the attributes A2 and A5. Order these attributes based on the info-gain. Hence the attributes will be considered in the order A5 and A2. Create a path in the decision tree with the values of the rule in the order A5 and A2. The tree is as shown in Figure 3.In figures False is indicated with "F" and True is indicated with "T".

Table 1: The Sample Database.

Tuple.No	A0	A1	A2	A3	A4	A5	Class
1	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	C0
2	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
3	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	C1
4	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
5	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
6	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	C1
7	TRUE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
8	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	C1
9	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	C0
10	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
11	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
12	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	C0
13	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	C1
14	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	C1
15	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	C0
16	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	C0
17	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	C1
18	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	C0
19	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	C0
20	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	C1
21	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	C1
21	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	C1
22	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
23	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	C0
24	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	C0
25	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	C0
26	TRUE	FALSE	TRUE	TRUE	FALSE	FALSE	C0
27	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	C0
28	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
29	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	C0
30	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	C0

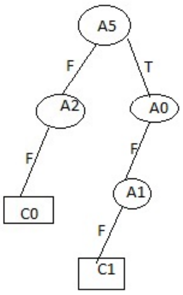


Figure 4: The Decision Tree Path for Rule 2 in Table 2.

Then consider the next rule with maximum capability 5 which is A0=False & A1=False & A5 =True ⇒ C1. Create a path in the decision tree corresponding to this rule as shown in Figure 4.

The next rule in order is A0=True & A2=False & A4=False & A5=True ⇒ C0 with next maximum capability 5. The tree after creating a path in the decreasing order of their info-gain values is as shown in Figure 5.

By repeating the process for all the non-sensitive rules the complete decision tree can be constructed. The complete decision is as shown in Figure 6.

The first path in the decision tree which is with A2 and A5 attributes with false value and class label as C0 is considered and corresponding to this path, 8 (the capability of rule) tuples are inserted into the reconstructed database. The remaining attributes are indicated by null values. To fill these null values consider the combination of the values in the attributes, in which values are available. In this case it is False, False, C0 for the attributes A2, A5 and class correspondingly. By comparing this combination in the original database, the number of tuples found is 8. Count the number of occurrences of each distinct value in each of the attributes A0, A1, A3 and A4. The value True occurred 4, 5, 6 and 6 times in A0, A1, A3 and A4 attributes respectively. The value False occurred 4, 3, 2 and 2 times in A0, A1, A3 and A4 attributes respectively. Since, the majority of the occurrences are for A3 and A4 by value True, the

Table 2: Capability Values of the Classification Rules.

Rule.No	Classification Rules	Capability
1	A2 = False & A5 = False ⇒ C0	8
2	A1 = False & A2 = True & A5 = False ⇒ C0	3
3	A0 = False & A1 = True & A2 = True & A5 = False ⇒ C0	1
4	A0 = True & A1 = True & A2 = True & A5 = False ⇒ C1	1
5	A0 = False & A1 = False & A5 = True ⇒ C1	5
6	A0 = False & A1 = True & A3 = False & A5 = True ⇒ C0	1
7	A0 = False & A1 = True & A3 = True & A5 = True ⇒ C1	–
8	A0 = True & A2 = False & A4 = False & A5 = True ⇒ C0	5
9	A0 = True & A2 = True & A4 = False & A5 = True ⇒ C0	1
10	A0 = True & A1 = False & A4 = True & A5 = True ⇒ C1	1
11	A0 = True & A1 = True & A2 = False & A4 = True & A5 = True ⇒ C1	1
12	A0 = True & A1 = True & A2 = True & A4 = True & A5 = True ⇒ C0	1

Table 3: Information Gain of the Attributes in Table 1.

S.No	Attribute Name	Info - Gain
1	A0	0.0598
2	A1	0.0258
3	A2	0.0598
4	A3	0.0304
5	A4	0.0258
6	A5	0.1835

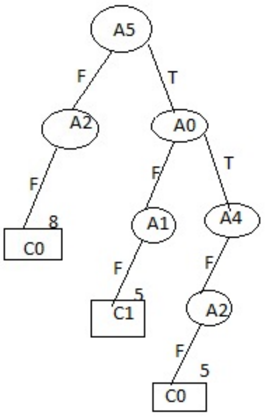


Figure 5: The Decision Tree Path for Rule 3 in Table 2.

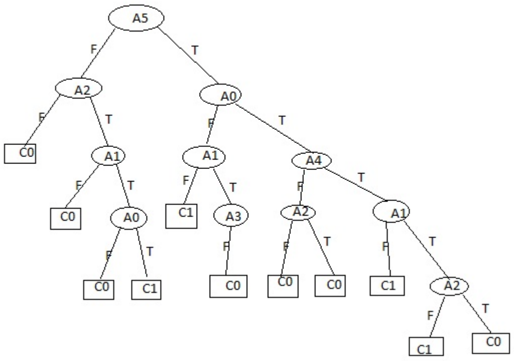


Figure 6: The Complete Decision Tree of all the Rules in Table 2.

missing values of A3 and A4 are filled with value True. Now for the tuples corresponding to the first path, the values are available for A2, A3, A4, A5 and class. Repeat the process for filling of A0 and A1 by considering the combination values in these attributes. After all the attributes are filled up the next path in the tree will be considered in the similar manner until the process of generation and filling will be completed for all the paths in the constructed decision tree. Finally, the reconstructed database obtained is shown in Table 4.

4 Evaluation measures

To assess the performance or efficiency of an algorithm some metrics are to be considered. Classification rule hiding algorithms are also be assessed with a set of measures. The four metrics for the evaluation of the proposed method are as follows:

The first measure is Hiding Failure, which measures the fraction of sensitive classification rules that are revealed from the reconstructed database. Through this, the amount

Table 4: The Reconstructed Database.

Tuple.No	A0	A1	A2	A3	A4	A5	Class
1	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
2	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
3	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
4	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
5	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
6	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
7	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
8	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	C0
9	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	C0
10	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	C0
11	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	C0
12	FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	C0
13	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	C0
14	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
15	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
16	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
17	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
18	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
19	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
20	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
21	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
22	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	C0
23	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	C0
24	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	C0
25	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	C0
26	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	C1
27	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	C0
28	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	C0
29	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	C1
30	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	C0

of sensitive knowledge that is preserved can be also be estimated.

The second and third measures are related to the side-effects of the hiding process. Second metric Miss Cost is one that deals with the fraction of the non-sensitive classification rules which are mined from D and cannot be mined from the reconstructed database D^1. The third metric Artifactual Rules is the fraction of the rules which are not derived from the original database D, but can be derived from the reconstructed database D^1.

The fourth measure is the Usability of the reconstructed database. It is measured through the ability of an attribute to classify the database. In order to increase the usability of the reconstructed database the classification model constructed from the reconstructed database should be as close as to the model constructed with the original database. It means the parameter information gain of the attributes in the reconstructed database must be with the minimum difference with the information gain of the attributes in the original database. Hence usability is calculated as the sum of the differences between the information gains of the attributes in D and D^1.

4.1 Hiding Failure (HF)

The hiding failure is calculated as follows:

$$HF = \frac{|SCR(D^1)|}{|SCR(D)|}$$

where $|SCR(D^1)|$ indicates the number of sensitive classification rules revealed from D^1, and $|SCR(D)|$denotes the number of sensitive classification rules discovered from D.

4.2 Miss Cost (MC)

The miss cost is calculated as:

$$MC = \frac{|NSCR(D)| - |NSCR(D^1)|}{|NSCR(D^1)|}$$

Where $|NSCR(D)|$ refers to the number of non-sensitive classification rules revealed from D and $|NSCR(D^1)|$

Table 5: Characteristics of the Datasets.

S.No	**Name of the Database**	**No.of Instances**	**No.of Attributes**
1	PIMA - DIABETES	768	9
2	GERMAN CREDIT RATING	1000	21
3	CONGRESSIONAL VOTING RECORDS	435	17
4	MUSHROOM	8124	23

refers to the number of non-sensitive classification rules discovered from D^1.

4.2.1 Artifactual Rules (AR)

This is measured as:

$$AR = \frac{|CR'| - |CR \cap CR'|}{|CR'|}$$

Where $|CR|$ and $|CR'|$ stands for, number of classification rules that are generated from D and D^1 respectively.

4.2.2 Usability

The difference between the gains of the attributes is measured as:

$$U = \sqrt{\frac{\sum_{i=1}^{m} \left(\frac{o_i - r_i}{o_i}\right)^2}{m}} * 100$$

Where o_i and r_i are the gain ratios for the i^{th} attribute on D and D^1 and m is the number of attributes in D.

A classification rule hiding algorithm with no hiding failure and artifactual rules i.e. 0% of HF and AR and with reduced miss cost and high usability of the D^1 is considered as an efficient algorithm.

5 Experimental results

Experiments were conducted by considering the real life databases PIMA-DIABETES, GERMAN CREDIT RATING, CONGRESSIONAL VOTING RECORDS and MUSHROOM which are available in UCI data repository[10]. The characteristics of the databases used in the experiments were shown in Table 5.

The results of the proposed method are compared with a classification rule hiding method by considering gain ratios, proposed by Natwichai in [3]. The Natwichai(Gain) method was also a reconstruction based method. Initially it constructs a decision tree from non-sensitive classification rules, and then each path is simply generated as a set of tuples in reconstructed database. In the proposed method, after constructing the tree from the non-sensitive classification rules and at the time of reconstructing the database the missing values are identified efficiently by considering the probability of the possible values in the original database. Hence the usability of the reconstructed database increases by reducing the miss cost and artifactual rules.

Experiments were conducted with four classification algorithms: C4.5(J48), PART, BF TREE and AD TREE which are rule based algorithms available in weka tool. In the experiments, same classification algorithm was used twice i.e. once on D and second on D^1 to discover the classification rules which are used to evaluate the performance measures. All the experiments were done by selecting only one classification rule as sensitive rule while all the remaining as non-sensitive rules. After the classification rules are generated by the algorithm, randomly one rule is selected as sensitive.

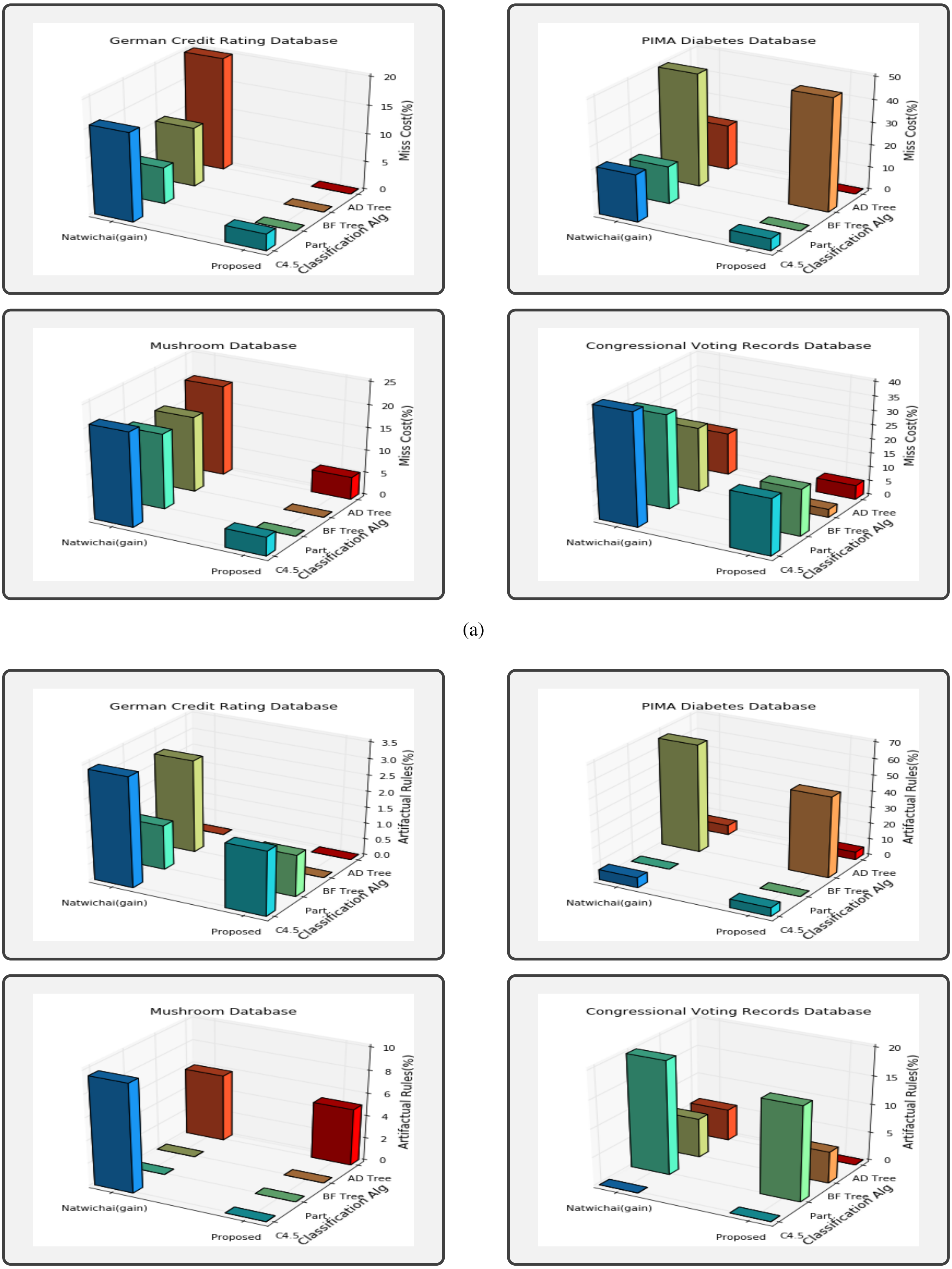
By applying C4.5, PART, BF TREE and AD TREE algorithms on the PIMA-DIABETES database the generated classification rules are 20, 13, 3 and 21 with an accuracy of 84.5, 81.25, 77.21 and 79.69 respectively. After reconstructing the database by using the proposed algorithm, the rules generated are 20, 12, 2 and 20 with an accuracy of 83.98, 80.48, 76.02 and 79.04 respectively. With C4.5 on the reconstructed PIMA-DIABETES database one non-sensitive rule was loosed, and one new rule was generated.

Similarly, by applying C4.5, PART, BF TREE and AD TREE algorithms on the GERMAN CREDIT RATING database the generated classification rules are 103, 78, 39 and 21 with an accuracy of 85.5, 89.7.84.2 and 75.4 respectively. After reconstructing the database by using the proposed algorithm, the rules generated are 101, 77 38 and 20 with an accuracy of 84.9, 89.01, 87.6 and 75.1 respectively. With C4.5 on GERMAN CREDIT RATING reconstructed database one non-sensitive rule was loosed, and three new rules were generated.

By applying C4.5, PART, BF TREE and AD TREE algorithms on MUSHROOM database the generated classification rules are 25, 13, 7 and 21 with an accuracy of 100, 100, 99.95 and 99.9 respectively. After reconstructing the database by using the proposed algorithm, the rules generated are 23, 12, 6 and 20 with an accuracy of 100, 100, 98.53 and 98.14 respectively. With C4.5 and AD TREE on Mushroom reconstructed database one non-sensitive rule was loosed, and with AD TREE one new rule was generated.

By applying C4.5, PART, BF TREE and AD TREE algorithms on CONGRESSIONAL VOTING database the generated classification rules are 6, 7, 36 and 21 with an accuracy of 97.24, 97.47, 98.39 and 97.93 respectively. After reconstructing the database by using the proposed algorithm, the rules generated are 4, 6, 36 and 19 with an accuracy of 96.25, 95.87, 98.14 and 96.89 respectively. With all the four algorithms on CONGRESSIONAL VOTING reconstructed database one non-sensitive rule was loosed, and 1 and 2 new rules were generated with PART and BF TREE respectively.

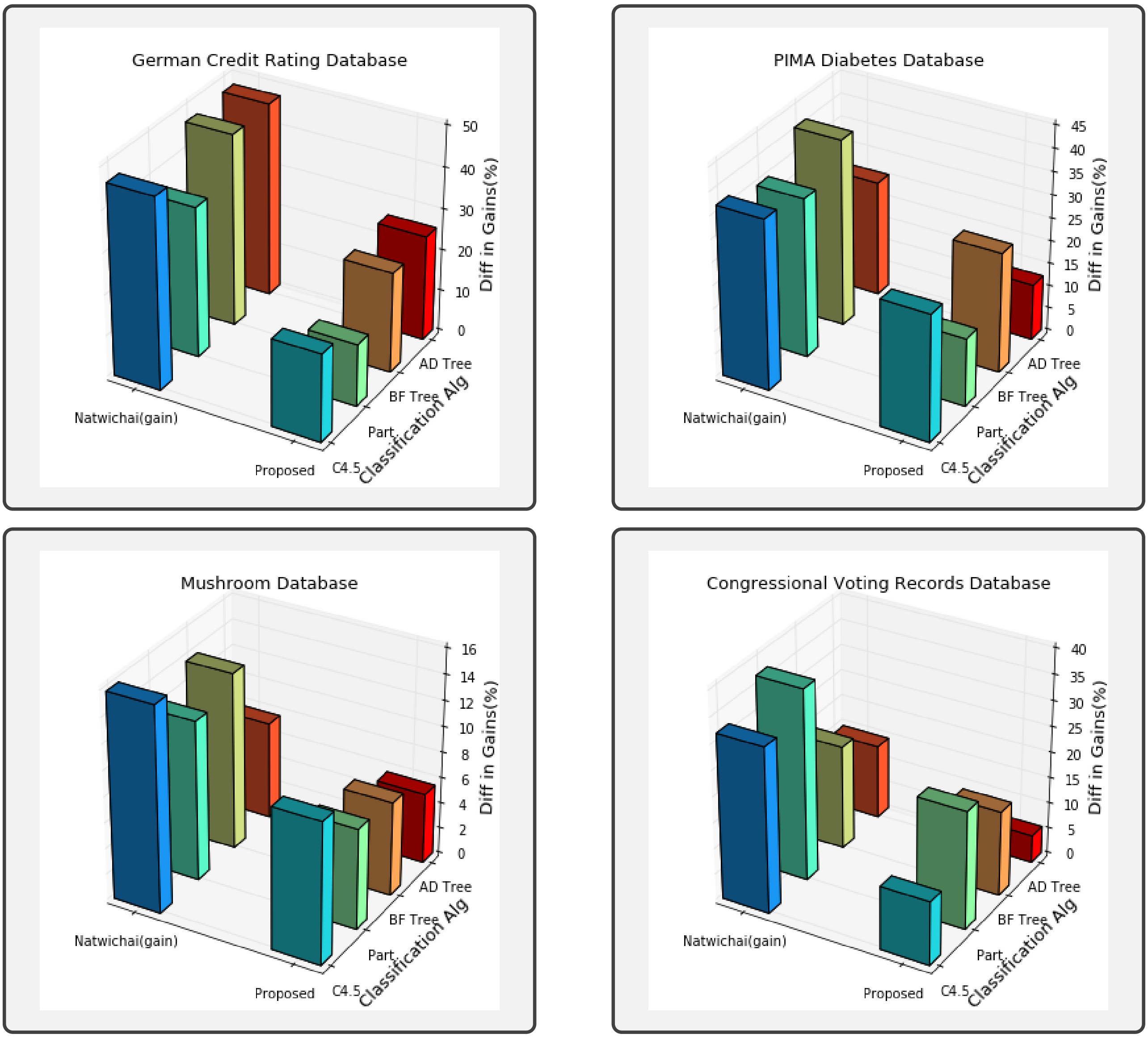
The results of the experiments with the proposed method and Natwichai (Gain) method [3] on four databases with four classification algorithms were shown in Table 6 and Table 7 respectively.



(a)

(b)

Figure 7: (a)Comparison of Miss Cost on Four Databases. (b)Comparison of Artifactual Rules on Four Databases.



(a)

Figure 8: Comparison of Difference between Gains of the Attributes on Four Databases.

Table 6: Experimental Values of Proposed Method.

Database	Classification Algorithm	Original Database		Reconstructed Database		Performance Measures		
		No.of Rules	Accuracy of the Model	No. of Rules	Accuracy of the Model	HF	MC	AR
PIMQ-DIABETES	C4.5	20	84.15	19	83.98	0	1	1
	PART	13	81.25	12	80.48	0	0	0
	BF TREE	3	77.21	2	76.02	0	1	1
	AD TREE	21	79.69	21	79.04	0	0	1
GERMAN CREDIT RATING	C4.5	103	85.5	101	84.9	0	3	2
	PART	78	89.7	78	89.01	0	0	1
	BF TREE	39	84.2	38	87.6	0	0	0
	AD TREE	21	75.4	20	75.1	0	0	0
CONGRESSIONAL VOTING RECORDS	C4.5	6	97.24	4	96.25	0	1	0
	PART	7	97.47	6	95.87	0	1	1
	BF TREE	36	98.39	36	98.14	0	1	2
	AD TREE	21	97.93	19	96.89	0	1	0
MUSHROOM	C4.5	25	100	23	100	0	1	0
	PART	13	100	12	100	0	0	0
	BF TREE	7	99.95	6	98.53	0	0	0
	AD TREE	21	99.9	20	98.14	0	1	1

Table 7: Experimental Values of Natwichai (Gain) Method.

Database	Classification Algorithm	Original Database		Reconstructed Database		Performance Measures		
		No. of Rules	Accuracy of the Model	No. of Rules	Accuracy of the Model	HF	MC	AR
PIMQ-DIABETES	C4.5	20	84.15	16	71.32	0	4	1
	PART	13	81.25	10	66.25	0	2	0
	BF TREE	3	77.21	3	25.73	0	1	2
	AD TREE	21	79.69	17	64.51	0	4	1
GERMAN CREDIT RATING	C4.5	103	85.5	89	71.87	0	16	3
	PART	78	89.7	73	81.93	0	5	1
	BF TREE	39	84.2	35	77.56	0	4	1
	AD TREE	21	75.4	16	61.03	0	4	0
CONGRESSIONAL VOTING RECORDS	C4.5	6	97.24	3	68.62	0	3	0
	PART	7	97.47	5	75.69	0	2	1
	BF TREE	36	98.39	29	80.25	0	8	2
	AD TREE	21	97.93	18	84.93	0	3	1
MUSHROOM	C4.5	25	100	21	89.06	0	5	2
	PART	13	100	10	86.92	0	2	0
	BF TREE	7	99.95	5	81.39	0	1	0
	AD TREE	21	99.9	17	89.62	0	4	1

Generally the performance metrics are to be evaluated in terms of the percentage as % of hiding failure, % of miss cost, % of artifactual rules and % of the difference between the gains of the attributes. The comparison of these parameters for both proposed and Natwichai (Gain) methods was plotted in the Graphs. In both the methods, percentage of hiding failure was zero i.e. no sensitive rules will be generated from the reconstructed databases. So the Graphs are included only for the other two parameters i.e. miss cost, artifactual rules and difference in gains of the attributes in D and D^1. The graphs were drawn in python by considering the Natwichai (Gain) and proposed method on X-axis, the classification algorithms used to generate the rules from the databases are on Z-axis and the parameter used for comparison in terms of percentages on Y-axis.

The comparison of the miss cost on four databases is shown in Figure 7(a). with C4.5, PART, BF TREE and AD TREE algorithms on PIMA-DIABETES the % of miss cost with proposed method was 5.3, 0, 50 and 0 respectively. with same algorithms on GERMAN CREDIT RATING database the % of miss cost with proposed method was 2.9, 0, 0 and 0 respectively. with same algorithms on CONGRESSIONAL VOTING RECORDS database the % of miss cost with proposed method was 20, 16.67, 2.8 and 5 respectively. with same algorithms on MUSHROOM database the % of miss cost with proposed method was 4.17, 0, 0 and 5 respectively. In all the four databases the percentage of miss cost was reduced in proposed method when compared to the existing method.

The comparison of artifactual rules on four databases is shown in Figure 7(b). with C4.5, PART, BF TREE and AD TREE algorithms on PIMA-DIABETES the % of artifactual rules with proposed method was 5.3, 0, 50 and 4.8 respectively. with same algorithms on GERMAN CREDIT RATING database the % of artifactual rules with proposed method was 1.9, 1.3, 0 and 0 respectively. with same algorithms on CONGRESSIONAL VOTING RECORDS database the % of artifactual rules with proposed method was 0, 16.67, 5.7 and 0 respectively. with same algorithms on MUSHROOM database the % of artifactual rules with proposed method was 0, 0, 0 and 5 respectively. In all the four databases the percentage of ghost rules generated was reduced in the proposed method when compared to the existing method.

The comparison of the difference between the information gains of the attributes in four databases is shown in Figure 8(a). with C4.5, PART, BF TREE and AD TREE algorithms on PIMA-DIABETES the % of difference between the information gains with proposed method was 27.42, 14.83, 26.12 and 12.23 respectively. with same algorithms on GERMAN CREDIT RATING database the % of difference between the information gains with proposed method was 21.19, 15.1, 24.3 and 25.5 respectively. with same algorithms on CONGRESSIONAL VOTING RECORDS database the 5.3 respectively. with same algorithms on MUSHROOM database the % of difference between the information gains with proposed method was 10.9, 7.7, 7.3 and 5.5 respectively. The proposed algorithm reduces the difference in gains of the attributes thereby increasing the usability of the reconstructed database which is going to be released without compromising on privacy of the sensitive rules.

Hence, the experimental assessment clearly indicates that the proposed method will reconstruct a database by hiding all the sensitive rules, with minimum loss in non-sensitive rules, minimum artifactual rules generated and by improving the usability of the reconstructed database.

6 Conclusion

Preserving the privacy of sensitive classification rules is a very important issue in application areas that involves collaboration with data sharing. A new algorithm is projected for defending the sensitive classification rules from disclosure. With the projected method which is reconstruction based classification rule hiding, new database will be reconstructed from which sensitive rules will not be disclosed and the side effects of the hiding process miss cost and artifactual rules are kept minimal. Moreover, the usability of reconstructed database will be maximized to make it useful with valid data mining results for a data analyst. The experimental analysis of the results is the evidence to indicate that the proposed algorithm is effective, i.e. it can preserve the privacy and data utility very well.

References

[1] Mahdi Aghasi and Rozita Jamili Oskouei (2016),Privacy Preserving Data Mining Survey of Classifications, *Soft Computing Applications, Advances in Intelligent Systems and Computing*,Springer.

[2] Neha Jain and Lalit Sen Sharma, An Ontology based on the Methodology Proposed by Ushold and King, International Journal of Synthetic Emotions, Volume 7 Issue 1, January 2016 , Pages 13-26, DOI: 10.4018/ IJSE.2016010102.

[3] Reena, Raman Kumar, Effect of Randomization for Privacy Preservation on Classification Tasks, Proceedings of the International Conference on Informatics and Analytics(ICIA-2016), ICPS: ACM International Conference Proceeding Series.

[4] Kalles, Dimitris, Vassilios S. Verykios, and Athanasios Papagelis. "Hiding decision tree rules by data set operations." Information, Intelligence, Systems and Applications (IISA), 2015 6th International Conference on. IEEE, 2015.

[5] Aldeen, Yousra Abdul Alsahib S., Mazleena Salleh, and Mohammad Abdur Razzaque. "A comprehensive review on privacy preserving data mining." SpringerPlus 4.1 (2015): 694.

[6] Xu L.,Jiang C.,Wang J.,Yuan J.and Ren Y.(2014).Information security in big data: privacy and data mining.IEEE, 1149-1176.

[7] Dhanalakshmi, M., and E. Siva Sankari. (2014).Privacy preserving data mining techniques-survey.*In Proc.of International Conference on Information Communication and Embedded Systems(ICICES)*,IEEE.

[8] Chouragade, Komal N., and Trupti H. Gurav. (2014).A Survey on Privacy-Preserving Data Mining using Random Decision Tree.*Int.J.Science and Research*,pp 2891-2894.

[9] Taneja S., Khanna S.,Tilwalia S. and Ankita .(2014).A Review on Privacy Preserving Data Mining:Techniques and Research Challenges.*International Journal of Computer Science and Information Technologies* pp 2310-2315.

[10] Lichman M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[11] Stephen O'Shaughnessy and Geraldine Gray, Development and Evaluation of a Dataset Generator Tool for Generating Synthetic Log Files Containing Computer Attack Signatures, : International Journal of Ambient Computing and Intelligence (IJACI),2011, DOI: 10.4018/jaci.2011040105.

[12] Alka Gangrade , Durgesh Kumar Mishra , Ravindra Patel, Classification Rule Mining through SMC for Preserving Privacy Data Mining: A Review, International Conference on Machine Learning and Computing IPCSIT vol.3 (2011) © (2011) IACSIT Press, Singapore.

[13] Hatice Gunes and Maja Pantic, Automatic, dimensional and Continuous Emotion recognition, International Journal of Synthetic Emotions, Volume 1, Issue 1 © 2010, IGI Global, DOI: 10.4018/jse.2010101605.

[14] Delis A, Verykios V.S, Tsitsonis A.(2010) A Data Perturbation Approach to Sensitive Classification Rule Hiding,*25th Symposium On Applied Computing*.

[15] Gkoulalas-Divanis A, Verykios VS (2010) Association rule hiding for data mining. Springer, New York.

[16] Kadampur M., D.V.L.N S.(2010)A noise Addition scheme in Decision tree for privacy preserving data mining. *Journal of Computing*,137-144.

[17] Upadhayay A. K., Aggarwal A.,Masand R. and Gupta R. (2009).Privacy Preserving Data Mining: A New Methodology for Data Transformation. *Proc. of the First International Conference on Intelligent Human Computer Interaction*, Springer India.

[18] Aliki Katsarou, Aris Gkoulalas-Divanis, and Vassilios S. Verykios (2009) Reconstruction-based Classification Rule Hiding through Controlled Data Modification, *IFIP International Federation for Information Processing, Volume 296; Artificial Intelligence Applications and Innovations III*,Springer pp. 449–458.

[19] Juggapong Natwichai Xue Li Maria E. Orlowska (2006) Reconstruction-based Algorithm for Classification Rules Hiding, *Seventeenth Australasian Database Conference (ADC2006), Hobart, Australia. Conferences in Research and Practice in Information Technology (CRPIT)*, Vol.49.

[20] K. Wang, B.C.M. Fung, P.S. Yu (2005) Template-Based Privacy Preservation in Classification Problems, *5th IEEE International Conference on Data Mining*, pp. 466-473.

[21] Natwichai J., Li X.,Orlowska M. (2005), Hiding classification rules for data sharing with privacy preservation, *Proceedings of 7th International Conference on Data Warehousing and Knowledge Discovery*, Lecture Notes in Computer Science, Springer, pp. 468–467.

Accelerating XML Query Processing on Views

Yin-Fu Huang and Yu-Hsien Cho
National Yunlin University of Science and Technology
123 University Road, Section 3, Touliu, Yunlin, Taiwan 640, R.O.C.
E-mail: huangyf@yuntech.edu.tw, http://mdb.csie.yuntech.edu.tw

Keywords: XPath, labeling schemes, materialized views, T-Bitmap, tag index, value index, navigation, twig query

Received: April 13, 2016

With the widespread use of the eXtensible Markup Language (XML), more and more applications store and query XML documents in XML database systems. Thus, how to efficiently process a query and find the specified patterns conforming the query from XML documents is a crucial issue. In this paper, some processing methods are employed on XML documents to improve document retrieval. First, a materialized view is built from an original document for each query. Then, on each materialized view, auxiliary structures such as T-Bitmap and indexes are also built to further accelerate query processing. Finally, four experiments are conducted to show the superiority of the proposed approach.

Povzetek: Predstavljena je metoda za hitrejše iskanje po bazah XML dokumentov.

1 Introduction

Since XML (eXtensible Markup Language) was widely used to exchange data over the web, more and more applications store and query XML documents in XML database systems. Different from other data formats, an XML document is composed of elements and values with a nested structure, and could be modeled as a tree structure. XPath and XQuery are the standard XML query languages proposed by W3C. They can be used to describe patterns with specified predicates on multiple elements with tree structured relationships. However, how to efficiently process a query and find the specified patterns conforming the query from XML documents is a crucial issue.

In the past, different methods have been proposed in querying XML documents. One of research directions was to build materialized views on XML documents. The goal is to reduce the number of visited nodes during tree traversing by searching from the root of a materialized view, rather than from the root of an original XML document tree. Another research direction was to construct index or access methods to query XML documents for facilitating query processing. In this paper, we integrate the methods from these two research directions as our motivation for accelerating XML query processing on views. The reason is that the performance of a materialized view is better than a non-materialized view because not only these data can be accessed without re-materialization, but also they can be fetched faster by building indexes on these data beforehand. Besides, a materialized view is usually used in accessing a large amount of data, such as data warehouse applications, in support of management's decision-making process through OLAP queries, almost read operations In short, the motivation is for decision makers to accelerate XML query processing in a data warehouse.

In summary, we highlight the contributions of this paper as follows:

1) In this study, we build **materialized views** from an XML document for each query to reduce the search space of queries, and also build **auxiliary structures** such as T-Bitmap and indexes to further accelerate query processing.

2) **Comprehensive experiments** are conducted to verify the superiority of the proposed approach.

3) **The space vs. time issue** is explored when multiple materialized views are integrated together to save the space.

The remainder of this paper is organized as follows. Section 2 presents the previous work proposed in querying XML documents. In Section 3, basic concepts such as query processing and materialized views on XML documents are introduced. In Section 4, we propose a system architecture consisting of view processing and query processing. In Section 5, four experiments are conducted to show the superiority of our approach. Finally, we make conclusions in Section 6.

2 Previous work

As mentioned in Section 1, one research direction on querying XML documents was to build materialized views on XML documents to reduce the number of visited nodes during tree traversing, thereby leading to faster query processing. Godfrey et al. [1], and Murthy and Banerjee [2] proposed SQL/XML syntax for query processing on views, whereas Halevy [3] and Jayavel et al. [4] proposed various query syntax such as join to handle views and focused on the problem of evaluating XML queries over XML views of relational data. However, users must be familiar with these various query syntax. Katsifodimos et al. [5] considered choosing the best views to materialize within a given space budget to improve the performance of a query. Roantree and Liu [6] approach is to segment a materialized view into fragments to minimize the effect of view changes. Bonifati et al. [7] presented an algebraic approach for propagating source updates to materialized views. Wu et al. [8, 9] proposed a bitmapped materialized views approach for optimizing XML queries. Gosain et al. [10] provided a survey of materialized view evolution methods, which aims at studying the materialized view evolution in relational databases and data warehouses as well as in a distributed setting. Gosain and Sachdeva [11] drew several conclusions about the status quo of materialized view selection and a future outlook is predicted on bridging the large gaps that were found in the existing methods.

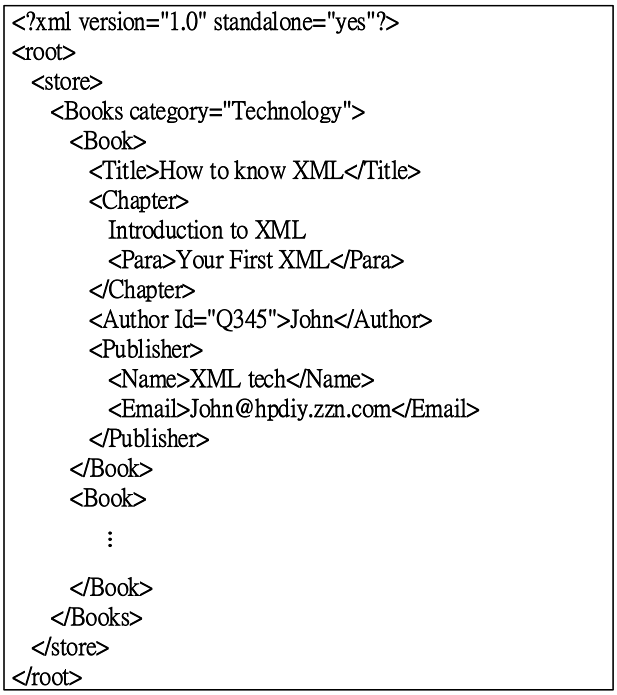
Another research direction was to construct index or access methods to query XML documents, also improving query processing. Some studies investigated constructing index methods to query XML documents [12-15]. Bruno et al. [12] and Jiang et al. [13] used a structure join method to determine element relationships based on the numbering scheme. This method has good performances for an ancestor-descendant axis, but it might fetch useless nodes for a parent-child axis, because all descendant nodes must be accessed to check if they are real children. Therefore, Huang and Wang developed an efficient query processing algorithm for retrieving XML documents [14]. Hsu et al. also proposed a path clustering method based on the concept of summary indexes for the processing of both structural and content queries on XML documents [15]. Karthiga and Gunasekaran [16] used tree-based association rules to mine the semantics from XML documents, which provide information on both the structure and the content of XML documents. The mined knowledge is used to provide the quick answers to queries and an approach called path based indexing is used to improve the speed of data retrieval. Alghamdi et al. [17], and Thi Le et al. [18] proposed approaches to optimizing twig queries by utilizing the semantics/constraints defined in XML schemas. Furthermore, Ordonez focused on the optimization of linear recursive queries in SQL [19]. Subramaniam and Haw [20] proposed an XML labeling scheme that helps quick determination of structural relationship among XML nodes and supports dynamic updates without relabeling nodes in case of update occurrences. Belgamwar et al. [21] follows an upside down approach which explicitly stores the values and only reconstructs the internal nodes, if needed. As a solution, they proposed a compressed internal storage format for native XML database systems where the inner structure of the gathered documents is virtualized. Ferro and Silvello [22] introduced a new paradigm where traditional approaches based on traversing trees are replaced by a brand new one based on basic set operations which directly return the desired subtree, avoiding to create it. Tudor [23] proposed an optimization model for XML data processing based on a heuristic algorithm to extract data from XPath views.

3 Basic concepts

3.1 XML documents

XML is a markup language which was proposed by W3C in 1996. The main purpose of the standard language is to provide data descriptions and data exchanges across different platforms. Like other markup languages, the contexts of XML are declared between start and end tags; however, especially different from others, the tags can be flexibly defined by users to describe data, and furthermore XML is supported in different platforms and systems. That is why it becomes the most common format for data exchanges.

An XML document is with a nested structure, and it could be represented as a rooted, ordered, and labeled tree structure. Figure 1 and Figure 2 illustrate an XML document and its corresponding tree representation, respectively. In the document, there is a unique root element called “root” and one of the descendant elements, called “Book”, has seven child element nodes; i.e., Title, Chapter, Para, Author with an attribute node “Id”, Publisher, Name, Email, and their texts. The symbols as shown in Figure 2 are circles, rectangles, and triangles; they represent elements, texts, and attributes, respectively.



```
<?xml version="1.0" standalone="yes"?>
<root>
  <store>
    <Books category="Technology">
      <Book>
        <Title>How to know XML</Title>
        <Chapter>
          Introduction to XML
          <Para>Your First XML</Para>
        </Chapter>
        <Author Id="Q345">John</Author>
        <Publisher>
          <Name>XML tech</Name>
          <Email>John@hpdiy.zzn.com</Email>
        </Publisher>
      </Book>
      <Book>
          ⋮
      </Book>
    </Books>
  </store>
</root>
```

Figure 1: XML document.

3.2 XPath

XPath (XML Path Language) is an expression language for addressing and querying an XML document. In XPath expressions, each step is separated by "/" and contains three components: **axis**, **node test**, and **predicate**. **Axis** defines the relationship to be followed in the document tree. **Node test** defines what kind of nodes is required. **Predicate** is optional and provides the capability to filter nodes, according to selection criteria.

Given an XPath example "//child::Publisher [child::Name='XML tech'] /child::Email" , it is to get the email of the publisher whose name is "XML tech". When navigating the XML document, it must start from the root element "root", then the descendant node "Publisher". Beneath "Publisher", we search the child nodes to find the node called "Email". Besides, during the search, it must have a child node called "Name" whose text matches with the specified predicate "XML tech". In general, the example above can be abbreviated to "//Publisher [Name ='XML tech'] /Email".

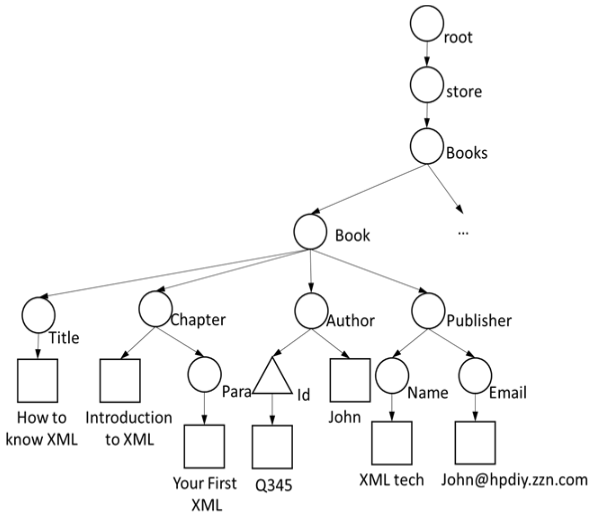


Figure 2: XML document tree.

3.3 Labeling schemes

One of the major query searches is to determine the relationships between nodes. In order to determine element relationships quickly, several different labeling schemes have been proposed. O'Connor and Roantree categorized labeling schemes into containment schemes, prefix schemes and prime number schemes [24]. Here, labeling schemes are classified into prefix-based ones and region-based ones (or containment schemes).

Dewey code [25] is a prefix-based labeling scheme that records the position information of a node, according to the path from the root to the node. For example, Dewey-id of node "Para" is 1.1.1.1.2.2, and indicates that we can get node "Para" if we search alone the path (the first node of level 1, the first node of level 2, the first node of level 3, the first node of level 4, the second node of level 5, the second node of level 6). Besides, since (1.1.1.1.2) is the prefix of (1.1.1.1.2.2), the relationship between node "Chapter" (1.1.1.1.2) and "Para" (1.1.1.1.2.2) can be deduced as a parent-child one. However, the drawback of the prefix-based labeling scheme is its lengthy Dewey codes, especially when the levels of an XML document tree are too deep.

The region-based labeling scheme [12] is another numbering scheme. The label contains three elements (start, end, level) where the start value and end value forms a region. The region of an upper-level node (i.e., ancestor or parent) must cover those of lower-level nodes (i.e. children or descendants). In other words, if node A covers node B, then *A.start* < *B.start* and *B.end* < *A.end*. Besides, the level value represents the node level in a document tree. With the coverage information, we can determine the relationships between nodes quickly. As for the labeling, we can label each node by traversing an XML document tree in a depth-first search way.

3.4 XML document storage

An XML documents can be stored in a few different forms, such as in flat files, in relational databases, and in native XML databases. For an XML document to be stored in flat files, we need to parse the files in advance before accessing them. Although it is the simplest form, the parsing time would be very lengthy when the XML document size is too large. Besides, it also incurs multi-user access and concurrency control problems. For an XML document to be stored in relational database, since the XML document is a tree structure, it must use some middleware to translate the XML format into relational tables. Besides, when querying the XML document, it is also necessary to translate a query into an SQL statement, and execute join operations repeatedly among different relation tables, so that it exposes lower efficiency. Native XML databases aim to provide complete XML document storage and manipulation. Different from other database systems, native XML databases use an XML document as a basic unit of storage, and defines an XML model used to store and retrieve XML documents.

3.5 Materialized views

A view is a virtual and derived table defined by users for facilitating to express a complicated query. Rather than physically stored as parts of a database, a view definition is merely recorded by the database system. It is evaluated only when a user issues a query involving this view. However, a materialized view is the one which is physically stored in the database, in addition to its definition. Absolutely, the performance of a materialized view is better than a non-materialized view because not only these data can be accessed without re-materialization, but also they can be fetched faster by building indexes on these data beforehand. Thus, a materialized view is usually used in accessing a large amount of data, such as a data warehouse or in business intelligence applications, where we need to take more time to query them. A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process. It can be accessed by decision makers through OLAP queries, almost read operations. In short, our design is for decision makers to accelerate XML query processing in a data warehouse.

In this paper, based on native XML databases, we use materialized views to query required data from an original document. Here, a materialized view can be defined using the "CREATE MATERIALIZED VIEW" function and an XPath expression. For the materialized views on an original document, we build auxiliary files and construct indexes using numbering schemes to avoid unnecessary sub-tree traversal, thereby improving the navigation efficiency of a query.

4 System architecture

4.1 Overview

In order to achieve faster query processing on the views defined in a native XML database, we propose a system architecture consisting of an offline phase and an online phase, as shown in Figure 3. In the offline phase called view processing, we build view-relevant structures such as T-Bitmap and indexes to accelerate later query processing. In the online phase called query processing, the system can promptly respond to view-based queries, utilizing the T-Bitmap and indexes built beforehand.

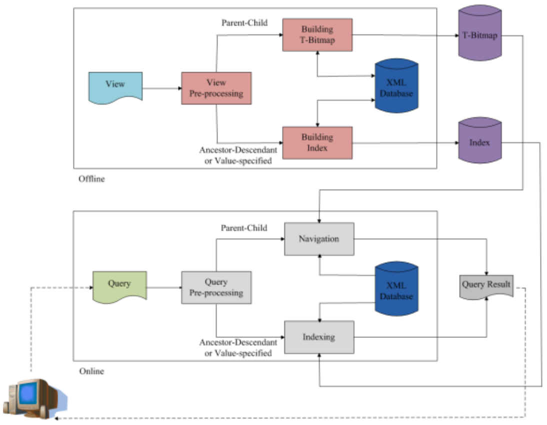


Figure 3: System architecture.

4.2 View processing

In this section, the motivation of view materialization is introduced first. Then, we build relevant structures such as T-Bitmap and indexes on materialized views to further accelerate query processing.

4.2.1 View pre-processing

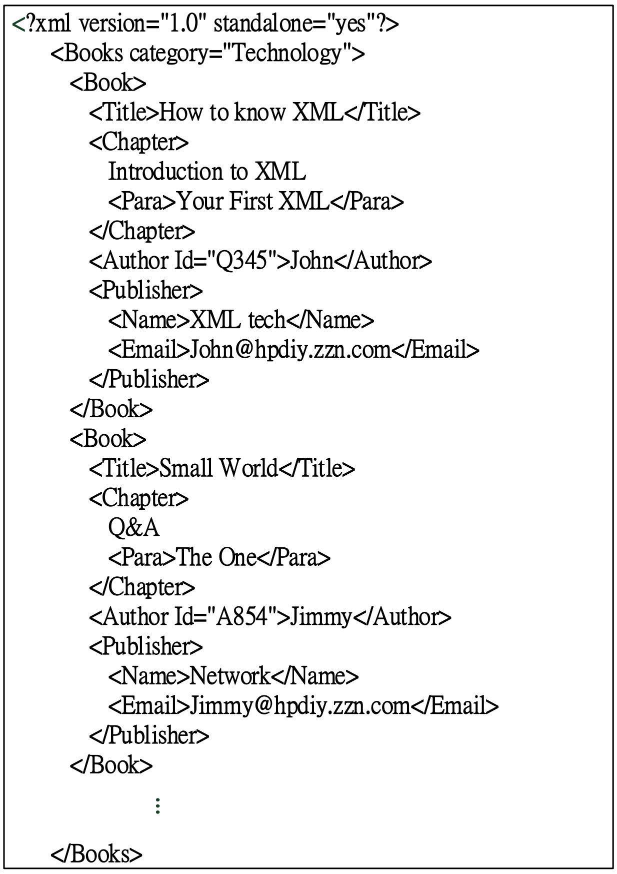
Usually, an Xpath expression is used to address and query an XML document. However, for the query execution, the system always searches an XML document tree from the root. When a query is frequently executed, the system performance would be degraded since a large amount of unnecessary sub-tree traversal cannot be avoided. For the query with an Xpath expression as shown in Figure 4, we can define a materialized view beforehand, which is rooted from node "Books" with an attribute "category" matching with the specified predicate "Technology", as shown in Figure 5. Then, the materialized view can be created from the original document, as shown in Figure 6. Thus, rather than traversing the original document tree always from the root, the system only needs to search the materialized view, thereby improving the navigation efficiency of the query.

```
XPath:/root/store/Books[@category="Technology"]
/Book[Title="How to know XML"]/Publisher
/Email='John@hpdiy.zzn.com'
```

Figure 4: Query with an XPath expression.

```
CREATE MATERIALIZED VIEW mv AS(
SELECT extract(sys_nc_rowinfo$,
'/root/store/Books[@category="Technology"]')
FROM XMLTABLE);
```

Figure 5: View definition.



```
<?xml version="1.0" standalone="yes"?>
    <Books category="Technology">
      <Book>
        <Title>How to know XML</Title>
        <Chapter>
          Introduction to XML
          <Para>Your First XML</Para>
        </Chapter>
        <Author Id="Q345">John</Author>
        <Publisher>
          <Name>XML tech</Name>
          <Email>John@hpdiy.zzn.com</Email>
        </Publisher>
      </Book>
      <Book>
        <Title>Small World</Title>
        <Chapter>
          Q&A
          <Para>The One</Para>
        </Chapter>
        <Author Id="A854">Jimmy</Author>
        <Publisher>
          <Name>Network</Name>
          <Email>Jimmy@hpdiy.zzn.com</Email>
        </Publisher>
      </Book>
                ⋮

    </Books>
```

Figure 6: Materialized view.

Before building T-Bitmap and indexes on a materialized view to further accelerate query processing, we must determine the relationships between nodes (i.e., parent-child axes and ancestor-descendant axes) in a materialized view using the region-based labeling scheme as mentioned in Section 3.3. We traverse a materialized view and label nodes in a depth-first search way. When a node is visited first, its start value is created; when we leave the node, the end value is labeled. After traversing the whole materialized view, all the nodes in the view are completely labeled as shown in Figure 7.

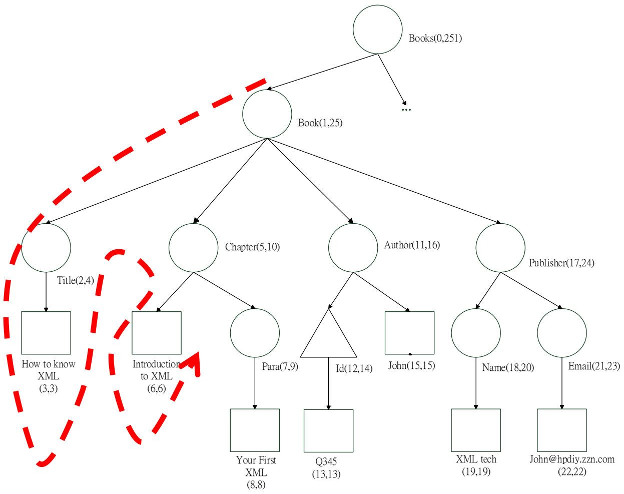


Figure 7: Labeling nodes in a depth-first search way.

4.2.2 Building T-Bitmap

T-Bitmap is a bit string type, which is used to record what descendant nodes are beneath a current node. First, a dictionary recording the positions in T-Bitmap and the corresponding tags is created, as shown in Table 1. Then, the T-Bitmap value on each node can be calculated using OR operators. For an example as shown in Figure 8, the T-Bitmap of node "Publisher" can be calculated by combining the T-Bitmaps of "Name", "Email", and itself using OR operators.

Table 1: Dictionary: positions in T-Bitmap and corresponding tags.

Position	0	1	2	3
Tag	Books	Book	Title	Chapter
Position	4	5	6	7
Tag	Para	Author	Id	Publisher
Position	8	9	-	-
Tag	Name	Email	-	-

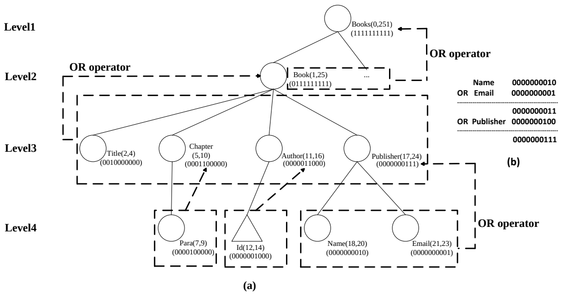


Figure 8: Combing T-Bitmaps using OR operators.

4.2.3 Building index

Here, we use labeling codes to build two kinds of index trees; i.e., tag index trees and value index trees. To illustrate the tag index construction, we extend the storage model in Figure 7. As shown in Figure 9, we can see a lot of nodes with the same tag names but with the different labeling codes; e.g., node "Email(21, 23)" and "Email(46, 48)". We can build the index tree of each tag using the start values in labeling codes as keys and the well-known B+-tree algorithm, as shown in Figure 10 where the pointers of a leaf node in the tag index tree indicate the positions of corresponding nodes in the materialized view. For the query with an XPath: "Book//Email", when processing the current node "Book(26, 50)", we can use the tag index tree of "Email" to locate each leaf node by following the dotted path, and find out node "Email(46, 48)" covered by node "Book(26, 50)".

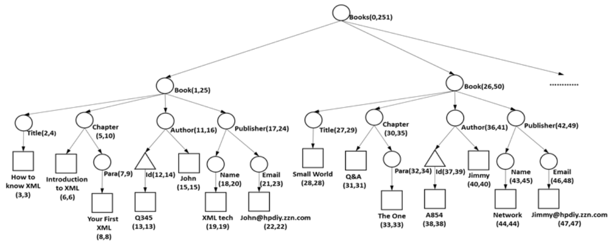


Figure 9: Extended storage model.

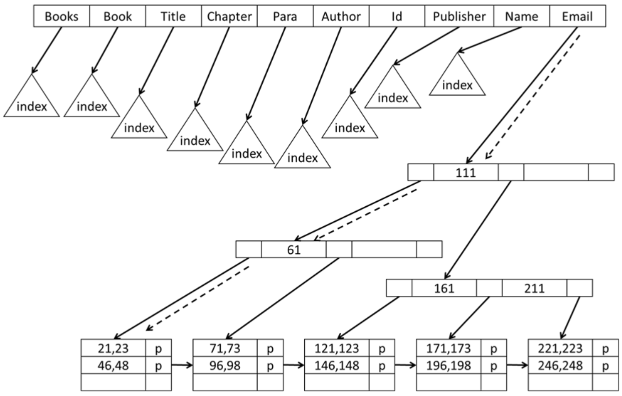


Figure 10: Tag index tree.

Besides, we can also build a value index tree according to the text values of nodes in the document, as shown in Figure 11. The construction method is the same as that used to build tag index trees. However, we generate only one value index tree for each materialized view, and the records of a leaf node are with the [text, start, pointer] format where the pointers also indicate the positions of corresponding nodes in the materialized view.

4.3 Query processing

In this section, the query transformation based on a materialized view is introduced first. Then, according to different axes specified in the transformed query, we make use of the T-Bitmap and indexes built in the view processing to accelerate query processing. Finally, we also introduce subsequent processing for a query specifying the particular predicate in an Xpath expression.

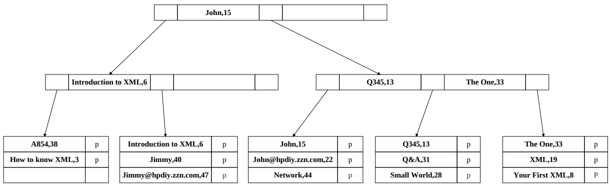


Figure 11: Value index tree.

4.3.1 Query pre-processing

After materialized views are created, a query should be transformed based on its corresponding materialized view. For the query as shown in Figure 4, its Xpath expression (traversing from node root) is transformed into a new one (traversing from node "Books") as shown in Figure 12.

```
SELECT extract(sys_nc_rowinfo$,'/Books
/Book[Title="How to know XML"]/Publisher
/Email=' John@hpdiy.zzn.com '')
FROM mv;
```

Figure 12: Query transformation based on a materialized view.

Before utilizing T-Bitmap and indexes to accelerate query processing, we must deal with parent-child axes and/or ancestor-descendant axes specified in the transformed query. We execute navigation or indexing according to different axes, and recursively check nodes in the document tree.

4.3.2 Navigation

For a parent-child axis, we use a navigation way to search nodes in the document tree. Here, T-Bitmap can be used to avoid unnecessary search during the navigation, since it provides the information whether result nodes are beneath the current processing node. For a query "/R/A/C" as shown in Figure 13, we can use an AND operator to determine whether node C is beneath the current processing node. First, for current node R, Query(11010)^R(11111)=Query(11010) indicates node C is beneath node R. Then, for the next node A1, Query(01010)^A1(01100)≠Query(01010) indicates node C cannot be beneath node A1, and we do not need to search the sub-tree rooted at node A1. Next, for node A2, Query(01010)^A2(01110)=Query(01010) indicates node C is beneath node A2. Finally, we recursively check nodes in the document tree until node C is found.

4.3.3 Indexing

For a query "/A//B· · ·" specifying an ancestor-descendant axis as shown in Figure 14, although we can also use T-Bitmap to search node B, the search based on a parent-child axis would go through a lot of unnecessary intermediate nodes. Therefore, we use indexes to directly search a descendant node, instead of using T-Bitmap. As mentioned in Section 4.2.3, we use the start value in the labeling code of the ancestor as the key to search the tag index tree of the descendant. Then, we check whether the descendant node is covered by the ancestor node; if yes, we fetch the descendant node and proceed to parse the query downward.

4.3.4 Subsequent processing

In this section, we investigate the processing for the query specifying a particular predicate. One is the query specifying values, and another is the twig query.

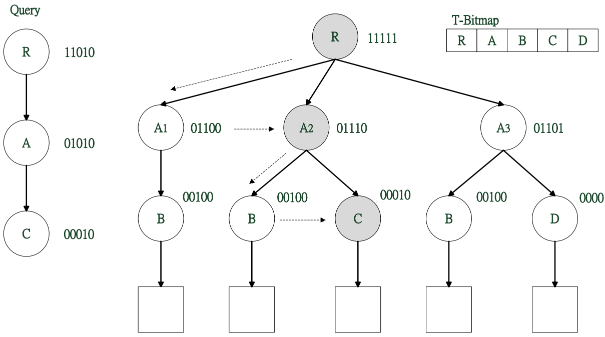


Figure 13: Navigation using T-Bitmap.

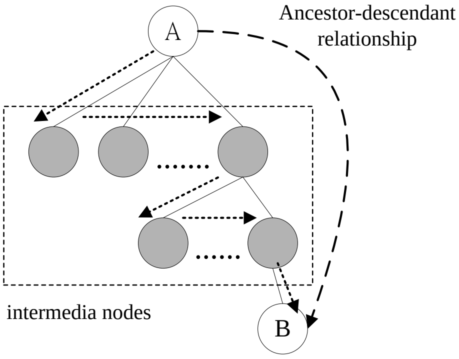


Figure 14: Unnecessary intermediate nodes.

4.3.4.1 Query specifying values

For a query "/Books/Book[Author = 'Jimmy']/Publisher = 'XML tech' ", two values are specified for filtering nodes in the document. As shown in Figure 15, we use the value index tree as mentioned in Section 4.2.3 to find out value nodes "Jimmy(12)", "XML tech1(7)", and "XML tech2(15)", and then put them into their corresponding queues, respectively. When processing node "Book" in the query, we fetch node "Book1", and find that node "Jimmy(12)" is not covered by node "Book1"; i.e., [2,9] < 12. Then, for the next node "Book2", although node "Book2" covers node "Jimmy(12)", node "XML tech1(7)" is not covered by node "Book2". Then, we choose the next value node "XML tech2(15)" and do the same inspection. Finally, we find out node "Book2" is the result node.

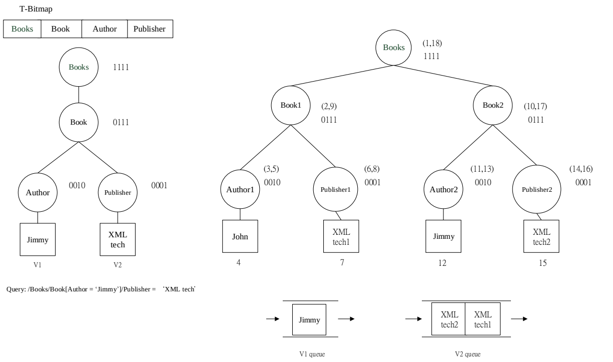


Figure 15: Value node processing.

4.3.4.2 Twig Query

Besides, for a query "/Book[Publisher]/Author" as shown in Figure 16(a), we can also process it in the same way as done in Section 4.3.2. As shown in Figure 16(b), two solutions "Book, Publisher, Author1, Jim" and "Book, Publisher, Author2, Jimmy" exist in the document. They can be found by 1) merging Path1 and Path2, and 2) merging Path1 and Path3, as shown in Figure 16(c).

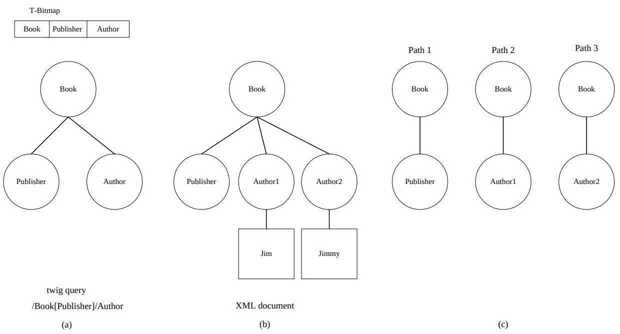


Figure 16: Twig query processing.

5 Experiments

In this section, four experiments are conducted to show the superiority of our approach proposed in this paper. These experiments are written in Java (JDK1.7) and conducted on an Intel Pentium4 3GHz CPU with 3G main memory in Windows 7. In the first experiment, we present the comparisons among different methods. In the second experiment, we investigate the effect of query types on different methods, especially on our method. In the third experiment, we use synthesis documents to analyze our method, and try to find some characteristics. In the last experiment, we address the space vs. time issue if multiple materialized views can be integrated together to save the space; in other words, more than one query would search from a materialized view.

5.1 Comparisons among different methods

In this experiment, we compare the search ways in different methods as shown in Figure 17. The first method is the original search way which is always from the root of a document tree. The second method was proposed by Godfrey et al. [1], which searches from the root of a materialized view. The last method is ours which also searches from the root of a materialized view, but with the aid of auxiliary data structures.

To fairly compare with the method proposed by Godfrey et al., an XML benchmark available on the XMark site is used in the experiment, which has data size 113.794MB and 1,513,518 nodes. Also, we follow the similar query types and comparison ways used in the experiments conducted by Godfrey et al. As shown in Table 2, there are twelve different types of queries tested in the experiment. To contrast with the searching ways as shown in Figure 17, the columns as shown in Table 3 are 1) query types, 2) searching time on the original tree, 3) view creation time, 4) searching time using Godfrey et al.' method, and 5) searching time using our method. The experimental results show that our method performs much better than the Godfrey et al.' method in all the queries, especially for long-path and twig queries Q3, Q7, Q8, Q9, and Q11. This is why our method uses the T-Bitmap and index structures to accelerate query processing.

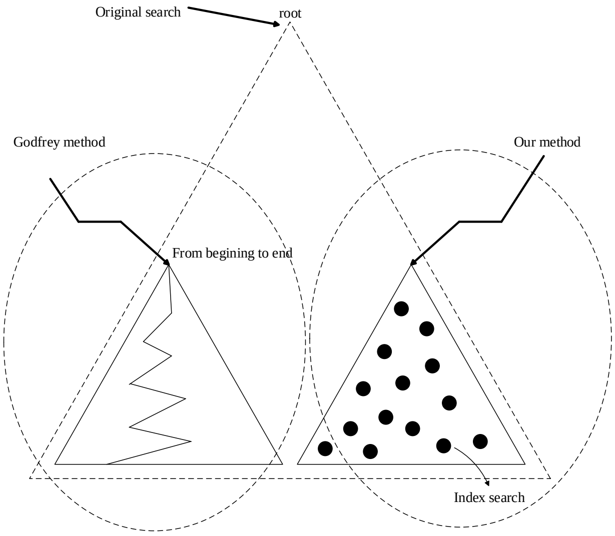


Figure 17: Search ways in different methods.

5.2 Effect of query types on different methods

In this experiment, we use the same XML benchmark as the first experiment, but different queries as shown in Table 4. For the searching axis, Q1 and Q2 are based on the parent-child axis, Q3 and Q4 on the ancestor-descendant axis, and the others on mixed axes. For the query types, Q1, Q3, Q5, Q6, and Q7 are path queries, whereas the others are twig queries. Furthermore, Q1, Q3, Q4, Q5, Q6, Q7, and Q10 are with value predicates.

In this experiment, as shown in Table 5, our method is still the best one among different methods in all the queries. Even taking the worst case Q8 as an example, our method is 206 times faster than the original way, and 88 times faster than the Godfrey et al.' method. The reason is, as shown in Table 6, the number of nodes visited for Q8 in our method is only 1/28 times of the original way, and is only 1/10 times of the Godfrey et al.' method.

For the path queries (i.e., Q1, Q3, Q5, Q6, and Q7), both execution time of the Godfrey et al.' method and ours is less than one second. For the twig queries (i.e., Q2, Q8, and Q9), they cost more execution time than the path queries since a large amount of nodes are visited. However, for the similar twig queries (i.e., Q4 and Q10), they do not cost much execution time since only a very small amount of nodes are required to visit. In summary, the advantages of our method are 1) when dealing with twig queries, we only need to check T-Bitmap and skip an entire sub-tree if not matched, and 2) when dealing with the queries with value predicates, we can use the value index tree to achieve efficient processing.

Table 2: Twelve kinds of queries.

Q1	/site/regions/europe/item/mailbox/mail
Q2	/site//item/mailbox/mail
Q3	/site//africa/item/description/parlist/listitem
Q4	/site//person/profile/interest[@category]
Q5	/site//person/profile[age]/interest[@category="category620"]
Q6	/site//person/profile[contains(age,"18")]/education
Q7	/site//open_auction[@id="open_auction5"]//date
Q8	/site//category/description[text]/parlist/listitem
Q9	/site//category/description[text/keyword]/parlist/listitem
Q10	/site/*/*/item/mailbox/mail
Q11	/site//*//africa/item/name
Q12	/site//item/mailbox[count(mail)]

Table 3: Comparisons among different methods.

Time(ms)				
Query	Original tree	$^{\%}$ View creation	* View non-indexed	$^{\#}$ View indexed
Q1	193718	30131	64980	5752
Q2	195146	32147	84574	20599
Q3	310315	31137	100130	593
Q4	185361	32890	99890	29436
Q5	193474	31147	147344	6865
Q6	191034	29702	65248	9106
Q7	203447	27112	132522	279
Q8	212511	28317	60353	232
Q9	241417	30169	64384	916
Q10	185357	31603	80727	19811
Q11	199274	28417	65059	320
Q12	209115	31731	99283	20687

$^{\%}$ View creation: Views created for both methods

* View non-indexed: Godfrey et al.' method

$^{\#}$ View indexed: Ours

5.3 Experiments on synthesis documents

In this experiment, six synthesis documents with different fanout are used to analyze our method for three different types of queries as shown in Table 7. Q1 is based on the parent-child axis, Q2 is based on the ancestor-descendant axis, and Q3 is a twig query with three predicates and based on mixed axes. As shown in Table 8, we find that the execution time of each query increases as the fanout increases. Moreover, regardless of the complexity in Q3, it still costs almost the same time as Q1 and Q2 using our method; i.e., its execution time would not increase significantly even if it is a twig query with three predicates. However, for Q3 using the original way and/or the Godfrey et al.' method, their execution time increases seriously as shown in Table 9. Especially for the Godfrey et al.' method, the execution time for fanout30 is 202 times slower than that for fanout5.

5.4 Experiments on space vs. time

In this experiment, we explore the space vs. time issue when multiple materialized views are integrated together. In order to achieve the premise that more than one query can search from a materialized view, we reuse seven queries (i.e., Q3, Q4, Q5, Q6, Q7, Q8 and Q10) as shown in Table 4. Then, we find that 1) Q3, Q6, and Q7 can search from the materialized view built based on Q3, 2) Q4 and Q10 can search from the materialized view built based on Q4, and 3) Q5 and Q8 can search from the materialized view built based on Q8. Thus, after the integration, we have three materialized views for these seven queries. The data space and execution time between no integration and integration are shown in Table 10. Taking the group (Q3, Q6, Q7) as an example, the overall data space is 3+1+3=7(KB) and the total execution time is 3+2+12=17(ms) if each query has its own materialized view; however, after the integration, the data space is only 3(KB), but the total execution time increases to 3+5+26=34(ms).

In order to explore the relationship between data space and execution time for these two strategies (i.e., no-integration and integration), we define two terms: 1) amount ratio for data space and 2) speed ratio for execution time as follows.

Table 4: Different kinds of queries.

Q1	/site/open_auctions/open_auction[@id="open_auction5"]/initial
Q2	/site/open_auctions/open_auction[annotation/author]/bidder/date
Q3	//site//open_auctions//open_auction[@id="open_auction0"]//current
Q4	//person[@id="person0"][creditcard]//watch
Q5	/site/regions//item[@id="item0"]//mail
Q6	//open_auction[@id="open_auction0"]/bidder/date
Q7	/site/open_auctions/open_auction[@id="open_auction0"]/../end
Q8	/site/regions//item[//text/bold]//location
Q9	//closed_auctions/closed_auction[//description/text]/seller
Q10	//people/person[@id="person0"][//business]/name

Table 5: Execution time.

Time(ms)				
Query	Original tree	View creation	View non-indexed	View indexed
Q1	31818	14920	193	8
Q2	68769	15731	37949	1883
Q3	29718	11787	140	3
Q4	30989	10056	103	3
Q5	27976	10705	119	3
Q6	30123	11723	48	2
Q7	32680	12135	139	12
Q8	4344137	222996	1853537	21111
Q9	1560306	246697	201444	16185
Q10	28425	10374	70	2

Table 6: Number of nodes visited.

Nodes			
Query	Original tree	View non-indexed	View indexed
Q1	48005	57	5
Q2	259604	87331	23794
Q3	24001	64	3
Q4	31502	24	8
Q5	34124	50	3
Q6	25960	8	5
Q7	24005	64	3
Q8	1204343	432651	43502
Q9	736217	102736	39003
Q10	25647	24	5

Table 7: Three kinds of queries.

Q1	/root/L1/R
Q2	//root//R
Q3	/root//L1[//R][//Q][//S]

$$amount-ratio_{strategy} = \frac{space(strategy)}{space(original)} \qquad (1)$$

$$speed-ratio_{strategy} = \frac{time(strategy)}{time(original)} \qquad (2)$$

where *space(strategy)* is the overall data space used in the no-integration or integration strategies, *time(strategy)* is the total execution time required in the no-integration or integration strategies, *space(original)* is the data space of the original document, and *time(original)* is the execution time on the original document.

Then, we can use these two terms to judge which strategy is better for the system performance as follows.

$$\frac{amount-ratio_{integration}}{amount-ratio_{no-integration}} < \frac{speed-ratio_{no-integration}}{speed-ratio_{integration}}$$
(3)

or $$\frac{space(integration)}{space(no-integration)} < \frac{time(no-integration)}{time(integration)} \qquad (4)$$

For Equation (4), the former term represents that the data space benefits for the integration strategy can be

Table 8: Comparisons for different fanout in our method.

Time(ms)			
	Q1	Q2	Q3
Fanout5	358	326	392
Fanout10	571	554	569
Fanout15	1065	969	1023
Fanout20	1514	1432	1486
Fanout25	2164	2023	2137
Fanout30	2701	2579	2658

Table 9: Comparisons for different fanout in different methods.

Q3 Time(ms)			
	Original tree	View non-indexed	View indexed
Fanout5	143618	419	392
Fanout10	278615	796	569
Fanout15	436672	2700	1023
Fanout20	655059	10815	1486
Fanout25	678379	38339	2137
Fanout30	879231	84779	2658

Table 10: Comparisons between no integration and integration.

No integration	Space(KB)	Time(ms)
Q3	3	3
Q4	1	3
Q5	2	3
Q6	1	2
Q7	3	12
Q8	56226	21111
Q10	1	2
Integration	Space(KB)	Time(ms)
(Q3, Q6, Q7)	3	3+5+26
(Q4, Q10)	1	3+2
(Q5, Q8)	56226	16491+21111

Table 11: Comparisons based on Equation (4).

	(Q3, Q6, Q7)	(Q4, Q10)	(Q5, Q8)
Former term	3/7=0.43	1/2=0.5	56226/56228=1
Latter term	17/34=0.5	5/5=1	21114/37602=0.56

gained (i.e., data space reduced) whereas the latter term represents that the execution time benefits for the no-integration strategy can be gained (i.e., execution time reduced). If Equation (4) with the equal weighting between space and time is true, the integration strategy should be adopted. According to the data taken from Table 10, we can calculate the former term and latter term in Equation (4), as shown in Table 11. From the statistical data, we find that the integration strategy is better for group (Q3, Q6, Q7) and group (Q4, Q10), but the no-integration strategy is better for group (Q5, Q8). Absolutely, different strategies can be adopted for different query groups at the same time to make the system performance in the best status.

6 Conclusions

In this paper, we employ some processing methods on XML documents to improve document retrieval. The goal is to reduce the number of visited nodes during tree traversing, thereby leading to faster query processing. To achieve this goal, we focus on the usage of database views. First, we build a materialized view from an original document for each query. Then, on each materialized view, we also build auxiliary structures such as T-Bitmap and indexes to further accelerate query processing. According to different axes specified in an Xpath expression, we have different techniques to handle them. Finally, through the experiments, we 1) compare the performances among different methods, 2) investigate the effect of query types on them, 3) use synthesis documents to analyze our method, and 4) address the space vs. time issue if materialized views are integrated together.

References

[1] Godfrey P, Gryz J, Hoppe A, et al. Query rewrites with views for XML in DB2. In: Ioannidis Y, Lee D, Ng R, eds. Proceedings of the IEEE 25th International Conference on Data Engineering, Shanghai, China, 2009. 1339-1350

[2] Murthy R, Banerjee S. XML schemas in Oracle XML DB. In: VLDB Endowment, eds. Proceedings of the 29th International Conference on Very Large Data Bases, Berlin, Germany, 2003. 1009-1018

[3] Halevy Y. Answering queries using views: a survey. Very Large Data Bases Journal, 2001, 10: 270-294

[4] Jayavel S, Jerry K, Eugene S, et al. Querying XML views of relational data. In: VLDB Endowment, eds. Proceedings of the 27th International Conference on Very Large Data Bases, Rome, Italy, 2001. 261-270

[5] Katsifodimos A, Manolescu I, Vassalos V. Materialized view selection for XQuery workloads. In: Fuxman A, eds. Proceedings of ACM SIGMOD International Conference on Management of Data, Scottsdale, Arizona, USA, 2012. 565-576

[6] Roantree M, Liu J. A heuristic approach to selecting views for materialization. Software: Practice and Experience, 2014, 44: 1157-1179

[7] Bonifati A, Goodfellow M, Manolescu I, et al. Algebraic incremental maintenance of XML views. ACM Transactions on Database Systems, 2013, 38: 14:1-14:45

[8] Wu X, Theodoratos D, Wang W H, et al. Optimizing XML queries: bitmapped materialized views vs. indexes. Information Systems, 2013, 38: 863-884

[9] Wu X, Theodoratos D, Kementsietsidis A. Configuring bitmap materialized views for optimizing XML queries. World Wide Web, 2015, 18: 607-632

[10] Gosain A, Sabharwal S, Gupta R. Architecture based materialized view evolution: a review. Procedia Computer Science, 2015, 48:256-262

[11] Gosain A, Sachdeva K. A systematic review on materialized view selection. In: Satapathy S et al., eds. Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications. Advances in Intelligent Systems and Computing, Vol. 515. Springer, Singapore, 2017. 663-671

[12] Bruno N, Koudas N, Srivastava D. Holistic twig joins: optimal XML pattern matching. In: Franklin M J, eds. Proceedings of ACM SIGMOD International Conference on Management of Data, Madison, WI, USA, 2002. 310-321

[13] Jiang H, Wang W, Lu H, et al. Holistic twig joins on indexed XML documents. In: VLDB Endowment, eds. Proceedings of the 29th International Conference on Very Large Data Bases, Berlin, Germany, 2003. 273-284

[14] Huang Y F, Wang S H. An efficient XML processing based on combining T-Bitmap and index techniques. In: Biaz S, Bellaachia A, eds. Proceedings of the IEEE International Symposium on Computers and Communication, Marrakech, Morocco, 2008. 858-863

[15] Hsu W C, Liao I E, Wu S Y, et al. An efficient XML indexing method based on path clustering. In: Alhajj R S, eds. Proceedings of the 20th IASTED International Conference on Modeling and Simulation, Banff, Alberta, Canada, 2009. 339-344

[16] Karthiga D, Gunasekaran S. Optimization of query processing in XML document using TAR and path based indexing. International Journal of Computer Science and Network Security, 2013, 13: 119-127

[17] Alghamdi N S, Rahayu W, Pardede E. Object-based semantic partitioning for XML twig query optimization. In: Barolli L et al., eds. Proceedings of the IEEE 27th International Conference on Advanced Information Networking and Applications, Barcelona, Spain, 2013. 846-853

[18] Thi Le D X, Maghaydah M, Orgun M A, et al. Optimization of XML queries by using semantics in XML schemas and the document structure. In: Lin X et al., eds. Proceedings of the 14th International Conference on Web Information Systems Engineering, Nanjing, China, 2013. 343-353

[19] Ordonez C. Optimization of linear recursive queries in SQL. IEEE Transactions on Knowledge and Data Engineering, 2010, 22: 264-277

[20] Subramaniam S, Haw S C. ME labeling: a robust hybrid scheme for dynamic update in XML databases. In: Ismail M, Ramli N, eds. Proceedings of the IEEE 2nd International Symposium on Telecommunication Technologies, Langkawi, Malaysia, 2014. 126-131

[21] Belgamwar H C, Dhore S M, Rathod P U, Deshmukh S S, Nandanwar G S. Review on storing and indexing XML documents upside down. International Journal for Engineering Applications and Technology, 2015, Manthan-15

[22] Ferro N, Silvello G. Descendants, ancestors, children and parent: a set-based approach to efficiently address XPath primitives. Information Processing & Management, 2016, 52:399-429

[23] Tudor N L. Query optimization against XML data. Studies in Informatics and Control, 2016, 25:173-180

[24] O'Connor M F, Roantree M. Desirable properties for XML update mechanisms. In: Proceedings of the 2010 EDBT/ICDT Workshops, Lausanne, Switzerland, 2010

[25] Lu J, Ling T W, Chan C Y, et al. From region encoding to extended Dewey: on efficient processing of XML twig pattern matching. In: VLDB Endowment, eds. Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, 2005. 193-204

Optimization, Modeling and Simulation of Microclimate and Energy Management of the Greenhouse by Modeling the Associated Heating and Cooling Systems and Implemented by a Fuzzy Logic Controller using Artificial Intelligence

Didi Faouzi
Materials and Renewable Energy Research Unit M.R.E.R.U University of Abou-bakr Belkaïd
B.P. 119, Tlemcen, Algeria
E-mail: didifouzi19@yahoo.com / didifouzi19@gmail.com

Nacereddine Bibi-Triki
Materials and Renewable Energy Research Unit M.R.E.R.U University of Abou-bakr Belkaïd
B.P. 119, Tlemcen, Algeria
E-mail: n_bibitriki@hotmail.fr

Bentchikou Mohamed
University of Yahia Fares, Médéa, Laboratory Director (LBMPT), Médéa Algeria
E-mail: bentchikou.mohamed@univ-medea.dz

Abderrahmane Abène
Euro-Mediterranean Institute of Environment and Renewable Energies, University of Valenciennes, France
E-mail: a.abene@yahoo.fr

Keywords: optimization, modelling, simulation, heater systems, intelligent control, fuzzy logic, greenhouse, cooling pads

Received: May 1, 2017

Agricultural greenhouse aims to create a favorable microclimate to the requirements of growth and development of culture, from the surrounding weather conditions, produce according to the cropping calendars fruits, vegetables and flower species out of season and widely available along the year. It is defined by its structural and functional architecture, the quality thermal, mechanical and optical of its wall, with its sealing level and the technical and technological accompanying. The greenhouse is a very confined environment, where multiple components are exchanged between key stakeholders and them factors are light, temperature and relative humidity. This state of thermal evolution is the level sealing of the cover of its physical characteristics to be transparent to solar, absorbent and reflective of infrared radiation emitted by the enclosure where the solar radiation trapping effect otherwise called "greenhouse effect" and its technical and technological means of air that accompany. New climate driving techniques have emerged, including the use of control devices from the classic to the use of artificial intelligence such as neural networks and / or fuzzy logic, etc... As a result, the greenhouse growers prefer these new technologies while optimizing the investment in the field to effectively meet the supply and demand of these fresh products cheaply and widely available throughout the year. In north Africa, greenhouse cultivation is undergoing significant development. To meet an increasingly competitive market and conditioned by increasingly stringent quality standards, "Greenhouse" production systems (heating and air-conditioning systems) Become considerably sophisticated and then disproportionately expensive. That is why locks who want to remain competitive must optimize their investment by controlling production conditions. The aim of our work is to model heating and air conditioning systems whose goal of heating and cooling the air inside our model and implemented in our application of climate control are due to the fuzzy logic that Has the role of optimizing the cost of the energy supplied using MATLAB software.

Povzetek: V Matlabu je razvit inteligentni sistem/dom za steklenjake v Alžiriji.

1 Introduction

Increased demand and requirement of fresh products consumers throughout the year, led parallelly to a rapid development of agricultural greenhouse, which is today modern and quite sophisticated.

Agricultural greenhouse aims to create a favorable microclimate to the requirements of the plant, necessary for its growth and development, from the surrounding weather conditions. it produces based cropping calendars, off-season products, cheap and widely available along the year. [8]

It is defined by its structural and functional architecture, the optical quality, thermal and mechanical coverage and the accompanying technical means. it is considered as a very confined environment where many components are exchanged between them, and in which the main factor involved in this medium is light, temperature and relative humidity [7-9]. to manage the greenhouse microclimate, greenhouse growers often use methods such as passive static ventilation (opening), shade screens, evaporative cooling etc ... and occasionally the active type. these methods are less expensive but more difficult to manage and optimize [11-14].

The first objective is to improve the thermal capacity of the greenhouse (greenhouse).

This is, to characterize the behavior of the complex system that is the greenhouse with its various compartments (ground, culture, cover, indoor and outdoor environment). To develop non-stationary mathematical models usable for simulation, optimization and the establishment of laws and control of simple and effective regulation.

These models must reproduce the essential properties of the mechanisms and interactions between different compartments. they must be both specific enough to obey the dynamic and real behavior of the greenhouse system, and fairly small to be easily adaptable to the phases of the simulation.

Good modulation instructions depending on the requirements of the plants to grow under shelter and outdoor climatic conditions, result in a more rational and efficient use of inputs and equip the best production performance.

The greenhouse climate is modified by artificial actuators, thus providing the best conditions in the immediate environment of energy costs and it requires a controller, which minimizes the power consumption while keeping the state variables as close as possible optimal harvest.

In this paper, or using fuzzy logic which is a powerful way to optimize and facilitate the global management of modern greenhouse, while providing through simulation interesting and encouraging which results in an optimization of favorable state variable values for the growth and development of protected cultivation [10-12-13].

The greenhouse originally conceived as an enclosure bounded by a wall transparent to solar radiation, as is the case with the conventional greenhouse, which is widely used in our country, amplifies certain parameters of the surrounding climate and shows conditions that are not favorable to Growth and the development of protected crops. This type of traditional greenhouses answered fairly well in the countries of the Mediterranean basin, is confronted to the intense nocturnal cooling, which sometimes results in the reversal of the internal temperatures and complications of overheating and hygrometric variations According to the seasons. Extreme variations in these parameters, often observed within shelters, constitute a nuisance that can hinder growth and crop development and, at best, penalize yield and product quality. To meet this equation of supply and demand, greenhouse systems have developed over time, thus imposing a great mastery of management and knowledge to achieve a better production [4].

This type of greenhouse, equipped and materialized by climate support; Are a means of transforming local conditions into an operational microclimate favorable to the growth and development of sheltered crops [1].

Technological progress has made considerable progress in the development of agricultural greenhouses. They become very sophisticated (heating systems, air conditioning, accessories and accompanying technical equipment, control computer etc.).

New climate control techniques have emerged, including the use of control devices, ranging from the classical to the application of artificial intelligence, now known as neural networks and / or fuzzy logic [2].

The air conditioning of modern greenhouses, allows to keep the crops under shelters under conditions compatible with the agronomic and economic objectives. Serrists opt for competitiveness.

They must optimize their investments, the cost of which is becoming more and more expensive [3].

The agricultural greenhouse can be profitable insofar as its structure is improved, the materials of the well chosen walls, depending on the nature and type of production, the technical installations and accompanying equipment must be judiciously defined.

Many equipment and accessories have appeared to regulate and control the state variables [6] such as temperature relative humidity, CO_2 concentration etc ...

At present the climatic computers of the greenhouses, solve the problems of regulation and ensure the observance of the climatic constraints Required by plants. From now on, the climate computer is a tool for dynamic production management, able to choose the most appropriate climate route, to meet the targets set, while minimizing inputs.

- Physiological aspect: This relatively complex and insufficiently developed field requires total management and extensive scientific and experimental treatment. This allows us to characterize the behavior of the plant during its evolution, from growth to final development; This allows us to establish an operational model
- Technical aspects: The greenhouse system is subject to a large number of data, decisions and actions to be taken on the immediate climatic environment of the plant (temperature, hygrometry, CO_2 enrichment, misting, etc.). The complexity of managing this environment requires an analytical, operational, numerical and computer-based approach to the system
- Socio-economic aspect: The social evolution, will be legitimated by a demanding and pressing demand of fresh products throughout the year; This state of affairs, involves all socio-economic operators [5], to be part of a scientific, technological and kitchen

dynamics. This dynamic demands high professionalism.

New techniques have emerged, including the use of climate control devices in a greenhouse (temperature, humidity, CO_2 concentration, etc.). Up to the exploitation of artificial intelligence That the neural networks and / or fuzzy logic.

The application of artificial intelligence in the industry has grown considerably, which is not the case in the field of agricultural greenhouses, where its application remains timid. It is from this state of affairs that we initiate research in this field and carry out a modeling based on meteorological data through MATLAB Simulink (Didi Faouzi, et al., 2016), to finally analyze the thermo - energy behavior of the greenhouse microclimate.

In our work we have modeled greenhouse systems (heating and cooling system and optimized the use of energy in a greenhouse by a defined intelligent controller such as fuzzy logic (FLC) using the method of Mamdani (Didi Faouzi , Et al, 2016).

2 Modeling of the greenhouse

Our model is parameterized (state variables), meaning that spatial heterogeneity is ignored and that the internal content of flows at the boundary of the system boundary is uniformly distributed [1].

The model consists of a set of differential equations formulated as follows [4]:

$$cap * \frac{\partial \mathrm{T}}{\partial \mathrm{t}} = \sum(puissance_{in} - puissance_{out}) \text{ [W]} \quad (1)$$

Where:

T: Is the temperature of the element under consideration (C °).

cap (J K-1): Is its thermal capacity and the incoming and outgoing thermal power are expressed in watts.

3 Modeling of heating and cooling systems

3.1 Modeling of heating systems

The modeled heating system consists of two independent heating pipes: one under the canopy (lower pipes) and the other in the canopy (upper pipes).

The heating system located under the canopy; Whose pipes are installed beneath the benches and on the sides of the walking paths must be dimensioned correctly, in order to contribute effectively to the internal climate of the greenhouse.

Due to the importance of this heating system, it was modeled with a proportional controller described by [8]:

$$\mathrm{Q}_{fournie\_tuyaux} = A_{\mathrm{sol}} * \left\| K_{p\_tuyaux} * (T_{ref} - T_{air}) \right\|_{0}^{250}$$
(2)

$\mathrm{Q}_{fournie\_tuyaux}$: : Is the thermal power entering the pipes [W].

K_{p_tuyaux} = 125 [W K^{-1} m^{-2}] : Is the constant of proportionality.

A_{sol}: Is the ground surface of the greenhouse [m2].

T_{air} [° C] : Is the parameter controlled and T_{ref} [° C] is the desired value of the controlled variable.

The term in parentheses $\| \quad \|_0^{250}$ is limited to a value between zero and 250 [W m-2]. This limitation is made using the "Saturation Simulink®" block, which indicates the maximum and minimum power, That the generator can supply per square meter.

The desired temperature T_{ref} is 20 ° C during the day period and 18 ° C during the night period.

3.2 Modeling of cooling systems

There are three common methods for cooling greenhouses: (1) natural ventilation (2) mechanical ventilation (3) mist cooling (misting). In our work mechanical ventilation is used [8]:

The control system selected is described by:

$$\mathrm{Q}_{ouverture} = A_{\mathrm{sol}} * 1 * 10^{-3} + A_{\mathrm{sol}} \left\| K_{p\_ouverture} (\mathrm{C\_H2O}_{air} - \mathrm{C\_H2O}_{ref}) \right\|_{0}^{1*10^{-3}} \quad (3)$$

Where :

$\mathrm{Q}_{H2O\_brouillard}$: Is the airflow through the opening [W].

A_{sol} [m^2] : Is the ground surface of the greenhouse [m2].

$K_{p\_brouillard}$ = 1 [m s^{-1}] : Is the minimum air flow.

$K_{p\_ouverture}$ = 0,5 [m^4 s^{-1} kg^{-1}] : Is the constant of proportionality.

$\mathrm{C\_H2O}_{air}$ [Kg m^{-3}] : Is the concentration of water vapor in air. And $\mathrm{C\_H2O}_{ref}$, is the desired moisture.

4 Organization of the model

Our model was developed using a form of organization according to the model proposed by (Jamisson M. Hill, 2006) [7]. The model of the plant used was set for Douglas fir planting. [7] The plants were started at 0.57 g dry weight and harvested at 1.67 g dry weight; A new growing season was recorded at each harvest.

So after we got a complete list of equations that can show the relationships between quantities, it does not tell me how these equations need to be solved numerically on the computer. And even less, how they should be expressed and organized as part of the global model software. Mathematical equations must be translated into computer code, which, when compiled and executed, translates the raw input data into meaningful data.

In our model, each block is defined by three sets of variable sets: inputs, state variables that describe the state of behavior and output that are directly dependent on that state. At each time step, the block may be called to execute the following commands [7]:

1. Initialization / reset of outputs and states.
2. Calculation of state derivatives.

3. Integrate the state derivatives to calculate the next state.
4. Calculates outputs according to the current state.

This methodology is robust and simple and can be applied to a wide range of processes, particularly those involving weighted parameters (mass / energy balance) or transfer functions, so it is very sensitive to crop models . The main impetus for using Simulink is that this methodology is built into the program structure, allowing the user to focus on the side of the model diagram.

```
% GUESS.m
% Core routine GUESS model
% GUESS (Greenhouse Use of Energy Seedling
     Simulation)
% GUESS is a dynamic lumped parameter process
     based model of a Douglas Fir
% seedling production greenhouse. GUESS models
     the dynamics of
% photosynthesis, and carbon allocation, climate
     control, and energy use.

t1 = cputime;
orgpath=path;
try
path(orgpath,genpath('Subfunctions'));
guessinit;             % User Defined Parameters
guessread;              % Load, and process weather data
guessmodel;              % Execute Simulink model
guessoutput;            % Display results
t2 = cputime;
   t3 = t2-t1;
catch err
path(orgpath);
rethrow(err);
end
path(orgpath);
fprintf('Model took %4.2f seconds to execute\n', t3);
```

5 Modeling of the fuzzy controller

The fuzzy logic control (FLC) is very robust, it is a flexible method that can be easily modified, and can use several inputs and outputs. It is much simpler than its predecessors (linear algebraic equations), and still very fast And less costly to implement. Then the controllers by fuzzy logic are very simple and easy to use. This method basically consists of three parts: an input, a processing part and an output part [2]:

1) The first part is an input: Indeed, it is represented in the membership functions.
2) The second part is a part of treatment, so-called rules of decisions.
3) The third and final part, is the exit step. The controller converts the results into specific values, which can be managed by another system.

One of the first questions to ask when designing a Fuzzy Logic Controller (FLC) is: What are my inputs and outputs? Once this issue is resolved, the next item to deal with is the range of inputs and outputs. When we speak of fuzzy sets, this range is called universal space [4].

An output value controlled by the fuzzy logic theoretical (FLC) is developed using the MATLAB Simulink software.

FLC is widely used when modeling the system implies that information is scarce and inaccurate, or when the system is described by a complex mathematical model. An example of this type of structure is the agricultural greenhouse and its variables such as the internal temperature. This state variable influences and activates the dynamic behavior of the greenhouse, it is non-linear. The internal temperature is one of the important and even main variables in the control and modeling of greenhouses.

In addition, a FLC is efficient to deal with continuous functions using the membership function (MF) and the IF-THEN rules. In general, a FLC contains four parts: fuzzifier, rules of decisions. , Fuzzy inference engine and defuzzify.

First, a set of input data is gathered and converted to a fuzzy set using fuzzy linguistic variables, fuzzy linguistic terms, and membership functions. This step is known as Fuzzification. Then, an inference is made on the basis of a set of rules. Finally, the resulting fuzzy output is matched to a net output using the MF (membership functions) in the defuzzification step.

Mamdani is method of fuzzy inference. Is the method we used and applied in our work to optimize the management of the microclimate of our agricultural greenhouse model. This method has fuzzy rules of form (IF-THEN) that have been used to implement the modeling of the fuzzy controller (FLC).

In many fuzzy applications, membership functions (MF) have been arbitrarily chosen as trapezoidal, triangular or Gaussian curves depending on the selected ranges.

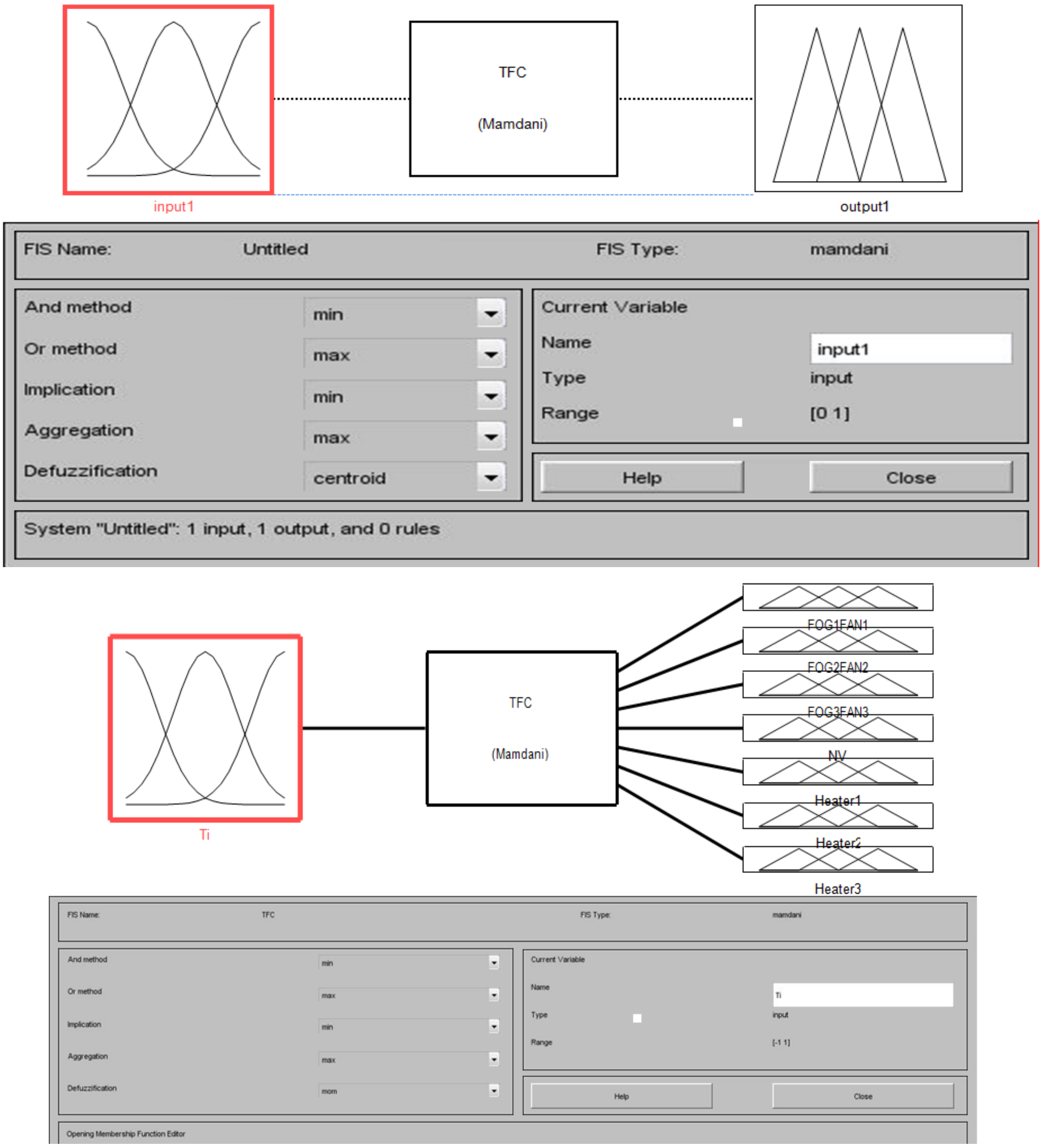
In our model, the sigmoid membership function is considered to define the input and triangular variables for the output variables (Figure 2).

All membership functions are defined on the normalized domain [-1, 1] in the discourse universe. With eight linguistic values, as shown in Figure 1.

This figure illustrates the fuzzy sets of membership functions that contain seven fuzzy sets. The linguistic values of the fuzzy sets used are:

Very cold (TVCOLD), COLD (TCOLD), Uncooked (TCOOL), OK (TGOOD), Low warm (TSH), Warm (TH), Very hot (HST) Designed on the basis of expert knowledge and in specialized literature.

We added to our model of the greenhouse an intelligent regulator using the fuzzy logic and we chose the Mamdani method with a single input, we started by first defining the input data and the outputs, and by the following has been attempted to link the membership functions in a logical manner in order to respond to the following steps. The characteristic variables of the system to be controlled and the instructions define the input variables of the fuzzy controller. The characteristic variables are in general the output



FIS Name: Untitled FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name: input1
Type: input
Range: [0 1]

Help Close

System "Untitled": 1 input, 1 output, and 0 rules

FIS Name: TFC FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: mom

Current Variable
Name: Ti
Type: input
Range: [-1 1]

Help Close

Opening Membership Function Editor

Figure 1: Creating Input and Output.

variables and, where appropriate, other measures the dynamic evolution of the process. The output variables of the fuzzy controller are the commands to apply to the process. The knowledge base consists of a database and a rule base. The database includes:

- Fuzzy sets associated with the input and output variables of the fuzzy controller,
- Scaling factors (input) (normalization) and output (demoralization).

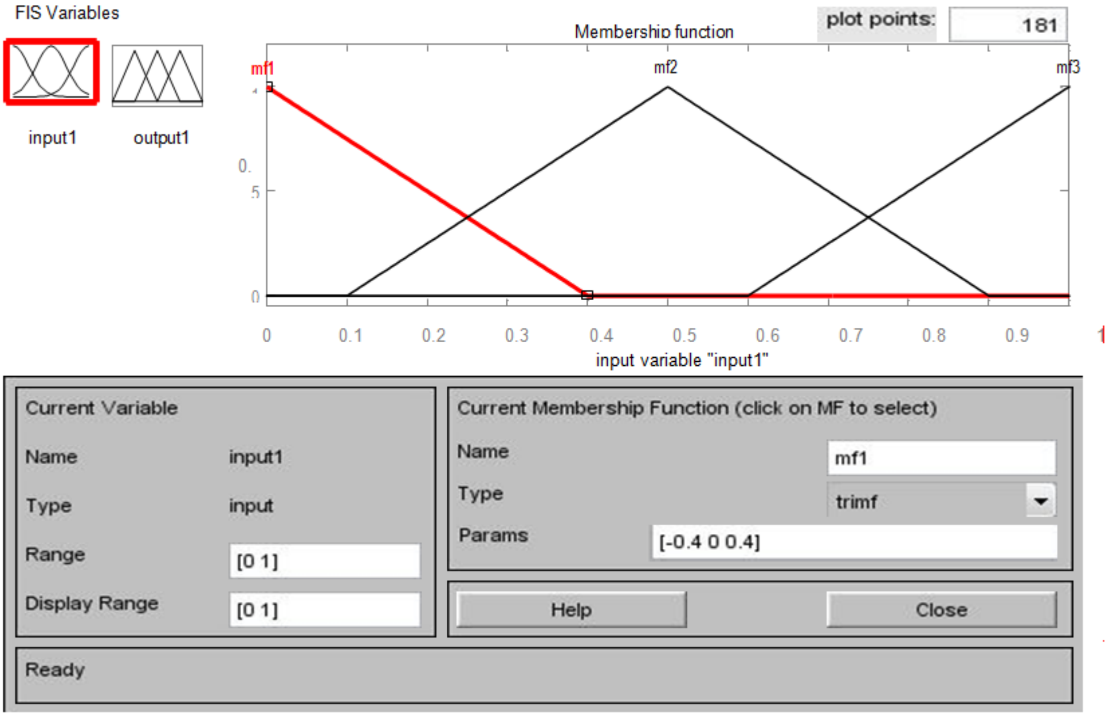
Then the range of variations (the fuzzy sets) and the membership functions for the input and the output were defined, and each part of the membership function was called by a significant name (Figure 3).

After defining the membership functions, the inference rules have been implemented in such a way as to achieve optimum control as desired, for example if the climate inside the greenhouse becomes lime the regulator will automatically Lowering the temperature by closing a heating system or opening a cooling system or by any other means and in order to keep the required instruction which will be translated by the following command [5]:

1. If (Ti is TVCOLD) then (FOG1FAN1 is OFF)(FOG2FAN2 is OFF)(FOG3FAN3 is OFF)(NV is OFF)(Heater1 is ON)(Heater2 is ON)(Heater3 is ON) (1)

2. If (Ti is TCOLD) then (FOG1FAN1 is OFF)(FOG2FAN2 is OFF)(FOG3FAN3 is OFF)(NV is OFF)(Heater1 is ON)(Heater2 is ON)(Heater3 is OFF) (1)

3. If (Ti is TCOOL) then (FOG1FAN1 is OFF)(FOG2FAN2 is OFF)(FOG3FAN3 is OFF)(NV is OFF)(Heater1 is ON)(Heater2 is OFF)(Heater3 is OFF) (1)



Current Variable

Name input1

Type input

Range [0 1]

Display Range [0 1]

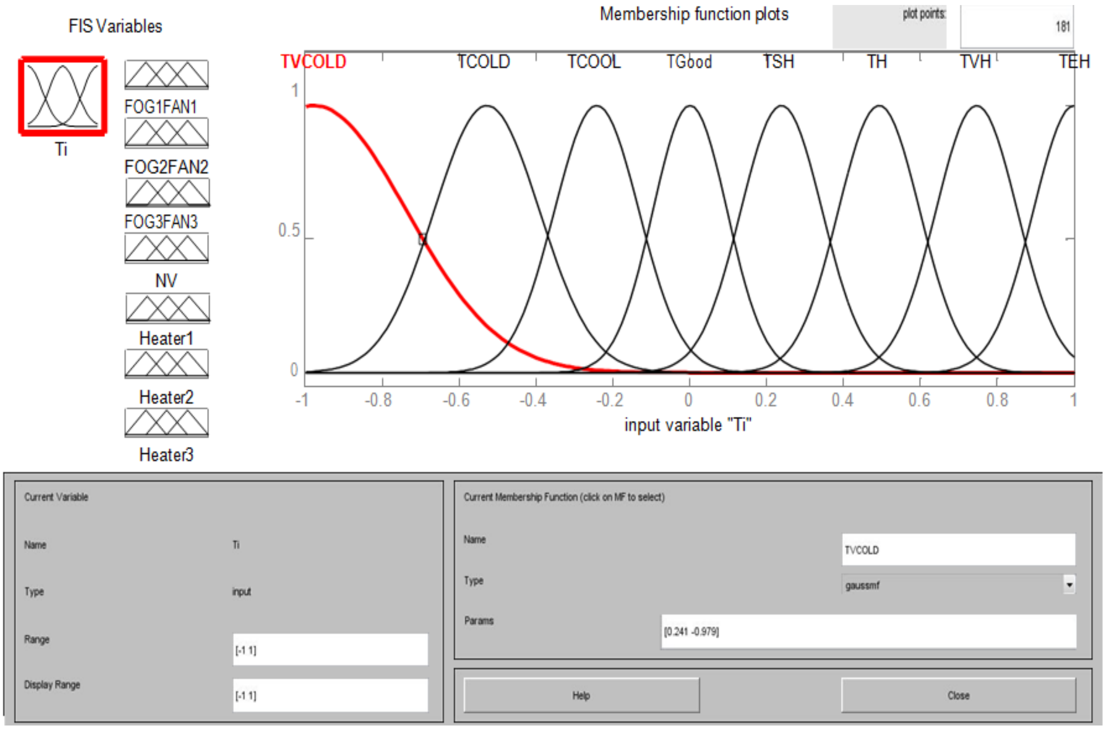
Current Membership Function (click on MF to select)

Name mf1

Type trimf

Params [-0.4 0 0.4]

Figure 2: Membership function of the command.



Current Variable

Name Ti

Type input

Range [-1 1]

Display Range [-1 1]

Current Membership Function (click on MF to select)

Name TVCOLD

Type gaussmf

Params [0.241 -0.979]

Figure 3. Membership functions for input and output variables.

4. If (Ti is TGood) then (FOG1FAN1 is OFF)(FOG2FAN2 is OFF)(FOG3FAN3 is OFF)(NV is OFF)(Heater1 is OFF)(Heater2 is OFF)(Heater3 is OFF) (1)

5. If (Ti is TSH) then (FOG1FAN1 is OFF)(FOG2FAN2 is OFF)(FOG3FAN3 is OFF)(NV is ON)(Heater1 is OFF)(Heater2 is OFF)(Heater3 is OFF) (1)

6. If (Ti is TH) then (FOG1FAN1 is ON)(FOG2FAN2 is OFF)(FOG3FAN3 is OFF)(NV is OFF)(Heater1 is OFF)(Heater2 is OFF)(Heater3 is OFF) (1)

7. If (Ti is TVH) then (FOG1FAN1 is ON)(FOG2FAN2 is ON)(FOG3FAN3 is OFF)(NV is OFF)(Heater1 is OFF)(Heater2 is OFF)(Heater3 is OFF) (1)

8. If (Ti is TEH) then (FOG1FAN1 is ON)(FOG2FAN2 is ON)(FOG3FAN3 is ON)(NV is OFF)(Heater1 is OFF)(Heater2 is OFF)(Heater3 is OFF) (1)

T_i : Indoor temperature.
TVCOLD : Temperature very cold.
TCOLD : Temperature cold.
TCOOL : Temperature is cool.
TSH : Temperature increases slowly.
TH : Temperature is hot.
TVH : Temperature is very hot.
HE : super hot temperature.

The explanation of the previous decision rules is as follows:

1) If the temperature (TVCOL) inside the greenhouse is very cold, then it is lower than the set temperature, the fuzzy controller sends a signal to automatically trigger all mechanical ventilation and ventilation systems and gives the order of " Open all heating systems".
2) If the temperature (TCOLD) inside the greenhouse is cold , then it is therefore somewhat below the set temperature then the fuzzy controller sends a signal to automatically trigger all mechanical cooling and ventilation systems and gives an order for " Open a single heating system".
3) If the temperature inside the greenhouse is cool therefore the controller gives the same previous order.
4) If the temperature (TSH) inside the greenhouse increases slowly, then the controller gives the order to stop all heating, cooling and mechanical ventilation systems and leave the operation of the ventilation natural.
5) If the temperature (TH) inside the greenhouse is hot, then it is slightly higher than the temperature of the set point, then the fuzzy controller sends a signal to automatically trigger all the cooling, heating and ventilation systems and gives Order to "open mechanical ventilation system" (forced).
6) If the temperature (HVT) inside the greenhouse is very hot, it is higher than the set point temperature, the fuzzy controller sends a signal to automatically trigger all heating systems and allow the operation of two mechanical ventilation systems.
7) If the temperature (TEH) inside the greenhouse is super hot, it is therefore superior to the set point temperature. Then the blurred controller sends a signal to automatically open all mechanical cooling and ventilation systems and stop " Operation of heating systems" and natural ventilation mechanical ventilation.

We save the file (.fis) to load it into the workspace and retrieve it in the Simulink Fuzzy block under the same name of the saved file.

The simulation of our system was done by MATLAB SIMULINK. The results of the MATLAB / SIMULINK software indicate the high capacity of the proposed technique to control the internal temperature of the greenhouse even in the event of a rapid change of atmospheric conditions. The modeling of the system Is defined in the form of this block diagram introduced in our Simulink shown in Figure (4 and 5). Its goal is to achieve the set temperature of 20 ° C required by the internal environment of our greenhouse. Indeed, by varying the ranges of inferences, the efficiency of the regulator has been increased around this set point. It would also be possible to modify the inference rules or the forms of the membership functions used.

For the validation of our model we used the full windows version of MATLAB Simulink R2012b (8.0.0.783), 64bit (win64). The simulation was performed on a TOSHIBA laptop. The latter is equipped with a 700 GB hard drive, and 5 GB of RAM. Simulink parts of the model were performed in "Accelerator" mode which first generated a compact representation of C code in the diagram.

Then compiled and executed. Simulink diagrams are obtained in the form of a sub-model integrated into blocks, thus decomposing the global model.

Our model is validated independently. Simulink diagrams resulting from the implementation and validation of our greenhouse model and its systems (heating, ventilation, cooling and misting, etc.) are shown in the figures below.

The informatics code (program) that gives the order to start the simulation for the figures below is as follows:

```
% guessread
% Didi Faouzi
% 1-1-2015 -- 5-24-2016
% Opens, reads, processes, and interpolates (per
minute basis) hourly
% weather data from a text file and converts it into
weather vectors for
% use by the Simulink model.  Also calculates vapor
pressures, and
% Weather data must be in tab, space, or comma
delimited text format
% And MUST include in the following columns in this
order
% date column
% time column
% Running time in days from first data point
% Outdoor temperature F or C
% Outdoor Relative Humidity % of VP Saturation
% Dew point temperature
% Wind Direction in degrees/radians E of N(azimuth)
% Wind Speed in mph or m/s
% Solar radiation in Langley s/hr or W/m2
%
% start cool = column with running time in hrs, min
or seconds
% start row = column at end of header
% NOTE:---------
% This M-File cannot be executed standalone, and
must be called
% from within the main GUESS module.  To run
GUESS, type GUESS in the
% command window prompt, and press <enter>.
```

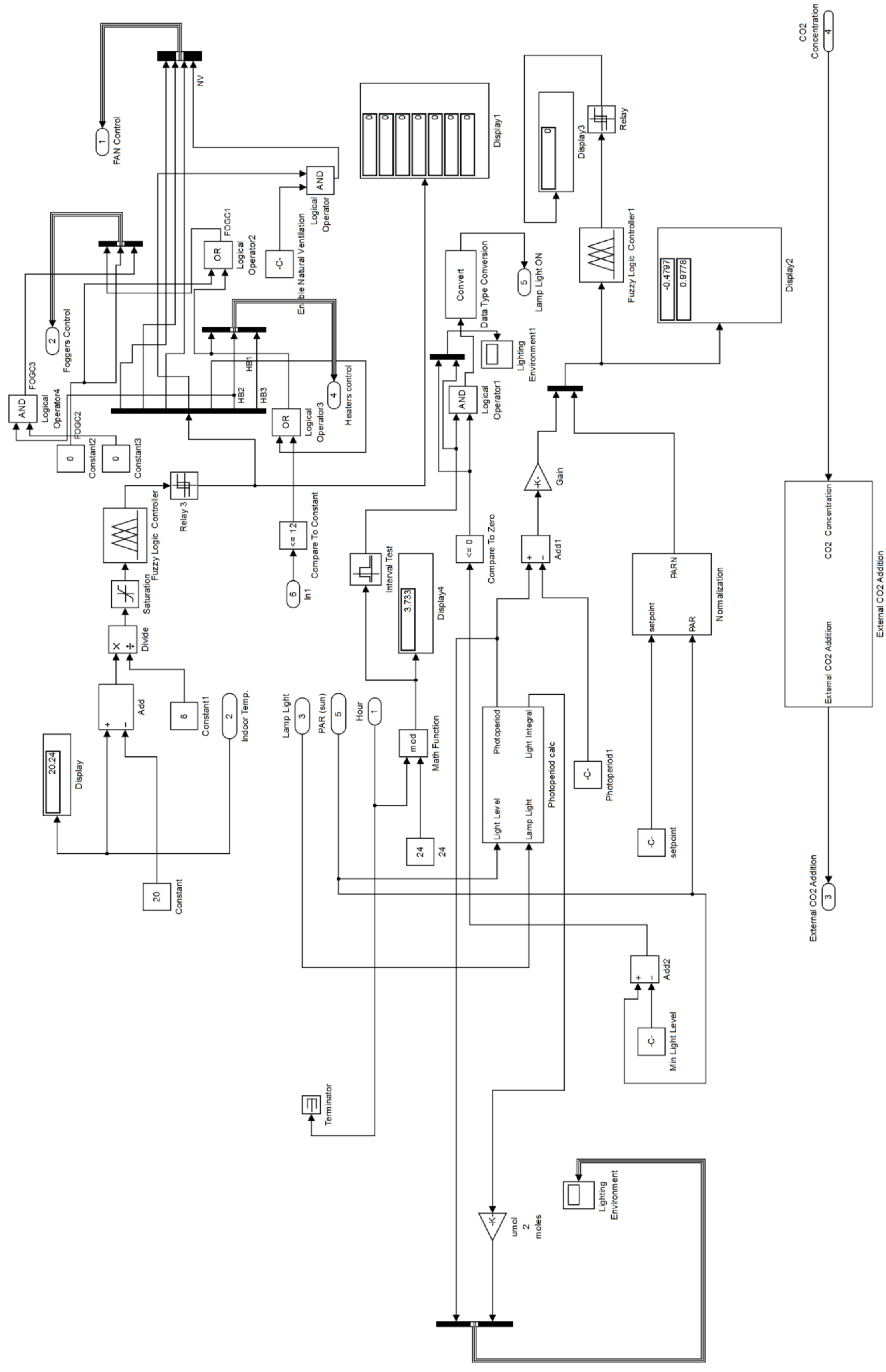


Figure 4: Schema Simulink represents our Fuzzy Logic Controller.

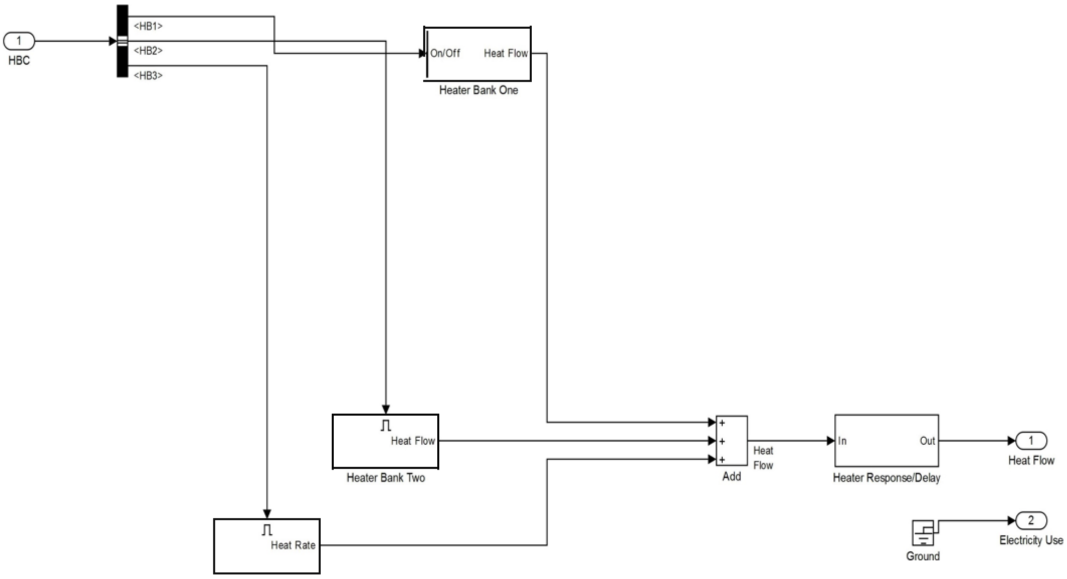


Figure 5. Simulink representation of the heating system model.

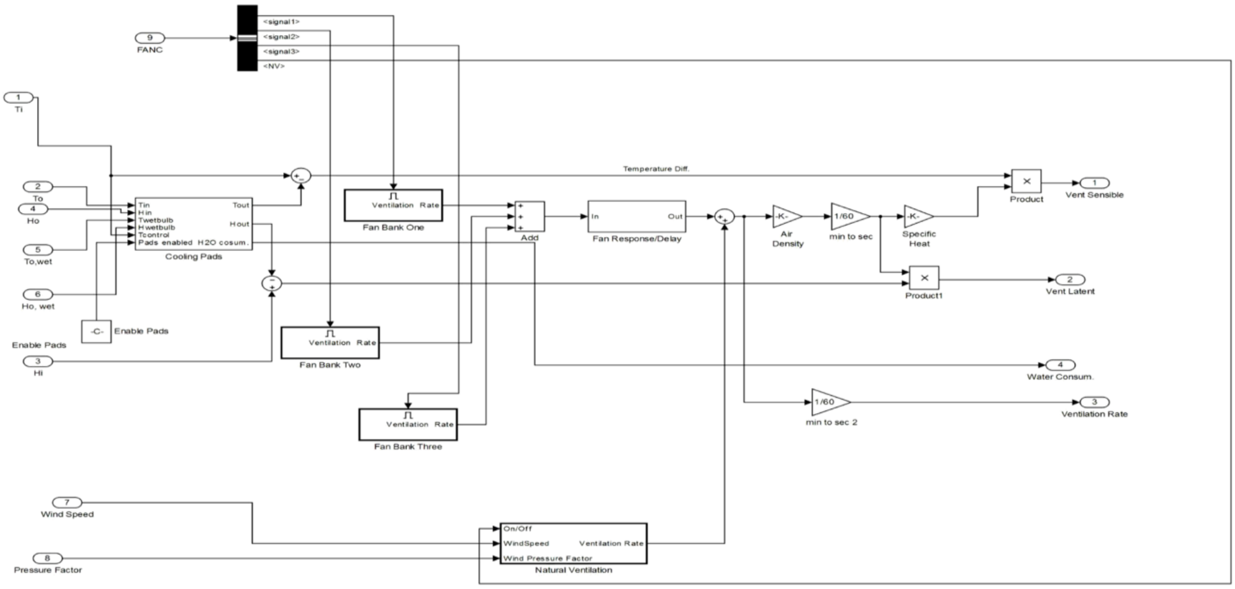


Figure 6: Simulink Representation of the Ventilation and Cooling Systems Model.

```
--
% last modified 4-30-06
%-----------
% Read data
%-----------
%try                    % Look for errors
tic                     % Start timer
warning off
fprintf('\n Weather Data File Processing\n')
fprintf('Reading File......');
W = dim read(Settings. file. file path, Settings. file. delimiter, ...
    Settings. file. start row, Settings. file. start cool);
switch Settings. sim. frequency
    case 'hourly'
        Hours = W(:,1);             % Convert to hourly data set
    case '15min'
        Quarters = W(:,1);
    %case '5min'
    %case '2min'
    case '1min'
        Minutes = W(:,1);
    otherwise
        error('\n Invalid time step \n');
    return
end
Date Raw     = W(:,2);              % Days elapsed since start of growing season
Temp Raw     = W(:,3);              % in deg C or F
Reel H Raw    = W(:,4);             % rel. Humidity %
%Dew P Raw     = W(:,5);            % Dew point Temp C or F; eliminate dew point
WindDirRaw1 = W(:,6);               % Wind Direction in degrees from South, azimuth
Wind Raw     = W(:,7);              % Wind Speed
```

```
Solar Raw     = W(:,8);              % Solar insulation in
Langley's or watts/m^2
fprintf('DONE\n')
% ----------------
% Unit Conversion
% ----------------
% ---------------
% Convert date into metric for ease of calculations
% ---------------
if units. temp == 'F'
   Temp_C  = conv Fto C(Temp Raw);
%    Dew P_C  = convFtoC(Dew P Raw);
else if units. temp == 'C'
   Temp_C = Temp Raw;
%    Dew P_C = convFtoC(Dew P Raw);
else
   error ('Invalid temperature units');
end
switch units. wind
   case 'mph'
      Wind Raw = convmph2mps (Wind Raw);
   case 'mps'
      % do nothing
   otherwise
      error ('Invalid wind speed units');
end
   %Wind Raw = Power Law Wind Conversion
switch units. solar
   case 'Wm2'
      % do nothing
   case 'ly'
      Solar Old = Solar Raw;
      Solar Raw = convLyhr2Wm2 (Solar Raw);
   case 'Btuf2h'
      Solar Raw = convBtuhrft2toWm2(Solar Raw);
   otherwise
      error('Invalid solar units');
end
switch units.rel H
   case '%'
      Reel H Raw = Reel H Raw / 100;
   case '1'% do nothing
   otherwise
      error('Invalid humidity unit');
end
switch units. wind dir
   case 'deg'
      Wind Dir Raw = WindDirRaw1 / 180*pi();
   case 'read' %do nothing
   otherwise
      error('Invalid wind direction unit');
end
%------------------------
% Check sampling rate
%------------------------
switch Settings. sim. frequency
   case 'hourly'           % 1hr. sampling rate
       Time Raw = Hours .*(60/Settings.sim.timestep);
   case '15min'            % 15 min. sampling rate
       Time Raw = Quarters
.*(15/Settings.sim.timestep);
```

```
   case '1min'             % 1 min. sampling rate
       Time Raw = Min;
end
EO Time = Time Raw(length(Time Raw));
% PAR Conversion
Light Raw     = conv2PAR('sunlight', Solar Raw,
'W/m2');
fprintf('Vapor Pressure, Wet bulb Calculations ......');
% Calculate humidity measurements: vapor pressure
or humidity ratios
% Calculate Saturation Vapor Pressures
Sat VP Raw   = p sat(Temp_C);              % Saturation
Vapor Pressures
% modify Sat VP to use Teten's formula (will run
faster)
% modify all to run with arrays; add *. and ./
VP Raw     = Reel H Raw .* Sat VP Raw;     % Vapor
Pressures
Hum Raw  = hum ratio (location. pressure, VP Raw);
% humidity ratio (mass H2O/mass air)
Wet Bulb Raw   = (wet bulb (Temp_C, Reel H Raw,
location. pressure))';
Wet Bulb VP    = p sat (Wet Bulb Raw);
Wet Bulb Hum Raw = hum ratio (location. pressure,
Wet Bulb VP);
fprintf('DONE\n');
fprintf('Wind Pressure Calculations ......');
% Calculate Wind Incidence Angle and Natural
Ventilation Pressure Coeff.
IncidenceAngleRaw1 = Wind Dir Raw - Greenhouse.
Azimuth;   % Inlet 1
IncidenceAngleRaw2 = Wind Dir Raw + Greenhouse.
Azimuth;   % Inlet 2
% Coefficient of Pressures
CpRaw2 = ones(length(IncidenceAngleRaw1),1);
CpRaw1 = ones(length(IncidenceAngleRaw1),1);
for X = 1:length(IncidenceAngleRaw1)
  CpRaw1(X)      =      calc      Wind      Press
Coeff(IncidenceAngleRaw1(X));
   CpRaw2(X)      =      calc      Wind      Press
 Coeff(IncidenceAngleRaw2(X));
end
% Ventilation Rate
 Wind     Factor     Raw     =     abs(CpRaw1     -
 CpRaw2)./(sort(abs(CpRaw1 - CpRaw2)));
U Nat V Raw = Ventilation. Natural. CD is charge .*
Wind Factor Raw .* Wind Raw;   % Ventilation Rate
% ----------------------------------
% Calculate Wind Pressures in Pascals
% Find Wind speed at eave height
Wind Raw = WS Convert(Wind Raw, Settings.
Climate. Wind. measured height, ...
        Ventilation.    Natural.    Height,    Settings.
Climate. Wind. exponent);
% Wind Pressure Inlet 1
WindPressure1  =  0.5*Air  D(Temp_C,  location.
pressure)*WindRaw.^2* CpRaw1;
% Wind Pressure Inlet 2
WindPressure2  =   0.5*Air  D(Temp_C,  location.
pressure)*WindRaw.^2* CpRaw2;
Wind    Pressure    Raw    =    WindPressure1    -
```

```
WindPressure2;  %Flow driven by pressure diff.
fprintf('DONE\n');
%
fprintf('Solar Radiation Calculations ......');

% --- Solar Altitude & Clearness Index ---
% Solar Time
Hrs    =    Time    Raw/(60/Settings.sim.timestep)    +
Settings. sim. time lag;
Clock time = mod(Hrs, 24);
Day Raw = 1+ Hrs ./ 24;
DayRaw2 = floor (Clock time ./ 24);
Hr Angle = Hour Angle Correct(Clock time, Day
Raw,...
   location. long - location. std long);
Declination = declination (Day Raw);
% Solar Altitude
Altitude = solar altitude (Declination, location.lat, Hr
Angle);
Altitude(Altitude < 0) = 0;    % can use -6 deg for civil
twilight
Altitude = (pi/180)* Altitude; % convert to radians
% Clearness Index
ET Solar = sin(Altitude) * Properties. Solar Constant;
% Calculate ET Radiation
K Index(1:length(Solar Raw),1)= 0.8;
%K Index = Solar Raw./ET Solar;    % clearness index
for i = 1:length(Time Raw)          % Correct for div
by/zero
   if Solar Raw(i) > ET Solar(i) % Can't have clearness
index > 1
      K  Index(i,1) = 1;
   else if ET Solar(i) == 0 && i > 2        % Maintain
previous value
      K Index(i,1) = K Index(i-1,1);       % throughout
the night
   else
      K   Index(i,1)   =   Solar   Raw(i)./ET   Solar(i);
%clearness index
   end
end
% Diffuse vs. Direct
[Diffuse Raw Direct Raw] = Split Beam(Solar Raw,
K Index); %Rad. splitting
fract Diff Raw = f Diffuse(K Index);
% f Direct  = 1 - f Diffuse;
fprintf('DONE\n');
%---------------------
% Long wave Sky Balance
%---------------------
fprintf('Long wave Calculations ......');
e_sky = e Sky B(Temp Raw, VP Raw, K Index);
T Sky Raw =  e_sky.^(1/4) .* Temp Raw;
fprintf('DONE\n');
fprintf('Interpolating ......');
%------------------
% Interpolate to 1 minute time step for simulation
%------------------
Settings. sim. Max time = EO Time;
Time     = ((1:EOTime).');
Date    = interp1(Time Raw, Date Raw, Time, 'linear');
Temp          = interp1(Time  Raw,  Temp_C,  Time,
method);
Reel H      = interp1(Time Raw, Reel H Raw, Time,
method);
%  Dew P      = interp1(Time Raw, Dew P_C, Time,
method);
Wind         = interp1(Time  Raw,  Wind  Raw,  Time,
method);
Solar        = interp1(Time  Raw,  Solar  Raw,  Time,
method);
U Nat V    = interp1(Time Raw, U Nat V Raw, Time,
method);
Wind Fact = interp1(Time Raw, Wind Factor Raw,
Time, method);
Diffuse   = interp1(Time Raw, Diffuse Raw, Time,
methods);
fract Diffuse = Diffuse ./Solar;
fract Diffuse(~is finit(fract Diffuse))= 0;
%Direct    = interp1(Time Raw, Direct Raw, Time,
method);
Sat VP     = interp1(Time Raw, Sat  V Raw, Time,
method);
Humidity  =  interp1(Time  Raw,  Hum  Raw,  Time,
method);
Wet bulbs      = interp1(Time Raw, Wet Bulb Raw,
Time, method);
WB Humidity  = interp1(Time Raw, Wet Bulb Hum
Raw, Time, method);
VP                = interp1(Time Raw, VP Raw, Time,
method);
Light          = interp1(Time Raw, Light Raw, Time,
method);   % Light in PAR
Angle             = interp1(Time  Raw,  Altitude,  Time,
method);
T Sky          = interp1(Time Raw, T Sky Raw, Time,
method);
%--- Wind and Natural Ventilation  ---
Wind Dir  = interp1(Time Raw, Wind Dir Raw, Time,
method);
Wind Pressure = interp1(Time Raw, Wind Pressure
Raw, Time, method);
fprintf('DONE\n');
fprintf('Generating Lookup Tables ......\n');
fprintf('Saturation Humidity ......');
%--- Saturation Vapor Look up Table ---
P Sat Look Up Table. T = 0:0.5:55;
P Sat Look Up Table. P = p sat(P Sat Look Up Table.
T);
%--- dP Sat Look Up Table ---
dP Sat Look Up Table. T = 0:0.5:55;
dP  Sat  Look  Up  Table.  P  =  dp  sat(dP  Sat  Look  Up
Table. T);
%--- Humidity at Wet Bulb Table ---
equiv WB Look Up Table. T = linspace(0,50,100);
equiv WB Look Up Table .H = linspace(0,1,100);
[T2  H]  =  mesh  grid(equiv  WB  Look  Up  Table.  T,
equiv WB Look Up Table .H);
equiv   WB   Look   Up   Table.   Values   =   equiv   WB
humidity(T2, H/100, location. pressure);
%--- Plant Stuff ---
Init Plant;
```

```
fprintf('DONE\n');
fprintf('Packing and Cleanup ......');
%--------------
% Create Weather structure in format wanted by
Simulink
% and organize into structure for ease of packaging
% Simulink Format
% Signal = [time step data];  use column vectors for
both.
% ---------------
Weather. Temp        = [Time Temp];
Weather. Reel H        = [Time Reel H];
Weather. Solar       = [Time Solar];
Weather. Wind        = [Time Wind];
Weather. Wind Fact     = [Time Wind Fact];
Weather. U Nat V       = [Time U Nat V];
Weather. Sat VP       = [Time Sat VP];
Weather. Humidity     = [Time Humidity];
Weather. Wet Bulb      = [Time Wet bulbs];
Weather. VP          = [Time VP];
Weather. Wind Dir      = [Time Wind Dir];
Weather. WB Humidity   = [Time WB Humidity];
Weather. Wind P       = [Time Wind Pressure];
Weather. Angle       = [Time Angle];
Weather. fract Diffuse = [Time fract Diffuse];
Weather. T Sky        = [Time T Sky];
%Weather. Direct      = [Time Direct];
%---------------
% Create timer object to override Simulink built-in
clock for output
% graphing and scoping
% -----------------
%-------------
% Cleanup
% Clear unneeded data
%-------------
clear W Date Raw Reel H Raw  Dew P Raw  T_C Cp
Raw VP Light Solar Old Temp Raw
clear Hours Quarters Minutes  Light Reel H Sat VP
Humidity Wet bulbs Wind
clear Wind Humidity Dew P Dew P_C Incidence
Angle  Hum Raw Cp Wet bulb
clear Wind Dir WindDirRaw1  VP Raw Sat VP Raw
VP Solar Old Wind Pressure Solar Raw
clear Incidence Angle Raw Wet Bulb Raw Temp  Wet
Bulb Hum Raw WB Humidity  Angle
clear Diffuse Raw Direct Raw  Direct CpRaw1
CpRaw2 Day Raw Altitude
clear DayRaw2 WB Humidity Time Raw Temp_C
Time Fill  K Index pack;
fprintf('DONE\n\n');
disp('Ready for simulation!'); to warning on
%catch
%    disp('Corrupt/Invalid Weather data file OR');
%     disp('guessread is not a standalone m-file, run
guessinit first');
%end.
% Plant Growth Diagram
figure(5)
% Height Graph, subplot 1
subplot(2,2,1)
```

```
plot(Guess Output. date, Guess Output. Plant.
Height);
x label('Day');
y label('Height (cm)');
subplot(2,2,2)
plot(Guess Output. date, Guess Output .Plant. Diam);
x label('Day');
y label('Stem Diameter (mm)');
subplot(2,2,3)
plot(Guess Output. date, Guess Output. Plant.
Biomass);
x label('Day');
y label('Total Dry Biomass (g)');
subplot(2,2,4)
plot(Guess Output. date, Guess Output. Plant. Crops);
x label('Day');
y label('Crops Harvested(#)');
h = top title('Plant Growth Characteristics');
set(h, 'Font Size', 14);
% Indoor Temperature Distribution
figure
plot(Guess Output. date, Guess Output. temp)
title('Temperatures');
x label('Day');
y label(axis);
legend('outdoor',' indoor');
figure
hold on
hits (Guess Output. temp(:,2));
title('Indoor Temperature Distribution');
x label(axis);
y label('freq.');
mean temp = mean(Guess Output. temp(:,2));
disp(sprint('Mean Indoor Temperature: %3.2f', mean
temp));
s tdev = std(Guess Output. temp(:,2));
disp(sprint('Standard Deviation Indoor Temperature:
%3.2f', stdev));
% Costs Diagram
figure(2)
plot(GuessOutput.date,[GuessOutput.Costs.total,
Guess Output.Costs.gas,...
 GuessOutput.Costs.electricity,
GuessOutput.Costs.water])
legend('Total', 'Natural Gas', 'Electricity', 'Water',
'Orientation', ... 'Horizontal', 'Location', 'Best')
h = title('Energy Costs');
x label('Day')
y label('Cost ($)')
set(h, 'Font Size', 14)
% Quantities Diagram
figure(4)
plot(GuessOutput.date,[GuessOutput.Quants.gas*Fue
l Converter,...
GuessOutput.Quants.electricity,
GuessOutput.Quants.water])
legend('Natural Gas(ft^3)', 'Electricity(kWh)',
'Water(gal)', 'Orientation', ...
     'Horizontal', 'Location', 'Best')
  x label('Day')
  y label('Energy Quantity')
```

```
h = title('Energy Quantities');
set(h, 'Font Size', 14).
```

6 Simulation results

6.1 Discussions on figures

The simulation results demonstrate the capabilities and performance of the intelligent controller, as well as the robustness of the fuzzy control. They illustrate in FIG (7) the stability of the variation of internal temperatures during the day and at night ranging from 15 ° C. to 25 ° C. and a relative humidity ranging from 50 % To 80% in the greenhouses of the two regions of Dar El Beida (wetland) and Biskra (arid zone). The temperature gradient due to the greenhouse effect recorded between the interior and the external environment was positive and varied from + 2 ° C to + 14 ° C. It should be noted that the external moisture content of the Dar El Beida zone, varying from 50% to 95%, was higher than that of Biskra, which varies from 35% to 80%. Temperature disturbances were recorded during the fifth season from

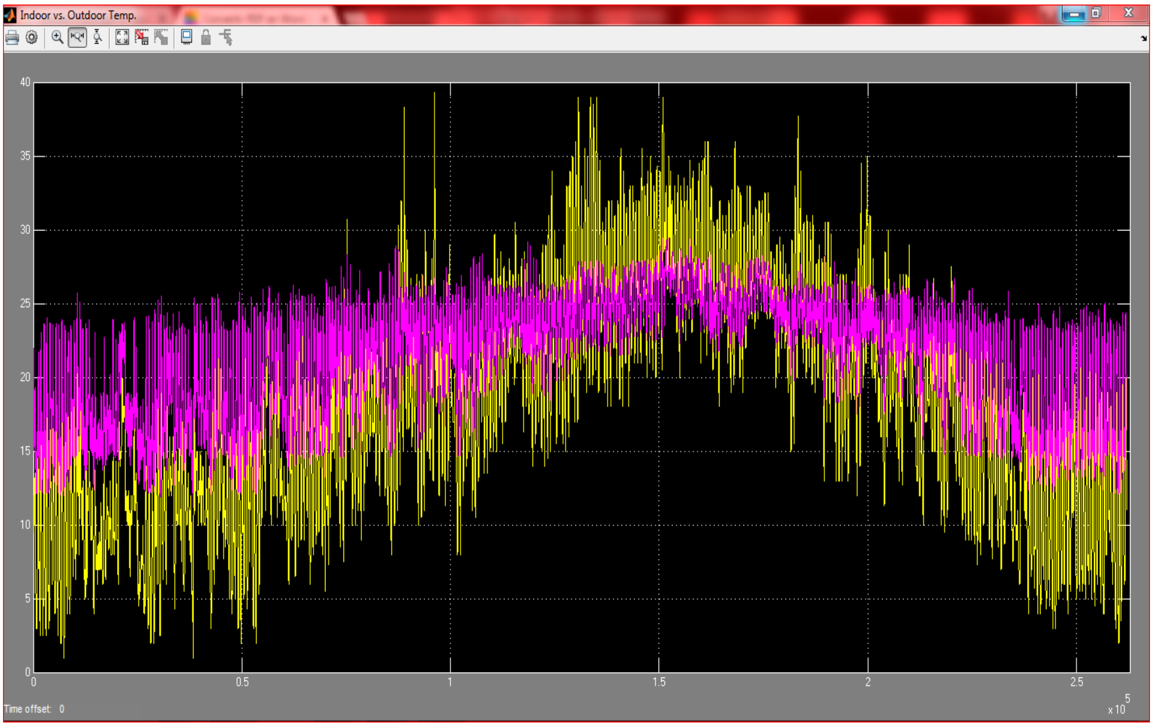


Figure 7. The evolution of the indoor and outdoor temperature in the form of scoop.

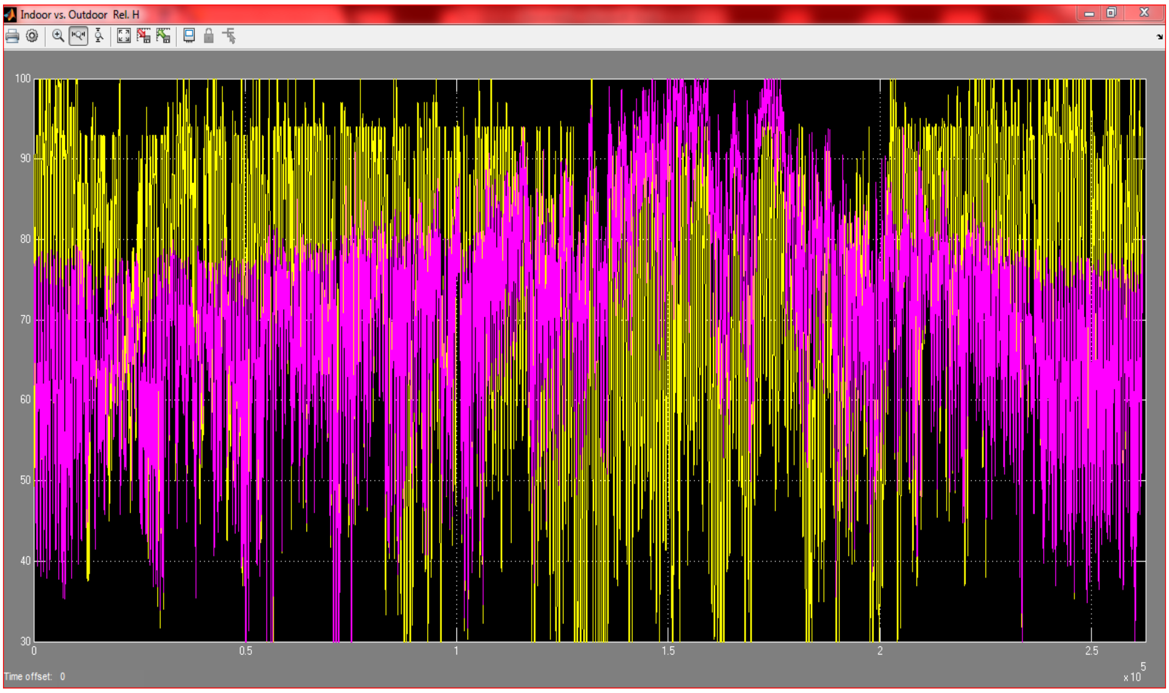


Figure 8. Evolution of indoor and outdoor moisture in the form of scoop.

mid-autumn to early winter, between 20 November and 20 January of the year. This was reflected in peaks in water temperatures and saturations due mainly to heat losses and signal noise caused by repeated cyclic activation and deactivation of heating and ventilation systems and equipment. On the whole, the application of the fuzzy method presents rather satisfactory results.

The need to improve the thermal insulation of the greenhouse is essential to reduce the heat losses that generally occur during this cold season in the greenhouse. Improving the greenhouse effect means stabilizing the microclimate and reducing the number of activations and deactivations of the accompanying systems that partly pose the problem of climate management.

In the figure (8), the moisture content within the greenhouse in the two wetlands and arid areas generally remains close to the optimum except in summer, where the rate of Humidity sometimes falls below the minimum threshold due to the ventilation necessary to ventilate, compensate and regulate the internal temperature.

7 Conclusion

The research work was initiated by a rich and interesting bibliographical study, which allowed us to discover this area of current affairs. A description of the types and models of agricultural greenhouses has been developed. Thermo hydric interactions, which occur within the greenhouse have been approached. The biophysical and physiological state of the plants through photosynthesis, respiration and evapotranspiration were exposed while taking into account their influences on the immediate environment and the mode of air conditioning. The models of regulation and climatic control have been approached from the use of conventional equipment to the use of artificial intelligence and / or fuzzy logic. Knowledge models and computer techniques have been established with a well-defined approach and hierarchy for optimal climate management of greenhouse systems, while naturally adopting the Mamdani method.

The aim is to develop a robust and robust air-conditioning control technology to deal with disturbances that may occur in the external and internal environments of the greenhouse system

Our assignments are:

1) To define the functioning of the complex system of the greenhouse with its various components (culture-immediate environment) using control models that optimally regulate the climate inside the greenhouse and reproduce the essential Of the properties, mechanisms and interactions between culture and its environment
2) To solve the problem of follow-up by the modeling, design and development of an intelligent controller able to regulate the system by following a desired reference trajectory and a hierarchy of control and regulation of the couplings between The different inputs and / or outputs

In this work we develop a climatic control by the Mamdani method based on multi-variable models in line for an adaptive neural model structure. The influence of random perturbations on system performance and optimization must be supported.

We have conceptualized, modeled, configured and developed an intelligent controller by fuzzy logic based on the Mamdani method. The simulation of the optimal climate management of the indoor environment of the agricultural greenhouse was carried out for two different regions, one wet, and this is Dar El Beida in ALGER; The other arid and concerns the region of BISKRA.

The simulation results highlight the capabilities and performance of the intelligent controller, as well as the robustness of the fuzzy control. They also illustrate the intelligibility of this control for optimal management during the four production seasons. We have also noted deficiencies that have arisen during the operation of the system during the fifth season, from mid-autumn to early winter (from 20 November to 20 January) .This is mainly due to heat losses and Signaling caused by repeated cyclic activation and deactivation of air conditioning equipment (heating and ventilation). This is because these state variables are highly correlated and influenced by the external environment, Solar radiation in the visible and infrared and the physiological response of the culture, a reaction quite natural and implicit.

The use of conventional controllers, the configuration of which no longer meets our expectations, nor the optimum climate regulation, but rather to some extent to the artificial intelligence technique, easily handled, to the serrists Characterized by its reliability and robustness in optimal climate management

The advantages of fuzzy control must be listed and treated without ignoring the conventional approaches of the classical automatic. All this involves the search for a compromise between complexity, human experience, systems mastery, model realism, configuration mode, and the robustness of the control method for predictive performance.

It should be noted that building robust models and practical, robust control methods are not limited by computer and / or digital tools, but rather by our knowledge and control of the dynamics of the ecosystem and its impact on the environment. Optimal climatic management of the greenhouse system. This implies that it is rather limited by the nature and quality of information on the ecosystem and its environment and by the faithful reproduction of this information for decision-making.

We remain optimistic in the near future, as regards the use of artificial intelligence technique and its fuzzy logic branch, which is indicated by:

1) Optimum climate control and regulation.
2) The operating efficiency of the energy reserve due to the greenhouse effect.

3) Optimal management of the energy input necessary for the operation of the accompanying systems and equipment.
4) Better productivity of sheltered crops.
5) A significant decrease in human intervention.

In the same way, it is necessary to point out the insufficiencies which may arise in the application of the fuzzy logic method and which are due at the moment to a misunderstanding or insufficient information of the ecosystems and their environments, Signal noise problems, the robustness of the fuzzy control, the peaks of the dominant variables; But for the moment all these interference problems and deficiencies can be solved by a realistic approach of the system.

As for the prospects, they are numerous. The application of Artificial Intelligence was reserved mainly in the fields of industry, robotics and especially in the Agri-food industry, whereas it can intervene in the management of several systems and processes not or can be tackled up to this day. We propose in our field to promote these techniques to multiply the harvest seasons and to make the exploitation of our agricultural land profitable.

8 References

[1] Didi Faouzi, N. Bibi Triki and A. Chermitti, 2016. Optimizing the greenhouse micro-climate management by the introduction of artificial intelligence using fuzzy logic. Int. J. Computer Eng. Technology, 7: 78-92 , Volume 7, Issue 3, May-June 2016, pp. 78–92, Article ID: IJCET_07_03_007.

[2] Didi Faouzi, N. Bibi-Triki, B. Draoui, A. Abène, 2016 , Modeling, Simulation and Optimization of-agricultural greenhouse microclimate by the application of-artificial intelligence and/or fuzzy logic, International journal of scientific & engineering research, volume 7, issue 8, august-2016 issn 2229-5518.

[3] Didi Faouzi, N. Bibi-Triki, B. Draoui, A. Abène, 2016 Comparison of modeling and simulation results management micro climate of the greenhouse by fuzzy logic between a wetland and arid region, International Journal of Multidisciplinary Research and Modern Education (IJMRME) ISSN (Online): 2454 - 6119, Volume II, Issue II, 2016 .

[4] Didi Faouzi, N. Bibi-Triki, B. Draoui, A. Abène, 2016, Modeling and Simulation of Fuzzy Logic Controller for the purpose of Optimizing the Management Micro Climate of the Agricultural Greenhouse, MAYFEB Journal of Agricultural Science Vol 2 (2016).

[5] Didi Faouzi , N. Bibi-Triki , B. Draoui , A. Abène, 2017**,** Greenhouse Environmental Control Using Optimized, Modeled and Simulated Fuzzy Logic Controller Technique in MATLAB SIMULINK, Computer Technology and Application 7 (2016) 273-286, doi: 10.17265/1934-7332/2016.06.002.

[6] Didi Faouzi, N. Bibi-Triki, B. Draoui, A. Abène. Dated 10th March 2017**,** The Optimal Management of the Micro Climate of the Agricultural Greenhouse through the Modeling of a Fuzzy Logic Controller, International Knowledge Press, Journal of Global Agriculture and Ecology (JOGAE), 7(1): 1-15, 2017, ISSN: 2454-4205, Ref. No. IKP/JOGAE/17/0102.

[7] Jamisson M.Hill, dynamic modeling of tree growth and energy use in a nursery greenhouse using MTLAB and Simulink, Cornell University, 7/31/2006.

[8] Marco Binotto (May 2014), "Greenhouse climate model an aid to estimate the influence of supplemental lighting on greenhouse climate", School of Science and Engineering at Reykjavík University.

[9] Babuska, R., & Mamdani, E. H. (2008). Fuzzy Control. http://www.scholarpedia.org/article/Fuzzy_control.

[10] [10] Breemen, A. v., & Vries, T. d. (2000). An Agent-Based Framework for Designing Multi-Controller Systems. Paper presented at the Proceedings of the Fifth International Conference on The Practical Applications of Intelligent Agents and Multi-Agent Technology, Manchester, U.K.

[11] [11] Tan, V., Yoo, D.-S., & Yi, M.-J. (2008a). A Multiagent-System Framework for Hierarchical Control and Monitoring of Complex Process Control Systems. Paper presented at the Proceedings of the 11th Pacific Rim International Conference on Multi-Agents: Intelligent Agents and Multi-Agent Systems, Hanoi, Vietnam.

[12] [12] Choi, J., Oh, S., & Horowitz, R. (2009). Distributed learning and cooperative control for multi-agent systems. Automatica, 45(12), 2802-2814. doi: 10.1016/j.automatica.2009.09.025.

[13] [13] McArthur, S. D. J., Davidson, E. M., Catterson, V. M., Dimeas, A. L., Hatziargyriou, N. D., Ponci, F., & Funabashi, T. (2007). Multi-Agent Systems for Power Engineering Applications-Part I: Concepts, Approaches, and Technical Challenges. *22*, 1743- 1752.doi: 10.1109/tpwrs.2007.908471.

[14] [14] Kelly, I. D., & Keating, D. A. (1998). Faster learning of control parameters through sharing experiences of autonomous mobile robots. International Journal of Systems Science 29(7), 783-793.

Improving Visual Vocabularies: a More Discriminative, Representative and Compact Bag of Visual Words

Leonardo Chang, Airel Pérez-Suárez and José Hernández-Palancar
Advanced Technologies Application Center, 7A #21406, Siboney, Playa, Havana, Cuba, C.P. 12220
E-mail: {lchang,asuarez,jpalancar}@cenatav.co.cu

Miguel Arias-Estrada and L. Enrique Sucar
Instituto Nacional de Astrofísica, Óptica y Electrónica
Luis Enrique Erro No. 1, Sta. María Tonantzintla, Puebla, México, C.P. 72840
E-mail: {ariasmo,esucar}@inaoep.mx

Keywords: bag of visual words improvement, bag of features, visual vocabulary, visual codebook, object categorization, object class recognition, object classification

Received: July 21, 2016

In this paper, we introduce three properties and their corresponding quantitative evaluation measures to assess the ability of a visual word to represent and discriminate an object class, in the context of the BoW approach. Also, based on these properties, we propose a methodology for reducing the size of the visual vocabulary, retaining those visual words that best describe an object class. Reducing the vocabulary will provide a more reliable and compact image representation. Our proposal does not depend on the quantization method used for building the set of visual words, the feature descriptor or the weighting scheme used, which makes our approach suitable to any visual vocabulary. Throughout the experiments we show that using only the most discriminative and representative visual words obtained by our proposed methodology improves the classification performance; the best results obtained with our proposed method are statistically superior to those obtained with the entire vocabularies. In the Caltech-101 dataset, average best results outperformed the baseline by a 4.6% and 4.8% in mean classification accuracy using SVM and KNN, respectively. In the Pascal VOC 2006 dataset there was a 1.6% and 4.7% improvement for SVM and KNN, respectively. Furthermore, these accuracy improvements were always obtained with more compact representations. Vocabularies 10 times smaller always obtained better accuracy results than the baseline vocabularies in the Caltech-101 dataset, and in the 93.75% of the experiments on the Pascal VOC dataset.

Povzetek: S pomočjo rudarjenja podatkov se prispevek ukvarja z iskanjem besed za razločevanje razredov objektov.

1 Introduction

One of the most widely used approaches for representing images for object categorization is the Bag of Words (BoW) approach [5]. BoW-based methods have obtained remarkable results in recent years and they even obtained the best results for several classes in the recent PASCAL Visual Object Classes Challenge on object classification [8]. The key idea of BoW approaches is to discretize the entire space of local features (e.g., SIFT [22]) extracted from a training set at interest points or densely sampled in the image. With this aim, clustering is performed over the set of features extracted from a training set in order to identify features that are visually equivalent. Each cluster is interpreted as a visual word, and all clusters form a so-called visual vocabulary. Later, in order to represent an unseen image, each feature extracted from the image is assigned to a visual word of the visual vocabulary; from which a histogram of occurrences of each visual word in the image is obtained, as illustrated in Figure 1.

One of the main limitations of the BoW approach is that the visual vocabulary is built using features that belong to both the object and the background. This implies that the noise extracted from the image background is also considered as part of the object class description. Also, in the BoW representation, every visual word is used, regardless of its low representativeness or discriminative power. These elements may limit the quality of further classification processes. In addition, there is no consensus about which is the optimal way for building the visual vocabulary, i.e., the clustering algorithm used, the number of clusters (visual words) that best describe the object classes, etc. When dealing with relatively small vocabularies, clustering can be executed several times and the best performing vocabulary can be selected through a validation phase. However, this becomes intractable for large image collections.

In this paper, we propose three properties to assess the ability of a visual word to represent and discriminate an

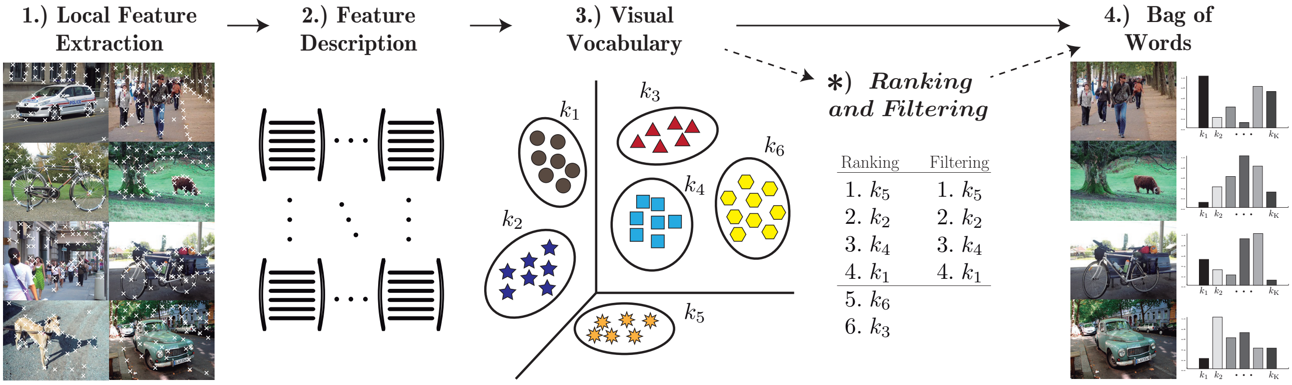


Figure 1: Classical BoW approach overview (steps 1 to 4). First, regions/points of interest are automatically detected and local descriptors over those regions/points are computed (step 1 and 2). Later in step 3, the descriptors are quantized into visual words to form the visual vocabulary. Finally, in step 4, the occurrences in the image of each specific word in the vocabulary for constructing the BoW feature are found. In this work, we propose to introduce step ($*$) in order to use only the most discriminative and representative visual words from the visual vocabulary in the BoW representation.

object class in the context of the BoW approach. We define three measures in order to quantitatively evaluate each of these properties. The visual words that best represent a class, best generalize over intra-class variability and best differentiate between object classes will obtain the highest scores for these measures. A methodology for reducing the size of the visual vocabulary based on these properties is also proposed. Our proposal does not depend on the clustering method used to create the visual vocabulary, the descriptor used (e.g., SIFT, SURF, etc.) or the weighting scheme used (e.g., *tf*, *tf-idf*, etc.) Therefore, it can be applied to any visual vocabulary to improve its representativeness, since it does not build a new visual vocabulary, it rather finds the best visual words of a given visual vocabulary.

Experiments conducted on the Caltech-101 [10] and Pascal VOC 2006 [9] datasets, in a classification task, demonstrate the improvement introduced by the proposed method. Tested with different vocabulary sizes, different interest points extraction and description methods, and different weighting schemas, the classification accuracies achieved using the entire vocabulary were always statistically inferior to those achieved by several of the vocabularies obtained by filtering the baseline vocabulary, using our proposed vocabulary size reducing methodology. Moreover, the best results were obtained with as few as the 13.4% and 17.2%, in average, of the baseline visual words for the Caltech-101 and Pascal VOC 2006 datasets, respectively. Compared with a state-of-the-art mutual information based method for feature selection our proposal obtains superior classification accuracy results for the highest compression rates and comparable results for the other filtering sizes.

The paper is organized as follows: Section 2 gives an overview on related works for building more discriminative and representative visual vocabularies. Section 3 introduces the proposed properties and measures for the evaluation of the representativeness and distinctiveness of visual words. The performance of our proposed method on two data sets and a discussion of the obtained results are presented in Section 4. Finally, Section 5 concludes the paper with a summary of our findings and a discussion of future work.

2 Related work

Several methods have been proposed in the literature to overcome the limitations of the BoW approach [25]. These include part generative models and frameworks that use geometric correspondence [30, 23], works that deal with the quantization artifacts introduced while assigning features to visual words [15, 11], techniques that explore different features and descriptors [24, 12], among many others. In this section, we briefly review some recent methods aimed to build more discriminative and representative visual vocabularies, which are more related to our work.

Kersorn and Poslad [17] presented a framework to improve the quality of visual words by constructing visual words from representative keypoints. Also, domain specific non-informative visual words are detected using two main characteristics for non-informative visual words: high document frequency and a small statistical association with all the concepts in the collection. In addition, the vector space model of visual words is restructured with respect to a structural ontology model in order to solve visual synonym and polysemy problems.

Zhang *et al.* [29] proposed to obtain a visual vocabulary comprised of descriptive visual words and descriptive visual phrases as the visual correspondences to text words and phrases. Authors state that a descriptive visual element can be composed by the visual words and their combinations and that these combinations are effective in representing certain visual objects or scenes. Therefore, they define visual phrases as frequently co-occurring visual word pairs.

Lopez-Sastre *et al.* [21] presented a method for building a more discriminative visual vocabulary by taking into account the class labels of images. The authors proposed a cluster precision criterion based on class labels in order to obtain class representative visual words through a Reciprocal Nearest Neighbors clustering algorithm. Also, they introduced an adaptive threshold refinement scheme aimed to increase vocabulary compactness.

Liu [19] builds a visual vocabulary based on a Gaussian Mixed Model (GMM). After K-Means clusters are obtained, GMM is then used to model the distribution of each cluster. Each GMM will be used as a visual word of the visual vocabulary. Also, a soft assignment schema for the bag of words is proposed based on the soft assignment of image features to each GMM visual word.

Liu and Shah [20] exploit mutual information maximization techniques to learn a compact set of visual words and to determine the size of the codebook. In their proposal two codebook entries are merged if they have comparable distributions. In addition, spatio-temporal pyramid matching is used to exploit temporal information in videos.

Most popular visual descriptors are histograms of image measurements. It has been shown that with histogram features, the Histogram Intersection Kernel (HIK) is more effective than the Euclidean distance in supervised learning tasks. Based on this assumption, Wu *et al.* [28] proposed a histogram kernel k-means algorithm which use HIK in an unsupervised manner to improve the generation of visual codebooks.

In [4], in order to use low level features extracted from images to create higher level features, Chandra *et al.* proposed a hierarchical feature learning framework that uses a Naive Bayes clustering algorithm. First, SIFT features over a dense grid are quantized using K-Means to obtain the first level symbol image. Later, features from the current level are clustered using a Naive Bayes-based clustering and quantized to get the symbol image at the next level. Bag of words representations can be computed using the symbol image at any level of the hierarchy.

Jiu *et al.* [16], motivated by obtaining a visual vocabulary highly correlated to the recognition problem, proposed a supervised method for joint visual vocabulary creation and class learning, which uses the class labels of the training set to learn the visual words. In order to achieve that, they proposed two different learning algorithms, one based on error backpropagation and the other one based on cluster label reassignment.

In [27], the authors propose a hierarchical visual word mergence framework based on graph-embedding. Given a predefined large set of visual words, their goal is to hierarchically merge them into a small number of visual words, such that the lower dimensional image representation obtained based on these new words can maximally maintain classification performance.

Zhang *et al.* [31] proposed a supervised Mutual Information (MI) based feature selection method. This algorithm uses MI between each dimension of the image descriptor and the image class label to compute the dimension importance. Finally, using the highest importance values, they reduce the image representation size. This method achieve higher accuracy and less computational cost than feature compression methods such as product quantization [14] and BPBC [13].

In our work, similarly to [17, 21, 16], we also use the class labels of images. However, we do not use the class labels to create a new visual vocabulary but for scoring the set of visual words, according to their distinctiveness and representativeness for each class. It is important to emphasize that our proposal does not depend on the algorithm used for building the set of visual words, the descriptor used nor the weighting scheme used. The previously mentioned characteristics make our approach suitable to any visual vocabulary since it does not build a new visual vocabulary, it rather finds the best visual words of a given visual vocabulary. In fact, our proposal could directly complement all the above discussed methods, by ranking their resulting vocabularies according to the distinctiveness and representativeness of the obtained visual words, although is out of the scope of this paper to explore it.

3 Proposed method

Visual vocabularies are commonly comprised by a lot of noisy visual words due to intra-class variability and the inclusion of features from the background during the vocabulary building process, among others. Later, for image representation every visual word is used, which may lead to an error-prone image representation.

In order to improve image representations, we introduce three properties and their corresponding quantitative evaluations to assess the ability of a visual word to represent and discriminate an object class in the context of the BoW approach. We also propose a methodology, based on these properties, for reducing the size of the visual vocabulary, discarding those visual words that worst describe an object class (i.e., noisy visual words). Reducing the vocabulary in such a manner will allow to have a more reliable and compact image representation.

We would like to emphasize that all the measures proposed in this section are used during the training phase; therefore, we can use all the knowledge about the data that is available during this phase.

3.1 Inter-class representativeness measure

A visual word could be comprised of features from different object classes, representing visual concepts or parts of objects common to those different classes. These common parts or concepts do not have necessarily to be equally represented inside the visual word because, even when similar, object classes should also have attributes that differentiate them. Therefore, we can say that, in order to represent an object class the best, a property that a visual word must satisfy is to have a high representativeness of this class. In order to measure the representativeness of a class c_j in visual word k, the measure $\mathcal{M}_1$ is proposed:

$$\mathcal{M}_1(k, c_j) = \frac{f_{k,c_j}}{n_k}, \tag{1}$$

where f_{k,c_j} represents the number of features of class c_j in visual word k and n_k is the total number of features in visual word k.

Figure 2 shows $\mathcal{M}_1$ values for two example visual words. In Figure 2 a) the 'blue' class has a very high value of $\mathcal{M}_1$ because most of the features in the visual word belong to the ○ class, being the opposite for the classes □ and △ that are poorly represented in the visual word. Figure 2 b) shows an example visual word where every class is nearly equally represented, therefore every class have similar $\mathcal{M}_1$ values.

3.2 Intra-class representativeness measure

A visual word could be comprised of features from different objects, many of them probably belonging to the same object class. Even when different, object instances from the same class should share several visual concepts. Taking this into account, we can state that a visual word best describes a specific object class while more balanced are the features from that object class comprising the visual word, with respect to the number of different training objects belonging to that class. Therefore, we could say that, in order to represent an object class the best, a property that a visual word must satisfy is to have a high generalization or intra-class representativeness over this class.

To measure the intra-class representativeness of a visual word k for a given object category c_j, the measure μ is proposed:

$$\mu(k, c_j) = \frac{1}{O_{c_j}} \sum_{m=1}^{O_{c_j}} \left| \frac{o_{m,k,c_j}}{f_{k,c_j}} - \frac{1}{O_{c_j}} \right|, \tag{2}$$

where O_{c_j} is the number of objects (images) of class c_j in the training set. o_{m,k,c_j} is the number of features extracted from object m of class c_j in visual word k, and f_{k,c_j} is the number of features of class c_j that belong to visual word k. The term $1/O_{c_j}$ represents the ideal ratio of features of class c_j that guarantees the best balance, i.e., the case where each object of class c_j is equally represented in visual word k.

The measure μ evaluates how much a given class deviates from its ideal value of intra-class variability balance. In order to make this value comparable with other classes and visual words, μ could be normalized using its maximum possible value, which is $\frac{2 \cdot O_{c_j} - 2}{O_{c_j}^2}$.

Taking into account that μ takes its maximum value in the worst case of intra-class representativeness, the measure $\mathcal{M}_2$ is defined to take its maximum value in the case of ideal intra-class variability balance and to be normalized by $\max(\mu(k, c_j))$:

$$\mathcal{M}_2(k, c_j) = 1 - \frac{O_{c_j}}{2 \cdot (O_{c_j} - 1)} \sum_{m=1}^{O_{c_j}} \left| \frac{o_{m,k,c_j}}{f_{k,c_j}} - \frac{1}{O_{c_j}} \right|. \tag{3}$$

Figure 3 shows the values of $\mathcal{M}_2$ on two example visual words. In Figure 3 a), the number of features from the different images of the ○ class in the visual word is well balanced, i.e., the visual word generalizes well over intra-class variability for the ○ class, hence this class presents a high $\mathcal{M}_2$ value. In contrast, in Figure 3 b) only one image from the ○ class is well represented by the visual word. As the visual word represents a visual characteristic only present in one image, it is not able to well represent intra-class variability, therefore, the ○ class will have a low value of $\mathcal{M}_2$ in this visual word.

3.3 Inter-class distinctiveness measure

$\mathcal{M}_1$ and $\mathcal{M}_2$ provide, under different perspectives, a quantitative evaluation of the ability of a visual word to describe a given class. However, we should not build a vocabulary just by selecting those visual words that best represent each object class, because this fact does not directly imply that the more representative words will be able to differentiate well one class from another, as a visual vocabulary is expected to do. Therefore, we can state that, in order to be used as part of a visual vocabulary, a desired property of a visual word is that it should have high values of $\mathcal{M}_1(k, c_j)$ and $\mathcal{M}_2(k, c_j)$ (represents well the object class), while having low values of $\mathcal{M}_1(k, \{c_j\}^C)$ and $\mathcal{M}_2(k, \{c_j\}^C)$ (misrepresents the rest of the classes), i.e., it must have high discriminative power.

In order to quantify the distinctiveness of a visual word for a given class, the measure $\mathcal{M}_3$ is proposed. $\mathcal{M}_3$ expresses how much the object class that is best represented by visual word k is separated from the other classes in the $\mathcal{M}_1$ and $\mathcal{M}_2$ rankings.

Let $\Theta_{\mathcal{M}}(K, c_j)$ be the set of values of a given measure $\mathcal{M}$ for the set of visual words $K = \{k_1, k_2, ..., k_N\}$ and the object class c_j, sorted in descending order of the value of $\mathcal{M}$. Let $\Phi(k, c_j)$ be the position of visual word $k \in K$ in $\Theta_{\mathcal{M}}(K, c_j)$. Let $P_k = \min_{c_j \in C}(\Phi(k, c_j))$ be the best position of visual word k in the set of all object classes $C = \{c_1, c_2, ..., c_Q\}$. Let $c_k = \arg\min_{c_j \in C}(\Phi(k, c_j))$ be the object class where k has position P_k. Then, the inter-class distinctiveness (measure $\mathcal{M}_3$), of a given visual word k for a given measure $\mathcal{M}$, is defined as:

$$\mathcal{M}_3(k, \mathcal{M}) = \frac{1}{(|C| - 1)(|K| - 1)} \sum_{c_j \neq c_k} (\Phi(k, c_j) - P_k). \tag{4}$$

In Figure 4, the $\mathcal{M}_3$ measure is calculated for two visual words (i.e., k_2 and k_5) of a six visual words and three

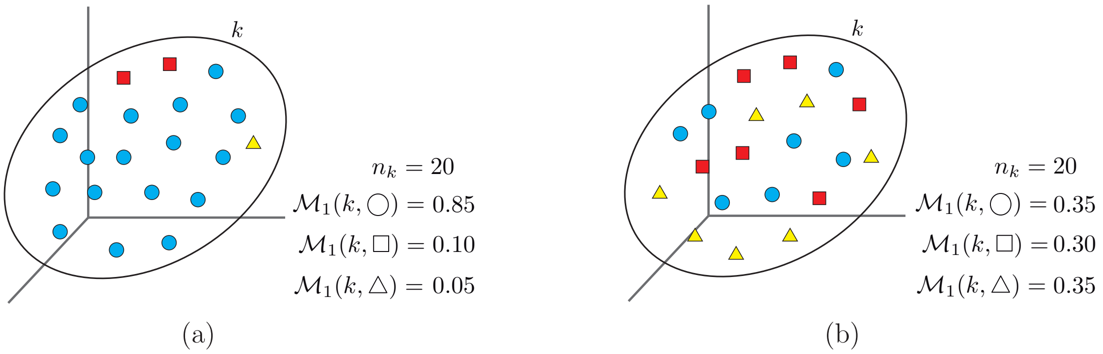

k
$n_k = 20$
$\mathcal{M}_1(k, \bigcirc) = 0.85$
$\mathcal{M}_1(k, \square) = 0.10$
$\mathcal{M}_1(k, \triangle) = 0.05$
(a)
k
$n_k = 20$
$\mathcal{M}_1(k, \bigcirc) = 0.35$
$\mathcal{M}_1(k, \square) = 0.30$
$\mathcal{M}_1(k, \triangle) = 0.35$
(b)

Figure 2: *(best seen in color.)* Examples of $\mathcal{M}_1$ measure values for a) a visual word with a well-defined representative class (○ class with high $\mathcal{M}_1$ value, □ and △ classes with low $\mathcal{M}_1$ values) and b) a visual word without any highly representative class (○, □ and △ classes have low and very similar $\mathcal{M}_1$ values).

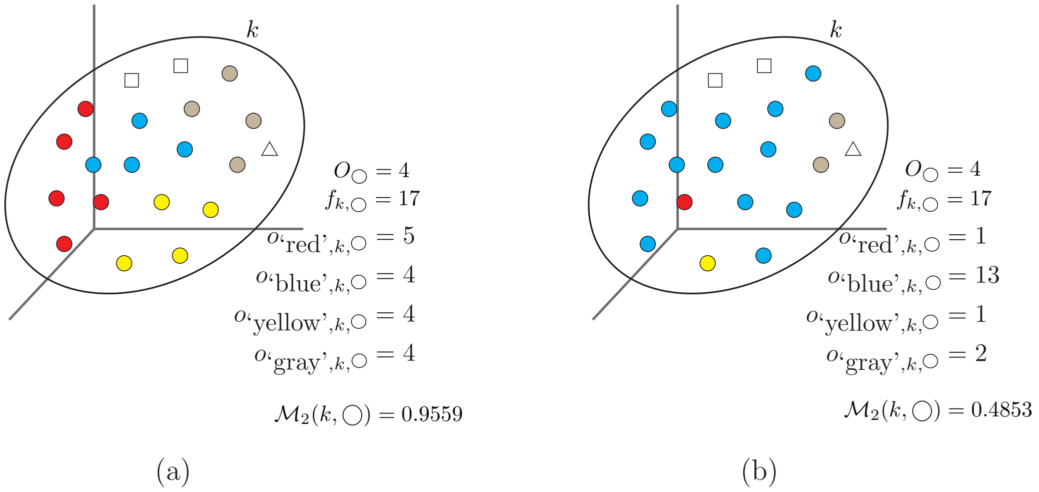

k
$O_{\bigcirc} = 4$
$f_{k,\bigcirc} = 17$
$o_{\text{'red'},k,\bigcirc} = 5$
$o_{\text{'blue'},k,\bigcirc} = 4$
$o_{\text{'yellow'},k,\bigcirc} = 4$
$o_{\text{'gray'},k,\bigcirc} = 4$
$\mathcal{M}_2(k, \bigcirc) = 0.9559$
(a)
k
$O_{\bigcirc} = 4$
$f_{k,\bigcirc} = 17$
$o_{\text{'red'},k,\bigcirc} = 1$
$o_{\text{'blue'},k,\bigcirc} = 13$
$o_{\text{'yellow'},k,\bigcirc} = 1$
$o_{\text{'gray'},k,\bigcirc} = 2$
$\mathcal{M}_2(k, \bigcirc) = 0.4853$
(b)

Figure 3: *(best seen in color.)* Examples of $\mathcal{M}_2$ measure values for the ○ class in a) a visual word where there is a good balance between the number of features of different images of the ○ class (high $\mathcal{M}_2$ value), and in b) the opposite case where only one image for the ○ class is predominantly represented in the visual word (low $\mathcal{M}_2$ value). In the figure, different fill colors of each feature in the visual word represent features extracted from different object images of the same class.

classes example. Visual word k_2 is among the top items of the representativeness ranking for every class in the example. Despite this, k_2 has low discriminative power because describing well several classes makes harder the process of differentiate one class from another. In contrast, visual word k_5 is highly discriminative because it describes well only one class.

3.4 On ranking and reducing the size of visual vocabularies

The proposed measures, provide a quantitative evaluation of the representativeness and distinctiveness of the visual words in a vocabulary for each class. The visual words that best represent a class, best generalize over intra-class variability and best differentiate between object classes will obtain the highest scores for these measures. In this section, we present a methodology for ranking and reducing the size of the visual vocabularies, towards more reliable and compact image representations.

Let $\Theta^{\mathcal{M}_1}(K)$ and $\Theta^{\mathcal{M}_2}(K)$ be the rankings of vocabulary K, using measures $\mathcal{M}_3(K, \mathcal{M}_1)$ and $\mathcal{M}_3(K, \mathcal{M}_2)$, respectively. $\Theta^{\mathcal{M}_1}(K)$ and $\Theta^{\mathcal{M}_2}(K)$ provide a ranking of the vocabulary based on the distinctiveness of visual words according to inter-class and intra-class variability, respectively.

In order to find a consensus, $\Theta(K)$, between both rankings $\Theta^{\mathcal{M}_1}(K)$ and $\Theta^{\mathcal{M}_2}(K)$ a consensus-based voting method can be used; in our case, we decided to use the Borda Count algorithm [7] although any other can be used as well. The Borda Count algorithm obtains a final ranking from multiple rankings over the same set. Given $|K|$ visual words, a visual word receive $|K|$ points for a first preference, $|K| - 1$ points for a second preference, $|K| - 2$ for a third, and so on for each ranking independently. Later, individual values for each visual word are added and a final ranking obtained.

From this final ranking a reduced vocabulary can be obtained by selecting the first N visual words. As pointed in [20], the size of the vocabulary affects the performance and there is a vocabulary size which can achieve maximal accuracy, which depends on the dataset, the number of classes and the data nature, among others. In our experiments, we explore different vocabulary sizes, over different datasets, different interest points extraction and description methods, different weighting schemas, and different classifiers.

4 Experimental evaluation

In this work we have presented a methodology for improving BoW-based image representation by using only the most representative and discriminative visual words in the vocabulary. As it was stated in previous sections, our proposal does not depend on the algorithm used for building the set of visual words, the descriptor used nor the weighting scheme used. Therefore, the proposed methodology could be applied for improving the accuracy of any of the methods reported in the literature, which are based on a BoW approach.

The main goal of the experiments we present in this section is to quantitatively evaluate the improvement introduced by our proposal to the BoW-based image representation, over two standard datasets commonly used in object categorization. The experiments were focused on: a) to assess the validity of our proposal in a classic BoW-based classification task, b) to evaluate the methodology directly with respect to other kinds of feature selection algorithms, and c) to measure the time our methodology spent in order to filter the visual vocabulary built for each dataset. All the experiments were done on a single thread of a 3.6 GHz Intel i7 processor and 64GB RAM PC.

The experiments conducted in order to evaluate our proposal were done in two well-known datasets: Caltech-101 [10] and Pascal VOC 2006 [9].

The Caltech-101 dataset [10] consists of 102 object categories. There are about 40 to 800 images per category and most categories have about 50 images. The Pascal VOC 2006 dataset [9] consists of 10 object categories. In total, there are 5304 images, split into 50% for training/validation and 50% for testing. The distributions of images and objects by class are approximately equal across the training/validation and test sets.

4.1 Assessing the validity in a BoW-based classification task

As it was mentioned before, the goal of the first experiment is to assess the validity of our proposal. With this aim, we evaluate the accuracy in a classic BoW-based classification task, with and without applying our vocabulary filtering methodology.

In the experiments presented here, we use for image representation the BoW schema presented in Figure 1 with the following specifications:

- Interest points are detected and described using two methods: SIFT [22] and SURF [2].
- K-means, with four different K values, is used to build the visual vocabularies; these vocabularies constitute the baseline. For both Caltech-101 and Pascal VOC 2006 datasets we used K=10000, 15000, 20000 and 25000.
- Each of the baseline vocabularies is ranked using our proposed visual words ranking methodology.
- Later, nine new vocabularies are obtained by filtering each baseline vocabulary, leaving the 10%, 20%, ..., 90%, respectively, of the most representative and discriminative visual words based on the obtained ranking.
- Two weighting schemas are used for image representation: *tf* and *tf-idf*.

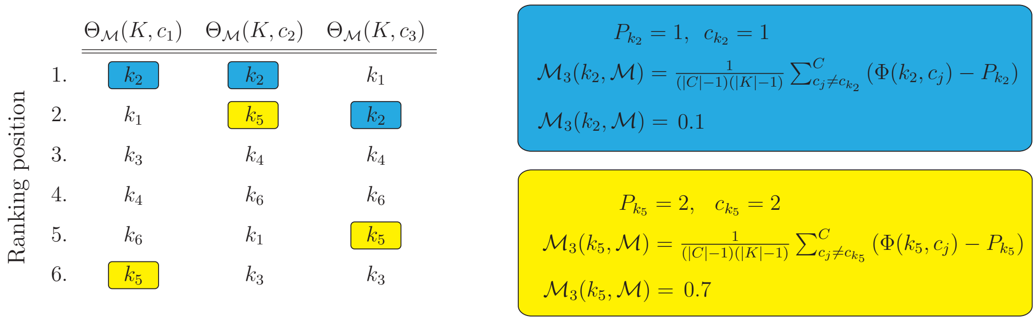

$\Theta_{\mathcal{M}}(K,c_1)$
$\Theta_{\mathcal{M}}(K,c_2)$
$\Theta_{\mathcal{M}}(K,c_3)$
Ranking position
1. k_2 k_2 k_1
2. k_1 k_5 k_2
3. k_3 k_4 k_4
4. k_4 k_6 k_6
5. k_6 k_1 k_5
6. k_5 k_3 k_3
$P_{k_2} = 1, \ c_{k_2} = 1$
$\mathcal{M}_3(k_2,\mathcal{M}) = \frac{1}{(|C|-1)(|K|-1)} \sum_{c_j \neq c_{k_2}}^{C} (\Phi(k_2,c_j) - P_{k_2})$
$\mathcal{M}_3(k_2,\mathcal{M}) = 0.1$
$P_{k_5} = 2, \ c_{k_5} = 2$
$\mathcal{M}_3(k_5,\mathcal{M}) = \frac{1}{(|C|-1)(|K|-1)} \sum_{c_j \neq c_{k_5}}^{C} (\Phi(k_5,c_j) - P_{k_5})$
$\mathcal{M}_3(k_5,\mathcal{M}) = 0.7$

Figure 4: *(best seen in color.)* Example of $\mathcal{M}_3$ measure for two visual words. k_2 has low discriminative power because it represents well several classes, while k_5 has high discriminative power because it describes well only one class.

– For both datasets we randomly selected 10 images from all the categories for building the visual vocabularies. The rest of the images were used as test images; but, as in [18], we limited to 50 the number of test images per category.

After that, we tested the obtained visual vocabularies in a classification task, using SVM (with a linear kernel) and KNN (where K is optimized with respect to the leave-one-out error) as classifiers. For each visual vocabulary, test images are represented using this vocabulary and, a 10-fold 10-times cross-validation process is conducted, where nine of the ten partitions are used for training and the other one for testing the trained classifier. The mean classification accuracy along the ten iterations is reported.

Figures 5 and 6 show the mean classification accuracy results over the cross-validation process using SVM and KNN, respectively, on the Caltech-101 dataset. Figures 7 and 8 show the same for the Pascal VOC 2006 dataset. In Figures 5 to 8, subfigures (a) and (b) show the results using SIFT descriptor; results for SURF are shown in subfigures (c) and (d). Results for the two different weighting schemas, i.e., *tf* and *tf-idf*, are shown in subfigures (a) (c), and (b) (d), respectively.

It can be seen that in both datasets, for every configuration, our proposed methodology allows to obtain reduced vocabularies that outperformed the classical BoW approach (baseline).

Table 2 summarizes the results presented in Figures 5 and 6 for the Caltech-101 dataset. The results in Figures 7 and 8 are summarized in Table 3. For every experiment configuration, Tables 2 and 3 show the baseline classification accuracy against the best result obtained by the proposed method with both SVM and KNN classifiers. The size of the filtered vocabulary in which the best result was obtained is also showed.

4.1.1 Discussion

The experimental results presented in this section validate the claimed contributions of our proposed method. As it can be seen in Tables 2 and 3, the best results obtained with our proposed method outperform those obtained with the whole vocabularies. For the experiments conducted in the Caltech-101 dataset, our average best results outperformed the baseline by a 4.6% and 4.8% in mean classification accuracy using SVM and KNN, respectively. In the Pascal VOC 2006 dataset there was a 1.6% and 4.7% improvement for SVM and KNN, respectively. As noticed on Figures 5 to 8, there is a trend of the performance with respect to the filter size in the two considered datasets, i.e., for smaller filter sizes higher accuracy.

In order to validate the improvement obtained by the proposed method, the statistical significance of the obtained results was verified. For testing the statistical significance we used the Mann-Whitney test, with a 95% of confidence. A detailed explanation about Mann-Whitney test, as well as an implementation, can be found in [1]. As a result of this test, it has been verified that the results obtained in both datasets, by the proposed method, are statistically superior to those obtained by the baseline.

In addition, the best results using the filtered vocabularies were obtained with vocabularies several times smaller than the baseline vocabularies, i.e., 6 and 10 times smaller in average using SVM and KNN, respectively, for the Caltech-101 datasets, and 8 and 5 times smaller in average for the Pascal VOC 2006 dataset with SVM and KNN, respectively. Furthermore, vocabularies 10 times smaller always obtained better accuracy results than the baseline vocabularies in the Caltech-101 dataset, and in the 93.75% of the experiments on the Pascal VOC 2006 dataset. Obtaining smaller vocabularies implies more compact image representations, that will have a direct impact on the efficiency of further processing based on these image representations, and less memory usage.

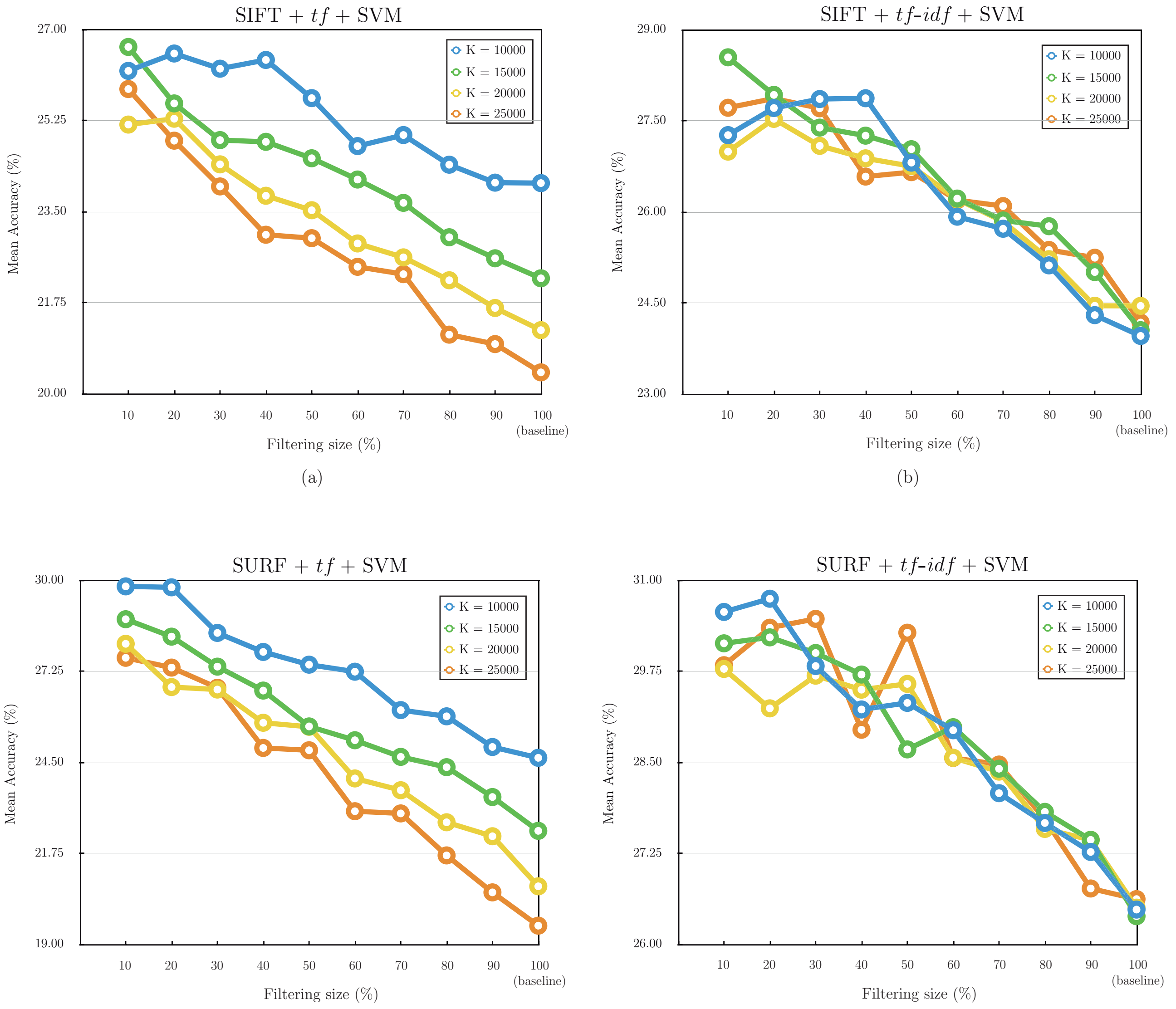
Also, the conducted experiments provide evidence that a large number of visual words in a vocabulary are noisy or little discriminative. Discarding these visual words allows for a better and more compact image representations.

4.2 Comparison with other kinds of feature selection algorithms

The aim of the second experiment is to compare our proposal with respect to other kind of feature selection algorithm. With this purpose, we compare the accuracy of our vocabulary filtering methodology with respect to the accu-

Table 1: Computation time (in seconds) of visual vocabulary ranking compared to vocabulary building.

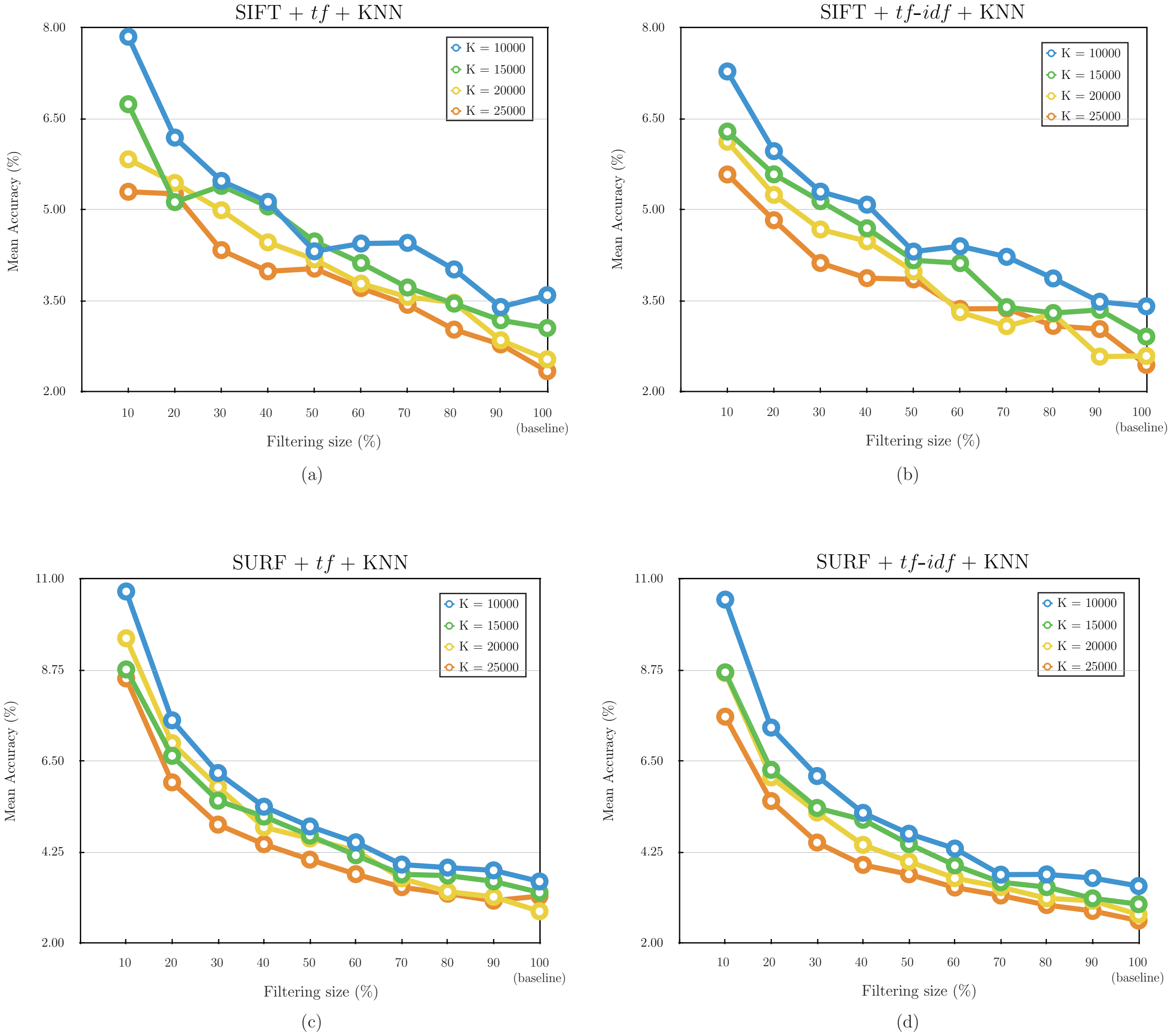
Dataset	K	Vocabulary building (K-means) computation time (s)	Vocabulary ranking (proposed method) computation time (s)	Vocabulary ranking (MI-based method [31]) computation time (s)
Caltech-101 (188 248 training features)	10000	4723.452	**8.111**	5.754
	15000	6711.089	**18.622**	16.825
	20000	7237.885	**33.890**	27.478
	25000	9024.024	**54.338**	49.963
Pascal VOC 2006 (114 697 training features)	10000	4126.487	**7.985**	5.285
	15000	5922.134	**18.320**	15.879
	20000	6441.980	**30.259**	28.458
	25000	8563.692	**51.743**	49.132



(a) (b)

(c) (d)

Figure 5: Mean classification accuracy results for SVM cross-validation on the Caltech-101 dataset. As can be seen, using the reduced vocabularies always resulted in better classification accuracies.



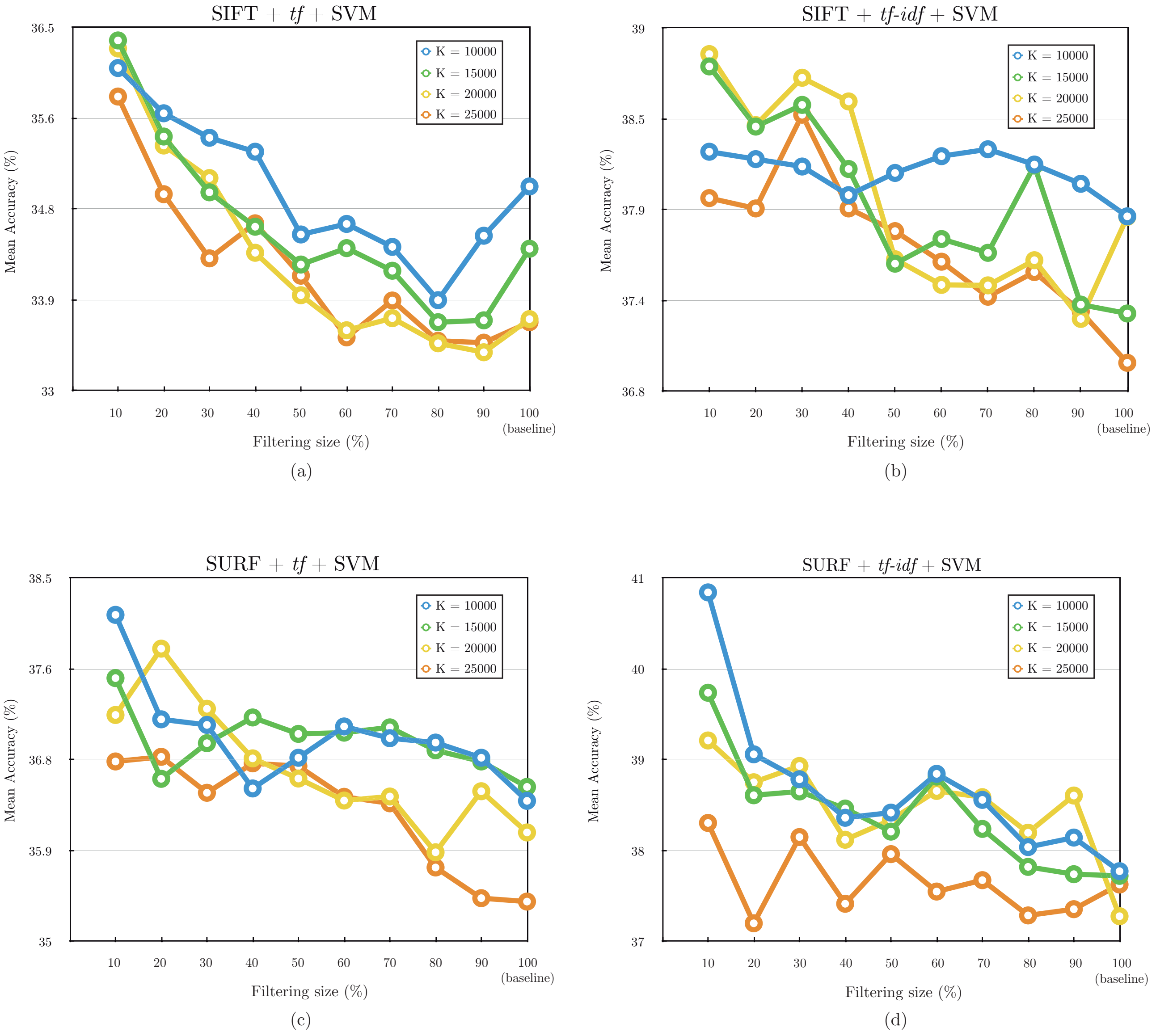
(a)

(b)

(c)

(d)

Figure 6: Mean classification accuracy results for KNN cross-validation on the Caltech-101 dataset. As can be seen, using the reduced vocabularies always resulted in better classification accuracies.



(a)

(b)

(c)

(d)

Figure 7: Mean classification accuracy results for SVM cross-validation on the Pascal VOC 2006 dataset. As can be seen, using the reduced vocabularies always resulted in better classification accuracies.

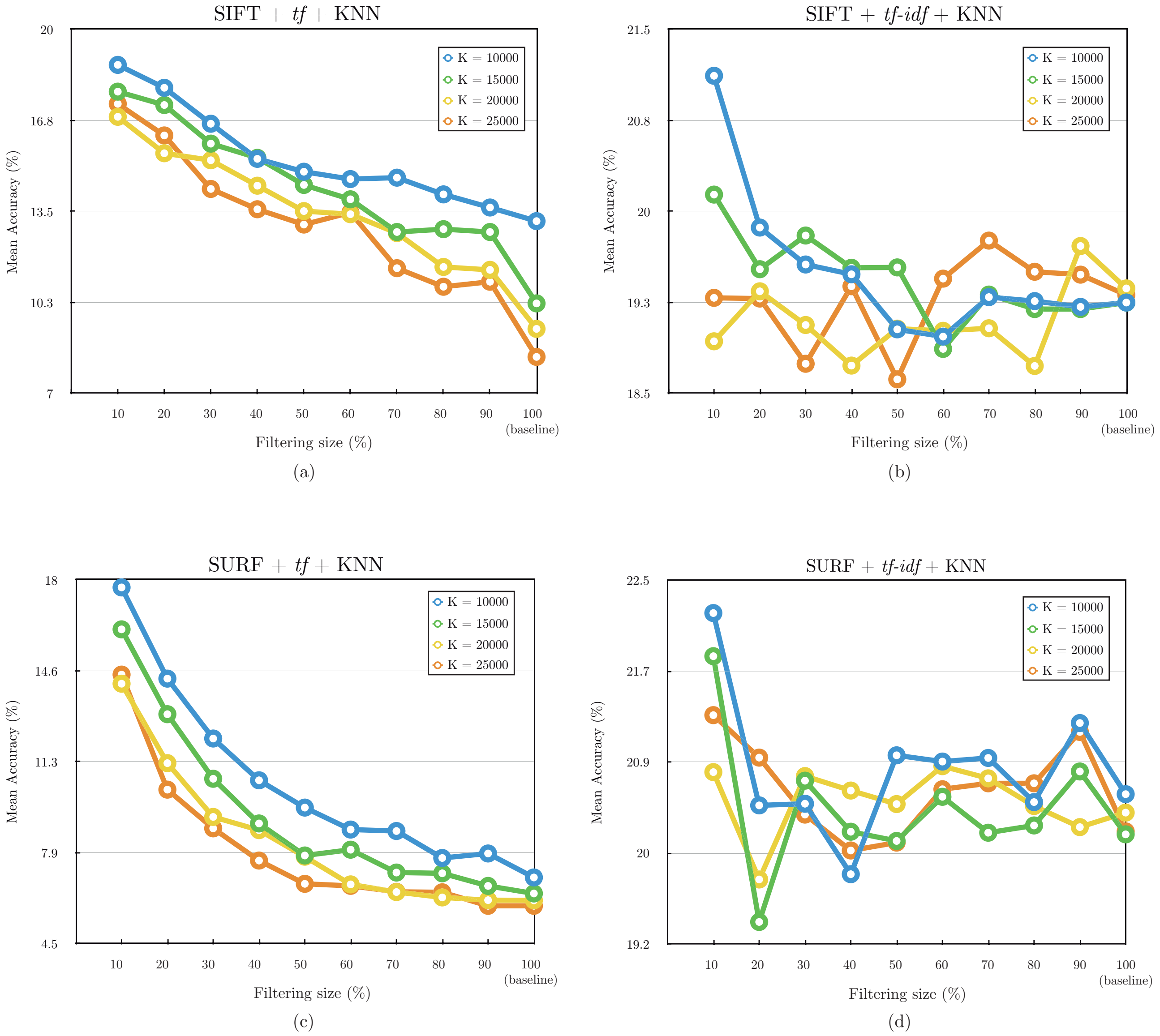


Figure 8: Mean classification accuracy results for KNN cross-validation on the Pascal VOC 2006 dataset. As can be seen, using the reduced vocabularies always resulted in better classification accuracies.

Table 2: Summarized results for the Caltech-101 dataset.

Descriptor	**Weighting schema**	**K**	**SVM**			**KNN**		
			Base-line	**Best result**	**Best filter size (%)**	**Base-line**	**Best result**	**Best filter size (%)**
SIFT	*tf*	10000	24.05	**26.54**	20	3.60	**7.86**	10
		15000	22.22	**26.66**	10	3.06	**6.74**	10
		20000	21.22	**25.28**	20	2.54	**5.84**	10
		25000	20.41	**25.85**	10	2.34	**5.30**	10
	tf-idf	10000	23.96	**27.87**	40	3.41	**7.28**	10
		15000	24.05	**28.55**	10	2.91	**6.29**	10
		20000	24.45	**27.53**	20	2.60	**6.13**	10
		25000	24.18	**27.87**	20	2.45	**5.59**	10
SURF	*tf*	10000	24.63	**29.81**	10	3.53	**10.70**	10
		15000	22.43	**28.82**	10	3.26	**8.77**	10
		20000	20.75	**28.08**	10	2.80	**9.54**	10
		25000	19.56	**27.66**	10	3.17	**8.55**	10
	tf-idf	10000	26.48	**30.74**	20	3.42	**10.50**	10
		15000	26.39	**30.21**	20	2.97	**8.70**	10
		20000	26.50	**29.78**	10	2.72	**8.69**	10
		25000	26.62	**30.47**	30	2.57	**7.61**	10
Average			23.62	**28.23**	16.8	2.96	**7.76**	10

Table 3: Summarized results for the Pascal VOC 2006 dataset.

Descriptor	**Weighting schema**	**K**	**SVM**			**KNN**		
			Base-line	**Best result**	**Best filter size (%)**	**Base-line**	**Best result**	**Best filter size (%)**
SIFT	*tf*	10000	34.97	**36.11**	10	13.16	**18.74**	10
		15000	34.37	**36.38**	10	10.22	**17.79**	10
		20000	33.69	**36.30**	10	9.31	**16.89**	10
		25000	33.66	**35.84**	10	8.31	**17.35**	10
	tf-idf	10000	37.86	**38.27**	10	19.25	**21.12**	10
		15000	37.27	**38.77**	10	19.25	**20.14**	10
		20000	37.86	**38.84**	10	19.37	**19.72**	90
		25000	36.97	**38.48**	30	19.31	**19.76**	70
SURF	*tf*	10000	36.36	**38.15**	10	6.96	**17.72**	10
		15000	36.49	**37.54**	10	6.37	**16.17**	10
		20000	36.05	**37.82**	20	6.11	**14.15**	10
		25000	35.39	**36.78**	20	5.91	**14.49**	10
	tf-idf	10000	37.77	**40.85**	10	20.56	**22.20**	10
		15000	37.72	**39.74**	10	20.19	**21.81**	10
		20000	37.28	**39.21**	10	20.39	**20.81**	60
		25000	37.63	**38.31**	10	20.22	**21.28**	10
Average			36.33	**37.96**	12.5	14.06	**18.76**	21.88

racy of the MI-based method proposed in [31], in a classification task; the experiment was done over the Caltech-101 dataset. As it was mentioned in Section 2, the MI-based method proposed in [31] obtains the best results among the feature selection and compression methods of image representation for object categorization.

In the experiments presented here, we use for image representation a BoW-based schema with the following specifications:

- PHOW features (dense multi-scale SIFT descriptors) [3].
- Spatial histograms as image descriptors.
- Elkan K-means [6], with five different K values (K= 256, 512, 1024, 2048 and 4096), is used to build the visual vocabularies; these vocabularies constitute the baseline.
- Each of the baseline vocabularies is ranked using the MI-based method proposed in [31] and our proposed visual vocabulary ranking methodology.
- Later, nine new vocabularies are obtained by filtering each baseline vocabulary, leaving the 10%, 20%, ..., 90%, respectively.
- We randomly selected 15 images from each of the 102 categories of Caltech-101 dataset, in order to build the visual vocabularies. For each category, 15 images were randomly selected as test images.

We tested the obtained visual vocabularies in a classification task, using a homogeneous kernel map to transform a χ^2 Support Vector Machine (SVM) into a linear one [26]. The classification accuracy is reported in Figure 9.

As it can be seen in Figure 9, for each value of K used in the experiment, our proposal obtains the best classification accuracy results for the highest compression rates. Besides, for the other filtering sizes our proposal and the MI-based method attains comparable results.

4.3 Computation time of the visual vocabulary ranking

The computation time of the visual vocabulary ranking methodology has also been evaluated. Table 1 shows the time in seconds taken for the ranking method in different size vocabularies, for the Caltech-101 and the Pascal VOC 2006 dataset. In Table 1, the ranking time is compared with the time needed to build the visual vocabulary.

As can be seen in Table 1, the proposed methodology can be used to improve visual vocabularies without requiring much extra computation time.

5 Conclusion and future work

In this work we devised a methodology for reducing the size of visual vocabularies that allows to obtain more discriminative and representative visual vocabularies for BoW image representation. The vocabulary reduction is based on three properties and their corresponding quantitative measures that express the inter-class representativeness, the intra-class representativeness and inter-class distinctiveness of visual words. The experimental results presented in this paper showed that, in average, with only 25% of the ranked vocabulary, statistically superior classification results can be obtained, compared to the classical BoW representation using the entire vocabularies. Therefore, the proposed method, in addition to providing accuracy improvements, provides a substantial efficiency improvement. Also, compared with a mutual information based method our proposal obtained superior results for the highest compression rates and comparable results for the other filtering sizes.

As future work, we aim to propose a weighting schema that takes advantage of the proposed measures, in order to improve image representation. Also we would like to explore the use of hierarchical classifiers for dealing with inter-class variability. Finally, we also aim to define a measure that help us to automatically choose the filter size.

References

[1] Concepts and applications of inferential statistics, 2013.

[2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.*, 110(3):346–359, 2008.

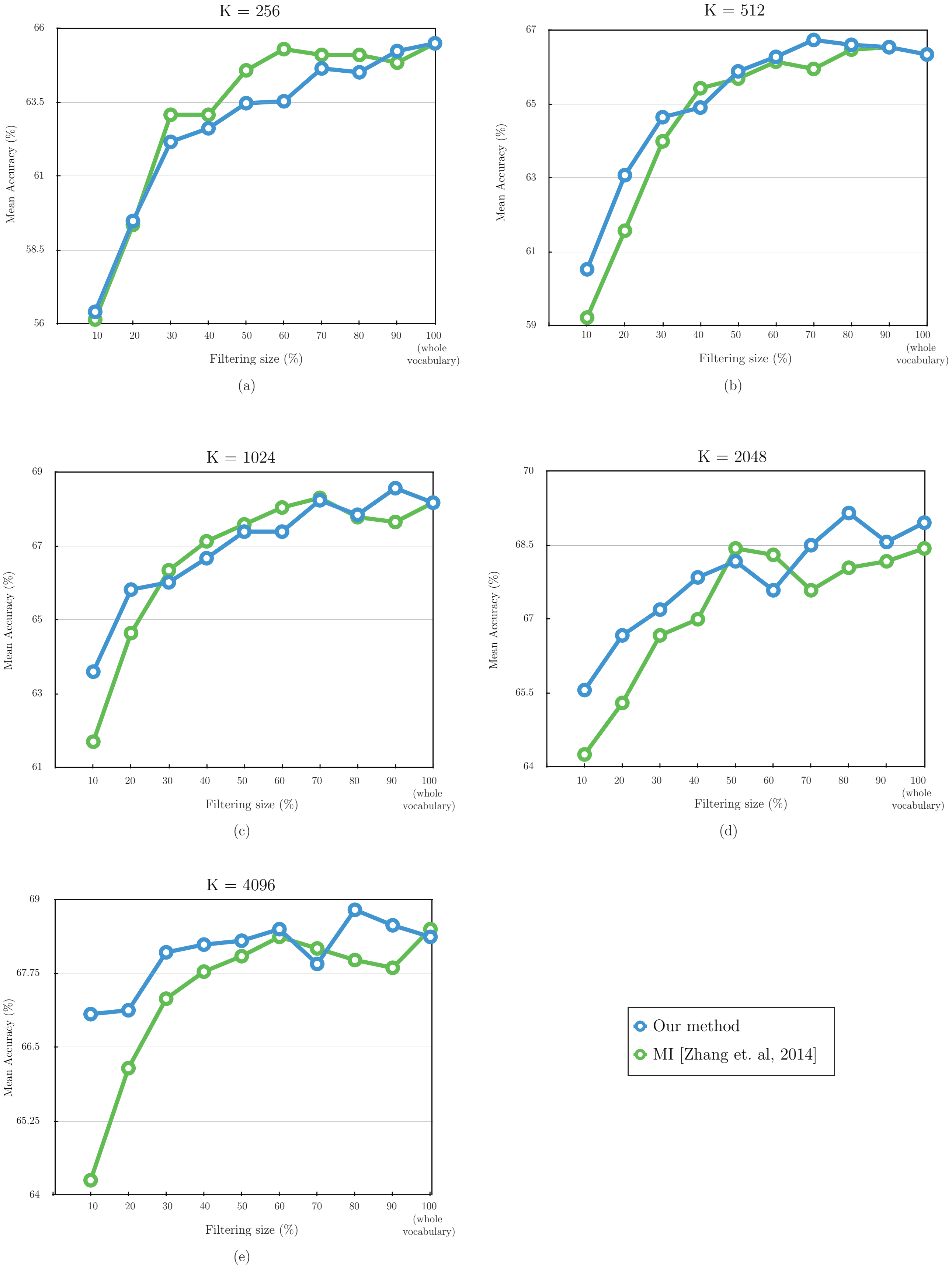
[3] A Bosch, Andrew Zisserman, and X Munoz. Image classification using random forests and ferns. *IEEE 11th International Conference on Computer Vision (2007)*, 23(1):1–8, 2007.

[4] Siddhartha Chandra, Shailesh Kumar, and C. V. Jawahar. Learning hierarchical bag of words using naive bayes clustering. In *Asian Conference on Computer Vision*, pages 382–395, 2012.

[5] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[6] Charles Elkan. Using the triangle inequality to accelerate k-means. In Tom Fawcett and Nina Mishra, editors, *ICML*, pages 147–153. AAAI Press, 2003.

[7] Peter Emerson. The original borda count and partial voting. *Social Choice and Welfare*, 40(2):353–358, 2013.



(a)

(b)

(c)

(d)

(e)

Figure 9: Comparison of mean classification accuracy results, on the Caltech-101 dataset, between the proposed methodology and the MI-based method proposed in [31].

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/ workshop/index.html.

[9] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. http://www.pascal-network.org/challenges/VOC/voc2006/ results.pdf.

[10] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Comput. Vis. Image Underst.*, 106(1):59–70, April 2007.

[11] Basura Fernando, Élisa Fromont, Damien Muselet, and Marc Sebban. Supervised learning of gaussian mixture models for visual vocabulary generation. *Pattern Recognition*, 45(2):897–907, 2012.

[12] Peter V. Gehler and Sebastian Nowozin. On feature combination for multiclass object classification. In *ICCV*, pages 221–228. IEEE, 2009.

[13] Y. Gong, S. Kumar, H. A. Rowley, and S. Lazebnik. Learning binary codes for high-dimensional data using bilinear projections. In *CVPR 2013*, 2013.

[14] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *Pattern Analysis and Machine Intellingence*, 33(1):117?128, 2011.

[15] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.

[16] Mingyuan Jiu, Christian Wolf, Christophe Garcia, and Atilla Baskurt. Supervised learning and codebook optimization for bag of words models. *Cognitive Computation*, 4:409–419, December 2012.

[17] Kraisak Kesorn and Stefan Poslad. An enhanced bag-of-visual word vector space model to represent visual content in athletics images. *IEEE Transactions on Multimedia*, 14(1):211–222, 2012.

[18] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Volume 2 CVPR06*, 2(2169-2178):2169–2178, 2006.

[19] Gang Liu. Improved bags-of-words algorithm for scene recognition. *Journal of Computational Information Systems*, 6(14):4933 – 4940, 2010.

[20] Jingen Liu and Mubarak Shah. Learning human actions via information maximization. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 0:1–8, 2008.

[21] R.J. Lopez-Sastre, T. Tuytelaars, F.J. Acevedo-Rodriguez, and S. Maldonado-Bascon. Towards a more discriminative and semantic visual vocabulary. *Computer Vision and Image Understanding*, 115(3):415 – 425, 2011. Special issue on Feature-Oriented Image and Video Computing for Extracting Contexts and Semantics.

[22] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[23] Zhiwu Lu and Horace Ho-Shing Ip. Image categorization with spatial mismatch kernels. In *CVPR*, pages 397–404. IEEE, 2009.

[24] Jianzhao Qin and Nelson Hon Ching Yung. Feature fusion within local region using localized maximum-margin learning for scene categorization. *Pattern Recognition*, 45(4):1671–1683, 2012.

[25] Chih-Fong Tsai. Bag-of-words representation in image annotation: A review. *ISRN Artificial Intelligence*, 2012, 2012.

[26] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *Pattern Analysis and Machine Intellingence*, 34(3), 2011.

[27] Lei Wang, Lingqiao Liu, and Luping Zhou. A graph-embedding approach to hierarchical visual word mergence. *IEEE Trans. Neural Netw. Learning Syst.*, 28(2):308–320, 2017.

[28] Jianxin Wu, Wei-Chian Tan, and James M. Rehg. Efficient and effective visual codebook generation using additive kernels. *Journal of Machine Learning Research*, 12:3097–3118, 2011.

[29] Shiliang Zhang, Qi Tian, Gang Hua, Qingming Huang, and Wen Gao. Generating descriptive visual words and visual phrases for large-scale image applications. *IEEE Transactions on Image Processing*, 20(9):2664–2677, 2011.

[30] Shiliang Zhang, Qi Tian, Gang Hua, Wengang Zhou, Qingming Huang, Houqiang Li, and Wen Gao. Modeling spatial and semantic cues for large-scale near-duplicated image retrieval. *Computer Vision and Image Understanding*, 115(3):403–414, 2011.

[31] Y. Zhang, J. Wu, and J. Cai. Compact representation for image classification: To choose or to compress? In *CVPR 2014*, 2014.

An Output Instruction Based PLC Source Code Transformation Approach For Program Logic Simplification

Arup Ghosh and Shiming Qin
Unified Digital Manufacturing Laboratory, Department of Industrial Engineering, Ajou University
Suwon 443-749, Republic of Korea
E-mail: arupghosh22.3.89@gmail.com, taihejiang11@ajou.ac.kr

Jooyeoun Lee and Gi-Nam Wang
Department of Industrial Engineering, Ajou University, Suwon 443-749, Republic of Korea
E-mail: jooyeoun325@ajou.ac.kr, gnwang@ajou.ac.kr

Keywords: programmable logic controller, reengineering, ladder logic diagram, instruction list, XML

Received: August 26, 2016

Due to the growing size and complexity of the PLC (Programmable Logic Controller) programs used for controlling the industrial processes, there is an increasing need for an approach that can help the users to understand the control logics of the PLC programs easily, and can assist them to analyze the programming errors effectively. In this paper, we propose an approach that takes the source code file of PLC program as the input; and transforms it into a hierarchical-structured XML (extensible markup language) file. The XML file format is based on the PLC output instructions and their corresponding conditions. It helps the users to identify the actual cause of a programming error quickly. In addition, a novel technique is applied that decomposes the PLC program into several smaller and modular sub-logic blocks. This makes the control logic simpler and easier to follow. An additional software application has also been developed for state-based graphical visualization of the XML file.

Povzetek: Prispevek opisuje metodo za poenostavitev PLC programov za industrijske procese.

1 Introduction

The PLCs are a special type of computers that are used for automation of the industrial processes. A PLC controls an industrial process according to the control program embedded in its controller. In each execution of the PLC program, it takes the sensor signals as the inputs and produces a set of output control signals to the actuators. So, the program outputs and their corresponding conditions (which must be satisfied in order to receive that particular output) are the basis of a PLC program. The PLCs can be programmed by using several programming languages under the international standard IEC 61131-3, such as Ladder Logic Diagram (LLD), Function Block Diagram (FBD), Structured Text (ST), Instruction List (IL) etc. [1]. Among these languages, the LLD is the most popular PLC programming language in industries; and the IL is the most commonly used PLC programming language in Europe [1]–[3]. On various occasions, the programmers use a combination of these languages to write a PLC program. With the growing size and complexity of the PLC programs, it becomes more difficult to understand the program logics because of the low-level PLC programming languages. Moreover, if an error is detected in any PLC output, then the programmers have to analyze the complete program manually to find out the conditions that can cause such an error. It is very complicated and time-consuming job for a programmer to determine all the conditions that can affect a particular output. The situation becomes more critical when many programmers work together to develop the project. Moreover, if the routines of the PLC program are written in different languages, then understanding the control logics and/or determining the conditions associated with a program output become even harder.

Our main aim is to transform the PLC program source code into a programming language and vendor independent XML file format that can help the users to understand the program logic easily; and can assist the programmers to analyze the programming errors quickly. In this paper, we present a PLC program source code reengineering approach, called Program Output based Source-code Transformation (POST) approach that takes the source code file of the PLC program saved in the IL language as the input, and produces a hierarchical-structured XML file as the output. In that XML file, the program logic is interpreted in terms of the program output instructions and their corresponding conditions, thus the programmers can analyze the programming errors easily. In addition, POST applies a novel technique that subdivides the program logic blocks into several smaller and modular sub-logic blocks in order to make the program logic more simpler, clearer and well-organized. POST is applicable to all the programming languages and the PLC software vendors where the program source code can be saved in the IL language. For example, in case of Siemens PLC software [4], the programs written in the LLD, IL and FBD languages can automatically be saved

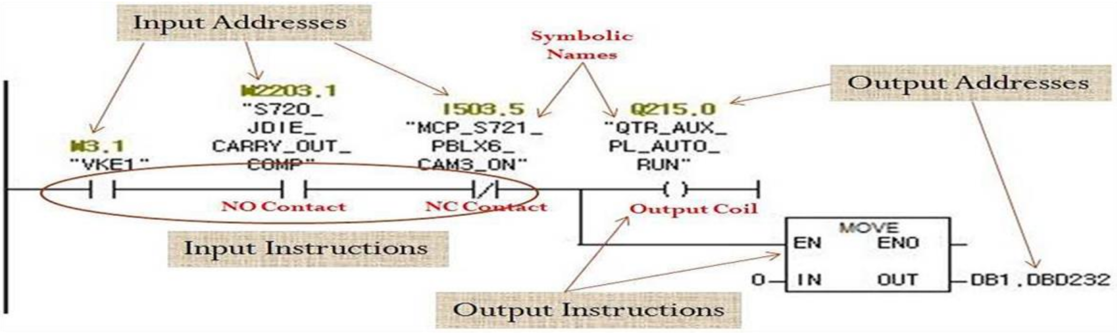
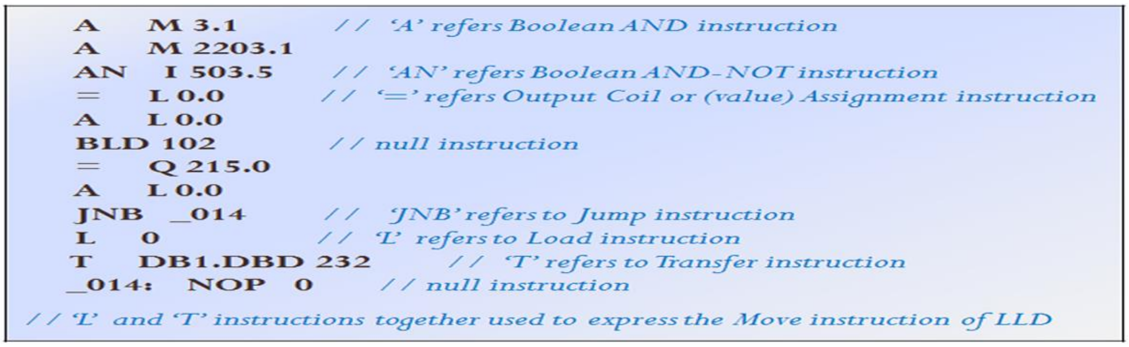


Figure 1: A rung of a LLD program.



```
A     M 3.1           // 'A' refers Boolean AND instruction
A     M 2203.1
AN     I 503.5        // 'AN' refers Boolean AND-NOT instruction
=      L 0.0          // '=' refers Output Coil or (value) Assignment instruction
A      L 0.0
BLD 102               // null instruction
=      Q 215.0
A      L 0.0
JNB   _014            // 'JNB' refers to Jump instruction
L     0               // 'L' refers to Load instruction
T     DB1.DBD 232          // 'T' refers to Transfer instruction
_014:   NOP  0        // null instruction
// 'L' and 'T' instructions together used to express the Move instruction of LLD
```

Figure 2: The IL language representation of the LLD rung of Figure 1.

in the IL language format. We have implemented and tested POST for Siemens and Allen-Bradley PLC software [5]. It can easily be extended for other types of PLC software as well.

2 Problem Description

In Figure 1, an example rung of a LLD program (written using Siemens Simatic Step 7 software [4]) is given. Each rung of a LLD program characterizes a specific rule or a set of rules. As can be seen in Figure 1, the rung has two output instructions and those outputs are dependent on three input instructions (or conditions) i.e., two Normally Open (NO) contacts and one Normally Closed (NC) contact. The NO and NC contacts actually represent the AND and AND-NOT boolean logic operations, respectively. These conditions are evaluated at the time of program execution in order to determine the data values of the output addresses. In practice, a LLD program can have thousands of such rungs partitioned into several program blocks, such as the Organization Blocks (OBs), Functions (FCs), Function Blocks (FBs) etc. It is very time-consuming and laborious task for the programmers to identify the real cause of a programming error. This is because, if an error is found in any PLC output signal, then the programmers have to examine the complete program (i.e., each rung of every program blocks) manually to find out the exact conditions that can affect the value of the corresponding output address. In that condition candidate set, if an erroneous data value is found in the address field of an input instruction which is not a direct sensor input, then the programmers have to search again for the conditions that can affect the value of that address. This process continues until the root causes of the error (in other words, the faulty sensor inputs and/or the flaws in the program logic) are identified. In order to overcome this kind of difficulties, an attempt is given to transform the source code file of the PLC program into a well-organized and well-structured XML file, thus all the conditions attached to an output address can be determined automatically. This can help the users to fix the programming errors very quickly.

The PLC programs are often written in a combination of different languages. The input-output instructions of a particular programming language also vary depending on the PLC software vendors. So, it is necessary to transform the PLC code into a vendor and language independent format, thus the users can understand the program instructions quickly and easily. An automated approach for program logic simplification is another important requirement for industries. Often PLC programs are written in a very low-level, non-graphical language such as the IL language. An IL language code equivalent to the LLD rung of Figure 1 is given in Figure 2. As can be seen, it is very hard to understand the rung logic from this kind of non-graphical PLC programs. Even for the programs written in graphical languages such as the LLD, FBD etc., it becomes difficult to understand the program logic with the growing size and complexity of the rung diagram (especially if the rung has several outputs, parallel branches and sub-branches). An example of such complex LLD rung is presented in Figure 3. It is easy to perceive, identifying the conditions or understanding the program logic behind a particular output is very difficult from this kind of ladder rungs. Therefore, an automated, systematic approach is required that can simplify the program logic of this kind of complex ladder rungs in an efficient way, thus the users can understand the program logic behind a

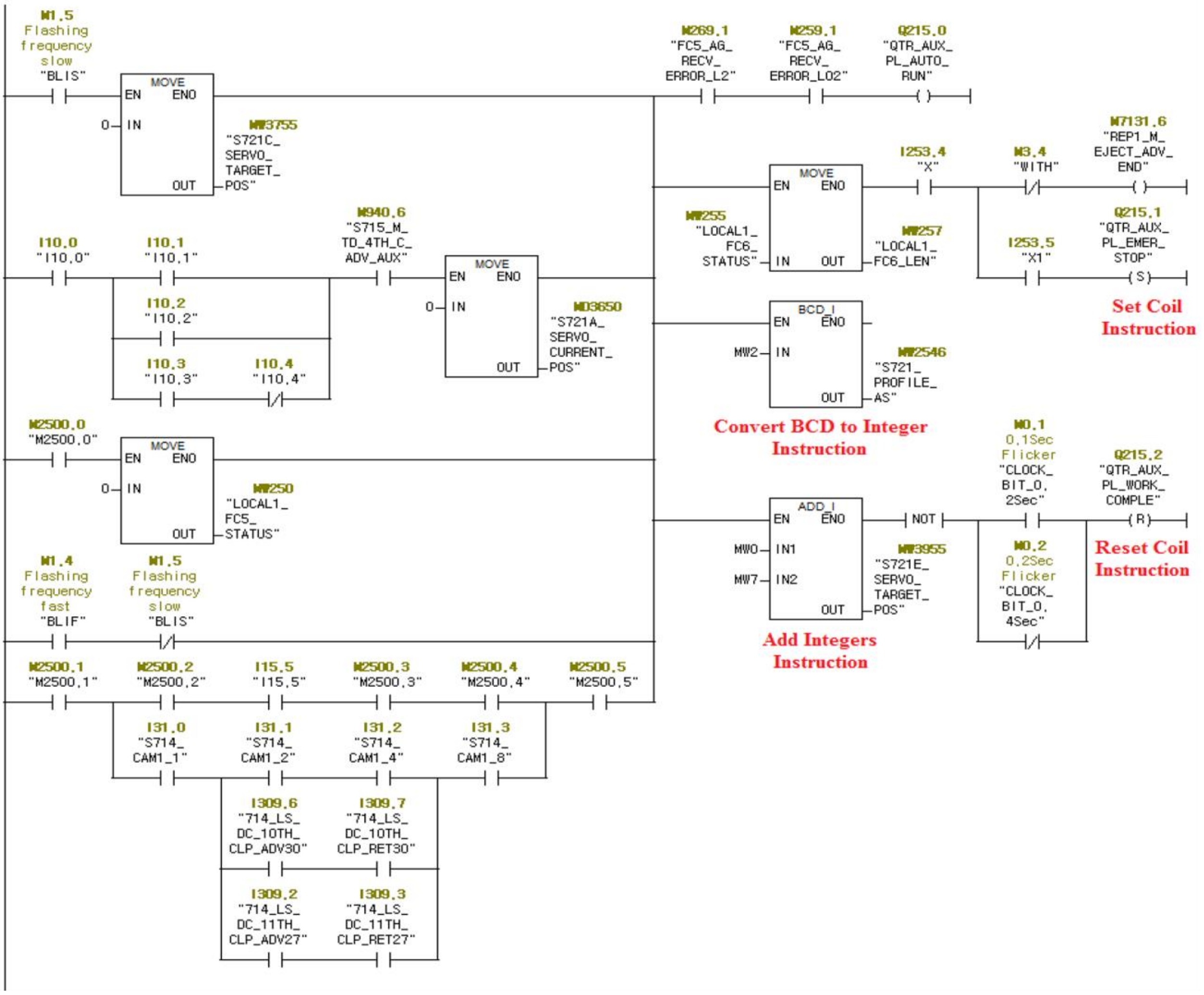


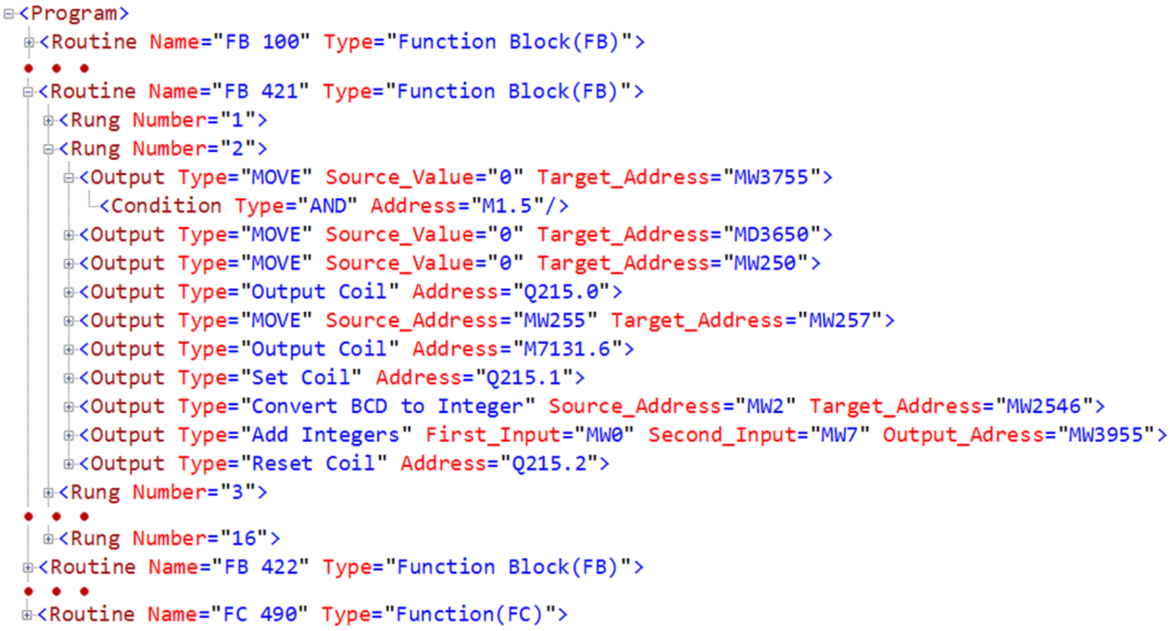
Figure 3: An example complex ladder rung.

particular output easily. In this work, we have successfully addressed those needs. The rest of this paper is organized as follows: an overview about the existing works in this field is presented in Section 3. In Section 4, we have discussed POST approach in details. Section 5 contains our conclusive remarks of the work followed by a list of relevant references.

3 Background Study

Several approaches have been proposed in literatures for reengineering the PLC programs. They can broadly be classified into the following three categories:

- Approaches focused on source-to-source translation: this type of approaches transform the PLC programs written in a particular programming language into another programming language. For example, the approaches proposed in [6]–[9] transform the LLD program into the IL language code. The main objective of these research works is to convert the PLC program into the IL language code thus it can be executed directly by the PLCs. In [10], a different type of approach was proposed that converts the LLD program into the ST language code. The main aim of this research work is to promote a particular type of technology and hardware.
- Approaches focused on vendor interoperability: these research works propose an approach that can accomplish transferring the program source code among different vendors of PLC programming tools. In these works, the interoperability between the PLC programming tools is achieved by means of a middleware. In most of the cases, the XML technologies have been used for developing the interoperability middleware. Examples of such works include: [11]–[15].
- Approaches focused on alternative visualization: this type of approaches transforms the PLC program source code file into another file format for more efficient graphical visualization. For example, in earlier works [3] and [16], an approach was proposed that transforms the PLC program into a vendor and platform independent XML file format. In [17] and [18], an approach was proposed that transforms the PLC programs into the Finite State Machines (FSMs). In another article [19], the UML (Unified Modelling Language) state diagrams are used in place of FSMs for more efficient graphical visualization of the rung logic.



```
⊟<Program>
 ⊞<Routine Name="FB 100" Type="Function Block(FB)">
 •  •  •
 ⊟<Routine Name="FB 421" Type="Function Block(FB)">
   ⊞<Rung Number="1">
   ⊟<Rung Number="2">
     ⊟<Output Type="MOVE" Source_Value="0" Target_Address="MW3755">
        └<Condition Type="AND" Address="M1.5"/>
     ⊞<Output Type="MOVE" Source_Value="0" Target_Address="MD3650">
     ⊞<Output Type="MOVE" Source_Value="0" Target_Address="MW250">
     ⊞<Output Type="Output Coil" Address="Q215.0">
     ⊞<Output Type="MOVE" Source_Address="MW255" Target_Address="MW257">
     ⊞<Output Type="Output Coil" Address="M7131.6">
     ⊞<Output Type="Set Coil" Address="Q215.1">
     ⊞<Output Type="Convert BCD to Integer" Source_Address="MW2" Target_Address="MW2546">
     ⊞<Output Type="Add Integers" First_Input="MW0" Second_Input="MW7" Output_Adress="MW3955">
     ⊞<Output Type="Reset Coil" Address="Q215.2">
   ⊞<Rung Number="3">
 •  •  •
   ⊞<Rung Number="16">
 ⊞<Routine Name="FB 422" Type="Function Block(FB)">
 •  •  •
 ⊞<Routine Name="FC 490" Type="Function(FC)">
```

Figure 4: The output XML file format.

Unfortunately, all the mentioned approaches are focused on an efficient graphical representation of the PLC program and/or the vendor and platform interoperability. None of these approaches fulfils all the requirements stated in Section 1 and Section 2, and hence, a completely different type of approach is needed. Our proposed approach POST can solve all those needs effectively.

4 POST approach

This section is divided into five subsections. In Subsection 4.1, the overall structure of the output XML file (produced by POST approach) is given and in Subsection 4.2, the program logic simplification procedure of POST is discussed. The program error analysis procedure is presented in Subsection 4.3. In Subsection 4.4, the implementation details of POST approach is discussed and in Subsection 4.5, the output XML file format for a special instruction i.e., the block call instruction is given. In this paper, we discuss POST approach using the ladder rung diagrams of Siemens PLC programs (just for exemplification purpose).

4.1 The Overall XML file structure

POST takes the source code file of a PLC program saved in the IL language as the input and produces a well-structured and well-organized XML file. It gives an efficient tree-based representation of the program logic to the users. The XML file structure outputted by POST is based on the output instructions and their corresponding conditions of each rung of the PLC program. The overall structure of the output XML file is given in Figure 4 (some XML nodes are not expanded in order to maintain the clarity of the image). As we can see, under the root node i.e. Program node, the Routine nodes are defined. A routine actually refers to a block of the PLC program. The Type attribute of the Routine nodes specifies the type of that routine i.e., OB or FB or FC etc. In a LLD program, the ladder rungs are always declared inside a routine and hence, under the Routine node, the Rung nodes are defined. The Number attribute represents the corresponding rung number in the routine. As can be seen in Figure 4, under the Rung node, the Output nodes are characterized. Each Output node basically represents a separate output of the corresponding rung. The Type attribute of the Output nodes refers to the type of that output instruction such as the Output Coil, Convert BCD to Integer (CBI), Move, Set or Reset Coil instruction etc. (see for instance: [20] and [21]). The Move and the CBI type instructions have the following two additional attributes: i) Source_Address or Source_Value attribute: represents the address or the value specified at the IN input; and ii) Target_Address attribute: represents the address specified at the OUT output (see Figure 3 and Figure 4). Similarly, the Output Coil type instructions have one additional attribute i.e., the Address attribute which characterizes the output address of the corresponding instruction (the same is also true for the Set and Reset Coil instructions). The additional attributes associated with an instruction (or an Output node) actually represent the addresses and the data values associated with that instruction. As can be seen in Figure 4, POST determines the number of additional attributes and their names (or formats) based on that particular type of instruction (also see [20] and [21]).

In our original PLC program, the ladder rung of Figure 3 is actually the second rung of the function block FB 421. In Figure 4, we can find the Output nodes corresponding to the ladder rung of Figure 3 under the rung number 2 node. As can be seen in Figure 3, the rung consists of ten output instructions and hence, ten Output nodes are created under the rung number 2 node in the XML file of Figure 4. Actually, in the output XML file, a rung diagram is characterized on the basis of its output instructions and hence, under each Output node, we can find its corresponding conditions. For example, as can be

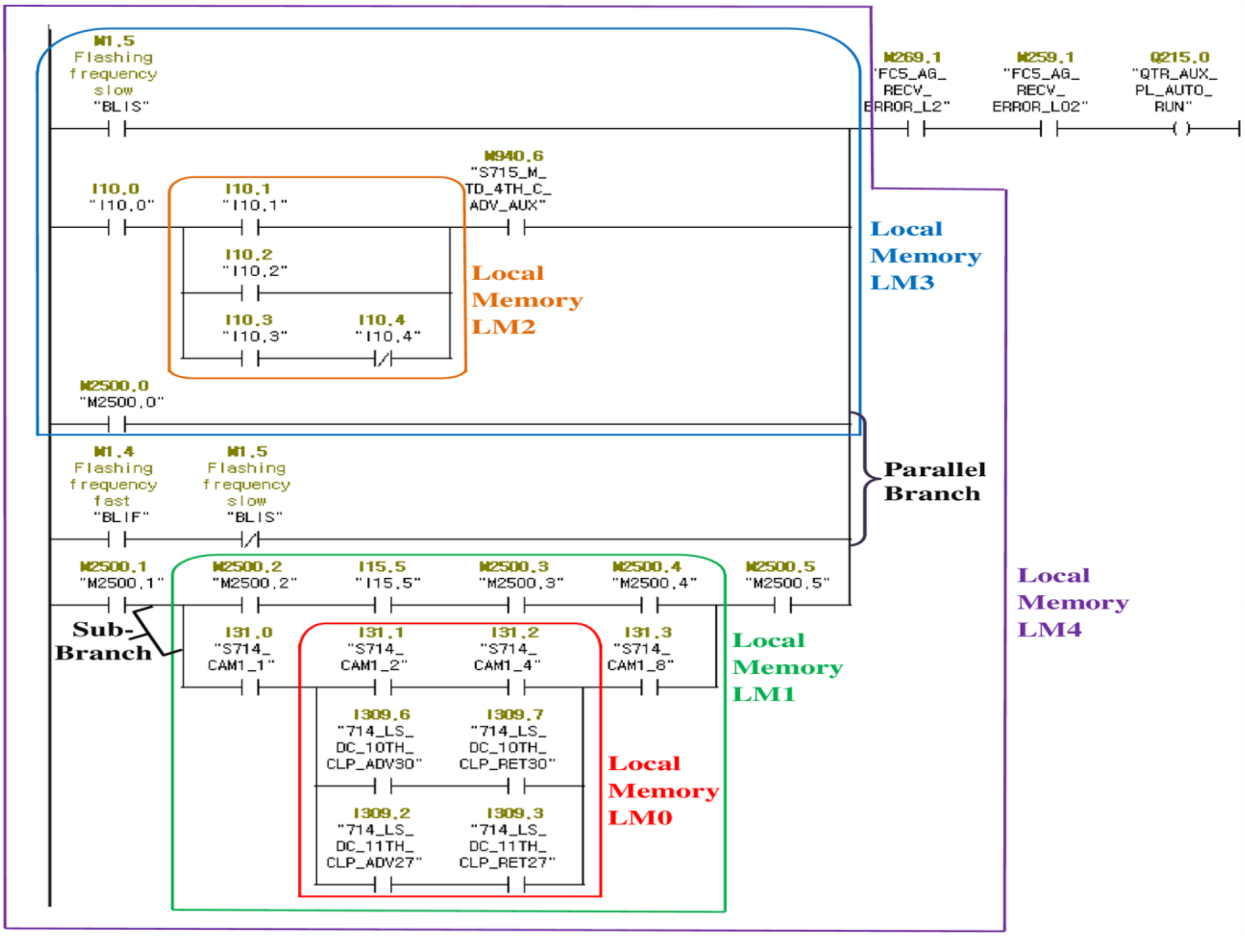
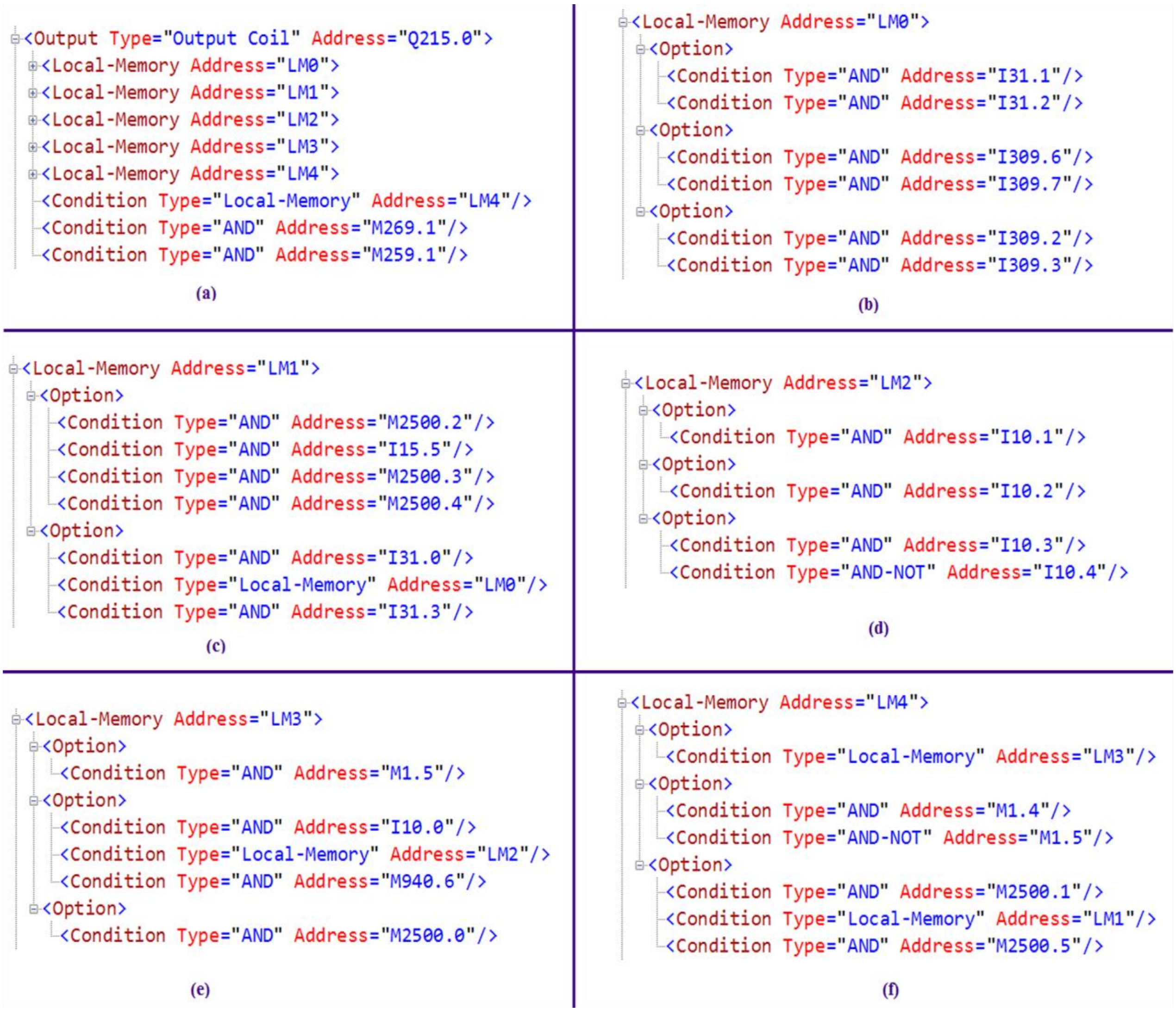


Figure 5: The rung diagram (or rung logic) associated with the first Output Coil instruction (address Q215.0).

seen in Figure 3, the first Move type output instruction (see the first branch) has only one corresponding condition (the condition type AND with address argument M1.5). As can be seen from Figure 4, this condition is correctly placed right below the corresponding Output node in the XML file. It is easy to perceive from the format of the output XML file, it follows exactly the same logic structure as in the original PLC program. However, if any rung has more than one output instruction, then the rung logic is split based on the corresponding output instructions. This is the first logic simplification measure taken by POST (this also simplifies the programming error analysis task – we will discuss on it later). In addition, as can be seen in Figure 4, the instructions and their corresponding properties are described by using simple descriptive language. This makes the program logic easy understandable, and programming language and platform independent. Please note that POST can also produce the output XML file based on the symbolic names (see Figure 1).

4.2 Program Logic Simplification by utilizing the Local Memory Definition

The program output based source code transformation method simplifies the rung structure to a great extent. However, further logic simplification measures are needed to be taken particularly for the rungs with a large number of parallel and high depth sub-branches. The parallel branches and sub-branches of a LLD rung represent the OR boolean logic operations. This means that the Result of Logic Operation (RLO) of the parallel branches (respectively, sub-branches) is true, if RLO of any of those branch (respectively, sub-branch) is true [20, 21]. POST simplifies the logic of a rung diagram with parallel branches and sub-branches by using a bottom-up hierarchical decomposition procedure. More specifically, it characterizes a certain portion of the complete rung diagram (a sub-logic block) by utilizing the Local Memory Definition (LMD) [a local memory can be thought of as a virtual memory location where the RLO of its corresponding sub-logic block is stored]. The LMDs are then used successively (reused in a modular fashion) to define the complete rung logic. In Figure 5, the relevant part of the rung diagram of Figure 3 that depicts only the conditions associated with the first Output Coil instruction (address Q215.0) is given. As can be seen from Figure 5, even after the above mentioned logic simplification step (as stated in Subsection 4.1), the rung diagram has several parallel branches and sub-branches. For this reason, the program logic behind the output is difficult to follow and hence, is needed to be simplified further.



```
<Output Type="Output Coil" Address="Q215.0">
  <Local-Memory Address="LM0">
  <Local-Memory Address="LM1">
  <Local-Memory Address="LM2">
  <Local-Memory Address="LM3">
  <Local-Memory Address="LM4">
  <Condition Type="Local-Memory" Address="LM4"/>
  <Condition Type="AND" Address="M269.1"/>
  <Condition Type="AND" Address="M259.1"/>
```

(a)

```
<Local-Memory Address="LM0">
  <Option>
    <Condition Type="AND" Address="I31.1"/>
    <Condition Type="AND" Address="I31.2"/>
  <Option>
    <Condition Type="AND" Address="I309.6"/>
    <Condition Type="AND" Address="I309.7"/>
  <Option>
    <Condition Type="AND" Address="I309.2"/>
    <Condition Type="AND" Address="I309.3"/>
```

(b)

```
<Local-Memory Address="LM1">
  <Option>
    <Condition Type="AND" Address="M2500.2"/>
    <Condition Type="AND" Address="I15.5"/>
    <Condition Type="AND" Address="M2500.3"/>
    <Condition Type="AND" Address="M2500.4"/>
  <Option>
    <Condition Type="AND" Address="I31.0"/>
    <Condition Type="Local-Memory" Address="LM0"/>
    <Condition Type="AND" Address="I31.3"/>
```

(c)

```
<Local-Memory Address="LM2">
  <Option>
    <Condition Type="AND" Address="I10.1"/>
  <Option>
    <Condition Type="AND" Address="I10.2"/>
  <Option>
    <Condition Type="AND" Address="I10.3"/>
    <Condition Type="AND-NOT" Address="I10.4"/>
```

(d)

```
<Local-Memory Address="LM3">
  <Option>
    <Condition Type="AND" Address="M1.5"/>
  <Option>
    <Condition Type="AND" Address="I10.0"/>
    <Condition Type="Local-Memory" Address="LM2"/>
    <Condition Type="AND" Address="M940.6"/>
  <Option>
    <Condition Type="AND" Address="M2500.0"/>
```

(e)

```
<Local-Memory Address="LM4">
  <Option>
    <Condition Type="Local-Memory" Address="LM3"/>
  <Option>
    <Condition Type="AND" Address="M1.4"/>
    <Condition Type="AND-NOT" Address="M1.5"/>
  <Option>
    <Condition Type="AND" Address="M2500.1"/>
    <Condition Type="Local-Memory" Address="LM1"/>
    <Condition Type="AND" Address="M2500.5"/>
```

(f)

Figure 6: XML file format for the rung diagrams with parallel branches and sub-branches. (a) The condition set corresponding to the first Output Coil instruction. (b) Local memory LM0 definition. (c) Local memory LM1 definition. (d) Local memory LM2 definition. (e) Local memory LM3 definition. (f) Local memory LM4 definition.

From our practical experience, we have seen that the conventional procedure to understand this kind of complicated rung structure (as in Figure 5) is to analyze the rung diagram starting from its highest depth sub-branches, and then consecutively proceeding towards the main branch and its parallel branches (in a bottom-up fashion). The LMD based logic simplification procedure of POST exactly follows this natural bottom-up modular decomposition approach. As we can see from Figure 5, the (relatively straightforward) rung logic corresponding to the highest depth sub-branches (branches inside the red colour box) will be defined by using the local memory LM0. Similarly, the rung logic corresponding to the next highest depth sub-branches (branches inside the green colour box) will be characterized by using the local memory LM1. It is easy to perceive, the definition of local memory LM0 can successively be utilized in the definition of local memory LM1. As can be seen from Figure 5, this LMD formulation procedure will be repeated in a bottom-up fashion until all the parallel branches and sub-branches are characterized by using the LMDs. In Figure 5, the boxes and its associated local memory names represent how the rung structure can further be simplified by using the LMDs [The LMDs are restricted to maximum three parallel branches. As an example, see the branches inside the blue colour box. We will discuss more on it later.]. For simplicity, we can suppose that the RLO of the parallel sub-branches of a branch (respectively, the parallel branches of the main branch) is stored in a virtual memory location of the type local memory, and is used successively to evaluate the RLO of that branch (respectively, the main branch) by applying an AND boolean logic operation.

The output XML file shown in Figure 6 depicts the condition set (or the rung logic) corresponding to the first Output Coil instruction of Figure 3 (also see the simplified rung diagram of Figure 5). As can be seen in Figure 6 (a), the rung logic or the rung diagram associated with the first Output Coil instruction is characterized based on the definition of local memory LM4. The definition of local memory LM0, LM1, LM2, LM3 and LM4 are shown

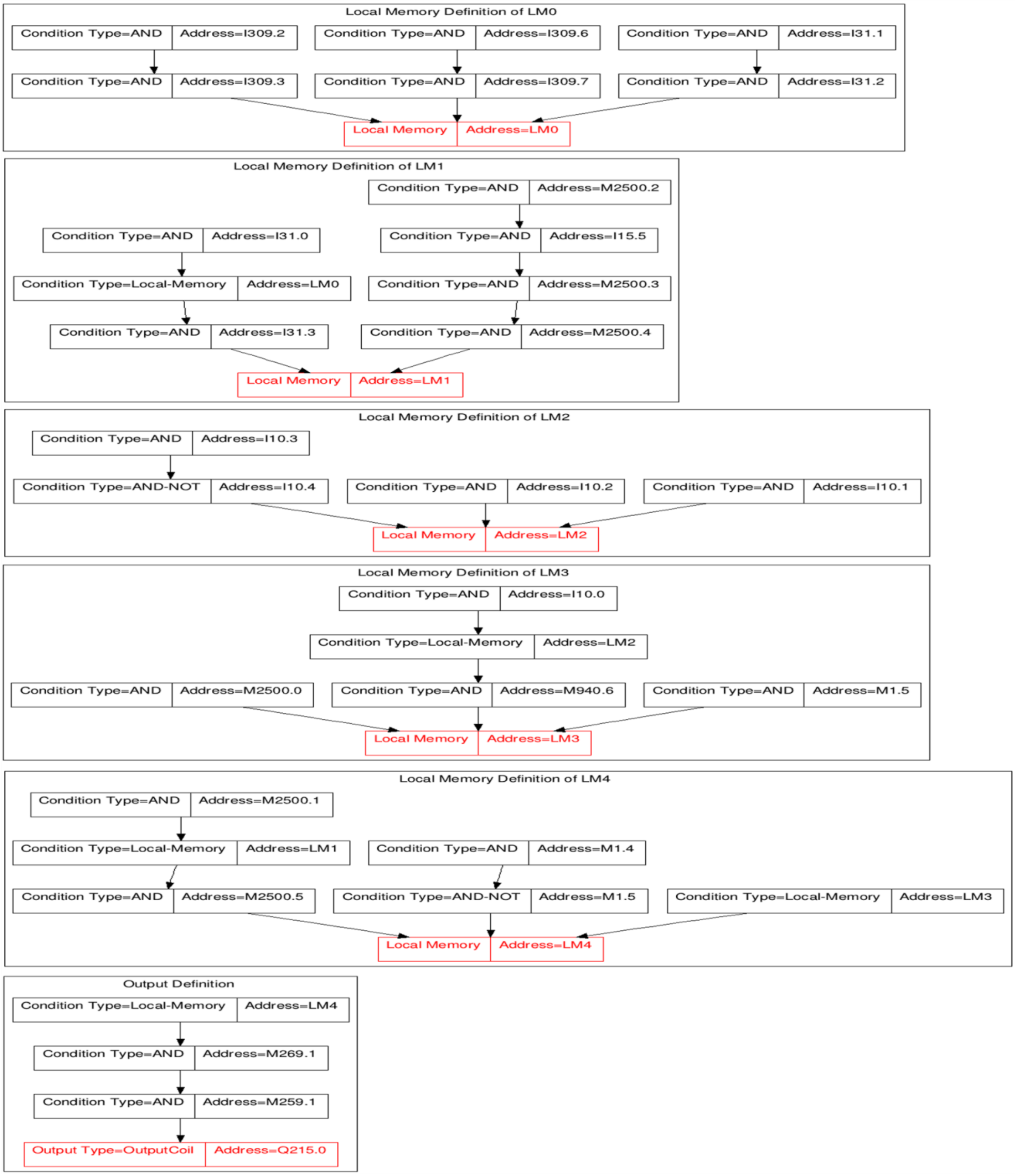
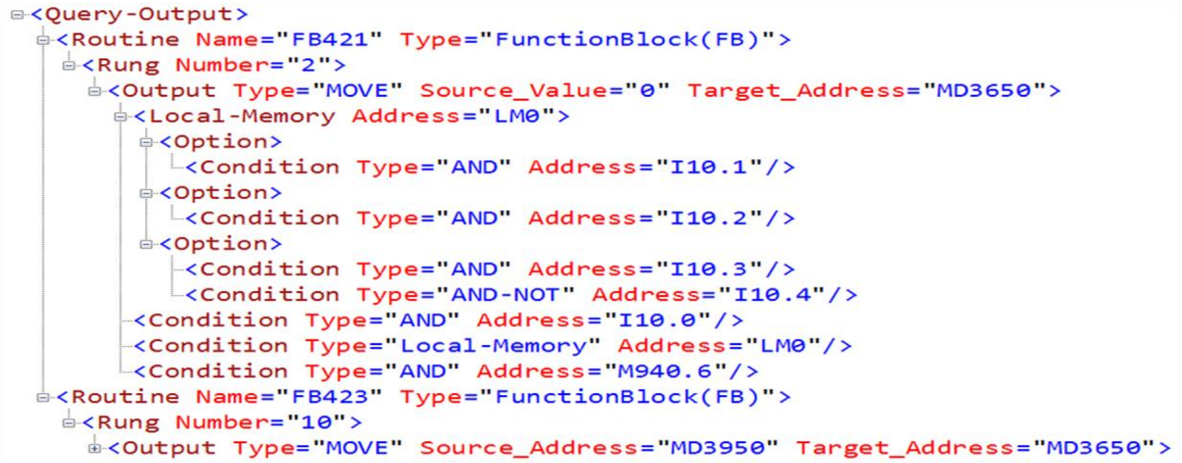

Local Memory Definition of LM0
Condition Type=AND | Address=I309.2
Condition Type=AND | Address=I309.6
Condition Type=AND | Address=I31.1
Condition Type=AND | Address=I309.3
Condition Type=AND | Address=I309.7
Condition Type=AND | Address=I31.2
Local Memory | Address=LM0
Local Memory Definition of LM1
Condition Type=AND | Address=M2500.2
Condition Type=AND | Address=I31.0
Condition Type=AND | Address=I15.5
Condition Type=Local-Memory | Address=LM0
Condition Type=AND | Address=M2500.3
Condition Type=AND | Address=I31.3
Condition Type=AND | Address=M2500.4
Local Memory | Address=LM1
Local Memory Definition of LM2
Condition Type=AND | Address=I10.3
Condition Type=AND-NOT | Address=I10.4
Condition Type=AND | Address=I10.2
Condition Type=AND | Address=I10.1
Local Memory | Address=LM2
Local Memory Definition of LM3
Condition Type=AND | Address=I10.0
Condition Type=Local-Memory | Address=LM2
Condition Type=AND | Address=M2500.0
Condition Type=AND | Address=M940.6
Condition Type=AND | Address=M1.5
Local Memory | Address=LM3
Local Memory Definition of LM4
Condition Type=AND | Address=M2500.1
Condition Type=Local-Memory | Address=LM1
Condition Type=AND | Address=M1.4
Condition Type=AND | Address=M2500.5
Condition Type=AND-NOT | Address=M1.5
Condition Type=Local-Memory | Address=LM3
Local Memory | Address=LM4
Output Definition
Condition Type=Local-Memory | Address=LM4
Condition Type=AND | Address=M269.1
Condition Type=AND | Address=M259.1
Output Type=OutputCoil | Address=Q215.0

Figure 7: The state-based graphical representation of the rung logic corresponding to the first Output Coil instruction.

separately in Figure 6 (b), Figure 6 (c), Figure 6 (d), Figure 6 (e) and Figure 6 (f), respectively. As can be seen in those figures, under the Local-Memory node, the definition (or the rung logic) of the corresponding local memory is given. The Address attribute of the Local-Memory node represents the virtual address (or name) of the local memory. It is easy to see from Figure 6, the definitions of the local memories characterize the rung logic of exactly the same branches (or the sub-logic blocks) as depicted in Figure 5. For example, the definition of local memory LM0 (presented in Figure 6 (b)) covers the rung logic of the highest depth parallel sub-branches of the rung diagram of Figure 5 (see the red colour box). As can be seen in Figure 6 (b), the condition set of each parallel branches are presented under a separate Option node. The Option nodes basically represent the OR boolean logic operations. So, if the condition set under any Option nodes associated with a particular local memory is true, then the RLO value stored in that local memory is also true. In the same way, the local memory LM1 is defined (shown in Figure 6 (c)). Please note that the definition of local memory LM1 is characterized by using the definition of



```
<Query-Output>
  <Routine Name="FB421" Type="FunctionBlock(FB)">
    <Rung Number="2">
      <Output Type="MOVE" Source_Value="0" Target_Address="MD3650">
        <Local-Memory Address="LM0">
          <Option>
            <Condition Type="AND" Address="I10.1"/>
          <Option>
            <Condition Type="AND" Address="I10.2"/>
          <Option>
            <Condition Type="AND" Address="I10.3"/>
            <Condition Type="AND-NOT" Address="I10.4"/>
        <Condition Type="AND" Address="I10.0"/>
        <Condition Type="Local-Memory" Address="LM0"/>
        <Condition Type="AND" Address="M940.6"/>
  <Routine Name="FB423" Type="FunctionBlock(FB)">
    <Rung Number="10">
      <Output Type="MOVE" Source_Address="MD3950" Target_Address="MD3650">
```

Figure 8: The format of the query output XML file (retrieved as a result of the query input: address MD3650).

local memory LM0 (in a modular fashion). The RLO value saved in the local memory LM1 can easily be determined by performing an AND boolean logic operation between the RLO value saved in the local memory LM0 and the RLO value of the other conditions (for simplicity, we can assume that the Local-Memory type attribute represents the AND boolean logic operation). It is easy to realize, this bottom-up hierarchical logic decomposition procedure provides an easy, systematic, step-by-step interpretation of the rung logic to the users. As can be seen in Figure 6 (a), the overall rung logic corresponding to the first Output Coil instruction is characterized by using only a very few conditions (the same is also true for the LMDs – see Figure 5 and Figure 6). This indeed makes the program logic behind an output easier to follow.

The rung diagram connected with a particular output can have many same-depth parallel branches and sub-branches. It is easy to perceive, if a large number of parallel branches or sub-branches are characterized by using a single local memory, then it can generate a very complex local memory definition. In order to avoid such issues, POST allows the users to explicitly bind the complexity of the LMDs through restricting (or setting) the number of parallel branches that the definition can cover. For example, as can be seen from Figure 5 and Figure 6, the LMDs are always restricted to maximum three parallel branches. We have also developed a software interface module (with the help of Graphviz Software [22]) that can provide an efficient graphical representation of the rung logic. As an example, in Figure 7, the graphical representation of the rung logic associated with the first Output Coil instruction is shown. It is actually the state-based graphical representation of the XML file presented in Figure 6. A state or a node in the graph of Figure 7 actually represents a particular condition (in other words, indicates a program instruction and the associated addresses or data values). Please note that the RLO values corresponding to the state nodes of a particular path are needed to be ANDed in order to get the resultant RLO value of that path; and the RLO values of the different paths are needed to be ORed in order to determine the resultant RLO value of the corresponding sub-logic block (see Figure 5 and Figure 7).

4.3 Program Error Analysis Procedure

The program output instructions and their corresponding conditions based output XML file format not only provides an efficient program logic interpretation, but also makes it possible for a software module to accumulate all the conditions corresponding to an output automatically. If an incorrect data value is found in any output address, then the user has to pass that address (or any other output address) as the query input value to the condition search engine of POST. The condition search engine of POST analyzes the output address attribute values of all the Output nodes of the above stated XML file (as shown in Figure 4 and Figure 6), and generates a query output XML file that contains all the conditions (i.e., the program instructions and the associated addresses or data values) that can directly affect the value stored at that particular input address. Recall that the output address attribute of the Output nodes refers to the attribute that denotes the output address of the corresponding program instruction. For example, the Target_Address attribute is the output address attribute of the Move type instructions (see Figure 3 and Figure 4).

For the convenience of the readers, an example query output XML file is presented in Figure 8. The condition search engine of POST produced that XML file for the query input address MD3650 (see Figure 3 and Figure 4). As we can see, under the root node i.e. Query-Output node, the Routine and the Rung nodes are defined. It helps the users to identify the routine and the rung in which the output instruction is declared. The Output nodes and their corresponding Local-Memory and Condition nodes help the users to explicitly determine the output instructions and the conditions that can affect the value stored in that output address. As can be seen in Figure 8, two Move type output instructions (and their corresponding condition sets) can directly alter the value stored in the query input address MD3650. The first Move instruction is located in the second rung of the function block FB421 (shown in Figure 3 – see the second branch of the rung diagram), and the second Move instruction is located in the tenth rung of the function block FB423 (not illustrated for the space

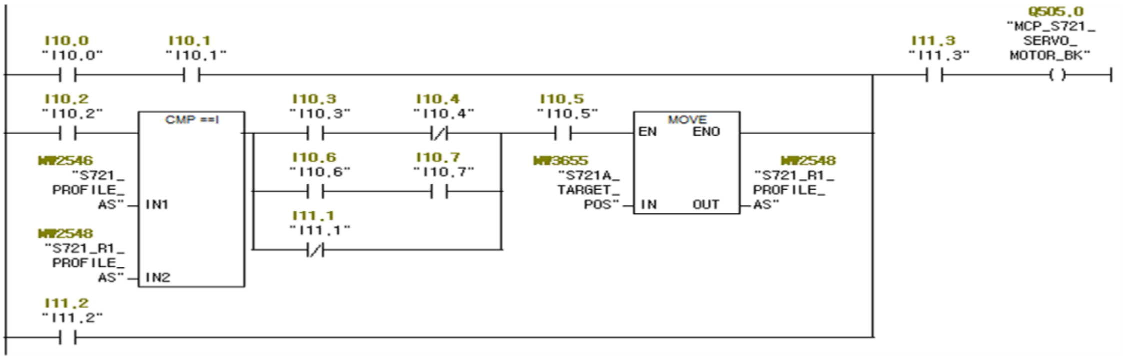


Figure 9: A simple example ladder rung diagram.

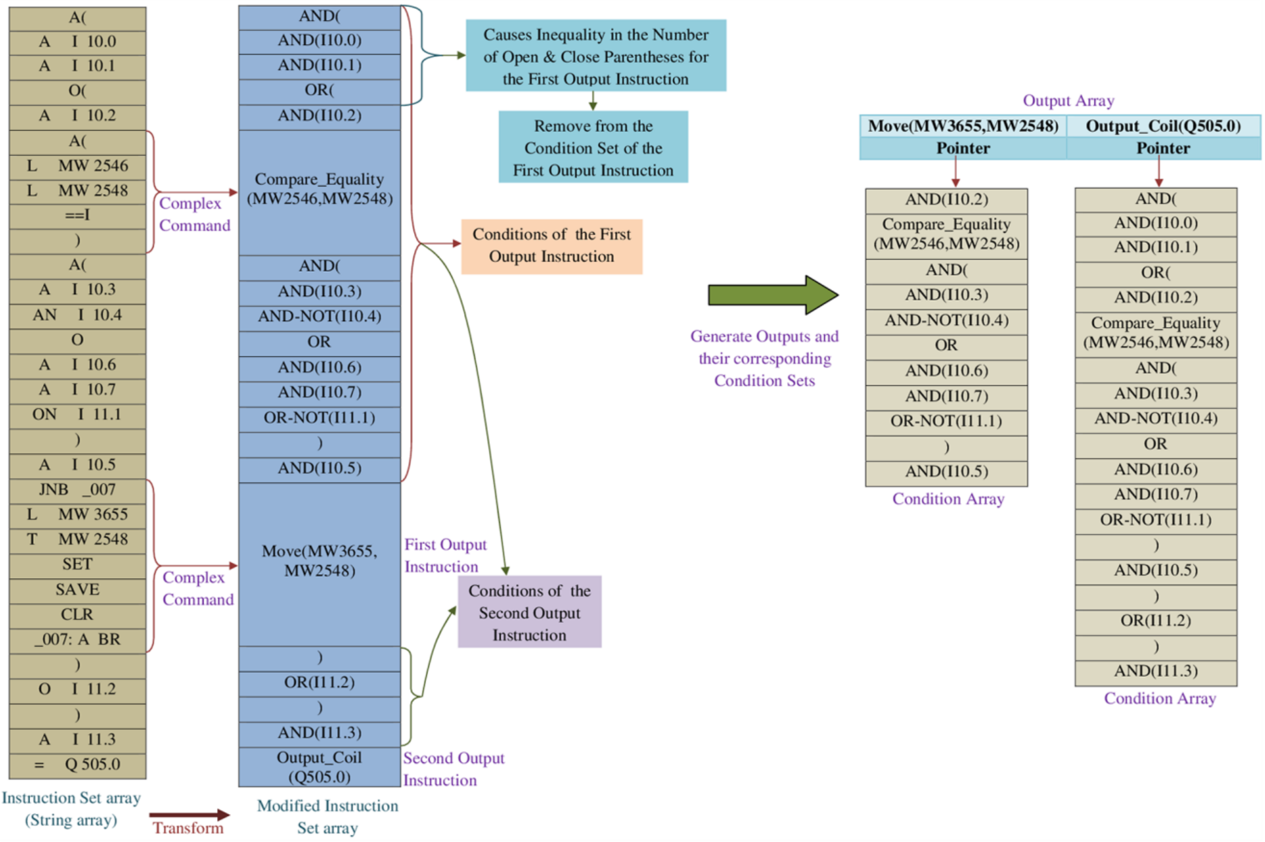


Figure 10: Step 1 – Identifying the program output instructions and their corresponding condition sets.

reasons). The users then have to thoroughly inspect these condition sets to find out: i) if any sensor has failed or is transmitting an inaccurate reading; and ii) if there is any logical or conceptual flaw in the rung diagram (i.e., the output instructions and their corresponding conditions). If not then the users have to search again for the conditions corresponding to the output address (or addresses) from where an erroneous value is obtained as the input (please note that this address must have to the output address of an instruction that belongs to the above stated condition sets). This process continues until the actual cause of the error is detected (as mentioned in Section 2). It is easy to realize, POST makes this entire condition search process automatic and oversight easy; and hence, the program error analysis process becomes simple and fast. Please note that this becomes possible only because of the program output instructions and their corresponding conditions based source code transformation approach of POST.

4.4 Implementation Details of POST Approach

We have implemented POST approach in C++ language and tested it on the program source codes of Siemens and Allen-Bradley PLC software. POST takes the PLC program source code file saved in the IL language format as the input, and converts it into the above stated XML file format (as discussed in Subsection 4.1 and Subsection 4.2). Whenever the starting tag of a routine (respectively, rung) is encountered in the program source code file, POST enters the name and type (respectively, number) information of that routine (respectively, rung) into the output XML file, following the same format as shown in Figure 4. Similarly, when the ending tag is detected, it

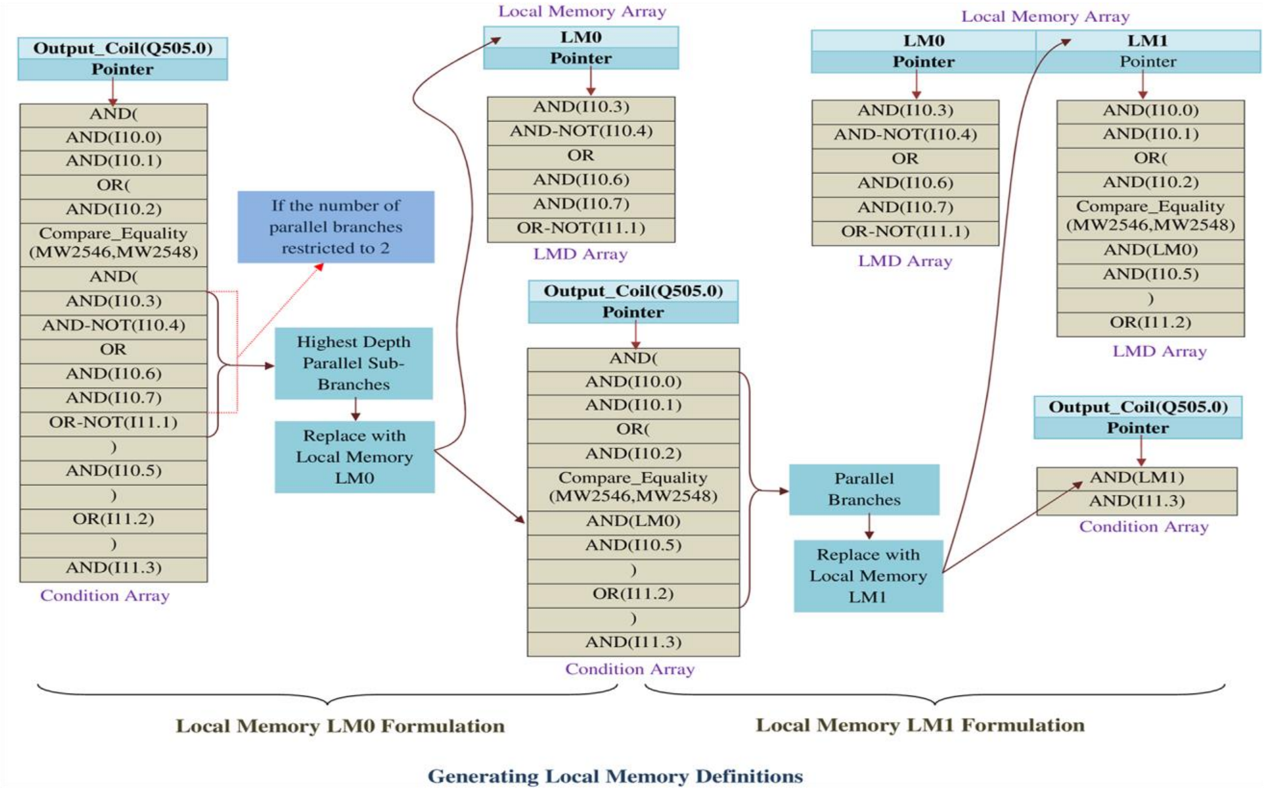
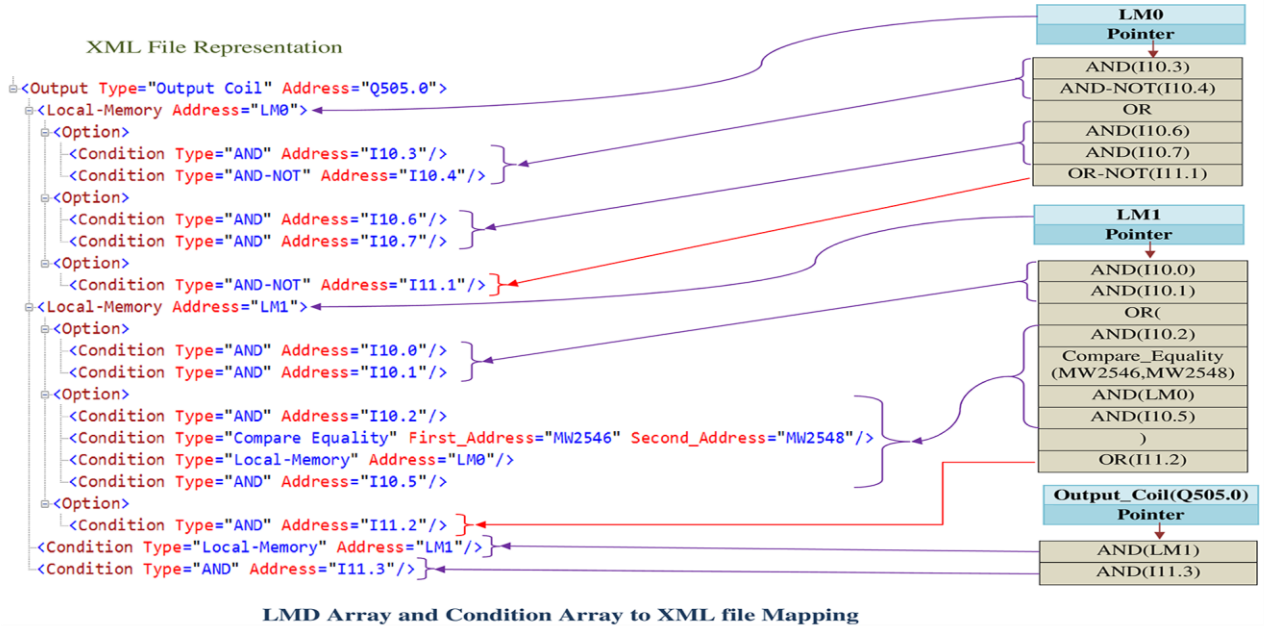


Figure 11: Step 2 – Formulating the local memory definitions.



```
<Output Type="Output Coil" Address="Q505.0">
  <Local-Memory Address="LM0">
    <Option>
      <Condition Type="AND" Address="I10.3"/>
      <Condition Type="AND-NOT" Address="I10.4"/>
    <Option>
      <Condition Type="AND" Address="I10.6"/>
      <Condition Type="AND" Address="I10.7"/>
    <Option>
      <Condition Type="AND-NOT" Address="I11.1"/>
  <Local-Memory Address="LM1">
    <Option>
      <Condition Type="AND" Address="I10.0"/>
      <Condition Type="AND" Address="I10.1"/>
    <Option>
      <Condition Type="AND" Address="I10.2"/>
      <Condition Type="Compare Equality" First_Address="MW2546" Second_Address="MW2548"/>
      <Condition Type="Local-Memory" Address="LM0"/>
      <Condition Type="AND" Address="I10.5"/>
    <Option>
      <Condition Type="AND" Address="I11.2"/>
  <Condition Type="Local-Memory" Address="LM1"/>
  <Condition Type="AND" Address="I11.3"/>
```

Figure 12: Step 3 – Transforming the results into the output XML file format.

closes the corresponding XML file node. The rung logic defined inside the starting and ending tag of a rung is copied into the computer memory for further processing. The rung logic (or IL code) to XML file conversion is a three-phase procedure. We discuss this three-phase procedure with the help of a simple ladder rung given in Figure 9.

The first phase of the above stated three-phase source code transformation process is illustrated in Figure 10. As we can see, a string array, named Instruction Set array is used to store the source code of the ladder rung presented in Figure 9. In the first phase, as can be seen in Figure 10, the output instructions and their corresponding condition sets are determined. In order to accomplish this, at first, the program instructions stored in the Instruction Set array are converted into the specific instruction name format of POST (as discussed in Subsection 4.1 and Subsection 4.2). As can be seen in Figure 10, the IL language instructions are converted into the descriptive language instructions and are stored in a string array, called the Modified

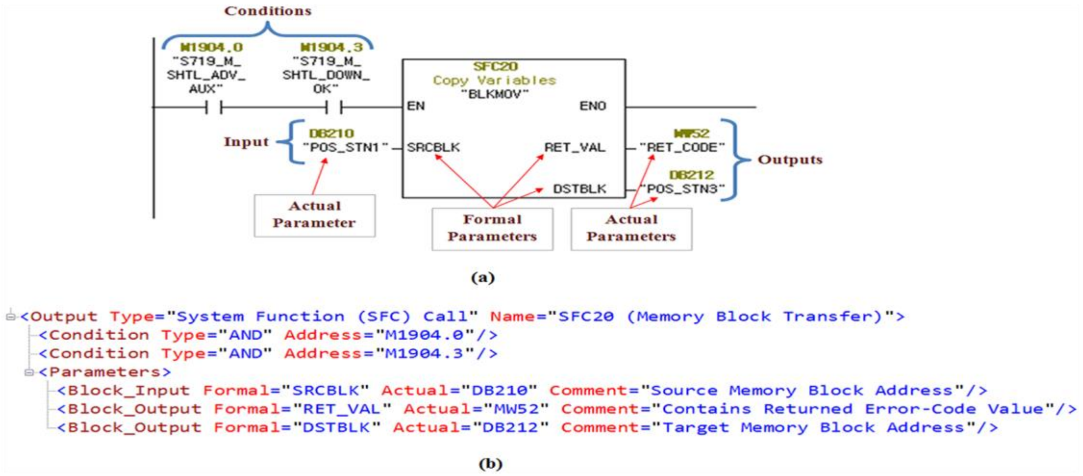
Instruction Set array. In case of the source code of Siemens PLC software, the complex (or compound) instructions are decomposed into the core or basic instructions. For example, a Move type instruction is decomposed into the L, T, (optional) JNB instructions etc. [20, 21] (also see Figure 2). For this reason, POST has to inspect whether a set of core instructions in the Instruction Set array is actually equivalent to any such complex program instruction or not. If so, then POST replaces that instruction set with its corresponding descriptive language instruction and stores it in the Modified Instruction Set array (see Figure 10). The same measure is also taken for the instructions that have multiple inputs and outputs (such as the block call instructions – we will discuss more on it later). This sub-phase is skipped for the source code of Allen-Bradley PLC software. This is because, in that case, the complex instructions are not broken down into the core or basic instructions.

In the next sub-phase of the first phase, the outputs and their corresponding condition sets are formulated from the Modified Instruction Set array. As we can see in Figure 10, POST first identifies all the output instructions in the Modified Instruction Set array, and then copies them into another array, named the Output Array. The condition set corresponding to each output is stored in a separate array (by using a pointer), named the Condition Array. As can be seen in Figure 10, all the instructions (except the output instructions) prior to an output instruction form the condition set corresponding to that output instruction, and are stored in the corresponding Condition Array. However, this axiom is not correct for the output instructions that are declared in a parallel branch or a sub-branch. For example, as can be seen in Figure 9, the Move output instruction is declared in a parallel branch of the main branch. For this reason, the conditions declared in its previous branches that appear prior to it in the Modified Instruction Set array, cannot be considered as its corresponding conditions. As can be seen from Figure 10, in this case, we get an inequality in the number of open and close parentheses (three open and one close parentheses). So, all the conditions up to the second open parenthesis are needed to be eliminated in order to get the equality in the number of open and close parentheses (in other words, in order to obtain the actual condition set). As can be seen in Figure 10, after performing this elimination operation, the condition set corresponding to an output is stored in its corresponding Condition Array for further processing.

In the second phase, POST further simplifies the program logic stored in each Condition Array by formulating the LMDs following the same procedure as discussed in Subsection 4.2. This phase in details is illustrated in Figure 11 (for the interest of space, the LMD formulation procedure is shown only for the Output Coil instruction). If the Condition Array does not hold any OR instruction, then this phase is skipped. As can be seen in Figure 11, POST first identifies the condition set associated with the highest depth parallel branches present in the Condition Array (the branch depth can easily be calculated from the number of open and close parentheses); and then replaces it with a new local memory definition. Recall from Subsection 4.2 that the RLO value stored in the local memory address associated with a LMD is needed to be ANDed with the RLO value of the other conditions present in the Condition Array in order to get the resultant RLO value of the Condition Array. It is easy to see, the exact same principle is followed in Figure 11. As can be seen in Figure 11, the local memory addresses are saved in a separate array, called the Local Memory Array and in the second dimension of that array, a pointer to another array, called the LMD Array is stored. In the LMD Array, the definition (or the condition set) of its corresponding local memory is stored. This process is repeated until all the parallel branches of the main branch are defined by using the local memories (in other words, until all the OR instructions are eliminated from the Condition Array – see Subsection 4.2). Recall that in POST, the LMDs can be restricted to a limited number of parallel branches. If we set that number to two (implies that at most one OR instruction can be present in a particular LMD Array), then only the conditions inside the red colour arrows (see Figure 11) are characterized by using the local memory LM0 (and so on).

In the third phase of the source code transformation process, the outputs and their corresponding condition sets are mapped into the output XML file. This phase is depicted in Figure 12. As can be seen, the condition sets associated with the local memories and the outputs are respectively written into the XML file following the same format as discussed throughout Subsection 4.1 and Subsection 4.2. In the output XML file, the OR instructions in the LMD Arrays are represented by using the Option nodes (as stated earlier). However, as can be seen in Figure 12, an OR (respectively, OR-NOT) instruction with an address argument is characterized by using the Condition node that has the Type attribute value AND (respectively, AND-NOT) and is defined under a separate Option node (this type of mapping is shown by using the red colour arrows). This is because, in the program source code of Siemens PLC software, an OR (respectively, OR-NOT) instruction with an address argument actually represents a parallel branch where only one AND (respectively, AND-NOT) instruction is declared (see Figure 9 and Figure 12). In the output XML file, POST always keeps the exact same logic structure as in the original PLC program (as stated earlier). Also note that in the output XML file, a Local-Memory type condition basically indicates an AND boolean logic operation and hence, the corresponding RLO value is needed to be ANDed with the RLO value of the other conditions in order to determine the resultant RLO value of the corresponding sub-logic block.

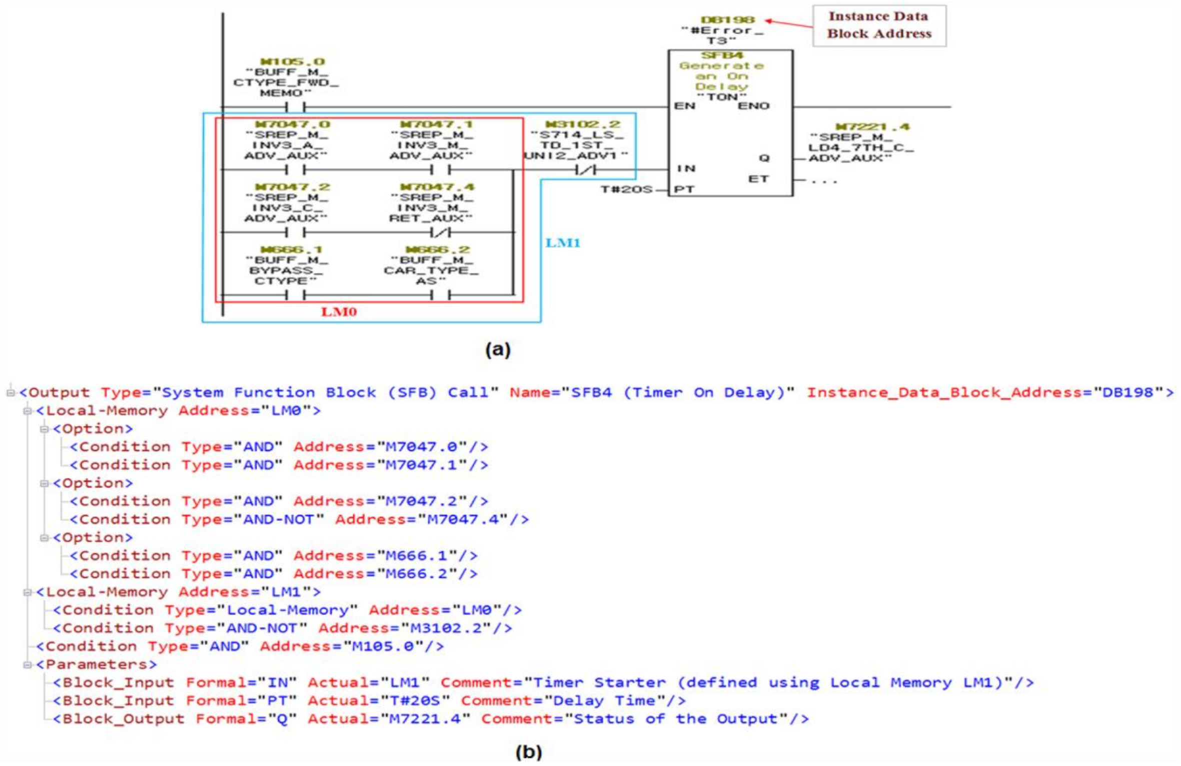
In the above, we have discussed the implementation details of POST based on Siemens and Allen-Bradley PLC software. However, the proposed three-phase procedure is logically applicable to the source code of other PLC software as well (a little modification may be needed based on that particular software).



```
<Output Type="System Function (SFC) Call" Name="SFC20 (Memory Block Transfer)">
  <Condition Type="AND" Address="M1904.0"/>
  <Condition Type="AND" Address="M1904.3"/>
  <Parameters>
    <Block_Input Formal="SRCBLK" Actual="DB210" Comment="Source Memory Block Address"/>
    <Block_Output Formal="RET_VAL" Actual="MW52" Comment="Contains Returned Error-Code Value"/>
    <Block_Output Formal="DSTBLK" Actual="DB212" Comment="Target Memory Block Address"/>
```

(b)

Figure 13: The output XML file format for the block call instructions. (a) The calling or invocation of the System Function SFC20. (b) The output XML file format for SFC20 block.



```
<Output Type="System Function Block (SFB) Call" Name="SFB4 (Timer On Delay)" Instance_Data_Block_Address="DB198">
  <Local-Memory Address="LM0">
    <Option>
      <Condition Type="AND" Address="M7047.0"/>
      <Condition Type="AND" Address="M7047.1"/>
    <Option>
      <Condition Type="AND" Address="M7047.2"/>
      <Condition Type="AND-NOT" Address="M7047.4"/>
    <Option>
      <Condition Type="AND" Address="M666.1"/>
      <Condition Type="AND" Address="M666.2"/>
  <Local-Memory Address="LM1">
    <Condition Type="Local-Memory" Address="LM0"/>
    <Condition Type="AND-NOT" Address="M3102.2"/>
  <Condition Type="AND" Address="M105.0"/>
  <Parameters>
    <Block_Input Formal="IN" Actual="LM1" Comment="Timer Starter (defined using Local Memory LM1)"/>
    <Block_Input Formal="PT" Actual="T#20S" Comment="Delay Time"/>
    <Block_Output Formal="Q" Actual="M7221.4" Comment="Status of the Output"/>
```

(b)

Figure 14: An example of a block call instruction with an input parameter that is defined based on a set of conditions. (a) The calling or invocation of the System Function Block SFB4. (b) The output XML file format for SFB4 block.

4.5 Dealing with the Block Call Instructions

In this subsection, we discuss the output XML file format for a special instruction i.e., the block call instruction. The block call instructions are used to call (or invoke) the program blocks, such as the FCs, FBs, System FCs (SFCs), System FBs (SFBs) etc. [20, 21, 23]. Unlike other instructions, the block call instructions have two types of parameters, namely the formal and the actual parameter. In addition, a program block can have multiple inputs and outputs. An example of such program block call is presented in Figure 13 (a) [the SFC20 block is used to copy the contents of a memory area given at input SRCBLK to another memory area given at output DSTBLK. If an error occurs, the returned error code is stored at the address given at output RET_VAL.]. As can be seen in Figure 13 (a), the SFC20 block has only one input and two outputs. However, some program blocks can have dozens of inputs and outputs. If POST characterizes all the inputs and outputs associated with a program block inside the Output node, it may generate a very complex XML node structure. In addition, POST has to define both the formal and actual parameters inside the Output node. In order to overcome this type of issues, under the Output node, an additional XML file node, called the Parameters node is created, inside which all the information related to the parameters of a program block is put together. In Figure 13 (b), the output XML file format for SFC20 block is shown. As we can see, only the name and type information of the block call instruction is specified inside the Output node. The corresponding condition set is defined under the Output node following the same way as done before. As can be seen in Figure 13 (b), under Parameters node, all the parameters of SFC20 block are characterized. The Block_Input and the Block_Output nodes are created to define the inputs and the outputs of the program block. The Formal and the Actual attributes represent the formal and the actual parameters, respectively (the Actual attribute of the Block_Output node is the output address attribute of the block call instruction – also see Subsection 4.3). Please note that inside the Block_Input and the Block_Output nodes, an additional attribute i.e., the Comment attribute is incorporated in order to define the objectives of the parameters. However, the Comment attribute is an optional attribute and is generated only for the system library blocks (since the objectives of the parameters are known in advance).

Another example of a program block call (a SFB4 block call) and its corresponding XML file format are shown in Figure 14 (a) and Figure 14 (b), respectively. As can be seen in Figure 14 (b), the output XML file follows exactly the same structure as discussed above. In the Output node, an additional attribute i.e., the Instance_Data_Block_Address attribute is included thus the address of the corresponding instance data block can be incorporated (for more information, see [21] and [23]). However, this change is instruction specific (recall that POST determines the format of the XML node according to the functional specification of the corresponding instruction). A distinctive feature of this SFB4 block is that it has an input i.e., the input IN which is defined on the basis of a set of conditions (see Figure 14 (a)). Actually, at the time of program execution, the RLO value of the condition set is passed as the input IN value to the SFB4 block. If we define all these conditions inside the corresponding Block_Input node, then it generates a very complex XML node structure. In order to avoid this sort of problems, POST utilizes the concept of the local memory definitions. The local memory definitions are used to characterize the inputs that are defined on the basis of multiple conditions (following the same way as discussed in Subsection 4.2). As can be seen from Figure 14 (a) and Figure 14 (b), the complete condition set corresponding to the actual parameter of input IN is defined by using the local memory LM1. Please note that the parallel branches of the main branch are characterized by using the local memory LM0 following exactly the same procedure as described in Subsection 4.2.

It is easy to perceive from the above discussions:

- the output XML file format is designed very carefully in such a way that the condition search engine of POST can accumulate all the conditions associated with a program output automatically and in a straightforward way
- the rung logic associated with a program output is simplified further whenever it gets complicated (in other words, whenever the number of parallel branches and sub-branches exceeds a certain limit)
- each type of node in the output XML file is designed keeping in mind the objective and the functional specification of the corresponding instruction

These features of the output XML file indeed make the programming error analysis task simple, fast and oversight easy (because, there is no need to inspect each rung of every program blocks manually). In addition, the above discussed XML file format provides an easy, systematic and step-by-step interpretation of the program logic to the users which makes the error analysis task even more simpler.

5 Conclusion

This work is motivated by the need of an approach that can help the users to understand the PLC programs easily, and can assist them to analyze the programming errors in an efficient manner. In this paper, we have proposed a new approach, called POST that can satisfy all the mentioned needs effectively. POST takes the PLC program source code file as the input, and converts it into a program output instruction and its corresponding conditions based well-structured XML file. In the XML file, the rung logic corresponding to an output is further simplified by using a novel local memory based technique, and is presented in a programming language and platform independent format. The proposed XML file format provides a systematic and step-by-step interpretation (in a bottom-up fashion) of the program logic to the users. In addition, the XML file format is designed in such a way that the condition search engine of POST can accumulate all the conditions that can affect the value stored at a given output address automatically. These features of POST indeed help the users to identify the actual cause of a programming error quickly and reliably. A software interface module has also been developed in order to provide an efficient state-based graphical representation of the rung logic to the users.

Acknowledgement

The authors wish to thank UDMTEK Co., Ltd. for supplying all the required software, tools and PLC programs used in this research work. This work was supported in part by the Ministry of Trade, Industry and Energy (MOTIE), Republic of Korea, under Grant 10051146 and 10065737; in the part by the Small and Medium Business Administration, Republic of Korea, under Grant S2408982; and in part by the MOTIE and the Korea Institute for Advancement of Technology, Republic of Korea, under Grant N0001083.

References

[1] Liu, J. & Darabi. H. (2002). Ladder Logic Implementation of Ramadge-Wonham Supervisory Controller. *Proceedings of the 6th International Workshop on Discrete Event Systems (WODES’02)*, Zaragoza, Spain, pp. 383–389.

[2] Du, D., Liu, Y., Guo, X., Yamazaki, K., & Fujishima, M. (2009). Study on LD-VHDL conversion for FPGA-based PLC implementation. *The International Journal of Advanced Manufacturing Technology*, vol. 40, no. 11-12, pp. 1181–1190.

[3] Bani Younis, M. & Frey, G. (2004). Visualization of PLC Programs Using XML. *Proceedings of the American Control Conference (ACC'04)*, Boston, USA, pp. 3082–3087.

[4] Siemens Simatic Step S7 Software. Website: http://w3.siemens.com/mcms/simatic-controller-software/en/step7/pages/default.aspx, last retrieved on 6th July, 2017.

[5] Allen-Bradley RSLogix Software. Website: http://www.rockwellautomation.com/rockwellsoftware/products/rslogix.page, last retrieved on 6th July, 2017.

[6] Fen, G. & Ning, W. (2006). A Transformation Algorithm of Ladder Diagram into Instruction List Based on AOV Digraph and Binary Tree. *Proceedings of the IEEE Region 10 International Conference (TENCON'06)*, Hong Kong, China, pp. 1–4.

[7] Hu, F., Fu, L., Liu, L., & Zhang, G. (2008). An Algorithm about Transforming PLC Ladder Diagram to Instruction List Based on Series-Parallel Merging Method. *Proceedings of the Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA'08)*, Wuhan, China, pp. 812–816.

[8] Tan, A. & Ju, C. (2011). The Application of Maze algorithm in Translating Ladder Diagram into Instruction Lists of Programmable Logical Controller. *Procedia Engineering*, vol. 15, no. 1, pp. 264–268.

[9] Yan, Y. & Zhang, H. (2010). Compiling ladder diagram into instruction list to comply with IEC 61131-3. *Computers in Industry*, vol. 61, no. 5, pp. 448–462.

[10] Huang, L., Liu, W., & Liu, Z. (2009). Algorithm of transformation from PLC ladder diagram to structured text. *Proceedings of the 9th International Conference on Electronic Measurement & Instruments (ICEMI'09)*, Beijing, China, pp. 4-778–4-782.

[11] Estevez, E., Marcos, M., Iriondo, N., & Orive, D. (2007). Graphical Modelling of PLC-based Industrial Control Applications. *Proceedings of the 26th American Control Conference (ACC'07)*, New York City, USA, pp. 220–225.

[12] Estevez, E., Marcos, M., Orive, D., Irisarri, E., & Lopez, F. (2007). XML based Visualization of the IEC 61131-3 Graphical Languages. *Proceedings of the 5th IEEE International Conference on Industrial Informatics (INDIN'07)*, Vienna, Austria, pp. 279–284.

[13] Estevez, E., Marcos, M., Irisarri, E., Lopez, F., Sarachaga, I., & Burgos, A. (2008). A novel Approach to attain the true reusability of the code between different PLC programming Tools. *Proceedings of the 7th IEEE International Workshop on Factory Communication Systems (WFCS'08)*, Dresden, Germany, pp. 315–322.

[14] Estevez, E., Marcos, M., Orive, D., Lopez, F., Irisarri, E., & Perez, F. (2008). Middleware based on XML technologies for achieving true interoperability between PLC programming tools. *Proceedings of the 17th World Congress of the International Federation of Automatic Control (IFAC'08)*, Seoul, Republic of Korea, pp. 8461–8466.

[15] Marcos, M., Estevez, E., Perez, F., & Der Wal, E. (2009). XML exchange of control programs. *IEEE Industrial Electronics Magazine*, vol. 3, no. 4, pp. 32–35.

[16] Lopez, F., Irisarri, E., Estevez, E., & Marcos, M. (2008). Graphical representation of factory automation Markup Languages. *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'08)*, Hamburg, Germany, pp. 29–32.

[17] Frey, G. & Bani Younis, M. (2004). A Re-Engineering Approach for PLC Programs using Finite Automata and UML. *Proceedings of the IEEE International Conference on Information Reuse and Integration (IRI'04)*, Las Vegas, USA, pp. 24–29.

[18] Bani Younis, M. & Frey, G. (2005). Formalization and Visualization of Non-binary PLC Programs. *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC'05)*, Seville, Spain, pp. 8367–8372.

[19] Bani Younis, M. & Frey, G. (2006). UML-based approach for the re-engineering of PLC programs. *Proceedings of the 32nd Annual Conference on IEEE Industrial Electronics (IECON'06)*, Paris, France, pp. 3691–3696.

[20] Siemens Simatic Ladder Logic (LAD) for S7-300 and S7-400 Programming (Reference Manual). URL: https://cache.industry.siemens.com/dl/files/822/45523822/att_82001/v1/s7kop__b.pdf, last retrieved on 6th July, 2017.

[21] Siemens Simatic Statement List (STL) for S7-300 and S7-400 Programming (Reference Manual). URL: https://cache.industry.siemens.com/dl/files/446/45523446/att_79269/v1/s7awl__b.pdf, last retrieved on 6th July, 2017.

[22] Graphviz Software. Website: http://www.graphviz.org, last retrieved on 6th July, 2017.

[23] Siemens Simatic System Software for S7-300/400 System and Standard Functions (Volume 1/2, Reference Manual). URL: https://cache.industry.siemens.com/dl/files/574/1214574/att_44504/v1/SFC_e.pdf, last retrieved on 6th July, 2017.

An Effective Meta-Heuristic Cuckoo Search Algorithm for Test Suite Optimization

Manju Khari
Department of Computer Engineering,
Ambedkar Institute Of Advanced Communication Technologies and Research, Delhi, India
E-mail: manjukhari@yahoo.co.in

Prabhat Kumar
Department of Computer Engineering,
National Institute of Technology Patna, Bihar, India
E-mail: prabhat@nitp.ac.in

Keywords: optimization, nature inspired algorithm, cuckoo search, test suite, test data

Received: March 12, 2016

Automation testing is the process of generating test data without any human interventions. In recent times, nature-inspired solutions are planned, tested and validated successfully in many areas for the purpose of optimization. One such meta heuristic technique is Cuckoo Algorithm (CA) that receives its sole inspiration from the behavior of cuckoo, who has the ability to resolve complex issues using simple initial conditions and limited knowledge of the search space. This paper presents a cost effective and time efficient algorithm inspired from cuckoo for optimizing the test data. On comparing the proposed algorithm with existing Firefly Algorithm (FA) and Hill Climbing (HC) algorithms, it was found that CA outperforms both FA and HC in terms of the test data optimization process. The work done in the current study would be helpful to testers in generating optimized test data which would result in saving of both testing cost and time.

Povzetek: V prispevku je razvita izpopolnjena meta-hevristična metoda kukavičjega algoritma (Cockoo algorithm), ki temelji na reševanju zapletenih problemov z reševanjem več lokalnih preprostih.

1 Introduction

Software testing is done with the intention of finding bugs and enhancing the quality before delivering it to the client [1]. As testing is very tedious and time consuming, it is highly desirable that this cost be controlled as much as possible [2]. One way to control this cost is to reduce the data which is used for testing. Test data optimization is a process of reducing the test data sets and it can be successfully applied to black box as well as white box testing [3].

Optimization mainly focuses on reducing the number of available solutions depending upon certain parameters. Merely reducing the number of test data does not accomplish the purpose of software testing. The test data should effectively uncover all potential lapses that exist in the software or product. For carrying out test data optimization, mathematical functions are applied that can effectively filter out the required test data. The methods of finding the optimal solutions among the various available options can be regarded as the optimization problem [20]. This problem aims at maximizing or minimizing a real value function from the allowed set of solutions.

With the help of optimization, we can filter out the fittest test data that can easily test various properties related to a software or product. Generally, test data optimization is carried out with the help of various meta-heuristic algorithms. They offer better solutions in comparison to the traditional algorithm which finds a solution based on hit and trial method. These algorithms have two components, exploration (intensification) and exploitation (diversification). Exploration means to search solution at a global scale while exploitation aims at finding a good solution in a local region. The combination of these two ensures a global optimal solution which is achievable.

This paper deals with implementation of a meta-heuristic CA for optimizing test data. In the year 2009, the behavior of a cuckoo was initially modeled by Yang [25]. Since then CA has been successfully used for solving various optimization problems [26][27][28][31][33][35][37], but to the best of author's knowledge, no study has explored the application of CA for test data optimization at such a large scale. In order to reduce the testing efforts, CA is applied for test data generation and compared with existing traditional algorithms such as FA and HC. The algorithms were applied to 50 JAVA programs. Various parameters, such as time taken by each algorithm to perform optimization, were compared using a number of iterations. It was found that CA outperformed FA and HC in each and every aspect.

The rest of the paper is organized as follows: Section 2 describes the related work that has been done with context to CA, Section 3 discusses various concepts related to test data optimization, Section 4 explains the proposed methodology, Section 5 describes that has been collected for showing that proposed methodology outperformed FA and HC, Section 6 provides a detailed case study of the proposal along with the statistics applied to JAVA programs for all three algorithms, Section 7 describes the potential threats and precautions that need to be taken while implementing this algorithm and finally, Section 8 concludes the study with a discussion of the future scope.

2 Related work

Developments in the field of 'automated test data generation' were initiated in the early 70's. Articles like 'Testing large software with automated software evaluation systems' by Ramamoorthy et al., in the year 1975 [4] and 'Automatic generation of floating-point test data' by Miller et al., in the year 1976 [5], are a few examples of the early work in this field. Nevertheless, Clarke's examines [6] of the year 1976 is considered to be the first of its kind to propose an algorithm for automated test data generation which is written in FORTRAN. The automated test-data generators can be divided into three classes - random, static and dynamic. Random test data generation is easy to automate, but problematic [7][9]. Firstly, it produces a sample of the possible paths through the software under test (SUT). Secondly, it may be expensive to generate the expected output data for a large amount of input data produced. Finally, provided exceptions occur only rarely, the input domain which causes an exception is likely to be small. Random test-data generators may never hit on this small area of the input domain.

During 70's up to mid-80's, people did research on test data generation using symbolic execution. Researchers have been working on test data generation since 1975, but unfortunately, there is hardly any fully automated test data generation working tool available in the current software industry. At that time the language used for test data generation was FORTRAN. In the year 1987, Parther [7] proposed a new idea for test data generation called path prefix method. In the year 1990, Korel [8] provided a revolutionary change by generating test data dynamically based on actual value using pattern and explanatory search. In the year 1996, Ferguson [9] examined an assertion oriented and chaining approach i.e. goal-oriented test data generation [10]. Test data generation from dynamic data structure [5] has encouraged many authors to work on hybrid testing methods for detecting infeasible paths, thereby saving computational time. Mutation testing technique is used to improve the reliability of object-oriented software, authors understand the traditional mutation testing method and apply on object-oriented programs that are known as class mutation [12]. Xiao et al. [13] proposed a technique for test data generation even if more or less path predicate is unsolvable. However, the proposed method could not provide a good coverage.

A better way can be the combination of evolutionary methods with dynamic symbolic execution [15][18], which would wipe away the disadvantages of both the approaches. Search-based methods have previously been applied for testing object-oriented software by making use of method sequences [19][20] or by the help of strongly typed genetic programming [21][22]. While creating test data for object-oriented software, since the early work of Tonella [23], the authors experimented on issues of handling the techniques that can be used to reverse engineer several design views from the source code. These techniques aim at generating test suites to achieve high quality test data using mutation testing [16][46][47][48] deployed in a search-based test generation environment. Lakhotia et al. [24] established a search based multi-objective approach in which a random search, Pareto GA and weighted GA algorithm is used for branch coverage. Various algorithms were also proposed to optimize and prioritize the test suite. Evolutionary algorithms (like CA) were among them. Amir et al. used cuckoo search for solving structural optimization tasks. It has been used in non-linear constrained optimization problems [44].

In the year 2010, Yang et al. [25] examined CA to resolve the issue of engineering design optimization. The results that were obtained proved to be better than particle swarm optimizer. In the year 2011, Rajabioun [26] analyzed CA to solve nonlinear optimization issues which are used to solve difficult problems. Chandrasekaran et al. in the year 2011 [27], substantiated a hybrid CA integrated with fuzzy logic for solving multi-objective unit commitment problems. Yildiz [28] used this algorithm select optimal machining parameters for mining operations in the year 2013. Valian et al. [29] verified CA to the forward network for two classification problems.

Sur et al. [33] compared CA to particle swarm optimization, differential evolution, and artificial bee colony algorithm and establish CA on optimizing vehicle route in the graph-based network. Gandomi et al. [31] proposed CA for the purpose of truss structure optimization. Bulatovic et al. [32] interpreted CA to solve the issues related to optimum synthesis of a six bar double dwell linkage. Swain et al. in the year 2014 [34] critiques bio-inspired CA for a neural network and applied it on noise cancellation. Prakash et al. [35] examined CA for finding out optimal scheduling in computational grid. Walton et al. [27] proved CA for gradient-free optimization. Bhandaria et al. [37] explores CA and wind driven optimization for satellite image segmentation for multilevel thresholding using Kapur's entropy. In the year 2015 Ahmed et al. [43] verified a combinatorial test suite technique using CA. This method is only applicable to the functional testing process. Wang et al. [36] investigates a robust algorithm used to enhance the searching process of CA. Elazim et al. [45] maintained CA in the multi-machine power system. Table 1 summarizes the literature work.

The above studies show that CA has been applied for finding optimal solutions to various problems. It has been used in various applications from mining to finding routes. However, it has not been used for generating test data which can be used in testing the software products thereby reducing the efforts of the testers and developers. We propose an approach to generate test data that is optimal for performing testing in software development life cycle by using the CA.

3 Key research concepts

The concepts involved in developing the proposed algorithm along have been described in this section. This section discusses the concepts of nature inspired CA, FA, HC, and the objective function.

3.1 Cuckoo biological algorithm

It is an optimization algorithm developed by Yang et al. in the year 2009. Cuckoos are by and large known for their sweet voices, yet they have a forceful proliferation method. They lay their eggs in the nest of other host birds. If the host bird discovers that the eggs are alien then the host bird shall either discard them or abandon the nest. CA has been applied to various optimization problems. It is a nature inspired meta-heuristic algorithm which supports the theory of 'survival of the fittest'[38][41]. A number of algorithms work by beginning with an essential result and progressively adding more and more data that prompts creation of the best result from the pool of results. On the other hand, a few algorithms begin with a pool of results and come down to the best result, via disposing of the most exceedingly bad results from the pool, by thinking about among the results. CA falls into the second category of algorithms. The biological algorithms have been discussed in [39].

Each egg in a nest represents a solution and cuckoo egg represents a new solution. The algorithm aims at using the new and better solutions to replace the less good solutions. It is based on three idealized rules which are given as:

1. Cuckoo chooses one egg at a time which has to be dumped into a randomly chosen nest.
2. The nests containing high quality of eggs has to be carried forward to the next generation.
3. Available host nests is of fixed quota and laid eggs are discovered with the probability of (0, 1).

CA consists of two search capabilities: local search and global search which is controlled by discovery probability. In relation to testing, using CA, we can represent the eggs present in nest, as test data giving a solution. CA will replace high quality with the low-quality eggs already present in the nest. It increases the potential of getting good quality test data for the SUT.

The advantage of using CA is that it uses only a single parameter for optimization, unlike other algorithms which makes it easier to implement and since it consists of local search and a global search, it tends to give global optimal solution to the problem under test or SUT.

3.2 Firefly algorithm

This algorithm is inspired by the nature of fireflies. It is based on the collective behavior of fireflies. Fireflies are known for their flashing behavior. The variation of the light intensity and the formulation of attractiveness are the two major factors affecting the behavior of the fireflies. They use their flashlights to attract other fireflies. The algorithm was developed by Yang in the year 2008.The main purpose of flashing lights is to attract mating partners and warn off potential predators [26][27].

The FA algorithm is based on the idealized rules. First, all the fireflies are unisexual and the attraction between is irrespective of gender. Attraction is based on the brightness of fireflies such that low brightness firefly will move towards high brightness firefly. Since light intensity tends to decrease with the increase in distance between fireflies; hence, the attractiveness is inversely proportional to the distance between the fireflies. Brightness and intensity of the fireflies are determined using the objective function.

3.3 Hill climbing algorithm

This algorithm aims at finding superior results in an incremental way. It transforms its state by software under test and if the change delivers a superior result then the addition is carried out for performing the further evaluation. It aims at finding local optimal solutions, thereby achieving a result that is globally optimal. It is an iterative algorithm that starts with a random solution to the problem with the aim of finding out a better solution. The solution is changed if an improvement is found. This process continues until no further improvement can be made in the solution. It is used widely where you want to reach goal state from starting node. It also aims at maximizing or minimizing the target function [40]. Many variants of HC are available like steepest ascent HC, stochastic HC, random restart HC [20].

HC starts with the process of assigning the random coordinates to each test data. Then, first test data is taken as input and its neighbors are discovered. The objective function values are compared to the neighbors and the selected test data and best one is chosen as optimal test data. This process is iterated for every original test data. Finally, the optimized test data is produced by HC for the given SUT.

3.4 Objective function

This function is used to maximize / minimize some numerical values and is often used for finding the optimal solution for a given problem. In the provided domain the objective function's best value is selected by evaluating its objective

Year	Author	Key points
1975	Ramamoorthy et al. [4]	• The paper explores the main features of automated software tools and various software evaluation system, which were available. • Automated software tools were chosen because it has been found to be valuable to improve the reliability of a software.
1976	Miller et al. [5]	• Two examples i.e. a matrix factorization subroutine and a sorting method are used to describe the types of data generation problems. They are used instead of symbolic execution to generate test data. • The programs with floating-point data are used for large savings of time and storage are made possible.
1976	Clarke [6]	• The system proposed to generate test data for programs written in ANSI Fortran. • System symbolically executes the path and creates a set of constraints on the program's input variables. • It uses linear programming when the set of constraints are linear.
1987	Prather et al. [7]	• The novel technique i.e. "adaptive" is analysed for selection of subsequent paths and offers considerable advantages over existing strategies in its computational requirements. • Method ensures branch coverage and offers a considerable advantage in its computational requirements.
1990	Korel [8]	• The approach for generating test data is extended to programs with dynamic data structures and a search based method on dynamic data-flow analysis, along with backtracking is presented. • In the approach, values of array indexes and pointers are used.
1996	Ferguson et al. [9]	• The chaining approach for automated software test data generation, which is based on the theory of execution-oriented test data generation and also used for the search process. • The approach used significantly improves the test data generation compared to the existing methods.
1996	Korel et al. [10]	• The assertions are used to generate test data and is considered a tool for automatic runtime detection of software errors. • The assertion is violated reducing to the problem of finding program input on which a selected statement is executed. It is done with the help of white box testing.
2000	Frohlich et al. [11]	• Authors experiments, how test suites with a given coverage level can be automatically generated from state chart diagrams. • It is done by mapping the state chart elements to the STRIPS planning language.
2000	Kim et al. [12]	• The authors examine the Class Mutation technique that assesses the quality of test data distinguish between mutated programs from the original program. • It is complimented with the help of the results of the case study, which were tested to investigate the applicability of the technique.
2001	Ernst et al. [15]	Authors explore three results • Describes techniques for dynamically discovering invariants. • Reports on the Daikin's application to two sets of the program. • Analyzes scalability issues.
2004	McMinn [20]	• The authors reviewed Meta-heuristic search techniques are high-level frameworks. • It uses heuristics to seek solutions for combinatorial problems at a reasonable computational cost.
2005	Tonella [23]	• This proceeding describes some of the most advanced techniques that can be employed to reverse engineer several design views from the source code.
2006	Wappler et al. [21]	• The authors investigate a tree-based representation of method call sequences that search for numeric test data. • It automatically generates test programs that represent object-oriented unit test data.
2007	Xiao et al. [13]	• The paper experiments various automated test generation techniques but chooses goal oriented approach. • The goal-oriented approach as a promising approach to devising automated test-data generators using optimization techniques.
2007	Harman [14]	• Authors examine optimization techniques on seven application of software engineering. • Optimization techniques evolved from the operational research and metaheuristic research.
2008	Sofokleous et al. [18]	• Authors prove dynamic test data generation framework based on genetic algorithms. • They are the Batch-Optimistic and the Close-Up that provide an optimum set of test data with respect to the condition coverage criterion.
2010	Papadakis et al. [16]	• Authors compares an approach conjoins program transformation and dynamic symbolic execution techniques in order to automate successfully the test generation process.
2010	Yang et al. [25]	• Authors provide extensive comparison study using some standard test functions and newly designed stochastic test functions. • Examines CA to solve engineering design optimization problems.
2011	Fraser et al. [17]	• EVOSUITE is critiqued, a search-based approach that optimizes test suites for satisfying distinct coverage goals. • It achieves up to 18 times the coverage of a traditional approach.
2011	Rajabioun [26]	• Authors observe CA which is suitable for continuous nonlinear optimization problems.
2011	Valian [29]	• Authors interpret an algorithm which is employed for training feedforward neural networks for two benchmark classification problems.
2012	Walton et al. [27]	• New modified CA robust algorithm has been analysed, modification of CA involves the addition of information exchange between the best solutions.
2012	Prakash et al. [35]	• CA examines an Optimal Job Scheduling in Grid computational resource allocation and Resource Discovery.
2013	Gandomi et al. [44]	• The CA proves for solving structural optimization tasks with Leivy flights is first verified using a nonlinear constrained optimization problem and also validation against structural engineering optimization problem.
2013	Civicioglu et al. [30]	• The numerical optimization problem-solving successes of the proposed CA has been compared statistically over 50 different benchmark functions.
2013	Yildiz et al. [28]	• Authors examine a hybrid optimization approach based on differential evolution algorithm and also having receptor editing property of the immune system.
2013	Gandomi et al. [31]	• Authors examine CA to solve truss optimization problems. • It also adds unique search features used in the proposed CA.
2013	Bulatovic et al. [32]	• CA scrutinizes the procedure of optimum synthesis of mechanism parameters. • The paper also highlights the dimensional synthesis of a six-bar linkage with turning kinematic pairs.
2014	Sur et al. [33]	• Modified CA analyzes for discrete problem domain like that of the graph based problem and other combinatorial optimization problems.
2014	Swain et al. [34]	• The authors verify CA for noise removal from a signal. • It uses trained network to remove noise from sine signal, which was contaminated, with white Gaussian noise.
2014	Bhandaria et al. [37]	• Authors examine on image segmentation is to extract meaningful objects by CA and wind driven optimization using entropy.
2015	Ahmed et al. [43]	• CA maintains to construct optimized combinatorial sets. • The strategy consists of different algorithms for construction.
2016	Wang et al. [36]	• In robust CS, the pitch adjustment operation in harmony search (HS) that can be considered as a mutation operator is added to the process of the cuckoo updating to speed up convergence. • It is used for enhancing the capability of CS.
2016	Elazim et al. [45]	• CS verifies for optimal Power System Stabilizers (PSSs) design in a multi-machine power system. • The design problem of PSS is formulated as an optimization problem.

Table 1: List of research paper included for literature work.

function [42].The objective function is used to describe the closeness of a given design solution to achieving its aims in terms of numerical value. It works as a single figure of merit for determining the quality of the test data which can be compared to the objective function value of other test data. By this, we can compare the results and choose the better quality test data, hence optimal test data.

In this paper, we have considered the Sphere objective function [43]. The two major types of objective functions are:

- For Multi-Objective Optimization Problems: Some problems have multiple criteria to fulfill a particular objective[42]. They are: "Chakong and Haimes Function, Binh and Korn Function, Fonseca and Fleming Function, Test Function, Kursave Function, Schaffer Function N.1, Poloni's Two Objective Function".
- For Single Objective Optimization Problems: Some problems have single criteria to fulfill a particular objective hence it is not guaranteed that there may exist a single solution that will satisfy a particular objective [42]. They are: "Levi Function, Rosenbrock Function, Booth Function, Beale Function, Sphere Function, Three Hump Camel Function, Goldstein Price Function, and Cross in Tray Function".

4 Proposed algorithm

This section explains the proposed methodology that has been applied for optimizing test data. The technique is based upon the natural behavior of cuckoo for producing optimized test data that can be used for carrying out software testing efficiently thereby, reducing the cost of the product. The proposed technique uses the concept of exploration and exploitation for carrying out test data analysis and finding out the result which is optimal in nature.

The approach devised is mainly applicable to path testing. It ensures that proper path and code coverage is achieved with the help of optimized test data. Initially, ample amount of test data for the program or software under test is generated. These initial test data are analogous to the eggs which are laid by cuckoo. The paths on which testing needs to be performed are equivalent to the nests where the fitness of the eggs needs to be judged. Just as a cuckoo searches for a random nest for disposing of its eggs, in the same way, a test data is picked up randomly which is disposed of in a random path. The host bird can discard the egg or the path itself depending upon the conditions. Similarly, the chosen test data is validated with the chosen path. If they are found to be compatible then the test data is kept in that path else it is kept in another path named buffer.

In simple words, a test data and a path are chosen at random. If the chosen test data belongs to that path then the test data is kept there itself, else it is kept in another path called buffer. Initially, each test data undergoes this process unless the initial list of test data becomes empty. Once this initial list of test data becomes empty, we consider another list called buffer. This is an extra path or nest which keeps those test data for which a compatible path was not found in first go. Before further picking of nests and eggs, the test data with a minimum value of objective function is discarded from each path. For further evaluations, random picking of test data is done from the buffer list and not from the initial list. A test data is picked up randomly along with a path. If the test data belongs to that path then that test data is kept in that path only, else it remains in the buffer. This process continues until the buffer list becomes empty. Every time when a match is found, the objective function is applied to the available test data. The test data with a minimum value of the objective function is discarded every time. At the end, when the list empties, the optimized test data belonging to a particular path are obtained. There might be a case that a path might be discarded which means that no test data might be available for evaluation purpose. In such data, the path becomes obsolete and it becomes ineligible for performing testing.

Depending upon the type of test data, different values are fed into the objective function for reaching optimized results for a particular path. If the test data is of a numerical type, then the values are directly fed into the objective function for carrying out analysis and finding the value. For string type of test data, the value that is fed into the objective function is the average of the ASCII values of individual characters. Characters can be alphabetical or special characters. For example, let the test data which has been given for evaluation be 'mam' then the value that we will be opting for putting into objective function can evaluated as mentioned in equation 1

m=109 (ASCII value of m), a=97 (ASCII value of a)

$$mam = (109 + 97 + 109)/3 = 105. \tag{1}$$

105 is given as input to the objective function which helps in determining the optimality of the test data which are of type string. The pseudo code for the CA is depicted below:

Pseudo code of CA

```
Declaration
initlist: initial list.
buff: buffer or temporary list.
Input:Program under test.
Output:Optimize test data.

1. Initialize test data for the whole program in a path or nest in a list
   called initlist.
2. Separate paths or nests for the program.
3. Initialize objective function.
4. While (initlist!=empty )
   (a) Pick a random test data or egg
   (b) Pick a random path or nest
       i. If ( the chosen test data belong to the chosen path )
          A. Keep it in the path or nest
          B. Remove it from the initlist or path
       ii. Else
          A. Keep it in a list or nest call buff
          B. Remove it from the initlist
       iii. End if
5. End while
6. For each path or nest
   (a) Find objective function for each test data using formula in
       equation 1
   (b) Find out test data or egg with minimum value of objective
       function and remove it
7. End for
8. While (buff!=empty )
   (a) Select a test data or egg randomly
   (b) Select a path or nest randomly
       i. If ( test data or egg belong to that path or nest )
          A. Keep it in that path or nest
          B. Remove it from buff list or nest
       ii. Else
          A. Put it back in the buff list or nest
       iii. End if
   (c) Calculate value of the objective function for each test or egg
       data for the path or nest opted.
   (d) Remove the test data or egg with a minimum value of the
       objective function from the path or nest.
9. End while
10. Optimized test data or eggs for each path or nest.
```

5 Data collection

Evaluating the performance of any technique requires selecting certain subject programs which form the basis for evaluation. To evaluate the performance of our proposed algorithm and to compare it with other algorithms, we have selected fifty real time programs written in Java language. The subject programs were chosen for test data generation and optimization activity has been discussed in the following subsection. The size of programs ranges from 30 to 250 lines of source code. A diversified range of programs were chosen including mathematical problems such as finding roots of quadratic equation, triangle classification problem, computing the median of the triangle; general logical problems such as checking for the Armstrong number, magic number, palindrome number; Pythagorean triplet, convert a number to hexadecimal, octal or binary. All these programs are written in standard Java language that makes it easier to work with.

To discuss the advantage of our algorithm over other techniques, we have developed an analytical framework. This framework evaluates our algorithm and compares it with other algorithms on the basis of four parameters: optimized test data, iteration, duration, and complexity.

5.1 List of programs

Table 2 lists the JAVA programs used for the analysis of the algorithms.

Program	Description
P1.	Find greatest of three given numbers.
P2.	Find if a number is even or odd.
P3.	Find if a number is positive or negative.
P4.	Find if a year is a leap or not.
P5.	Find if a number is multiple of 2,3,5,7 or not.
P6.	Find smallest of three numbers.
P7.	Find the area of figures depending upon choice entered by the user.
P8.	Find if a number id perfect square or not.
P9.	Find if a number is in powers of two or not.
P10.	Check if the number entered is palindrome or not.
P11.	Find if a number is Armstrong number or not.
P12.	Find if a number is perfect or not.
P13.	Find if a number is a magic number or not.
P14.	Find if a number is prime or not.
P15.	Find the type of character entered by the user (character, special character, number).
P16.	Find if a string entered is palindrome or not.
P17.	Perform a calculation on the entered numbers depending upon the choice.
P18.	Find if a triangle is a scalene, equilateral or isosceles.
P19.	Find if the given three numbers entered from a Pythagorean triplet or not.
P20.	Find if the number entered is a single digit number or not.
P21.	Calculate bonus depending upon the number of extra days entered by the user.
P22.	Find the quadrant of the points entered by the user.
P23.	Find types of roots entered by the user.
P24.	Implement linear search.
P25.	Implement a binary search
P26.	Convert a number to binary, hexadecimal or octal entered by the user.
P27.	Find greatest common divisor of the entered two numbers.
P28.	Find least common multiple of entered two numbers.
P29.	Find if the number entered is strong or not.
P30.	Calculate bill depending upon the units entered by the user.
P31.	Find if a triangle is right-angled or not.
P32.	Find the volume of the given figure.
P33.	Find the total surface area of a given figure.
P34.	Find the total surface area of a given figure.
P35.	Convert the temperature from Celsius to Fahrenheit, Kelvin to Celsius and vice-versa.
P36.	Find S.T.D codes or states depending upon the value entered by the user.
P37.	To allot section to a student depending upon the marks obtained.
P38.	Calculate profit or loss.
P39.	Find details of any element depending upon the element entered by the user.
P40.	Find if the string entered by the user starts with a vowel or not.
P41.	Find the strength of a password.
P42.	Find the sum of three numbers entered by the user.

Continued on next page

Continued from previous page

Program	Description
P43.	Interconvert currencies.
P44.	Calculate B.M.I after entering height and weight.
P45.	Convert interconvert seconds, minutes and hours.
P46.	Calculate simple interest.
P47.	Calculate compound interest.
P48.	Calculate factors of given number.
P49.	Determine if the given series are in A.P or G.P.
P50.	Find the capital of a given state or vice versa.

Table 2: List of JAVA program used as a program under test

5.2 Metrics for evaluation

The following metrics were considered for the evaluation and comparison of the proposed work with the existing FA and HC algorithms:

1. *Number of Optimized Test data:* This is the main focus of this research paper as the proposed algorithm aims at reducing the number of test data for any given path of any particular program. FA, HC and proposed algorithm using CA were implemented on the programs and it was observed that good results were given by proposed algorithm. The number of optimized test data was counted by embedding a counter in the source code of each program. For each optimized test data *optCount (O_C)* can be defined as in equation 2:
$$O_C = O_C + 1 \qquad (2)$$

2. *Iterations:* Number of iterations while applying the algorithm and finding out test data was calculated for FA, HC and proposed CA. A counter for counting a number of lines of execution was embedded in each program for all three algorithms. After the execution of each program under test, the counter was incremented for finding out a number of lines of execution. For each optimized test data *iterations Count* ($\mathrm{I_C}$) can be defined as in equation 3:
$$I_C = I_C + 1 \qquad (3)$$

3. *Duration:*This metric refers to the execution time of the program. The code for calculating the time of execution was embedded in every program. *Initial time* ($\mathrm{I_T}$) and *final time* ($\mathrm{F_T}$) after completion were noted and the two were subtracted for finding out the time period of execution. Therefore, *Time of execution/ Duration* ($\mathrm{E_T}$) can be represented in equation 4:
$$E_T = F_T - I_T \qquad (4)$$

4. *Cyclomatic Complexity:* The complexity of the codes were calculated using a plugin of Eclipse IDE tool, Metrics. It takes the programs for which the complexity has to be generated as the input. It was found that CA gave better results as compared to FA and HC. McCabe Cyclomatic Complexity ($\mathrm{C_{MC}}$) where, number of edges (E) in control flow graph and number of nodes (N) as in equation 5:
$$C_{MC} = E - N + 2 \qquad (5)$$

6 Analytical evaluation and comparison

6.1 Case study

In this section, we consider a case study of calculating bonus of an employee depending upon the number of days he has worked extra. The detailed results are generated after evaluating a particular case study with the help of CA. The example shows how CA can be employed to find out optimized test data. The case study has three paths; one in which the number of days is in between 1-5, second in which the number of days is in between 6-10, and the third path in which the number of days is in between 11-15. The following sections show the detailed view of each and every iteration involved while evaluating the case study. The initial list (*initlist*) of test data is given as 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16. There are three paths in the program that are named as *Path 1, Path 2, and Path 3.*The test data in terms of path are given as: *Path 1 = 1,2,3,4,5,6 ; Path 2 = 7,8,9,10 ; Path 3 = 11,12,13,14,15,16.*

Now in the first iteration, the cuckoo bird picks up a test data and a particular nest which is termed as a path. The lists of the three paths are given as one, two and three. If both of them are compatible then the chosen test data is put into that path else it is kept in a nest called buff (buffer/ temporary list) which does not discard them completely. The first iteration result is shown in the following Table 3, which gives the chosen test data, chosen path and the path to which the test data has been added.

Table 4 shows the lists after the first iteration along with the objective function value.

After the first iteration, the test data / eggs with a minimum value of objective functions are removed from the lists. The modified lists are mentioned in table 5.

Now the test data are again picked randomly from the buff list and again a random path is chosen. If both are compatible then the test data is added to the list of the corresponding path else it is kept in the buff list. This process continues until the buff list becomes empty. Every time the nest and the egg are compatible then the egg with the least value of the objective function will be removed from the nest or path. The egg was chosen, the nest was chosen, status if it has been added or not in the list, and value removed from the list has been shown in the following table 6.

Once the buff list is empty then the optimized test data are obtained for each path. The number of optimized test data for each and every path is 1. Table 7 shows the path-wise total number of test data along with optimized test

Chosen egg/test data	Chosen path/nest	Nest to which the egg/test data has been added
8	Two	Two
11	One	Buff
3	Three	Buff
2	One	One
4	Two	Buff
13	Three	Three
15	One	Buff
5	One	One
6	Two	Buff
16	Three	Three
12	One	Buff
14	One	Buff
9	One	Buff
1	Two	Buff
7	Two	Two
10	Two	Two

Table 3: Results after Iteration 1 of CA

data, Iteration and Complexity results obtained after applying FA, HC, and CA on the case study mentioned in section 6.

6.2 Results for 50 programs

Table 7 shows comparative results for JAVA programs after applying FA, HC, and CA on them. The results for various parameters have been depicted in Table 8. The obtained results show that CA achieves best results in terms of optimized test data, duration, iteration, and complexity. Fig1 - Fig4 provides graphical comparisons of the algorithms using the aforementioned metrics.

Figure 1 shows the number of optimized test data using FA, HC, and CA along with the total number of test data. In this figure line plots the number of optimized test data and a number of programs showing the maximum optimized result of CA. The square line represents test data using FA, triangle line shows the number of optimized test data using HC, cross line represent a number of test data using CA and the total number of test data are represented by diamond lines.

Through statically analysis, we can check whether the performance of all the algorithms is similar or not. As the theory of ANOVA states that for the larger value of computed value of F proves its better capability. To fulfill this objective ANOVA two-factor without replication for that we have to set null hypothesis, which states: H_0: There is not a significant difference between the various programs among different algorithms. To check the significance of the algorithms, authors are using ANOVA for optimization of test suite. In Table 9, 10, 11 and 12 Rows are Numbers of programs and Columns indicated all algorithms. The result is shown in Table 9.

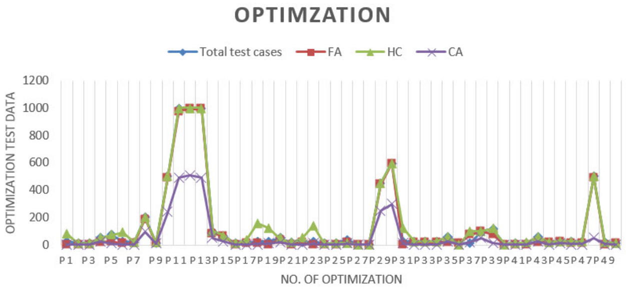


Figure 1: The comparison of FA, HC, and CA on number of optimized test data

Since the computed values of F test statistic (21) is greater than that of critical value (1.48) in terms of number of programs and in terms of algorithms computed values of F test statistic (15.7) is greater than that of critical value (3) at 5 % level of significance, H_0 is not accepted and hence the result of optimized test suite differs significantly.

Similarly, other performance parameters like execution time and H_0 rejected their respective null hypothesis at 5% level of significance. For execution time the computed values of F test statistics (3.14) is greater than that of critical value (1.48) for number of programs and for different algorithms computed values of F test statistics (18.5) is greater than that of critical value (3) at 5% level of significance, so H_0 cannot accept and hence the duration of execution time of the test suite differs significantly. Table 10 shows the result.

Figure 2 line plots show the comparison between FA, HC, and CA based on the time taken by the respective algorithms to optimize the test suite of the JAVA programs. The diamond line represents time taken by FA and the square line shows time taken by HC and triangle line plots in their presentation of CA.

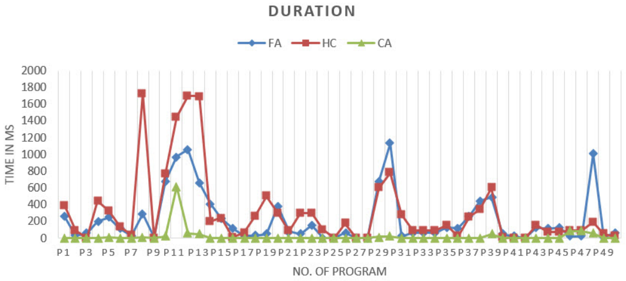


Figure 2: The comparison of FA, HC, and CA on duration of test data

Figure 3 gives the graphical representation of compared result in terms of iteration according to the result is shown in table 8. We analyze that the CA performs the best due to its cumulative response. FA always calculate iteration depending on each path separately. The diamond line represents FA, the square line represents HC, and the triangle line plotting represents CA.

List one	Objective Function value	List two	Objective function value	List three	Objective function value	Buff
2	4	8	64	13	169	11
5	25	7	49	16	256	3
-	-	10	100	-	-	4
-	-	-	-	-	-	15
-	-	-	-	-	-	6
-	-	-	-	-	-	12
-	-	-	-	-	-	14
-	-	-	-	-	-	9
-	-	-	-	-	-	1

Table 4: lists which include objective function

List one	List two	List three
5	8	16
-	10	-

Table 5: Updated lists after removing minimum value objective function test data.

Test Data	Path	Status	Value removed
9	One	Not compatible	-
4	Three	Not compatible	-
12	Three	Compatible	12
9	Two	Compatible	8
6	Two	Not compatible	-
3	Three	Not compatible	-
15	One	Not compatible	-
3	Two	Not compatible	-
6	Three	Not compatible	-
4	Two	Not compatible	-
14	Two	Not compatible	-
15	Two	Not compatible	-
15	Three	Compatible	15
6	Three	Not compatible	-
11	Three	Compatible	11
6	One	Compatible	6
1	One	Compatible	1
14	One	Not compatible	-
3	One	Compatible	2
14	Three	Compatible	14

Table 6: Status of egg and nest chosen

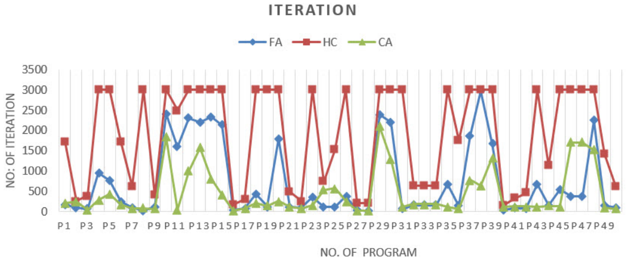


Figure 3: The comparison of FA, HC, and CA on Iteration of test data

For the Iteration, statically the computed value of F test statistics (5.5) is greater than that of critical value (1.48) for Rows and in Columns the computed value of F test statistics (71.7) is greater than that of critical value (3) at 5% level of significance, H_0 is not accepted and hence, the iteration differs significantly. Table 11 shows the result.

Figure 4 gives the graphical representation of running time complexity of JAVA programs. CA run time complexity is based on Big O notation. CA outperforms as compared to rest of the algorithms. The diamond line represents time taken by FA, the square line shows time taken by HC and triangle line plotting represents CA.

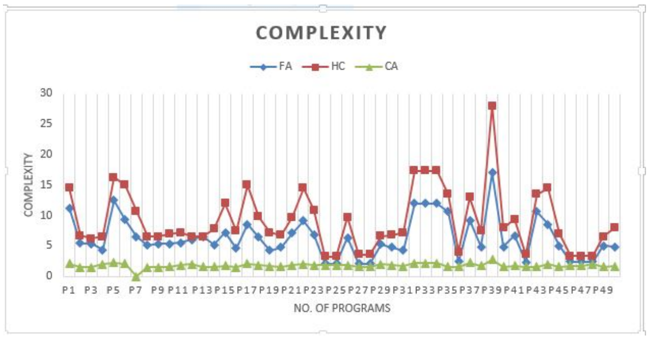


Figure 4: The comparison of FA, HC, and CA on complexity of test data

Statically analyzing the cyclomatic Complexity, we find

Main Iteration		Optimized test data			Iterations			Complexity		
Path number	Test data	HC	CA	FA	HC	CA	FA	HC	CA	FA
1	6	6	1	2	204	114	47	3.25	1.88	2.38
2	4	4	1	1	76		27	3.25		2.38
3	6	6	1	2	211		47	3.25		2.38

Table 7: Final compared result using the three algorithms on a case study

that, the computed value of F test statistics (3.8) is greater than that of critical value (1.48) for Rows and in Columns the computed value of F test statistics (113) is greater than that of critical value (3) at 5% level of significance, H_0 is not accepted and hence, the iteration differs significantly. Table 12 shows the result.

In Table 13, descriptive statistics for the result of test data of FA, HC, and CA algorithm are been mentioned. The given descriptive statistics include: mean, standard error, median, mode, and standard deviation. At 95 % confidence, authors can say that the result of optimized test data of CA which shows much-improved performance as compared to FA and HC.

The descriptive statistics for the execution time of test data using FA, HA, and CA algorithm is mentioned in Table 14. At 95% confidence, authors can say that the duration time of CA is shown much-improved performance as compared to FA and HA.

The descriptive statistics for the Iteration of test data using FA, HC, and CA algorithm is mentioned in Table 15. At 95% confidence, authors can say that the execution time of CA is testifying much-improved performance as compared to FA and HC.

The descriptive statistics for the Cyclomatic complexity of test data using FA, HC, and CA algorithm is mentioned in Table 16. At 95% confidence, authors can say that the execution time of CA is testifying much-improved performance as compared to FA and HC.

The performance of the algorithms has been analyzed statistically. ANOVA two-factor without replication has been implemented to check the significant difference among the various algorithms. Descriptive statistics also be performed and the confidence interval is built on the basis of average and standard error. Statistically, it was found that the performance of CA is better than that of FA and HA algorithms.

7 Threats

7.1 Internal threats

- The initial test data were generated by making use of the worst case analysis.
- The optimized test data were produced although the approach that we have adopted is purely randomized.
- This algorithm might take some amount of time to make the buff list empty and hence, the process needs to be continued till it becomes empty.
- It may happen that a test data which does not belong to any path will be there in the buff list which would mean that the desired results might not be obtained.

External threats

- This algorithm is only applied to Java based programs and has also not been applied to any other application.

Construction threats

- The above program needs Java development kit and Java runtime environment.
- The calculated computation time might change every time as it could depend on the operating system and other system features.

8 Conclusion and future scope

CA has been proved to be effective in producing optimized test data. A large number of test data for all possible paths in a program under test were reduced without any redundancy. The reduced number of test data were effective in minimizing the overall cost, effort, and time of the testing phase in software development life cycle. In future, attempts can be made to extend the work by applying the algorithm to large datasets and for real-time applications as well. This potentially powerful optimization technique can be extended to study multi-objective optimization applications with various constraints, even to NP-hard problems. The studies can also focus on hybridization of CA with other metaheuristic algorithms.

References

[1] Sommerville, I. (2004) *Software engineering*, International computer science series. ed: Addison Wesley.

[2] Mathur, A. P. (2008) *Foundations of Software Testing*, Pearson Education India.

[3] Mall, R. (2014) *Fundamentals of software engineering*, PHI Learning Pvt. Ltd.

No.	Total Data	Optimization			Duration(in ms)			Iterations			Complexity		
-	-	FA	HC	CA	FA	HC	CA	FA	HC	CA	FA	HC	CA
P1	27	6	83	4	266	395	3	183	1715	214	11.2	14.48	2.11
P2	12	4	11	5	33	93	1	106	264	249	5.42	6.75	1.5
P3	12	2	12	4	58	4	2	68	396	48	5.34	6.24	1.5
P4	50	18	50	24	197	443	3	962	3000	280	4.4	6.5	2
P5	75	29	75	8	247	324	10	764	3000	436	12.5	16.25	2.25
P6	27	9	89	5	109	137	6	246	1715	179	9.32	15	2.11
P7	18	10	18	4	29	43	4	109	618	85	6.57	10.74	1.89
P8	200	190	200	98	290	1729	8	19	3000	91	5.14	6.54	1.5
P9	15	10	16	8	10	7	1	119	400	70	5.34	6.5	1.5
P10	500	492	500	241	677	772	20	2420	3000	1840	5.37	7	1.62
P11	1000	977	1000	491	975	1446	620	1600	2500	53	5.5	7.22	1.86
P12	1000	994	1000	509	1056	1695	61	2300	3000	1001	6	6.54	2
P13	1000	994	1000	489	659	1694	47	2200	3000	1577	6.5	6.58	1.62
P14	100	86	100	49	406	202	4	2341	3000	812	5.2	7.82	1.62
P15	70	69	70	25	234	246	7	2141	3000	419	7.25	12	1.75
P16	8	2	8	2	117	8	1	38	172	24	4.58	7.5	1.5
P17	16	8	38	4	28	69	2	80	305	78	8.56	15	2.11
P18	27	11	157	7	34	267	3	434	3000	205	6.42	9.82	1.89
P19	27	6	126	11	48	503	2	137	3000	142	4.28	7.25	1.67
P20	50	35	50	22	384	304	4	1800	3000	267	4.76	6.76	1.62
P21	16	5	16	3	82	87	6	121	491	114	7.14	9.75	1.88
P22	16	16	52	4	51	299	2	104	264	83	9.16	14.48	2
P23	27	1	143	9	153	298	3	361	3000	148	6.87	10.86	1.89
P24	10	3	10	1	2	100	5	123	738	538	2.12	3.25	1.88
P25	13	7	13	3	3	5	5	124	1527	578	2.12	3.25	1.88
P26	39	16	13	11	62	173	5	377	3000	260	6.36	9.75	1.88
P27	6	6	6	2	2	3	1	39	210	29	2.14	3.62	1.56
P28	6	6	6	2	2	2	1	12	210	31	2.14	3.62	1.56
P29	450	442	450	253	681	599	17	2400	3000	2300	5.29	6.63	2
P30	600	593	600	297	1130	776	23	2200	3000	1300	4.76	6.76	1.88
P31	27	6	126	10	22	279	2	60	3000	125	4.28	7.24	1.67
P32	29	18	29	3	70	93	4	164	650	187	12.01	17.36	2.22
P33	29	18	29	3	70	93	4	164	650	187	12.01	17.38	2.22
P34	29	18	29	3	70	93	4	164	650	187	12.01	17.37	2.22
P35	60	22	60	24	128	158	7	682	3000	124	10.68	13.52	1.62
P36	14	12	6	6	117	24	2	145	1762	66	2.43	4	1.62
P37	10	74	100	18	264	255	6	1874	3000	786	9.26	13	2.25
P38	100	100	100	50	442	344	4	2975	3000	645	4.86	7.5	1.75
P39	118	77	120	13	489	602	49	1684	3000	1325	17.01	28	2.78
P40	7	4	7	2	49	11	7	36	162	125	4.86	8	1.62
P41	15	5	14	6	22	7	2	88	330	137	6.75	9.36	1.75
P42	8	3	16	4	3	3	2	82	464	137	2.33	3.62	1.56
P43	60	22	60	24	128	158	7	682	3000	124	10.68	13.52	1.62
P44	23	23	23	5	113	73	3	161	1126	141	8.56	14.48	2
P45	30	30	30	14	131	77	7	540	3000	130	5	7.04	1.62
P46	24	13	24	7	20	89	87	385	3000	1712	2.43	3.38	1.75
P47	24	13	24	7	20	89	87	385	3000	1712	2.43	3.38	1.75
P48	500	496	500	51	1007	193	60	2246	3000	1518	2.43	3.38	2
P49	20	4	20	7	8	51	7	154	1419	92	5	6.5	1.56
P50	14	10	14	4	65	24	6	94	624	61	4.86	8	1.67

Table 8: Comparison results for JAVA programs

Source of Variation	SS	df	MS	F	P-value	F crit
Rows	6715135	49	137043.6	21	7.31	1.48
Columns	204886.9	2	102443.4	15.7	1.15	3.08
Error	636560.4	98	6495.515	-	-	-
Total	7556583	149	-	-	-	-

Table 9: ANOVA results based on Optimization

Source of Variation	SS	df	MS	F	P-value	F crit
Rows	8844974	49	180509.7	3.14	7.18	1.48
Columns	2132012	2	1066006	18.56	1.46	3.08
Error	5627039	98	57418.77	-	-	-
Total	16604025	149	-	-	-	-

Table 10: ANOVA Result Of Execution Time/ Duration

Source of Variation	SS	df	MS	F	P-value	F crit
Rows	88715082	49	1810512	4.52	1.08	1.48
Columns	57403410	2	28701705	71.74	6.43	3.08
Error	39204947	98	400050.5	-	-	-
Total	185000000	149	-	-	-	-

Table 11: ANOVA Result Of Iteration

Source of Variation	SS	df	MS	F	P-value	F crit
Rows	1160.181	49	23.67	3.88	5.7	1.48
Columns	1386.396	2	693.19	113.63	2.96	3.08
Error	597.8448	98	6.1	-	-	-
Total	3144.421	149	-	-	-	-

Table 12: ANOVA Result Of Cyclomatic Complexity

-	FA	HC	CA
Mean	120.3	144.86	57.12
Standard Error	36.80852	36.27629	18.16194
Median	14.5	44	7
Mode	6	16	4
Standard Deviation	260.2755	256.5121	128.4243
Minimum	1	6	1
Maximum	994	1000	509
Sum	6015	7243	2856
Count	50	50	50

Table 13: Descriptive Statistics For The Result Of Optimized Test data

-	FA	HC	CA
Mean	225.26	308.78	24.68
Standard Error	42.72953	62.64793	12.50474
Median	95.5	147.5	4.5
Mode	2	93	2
Standard Deviation	302.1434	442.9877	88.42186
Minimum	2	2	1
Maximum	1130	1729	620
Sum	11263	15439	1234
Count	50	50	50

Table 14: Descriptive Statistics For The Duration Of Test data

-	FA	HC	CA
Mean	733.86	1887.24	459.44
Standard Error	127.2218	170.9619	82.45547
Median	173.5	2750	183
Mode	164	3000	187
Standard Deviation	899.5941	1208.883	583.0482
Minimum	12	162	24
Maximum	2975	3000	2300
Sum	36693	94362	22972
Count	50	50	50

Table 15: Descriptive Statistics For The Iteration

-	FA	HC	CA
Mean	6.3526	9.2098	1.8256
Standard Error	0.466785	0.705869	0.037717
Median	5.355	7.375	1.75
Mode	2.43	14.48	1.62
Standard Deviation	3.300668	4.991246	0.266697
Minimum	2.12	3.25	1.5
Maximum	17.01	28	2.78
Sum	317.63	460.49	91.28
Count	50	50	50

Table 16: Descriptive Statistics For Complexity

[4] Ramamoorthy, C. V., and Ho, S. B. F. (1975) Testing large software with automated software evaluation systems, *In ACM SIGPLAN Notices*, ACM, pp. 382-394.

[5] Miller, W., and Spooner, D. L. (1976) Automatic generation of floating-point test data, *IEEE Transactions on Software Engineering*, IEEE, pp. 223.

[6] Clarke, L. A. (1976) A system to generate test data and symbolically execute programs, *IEEE Transactions on Software Engineering*, IEEE, pp. 215-222.

[7] Prather, R. E., and Myers Jr, J. P. (1987) The path prefix software testing strategy, *IEEE Transactions on Software Engineering*, IEEE, pp. 761-766.

[8] Korel, B. (1990) The path prefix software testing strategy, *IEEE Transactions on Software Engineering*, IEEE, pp. 870-879.

[9] Ferguson, R., and Korel, B. (1996) The chaining approach for software test data generation, *IACM Transactions on Software Engineering and Methodology (TOSEM)*, ACM, pp. 63-86.

[10] Korel, B., and Al-Yami, A. M. (1996) Assertion-oriented automated test data generation, *18th international conference on Software engineering*, IEEE, pp. 71-80.

[11] Fröhlich, P., and Link, J. (2000) Automated test case generation from dynamic models, *In ECOOP 2000 Object-Oriented Programming*, Springer Berlin Heidelberg, pp. 472-491.

[12] Kim, S., Clark, J. A., and McDermid, J. A. (2000) Class mutation: Mutation testing for object-oriented programs, *Proc. Net. ObjectDays*, Citeseer, pp. 9-12.

[13] Xiao, M., El-Attar, M., Reformat, M., and Miller, J. (2007) Empirical evaluation of optimization algorithms when used in goal-oriented automated test data generation techniques, *Empirical Software Engineering*, Springer, pp. 183–239.

[14] Harman, M. (2007) The current state and future of search based software engineering, *Future of Software Engineering*, IEEE, pp. 342-357.

[15] Ernst, M. D., Cockrell, J., Griswold, W. G., and Notkin, D. (2001) The current state and future of search based software engineering, *IEEE Transactions on Software Engineering*, IEEE, pp. 99-123.

[16] Papadakis, M., and Malevris, N. (2010) Automatic mutation test case generation via dynamic symbolic execution, *IEEE 21st international symposium on Software reliability engineering (ISSRE)*, IEEE, pp. 121-130.

[17] Fraser, G., and Arcuri, A. (2011) Evolutionary generation of whole test suites, *11th International Conference on Quality Software (QSIC)*, IEEE, pp. 31-40.

[18] Sofokleous, A. A., and Andreou, A. S. (2008) Automatic, evolutionary test data generation for dynamic software testing, *Journal of Systems and Software*, IEEE, pp. 1883-1898.

[19] Fraser, G., and Arcuri, A. (2011) Evosuite: automatic test suite generation for object-oriented software, *In Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, ACM, pp. 416-419.

[20] McMinn, P. (2004) Search-based software test data generation: A survey, *Software Testing Verification and Reliability*, Wiley, pp. 105-156.

[21] Wappler, S., and Wegener, J. (2006) Evolutionary unit testing of object-oriented software using strongly-typed genetic programming, *In Proceedings of the 8th annual conference on Genetic and evolutionary computation*, ACM, pp. 1925-1932.

[22] Ribeiro, J. C. B. (2008) Search-based test case generation for object-oriented java software using strongly-typed genetic programming, *In Proceedings of the 10th annual conference companion on Genetic and evolutionary computation*, ACM, pp. 1819-1822.

[23] Tonella, P. (2005) Reverse engineering of object oriented code, *In Proceedings of the 27th international conference on Software engineering*, ACM, pp. 724-725.

[24] Lakhotia, K., Harman, M., and McMinn, P. (2007) A multi-objective approach to search-based test data generation, *In Proceedings of the 9th annual conference on Genetic and evolutionary computation*, ACM, pp. 1098-1105.

[25] Yang, X. S., and Deb, S. (2010) Engineering optimisation by cuckoo search, *International Journal of Mathematical Modelling and Numerical Optimisation*, Inderscience, pp. 330-343.

[26] Rajabioun, R. (2011) Cuckoo optimization algorithm, *Applied soft computing*, Elsevier, pp. 5508-5518.

[27] Walton, S., Hassan, O., Morgan, K., and Brown, M. R. (2011) Modified cuckoo search: a new gradient free optimisation algorithm, *Chaos, Solitons and Fractals*, Elsevier, pp. 710-718.

[28] Yildiz, A. R. (2013) Cuckoo search algorithm for the selection of optimal machining parameters in milling operations, *The International Journal of Advanced Manufacturing Technology*, Springer, pp. 55-61.

[29] Valian, E., Mohanna, S., and Tavakoli, S. (2011) Improved cuckoo search algorithm for feedforward neural network training, *International Journal of Artificial Intelligence and Applications*, Elsevier, pp. 36-43.

[30] Civicioglu, P., and Besdok, E. (2013) A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms, *Artificial Intelligence Review*, Springer, pp. 315-346.

[31] Gandomi, A. H., Talatahari, S., Yang, X. S., and Deb, S. (2013) Design optimization of truss structures using cuckoo search algorithm, *The Structural Design of Tall and Special Buildings*, Wiley, pp. 1330-1349.

[32] Bulatovic, R. R., Đordevic, S. R., and Đordevic, V. S. (2013) Cuckoo search algorithm: a metaheuristic approach to solving the problem of optimum synthesis of a six-bar double dwell linkagem, *Mechanism and Machine Theory*, Elsevier, pp. 1-13.

[33] Sur, C., and Shukla, A. (2014) Discrete Cuckoo Search Optimization Algorithm for Combinatorial Optimization of Vehicle Route in Graph Based Road Network, *In Proceedings of the Third International Conference on Soft Computing for Problem Solving*, Springer, pp. 307-320.

[34] Swain, K. B., Solanki, S. S., and Mahakula, A. K. (2014) Bio inspired cuckoo search algorithm based neural network and its application to noise cancellation, *International Conference on Signal Processing and Integrated Networks*, IEEE, pp. 632-635.

[35] Prakash, M., Saranya, R., Jothi, K. R., and Vigneshwaran, A. (2012) An optimal job scheduling in grid using cuckoo algorithm, *International Journal Computer Science Telecomm*, pp. 65-69.

[36] Wang, G. G., Gandomi, A. H., Zhao, X., and Chu, H. C. E. (2016) Hybridizing harmony search algorithm with cuckoo search for global numerical optimization, *Soft Computing*, Springer, pp. 273-285.

[37] Bhandari, A. K., Singh, V. K., Kumar, A., and Singh, G. K. (2014) Cuckoo search algorithm and wind

driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy, *Expert Systems with Applications*, Elsevier, pp. 3538-3560.

[38] Mala, D. J., and Mohan, V. (2010) Quality improvement and optimization of test cases: a hybrid genetic algorithm based approach, *ACM SIGSOFT Software Engineering Notes*, ACM, pp. 1-14.

[39] Lotem, A., Nakamura, H., and Zahavi, A. (1992) Rejection of cuckoo eggs in relation to host age: a possible evolutionary equilibrium, *Behavioral Ecology*, ISBE, pp. 128-132.

[40] Yildiz, A. R. (2009) An effective hybrid immune-hill climbing optimization approach for solving design and manufacturing optimization problems in industry, *Journal of Materials Processing Technology*, Elsevier, pp. 2773-2780.

[41] Devadas, S., Ma, H. K. T., Newton, A. R., and Sangiovanni-Vincentelli, A. (1988) Synthesis and optimization procedures for fully and easily testable sequential machines, *In International Conference on New Frontiers in Testing*, IEEE, pp. 621-630.

[42] Chun, J. S., Jung, H. K., and Hahn, S. Y. (1998) A study on comparison of optimization performances between immune algorithm and other heuristic algorithms, *IEEE Transactions on Magnetics*, IEEE, pp. 2972-2975.

[43] Ahmed, B. S., Abdulsamad, T. S., and Potrus, M. Y. (2015) Achievement of minimized combinatorial test suite for configuration-aware software functional testing using the Cuckoo search algorithm, *Information and Software Technology*, Elsevier, pp. 13-29.

[44] Gandomi, A. H., Yang, X. S., and Alavi, A. H. (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, *Engineering with computers*, Springer, pp. 17-35.

[45] Elazim, S. A., and Ali, E. S. (2016) Optimal Power System Stabilizers design via Cuckoo Search algorithm, *International Journal of Electrical Power and Energy Systems*, Elsevier, pp. 99-107.

[46] Papadakis, M., and Malevris, N. (2012) Mutation based test case generation via a path selection strategy, *Information and Software Technology*, 54(9), pp. 915-932.

[47] Papadakis, M., and Malevris, N. (2011) Automatically performing weak mutation with the aid of symbolic execution, concolic testing and search-based testing, *Software Quality Journal*, 19(4), pp. 691-723.

[48] Papadakis, M., and Malevris, N. (2013) Searching and generating test inputs for mutation testing, *SpringerPlus*, 2(1), pp. 1.

Classification of Vegetation in Aerial LiDAR Data

Denis Horvat
Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia
E-mail: denis.horvat@um.si, Web: https://gemma.feri.um.si/

Thesis summary

Keywords: LiDAR, classification, vegetation, mathematical morphology, surface fitting

Received: April 12, 2017

This contribution summarises a doctoral dissertation which proposes an algorithm for the classification of vegetation points in aerial LiDAR data. The algorithm characterizes vegetated areas based on statistically large dispersion in elevations of points, and the context in which the points are located. The algorithm is able to classify vegetation in both rural and urban areas with an average F1 score of 97.9% and 91.0%, respectively. The point-clouds can contain different types of vegetation and various degrees of canopy densities.

Povzetek: V predlaganem prispevku povzamemo doktorsko disertacijo, ki predlaga algoritem za klasifikacijo točk vegetacije iz podatkov LiDAR. Algoritem ovrednoti območja vegetacije na podlagi statistično visoke razpršenosti višin točk v kombinacji s kontekstom v katerem se točke nahajajo. Algoritem klasificira točke vegetacije v urbanih in neurbanih področjih s 97% in 91% povprečnim rezultatom F1. Oblaki točk lahko vsebujejo različne tipe vegetetacije z različno gostoto olistanosti.

1 Introduction

The potential of the data obtained by the aerial LiDAR (**Li**ght **D**etection **a**nd **R**anging) systems has been utilised increasingly by a variety of scientific and industrial applications. Data acquisition using LiDAR produces a point-cloud, in which the individual points are calculated based on the time delay between the emitted and detected laser beam. While the obtained point-clouds from a plane-mounted LiDAR can represent the underlying Earth's surface accurately, the entity to which a given point belongs (e.g. ground, building, vegetation) is not known. This contextual knowledge is crucial for a variety of applications, such as environmental simulations, urban planning, or the generation of a canopy height model. Thus, preprocessing, using a classification algorithm is usually applied, in which each point is correlated with one of the predetermined classes.

This paper is a summary of a PhD thesis [1] (and the corresponding paper [4]), which proposed an algorithm for classification of vegetation points within LiDAR data. Vegetation can be particularly challenging to identify, as the classification model should be universal enough to cover the various sizes, shapes and canopy densities of different vegetation but, at the same time, still differentiate it from other surface objects (e.g. houses, cars, fences). The following section outlines the proposed classification algorithm, while section 3 evaluates it. Section 4 concludes the paper.

2 Classification algorithm

Clusters of points that represent vegetation are, in most cases, defined by statistically large dispersions in elevation. This is caused by the vegetation's non-linear shape and porosity, as the laser beam can usually penetrate the canopy and capture many points within, or even under, the vegetation. The mentioned properties can be characterized efficiently by modifying the LoFS (**Lo**cal **F**itting of **S**urfaces) method [2]. Namely, by locally fitting planes on the LiDAR-derived surface and evaluating the fitting error using all points in the fitted area, a distinction can be made between vegetation and most of other man-made objects. Larger fitting errors are expected in the former case, while the latter ones usually produce errors that are identifiably smaller.

The remaining non-vegetation that does not conform to this definition (i.e. also produces a larger fitting error) is handled using contextual analysis which defines: 1) Attached objects 2) Overgrowing vegetation and 3) Small objects. Attached objects represents a transition between areas (e.g. a wall, balcony or chimney). Such objects can be identified (and subsequently removed) using spatially-variant morphological dilation [3] where all non-ground areas with small fitting errors are dilated. The extent of the dilation is controlled locally, using a structuring element with the radius equal to the distance from the nearest ground. Spatially-variant dilation is used similarly on areas with high fitting errors to identify overgrown vegetation. However, the radius, in this case, is dependent on the height difference between a given area and the nearest non-ground area with a small fitting error. Lastly, small objects are removed using connected operators. The final result that includes the described fitting error evaluation and contextual analysis is then mapped onto the individual LiDAR points to get the classification.

3 Results

The algorithm was tested on multiple rural and urban datasets which contained different types of vegetation and degrees of canopy densities. The results were evaluated by counting the false positives, false negatives, true positives and true negatives which served as an input for the calculation of the well-established F1 score. An average F1 score of 97.9% was achieved for rural and 91.0% for urban environments which, because of the more complex geometry, tend to be more difficult to classify. The use of contextual analysis improves the results in urban areas significantly. Namely, by removing the attached object, small object and handling overgrown vegetation, the F1 score is improved, on average, by 8.8%, 1.1% and 1.0%, respectively.

4 Conclusion

The PhD thesis presented an algorithm for the classification of LiDAR data, which classifies vegetated areas with high accuracy in characteristically different point-clouds (rural environment, urban environment, different types of vegetation, various leaf-on conditions). Additionally, the algorithm can be used in most classification scenarios that contain aerial point-clouds, as it relies only on geometrical features of the point-cloud and the subsequent contextual analysis.

Acknowledgement

The authors acknowledge the financial support from the Slovenian Research Agency (research core funding No. P2-0041 and project J2-6764)

References

[1] D. Horvat (2017) Algorithm for classification of vegetation in LiDAR data, *Doctoral dissertation, Faculty of Electrical Engineering and Computer Science, University of Maribor (in Slovene).*

[2] D. Mongus, N. Lukač and B. Žalik (2014) Ground and building extraction from LiDAR data based on differential morphological profiles and locally fitted surfaces, *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 93, pp. 145–156.

[3] N. Bouaynaya and Mohammed Charif-Chefchaouni (2008) Theoretical Foundations of Spatially-Variant Mathematical Morphology Part I: Binary Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, No. 5, pp. 145–156.

[4] D. Horvat, B. Žalik and D. Mongus (2016) Context-dependent detection of non-linearly distributed points for vegetation classification in airborne LiDAR, *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 116, pp. 1–14.

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 900 staff, has 700 researchers, about 250 of whom are postgraduates, around 500 of whom have doctorates (Ph.D.), and around 200 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of **Slove**nia (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

From the Jožef Stefan Institute, the Technology park “Ljubljana” has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

Part of the Institute was reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project was developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park is a shareholding company hosting an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Higher Education, Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of the Economy, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel.:+386 1 4773 900, Fax.:+386 1 251 93 85
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Public relations: Polona Strnad

INFORMATICA

AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS

INVITATION, COOPERATION

Submissions and Refereeing

Please register as an author and submit a manuscript at: http://www.informatica.si. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible from typing errors to global philosophical disagreements. The chosen editor will send the author the obtained reviews. If the paper is accepted, the editor will also send an email to the managing editor. The executive board will inform the author that the paper has been accepted, and the author will send the paper to the managing editor. The paper will be published within one year of receipt of email with the text in Informatica MS Word format or Informatica LaTeX format and figures in .eps format. Style and examples of papers can be obtained from http://www.informatica.si. Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the managing editor.

SUBSCRIPTION

Please, complete the order form and send it to Dr. Drago Torkar, Informatica, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: drago.torkar@ijs.si

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than twentythree years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica web edition is free of charge and accessible at http://www.informatica.si.
Informatica print edition is free of charge for major scientific, educational and governmental institutions. Others should subscribe.

Informatica WWW:

http://www.informatica.si/

Referees from 2008 on:

Informatica

An International Journal of Computing and Informatics

Web edition of Informatica may be accessed at: http://www.informatica.si.

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Litostrojska cesta 54, 1000 Ljubljana, Slovenia.
The subscription rate for 2017 (Volume 41) is
– 60 EUR for institutions,
– 30 EUR for individuals, and
– 15 EUR for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

Typesetting: Borut Žnidar.
Printing: ABO grafika d.o.o., Ob železnici 16, 1000 Ljubljana.

Orders may be placed by email (drago.torkar@ijs.si), telephone (+386 1 477 3900) or fax (+386 1 251 93 85). The payment should be made to our bank account no.: 02083-0013014662 at NLB d.d., 1520 Ljubljana, Trg republike 2, Slovenija, IBAN no.: SI56020830013014662, SWIFT Code: LJBASI2X.

Informatica is published by Slovene Society Informatika (president Niko Schlamberger) in cooperation with the following societies (and contact persons):
Slovene Society for Pattern Recognition (Simon Dobrišek)
Slovenian Artificial Intelligence Society (Mitja Luštrek)
Cognitive Science Society (Olga Markič)
Slovenian Society of Mathematicians, Physicists and Astronomers (Marej Brešar)
Automatic Control Society of Slovenia (Nenad Muškinja)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Stane Pejovnik)
ACM Slovenia (Matjaž Gams)

Informatica is financially supported by the Slovenian research agency from the Call for co-financing of scientific periodical publications.

Informatica is surveyed by: ACM Digital Library, Citeseer, COBISS, Compendex, Computer & Information Systems Abstracts, Computer Database, Computer Science Index, Current Mathematical Publications, DBLP Computer Science Bibliography, Directory of Open Access Journals, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter, Scopus, Zentralblatt Math

Volume 41 Number 3 September 2017 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Informatica **41** (2017) Number 3, pp. 259–381