Timely segregation of critical/noncritical nodes is extremely crucial in mobile ad hoc and sensor networks. Most of the existing segregation schemes are centralized and require maintaining network wide information, which may not be feasible in large-scale dynamic networks. Moreover, these schemes lack rigorous validation and entirely rely on simulations. We present a localized algorithm for segregation of critical/noncritical nodes (LASCNN) to the network connectivity. LASCNN establishes and maintains a k-hop connection list and marks a node as critical if its k-hop neighbours become disconnected without the node and noncritical otherwise. A noncritical node with more than one connection is marked as intermediate and leaf noncritical otherwise. We use both formal and nonformal techniques for verification and validation of functional and nonfunctional properties. First, we model MAHSN as a dynamic graph and transform LASCNN to equivalent formal specification using Z notation. After analysing and validating the specification through Z eves tool, we simulate LASCNN specification to quantitatively demonstrate its efficiency. Simulation experiments demonstrate that the performance of LASCNN is scalable and is quite competitive compared to centralized scheme with global information. The accuracy of LASCNN in determining critical nodes is 87% (1-hop) and 93% (2-hop) and of noncritical nodes the accuracy is 91% (1-hop) and 93% (2-hop).

1. Introduction

Mobile ad hoc and sensor networks (MAHSNs) are captivating significant attention from research community because of their suitability for enormous range of applications. MAHSNs may include MANETs, wireless sensor networks, mobile sensor networks, wireless sensor and actor networks, and mobile robot networks. Nodes (stationary and mobile) in these networks are expected to establish and maintain a connected wireless network and operate autonomously in unattended setups. Due to ad hoc nature, limited energy and transmission range nodes have to rely on multihop communication; that is, a node forwards a message to the next node on the path towards the destination. The nodes organize themselves because there is no central entity to manage the network. Therefore, maintaining network connectivity is extremely desirable in order for the network to stay functional.

Nonetheless, random deployment, frequent mobility, and network dynamics inherently introduce critical (cut vertex in graph theory) nodes to the network connectivity. The term critical and cut vertex will be used interchangeably afterward. A node is cut vertex or critical if its removal disconnects the network into two or more isolated components, noncritical otherwise. Noncritical nodes can further be categorized into leaf and intermediate. A noncritical node with only one connection is called leaf, intermediate otherwise. Several reasons such as harsh and inhospitable application environment make nodes susceptible to physical damage and component malfunction. Moreover, these networks are subject to various attacks; therefore, failure of nodes is not surprising. Failure of a critical node partitions the network into two or more disjoint segments and may disrupt the network operation. The network stays connected despite the loss of a noncritical node. Thus, it is imperative to distinguish critical/noncritical
nodes to the network connectivity in order to assess vulnerabilities against critical node failures and provide precautionary means for survivability. The autonomous and self-organizing nature of these large-scale dynamic networks requires localized and distributed schemes. Moreover, nodes with limited resources should use agile and lightweight procedure and avoid sophisticated diagnostics.

Most of the published schemes are centralized and require globalized network state information for determining critical nodes. Centralized schemes require maintaining up-to-date network wide information that may not be practical for large-scale dynamic networks. Moreover, it becomes almost impossible to identify critical/noncritical nodes at regular intervals due to frequent topology changes (due to mobility and network dynamics). Furthermore, most of the existing localized schemes do not determine and segregate noncritical nodes and entirely rely on nonformal techniques such as simulation for performance evaluation of nonfunctional properties (e.g., critical node count) of algorithms. Simulation-based results only help in quantitative performance analysis but do not ensure correctness of the approach which is vital in complex and safety-critical applications.

Formal methods have been meritoriously used for modelling, specifying, analysing, and verifying the properties of complex and safety-critical systems. To describe and analyse properties of software systems, these methods employ discrete mathematics based notations and computer-assisted tools. Model-oriented formal methods such as Z notation [1] construct both statics and dynamics of a system. Unlike descriptive languages, Z uses standard mathematical notations for specification of abstract properties of complex systems and organizes them into smaller components. The abstraction and structuring features of Z minimizes the complexity of the system by eliminating unnecessary details and dangle a simplified model for simulation. However, formal methods can only help in verifying functional properties of the approach and do not provide quantitative means for performance validation which is important in resource-constraint networks. Therefore, we advocate and promote the use of complementary verification and validation techniques for dynamic, complex, distributed, and resource constraint MAHSN to be deployed for safety-critical applications.

This paper presents a fully distributed and localized algorithm for segregation of critical/noncritical nodes (LASCNN) that opts to distinguish connectivity-centric critical/noncritical nodes to assess the network vulnerability against critical node failures and provide precautionary means for survivability. Nodes establish and maintain a connection list based on localized information (i.e., k-hop) and employ LASCNN to process the list in determining critical/noncritical nodes. LASCNN examines the list and marks the node as critical if its k-hop neighbours become disconnected without the node, noncritical otherwise. LASCNN marks a noncritical node with more than one connection as intermediate, leaf otherwise. We use complementary formal specification and simulation for verification and validation of functional and nonfunctional properties of LASCNN. We model MAHSN as a dynamic graph and transform LASCNN to equivalent formal specification using Z notation.

After analysing and validating the specification through Z tool, we simulate the LASCNN specification to quantitatively demonstrate its efficiency. Simulation results confirm that the performance of LASCNN is scalable and is quite competitive compared to centralized scheme with global network information. The accuracy of LASCNN in determining critical nodes is 87% (1-hop) and 93% (2-hop) and noncritical nodes 91% (1-hop) and 93% (2-hop). To the best of our knowledge, this is the first work that employs formal and nonformal techniques for segregation of critical/noncritical nodes.

The rest of the paper is organized as follows. We elaborate the system model and problem statement in Section 2. Section 3 provides insights to works related to this research. The details of the LASCNN algorithm are presented in Section 4. Formal specification of the algorithm is described in Section 5. Section 6 contains the results and analysis. The conclusion of the paper is presented in Section 7.

2. System Model and Problem Statement

LASCNN is applicable to MAHSNs. The nodes are randomly deployed in an area of interest and are assumed to be able to move around. After deployment, nodes are assumed to discover each other through hello messages and form a connected network. Nodes have limited resources in terms of energy and computation. Nodes are assumed to acquire their positions through GPS or other range-based localization techniques, for example, [2], and periodically update their current location to their neighbours. The communication range ($R_c$) of a node is limited and refers to a maximum Euclidean distance that its radio can reach and is equal for all nodes.

The network survivability primarily depends on the position of nodes in the network topology. For example, failure of a critical node, such as N4 in Figure 1, will divide the network into three disjoint segments. Meanwhile, loss of a noncritical node, such as N17 (intermediate) and N18 (leaf), does not affect network connectivity. To identify critical nodes, the published schemes can be categorized based on maintaining topology information into globalized and localized. Globalized schemes require maintaining network wide information. However, maintaining such a level of information is infeasible for large-scale dynamic networks. On the other hand, localized schemes only require k-hop information. We argue that localized schemes better suit MAHSNs because of their
distinguished characteristics such as size and dynamic nature. LASCNN belongs to this category.

3. Related Work

Identifying cut vertices in graph theory with respect to connectivity has been extensively investigated [3, 4]. To determine cut vertices in graph, a number of algorithms have been proposed in the literature. However, these algorithms are difficult or impossible to be adopted for MAHSNs. These algorithms can be categorized into centralized and distributed based on execution. Centralized algorithms [5, 6] are required to maintain entire topology information which involves very high communication overhead and thus may not be suitable for large-scale dynamic networks. On the other hand, a globalized distributed implementation of centralized algorithm based on depth first search (DFS) does not require network wide information at any node and can be used for MAHSNs. For example, DDFS (distributed depth-first search) and CAM [7] build a DFS tree in a distributed fashion. However, results [8] show that these algorithms are not suitable for large-scale dynamic networks due to long time execution and high communication cost. Few localized and distributed algorithms to determine critical nodes in ad hoc and sensor networks have been proposed in [8, 9]. These algorithms only indicate network vulnerabilities by identifying critical nodes and do not provide survivability mechanism by determining noncritical nodes. Moreover, the accuracy of determining critical nodes is less.

Generally, determining critical nodes has been investigated in various network contexts such as P2P [10] and overlay networks [7, 11–13]. Several reasons such as ad hoc and wireless nature and dynamic conditions make these approaches unsuitable for MAHSNs. The existing critical node detection schemes can broadly be categorized into prepartitioning and postpartitioning. Prepartitioning schemes are designed to determine critical nodes in advance, while postpartitioning schemes perform cut detection after it occurs. Most of the published work focuses on postpartitioning schemes. Postpartitioning schemes are designed either to perform cut detection [14–16] or only investigate the impact of network disconnection [17, 18]. Prepartitioning schemes to determine critical nodes based on limited information have received little attention. Prepartitioning algorithms such as [8, 9] only provide advance network vulnerability assessment by identifying critical nodes to the network connectivity. However, our proposed LASCNN algorithm inherently dangles survivability mechanism by accurately segregating noncritical nodes as well and can be used for partitioning avoidance [19], coverage, and connectivity restoration [20–22].

Most of the existing critical node detection schemes including [8, 9] lack rigorous validation and only use nonformal techniques (e.g., simulation) [23]. Simulation results are only effective for quantitative performance validation and do not guarantee correctness of the approach that is vital for safety-critical applications. Moreover, recent investigations [24, 25] have shown increasing concerns on the dependability of simulation results for wireless networks and argued the scantiness of experimental results [26]. These studies observed huge diversity in the simulation results compared to real experiments. This is because focus of nonformal techniques is on performance analysis of nonfunctional properties (e.g., performance-related metrics, such as resource and delay). Nonformal techniques are unable to verify the correctness and detect design errors while modeling functional behaviors (e.g., interactions) of algorithms. Recently, some researchers have advocated the use of formal methods for algorithm validation in wireless networks [23, 27]. However, most schemes only use formal specification and did not use complementary verification and validation techniques. A formal specification based conformance testing methodology was proposed in [28] to validate the DSR routing protocol for MANET. The authors in [29] used real-time maude for modeling, performance estimation, and model checking of the OGDC algorithm for sensor networks.

The authors in [30] proposed a critical link (i.e., bridge) detection algorithm. Unlike [30], focus of our work is on segregating critical/noncritical nodes. A preliminary version of this work was presented in [31]. However, to the best of our knowledge, none of the existing critical node detection scheme including [31] further segregates noncritical nodes into intermediate/leaf and has been analysed and validated through formal and nonformal techniques.

4. Localized Algorithm for Segregation of Critical/Noncritical Nodes

The distributed and highly localized algorithms are more suitable for large-scale dynamic networks such as MAHSNs to support high mobility, scalability, robustness, and resource optimization. Localized algorithms can quickly determine critical/noncritical nodes with far less computation and communication overhead as they only process localized information. The underlying principle of the proposed LASCNN is to check the connectivity of neighbours for each node based on localized information to determine if the node is critical/noncritical. The balance of this section explains our LASCNN algorithm.

4.1. Establish and Maintain Connection List. Unlike globalized schemes, LASCNN establishes and maintains a connection list based on localized information to quickly determine critical/noncritical nodes with minimum communication overhead. Moreover, LASCNN avoids unnecessary computation overhead (i.e., memory and processing) in maintaining adjacency matrix. Instead, it maintains a duplication-free pair-wise connection list (ConnList) for each node based on k-hop knowledge.

After deployment, nodes discover each other through hello messages and establish a 1-hop and 2-hop ConnList. The 1-hop ConnList contains the direct neighbours of a node in the form of a pair. For example, a 1-hop ConnList of node N1 for the network segment of Figure 1 is shown in Table 1 where each row represents an undirected (or bidirected) connection between pair of nodes. Similarly, 2-hop connections can be acquired through 1-hop neighbours. These are the
connections between 1-hop neighbours and between a 1-hop and a 2-hop neighbour, and possibly connection between 2-hop neighbours can be determined based on the position of nodes. Table 1 shows a 2-hop ConnList of N11 for the similar network segment. For example, N11 can determine the connection between its 2-hop neighbours N13 and N14 based on their position. Similarly, generalizing k-hop ConnList will contain connections between k-hop neighbours, between a k-hop and a (k-1)-hop neighbour, and if some k-hop nodes are connected.

The accuracy of determining critical/noncritical nodes depends on the connection list besides algorithm. Since, topology of MAHSNs continuously changes due to network dynamics such as mobility; therefore, the connection list must be updated. Nodes in these networks continuously exchange status update messages containing ID, location, and so forth based on the localized information a network is supposed to maintain. Upon receiving an update message from a neighbour, a node forwards the message to (k-1) hop neighbours further depending on the localized information a network is supposed to maintain. For example, while maintaining 2-hop ConnList, N12 will update status of N11 to N13. A noticeable point is that all nodes need to maintain their connection list based on status updates; therefore, it becomes infeasible to maintain more topology information in large-scale dynamic networks.

### 4.2. LASCNN Algorithm

Once all nodes establish a k-hop connection list (e.g., 1-hop and 2-hop), each node independently processes its list to determine whether it is critical/noncritical. Basically, a node N needs to determine from the k-hop ConnList whether its k-hop neighbours remain connected or not. If k-hop neighbours of N remain connected without it, N is critical, noncritical otherwise. While processing ConnList, each node populates a k-hop connected neighbours list ConnNeighbors that contains entries of connected neighbours for the node. The detail process of LASCNN is as follows.

If there is only one connection in the ConnList then the node is a leaf noncritical because absence of such node will not affect network connectivity. In case of more connections, N starts processing its list from first connection, that is, ActiveConn. The processing of ActiveConn only proceeds if N is not involved in the connection pair; otherwise, the next connection becomes the ActiveConn. The node pair in ActiveConn will be inserted in the ConnNeighbors if it is empty. Otherwise, if there is a common node in ActiveConn and ConnNeighbors, the other node will be put into ConnNeighbors if it is not already there. The next connection will become the ActiveConn and will be processed in the same manner until all the connections are processed. The next iteration of processing ConnList will commence if size of ConnNeighbors increased in the current iteration. In this way, all the connected neighbours of N will become part of ConnNeighbors. If the size of k-hop ConnNeighbors is less than the number of k-hop neighbours of N, N is critical, intermediate noncritical otherwise.

For example, N11 processes its 1-hop ConnList provided in Table 1 and executes LASCNN to determine whether it is critical or not. Since there is more than one connection in the list, therefore it is not a leaf noncritical node. Since N11 is engaged in the ActiveConn (i.e., N7 ↔ N11), therefore it will not be processed and the next connection becomes the ActiveConn (i.e., N8 ↔ N11). Similarly, none of the ActiveConn will be processed because all the four connections in the 1-hop ConnList involve N11. Consequently, none of the nodes will be inserted in the 1-hop ConnNeighbors list of N11. Therefore, N11 is 1-hop critical because the size of its ConnNeighbors list is less than the number of its neighbours.

Similarly, 2-hop ConnList (i.e., Table 1) is processed to determine whether N11 is critical or noncritical. Again, the node is not a leaf noncritical and the first four connections will not be processed because they involve N11. However, N5 and N7 will be inserted in to the 2-hop ConnNeighbors list because all the four connections in the 1-hop ConnList involve N11. Consequently, none of the nodes will be inserted in the 1-hop ConnNeighbors list of N11. Therefore, N11 is 2-hop critical because the size of its ConnNeighbors list is equal to the number of its neighbours.

| 1-hop ConnList | 2-hop ConnList |
|----------------|----------------|
| N7 ↔ N11       | N7 ↔ N11       |
| N8 ↔ N11       | N8 ↔ N11       |
| N10 ↔ N11      | N10 ↔ N11      |
| N11 ↔ N12      | N10 ↔ N16      |
| N5 ↔ N7        | N12 ↔ N13      |
| N6 ↔ N8        | N12 ↔ N15      |
| N13 ↔ N14      |                |

**Table 1: 1-hop and 2-hop connection list of node N11.**
4.3. Pseudo Code of LASCNN Algorithm. Pseudo code of LASCNN is shown in Algorithm 1. The algorithm is to run on all nodes of MAHSN in a distributed manner (line 1). A node \( A \) determines itself as a leaf noncritical if the size of its connection list is equal to 1 (lines 2-3). Otherwise, \( A \) picks each connection \( \text{ActiveConn} \) from the list and processes only those connections where \( A \) is not engaged. Pair of nodes involved in the \( \text{ActiveConn} \) will be added in the \( \text{ConnNeighbors} \) list if it is empty (lines 4–12). Otherwise, if there is a common node between \( \text{ActiveConn} \) and \( \text{ConnNeighbors} \), the other node will be included in connected neighbours list and next connection will be processed likewise (lines 13–22). When all the connections are processed, the node \( A \) checks if the number of nodes in the \( \text{ConnNeighbors} \) list are less than the number of neighbours; then, \( A \) is critical, intermediate noncritical otherwise (23–28).

5. Formal Specification of LASCNN

As stated earlier, most of the published research on MAHSNs entirely relies on conventional techniques such as simulation for performance evaluation of nonfunctional properties of algorithms. Such techniques are only useful for quantitative performance analysis and do not focus on the correctness of the approach which is vital in complex and safety-critical applications. This section emphasizes the importance and merits of integrating graph theory, formal methods, and Z notation for algorithm verification in MAHSNs. We model MAHSN as a dynamic graph and convert LASCNN to equivalent formal specification through Z notation. The LASCNN specification is analyzed and validated using Z eves tool.

5.1. Why Formal Specification? Traditional methods such as simulation are insufficient for rigorous verification and
5.2. Reasoning of Using Z Notation and Graph Theory. Z notation is a model centered approach based on sets, sequences, bags, relations, functions, and predicate logic which is used at an abstract level of specification of systems. The expressive power of Z makes it most feasible candidate for modeling and specification of distributed systems. Z allows organizing a system into its smaller components using a powerful data structure named schema. The schemas also support defining a way in which state space of a system can be described and manipulated. Formal specification described using Z can further be refined and transformed to an implemented system.

Applications of graph theory are significant in all areas of computer science. Communication and network systems cannot be analyzed and optimized without the usage of algorithms of graph theory. More important is that theory of graphs has been enriched in last couple of decades, particularly, in the area of computer science. However, its applications are not limited to only computer networks but are extended to other social and complex networks. The importance of using graphs for representation of the problem of MAHSN is obvious. For example, we would like to know whether the graph is connected or which one is the shortest path between any two objects. Further, the model in graph can be used to update the dynamic changes efficiently rather than recompute it from scratch every time.

As Z is based on set theory and first order predicate logic, the sets and relations are fundamentals of describing composite types in Z. On the other side, the structures of graphs are similar to Z structures; that is, it contains collection of objects and establishment of relations among the objects. Hence, any model in graph theory can easily be transformed to Z for further analysis and proof generation.

5.3. Formal Specification of LASCNN Algorithm. In this section, formal specification of LASCNN is described using Z notation. In MAHSNs, the topology is subject to change, for example, after joining or leaving of nodes or links. That is why the network topology of MAHSN is represented by a model using dynamic graph. Then, state space of the topology is updated if a node is connected or disconnected from the network. Next, formal definition of the network and its components is described. Finally, formal procedure of segregation of critical and noncritical nodes is presented. Formal analysis is provided using Z/Eves tool.

The topology is denoted by the schema DynamicGraph (see Scheme 1) which consists of set of nodes and set of edges. The set of nodes is a power set of Node where Node is assumed as a set at an abstract level of specification in Z notation. The set of edges is a relation over the node-set to itself. The node-set and edge-set are contained in the first part of the schema and invariants are defined in the second part.

The network topology is updated by the operation AddNodes (see Scheme 2) in terms of a schema that inserts nodes and associated edges of the graph. The schema consists of three components, namely, ΔDynamicGraph, nodes to be added, and connections which represent associated edges. The symbol Δ means that state of the network is changed; that is, it shows a relationship between previous and new states of the network. The symbol “?” at the end of variables shows that it is input to the operation.

Pre-/postconditions: (i) the node to be added is known to the network; that is, it belongs to the set of all the nodes of the network. (ii) Every node to be added is end point of an edge to be added. (iii) Every edge to be added has an end point which is a node to be added. (iv) Any node in the domain of connection relation does not belong to its image set under the relation. That is, node is not connected to itself. (v) The edge-set is updated by stating that one end point of an added edge is in the previous node-set and the other is in the new node-set.

In case of removal of edges, the network is updated by the operation DeleteNodes schema (see Scheme 3). The schema consists of two components, namely, ΔDynamicGraph and nodes, which are assumed as input to the operation. The first

---

**Scheme 1: DynamicGraph.**

```
[Node]
DynamicGraph
nodes: F Node
edges: Node ↔ Node
∀n: Node | n ∈ nodes·∃e: Node × Node | e ∈ edges· e.1 = n ∨ e.2 = n
∀e: Node × Node | e ∈ edges· ∃n1, n2: Node | n1 ∈ nodes ∧ n2 ∈ nodes· e = (n1,n2)
∀n1, n2: Node | (n1,n2) ∈ edges· (n2,n1) ∈ edges
```
component has same meaning as defined above and second one represents set of nodes to be deleted from the network. In precondition, it is stated that a node to be deleted is a member of the network. In postcondition, the network is updated by removing set of nodes and the associated set of edges.

The Sensor schema contains six variables, namely, identifier, type, neighbors, 1-hop list, 2-hop list, and connectivity, as in Scheme 4. The variable type has two values, that is, either critical or noncritical. As a sensor can communicate only with its neighbors, hence, neighbors’ information is required. The connectivity variable is used to check status of the node which is either connected or disconnected to the network. The above components are put in definition part and invariants are described in second part of the schema in terms of predicates.

Invariants: (i) the identifier of the sensor node does not belong to its neighbors. (ii) The first element in the ordered pairs of 1-hop list is the sensor itself. (iii) Either the first element in the ordered pairs of 2-hop list is the sensor itself or there exists another ordered pair, in the 2-hop list, whose first element is the sensor identifier. (iv) All the second elements in the 1-hop list of the sensor belong to its neighbors. (v) All the elements in the 2-hop list, except sensor identifier, belong to its set of neighbors. (vi) If 1-hop list is nonempty then node is assumed as 1-hop connected. (vii) If there exists an order pair in the 2-hop list in which sensor identifier is neither first nor second element of the ordered pair then sensor node is assumed as 2-hop connected.

The topology for mobile ad hoc and sensor network is denoted by the schema MAHSN given in Scheme 5 which consists of two components, that is, sensors and DynamicGraph. The first component defining dynamic graph is defined above. The second component sensors represents set of all the objects of the network, which is a type of power set of sensor.

Invariants: (i) every sensor identifier is a node in the network topology. (ii) Every node in the network topology is represented as a sensor identifier. (iii) Every link between sensor and its neighbor is represented as an edge in the network topology. All the direct neighbors of a sensor form an edge with the sensor in the network topology. (iv) For any two sensors there is a path in the network topology. That is, for every sensor there is, at least, one sensor responsible for receiving information and delivering to the required sensor through an appropriate and efficient path. The path is defined as a sequence of nodes. The first and last elements in the sequence are source and target sensors for sharing the required information. It is verified that for every pair of consecutive elements in the sequence, it is an ordered pair and an edge in the network topology. It is noted that there does not exist any loop in the communication path between source and target sensors of the sequence. That is, all the elements in path sequence are disjoint.

Formal specification of the proposed algorithm to segregate critical/noncritical nodes is described using LASCNN schema given in Scheme 6. The schema consists of four components, that is, ΔMAHSN, set of critical nodes criticals!, set of noncritical criticals!, and set of neighbors. The first component is defined above to represent MAHSN.
The second and third components are output variables used to store information for critical and noncritical nodes. The fourth and last component is a temporary variable used in the formal procedure to check if all the neighbors of the graph are discovered. The symbol exclamation mark "!" is used after output variables in Z notation.

Pre-/postconditions: (i) The node determines itself to be noncritical (i.e., leaf) if the size of its connection list is equal to 1. (ii) For every ordered pair in the connected list of a sensor, the ordered pair which does not contain the sensor itself is identified. Then, pair of nodes in the ordered pair is added in the temporary set of neighbors defined in first part of the schema. If there is any common node between elements of ordered pair and the temporary neighbors list, the other node of ordered pair will be included in connected neighbors list. This process is repeated for all the ordered pairs of the connected list. After processing of all the ordered pairs, it is checked; if the number of nodes in the temporary neighbor lists less than the number of neighbors of the sensor node, the node is identified as a critical; otherwise, it is intermediate noncritical. (iii) The lists of critical and noncritical nodes of the entire network are updated accordingly.

5.4. Model Analysis through Z Eves Tool. Model analysis is provided in this section for the formal specification of LASCNN using Z/Eves toolset. Although use of computer tools improves quality of software systems, it does not give guarantee about precision of the models. We are not aware about the availability of any software which may assure complete correctness of the computer models. It means even if the formal specification is well-defined and well-written using any of the formal specification language it may contain inconsistencies and potential errors. That is, an art of writing formal specification does not give guarantee that the proposed system is correct, complete, consistent, and reliable.
On the other hand, if the specification is analyzed and validated using rigorous computer tools it increases quality and confidence over the system to be developed. Of course it is important to note how the specification is analyzed and evaluated.

The Z/Eves is a powerful tool used for analysing the Z specification of the algorithm underhand. A snapshot of the specification analysis of the schema DynamicGraph is presented in Figure 2. The first column on the left side of the figure shows syntax checking of the specification. The second column represents its proof correctness. The third column describes formal specification to be checked and analyzed. Each part of the specification is evaluated separately. The symbol “Y” on the left of every part of the specification shows that the formal specification is correct syntactically and proof is also correct. The symbol “N” which does not appear in the figure represents that specification is not correct and properties of the specification need to be revised. It is noted that many inconsistencies were identified initially in the specification. A series of refinements were done to complete the model analysis reaching the final specification. All schemas of the specification of the proposed algorithm are checked and analysed to prove that specification is correct in syntax and has a correct proof. Some schemas of the formal specification were proved using reduction techniques available in the toolset.

Summary of the model analysis is presented in Table 2. In first column of the table, name of schema is given. The symbol “Y” in column 2, as stated above, indicates that all schemas
Table 2: Results of model analysis.

| Schema name   | Syntax type | Domain check | Reduction | Proof |
|---------------|-------------|--------------|-----------|-------|
| DynamicGraph  | Y           | Y            | Y         | Y     |
| AddNodes      | Y           | Y            | Y         | Y     |
| DeleteNodes   | Y           | Y            | Y         | Y     |
| Sensor        | Y           | Y            | Y*        | Y     |
| MAHSN         | Y           | Y            | Y         | Y*    |
| LASCNN        | Y           | Y            | Y         | Y*    |

are well-written syntactically. Similarly, domain checking, reduction, and proof by reduction are represented in columns 3, 4, and 5, respectively. The character “Y∗” annotated with “∗” notation shows that the schema is proved by performing reduction analysis on the predicates part to make specification more meaningful.

6. Results and Analysis

We developed a customized MAHSN simulator based on the formal specification presented in the previous section to validate the performance of LASCNN. This section describes the experiment setup, performance metrics, and experimental results.

6.1. Experiment Setup and Performance Metrics. In the experiments, we have generated connected topologies with varying network density that consist of (20–250) nodes. Nodes are deployed randomly in an area of 1000 m × 600 m. We have varied the transmission range of nodes (25–125) during experiments. The following metrics are used to assess the performance.

(i) The critical node detection ratio is the percentage of nodes that are detected as critical. This indicates the network connectivity and vulnerability. This metric is used to measure the effectiveness of the proposed algorithm.

(ii) The noncritical detection ratio is the proportion of nodes that are determined as noncritical (i.e., intermediate/leaf). This metric indicates the robustness of the network against node failures and help in provisioning survivability mechanism.

(iii) The accuracy of determining critical/noncritical nodes: this metric reflects the probability that a node detected by LASCNN is really a critical/noncritical confirmed through globalized algorithm. This gauges the accuracy of LASCNN in detecting critical/noncritical nodes.

We have used the following parameters to vary the network configuration in the experiments.

(i) The network density (N) affects the node count and the degree of network connectivity.

(ii) The radio range (r) is the reachability of nodes and affects the network connectivity.

To prove the effectiveness and efficiency, we compare the performance of our algorithm with localized information based on the above-mentioned metrics to that with globalized knowledge. The globalized approach uses network wide information in determining critical/noncritical nodes and thus is perfectly accurate.

6.2. Simulation Results and Analysis. The simulation experiments involve randomly generated connected topologies with varying network densities and radio ranges. The network densities have been set to 20, 40, 60, 80, 100, 150, 200, and 250. The radio range “r” of nodes is changed among 25, 50, 75, 100, and 125. For each experiment, simulation results of 15 different network topologies are averaged and reported. All results are subject to 90% confidence interval analysis and stays within 6%–10% of the sample mean.

6.2.1. Critical Node Detection Ratio. The results for the detection ratio of critical (C) nodes are shown in Figure 3. Both Figures 3(a) and 3(b) clearly show that the performance
of LASCNN with localized information (i.e., 1-hop and 2-hop) remains consistent and is not much affected while varying the network density and radio range. This suggests that our algorithm is equally effective for both sparse and dense networks. Moreover, the performance of LASCNN in determining critical nodes is quite competitive compared with the globalized information that can accurately detect all the critical nodes. The results suggest that the ratio of determining critical nodes hone with the more network state information available at each node. However, maintaining more topology information increases the computation (memory and processing) and communication cost. Basically,
some nodes with localized information determine themselves as critical while they are not cut vertices globally. More importantly, no critical node is missed. The number of critical nodes decreases with the increase in node density and communication range. This is because connectivity of the network improves in both cases, and thus alternative paths become available.

6.2.2. Noncritical Node Detection Ratio. Figure 4 shows the detection ratio of noncritical nodes while varying \( N \) and \( r \). Both Figures 4(a) and 4(b) clearly indicate that the performance of LASCNN is very close to a globalized scheme and is not much affected while changing \( N \) and \( r \) which shows great scalability. This is because a node locally determined as noncritical is indeed globally noncritical. Figure 4(a) shows that the performance of LASCNN almost matches with the globalized scheme for very low and high density networks. In both the cases, alternative paths are shorter and known to localized algorithm. The performance of localized algorithm slightly degrades with the long radio range as is evident from Figure 4(b). This is because it is almost impossible to discover long alternative paths through localized information.

Figures 4(c)–4(f) show the detection ratio of intermediate (NC_I) and leaf (NC_L) noncritical nodes while changing node density and radio range. As expected, Figures 4(c) and 4(d) show that the number of intermediate noncritical nodes increases with the increase in \( N \) and \( r \) because network connectivity improves in both cases. Again, the performance of LASCNN is marginally lower than globalized scheme. This is due to inaccuracy of marking some intermediate noncritical nodes as critical while they are not really critical as explained above. However, LASCNN segregate all the leaf noncritical nodes and there is no performance deviation from globalized scheme as is evident from Figures 4(e) and 4(f). Both the figures show that increasing network connectivity decreases the number of leaf nodes.
6.2.3. Accuracy of Determining Critical/Noncritical Nodes. Figures 5 and 6 report the accuracy of determining critical (C) and noncritical (NC) nodes, respectively. Both Figures 5(a) and 5(b) suggest that some nodes are inaccurately determined as critical while they are not globally cut vertices. More importantly, no critical node is missed, and hence such an inaccuracy is not a major concern compared to the overhead of maintaining network wide information at each node. Figure 5(a) shows that the maximum inaccuracy of determining critical nodes with 2-hop and 1-hop information is only 7% and 13%, respectively. In other words, at least 93% (2-hop) and 87% (1-hop) nodes are accurately determined as critical. Similar observation can be made for Figure 5(b). The results in Figure 5 indicate that the inaccuracy of determining critical nodes also depends on the network topology, node density, and radio range. The inaccuracy is low when alternative paths are shorter and are known locally.

Figure 6 shows the accuracy of determining noncritical nodes while varying the network density and radio range. Both Figures 6(a) and 6(b) clearly indicate that the noncritical nodes determined based on 1-hop and 2-hop information are indeed noncut vertices globally as well. In other words, our localized scheme determines 91% of the noncritical nodes with 1-hop and 93% with 2-hop as is evident from Figure 6(a). Figure 6(b) provides similar results while varying “r.” However, our localized scheme may determine some noncritical nodes as critical due to limited information.

7. Conclusion and Future Work

This paper has presented LASCNN, a localized and distributed algorithm to segregate critical/noncritical nodes. In LASCNN, each node establishes and maintains a k-hop neighbour connection list and processes the list to check neighbours’ connectivity. Based on the list, LASCNN determines a node as critical, if its neighbours become disconnected without the node, noncritical otherwise. A noncritical node with one connection is marked as leaf; intermediate otherwise. The functional and nonfunctional properties of LASCNN are verified and validated through formal and non-formal approaches. MAHSN is modelled as a dynamic graph to convert LASCNN into equivalent formal specification using Z notation. The specification is analyzed and validated using Z eves tool. A customized simulator based on specification was developed. The performance of LASCNN has been validated through extensive simulations. Simulation results confirmed the effectiveness of LASCNN for both sparse and dense networks.

It is worth mentioning that a well-written specification can effectively strangle erroneous design errors in simulations which ultimately improve the dependability of simulation results. Our future plan is to validate the results of LASCNN on prototype network of mobile robots.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Research Center of College of Computer and Information Sciences, King Saud University. The authors are grateful for this support.

References

[1] J. M. Spivey, The Z Notation: A Reference Manual, Prentice-Hall, 1989.
[2] A. Youssef, A. Agrawala, and M. Younis, “Accurate anchor-free node localization in wireless sensor networks,” in Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference (IPCCC ’05), pp. 465–470, Phoenix, Ariz, USA, April 2005.
[3] D. B. West, Introduction to Graph Theory, Prentice-Hall, 2nd edition, 2001.
[4] F. Buckley and M. Lewinter, A Friendly Introduction to Graph Theory, Prentice-Hall, 2003.
[5] M. Duque-Anton, F. Bruyaux, and P. Semal, “Measuring the survivability of a network: connectivity and rest-survivability,” European Transactions on Telecommunications, vol. 11, no. 2, pp. 149–159, 2000.
[6] D. Goyal and J. J. Caffery, “Partitioning avoidance in mobile ad hoc networks using network survivability concepts,” in Proceedings of the 7th International Symposium on Computers and Communications (ISCC ’02), pp. 553–558, Giardini Naxos, Italy, 2002.
[7] X. Liu, L. Xiao, A. Kreling, and Y. Liu, “Optimizing overlay topology by reducing cut vertices,” in Proceedings of the 16th Annual International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV ’06), Newport, RI, USA, May 2006.
[8] S. Xiong and J. Li, “An efficient algorithm for cut vertex detection in wireless sensor networks,” in Proceedings of the 30th IEEE International Conference on Distributed Computing Systems (ICDCS ’10), pp. 368–377, Genoa, Italy, June 2010.
[9] M. Jorgic, I. Stojmenovic, M. Hauspie, and D. Simplot-Ryl, “Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks,” in Proceedings of the 3rd Annual IFIP Mediterranean Ad Hoc Networking Workshop, pp. 360–371, Bodrum, Turkey, 2004.
[10] J. Cai, Z. Li, and X. Shao, “An analysis of the discovery and elimination algorithm in the P2P environment,” in Proceedings of the 2nd International Conference on Intelligent Computing Technology and Automation (ICICTA ’09), pp. 138–141, Hunan, China, October 2009.
[11] Y. He, H. Ren, Y. Liu, and B. Yang, “On the reliability of large-scale distributed systems—a topological view,” Computer Networks, vol. 53, no. 12, pp. 2140–2152, 2009.
[12] L. Xiaomei, X. Li, and A. Kreling, “A fully distributed method to detect and reduce cut vertices in large-scale overlay networks,” IEEE Transactions on Computers, vol. 61, no. 7, pp. 969–985, 2012.
[13] T.N. Dinh, Y. Xuan, M. T. Thai, P. M. Pardalos, and T. Znati, “On new approaches of assessing network vulnerability: hardness and approximation,” IEEE/ACM Transactions on Networking, vol. 20, no. 2, pp. 609–619, 2012.
[14] N. Shrivastava, S. Suri, and C. D. Toth, “Detecting cuts in sensor networks,” *ACM Transactions on Sensor Networks*, vol. 4, no. 2, article 10, pp. 1–25, 2008.

[15] H. Ritter, R. Winter, and J. Schiller, “A partition detection system for mobile ad-hoc networks,” in *Proceedings of the 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON ’04)*, pp. 489–497, Santa Clara, Calif, USA, October 2004.

[16] M. Hauspie, D. Simplot, and J. Carle, “Partition detection in mobile ad hoc networks,” in *Proceedings of the 2nd Mediterranean Workshop on Ad-Hoc Networks*, pp. 25–27, 2003.

[17] A. Cerpa and D. Estrin, “ASCENT: adaptive self-configuring sensor networks topologies,” *IEEE Transactions on Mobile Computing*, vol. 3, no. 3, pp. 272–285, 2004.

[18] P. Kumar, J. Kuri, P. Nuggehalli, M. Strasser, M. May, and B. Plattner, “Connectivity-aware routing in sensor networks,” in *Proceedings of the International Conference on Sensor Technologies and Applications (SensorComm ’07)*, pp. 387–392, Valencia, Spain, October 2007.

[19] S. Das, H. Liu, A. Nayak, and I. Stojmenović, “A localized algorithm for bi-connectivity of connected mobile robots,” *Telecommunication Systems*, vol. 40, no. 3-4, pp. 129–140, 2009.

[20] M. Imran, M. Younis, N. Haider, and M. A. Alnuem, “Resource efficient connectivity restoration algorithm for mobile sensor/actor networks,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, article 347, 2012.

[21] M. Imran, A. M. Said, M. Younis, and H. Hasbullah, “Application-centric recovery algorithm for wireless sensor and actor networks,” *International Journal of Communication Networks and Distributed Systems*, vol. 10, no. 4, pp. 379–401, 2013.

[22] N. Haider, M. Imran, and N. M. Saad, “CARE: coverage-aware connectivity restoration algorithm for mobile actor/robot networks,” in *Proceedings of the 19th Asia-Pacific Conference on Communications*, Bali, Indonesia, 2013.

[23] A. C. Viana, S. Maag, and F. Zaidi, “One step forward: linking wireless self-organizing network validation techniques with formal testing approaches,” *ACM Computing Surveys*, vol. 43, no. 2, article 7, pp. 1–36, 2011.

[24] S. Kurkowski, T. Camp, and M. Colagrosso, “MANET simulation studies: the incredibles,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 4, pp. 50–61, 2005.

[25] T. R. Andel and A. Yasinsac, “On the credibility of manet simulations,” *Computer*, vol. 39, no. 7, pp. 48–54, 2006.

[26] F. Ben Abdesslem, L. Iannone, M. D. de Amorim, K. Obaczkza, I. Solis, and S. Fdida, “A prototyping environment for wireless multihop networks,” in *Proceedings of the 3rd Asian Conference on Internet Engineering: Sustainable Internet (AINTEC ’07)*, Phuket, Thailand, November 2007.

[27] M. Imran and N. A. Zafar, “Formal specification and validation of a hybrid connectivity restoration algorithm for wireless sensor and actor networks,” *Sensors*, vol. 12, no. 9, pp. 11754–11781, 2012.

[28] S. Maag, C. Grepet, and A. Cavalli, “A formal validation methodology for MANET routing protocols based on nodes’ self similarity,” *Computer Communications*, vol. 31, no. 4, pp. 827–841, 2008.

[29] P. C. Ölveczky and S. Thorvaldsen, “Formal modeling, performance estimation, and model checking of wireless sensor network algorithms in Real-Time Maude,” *Theoretical Computer Science*, vol. 410, no. 2-3, pp. 254–280, 2009.

[30] O. Dagdeviren and V. K. Akram, “Energy-efficient bridge detection algorithms for wireless sensor networks,” *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 867903, 15 pages, 2013.

[31] M. Imran, M. A. Alnuem, M. S. Fayed, and A. Alamri, “Localized algorithm for segregation of critical/non-critical nodes in mobile ad hoc and sensor networks,” *Procedia Computer Science*, vol. 19, pp. 1167–1172, 2013.