EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

AOLSR: hybrid ad hoc routing protocol based on a modified Dijkstra's algorithm

Dhanalakshmi Natarajan[1*] and Alli P Rajendran[2]

Abstract

Mobile *ad hoc* networks (MANETs) are highly vulnerable to both link and node failures due to nodal mobility. The routing resilience against link and/or node failures needs to be enhanced to avoid the degradation of network performance. This can be achieved by multipath routing which uses multiple alternative paths to route the messages *via* multiple disjoint paths and result in increased bandwidth, fault-tolerance, and security. An optimized link state routing (OLSR) protocol is a proactive routing protocol. An advanced OLSR (AOLSR) protocol is proposed based on a modified Dijkstra's algorithm which enables routing in multiple paths of dense and sparse ne0074work topologies. The routing is based on the energy of nodes and links (implied from the lifetime) and the mobility of the nodes. It is a hybrid *ad hoc* routing protocol because it combines the proactive and reactive features. It is another form of source routing protocol which allows a sender of a data packet to partially or completely reveal the route the packets take through the network. Two cost functions are introduced to build link-disjoint or node-disjoint paths. Secondary functions, namely path recovery and loop discovery process are involved to manage the topology changes of the network. AOLSR protocol is analyzed and compared with the existing MANET routing protocols namely, dynamic source routing (DSR) and OLSR. Its performance is observed to be satisfactory in terms of average end-to-end delay, packet delivery ratio (PDR), average time in first-in-first-out (FIFO) queue, and throughput.

Keywords: Dynamic source routing (DSR); Dijkstra's algorithm; Mobile ad hoc network (MANET); Packet delivery ratio (PDR); Optimized link state routing (OLSR); Quality of service (QoS)

1. Introduction

Network connectivity is an important aspect of mobile technologies. An advantage in mobile *ad hoc* networks (MANETs) is that all the nodes can act as routers to forward the packets without any additional infrastructure. This is efficient due to its self-coordinated, self-maintained, and spontaneous nature. Data routing in such networks is a challenging task owing to the lifetime, scalability, and security issues.

Several routing protocols have been designed for *ad hoc* networks. The link failures and node failures in *ad hoc* networks form a major problem due to the depleted node mobility or node power and might break down the path for routing. The routing resiliency can be enhanced by simultaneous routing of a message through multiple disjoint routes. This will ensure that the destination node will receive the message. Multipath routing protocols are advantageous especially in large and dense *ad hoc* networks. They solve the limitations in bandwidth and energy consumption.

The primary goals of a multipath routing protocol are to balance the network load, decrease the intersection of nodes or connections among the parallel routes, enhance the quality of service (QoS), and ensure that reliable communication is provided, while the secondary goals are to decrease the delay, overhead, and increase the network lifetime. A link failure in one path should not affect other routes. The multiple paths utilized in this type of routing protocol can act as backup routes or additional routes for parallel data transmission.

A hybrid *ad hoc* routing protocol is a combination of proactive and reactive routing protocols. A proactive routing protocol maintains a routing table (next hop information) for all potential destinations and so it is also known as table-driven routing protocol. A reactive routing protocol determines a route only on demand by

* Correspondence: dhanammugi@gmail.com
[1]Department of Information Technology, NPR College of Engineering & Technology, Natham, Dindigul, Tamil Nadu 624003, India
Full list of author information is available at the end of the article

inundating the network with route request (RREQ) packets and so it is also known as an on-demand routing protocol.

An optimized link state routing (OLSR) protocol is a proactive routing protocol. An advanced OLSR (AOLSR) protocol is proposed based on a modified Dijkstra's algorithm which permits routing in multiple paths of dense and sparse network topologies. The routing is based on the energy of nodes and links (implied from the lifetime) and the mobility of the nodes. Energy factors are used to determine the multiple parallel and disjoint routes. AOLSR is a hybrid *ad hoc* routing protocol because it integrates the proactive and reactive characteristics. It is also a source routing protocol which permits the sender of a data packet to partially or completely reveal the route that the packets traverse through a network. This enables the discovery of all possible paths to a host. Two cost functions are introduced to construct link-disjoint or node-disjoint routes. Secondary functions namely, path recovery and loop discovery process are included to manage the topology changes of the network. The network topology varies frequently due to the movement of the mobile nodes and energy constraints.

The remaining part of the paper is organized as follows: section 2 involves a brief description of the existing methods - dynamic source routing (DSR) and OLSR - and the problems involved in them. Section 3 involves the works related to probable solutions for problems in DSR and OLSR in terms of routing overhead and QoS. Section 4 involves the description of the proposed method - advanced OLSR (AOLSR). Section 5 involves the performance evaluation and comparison of AOLSR and existing techniques based on DSR and OLSR. The paper is concluded in section 6.

2. Existing methods

Two existing routing protocols for MANET are considered. One is the DSR which is a reactive routing protocol and another one is the OLSR protocol which is a proactive routing protocol.

2.1 Dynamic source routing protocol

In DSR [1], the mobile nodes maintain the path caches that comprise the pre-known source routes. The elements of the path cache are updated as the new paths are discovered. This protocol consists of two functions namely, path discovery and path maintenance. When a packet is to be transmitted to a destination, the source node first determines whether its path cache already consists of an existing path to the destination. If a path to the destination is available, the packet is routed using that path. Otherwise, the node starts a path discovery process by RREQ broadcast.

The maintenance of path caches extends the validity of the paths. The information in the path caches can also be extracted by the intermediate nodes for effective reduction of control overhead.

2.2 Demerits of DSR protocol

The following are the disadvantages of using the DSR protocol:

- The disconnected links cannot be fixed by the local path maintenance scheme.
- The idle path cache information leads to variations during the path reconstruction phase.
- Higher connection setup delay compared to table-driven routing protocols.
- Degradation of performance with higher mobility of nodes.
- Higher routing overhead.

2.3 Optimized link state routing protocol

The OLSR protocol characterizes low bandwidth and high mobility [2]. It involves a novel periodic flooding of control information using multipoint relays (MPRs), which decrease the number of transmissions in the network.

The OLSR daemons regularly exchange the various messages namely, *HELLO*, multiple interface declaration (*MID*), and topology control (*TC*). These messages maintain the network topology information under the link failure and mobility conditions.

- HELLO messages are interchanged between each neighboring nodes which are at a distance of (1 – hop), where 'hop' is the minimum hop distance between two nodes.
- TC messages are produced periodically by the MPRs to identify the other nodes which have been MPRs.
- MID messages are transmitted by the nodes to inform about the involvement of network interfaces.

The timeouts before transmitting HELLO, MID, and TC messages are *HELLO_PERIOD*, *REFRESH_ PERIOD*, and *TC_ PERIOD*, respectively. The validity period of the information obtained from the three messages are given by the variables *NEIGHB_HOLD_PERIOD*, *MID_HOLD_PERIOD*, and *TC_HOLD_PERIOD*.

2.4 Demerits of OLSR protocol

The following are the disadvantages of using the OLSR protocol:

- No provision for sensing of the link quality,
- High consumption of network and power resources,
- Large amount of bandwidth required to estimate the optimal routes.
- Limited number of control traffic messages [3],
- Possibility of network compromise [3].

3. Related work

This section deals with the probable solutions (various existing multipath routing protocols and variants of Dijkstra's algorithm) considered for solving the problems that were earlier faced by DSR [1] and OLSR [2] during the estimation of the shortest routes in MANETs and other communication fields. The conventional MANET routing protocols determine only a single route from a source to a destination [4]. Multiple disjoint routes between the source and destination are estimated during the path discovery phase. Every node constructs a map of the whole network and uses Dijkstra's algorithm to discover the best routes to each destination. This will decrease the overhead and packet loss rate and enhance the network reliability.

Banimelhem and Khasawneh designed a grid-based multipath routing protocol integrated with congestion avoidance (GMCAR) [5]. This is suited for grid-based sensor networks focusing on energy efficiency. The network was divided into grids, where each grid composes of a master node. The master node is responsible for delivery of data obtained from any node in the corresponding grid. A master node also routes the data from the other master nodes in the surrounding grids. Each master node stores the multiple diagonal routes to its sink in a routing table. A congestion control scheme combined with grid densities and hop count enhances the performance of this protocol. It consumed an average of 22% energy of the total stored energy. The average delivery ratio was around 50% and the average end-to-end packet delay was about 280 ms.

Thulasiraman, et al. proposed a multipath routing scheme in wireless multihop networks [6]. The QoS was enhanced by a fair max-min bandwidth allocation algorithm based on different routing metrics. An optimization formulation was designed to solve the multi-commodity flow problem in the bandwidth allocation algorithm. Dijkstra's algorithm with some prominent edge lengths was also considered for reducing the communication overhead (determination of single source shortest route) in multipath routing algorithms [7]. A MANET was modeled as a graph (abstract data type) comprising of edges with positive length and the prominent edge lengths. Considering a graph with x vertices, y edges, and Z prominent edge lengths, this Dijkstra's algorithm possesses the following communication complexity:

$$\text{Commun.Comp.} = \begin{cases} O(y), xZ \leq 2y \\ O\left(y \log \dfrac{xZ}{y}\right), \text{otherwise} \end{cases} \quad (1)$$

Yang, et al. proposed a disjoint, integrated, and reliable multipath routing scheme for sensor networks [8]. The multiple paths were constructed using a hop-by-hop technique. This scheme maintains only the local route information on each node before estimating the end-to-end paths. The neighbors were clustered into groups according to their hop count. This enhanced the network traffic balance. The local nodes chose their own backup nodes to construct additional logical routes using an integrated multipath model. This method effectively guaranteed the load balance of the network and decreased the number of transmission routes and nodal energy consumption. Sermpezis, et al. investigated a junction-based multipath source routing algorithm for vehicular *ad hoc* networks (VANETs) [9]. The adoption of the junction-focused logic and source routing schemes resulted in an average packet delivery ratio (PDR) of 83% and an average delay of 0.425 s. This technique chooses the paths according to Dijkstra's algorithm. Dijkstra's algorithm is used to perform a bidirectional search on time-dependent road networks [10] and plan the motion of unmanned aerial vehicles (UAVs) based on terrain elevation [11]. Dijkstra's algorithm was also recently used for solving mathematical problems like L-concave function maximization [12].

Zuo, et al. proposed a hybrid multipath routing protocol for industrial wireless mesh networks [13]. Usually multipath routing methods are designed only for enhancement of reliability and not guaranteed transmission. This hybrid multipath technique enhances both the reliability and trust of data transmission. This method uses an enhanced Dijkstra's algorithm for the determination of the shortest path from the gateway to each end node. The multiple routes are estimated using the ant colony optimization algorithm and the link failures are managed using the path maintenance scheme. Some other multipath routing protocols include concepts like independent directed acyclic graphs [14], forward error correction [15], border gateway protocol (BGP) [16], inter-domain routing [16], multiple-exit discriminator (MED) [16], hiding routes [17], and adaptive multi-metric *ad hoc* on-demand multipath distance vector (AM-AOMDV) routing protocol [18].

4. Hybrid *ad hoc* routing protocol

The primary functions of AOLSR protocol are *topology detection* and *path estimation*. The network topology is sensed to inform the nodes the topology information. The path estimation utilizes the modified Dijkstra's algorithm to compute the various paths based on the information from topology detection. A link failure in one path should not affect other routes. The source path (route from source to destination including all the hops) is always preserved in the header of the data packets. The data flow diagram of AOLSR protocol is shown in Figure 1.

Topology detection and path estimation are responsible for the determination of the multiple paths from the source to the destination. The instability of the wireless medium and the variations in the network topology necessitates the auxiliary functions of the OLSR protocol such as, *path recovery* and *loop discovery*. Path recovery is used for the

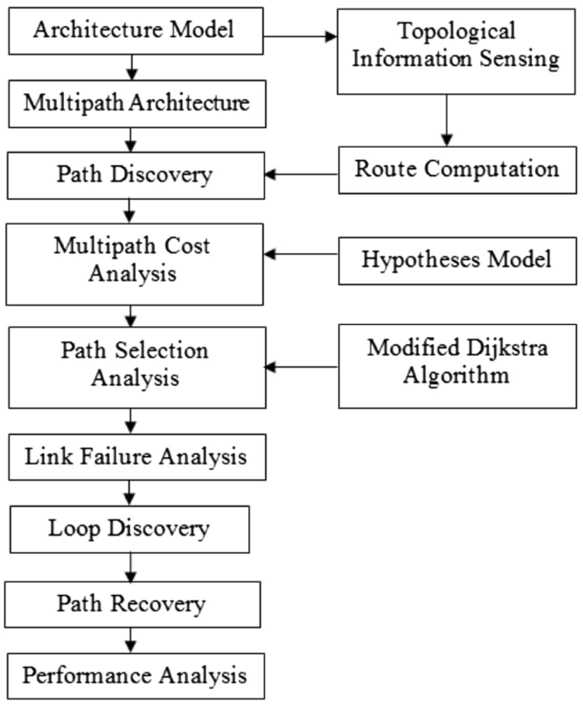

Figure 1 Data flow diagram of AOLSR protocol.

effective decrease of the packet loss and loop discovery is used to detect and avoid loops in the routing paths.

4.1 Topology detection

The information in the network topology is sensed by topology detection. The procedure in AOLSR consists of an additional process known as energy detection to that of OLSR, with processes namely, link detection, neighbor detection, and topology discovery. Link detection constructs the *link set* consisting of local link information. It concentrates on the packet communication between the OLSR interfaces and their addresses. Neighbor detection constructs a *single-hop neighbor set* and a *double-hop neighbor set* consisting of network information and node addresses. Topology discovery constructs a *topology set* containing the information about the nodes that are greater than double hops. The topology set construction depends on the TC message flooding.

Each sensor node in the WSN obtains sufficient topological information to enable routing. The AOLSR protocol estimates the route quality and energy usage according to the number of hops and maintains the link information. The hop count and energy consumption are used as the link metrics in the AOLSR protocol.

4.2 Path estimation

The determination of paths is based on an on-demand methodology to obviate the density estimation of multiple paths for every feasible destination. The lifetime of the nodes and links determine the path selection. The routes with longer lifetime (energy) are chosen to prohibit the failure of the entire route in case of energy exhaustion of certain nodes or links. A multipath estimation hypotheses model is introduced as a prerequisite for the modified Dijkstra's algorithm.

4.2.1 Hypotheses model

A multipath routing protocol constructs a group g_n of n paths without any loops. These paths connect a source node S to a destination node D.

An *ad hoc* network is defined by a directed energy graph (an abstract data type) $G = (g_v, g_a, f_c)$, where g_v is the group of vertices, $g_a \subset g_v \times g_v$ the group of arcs, and f_c: $g_v \rightarrow \mathbb{R}^{*+}$ a rigidly real-positive cost function. The graph is initialized to be undirected, i.e., $(v_1, v_2) \in g_a \Rightarrow (v_2, v_1) \in g_a$ and $f_c(v_1, v_2) = f_c(v_2, v_1)$ and loop-free, i.e., No arcs from a node connect to itself. It is also assumed that a pair of vertices cannot be linked by more than one arc. A path between S and D is defined as a sequential order of vertices $(v_1, v_2, \ldots, v_D)$ so that consecutive vertices are elements of g_a, where $v_1 = S$ and $v_D = D$. These hypotheses define the cost function f_c in an *ad hoc* manner.

4.2.2 Modified Dijkstra's algorithm

The AOLSR maintains a *status flag* for every sensor node in the WSN to learn the validity of the routes to the relative sensor node. The status flag of every node x (*status_flag*$_x$) is initially set to 0, which implies the path to the corresponding destination needs to be refreshed or does not prevail. The source node will initially check *status_flag*$_x$ when a RREQ to node x is placed.

- When *status_flag*$_x$ is equal to 0, the node executes Algorithm I to obtain the multiple routes to node x. These routes are stored in the multipath routing table and the relative *status_flag*$_x$ is updated to 1.
- When *status_flag*$_x$ is equal to 1, the node determines a valid path to node x in the multipath routing table.

Whenever the node receives a new HELLO or TC message, variations occur in the topology set, and all the status flags will be set to 0. Algorithm I briefs the steps for the determination of n routes from S to D.

This modified Dijkstra's algorithm is applied to a graph $G = (g_v, g_a, f_c)$, two vertices $(S, D) \in g_a^2$, and a rigidly real-positive integer n. It results in n paths $(P_1, P_2, \ldots, P_n)$ from S to D obtained from G. The following predefined functions or conventions are used:

- *Dijkstra*(G, S) is the conventional Dijkstra algorithm [19] which yields the source tree (T_s) of the shortest route from vertex S in a graph G, where a tree is a type of data structure.
- *Get_Path*(T_s, D) is the function to obtain the shortest path to D from T_s.

- a^{-1} means the opposite edge of a.
- *Vertex_Head*(a) gives the head (forward vertex edge) of a.

```
ALGORITHM I – Modified Dijkstra's Algorithm to compute $n$ paths in $G$ from $S$ to $D$
Input:  Source node $S$, Destination node $D$, Graph (Ad hoc network) $G$, Number of paths $n$, Cost
        function $f_c$, Group of vertices $g_v$, and Group of arcs $g_a$.
Output: Multiple paths $P_1, P_2 \dots P_n$
begin
    $f_{c\_1} = f_c$            // Initial $f_c$
    $G_1 = G$                  // Initial $G$
    for $x = 1$ to $n$ do
        $T_{s\_x} = Dijkstra(G_x, S)$
        $P_x = Get\_Path(T_{s\_x}, D)$
        for each arc $a$ in $g_a$
            $t_{l\_a} = (1 - W_l) . t_{l\_a}$
        end for
        for each vertex $v$ in $g_v$
            $t_{l\_v} = (1 - W_n) . t_{l\_v}$
        end for
        for each arc $a$ in $g_a$
            if $a$ or $a^{-1}$ is in $P_x$
                $f_{c\_x+1} = f_r(f_{c\_x}(a))$
            else if $Vertex\_Head(a)$ is in $P_x$
                $f_{c\_x+1} = f_a(f_{c\_x}(a))$
            else
                $f_{c\_x+1} = f_{c\_x}$
            end if-else
        end for
        $G_{x+1} = (g_v, g_a, f_{c\_x+1})$
    end for
    return $(P_1, P_2 \dots P_n)$
end
```

Initially, the links are selected according to the maximum lifetime as per the following three conditions:

- The link with the maximum lifetime is chosen from the former links which initiate from a specific node x.
- When there are many links with equal lifetime, the lifetimes of the neighbors in each link are compared, and the link whose neighbor possesses the maximum lifetime is chosen.
- When there are various links with equal lifetime and whose neighbors also possess equal lifetime, a link is chosen at random.

The links are selected are selected every time and further path selection is performed by using the two incremental functions. The energy factors are given by two iteration factor weights: weight for links (W_l) and weight for nodes (W_n). The range of these weights is [0, 1]. The iteration factor weights are given by the following formulae:

$$W_l = \left(t_{l\_\max} - t_{l\_\min}\right)/t_{l\_\text{avg}} \tag{2}$$

$$W_n = \left(t_{n\_x} - t_{n\_\text{threshold}}\right)/t_{n\_\text{threshold}} \tag{3}$$

In (2), $t_{l\_\max}$, $t_{l\_\min}$, and t_{l_avg} respectively denote the maximum, minimum, and average lifetime of the links in the entire route. In (3), t_{n_x} denotes the lifetime of node x and $t_{n\_\text{threshold}}$ denotes the threshold of the lifetime for any node in the route. The computation of the energy factors can be altered according to the various inclines for the disconnectivity of the links or disconnectivity of the nodes during path selection.

Two incremental functions $f_a, f_r : \mathbb{R}^{*+} \to \mathbb{R}^{*+}$ are introduced at each round to obtain a disjoint route between S and D. f_a is used to increase the arc costs that converge to the vertices of the previous path P_x. f_r is used to increase the arc costs that belong to the previous path P_x (or the opposite arcs belonging to it). This will enable the further routes to utilize various arcs. The three possible conditions are as follows:

- When $f_i = f_a < f_r$, the routes become arc-disjoint.
- When $f_i < f_a = f_r$, the routes become vertex-disjoint.
- When $f_i < f_a < f_r$, the routes try to be vertex-disjoint, but if not possible, they become arc-disjoint.

In the above conditions, f_i denotes the identity function. The cost functions determine the variety in the n paths of the network topology. There is no necessity that the multiple paths estimated by this algorithm require being completely disjointed. This is because the number of disjoint routes is bounded to the (S, D) minimal cut. The minimal cut of (S, D) is the dimension of the smallest subset of edges necessary to link S to D. The minimal cut is estimated by the neighborhoods of source and destination. A demerit of completely disjoint algorithm is the generation of longer routes because each local cutoff can be applied only once.

An illustration of the modified Dijkstra's algorithm is shown in Figure 2. The number of hops is used as the connection cost metric, $f_a(c) = 2c$, and $f_r(c) = 3c$. More penalties are assigned to the traversed links. The cost of the links is set to unity initially.

First, the shortest route $S \to E \to F \to G \to D$ is determined. Then, the cost functions are applied to increase the cost of the corresponding arcs:

- $S \to E$, $E \to F$, $F \to G$, $G \to D$ will be modified from 1 to 3 as per f_r.
- $S \to A$ and $C \to G$ will be altered from 1 to 2 as per f_a.

Next, the second shortest route $S \to A \to B \to C \to G \to D$ is determined. Different cost functions can determine numerous multipath sets (link-disjoint or node-disjoint) according to the network preferences. Another network topology example is shown in Figure 3.

- When $f_a(c) = c$, and $f_r(c) = 3c$ are chosen, the penalty is assigned only to the traversed links, and the two link-disjoint routes obtained are $S \to E \to A \to F \to D$ and $S \to B \to A \to H \to D$.
- When $f_a(c) = 2c$, and $f_r(c) = 3c$ are chosen, the penalty is assigned only to the traversed nodes, and the two node-disjoint routes obtained are $S \to E \to A \to F \to D$ and $S \to B \to C \to G \to H \to D$.

4.3 Path recovery

Topology detection determines the network topology details with the communication of HELLO and TC messages. This information is stored in the link set, neighbor set, or topology set of the local node. Practically, the topology information base is not competent compared to the real network topology owing to the mobile nature of the ad hoc network.

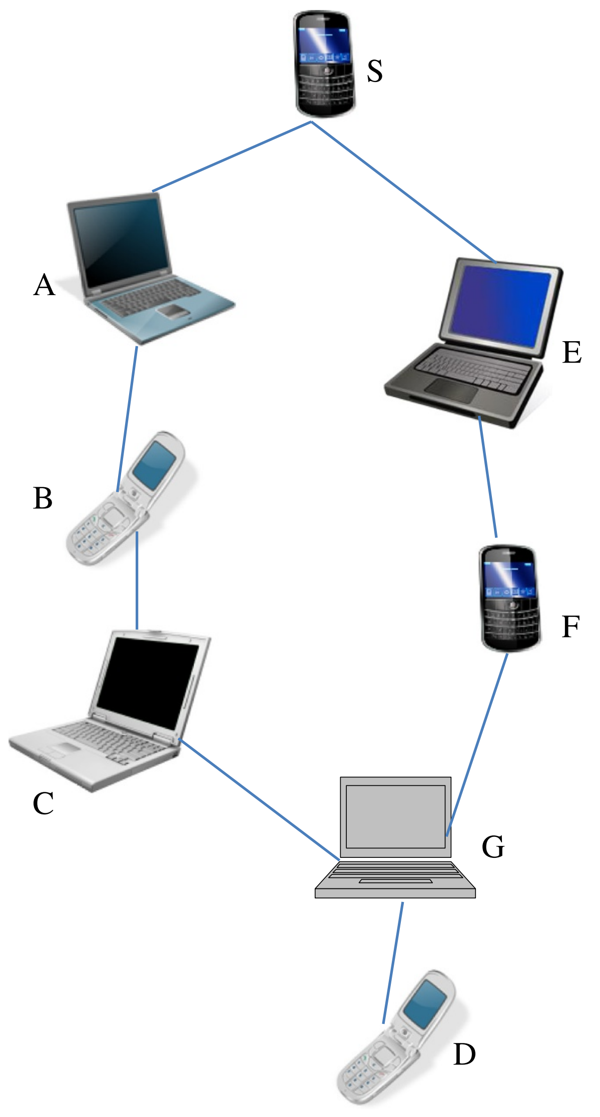

Figure 2 Illustration of modified Dijkstra's algorithm.

During the message generation time interval of the HELLO and TC messages, the topology may vary due to nodal movement. The control message can get expired or even get lost due to the possible collision or delay in the control messages. These conditions are the reasons for the variability between the actual network topology and the network topology information base.

Path recovery is used to fulfill the gap between the actual network topology and the network topology information base. First, a node checks whether the next hop in the source path is one of its neighbors. If so, the

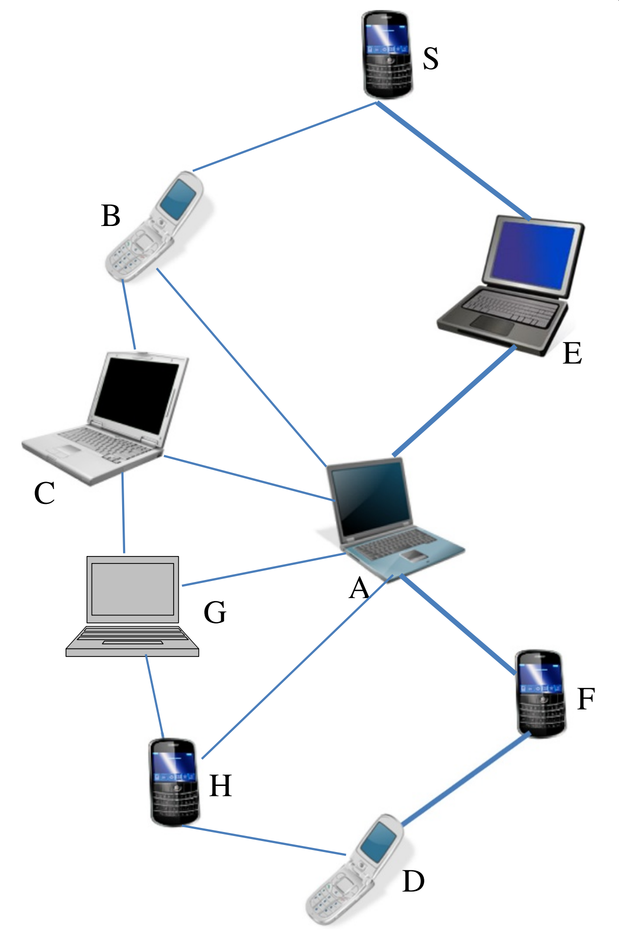

Figure 3 Various route sets using different cost functions.

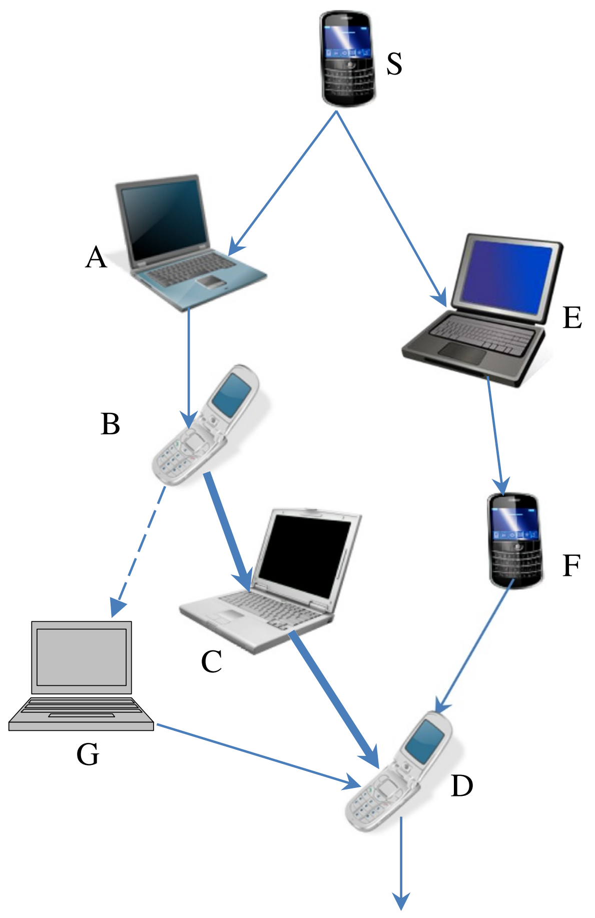

Figure 4 Example of path recovery.

packet is transmitted normally; otherwise, it is decided that the 'next hop' node is not available. Next, the node will recalculate the path and transmit the packet in the new route.

An example of path recovery is given in Figure 4. The packets are routed from source *S* to destination *D*. The actual paths are $S \to E \to F \to D$ and $S \to A \to B \to G \to D$. But, the node *G* displaces out of the transmission range of node *B* and loses the second path. Now, the link failure cannot be detected immediately by the source node because of the delay in TC messages. So, the source node keeps on transmitting the packets along the same path which are dropped consequently. When path recovery is included, the node *B* will check upon the incoming packet whether node *G* is yet one of its neighbors before transmitting the packet according to the primary route. When node *G* is not one of the neighbors of node *B*, the former node will recalculate the path to node *D*, and find $B \to C \to D$. Then the incoming packets are transmitted through the new route.

This technique does not introduce much additional delay because path recovery checks only the topology information stored in the local node. It will also greatly enhance the PDR of the network. The delivery ratio of the AOLSR protocol with path recovery is about 50% higher than that without path recovery.

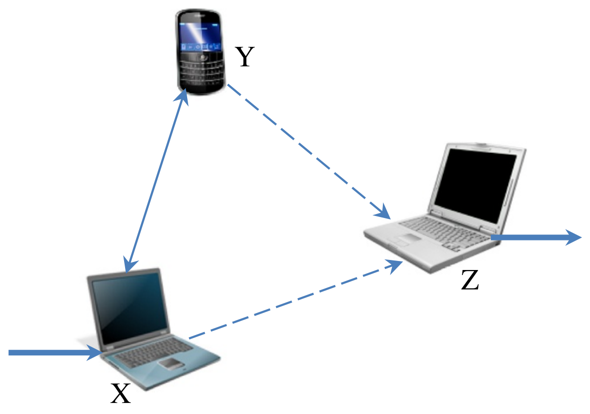

Figure 5 Example of a loop in the network.

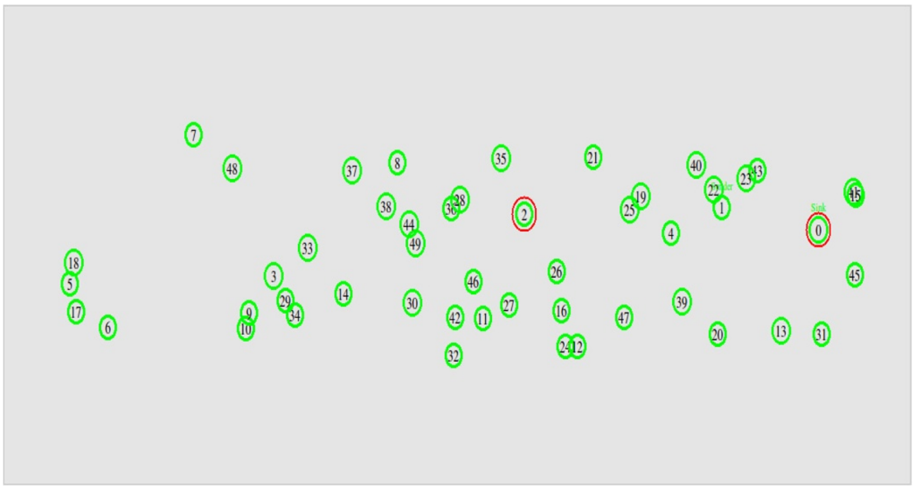

Figure 6 Network topology with nodal distribution.

4.4 Loop discovery

Loops in the network are an essential issue in routing because they result in data redundancies and communication latency. Link layer notification (LLN) must be mandatorily discussed for the enhancement of the packet delivery ratio, before handling the problem of loops in the network [20]. The link layer information briefs about the linkage to the available neighboring nodes. This information is additionally used to that of the HELLO message for the maintenance of the neighbor set and MPR selector group. The AOLSR protocol acts on the Acknowledgement (ACK) message from LLN and deletes the related connections from its information base.

Theoretically, the routes produced by Dijkstra's algorithm and the AOLSR protocol are devoid of loops. But, practically due to LLN and path recovery, loops in the network are possible. A node attempts to transmit a packet over a link but does not succeed in the end and so the link layer will give a feedback to the AOLSR protocol to apprise about the link loss. This sort of sudden interruption will require additional executions on the topology information base besides the normal HELLO and TC messages. This implies that other nodes do not know about these modifications immediately. Consequentially, the LLN might result in variable topology information in the various nodes. The inclusion of path recovery enables the modification of the route in intermediate nodes and the loop in the network is temporarily constrained.

An example of loop production is shown in Figure 5. Here, node X is taken as an intermediate node of some former route. The packets with the primary path $X \rightarrow Z$ arrive at node X and require to be transmitted to node Z. Next, node Z displaces out of the transmission coverage of nodes X and Y, which expires the links $X \rightarrow Z$ and $Y \rightarrow Z$.

The transmission of the incoming packets at node X to node Z will be failed, as a result of which the AOLSR protocol in node X will be accepted by LLN, and the path $X \rightarrow Z$ is removed from the link set of node X. For

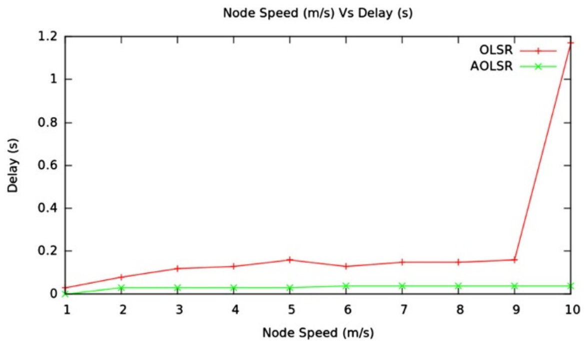

Figure 7 Comparison of average delay for OLSR and AOLSR protocols.

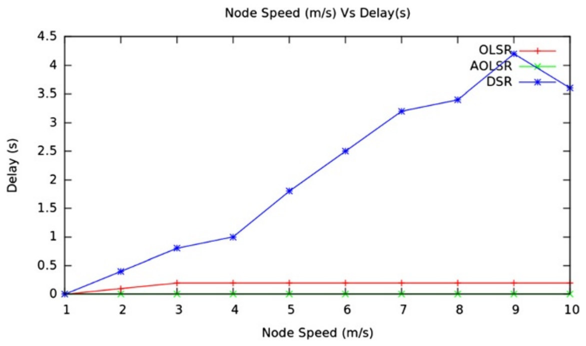

Figure 8 Comparison of average delay for DSR, OLSR, and AOLSR protocols.

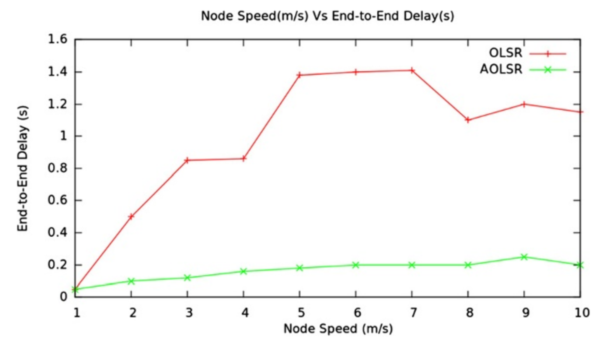

Figure 9 Comparison of average end-to-end delay for OLSR and AOLSR protocols.

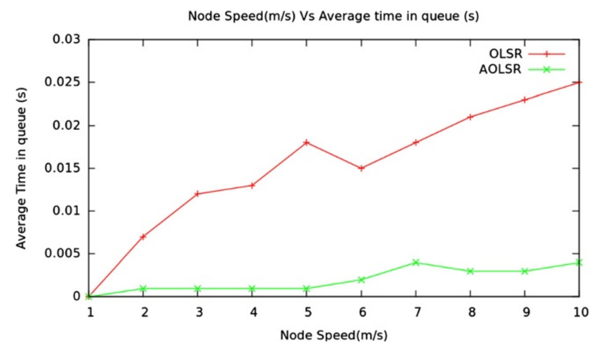

Figure 11 Comparison of average time in a FIFO queue for OLSR and AOLSR protocols.

node X, although it identifies the link failure of $X \rightarrow Z$ by LLN, it is tough to detect the failure of $Y \rightarrow Z$ instantly. This is due to the late removal of link $Y \rightarrow Z$ because of the high expiry time of *NEIGHB_HOLD_PERIOD*. Simultaneously, the path recovery process will be triggered and a fresh path $X \rightarrow Y \rightarrow Z$ will be determined. The forthcoming packets will be transmitted through this new path and will be rerouted to node Y. These operations are also repeated in node Y. Since node Y cannot identify the link failure of $X \rightarrow Z$ immediately, the recent route discovered by path recovery is $Y \rightarrow X \rightarrow Z$. This creates a loop as the packet will traverse $Y \rightarrow X \rightarrow Y$. This is only a transient loop existing for several seconds and will diminish when the corresponding link expires. But this sort of temporary loops will hinder the links in the loop and choke the respective transmission area.

To overcome the previous disadvantage, a loop discovery technique based on source routing is applied which does not accumulate much memory overhead. After the process of path recovery, a new route will be estimated from the present node to the destination. This method will utilize the new route when there are no loops in the network; otherwise, it will determine another route according to the modified Dijkstra's algorithm. Suppose no suitable route exists, the packet will be discarded.

Node X will obtain a route $X \rightarrow Y \rightarrow Z$ by path recovery. Now, when the packet arrives at node Y, a new route $Y \rightarrow X \rightarrow Z$ will be produced because of the connection failure of $Y \rightarrow Z$. Node Y will examine the new route with the former primary route $X \rightarrow Y \rightarrow Z$ in the packet. The reduction of loops in the network will efficiently decrease the network congestion and end-to-end delay.

5. Performance analysis

The network topology with the distribution of nodes is given in Figure 6. The AOLSR protocol is analyzed and compared with the existing MANET routing protocols namely, dynamic source routing (DSR) [1] and OLSR [2]. Its performance is observed to be satisfactory in terms of average end-to-end delay, packet delivery ratio (PDR), average time in first-in-first-out (FIFO) queue, and throughput.

5.1 Average delay

The average delay for the OLSR and AOLSR protocols is analyzed and compared in Figure 7, while the average delay for all the three protocols DSR, OLSR, and AOLSR are analyzed and compared in Figure 8.

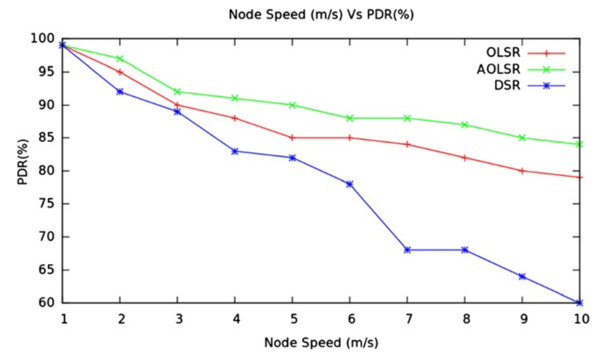

Figure 10 Comparison of PDR for DSR, OLSR, and AOLSR protocols.

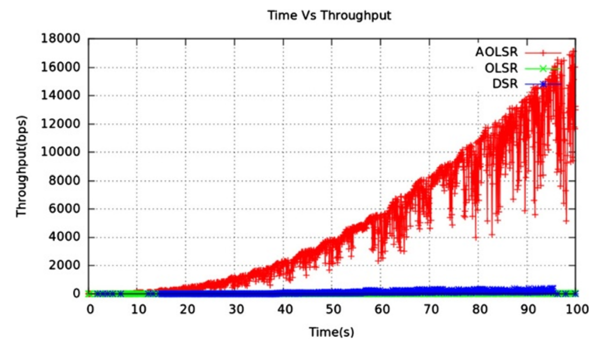

Figure 12 Comparison of throughput for DSR, OLSR, and AOLSR protocols.

5.2 Average end-to-end delay
The average end-to-end delay for OLSR and AOLSR protocols is analyzed and compared in Figure 9.

5.3 Packet delivery ratio
The PDR for all the three protocols DSR, OLSR, and AOLSR are analyzed and compared in Figure 10.

5.4 Average time in FIFO queue
The average time in a FIFO queue for all the three protocols DSR, OLSR, and AOLSR are analyzed and compared in Figure 11.

5.5 Throughput
The throughput for all the three protocols DSR, OLSR, and AOLSR are analyzed and compared in Figure 12.

6. Conclusion
In this paper, a hybrid *ad hoc* routing protocol, i.e., an advanced OLSR (AOLSR) protocol, is proposed based on a modified version of Dijkstra's algorithm. The major additions to the conventional Dijkstra's algorithm are the two cost functions to create the multiple disjoint or connected routes, secondary functions, i.e., path recovery and loop discovery to ensure the QoS. The routing is based on the energy of nodes and links (implied from the lifetime) and the mobility of the nodes. This routing protocol effectively enhances the network performance in the case of heavy network traffic and high mobility.

The main aspects in a MANET such as confidentiality, network lifetime, scalability, and reliability are satisfied by the AOLSR protocol. The network lifetime is enhanced by decreasing the number of transmitted packets per node. The future work involves the enhancement of this model in terms of security *via* a partial network topology to detect attacks like spoofing attack, invalid MPR attack, disruption attack, and hop limit attack.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Department of Information Technology, NPR College of Engineering & Technology, Natham, Dindigul, Tamil Nadu 624003, India. [2]Department of Computer Science and Engineering, Velammal College of Engineering & Technology, Madurai, Tamil Nadu 625009, India.

Received: 9 December 2013 Accepted: 8 May 2014
Published: 5 June 2014

References
1. N Qadri, A Liotta, Analysis of Pervasive Mobile Ad Hoc Routing Protocols, in *Pervasive Computing*, ed. by AE Hassanien, JH Abawajy, A Abraham, H Hagras (Springer, London, 2010), pp. 433–453
2. J Toutouh, J Garcia-Nieto, E Alba, Intelligent OLSR routing protocol optimization for VANETs. IEEE Trans on Vehicular Technol **61**(4), 1884–1894 (2012)
3. G Cervera, B Michel, GA Joaquin, K Evangelos, Security issues in link state routing protocols for MANETs, in *Advances in Network Analysis and its Applications*, ed. by E Kranakis. vol. 18 (Springer, Berlin Heidelberg, 2013), pp. 117–148
4. LML Oliveira, F de Amaro, JPC Joel, Routing and mobility approaches in IPv6 over LoWPAN mesh networks. Int J Commun Syst **24**(11), 1445–1466 (2011)
5. O Banimelhem, S Khasawneh, GMCAR: Grid-based multipath with congestion avoidance routing protocol in wireless sensor networks. Ad Hoc Netw **10**(7), 1346–1361 (2012)
6. P Thulasiraman, J Chen, X Shen, Multipath Routing and Max-Min Fair QoS Provisioning under Interference Constraints in Wireless Multihop Networks. IEEE Trans on Parallel and Distributed Systems **22**(5), 716–728 (2011)
7. JB Orlin, K Madduri, K Subramani, M Williamson, A faster algorithm for the single source shortest path problem with few distinct positive lengths. J Discrete Algorithms **8**(2), 189–198 (2010)
8. Y Yang, C Zhong, Y Sun, J Yang, Network coding based reliable disjoint and braided multipath routing for sensor networks. J Netw Comput Appl **33**(4), 422–432 (2010)
9. P Sermpezis, G Koltsidas, F-N Pavlidou, Investigating a junction-based multipath source routing algorithm for VANETs. Commun Lett, IEEE **17**(3), 600–603 (2013)
10. G Nannicini, D Delling, D Schultes, L Liberti, Bidirectional A search on time-dependent road networks. Netw **59**(2), 240–251 (2012)
11. F Medeiros, J Silva, A Dijkstra algorithm for fixed-wing UAV motion planning based on terrain elevation, in *Advances in Artificial Intelligence – SBIA 2010. Vol. 6404*, ed. by A Rocha Costa, R Vicari, F Tonidandel (Springer, Berlin Heidelberg, 2011), pp. 213–222
12. K Murota, A Shioura, Dijkstra's algorithm and L-concave function maximization, in *Mathematical Programming vol. 145*, ed. by (Springer, Berlin Heidelberg, 2013), pp. 1–15
13. Y Zuo, Z Ling, Y Yuan, A hybrid multi-path routing algorithm for industrial wireless mesh networks. EURASIP J Wirel Commun Netw **2013**(1), 1–12 (2013)
14. S Cho, T Elhourani, S Ramasubramanian, Independent directed acyclic graphs for resilient multipath routing. IEEE/ACM Trans Netw **20**(1), 153–162 (2012)
15. J Ben-Othman, B Yahya, Energy efficient and QoS based routing protocol for wireless sensor networks. J Parallel Distributed Comput **70**(8), 849–857 (2010)
16. S Secci, J Rougier, A Pattavina, F Patrone, G Maier, Peering equilibrium multipath routing: a game theory framework for internet peering settlements. IEEE/ACM Trans on Netw **19**(2), 419–432 (2011)
17. I Ganichev, B Dai, PB Godfrey, S Shenker, YAMR: yet another multipath routing protocol. SIGCOMM Comput Commun Rev **40**(5), 13–19 (2010)
18. S Khimsara, KR Kambhatla, J Hwang, S Kumar, JD Matyjas, AM-AOMDV: adaptive multi-metric ad-hoc on-demand multipath distance vector routing, in *Ad Hoc Networks. vol. 28*, ed. by J Zheng, S Mao, S Midkiff, H Zhu (Springer, Berlin Heidelberg, 2010), pp. 884–895
19. Y Gao, Shortest path problem with uncertain arc lengths. Comput Math Appl **62**(6), 2591–2600 (2011)
20. P Bucciol, FY Li, N Fragoulis, J Carlos, Hierarchical and QoS-aware routing in multihop wireless mesh networks, in *Guide to Wireless Mesh Networks*, ed. by S Misra, S Misra, I Woungang (Springer, London, 2009), pp. 49–75

doi:10.1186/1687-1499-2014-90
Cite this article as: Natarajan and Rajendran: **AOLSR: hybrid ad hoc routing protocol based on a modified Dijkstra's algorithm.** *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:90.