Link State Routing Protocol with Cluster Based Flooding for Mobile Ad-hoc Computer Networks

Boris Mitelman, Arkady Zaslavsky,
School of Computer Science and Software Engineering
Monash University, Melbourne, Australia.
{boris.mitelman, arkady.zaslavsky}@infotech.monash.edu.au

Abstract

Ability of a routing algorithm to converge quickly when network topology changes frequently is a critical requirement for routing in multi-hop wireless networks. Due to its exceptional convergence performance, Link State routing technology is the state of the art in wired network routing. There is, however, a concern that Link State routing will generate too much overhead traffic in bandwidth-depleted wireless networks. This paper presents a protocol for Link State routing in a mobile wireless ad-hoc network. The protocol is designed to operate in small to medium size ad-hoc wireless networks, with each node being a router at the same time. While most routing algorithms for ad-hoc networks use minimum hop count as their routing metric, our protocol uses link costs that are proportional to the power that the node needs to generate to reach the next hop. Although this approach produces routes with more hops, it allows to minimize the congestion on the link layer (OSI reference architecture), preserves battery power, and creates routes that are less likely to be broken due to node mobility. In order to ensure efficient and reliable flooding of Link State Updates, the cluster based algorithm is used. Once global up-to-date information is delivered to every router, Shortest Path First algorithm ensures that the optimal route is selected.

1. Introduction

Wide availability of portable computers and the demand for flexible mobile networking options stimulated renewed interest in mobile packet radio networks. Most technologies available today for mobile computing are based on existing cellular communication architectures, eg., Cellular Digital Packet Data (CDPD) for connection to packet-switched networks [3], and Global System for Mobiles (GSM) [17] for connections to circuit switched networks. For the Internet, Mobile IP [20] provides a one hop wireless connection of mobile computers to the fixed network.

Ad-hoc wireless networks deal with environments where no fixed network infrastructure exists. Participating computers have to organise themselves into networked groups. Routing becomes an issue in multi-hop wireless networks, where not all of the communicating computers can directly communicate with each other. For the Internet, Mobile Ad-hoc Networking (MANET) [4] project has been initiated to meet the demand for such networks.

"Ad-hoc wireless networks" is a new term for packet radio networks, which have been around for some time [15, 13]. First multi-hop packet radio networks used a special routing protocol, called tier routing [28, 11], which was a Distance Vector protocol adapted specifically for a broadcast environment. Tier routing was designed for small-to-medium size networks operating over a shared channel. Later developments for Packet Radio Networks took advantage of Spread Spectrum radio communication technology [22]. Protocols for the United States' Defence Advance Research Project Agency (DARPA) Survivable Adaptive Networks (SURAN) program [13, 25] utilized receiver-specific codes for Direct Sequence Spread Spectrum. An efficient radio channel access scheme, which is based on Spread Spectrum technology, is called Code Division Multiple Access (CDMA) [14, 7]. It allows transmitting a packet to a receiver in such a way that it appears as white noise to stations listening to orthogonal codes. In addition, the hierarchical routing models for large packet radio networks were developed in SURAN protocols. Clustering techniques were used in SURAN to limit the areas of propagating routing information. Hierarchical routing protocols in SURAN utilized tier protocol for intra-cluster routing, and Link State for inter-cluster routing [26].

Link State Routing, first published in [16] is the state of the art routing technology for wired networks. The Internet Link State routing protocol, Open Shortest Path First (OSPF) [24] is the most advanced Interior Gateway Protocol for the Inter-

Proceedings of the Workshop on Computer Science and Information Technologies CSIT'99
Moscow, Russia, 1999

net. It requires more calculations in the routers and creates slightly higher routing traffic than conventional Distance Vector Routing protocols like Routing Information Protocol (RIP) [23]. The compensation for extra overheads comes in the form of superior network convergence, which results in loop-free routing. Due to extra complexity involved in building networks that use Link State routing, Distance Vector protocol implementations usually come first. There have been several novel algorithms proposed for routing in mobile ad-hoc networks [10, 18, 27]. Motivation behind creating the new routing schemes is to provide for loop-free routing. Link-state is usually discarded as being known for high routing traffic overhead. In order to ensure loop free routing, all mentioned protocols use network flooding in one form or another. If used over a broadcast channel, network flooding can severely stress media access control mechanisms. Unfortunately, congestion problems on the data link layer cannot be seen in the network layer simulation models, which are used to verify modern algorithms.

In this paper, we will adapt Link State routing technology to wireless ad-hoc networks where the data link layer enables power control. The remainder of this paper is organised as follows. Section 2 discusses graph algorithms behind Distance Vector and Link State routing. Section 3 reviews common routing techniques for mobile ad-hoc networks. Section 4 presents a rationale for selected routing metric, and describes a protocol for flooding Link State updates in packet radio networks. Section 5 contains conclusions, while section 6 discusses directions for further research.

2. Graph Algorithms

A network can be represented as a graph G(V,E), where V is the set of vertices, which represent network nodes, and E is the set of edges that represent communication links between the nodes. This section describes basic algorithms that were developed for routing in wired communication networks.

2.1 Distance Vector (distributed algorithm)

In Distance Vector algorithm, nodes of the network take part in the distributed routing computation. The following fields are identified for node's routing table [23]:

- destination address
- next hop address
- interface for next hop (in wired networks routers have multiple interfaces)
- metric (distance to the destination)
- timer (the age of the entry)

The Distance Vector algorithm is also referred to as the Bellman-Ford algorithm, or Ford-Fulkerson algorithm. It can be described in terms of the Bellman's equation [2]. Let d(i,j) be the cost of a hop (distance) from the node *i to node j*. If direct link between *i and j* exists, the distance is finite. Otherwise, it is infinity. *Let D(i,j)* be an additive cost of all hop costs (distances) on the route from *i to j*. Nodes which participate in Distance Vector algorithm calculate the costs D as follows:

- $D(i,i) = 0$
- $D(i,j) = \min[d(i,k)+D(j,k)]$ for all k.

Since only for neighbours the d(i,k) is finite, the distance to the destination is always the one with best metrics via node's neighbours. The algorithm description is thus complete: the node requests routing tables from all its neighbours, and then calculates its own using the formula above.

In all known implementations of the Distance Vector algorithm, all routers in the network advertise their routing tables periodically. Instead of requesting tables from their neighbours, nodes simply listen for the advertisements, and then compare each destination metric in the advertisement with the metric already stored in the table for this destination. If the path to the destination is through this neighbour, the new metric is written to the table. If the best known metric so far was from another neighbour, new metric is written only if it is better than the existing one. The major shortcoming of the basic Distance Vector Algorithm is the possibility of developing long lived routing loops. The trivial case can be illustrated in figure 1.

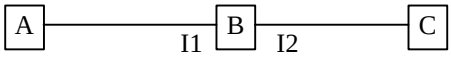

Figure 1. Illustrating Distance vector Algorithm

Suppose that the node B knows that its distance to A is one hop via the interface I1. Node C knows that its distance to node A is two hops via node B. Now suppose that the link between A and B breaks. Now the node B knows that A is unreachable through interface I1 (timer expires), and sets distance to A to infinity. Then it gets a routing update from node C, which would contain distance 2 to A (via B, but there is no information in the routing update that the original route was via B). B then updates its table with a record showing a 3-hop route to A via C. B will then send the update to C, and C updates its route to A to four hops via B. The process continues until the "infinity" of 16 hops is reached. The problem can become much more complicated if there is more than one node involved in counting to infinity.

The following techniques are used to speed up convergence in routing information protocols (RIP):

- **Split Horizon**: a technique where routes are not advertised on the interface from which they were received. The problem described above is then solved.
- **Split Horizon with Poisoned Reverse**: routes are advertised on the interface they were received, but with metric of infinity. This breaks routing loop immediately, rather than waiting for a timeout. The disadvantage is increased size of routing update. (Classic routing overhead versus convergence speed trade-off).
- **Triggered updates**: routers are required to send an update immediately after a distance to any router is changed.

Essentially, the network is flooded every time the topology changes.

A distinguishing feature of Distance Vector as a distributed algorithm is that participating nodes do not maintain full information about the network. Instead, just the distance to the destination, and the direction of the next hop is stored (hence, the name Distance Vector).

2.2 Link State (decentralised algorithm).

In the Link State algorithm, routers in the network maintain the full network map. Each node runs its own copy of the Shortest Path algorithm to construct its routing table. In order to distribute the information about Link States, routers broadcast Link State updates (LSU). To ensure that the LSUs are delivered to all the routers in the network, a flooding algorithm is used. The flooding algorithm works as follows:

1. When a node receives an LSU, it checks its log for an LSU from the same source with the same sequence number.
2. If the LSU is found in the log, the newly arrived copy of the LSU is discarded.
3. If the LSU is new, the router transmits it on all the interfaces, except the one from which it received the LSU.
4. The router then updates its network database with the Link States from the LSU.
5. The router performs calculations of the Shortest Path tree on the network graph, and updates its routing table

The Shortest Path tree calculations in Link State routing are usually performed using the Dijkstra algorithm [2, 16]. This algorithm fills the Shortest Path tree, starting with the Shortest Paths first, hence its name. The algorithm can be described as follows:

1. The node doing the calculations puts itself in the tree as a root
2. It puts all the neighbours on the candidate list
3. From the candidate list, the node (let us call it node X) with the shortest distance from the root is selected and placed in the tree.
4. All neighbours of the node X are added to the candidate list. If the neighbour is already in the candidate list, its old distance from the root is compared with one through X, and the shorter distance is selected.
5. Steps 3 & 4 are repeated until all nodes are moved to the tree.

3. Wireless Routing Techniques

3.1 Tier Routing

DARPA packet radio utilises *tier routing* [28, 11, 13]. Every 7.5 seconds nodes broadcast Packet Radio Organisation Packets (PROP), which allow neighbours to determine the link quality and to receive routing information. Tier routing is, in essence, a classic Distance Vector algorithm with hop count as a distance metric. Nodes with the same distance are said to belong to the same tier. The only addition in DARPA routing table is a flag to indicate whether the route contains poor quality links. The following three techniques are used to utilize broadcast environment:

- **Passive acknowledgment on link level**: when a node hears that the next-hop-node has transmitted the packet, it considers it as the passive acknowledgment
- **Alternative path routing**. When a node fails to receive an acknowledgment from the next-hop, it broadcasts packet with a flag for alternative route request set.
- **Overheard routed traffic** is used to update tier tables. In order to allow that, each packet contains source and destination addresses, the number of hops already travelled, and the number of hops yet to travel.

The advantage of the tier routing protocol is in its simplicity. However, the possibility of long-lived routing loops is much greater in wireless networks, especially with a likelihood of highly connected topology. The Special techniques for Distance Vector protocols, Split Horizon and Poisoned Reverse do not break multi-node routing loops. It is, therefore, understandable that better routing solutions are being sought for present-day ad-hoc wireless networks.

3.2 Destination Sequenced Distance Vector Protocol

Perkins and Bhagwat [18] describe Direct Sequence Distance Vector (DSDV) routing protocol, which is a modification to the basic Distance Vector protocol. In DSDV, an extra field, called *destination sequence number*, is added to the routing table.

Every node in the basic Distance Vector algorithm advertises a route to itself with distance 0. In DSDV, the routers increment the *destination sequence number* field every time they advertise a route to themselves.

The routing table at each host is updated either if the sequence number to a particular destination is higher than the previously stored in the table, or if the sequence number is the same, but the distance metric is better. Adding destination sequence numbers to the routing updates improves protocol convergence.

Counting to infinity problem is eliminated by a special technique. When a link is broken, then, for all the unreachable destinations, intermediate nodes increment destination sequence numbers by one, and generate routing updates with infinity metric. On receiving this update, any node in the network has to discard all routes with the previous destination sequence number.

Two types of routing updates are used: full dump, which is broadcasted relatively infrequently, and incremental update, which contains all the routes that have changed after the full dump.

Damping Fluctuations technique is suggested to eliminate a problem of advertising short-lived sub-optimal routes. If for some reason a node consistently receives a route with the worse metric first, it delays rebroadcasting the new route entry until it receives the better route. This technique reduces

the protocol traffic overheads. However, average delay for route propagation will be increased, thus negatively affect the convergence time.

3.3 Dynamic Source Routing

On-demand source routing is not new for networking. Source routing was previously used in Token Ring 802.5 Source Routing Bridges to discover a route that is different from the one that follows branches of the spanning tree [21]. Johnson and Maltz [10] suggest *dynamic source routing* as a routing strategy for ad-hoc wireless networks. A similar technique is described in [19]. Instead of running a protocol for maintaining routing tables, as in Distance Vector or Link State algorithm, on-demand *route discovery* is suggested. *Route maintenance* protocol is run to update the routing information. Source node keeps discovered previously routes in its *route cache*. Entries are deleted from the cache either if they are expired, or if one of the links in the route brakes down. In the latter case, new route discovery is initiated. The idea is to minimise overheads associated with network traffic created by routing updates. When new routes are not required, no routing updates are initiated. Hosts initiate route discovery when they need to send a packet to a new destination. Route discovery is, in effect, a network flooding protocol. As route discovery propagates through the network, intermediate nodes append their addresses to the path in the packet and rebroadcast the request, until it reaches the destination. Then destination initiates route reply. For networks with unidirectional links, route reply is piggybacked to the route discovery from the destination to the originator. The following checks are used to eliminate routing anomalies:

1. On receipt of the route discovery packets, network nodes check the <initiator, request ID> pairs and discard all the duplicates.
2. If node's address is already in the path, requests are also discarded.

The following are the concerns related to the dynamic source routing technique:

- The user traffic carries extra overhead in the packet header: in source routing, addresses of all intermediate routers have to be included.
- Routing cache in each node can be seen as a virtual circuit connection. Virtual circuits are vulnerable to failures of intermediate nodes, because in case of a failure, the packets in all nodes along the virtual circuit are lost. In mobile networks, where node failures are very likely, a datagram service can be a better alternative. In case of a node failure in datagram networks, only packets that were queued up at the failed node are lost.
- In high-density networks, route discovery flooding can have a devastating effect on the network. Perlman [21] expresses concern for using source routing bridges for highly connected topology. In wireless networks, the network capacity will be exceeded much quicker than in wired networks. When this occurs, the route discovery will fail, and the user traffic will be lost in the congestion. Jiang et al [9] suggest using clusters to overcome this flooding problem. Only Cluster Heads and gateways participate in flooding. However, since flooding creates the source routes as it goes, it essentially means that after the cluster formation, only gateways and Cluster Heads will become routers, while all ordinary members will be just end nodes. We believe that this feature is undesirable in ad-hoc wireless networks. Where wired infrastructure exists, gateways and base stations are usually connected to backbone networks with higher available bandwidth. In the wireless ad-hoc networks, Cluster Heads and gateways share the same communication media with other nodes, and, therefore, routing packets through them would produce uneven load distribution in the network.

4. Routing Algorithm

4.1 Link costs

In Link State routing for wired networks, link costs are traditionally associated with communication delays. The main purpose for dynamic metrics is to avoid congestion at the network layer. Link delays depend on the time packets are buffered in routers before then can be transmitted over the link. This technique is effective when links between routers are provided by dedicated communication lines. In mobile wireless networks, congestion is more likely to occur on the data link layer, where all the links share the wireless medium. We select our routing metric to enable the routing algorithm to adapt to the network topology created by nodes' mobility. We rely on the data link layer to provide the information about received power measurements.

Figure 2 illustrates the advantage of wireless networks where transmitters can adjust their power. In the systems with power control, before transmitting a packet the transmitter sets its power level to comfortably reach the receiver, but not more. Studies on the radio propagation show that in order to double the transmission range, the transmitter has to use 2^4=16 times more power [7]. In figure 2, the four boundary nodes are not affected if the packet is forwarded over two hops. One hop transmission, however, generates radio interference for the boundary nodes, which effectively means that there is less bandwidth available on other links. Apart from minimizing interference with the other links, limiting transmission power also helps to prolong the mobile node's battery life.

Another advantage of short hops is the longer link lifetime. Even if the next hop node moves further apart, it will still be within the range. On the other hand, routing protocols based on hop count, or minimum delay, would often have to perform route maintenance to recover from node movements out of range. We assign link costs to be proportional to the transmission power that is necessary to reach the neighbour node. In the figure two, the two-hop path has distance metric of two, whereas the one-hop path has a metric of 16.

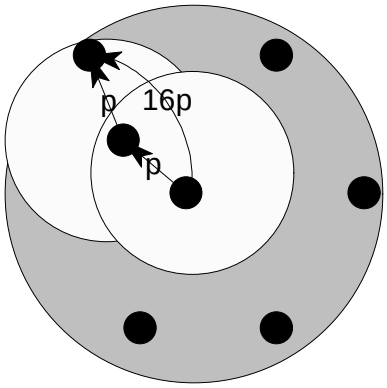

Figure 2. Adaptive power control in wireless networks

4.2 Cluster Formation

Our cluster formation is based on the Link Cluster Algorithm (LCA) from [1] (see also [5, 6, 9]). An alternative clustering algorithm can be found in [12], where clusters were used to improve the Distance Vector protocol performance. In [12] Krishna et al define clusters as fully connected sub-graphs. The nodes connected to other clusters are called boundary nodes. The non-boundary nodes no not have to advertise their routing tables, and passively listen for the updates from the boundary nodes. Thus, the boundary nodes act as routers in wired networks, while the non-boundary nodes act as hosts. Krishna's clusters also provide a framework for flooding, since the non-boundary nodes do not broadcast flooding packets.

For this work, we selected LCA from [1] for several reasons. Firstly, it creates bigger clusters and requires less frequent cluster reorganisations. Bigger clusters mean less broadcast transmissions during flooding. Secondly, the data link layer can also use LCA for the channel and code allocation [6]. Keeping in mind an infrastructure for efficient flooding, our objective is to designate nodes that would perform the following two functions:

- *Cluster Heads*, to broadcast flooding packets to the other nodes in their cluster
- *Gateways*, to deliver the flooding packets to the neighbouring Cluster Heads.

Functions of the Cluster Heads in the flooding algorithm will be similar to those performed by the designated routers in Open Shortest Path First (OSPF) [24]. Gateways will act as ordinary routers in OSPF. At any point in time, a node in the mobile network associates itself with a cluster. The clusters are formed around cluster heads, and are identified by the Cluster Head ID.

To enable the cluster formation and maintenance, all nodes keep the information about their neighbours in the neighbour table.

Neighbour ID	Neighbour's Cluster Head ID	Timer	Received power measurement
…	…	…	…

Table 1. Neighbour Table Example

4.2.1 Maintaining the Neighbour Table

Each node periodically broadcasts Hello packets as follows:

Node ID	Node's Cluster Head ID

The Hello packets are only delivered to the immediate neighbours and are not passed any further. When a node receives a Hello packet, it updates its neighbour table. If the entry with the received Node ID does not exists in the table, a new entry is added, and the timer is initialised. If the entry already exists, then the neighbour's Cluster Head ID and the received power measurements fields are updated, and the timer field is reset. When the timer expires, the corresponding entry is removed from the neighbour table.

4.2.2 Cluster Head Selection

Let t be the period between the Hello broadcasts. When a node first switches on, it first listens to Hello packets on the broadcast channel. If any other node on the broadcast channel is already advertising itself as a Cluster Head (Node ID = Node's Cluster Head ID), the new node saves the heard Cluster Head ID in its Cluster Head ID field. If consequently the new node receives a Hello packet from another Cluster Head, it is simply added to the neighbour table.

The new node has to listen on the broadcast channel for long enough, to be able to detect existing Cluster Heads with a high probability. Before the node decided on its Cluster Head, it can advertise Hello packets with zeros in the Cluster Head ID field. Suppose the new node does not hear Hello packets from any Cluster Head for the period of $20t$ [1] after it switches on. Then the new node becomes a Cluster Head itself. To ensure quick formation of clusters, the time period t between the Hello broadcasts should be chosen rather small. This should not be a problem, because the Hello packets are very short and would not consume much bandwidth

When the timer in the Neighbour Table expires, and the entry is being removed, the node is expected to check whether removed entry belonged to the node's Cluster Head. If so, the node searches its neighbour table for the first entry with (Neighbour ID = Neighbour Cluster Head ID), and accepts this neighbour as its new Cluster Head. If no entry is found, the node itself should become a Cluster Head.

If a Cluster Head hears a Hello packet from another Cluster Head, it should compare its ID with the one of the heard Cluster Head. The node with higher ID will then resign as a

[1] Under a reasonable channel access protocol, 20 Hello periods would allow to receive Hello messages from at least 10 neighbours.

Cluster Head, and will advertise its new status on the network.

In our protocol, a node that joined the network earlier is more likely to become a Cluster Head. This is a small optimization to the pure lowest ID protocol, where nodes have to give up their status every time they hear from a lower ID node. This will reduce number of Cluster Head changes, because if a node with lower ID switches on inside the cluster, it does not force the old Cluster Head to give up its status.

This clustering algorithm guarantees that:

- Cluster Heads are not in the direct communication range of each other.
- Any node in a cluster either is a Cluster Head itself, or directly connected to at least one Cluster Head.

4.2.3 Gateway assignment.

The Cluster Head is responsible for assigning gateways to the neighbouring clusters. The Cluster Head learns information about network connectivity from Link State Updates (LSU). Each node in the network learns the information about its neighbours from received Hello packets, and stores this information in its neighbour table (Table 1). Periodically, it sends the update about its incoming links to its Cluster Head in the following form:

Node ID	Node's Cluster Head ID	
Neighbour ID	Neighbour's Cluster Head ID	Received power measurement
…	…	…
…	…	…

Among the nodes that have connections to neighbouring clusters, outbound gateways are selected. For each of the neighbouring clusters, the following selection is performed:

1. Among nodes with direct link to the neighbour Cluster Head, the one with minimal cumulative link cost is selected.
2. If there are no nodes with direct link to the neighbouring Cluster Head, a node that has a link to any other node in the neighbouring cluster is selected.

An example of cluster infrastructure that can be obtained from the above algorithm is shown in figure 4.

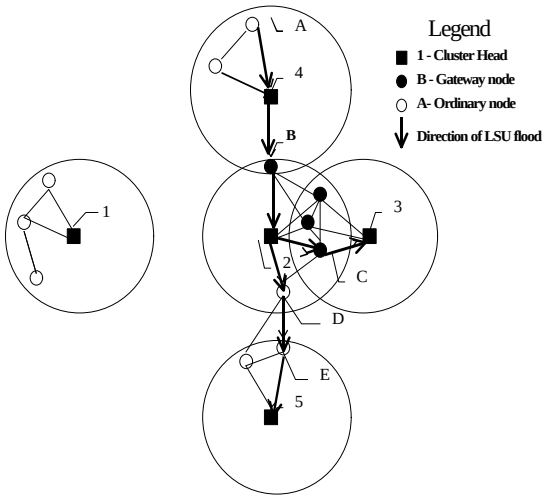

Figure 3. Example of clusters.

Clusters 2 and 3 can be connected via three gateway nodes. Suppose, the Cluster Head 2 has assigned node C as a gateway to the cluster 3. Also, suppose that node 2 has assigned node D as its gateway to the cluster 5. B is the Cluster Head 4's only option for connecting to cluster 2. Cluster 1 is disconnected.

4.3 Flooding

The flooding protocol is using the cluster infrastructure formed above. We will describe the flooding protocol using an example of the network in figure 4. The example shows how an LSU is distributed to all the nodes in the network. Nodes B, C and D in figure 4 are assigned gateways. The algorithm is described as follows:

1. Node A sends its (LSU) to its Cluster Head, node 4.
2. Cluster Head 4 receives the LSU, sends an acknowledgment to node A, and broadcasts the LSU for its cluster.
3. Node B, knowing that it is an assigned gateway, sends an acknowledgment to node 4, and forwards the LSU to Cluster Head 2.
4. Cluster Head 2 receives the LSU, sends an acknowledgment to node B, and broadcasts the LSU for its cluster.
5. Node C, knowing that it is an assigned gateway, sends an acknowledgment to node 2, and forwards the LSU to Cluster Head 3. Node D, knowing that it is an assigned gateway, sends an acknowledgment to node 2, and forwards the LSU to Node E.
6. Node E sends an acknowledgment to node D, and forwards the LSU it to Cluster Head 5.
7. Cluster Head 5 receives the LSU, sends an acknowledgment to node D, and broadcasts the LSU for its cluster

Flooding procedure ends here ■.

For clarity, we stopped the description of the flooding procedure at the point when it was delivered to all the nodes in the connected segment of the network. One can notice that the

flooding would continue, because assigned gateways from clusters 3 and 5 to cluster 2 will continue forwarding packets. Cluster Head 2 is responsible for discarding the LSU duplicates when they arrive from clusters 3 and 5.

All nodes in the cluster are expected to acknowledge the LSU. If one of the nodes does not send an acknowledgment, the Cluster Head retransmits the LSU to that particular node.

4.4 Routing

Now when the procedure for flooding is developed, routing is straightforward. Each node in the network maintains full network database. After receiving every LSU, it recalculates best routes using the Dijkstra algorithm as described in section 2.2, and updates its routing table. The link costs are set to be inversely proportional to the received power readings.

A node obtains a feedback from a neighbour about the received power level via the neighbour's LSU. When forwarding the user packets to its neighbours, the node is required to adjust its transmission power so that it is sufficient to create the necessary, but not excessive signal power level at the receiver node. The power control is thus performed using closed loop method [7]. Dijkstra's algorithm guarantees consistency of the shortest path along the route. That is, intermediate nodes' routing tables will contain exactly the same nodes for the next hop towards the destination as it was calculated by the packet source. Thus, Routing headers only need to contain destination ID. It is an advantage over on-demand source routing protocol, which create significant header overhead in the user packets.

4.5 Uni-directional links

All the links that carry LSU have to be bi-directional, because reliable flooding procedure requires LSU acknowledgments. For user traffic, however, unidirectional links can be used. This is how it works: Suppose node X received a Hello packet from node Y. Regardless of whether Y can hear X or not, X includes link from Y in its LSU. Y will receive LSU from its Cluster Head via flooding. It will update its database with a uni-directional link from Y to X. It will then use this link in its Shortest Path calculation, which may result in X being next hop for some of the destinations.

5. Conclusions

Scarce communication bandwidth and the high likelihood of frequent topology changes impose new requirements on routing algorithms for mobile ad-hoc networks. Several routing algorithms have been developed in an attempt to improve convergence and, at the same time, minimize routing traffic. Various modifications to Distance Vector algorithms would have done extremely well if implemented in wired networks. However, many attempts to apply routing algorithms to wireless networks fail to recognise substantial differences in underlying communication technology. All protocols use network flooding in order to eliminate possibility of routing loops. In case of broadcast flooding, achievements in reducing overhead on the network layer will be wasted due to increased interference at the physical layer. If flooding is done in a more efficient way, eg., [9], selected nodes have to handle more traffic without having extra bandwidth for it.

The routing protocol presented in this paper does not impose extra burden on lower layer protocols. The problem of flooding is resolved using clusters. Regardless of the node's position in the clustered hierarchy, each node in the network acts as a router. Table 2 shows a comparison between different proposed algorithms in wireless ad-hoc networks. The table shows that flooding is an inherent feature for any routing protocol adapted for networks with dynamic topology. If the clustering structure is utilized, the Link State Algorithm with Cluster Based Flooding allows for more even distribution of the network load. In Distance Vector protocols, the user traffic has to follow the routing traffic, and, therefore, will create congestions in Cluster Heads. Inherent for Link State routing flexible link costs allow the routing algorithms to be tightly coupled with link-layer optimisations like variable transmission power.

Routing Algorithms	**Routing Loops**	**Sub-optimal Routes**	**Flooding employed**	**Action that needs flooding**	**Non-flooding routers**	**Unidirectional links**	**Preferred Routing Metric**
Tier Routing (Distance Vector)	Yes	Yes	no	N/A	N/A	No	Hop Count
Destination Sequenced Distance Vector	No	No	yes	Triggered update	No	No	Hop Count
On Demand Source Routing	No	Yes	Yes	Route discovery	No	Yes	Minimum Delay
Link State with Cluster Based Flooding	No	No	Yes	Link State updates	**Yes**	For non-flooded links	**Transmission power**

Table 2 Presence of Flooding in different routing algorithms

Although the nodes in the proposed protocol participate in cluster formation, the routing architecture remains flat and, therefore, only suitable for small to medium size wireless ad-hoc networks. Hierarchical structures for routing in large packet radio networks were developed for SURAN program [26].

6. Future Work

Building wireless networks is an expensive endeavor. We recognize the importance of simulation for validation and verification of network algorithms. We believe that simulation for wireless ad-hoc routing protocols should be taking into account the protocol stack's operation on the data link layer.

We reckon that the protocol presented in this paper will work under a variety of conditions, but would be most beneficial with the link layer supporting Spread Spectrum CDMA, and transmission power control. Our goal is to build the simulation model to be as realistic as possible, and to compare the Link State routing with other related methods.

References

1. Baker D.J., Ephremides A., Flynn J.A., "The Design and Simulation of a Mobile Radio Network with Distributed Control", *IEEE Journal of Selected Areas in Communications*, Vol. SAC-2, No1, January 1984, pp 226-237
2. Bertsekas D., Gallager A., "Data Networks", Prentice Hall 1992
3. "Cellular Digital Packet Data System Specification", *CDPD Forum*. Release 1.0, July 19, 1993
4. Corson M.S., "Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Considerations", *IETF Internet Draft*, work in progress, September 1997
5. Ephremides A., Wieselthier J.E., Baker D.J., "A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signalling", *Proceedings of the IEEE*, vol 75, No1, January 1987
6. Gerla M., Tsai J.T.C., "Multicluster, mobile, multimedia radio network", *Wireless Networks*, Baltzer, No1, 1995, pp 255-265
7. Glisic S, Vucetic B, "Spread Spectrum CDMA Systems for Wireless Communications", Artech House, 1997
8. Gower N., Jubin J., "Congestion Control Using Pacing in a Packet Radio Network", in *Proceedings of Milcom 82* pp 23.1-1 - 23.1-6
9. Jiang M., Li J., Tai Y.C., "CBRP functional specifications", *IETF Internet Draft*, work in progress, August 1998.
10. Johnson D. B., Maltz D.A., "Dynamic Source Routing in Ad Hoc Wireless Networks" in *Mobile Computing*, Kluwer Academic Publishers, 1996, pp 153-181
11. Jubin J, "Current Packet Radio Network Protocols", in *Proceedings of IEEE Infocom 85* pp 86-92
12. Krishna P., Chatterjee M., Vaidya N.H., Pradhan D.K., "A Cluster-based Approach for Routing in Ad-hoc Networks", *Second Usenix Symposium on Mobile and Location Independent Computing, 1995*.
13. Lauer G.S., "Packet Radio Routing", in (ed) Steenstrup Marta, *Routing in Communications Networks*, Prentice-Hall, 1995, Chapter 11
14. Lee W. C. Y., "Mobile Communications Design Fundamentals", John Wiley & Sons, 1993.
15. Lynch C.A. and Brownrigg E.B., "Packet Radio Networks, Architectures, Protocols, Technologies and Applications", Pergamon Press, 1987.
16. McQuillan J.M., Richer I., Rosen E.C., "The New Routing Algorithm for the ARPANET", *IEEE Transactions on Communications*, vol COM-28, No5, pp 711-719.
17. Mouly, M. & Pautet M.B., "The GSM System for Mobile Communications", published by authors, 1992.
18. Perkins, C.E., Bhagwat, P., "Routing Over Multi-Hop Wireless Network of Mobile Computers" in *Mobile Computing*, Kluwer Academic Publishers, 1996. , pp 183-205
19. Perkins, C.E., "Ad-hoc on-demand Distance Vector routing", *IETF Internet Draft*, work in progress, November 1997
20. Perkins C E., Mobile Networking Through Mobile IP, *IEEE Internet Computing*, January-February 1998, pp 58-69
21. Perlman R., "Interconnections, Bridges and Routers", Addison-Wesley, 1996
22. Pursley, M.B., "The Role of Spread Spectrum in Packet Radio Networks", *Proceedings IEEE*, vol 75, no1, pp 116-134, Jan1987.
23. Hendric C., "Routing Information Protocol", *IETF Network Working Group. Request for Comments 1058*, June 1988.
24. Moy J., "OSPF version 2", *IETF Network Working Group, Request for Comments: 2178*, July 1997.
25. Shacham N., Tornow J., "Future Directions in Packet Radio Technology", *Proceedings of IEEE Infocom 85*, pp 93-98.
26. Shacham N., Westcott J., "Future Directions in Packet Radio Architectures and Protocols", *Proceedings of the IEEE*, vol 75, No1, January 1987, pp 83-99.
27. Toh C.K., "A Novel Distributed Routing Protocol to Support Ad-hoc Mobile Computing", *International Phoenix Conference on Computers and Communications*, 1996, pp 480-486
28. Westcott J., Jubin J., "A Distributed Routing Design for a Broadcast Environment", *Proceedings of Milcom 82*, pp 10.4-1 - 10.4-5.