Project on Mobile Ad-hoc Networking and Clustering

for the course EDA 390 Computer Communication and Distributed Systems

Contents

Introduction

A mobile ad-hoc network, or MANET, consists of identical nodes that travel around freely and independently and communicate wirelessly. The obvious problem in ad-hoc networking is how to transfer messages between two nodes that are not in range to communicate directly. Ad-hoc is a very hot topic at the moment and is already expanding greatly in use. It has a lot of applications and many good reasons for its popularity. Ad-hoc networks are easy to deploy and quickly self organized. It is decentralized and independent on infrastructure. Presented in this rapport are a few of the many different routing protocols and clustering algorithms.

History

In the late 1960s the Air Force Office of Scientific Research started to do some research on the subject of using radio for a packet switched network communication. In the early 1970s DARPA provided continued funding for this research at the University of Hawaii. The project was called ALOHA (Areal Locations of Hazardous Atmospheres) and here not only were new ideas of mobile packet radio born but also the basis of Ethernet was developed. In 1977 there was a big demonstration of the Packet Radio net, SATNET, and the ARPANET where messages were sent across the US.

In 1983 another DARPA funded project called SURAN (Survivable Radio Network) started. In this project the goals were to:

* develop a small, low-cost, low-power radio that would support more sophisticated packet radio protocols than the DARPA Packet Radio project from the 1970s
* develop and demonstrate algorithms that could scale to tens of thousands of nodes
* develop and demonstrate techniques for robust and survivable packet networking in sophisticated electronic attacks

There were several follow-up projects funded by DARPA projects. Two examples of there are Low-cost Packet Radio (LPR) and GloMo.

Applications/Uses

The original applications of ad-hoc networks were of a military kind. Vehicles on a battlefield are certainly mobile and move around in unpredicable manners. This is the original scene from where the very idea of ad-hoc networking was born. Other applications are in rescue opperations where ad-hoc networks could be used by both police and firefighters. Search and rescue missions are also suitible applications. Possible commercial purposes could be for taxi communication, on boast, aircrafts and in sports stadiums. Personal uses are for laptops and notebook computers. Ad-hoc networking has even reached the entertainmentbusiness with sonys playstation portable which uses this for multiplayer gaming.

Routing

The obvious problem with ad-hoc networking is how to send a message from one node to another with no direct link. This is the problem of routing. Because the nodes in the network are moving around unpredictibly, which nodes that are directly linked together are changing all the time. This means that the topology of an ad-hoc network is constantly changing and this is what makes routing so difficult.

There are two main approaches to this problem. There are also combinations of the two. The first approach is a pro-active approach which is table driven and uses periodic protocols. This means that all nodes have tables with routing information which are updated at intervals. The second approach is re-active, source-initiated or on-demand. This means that every time a message is sent it first has to find a path by searching the entire network.

There are many different protocols that are in accordance with the two different routing approaches. Different protocols are specialized in different aspects of the routing. Other aspects than finding a short path are low overhead communication and loadbalacing.

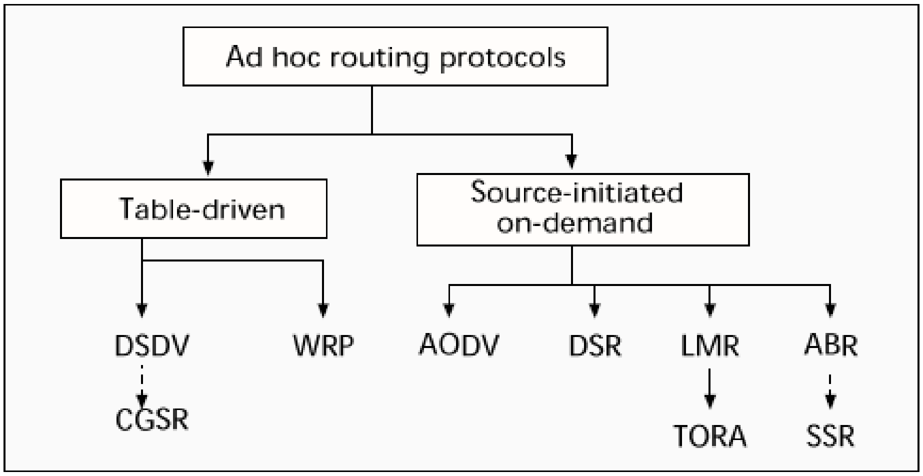
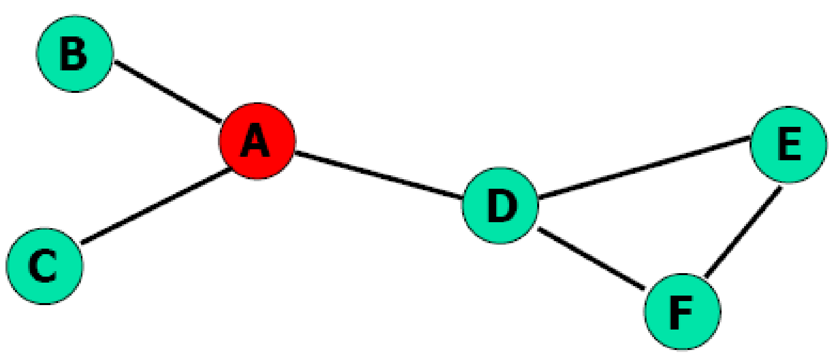


Table Driven Routing / Periodic Protocols

DSDV (Destination-Sequenced Distance-Vector Routing)

DSDV routing [1] uses a distributed variant of the Bellman-Ford algorithm. Every node in the DSDV network has a table that holds information about every other reachable node in the network. For every node the information stored is: the next hop node, the hop distance and a sequence number. In the Bellman-Ford algorithm, every node sends its table to its neighbours at time intervals. When a node receives a table from a neighbouring node it updates its own table. At every new time interval when a node is to broadcast its table, the sequence number is incremented. This way a node receiving a table knows how up-to-date the information is. New information replaces old information, and better information (a quicker route) replaces worse, if the table information has the same sequence number. In order to keep the network traffic due to updating information down, the whole routing table is not sent every time (full dump) but smaller incremental packages only containing the changed information since the last full dump.



Destination	Next Hop	Distance	Sequence Number
A	A	0	S556_A
B	B	1	S854_B
C	C	1	S226_C
D	D	1	S273_D
E	D	2	S280_E
F	D	2	S823_F

CGSR (Clusterhead Gateway Switch Routing)

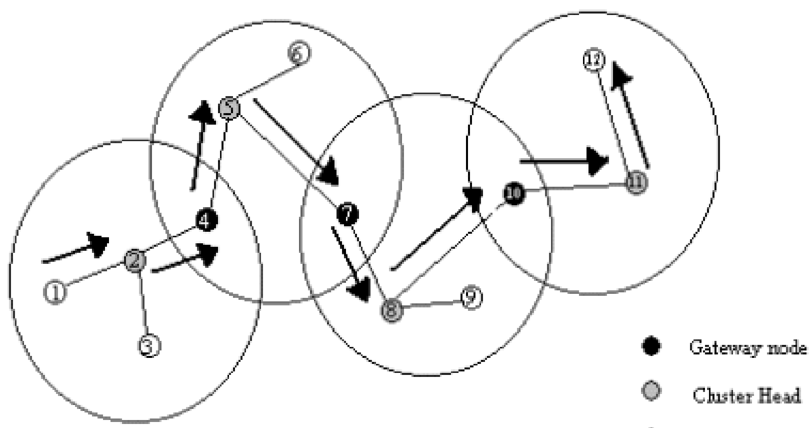
In CGSR [2] the ad-hoc network is divided into clusters. Every cluster contains a node that is given the role of being a clusterhead and nodes that have direct links to two or more clusterheads are gateways. Routing is then done through the clusterheads using the same method as in DSDV. Dividing up the network into clusters can be done in many ways (more on this later) but the most important thing is that the clusterhead roles should not be changed too frequently. This leads to an unnecessary amount of communication for just organizing the network.

To deal with this problem the LCC (Least Cluster Change) algorithm is therefore used. What this algorithm does is that the clusterhead election is only performed at two occasions. The first occasion is when two clusterheads get so close together that they can communicate directly. The second is when a node loses contact with a clusterhead.

Every node has two tables. One table contains the information of which clusterhead every node in the network belongs to. The other table is a distance-vector-routing table with the information of the next hop node for all clusterheads.

When a message is to be delivered from node 1 to node 12 in the example below, routing is done by the following steps. First the clusterhead of node 12 is looked up in the cluster member table to find node 11. Second the next hop node for clusterhead-node 11 is looked up in the distance-vector table to find node 2. The message is then sent to node 2 which passes it on to its gateway-node leading to clusterhead-node 11. It is them passed on between gateways and clusterheads until it reaches node 11. Node 11 then passes is on to the destination node 12.

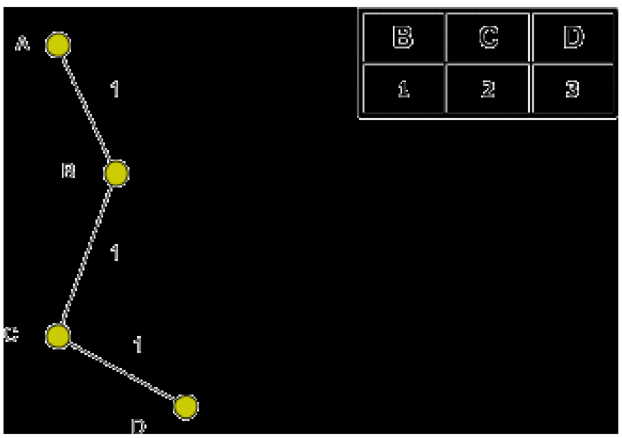
The advantages of CGSR over DSDV are that routing is performed only through clusterheads reducing the size of the distance-vector tables. There is still the problem of much bandwidth used up by keeping the cluster member table up to date.

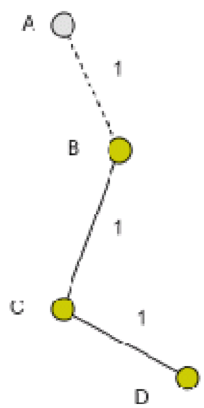


WRP (Wireless Routing Protocol)

A problem of the Bellman-Ford protocol is the count-to-infinity problem. This is a problem that is solved by WRP [3]. When the link between a node and its only neighbour is broken it loses contact with the network. In the example below node A loses contact with node B. When B realizes that it has lost direct contact with A it looks for other paths. When B receives the table information of C it sees that, C has a link to A by 2 hops. B updates its table by adding one hop for the distance to A, and sets the next hop to be C. When C receives B updated table, it notices that the distance from B to A has increased. Since the entry for next hop node going to A is B, the entry for the hop distance to A has to be changed to what B has plus one. As tables are updated no one will realize that they have lost contact with A but the distance to A will count up to infinity.

WRP avoids this count-to-infinity problem by doing consistency checks of predecessor information reported by all its neighbours.





B	C	D
3	2	3

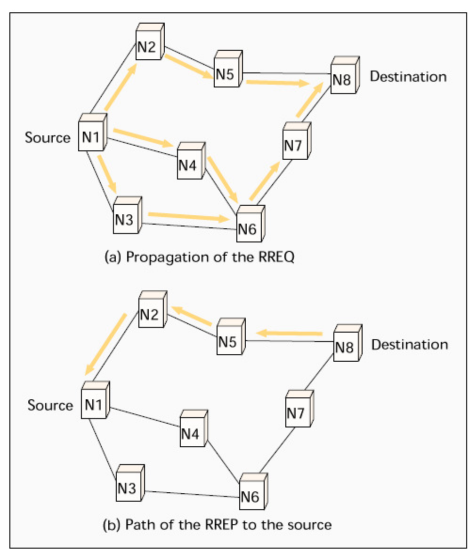
B	C	D
3	4	3

B	C	D
5	4	5

On-Demand Source-Initiated Routing

AODV (Ad hoc On-Demand Distance Vector Routing)

AODV [4] is an improvement of DSDV. Since it is on-demand routing information betweens nodes that never communicate will not be considered. When one node is to communicate with another and it doesn't know of a route, it broadcasts a route request RREQ to all neighbours. The RREQ is forwarded until it reaches its destination or it finds a node with a fresh enough route to the destination.



(a) Propagation of the RREQ

(b) Path of the RREP to the source

AODV uses destination sequence numbers to make sure all routes are loop-free and contain the most recent routing information. Each node keeps track of its own sequence number and a broadcast ID which is incremented every time a RREQ is sent. The RREQ contains information of the source IP address, the source sequence number, the broadcast ID and the most recent sequence number known for the destination address.

When an intermediate node receives a RREQ it can respond with a route reply RREP if it has a route to the destination with a sequence number that is greater of equal to the sequence number of the RREQ. Since the RREQ propagates by nodes passing it forward to its neighbours, an intermediate node will receive the same RREQ from many neighbours. Only the first copy of the RREQ is treated all others are discarded.

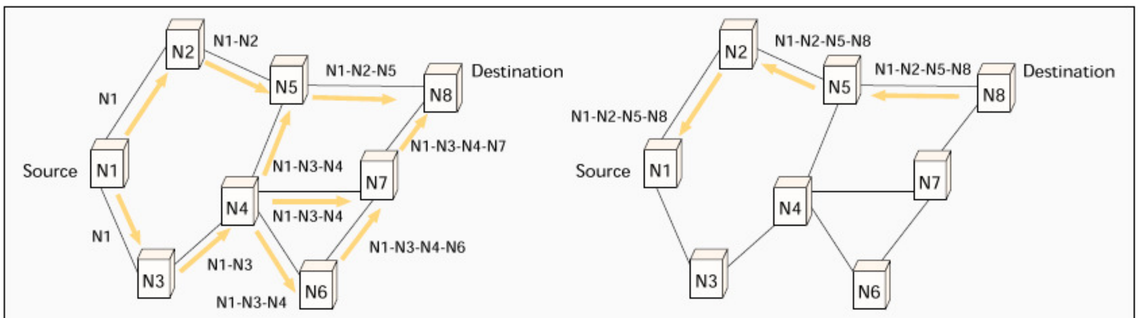
An intermediate node receiving the first copy of a RREQ stores the information of which neighbour sent it. When the RREQ reaches the destination node or an intermediate node with a fresh enough route, the RREP message is sent back to the source by backtracking the RREQs path using the information of from where all intermediate nodes received their first RREQ. The nodes on the path of the RREP store the information of from which neighbour came, setting up an active forward route.

When the link between the source node and its next hop node is broken, the source node simply sends another RREQ to its remaining neighbours. If a link between two intermediate nodes is broken a message of this event is sent back to the source node which then can decide if it wants to send a RREQ to re establish the route to the destination or not.

DSR (Dynamic Source Routing)

In DSR [5] all nodes keep a route cache which holds routing information of other nodes. Entries in the routing cache hold the entire routing information of a route, not just the next hop node. If a node is not in the routing cache that a source node wants to communicate with, the source node broadcasts a route request, much like in AODV. This route request holds the information of source and destination but also the nodes on the path, called a route record. When an intermediate node receives a route request, it checks if it is in this route record. If it is, the message is discarded; otherwise it adds itself to the route record and sends the route request on to its neighbours.

When a route request reaches a node that has a route to the destination in its routing cache, the node adds itself and the routing cache information to the route record and sent back a route reply, containing the route record, to the source. If the route request reaches the destination node, it also adds itself to the route record and sends back a route reply.



(a) Building of the route record during route discovery (b) Propagation of the route reply with the route record

TORA (Temporally Ordered Routing Algorithm)

TORA [6] is a routing protocol suitable for very mobile ad-hoc networks. What is unique for TORA is that it focuses on solving the routing problem at the local area where the topology has changed. It does this by letting all nodes hold routing information of the neighbouring nodes. TORA has a problem of stability. There is a risk of oscillations occurring, very similar to the problem of count-to-infinity in DSDV.

ABR (Associativity-Based Routing)

ABR [7] differs from other routing protocols by introducing the idea of degree of association stability. All nodes hold a measure of the neighbours associativity. At time intervals nodes send out a signal to all its neighbours, telling them that there still there and the link is not broken. Every time such a signal is received the associativity of the sending is increased. When the link to a node is broken the associativity of that node is reset.

In this way the associativity of a node says something about its mobility. A node that moves around a lot will get new neighbours and lose old neighbours at a higher pace. This means that the links aren't very old and therefore have a lower degree of associativity. The opposite case is a node which is not moving, have links that have been up a long time and thus high associativity. This measure is then used when routing to find routes that are stable.

SSR (Signal Stability Routing)

SSR [8] similarly to ABR also introduce new ideas of how to measure which is the best route. SSR uses the measure of signal strength of a link. A link with a strong signal reduces the risk of packet loss and also indicates that the two nodes are relatively close. If two nodes are close the time for them to be separated is greater, thus the chance that the link is going to stay up longer is greater too.

Summery of Table-Driven Protocols

Because in DSDV every node holds information of routing to everyone and this information is updated at intervals, it has problems with a large amount of overhead wish grows as $O(n^2)$. In CGSR routing is limited to the clusterheads. This allows for more nodes than in DSDV but there is still a lot of data maintained in the two tables that every node holds. Also routing is dependant on clusterheads, making the organization less distributed. This causes a lot of resources to be consumed when running the clustering algorithm.

In WRP every node has four tables to avoid the temporary routing loops that the other table-driven protocols have problem with. However the time to repair a route due to a broken link is much lower in WRP since it only notifies the neighbouring nodes about the changes.

Summery of Source-Initiated On-Demand Protocols

The overhead of DSR is greater than that of AODV because the packages contain the whole route information where as in AODV only the next hop node is considered. The memory overhead is also larger because of the same argument. An advantage of DSR however is that it doesn't send periodic routing messages. This means that power is saved and the utilization of the bandwidth is better. DSR can hold several route cache entries for the same destination. This is the fastest way of solving the problem with a broken link, since no repair procedure has to be invoked.

Both AODV and TORA can support multicast. The main advantage of TORA is, like DSR, its support for multiple routes. The main drawback of TORA is that it demands the nodes to have synchronized clocks. So if the nodes don't have GPS or some other way of synchronizing clocks TORA cannot be used. Also if the synchronization fails the algorithm fails.

ABR doesn't always chose the route with the least hops, but the most stable route. This means that the route will be more long-lived, thus fewer route reconstructions have to be made. Another advantage of ABR, in contrast to all other protocols, is that it avoids duplicate packages. The drawback on the other hand is the power consumption of the frequent beaconing messages.

SSR has properties similar to ABR. A great drawback is that intermediate nodes cannot reply to a route request, leading to long delays in finding a route. A similar problem arises during reconstruction where the reconstruction cannot start from where the link was broken, but has to start all the way back from the source.

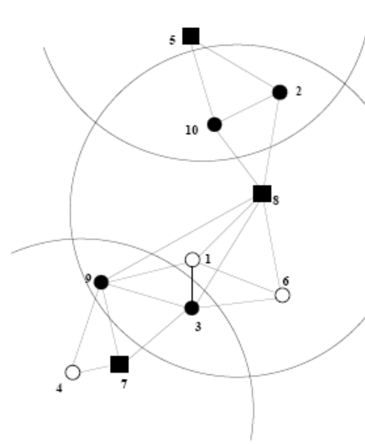
Clustering

How to divide the network up into clusters is a problem of the CGDV algorithm. There are many ways of doing this and many aspects to consider. Different consideration are; stability, loadbalancing, maximum clustersize and maximum number of hops to the nearest clusterhead. Here are presented only the earliest algorithms and the main concepts. One of the very first being LCA [9].

LCA (Linked Cluster Algorithm)

In this algoritm evey node is assigned an ID number. There are two ways of becoming a clusterhead. The first way is if you have the highest ID of yourself and your neighbours. The second way is if you have a neighbour in who's neighbourhood you have the highest ID.

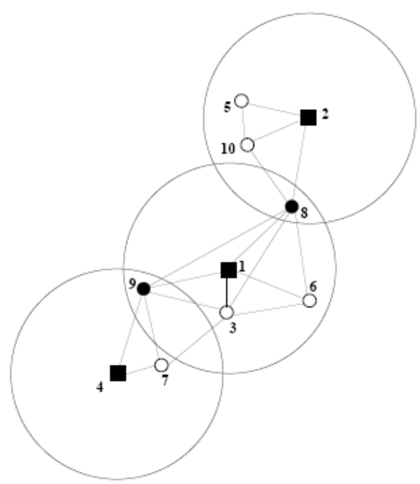
This algorithm turned out to elect an unnecessary number of clusterheads. A modification was made to form LCA2. Here the notation of being covered and non-covered was introduced. A node with a clusterhead as a neighbour is considered covered. Starting at the lowest ID a node is elected clusterhead if it has the lowest ID among its non-covered neighbours.



Highest-Connectivity Cluster Algorithm

Every node broadcasts the number of neighbours it is connected to. Then election for clusterheads work much like the LCA algorithm. Instead of looking at the ID, the connectivity is considered, that is the number of nodes one is connected to. The node with the highest connectivity is elected clusterhead. In the case of a tie, the node with the lowest ID prevails.

Good properties of both of these algorithms are that no clusterheads are directly linked and that two nodes in a cluster are only two hops away. What is good about the fact that no two clusteheads are directly linked is that it is an ensureance that the number of clusterheads cannot be too great. These algorithms are meant to be used for relatively small networks with a maximum number of nodes of about a hundred. This is because all clusters are of radius one. That means that all members of a cluster are directly linked to the clusterhead.



A problem of the algorithms above is the disadvantage of some nodes being elected clusterheads more often then others. Another algorithm that deals with this loadbalancing problem has suggested that each node has both a Physical ID (PID) and a Virtual ID (VID). The VID is used in the clusterhead election and is increased every time an election occurs. When a node has been a clusterhead the VID is set to 0. This way the probability of being elected clusterhead becomes greater the longer it has not been a clusterhead.

Other Clustering Algorithms

Other algorithms are for example ones that focus on making a very good approximation of the optimal cluster structure. The optimal cluster structure, in a network with cluster-radius-size one, is the structure that minimizes the number of clusters to cover the network. There are algorithms that provide a certain probability of staying connected at a certain time intervall. And yet another that control cluster size. All these algorithms focus on different problems making them suitable for different environments. There is no way of saying which clustering algorithm is better or worse. The answer is that it depends on what the ad-hoc network will be used for.

Conclusion

Ad-hoc networking and clustering are very hot areas at the moment. There is a lot of research going on and not just in the military, where it all started. One can imagine how different, different ad-hoc networks can behave. There can be a generally high connectivity, that is many nodes in a small area, and there can be many aspects of the mobility. In a military application one can imagine a great deal of mobility of the nodes (vehicles of soldiers) in the middle of a battle. One can also expect certain nodes to stick together if the soldiers are organized in squads. In a non military scenario when nodes are people sitting with a laptop in a park for example one expects much less mobility. All these properties affect the choice of routing technique.

Even if these different techniques are specialized in different areas some of them do have problems. Ad-hoc routing is a delicate problem and its not obvious how to do it, but because of the large number of applications and the bright future prospects, this is something I am sure we will see much more of in the future.

References

[1] Highly dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for mobile Computers
[2] Routing in Clustered Multihop. Mobile Wireless Networks with Fading Channel (CGSR)
[3] An Efficient Routing Protocol for Wireless Networks (WRP)
[4] Ad-hoc On-Demand Distance Vector Routing (AODV)
[5] Dynamic Source Routing in Ad-Hoc Wireless Networks (DSR)
[6] A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks (TORA)
[7] A Novel Distributed Routing Protocol To Support Ad-Hoc Mobile Computing (ABR)
[8] Signal Stability based Adaptive Routing (SSA) for Ad-Hoc Mobile Networks
[9] Multicluster, Mobile, Multimedia Radio Network (LCA)