Congestion control in Mobile Ad-Hoc Networks (MANETs)

COMPUTER SCIENCE AND ENGINEERING

April - 2015

Index:

1.1 Introduction

In this paper we consider the problem of congestion control in mobile ad-hoc networks (MANETs). For the different structure TCP do not work properly with the specific effects occurring in MANETs. This is because TCP has originally been designed for the Internet, a network with different properties. As a consequence, appropriate congestion control is widely considered to be a key problem for MANETs.

A mobile ad-hoc network is a collection of mobile nodes forming an ad-hoc network without the assistance of any centralized structures. These networks introduced a new art of network establishment and can be well suited for an environment where either the infrastructure is lost or where deploy an infrastructure is not very cost effective. The popular IEEE 802.11 "WI-FI" protocol is capable of providing ad-hoc network facilities at low level, when no access point is available. However in this case, the nodes are limited to send and receive information but do not route anything across the network. Mobile ad-hoc networks can operate in a standalone fashion or could possibly be connected to a larger network such as the Internet.

Mobile ad-hoc networks can turn the dream of getting connected "anywhere and at any time" into reality. Typical application examples include a disaster recovery or a military operation. Not bound to specific situations, these networks may equally show better performance in other places. As an example, we can imagine a group of peoples with laptops, in a business meeting at a place where no network services is present. They can easily network their machines by forming an ad-hoc network. This is one of the many examples where these networks may possibly be used.

1.2 PROJECT DESCRIPTION

The ad hoc routing protocols AODV, DSR and DSDV are three of the promising routing protocols. They can be used in mobile ad hoc networks to rout packets between mobile nodes.

The main objectives of this thesis project are:

(1) Implementing the existing AODV, DSR and DSDV routing protocols in ns2

(2) Comparing the performance the protocols under following TCP variants

(i) TCP RENO

(ii) New RENO

(iii) TCP SACK

(iv) TCP FACK

(v) TCP VEGAS

2.1 Features of Mobile Ad-hoc Networks

MANETs is an IEEE 802.11 framework. It is an interconnected collection of wireless nodes where there is no networking infrastructure in the form of base stations, devices do not need to be within each other‘s communication range to communicate, the end -users devices also act as routers, nodes can enter and leave over time, data packets are forwarded by intermediate nodes to their final destination.

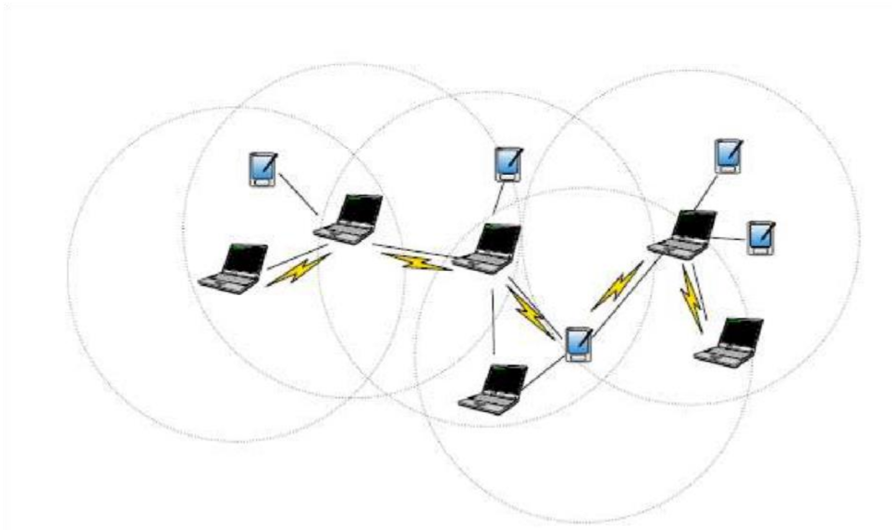


Figure1: Mobile Ad-hoc Networks (MANETs)

2.2 Characteristics of MANETs

Mobile ad hoc network nodes are furnished with wireless transmitters and receivers using antennas, which may be highly directional (point-to-point), omni directional (broadcast), probably steer able, or some combination there of [1]. At a given point in time, depending on positions of nodes, their transmitter and receiver coverage patterns, communication power levels and co-channel interference levels, a wireless connectivity in the form of a random, multihop graph or "ad hoc" network exists among the nodes. This ad hoc topology may modify with time as the nodes move or adjust their transmission and reception parameters.

The characteristics of these networks are summarized as follows:

- Communication via wireless means
- Nodes can perform the roles of both hosts and routers
- Bandwidth-constrained, variable capacity links
- Energy-constrained Operation
- Limited Physical Security
- Dynamic network topology
- Frequent routing updates

2.3 Advantages of MANETs

The following are the advantages of MANETs:

- They provide access to information and services regardless of geographic position.
- These networks can be set up at any place and time.

2.4 Disadvantages of MANETs

Some of the disadvantages of MANETs are as follows:

- Limited resources and physical security.
- Intrinsic mutual trust vulnerable to attacks.
- Lack of authorization facilities.
- Volatile network topology makes it hard to detect malicious nodes.
- Security protocols for wired networks cannot work for ad hoc networks.

2.5 Applications of MANET

Some of the applications of MANETs are as follows:

- Military or police exercises.
- Disaster relief operations.
- Mine cite operations.
- Urgent Business meetings.

3.1 Routing in MANET

Mobile Ad-hoc networks are self-organizing and self-configuring multihop wireless networks, where the structure of the network changes dynamically. This is mainly due to the mobility of the nodes [3]. Nodes in these networks utilize the same random access wireless channel, cooperating in an intimate manner to engaging themselves in multihop forwarding. The node in the network not only acts as hosts but also as routers that route data to/from other nodes in network [6]. In mobile ad-hoc networks there is no infrastructure support as is the case with wireless networks, and since a destination node might be out of range of a source node transferring packets; so there is need of a routing procedure. This is always ready to find a path so as to forward the packets appropriately between the source and the destination. Within a cell, a base station can reach all mobile nodes without routing via broadcast in common wireless networks. In the case of ad-hoc networks, each node must be able to forward data for other nodes. This creates additional problems along with the problems of dynamic topology which is unpredictable connectivity changes [8].

3.2 Properties of Ad-Hoc Routing protocols

The properties that are desirable in Ad-Hoc Routing protocols are [4]:

i). Distributed operation: The protocol should be distributed. It should not be dependent on a centralized controlling node. This is the case even for stationary networks. The dissimilarity is that the nodes in an ad-hoc network can enter or leave the network very easily and because of mobility the network can be partitioned.

ii). Loop free: To improve the overall performance, the routing protocol should assurance that the routes supplied are loop free. This avoids any misuse of bandwidth or CPU consumption.

iii). Demand based operation: To minimize the control overhead in the network and thus not misuse the network resources the protocol should be reactive. This means that the protocol should react only when needed and should not periodically broadcast control information.

iv). Unidirectional link support: The radio environment can cause the formation of unidirectional links. Utilization of these links and not only the bi-directional links improves the routing protocol performance.

v). Security: The radio environment is especially vulnerable to impersonation attacks so to ensure the wanted behavior of the routing protocol we need some sort of security measures. Authentication and encryption is the way to go and problem here lies within distributing the keys among the nodes in the ad-hoc network.

vi). Power conservation: The nodes in the ad-hoc network can be laptops and thin clients such as PDA‗s that are limited in battery power and therefore uses some standby mode to save the power. It is therefore very important that the routing protocol has support for these sleep modes.

vii). Multiple routes: To reduce the number of reactions to topological changes and congestion multiple routes can be used. If one route becomes invalid, it is possible that another stored route could still be valid and thus saving the routing protocol from initiating another route discovery procedure.

viii). Quality of Service Support: Some sort of Quality of service is necessary to incorporate into the routing protocol. This helps to find what these networks will be used for. It could be for instance real time traffic support.

3.3 Problems in routing with Mobile Ad hoc Networks

i). Asymmetric links: Most of the wired networks rely on the symmetric links which are always fixed. But this is not a case with ad-hoc networks as the nodes are mobile and constantly changing their position within network

ii). Routing Overhead: In wireless ad hoc networks, nodes often change their location within network. So, some stale routes are generated in the routing table which leads to unnecessary routing overhead.

iii). Interference: This is the major problem with mobile ad-hoc networks as links come and go depending on the transmission characteristics, one transmission might interfere with another one and node might overhear transmissions of other nodes and can corrupt the total transmission.

iv). Dynamic Topology: Since the topology is not constant; so the mobile node might move or medium characteristics might change. In ad-hoc networks, routing tables must somehow reflect these changes in topology and routing algorithms have to be adapted. For example in a fixed network routing table updating takes place for every 30sec. This updating frequency might be very low for ad-hoc networks.

4.1 Types of MANETs Routing Protocols

Classification of routing protocols in mobile ad hoc network can be done in many ways, but most of these are done depending on routing strategy and network structure [2] [3] [10]. The routing protocols can be categorized as flat routing, hierarchical routing and geographic position assisted routing while depending on the network structure.
According to the routing strategy routing protocols can be classified as Table-driven and source initiated. The classification of routing protocols is shown in the Figure 2.

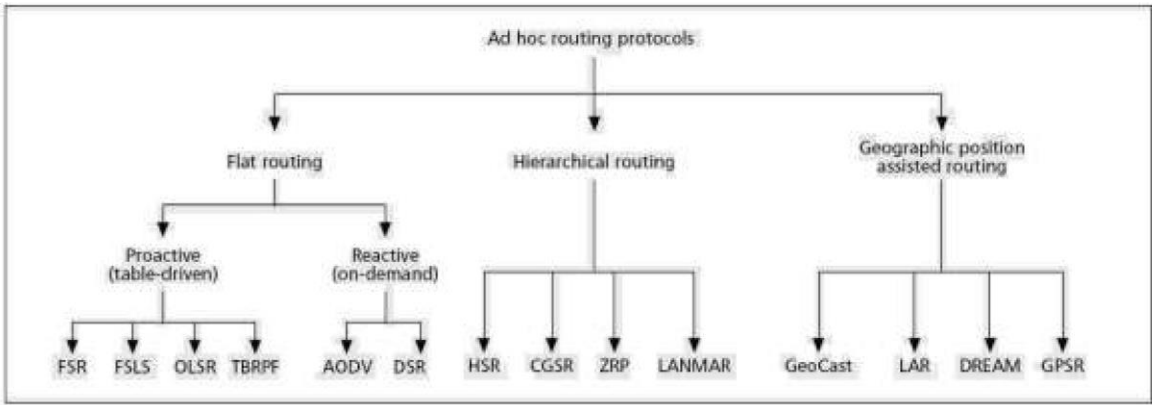


Figure 2: Classification of Routing Protocols in Mobile Ad-hoc Networks

4.2 Flat Routing Protocols

Flat routing [3] protocols are divided mainly into two classes; the first one is proactive routing (table driven) protocols and other is reactive (on-demand) routing protocols. One thing is general for both protocol classes is that every node participating in routing play an equal role. They have further been classified after their design principles; proactive routing is mostly based on LS (link-state) while on-demand routing is based on DV (distance-vector).

4.2.1 Pro-Active / Table Driven routing Protocols

Proactive MANETs protocols are also called as table-driven protocols and will actively determine the layout of the network. Through a regular exchange of network topology packets between the nodes of the network, at every single node an absolute picture of the network is maintained. There is hence minimal delay in determining the route to be taken.

This is especially important for time-critical traffic [3].

When the routing information becomes worthless quickly, there are many short-lived routes that are being determined and not used before they turn invalid. Therefore, another drawback resulting from the increased mobility is the amount of traffic overhead generated when evaluating these unnecessary routes. This is especially altered when the network size increases. The portion of the total control traffic that consists of actual practical data is further decreased.

Lastly, if the nodes transmit infrequently, most of the routing information is considered redundant. The nodes, however, continue to expend energy by continually updating these unused entries in their routing tables as mentioned, energy conservation is very important in a MANET system design. Therefore, this excessive expenditure of energy is not desired. Thus, proactive MANET protocols work best in networks that have low node mobility or where the nodes transmit data frequently. Examples of Proactive MANET Protocols include:

- Optimized Link State Routing (OLSR)
- Fish-eye State Routing (FSR)
- Destination-Sequenced Distance Vector (DSDV)
- Cluster-head Gateway Switch Routing Protocol (CGSR)

4.2.2 Reactive (On Demand) protocols

Portable nodes- Notebooks, palmtops or even mobile phones usually compose wireless ad-hoc networks. This portability also brings a significant issue of mobility. This is a key issue in ad-hoc networks. The mobility of the nodes causes the topology of the network to change constantly. Keeping track of this topology is not an easy task, and too many resources may be consumed in signaling. Reactive routing protocols were intended for these types of environments. These are based on the design that there is no point on trying to have an image of the entire network topology, since it will be constantly changing. Instead, whenever a node needs a route to a given target, it initiates a route discovery process on the fly, for discovering out a pathway [8].

Reactive protocols start to set up routes on-demand. The routing protocol will try to establish such a route, whenever any node wants to initiate communication with another node to which it has no route. This kind of protocols is usually based on flooding the network with Route Request (RREQ) and Route reply (RERP) messages .By the help of Route request message the route is discovered from source to target node; and as the target node gets a RREQ message it send RERP message for the confirmation that the route has been established. This kind of protocol is usually very effective on single-rate networks. It usually minimizes the number of hops of the selected path. However, on multi-rate networks, the number of hops is not as important as the throughput that can be obtained on a given path [15].

The different types of On Demand driven protocols are:

- Ad hoc On Demand Distance Vector (AODV)
- Dynamic Source routing protocol (DSR)
- Temporally ordered routing algorithm (TORA)
- Associativity based routing (ABR)
- Signal Stability-Based Adaptive Routing (SSA)
- Location-Aided Routing Protocol (LAR)

4.3 Hybrid Routing Protocols

Since proactive and reactive protocols each work best in oppositely different scenarios, hybrid method uses both. It is used to find a balance between both protocols. Proactive operations are restricted to small domain, whereas, reactive protocols are used for locating nodes outside those domains [8]. Examples of hybrid protocols are:

- Zone Routing Protocol, (ZRP)
- Wireless Ad hoc Routing Protocol, (WARP)

4.4 Hierarchical Routing Protocols

As the size of the wireless network increases, the flat routing protocols may produce too much overhead for the MANET. In this case a hierarchical solution may be preferable

[8].

- Hierarchical State Routing (HSR)
- Zone Routing Protocol (ZRP)
- Cluster-head Gateway Switch Routing Protocol (CGSR)
- Landmark Ad Hoc Routing Protocol (LANMAR)

4.5 Geographical Routing Protocols

There are two approaches to geographic mobile ad hoc networks:

1. Actual geographic coordinates (as obtained through GPS – the Global Positioning System).

2. Reference points in some fixed coordinate system.

An advantage of geographic routing protocols [8] is that they prevent network-wide searches for destinations. If the recent geographical coordinates are known then control and data packets can be sent in the general direction of the destination. This trim downs control overhead in the network. A disadvantage is that all nodes must have access to their geographical coordinates all the time to make the geographical routing protocols useful. The routing updates must be done faster in compare of the network mobility rate to consider the location-based routing effective. This is because locations of nodes may change quickly in a MANET. Examples of geographical routing protocols are:

- GeoCast (Geographic Addressing and Routing)
- DREAM (Distance Routing Effect Algorithm for Mobility)
- GPSR (Greedy Perimeter Stateless Routing)

5.1 Ad hoc On-demand Distance-Vector Routing (AODV)

Ad hoc On-Demand Distance Vector (AODV) routing is a routing protocol for mobile ad hoc networks and other wireless ad-hoc networks. It is jointly developed in Nokia Research Centre of University of California, Santa Barbara and University of Cincinnati by C. Perkins and S. Das. It is an on-demand and distance-vector routing protocol, meaning that a route is established by AODV from a destination only on demand [24]. AODV is capable of both unicast and multicast routing [17]. It keeps these routes as long as they are desirable by the sources. Additionally, AODV creates trees which connect multicast group members. The trees are composed of the group members and the nodes needed to connect the members. The sequence numbers are used by AODV to ensure the freshness of routes. It is loop-free, self-starting, and scales to large numbers of mobile nodes [17] [25]. AODV defines three types of control messages for route maintenance:

RREQ- A route request message is transmitted by a node requiring a route to a node. As an optimization AODV uses an expanding ring technique when flooding these messages. Every RREQ carries a time to live (TTL) value that states for how many hops this message should be forwarded. This value is set to a predefined value at the first transmission and increased at retransmissions. Retransmissions occur if no replies are received. Data packets waiting to be transmitted (i.e. the packets that initiated the RREQ). Every node maintains two separate counters: a node sequence number and a broadcast_ id. The RREQ contains the following fields [17]:-

source address	broadcast ID	source sequence no.	destination address	destination sequence no.	Hop count

The pair <source address, broadcast ID> uniquely identifies a RREQ. Broadcast id is incremented whenever the source issues a new RREQ [7].

RREP- A route reply message is unicasted back to the originator of a RREQ if the receiver is either the node using the requested address, or it has a valid route to the requested address. The reason one can unicast the message back, is that every route forwarding a RREQ caches a route back to the originator.

RERR- Nodes monitor the link status of next hops in active routes. When a link breakage in an active route is detected, a RERR message is used to notify other nodes of the loss of the link. In order to enable this reporting mechanism, each node keeps a ―precursor list", containing the IP address for each its neighbours that are likely to use it as a next hop towards each destination.

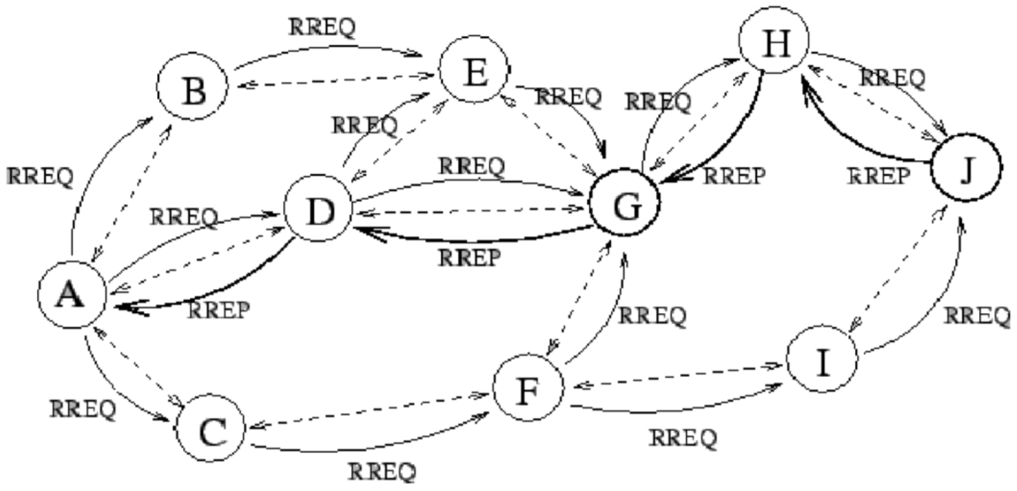


Figure 3: A possible path for a route replies if A wishes to find a route to J [9]

The above Figure3 illustrates an AODV route lookup session. Node A wants to initiate traffic to node J for which it has no route. A transmit of a RREQ has been done, which is flooded to all nodes in the network. When this request is forwarded to J from H, J generates a RREP. This RREP is then unicasted back to A using the cached entries in nodes H, G and D.

AODV builds routes using a route request/route reply query cycle. When a source node desires a route to a destination for which it does not already have a route, it broadcasts a route request (RREQ) packet across the network [18]. Nodes receiving this packet update their information for the source node and set up backwards pointers to the source node in the route tables. In addition to the source node's IP address, current sequence number, and broadcast ID, the RREQ also contains the most recent sequence number for the destination of which the source node is aware [7] [25]. A node getting the RREQ may send a route reply (RREP) if it is either the destination or if it has a route to the destination with corresponding sequence number greater than or equal to that contained in the RREQ. If this is the case, it unicasts a RREP back to the source. Otherwise, it rebroadcasts the RREQ. Nodes keep track of the RREQ's source IP address and broadcast ID. If they receive a RREQ which they have already processed, they discard the RREQ and do not forward it.

As the RREP propagates back to the source, nodes set up forward pointers to the destination [19]. Once the source node receives the RREP, it may begin to forward data packets to the destination. If the source later receives a RREP containing a greater sequence number or contains the same sequence number with a smaller hop count, it may update its routing information for that destination and begin using the better route.

As long as the route remains active, it will continue to be maintained. A route is considered active as long as there are data packets periodically travelling from the source to the destination along that path. Once the source stops sending data packets, the links will time out and eventually be deleted from the intermediate node routing tables. If a link break occurs while the route is active, the node upstream of the break propagates a route error (RERR) message to the source node to inform it of the now unreachable destinations. After receiving the RERR, if the source node still desires the route, it can reinitiate route discovery.

Multicast routes are set up in a similar manner. A node wishing to join a multicast group broadcasts a RREQ with the destination IP address set to that of the multicast group and with the 'J'(join) flag set to indicate that it would like to join the group. Any node receiving this RREQ that is a member of the multicast tree that has a fresh enough sequence number for the multicast group may send a RREP. As the RREPs propagate back to the source, the nodes forwarding the message set up pointers in their multicast route tables. As the source node receives the RREPs, it keeps track of the route with the freshest sequence number, and beyond that the smallest hop count to the next multicast group member. After the specified discovery period, the source nodes will unicast a Multicast Activation (MACT) [15] message to its selected next hop. This message serves the purpose of activating the route. A node that does not receive this message that had set up a multicast route pointer will timeout and delete the pointer. If the node receiving the MACT was not already a part of the multicast tree, it will also have been keeping track of the best route from the RREPs it received. Hence it must also unicast a MACT to its next hop, and so on until a node that was previously a member of the multicast tree is reached.
AODV maintains routes for as long as the route is active. This includes maintaining a multicast tree for the life of the multicast group. Because the network nodes are mobile, it is likely that many link breakages along a route will occur during the lifetime of that route [20].

The ―counting to infinity problem is avoided by AODV [9] from the classical distance vector algorithm by using sequence numbers for every route. The counting to infinity problem is the situation where nodes update each other in a loop.

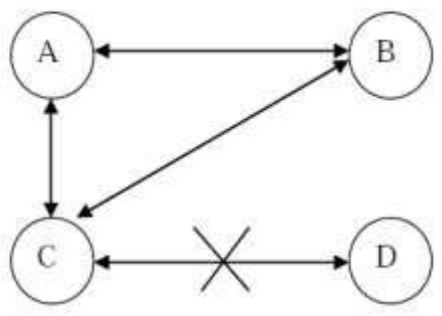


Figure 3.2 problem [9]

Figure 4: counting to infinity problem [9]

Consider nodes A, B, C and D making up a MANET as illustrated in Figure3.2. A is not updated on the fact that its route to D via C is broken. This means that A has a registered route, with a metric of 2, to D. C has registered that the link to D is down, so once node B is updated on the link breakage between C and D, it will calculate the shortest path to D to be via A using a metric of 3.C receives information that B can reach D in 3 hops and updates its metric to 4 hops. A then registers an update in hop-count for its route to D via C and updates the metric to 5. So they continue to increment the metric in a loop. The way this is avoided in AODV, for the example described, is by B noticing that as route to D is old based on a sequence number. B will then discard the route and C will be the node with the most recent routing information by which B will update its routing table.

5.1.1 Characteristics of AODV

- Unicast, Broadcast, and Multicast communication.
- On-demand route establishment with small delay.
- Multicast trees connecting group members maintained for lifetime of multicast group.
- Link breakages in active routes efficiently repaired.
- All routes are loop-free through use of sequence numbers.
- Use of Sequence numbers to track accuracy of information.
- Only keeps track of next hop for a route instead of the entire route.
- Use of periodic HELLO messages to track neighbours [21].

5.1.2 Advantages and Disadvantages

The main advantage of AODV protocol is that routes are established on demand and destination sequence numbers are used to find the latest route to the destination. The connection setup delay is less. The HELLO messages supporting the routes maintenance are range-limited, so they do not cause unnecessary overhead in the network. One of the disadvantages of this protocol is that intermediate nodes can lead to inconsistent routes if the source sequence number is very old and the intermediate nodes have a higher but not the latest destination sequence number, thereby having stale entries. Also multiple RouteReply packets in response to a single RouteRequest packet can lead to heavy control overhead [21]. Another disadvantage of AODV is that the periodic beaconing leads to unnecessary bandwidth consumption.

5.2 Dynamic Source Routing (DSR)

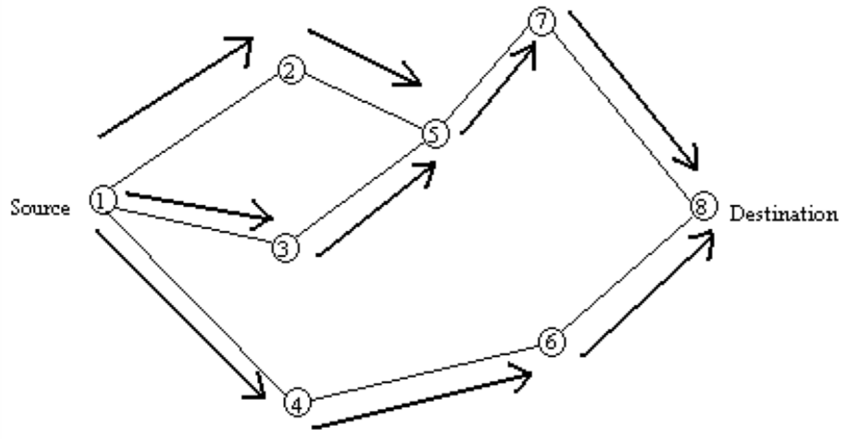
Dynamic Source Routing (DSR) is a routing protocol for wireless mesh networks. It is similar to AODV in that it establishes a route on-demand when a transmitting mobile node requests one. However, it uses source routing instead of relying on the routing table at each intermediate device [13].

Dynamic source routing protocol (DSR) is an on-demand, source routing protocol [27], whereby all the routing information is maintained (continually updated) at mobile nodes. DSR allows the network to be completely self-organizing and self-configuring, without the need for any existing network infrastructure or administration. The protocol is composed of the two main mechanisms of "Route Discovery" and "Route Maintenance", which work together to allow nodes to discover and maintain routes to arbitrary destinations in the ad hoc network [35].

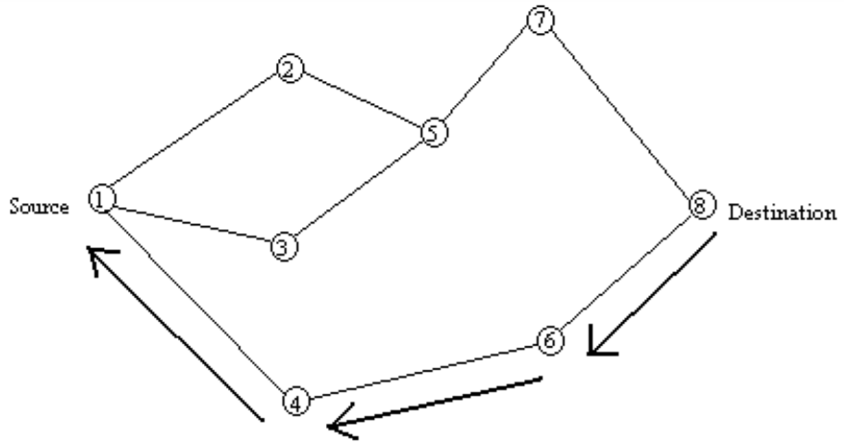
An optimum path for a communication between a source node and target node is determined by Route Discovery process. Route Maintenance ensures that the communication path remains optimum and loop-free according the change in network conditions, even if this requires altering the route during a transmission. Route Reply would only be generated if the message has reached the projected destination node (route record which is firstly contained in Route Request would be inserted into the Route Reply).

To return the Route Reply, the destination node must have a route to the source node. If the route is in the route cache of target node, the route would be used. Otherwise, the node will reverse the route based on the route record in the Route Reply message header (symmetric links). In the event of fatal transmission, the Route Maintenance Phase is initiated whereby the Route Error packets are generated at a node. The incorrect hop will be detached from the node's route cache; all routes containing the hop are reduced at that point. Again, the Route Discovery Phase is initiated to determine the most viable route.

The major dissimilarity between this and the other on-demand routing protocols is that it is beacon-less and hence it does not have need of periodic hello packet (beacon) transmissions, which are used by a node to inform its neighbours of its presence. The fundamental approach of this protocol during the route creation phase is to launch a route by flooding RouteRequest packets in the network. The destination node, on getting a RouteRequest packet, responds by transferring a RouteReply packet back to the source, which carries the route traversed by the RouteRequest packet received.



(a). Propagation of request (PREQ) packet



(b). Path taken by the Route Reply (RREP) packet

Figure 5: Creation of route in DSR [14]

A destination node, after receiving the first RouteRequest packet, replies to the source node through the reverse path the RouteRequest packet had traversed. Nodes can also be trained about the neighboring routes traversed by data packets if operated in the promiscuous mode. This route cache is also used during the route construction phase. If an intermediary node receiving a RouteRequest has a route to the destination node in its route cache, then it replies to the source node by sending a RouteReply with the entire route information from the source node to the destination node.

5.2.1 Advantages and Disadvantages

DSR uses a reactive approach which eliminates the need to periodically flood the network with table update messages which are required in a table-driven approach. The intermediate nodes also utilize the route cache information efficiently to reduce the control overhead.

The disadvantage of DSR is that the route maintenance mechanism does not locally repair a broken down link. The connection setup delay is higher than in table-driven protocols. Even though the protocol performs well in static and low-mobility environments, the performance degrades rapidly with increasing mobility. Also, considerable routing overhead is involved due to the source-routing mechanism employed in DSR. This routing overhead is directly proportional to the path length.

5.3 Destination Sequenced Distance Vector (DSDV)

DSDV is one of the most well known table-driven routing algorithms for MANETs. It is a distance vector protocol. In distance vector protocols, every node *i* maintains for each destination *x* a set of distances $\{dij(x)\}$ for each node *j* that is a neighbor of *i*. Node *i* treats neighbor *k* as a next hop for a packet destined to x if $dik(x)$ equals $minj\{dij(x)\}$. The succession of next hops chosen in this manner leads to *x* along the shortest path. In order to keep the distance estimates up to date, each node monitors the cost of its outgoing links and periodically broadcasts to all of its neighbors its current estimate of the shortest distance to every other node in the network. The distance vector which is periodically broadcasted contains one entry for each node in the network which includes the distance from the advertising node to the destination. The distance vector algorithm described above is a classical Distributed Bellman-Ford (DBF) algorithm [36][37]. DSDV is a distance vector algorithm which uses sequence numbers originated and updated by the destination, to avoid the looping problem caused by stale routing information. In DSDV, each node maintains a routing table which is constantly and periodically updated (not ondemand) and advertised to each of the node‘s current neighbors. Each entry in the routing table has the last known destination sequence number. Each node periodically transmits updates, and it does so immediately when significant new information is available. The data broadcasted by each node will contain its new sequence number and the following information for each new route: the destinations address the number of hops to reach the destination and the sequence number of the information received regarding that destination, as originally stamped by the destination. No assumptions about mobile hosts maintaining any sort of time synchronization or about the phase relationship of the update periods between the mobile nodes are made. Following the traditional distance-vector routing algorithms, these update packets contain information about which nodes are accessible from each node and the number of hops necessary to reach them. Routes with more recent sequence numbers are always the preferred basis for forwarding decisions. Of the paths with the same sequence number, those with the smallest metric (number of hops to the destination) will be used. The addresses stored in the route tables will correspond to the layer at which the DSDV protocol is operated. Operation at layer 3 will use network layer addresses for the next hop and destination addresses, and operation at layer 2 will use layer-2 MAC addresses [37].

5.3.1 Advantages and Disadvantages

DSDV was one of the early algorithms available. It is quite suitable for creating ad hoc networks with small number of nodes. Since no formal specification of this algorithm is present there is no commercial implementation of this algorithm. DSDV guarantees for loop free path. DSDV requires a regular update of its routing tables, which uses up battery power and a small amount of bandwidth even when the network is idle. Whenever the topology of the network changes, a new sequence number is necessary before the network re-converges; thus, DSDV is not suitable for highly dynamic networks. (As in all distance-vector protocols, this does not perturb traffic in regions of the network that are not concerned by the topology change.)

6.1 Congestion control in MANET

To maintain and allocate network resources effectively and fairly among a collection of users is a major issue. The resources shared mostly are the bandwidth of the links and the queues on the routers or switches. Packets are queued in these queues awaiting transmission. When too many packets are contending for the same link, the queue overflows and packets have to be dropped. When such drops become common events, the network is said to be congested [10].

In Ad-hoc networks, since there is no fixed infrastructure there are no separate network elements called routers and hence the mobile nodes themselves act as the routers (i.e. they are responsible for routing the packets). Congestion control methods [11] can be router centric or host/node centric. In existing congestion control methods, the source is informed about the congestion in the network so that either it may slow down the packet transmission rate or find an alternate route which may not necessarily be an optimal route. It must be pointed out that all the congestion control methods are able to inform the source about the congestion problem because they use Transmission Control Protocol (TCP).

6.1.1 Problem Identification

TCP/IP protocol was designed for wired networks which provides end to end reliable communication between nodes and assures ordered delivery of packets. It also provides flow control and error control mechanisms. As it is still a successful protocol in wired networks, losses are mainly due to congestion. But in case of ad hoc networks packet losses are due to congestion in the network and due to frequent link failures so when we adapt TCP to ad hoc networks it misinterprets the packet losses due to link failure as packet losses due to congestion and in the instance of a timeout, backing-off its retransmission timeout (RTO).This results in unnecessary reduction of transmission rate because of which throughput of the whole network degrades. Therefore, route changes due to host mobility can have a detrimental impact on TCP performance.

On-demand Routing Protocols such as AODV and DSR are used for this performance analysis of TCP. These types of routing protocols create routes only when requested by a source node. When a node wants to establish a route to a destination, it initiates a route discovery process within the network. One the route has been established, it is maintained until either destination becomes inaccessible or the route is no longer desired.

We will analyze the performance of two on-demand routing protocols for ad hoc networks, namely Dynamic Source Routing(DSR) [13] and Ad Hoc On-demand Distance Vector (AODV) [13] routing protocols, based on TCP traffic flows[14]. We also use DSDV which is a table driven routing protocol.

6.1.2 Related Works

TCP optimization in MANETs has been investigated in several studies. TCP does not have any resilience mechanisms that are specially designed to deal with link failures. From the viewpoint of TCP, there is no difference between link failure and network congestion. As a result, when part of the network fails and some segments are dropped, TCP will assume that there is congestion somewhere in the network, and the TCP congestion control mechanisms will start dealing with the segment loss. TCP congestion control mechanisms have improved over time. The main versions of TCP are Tahoe TCP, Reno TCP, NewReno TCP and SACK TCP. Tahoe TCP is the oldest version and only a few old systems use it. NewReno TCP and SACK TCP are widely implemented. We focus on another two new TCP variants TCP FACK and TCP vegas because they are the newer versions and are being deployed. And also the analysis was not on just one routing protocol but three widely used routing protocols, AODV, DSR and DSDV

6.2 Congestion Control Mechanisms

The predominant example of end-to-end congestion control [11] in use today, that implemented by TCP. The essential strategy of TCP is to send packets into the network without a reservation and then to react to observable events that occur. TCP assumes only

FIFO queuing in the network_s routers, but also works with fair queuing.

6.2.1 Additive Increase/Multiplicative Decrease

TCP maintains a new state variable for each connection, called Congestion Window [3], which is used by the source to limit how much data it is allowed to have in transit at a given time. The congestion window is congestion control's counterpart to flow control's advertised window. TCP is modified such that the maximum number of bytes of unacknowledged data allowed is now the minimum of the congestion window and the advertised window.

MaxWindow = MIN (CongestionWindow, AdvertisedWindow)

EffectiveWindow = MaxWindow - (LastByteSent - LastByteAcked).

That is, MaxWindow replaces AdvertisedWindow in the calculation of EffectiveWindow. Thus, a TCP source is allowed to send no faster than the slowest component—the network or the destination host—can accommodate. The problem, of course, is how TCP comes to learn an appropriate value for CongestionWindow. Unlike the

AdvertisedWindow, sent by receiving side of the connection, there is no one to send a suitable CongestionWindow to the sending side of TCP.

TCP does not wait for an entire window_s worth of ACKs to add one packet_s worth to the congestion window, but instead increments CongestionWindow by a little for each ACK that arrives. Specifically, the congestion window is incremented as follows each time an ACK arrives:

Increment = MSS × (MSS/CongestionWindow)
CongestionWindow + = Increment

That is, rather than incrementing CongestionWindow by an entire MSS bytes each RTT, we increment it by a fraction of MSS every time an ACK is received. The important concept to understand about AIMD is that the source is willing to reduce its congestion window at a much faster rate than it is willing to increase its congestion window. [12]

6.2.2 Fast Retransmit and Fast Recovery

The mechanisms described so far were part of the original proposal to add congestion control to TCP. It was soon discovered, however, that the coarse-grained implementation of TCP timeouts led to long periods of time during which the connection went dead while waiting for a timer to expire. Because of this, a new mechanism called fast re- transmit was added to TCP. Fast retransmit is a heuristic that sometimes triggers the retransmission of a dropped packet sooner than the regular timeout mechanism.

6.3 Variants of TCP

Transmission Control Protocol (TCP)[38][41] is the predominant Internet protocol and it carries approximately 90% of Internet traffic in today‘s heterogeneous wireless and wired networks. TCP is reliable end to end protocol because TCP is trying to provide reliable data transmission between two entities. TCP is widely used as a connection oriented transport layer protocol that provides reliable data packet delivery over unreliable links. TCP primary purpose is to provide a connection oriented reliable data transfer service between different applications to be able to provide these services on top of an unreliable communication system. TCP needs to consider data transfer, reliability flow control, multiplexing, TCP segment, and congestion control and connection management. TCP does not depend on the underlying network layers and, hence, design of various TCP variants is based on the properties of wired networks. However, TCP congestion control algorithms may not perform well in heterogeneous networks. The TCP protocol has been extensively tuned to give good performance at the transport layer in the traditional wired network environment. However, TCP in its present form is not well suited for mobile ad hoc networks (MANETs) where packet loss due to broken routes can result in the counterproductive invocation of TCP‘s congestion control mechanisms.

Although a number of studies have been conducted and protocol modifications suggested. The reason behind the variations of TCP is that each type possesses some special criteria, such as the traditional TCP has become known as TCP Tahoe. TCP Reno adds one new mechanism called Fast Recovery to TCP Tahoe [41]. TCP New Reno uses the newest retransmission mechanism of TCP Reno [8]. The uses of TCP Sack permits the receiver to specify several additional data packets that have been received out-of-order within one dup ACK, instead of only the last in order packet received [40]. TCP Vegas proposes its own new unique retransmission and congestion control strategies. TCP FACK is Reno TCP with forward acknowledgement [39].

6.3.1 TCP RENO

This RENO retains the basic principle of Tahoe, such as slow starts and the coarse grain retransmit timer. However it adds some intelligence over it so that lost packets are detected earlier and the pipeline is not emptied every time a packet is lost. Reno requires that we receive immediate acknowledgement whenever a segment is received. The logic behind this is that whenever we receive a duplicate acknowledgment, then his duplicate acknowledgment could have been received if the next segment in sequence expected, has been delayed in the network and the segments reached there out of order or else that the packet is lost. If we receive a number of duplicate acknowledgements then that means that sufficient time have passed and even if the segment had taken a longer path, it should have gotten to the receiver by now. There is a very high probability that it was lost. So

Reno suggests an algorithm called _Fast Re-Transmit'. Whenever we receive 3 duplicate ACK_s we take it as a sign that the segment was lost, so we re-transmit the segment without waiting for timeout. Thus we manage to re-transmit the segment with the pipe almost full. Another modification that RENO makes is in that after a packet loss, it does not reduce the congestion window to 1. Since this empties the pipe. It enters into an algorithm which we call _Fast-Re-Transmit'.

6.3.1.1 Disadvantages

RENO performs very well over TCP when the packet losses are small. But when we have multiple packet losses in one window then RENO doesn_t perform too well and its performance is almost the same as Tahoe under conditions of high packet loss. Another problem is that if the window is very small when the loss occurs then we would never receive enough duplicate acknowledgements for a fast retransmit and we would have to wait for a coarse grained timeout. Thus is cannot effectively detect multiple packet losses.

6.3.2 New RENO

New RENO is a slight modification over TCP-RENO. It is able to detect multiple packet losses and thus is much more efficient that RENO in the event of multiple packet losses. Like RENO, New-RENO also enters into fast-retransmit when it receives multiple duplicate packets, however it differs from RENO in that it doesn_t exit fast-recovery until all the data which was out standing at the time it entered fast recovery is acknowledged. The fast-recovery phase proceeds as in Reno, however when a fresh ACK is received then there are two cases:

- If it ACK_s all the segments which were outstanding when we entered fast recovery then it exits fast recovery and sets CWD to threshold value and continues congestion avoidance like Tahoe.
- If the ACK is a partial ACK then it deduces that the next segment in line was lost and it re-transmits that segment and sets the number of duplicate ACKS received to zero. It exits Fast recovery when all the data in the window is acknowledged

6.3.2.1 Disadvantages

New-Reno suffers from the fact that it takes one RTT to detect each packet loss. When the ACK for the first retransmitted segment is received only then can we deduce which other segment was lost.

6.3.3 TCP SACK

TCP with =Selective Acknowledgments_ is an extension of TCP RENO and it works around the problems face by TCP RENO and TCP New-RENO, namely detection of multiple lost packets, and re-transmission of more than one lost packet per RTT. SACK retains the slow-start and fast retransmits parts of RENO. It also has the coarse grained timeout of Tahoe to fall back on, in case a packet loss is not detected by the modified algorithm. SACK TCP requires that segments not be acknowledged cumulatively but should be acknowledged selectively. If there are no such segments outstanding then it sends a new packet**.** Thus more than one lost segment can be sent in one RTT.

6.3.3.1Disadvantages

The biggest problem with SACK is that currently selective acknowledgements are not provided by the receiver to implement SACK we_ll need to implement selective acknowledgment which is not a very easy task.

6.3.4 TCP FACK

FACK or Forward Acknowledgement is a special algorithm that works on top of the SACK options, and is geared at congestion controlling. FACK algorithm uses information provided by SACK to add more precise control to the injection of data into the network during recovery – this is achieved by explicitly measuring the total number of bytes of data outstanding in the network. FACK decouples congestion control from data recovery thereby attaining more precise control over the data flow in the network. The main idea of FACK algorithm is to consider the most forward selective acknowledgement sequence number as a sign that all the previous un-(selectively)acknowledged segments were lost. This observation allows improving recovery of losses significantly.

6.3.4.1 Disadvantages

TCP FACK provides congestion avoidance and fast retransmission mechanism, it faces so many situations for recovery and it cannot implement easily.

6.3.5 TCP VEGAS

Vegas is a TCP implementation which is a modification of RENO. It builds on the fact that proactive measure to encounter congestion is much more efficient than reactive ones. It tried to get around the problem of coarse grain timeouts by suggesting an algorithm which checks for timeouts at a very efficient schedule.

Also it overcomes the problem of requiring enough duplicate acknowledgements to detect a packet loss, and it also suggests a modified slow start algorithm which prevents it from congesting the network. The three major changes induced by Vegas are:

- **New Re-Transmission Mechanism**: Vegas extend on the retransmission mechanism of RENO. It keeps track of when each segment was sent and it also calculates an estimate of the RTT by keeping track of how long it takes for the acknowledgment to get back.
- **Congestion avoidance:** TCP Vegas is different from all the other implementation in its behavior during congestion avoidance. It does not use the loss of segment to signal that there is congestion. It determines congestion by a decrease in sending rate as compared to the expected rate, as result of large queues building up in the routers. It uses a variation of Wang and crow croft_s Tri-S scheme.
- **Modified Slow-start:** TCP Vegas differs from the other algorithms during its slow-start phase. The reason for this modification is that when a connection first starts it has no idea of the available bandwidth and it is possible that during exponential increase it over shoots the bandwidth by a big amount and thus induces congestion. To this end Vegas increases exponentially only every other RTT, between that it calculates the actual sending through put to the expected and when the difference goes above a certain threshold it exits slow start and enters the congestion avoidance phase.

6.3.5.1 Disadvantages

If there are enough buffer in the routers it means that Vegas congestion avoidance mechanism can function effectively a higher throughput and a faster response time result. As the load increase or the number or router buffer decrease, Vegas congestion avoidance mechanism is not as effective and Vegas start to behave more like Reno. Vegas is less aggressive in its use of router buffer than Reno because Vegas is limited. Finally Vegas congestion detection algorithm depends on the accurate value for Base RTT.

6.4 COMPARISION OF TCP VARIANTS

TCP Reno: More than half of the coarse grained timeouts of Reno are prevented by Vegas as it detects and re-transmits more than one lost packet before timeout occurs. It doesn‘t have to always wait for 3 duplicate packets so it can retransmit sooner. It doesn‘t reduce the congestion window too much prematurely. The advantages that it has in congestion avoidance and bandwidth utilization over Tahoe exist here as well. It can suffer from performance problems when multiple packets are dropped from a window of data.

TCP New-Reno: It prevents many of the coarse grained timeouts of New-Reno as it doesn‘t need to wait for 3duplicate ACK‘s before it retransmits a lost packet. Its congestion avoidance mechanisms to detect _incipient‘ congestion are very efficient and utilize network resources much more efficiently. Because of its modified congestion avoidance and slow start algorithm there are fewer retransmits.

TCP Sack: TCP Vegas doesn‘t have a clear cut advantage over SACK TCP. The only field where it appears to outperform SACK is: In its estimation of incipient congestion, and its efficient estimation of congestion by measuring change in throughput rather than packet loss. This would result in a better utilization of bandwidth and lesser congestion. Also it appears more stable than SACK, The reason for this being that SACK uses packet losses to denote congestion. So that the sender continually increase sending rate until there is congestion. This cycle continues and the system keeps on oscillating**.** Another advantage of TCP Vegas or rather the disadvantage of SACK is that it is not very easy to incorporate SACK in the current TCP.

TCP FACK: TCP FACK provides a better way to detect the congestion in network rather than TCP West Wood. TCP FACK performs first halved to estimate the correct congestion window that should be further decreased. RTT can be avoided when the window is gradually decreased. TCP FACK introduces a better way to halve the window when congestion is detected.

TCP Vegas: TCP Vegas provides fast recovery mechanism to overcome packet loss and network congestion.

An improved congestion avoidance mechanism that controls buffer occupies. To solve the problem of coarse gain timeout, TCP Vegas uses a modified retransmission strategies, which is based on the Fire-Gained measurements of RTT.

N- Normal, E V-Enhanced Version, N M- New Mechanism

Variants	TCP Reno	TCP New Reno	TCP SACK	TCP FACK	TCP Vegas
Slow Start	Yes	Yes	Yes		E V
Congestion Avoidance	Yes	Yes	Yes	Yes	E V
Fast Retransmit	Yes	Yes	Yes	Yes	Yes
Fast Recovery	Yes	E V	E V	E V	Yes
Retransmission	N	N	N	N	N M
Congestion Control	N	N	N	N M	N M
Selective ACK	No	No	Yes	Yes	No

Table-1: Comparison of TCP variants

7.1 Simulation Tool

The simulation study is done by using widely recognized and improved network simulator NS-2 version 2.29.3 for Mobile Ad-hoc Networks (MANETs). NS-2 is powerful for simulating ad-hoc networks. In NS-2 the user has to imagine of a scenario, the number of nodes to be placed in the scenario, and then write the TCL scripts (*.tcl* file) specifying the node configurations parameters and some other *ns* commands required to start and stop *ns*. The user has also to create the movement and connection files that together represent the scenario. The output of the simulation is a trace file (.tr), which is logged with each and every event that took place during the simulation. This file can than be used for obtaining measures such as mobility, throughput, end-to-end delay, and packet loss measurement. An optional output is the NAM [15] supported file (.nam) that logs the necessary events to help visualize the scenario using the NAM. The NAM is a post simulation process that shows how the nodes moved and how they were connected during the simulation. Another optional output is xgraph [15], which shows a graphical output for a specific measurement. The AODV, DSR and DSDV protocols are also provided as part of the NS-2 installation. The TCP congestion control techniques were implemented by editing patch files in the NS2 codes.

7.2 Simulation Environment

The setdest tool in ns2 is used to generate the random topologies for the simulations. All simulations are performed for a 1000m × 1000m grid consisting of 50nodes, distributed randomly over the two-dimensional grid. The source destination pairs are randomly chosen from the set of 50 nodes in the network. We consider seven different speeds of 10m/s, 20m/s, 30m/s, 40m/s in our simulations all with pause time of 0. Two runs were conducted for each of the average speeds and we used, resulting different node density of 40, 60, 80 and 100 nodes resulting 128 different movement patterns. Pause time 0 means each node moves constantly throughout the simulation. TCP packet size of 1460 is considered in our analysis. The TCP clock granularity is set to 200ms. The queue sizes are set to 50 packets to avoid frequent drop of packets due to buffering. We measured the TCP throughput for each setup when operating over a wireless system.

8.1 Performance Evaluation

For different node density in a fixed area, we performed a series of three simulation runs. Each simulation run tested a different technique: TCP Newreno, TCP SACK, TCP FACK and TCP Vegas. In each run a set of performance measurements were made for each of the two routing protocols at each of several node densities from 40 nodes to 100 nodes.

Table 2: TCP Newreno for various node densities and different routing Protocols

Routing Protocols	**No. of Nodes**			
	40	60	80	100
	Through put (Kbps)			

AODV	160	240	260	300
DSR	160	310	140	260
DSDV	150	180	160	240

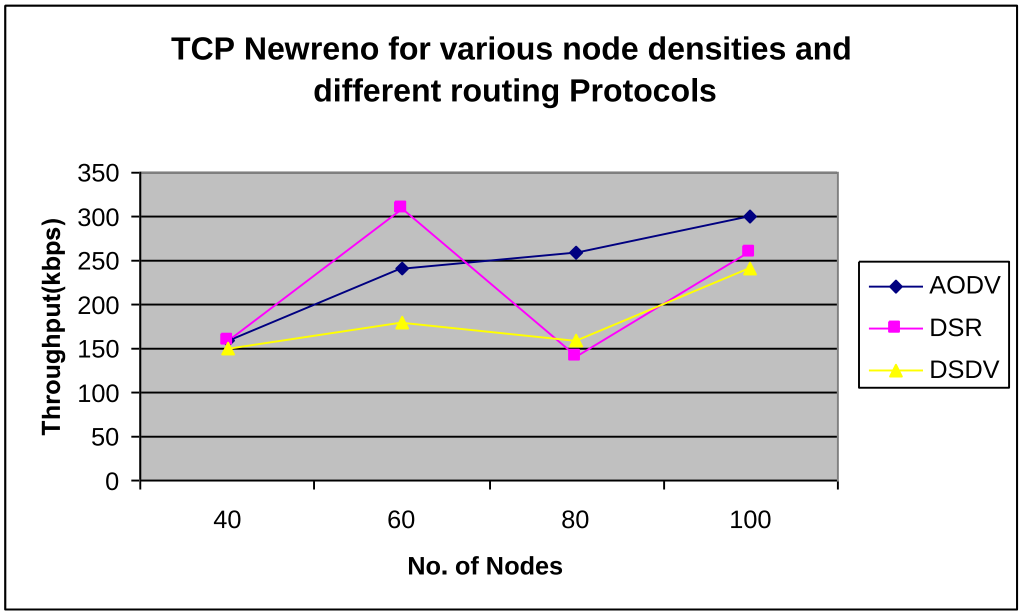


Figure No. 6: TCP Newreno for various node densities and different routing Protocols

8.2 Evaluation of TCP Newreno

Figure no. 6 represents TCP Newreno for all the measurement of throughput. When 20 nodes are used in the grid, DSR routing protocol performs very well then the other two protocols i.e. AODV and DSDV. AODV and DSDV almost give same throughput. However, when the nodes increase in the grid throughput also changes. Now when we consider the other set of measurement we notice that the throughput of the network gradually increases on using AODV routing protocol and also for DSDV but for DSR routing protocol the throughput is continuously varying at different node densities and it provides high throughput when we use 60nodes in the grid. On an average DSR routing protocol provides a better throughput compared to the other routing protocols.

8.3 Evaluation of TCP SACK

Table 3: TCP SACK for various node densities and different routing Protocols

Routing Protocols	No. of Nodes			
	40	60	80	100
	Through put (Kbps)			
AODV	240	380	420	320
DSR	250	340	380	300
DSDV	340	360	400	350

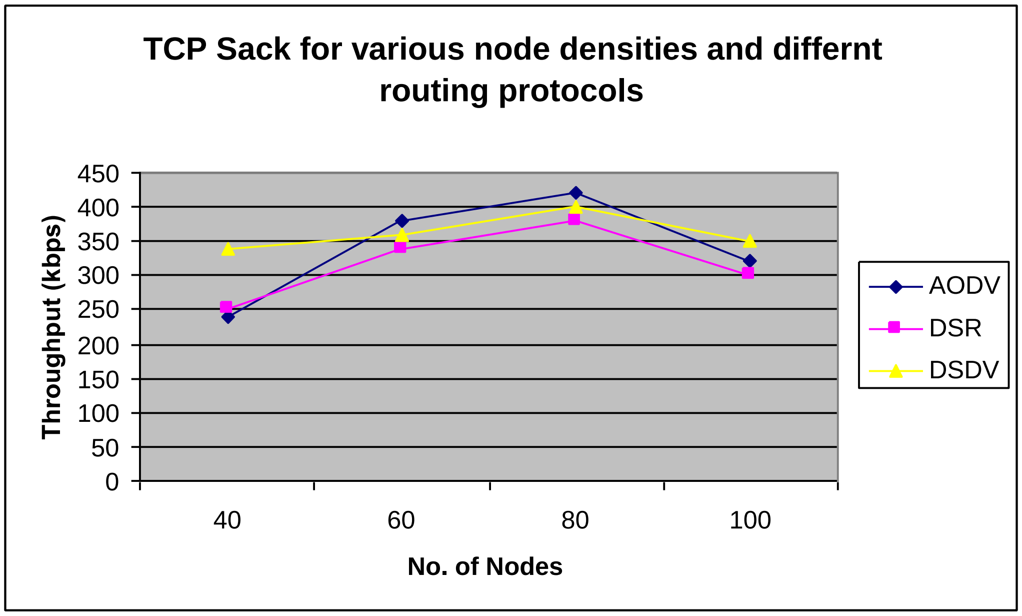


Figure No. 7: TCP SACK for various node densities and different routing Protocols

Figure no. 7 represents TCP SACK for all the measurement of throughput. TCP SACK gives better throughput than other variants in most of the scenarios. This is because it avoids frequent retransmission of packets by sending selective acknowledgements. This mechanism is better than the mechanisms used in TCP New RENO where in multiple packet losses lead to frequent retransmission of packets. But we also see that performance decreases as the nodes increases.

8.4 Evaluation of TCP FACK

Table 4: TCP FACK for various node densities and different routing Protocols

Routing Protocols	No. of Nodes			
	40	60	80	100
	Throughput (Kbps)			
AODV	310	480	430	400
DSR	300	460	480	340
DSDV	320	420	480	390

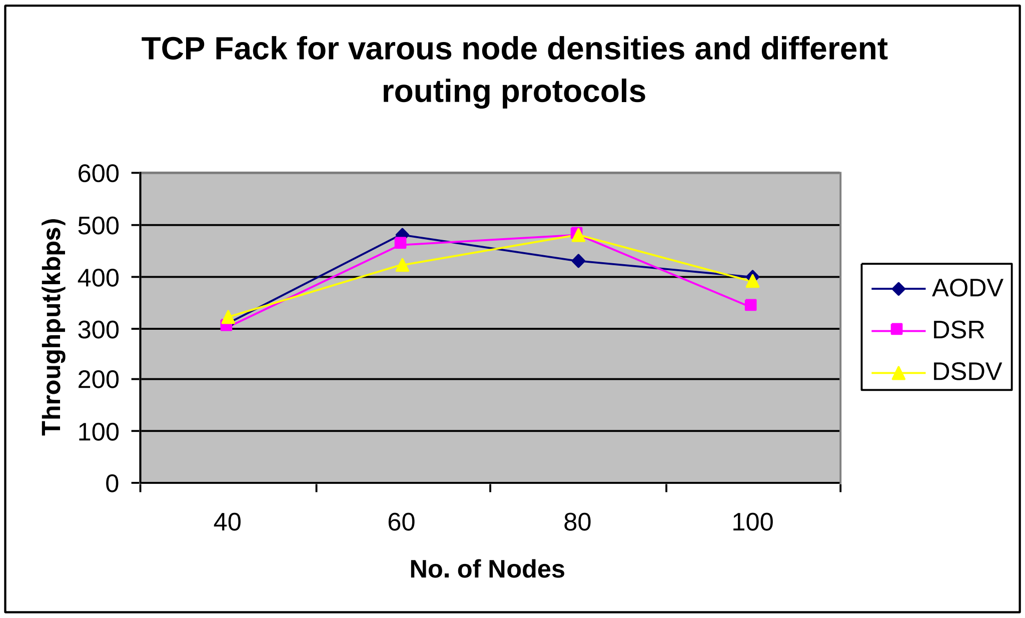


Figure No. 8: TCP FACK for various node densities and different routing Protocols

Figure no. 8 represents TCP FACK shows a similar behavior as TCP Sack. But it has a higher throughput than TCP SACK. This is because TCP FACK uses information provided by SACK to add more precise control to the injection of data into the network during recovery. This improves the recovery of losses significantly.

8.5 Evaluation of TCP VAGAS

Table 5: TCP VAGAS for various node densities and different routing Protocols

Routing Protocols	**No. of Nodes**			
	40	60	80	100
	Through put (Kbps)			
AODV	320	390	430	450
DSR	350	420	380	420
DSDV	320	350	380	400

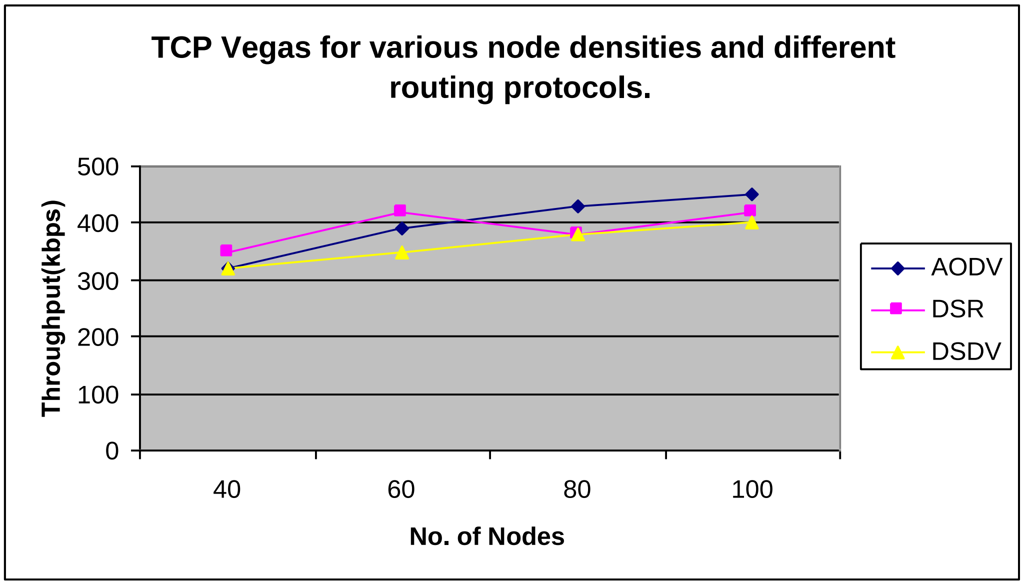


Figure No. 9: TCP VAGAS for various node densities and different routing Protocols

Figure no. 9 represents TCP VAGAS even though having a lower through than the other variants, shows a stable increase when the nodes increases. Where else the other variants the throughput decreases. When we increase the number of connections in a network (keeping number of nodes fixed) more packets are dropped in the network due to collision. TCP Vegas has a proactive behavior that prevents the packets being dropped in the network. Due to this nature it restricts the amount of data that it transmits in the network. Thus TCP Vegas achieves low throughput as compared to other variants.

9.1 Conclusion

We calculated the performance of four TCP variants; they are TCP New RENO, TCP SACK, TCP FACK and TCP Vegas for three different routing protocols. After analyzing the performance from simulated data and graphs obtained, we found that TCP Vegas is better than any other TCP variants for sending data and information due to its better Packet Delivery Fraction and Avg. end- to- end delay. TCP Vegas also showed a consistent performance for all three routing protocols as an average. Where as the others had varied response to the different routing protocols specially DSR showed a drop in throughput as nodes number increased. DSR is not good for high mobility and high number of nodes.   This is due to fine tuning of congestion window size by taking into consideration the RTT of a packet, whereas other reactive protocols like TCP New RENO, SACK, and FACK continue to increase their window size until packet loss is detected. We hope these results will be of some use in future study in this area helping the growing interest and resulting in the required protocol for today_s high demanding world.

9.2 Future Field

The future work of this project can be done in following areas:

1) The performance analysis of variants of TCP under other routing protocols like DYMO, DSR, OLSR.
2) Expanding the range of analysis by considering other new TCP_s like HS-TCP, TCP WESTWOOD etc

References

[1] Mobile Ad-hoc Networks (MANET), http://www.ietf.org/html.charters/manetcharter. html. (1998-11-29).

[2] Hadi Sargolzaey, Ayyoub Akbari Moghanjoughi and Sabira Khatun, ―A Review and Comparison of Reliable Unicast Routing Protocols For Mobile Ad Hoc Networks‖ , IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.1, pp. 186-196, January 2009.

[3] Krishna Gorantala, ―Routing in Mobile Ad-hoc Networks‖ , Umea
University,Sweden, June-2006.

[4] Geetha Jayakumar and Gopinath Ganapathy, ―Performance Comparison of Mobile Ad-hoc Network Routing Protocol‖ , International Journal of Computer

Science and Network Security (IJCSNS), VOL.7 No.11, pp. 77-84 November 2007.

[5] Azizol Abdullah, Norlida Ramly, Abdullah Muhammed, Mohd Noor Derahman,
―Performance Comparison Study of Routing Protocols for Mobile Grid Environment‖ ,
IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.2,
pp.82-88, February 2008.

[6] Laura Marie Feeney, ―A taxonomy for routing protocols in mobile ad hoc networks‖ , Technical report, Swedish Institute of Computer Science, Sweden, 1999.

[7] C.E. Perkins and E.M. Royer, ―Ad hoc On Demand Distance Vector Routing‖ , University of California, Santa Barbara.

[8] Elizabeth M. Royer and Chai-Keong Toh, ―A review of current routing protocols for ad hoc mobile wireless networks‖ , Technical report, University of California and Georgia Institute of Technology, USA, 1999.

[9] Ad-hoc distance vector protocol for mobile ad hoc network http://www.olsr.org/docs/report_html/node16.html#adhoc_aodv#adhoc_aodv – accessed on may 2009.

[10] Xiaoqin Chen, Haley M. Jones, A.D.S Jayalath, ―Congestion Aware Routing Protocol for Mobile Ad-hoc Networks‖, Department of Information Engineering, National University, Canberra.

[11] Raju Kumar, Riccardo Crepaldi, Hosam Rowaihy, Albert F. Harris III, Guohong Cao, Michele Zorzi, Thomas F. La Porta, ―Mitigating Performance Degradation in Congested Sensor Networks.‖, IEEE Transactions on Mobile Computing, Vol. 7, No. 6, June 2008.

[12] Yuvaraju B N, Dr. Niranjan N Chiplunkar, ―Scenario Based Performance Analysis of Variants of TCP‖, International Journal of Computer Applications (0975 – 8887), Volume 4– No.9, August 2010.

[13] C.Perkins, E.Royer and S Das, ―Ad-hoc on demand distance vector(aodv) routing,‖ in IETF Internet Draft(work in progress), Nov.2000.

[14] Sung Jae Jung Mun Gi Kim Byung Ho Rhee, ―Performance Enhancement Technique for TCP over Mobile Ad Hoc Networks‖, IEEE Hybrid Information Technology, pp.210-215, 2006.

[15] E.M. Royer and Charles E. Perkins, ―Multicast Operation of the Ad-hoc On- Demand Distance Vector Routing Algorithm|| , University of California, Santa Barbara.

[16] Rohit Dube, Cynthia D.Rais, Kuang-Yeh Wang, and Satish K.Tripathi, ―Signal Stability-based adaptive routing (SSA) for ad hoc mobile networks, IEEE Personal Communications, pp. 36-45, Feb. 1997.

[17] C.E. Perkins and E.M. Royer, ―Ad-hoc On-Demand Distance Vector Routing‖, Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, pp. 90-100, February 1999.

[18] P. Johansson,T. Larsson, N. Hedman, B. Mielczarek, M. Degermark,
―Scenariobased
Performance Analysis of Routing Protocols for Mobile Ad-hoc Networks‖,
Mobicom ‗99 Scattlc Washington USA, pp. 195-206

[20] Charles E. Perkins, ―Ad Hoc On Demand Distance Vector (AODV) Routing‖.
Internet draft, draft-ietf-manetaodv-02.txt, November 1998.

[21] Luke Klein-Bernd, ―A Quick Guide to AODV Routing‖, National Institute of Standards and Technology, US.

[22] Y.-B. Ko and N.H. Vaidya, Location-aided routing (LAR) in mobile ad hoc networks, in: Proc. of Mobicom‗98 (1998).

[23] C-K. Toh, ―A Novel Distributed Routing Protocol To Support Ad-Hoc Mobile
Computing‖, Proc. 1996 IEEE 15th Annual Int‗l. Phoenix Conf.Comp. and Communication., pp. 480–86, Mar. 1996.

[24] C.E Perkins, E.M. Royer, and S. Das, ―Ad hoc On-demand Distance Vector (AODV),‖ RFC 3561, July 2005.

[25] Abdul Hadi Abd Rahman and Zuriati Ahmad Zukarnain, ―Performance
Comparison of AODV, DSDV and I-DSDV Routing Protocols in Mobile Ad
Hoc Networks‖, European Journal of Scientific Research, ISSN 1450-216X Vol.31 No.4, pp.566-576, 2009.

[26] Josh Broch, David B. Johnsson, David A. Maltz, ―The Dynamic Source Routing
Protocol for Mobile Ad Hoc Networks‖ . Draft, draft-ietf-manet-dsr-00.txt,
October 1999. Work in progress

[27] D. B. Johnson and D. A. Maltz, ―Dynamic Source Routing in Ad Hoc Wireless
Networks, Mobile Computing‖ , Chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996,

[35] http://www.networksorcery.com/enp/protocol/dsr.htm-- Dynamic source routing protocol (DSR).

[36] N Vetrivelan, Dr. A V Reddy ―Performance Analysis of Three Routing Protocols for
Varying MANET Size‖ Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol II IMECS 2008, 19-21 March, 2008, Hong Kong.

[37] S.Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, ―A distance routing effect algorithm for mobility (dream),‖ in Proceedings of the IEEE/ACM international Conference on Mobile Computing and

Networking (MOBICOM'98), 1998, pp. 76–84

[38] J. Postel, ―Transmission Control Protocol‖, RFC 793, Sep 1981.

[39] Renaud Bruyeron, Bruno Hemon, Lixia Zhang: ―Experimentations with TCP
Selective Acknowledgment‖, ACM SIGCOMM
Computer Communication Review, April 1988

[40] K. Fall, S. Floyd **"**Simulation Based Comparison of Tahoe, Reno and SACK TCP**",** 1998**.**

[41] M. Mathis, J. Mahdavi, S. Floyd, A. Roma now.RFC 2018: TCP Selective acknowledgment options, October 1996.