Multi-Retransmission Route Discovery Schemes for Ad Hoc Wireless Network with a Realistic Physical Layer

by

Xiangyang Jin

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the Master degree of Computer Science (MCS)

School of Information Technology and Engineering
University of Ottawa

© Xiangyang Jin, Ottawa, Canada, 2011

Table of Contents

List of Abbreviations

ABR	Associativity Based Routing
AODV	Ad-hoc On-demand Distance Vector Routing
BRP	Bordercase Resolution Protocol
CGSR	Cluster Gateway Switching Routing
DAG	Directed Acyclic Graph
DSDV	Destination-Sequenced Distance-Vector
DSR	Dynamic Source Routing
EHC	Expected Hop Count
FSR	Fisheye State Routing
IARP	Intra-zone Routing Protocol
IERP	Inter-zone Routing Protocol
LCC	Least Cluster Change

MAC	Media Access Control
MANET	Mobile Ad-hoc Network
MRL	Message Retransmission List
NPDU	Network Protocol Data Units
NS2	Network Simulator
RReq	Route Request Packet
RT	Routing Table
RRep	Route Reply Packet
SSA	Signal Stability-based Adaptive Routing
SST	Signal Stability Table
TORA	Temporally Ordered Routing Algorithm
WRP	Wireless Routing Protocol
ZRP	Zone Routing Protocol

List of Figures and Charts

Abstract

In reactive routing protocols, nodes follow the flooding based route discovery procedure to find the route to the destination. During the route discovery process, each node receiving the route request packet (RReq) will retransmit RReq exactly once. A distant neighbor may accidentally receive/loose the only RReq and use it to announce a new route, although that link is inferior/superior for route reply packet (RRep) or actual message routing. Overall, the constructed route may be far from the optimal. All existing route discovery schemes (including DSR/AODV) apply retransmission during route discovery exactly once (1R). Based on a realistic physical layer model, we propose two new route discovery schemes: *n-retransmission* (*n*R, retransmitting exactly *n* times) and *n-retransmission c-reception* (*nc*RR), retransmitting until either a total of *n* own retransmissions or *c* copies from neighbors are heard. We compare our two new scheme with traditional one, under otherwise same conditions (same metric, same packet reception probability on each link) and same choices about possibly retransmitting again upon discovering better route (R+) or discarding it (R1), generating route reply packet for every received RRep (B*), or for first and better discovered route (B+), and retransmitting RRep exactly once (A1), maximum three times (A3), or optimally *u* times decided by link quality (A*u*). Experimental results

also show that the proposed *nc*RR scheme achieves the best tradeoff between quality of route, success rate and message overhead in route discovery process, followed by *n*R scheme, and both of them are superior to the existing traditional based schemes. Based on simulation results, the optimal number of retransmissions should be $n=2$ and the counter threshold for the received RReq copies should be $c=3$ or $c=4$.

Chapter 1 Introduction

A mobile ad hoc network (MANET) is an autonomous system of mobile routers (wireless devices such as portable or handheld computers, PDAs, and cell phones) connected by wireless links – the union of which forms an arbitrary graph. The routers are free to move randomly and organize themselves arbitrarily; thus, the network's wireless topology may change rapidly and unpredictably. Such a network may operate in a standalone fashion, or may be connected to the larger Internet [1]. It is composed only of nodes, without Access Point. Messages are exchanged and relayed between nodes. Figure 1 shows a small ad hoc wireless network. The circles represent the radio transmission range of the nodes. Node *A* can communicate with node *D* via nodes *B* and *C*, and vice versa.

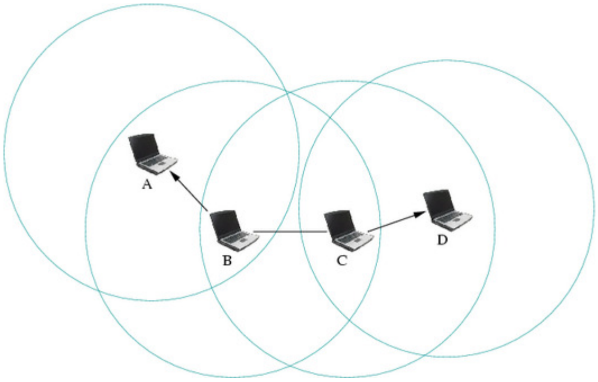


Figure 1 A Small Ad Hoc Wireless Network

The routing protocol in the ad hoc network is to find a feasible path from source to destination based on certain criteria, such as hop counts, power requirements and lifetime of wireless line etc. Unlike routing in infrastructure-based wireless network, in ad hoc wireless network, it needs to be handled by ordinary hosts that have neither specialized equipment nor a fixed, privileged position in the network. Every mobile host participating in a MANET, needs to serve as a network router to transmit packets for other hosts as well. The functionalities generally assigned to infrastructure components, such as switches and routers, need to be accomplished by the regular nodes participating in the network. Due to the inherent non-uniform propagation characteristics of wireless transmissions and the possibility that any or all of the hosts involved may move at any time, the routing problem in a real MANET may be very complicated. Therefore, designing routing for ad hoc wireless network could be very challenging. The issues, such as the mobility of the nodes, their limited energy resources and so on, all need to be taken into consideration. The basic goals while designing solid routing protocols are to minimize control overhead, packet loss ratio, end-to-end delay and energy usage while maximizing the throughput.

1.1 Motivation and Objective

There are plenty of existing routing protocols for MANET which are designed to emphasize different implementation scenarios. They are categorized based on their underlying architectural framework. We discuss here a type of source-initiated routing, also known as *reactive*, or *on-demand*, routing in which a route discovery procedure will be invoked by the source to find a feasible path to the destination. They do not maintain the network topology, but obtain the path information by using a connection establishment process. Hence, periodic update of route information is not required. In *on-demand* dynamic routing protocols, hosts follow the flooding based route discovery procedure to find the route to the destination. In a traditional route discovery process, the source host, also called the initiator, broadcasts a route request packet (RReq) to search for the destination host. Each host in the network that receives the route request will rebroadcast the RReq only once if it is not the destination host. When the intermediate hosts rebroadcast the RReq, they append their own address on the RReq. When the destination host receives the first arrival route request, it sends back a route reply using the memorized path (or a single memorized hop toward source node) recorded in the RReq.

In all existing *on-demand* routing schemes, during route discovery, each host that receives the RReq will check if it is the destination. If it is not, after attaching its address to the route record, the host will rebroadcast the RReq to its immediate neighbors, but only once. We refer to this scheme as 1R (1-Retransmission RReq). More precisely, this option is described as follows: when a node decides to retransmit, it does so only once. In 1R, distant neighbor may accidentally receive the only route discovery message and happily use it to announce a new route, although in reality it is not able to provide such a good service for the real traffic. Another issue is that only one communication can miss an important neighbor and therefore the route may not be found. Overall, the constructed route may be far from the optimal one. Our main motivation is to address this 1R design and propose alternatives with better tradeoffs between success rates, overheads in route request process, and quality of discovered routes. We are not aware of any existing effort to study the effectiveness of rebroadcasting the RReq more than once.

Detailed protocol description (e.g. for DSR) include some features for repeated RReq. Sources *should* occasionally initiate a new Route Discovery for the packet's destination address. In the case of the near-partitions, protocols have features to “try again”, which should increase the success probability. These features, however, are orthogonal to the main novelty introduced here, and can be added to them as well. To keep protocol design, discussion, and simulations simple, tractable, and explicable, we do not consider these features in existing or new protocols. We also make some further simplification, by basically considering the effect of a single routing task rather than moving discussion to the transport layer. For instance, we do not consider limitations arising from the ‘send buffer’ size and dropping some packets from some flows.

The principle objective of this research is to design an improved route discovery scheme for reactive routing compared to the traditional (e.g. dynamic source routing) protocol. The proposed route discovery scheme will be simulated running on a realistic physical layer model, the *shadowing propagation model*. Meanwhile, the traditional dynamic source route discovery scheme will also be simulated on the realistic physical layer model for comparison purpose. The proposed new route discovery scheme is to rebroadcast the RReq multiple times, instead of only once. Rebroadcasting more than once is likely to find better routes, also it may help to find a distant neighbor, thus increase the success rate of route discovery. To reduce the overhead, the *n-retransmission c-reception* scheme is proposed. A home-made simulator is specially designed in MATLAB to suit the comparing purpose of this research.

1.2 Contribution

In the proposed *n-retransmission* scheme (*n*R), once a node decides to retransmit RReq, it will do so exactly *n* times. Therefore for *n*=1 this becomes equivalent to 1R scheme that corresponds to existing solutions (e.g. DSR). We also propose *n*cRR based on *n*R, a *n-retransmission c-reception* scheme, where, once a node decides to retransmit, it will do so until the number of its retransmissions reaches *n*, or the number of received copies of same RReq reaches *c* (*c*>=2). In case of 12RR, nodes will either make one retransmission, or will make no retransmission at all if, while waiting for its own scheduled retransmission to occur, it does receive yet another, second, copy of the same RReq. This is an appealing alternative for high-density ad hoc networks. If there are hundreds nodes in a same very small area, there is no need for all of them to retransmit before a good route has been constructed. In case of 22RR, nodes will make up to 2 retransmissions. When the second copy of the same RReq is received, it will stop retransmitting. In *n*3RR, node will keep retransmitting, at most *n* times, until two more copies of the same RReq are received.

The existing 1R (e.g. DSR), its generalization *n*R, and improvement *nc*RR (improvement is expected for dense networks), are three basic options in this article. For each of them, we discuss additional options based on some route discovery and route reply decisions. It is commonly agreed that nodes do not send acknowledgments for received RReq packets. Route discovery options are with respect to making decision on whether or not to retransmit RReq. We consider two options, R1 and R+.

R1: The sender retransmits only upon receiving the first copy of received RReq. When route discovery message is received again, it is simply ignored. This is how DSR handles the same copies of received RReq. Therefore, [1R, R1] is equivalent to DSR in terms of handling RReq packets.

R+: If a node receives route request with a better cumulative cost upon arrival, it will retransmit once again.

Note the distinction between 1R and R1. In R1, the same node may decide several times to send RReq, upon receiving any route with better cost. If R1 is coupled with 1R then one retransmission is sent each time. If R1 is used in conjunction to *n*R, then each time when a better route is found, the node retransmits the corresponding RReq for *n* times. We note here that it makes perfect sense to always retransmit only the best available offer. Therefore, in *n*R, if the current best route is still not retransmitted *n* times, and a route with better route cost arrives, then the counter *n* restarts with R+, and continues with R1, and only better routes are advertised afterwards. Similar adjustment is made for the corresponding *nc*RR scheme, which is acting like *n*R, except that each received copy of RReq is counted and it is checked if the amount of received copies hits the threshold *c*.

Our proposed *n*R and *n*cRR schemes may be combined with both options R1 and R+. Thus we can define algorithms [*n*R, R1]=*n*R1, [*n*R, R+]=*n*R+, [*n*cRR, R1]=*n*cRR1 and [*n*cRR, R+]=*n*cRR+. In *n*R+ and *n*cRR+, whenever a better route is detected, retransmission counter is reset, as if the RReq came for the very first time. We consider these options as main ones for our new schemes, although they can be defined also with the alternative R1. In *n*R1 and *n*cRR1 variants, the retransmission counter is not reset. If that counter reaches *n*, and better route arrives, it will not be advertised.

We now consider options stemming from route reply process. The destination node might receive multiple copies of the same RReq. However, they may be bearing different path information. Correspondingly the costs in those RReq are also different. According to RFC, “A node originates a Route Reply in order to reply to a received and processed Route Request, according to the procedures described in Sections 8.2.2 and 8.2.3. The Route Reply is returned in a Route Reply option (Section6.3).” It implies that a destination node responds to every request packet received. However, this may cause more overhead. An alternative is to respond to the request which only contains the better cost. Here are the two options when determining which route reply packets are supposed to be responded.

B*: The destination node reports back *all* received RReqs; according to above discussion, this is the main option.

B+: The destination node reports back the first time, and any time measured value of path metric bearing better path than previously received ones.

While a route reply packet is being transmitted back to the source node, only nodes on the memorized path will perform the transmission of the reply packet. Because the realistic physical layer is applied, whether a route reply packet could be successfully sent back to the source node depends on the reception probabilities between nodes along the discovered path. Here are three options of how a route reply packet is processed by the nodes on the path.

A1: Each node on the discovered path transmits the route reply packet exactly ones. If any failure happens, process stops;

A3: Each node on the discovered path transmits the route reply packet once. If there is no transmission overheard by a neighbor, it transmits again. If again no transmission overheard, it transmits the third time. No more attempts made if again no acknowledgment is received; this is considered the main option in our analysis.

Au: Each node on the discovered path transmits the route reply packet for maximum u times until first transmission is overheard, where u=round((1/p(x))-0.1), where p(x) is the packet reception probability on link x. It can be shown that this is the optimal choice for the number of retransmissions u.

In reactive routing protocols, hosts follow the flooding based route discovery procedure (such as DSR or AODV) to find the route to the destination. During the route discovery process, each host, receiving RReq, will find the cumulative cost of the route from the source to itself (measured as hop count, or power consumption, or any other metric), and decide whether the cost is lower than the best cost it has received so far for the same task. If so, it will retransmit the message according to the selected behavior described above. The destination will report back to the source using the memorized best path. The cost metric used in this article is the expected hop count (EHC). EHC is based on the probability p(x) that two nodes at distance x receive packet sent from one to another. Note that EHC cost of a link is not the cost of RReq during route discovery process where it is used, but the cost of future actual message over the same link (these two differ in packet length and therefore do not have same costs; however in this article we used same packet lengths for any message anywhere). For the purpose of calculating this EHC cost, option A*u* was assumed for actual message being routed over discovered path. Therefore during the routing process, all forwarders will try up to u=round((1/p(x))-0.1) times to transmit to their neighbors, and each time wait for acknowledgment before trying again. So the EHC metric may depend on the choices that are made above. It is metric applied while discovering route (no acknowledgments), but using that link as part of a route (with acknowledgments). We assume that messages and acknowledgments have same probability p(x), and all route discovery messages are of same packet length as messages later in routing, which means that probabilities during route discovery processes are same as probabilities used in EHC metric for route costs. Note also that p(x) depends also on the packet lengths. For some existing algorithms, e.g. DSR, the whole route is recorded and forwarded in RReq, which gradually increases packet lengths and decreases p(x). However, here we assumed that all packets are of same length, thus implicitly using version where only previous and next hops on a path are recorded (e.g. AODV) therefore preserving packet length and p(x).

This thesis is the outcome of joint research of candidate and supervisor Dr. Ivan Stojmenovic. The external examiner Dr. Thomas Kunz (Carleton University) also gave technical contribution by expanding our original model to include route reply options in the algorithm and appropriate costs in the analysis.

1.3 Evaluation

The ultimate goal of this thesis is to prove that the proposed *n*R and *nc*RR are superior to DSR(1R). The methods being compared are 1R, *n*R for *n*=2, 3, 4 (that is, 2R, 3R, 4R) and *nc*RR for *c*=2, 3,4,5,6. Since we have 1) R1 and R+, two RReq handling options; 2) B* and B+, two options that destination nodes handle RReq; 3) A1, A3 and Au, three options that nodes on the discovered path handle RRep, so in total there are 2*2*3=12 option combinations that could be considered. But we need to be systematic (to avoid data overload) and look at desirable outcomes.

In the first round, RReq and RRep handling options R1, B* and A3 are selected as the main ones to test 1R and *n*R. Then, we run these options for selected *n*R and *nc*RR methods in the second round. Measured values are EHC, success rates, and message overhead during route discovery. After these two rounds, we will make some conclusions about the best value of *n* in *n*R and *nc*RR, and the best value of *c* in *nc*RR. Next, it is interesting to consider alternatives as competing to each other, while fixing best values of *n*. For example, combinations [Au, B*, R1] vs [A3, B*, R1] with some best new method. Similarly, [B*, A3, R1] vs [B+, A3, R1]. Finally, [R1, A3, B*] vs [R+, A3, B*] also on few best methods.

The simulation results in the first round show retransmitting route discovery message more than once will improve quality of the discovered routes by up to 31% and route discovery success rate by up to 58%. The best amount of RReq retransmission *n* is 2. However, the overhead rises tremendously as *n* increases. The simulation results in the second round show that the 2*c*RR scheme reduces message overhead by about 19.6% ~ 79.2%, depending on the network density. Overall, the *nc*RR scheme has the best tradeoff between quality of route, success rate and message overhead in route discovery process when the counter threshold *c* is 3 or 4. Finally, we compared option combinations while best *n* and *c* are chosen, namely *n*=2 and *c*=3. The results show that option R+, B* and A3 perform relatively better than others.

1.4 Thesis Organization

The rest of the thesis is organized as follow:

- Chapter 2 presents the background literature and related work for the stated problem, and gives a brief introduction on the *log-normal shadowing* model as the realistic physical layer model. Packet reception probability P(x) and expected hop count (EHC) are defined.
- Chapter 3 first proposes a new route discovery scheme, *n-retransmission* scheme. Then its performance in near-partitioned networks is analyzed. To handle the overhead issue cause by multiple retransmissions, another new scheme, *nc*RR scheme is proposed.
- Chapter 4 introduces how the simulation is implemented, and presents the simulation results from both the proposed route discovery schemes and the traditional one (DSR), while coupling with a few of combinations of R*z* (R1 and R+), B*y* (B* and B+) and A*x* (A1, A3 and A*u*), followed by some analysis regarding the simulation results.
- Chapter 5 draws the conclusions of this research and suggests some future work to further this research.

Chapter 2 Background

In MANET, the main goal of routing algorithm is to correctly and efficiently establish a route between a pair of nodes in the network so that a message can be delivered according to the expected QoS parameters. Ideally, a routing algorithm for MANET should not only have the general characteristics of any routing protocol but also consider the specific characteristics of a mobile environment—in particular, bandwidth and energy limitations and mobility. Some of the characteristics are: fast route convergence; scalability; QoS support; power, bandwidth, and computing efficient with minimum overhead; reliability; and security. All these challenges signaled the rising interest in routing algorithm designing for MANET.

2.1 Routing Protocols for Ad hoc Wireless Networks

So far, there have been many existing routing protocols for ad hoc networks emphasizing different implementation scenarios. Based on their underlying architectural framework, there are mainly the following three categories.

- ***Source-initiated (reactive or on-demand) protocols***
- ***Table-driven (pro-active) protocols***

- ***Hybrid protocols***

2.1.1 Source-initiated (reactive or on-demand) protocols

Source-initiated routing is also known as source routing, in which the route is created only when the source node requests a route to a destination node. The source node invokes a route discovery procedure by broadcasting a RReq when the route is requested. The RReqs are flooded to the network. When the source node receives one or multiple discovered routes to the destination, the route discovery process is over. The active routes are maintained in route maintenance procedure until it is not needed.

2.1.1.1 Dynamic Source Routing

Dynamic Source Routing (DSR) [4] is one of the most widely referred reactive routing protocols, which is also the one that this research is trying to improve. Dynamic source routing (DSR) is a reactive routing protocol, which establishes a route from a source node to a destination node only on demand. One feature DSR different from conventional routing protocols is DSR doesn’t use periodic routing advertisement messages. Unlike another typical reactive routing protocol, Ad hoc On-Demand Distance Vector routing (AODV) [5], in which the source node and the intermediate nodes store the next-hop information corresponding to each flow for data packet transmission, DSR uses source routing in which a data packet carries the complete path to be traversed. Source routing is a routing technique in which the sender of a packet determines the complete sequence of nodes through which to forward the packet. A route is explicitly listed in the packet's header by the sender, identifying each forwarding hop by the address of the next node to which to transmit the packet on its way to the destination node.

DSR has two major phases, Route Discovery and Route Maintenance. In Route Discovery phase, the source node initiates a route discovery process by generating and broadcasting a RReq. The RReq contains the address of the original initiator of the request and the target of the request, a route record which accumulates a record of the sequence of hops taken by the RReq as it is propagated through the ad hoc network during this route discovery, and also a unique request id which set by the initiator. In order to detect duplicate route requests received, each node maintains a list of the <initiator address, request id> pairs that it has recently received on any route request. When any host receives a RReq, it will check if the pair <initiator address, request id> is in this host's list of recently received requests. If it is, discard the request packet and process no further. Otherwise, it will check if this host's address is listed in the route record in the request already. If it is, discard the request packet and process no further. Otherwise, it will check if this host is the target. If it is, return a copy of this route in a route reply packet to the initiator. Otherwise, append this host's own address to the route record in the request packet and rebroadcast it. Since each host also maintains a route cache, when the RReq reaches a host that has a recent route to the destination, the host will return a route reply message to the source host rather than forwarding the route request message further. A route reply packet that is sent back to the source node contains the complete route from the source to the destination. Thus, the source node can initiate the routing of the data packets and meanwhile caches this route in its route cache. The Figure 2 is a flowchart of how a RReq is processed when received by any node in the wireless network.

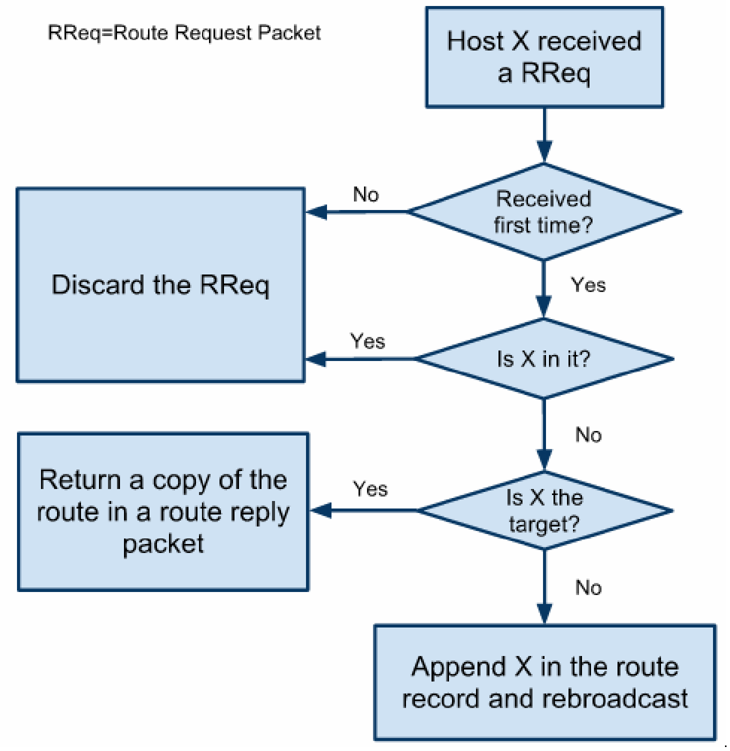
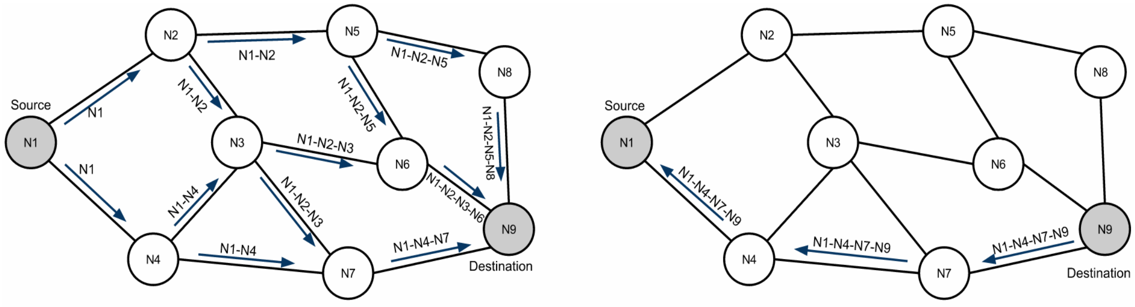


Figure 2 Flowchart of RReq Processing in DSR

The following Figure 3 is an example of the creation of a route record in DSR [23].



(a) Building of the route record during route discovery

(b) Propagation of the route reply with the route record

Figure 3 Creation of a Route Record in DSR

The route maintenance procedure is to monitor the operation of the route and informs the sender of any routing errors while a route is in use. To provide early detection and retransmission of lost or corrupted packets, a lot of wireless networks utilize a hop-by-hop acknowledgement at the data link level, such as IEEE 802.11. If such a lower-level acknowledgement mechanism is not supported, there are some other alternatives: a) passive acknowledgements; b) existing transport or application level replies or acknowledgements; c) adding a bit in the packet header to request an explicit acknowledgement from the next-hop receiver. Route maintenance can also be conducted by using end-to-end acknowledgements rather than hop-by-hop acknowledgements.

2.1.1.2 Ad hoc On-Demand Distance Vector

Ad hoc On-Demand Distance Vector (AODV) [5] is an improvement on the Destination-Sequenced Distance-Vector (DSDV) routing protocol [2]. It is essentially a combination of DSR and DSDV because it had the sequence numbers from DSDV and the on-demand route discovery and route maintenance from DSR. AODV tries to minimize the amount of broadcasting by creating route on-demand as opposed to DSDV which keeps the list of all routes. To discover a route to the target node, the source node invokes a route discovery procedure by broadcasting the RREQ. Its immediate neighbors then retransmit the packet to their neighbors till it hits either the destination node or some node which holds an up-to-date route to the destination node. If any intermediate node receives the same RREQ, simply discards it. Sequence numbers are used by RREQ to 1) ensure that the routes are loop free and 2) make sure that the latest information are replied if the intermediate nodes reply to RREQ. A node also records in its tables the node from which the first copy of the RREQ came when it forwards a RREQ to its neighbors. This information is used to construct the reverse path for the route reply packet (RREP). Because the RREP follows the reverse path of the RREQ, AODV uses only symmetric links. The nodes on the discovered route record the forward route info into their tables as the RREP traverses back to the source node. A new route discovery procedure will be invoked if the source moves. When one of the intermediate node moves, it will send a link failure notification message to each of its upstream neighbors and so on till it reaches the source so that the source can revoke another route discovery if required. What makes AODV different from other routing protocols is its integration of multicast routing. It combines unicast, multicast and broadcast communication; use only symmetric links between neighboring nodes. To ensure loop free and route information up-to-date, destination sequence numbers are used whenever route discovery is executed. AODV deletes invalid route information by using a route error message called Route Error (RERR).

2.1.1.3 Temporally Ordered Routing Algorithm

Temporally Ordered Routing Algorithm (TORA) [6] is a highly adaptive, efficient and scalable routing protocol based on the concept of link reversal. It is designed for highly dynamic mobile, multihop wireless networks. TORA is a source-initiated on-demand routing protocol which finds multiple routes from a source node to a destination node. The main feature of this protocol is that the control messages are localized to a very small set of nodes near the occurrence of a topological change. In order to achieve this, the nodes maintain routing information about adjacent nodes. TORA has three major functions: route creation, route maintenance and route erasure. The metric this protocol uses contains the following five elements: 1) logical time of a link failure; 2) the unique ID of the node that defined the new reference level; 3) a reflection indicator bit; 4) a propagation ordering parameter; 5) the unique ID of the node. A directed acyclic graph (DAG) rooted at the destination node will be established during the route creation and route maintenance phases. Based on the relative height metric of the adjacent nodes, the link can be either an upstream or downstream. The first three elements collectively represent the reference level. Each time a node loses its last downstream link due to a link failure, a new reference level is defined. The last two elements define a delta regarding the reference level. There might be oscillations occurring in TORA, especially when multiple sets of coordinating nodes are concurrently detecting partitions, erasing routes, and building new routes based on each other. Similar to the "count-to-infinity" problem in distance-vector routing protocols, TORA has instability problem because it uses intermodal coordination. However, such oscillations are temporary and route convergence will ultimately occur.

2.1.1.4 Associativity-Based Routing

Associativity-Based Routing (ABR) protocol [7] is defines a new metric for routing known as the degree of association stability, "associativity", which means, in ABR, a route is selected based on associativity stats of nodes, in which way the routes are likely to be long-lived. It is free from loops, deadlock, and packet duplicates. There are also three phases in ABR which are Route discovery, route reconstruction and route deletion. Associativity ticks are used to maintain a “degree of associativity” in ABR. All nodes in the network periodically generate beacons to signify their existence. A neighbor node updates its associativity tables when a beacon is received. For every received beacon, a node increments the associativity tick with respect to the node from which it received the beacon. High association stability means high connection stability. A low value of associativity tick may indicate a high stat of node mobility. When the neighbors of a node or the node itself move out of proximity, the associativity ticks are reset. The fundamental goal of ABR is to discover longer-lived routes. However, there are some inherent shortcomings about ABR which includes memory requirements for the routing tables, excessive storage needs for storing the associativity ticks, and additional computation to maintain the tick count along with great power requirements.

2.1.1.5 Signal Stability-Based Adaptive Routing

Signal Stability-Based Adaptive Routing (SSA) protocol [8] selects routes based on the signal strength between nodes and a nodes’ location stability, among which the signal strength (link quality) plays a more important role in the route selection process in this protocol. Functionally, the SSA protocol comprises of two cooperative protocols: the Forwarding Protocol (FP) and the Dynamic Routing Protocol (DRP), which utilize the extended device driver interface. This interface is responsible for making available to routing protocols the signal strength information from the device. DRP maintains the routing table by interacting with the DRP on other hosts. FP performs the actual routing table lookup to forward a packet onto the next hop. Two tables are maintained in the SSA protocol: the Signal Stability Table (SST) and Routing Table (RT). The SST stores the signal strength of neighbor nodes obtained by periodic beacons from the link layer of each neighbor node. Signal strength is either recorded as a strong or weak channel. DRP receives all transmissions and process them. And then it passes the packet to the SRP after updating the corresponding table entries. The packet is passed up the stack by the SRP if it is the intended receiver. If not, it looks up the destination in the RT and forwards the packet. A route-search process will be invoked if there is no entry for the destination in the RT. RReq are forwarded to the next hop only if they are received over strong channels and have not been previously processed (to avoid looping). The destination chooses the first arriving route-search packet to send back as it is very likely that the packet arrived over the shortest and/or least congested path. The DRP reverses the selected route and sends a route-reply message back to the initiator of route request. All nodes on the discovered path update their RTs accordingly. When a host moves out of range of its neighbors or shuts down, the neighbors will recognize that the node is not reachable because they no longer receive beacons from that node. The DRP will modify the SST and RT to reflect the changes. The node detecting the failure sends an error packet to the source. The source FP will send a message to erase the invalid route, and will also initiate a new route discovery to find an available route.

2.1.2 Table-Driven (Proactive)

Table-driven routing protocols let each node in the network maintain one or more tables holding routing information to every other node, and they respond to changes in network topology by propagating updates throughout the network in order to maintain a consistent, up-to-date view of the network. These table-driver routing protocols differ in the method by which the topology change information is distributed across the network and the amount of necessary routing-related tables. Basically, they all have the same goal of reducing route maintenance overhead as much as possible. This type of protocols are not suitable for highly dynamic networks due to the extra control overhead generated to keep the routing tables consistent and fresh for each node in the network.

2.1.2.1 Destination-Sequenced Distance-Vector

Destination-Sequenced Distance-Vector (DSDV) [2] is based on the classical Belman-Ford routing algorithm [3]. The major improvement made by DSDV is the avoidance of loops in routing tables. Each node in the network maintains a routing table that lists all available destinations, the amount of hops to reach the destination and the sequence number assigned by the destination node. The sequence number is to make sure the mobile nodes to distinguish old routes from new ones, thereby avoiding the formation of routing loops. To maintain the table consistency, DSDV adopts both periodic and triggered routing updates. Triggered routing updates are used when network topology changes are detected in order to propagate the routing information as fast as possible. Route updates have two possible types of packets to help reduce the potentially large amount of network traffic that such updates generate. The first is known as a “full dump”. This type of packet carries all available routing information and can require multiple network protocol data units (NPDUs). Another type of packet, called “incremental”, will only carry the information changed since the last full dump and should fit in one NPDU in order to reduce the traffic amount generated. If there is space in the incremental update packet, then those entries may be included whose sequence number has changed. When the network is relatively stable, incremental updates are sent to avoid extra traffic and full dump are relatively infrequent. In a network with high mobility, incremental packets can grow big, so full dumps will be more frequent. Each route update packet, in addition to the routing table information, also contains a unique sequence number assigned by the transmitter. The route labeled with the highest (i.e. most recent) sequence number is used. If two routes have the same sequence number then the route with the best metric (i.e. shortest route) is used. In an environment where many independent nodes transmit routing tables asynchronously, some fluctuations could develop. DSDV also uses settling time to prevent fluctuations of routing table updates. The settling time is used to decide how long to wait before advertising new routes.

2.1.2.2 Clusterhead Gateway Switch Routing

Clusterhead Gateway Switch Routing (CGSR) [9] is a clustering scheme using a distributed algorithm called Least Cluster Change (LCC). Mobile nodes in a wireless network are aggregated into clusters and a cluster-head is selected. All nodes that are in the communication range of a cluster-head belong to its cluster. A gateway node is a node in the communication rage of two or more cluster-heads. In a dynamic network, cluster head scheme can cause performance degradation due to frequent cluster-head selections. LCC algorithm is introduced to prevent frequent cluster head changes. According to the algorithm, only two situations cause the cluster head to change. One is when a node becomes disconnected from any cluster, and the other is when two cluster heads come within range of each other. Each node has two tables for route packets. One is the cluster member table which is used to map a destination address to the destination cluster head address, and the other one is the routing table which is used to select the next node to reach the destination cluster. The source node transmits the packet to its cluster-head first. From the cluster-head, the packet is sent to the gateway node that connects this cluster-head with the next cluster-head along the route to the destination. Then the packet is transmitted through gateways and cluster-head one after another till it reaches the cluster-head that has the destination information. Finally the destination cluster-head transmits the packet to the destination node. On receiving a packet, a node tries to discover the nearest cluster-head along the route to the destination according to the cluster member table and the routing table. Then through the routing table, it finds the next hop in order to reach the cluster-head selected earlier and then transmits the packet to that node.

2.1.2.3 Wireless Routing Protocol

Wireless Routing Protocol (WRP) [10] is one of the earliest works on routing algorithms. It is similar to the distributed Bellman-Ford algorithm. Each node in the network maintains a Distance table, a Routing table, a Link-Cost table and a Message Retransmission list. The Distance table of a node *x* contains the distance of each destination node *y* via each neighbor *z* of *x*. It also contains the downstream neighbor of *z* through which this path is realized. The Routing table of node *x* contains the distance of each destination node *y* from node *x*, the predecessor and the successor of node *x* on this path. It also contains a tag to identify if the entry is a simple path, a loop or invalid. Storing predecessor and successor in the table is beneficial in detecting loops and avoiding counting-to-infinity problems. The Link-Cost table contains cost of link to each neighbor of the node and the number of timeouts since an error-free message was received from that neighbor. The Message Retransmission list (MRL) contains information to let a node know which of its neighbor has not acknowledged its update message and to retransmit update message to that neighbor. Nodes exchange routing tables with their neighbors using update messages periodically as well as on link changes. The nodes present on the response list of update message (formed using MRL) are required to acknowledge the receipt of update message. If there is no change in routing table since last update, the node is required to send an idle “Hello” message to ensure connectivity. On receiving an update message, the node modifies its distance table and looks for better paths using new information. Any new path so found is relayed back to the original nodes so that they can update their tables. The node also updates its routing table if the new path is better than the existing path. On receiving an ACK, the mode updates its MRL. A unique feature of this algorithm is that it checks the consistency of all its neighbors every time it detects a change in link of any of its neighbors. Consistency check in this manner helps eliminate looping situations in a better way and also has fast convergence.

2.1.3 Hybrid Protocols

Hybrid routing protocols combine the power of both reactive and proactive routing protocols. Static routing is generally used at the fringes of the network where route changes are not frequent while in the core of the network on-demand routing has more significance. These schemes create a bridge between the two major types of routing protocols and the overall performance obtained can be further improved.

2.1.3.1 Zone Routing Protocol

Zone Routing Protocol (ZRP) [11] is a well-known hybrid routing protocol and most suitable for large scale networks. Its name is derived from the use of “zones” which define the transmission radius for every participating node. This protocol uses a pro-active mechanism of node discovery within a node's immediate neighborhood while inter-zone communication is carried out by using reactive approaches. ZRP utilizes the fact that node communication in ad hoc networks is mostly localized, thus the changes in the node topology within the vicinity of a node are of primary importance. ZRP makes use of this characteristic to define a framework for node communication with other existing protocols. Local neighborhoods, called “zones”, are defined for nodes. Neighbor discovery is accomplished by either Intra-zone Routing Protocol (IARP) or simple “Hello” packets. IARP is pro-active approach and always maintains up-to-date routing tables. Since the scope of IARP is restricted within a zone, it is also referred to as “limited scope pro-active routing protocol”. Route queries outside the zone are propagated by the route requests based on the perimeter of the zone, instead of flooding the network. The Inter-zone Routing Protocol (IERP) uses a reactive approach for communicating with nodes in different zones. Route queries are sent to peripheral nodes using the Bordercast Resolution Protocol (BRP). Since a node does not resend the query to the node in which it received the query originally, the control overhead is significantly reduced and redundant queries are also minimized. ZRP provides a hybrid framework of protocols, which enables a use of any routing strategy according to various situations. It can be optimized to take full advantage of the strengths of any current protocols.

2.1.3.2 Fisheye State Routing

Fisheye State Routing (FSR) [12] is a hierarchical routing protocol that aims at reducing control packet overhead by introducing the multilevel scopes. It is very effective to decrease the size of information required to represent graphical data. The concept it uses is that the eye of a fish captures with greater details the view nearer to the focal point while detail decreases as the distance from the focal point increases. FSR introduces the scopes concept, which depends on the number of hops a packet traveled from its source. A higher frequency of update packets are generated for nodes within smaller scope whereas for farther-away nodes, updates are fewer in general. Each node maintains a local topology map of the shortest paths which is exchanged periodically between the nodes. The protocol scales well to large size of networks while keeping the control overhead low without compromising on the accuracy of route calculations. Routes to farther destinations may seem stale. However, they become increasingly accurate as a packet approaches its destination.

2.2 Realistic Physical Layer Model

2.2.1 Propagation Models

In a wireless channel, factors such as path loss, interference, and blockage affect the range, data rate, and reliability of the wireless transmission. Expect the direct path between the transmitter and receiver, other components known as multipath components caused by reflection, diffraction and scattering also add to the receiver.

Free space propagation model assumes there is only direct-path signal between the transmitter and the receiver, so the received signal strength depends only on the distance. The relationship between the transmitted power P_t and the received power is P_r is given by

$$P_r = P_t G_t G_r (\frac{\lambda}{4\pi d})^2 \qquad (2.1)$$

G_t and G_r are the transmitter and receiver antenna gains, respectively; d is the distance between the transmitter and receiver. λ is the wavelength of radio signal.

In reality, the signal reaches the receiver through multipath. The two-ray model or the two-path model is more accurate, since it includes both the direct-path and another path cause by reflection (or refraction, or scattering). In the two-ray model, the received power is given by

$$P_r = P_t G_g G_r (\frac{h_t h_r}{d^2})^2 \qquad (2.2)$$

Most published results in ad hoc network use these simplistic and idealistic propagation models as the physical layer model. However in reality, the received signal strength is not only dependent on the distance between the transmitter and the receiver but also on the environment.

A more realistic physical layer model is used here, using the *Log-Normal Shadowing Model* [16]. The noise element is modeled by a Gaussian distribution. The *Log-Normal Shadowing model*, also called the shadowing model is a statistical model. The mean received power at a distance d is calculated related to $P_r(d_0)$ as

$$\left[\frac{P_r(d)}{P_r(d_0)}\right] db = -10\beta \log(\frac{d}{d_0}) + X_\sigma \qquad (2.3)$$

where d_0 is the reference distance, β is the loss exponent. X_σ is a Gaussian random variable with zero mean and standard deviation σ.

The average path loss can be expressed as a function of distance d, so the received power will be

$$P_r(d)[dBm] = P_t[dBm] - \overline{PL}(d_0) - 10\beta \log\left[\frac{d}{d_0}\right] - X_\sigma \qquad (2.4)$$

2.2.2 Probability of Packet Reception

The shadowing model can be used for area coverage calculations, the probability that the received power at a location *d* exceeds *γ* can be given as:

$$P_r\left[P_r(d) > \gamma\right] = 0.5(1 - erf\left[\frac{\gamma - \overline{P_r(d)}}{\sqrt{2}\sigma}\right]) \qquad (2.5)$$

The signals fade with distance and noise, the receiver has a certain probability for proper signal reception. The probability of packet reception p(*x*) depends on the length *L* of the packet and is given by p(*x*) = b(*x*) ^{L}. The packet transmission radius *R* is defined as the distance for which p(*R*) = 0.5 is satisfied [16]. Authors in [16] provide an approximation function for packet length *L*=120 and an error within 4% for quick calculation. The approximation function for the packet reception probability is

$$P(x) = (1 - \frac{(\frac{x}{R})^{2\beta}}{2}) \quad \text{for } 0 < x < R, \qquad .$$

$$P(x) = (\frac{(\frac{2R - x}{R})^{2\beta}}{2}) \quad \text{for } R \le x < 2R, \qquad (2.6)$$

$$\text{and} \quad P(x) = 0 \quad \text{for } 2R \le x \qquad .$$

The power attenuation factor in the (2.6) is 2*β*, because this is an approximation of packet probability rate rather than bit probability rate.

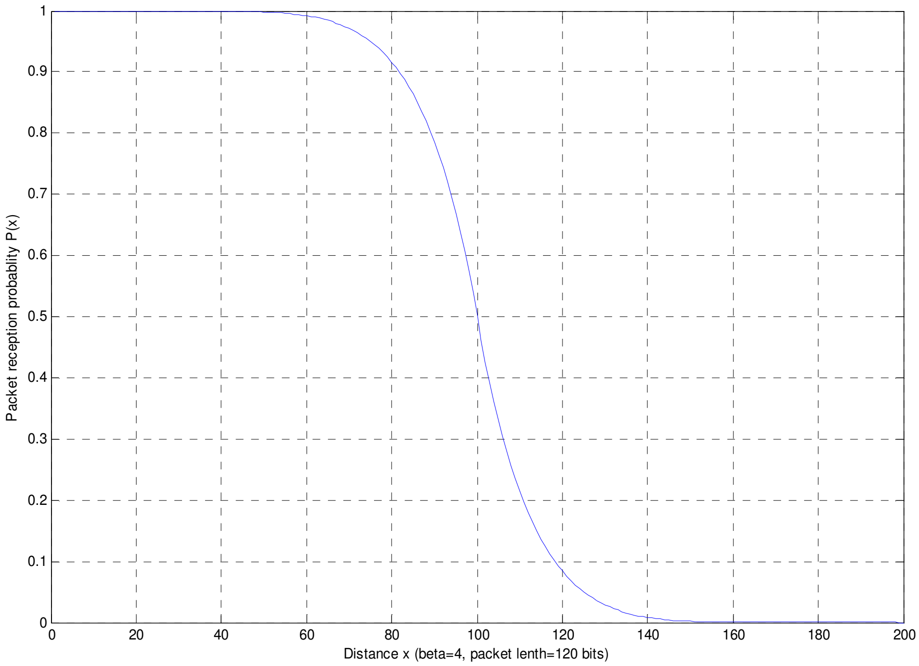


Figure 4 P(*x*) for *β*=4, *L*=120, *R*=100

Figure 4 shows the approximation of packet reception probability P(*x*) for *β*=4, *L*=120. *R* is defined as the distance where reception probability P(*x*) = 0.5. This means even outside the transmission radius *R* towards 2*R*, hosts still have the reception probability from 0.5 to 0.

Average network degree is defined as the average neighbors a host has within its transmission radius *R*. In this work, it is further defined that hosts in distance within from *R* to 2*R* can still receive the signal with perception probability from 0.5 to 0. However, according to [15], hosts are neighbors if the packet reception probability between them is above a certain minimum threshold, and they used a threshold of 0.05. Therefore, distance close to 2*R* with perception probability ≤ 0.05 is regarded as unreachable.

2.2.3 Expected Hop Count

In this work, the traditional hop count as the metric for a route is replaced by Expected Hop Count. The Expected Hop Count is derived as follow.

In [16], the authors consider a hop-by-hop retransmission routing protocol, i.e. receiving host needs to send separate acknowledgement and forwarding packets to the previous and the next hosts on the route. The receiver will send u acknowledgements for each packet received. The sender will keep sending packets until it receives any acknowledgement. The expected number of messages is the proposed measure of hop count between two hosts. The hop count is then used as weight in the routing algorithm. The expected hop count between two hosts at distance x is

$$\frac{1}{\left[p(x)(1-(1-p(x))^{u})\right]}+\frac{u}{\left[1-(1-p(x))^{u}\right]} \qquad (2.7)$$

where the first term represents expected number of packets sent by the sender, the second term represents the expected number of acknowledgments sent by the receiver. The authors in [16] proposed the optimal value for u should be dynamically calculated to satisfy u * p(x) ≈ 1. Then they derived that the best value for u is u = round((1/p(x))-0.1).

The expected hop count is used as the metric and a comparison with other routing schemes. The expected hop count h(x) that gives u * p(x) ≈ 1 is given by

$$h(x) = \frac{1}{x}\left[\frac{1}{p(x)(1-(1-p(x))^u)} + \frac{u}{(1-(1-p(x))^u)}\right] \qquad (2.8)$$

The authors in [20] stated how to obtain the distance information in ad hoc network without the aid of position information. Based on signal strength received, the receiver can estimate the distance. Let P_t and P_r be the power levels on which a message is sent and received, respectively. The received signal power is

$$P_r = P_t(\frac{c1}{d})^n c2 \qquad (2.9)$$

where n, $c1$ and $c2$ are constants related to physical environment, the carrier's wavelength, and antenna gains, respectively. Since P_r and P_t can be measured, the distance d can be estimated from formula (2.9).

Each host in the ad hoc network after receiving message, can calculated the distance from the sender based on the receiving power and derived the expected hop count from the receiver to the sender. This information is passed to upper network layer for decision on the routing process.

2.3 Related Work

In [13], the authors questioned the idealized binary perfect-reception-within-range models used in common ad hoc network simulation tools. They present mathematical link layer models for the statistical variation of packet reception rates with respect to distance, the analysis of which yields there are three distinct reception regions in a wireless link: connected, transitional, and disconnected. The transitional region is often quite significant in size, and is generally characterized by high-variance in reception rates and asymmetric connectivity. Particularly, in dense deployments such as those envisioned for sensor networks, a large number of the links in the network (even higher than 50%) can be unreliable due to the transitional region. Then they derived expressions for the packet reception rate as a function of distance and for the width of the transitional region to show how the transitional region is impacted by important radio parameters such as modulation, encoding, output power, frame size, noise floor, and channel parameters. The authors claimed that one of the key findings in their work is, for radios using narrow-band modulation, because of multi-path fading, the transitional region would exist even with perfect-threshold receivers.

The authors in [14] concentrate on the impact of a realistic physical layer (shadowing propagation model) on simulating the performance of AODV and DSR on-demand wireless routing protocols. They proposed to use new signal power thresholds to enable the selection of links with strong enough signal strength and reduce some protocol control messages. The simulation results from the research showed that in most cases there are significant increase in the packet delivery ratio and decrease in packet latency and control messages. They also suggested that the link status is a better metric than hop count for selecting routes in shadowing models. However, the drawback of this approach is that increasing threshold will correspondingly decrease the transmission radius, thus the network is likely to be partitioned even if it is not, and also increase the hop count metric. The authors believe that the best route approach should be used instead of the threshold-based one, since thresholds may prevent operational links from being included in possibly the best or sometimes the only route.

Some guidelines are presented in [15], to design network protocols when Unit Disk Graph (UDG) model is replaced with a more realistic physical layer model. They proposed that physical, MAC and network layers share the information about a bit/packet reception probability as a function of distance between hosts. It is suggested that an optimal route discovery protocol cannot be based on a single retransmission by each host, because such a search may fail to reach the destination or find the optimal path. A guideline for the design of greedy position-based routing protocols with known destination locations is proposed. The node currently holding the message will forward it to a neighbor (closer to the destination than itself) that minimizes the ration of cost over progress, where the cost measure depends on the assumptions and metrics used, while the progress measures the difference in distance to the destination.

In [16], the authors used the log normal shadow fading model to represent a realistic physical layer to derive the reception probability p(x) (as function of distance x between two hosts.) They defined the transmission radius R as the distance at which p(R) = 0.5. They suggested that forwarding to neighbor closest to destination is suboptimal. The optimal forwarding distance is one that minimizes Expected Hop Count (EHC) per progress made. Thus they derived that the optimal forwarding radius is between 0.7R and 0.8R. Then they proposed the method to use optimal hop count function h(x) in route discovery.

Two protocols, s-hello protocol and target density protocol, are proposed in [17] to address the problem of gathering neighbor information when a realistic physical layer is applied. The basic idea behind those two protocols is that every node in the network sends a fixed amount of messages in order to collect accurate local knowledge information. And in the end, the authors suggested the same idea could be further applied in the related problem of route discovery in reactive routing schemes, route discovery packet being retransmitted only once after it is received, which is addressed in this work.

In [18], the authors provided a general theoretical model of the attainable throughput in multi-rate ad hoc wireless networks. Then they proposed to use Medium Time Metric

(MTM) that selects optimal throughput and to avoid long unreliable links. They concluded that link rate information from the MAC layer can be utilized by routing protocols to significantly increase network performance.

In [19], the authors stated that countless experimental studies already published cannot be replicated because they do not fully report the conditions in which they were carried out. This situation is especially troublesome when the number of parameters in the simulation is large. It is not clear what the desired combinations are in parameter space. One should be aware that model parameters have default values hidden within the simulator. These defaults may not set the exact scenario the experiment has in mind.

In [20], the authors showed the problem of redundancy, contention, and collision in ad hoc network caused by broadcasting by flooding. They defined the problems associated with flooding as the broadcast storm problem. Then they proposed several schemes to reduce redundant rebroadcasts and differentiated timing of rebroadcasts to alleviate the problem.

Chapter 3 Route Discovery with *n*-Retransmission schemes

3.1 Assumption

Expected hop count is derived based on hop-by-hop routing. Route discovery process has no acknowledgements, but actual traffic may have. This means that, if actual traffic has acknowledgements, then the costs to be used in route discovery should be hop-by-hop routing. It is assumed that there are no acknowledgements in route discovery process, but the costs of each link will be those from acknowledged routing to be used after route is found. So the expected hop count is used as the costs for each link in route discovery process.

The development of the proposal and the simulation are based on the following assumptions:

- All hosts fully participate in the protocols of the network, and particularly, will forward packets for other hosts;
- Distances between neighboring nodes are known;
- All packets' length are fixed and same;
- Hosts can enable a promiscuous receive mode;

- The receiving host can derive the expected hop count to the sending host based on the distance between them;
- No obstacles between any two hosts;
- All hosts are static while a broadcast is in progress;
- Ideally there are no collisions between neighbors;
- Messages are sent at different time slot.

3.2 Impact of *n*-Retransmission on Selecting Best Route

In traditional route discovery process, intermediate hosts, after receiving the RReq, retransmit it once only. It does not seem to be a problem in route protocol design that uses an ideal physical layer model, which assumes all hosts within the transmission radius can receive the message.

Recent studies ([13] [14] [15] [16] [17]) using realistic physical layer show that packet reception probability P(x) depends on distance x.

In route discovery process, an intermediate host in ad hoc network retransmits the RReq only once. Since the neighboring hosts may miss the packet, only one retransmission can construct an inferior route. Moreover, hosts closer to the transmitter have higher probability to receive the message, thus the constructed route tends to be a sequence of hosts with short distance apart, which leads to high expected hop count.

Retransmitting the RReq more than once will increase the packet reception probability, thus leads to a better route. Figure 5 shows four constructed routes and their cumulative expected hop count, in a 40-host ad hoc network with average network degree of 6. The host with two dotted circle is the initiator and the black up-triangle is the target host. The inner circle is the transmission area of the initiator with radius R; the outer circle has the radius $2R$. Any point inside the two circles has the possibility to receive the RReq from the initiator. The packet perception probability decreases from the center to the border as depicted in Figure 4. All other hosts in the network have the same transmission radius.

Figure 5 shows that the traditional route discovery scheme takes the most direct route, which is correct if the hop count metric is used. While the metric is changed to expected hop count, *n-retransmission* (n>1) discovers route with better Expected Hop Count. The second hop of *n-retransmission* from the initiator is a host closer than the second hop in DSR, since this host provides better EHC, even if it adds one more hop on the route.

This finding is in accordance with the design principle stated in [15], that “Host A, currently holding the packet, will forward it to neighbor B, closer to the destination than itself, that minimizes the ratio of cost over progress.” The cost considered here is EHC.

Figure 5 shows that even without the position information, the constructed route will take the path which gives a better EHC.

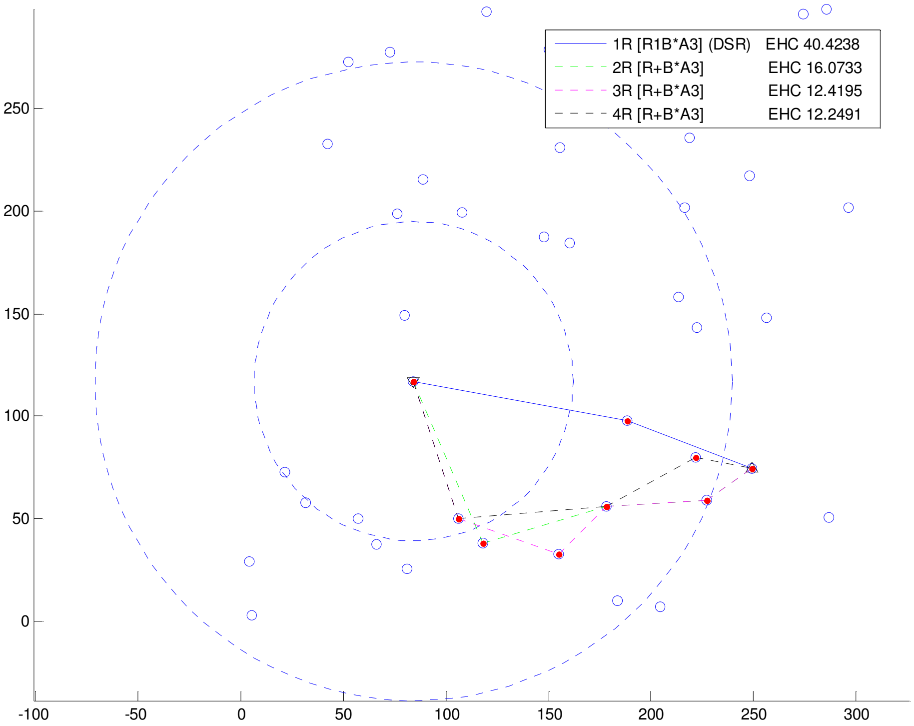


Figure 5 Discovered Routes with DSR and *n*R Scheme

3.3 Impact of *n*-Retransmission on Routing in Nearly Partitioned Network

In a sparse network, hosts tend to group into several sub-networks. It is very likely that a route discovery process will fail if the source host and destination host are located in two different sub-networks. DSR route discovery schemes where each host retransmits RReq exactly once can be problematic, since only one communication can miss an important neighbor and therefore the route may not be found. Retransmitting the RReq more than once increases the probability of finding ‘hidden’ hosts thus increases the success rate of route discovery across two semi-partitioned networks.

Figure 6 and Figure 7 show the route discoveries in a nearly partitioned network. There are no hosts existed in the stripe marked by two dotted lines in the middle of the graph. The source host and destination host are located on left and right side of the no-hosts zone. The two dotted circle around the source host indicate the transmission radius *R* and 2*R*, respectively. The task is to discovery the route from the source to the destination.

In Figure 6, traditional route discovery with only one retransmission failed to found the route to the destination. The route is discovered in Figure 7 with 2-retransmission. In Figure 7, retransmission more than once helps to find the host close to the partition board on the right subnet, which has a high probability of missing the RReq. The route cross the partition zone has high EHC due to its long distance. However, it is the only way that a route can be established between source and destination.

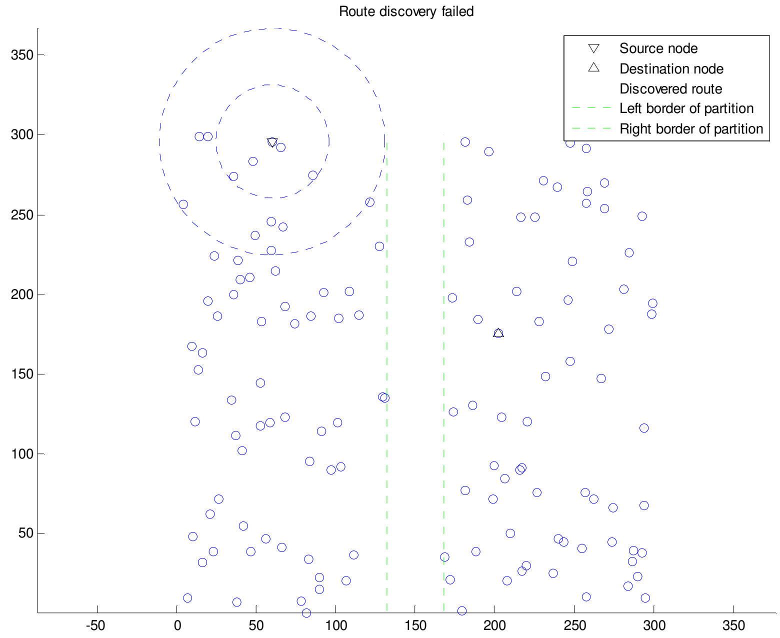


Figure 6 DSR Failed in a Nearly Partitioned Network

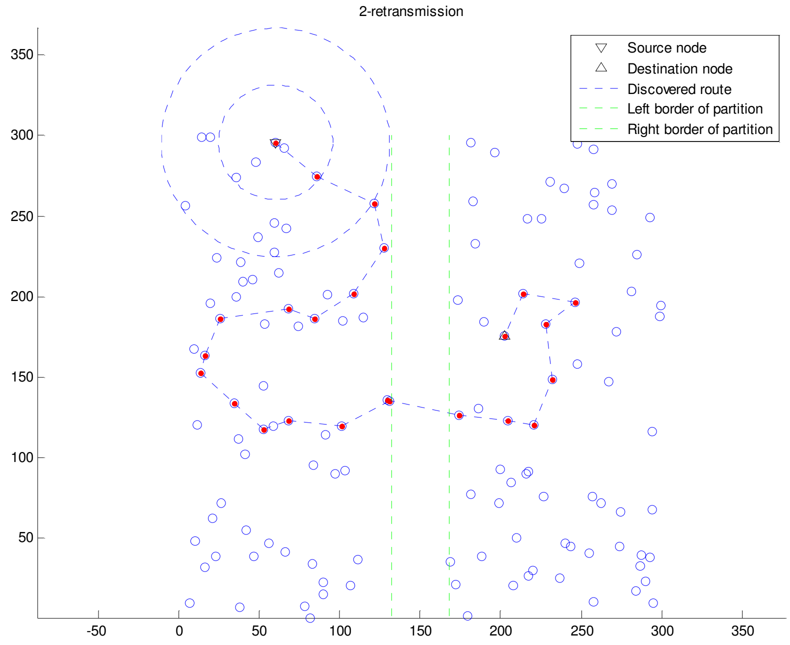


Figure 7 2-Retransmission Found One Route

3.4 Proposed *n*-Retransmission (*n*R) Route Discover Scheme

The proposed *n-retransmission* route discovery scheme is based on route discovery protocol in DSR.

The source host, also called the initiator, invokes a route discovery procedure by constructing a RReq and meanwhile scheduling *n* times broadcast. The format of RReq is shown in Figure 8. The *Target* identifies the destination address of host of the route request. Next entry is the address of the *Initiator* itself. The *Request id* is unique for each RReq, set by the initiator. *EHC* is the cumulative expected hop count from the initiator to the receiving host. *Preceding Node* is the node from which the current host received the RReq.

Route Request Packet (RReq)

Target	Initiator	Request Id	EHC	Preceding Node

Figure 8 Format of RReq

The initiator broadcasts the RReq; hosts within the initiator's wireless transmission range may receive it based on their packet reception probability. Every host maintains a rebroadcast count. After rebroadcasting the RReq for the first time, the counter is incremented and the RReq will be rebroadcasted up to *n* times. If the target host receives the RReq, it responds with a route reply packet.

This is different from what proposed in the *n-retransmission* scheme. In the step 3 of this proposed route discovery scheme, whether a received copy of the same RReq packet being discarded or not will depend on the cumulative EHC. If the accumulated EHC is better based on the newly received copy of the same RReq packet, then it will be retransmitted again. Otherwise, it will be discarded. The probability of finding a better EHC route will be increased in this way.

In DSR, when another copy of the same RReq packet is received, it will be simply discarded. However, it is very likely this copy is coming from a different preceding node and carrying a better EHC. Rebroadcasting this copy is likely to increase the probability of finding the source node a route with lower cost (better EHC). Thus, we have two options with respect to making decision on whether or not to send other copies of the same RReq message. They are

R1: The sender retransmits only upon receiving the first copy of received RReq. When route discovery message is received again, it is simply ignored;

R+: If a node receives route request with a better cumulative cost upon arrival, it will retransmit once again.

Presumably, R+ should yield better results. Thus, being combined with option R+, it is proposed that when any host receives a RReq during the route request process, it processes the RReq according to the following steps:

1. Check whether the *Target* of the RReq matches this host's own address. If it does, it means the destination host has been reached and a route from the source host to the destination host has been discovered. Then calculate and set the new cumulative *EHC*. The destination host proceeds to route reply process with this RReq; (See below option B* and B+)
2. Otherwise, check whether the RReq has been received before, by using *Initiator* and *Request id*. If it has not, the host firstly adds the cost (EHC) of the last hop to the cumulative *EHC* that is sent with the packet, set the *Preceding Node* as the current host, and then schedule *n* times rebroadcast, following adopted MAC protocol, each containing new summary EHC; (A copy of this RReq is kept in the host, in which the *Preceding Node* is pointing at the node from which the RReq is received.)
3. Otherwise, it means this RReq is a new copy of the same route discovery that has been received recently. Then the host adds the cost (EHC) of the last hop to the cumulative *EHC* that is sent with the packet, estimates whether or not new cumulative *EHC* is better than the previously best one. If the new *EHC* is better, cancel the rest rebroadcast

of the previous RReq if any, reset the rebroadcast counter and reschedule *n* times rebroadcast with the new better cumulative *EHC*;

4. Otherwise, discard the RReq and do not process it further.

We can define the above algorithm as [*n*R, R+]=*n*R+. Here is the flowchart of how the RReq are processed in *n*R+.

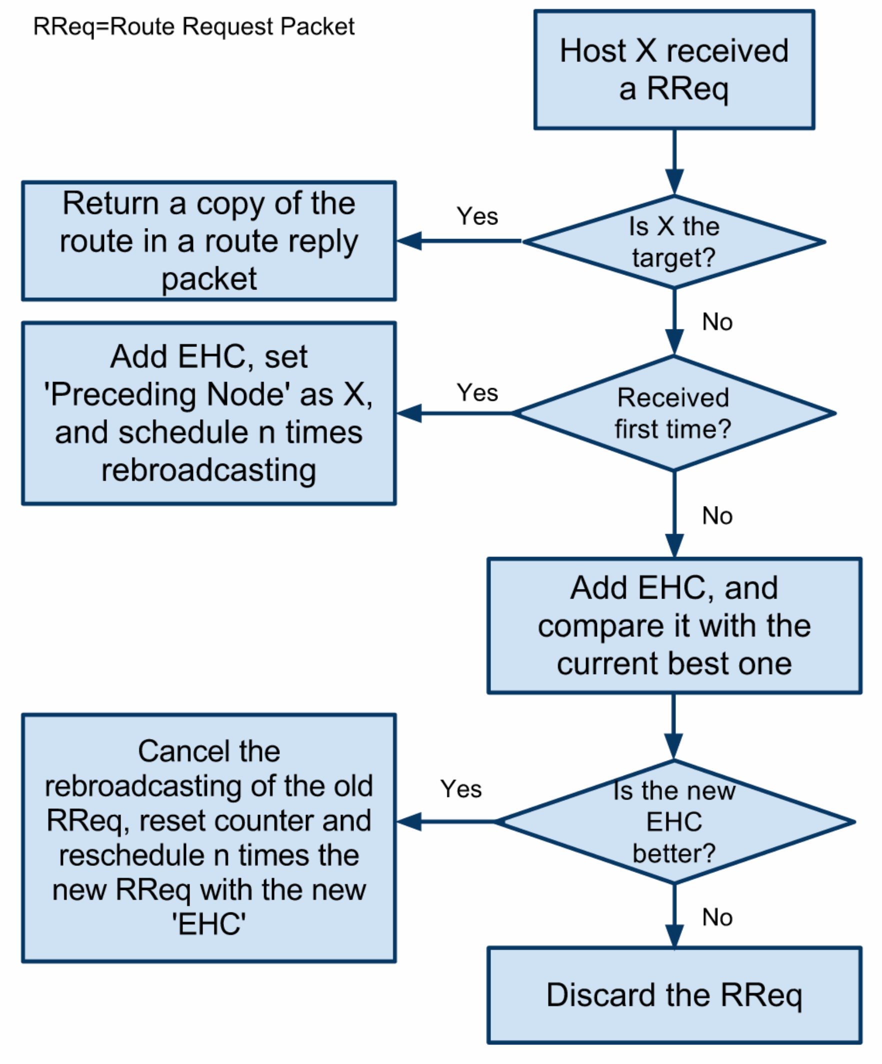
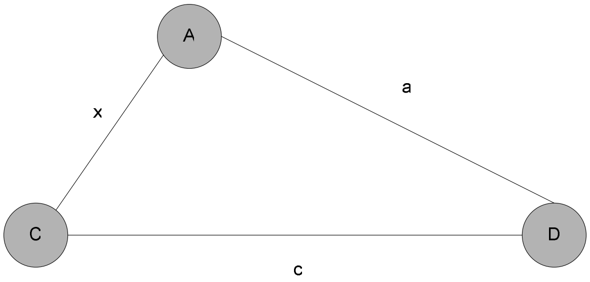


Figure 9 Flowchart of RReq Processing in *n*R+

The RReq from the initiator is received only by the hosts within the transmission range of the initiator. However, not every host in the transmission range will receive the route request. The reception depends on the packet reception probability P(x) based on the distance from the transmitter to the receiver. Each of these hosts that received the route request will propagate the request if it is not the target. All hosts except the target host that hold the RReq retransmit for n times.

Here is an example to demonstrate how this n-transmission route discovery scheme works.



Node C is broadcasting a RReq. The distance between these nodes are known based on assumption. The x=1.2, a=1.8 and c=2.4 are the distance of |CA|, |AD| and |CD| respectively. Let the transmission radius of each node R=2, and β=2. Then according to the approximation function (2.6) for the packet reception probability in Chapter 2,

$$P(x) = (1 - \frac{(\frac{x}{R})^{2\beta}}{2}) \quad \text{for } 0 < x < R \,, \quad .$$

$$P(x) = (\frac{(\frac{2R - x}{R})^{2\beta}}{2}) \quad \text{for } R \le x < 2R \,,$$

$$\text{and} \quad P(x) = 0 \quad \text{for } 2R \le x \quad .$$

P(|CA|)=0.9352, P(|AD|)=0.6719 and P(|CD|)=0.2048. Then according to the equation (2.7),

$$\frac{1}{\left[p(x)(1-(1-p(x))^{u})\right]}+\frac{u}{\left[1-(1-p(x))^{u}\right]}$$

EHC(|CA|)=2.2127, EHC(|AD|)=3.7030 and EHC(|CD|)=14.4902, in which u was dynamically calculated by u=round((1/p(x))-0.1). Obviously, EHC(|CA|)+ EHC(|AD|) < EHC(|CD|). Therefore, when node D received the RREQs from C and A, it kept the one from A and discard the one from C, then schedule n times and retransmitted the RReq with the better EHC.

Now we are considering route reply process. When the destination host receives a RReq, a route reply packet is generated and sent back to the *Initiator* according to the *Preceding Node* in the RReq of every node on the discovered route. But very likely, the destination node might receive multiple copies of the same RReq which, however, may be carrying different path information. And correspondingly, the costs (EHC) in those RReq are also different. According to RFC, “A node originates a Route Reply in order to reply to a received and processed Route Request, according to the procedures described in Sections 8.2.2 and 8.2.3. The Route Reply is returned in a Route Reply option (Section6.3).” It implies that a destination node responds to every request packet received. However, this may cause more overhead, especially in networks high in density. An alternative is to respond to the RReq packets that only contain the better cost. Here are the two options when determining which route reply packets are supposed to be responded.

B*: The destination node reports back *all* received RReqs; according to above discussion, this is the main option.

B+: The destination node reports back the first time, and any time measured value of path metric bearing better path than previously received ones.

While a route reply packet is being transmitted back to the source node, only nodes on the memorized path will perform the transmission of the reply packet. Because the realistic physical layer is applied, whether a route reply packet could be successfully sent back to the source node depends on the reception probabilities between nodes along the discovered path. Here are three options of how a route reply packet is processed by the nodes on the path.

A1: Each node on the discovered path transmits the route reply packet exactly ones. If any failure happens, process stops;

A3: Each node on the discovered path transmits the route reply packet once. If there is no transmission overheard by a neighbor, it transmits again. If again no transmission overheard, it transmits the third time. No more attempts made if again no acknowledgment is received; this is considered the main option in our analysis.

Au: Each node on the discovered path transmits the route reply packet for maximum u times until first transmission is overheard, where u=round((1/p(x))-0.1), where p(x) is the packet reception probability on link x. It can be shown that this is the optimal choice for the number of retransmissions u.

All the above options could be coupled with *n*R+, and will be evaluated in the simulation to see which one is better in performance.

3.5 Overhead Caused By *n*-Retransmission

Hosts in ad hoc network are assumed to share a single common channel with carrier sense multiple access with collision detection (CSMA/CD). In a geographical area, radio signals are likely to overlap with each other. Heavy contention for accessing the media is possible when hosts are close to each other. Collision may happen since the timing of rebroadcasts is highly correlated [20].

Retransmitting the RReq more than once will only add more overhead to the ad hoc network. The extra overhead may counterbalance the benefit of the fixed *n-retransmission* route discovery scheme, which leads to redundant rebroadcasts, contention and collision from the extra RReq flooded in the network.

Extensive study has been done to address the problem from broadcasts. In [20], the authors proposed a counter-based scheme: A host holds a message to rebroadcast may be blocked by medium contention, backoff procedure, and other queued message. It’s possible that the host hears the same message from other rebroadcasting hosts before the host actually starts transmitting the message. A counter C is used to keep track of the number of times the broadcast message is received. A counter threshold c is chosen. The host will stop and discard the message if $C \geq c$.

3.6 Proposed *n*-Retransmission *c*-Reception Route Discovery Scheme (*nc*RR)

Being inspired by [20], *n*-retransmssion *c*-reception scheme (*nc*RR) is proposed, intending to lower the overhead that the *n-retransmission* route discovery scheme causes. This proposed scheme is a modification to the *n-retransmission* route discovery scheme.

The modification is that each host sets up a counter C to keep track of number of times the same RReq is received from its neighboring hosts. A counter threshold c is introduced. If $C \geq c$, the host terminates extra retransmissions.

During the route request process, when any host receives a RReq, it processes the request with the following steps:

1. Host first adds the cost (EHC) of the last hop to the cumulative *EHC* that is sent with the packet, and sets the *Preceding Node* as the current host;

2. Host checks whether the *Target* of the RReq matches this host's own address. If it does, it means the destination host has been reached and a route from the source host to the destination host has been discovered. Then it proceeds to route reply process with this RReq;
3. Otherwise, check whether the RReq has been received before, by using *Initiator* and *Request id*. If the RReq is received for the first time then initialize the counter C=1, schedule n times retransmission. (A copy of this RReq is kept in the host, in which the *Preceding Node* is pointing at the node from which the RReq is received.)
4. Otherwise, this RReq is just another copy of the same route discovery that has been received recently. Then increment the counter C and check whether $C \geq c$. If so, cancel the retransmission and inhibit rebroadcasting.
5. Otherwise, the host estimates whether or not new cumulative *EHC* is better than the previously best one. If so, it cancels the rest of rebroadcasts of the previous RReq if any left, and reschedules n times rebroadcast with the new better cumulative *EHC*.
6. Otherwise, discard the RReq and do not process it further.

The threshold c should be more than 1 because at least one RReq is allowable to be retransmitted. But, as we discussed earlier, it is likely that a node in a dense network will never have a chance to retransmit RReq because the threshold *c* might have been reached before the retransmission could happen. As we can see, the option R+ is coupled with the above steps. Then we define algorithms [*nc*RR, R+]=*nc*RR+. During route reply process, the *nc*RR scheme adopts the same options as *n*R scheme does, B* and B+; A1, A3 and Au. Figure 10 is the flowchart of how RReq processed in *nc*RR+.

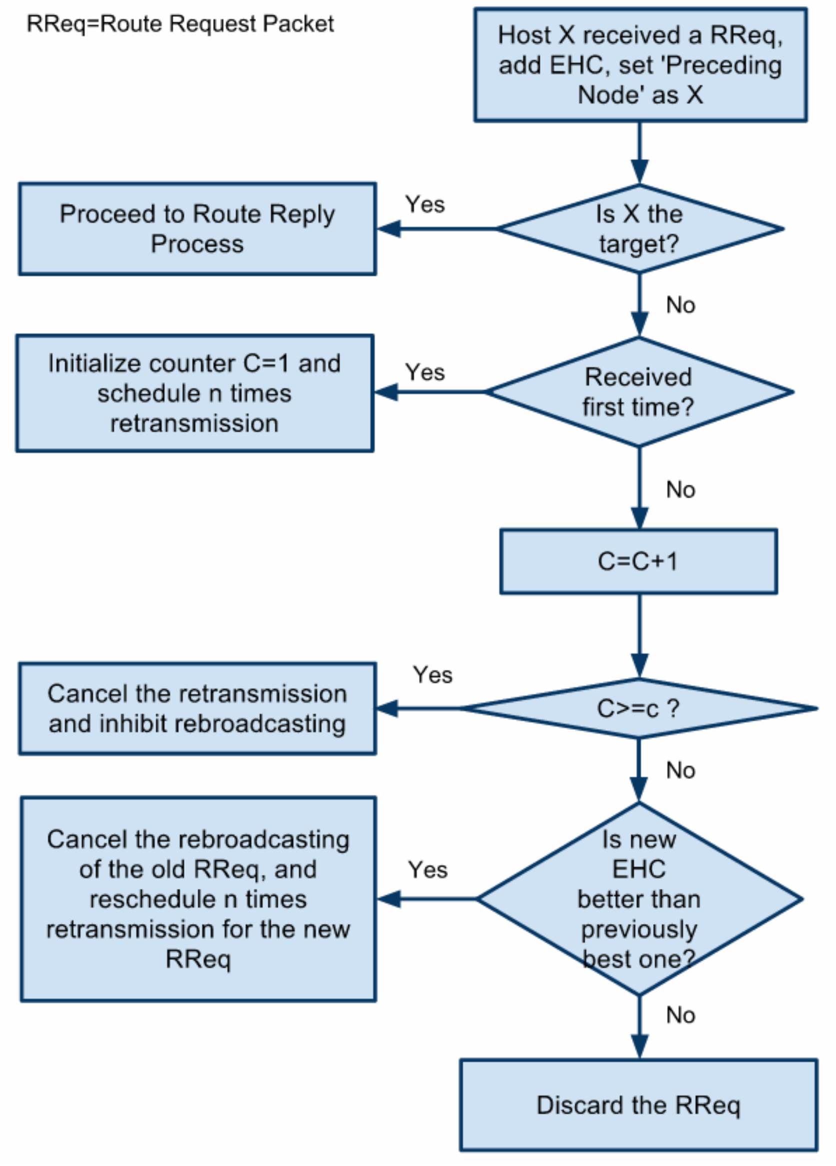


Figure 10 Flowchart of RReq Processing in *nc*RR+

Although both proposed *n*R and *nc*RR schemes could be combined with R1, namely [*n*R, R1]=*n*R1 and [*nc*RR, R1]=*nc*RR1, we consider *n*R+ and *nc*RR+ as main ones for our new schemes because we presume adopting R+ option will help source nodes to discover routes with lower cost. As we mentioned earlier, in the traditional DSR, any received RReq is retransmitted only once, and any other copies of the same RReq than the very first one will be ignored, so it is equivalent to [1R, R1]=1R1. Thus, the proposed *n*R+ and *nc*RR+ will be challenging 1R1 in the simulation.

Chapter 4 Performance Evaluation

4.1 The Simulation

To test the efficiency of the proposed route discovery scheme, a home-made ad hoc network simulator is built in this research. Some existing network simulator, such as OPNET, NS-2, GloMoSim and JSIM are general-purpose network simulators, which target a wide range of network protocols. In [21], the authors reveal that different simulation tools yield quite different results even when they are configured with the same set of protocols, and such differences are mostly derived from different assumptions made at the physical layer. Also in [19], the authors state that countless experimental studies already published cannot be replicated because they do not fully report the conditions in which they were carried out. Furthermore, these powerful and complex simulators have lots of black-box parameter setting that without proper setting can change the simulation results significantly. To have a ‘clear-cut’ simulator with reproducible results, simulation in this work is done by the author’s own ad hoc network simulator which is coded in MATLAB. It is lightweight and easy to modify according to requirements. Most importantly, the design of the simulator is dedicated to comparing behaviors in network layer based on the realistic physical layer. So it mainly focuses on the route discovery process in ad hoc network. Only the behaviors of an ad hoc network in the physical layer and the network layer are simulated, assuming no collisions (messages are sent at different times, without simultaneous network activities) when messages are being transmitted.

Simulation work is performed over static networks. Simulation area for placement of nodes is a l x l square area where l=300, over which 200 nodes (m=200) are generated through algorithm MIN-DPA [22]. *Minimum degree proximity algorithm* (MIN-DPA) aims to distribute node degrees more uniformly while maintaining connectivity. This algorithm generates graphs by placing each new node around the node that has the smallest number of neighbors. According to this algorithm, the transmission radius R is given by $R = \mathrm{sqrt}(dl2/((m\text{-}1)\,\pi\,))$ so that the expected node degree is equal to the desired average degree d..

4.2 Simulation Model

Physical Layer:

A physical layer model is used. After all nodes are generated through MIN-DPA algorithm, the distance vector of each node to all other nodes is calculated. The simulator uses Equation (2.6), to obtain the reception probability $P(x)$ of each node to all of its neighboring nodes. Path loss exponent β is set to 4. In such calculation, x is the distance between each pair of nodes. Neighboring nodes are defined as the nodes in the transmission range of *2R*. Each node makes use of Equation (2.7) to calculate the *EHC* between itself and its neighboring nodes, in which u is dynamically calculated by u=round((1/p(x))-0.1). In any time slot, non-transmitting nodes listen to the channel to make sure there is no other node transmitting messages. They receive message from the transmitting neighbors only if the reception probability $P(x)$ is greater than a randomly generated number r ($0 \leq r \leq 1$).

Network Layer:

The network layer implements the proposed route discovery schemes, *n-retransmission* (*n*R) scheme and *n-retransmission n-reception* (*nc*RR) scheme in Chapter 3. The traditional DSR, equivalent to 1R1, is also implemented for comparison. As we mentioned in Chapter 3, we consider *n*R+ and *nc*RR+ as the main algorithms in the proposed schemes. Meanwhile, all the options during the route reply process discussed in Chapter 3, B* and B+, A1, A3 and A*u* are also implemented with the proposed route discovery schemes. We consider R1, B* and A3 as the main options to evaluate *n*R and *nc*RR.

4.3 Simulation Results

The following Chart 1, 2, and 3 present the simulation results of average EHC, success rate and message overhead during route discovery. The methods being compared are 1R, *n*R for *n*=2, 3, 4, and the RReq and RReq handling options are [R1, B*, A3]. The amount of nodes *m* is tested for *m*=200. The average node degree *d* is tested for *d*=6, 8, 10, 16, 20, 24, 32 and 40, which simulate a network from sparse to dense. The average expected hop count for each combination of parameters *n* and *d* is measured over 30 graphs and 100 successful source-destination route discovery processes (100 successful source-destination routing on each of the 30 graphs generated). The source and destination nodes are chosen randomly for each route discovery process.

From Chart 1 we can see, average EHC decreases as the average network degree *d* increases. The EHC has 0.5% ~ 20% decrease in the proposed *n-retransmission* schemes for different value of *d*. Smaller EHC means better route. However, the results show that retransmitting RReq packets more then 2 times doesn't lower EHC further significantly. Therefore, retransmitting RReq 2 times, namely 2R, should be the best choice. The Chart 2 shows that the success rate is improved greatly when the proposed route discovery scheme applied in the sparse networks. In networks high in density, retransmitting more than one time does not make a lot of difference. Chart 3 shows packages flooded in the network during route discovery are drastically getting more as nodes retransmitting more RReq. And the overhead rises as the average network degree increases.

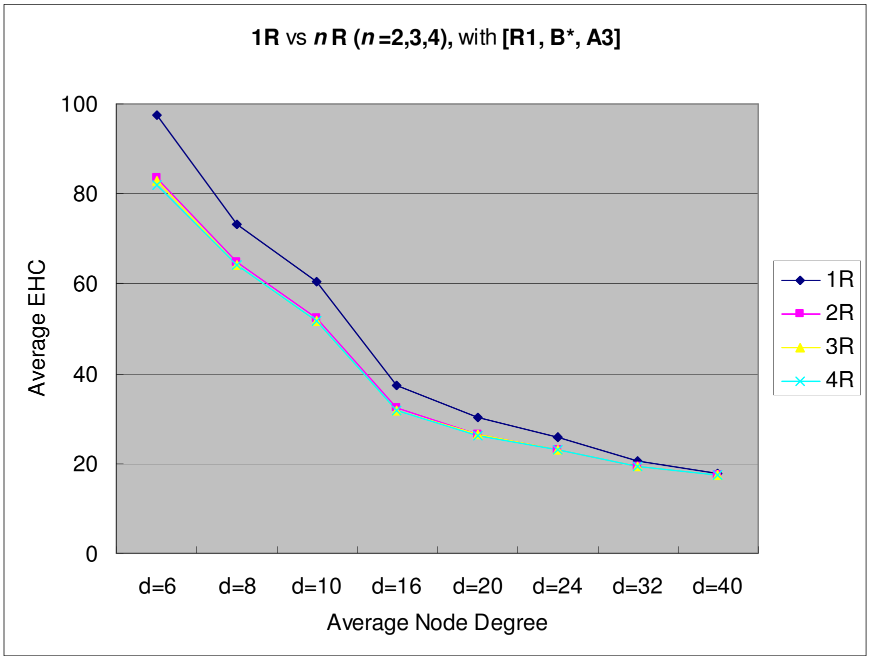


Chart 1 Average EHC between 1R, 2R, 3R and 4R with [R1, B*, A3]

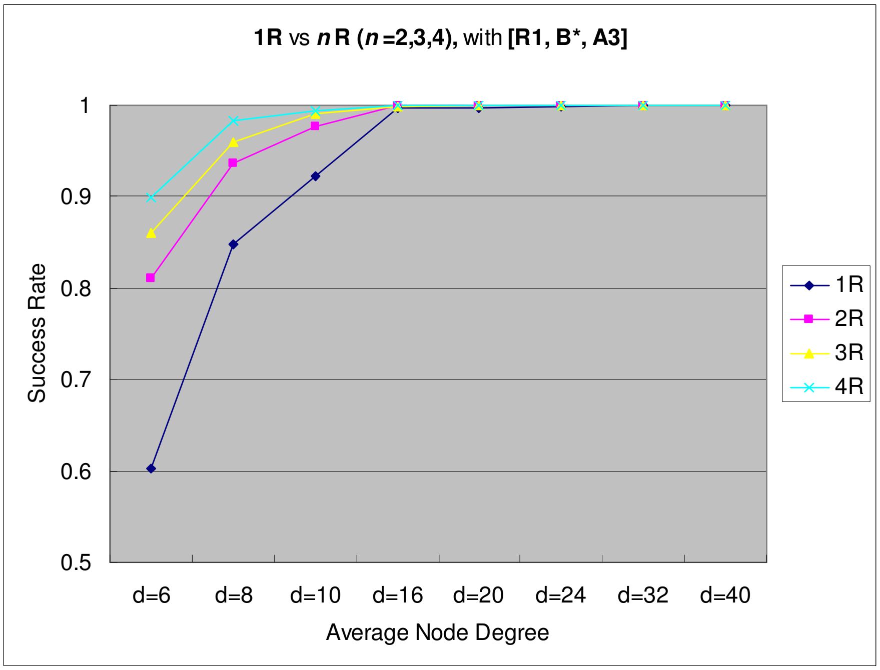


Chart 2 Success Rate between 1R, 2R, 3R and 4R with [R1, B*, A3]

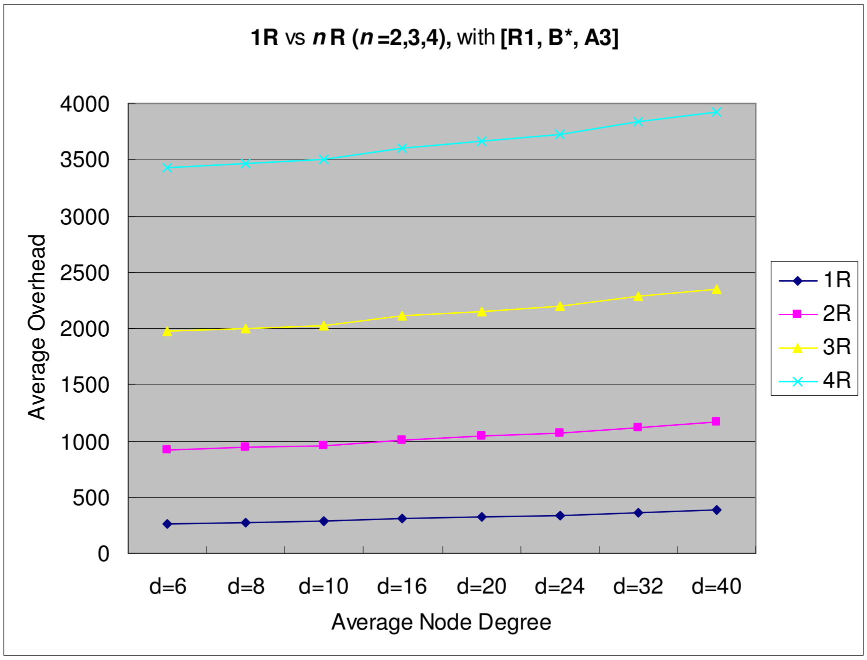


Chart 3 Average Overhead between 1R, 2R, 3R and 4R with [R1, B*, A3]

The first round simulation results let us know *n*R performs better than 1R, and the best value of *n* is 2 for *n*R. However, the simulation only conducted route discovery from one pair source-destination nodes. The message overhead caused by 2R almost got tripled already comparing to 1R. We can imagine how serious the overhead problem would be if multiple route discoveries are involved. We proposed *nc*RR to fix this problem. Based on the simulation results from the first round, we choose 2R, coupling with main options R1, B* and A3 to evaluate *nc*RR, in which case the value of *n* is 2 and the counter threshold for reception is 2, 3, 4, 5 and 6.

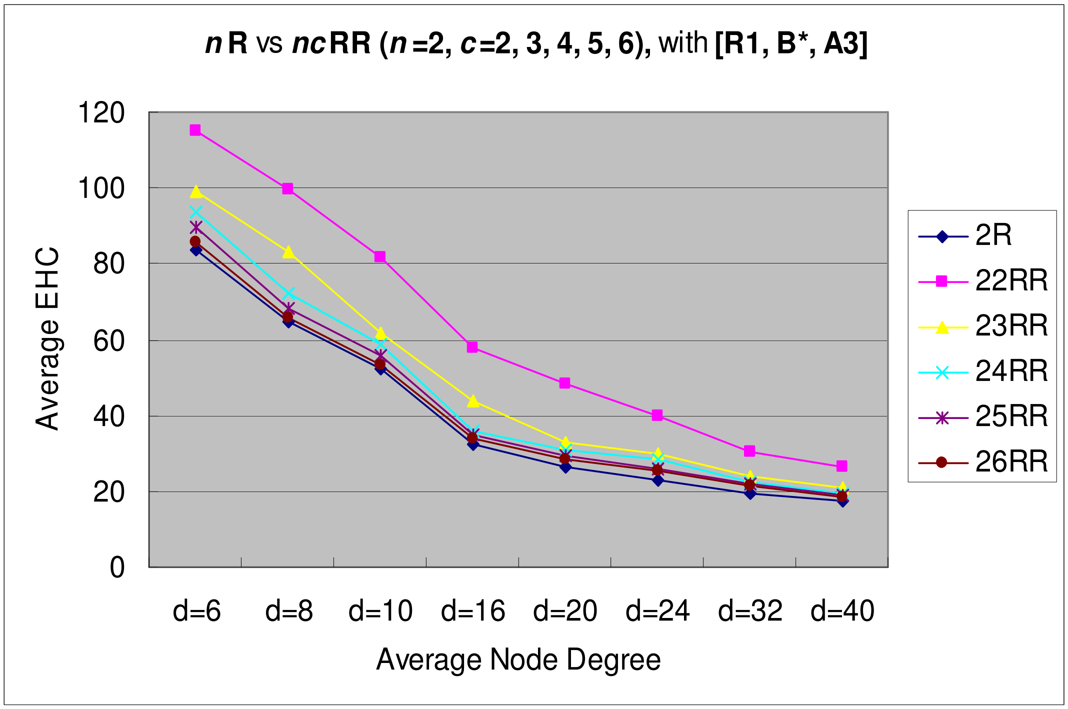


Chart 4 Average EHC between *n*R and *nc*RR(*n*=2, *c*=3,4,5,6) with [R1, B*, A3]

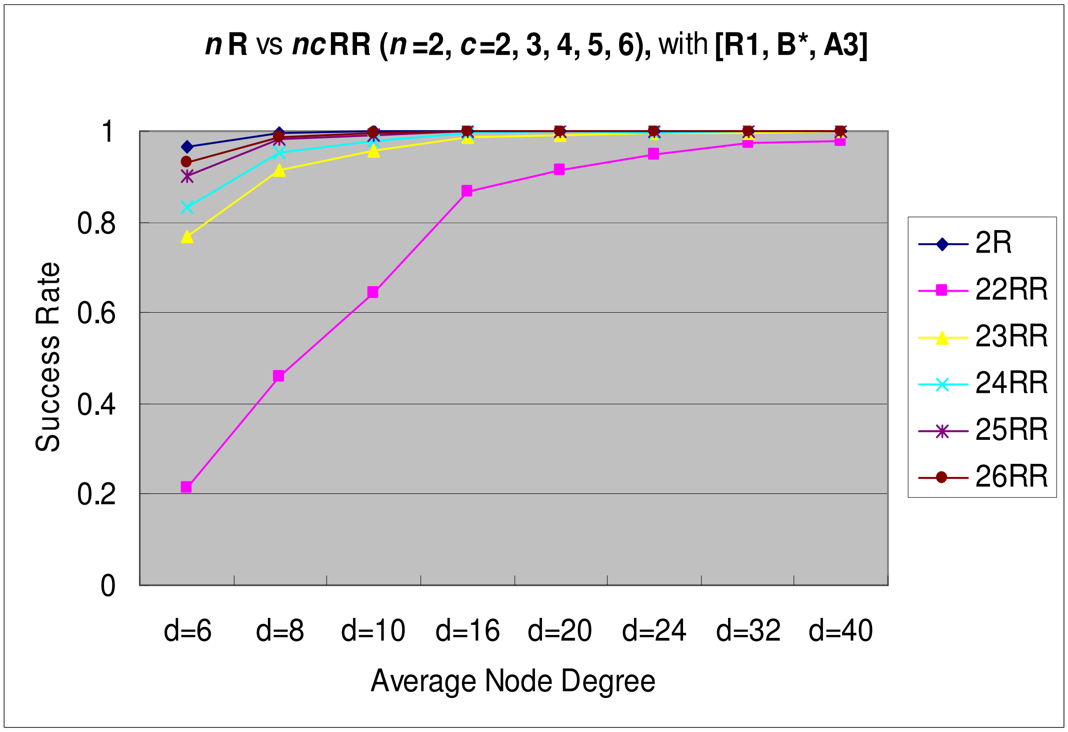


Chart 5 Success Rate between *n*R and *nc*RR(*n*=2, *c*=3,4,5,6) with [R1, B*, A3]

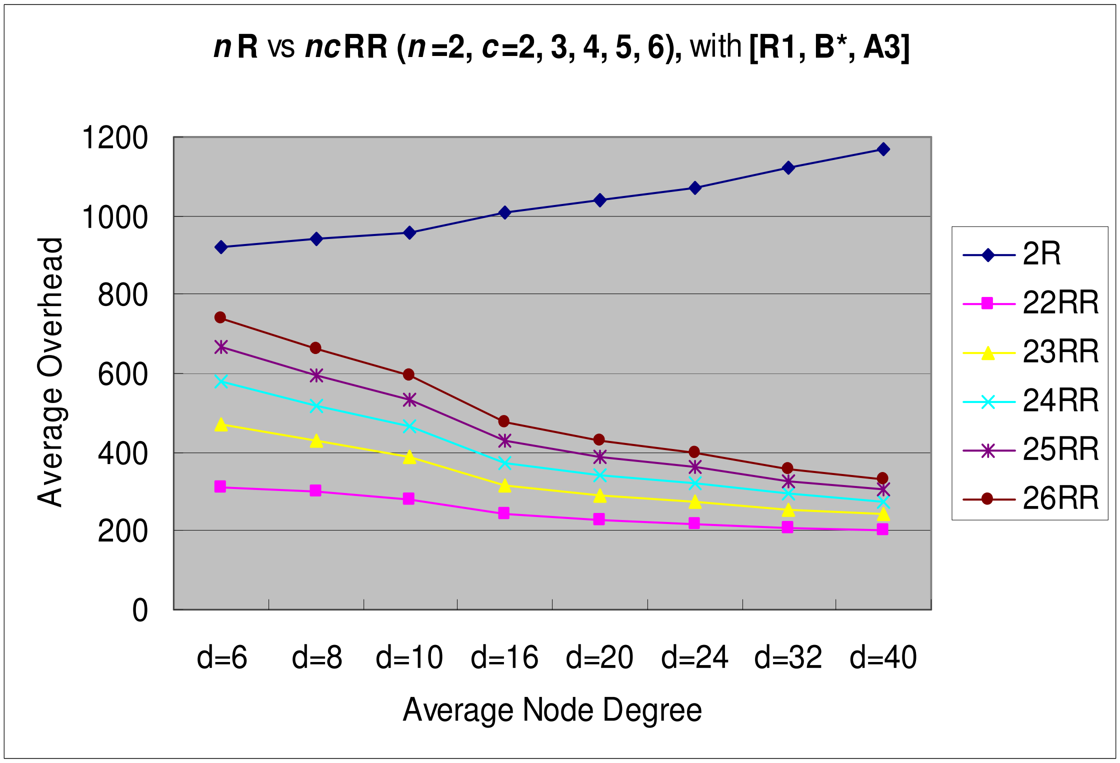


Chart 6 Average Overhead between *n*R and *nc*RR(*n*=2, *c*=3,4,5,6) with [R1, B*, A3]

Chart 4, Chart 5 and Chart 6 respectively present the average EHC, success rate and average overhead of route discovery by applying *n*R and *nc*RR (*n*=2, *c*=2,3,4,5,6), with options [R1, B*, A3]. From the charts we can see the counter threshold has great impact on average EHC and route discovery success rate although the message overhead situation got effectively controlled. The average EHC yielded from 2-retransmission 2-reception scheme, namely 22RR, is quite higher than that from 2R scheme without counter threshold, especially in low density network. But when average node degree *d*>=16 and the threshold *c*>=3, the average EHC and success rate reach the same level as that of 2R. When *c*=2, the average EHC is highly compromised although the overhead is sharply reduced. Overall, the amount of overhead decreases by about 19.6% ~ 79.2% when *c*>=3. According to this simulation result, a threshold *c* of 3 or 4 will be an appropriate choice.

Now let's take a look at the performance comparison between option R1 and R+ while fixing best values of *n* for *n*R. Since we have found out 2R is the best choice for *n*R, then R1 and R+ are compared while the other options are fixed at [2R, B*, A3]. Chart 7 and Chart 8 show the simulation results of average EHC and success rate comparison between option R1 and R+. As we can see, the improvement is quite obvious in sparse networks as presumed. When *d*≤16, the average EHC are lowered by 7.6% to 18.1%. When *d*≤10, the success rates rose by 2.6% to 19.6%. There is no noticeable improvement when networks are high in density. Therefore, retransmitting the same copies of a RReq message that hold better EHC, namely R+, is more likely to find the source node a route with a smaller EHC. This works more effectively in sparse networks.

Chart 9 and Chart 10 are the simulation results to show which option, B* or B+, can yield better average EHC and success rate while the other options are set to [2R, R1, A3]. Chart 11 and Chart 12 are the simulation results to show which option, A1, A3 or Au, can yield better average EHC and success rate while the other options are set to [2R, R1, B*].

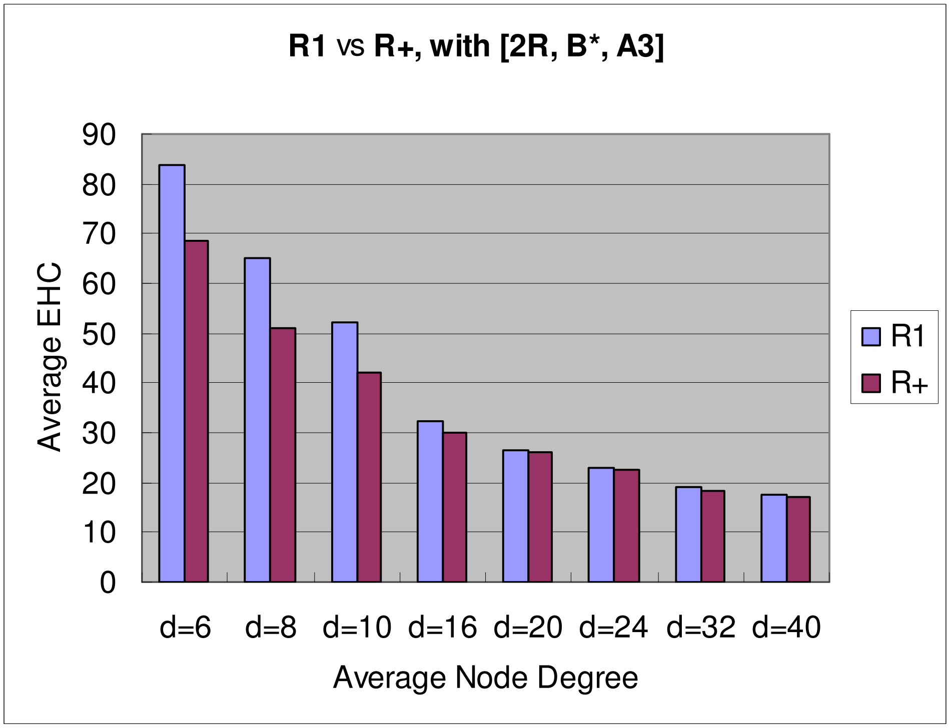


Chart 7 Average EHC between R1 and R+ with [2R, B*, A3]

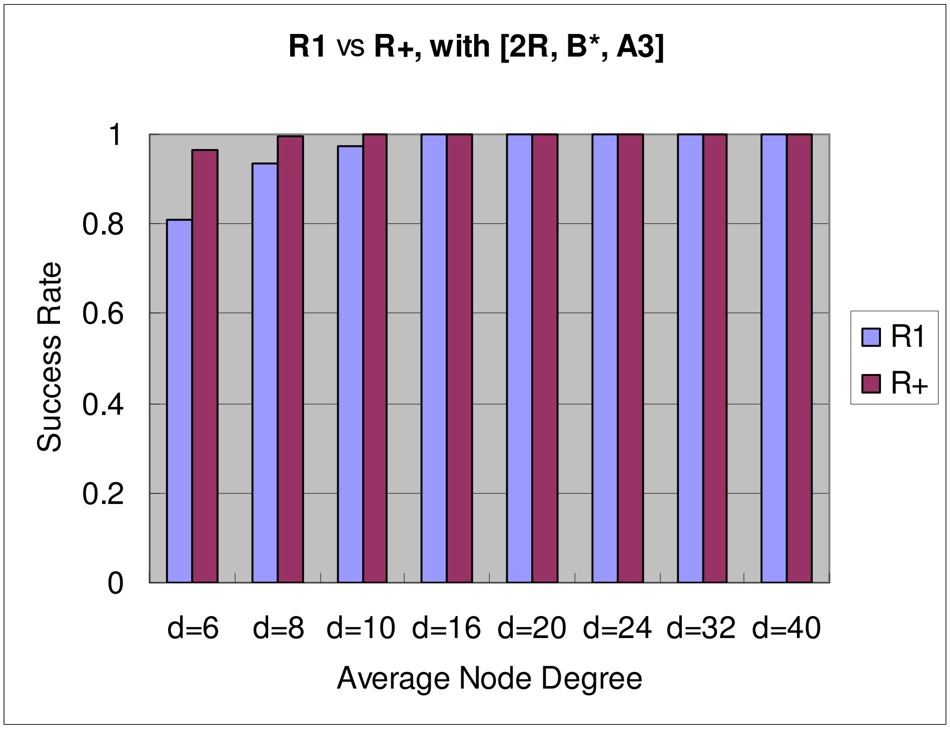


Chart 8 Success Rate between R1 and R+ with [2R, B*, A3]

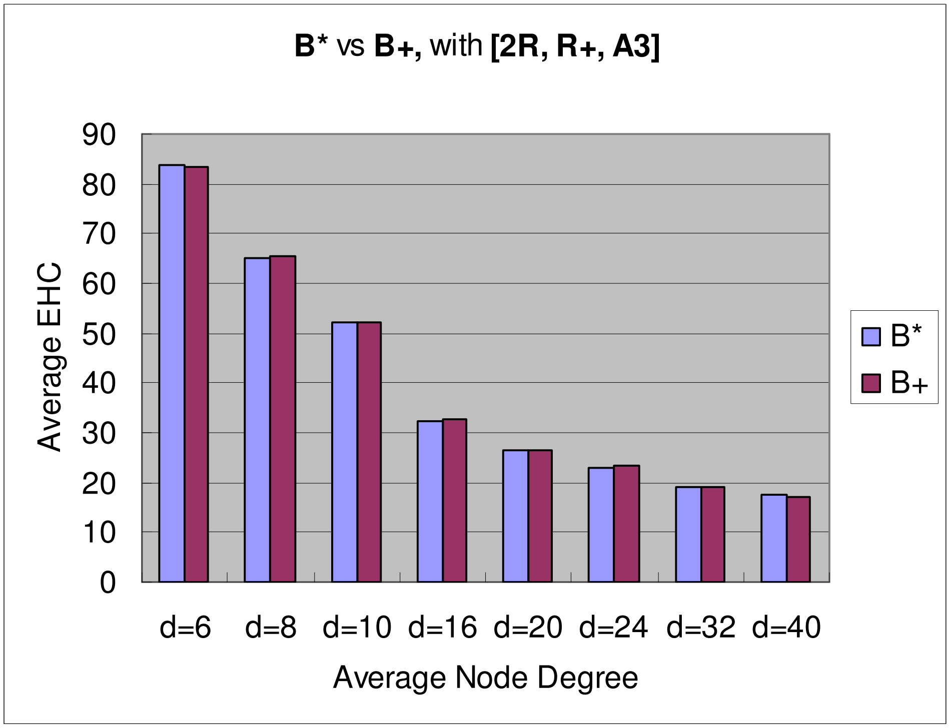


Chart 9 Average EHC between B* and B+ with [2R, R+, A3]

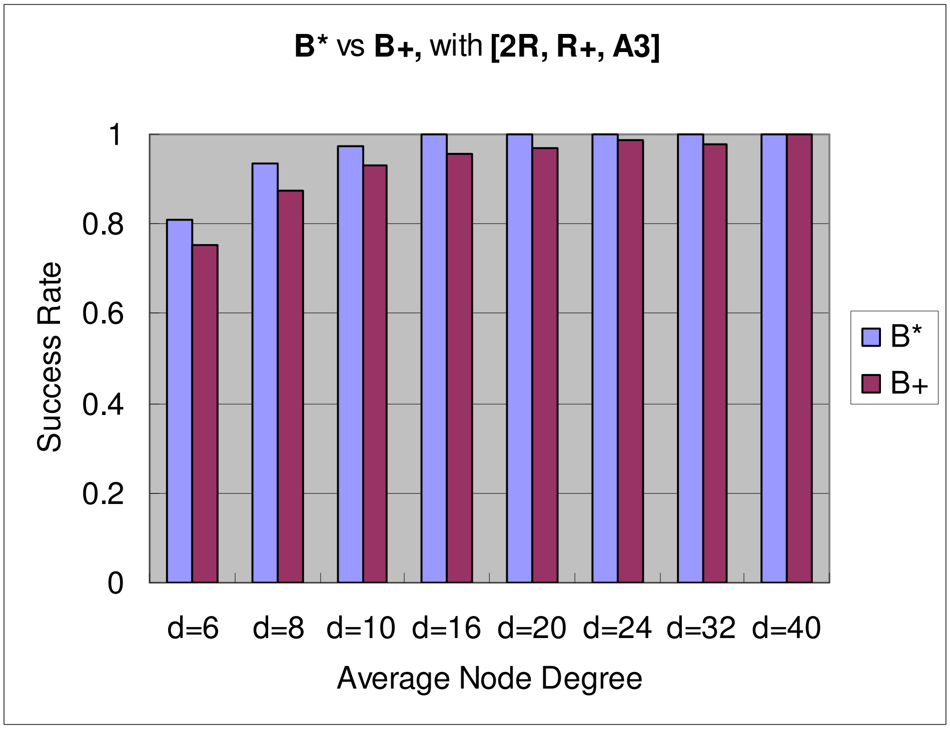


Chart 10 Success Rate between B* and B+ with [2R, R+, A3]

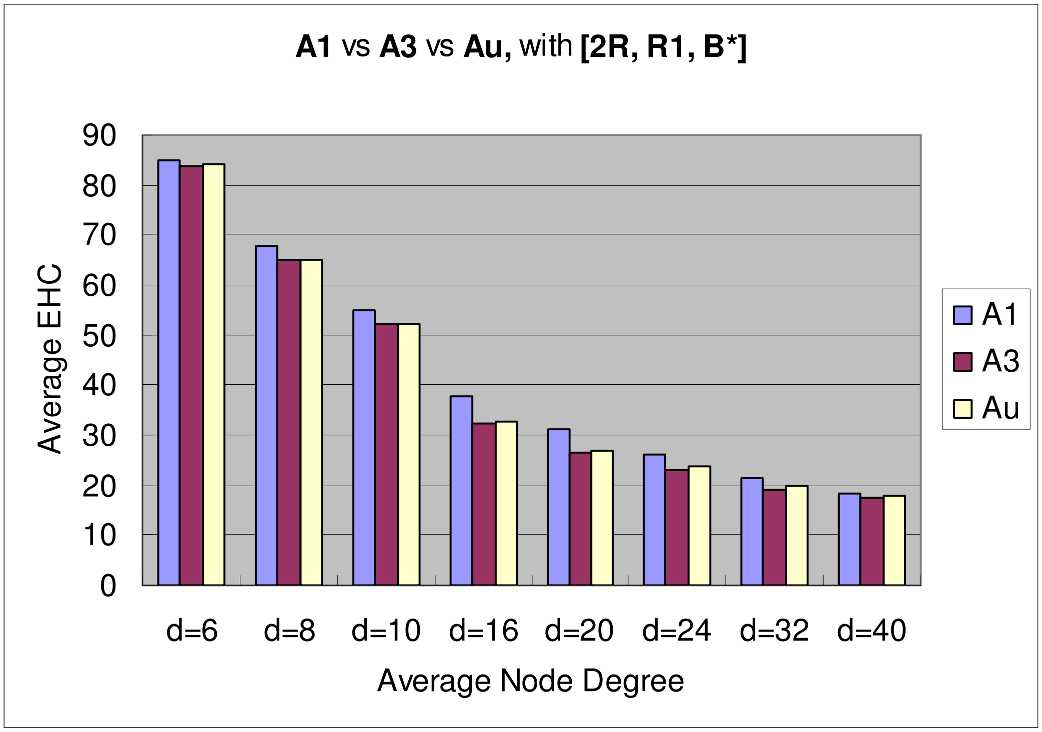


Chart 11 Average EHC between A1, A3 and Au with [2R, R1, B*]

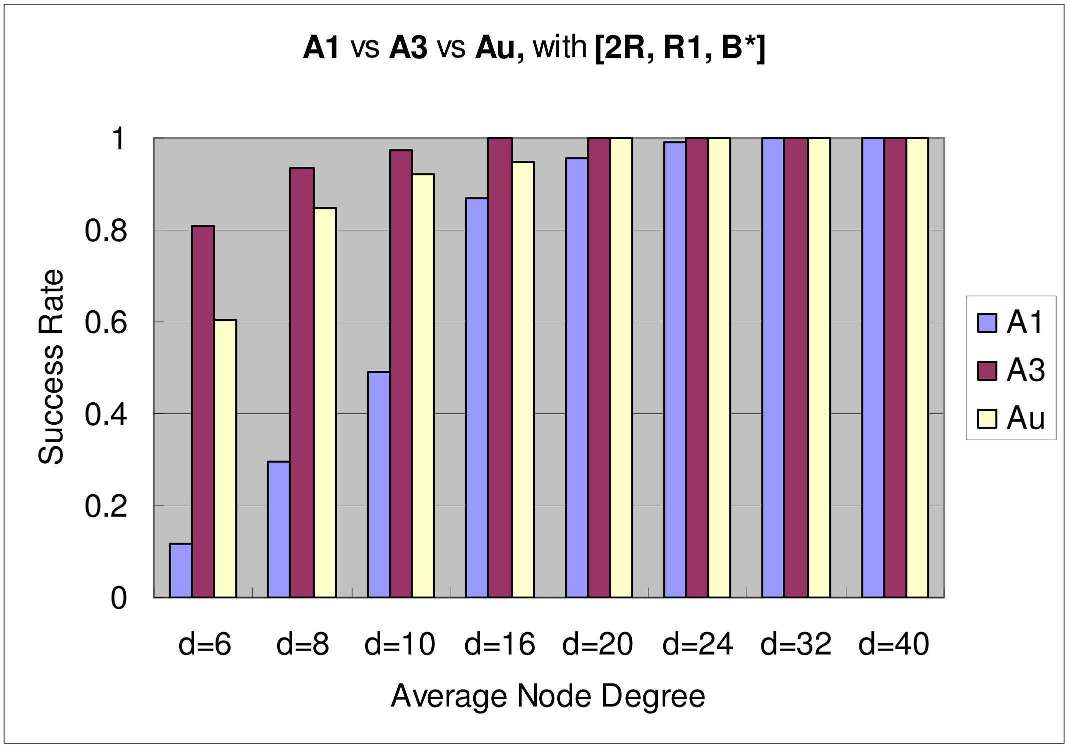


Chart 12 Average EHC Comparison A1, A3 and Au with [2R, R1, B*]

Chart 9 and Chart 10 show that option B+ yields pretty much the same result as B* in terms of average EHC. But the success rate with B+ is slightly lower than B*. Chart 10 and Chart 11 show that option A1 slight raises the average EHC more or less comparing A3 and Au, and A3 and Au have the similar performance on average EHC. However, the success rates among the three alternatives have significant difference. Apparently, retransmitting RRep only once leads to a relatively low success rate, so A1 is the worst scenario, especially when networks are sparse. Au is slightly lower in success rate than A3 in most cases.

Now that the above simulation results have shown 2R is the best choice for *n*R and R+ is superior to R1, we may want to compare the proposed *n*R+ with 1R1 which is equivalent to DSR. Chart 13 and Chart 14 present the simulation results of 1R1 vs nR+ while the other options are [B*, A3]. The improvement of average EHC is about 0.9% ~ 31%, and success rate is raised up to about 58% higher.

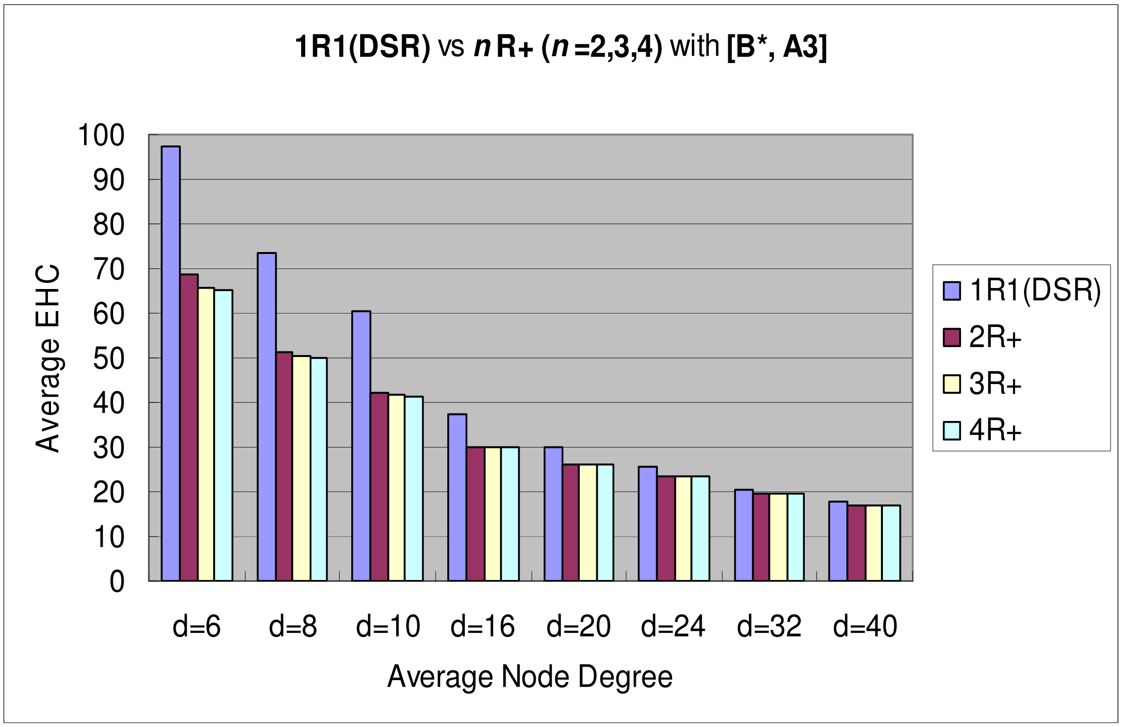


Chart 13 Average EHC between 1R1(DSR) and *n*R+ with [B*, A3]

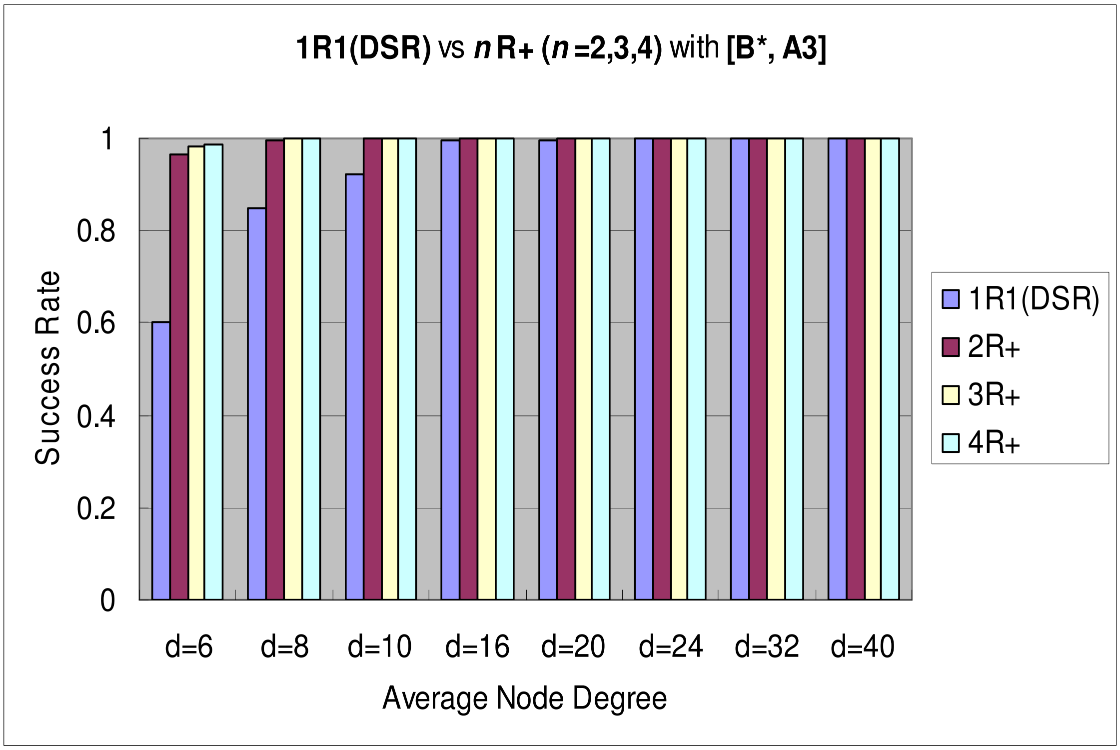


Chart 14 Success Rate between 1R1(DSR) and *n*R+ with [B*, A3]

Chart 15 presents the simulation result of route discovery in a nearly partitioned network. The near-partitioned network is created as follow: a network with 150 nodes is created in the same method as before. To make a partition in the network, nodes in the area of (*R*/2±150)*300, i.e. the stripe with the dimension of *R**300 along the centerline, are removed. The generated graphs have two sub-networks on the left and right sides, nodes in sub-networks are apart at least with distance *R*. The source and destination nodes are chosen randomly in the left and right sub-networks, respectively; a successful source-destination route discovery has to cross the partition zone. The number of nodes in the partitioned network, therefore, is always equal to or less than 150. Also, 30 graphs are generated. On each generated graph, 100 route discovery attempts with different schemes, namely DSR [1R1, B*, A3], 2-retransmission [2R+, B*, A3], 3-retransmision [4R+, B*, A3], and 4-retransmission [4R+, B*, A3], are performed on the same graph. The success rate is the percentage of the number of times RReq reaches destination node to the whole routing attempts (30*100=3000 attempts). In *n-retransmission* schemes, destination nodes send back route reply packets in the way that was discussed in option B* and A1.

The result shows, comparing DSR and 2R+ scheme in nearly partitioned networks, there is significant improvement in route discovery success rate. However, more retransmissions only have minor increase in the route discovery success rate while the overhead rise rapidly.

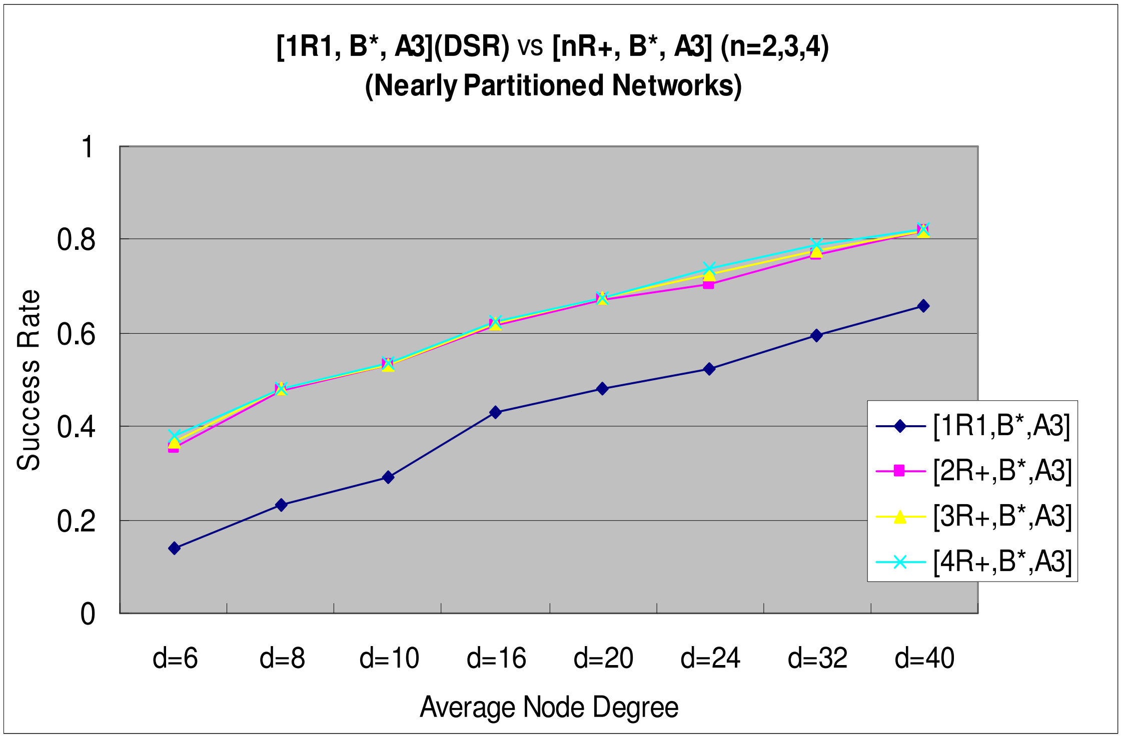


Chart 15 Success Rate in Nearly Partitioned Networks

As we can see, retransmitting RReq more than one time could boost the route discovery success rate from about 24% to 106% when the networks are nearly partitioned.

4.4 Summary

The overall simulation results verify that under the stated assumptions, the proposed route discovery scheme is superior to existing ones, in such that:

- The proposed *n-retransmission* scheme significantly decreases the expected hop count, i.e. better routes are discovered by such schemes.
- Simple *n-retransmission* scheme causes a great deal of overhead in the network.
- *N-retransmission c-reception* scheme solves the problem of extra overhead in the price of a slightly increase in the expected hop count level.
- The optimal value for retransmission, found heuristically, should be 2 with significant decrease in EHC and minor increase in overhead.
- The value for counter threshold *c* should be 3 or 4, in which case the overhead is reduced and meanwhile the average EHC is not compromised a lot. It is also in accordance with the recommended value in [20].
- The *n-retransmission* scheme significantly raises the route discovery success rate in nearly partitioned networks.

Chapter 5 Conclusion and Future Work

Most existing routing protocols in ad hoc network assume an ideal physical layer model. However, recent study shows that the packet reception probability P(x) decreases from 1 to 0 gradually according to the distance of the receiver from the transmitter. Furthermore, there is a transition area around P(x) = 0.5. Routing protocols based on ideal physical layer model have problems in simulation and implementation when the idea physical layer model is replaced by a more realistic one. The *Log-Normal Shadowing* model is a realistic physical layer model. Expected Hop Count based on this model is a better weight for routing decision than the traditional hop count.

Several existing reactive routing schemes using realistic physical layer model continue to apply only one retransmission by each node. The fact is that only one communication is very likely to miss an important neighbor. Also in such schemes, a distance neighbor may accidentally receive the route discovery message and use it to announce a new route, although in reality it is not able to provide such a good service for the real traffic.

The proposed *n-retransmission* route discovery scheme and *n-retransmission c-reception* route discovery scheme retransmit the RReq n times rather than one time. In *n-retransmission* scheme, nodes keep retransmitting the route discovery message for a certain amount of times, e.g. 2, 3, 4… times. Each node restarts the retransmission process if it receives a route discovery message with a better EHC.

In *n-retransmission c-reception* scheme, nodes retransmit the route discovery message for a certain amount of times. When retransmitting, every node in the network listens to its neighbors. If the threshold amount of the same route discovery message has been received, it terminates the retransmission process.

The simulation results from both schemes are very encouraging. The EHC from the proposed *n-retransmission* scheme in most network density conditions are better than the existing routing schemes. Also it shows significant rise in route discovery success rate comparing to the traditional route discovery scheme in nearly partitioned networks.

The *n-retransmission c-reception* scheme has relatively effective control on overhead caused by extra retransmissions. The simulation results show the traffic in such scheme is reduced from about 19.6% ~ 79.2% when c>=3, while the EHC value is only slightly compromised.

The experimental results show that the best value for the amount of retransmission is 2 in all proposed schemes. The best value of reception threshold c in *nc*RR scheme is 3 or 4. Therefore, a 2-retransmission scheme with the reception threshold set to 3 or 4 is the best one among all the proposed schemes in this research.

A more realistic MAC layer can be implemented in the simulator to test if the proposed two *n-retransmission* route discovery schemes are still able to yield better results. In that case, the full feature of IEEE 802.11 behavior, with delay decided by a random function, can be simulated. Also one can try to use a wait function depending on some evaluation, instead of random delay.

Further options are to consider use variable bit rate which is already existed in IEEE 802.11 standard and its implementations. In that way, instead of EHC, other metric such as Medium Time Metric (MTM) in [18] may be used.

Our counter based retransmission scheme is based on two parameters: either exceeding number of retransmissions *n* or number of received RReq copies *c*. It is possible to consider also a version with a single parameter, n. In *counter-based n-retransmission* (*n*CB) scheme, node is retransmitting RReq until a total of *n* own or retransmissions from neighbors are heard. Thus in *n*CB, once a node decides to retransmit, it will do so until the number of its retransmissions, or the number of received copies of same RReq, reaches *n*. 1CB means no retransmissions at all, since one copy of message was already received. In case of 2CB, node will either make one retransmission, or will make no retransmission at all if, while waiting for its own scheduled retransmission to occur, it does receive yet another, second, copy of the same RReq. In case of 3CB, node will either make 2 retransmissions, or hear two more copies of same RReq, or make one retransmission and hear one more copy of the same RReq. We left the study of *n*CB for future work.

References

[1] http://www.ietf.org

[2] Charles E. Perkins, Pravin Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers”. Proceedings of the Conference on Communications Architectures, Protocols and Applications, pages 234-244, London, England, August 1994.

[3] John Jubin and Janet D. Tornow, “The DARPA Packet Radio network Protocols”. Proceedings of IEEE, 75(1), pages 21-32, January 1987.

[4] David B. Johnson and David A. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks”. In Mobile Computing, edited by Tomas Imielinski and Hank Korth, Chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[5] Charles E. Perkins and Elizabeth M. Royer, “Ad-hoc On-Demand Distance Vector Routing”. Proceedings of the 2nd IEEE Workshop on Mobile Computing systems and Applications, pages 90-100, New Orleans, LA, February 1999.

[6] Vincent D. Park, M. Scott Corson, “A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks”. Proceedings of the IEEE Conference on Computer Communications, pages 1405-1413, Kobe, Japan, April 1997.

[7] Chai-Keong Toh, “Associativity-Based routing for Ad-hoc Mobile Networks”, Wireless Personal Communications Journal, Special Issue on Mobile Networking and Computing Systems, pages 103-139, vol.4, no.2, March 1997.

[8] Rohit Dube, Cynthia D. Rais, Kuang-Yeh Wang and Satish K. Tripathi, “Signal Stability-Based Adaptive routing (SSA) for Ad-hoc Networks”. IEEE Personal Communications, pages 36-45, February 1997.

[9] Ching-Chuan Chiang and Mario Gerla, “Routing in Clustered Multihop, Mobiles Wireless Networks”. Proceedings of IEEE Singapore International Conference on Networks, pages 197-211, Singapore, April 1997.

[10] Shree Murthy and J.J. Garcia-Luna-Aceves, “An Efficient Routing Protocol for Wireless Networks”, ACM Mobile Networks and Applications 1, Special Issue on Routing in Mobile Communication Networks, pages 183-197, Oct. 1995.

[11] P. Sammer, M. Pearlman, S. Haas, “Independent Zone Routing: an Adaptive Hybrid Routing Framework for Ad-hoc Wireless Networks”, IEEE/ACM Transactions on Networking, 12(4), pages 595-608, 2004.

[12] G. Pei, M. Gerla, and T.-W. Chen, “Fisheye State Routing in Mobile Ad-hoc Networks”, Proceedings of ICDCS Workshop on Wireless Networks and Mobile Compution, pages D71-D78, 2000.

[13] Marco Zuniga, Bhaskar Krishnamachari, "Analyzing the Transitional Region in Low Power Wireless Links", First IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON), Santa Clara, CA, October 2004.

[14] L. Quin, T. Kunz, "On-demand Routing in MANETs: The Impact of a Realistic Physical Layer Model," Proceedings of the Second International Conference ADHOC-NOW, 2003, pp. 37-48

[15] Ivan Stojmenovic, Amiya Nayak and Johnson Kuruvila, "Design guidelines for routing protocols in ad hoc and sensor networks with a realistic physical layer", IEEE Communications Magazine (Ad Hoc and Sensor Networks Series), 43, 3, March 2005, 101-106

[16] Stojmenovic I., A. Nayak, J. Kuruvila, F. Ovalle-Martinez, E. Villanueva-Pena, "Physical layer impact on the design and performance of routing and broadcasting protocols in ad hoc and sensor networks", Computer Communications, Volume 28, Number 10, June 2005, 1138-1151

[17] Nishith Goel, Kalai Kalaichelvan, Eric Karmouch, Amiya Nayak, Ivan Stojmanovic, Eduardo Villanueva-Pena, "Physical layer impact on finding local knowledge information in ad hoc and sensor networks", International Journal of Computational Science, Volume 2, Number 2, April 2008, 233-249

[18] B. Awerbuch, D. Holmer, H.Rubens, "High Throughput Route Selection in Multi-Rate Ad Hoc Wireless Networks," Johns Hopkins U. Technical Report, March 2003

[19] Luiz F. Perrone, yougu Yuan, "Modeling and simulation best practices for wireless ad hoc networks". In Proc. of the 2003 Winter Simulation Conference, pp. 685-693, 2003.

[20] S. Ni, Y. Tseng, Y. Chen, and J. Sheu. "The broadcast storm problem in a mobile ad hoc network." In ACM MOBICOM '99, August 1999

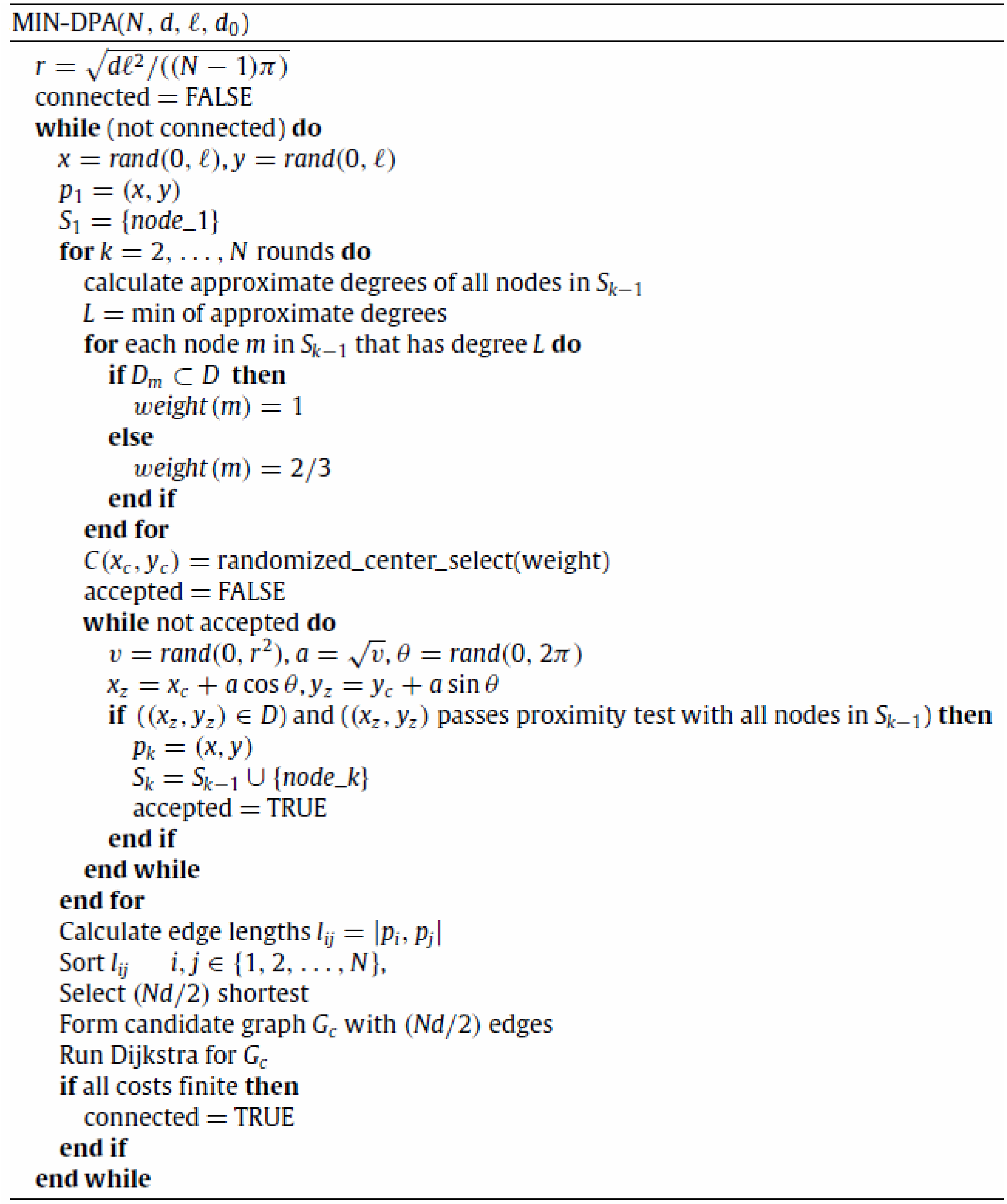
[21] M. Takai, J. Martin, and R. Bagrodia. "Effects of wireless physical layer modeling in mobile ad hoc networks," ACM int. Symp. On Mobile Ad hopc Networking and Computing (MobiHoc), Oct 2001

[22] Furuzan Atay Onat, Ivan Stojmenovic, Halim Yanikomeroglu, "Generating Random Graphs for the Simulation of Wireless Ad-hoc, Actuator, Sensor, and Internet Networks", Pervasive and Mobile Compputing 4, pages 597-615, 2008.

[23] Elizabeth M. Royer and Chai-Keong Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", IEEE Personal Communications Magazine, pages 46-55, April 1999.

Appendix MIN-DPA Algorithm

All the networks (graphs) in this simulation are generated through Minimum Degree Proximity Algorithm (MIN-DPA). The following are the procedure of this algorithm.



```
MIN-DPA($N$, $d$, $\ell$, $d_0$)
r = $\sqrt{d\ell^2/((N-1)\pi)}$
connected = FALSE
while (not connected) do
  $x = rand(0, \ell), y = rand(0, \ell)$
  $p_1 = (x, y)$
  $S_1 = \{node\_1\}$
  for $k = 2, \ldots, N$ rounds do
    calculate approximate degrees of all nodes in $S_{k-1}$
    $L$ = min of approximate degrees
    for each node $m$ in $S_{k-1}$ that has degree $L$ do
      if $D_m \subset D$ then
        $weight(m) = 1$
      else
        $weight(m) = 2/3$
      end if
    end for
    $C(x_c, y_c)$ = randomized_center_select(weight)
    accepted = FALSE
    while not accepted do
      $v = rand(0, r^2), a = \sqrt{v}, \theta = rand(0, 2\pi)$
      $x_z = x_c + a\cos\theta, y_z = y_c + a\sin\theta$
      if $((x_z, y_z) \in D)$ and $((x_z, y_z))$ passes proximity test with all nodes in $S_{k-1}$) then
        $p_k = (x, y)$
        $S_k = S_{k-1} \cup \{node\_k\}$
        accepted = TRUE
      end if
    end while
  end for
  Calculate edge lengths $l_{ij} = |p_i, p_j|$
  Sort $l_{ij}$    $i, j \in \{1, 2, \ldots, N\}$,
  Select $(Nd/2)$ shortest
  Form candidate graph $G_c$ with $(Nd/2)$ edges
  Run Dijkstra for $G_c$
  if all costs finite then
    connected = TRUE
  end if
end while
```