AALTO UNIVERSITY

Delay Tolerant Networks

Assignment 2: Routing Protocols

Robin Babujee Jerome (335623), Bastian Arjun Shajit (338303)

01-Dec-13

1. OBJECTIVE

The objective of this assignment is to study and analyse the DTN routing protocols (such as Spray & Wait, Epidemic, Direct Delivery and PROPHET) in two scenarios, namely, urban area and simple plain simulation area. The work also includes novel designs for two routing protocols, their behavioural analysis and performance for the two scenarios mentioned above, respectively.

The performance analysis shows the following characteristics:

- Delivery ratio vs. Offered load
- Message delivery delay vs. Offered load

2. SCENARIOS

2.1. Plain Area Scenario Overview

In scenario 1, the simulation is run for plain area with no geographical restrictions. The movement model used here is Gauss Markov mobility model. In brief, this model is a temporally dependent mobility model where the new velocity & direction (v, d) of the node depends on the previous (v,d) by a memory level parameter. Hence in an area with no obstacles each node is allowed to choose its path and speed randomly. This model has a total of 50 nodes, each with a memory factor = 0.5.

2.2. Urban Scenario Overview

Scenario 2 provides Helsinki city scenario settings to run the urban area scenario. The movement model used for hosts in this scenario is Shortest Path Map Based. The whole simulation area is divided into 4 districts and hosts do not travel between districts. Bundle transport between the districts is provided by 4 tram lines.

In total there are 176 hosts which include:

- 16 trams running on 4 tram lines
- 40 hosts in each of the 4 districts (c,e,w,sw)

The tram groups have much higher velocity when compared to the other groups. They have a bigger buffer size of 1G when compared to 30M of the other normal hosts. These trams can be considered as data mules. These trams travel along fixed path and their motion can be considered as close to periodic motion. The other hosts move haphazardly in their regions and hence their motion cannot be considered as periodic or predictable. However if a host meets another host which is restricted to its region, there is a good chance that these two hosts will come in contact with each other yet again.

3. PERFORMANCE EVALUATION

3.1. Plain Area Scenario

In this section we will be studying the performance of different routing algorithm that is implemented in the nodes (which follow a Gauss Markov mobility pattern) for routing messages.

The performance of each of the routing algorithm was analyzed by subjecting them to various load conditions in the area. The consolidated performance report for all the scenarios can be found in Table 1.

3.1.1. Direct Delivery

In DD router, the message reaches the destination only when the origin node meets the destination node. Hence it is evident from the underlying principle that the hop count average should be 1 and since there is absolutely no transfer of messages between the origin node and any other non-destination node, there is absolutely no overhead in the network, hence this algorithm has the least overhead compared to all other routing algorithms. Due to the specific property of DD routers, the latency or the time taken by the message to get delivered to the destination is very high compared to all the other routing algorithms. Any node is allowed to have a specific buffer size of around 30M as per the configuration file and as more messages remain undelivered in the node, the more the message will likely be dropped thus reducing the likelihood of delivering the message to the destination. Due to this characteristic of Direct Delivery method, it has lower delivery probability compared to Spray ‘n Wait, Prophet but however is slightly better than Epidemic router since it will only deliver the message on contacting the destination and this may or may not happen within the simulation period. Also it is evident that as the load on the network decreases, the amount of messages that need to be delivered decreases, hence the likelihood of dropping the message due to filled buffer also decreases. This can be seen from the statistics that the dropped rate is 0 for DD router as the number of messages in the network is greatly reduced, but this also leads to increased latency in delivering the message. DD router is thus a very rudimentary routing algorithm with the lowest delivery probability, highest latency and buffer time.

3.1.2. Epidemic Router

The main feature of epidemic router is that it indiscriminately exchanges all messages in its buffer with the next contact node. This uncontrolled spreading of messages can have several negative effects; it greatly increases the number of copies of the same message in the network, increases the total number of messages in the buffer drastically and hence leads to increased drop in messages. This drop rate reduces drastically as the total load in the network also reduces. The load increase in the network is represented by the total number of new messages created in the network. Due to the above mentioned drawbacks, the epidemic router has the least delivery probability compared to all other routers because of the huge drop ratio seen from increased messages in a small buffer. However this probability increases with fewer new message load in the network due to the reduced drop rate, but is still lesser than all the other routers for the above mentioned reason. We can also conclude from epidemic’s behaviour that it should have one of the lowest latency, but unfortunately the increased number of copies of a particular message in the network might cause the node to drop a message which was supposed to be delivered to the next node which it might have come in contact with, thus increasing the delay of delivering the message. Compared to other routing algorithms, epidemic has a fairly lower latency but is still higher than Spray ‘n Wait.

3.1.3. Spray ‘n Wait

One of the biggest advantages of spray and wait is its ability to reduce the number of copies per message in the network. This intuitively curbs the drawbacks seen in the Epidemic

router which suffers from huge drop rate. Spray and Wait router, maintains a total of ‘n’ copies in the entire network and like Epidemic , transfers one copy of a packet or ‘n/2’ packets to the next host it comes in contact with. This ability to spray and wait for the message to reach its destination and also the ability to reduce the copies per messages are some of the biggest advantages of S’nW which takes care of the drawback seen in Epidemic. Hence with very low drop rate and a low over-head in the network (though it is not the least), the messages tend to remain in the buffer for longer time, thus avoiding unnecessary drops which allows the spray and wait to have the highest delivery and the lowest latency among all the other algorithms. It can be seen from the observation that the drop rate of S’nW decreases drastically as the number of new messages in the network reduces thus boosting the delivery probability with reduce load. It can also be seen that Spray ‘n Wait provides the best performance with decreased overhead, lesser drop rate and reduced latency in delivering the message to destination.

3.1.4. Prophet

Prophet is a probabilistic routing algorithm, which estimates the probability of meeting the same node based on how frequently a particular node has come in contact with the destination node and also on an aging factor for that probability. This notable characteristic of Prophet Router (is actually a mechanism to control the number of copies per message in the network), which allows it to give a copy of the message to the next node, only if its probability of meeting the destination node is higher than itself. This controlled mechanism is the reason why prophet router adds lower overhead to the network compared to Spray N wait and Epidemic. Now, this probabilistic assumption holds well only if the node actually follows a fixed pattern of movement. But a Gauss Markov model, is a completely random model with a memory factor of 0.5 (as per the configuration), which completely nullifies this great algorithm’s assumptions and hence its delivery probability is lesser compared to Spray ‘n Wait. Due to the low overhead in the network, the buffer time of the message also increases slightly more than epidemic router, but the time taken to or the latency in delivering the message to the destination is greatly reduced because of the unrealistic probabilistic assumption in a completely random environment. However as the load in the network decreases, the probability of delivery increases (but is still lesser than Spray ‘n Wait) and the average latency also increases as there are fewer messages to pass around for delivery. Next to Direct delivery mechanism, PROPHET has the highest latency while delivering the messages.

3.2. Urban Area Scenario

The first step towards designing an efficient routing protocol for Urban Area Scenario with high delivery probability, low overhead, low latency etc, is to analyze the performance of existing routing protocols that were bundled along with the ONE simulator. The consolidated performance report for the urban scenario can be found in Table 2.

3.2.1. Direct Delivery

Since direct delivery routers transfer packets only to the final destination, it was clear that this routing algorithm would be a bad approach for a city scenario where hosts are restricted to move within their region. In such a constrained routing algorithm, a packet generated at a host in a particular region will never be delivered to a destination host in another disconnected region. Also this meant that for those packets which eventually get delivered, the average hop-count and the median of hop count will be 1. This protocol also implies that the overhead ratio will always

be 0. With regards to delivery probability this router has the lowest delivery probability as well, which is quite intuitive from the above mentioned reasons. The latency of this router is higher than that of Prophet and Spray 'n Wait. The reason for this is that both Prophet & Spray 'n Wait algorithms try to spread packets in an intelligent way. Prophet on one hand only gives packets to those hosts who have a higher probability of meeting the destination host than the current host carrying the packet. Spray 'n Wait, on the other hand tries to spread packets keeping the total number of packets in the network under check. Since the number of packets in the network is kept under check, there are lesser number of buffer overflows and packet drops. Epidemic Router on the other hand has higher latency than direct delivery router. This could be because epidemic tries to spread packets in an epidemic fashion and thus queues will remain close to full and more packets will be dropped and unwanted latency is introduced because of this.

3.2.2. Epidemic Router

Since epidemic routing makes very few assumptions about the underlying connectivity of the network, there is a trade-off between message delivery and resource usage. Since each of the hosts have a fixed buffer space, epidemic routing would not be a good routing protocol for the city scenario under consideration. Yet another reason why this approach might not be the best is that the buffer sizes of nodes vary highly (from 30M for pedestrians to 1G for trams). So when a tram host (with higher mobility), meets a pedestrian host, it transfers even those packets which the pedestrian will never be able to deliver to the final destination. This could also result in buffer overflow at pedestrian hosts and the dropping of critical messages.

The results speak for these assumptions as epidemic routers have the highest number of started packet transfers and aborted/dropped packets. The delivery probability is higher than only the Direct Delivery router. The number of packets delivered also follows the same trend as delivery probability. The overhead ratio is also the highest in the lot. Since these packets are very often dropped and un-necessarily transferred, the average hop-count, median hop-count as well as latency are the highest among all routing algorithms that were analyzed.

3.2.3. Spray N Wait (S'nW)

This routing algorithm limits resource usage by setting an explicit maximum number of copies and hence should intuitively perform better than Epidemic router in all aspects. This routing algorithm performs better than the two algorithms mentioned above and its performance can be compared with that of Prophet Router. When the load on the network is higher (when interval is 5-10), S'nW algorithm outperforms the prophet router and as the number of packets generated becomes lower (when interval is 30-70), the performance of this protocol slips behind the Prophet routing algorithm. On all cases, Prophet starts much more packet transfers when compared to S'nW. The number of packets aborted and dropped in S'nW is only a fraction of that seen in Prophet. The overhead of S'nW also follows the same trend. The latency of S'nW is also lower than that of Prophet and the reason could be that there is less number of packets unnecessarily transferred.

3.2.4. Prophet

This routing algorithm decides to forward packet based on the delivery predictability of the packet through an intermediate node. As per the results obtained from the experiment, Prophet Router performs well under conditions of lower network load. In the city environment when nodes restricted to an area are in contact with each other very frequently, maybe it's not

always wise to spread packets only based on increased probability of delivery alone. Other factors like free buffer size of the peer node and also the margin of difference between the probabilities could also be some things to consider. Prophet router transfers packets through an intermediate node even if the difference in delivery probability is miniscule. This might be a good approach when the number of packets is low and buffer overflows do not happen frequently. But in cases where the number of packets is too high and buffer overflows are common, such an approach can have some serious drawbacks. This can be seen in the comparison between the results of S'nW and Prophet, as the number of packets in the network increases. The number of packets dropped in Prophet Router is second only to Epidemic router. The latency is lower than S'nW when the number of packets is low, but as the number of packets becomes large, latency becomes slightly larger than S'nW router.

Scenario	Algorithm	Seed Interval	created	delivery_prob	overhead_ratio	latency_avg
Simple Plain Scenario	Direct Delivery	5	17280	0.1975	0	4546.9408
		10	8640	0.3477	0	8466.5192
		30	2880	0.5913	0	17351.4559
		70	1234	0.6361	0	18506.9391
	SnW	5	17280	0.2004	1.8094	4548.9218
		10	8640	0.3553	2.1599	7622.5586
		30	2880	0.6934	3.6795	10082.448
		70	1234	0.8849	5.0623	7246.3048
	Epidemic	5	17280	0.1994	1.8406	4503.4218
		10	8640	0.3444	2.3253	7398.764
		30	2880	0.5431	5.2494	10500.9869
		70	1234	0.6143	11.938	10555.9169
	Prophet	5	17280	0.1999	1.774	4555.899
		10	8640	0.3444	2.2228	7445.3166
		30	2880	0.5771	4.633	10649.7237
		70	1234	0.6564	10.5938	10817.7322
	Circular 5	5	17280	0.2035	1.766	4537.306
		10	8640	0.364	2.0766	7460.4278
		30	2880	0.7073	3.4914	10042.579
		70	1234	0.8938	4.7715	7604.5241
	City Router	5	17280	0.1981	0.0064	4628.2242
		10	8640	0.3488	0.007	8576.7992
		30	2880	0.5927	0.0094	17357.7037
		70	1234	0.6361	0	18537.3499

Table 1 : Plain Area Performance Evaluation

4. CIRCULAR SPREAD ROUTING ALGORITHM

4.1. Design

It is important to understand the characteristic of the underlying network and the movement of nodes in an environment before one can actually design a protocol which has the highest delivery probability with the lowest overhead and latency. The results that were observed for different routing algorithms mentioned in the previous section helped us analyze the strengths and drawbacks of each protocol. This greatly enabled us with the required knowledge to design an algorithm, which works best in a completely random environment. The basis of our algorithm is that a node N carrying a copy of a message, travelling in direction 1, has higher probability of delivering the message to a destination node D if it tries to give copies of the packet to nodes travelling in diverse directions. The section below discusses the heuristic design:

1. Identifying the cardinal directions for both the message and the node.
2. Spreading Algorithm:
    a. Assume a total of 'n' copies in the network.
    b. Each origin Node distributes 1 or n/2 number of copies to the next node it meets and reduces the total count of message copies it has. This is done using the MSG_COUNT_PROPERTY in the message.
3. Spread Limiter Algorithm:
    a. The limiter algorithm helps limit the spread of copy of messages based on two limiting assumptions
        i. *Absolute message cardinal direction*:
            1. Divide the entire region into 'x' number of cardinal directions, based on the direction coefficient value in the configuration. (x <= 8). Based on experiments, an optimal value is found to be x = 5. Let's call these divided regions as *absolutely cardinal.*
            2. Each message has a property, MSG_SENT_DIRECTIONS, where each direction is initialized to Boolean *FALSE*.
            3. When a message has been sent in direction 1, mark that direction as *TRUE* in the message properties.
            4. Now when a node A meets another node B, a copy of the message is given to node B only if the message was not already send in the direction in which 'B' is travelling.
        ii. *Relative node cardinal direction*:
            1. Taking the direction of travel of host node A as reference, divide the sections around node A into 'x' number of cardinal directions. In simpler words, if the direction of travel of 'A' is 270 degree downwards, then this is taken as reference (i.e., it is assumed as 0). Hence for 'A', the direction *north* is actually in the *east* and direction *east* is its actual direction of travel. Let's call this the *relatively cardinal.*
            2. When node A meets node B, pass a copy of the message only if the direction of B, is in one of the cardinal directions different from the direction of A.

3. If node B is travelling in the same cardinal direction as node A, then pass a copy of the message to node B, only if the speed of B is greater than A.

b. Hence when a host node meets a travelling node, first filter all messages that can be transferred to the travelling node by using '*Absolute message cardinal direction'* algorithm.
c. Now pass copies of all messages obtained from the previous step to the travelling node, only if the node is found valid using the *Relative node cardinal direction algorithm*.

4.2. Design Rationale

The rationales behind choosing such a design are:

1. The Gauss Markov model is a completely random model; hence it will only be effective to pass a copy of the message to a node which is travelling in a direction completely different from the host node, since the host node may meet the destination node in its current direction.
2. It is possible that a host node might always meet a travelling which might be travelling in a different but always the same direction. In such a scenario, it is ineffective to distribute a copy of the message in the same direction. By adding a message direction property, we can ensure that the message is spread in all directions and can avoid being sent in the same direction again.
3. Now, if a host node meets a travelling node going in the same direction, it is much more efficient to pass a copy of the message to the travelling node if it is moving at a much higher velocity since it can help deliver the message faster, if the destination node is the same direction as the travelling and host node.

4.3. Performance evaluation

4.3.1. Plain Area Scenario

The circular spread router algorithm indeed achieves very high delivery probability compared to all other routing protocols. This result was as expected since this protocol strives to effectively spread the copies of message in all directions possible, this way the message would definitely reach the destination. Also as seen in the observation, this new routing protocol has significantly reduced the amount of overhead in the network, this was only possible due to the limiting algorithms. Thus Circular spread algorithm performs significantly better than all other algorithms in terms of network overhead and delivery probability and the performance is fairly higher as the amount of load in the network also increases. The latency of the CS router is better than Epidemic and Prophet; it still is lesser than the Spray 'n Wait algorithm with reduced number of new messages in the network. The latency in delivering the message increases with the decrease in the network load. This is because the algorithm was not meant to be passive and reserved trying to pass a copy of the message to other travelling nodes.

4.3.2. Urban Area Scenario

Urban case scenario has a number of assumptions; a) Street Layouts, which forces and confines the nodes to move in fixed path/lanes and hence the underlying principle of totally

random motion for nodes on which this algorithm was designed does not hold good. Since the nodes have a fixed mobility within the city, the probability that the message will be spread will be confined only within the city. Let's assume that the destination node is in another city, if a particular host node having the message copy meets travelling nodes moving in different direction within the city before it meets a data mule, the message will not be sent to data mule and hence the message can never reach the destination. If by chance a message indeed gets into the data mule, the chance that the message will be spread is again limited since the message would have already been sent in all the direction within the city. There are many such drawbacks which make the circular spread algorithm designed for random movement a complete failure when applied on a city due to the cardinal direction assumption limitation. This is also the reason why the CS router has a very low delivery probability and very high latency. The performance seen is very similar to Direct Delivery because the chance that the message can get transferred out of the city is very low and also only messages destined to nodes within the same city will have a higher probability of getting the message.

City Routing Scenario	**SnW**	5	17280	0.4505	10.7274	5820.2473
		10	8640	0.505	9.6892	5774.8949
		30	2880	0.5674	8.6597	6856.3928
		70	1234	0.611	8.0464	8448.4523
	Epidemic	5	17280	0.3067	53.3812	8436.0913
		10	8640	0.3941	84.267	8641.5685
		30	2880	0.5899	166.0359	7213.3636
		70	1234	0.7212	297.164	4579.9974
	Prophet	5	17280	0.3392	42.9719	5913.5915
		10	8640	0.4338	67.7271	6617.4662
		30	2880	0.6097	138.9863	5837.4897
		70	1234	0.7269	243.3835	4290.1962
	Direct Delivery	5	17280	0.2863	0	6072.1661
		10	8640	0.316	0	9190.4727
		30	2880	0.3316	0	10791.143
		70	1234	0.3274	0	11248.9666
	City Router	5	17280	0.4457	4.7368	9243.5114
		10	8640	0.5693	7.1594	11134.0981
		30	2880	0.7684	11.7347	9312.9874
		70	1234	0.9246	14.0982	5907.627
	Circular 5	5	17280	0.2863	0	6072.1661
		10	8640	0.3161	0.0018	9194.1583
		30	2880	0.3316	0	10791.143
		70	1234	0.3298	0.0369	11433.2931

Table 2 : Urban Area Performance Evaluation

5. CITY ROUTING ALGORITHM

5.1. Design

City routing protocol is a protocol designed with the objective of attaining a high delivery probability with the lowest possible overhead. The key concepts that are involved in design of this algorithm are

- **Identification of data mules**: Data mules or trams are identified using their properties like 1G buffer size and average speed. When a node A meets a data mule B, A adds B to its list of known data mules.
- **Stratification of hosts:** Each host tries to stratify all other hosts in the simulation area that it knows of. This stratification is done into different strata like Single hop strata, Multi-hop strata and Data Mules. This is established through exchange of contacts when 2 nodes meet. When two nodes meet, they add each other to their single hop strata.
- **Maintaining history of contacts:** Each host maintains a count of the number of times it has met a host in its single hop strata.
- **Identifying nodes that are running low on free buffer space:** A configurable parameter 'low-buffer-factor' is used to identify if a node is running low on free buffer space. This factor was kept as 100 for the simulation. If a node had less than $1/100^{th}$ of free total buffer size, it is considered as to be running low on free buffer space.
- **Identifying nodes that are running high on free buffer space:** A configurable parameter 'high-buffer-factor' is used to identify if a node is running high on free buffer space. This factor was kept as 2 for the simulation. If a node has at least ½ of its total buffer size as free, it is considered to be running high on free buffer space.
- **Multiple packet forwarding strategies based on peer type, history of contacts & free buffer size of peer:**
    - **Peer is a data mule:**
        - When a host P (pedestrian) meets a host M (mule), P transfers a packet to M if and only if M has the destination node in any of its strata.
    - **Peer is not a data mule:**
        - When a host P1 meets another host P2, if the destination of the message (D) is unknown to P1 but P2 has this host in any of its strata, the message is forwarded to P2.
        - When a host P1 meets another host P2, if the destination of the message (D) is in the first hop strata of both P1 and P2, the message is forwarded to P2 if and only if P2 has met the destination node more times than P1 and also if P2 has high free buffer size and P1 is running low on free buffer size.
        - When a host P1 meets another host P2, if the destination of the message (D) is in the first hop strata of P2 but only in the multi-hop strata of P1, the message is forwarded to P2 if and only if P2 has high free buffer size and P1 is running low on free buffer size.

5.2. Design Rationale

There were multiple reasons why the above mentioned design decisions were chosen.

1. Identification of data mules was necessary as these nodes have higher buffer size and also because they are the only means of carrying packets to nodes in other regions. Since the normal pedestrian nodes and data mules have different movement model characteristics and buffer size characteristics it was only reasonable to adopt different packet forwarding strategy for them.
2. Stratification of hosts into first-hop, multi-hop and data mules is representative of the proximity or nearness between the hosts. Since the pedestrian nodes are restricted to move only within a region, we assumed that it was likely that a good number of hosts in a region knew that they were in each other's first hop strata. In other words, it can be deduced that these nodes were from the same region. Such sort of an inference cannot be made from a probabilistic approach like Prophet. Multi-hop strata were included because a pedestrian node in one region can only be present in the multi-hop strata of pedestrians in another region. If the algorithm runs for sufficient amount of time it can be inferred that a host present in multi-hop strata is most likely in another region and hence the best way to route packets through to that node is through a data mule.
3. If two nodes (A, B) both of which have had contact with the destination node (D), meet each other, should A transfer the packet to B ? The answer to this depends on who has met the node more number of times. But forwarding packet only based on number of times of previous contact should not be the only approach. In such cases it is always better to transfer the packet only if the node A is running low on free buffer space and node B is more likely to transfer the packet to B.
4. The key learning from running different algorithms for city scenario was that if the packet transferring is not selective and intelligent enough, there is unwanted overhead and the delivery probability is poor. To prevent unwanted relays and dropping of packets it was decided to minimize dropping of packets by forwarding only if the pedestrian node is likely to lose the packet because of overflow. To achieve this it was needed to qualify a node for transfer based on its available free buffer size.

5.3. Other Concepts Explored

Some of the other concepts that were explored but were dropped out eventually were

1. Splitting the entire simulation area into four quadrants and each node, when it generates a message also adds its x & y coordinates as message property so that an intermediate node gets to know how far the nodes could be or which quadrant they belong to. Further messages would then be forwarded to nodes having history of visiting areas near the destination host.
2. Setting a flag in a message which indicated if the message had passed through a data mule. If this message came through a mule, then i for sending further messages to that node, the same mule could be used.
3. Controlling the number of packets by using some extension of Spray 'n Wait. When a packet is sent from a mule to a normal pedestrian, it would have multiple numbers of copies (a fixed number). The pedestrians will then distribute this in their regions using Binary Spray 'n Wait algorithm.

5.4. Performance evaluation

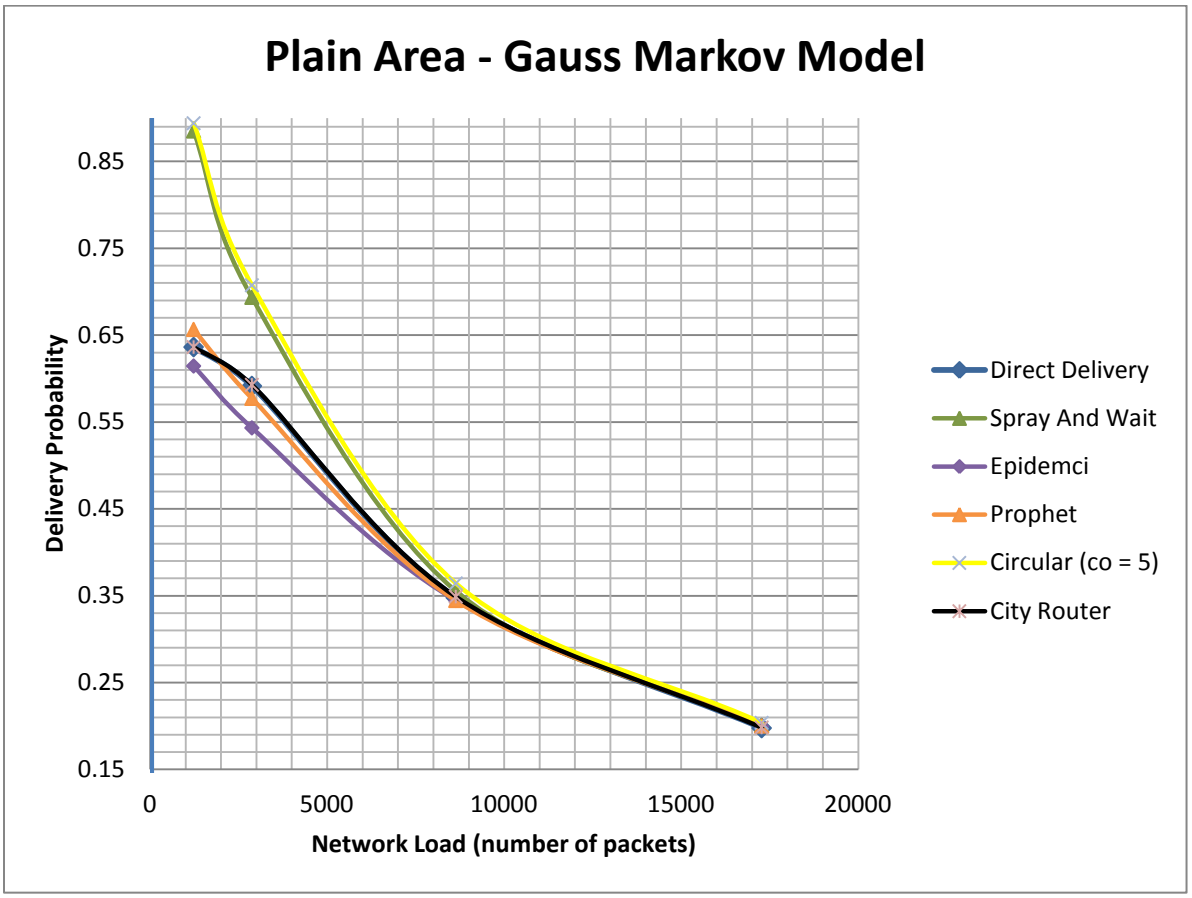
5.4.1. Plain Area Scenario

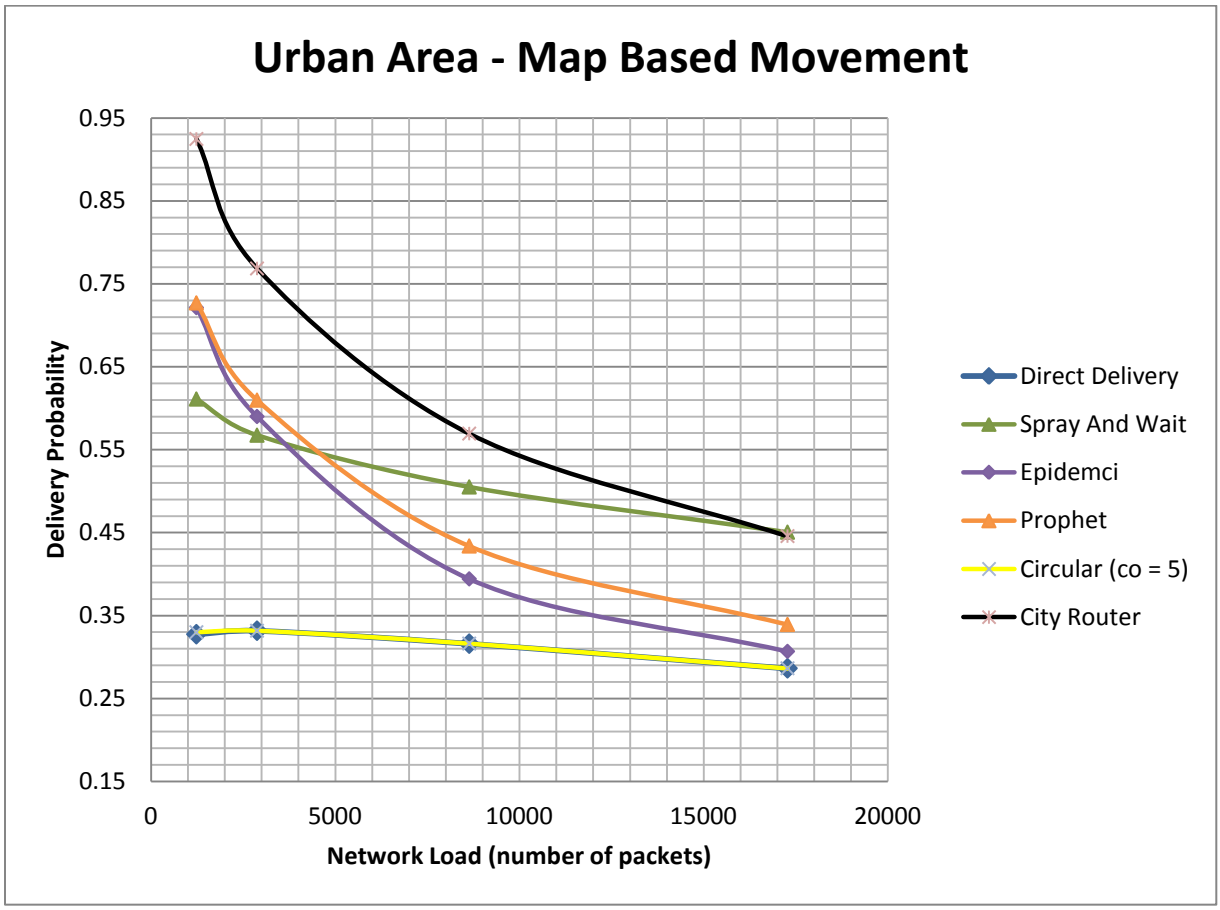
The basic assumptions, under which the City Router was developed, do not hold good for a mobility model like the Gauss Markov Mobility Model. There are no data mules in the Gauss Markov data model. All hosts move randomly and hence the assumption that nodes are restricted to motion within regions is no longer valid. All hosts have the same level of mobility this is unlike the Urban scenario where the data mules have higher mobility than the pedestrians. The results of running the City Router in plain Gauss Markov mobility model scenario were not surprising. The algorithm only managed to outperform Direct Delivery routing in terms of delivery probability. When the network load was the least (interval 70;100), the City Router defaults to the performance of Direct Delivery router as the results obtained by both are very similar. The City Router and the Epidemic router performances can be compared too. Except for the case when the number of messages is the highest, City Router has a better delivery probability. Intuitively the latency was higher for City Router. This is because City Router is very choosy in forwarding packets to peers. This results in lower overhead and lower number of dropped packets in City Router. Spray ‘n Wait router outperforms the City router in all conditions because it tries to spread packets more freely and the same time keeping the load on the network under check. Prophet Router performs better than City Router in terms of delivery probability in cases when the delivery load on the network is highest (interval 5;10) and lowest (70;100). But in other cases of moderate load (interval 10;30 & interval 30;70), City router has a higher delivery probability. The Circular Spread Router has better performance than City Router as it tries to spread packets intelligently in different directions keeping the number of dropped packets under check.

5.4.2. Urban Area Scenario

City Router algorithm results show significant improvement over other Routing Algorithms tested as part of this experiment. The delivery probability of City Router is higher than that of Prophet under all scenarios. City Router manages to achieve this delivery probability rate at a fraction of the overhead of Prophet. However the latency of City Router is higher than that of Prophet. This is as per the design of City Router routing protocol. City Router does not try to aggressively transfer packets to decrease the delay. On the other hand, it tries to achieve low overhead with a good amount of delivery probability. It does so by sending packets considering the receiving nodes free buffer capacity. Spray ‘n Wait algorithm gets a slightly higher delivery probability than City Router when the network load increases to its maximum. But the difference is too minute to analyze in depth. But the overhead incurred in achieving this rate for City router is less than half of S’nW router. On all other scenarios when the number of messages is lesser, City Router significantly outperforms S’nW and also other algorithms (in terms of delivery probability, delivery overhead). This is because of the reasons mentioned in Design section. However City Router performs poorly in terms of latency of delivered message in cases when the number of messages is pretty high.

Delivery Probability vs. Network Load





Average Latency vs. Network Load