Throughput Comparison of AODV-UU and DSR-UU Protocol Implementations in Multi-hop Static Environments

Josip Lorincz*, Nenad Ukić and Dinko Begušić*
Ericsson Nikola Tesla d.d., R&D Centre, ETK/DR, Split, Croatia
*FESB-Split, University of Split, Split, Croatia
E-mail: nenad.ukic@ericsson.com, *josip.lerinc@fesb.hr

***Abstract* – In this paper, we present throughput analysis of two popular Mobile Ad hoc NETwork (MANET) routing protocol implementations, AODV-UU (Uppsala University) and DSR-UU. Both implementations are Linux based user space implementations developed using C language. Instead of using simulator for throughput analysis, the investigated scenario involves a stationary test-bed that consists of real wireless nodes in ad-hoc mode connected using IEEE 802.11b standard. Influence of number of wireless nodes on overall throughput, with and without predefined static routes between nodes has been investigated. Specially developed java-based configuration and management utility was used for static route selection. Disabling or enabling of predefined wireless connections is accomplished by discarding frames according to their hardware address at MAC layer. Throughput performance of both protocols during FTP data transfer over real multi-hop network of wireless nodes has been analyzed. Additionally, a delay introduced by AODV-UU and DSR-UU protocol implementations in dynamic route change situations caused by wireless link breakage has been studied.**

***Keywords:* DSR, AODV, MANET, Ad-hoc, throughput analysis, static routes, route discovery time.**

I. INTRUDUCTION

A Mobile Ad hoc NETwork (MANET) consists of a set of wireless mobile nodes (PDA, laptops, smart phones…) communicating with each other using wireless links without any centralized control or fixed network infrastructure (e.g. access point, base station). In such a network, mobile nodes form dynamic, autonomous, self-configuring network where each node acts both as a host and also as a router forwarding data packets for other nodes. Such a network may operate in standalone fashion, or may be connected over fixed network infrastructure to the larger Internet.

While several ad-hoc routing protocols have bean proposed, we chose for analyses two of these protocols: the *Dynamic Source Routing* (DSR) [6] and the *Ad Hoc On-Demand Distance Vector* protocol (AODV) [5], because of their prominence in the ad hoc networking research community. DSR and AODV share an interesting common characteristic — they both initiate routing activities on an *on demand* basis. This *reactive* nature of these protocols is a significant departure from more traditional *proactive* protocols, which find routes between all source-destination pairs regardless of the use or need for such routes [13].

Heretofore, lots of research efforts have been invested in simultaneous performance comparison of different MANET routing protocols in order to improve its routing characteristics. In previous works [7], extended performance analysis of AODV and DSR protocols has been shown. To improve performance of ad-hoc networks different modifications of AODV and DSR protocols have been proposed and enhancements have been compared with basic AODV and DSR protocols [11, 8, 10, 12]. Also, performance comparison of AODV and DSR protocols in terms of energy consumption has been analyzed [9].

Most of previous analyses have been performed using ad hoc network simulators. Instead of using simulation model, our idea was to investigate throughput performance of AODV and DSR in real multi-hop static environment during real FTP transfer of fixed size data files. Specificity of our investigation is that we used predefined static routes for FTP transmission between nodes placed in the same coverage area. Although a lot of work on comparison of these two protocols has been published, only few of them are experimental and application-oriented. Moreover, static routes in MANET, if used, are considered as limitation rather than exploited for possible applications. We found usage of policy-based frame filtering for static route scheduling as useful feature of any ad-hoc network.

Usage of real multi-hop environment, rather than simulation, puts some constraints to our experiment regarding node mobility and hop number. Hopefully, that gives more insight to real-life scenarios than simulation does. Limited number of hops (6 in our test-bed network), limited node mobility and usage of static routes may seem to introduce rigid implementation restrictions, but in our case, that covers high number of future real-life practical scenarios. Among many applications, we see MANET networks also as the edge or last-step networks used to extend coverage areas of existing wireless, infrastructure based networks. In this context, foreseen applications may include Internet access to public areas in temporary manner. Such areas may comprise fairs, airport or other waiting rooms, conference rooms and other similar public places. Maximum number of 6 nodes in this sense doesn't play important role, since static routes may be used for ISP selection, network monitoring, charging or firewall purposes.

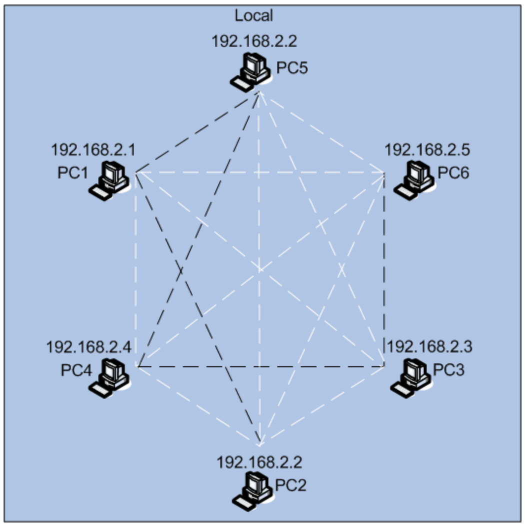

Fig. 1 "Forced multi-hop" example (software application screenshot)

This paper is structured as follows: in Section II, a brief description of AODV and DSR is presented. Section III describes testbed environment used for measurement. Obtained results are presented and discussed in Section IV. Finally in Section V conclusion remarks are given.

II. OVERVIEW OF AODV AND DSR

While DSR and AODV share the on-demand behavior in that they initiate routing activities only in the presence of data packets in need of a route, many of their routing mechanisms are very different [7]. In particular, DSR uses source routing, whereas AODV uses a table-driven routing framework and destination sequence numbers. DSR does not rely on any time based activities, while AODV does to a certain extent.

A. AODV protocol

AODV is an on-demand dynamic routing protocol that uses routing tables with one entry per destination. This is in contrast to DSR, which can maintain multiple route cache entries for each destination. When a source node needs a route to a destination, it initiates a route discovery process to locate the destination node. The source node floods a query packet requesting a route to be set up to the destination.

A reply is sent back directly to the source node either by the destination itself or any other intermediate node that has a current route to the destination. On receiving a route request (RREQ), intermediate nodes update their routing table for a reverse route to the source. Similarly, the forward route to the destination is updated on receiving a route reply (RREP) packet. AODV uses sequence numbers to determine the timelines of each packet and to prevent loops. Expiry timers are used to keep the route entries fresh.

Link failures are propagated by a route error (RERR) message from the site of a link break to the source node for that route. When the next hop link breaks, RERR packets are sent to a set of neighboring nodes that communicate over the broken link with the destination.

TABLE I. HARDWARE CONFIGURATIONS USED IN MEASUREMENTS

	RAM [MB]	CPU	Frequency
PC1	768	Intel® Pentium® 4	2.00 GHz
PC2	512	Intel® Pentium® M	1.73 GHz
PC3	512	Intel® Pentium® M	1.60 GHz
PC4	256	Intel® Pentium® 4	1.60 GHz
PC5	768	Intel® Pentium® 4	2.00 GHz
PC6	512	Intel® Pentium® 4	2.00 GHz

TABLE II. DSR-UU AND AODV-UU COMPARISON SUMMARY

MANET routing protocol implementation comparison		
	DSR	AODV
Implementation	Uppsala University (DSR-UU)	Uppsala University (AODV-UU)
Version	0.2	0.8.1
OS	Linux FC4 (2.6.X kernel)	Linux FC4 (2.6.X kernel)
Implemented as	Two kernel modules	Kernel module
Coexistence with non-multihop Ad-hoc network	YES (virtual interface)	NO
Multiple routes per destination	YES (Route cache)	NO (Routing table)
Route metric	hop count	hop count
Support multiple route metrics	YES	NO
Physical layer technology	802.11b	802.11b
Theoretical scaling law	1/N	1/N

This recursive process erases all broken entries in the routing table of the nodes. Since nodes reply to the first arriving RREQ, AODV favors the least congested route instead of the shortest route. The AODV on-demand approach minimizes routing table information. However, this potentially leads to a large number of route requests being generated.

B. DSR protocol

Dynamic Source Routing (DSR) utilizes source-based routing rather than table-based. That is, sender knows the complete hop-by-hop route to the destination. DSR is also an on-demand protocol and has a similar route discovery process to AODV. Route discovery works by flooding the network with *route request* (RREQ) packets where each node receiving an RREQ rebroadcasts it, unless it is the destination or it has a route to the destination in its route cache. One of the primary differences between DSR and AODV is that intermediate node addresses are accumulated on the DSR RREQ and RREP control packets. Every node in the network uses the information in the RREQ/RREP packets to learn about routes to other nodes in the network. These nodes store the routes in their route caches.

Once a RREP is received, the sender node knows the entire route to the destination. The route carried back by the RREP packet is cached at the source for future use. Data packets in DSR are routed by the intermediate nodes using the complete knowledge of the route to the destination contained in the packet header. If a link breaks and the next node on the source route is currently not its neighbor, the node reports an error back to the source using a route error (RERR) packet, and leaves it to the source to establish a new route. Alternatively, the node may try a different path, if it has an alternate route cached. DSR stores multiple paths per destination and does not use any expiry timers on route cache entries.

As an advantage, source routing in DSR eliminates routing tables and the aggressive caching reduces the overhead of DSR. However, there are two primary disadvantages of DSR, as found in [7]. Route reply flooding in DSR results in costly MAC layer overhead. Secondly, DSR is not scalable to large networks.

III. TESTBED ENVIRONMENT

Our experimental measurements have been carried out on S-Net laboratory network. S-Net Mobile Ad-hoc laboratory network is a testbed built up within S-Net Research project at Ericsson Nikola Tesla d.d., R&D center in order to evaluate usage of mobile ad-hoc networks. Special attention is given to measurements of throughput and route hot swap time comparison for AODV and DSR. Features of each MANET protocol used in our experimental analysis are listed in Table II.

A. Hardware and Software

In real MANET network wireless node hardware capabilities usually vary. Having this in mind, requirement for all nodes to have same hardware capabilities is not crucial. Hardware capabilities of hosts used in our laboratory measurements are listed in Table I. To ensure stability and compatibility all our measurements where done at 11 Mb/s (IEEE 802.11b standard) although all used wireless LAN cards (Zydas zd1211 chipset based 802.11g Wireless USB 2.0 Adapters) support IEEE 802.11g standard. Testbed network consists of 6 machines (nodes) that run Fedora Core 4 with 2.6.11-1.1369_FC4 kernel. We used Uppsala University routing protocol implementations; AODV-UU version 0.8.1 and DSR-UU version 0.2. Besides routing protocol implementations, the laboratory network comprises configuration and management utilities, software developed within S-Net project at Ericsson Nikola Tesla d.d. R&D Centre. It is java-based application with graphic user interface (GUI) that allows users to:

- Scan and visualize wireless network (nodes and links between them);
- Run selected routing protocol implementation on both, local and remote computers;
- Stop running routing protocol on all visible nodes;
- Disable (or enable) some direct wireless connections (forcing multi-hop connections);
- Disable all visibility constraints;
- Redraw (rescan) network topology to check if it is up-to-date.

B. Testbed Topology

All machines were located inside one laboratory room so visibility constraints necessary for multi-hop data transfer is artificially accomplished (Fig.1). This is sometimes called “forced multi-hop”. Data frames are filtered according to their MAC addresses via *iptables* Linux tool [1]:

```
#iptables –A INPUT –m mac –mac source
\ <MAC_ADDR> –j DROP
```

$$\begin{bmatrix} 1921682.1 & 1 & 1 & 1 & 1 & 1 & MAC\ 1 \\ 1 & 1921682.2 & 1 & 1 & 1 & 1 & MAC\ 2 \\ 1 & 1 & 1921682.3 & 1 & 1 & 1 & MAC\ 3 \\ 1 & 1 & 1 & 1921682.4 & 1 & 1 & MAC\ 4 \\ 1 & 1 & 1 & 1 & 1921682.5 & 1 & MAC\ 5 \\ 1 & 1 & 1 & 1 & 1 & 1921682.6 & MAC\ 6 \end{bmatrix}$$

Fig. 2. Topology matrix for full mesh topology

$$\begin{bmatrix} 1921682.1 & 1 & 0 & 0 & 0 & 0 & MAC\ 1 \\ 1 & 1921682.2 & 1 & 0 & 0 & 0 & MAC\ 2 \\ 0 & 1 & 1921682.3 & 1 & 0 & 0 & MAC\ 3 \\ 0 & 0 & 1 & 1921682.4 & 1 & 0 & MAC\ 4 \\ 0 & 0 & 0 & 1 & 1921682.5 & 1 & MAC\ 5 \\ 0 & 0 & 0 & 0 & 1 & 1921682.6 & MAC\ 6 \end{bmatrix}$$

Fig. 3. Topology matrix for “chain” topology

Since routing protocol implementation resides at higher level of OSI model (than MAC layer), routing protocol functions as it would function in real multi-hop environment. Radio frequency interference and node mobility effects are considered to be minimal and are ignored. To avoid unidirectional links, *iptables* command with corresponding MAC addresses should be entered on both machines participating in communication on a single link.

Our laboratory network with 6 nodes has totally 15 wireless links (Fig.1.). Configuration software application is used to set MAC address filters from graphic user interface (GUI) rather than from console. For network scanning *nmap* tool is used.

```
#nmap –sP –ttl 1 –n 192.168.2.1 -6
```

In the previous command, *Ttl* (time-to-live) part of command is set to 1 (hop), because we are interested only in direct (single-hop) connections. Part of previous command *sP* select ping scan (only determine if host is online, no port scanning). Part of `nmap` command *n* disables DNS resolution for faster output. If the command `nmap` is run as root user (as in our case), output additionally contains MAC address of WLAN interface for all discovered non-local hosts. The discovery cycle initiated from local node include remote execution of the *nmap* command on any discovered live node in the designated range of IP address regardless of the node from which it is seen. In that way we build up the topology matrix comprised of IP addresses of live nodes in matrix diagonal and ones at intersections of direct (single-hop) connected nodes at other positions of matrix. Every matrix row represents a node and column represents possible communication combination of that node (Fig. 2. and 3.). Besides topology matrices which define connectivity within the network, last column of the matrices contains MAC addresses of the hosts.

The topology matrix is a base for visualization and for full mesh topology where each node can potentially communicate with every other node topology matrix is shown on figure 2. Without any knowledge of their relative positions, configuration and management utility places all nodes into angles of a regular polygon as shown on Fig. 1. Nodes are represented with small icon

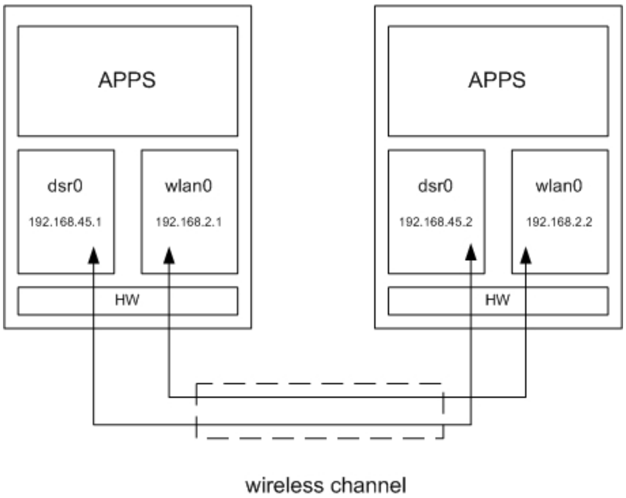

Fig. 4. DSR: Two (virtual) interfaces using same hardware

and IP address. Wireless links between nodes on Fig. 1. are represented with lines, where white links are disabled, and black links are enabled by configuration software utility (predefined static routes). Topology of interest in our measurements is so called "chain topology". This means that all nodes will be able to communicate only with two (pre-selected) neighboring nodes (data from other nodes is discarded at MAC layer), so we will have only one route from source to destination. Subsequently, end nodes will have only one neighbor. In this way we "force" pre-selected routes by disabling all other possible routes. Topology matrix for "chain topology" is shown on Fig. 3. Thus, by manually selecting communication paths between nodes working in ad-hoc mode, we can define communication paths between any communicating points. Although all communication principles between wireless nodes are based either on AODV-UU or DSR-UU MANET routing protocol, we manage to define communication links between nodes based on predefined static routes using specially developed software running on every node.

C. DSR Characteristics

Unlike AODV-UU, DSR-UU implements a *virtual* network interface (dsr0). This enables DSR network to coexist with the regular non-multihop ad hoc network. This feature may be very useful in some scenarios because it enables two independent logical networks to run over the same hardware at the same time. Naturally, trade-off with performance is expected if both interfaces (networks) are used simultaneously. From application and end user perspective, these two interfaces appear as two separate physical interfaces. This fact may be used in a number of applications.

D. Throughput Considerations

Since all hosts in the network operate at the same channel (frequency), in multi-hop transfer only time division multiplexing is possible. Ideally, we have static (non-moving) hosts that are in vicinity to each other. There is no frequency interference and no obstacles between hosts. Processing and buffering times in intermediate nodes are considered to be zero. In this case throughput will be inversely proportional to hop number

TABLE III. DSR-UU FTP THROUGHPUT MEASUREMENTS

Number of hops	Average [KB/s]	Relative deviation	Minimum [KB/s]	Maximum [KB/s]	Number of measurements
1	658,89	7,26%	580	730	20
2	330	4,79%	310	350	20
3	202,22	3,30%	190	210	20
4	150,91	9,58%	130	170	20
5	104,44	10,83%	89	120	20

TABLE IV. AODV-UU FTP THROUGHPUT MEASUREMENTS

Number of hops	Average [KB/s]	Relative deviation	Minimum [KB/s]	Maximum [KB/s]	Number of measurements
1	725	6,85%	630	770	20
2	326	17,79%	190	380	20
3	202	13,97%	160	250	20
4	144	11,89%	120	170	20
5	130	6,28%	120	140	20

N (1). Because all hosts are located within the same room every host can "hear" all other hosts on physical layer. Since visibility constraints are achieved artificially at MAC layer using specially developed software, only one transfer between two neighboring hosts in the chain is possible at the time. Thus, if we ignore processing and buffering times, total transfer time (T_{total}) is equal to a product of a single-hop transfer time (T_1) and number of hops (*N*). Constant C_1 is equal to throughput of single-hop transfer.

$$Throughput_{colocated} = \frac{Size}{T_{total}} = \frac{Size}{N * T_1} = \frac{C_1}{N}\left[\frac{KB}{Sec}\right] \quad (1)$$

$$C_1 = \frac{Size}{T_1}\left[\frac{KB}{Sec}\right] \quad (1a)$$

Coverage area of the nodes in our experiment forming *artificial chain topology* (collocated nodes) is almost the same as a single node's one, since all nodes are placed very close to each other. It may be useful to mention theoretical upper bound (2) for throughput in general wireless multihop (*artificial chain topology*) network (corresponds to C/SQRN(N) in rest of paper) [3][4]:

$$Throughput_{ideal} = \frac{C}{\sqrt{N}} \quad (2)$$

where *C* is constant and represents theoretical wireless channel capacity.

IV. RESULTS

A. Protocol Comparison

Comparing data from Table III. and IV., we can see that although DSR has somewhat lower total throughput for all hop numbers it also has lower throughput deviation. Only at 5 hop transfers DSR had more than 10% relative deviation (relative to average value) while AODV had more than 10% deviation in 2, 3 and 4 hop transfers (exceeding 17% in double-hop transfers). This indicates that this implementation of DSR-UU protocol is more stable than AODV-UU protocol implementation. These statements get more importance if we note that DSR-UU version used in measurements is an early 0.2 version versus older AODV-UU 0.8.1 version.

Possible explanation of AODV's throughput variations could be in somewhat shorter route *timeout* settings. DSR configuration value that defines single timeout value for

TABLE V. DSR AND AD-HOC SINGLE-HOP THROUGHPUT COMPARISON

chain	Ad hoc			DSR			Number of measurements
	Minimum [KB/s]	Average [KB/s]	Maximum [KB/s]	Minimum [KB/s]	Average [KB/s]	Maximum [KB/s]	
61	740	751	760	580	659	730	21
14	690	733	710	620	637	660	21
24	710	720	730	660	666	680	21
Average	713,33	734,66	733,33	620,00	654,00	690,00	
St. Deviation	3,53%	3,55%	3,43%	6,45%	2,31%	5,23%	

TABLE VI. AODV AND AD-HOC SINGLE-HOP THROUGHPUT COMPARISON

chain	Ad hoc			AODV			Number of measurements
	Minimum [KB/s]	Average [KB/s]	Maximum [KB/s]	Minimum [KB/s]	Average [KB/s]	Maximum [KB/s]	
23	670	682,86	690	630	655,71	680	21
63	740	741,43	750	710	734,29	740	21
61	750	760,00	770	750	767,14	770	21
51	760	767,14	770	770	770,00	770	21
65	670	720,00	750	750	752,86	760	21
Average	718,00	734,29	746,00	722,00	736,00	744,00	
St. Deviation	6,18%	4,64%	4,41%	7,74%	6,39%	5,08%	

all routes in the cache (*RouteCacheTimeout*) is set to 300 seconds and it had no operational influence on throughput. Meanwhile, AODV has several timeout configuration values ranging from 1 to 15 seconds. These values are relatively short indicating settings for mobile environment, while all our measurements were done in static environment (nodes do not move). It is possible for AODV to perform better if proper parameter tuning is applied [5], [6].

Other explanation can be found in basic operating differences of these two protocols. DSR utilizes source-based routing rather than table-based where sender knows the complete hop-by-hop route to the destination. Also, each node on source route has knowledge of entire route, thus requiring less time for route selection which results in lower relative deviation when DSR-UU protocol is used.

B. Throughput

Average achieved throughput in single hop case is 654 KB/s for DSR, 736 KB/s for AODV and 734 KB/s for *pure* Ad-hoc mode (Table V., VI.). All of these values are around 700KB/s (~5.7 Mb/s) which is typical single hop throughput of TCP based applications for IEEE 802.11b standard.

To simulate measurement environment closer to real situation, it is important to emphasize that multiple single hop throughput measurements have been performed for each protocol on different nodes. Thus, Tables III. and IV. presents both *pure* ad hoc and DSR (or AODV) single hop throughput results respectively, for different chains. Last octets of the IP address of all nodes in the chain are used to name different chains. For example, chain (24) defines communication link of hosts with 192.168.2.**2** and 192.168.2.**4** IP addresses (all hosts are in 192.168.2.x network).

AODV single hop has negligible difference in performance compared with *pure* single hop ad-hoc, achieving values near typical (single hop) throughput of TCP based applications for IEEE 802.11b standard. On the other hand, Table VI. shows small difference of measured throughput for single hop FTP transfer when comparing *pure* ad-hoc mode and DSR protocol. This can be explained with increased overhead that DSR protocol introduces during route discovery.

Fig 7. additionally shows that the major difference between AODV-UU and DSR-UU FTP throughput performance is in single hop case performance. We emphasize that measurements where done on real nonmoving nodes placed in the some coverage area, for real FTP fixed size file transfer over static predefined

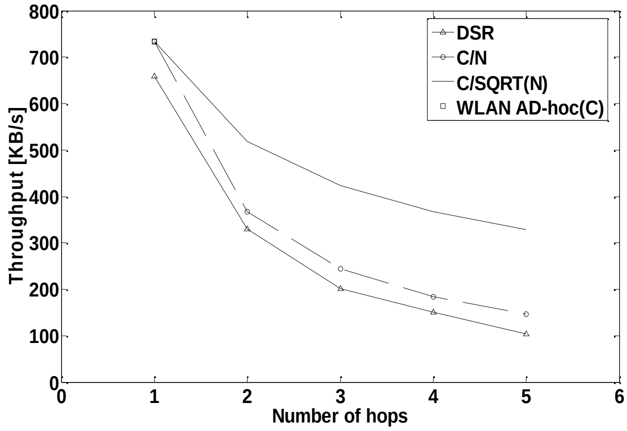

Fig. 5.Comparison of theoretical model and DSR-UU measurement results

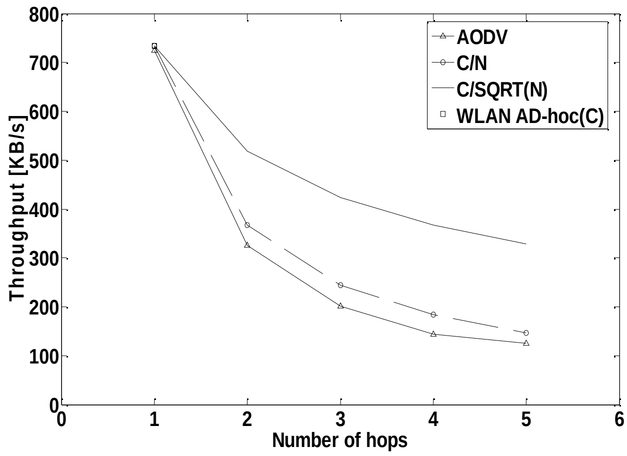

Fig. 6. Comparison of theoretical model and AODV-UU measurement results

routes. Multihop performance is almost the same and difference may be considered as statistical error due to a limited number of measurements performed. Advantages of source routing used by DSR-UU are obviously not utilized to the full extent in single hop case while additional processing needed by source routing paid of in multihop. Reasons for such measurement results can be found in different nature of both protocols and also in fact that preselected static routes have been used for FTP transmissions. Although DSR-UU has better throughput performance then AODV-UU [7] and is more scalable for small ad hoc networks, usage of preselected static routes defined by our software application impairs DSR-UU

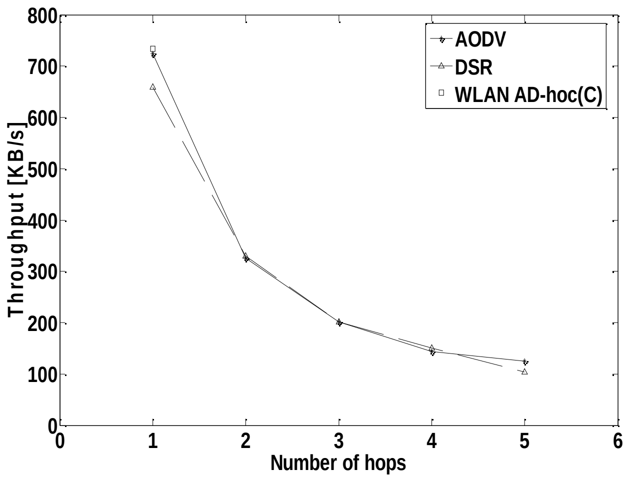

Fig. 7. Throughput comparison of AODV-UU and DSR-UU protocol

advantages leading to almost some multihop throughput performance.

If compared with theoretical scaling law C/N (1) and C/SQRT(N) (2), DSR-UU and AODV-UU scale almost the same (Fig. 5., 6.). Overall throughput is decreased with every new node introduced in ad-hoc network. That confirms accuracy of theoretical hypothesis. Throughput difference is almost the same regardless of the number of used hops. This can be explained by assumptions made in theoretical model. These assumptions include: ignored processing times at source and destination nodes, including FTP application delay as well as delay introduced by routing protocol in route discovery process.

C. Route Discovery Time

Route discovery time is defined as time needed for a routing protocol to find route to specified destination. If route breaks, and DSR have cached other routes besides primary route to the destination, DSR will switch to the first best available route according to DSR protocol algorithm. It will not initiate new route discovery process, as it will be the case with AODV since AODV doesn't have other alternative routes cached. However, DSR routing cache takes significantly more memory and processing power whereas source routing that is used by DSR also introduces additional delay in each traversed node. This implies that decision which one of these two protocols is better in broken route scenario (i.e. will converge in shorter time) is not trivial.

During determination of *route discovery/switching time* measurements methodology problem arises. We performed two types of measurements with different methodologies. At first we tried to compare FTP transfer times with and without route breakage. This seemed as promising way of measuring time needed for the protocols to switch to new route. However, measurements inconsistencies (FTP transfer time without route breakage sometimes took longer than FTP transfer time with route breakage), high deviation, measurement chain (hardware) dependency and fact that FTP transfer is connection-oriented (TCP based) enforce us to reconsider this methodology.

TABLE VII. DSR ROUTE DISCOVERY TIME

Chain	Last ICMP sequence number before route change	First ICMP sequence number before route change	Number of lost ICMP packets	ICMP packet transm-ission interval	Route discovery time (s)	Measurm. number
62(5)1	55	82	27	0.1	2.7	1
	55	81	26	0.1	2.6	2
	56	81	25	0.1	2.5	3
	56	81	25	0.1	2.5	4
	58	84	26	0.1	2.6	5
	59	86	27	0.1	2.7	6
	56	84	28	0.1	2.8	7
	31	59	28	0.1	2.8	8
	51	77	26	0.1	2.6	9
	31	57	26	0.1	2.6	10
Average route discovery time					**2.64**	
Standard deviation					**4,07%**	

TABLE VIII. AODV ROUTE DISCOVERY TIME

Chain	Last ICMP sequence number before route change	First ICMP sequence number before route change	Number of lost ICMP packets	ICMP packet transm-ission interval	Route discovery time (s)	Measurm. number
62(5)1	43	63	20	0.1	2	1
	56	76	20	0.1	2	2
	41	60	19	0.1	1.9	3
	55	75	20	0.1	2	4
	37	59	22	0.1	2.2	5
	38	58	20	0.1	2	6
	39	58	19	0.1	1.9	7
	42	62	20	0.1	2	8
	41	61	20	0.1	2	9
	58	78	20	0.1	2	10
Average route discovery time					**2**	
Standard deviation					**4.08%**	

```
[root@resta jars]# ping -R -i 0.1 192.168.45.1
PING 192.168.45.1 (192.168.45.1) 56(124) bytes of data.
64 bytes from 192.168.45.1: icmp_seq=0 ttl=63 time=3.65 ms
RR:     192.168.45.6
        192.168.45.2
        192.168.45.1
        192.168.45.1
        192.168.45.2
        192.168.45.6

64 bytes from 192.168.45.1: icmp_seq=1 ttl=63 time=2.35 ms
(same route)
64 bytes from 192.168.45.1: icmp_seq=2 ttl=63 time=2.47 ms
(same route)
...Some output lines omitted...
64 bytes from 192.168.45.1: icmp_seq=26 ttl=63 time=2.31 ms
64 bytes from 192.168.45.1: icmp_seq=27 ttl=63 time=3.42 ms
(route change)
64 bytes from 192.168.45.1: icmp_seq=54 ttl=63 time=3.76 ms
RR:     192.168.45.6
        192.168.45.5
        192.168.45.1
        192.168.45.1
        192.168.45.5
        192.168.45.6

64 bytes from 192.168.45.1: icmp_seq=55 ttl=63 time=2.27 ms
(same route)
64 bytes from 192.168.45.1: icmp_seq=56 ttl=63 time=2.45 ms
(same route)
...Some output lines omitted...
64 bytes from 192.168.45.1: icmp_seq=70 ttl=63 time=2.37 ms
(same route)
64 bytes from 192.168.45.1: icmp_seq=71 ttl=63 time=4.30 ms
(same route)

--- 192.168.45.1 ping statistics ---
72 packets transmitted, 46 received, 36% packet loss, time
7344ms
rtt min/avg/max/mdev = 2.084/2.933/4.303/0.674 ms, pipe 2
[root@resta jars]#
```

Fig. 8. List of ICMP replies during route switching/discovery process

Alternative methodology was to use frequent UDP-based ping (ICMP) packets with route record option:

```
# ping -R -i 0.1 192.168.45.1
```

All ICMP replies have sequence number (Fig. 8.). In each interval an ICMP request is sent to destination. With route record option and query interval small enough (0.1 second in our case) we could observe route switching as it happens, almost in real time. To simulate intermediate node crash, we unplugged the USB WLAN card from USB port after few ICMP ping packets sent. Since ICMP traffic is UDP based (connectionless), ICMP request packets sent during *route discovery/switching time* will not reach destination host so no ICMP reply will be sent to source host. By counting missing ICMP replies and knowing query interval time we can easily calculate time when no route to destination was available (during *route discovery/switching time*). Round Trip Time (RTT) of ICMP packets is millisecond in order (which is hundred times shorter than query interval time) so it can be ignored.

If we analyze list of ICMP replies shown in Fig. 8. we can see that chain “621” is used at beginning of transmission. Route brakes after 27 successfully received ICMP (ping) replies. If we analyze sequence numbers of received ICMP replies we can see that during route discovery process further 27 ICMP replies have been lost. Also, we can notice that new route has been established over chain “651” (starting with ICMP sequence number 54).

Results of FTP transfer time measurements indicate that AODV-UU is somewhat faster than DSR-UU. The whole process took 2 seconds on average for AODV-UU and 2.64 seconds for DSR-UU. However, very high deviation of these measurements (75% for AODV-UU and even 128% for DSR-UU) makes these results unreliable. Alternative, *ping* methodology appears to be more accurate (4,08% for AODV-UU and 4,07% deviation for DSR-UU) and that makes results more reliable (Table VII., VIII.). Measuring in this way, DSR-UU is somewhat slower than AODV-UU (2,64 seconds compared with 2,00 seconds).

Although one could say that DSR-UU has advantage in comparison with AODV-UU (route cache usage), measurements show that in case of link breakage, DSR-UU needs more time to redirect data flow to new route than AODV-UU. This may not be expected but source routing obviously has impact on overall DSR-UU performance. It should be noted that DSR-UU_v0.2 implementation we used was in early development stage compared with AODV-UU_v0.8.1, so future DSR-UU implementations may be more optimized and may achieve better results.

V. CONCLUDING REMARKS

In this paper we investigated throughput-number of hops dependency and route discovery time for two most popular MANET routing protocol (AODV-UU and DSR-UU) in stationary environment. Instead of using simulation model, our analysis is based on throughput performance comparison of AODV-UU and DSR-UU in real multi-hop static environments, during FTP transfer of fixed size data files over different wireless chains.

Although all communication principles between wireless nodes are still based on either AODV-UU or DSR-UU routing protocol, we schedule communication links between nodes using predefined static routes. Specially developed java based configuration and management software utility with GUI is used for selection of static routes between wireless nodes working in ad-hoc mode. Practical implementation of predefined static route selection in MANET may be foreseen in applications such as: choosing between multiple ISP wireless gateway links, network monitoring, firewall and charging purposes. Taking into account particularities of our testbed environments, we show that major differences of throughput performance between AODV-UU and DSR-UU are noticed during single hop transfer. We also show that throughput performance of AODV-UU and DSR-UU scale almost the same if compared with theoretical scaling law. In situations of link breakage, route discovery time when AODV-UU protocol is used is less in comparison with route discovery time for DSR-UU protocol for same wireless transmission chains.

REFERENCES

[1] Linux iptables tool homepage http://www.netfilter.org/

[2] Linux nmap tool homepage http://insecure.org/nmap/

[3] Piyush Gupta, Robert Gray, and P. R. Kumar, ``An Experimental Scaling Law for Ad Hoc Networks.", University of Illinois, Urbana-Chanpaign,May16,2001. http://decision.csl.uiuc.edu/~prkumar/html_files/postscript_files.html

[4] P. Gupta and P. Kumar, “ The capacity of the wireless networks”, IEEE Transactions on Information Theory, vol. IT-46, pp. 388-404, March 2000, http://pdos.csail.mit.edu/papers/grid:mobicom01/paper.pdf.

[5] RFC 3561 AODV Routing http://www.ietf.org/rfc/rfc3561.txt

[6] RFC 4728, (February 2007), The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4, http://www.ietf.org/rfc/rfc4728.txt

[7] Charles E. Perkins, Elizabeth M. Royer, Samir R. Das, Mahesh K. Marina, "Performance comparison of two on-demand routing protocols for ad hoc networks", *IEEE Personal Communications,* vol. 8, no. 1, February 2001 pp. 16-28

[8] Sumit Gwalani, Elizabeth M. Belding-Royer, Charles E. Perkins, "AODV-PA: AODV with path accumulation", *ICC 2003 - IEEE International Conference on Communications,* no. 1, May 2003 pp. 527-531

[9] F. Bertocchi, P. Bergamo, G. Mazzini, M. Zorzi, "Performance comparison of routing protocols for ad hoc networks", *GLOBECOM 2003 - IEEE Global Telecommunications Conference,* no. 1, Dec 2003 pp. 1033-1037

[10] Prasun Sinha, Raghupathy Sivakumar, Vaduvur Bharghavan, "Enhancing ad hoc routing with dynamic virtual infrastructures", *IEEE INFOCOM 2001 - The Conference on Computer Communications,* no. 1, April 2001 pp. 1763-1772

[11] Suli Zhao, Ivan Seskar, Dipankar Raychaudhuri, "Performance and scalability of self-organizing hierarchical ad hoc wireless networks", *WCNC 2004 - IEEE Wireless Communications and Networking Conference,* no. 1, March 2004 pp. 132-137

[12] Dmitri D. Perkins, Herman D. Hughes, Charles B. Owen, "Factors affecting the performance of Ad Hoc networks", *ICC 2002 - IEEE International Conference on Communications,* no. 1, April 2002 pp. 2048-2052

[13] Y.-H- Han, J. Choi, S.-H. Hwang, Reactive handover optimization in IPv6 –Based Mobile networks, IEEE Journal of Selected Areas in Communications, No. 9., Vol. 24, September 2006, pp.1758.-1772.