Multipath routing in MANETs using Multiple Description Coding

Eddy Cizeron and Salima Hamma
University Of Nantes, Nantes Atlantique Universits
IRCCyN, CNRS UMR 6597, Polytech'Nantes, rue Christian Pauc - BP50609
44306 Nantes cedex 3 France
{eddy.cizeron, salima.hamma}@univ-nantes.fr

***Abstract*—Routing in ad hoc networks is a well known issue. Most of the previous propositions to route data between two nodes aimed to define a path (sometimes several ones) on which packets are sent. In this paper we investigate a new strategy which combines a new multipath routing protocol, called Topology Multipath Routing (TMR) and multiple description coding (MDC). With this kind of coding methods, groups of original packets are turned into pieces of information called descriptions. Any subset with sufficient number of descriptions contains enough information to enable reconstruction of the original packets. Thus, by sending descriptions on different routes, we make each of theses routes less critical.**

Selected routes aims to be disjoint, but, contrary to usual multipath ad hoc routing protocols, this property is not mandatory. Simulations and evaluation of performances are made on NS2 for different values of MDC parameters. A comparative study with the well known reactive protocol DSR is also realized.

I. Introduction

A extensive litterature has been produced on routing protocols in ad hoc networks. Among popular protocol, DSR (Dynamic Source Routing) is often seen as the paragon of reactive protocols whereas OLSR (Optimized Link State Routing) is its proactive counterpart. Those classical protocols create a single route between communicating nodes and try to adapt it during transfer. Some multipath protocols have also been proposed (see [1], [2] and [3]). However, most of them aim to select disjoint routes (disjoint by links or by nodes). In fact, it may not be always possible to find strictly disjoint paths. Path diversity is generally limited, especially near the source or the destination: the number of neighbours of the source and of the destination are upper bounds for the number of possible disjoint routes. Furthermore, even if such routes are available, the disjointness strategy may favour long (and thus unstable) routes.

Concerning multipath protocols, the dispatching strategy is seldom mentioned; although papers like [6] focus on a recovery strategy, several routes are computed but only one is selected for use at a given time (in fact it is more a single path strategy). Some papers like [4] focus on the strategy for data transmission by using network coding methods. However, the route selection is quite simplistic. Countrary to DSR, the destination can answer to several request messages so that the source can receive several replies with different routes. This scheme is quite ineffective on a long distance as the requests that reach a local area have generally gone through very similar paths. In [7], we have checked that combining information redundancy to multipath routing can theorically improve the reliability of transmission. In [8], an extension of OLSR has been proposed (called MP-OLSR) and implemented on NS2. Coding methods have been added in order to transform data before dispatching the pieces. Simulation has showed some benefits can be obtained.

As reactive protocols differ from proactive ones, this paper proposes to check if the same method can be applied in a reactive context. Of course, as OLSR gathers a global vision of the network topology in each node, finding routes and controling the degree of their disjointness is easy (the source can design all paths at the same time). However, classical reactive protocols and their multipath counterparts generally let reply messages from the destination draw routes to the source. At the countrary, we propose in this paper to adopt an approach in which information is gathered at the source (although in a reactive way). The routing strategy, TMR, combine at the same time the classical request/reply method with a selection of routes by the source.

The rest of the paper is organised as follows: in section II, we present different strategies explaining how several routes can be used in parallel for a single transfer. Data may be encoded with a multiple description strategy. Then, section III is dedicated to the specification of our proposed routing method. In section IV, parameters and evaluation criteria for NS2 simulations are introduced. Simulation results are presented and analysed in section V. Finally, we present our conclusions in section VI.

II. Using route(s)

Using a single route is a simple issue: all data are sent on it. But if we have more than one route, how should data be distributed among them? This section deals with the possible strategies for dispatching information on several routes.

Suppose we know N routes from a source node `S` to a destination node `D`. In case of transfer, a simple way to transmit packets is to dispatch them on the N routes so that every routes eventually carries the same number of packets. In this case, the local rate λ_{loc} of data is smaller than the global one λ_{gl}. The routes may overlap so rate in one node may not always be λ_{gl}/N, however we can still consider it is smaller than λ_{gl}. If routers do not have the capacity to deal with a high rate,

978-0-7695-3841-9/09 $26.00 © 2009 IEEE
DOI 10.1109/WiMob.2009.55

traffic congestion may occur. By dividing the data flow we expect to improve transmission performances. On the other side, dividing data can require the use of longer and more unstable routes, that may disappear quickly. Let us imagine another strategy. By duplicating every packet on each route, we just have to expect than at least one of the duplicated version would reach the destination. Thus, the impact of the network unstability decreases as it becomes very unlikely that all routes fail together. The drawback of such a property is that global rate significantly increases and we may expect a lot of routes to be glutted. A compromise would be that we do not need all the routes to reconstruct the original flow but that a single route is not sufficient either.

A. MDC

Multiple description coding (MDC) is a set of coding methods in which data are not encoded into a single output but in several ones, called descriptions (see [9] for a general overview). The more descriptions that are received, the better original data can be recovered. Suppose we have a piece of information I that is encoded to produce N descriptions $(D_1, \cdots, D_N)$. Some descriptions may be lost during transmission. Because information is lacking we cannot reconstruct perfectly I but we may be able to reconstruct an approximation $\hat{I}$. Supposing the information space is fitted with a metric d, MDC implies that $d(I, \hat{I})$ decreases when Z, the number of descriptions received, increases.

Nevertheless, natural metrics are not defined on every kind of data. Furthermore, if we want to apply MDC to various kinds of data at the same time, finding a general notion of approximation might not be easy. However, one can consider a binary version of MDC: either distortion is maximal or it is null. This means that original data can be either perfectly recovered or it cannot be at all. Such a coding well fits situations where the nature of original data is not known. As the routing layer is generally supposed not to be concerned about this aspect, we may suppose that such binary MDC are suitable. In an optimal context, a set of N descriptions for which any subset of at least M elements allows reconstruction can be designed with every description having the same size $size(D_i) = 1/M \cdot size(I)$. Thus, all the N descriptions have size $N/M \cdot size(I)$. In such a context, the redundancy of information created by MDC has a similar impact on the increasing of information size and the ability to cope with loss.

Systematic coding is a special case in which parts of the original data I are directly used as descriptions. Redundancy descriptions are simply added such that a total of N descriptions can be used with the reconstruction property. A simple example, if I can split as $I = I_1 || I_2$, is to choose $D_1 = I_1$, $D_2 = I_2$ and $D_3 = I_1 \oplus I_2$ (this corresponds to $(M, N) = (2, 3)$). As a consequence, in systematic coding, all descriptions may not have the same importance. If information I_1 and I_2 can be processed individually (for example if they are UDP packets), they are more important than D_3 which is useless if alone. In the rest of our paper, we will consider that when systematic coding is applied, the M first descriptions are in fact original data packets.

B. Descriptions on routes

By sending one description on each one of the N routes from `S` to `D`, we can introduce a trade-off between reducing the local rate ($\lambda_{loc} = \lambda_{gl}/M$) and reducing the relative importance of routes ($N - M$ is the number of routes that can be lost without preventing correct data transmission). Figure 1 is an example that illustates such a configuration.

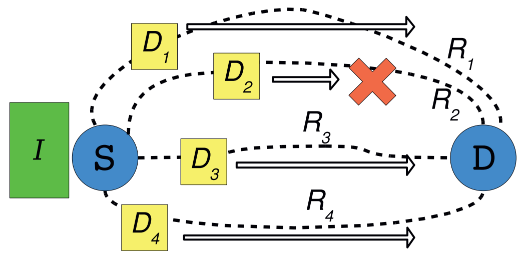


Fig. 1. Sending $N = 4$ descriptions such that at least $M = 3$ are sufficient

III. Specification of TMR

TMR (Topology Multipath Routing) is a multipath reactive protocol (i.e. each node begins to look for routes when it has data to send). Contrary to existing reactive protocols, the topology information is gathered in the source, which then defines routes to destination. TMR functioning can be split into 5 phases:

- **Request procedure**: The source `S` sends requests to reach the destination `D` and warn it that a transfer is expected to begin.
- **Reply procedure**: The destination sends replies back in order to collect topology information for the source.
- **Route computation**: Considering the information received, the source defines a predefined number of routes.
- **Data transmission**: Data are sent, by possibly converting original packets into descriptions and dispatching them on routes (several strategies are described in section IV).
- **Route maintenance**: If some routes disappear, intermediate nodes can deflect descriptions to new routes. Furthermore request and replies are periodically exchanged in order to update the source's vision of the network topology.

A. Request procedure

Just as in standard reactive protocols, requests are used by `S` to reach `D`. Network is thus flooded with requests. A specific sequence number defined by source is used by every node to check if the request corresponds to a new transfer. As usual, following requests are not broadcast anymore. Requests also allow every reached node to update its neighbourhood. Indeed, if node `V` received a request, even if this request is not the first one, `V` can infer that the previous sender `W` is one of its neigbour. Furthermore, requests contains information about its distance from S (by considering the minimum distance covered by requests) and a relay to S. The relay of node V to node S is the neighbour of V that is considered the most suitable to reach S. Here, the relay is chosen among the neighbours that send a request that has covered minimum distance (for this to work, we need to assume that most of links are symmetrical). Eventually, at the end of the request procedure (when every possibly reachable node has been reached), nodes have up-to-date knowledge about their neighbourhood, their distance to S and a possible relay to S.

B. Reply procedure

As usual, replies are broadcast from the destination with a specific sequence number to point out they are not out-of-date. The reply procedure of TMR differs from its counterpart in protocols like AODV and DSR. Its purpose is not anymore to let a single reply "draw" the route from D to S. Replies must gather "interesting topology" information in S. This means that it is not the whole topology but only part of it that is more likely to be used by multiple routes between S and D. This zone corresponds to the set of nodes that are not too far from the shortest path. We choose it to be the ellipse $E = \{\mathtt{V} : d(\mathtt{S},\mathtt{V}) + d(\mathtt{V},\mathtt{D}) < \xi \cdot d(\mathtt{S},\mathtt{D})\}$ (see figure 2), with $\xi > 1$ being a constant and d the hop count distance between nodes.

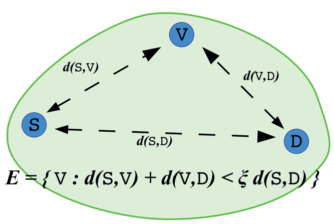


Fig. 2. Ellipse

Of course, the shape of the ellipse may not look as regular as in the figure, depending on the distribution of nodes and the existence of obstacles. However, we believe that an ellipse is a suitable choice. Most of the nodes inside are relevant to a transmission between S and D as they are close to at least one of them (relatively to the distance $d(\mathtt{S},\mathtt{D})$). This procedure share similarities with the EDSR-OPT method used in [4] where a diamond between S and D is used. However, for this to work, authors make the strong assumption that nodes are aware of their geographical position. In TMR reply, we do not need this information.

Gathering topology information about E is in fact equivalent to gathering information about neighbourhoods of nodes inside E. This task is accomplished by replies. The list of the neighbours of a node V will be called the token of V ($T_{\mathtt{V}}$). Contrary to requests, only the ellipse is flooded with replies. This property can be easily warranted (a node V receiving a reply can update its distance from D in a similar way it has already updated its distance from S with requests). We consider that replies also carry the distance $d(\mathtt{S},\mathtt{D})$, initialized by D. That way, every node V reached by a reply knows if it belongs to E. If not, V does not have to participate in the transfer and does not broadcast the reply. If $\mathtt{V} \in E$, node V may have to add its token $T_{\mathtt{V}}$ to the reply before broadcasting it. In fact, two situations are possible, depending on W, the previous node from which V has received the reply:

- If W has selected V as its relay to S, the sequence number of the request is checked. If older than the last known sequence number of D, the reply is discarded. If newer, V adds $T_{\mathtt{V}}$ to the request and then broadcasts it. If the received sequence number is equal to the last one known (meaning V has already received a reply and thus already sent its own token), the reply is directly broadcast without adding topology information.
- If W has not selected V as its relay to S (V is then called a *witness*), the sequence number of the request is also checked. If older than the last known sequence number of D or equal to it, the reply is discarded. If newer, V makes a reply with *no tokens* inside, adds $T_{\mathtt{V}}$ and broadcasts the reply.

To summarize, the purpose of the distinction between relays and witnesses is to guarantee that each node inside E transmits its token to S *only once*. Every token is then transported to S hop by hop from one relay to another. Every witness, if it still has its own token, makes a new reply with this token inside. At the end of the reply procedure, S has received tokens of all nodes inside the ellipse (including D). The replies might become large if they are added a lot of tokens or large tokens. Nevertheless, it is possible to split them before their transmission to the next node. Conversely, if two small replies reach the same intermediate node, they can be merged by gathering the tokens inside a single new reply.

C. Route computation

Information in tokens enables S to construct a partial image of the network topology. S can then defines routes. For this purpose, we use the algorithm described in [7]. This algorithm, based on well known Dijkstra algorithm requires that the image of the network is represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, cost)$ where $\mathcal{V}$ is the set of vertices (corresponding to nodes), $\mathcal{E}$ the set of edges (corresponding to physical links) and *cost* a function that gives to every edge a weight. A weight is a number in $\mathbb{R}^+$ that must represent the stability of the link. The weight values can be sent during response procedure inside tokens. If they are not, 1 is choosen as a default value. A metric like ETX, proposed in [5] is an possible example of a more complex metric. Computing N routes in $\mathcal{G}$ between S and D consists in the following steps:

1) apply Dijkstra algorithm to $\mathcal{G}$ to find the shortest route R_i;
2) increase all weights of links in $\mathcal{G}$ that belongs to R_i (using $newcost \leftarrow f_p(oldcost)$);
3) increase all weights of links in $\mathcal{G}$ that leads to a node of R_i (using $newcost \leftarrow f_e(oldcost)$);
4) restart step (1) if N routes have not been found yet.

The purpose of such an algorithm is to find multiple routes that are node-disjoint if possible or else link-disjoint if possible. Disjointness is expected, but contrary to most similar algorithms, not mandatory. The algorithm may even select a new route that is not disjoint from previous ones rather than a disjoint but long route. Of course, if two routes share a common segment, the probability of interference on this segment increases. However reliability is also due to the quality of links and the length of routes. So considering disjointness as a *sine qua non* and not a simple tendency can lead to use very long or poor quality routes. Theorically, the values of f_e and f_p have an impact on the degree of disjointness. If $f_e(x) = x$ paths tend to be node-disjoint only; if $f_e(x) = f_p(x)$ paths tend to be link-disjoint only. If $f_e(x)$ and $f_p(x)$ are far greater than x, longer but completely disjoint routes are prefered. Nevertheless, in [7], some of simulations have shown that this parameter does not have a very big impact on reliability, contrary to for example links stability. A simple explanation is that if there are no other possibilities but reusing a some links, the algorithm generally behaves the same way whatever is the increase of weights.

D. Route maintenance

Given that `S`'s view of the network topology may go out of date `S` tries to update its information. Then, as long as data are transmitted, it periodically sends new requests to `D` and waits for new replies. The new requests are only broadcast inside the ellipse and not in all the network.

However, this update procedure may not warrant that every selected route is still correct when being used by packets/descriptions. If an intermediate node `V` realise that `W` the next hop predicted in the header is not valid anymore (`W` is not one of the neighbours of `V`), `V` is allowed to deflect data on a new path to destination. If it cannot find such a path, data are discarded.

IV. Simulation context

In order to test MDC strategies for ad hoc reactive routing, an implementation of TMR on NS2 has been made. Results are also compared with DSR performances on the same scenarios.

A. Simulation parameters

Several scenarios have been simulated. We are mainly considering scenarios with different number of routes required (N) and different redundancy (corresponding to M). For each set of values, 8 different network configurations are tested in order to generate smooth result charts. Table I contains all the parameters values used in different simulations. We have focused on quite dense networks as multipath strategies can arguably be considered as irrelevant in sparse networks.

Scenario parameters	
Number of scenarios	8
Node numbers n_{tot}	50, 75 or 100
Area	$1000m \times 1000m$
Duration	300 s
Number of transmissions	30 CBR (on UDP)
Durations of transmissions	20 s
Packet rate	10, 25 packets/s
Size of every packet $size(P)$	512 B
Mobility parameters	
Mobility model	Random Waypoint
Pause time	5 s
Minimum speed	5 m/s
Maximum speed	10 m/s
Low layers parameters	
MAC Protocol	IEEE 802.11
Reflexion model	Two-ray ground
Range of nodes	175 or 250 m
TMR parameters	
TMR modes	Round robin, Systematic MDC, Non systematic MDC
N	1,2,3,4,5 or 6
M	$1 \leq M \leq N$
Weight increment function	$f_p(c) = c + 1000$ $f_e(c) = c + 500$

TABLE I
TESTS PARAMETERS

B. Coding and dispatching modes

There are three possible modes used in TMR:

1) **Round robin mode**: Data packets are just dispatched on the N available routes in such a way that every route carries the same number of packets.
2) **Non systematic coding mode**: Packets are grouped in sets of M elements (with $1 \leq M \leq N$). Each group is then transformed, using MDC, into N descriptions such that any subset of at least M descriptions allows reconstruction of the group. The N descriptions are dispatched on the N routes.
3) **Systematic coding mode**: Similar to previous mode except that among the N descriptions some are in fact the original packets. Considering a group of M packets, $N - M$ redundancy descriptions are generated. The N elements (M original packets + $(N - M)$ redundancy descriptions) form a set such that any subset of at least M descriptions allows reconstruction of the group. The N elements are dispatched on the N routes.

Coding and decoding procedures do require additional delays so it can theorically make transmission longer. However those delays are assumed to be negligible compared to the others. Some experimentations we have made with MP-OLSR protocol and coding methods has confirmed this hypothesis.

In order to guaranty that every packet/description follows the route selected by the source, the route is added to the packet/description using a dedicated header. This results in specific overhead (just like in DSR) and the size of descriptions is then increased. However, as for tested scenarios routes contains at most about 8 nodes, this overhead is not considered significant when compared to the size of a description.

C. Performance criteria

In order to compare performances, we need to define criteria:

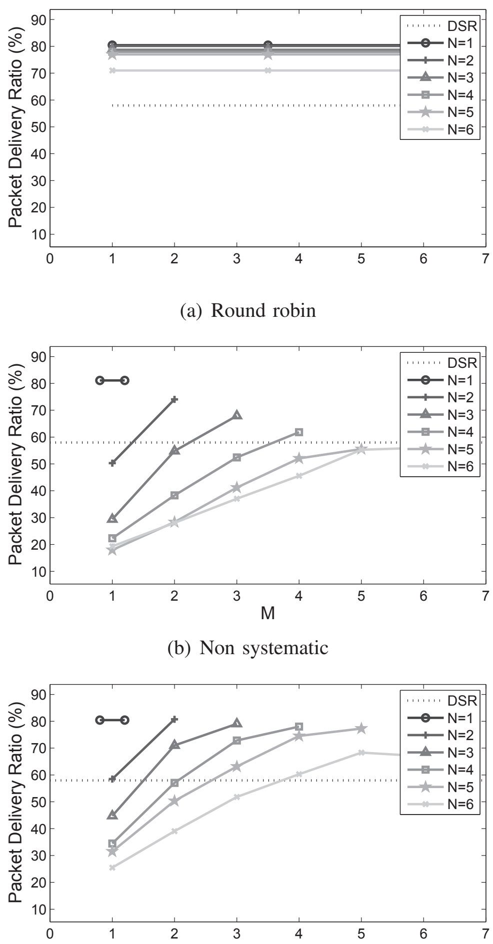
- the delivery ratio, defined as the ratio between the number of received data packets over the number of generated ones (indicates if the protocol can transmit data efficiently);

- the average delay, defined as the duration necessary for received data packets to reach their destination (indicates if the protocol is fast);
- the routing overhead, defined as the ratio beetwen the number of sent routing packets over the number of received data packets (indicates in which extent the protocol requires specific exchanges to work).

V. Results and analyses

A. Impact of coding and dispatching mode

We here consider simulations of 100 nodes exchanging 10 packets by seconds. The common range for nodes is 175 m.



(a) Round robin

(b) Non systematic

(c) Systematic

Fig. 3. Delivery ratio, 100 nodes, 10 pkt/s, range of 175 m

By analysing figures 3 (a), 3 (b) and 3 (c), we can see that dispatching information over different routes does not seem to improve delivery ratio in comparison with $N = 1$. In general the result with $N > 1$ is similar to $N = 1$, except when N is too big. By using non systematic coding, the delivery ratio is always better when M increases (when the redundancy is low) for a N constant, and is improved when N decreases (when few routes are used). It means that, in general, non systematic coding is not a good option. Concerning the systematic version, results are better, with cases $N = M$ providing a delivery ratio similar to the single route case (but which do not exceed it). On this point, one must keep in mind that systematic coding where $N = M$ is in fact similar to round robin. Thus, information of 3 (a) is contained in 3 (c). Furthermore, as the non systematic coding strategy performs poorly for delivery ratios, we can focus on the systematic coding one. Compared to DSR, in this context, TMR performs better for most of the value (M, N).

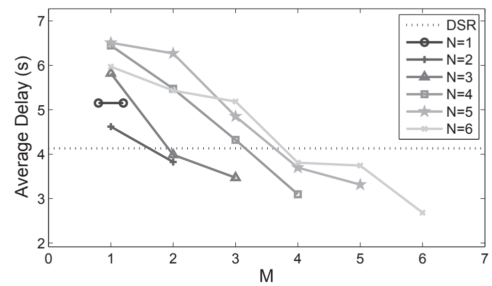


Fig. 4. Delay, 100 nodes, 10 pkt/s, range of 175 m, systematic coding

The analyse of the delay (see figure 4) shows that, although DSR has generally a shorter delay, multipath TMR (i.e. $M = N$) performs better (case $M = N = 6$ is irrelevant because the small delay may be explained by the decreasing delivery ratio). The coding strategy does not seem efficient.

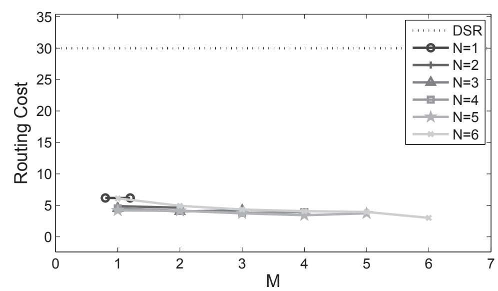


Fig. 5. Routing overhead, 100 nodes, 10 pkt/s, range of 175 m, systematic coding

Figure 5 shows that TMR requires far less routing packets than DSR, whatever the route strategy is (this is coherent with the fact that the ellipse is always the same no matter how many routes are searched).

B. Impact of node density

Figures 6, 7 and 8 present the delivery ratio for a range of 250 m with transfers of 25 pkt/s and respectively 50, 75 and 100 nodes. As the network goes denser, DSR performances decrease but TMR performances are not affected.

The delay becomes even shorter for multipath TMR (for example from 4.5 s to 3.7 s for $N = M = 3$) and increases

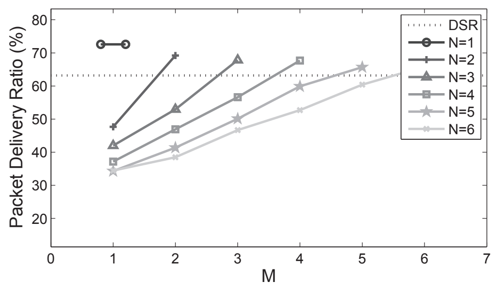


Fig. 6. Delivery ratio, 50 nodes, 25 pkt/s, range of 250 m, systematic coding

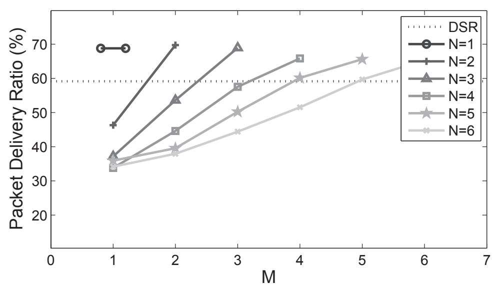


Fig. 7. Delivery ratio, 75 nodes, 25 pkt/s, range of 250 m, systematic coding

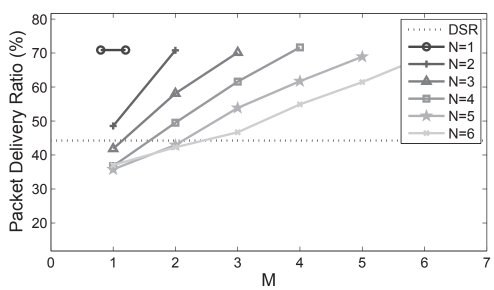


Fig. 8. Delivery ratio, 100 nodes, 25 pkt/s, range of 250 m, systematic coding

for DSR (from about 2.2 s to 4.5 s) so that TMR has shorter delay in a 100 node network (see figure 9). Routing overhead is always smaller for TMR (between 1 and 2) than for DSR (from 1.2 to 25).

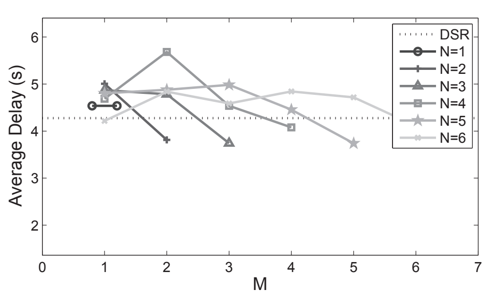


Fig. 9. Delay, 100 nodes, 25 pkt/s, range of 250 m, systematic coding

VI. Conclusion

This paper has introduced a multipath reactive routing protocol that looks for possibly (but not necessarily) disjoint routes. The purpose was to check if, in a reactive context, dispatching data on several paths with several coding methods was beneficial. We have thus compared three modes: using round robin to merely distribute packets among routes, using multiple description coding to create redundant descriptions of the original information and using a systematic coding in which redundancy descriptions were added to original packets.

As expected, systematic coding has always provided better performances than non systematic coding. Concerning delivery ratio, TMR has exceeded DSR especially when using a single path or in round robin cases. Delay has been reduced thanks to multipath. In case of limited range, it has even become shorter than DSR delay. Moreover, the routing overhead for TMR is globally far better than for DSR. This feature may be a consequence of the restriction of TMR request/reply flooding inside the ellipse, while DSR control messages are generally sent through all the network.

However, coding strategies do not improve TMR performance. One reason could be that the necessary increase of data size due to redundancy has a negative effect on transmission with a stronger impact than the benefit brought by coding protection properties.

A possible perspective could be to test MP-OLSR and TMR with the same scenarios in order to compare selected routes and determine if the stability of MP-OLSR, due to its proactive functioning, can explain its higher performance when using coding.

References

[1] Zhenqiang Ye, Srikanth V. Krishnamurthy, Satish K. Tripathi, *A routing framework for providing robustness to node failures in mobile ad hoc networks*, Ad Hoc Networks, volume 2, pp. 87-107, Elsevier 2004.

[2] Mahesh K. Marina, Samir R. Das, *On-Demand Multipath Distance Vector Routing in Ad Hoc Networks*, Proceedings of IEEE International Conference on Network Protocols (ICNP), pp. 14-23 2001.

[3] S.-J. Lee, M. Gerla, *Split multipath routing with maximally disjoint paths in ad hocnetworks*, Proceedings of IIEEE International Conference on Communications (ICC), volume 10, pp. 3201-3205 2001.

[4] Song Han, Zifei Zhong, Hongxing Li, Guihai Chen, Chan, E., Mok, A.K., *Coding-Aware Multi-path Routing in Multi-Hop Wireless Networks* Performance, Computing and Communications Conference, 2008. IPCCC 2008. IEEE International, pp. 93-100, 7-9 December 2008

[5] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris, *A High-Throughput Path Metric for Multi-Hop Wireless Routing*, Proceedings of ACM MobiCom 2003.

[6] A. Nasipuri, S.R. Das, *On-demand multipath routing for mobile ad hoc networks*, Proceedings of the Eight International Conference on Computer Communications and Networks, pp 64-70 1999.

[7] E. Cizeron, S. Hamma, *A Multiple Description Coding strategy for Multi-Path in Mobile Ad hoc Networks*, Proceedings of the Second International Conference on the Latest Advances in Networks - ICLAN Paris, December 2007.

[8] J. Yi, E. Cizeron, S. Hamma, B. Parrein, *Simulation and Performance Analysis of MP-OLSR for Mobile Ad hoc Networks*, Proceedings of the IEEE WCNC Las Vegas, March 2008.

[9] Vivek K. Goyal, *Multiple Description Coding: Compression Meets the Network*, IEEE Signal Processing Magazine, volume 18, pp 74-93 September 2001