Macrothink Institute™

Network Protocols and Algorithms
ISSN 1943-3581
2013, Vol. 5, No. 4

TCP Performance in Mobile Ad hoc Networks

Sofiane Hamrioui

UMMTO, USTHB, Algeria

University of Haute Alsace, IUT, GRTC, Colmar, France

Tel: +33760870175 E-mail: s.hamrioui@gmail.com

Jaime Lloret

Dept. of Communications, Polytechnic University of Valencia

Camino Vera s/n, 46022, Valencia (Spain)

Tel: 1-234-567-8910 E-mail: jlloret@dcom.upv.es

Pascal Lorenz

University of Haute Alsace, IUT, GRTC, Colmar, France

E-mail: pascal.lorenz@uha.fr

Mustapha Lalam

UMMTO, Algeria

E-mail: lalamustapha@yahoo.fr

Received: December 5, 2013 Accepted: December 28, 2013 Published: December 31, 2013

DOI: 10.5296/npa.v5i4.4773 URL: http://dx.doi.org/10.5296/ npa.v5i4.4773

Abstract

In this paper, we present a survey of TCP (Transmission Control Protocol) protocol for better performance in the MANET (Mobile Ad Hoc Network). After a short presentation of the main features of TCP, we give the most important problems from which TCP suffer in MANET. We present after that some approaches proposed in the literature in order to improve its performance. Our paper contains also a performance evaluation of TCP NewReno and TCP Vegas transport protocols under AODV and DSR routing protocols. The simulations

are conducted under varying conditions of number of TCP connections, number of nodes and mobility.

Keywords: Mobile Ad hoc Networks, TCP Protocol, Problems of TCP in MANET, Improving of TCP in MANET, TCP NewReno, TCP Vegas, Performance evaluation.

1. Introduction

Mobile Ad Hoc Network (MANET) [1] is complex distributed systems which consist of wireless mobile or static nodes. Transmission Control Protocol (TCP) [2], [3] is the transport protocol used in the most IP networks [4] and recently in MANET [5]. It is important to understand the TCP behavior in an ad hoc network. This protocol suffers from several problems related to wireless environments in general and MANET networks in particular.

Our contribution in this paper is to give a survey of TCP protocol in the MANET. After a short presentation of TCP protocol and the most important problems from which it is suffering, we present the various proposals which have been made in the literature to improve the performance of TCP in MANET. In the end of our paper, we present some simulations conducted with NS simulator to evaluate the performance of the versions of TCP which are NewReno and Vegas. We chose AODV and DSR as routing protocols and we use different conditions of the communication by varying the number of nodes, the mobility of nodes and the number of the TCP connections.

The paper is structured in five sections. Section two gives a short presentation of TCP protocol. In section three we present the most important sources of the degradation of this protocol. In section four we study the various approaches proposed in the literature to improve the TCP performance. In section five we present a performance evaluation of TCP NewReno and TCP Vegas. We finish our survey with section five which provides the conclusion and future work.

2. TCP in MANET

Many works [6] describe the functioning of TCP protocol. We will only discuss the most important principles for understanding the behavior of such protocol in MANET. TCP is a transport protocol which provides a mode connected service to the upper layers (session, presentation and application and cutting OSI). TCP provides reliable service and is therefore used in the end-to-end packets communications between two computers. TCP controls the transmitter flow in order to not exceed the capacity of the network (congestion control) and assures that this flow will not be too high relative to the receiver flow (flow control). To satisfy all of these properties, TCP is based on several mechanisms which we present in the following sections.

2.1 Mechanisms for the transmission reability

2.1.1 The use of the acknowledgments (ACK)

TCP is a connected mode protocol which means that the connection is established

between peer communicating entities. To establish this connection, a specific signaling is implemented in three phases (called the three-ways handshake): The issuer sends a synchronization message, acquitted by the receiving entity. The issuer then acquits the acknowledgments and the data transfer can begin.

TCP uses a Sequence Number (NS) to identify each one of the data packets. The NS is therefore used to define the order of packets transmission and at the connection step allows therefore to the end stations to share their NS, it will be incremented after each transmission.

When the volume of information to be transmitted is low, the connection establishment phase may represent a significant portion of the total connection time, especially when the time of the transmission channel is important, which is often the case with communications involving a wireless link.

TCP breaks the data to be transmitted in set of blocks called segments, which are then given to the IP layer. For each segment, TCP adds a header of a typical length of 20 bytes, this header can be extended by optional fields when necessary. The header contains in particular redundancy to test with a very high probability the presence of errors in the received segment. The error detection is performed from end to end. In addition, at the reception, TCP reorders the received packets, manage the duplications and transmits to the upper layer the packets correctly assembled.

For each correct reception of a data packet, the receiving station informs the transmitting station by sending an acknowledgment packet (ACK). The ACK is typically a packet of 40 bytes containing the NS of the next expected packet. This ACK will mean that all packets with NS less than the NS of the ACK were well received (acknowledgment cumulative concept).

It may be that the expected packet by the receiver (NS = X) is lost along the way or simply delayed. The packets with greater NS than X can successfully reach their destination, they are called OOO (Out Of Order), they will be buffered and acquitted, but the NS of their acknowledgment is equal to X. After the receipt of packet X, a reordering is done. An ACK whose the NS is equal to the largest NS of all the well received packets will be sent.

This process can degrade the TCP performance because when buffering a large number of packets when X is lost, the receiver has no way to inform that it has received the other and there is only the X which is missing. The “cumulative ACK” will require the issuer, after retransmission of X, to forward all packets coming after at least pending reordering and thus their ACK.

To avoid these unnecessary retransmissions, an option has been introduced. It is a new acknowledgment SACK (Selective ACK). This option is negotiated when establishing the connections, it allows the destination to select and specify the packet to be retransmitted.

A receiving station can reduce the number of ACK which it can send using a different type of appointed acquirement DACK (Delayed ACK). The DACK is an acknowledgment of a collection of data packets after receiving an X packet, the receiving station waits for the reception of other packets before acquit all. The ACK is delayed and the delay should not be too large to avoid the performance degradation. In practice, the period shall not exceed 0.5 seconds.

2.1.2 The use of the timeout and the transmissions

To achieve good reliability, when TCP sends a segment, it initiates a timer until the segment is acknowledged by the receiving entity. TCP is called auto-synchronized (self clocking). If the ACK does not arrive before that the timer expires, TCP considers that the segment is lost and retransmits it. This waiting period is called RTO (Retransmission Time Out).

Research has shown that the RTO must be determined dynamically. Fair and accurate RTO value improves TCP performance. The calculation of the RTO is based on RTT (Round Trip Time) which is the time required to receive a packet and its acknowledgment. When the RTO expires, the RTO is doubled and retransmission is attempted. The duration of the RTO may not exceed 64 seconds. After 12 attempts retransmissions of a packet without success, the packet is dropped.

2.2 Mechanisms for the flow control

TCP protocol is a self-regulating. It uses indeed in sending a sliding window of variable size. In general, new segments are transmitted, that is to say the transmission window can advance when the precedents packets are acknowledged. In addition, the window size is variable, allowing control of the flow of the transmitted packets. It is bounded by the rxwnd parameter (*Receiver window*) given by the receiving entity, reflecting its ability to handle a finite number of packets. So a small machine with a low processing speed and with only small memory buffer can control the flow of transmission and therefore communicate with a more powerful machine without being drowned by too rapid arrival of segments.

2.3 Mechanism of the congestion control

A typical case of congestion is when at least one station has a buffer saturated, such a station is called bottleneck. All packets to the destination station must be abandoned. In what follows, we will discuss four congestion control algorithms of TCP: Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery.

2.3.1 Slow start and congestion avoidance

The TCP congestion control is based on the use of a new window called the congestion window cwnd. The size of the transmit window is given by the minimum of cwnd and rxwnd. The growth in the size of the congestion window is built around two phases: slow start phase and congestion avoidance phase.

The purpose of the slow start phase is to test the ability of the network, especially at the beginning of connection when there is no a priori knowledge of the crossing network between the issuing entity and the receiving entity or after recovery a lost packet detected with RTO expires. The congestion window is initially set to one segment. After received acknowledgment, it is incremented by one segment. Therefore, this increase is exponentially because each time round trip (RTT Round Trip Time), its size is multiplied by 2. When the size of the congestion window reaches the threshold value (slow start threshold or ssthresh), the issuing entity enters into the congestion avoidance phase.

During this phase, the source increases its CWND using the following formula **(1):**

$$CWND = CWND + MSS * MSS / CWND \quad \textbf{(1)}$$

Congestion avoidance phase corresponds to the steady state. The increase of the congestion window is linear, segment by RTT. It is well adapted to the capacity of the network, without aggressive increase of the transmission rate.

The TCP source remains in this state until the detection of a loss. The decrease in the size of the congestion window, to limit the transmission rate, responds to packet losses which are interpreted by TCP as congestion events connection.

In practice TCP can detect the presence of error in two ways: after a timer expires or after receiving three duplicated acknowledgments. In the first case, the packet in question is not arrived at its destination and the following packets too (since no acknowledgment has been received). TCP deduces that the congestion is severe then it completely closes its congestion window and returns to the slow start phase as follow:

• Repeat the Slow Start phase.
• Position the SSTHRESH to the MAX (TWND / 2, 2 * SMSS)
• Replace the CWND by the size of a SMSS segment

In contrast, in the presence of duplicated acknowledgments, one packet is missing, the others are received as can testify the flow of acknowledgments which continue to arrive. The network is becoming congested. In this case, TCP will not restart from the initial state but reduces its congestion window by a factor of 2.

We summarize the two algorithms combined as follows

1. For each new connection initialize CWND to 1 and SSTHRESH to 64K
2. TWND = min (CWND, AWND)
3. Each time it receives an ACK: CWND = CWND + 1
4. If (CWND = SSTHRESH) then:
    While there is no loss reported by the RTO **do**:
        CWND = CWND + MSS * MSS / CWND
5. **If** (a loss is detected with the RTO) **then**:
    SSTHRESH = max (TWND / 2, 2 * SMSS)
    CWND = 1
    Repeat step **2**

The diagram of **Fig.1** explains more the two phases of slow start and congestion avoidance.

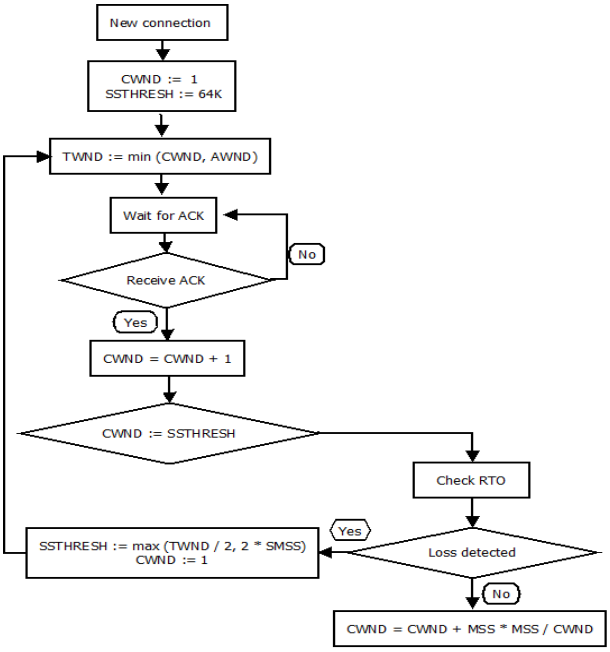

Fig.1. slow start and congestion avoidance phase

2.3.2 Fast retransmit and Fast recovery

TCP principles make that during the reception of an OOO packet, a double ACK is generated. TCP cannot understand whether this double-ACK is due to packets loss or to the disorder in their arrival. TCP expects to receive a triple ACK to make a decision.

In the case of OOO, the reordering time never exceed the receiving time of three ACK. To do this, beyond this period, a TCP source can use such situation as a strong indicator of packet loss. It will begin its retransmission without waiting for the expiration of the RTO. This step is the Fast Retrasmit.

The reception of a triple ACK indicates more than a packet loss. Indeed, it is a proof that a connection between the end stations is always maintained. So the source is not required to reduce its flow because the loss was a rare event and there is no congestion. The three received ACK were generated by the destination when it received three new packets but with a different NS than the one which is expected. Therefore the source deduces that there are three packets which have left the network.

Given that the resources are available, the source enters in the Fast Recovery phase which consist to inject new segments in the network until receive a new acknowledgment (with a NS different from the one of triple-ACK).

The Fast retransmit and Fast Recovery are combined in the implementation. The algorithm is as follows.

1. **Upon** receipt of a triple ACK **do**:

SSTHRESH = MAX (TWND / 2.2 * MSS)

2. Retransmission of lost packet

CWND = SSTHRESH + 3 * MSS (increase CWND with the number of segments that have left the network and which caused the triple ACK)

3. **Upon** receipt of each new duplicate acknowledgment **do**:

CWND = CWND + MSS (as new packets have left the network)

4. As the TWND permits, transmit new packets

5. Upon receipt of an ACK whose NS is different from the previously received duplicate ACK: CWND = SSTHRESH. This ACK will pay the retransmitted packet (the one that caused the Fast Retransmit) as it could pay all lost packets until after receiving the triple ACK packet.

The diagram of **Fig.2** explains more the two Fast retransmit and Fast recovery.

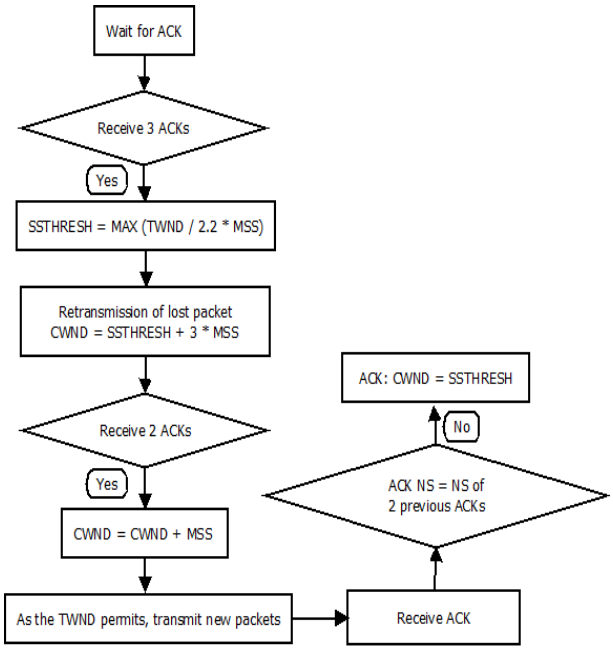

Fig.2. The two phases of Fast retransmit and Fast recovery

It should be noted that it is possible that the algorithm never reaches the final stage because the ACK in question may not arrive, a loss then is detected with the expiration of the RTO. This will remove the source from its state and makes it enter in the Slow Start mode.

3. Problems of TCP in MANET

The performance degradation of TCP protocol is due to the following problems [7][8][9].

3.1 The effects of the high BER (Bit Error Rate)

Wireless media, unlike the wired media knows very high BER (Bit Error Rate). This is mainly due to interference and signal attenuation which corrupt TCP packet (data and ACK) causing their losses. If the TCP source does not receive an ACK within the RTO time interval following the transmission of a data packet, it concludes that the network is congested, which will result: the retransmission of the lost packet, reducing CWND to 1, reducing the threshold

CWND to half and doubling the RTO.

The repeated errors at the transmission channel ensure that the CWND window of the TCP source remains too small and then causing a very low transmission rate and a coarse increase of RTO.

3.2 The effects of the nodes mobility

3.2.1 Effects of the recalculation of roads

Because of the mobility of nodes, the path between TCP source and destination can be broken at any time. The path breaking between the source and the destination initiates the route discovery mechanism, at the source, which will takes some time T.

If (T> RTO):

This will be performed at the source as congestion. TCP invokes then the congestion control mechanism and the retransmission of the lost packet. Thus, when a new road is discovered, the transmission rate continues to be very low during the Slow-Start phase. It is clear that this behavior is undesirable because in a highly mobile environment, the TCP connection will never have the opportunity to use the maximum capacity of the channel to transmit.

If (T <= RTO)

The TCP source continues to transmit in the new route using the old CWND. However, the old value of CWND may very well not be adapted to the new road, causing the loss of relation between CWND and transmission rate of data allowed by a road. Indeed, the value of CWND to the old road can be very large for the new, which will cause a sudden congestion of the network.

3.2.2 Effects of the partitioning of the network

It is very probable that the network is partitioned in a while due to moving or stopping of one or more nodes. If the TCP source and destination are in different partitions, the packets from the source shall be left by the network and invoking the congestion control mechanism of a timeout.

If the network partitioning take much time, the unnecessary retransmissions of the same data to a disconnected station doubles the RTO of the source until it reaches 64s. This will cause the inactivity of the source for long periods even if the link between the source and destination TCP is resets.

3.2.3 Effects of multi-path routing

Some routing protocols maintain multiple routes between the source and the destination (such as TORA) in order to minimize the frequency of recalculation of roads and distribute the load of a TCP connection on several routes. Unfortunately, this can sometimes cause a significant number of packets arrivals in out of order (OOO) at the receiver. This can then generate a duplicate acknowledgments for each packet received, the NS does not match the expected NS. Upon receipt of three duplicate ACK packets, the source invokes the Fast Retransmit / Fast Recovery mechanism (reduction of CWND and the threshold to half of

CWND). This lowers the transmission rate unnecessarily and degrades network performance.

3.2.4 Effects of the competition of the access to the wireless channel

Many researchers have shown that the MAC layer protocols seriously affect the performance of TCP. Indeed, the nature of the shared wireless medium in an ad hoc environment makes that the network stations are always in competition for access to the channel. If in a certain area the number of nodes is very large, collisions will be more frequent, bringing the node which tries to join another station, to conclude, after a number of MAC retransmissions, that the link is broken. This will trigger a useless route discovery mechanism.

Beside this, the schema of random exponential backoff of the IEEE 802.11 standard does not really suits the situation. Indeed, it has serious problems of equity for access to the channel when it favors the node which performed the last successful transmission, which could lead the node to monopolize the channel.

4. Improving of TCP in ad hoc networks

In this section we present the various proposals [10] [11] [12] [13] [14] [15] [16] which have been made in the literature to improve the performance of TCP in MANET. We classify these proposals into four groups according to the four issues which are:

1) TCP is unable to distinguish between the packets losses due to the route failures of roads and those due to network congestion.
2) The frequent roads failures
3) The contention on the wireless channel,
4) Non fairness of TCP.

Note that the problems (1) and (2) are the main causes of the degradation of TCP performance in MANET.

We classify the solutions which belong to the same set in two types: solutions based on the interactions between two layers and solutions based on one layer. The first type is based on the interactions between two layers of the Open Systems Interconnection (OSI) architecture. These solutions were based on the fact that "provide information from the below layer to the top layer should help the top layer to run better". This depends with the two OSI layers between which the information exchange will be. The solutions based on two layers can be further classified into four types: TCP and network layers solutions, TCP and MAC layers solutions, TCP and physical layers solutions, network and physical layers solutions. The solutions based on one layer solutions are based on the OSI layers adaptation independently from the other layers. Also, depending on which layer is involved, these solutions can be further classified into three types: TCP layer solutions, network layer solutions and link layer solutions.

In general, the solutions based on two layers report a better performance than those based only on one layer. But the solutions based on one layer respect the concept of isolation to design protocols, and they are considered as long-term solutions. To choose between these two solutions we need to know what the priority is for us, performance optimization or

architecture. The performance optimization can lead to time savings (short-term), while architecture is usually based on longer-term considerations. In addition, the solutions between layers are more complex to implement and develop than the solutions based on one layer. Because the implementation of solutions between layers requires at least two changes of OSI layers and their design requires the system to be considered in its entirety.

4.1 The solutions to distinguish between the losses dues to the road failures and the losses due to the network congestion.

The solutions which address the problem of the inability of TCP to distinguish losses due to failures of roads and network congestion in MANETs can be classified into two categories: solutions based on TCP layer and solutions based on the interactions between network and TCP layers.

4.1.1 Solutions based on the interactions between network and TCP layers

TCP-F (TCP-Feedback) **[17].** This solution offers the possibility to the issuer to distinguish between the routes failures and the network congestion. In this scheme, the issuer is forced to stop transmission without reducing the window size on the failure of route. As soon as the connection is restored, the fast retransmit is permitted. TCP-F is based on the network layer in an intermediate node to detect the route failure due to the mobility of its down neighbor along the route. An issuer may be in an active state or in a snooze state. In the active state, the transport layer is controlled by the normal TCP. Once an intermediate node detects a broken of route, it explicitly sends a notification packet of the failed route (RFN) to the sender and logs this event. On receiving the RFN, the sender enters the snooze state in which the issuer ceases completely to send other packets and freezes all timers and values of the state variables such as the RTO and the size of congestion window. Meanwhile, all the intermediate ascending nodes which receive RFN make invalid the particular route to avoid further packet loss. The transmitter remains in the snooze state until it is notified of the restoration of the road by the route recovery notification packet (RRN) of an intermediate node, then it resumes the transmission of the frozen state.

Technical based on ELFN. In this approach [18], TCP also interacts with the routing protocol to detect the failure of roads and take appropriate action when it is detected. This is done through explicit link failure notification messages (ELFN) which is returned to the sender node after detecting the failure. Such messages are broadcast by the routing protocol to be adapted for this purpose. In fact, the road failure message of DSR has been modified to carry a payload similar to ICMP (destination unreachable). Mainly, the ELFN messages contain the addresses of the sender and the receiver, the ports and the TCP sequence number. In this way, the modified TCP can distinguish the losses caused by congestion from that due to the mobility. When the TCP sender receives a ELFN message, it enters into standby mode, which means that the timers are disabled and probing packets are sent regularly to the destination to detect the restoration of road. Upon receipt of an ACK packet, the transmitter leaves the standby mode and resumes the transmission normally using its previous timers values.

***ATCP (Ad hoc TCP)* [19].** It uses feedback from the network layer as well. In addition to the route failures, ATCP tries to address the problem of high error rate bits (BER). The TCP sender can be in a persistent state, congestion control state or in the retransmission state. A layer called ATCP is inserted between the TCP and IP layers of the TCP source nodes. ATCP listens to the network status information provided by ECN messages (explicit congestion notification) and ICMP "destination unreachable" messages, so ATCP puts TCP agent in the appropriate state. Upon receipt of a message "Destination Unreachable" TCP agent enters to a persistent state. The TCP agent for this state is blocked and no packet is sent until a new route is found by probing the network. The ECN is used as a mechanism to explicitly inform the sender about network congestion along the route being used. On receipt of ECN congestion control of TCP is usually called without waiting for a timeout event. To detect packet loss due to channel errors, ATCP monitors the received ACKs. When ATCP sees three double ACKs have been received, it does not send the third duplicate ACK but puts TCP in the state of persistent and quickly retransmits the lost TCP packet from the buffer. After receiving the next ACK, ATCP will resume TCP to normal. Notice that ATCP allows interoperability with TCP sources or TCP destinations which don't run an ATCP application. In addition to the failure of road, ATCP tries to address the problem of high BER, network congestion, and replenishment package. These advantages make ATCP a more robust proposal for TCP in MANET.

TCP-BuS (Buffering capability and Sequence information). As in previous proposals, TCP-BuS [20] uses feedback from the network to detect the failure events of routes and make the required decision for this event. The new scheme in this proposal is the introduction of buffering opportunities in the mobile nodes. The authors choose the initiated source on demand routing protocol ABR (Associativity-Based Routing).

The following improvements are proposed:

- Explicit Notice: two control messages are used to inform the source about the route failure and route restoration. These messages are the Explicit Route Disconnection Notification (ERDN) and the Explicit Route Successful Notification (ERSN) messages. Upon receipt of the ERDN from the node which detected the route failure, called Pivoting Node (PN), the source stops sending. And similarly after the restoration of the route by the PN using Localized Query (LQ), the PN transmits an ERSN to the source. Upon receipt of the ERSN, the source continues the transmission of the data.
- Extension of the timeout values: for RCC (Route ReConstruction) phase, the packets along the path from the source to the PN are saved. To avoid timeout events during the RRC phase, the value of the retransmission timer for protected packets is doubled.
- Selective retransmission request: while the value of retransmission timer is doubled, lost packets along the path from the source to the PN are not transmitted until the expiration of the adjusted retransmission timer. To overcome this, an indication is made to the source to be able to retransmit these lost packets selectively.

4.1.2 Solutions based on TCP layer

Fixed RTO. The Fixed RTO [21] is a sender-based technique which does not rely on the feedback network. In fact, the authors use an heuristic to distinguish route failures and congestion. When two timeout expire in order, which corresponds to the situation where the missing ACK is not received before the second RTO expires, the sender concludes that route failure event occurred. The unacknowledged packet is retransmitted but the RTO is not doubled again. This is in contrast to TCP standard, where the exponential backoff algorithm is used. The RTO remains fixed until the route is restored and the retransmitted packet is acknowledged.

TCP DOOR (Detection of Out-of-Order and Response). [22] This is an end-to-end approach, it does not require the cooperation of intermediate nodes and it's based on the events of the next out-of-order (OOO). OOO events are interpreted as an indication of failure of roads. The detection of OOO events is accomplished by a transmitter based mechanism or receiver based mechanism. The transmitter based mechanism uses the non-decreasing property of the sequence number of ACKs to detect the OOO events. With double ACK packets, these ACKs have the same sequence number, so the issuer may need additional information to detect the OOO event. This information is an option byte which is added to the ACK sequence number called ACK Duplication Sequence Number (ADSN). The ADSN is incremented and transmitted with each duplicate ACK. However, the receiver based mechanism requires two additional TCP option bytes to detect OOO events, called TCP Packet Sequence Number (TPSN). The TPSN is incremented and transmitted with each packet including the retransmitted TCP packets. If the receiver detects an OOO event, it should inform the transmitter by placing a specific option bit, called OOO bit in the header of ACK packet. Once the TCP sender learns an OOO event, it takes the two following response steps: congestion control temporarily disabled, and instantaneous recovery for congestion avoidance. In the first action, the TCP sender neutralizes the congestion algorithm for a specific time period (T1). In the latest action, if the congestion control algorithm was called during a given time (T2), the TCP sender should recover to the state immediately before the invocation of congestion control.

4.1.3 Solutions based on the interactions between TCP and MAC layers

IB-MAC (Improvement of Backoff algorithm of MAC protocol). In [23] [24], the authors present an improvement of the interactions between MAC (Medium Access Control) and TCP (Transmission Control Protocol) protocols for better performance in MANET. This improvement is called IB-MAC (Improvement of Backoff algorithm of MAC protocol) and proposes a new backoff algorithm. The principle idea is to make dynamic the maximal limit of the backoff interval according to the number of nodes and their mobility. IB-MAC reduces the number of collisions between nodes. It is also able to distinguish between packets losses due to collisions and those due to the nodes mobility. The evaluation of IB-MAC solution and the study of its incidences on MANET performance are done with TCP New Reno transport protocol. The authors varied the network conditions such as the network density and the mobility of nodes. Obtained results are satisfactory and they showed that IB-MAC can outperform not only MAC standard, but also similar techniques that have been proposed in

the literature like and MAC-WCCP [25] and MAC-LDA [26].

Wireless Congestion Control Protocol (WCCP). In this work, the authors show that TCP suffers from severe performance degradation and unfairness. Realizing that the main reason is the poor interaction between traditional TCP and the MAC layer, they propose a systematic solution named Wireless Congestion Control Protocol (WCCP) [25] to address this problem in both layers. WCCP uses channel busyness ratio to allocate the shared resource and accordingly adjusts the sender's rate so the channel capacity can be fully utilized and fairness is improved.

MAC-layer LDA (Loss Differentiation Algorithm). The goal here is to adapt one of the MAC parameters, the Retry Limit (RL), in order to reduce the drop in performance due to the inappropriate triggering of TCP congestion control mechanisms. Starting from this, a MAC-layer LDA (Loss Differentiation Algorithm) is proposed [26]. This LDA scheme is based on the adaptation of the RL parameter depending on the quality of the 802.11 wireless channels.

4.2 The Solutions to reduce the routes failures

The proposals which address the problem of frequent failures of roads in MANETs can be classified into three categories: proposals between TCP and network layers, proposals between network and physical layers and proposals for network layer.

4.2.1 Solutions based on the interactions between network and TCP layers

Split TCP. TCP connections which have a large number of hops suffer from the frequent routes failures due to mobility. To improve the throughput of these connections and to solve the problem of non-equity, the Split TCP scheme [27] was introduced to cut the long TCP connections to shorter located segments. The interface node between two located segments is called proxy. The routing agent decides whether the node as a proxy using the inter-proxy distance setting. The proxy intercepts the TCP packets, stores them and after that acknowledges their reception to the source (or previous proxy) by sending a local acknowledgment (LACK). Also, the proxy is responsible for delivering the packets at an appropriate rate, to the next local segment. On receipt of a LACK (of the next proxy or final destination), the proxy will serve the packet's buffer. To ensure the reliability of the source to the destination, an ACK is sent from the destination to the source similarly to the standard TCP. In fact, this scheme also cut transport-layer functionality to those of congestion control and end-to-end reliability.

This is achieved by using two transmission windows at source which are the congestion window and the end-to-end window. The congestion window is a part of the end to end window. While the congestion window varies depending on the arrival rates of the next proxy LACKs, the end-to-end window vary depending on the arrival rates of the end-to-end ACK of the destination. In each proxy, there would be a congestion window which governs the sending rate between proxy.

4.2.2 Solutions based on the interactions between network and physical layers

***Preemptive routing in ad hoc networks*.** [28] In MANET, TCP can suffer from the long empty periods induced by frequent failures of routes. This proposal addresses this problem by reducing the number of failed routes. Also, it reduces the latency of route reconstruction. These are achieved by switching to a new route when it is expected that the link of the current route will fail in the future. This technique is coupled with AODV and DSR on demand routing protocols. The route failure detection mechanism is based on the use of power. More specifically, when an intermediate node along the route detects that the signal strength of a received packet from the ascending node falls below a given pre-emption threshold, this intermediate node detects a route failure.

On the detection of the event, the intermediate node will inform the source of the route. Upon receipt of such notice, the routing agent of source searches proactively a new route. When the new route is available, the routing agent switches to the new route. The threshold value of preemption appears to be critical. Indeed, in the case of a low threshold value, there will not be enough time to find an alternative route before the route fails. In the case of high value, the warning message will be generated too early. To overcome the fluctuations of the received signal strength due to fading channel and multipath effects, this can trigger a warning of route preemptive and cause an unnecessary flooding for a route request, the authors use a short and repeated and probing message to verify the accuracy of the warning message.

The links management based on the signal strength in the ad hoc network. This algorithm [29] is similar to the previous one. However, in this algorithm each node maintains a record of the received signal strengths from the one hop neighbor nodes. Using these records, the routing protocol predicts the event of the links failure in the near future (after 0.1 s), this prediction is called the proactive management of the link. On the detection of the event, the routing agent of the source is notified by Going Down message. Upon receipt of this message the routing agent of the source stops sending packets, and starts the process of route discovery. The novelty of this proposal is the reaction mechanism of the link management. This mechanism increases the transmission power to restore the broken link. Both reactive and proactive link management mechanisms can be coupled as follows: to predict that a link will be down, the routing agent of the node informs the source to stop sending, and this node increases its transmit power to handle the packets which are in transit and which are using this link.

4.2.3 Solutions based on the network layer

Backup path routing. This proposal [30] is to improve the availability of TCP path connections using multi-path routing. The authors found that the original multi-path routing degrades the performance of TCP. This is due to the inaccuracy in the measurement of average RTT and the out-of-order packet delivery. Thus, they introduce a new variation of the multi-path routing, called the backup path routing. The proposal of backup path routing maintains multiple paths from source to destination, but it only uses one path at any time. When the current path is broken, it can quickly switch to an alternate path. Using simulations, the authors found that the use of a primary path and an alternate path for each destination reported the best performance of TCP.

For the paths selection, the authors consider two schemes. The first scheme is to choose the shortest path with number of hops as primary and the shortest path in time as an alternative. The second scheme is the choice of the shortest path in delay as the primary path and the disjoint to maximum as an alternative. The disjoint to maximum alternative path is the path which has the few intermediate nodes covered with the primary path. Comparing the two techniques of choice, they found that the first scheme is more efficient than the second. This result is because with the second arrangement, the roads tend to be longer in number of hops. Comparing the performance of TCP over DSR routing protocol with backup path routing, the authors report an improvement in TCP throughput up to 30% with a reduction of routing overheads.

4.3 The Solutions to reduce the contention on the wireless channel

The proposals which address the problem of contention on the wireless channel can be classified into three categories: solutions at the TCP layer, solutions at network layer and solutions at link layer.

4.3.1 Solutions based on the TCP layer

Dynamic delayed ACK. This approach [31] aims to reduce the contention on the wireless channel, reducing the number of TCP ACKs sent by the receiver. This is a modification of the delayed ACK option (RFC 1122) which has a fixed coefficient d = 2. In fact, this number represents the number of TCP packets which the TCP receiver should receive before the acknowledgement of these packets. In this approach, the value of *d* is not fixed and changes dynamically with the sequence number of the TCP packet. For this reason, the authors define three levels l1, l2 and l3 such that *d* = 1 for the packets with the sequence number N smaller than l1, *d* = 2 for the packets with l1 <= N <= l2, *d* = 3 when l2 <= N <= l3 and *d* =4 when l3 <= N. They suggest, for best performance, ensure that *d* is a function of the congestion window of the sender instead to be function of the sequence number.

4.3.2 Solutions based on the network layer

COPAS (contention-based Path Selection proposal) [32]. This solution addresses the problem of poor TCP performance due to contention on the wireless channel. It uses two techniques. The first is disjoint routing and the reversed routes, which consist to choose disjoint routes for TCP data packets and TCP ACK packets. The second is the dynamic contention balancing, which consist to update dynamically the disjoint routes.

Once the route controversy exceeds a given threshold, called the backoff threshold, a new and less restraint route is selected to replace the higher contention route. In this proposal, the contention on the wireless channel is measured by the number of times which a node performs a backoff during each time interval. Also at any time a route is broken, in addition to the initialization of the routes recovery procedure, COPAS redirects TCP packets with the second alternative route.

Comparing COPAS and DSR, the authors found that COPAS is better than DSR in terms of routing overheads and TCP throughput. However, the use of COPAS, as reported by the authors is limited to static networks or networks with low mobility. Because, while the nodes

move faster using a disjoint forward and reverse routes increase the probability of routes failures known by the TCP connections. This may induce more routing overheads and more packet losses.

4.3.2 Solutions based on the MAC layer

Link RED (Random Early Detection Link) [33]. It aims to reduce the contention on the wireless channel. This is done by monitoring the average number of retransmissions (avg) at the link layer. When the number avg becomes larger than a given threshold, the probability of dropping / marking is calculated according to the RED algorithm. Since it marks packets, Link RED can be coupled to ECN to inform the TCP sender about the congestion level. However, instead of informing the TCP sender about the congestion level, the authors increase the backoff time at the MAC layer.

Adaptive pacing. The purpose of this approach [34] is to improve the spatial reuse of channels. In the current IEEE 802.11 protocol, a node is forced to challenge for the channel by a random backoff period, and one packet transmission period is announced by RTS or CTS frames. However, the problem of exposed receptor persists due to the absence of coordination between nodes which are far from each other in two jumps. The adaptive measure solves this problem by increasing the backoff period by an additional packet transmission time. This proposal works with the RED link as follows. Adaptive action is permitted by RED Link. When a node finds its average number of retransmission less than a threshold, it calculates its backoff time as usual. When the average number of trying exceeds this threshold, the adaptive measure is allowed and the backoff period is increased by a period equal to the period of transmission of the previous packet. Working together, link RED and adaptive measure reported an improvement in TCP throughput as well as fairness among multiple TCP sessions. However, in this solution, the additional backoff time is based on the packet size. Thus the existence of different sizes of data packet in the network should be inspected.

Neighborhood RED [35]. This proposal aims to increase the fairness of TCP in MANET. In contrast to wired networks, the authors show that RED does not solve the TCP unfairness in MANET, because the congestion does not occur in one node, but in an entire sector involving multiple nodes. The packets local queue within each individual node can not completely reflect the state of network congestion. For this reason, the authors define a new distributed queue, called neighborhood queue. At a node, the neighborhood queue should contain all the packets which transmissions affect its own transmission in addition to its packets.

Since it is difficult to obtain information on all these packets without presenting significant communication overhead, which may require an exchange of information in two jumps, a simplified neighborhood queue of node is introduced. It aggregates the local queue of the node, and the upstream and downstream queues of its one hop neighbors. Now the RED algorithm is based on the average size of the waiting queue of the neighborhood queue. The authors employ a distributed algorithm to calculate the average size of queue. In this algorithm, the time is divided into slots. During each slot, the channel idle period is measured.

Using this measurement, a node considers the channel utilization and the average size of the neighborhood waiting queue. The accuracy of the evaluation is controlled by the length of the slots. Using simulations, they check the effectiveness of their proposal and the improving of TCP fairness.

Non- work- conserving scheduling [36]. The purpose of this proposal is to improve the fairness between TCP flows in ad hoc wireless networks and wired networks. The authors adopt the policy of "non work conserves scheduling" for ad hoc networks instead "work conserving scheduling". The queue of the link layer sets up a timer each time it sends a data packet to MAC. Only after the timer expires, the queue product another packet to MAC. The duration of the timer is updated according to the value of output rate of the queue.

Specifically, the timer duration is the sum of three sides of D1, D2, and D3. D1 is equal to the length of the data packet divided by the channel bandwidth. D2 is a time whose value is determined by the recent output effeciency of the queue. The queue calculates the output rate by counting the number of bytes, C, of its output at each fixed interval T. To determine the value of D2, the authors place three thresholds X, Y, and Z (X < Y < Z) for C, and four delay values D21, D22, D23 and D24 (D21 < D22 < D23 < D24) for D2. According to the simulations, the authors report that their scheme significantly improves the fairness among TCP connections to the average total cost of damage output.

5. Evaluation of TCP Vegas and TCP NewReno performance in MANET

This section contains details of the simulations conducted and the results of the evaluation of TCP NewReno and TCP Vegas performance under AODV and DSR routing protocols. The simulations are conducted under varying conditions of number of TCP connections, number of nodes and mobility. First, the parameters of the simulations are presented and discussed. Then the results on the performance of each protocol are discussed.

5.1 The parameters of the evaluation

Important metrics were used, the throughput of TCP, the goodput, the average time from start to finish and routing traffic.

The goodput (%): The number of bits of the received data, excluding the number of data bits transmitted, by the simulation time.

The end-to-end delay (second): the ratio between the time for receipt of data minus the data transmission time and number of data packets received.

The routing traffic (normalized routing overhead): the ratio between the number of routed packets and the number of packets received.

The throughput (%): is given by the received data ratio taking into account all data sent

5.2 The effect of the mobility

Initially and as our interest focuses on MANET, it is clear that the evaluation function of the mobility becomes pretty obvious. The mobility was simulated by varying the maximum speed. The scenarios considered in this section consist of 50 nodes moving with a speed

between 0 and 40 m/s according to the Random Waypoint mobility model with a pause of 5 secs. Ten TCP connections were established between randomly chosen pairs. An average of twenty different mobility scenarios was performed to obtain each result.

5.2.1 The evaluation of TCP NewReno

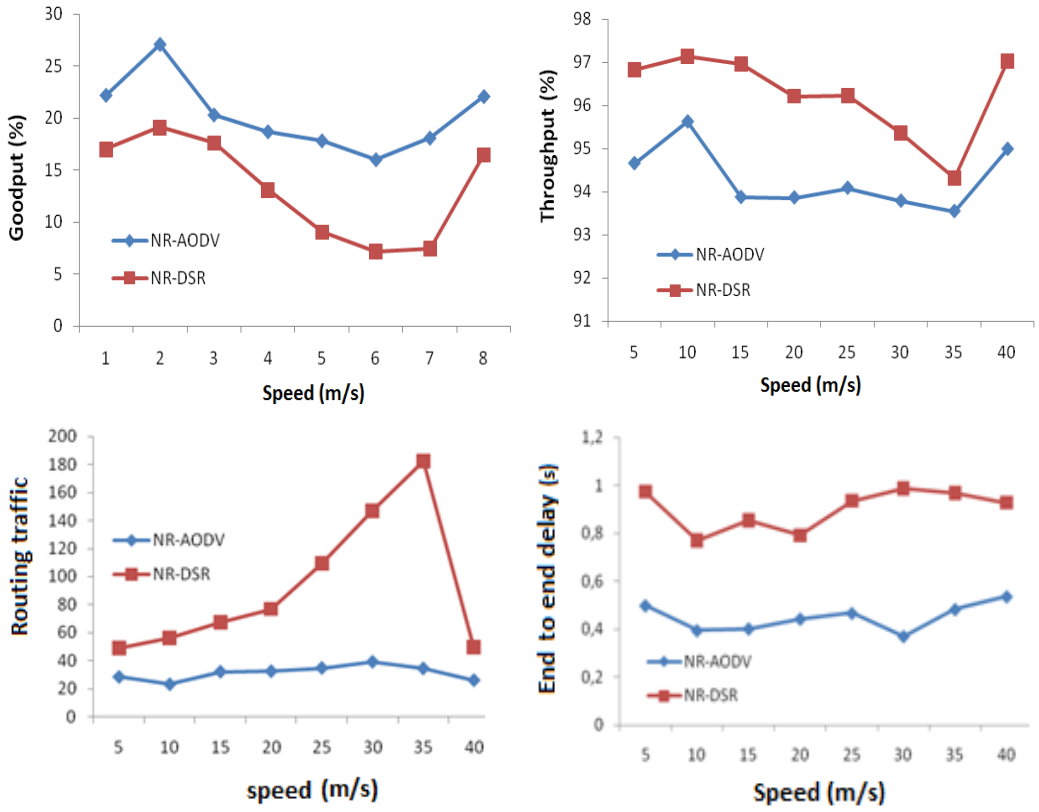

Fig.3. Effect of mobility on TCP New Reno with AODV and DSR

First, a throughput measurement of TCP New Reno with AODV and DSR is evaluated, and this as a function of node mobility. Regardless of speed, it is clear that DSR records better throughput for TCP New Reno compared to AODV. However, it saves a better goodput compared to DSR. It should be noted that for low speeds the goodput and throughput increase, but they also decrease soon as the speed increases and this with the two routing protocols. However, an increase is recorded again for both protocols interacting with New Reno for maximum speed exceeding 35 m/s.

The increasing of the throughput and goodput is due to the increasing of number of the received packets. The increase in speed, resulting in the break of the links and therefore the failures of transmission and packets losses. This involves the degradation of goodput. The problems of road failure in such environments lead to the launching of the congestion control system. This will reduce the congestion window and then the decreasing of the throughput.

It is clear that the restoration of roads due to disruptions caused by the mobility is heavy because both of the routing protocols used are reactive. However, DSR despite its mechanism, recorded a poor performance because of the invalid routes in the cache after changing the topology and then the need for alternative route or discover new route. Moreover DSR

registers a rather important routing traffic compared to AODV.

Regarding the performance of NewReno in terms of time, AODV enables it to record a lower delay compared to DSR. Certainly, DSR has the cache mechanism which should reduce the routing overhead at intermediate nodes, however, the available roads in these caches should be still valid. Mobility implies a change in the topology and generally broken links and then routes failure and the packets losses. However, a very interesting point must be raised, for certain speeds the delay may decrease and this means that this mobility can also be beneficial.

5.2.2 The evaluation of TCP Vegas

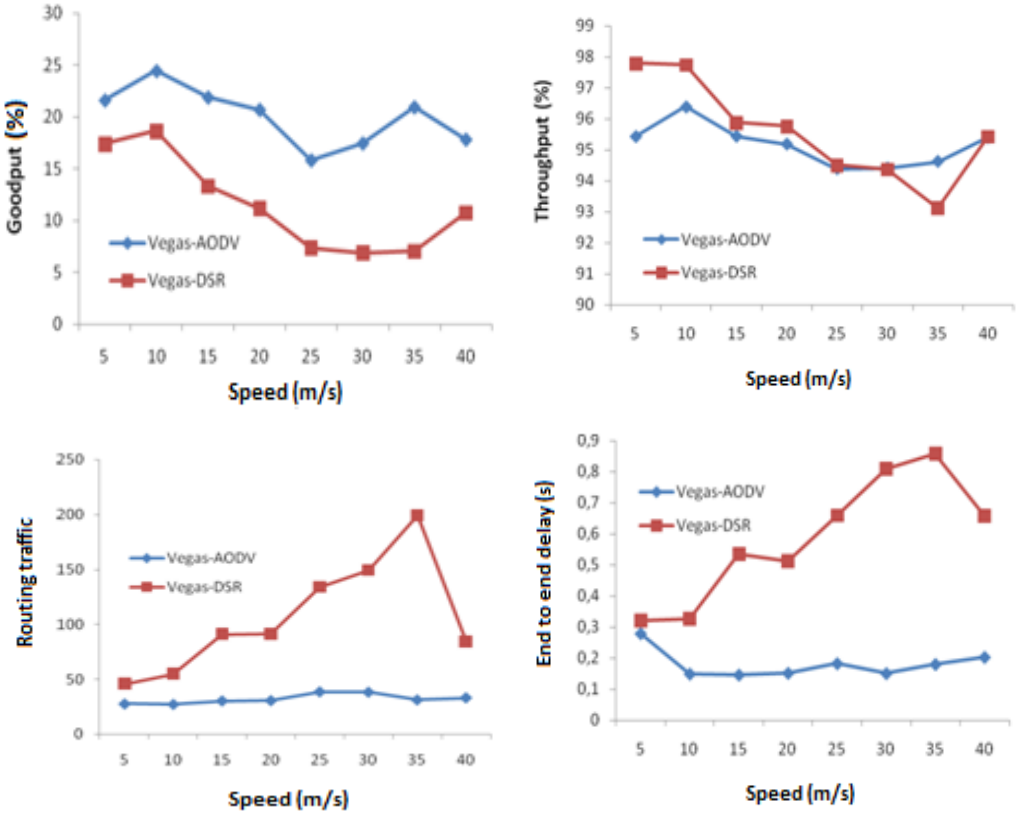

Fig.4. Effect of mobility on TCP Vegas with AODV and DSR

A measure of the throughput and goodput of TCP Vegas using two routing protocols results in an increase with both routing protocols for low speed but decreases with the increase of the speed of nodes.

For strong mobility, links break and nodes disconnect, thing which leads to packet loss and then the triggering of the congestion control system which reduces the congestion window. However, the difference observed between the two routing protocols is mainly due to the principle of restoration of roads used by everyone and which allows a better throughput for Vegas with DSR and a better goodput for AODV. The better goodput is due to the good reception therefore a better delivery and this is because the restoration mechanism which is more effective in such environment. However, the throughput of DSR decreases to approach even be lower compared to that of AODV. To increase again for an increase in goodput but

still always lower than AODV. With the DSR protocol, good performance is seen quickly deteriorated due to mobility.

The end to end delay of Vegas with DSR for low speed is very close to AODV and sometimes can be better, a result which has been observed in the simulations. This is due to the rapid restoration of roads from the caches of intermediate nodes. The average end to end delay for Vegas recorded with AODV remains much lower compared to that recorded with DSR with increasing speed. With the remark that the end to end delay with DSR increases rapidly as a function of mobility. For an increasing mobility, AODV proves better results compared to DSR which is penalized by its mechanism which should, at the base, improve its performance.

The routing traffic of DSR is always higher compared to that of AODV and whatever the TCP variant with which it interacts.

5.3 The effect of the network load (number of nodes)

In a second step an evaluation in function of the size of the network is performed. And for that the scenarios considered in this section consist of nodes moving with a speed of 15 m/s according to the Random Waypoint mobility model with a pause of 5 secs. 10 TCP connections were established between randomly chosen pairs. An average of 20 different scenarios was conducted for each outcome. The number of nodes being varied between 25 and 100.

5.3.1 The evaluation of TCP NewReno

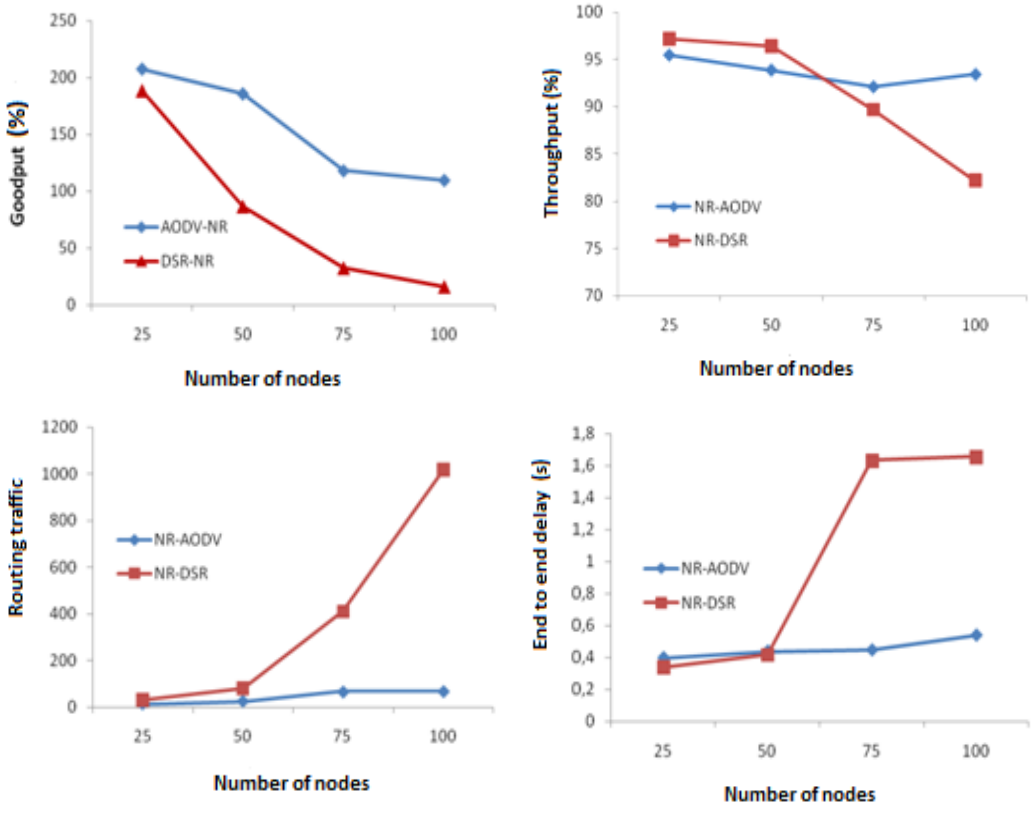

Fig.5. Effect of the network load on TCP NewReno with AODV and DSR

In large networks, deterioration in the overall throughput is noticed. With a remarkable performance degradation with DSR for large networks. A large number of nodes and a source routing protocol are ultimately not recommended too. The routing traffic increases considerably and thereby degrade the performance due to other contention caused.

In small networks and in case of error, AODV which has no caching mechanism must find a route, while DSR gets directly a new road. However, it is likely that this period is higher than that obtained by AODV and this because of the routing overload generated by DSR which adds a path field in the packet routing. The best performance obtained with DSR is due to the caching mechanism which hides its poorer transmission routes.

In networks of sufficient size the transmission delay of a packet is significantly higher than with DSR than AODV, due to overload generated by the path field. In addition the probability that route be invalid in the cache becomes important, which is explaining why the delay with DSR is higher than AODV.

5.3.1 The evaluation of TCP Vegas

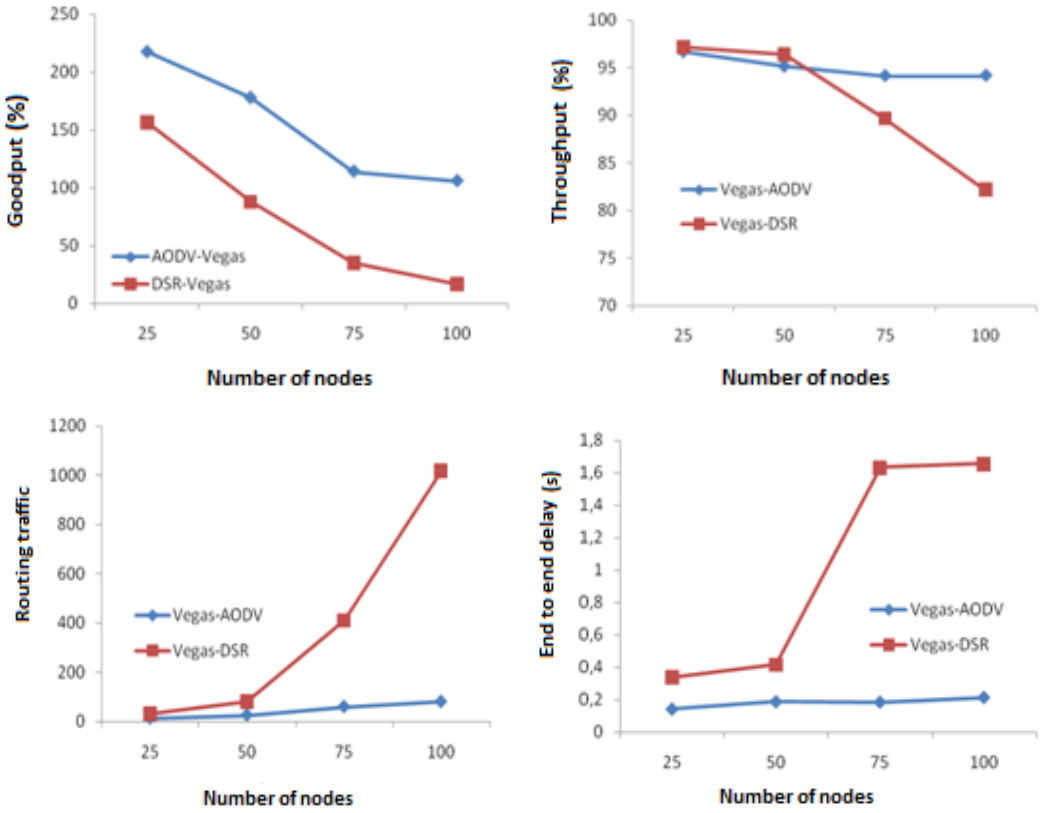

Fig.6. Effect of the network load on TCP Vegas with AODV and DSR

The results are essentially the same as those obtained with TCP New Reno with a higher average of end-to-end delay with DSR compared to that of AODV even for a small number of nodes. Vegas provide a better throughput with AODV compared to New Reno regardless of the number of nodes and the trend remains the same for the end-to-end delay.

5.4 The effect of the number of the connections

5.4.1 The evaluation of TCP NewReno

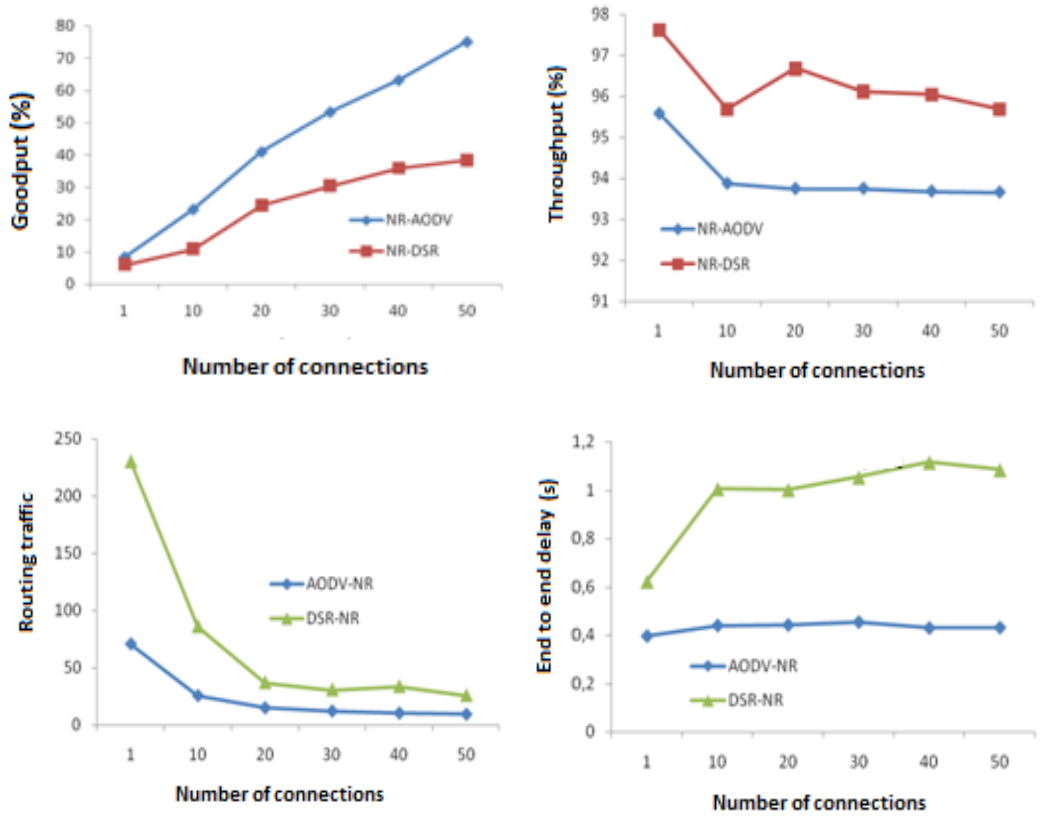

Fig.7. Effect of the number of connections on TCP NewReno with AODV and DSR

There is an increasing of the goodput with New Reno TCP for both routing protocols in function of the number of connections. Because the using of the routing table, AODV recorded a better performance depending on the number of connections. This leads to a reduction in losses and therefore less downtime and retransmission which reduces also the routing traffic.

The good performance of AODV makes that it records a better end-to-end delay which remains almost constant as a function of the number of connections. DSR records more delay which increases rapidly between one and ten connections which, thereafter, increases more slowly. The fact that the average delay with the AODV remains almost constant despite the increased number of connections is due to the delay between the source and destination which depends on the network conditions and the routing protocol. Although TCP NewReno recorded higher delay with DSR and this is because the characteristics of this routing protocol.

5.3.1 The evaluation of TCP Vegas

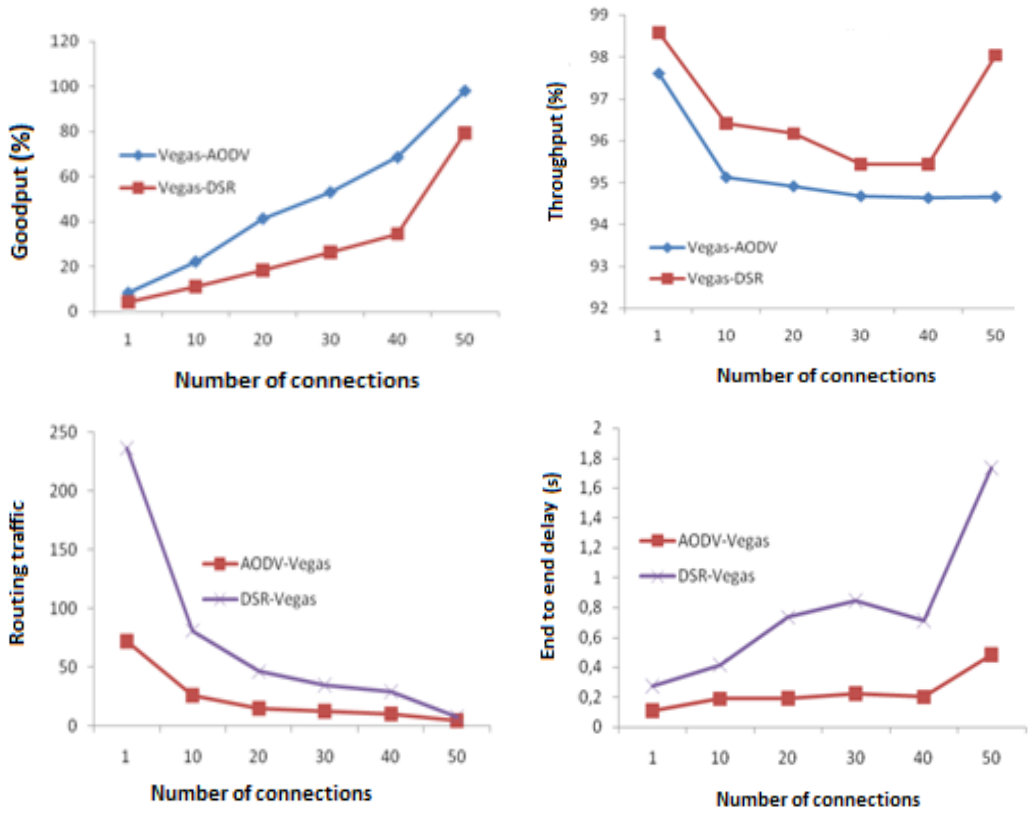

Fig.8. Effect of the number of connections on TCP Vegas with AODV and DSR

In general, the obtained results are very similar to those obtained with the transport protocol TCP New Reno. We should say that the routing aspect is quite dominant, the variations depend mainly on the behavior of the routing protocol.

5. Conclusion and future work

The behavior of TCP and its property of the flow control in a MANET environment were based on a vision of congestion in wired networks, which is quite different in ad hoc networks. In this paper, after a reminder of the main concepts and properties of TCP protocol, we then presented the sources of the performance degradation of such protocol in MANET and the recent efforts which have been made to improve the TCP performance TCP in context of MANET. All the solutions from these efforts are trying to improve TCP performance in one way or another, because the performance of TCP is very important to ensure a good quality of service for users of MANET.

We conducted also a study through which we showed that the performance of TCP is heavily influenced by the parameters of the communication environment such as network load, mobility of nodes and the number of TCP connections. This performance is also influenced by the protocols of the OSI model layers as the routing layer. TCP Vegas as proactive transport protocol shows good performance under different conditions. However, it provides much better results with the AODV routing protocol with DSR

We do not claim that the presented solutions in these papers are the optimal ones to better TCP performance, but the achieved results are indeed encouraging, justifying further investigation on this direction. The continuation of our work will consist in looking for a complete cross-layer solution in order to adapt dynamically and in a coordinated way the TCP protocol with the communication environment parameters for better quality of service in MANET.

References

[1] Basagni S., Conti M., Giordano S., Stojmenovic I., "Mobile Ad hoc Networking", Wiley-IEEE Press, ISBN: 0-471-37313-3, USA, 2004.

[2] Transmission Control Protocol, PROTOCOL SPECIFICATION, RFC 793, September 1981. http://www.ietf.org/rfc/rfc793.txt

[3] Hanbali A., Altman E., Nain P., "A Survey of TCP over Ad Hoc Networks", IEEE Communications Surveys & Tutorials, vol 7, no 3, pp 22-36, August 2005. http://dx.doi.org/10.1109/COMST.2005.1610548

[4] Kurose J., Ross K., "Computer Networking: A top-down approach featuring the Internet", Addison Wesley, 2005.

[5] Kawadia V., Kumar P., "Experimental investigations into TCP performance over wireless multihop networks", In SIGCOMM Workshop on experimental approaches to wireless network design and analysis (E-WIND), August 22, 2005. Philadelphia, PA, USA. http://dx.doi.org/10.1145/1080148.1080155

[6] STEVENS R., "TCP/IP illustrated", Vol. 1 The protocols, Addison –Wesley, 1994.

[7] Kuang T., Xiao F., Williamson C., "Diagnosing wireless TCP performance problems: A case study", In Proc. International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2003), July 20-24, 2003, Montreal, Canada.

[8] Fu Z., Zerfos P., Luo H., Lu S., Zhang L., Gerla M., "The impact of multihop wireless channel on TCP throughput and loss", Twenty-Second Annual Joint Conference of the IEEE Computer and Communications IEEE Societies (IEEE INFOCOM'03), San Francisco, USA, 30 March-3 April 2003. http://dx.doi.org/10.1109/INFCOM.2003.1209197

[9] Fu Z., Meng X., Lu S., "How bad TCP can perform in mobile ad-hoc networks", Seventh International Symposium on Computers and Communications (ISCC 2002), 1-4 July 2002, Taormina, Italy. http://dx.doi.org/10.1109/ISCC.2002.1021693

[10] Bakre A., Badrinath B., "I-tcp: Indirect tcp for mobile hosts", In Proc. of the 15th IEEE Int. Conf. on Distributed Computing Systems (IEEE ICDCS'95). Vancouver, Canada, Vancouver, British Columbia, Canada. May 30 - June 2, 1995. pages 136–143. http://dx.doi.org/10.1109/ICDCS.1995.500012

[11] Balakrishnan H., Seshan S., Amir E., Katz R.. "Improving tcp/ip performance over wireless networks", Proceedings of the 1st annual international conference on Mobile computing and networking (MobiCom '95), Vancouver, Canada, November 1995. http://dx.doi.org/10.1145/215530.215544

[12] Brown K., Singh S., "M-tcp: Tcp for mobile cellular networks", ACM Computer Communications Review, volume 27, Issue 5, pages 19-43. 1997.

http://dx.doi.org/10.1145/269790.269794
[13] Tsaoussidis V., Badr H., "Tcp-probing: Towards an error control schema with energy and throughput performance gains", 8th IEEE Conference on Network Protocols, Japan, November 14-17, 2000. http://dx.doi.org/10.1109/ICNP.2000.896288
[14] Zhang C., Tsaoussidis V., "Tcp-probing: Towards an error control schema with energy and throughput performance gains", 11th IEEE/ACM NOSSDAV 2001, New York, June 2001.
[15] Hamrioui S., Lorenz P., LIoret J., Lalam M., "A Cross Layer Solution for Better Interactions Between Routing and Transport Protocols in Manet", Journal of Computing and Information Technology, CIT 21(3), 2013, pp. 137-147.
[16] Hamrioui S., Lorenz P., LIoret J., Lalam M., "Incidence of the Improvement of the Interactions between MAC and Transport Protocols on MANET Performance", Book chapter, Wireless Communications and Networking-Theory and Practice, IGI Global, chapter 10, 2014, pp. 275-292.
[17] Chandran K., Ragbunathan, S.; Venkatesan, S.; Prakash, R., "A feedback based scheme for improving TCP performance in ad-hoc wireless networks," 18th International Conference on Proceeding International Conference on Distributed Computing Systems, Amsterdam, Netherlands, 26-29 May 1998. http://dx.doi.org/10.1109/ICDCS.1998.679778
[18] Holland G., Vaidya N., "Analysis of TCP performance over mobile ad hoc networks", ACM Mobile Communications Conf., Seattle, WA, USA, August 15–20, 1999, pp.219–230. http://dx.doi.org/10.1145/313451.313540
[19] Liu J., Singh S., "ATCP: TCP for mobile ad hoc networks", IEEE Journal on Selected Areas in Communications, vol. 19, no. 7, pp. 1300-1315, Jul. 2001. http://dx.doi.org/10.1109/49.932698
[20] Kim D., Toh C., Choi Y., "TCP-BuS: Improving TCP performance in wireless ad hoc networks", Journal of Communications and Networks, vol. 3, no. 2, pp. 175–186, Jun. 2001.
[21] Dyer T., Boppana R., "A comparison of TCP performance over three routing protocols for mobile ad hoc networks", in Proc. of ACM MOBIHOC, Long Beach, CA, USA, 2001, pp. 56–66. http://dx.doi.org/10.1145/501422.501425
[22] Wang F., Zhang Y., "Improving TCP Performance over Mobile Ad-Hoc Networks with Out-of-Order Detection and Response", 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc'02), Lausanne, Switzerland, Jun 2002. pp. 217-225. http://dx.doi.org/10.1145/513800.513827
[23] Hamrioui S., Lalam M., Arab D. K., Berqia, A., Lorenz P., "Improving tcp performance in manet by exploiting mac layer algorithms", IRACST - International Journal of Research in Management and Technology (IJRMT), vol 1, no 2, December 2011. Pp. 59-67.
[24] Hamrioui S., Lalam M., Lorenz P., "Ia-tcp : Improving acknowledgement mechanism of tcp for better performance in manet", International Journal on New Computer Architectures and Their Applications (IJNCAA), Vol. 2, Issue 2. Pp. 333-341. 2012.
[25] Zhai H., Chen X., Fang Y., "Improving Transport Layer Performance in Multihop Ad Hoc Networks by Exploiting MAC Layer Information", IEEE Transactions on Wireless Communications. Vol. 6, No. 5, 2007. Pp. 1692-1701.
[26] Lohier S., Doudane Y. G., Pujolle G., "MAC-layer Adaptation to Improve TCP Flow

Performance in 802.11 Wireless Networks", IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, (WiMob'2006). Montreal, Canada, 19-21 June 2006. http://dx.doi.org/10.1109/WIMOB.2006.1696392

[27] Kopparty S., Krishnamurthy S. V., Faloutsos M., Tripathi S. K., "Split-TCP for Ad Hoc Networks", UC Riverside Technical Report., 2002.

[28] Goff T., Abu-Ghazaleh N., Phatak D., Kahvecioglu R., "Preemptive routing in ad hoc networks", in Proc. of ACM MOBICOM, Rome, Italy, 2001, pp. 43–52

[29] Klemm F.,, Ye, Z., Krishnamurthy, S, V., Tripathi S. K., "Improving TCP performance in ad hoc networks using signal strength based link management", Ad Hoc Networks, Volume 3, Issue 2, March, 2005, Pages 175-191. http://dx.doi.org/10.1016/j.adhoc.2004.07.005

[30] Marina M., Das S., "On-demand multipath distance vector routing in ad hoc networks", IEEE International Conference on Network Protocols (ICNP'01), Riverside, California, 11-14 Nov. 2001. http://dx.doi.org/10.1109/ICNP.2001.992756.

[31] Oliveira R., Braun T., "A dynamic adaptive acknowledgment strategy for tcp over multihop networks", University of Bern, Technical Report IAM-04-005, July 2004.

[32] Cordeiro C., Das S., Agrawal, D., "COPAS: Dynamic contention-balancing to enhance the performance of tcp over multi-hop wireless networks", IEEE International Conference on Computer Communications and Networks (IC3N 2003), Miami, USA, October 2003, pp. 382–387.

[33] Floyd, S., Jacobson V., "Random Early Detection for Congestion Avoidance," IEEE/ACM Transactions on Networking. vol. 1, Pp. 397-413, August 1993.

[34] Sherif M. E., Alexander K., Christoph L., "TCP with Adaptive Pacing for Multihop Wireless Networks," MobiHoc'05, May 25–27, 2005, Urbana-Champaign, Illinois, USA. http://dx.doi.org/10.1145/1062689.1062726

[35] Al Hanbali A., Altman E., Nain P., "A Survey of TCP over Mobile Ad Hoc Networks," rapport technique N^{o} 5182 INRIA, May 2004.

[36] Luqing Y., Winston K.G. S., Qinghe Y., "Improving Fairness among TCP Flows crossing Wireless Ad Hoc and Wired Networks," MobiHoc'03, June 1-3, 2003, Annapolis, Maryland, USA. 2003. http://dx.doi.org/10.1145/778415.778423

Copyright Disclaimer