Computational Energy Cost of TCP in MANETs

Alaa SEDDIK-GHALEB
Networks & Multimedia Systems Research Group (LRSM), ENSIIE.
1 square de la résistance, 91025 Evry, CEDEX - France.
seddik@ensiie.fr

Yacine GHAMRI-DOUDANE
Networks & Multimedia Systems Research Group (LRSM), ENSIIE.
1 square de la résistance, 91025 Evry, CEDEX - France.
ghamri @ensiie.fr

Sidi-Mohammed SENOUCI
France Telecom R&D. 2 Av. Pierre Marzin, 22307, Lannion, France
sidimohammed.senouci@orange-ftgroup.com

***Abstract*— In this paper, we present the results from a detailed energy measurement study of different TCP variants when used in Mobile Ad hoc Network environments. More precisely, we focus on the node-level cost of the TCP protocol; also know as the computational energy cost. In fact, the computational energy consumption is the most important part of TCP energy consumption. This is already proven in previous work and our results confirm this fact. Sometimes, the computational energy cost is three times that of the communication energy cost. The studied TCP variants, in this work, are TCP New-Reno, Vegas, SACK, and Westwood. In our analysis, we draw a breakdown of the energy cost of the main congestion control algorithm (i.e. slow start, fast retransmit/fast recovery, and congestion avoidance) used by these TCP variants. The computational energy cost is studied using a hybrid approach, simulation/emulation, using the SEDLANE emulation tool. This study takes into consideration different data packet loss models (congestion, link loss, wireless signal loss, interference) within such environments when different ad-hoc routing protocols (reactive and proactive) are used. The performed study gives a set of results that are of high interest for future improvements of TCP in MANETs. Among the obtained results, we show that the computational energy cost of TCP varies according to the type of data packet loss model it comes through: network congestion, interference, link loss, or signal loss. The results demonstrate that the link loss scenario is the most severe situation for TCP connections to face. In addition to that, we show that the Fast Retransmit/Fast Recovery phase has much less energy cost than both Slow Start and Congestion Avoidance phases, due to the fact that it sends more TCP data bytes in a shorter period of time. Finally, the computational energy cost is quantified and compared to the TCP end to end performance for each TCP variant showing the link between both.**

I. INTRODUCTION

TCP is the most popular reliable transport control protocol. It is today supported by almost all Internet applications. However, TCP does not always have the best performances. In order to identify its performance limitations and thus to be able to correct them, it is important to study its behavior, categorize its performance metrics and quantify them in each of the different environments where TCP can be used. In this paper, we analyze a particular, but still important, performance metric and one importance-increasing environment in which TCP is targeted to be used. More precisely, we are interested in studying the computational energy cost of TCP when this one is used in Mobile Ad-hoc Networks (MANETs). The major motivation behind this study resides in the fact that mobile devices are battery-operated and it is important to optimize the energy-consumed by such devices in order to increase their lifetime. Prior to any improvement, there is a need to better understand how and where energy is consumed in the communications pipeline.

The computational energy cost of TCP is the energy spent within the node and its CPU unit in order to realize the various copy operations, compute checksums, and to respond to timeouts and triple duplicate ACKs, adjust timers, and perform the other book keeping operations. This cost is thus linked to the execution of the different TCP congestion control algorithms (Slow-Start, Fast Retransmit/Fast Recovery, and Congestion Avoidance). One should finally note that this work is complementary to the different researches targeting the evaluation of the radio-related energy cost of TCP variants (i.e. the energy consumption due to the transmission, retransmission and forwarding of TCP segments by ad-hoc nodes) [1][2][3].

In this work, the four major TCP variants, namely TCP New-Reno, Vegas, SACK and Westwood, are considered. In order to measure the computational energy cost while executing their different congestion control algorithms, we implement different data packet loss models and we take into consideration different types of ad-hoc routing protocols. Measuring the node-level energy consumption is realized using a realistic test-bed configuration. This configuration should introduce the effect of a real wireless mobile ad-hoc network environment (i.e. realistic data packet delays and losses). In this paper, we introduce such effects using a MANET delay and packet-loss emulation tool called SEDLANE [4] (Simple Emulation of Delays and Losses for Ad-hoc Networks Environment). This tool uses NS-2 simulation results in order to generate realistic data packet delays and losses in MANETs. The use of such a hybrid approach makes the evaluation approach taking advantage of each of these approaches: simulation and test-bed experiments. Hence, thanks to SEDLANE, the effect of different data packet loss models (congestion, interference,

link loss, and signal loss) and ad-hoc routing protocols (reactive vs. proactive) are introduced. Our study can have multiple benefits. The main foreseen benefits that motivated our work are: (1) to enable the understanding of the energy consumed by TCP at the CPU level and thus to facilitate the future development of new TCP congestion control algorithms for MANETs that are energy-efficient; and (2) to give a methodology, that extends the one defined by [5], to others that wants to evaluate the TCP energy cost in MANET-related specific scenario of use. The other benefits of such studies [6] are also: (3) to give to other researchers working on analytical modeling of TCP a set of results to develop energy models for TCP congestion control algorithms; and (4) to allow the incorporation of our node-level energy models into network simulators (such as NS-2) in order to obtain the overall energy cost (computational + radio) of TCP connections (Currently, network simulators only includes the radio energy cost).

The remainder of this paper is organized as follows: after presenting the related work in Section II, Section III gives an overview of SEDLANE emulation tool. It is followed by the description of the implemented test-bed and the methodology used to measure the node-level energy consumption. Section V introduces the results of our work. Finally, we summarize the main results and give some ideas for improving TCP performances in mobile ad-hoc networks in Section VI.

II. RELATED WORK

One of the earliest work analyzing the processing overhead of TCP was presented in [6]. The goal of this work was to analyse the effect of TCP processing on the end-to-end connection throughput. The study implied both the sender and receiver sides. The breakdown of the processing cost in both the sender and receiver sides gave insightful results and led to the development of a variety of techniques to enhance TCP efficiency [5]. Most of these improvements were operating system or hardware related. Many years later and with the advance of mobile computing devices, few researches started to look at the evaluation of the node-level energy consumption by wireless devices. Among these, some studies used realistic test-beds [5] [7] in order to get the energy consumption of TCP. More precisely, the authors in [7] have looked at the energy consumption of various wireless interface cards used by ad-hoc nodes; while [5] concentrated on the evaluation of the operating system and hardware level operations needed by TCP. Suggestions on how to improve the interactions between TCP and the underlying device platform had then been drawn. In our current study, we go a step further as we are more interested in TCP congestion control algorithms as a whole. Indeed, our target is to analyse their computational energy cost in order to identify potential improvements for TCP when it is used by mobile devices in general and by MANETs in particular. The advantage of using a test-bed configuration is to get the energy consumed at the node level (i.e. within the CPU unit) which is not straightforward to obtain when using simulations. In the mentioned studies [5][7], the authors implemented a test-bed configuration in order to compute the TCP energy consumption within wireless environments. Both studies used Dummynet [8] to introduce both data packet delays and losses in order to emulate the effect of a wireless environment. Dummynet parameters were randomly selected. This is not realistic as data packet losses are not random neither in wireless network in general nor in MANETs in particular. Furthermore, the delays are correlated and dependant on the effect of the wireless environment and the protocol suite. In our test-bed implementation, instead of using random data packet delay and loss values, we exploit the realism of a new wireless ad-hoc network emulator that is called SEDLANE [4]. SEDLANE allows us to introduce different data packet losses and delay effects that can appear within ad-hoc networks environment (congestion, interference, link loss, and signal loss).

III. OVERVIEW OF SEDLANE

As depicted in Figure 1, the main idea of SEDLANE [4] is a hybrid evaluation approach that takes benefit from simulation results in order to enhance real test-bed experiments. It allows configuring Dummynet [8] pipes (i.e. defining packet loss and delay rules) through NS-2 (Network Simulator-2) [9] trace files. More precisely, SEDLANE uses NS-2 TCP trace file to identify the classes of packets by gathering together the packets that have similar RTT values. Then, SEDLANE dedicates one pipe or communication channel for each group of packets. Hence, according to the identified packet classes, delay values (i.e. RTT/2 on each way) and loss rates are distributed among classes, SEDLANE dynamically generates the Dummynet rules to be applied on the packets. This way, we control the different ad-hoc network parameters using simulation approach in order to make our experiments more realistic compared to those previously used (i.e. in terms of data packet delays and losses).

Figure 1 SEDLANE Operation Concept

IV. TCP COMPUTATIONAL ENERGY CONSUMPTION

A. *Test-bed Configuration*

The methodology used in our energy consumption measurements test-bed extends the one previously used in [5] in which we add the use of SEDLANE.

Our test-bed configuration (as shown in Figure 2), is composed of a DELL LATITUDE D410 laptop as a sender point while the receiver end side is a DELL OPTIPLEX GX 520 Personal Computer (PC). Between the communicating nodes we implement SEDLANE (on a second DELL OPTIPLEX GX 520 PC), to get the effect of a wireless ad-hoc network environment between the sender and receiver sides. The laptop communicates with the PC over a wireless link channel. In order to calculate TCP energy consumption within the CPU unit: we measure both (i) the total energy consumption within the laptop, and (ii) the energy consumed within the wireless card for transmission and reception. The difference between the two measured values will be the computational energy consumption. Obviously, the measurements are taken at the TCP sender side. Synchronization ensured between the communicating end points and the PC where the measurements were taken. In

order to match this computational energy consumption to the TCP operations, we use a minimal Linux distribution in which we turn off the display, the power manager and the x-server in order to minimize the effect of any other running applications on the measured current. The reason for turning off power management as described in [5] is the fact that it helps to better determine the current draw that corresponds to TCP code execution. Last but not least, all the processes/daemons that are not necessary to TCP operations are simply removed from the Linux distribution making it minimal. By taking all these precautions, we ensured that the remaining energy consumption is due to TCP congestion control algorithms execution and timer adjustments.

Energy consumption is determined by measuring the input voltage and current draw using two Agilent 34401A digital multi-meters that have a resolution of one millisecond. We do not use the laptop's battery because avoiding the use of battery allows for a more steady voltage to be supplied to the device [10]. In order to directly measure the current and voltage draw of the wireless 802.11b PCMCIA card, the card was attached to a Sycard PCCextend 140A CardBus Extender [11] that in turn attaches to the PCMCIA slot in the laptop. This way, we can separately but simultaneously measure the current draw of the laptop and the current draw of the wireless 802.11b PCMCIA card[1].

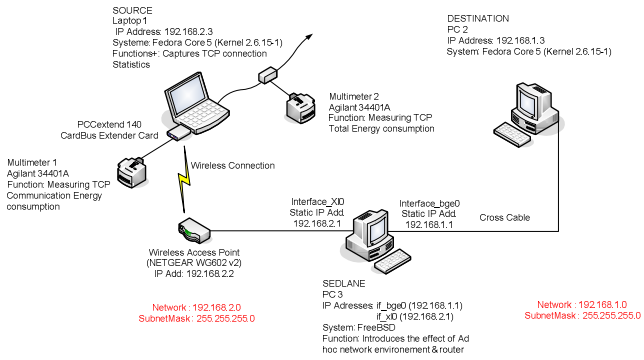

Figure 2 TCP Computational Energy Cost Measurements Test-bed

B. *Measurements Scenarios*

In order to have a wide range of results that help better understanding the behavior of TCP in front of different data packet loss models, we run our measurements using different loss scenarios. In the mean time, NS-2 simulation traces are obtained using different ad-hoc routing protocols (AODV, DSR, DSDV, OLSR), thus to get the effect of such routing protocols on TCP performances and in turn its computational energy cost. The models are defined to be run using NS-2. Then the TCP trace files are used by SEDLANE in order to provide the loss and delay effects (as described in Section III) within our realistic test-bed implementation. Our study is categorized by the nature of data packet loss models: (i) network congestion, (ii) interference, (iii) link loss, and (iv) signal loss.

[1] Sycard PCCextend 140 CardBus extender card is a debug tool for development and test of PC cards and hosts.

1) Creating Network Congestions: In this packet-loss model, we create a congested node at the middle of a five-node topology. This is done by generating three TCP data traffic flows that must pass by this intermediate node to reach the other communicating end (Figure 3). One should also note that, different levels of data congestion can be generated by controlling the number of TCP data flows crossing this particular network node at a certain time.

2) Interference Between Neighboring Nodes: In this case, two TCP connections are on-going in parallel. The main TCP connection (TCP data flow 1 in Figure 4) is disturbed by the interferences generated by the second TCP connection (TCP data flow 2 in Figure 4). Indeed, the node acting as forwarder for the main TCP connection is placed within the interference range of the second TCP connection sender. So, this situation creates interference and thus data packet drop.

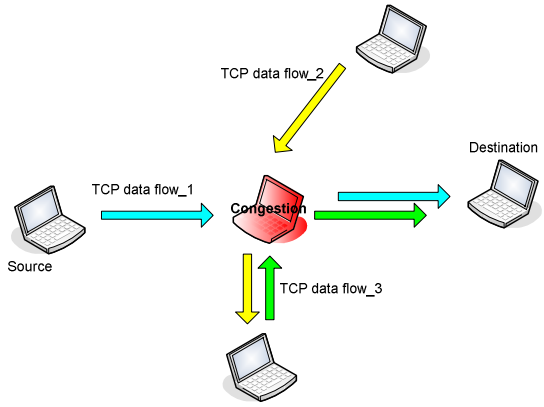

Figure 3 Creating network congestions

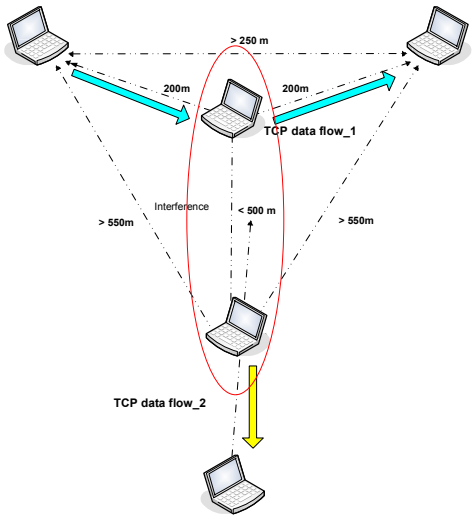

Figure 4 Interference Model

3) Link Loss and Communication Route Changes: In this model we force TCP to change its communication path by shutting down the intermediate node between the communicating end points. In addition, we imply routes with different number of hops (Figure 5). Thus, each time TCP

changes the communication route, the characteristics of the path between the communicating nodes change. It is obvious that the choice and the establishment delay of the new route will be dependant on the implemented ad-hoc routing protocol. Packet losses and delay changes will also be implied by the link loss and the new chosen route.

4) Signal Loss: this scenario illustrates the situation where the wireless signal is not stable. The communicating nodes loose the connection due to signal loss then they resume the communication when the signal comes back. Signal losses are generated by moving one of the intermediate nodes out of the radio range of its connection neighbor (Figure 6).

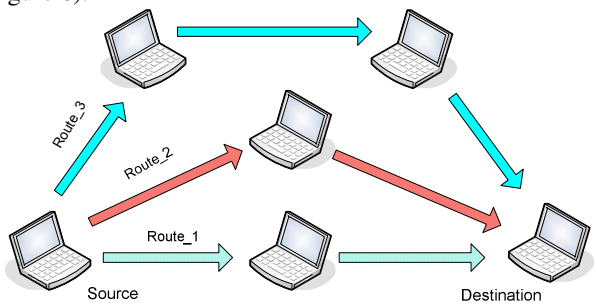

Figure 5 Link Failure Model

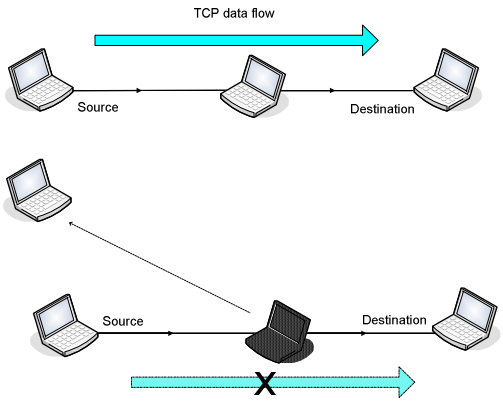

Figure 6 Signal loss Model

V. TEST-BED RESULTS

Contrarily to previous studies that concentrated on the operating system, hardware and device-level energy consumption due to TCP, the objective of our analysis is to analyse the energy cost of each TCP function and variant in order to facilitate improving their behaviour in MANETs. So, in the following we first analyse the computational energy cost of the main TCP functions: slow-start, congestion avoidance and fast recovery/fast retransmit. Then, a comparison of the different TCP variants in terms of computational energy cost is made. This one is realized according to the different data packet loss model: network congestion, interference, link loss, or signal loss. For each TCP variant, the computational energy cost is quantified and compared to the end to end performances. Finally, we identify and briefly discuss a set of design features that must have a new TCP variant to be energy and resource-efficient while used in MANETs.

A. *TCP Functions Computational Energy Cost*

In this section we analyze TCP New-Reno energy consumption using AODV as an ad-hoc routing protocol within the simulations. In order to get the energy consumption of the main TCP functions (Slow-start, Fast Retransmit/Fast Recovery, and Congestion Avoidance), we use log files at the sender side to log the start and end times of each TCP function. Then, we use this information to match the energy consumption with each process by using the energy consumption measurement record.

The results show that the computational energy cost of the Fast Retransmit/Fast Recovery phase is extremely high compared to that of both the Slow Start and Congestion Avoidance phases (Figure 7). Indeed, Figure 7 shows that the energy consumption is almost doubled. However, this is mitigated when we compare the energy consumption according to the amount of data sent by TCP (Figure 8). This is due to the fact that the TCP Fast Retransmit/Fast Recovery process consumes an important amount of energy when triggered but it does so for a short period of time during which it may send several TCP segments on one burst. This leads to a continuously high computational overhead while in the slow-start and congestion phases the computational overhead is not continuous (back-off). Indeed, the Fast Retransmit/Fast Recovery phase resumes the data transmission after data packet loss without minimizing data transmission rate to minimum which is the case in Slow Start phase. So, the trade-of energy-cost/data sent remains low in the Slow-Start phase. For its part, the Congestion Avoidance process has an acceptable trade-of between energy-cost and data sent (Figure 9). Indeed, during this phase, TCP is assumed to be close to its optimal throughput value. During this phase, TCP increases its transmission rate by one segment each RTT. It has a regular throughput and computational overhead that are lower than those of Fast Retransmit/fast Recovery phase (Figures 7 and 9). This leads to higher energy consumption per sent byte in this phase compared to Fast Retransmit/Fast Recovery phase (Figure 9).

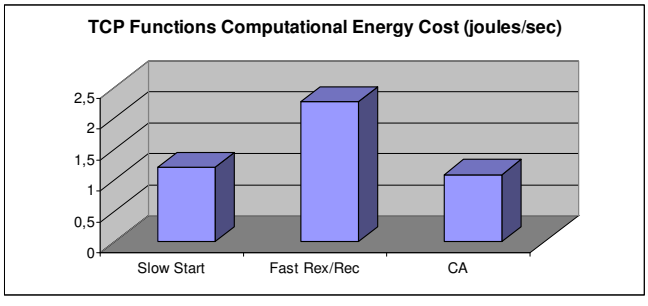

Figure 7 TCP Computational Energy Cost (joule/sec)

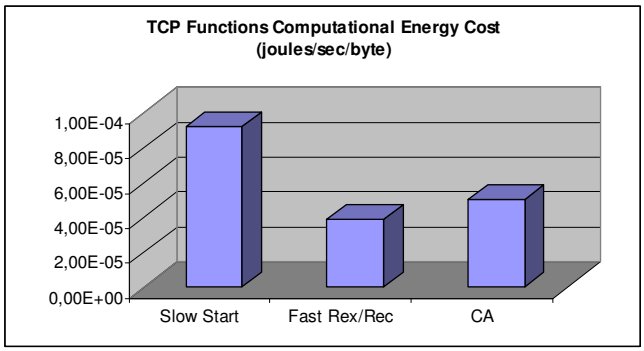

Figure 8 TCP Energy Cost (joule/sec/sent byte)

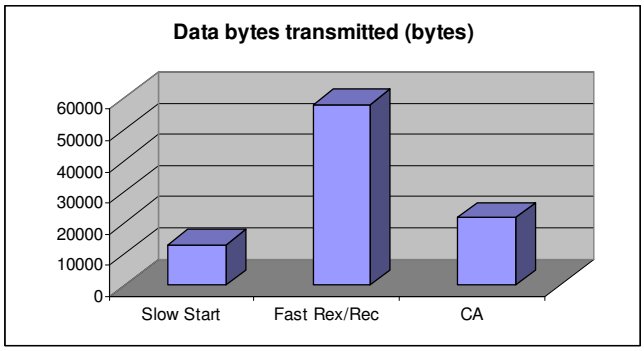

Figure 9 Data bytes transmitted/TCP function

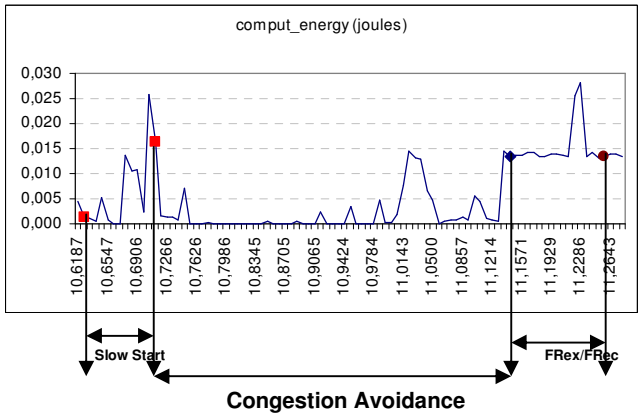

Figure 10 TCP New-Reno computational energy cost example (network congestion case)

Figure 10 shows an example of TCP New-Reno computational energy cost while facing packet losses due to network congestions. It can be obviously seen from the Figure that, the computational energy cost of the Fast Retransmit/Fast Recovery phase is higher than that of both Slow Start and Congestion Avoidance for the reasons mentioned above.

B. Computational Energy Cost of TCP Variants

This section aims at comparing the main TCP variants (New-Reno, SACK, Vegas, Westwood) in front of the different packet loss models they can face in ad-hoc networks: network congestion, interferences, link loss, and signal loss. As the two later may implies the triggering of the ad-hoc routing protocol to re-establish the network connection, the comparative study will be made according to different ad-hoc routing protocols. Contrarily, the two former are agnostic from the ad-hoc routing protocol.

1) Analyzing the Effect of Network Congestion and Interferences:

In order to isolate the effect of network congestion and interferences from other packet loss reasons, we used a static ad-hoc network without route changes. In this section, we choose to use the Ad-hoc On-demand Distance Vector (AODV) [12] as ad-hoc routing protocol. AODV triggers a route discovery only when the sender needs to send data to the destination. The ad-hoc routing protocol choice in a static ad-hoc network has no impact on the performances of the ongoing TCP connections.

a) Effects of Network Congestion:

The results demonstrate that TCP Vegas has the least number of TCP segments lost (almost no loss) among all the other variants (Figure 11). This is due to the fact that TCP Vegas is a variant that tries to avoid congestions. In order to achieve this, TCP Vegas calculates and modifies its TCP transmission parameters with each received acknowledgement (ACK). However, this reliability costs a lot in terms of processing which in turn can be translated into a high computational energy cost compared to all the other studied variants (Figure 12). On the other hand, we notice that TCP Westwood has better performance in terms of computational energy cost because it modifies its transmission parameters only when there is a data packet loss over the connection and not continuously as in TCP Vegas. This implies less computational overhead in spite of the increased number of retransmission compared to TCP Vegas. We also remark that TCP Westwood and New Reno have almost the same performances in terms of energy consumption per sent byte (Figure 12) even if the loss ratio is higher with TCP New Reno (Figure 11). From that we can conclude that the light computational cost (i.e. the one due to Fast Recovery/Fast retransmit process) of resending packets by TCP New Reno is neutralized by the computational overhead introduced by TCP Westwood (i.e. loss analysis to identify the packet loss cause).

Finally, one should note that even that TCP SACK has the ability to resend the lost data packets faster than TCP New Reno due to the Selective ACK option, Figure 12 demonstrates that the cost of SACK processing and storage (to extract the numbers of lost data packets at the sender side) is high in most cases, especially when the number of lost data packets is important as in network congestion cases.

b) Effect of Traffic Interference:

The results show that the number of data bytes lost due to traffic interference is higher than that due to network congestion (Figures 11 and 13). This result can be explained by the fact that TCP uses congestion control algorithms, meaning that TCP has the ability to better deal with network congestion conditions than traffic interference ones. The misbehavior of TCP in front of data packet losses due to interference leads to more computational energy cost. This result can be verified by comparing both Figures 12 and 14.

Referring to Figure 14 we recognize that TCP Vegas has the worst performance in terms of computational energy cost compared to other studied TCP variants. TCP Vegas depends on measured RTT values to adjust its performance parameters. In the case of traffic interference, measured RTT values do not change significantly, and then TCP Vegas does not recognize that there is a problem over the communication path and keeps increasing its data transmission rate normally which leads to more traffic interference over the connection. More traffic interference leads to more TCP data packet losses and higher computational energy cost. This important number of packet losses and retransmissions leads to more computations which has a high impact in the case of TCP Vegas. Even that TCP Westwood has the lowest loss ratio compared to other TCP variants; its computational energy consumption is higher than that of both TCP New Reno and TCP SACK. This is due to the complexity of the algorithms used by TCP Westwood and their continuous triggering by packet losses (i.e. re-calculates and modifies its data transmission rate after each data packet loss). Note that in front of congestions, TCP Westwood has the same behavior as TCP New Reno while in front of losses due to wireless effects its behavior is more complex. Surprisingly, we found that TCP New Reno and TCP SACK have almost the same performance in terms of computational energy cost. Although that the number of retransmitted data with TCP SACK is less than that in TCP New Reno, the processing overhead of TCP SACK neutralizes the advantages of using Selective acknowledgements.

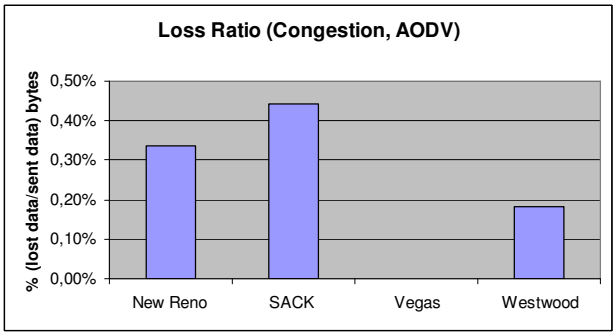

Figure 11 TCP Loss Ratio (network congestion model)

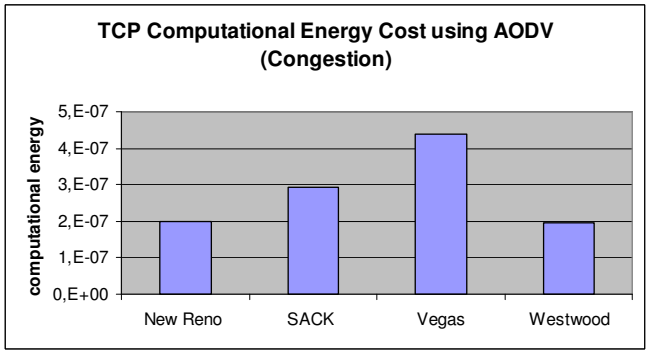

Figure 12 TCP Energy Cost (joule/sec/sent byte)

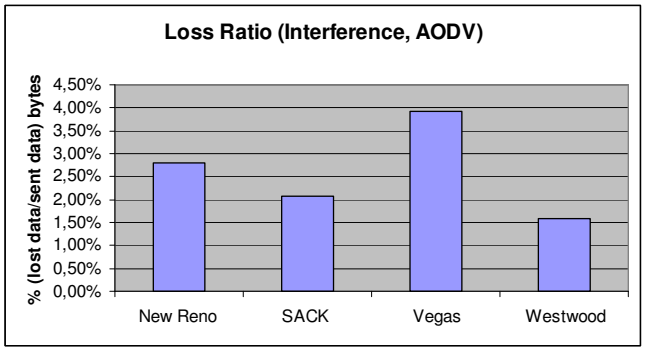

Figure 13 TCP Loss Ratio (interference model)

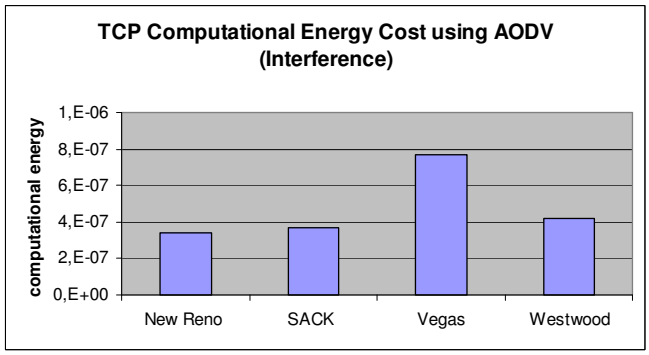

Figure 14 TCP Energy Cost (joule/sec/sent byte)

2) Analyzing the Effect of Link and Signal Losses:

In mobile wireless ad-hoc network, it is obvious that the nodes might have broken communication paths between the communicating end points (due to mobility or depletion of nodes' batteries). In addition, loosing the radio signal for short periods might be considered as another reason to get disconnected temporarily from the other communicating end. Signal loss case could be due to geographical obstacles such as high buildings or weather conditions such as raining. The above situations result in data packet losses over the TCP connections.

The choice of the ad-hoc routing protocol algorithm is important from two points of view: (i) its robustness to recover from a link failure, (ii) the overhead and frequency of its routing information update messages which might result in a congestion or traffic interference over the network links. For example, the overhead of ad-hoc routing update messages could aggravate the congestion situation over the TCP connection. This leads to more congestion control actions taken to recover from the packet losses.

TABLE I Comparative study of ad-hoc routing protocols

Ad-hoc routing protocol	Start-up time (sec)	Route recovery (sec)
AODV	≈ 0.03	≈ 1
DSDV	≈ 90	≈ 31
DSR	≈ 0.07	≈ 0.2
OLSR	≈ 6	≈ 7

As our objective is to analyze the effect of link loss and signal loss on the TCP variants performances while in MANETs using different ad-hoc routing protocols, let us first

recall the main performances of these routing protocols. Table I discusses the start-up time, i.e. the time needed by the ad-hoc routing protocol to build up its routing information table in order to start communicating, and the route recovery time of each of the four main ad-hoc routing protocols: AODV [12], the Dynamic Source Routing (DSR) [13], the Destination-Sequenced Distance Vector (DSDV) [14], and the Optimizes Link State Routing (OLSR) [15]. The figures depicted in Table 1 allows us to recall that in reactive protocols (AODV, DSR), the routing protocol triggers its route discovery process only when there is data to send towards the destination or when a used route is broken. Contrarily, Proactive protocols (DSDV, OLSR) needs longer time to build their routing table and also to recover from a route loss. This is due to the fact that it makes it for the whole network prior to any communication request is triggered.

c) Effect of Link Loss:

Figure 15 shows that the computational energy cost of most TCP variants increases compared to the above two studied scenarios. This is an expected observation because TCP as it is nowadays was not designed to cope with network link failures. In network link failure situations, we must expect high number of route re-computations. In this situation (link failure), the effect of the chosen ad-hoc routing protocol appears in its robustness and rapidity (promptness) to recover from link failures in order to resume the communication between the end points and avoid some TCP timeouts.

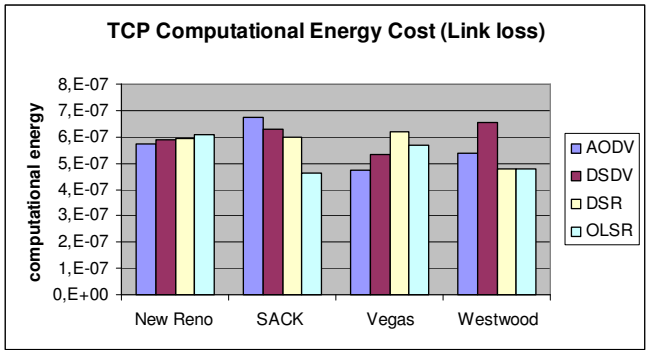

Figure 15 TCP Energy Cost (joule/sec/sent byte)

In the case of Link Loss, we remarked that all TCP variants in almost all cases recognize the data packet loss through TCP Retransmission Timeout (RTO). As they are not designed to cope with such situations (link losses), they all react similarly: i.e. classify the packet loss as due to strong network congestion and triggers the Slow-Start process. As mentioned previously (Section V.A), the slow start process is the less efficient one in terms of energy cost. Let us note here that theoretically triggering the slow-start phase is not necessary as the packet loss cause is not strong congestion. If we look to each variant separately, both TCP Vegas and TCP Westwood can be considered as well performing variants, in most cases. This is because both variants have the ability to rapidly re-adjust the data transmission rate over the connection according to the performances of the new chosen route. TCP New Reno or TCP SACK are proved to be less rapid in that.

d) Effects of Signal Loss:

Signal loss can be considered as a special case of link failure. In fact, we consider here the special case where when the signal is lost between two communicating end points, there is no way to resume the communication session unless the signal returns back. Thus, signal loss might be viewed as a network partitioning case where the communicating end points are totally disconnected from each other. The main difference between link failure and signal loss models is the ability to resume the communication session after the signal loss using the same route (that had also to be re-established by the routing protocol). In the link loss case, both nodes (sender and receiver) would search for another route to complete the session. While in the signal loss case, this is topologically not possible. After signal loss recovery, TCP sender will start the communication session from the beginning, starting from Slow Start phase. And this will be the case, each time the communicating nodes get disconnected in the absence of wireless signal. That's why almost all TCP variants stay most of the connection lifetime in Slow Start phase. In addition, TCP data packet losses would be recognized through RTO expiration.

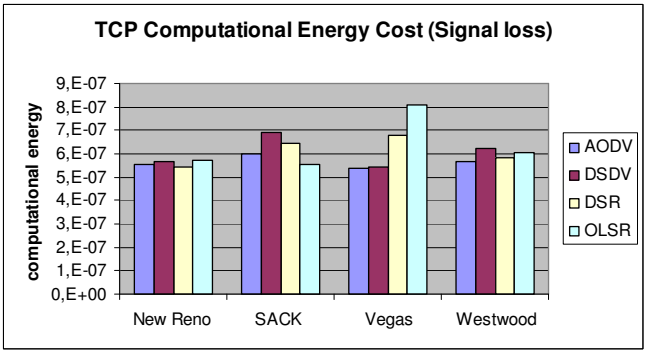

Figure 16 TCP Energy Cost (joule/sec/sent byte)

Figure 16 demonstrates that TCP Westwood is the best performing variant among all the other studied ones. TCP Westwood has the ability to differentiate between data packet losses due to congestion and those due to wireless signal problems. Thus, better adjusts its performance parameters. The trade-off between computational complexity and amount of data sent makes him more energy-efficient. This leads to the total computational energy cost of TCP Westwood is lower than other TCP studied variants. The correct classification of TCP Westwood of the cause of data packet losses is its main advantage over the other variants in this case. We have studied the signal loss model using different levels of signal loss duration time ranging from few seconds to few tens of seconds, and in all cases, TCP had the same behaviour.

C. Summary

In order to evaluate the performance of TCP within wireless networks, especially when studying mobile ad-hoc networks where the communication sessions could be interrupted due to nodes' mobility or even nodes' battery depletion, it is important to get a detailed idea about TCP

energy consumption. Our current study concentrated on evaluating the computational energy cost of the different TCP functions and the main TCP variants that implements them. While doing so we seen that there is a link between TCP end-2-end performances (i.e. achieved throughput), the complexity of the used algorithms and the computational energy-cost. Firstly, the results show that the Fast Retransmit/Fast Recovery phase has mush less energy cost than both Slow Start and Congestion Avoidance phases, due to its ability to send more TCP data bytes in a short period of time. Even though, the Congestion Avoidance phase can be considered as having a good balance in terms of computational energy cost which is not the case of the slow-start phase. Secondly, we showed that the performance of TCP varies according to the type of data packet loss model it comes through (network congestion, interference, link loss, or signal loss). The results demonstrate that the link loss scenario is the most severe situation to face for TCP. Link failure causes burst loss over the connection and TCP (as it is nowadays) mistakenly interprets and deals with it as it deals with strong congestions. This leads to the repeated triggering of the Slow-Start phase which has been proved as non-efficient in terms of computational energy cost. The simplicity of TCP Westwood and its ability to rapidly adjust its transmission parameter to match network conditions makes it one of the best performing TCP variant in terms of computational-energy cost. Finally, to sum up, the performance of TCP is highly affected by the loss model it comes through. Our result shows that, the reaction of TCP in most cases could not be the right one: for example dealing with data packet losses due to link loss as if it was a strong congestion is proved to be an erroneous reaction. From that, we suggest that TCP should have a data packet loss classification algorithm in order to classify the reason of data packet losses and accordingly triggering the most appropriate data loss recovery algorithm strategy. The loss differentiation algorithms should have the ability to recognize the different data packet loss causes within wireless mobile ad-hoc networks (network congestion, wireless channel errors, and link loss) with a minimum computational overhead (i.e. without storing and maintaining too much state information). Let us also remind that it had been shown in our results (§V.B.2.b) that the classification of signal losses as wireless channel errors by TCP Westwood had a good impact on the computational energy cost. So there is not a need to make explicitly the differentiation between both. The Loss Recovery should be as simple as possible (i.e. variation of the Fast Retransmit/Fast Recovery) avoiding un-necessary bandwidth reductions (i.e. reducing the bandwidth only in case of strong congestions).

VI. CONCLUSION

TCP was originally designed for wired networks. As a congestion control transport protocol, TCP can not cope with other data packet loss models that may be found within wireless ad-hoc networks (link failure, signal loss, and interference). Researches found that TCP performance highly degrades within such networks. In our work, we studied the performance of different TCP variants in terms of energy consumption at the node's level. The computational energy cost of TCP is the energy consumed in order to allow adjusting its parameters and execute its congestion control algorithms. We found that the complexity of these algorithms and their failure to cope adequately with certain loss causes are the main causes for unnecessary energy wastage at the node's level. We studied the TCP computational energy cost using a hybrid approach (i.e. using simulation results to configure a real test-bed and perform accurate experiments). The results show that TCP as it is suffers when dealing with different data loss models other than congestion. We also identified some tracks to follow in order to create a novel TCP variant that is energy-efficient in MANETs. Knowing where the most TCP energy consumption is spent is the main key to improve TCP functions and performance within MANETs. For example, helping TCP to avoid unnecessary re-transmissions or TCP CPU calculations would help minimize TCP energy consumption. In future work, we will develop this new TCP variant for MANET. This one should have an optimized behavior in regards of the different TCP performance parameters in such environments: throughput, radio-energy cost and the computational energy cost.

REFERENCES

[1] H. Singh and S. Singh, "Energy consumption of tcp reno, newreno, and sack in multi-hop wireless networks," *in ACM SIGMETRICS'02*, San Diego, CA, June 2002.

[2] A. Seddik-Ghaleb, Y. M. Ghamri Doudane, and S.-M. Senouci, "A Performance Study of TCP variants in terms of Energy Consumption and Average Goodput within a Static Ad Hoc Environment", *in ACM International Wireless Communications and Mobile Computing Conference, IWCMC'06*, Vancouver, Canada, July 2006.

[3] A. Seddik-Ghaleb, Y. M. Ghamri Doudane, and S.-M. Senouci, "Effect of Ad Hoc Routing Protocols on TCP Performance within MANETs," *in IEEE International Workshop on Wireless Ad-hoc and Sensor Networks, IWWAN'06*, New York, NY, June 2006.

[4] A. Seddik-Ghaleb, Y. M. Ghamri Doudane, and S.-M. Senouci, "Emulating End-to-End Losses and Delays for Ad Hoc Networks," *in IEEE International Conference on Communications, ICC'07*, (Glasgow, Scotland), June 2007.

[5] Bokyung Wang and Suresh Singh, "Computational energy cost of TCP", *In IEEE INFOCOM'04*, Hong Kong, March, 2004.

[6] D. D. Clark, V. Jacobson, J. Romkey, and H. Salwen, "An analysis of tcp processing overhead," *in IEEE Communications Magazine*, vol. 27, no. 6, pp. 23-29, June 1989.

[7] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," *in IEEE INFOCOM'01*, Anchorage, Alaska, April 2001.

[8] Dummynet, Available at http://info.iet.unipi.it/ luigi/ip dummynet/

[9] Network Simulator-NS-2. Available at www.isi.edu/nsnam/ns/

[10] P. Gauthier, D. Harada, and M. Stemm, "Reducing power consumption for the next generation of pdas: It's in the network interface," *in MoMuC'96*, Princeton, USA, Septembre 1996.

[11] http://www.sycard.com, "Sycard technologies, pccextend 140 cardbus extender," July 1996.

[12] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing", *In IEEE WMCSA'99*, Feb. 1999.

[13] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks", *Mobile Computing*, T. Imielinski and H. Korth, Eds., Kluer, 1996, pp.153-181.

[14] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", *Comp. Comm. Rev.*, Oct. 1994, pp. 234-44.

[15] Thomas Clausen, "Comparative Study of Routing Protocols for Mobile Ad-Hoc NETworks", *INRIA Research report*, RR-5135, March 2004.