# Performance Evaluation of Backpressure Routing in Integrated Satellite-terrestrial Backhaul for PPDR Networks

Natale Patriciello*, Carlo Augusto Grazia*, José Núñez-Martínez†, Jorge Baranda†, Josep Mangues-Bafalluy†, Maurizio Casoni*
*Department of Engineering *Enzo Ferrari*,
University of Modena and Reggio Emilia,
via Vignolese 905, Modena, Italy.
{natale.patriciello, carloaugusto.grazia, maurizio.casoni}@unimore.it
†Centre Tecnològic de Telecomunicacions de Catalunya (CTTC),
Av. Carl Friedrich Gauss, 7,
08860 Castelldefels, Barcelona, Spain.
{jose.nunez, jorge.baranda, josep.mangues}@cttc.cat

***Abstract*—After a disaster, it is unlikely to expect current terrestrial infrastructures to provide the coverage and the Quality of Service required by emergency operators. Currently, PPDR operators go towards a portable LTE infrastructure that provides an access network to field operators, backhauled by a wireless mesh network that can route traffic from and towards Internet through a satellite gateway. We firstly propose a new per-flow strategy for a backpressure-based routing protocol, BP-MR, and then in the aforementioned environment we analyse the original BP-MR per-packet, the proposed per-flow, and OLSR as routing protocol for the mesh network coupled with three TCP congestion control (Cubic, Hybla, Vegas) performing a typical HTTP transfer. We analysed the throughput, the latency, and the scalability over the number of the flows/operators. An extensive simulation set allows to conclude that employing BP-MR per-flow in the mesh network coupled with TCP Vegas for performing the end-to-end transfer improves the performance for PPDR professionals.**

***Index Terms*—backpressure, mesh, satellite, PPDR, rural**

## I. Introduction

After a natural or man-made disaster, commercial terrestrial networks often fail to provide the necessary support to public protection and disaster relief (PPDR) professionals. In the luckiest case they cannot sustain the sudden surge of resource demands due to congestion problems; but more often, the network simply get destroyed by the disaster.

The rising of LTE as the main wireless technology for broadband communication is now supported by the presence of portable infrastructures that provide flexible solutions for establishing coverage and increasing capacity after a disaster [1]. This portability finally concludes the process of investigation of the feasibility of LTE as Radio Access Network (RAN) during emergencies [2]. Mainly, the coverage is provided by a dense deployment formed by Small Cells (SCs), because increasing frequency re-use by reducing cell size has historically been the most simple and effective way to increase raw capacity [3].

Each mobile node that provides radio access (ENodeB) should be backhauled, in order to reach the Evolved Packet Core (EPC). Backhauling these (dense) LTE deployments is a challenging problem that can be addressed by incorporating wireless transport nodes into ENodeBs. These wireless transport nodes will feature one or multiple point-to-point interfaces, hence enabling the creation of a redundant path of wireless mesh backhaul, aiming to provide the capillarity required by the emergency operators. In this way, with the support of an appropriate routing protocol, the LTE traffic will be properly carried from/towards the EPC.

Recent developments in satellite technologies are bringing the availability of non-terrestrial high performance channels, in order to backhaul the EPC with high performance channels towards the Internet, allowing field operators to access on-line resources [4]. Overall, this system design allows to provide, after a disaster which destroys existing terrestrial infrastructure-based networks, an alternative but complete network usable for running existing services (e.g. Push-To-Talk, PTT) as well as innovative services (without the intention of being exhaustive, we can bring as examples health monitoring systems, on-line resource gathering, video streaming for field operations, ...).

In this paper we simulate an emergency deployment, which consists in an LTE access network backhauled by a wireless mesh that can access the Internet through a satellite channel. On this environment, we perform an evaluation of throughput and latency experienced by TCP connections, as well as monitoring the scalability as a function of the number of flows. We examine the interaction between routing protocols, used to carry traffic inside the mesh, and three common TCP congestion controls: Cubic, Hybla, and Vegas. The routing strategies analysed include Optimized Link State Routing (OLSR) and *BP-MR* (an existing backpressure-based routing protocol), proposing also a new variant to *BP-MR* that takes *per-flow* decisions, instead of the original *per-packet* strategy.

The experiments have been performed with ns-3[1] and show that a combination of TCP Vegas (a delay-based congestion control) with *BP-MR per-flow* offers the best performance to PPDR professionals, in the aforementioned emergency scenario.

The remainder of this paper is organized as follows. Section II contains the necessary background on *BP-MR* (both the original *per-packet* variant and the proposed *per-flow*). Section III describes the reference scenario and the methodology used to present the simulation results in Section IV. Section V covers related work, and finally, Section VI concludes the paper.

## II. BP-MR BACKGROUND

The path redundancy of the (potentially large) wireless mesh networks, that provides backhaul for the LTE access network, puts emphasis on the backhaul routing protocol ability to exploit this redundancy, since the routing protocol determines the way data is transported to/from the UEs. We have standard congestion-agnostic strategies such as Multiprotocol Label Switching (MPLS, RFC 5921) and Optimized Link State Routing (OLSR, RFC 3626, which in absence of node mobility and failures is equivalent to MPLS for the purpose of the paper), but also congestion-aware strategies such as backpressure-based ones [5]. The novelty in these strategies is that they take routing decisions by dynamically mapping the trajectory followed by each data packet to the most underutilized path, hence exploiting the network redundancy. However, these decisions may potentially make the path followed by consecutive packets of the same flow disjoint. This congestion-awareness, in theory, can help to avoid heavy utilized paths, hence smoothing the traffic from/to the field operators. For this reason, we choose to compare a standard, single and shortest path protocol OLSR and a backpressure-based protocol, *BP-MR*. In the first subsection we detail *BP-MR per-packet* [6] for sake of completeness, whereas in the second subsection we describe its new *per-flow* variant proposed in this paper.

*1) BP-MR per-packet:* For readers interested in the history of backpressure-based routing protocol, there is a comprehensive survey of backpressure state-of-the-art in [5]. The root concept consists in a centralized policy which routes traffic in a multi-hop network by minimizing the sum of the queue *backlog*s in the network among time slots. In the original proposal, there is a separate queue for each flow that passes through the node. Basically, if we define as *backlog* the queue size at nodes, the main idea of backpressure is to give priority to links and paths that have higher differential *backlog* between neighbouring nodes.

From this set of proposals we refer the reader to *BP-MR* [6], because of its proven scalability and performance improvements transporting UDP traffic over wireless mesh backhauls. Specifically, decentralized routing decisions are performed, in each node, following a two-stage process. Firstly, *BP-MR* classifies data packets in a per-interface queue system according to their final destination. Secondly, *BP-MR* employs geographic and congestion information to compute the best possible next-hop, on a *per-packet* basis, from all possible forwarding options in the multi-radio backhaul node. The per-interface queue system presents lower complexity than the original per-flow queuing system, a better delay performance compared to state-of-the-art backhauling routing protocols, and its distributed routing decision features contribute to the scalability and applicability capabilities of *BP-MR*.

*BP-MR* obtains information about surrounding network congestion conditions (queue backlogs) through the periodical exchange of control packets called *HELLO*. In a dense SC wireless mesh backhaul, with many concurrent flows and plentiful of available paths, it is likely to expect a high degree of variability in these information, so these control packets have to be exchanged quite often, for instance every 100 ms. In the worst case, every 100 ms the queue information is changed in such a way that the packets passing through the node can be redirected through another path. Hence, the *per-packet* evaluation performed by *BP-MR* could lead to different routing choices even for subsequent packets, despite the fact that they belong or not to the same end-to-end flow. Therefore, with a high probability all these packets, spread over many paths, will be received out-of-order by the end point, creating a problem for the TCP receiver.

One possible answer to this problem is selecting only similar (equal-hops/costs) paths in order to mitigate the different path delays. Unfortunately, one of the main feature of backpressure-based algorithms is their greedy approach: the nodes have only a local knowledge of the network, limited to their immediate surrounding neighbours (1-hop), which is incompatible with the full knowledge required to compare different end-to-end paths.

*2) BP-MR per-flow:* To overcome the packet reordering problem, without losing intrinsic characteristic of *BP-MR* and the capability of circumvent congested paths, we apply the concept of *per-flow* path selection strategy to *BP-MR* itself. Even if this strategy shares the name with the per-flow queuing system presented in the original backpressure proposal, our proposal does not apply to the queuing system (that continues to be per-interface, as the original *BP-MR*) but instead on the way the routing decisions are made. Through identifying a flow as an origin to destination packet stream of a transport layer connection between two end-hosts, each node maintains per-active flow state information, or in other words it maps the packets of a flow to its pre-assigned path, calculated the first time the node sees the flow. Nevertheless, a new flow has the flexibility to route dynamically to any of the available paths, and so is able to circumvent congested routes, without actually causing packet reordering at the destination. However, forwarding table size increases with the flows number, but we believe that with modern equipments and an efficient software implementation the saturation point of the network, in which there is no difference between any routing strategy, is reached well before the impossibility to store and manage the forwarding table. Note also that in terms of scalability, the state kept by each *BP-MR* node is smaller than the state required by the typical backpressure *per-flow* queuing system and the use of wild-cards rules can also significantly reduce the total state and forwarding table size kept by each backhaul node.

[1]https://www.nsnam.org

Furthermore, *BP-MR* introduces much less control overhead than schemes that need to set up and maintain end-to-end paths (e.g., *OLSR*), since *BP-MR* nodes do not need to acquire a complete view of the network.

## III. Scenario and Methodology

Our reference scenario depicted in Figure 1 is an emergency SC network deployment covering $2.5Km^2$ with inter-SC distance of five hundred meters. Note that every SC is composed by an LTE eNodeB and a wireless transport node. In particular, twenty-five LTE eNodeBs are deployed with peak downlink throughput of 350 Mb/s. The wireless transport node endows several 500 Mb/s point-to-point (PTP) interfaces that form wireless links of 4 ms of propagation delay and are connected among them to form a plain grid, as illustrated by Figure 1. The mesh is in turn connected to the mobile LTE Evolved Packet Core (EPC) through three PTP wired links, with 1 Gb/s of available bandwidth and 0.5 ms of propagation delay. The EPC is connected to the Internet through a satellite-connected gateway (the link is characterized by a bandwidth of 10 Mb/s and a propagation delay of 350 ms, with a Packet Error Rate of $10^{-4}$); the queue sizes of the backhaul network are set to the 100 % of each link bandwidth-delay product, making an effort to reduce default buffer size, as suggested in [7]. Regarding the Radio Access Network (RAN), we used European frequencies and a Okumura-Hata propagation model [8]: the LTE connection between UE and the eNodeB is modelled inside an open environment (rural) of a medium city.

We modelled a 2 MB file transfer from a remote server outside the Mobile Network to the UEs, which represents the download of a web page (including external object) from a remote web server. In all the simulations, we configured variable number of UEs from 1 to 16 attached to the Mobile Network, with one TCP transfer for each UE. Consequently, the number of TCP file transfers is equivalent to the number of attached UEs. To complete the picture, the UEs are uniformly distributed over the ENodeBs, and are placed in the left part of the mesh.

We conducted experiments for different TCP variants; we compared TCP Cubic [9], Hybla [10], and Vegas [11]. Our choice has been driven by the following facts: (i) Cubic is the default congestion control algorithm of Linux, and so is used by a great number of servers in the Internet; (ii) Hybla is a protocol specifically designed for satellite transmissions, (iii) Vegas is a delay-based protocol which focuses on maintaining low queuing delays, a property that can help in the presence of an high propagation delay to maintain an acceptable QoE for the users. Furthermore, for each TCP flavour we compared the performance of different underlying backhaul routing protocols: single-path based on *OLSR*, and backpressure *per-packet* and *per-flow* based on *BP-MR*. All the simulations have been conducted with ns-3 using latest version of LTE model[2], *BP-MR* routing protocol implemented in [6], and the different TCP variants presented in [12].

[2]http://networks.cttc.es/mobile-networks/software-tools/lena/

Our objective is to analyse the application throughput and the latency experienced among different TCP variants and routing protocols. Unfortunately, with TCP these values are biased by the protocol internals (retransmissions and sliding window mechanism). Therefore we present the download finish time as the throughput figure of merit (lower finish time means higher throughput) and the experienced RTT as worst-case latency indicator due to the network. TCP RTT is defined in RFC 6298, it is used to calculate RTO timer, and is updated in the data sender (the remote node in this case) each time it receives an ACK segment from the UE that acknowledge non-retransmitted and in-order new data. Therefore, retransmitted or out-of-order packets do not play any role in the calculation; through looking only to this value, a distracted reader can draw the wrong conclusion that, with more retransmission or reordering, the delay performance are better. So, why we use RTT and not directly an application latency measurements? Because application latency would be biased by the time that the data is waiting in the TCP buffer to be transmitted on the wire. At the end, combining download finish time with the RTT values gives the better overall view of the network performance experienced with the combinations of L3/L4 protocols.

## IV. Performance

The reported values of download time and RTT are represented in candlesticks, where the boxes stretches from the 20th to the 80th percentiles and the whiskers represent the maximum and minimum values, with the average value represented by a black horizontal line.

### *A. TCP response to different routing protocols*

In Figure 2 we present the results obtained by employing TCP Cubic, Hybla, and Vegas over different routing protocols during a 2 MB transfer from the remote node to 2 UEs (download), placed on the left side of the mesh network. On the left, we can see the reported RTT of the flows, measured during the transfer while, on the right, it is reported the completion time required to conclude the download by the UEs.

The first thing that is important to note is that the RTT is dominated by the propagation delay of the satellite, the baseline RTT is in fact roughly 750 ms. Vegas manifests a very polite behaviour with respect to the RTT figure of merit containing the entire candlesticks between 750 and 800 ms presenting, in general, a very stable performance regardless the routing protocol adopted. Also Cubic is particularly stable with respect to the RTT measurements; its values range from 750 to 950 ms, with candlesticks slightly wider if compared with Vegas. A little improvement here is reported when OLSR routing is adopted. To complete the RTT analysis of Figure 2, Hybla registered the worst values almost doubling the baseline RTT of 750 ms achieving 1.4 seconds of RTT. In this case, as well as Vegas, the routing protocol adopted during the experiment does not affect the final performance.

By moving from the RTT figure of merit to the download completion time, the evaluation changes. In fact, here it is

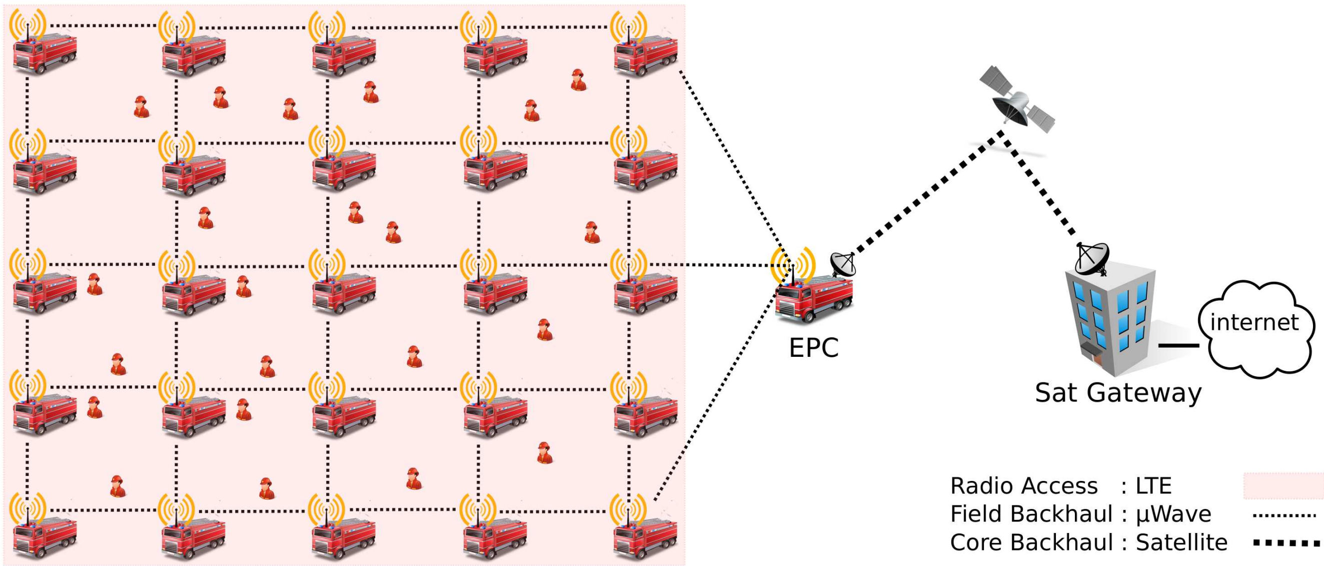


Fig. 1: Reference emergency scenario topology (UE number can vary).

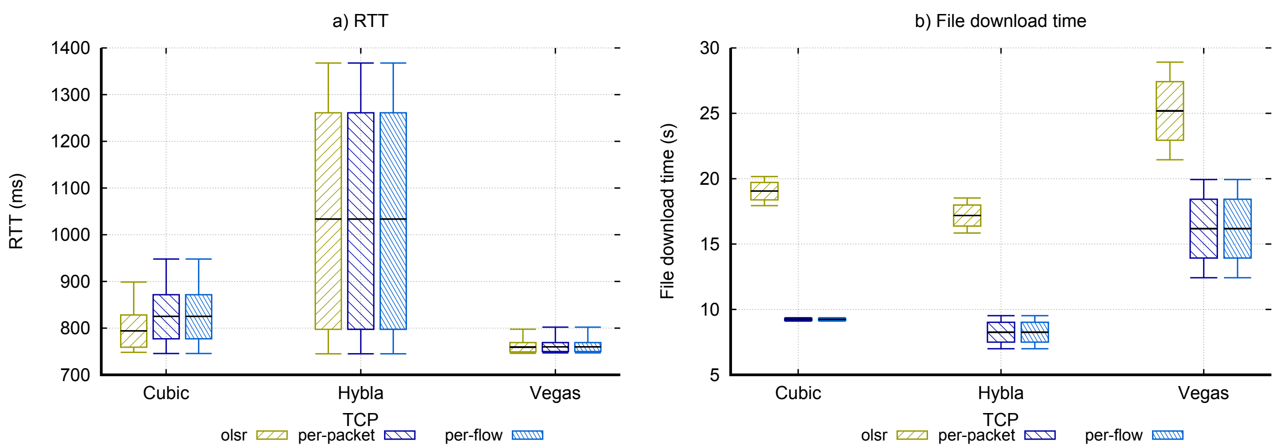


Fig. 2: RTT and file download time using different TCP for downloading 2 MB, 2 concurrent UE.

clearly visible as the choice of the routing protocol starts to affect the results. Vegas, which was outperforming the other TCP variants in terms of RTT, here suffers higher completion times if compared with Cubic and Hybla ones and higher variance (considering the wideness of the candlesticks). Hybla, which was the worst TCP algorithm in terms of RTT, here registers the best performance, with an average completion time of 7.5 seconds and 17.5 s when BP-MR and OLSR routing protocols are used respectively. It is remarkable as the use of BP-MR routing strategy saves 10 seconds of download time with respect the OLSR protocol. Finally Cubic, which was manifesting a quite good performance in terms of RTT, even with the file completion time is maintaining measures close to the best ones. The download time with the BP-MR routing strategy is slightly under 10 seconds, while the download time with the OLSR routing strategy is slightly under 20 seconds. The candlestick wideness of Cubic is the lowest of the experiment, resulting in an algorithm very stable in terms of completion time.

This first evaluation of Figure 2 gives us a preliminary idea about the impact of the routing strategy, that seems intervene mostly on the download completion time, and the impact of the TCP algorithm, that seems to mostly affect the RTT of the transmission.

In Figures 3 and 4 the above experiment is reproduced increasing the connected UEs to 8 and 16, respectively. By looking at these two figures, some results are confirmed while some others start to change. First of all, considering the RTT figure of merit of Figures 3 and 4, the better performance of Vegas, when compared with Cubic and Hybla, is confirmed. Hybla manifests a very stable behaviour with basically the same RTT range between 750 and 1400 ms regardless the routing strategy adopted. Cubic seems the TCP algorithm that suffers more the increase of the simultaneously active UEs. In the next subsections a detailed analysis regard the scalability will be provided to investigate more on this aspect.

The effect of increasing the UEs simultaneously active (and so the number of flows) is affecting in particular the download

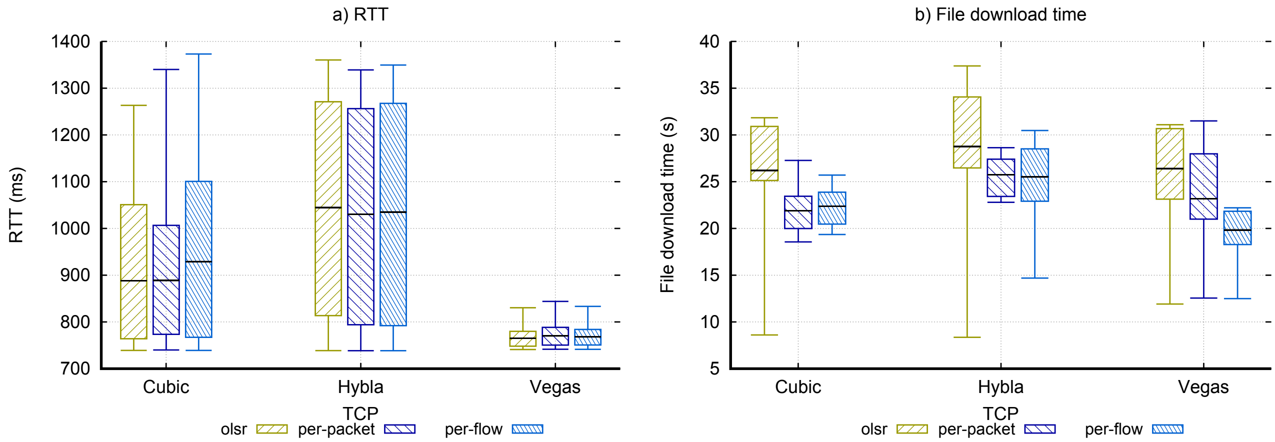


Fig. 3: RTT and file download time using different TCP for downloading 2 MB, 8 concurrent UE.

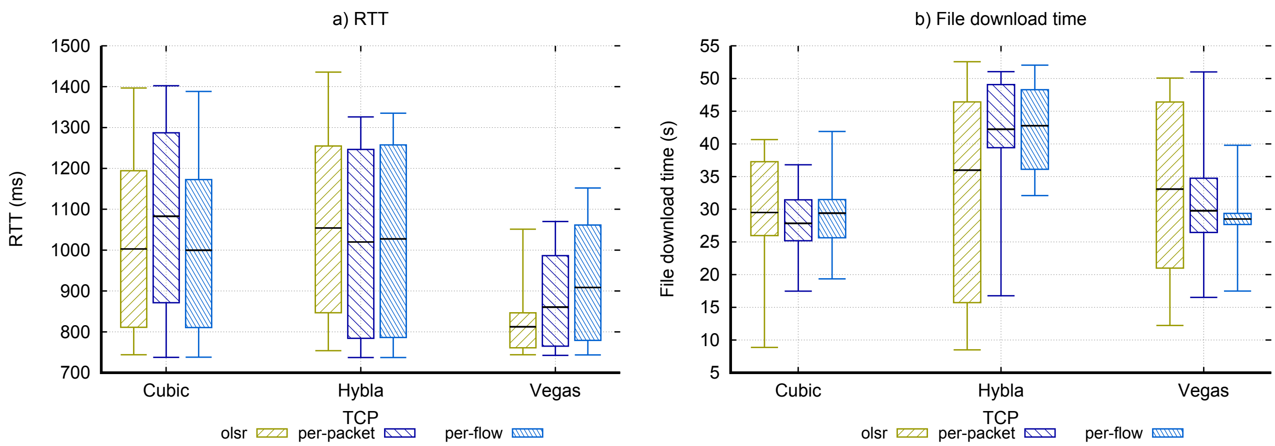


Fig. 4: RTT and file download time using different TCP for downloading 2 MB, 16 concurrent UE.

completion time results. In fact, Hybla moves from best to the worst performance inverting also the impact of the routing strategy. If in Figure 2 Hybla was reporting the best results, in conjunction with BP-MR routing strategy, regardless of the variant, in Figure 4 the same combination of routing strategy and TCP protocol is reporting the worst-case values with completion times of almost 45 seconds in average. Another remarkable change is registered by Vegas which moves from worst TCP protocol to the best. In fact, in Figures 3 and 4 it registers the lowest download time of 20 and 30 (average values) respectively. Both of these best-case values have been reported by adopting the BP-MR per-flow routing strategy. The Cubic algorithm remains very close to the best TCP in all the simulations. It was close to the result obtained by Hybla in Figure 2 and it is close to the results obtained by Vegas in Figures 3 and 4, manifesting also a very good robustness with respect to the routing strategy adopted in Figure 4.

### *B. Scalability of different TCP algorithms*

In this subsection we evaluate how the TCP variants scale as a function of the active UEs during the simulation, analysing also the impact of the routing strategy adopted. All the figures reported in this subsection are organized as in the previous ones. On the left, we report the RTT of the flows, measured during the transfer, while on the right it is reported the completion time required to conclude the download by the UEs. The download size considered for all the experiments is 2 MB.

In Figure 5 are reported the results obtained by the Cubic algorithm. Considering the RTT, it is easy to notice how Cubic starts to suffer in terms of scalability yet with 4 active UEs. In fact, the variance/range of the candlesticks move to 750-1400 ms while the average slightly increase as a function of the active UEs. The routing strategy adopted does not affect clearly the RTT performance of Cubic. Considering the download completion time, an easy thing to note is that OLSR does not provide good performance if compared with BP-MR strategies. This difference in terms of completion time performance between OLSR and BP-MR seems to be mitigated by the congestion level of the network, in fact the difference is reduced as function of the active UEs with almost equal results with 16 UEs.

In Figure 6 we presents the results obtained with TCP Hybla in place. Considering the RTT performance of Hybla, it has a very poor performance with a range between 750 and 1450 ms that is almost constant both a as a function of

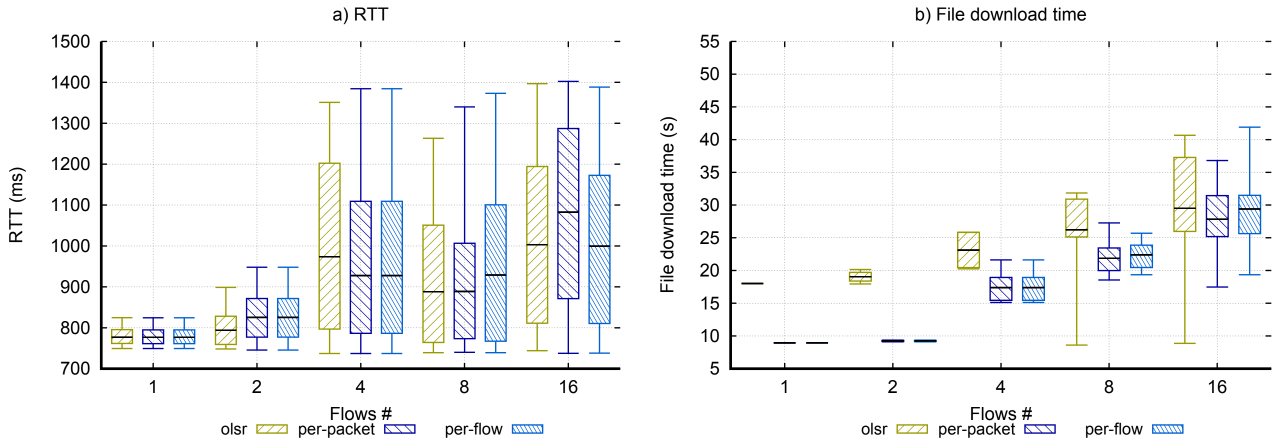


Fig. 5: RTT and file download time using TCP Cubic for downloading 2 MB, from 2 to 16 concurrent UE.

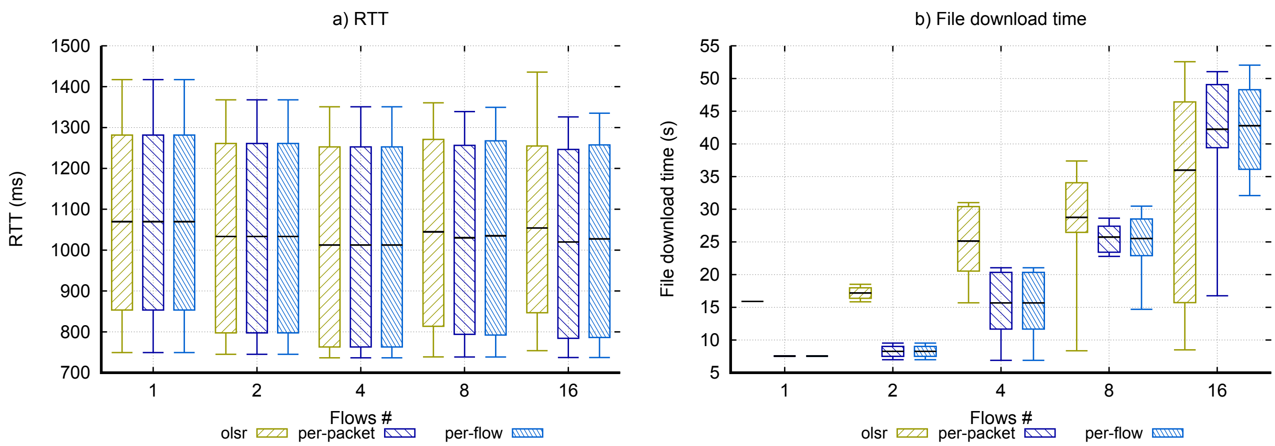


Fig. 6: RTT and file download time using TCP Hybla for downloading 2 MB, from 2 to 16 concurrent UE.

the number of active UEs and as a function of the routing strategy adopted. Minimum and maximum values as well as average and wideness scale well. The same is not true if we consider the download completion time, in fact Hybla starts with very similar result to Cubic with 1 and 2 nodes, but the growing trend of Hybla has a stronger impact than the Cubic one, i.e. Cubic scales better. In particular, with 16 UEs, Hybla reaches values higher than 40 seconds for the average download completion time. Again, as with Cubic, it is possible to note that OLSR routing strategy scales better than BP-MR strategies even when coupled with Hybla.

The Vegas algorithm is probably the most interesting one for both the figures of merit. In Figure 7 it is possible to see how Vegas scales well in terms of RTT with very stable and controlled candlesticks, a part for the once obtained with 16 UEs in which the range starts to increase. For the download completion time it is interesting how the Vegas algorithm maintains a baseline of more than 10 seconds of completion time, slightly higher than the other TCP variants that have a baseline under 10 seconds. The variance and the average of the completion times is always better for the BP-MR routing strategies instead of the OLSR one. In particular the BP-MR per-flow strategy is the best in all the experiments, with a lower maximum and average values as well as more controlled variance. The trend of the BP-MR per-flow strategy also scales better as a function of the network congestion growing less while the number of UEs increases.

## V. Related Work

The history of *BP-MR* started with *BP* [13], a self-organized backpressure routing protocol, that is a decentralized flavor of the original centralized backpressure algorithm. For dealing with sparse networks, Backpressure for Sparse Deployments (*BS*) [14] included additional extensions to *BP*. In particular, *BS* added a penalty function able to overcome dead ends in a scalable and decentralized way. However, *BS* was designed to tackle sparse topologies where nodes are equipped with a single backhaul radio and presented high inefficiencies in multi-radio deployments (i.e. with multiple backhaul interfaces). To mitigate such inefficiencies, we proposed in [6] *BP-MR*, used in this paper and detailed for sake of completeness in Section II.

TCP performance over the LTE RAN has been investigated in both simulated environment and over real data. For simulated environment, many works (such as [15], [16]) simulate the access network with simple point-to-point links, with

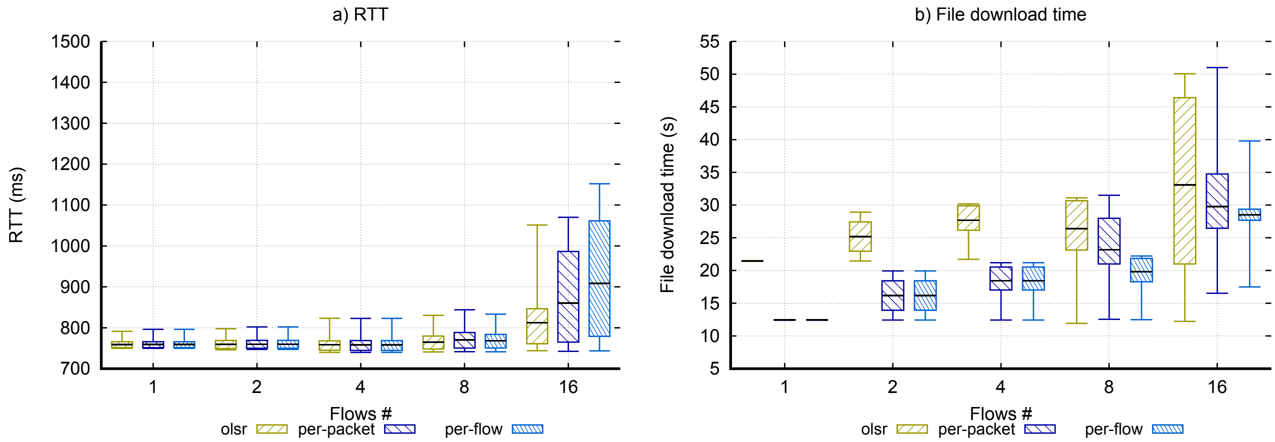


Fig. 7: RTT and file download time using TCP Vegas for downloading 2 MB, from 2 to 16 concurrent UE.

different properties, and so without taking into account the complex dynamics of the Radio Resource Control (RRC) state machine and the TCP protocol, thus invalidating the obtained results. Nguyen et al. in [17] investigates the performance of TCP over the full RAN stack, concluding that increasing the load in a cell can significantly throttle the bandwidth available to a UE, thus increasing the experienced delay, especially when the eNodeB maintains a large per-UE queue. This can invalidate the estimated RTO value, causing unnecessary TCP timeouts even when no packets are lost. Also, they concluded that radio-link handover can cause significant performance degradation. Similar conclusions are drawn in [18], where the authors analyze a large-scale real LTE data set to study the impact of protocol and application behaviors on the LTE network performance. For instance, they concluded that some TCP behaviors (such as not updating the RTT estimation using the duplicate ACKs) can cause severe performance issues; in addition, the bandwidth utilization ratio is usually below 50% for large flows. The work provides valuable insights on the interaction between TCP and LTE. However, the details of the backhaul network are out of the analysis (e.g. the routing algorithm). We aim to fill this hole by adding the analysis of different backhaul routing protocols over many TCP variants assuming a constrained wireless mesh backhaul.

For analysis of TCP traffic over backpressure routing we refer to [19], that identify the packet reordering at the receiver as the main issue with backpressure routing, and then proposes a delayed reordering algorithm at the destination for keeping packet reordering to a minimum. Other proposals use the MAC layer to perform such scheduling, for example [20]. While using the MAC layer ties the proposal to a specific technology (in [20] is used an IEEE 802.11-based wireless mesh network), we believe that avoiding reordering is more profitable than re-ordering packets in later stages. Under this light, we proposed the *BP-MR per-flow* variant. Furthermore, talking of backpressure-based algorithms, in [21] it is shown that TCP experiences incompatibilities with backpressure strategies that maintain per-flow queues, hence leading to unfairness between flows. In contrast to this work, we analyze the performance of *BP-MR*, that maintains per-interface queues, and that does not require changes at the TCP layer.

Moving to a satellite environment, the importance of routing protocols has been analyzed in [22]. In this work, a constellation of MEO or LEO satellite is considered, and performance evaluation of New Reno and SACK are carried out over two different routing strategies, shortest-path and an arbitrary multi-path protocol, that selects any minimum-hop path at a point in time. The choice is made to approximate the behavior of temporarily congested satellites; in our work, we effectively employ a real routing protocol to eliminate such approximation, and we use the state-of-art TCP protocols for satellite environments. In [23] the routing protocol is used to mitigate the effect of the handoff in LEO constellations, in order to avoid the blindly retransmission (due to the handoff). Another work that considers TCP and routing protocol in a satellite environment is [24]. The authors propose an improvement to a diversity routing strategy (i.e. a sublayer between TCP and the network that replicates each transmitted packet and sends the multiple copies along parallel paths) to overcome the issue of diversity routing over a congested network, with a satellite scenario.

## VI. Conclusion

In this paper we have proposed a new variant of the *BP-MR* routing algorithm that takes per-flow decisions instead of the originally proposed per-packet one. The goal of per-flow *BP-MR* is to reduce reordering at the destination and, in this way, to improve TCP performance. Our reference scenario has been composed by an emergency network, where field operators use an LTE RAN backhauled by a wireless mesh. The mesh is connected to the internet through a satellite gateway, in order to be portable, to provide connectivity and to be adaptable to whatever major disaster.

In such environment, we have evaluated the throughput and latency of TCP flows, providing also an insight over the scalability of the system as a function of the number of flows/users connected. The traffic has been generated by three different congestion control algorithms (Cubic, Hybla, Vegas) and routed inside the mesh by OLSR, *BP-MR per-packet*, and the newly proposed *BP-MR per-flow*.

The simulation results allow to conclude that employing TCP Vegas for the end-to-end transfers and *BP-MR per-flow* as routing protocol in the mesh help to provide better performance for field operators, compared with other combinations, maintaining also a scalable behaviour.

## VII. Acknowledgment

This work was partially funded by the EC under grant agreement no 645047 (H2020 SANSA project) and by the Spanish Ministry of Economy and Competitiveness under grant TEC2014-60491-R (5GNORM).

## References


[1] "Public Safety LTE Priorization and Preemption quality of service," Whitepaper, Motorola, July 2016. [Online]. Available: http://www.motorolasolutions.com/en_us/products/lte-broadband-systems/lte-portable-infrastructure.html#tabproductinfo

[2] M. B. Simić, "Feasibility of Long Term Evolution (LTE) as technology for public safety," in *Telecommunications Forum (TELFOR), 2012 20th*. IEEE, 2012, pp. 158–161.

[3] W. Webb, *Wireless Communications: The Future*. John Wiley & Sons, 2007.

[4] M. Casoni, C. A. Grazia, M. Klapez, N. Patriciello, A. Amditis, and E. Sdongos, "Integration of satellite and lte for disaster recovery," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 47–53, 2015.

[5] Z. Jiao, B. Zhang, C. Li, and H. T. Mouftah, "Backpressure-based routing and scheduling protocols for wireless multihop networks: A survey," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 102–110, February 2016.

[6] J. Baranda, J. Núñez-Martínez, and J. Mangues-Bafalluy, "BP-MR: Backpressure Routing for the Heterogeneous Multi-radio Backhaul of Small Cells," in *2015 8th IFIP Wireless and Mobile Networking Conference (WMNC)*, Oct 2015, pp. 48–55.

[7] K. Jamshaid, B. Shihada, A. Showail, and P. Levis, "Deflating link buffers in a wireless mesh network," *Ad Hoc Networks*, vol. 16, pp. 266–280, 2014.

[8] "The European table of frequency allocations and applications in the frequency range 8.3 kHz to 3000 GHz (ECA table)," European Conference of Postal and Telecommunications Administrations (CEPT), Electronic Communications Committee (ECC), May 2014. [Online]. Available: http://www.grss-ieee.org/wp-content/uploads/2014/07/ECA_European_Table_of_Frequency_Allocations_May_2014.pdf

[9] S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64–74, 2008.

[10] C. Caini and R. Firrincieli, "TCP Hybla: a TCP enhancement for heterogeneous networks," *International journal of satellite communications and networking*, vol. 22, no. 5, pp. 547–566, 2004.

[11] L. S. Brakmo and L. L. Peterson, "Tcp vegas: End to end congestion avoidance on a global internet," *IEEE Journal on selected Areas in communications*, vol. 13, no. 8, pp. 1465–1480, 1995.

[12] M. Casoni and N. Patriciello, "Next-generation TCP for ns-3 simulator," *Simulation Modelling Practice and Theory*, vol. 66, pp. 81–93, 2016.

[13] J. Núñez-Martínez, J. Baranda, and J. Mangues-Bafalluy, "Experimental evaluation of self-organized backpressure routing in a wireless mesh backhaul of small cells," *Ad Hoc Networks*, vol. 24, pp. 103–114, 2015.

[14] ——, "A self-organized backpressure routing scheme for dynamic small cell deployments," *Ad Hoc Networks*, vol. 25, pp. 130–140, 2015.

[15] G. A. Abed, M. Ismail, and K. Jumari, "Traffic Modeling of LTE Mobile Broadband Network Based on NS-2 Simulator," in *Computational Intelligence, Communication Systems and Networks (CICSyN), 2011 Third International Conference on*, July 2011, pp. 120–125.

[16] ——, "Behavior of cwnd for TCP source variants over parameters of LTE networks," *Information Technology Journal*, vol. 10, no. 3, pp. 663–668, 2011.

[17] B. Nguyen, A. Banerjee, V. Gopalakrishnan, S. Kasera, S. Lee, A. Shaikh, and J. Van der Merwe, "Towards Understanding TCP Performance on LTE/EPC Mobile Networks," in *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, & Challenges*, ser. AllThingsCellular '14. New York, NY, USA: ACM, 2014, pp. 41–46. [Online]. Available: http://doi.acm.org/10.1145/2627585.2627594

[18] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck, "An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 363–374. [Online]. Available: http://doi.acm.org/10.1145/2486001.2486006

[19] B. Radunović, C. Gkantsidis, D. Gunawardena, and P. Key, "Horizon: Balancing TCP over Multiple Paths in Wireless Mesh Network," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, ser. MobiCom '08. New York, NY, USA: ACM, 2008, pp. 247–258. [Online]. Available: http://doi.acm.org/10.1145/1409944.1409973

[20] F. Nawab, K. Jamshaid, B. Shihada, and P. H. Ho, "Fair packet scheduling in wireless mesh networks," *Ad Hoc Networks*, vol. 13, pp. 414–427, 2014.

[21] H. Seferoglu and E. Modiano, "TCP-aware backpressure routing and scheduling," in *Information Theory and Applications Workshop (ITA), 2014*. IEEE, 2014, pp. 1–9.

[22] L. Wood, G. Pavlou, and B. Evans, "Effects on tcp of routing strategies in satellite constellations," *IEEE Communications Magazine*, vol. 39, no. 3, pp. 172–181, 2001. [Online]. Available: www.scopus.com

[23] P. Wang and X.-M. Gu, "Routing strategy to avoid blind retransmission of tcp in leo satellite network," *Nanjing Li Gong Daxue Xuebao/Journal of Nanjing University of Science and Technology*, vol. 31, no. 1, pp. 85–88, 2007, cited By 1. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-34147216206&partnerID=40&md5=c911677b916b1f61a11a2a618a2b2499

[24] Y. Zhang, J. Chapin, and V. W. S. Chan, "Failure of tcp congestion control under diversity routing," in *2011 IEEE Wireless Communications and Networking Conference, WCNC 2011*, 2011, pp. 569–574. [Online]. Available: www.scopus.com