Efficient and Reliable Car-to-Cloud Data Transfer Empowered by BBR-enabled Network Coding

Johannes Pillmann, Daniel Behnke, Benjamin Sliwa, Matthias Priebe, Christian Wietfeld
TU Dortmund University, Communication Networks Institute (CNI)
Otto-Hahn-Str. 6, 44227 Dortmund, Germany
{johannes.pillmann, daniel.behnke, benjamin.sliwa, matthias.priebe, christian.wietfeld}@tu-dortmund.de

***Abstract*—Nowadays, vehicles are not only merely used as a transportation medium, but also as a highly mobile Internet of Things (IoT) node, collecting data from all types of sources. The delivery of aggregated vehicle sensor data into the cloud for further analysis is very fragile, as vehicles move fast in their environment and channel conditions vary heavily. Next to mastering possible packet loss, communication protocols need to quickly adapt to the frequent data rate changes. *Random Linear Network Coding* (RLNC) has been proven as an efficient and robust mechanism for reliable data transfer especially on lossy channels. In this paper, the authors extend the *Scalable Network Coding* (ScalaNC) framework with the novel congestion control *Bottleneck, Bandwidth and Round-Trip Time (BBR)* in order to quickly adapt to the frequent changes in data rate and allow effective transfer of high amounts of data from vehicles and into the cloud. ScalaNC is validated in a *Hardware-in-the-Loop* field test consisting of a Long Term Evolution (LTE) base station and channel emulator.**

I. Introduction

Todays vehicles are now being used as mobile sensor networks and collection platforms in the context of the IoT [1]. First data marketplaces are forming, e.g. the AutoMat marketplace[1], which aim at leveraging large-scale automotive sensor data for big data evaluation and analytics [2]. With an extensive amount of in-vehicle aggregated data, which sums up to several gigabytes per day, data delivery becomes more and more challenging. Vehicles move fast through their environment and thereby channel conditions are difficult and underly high frequent changes. Shadowing and multipath, next other effects, have a strong influence on link quality causing the available data rate to vary or even lead to data loss.

Since its first description in the year 2000, *Random Linear Network Coding (RLNC)* has received a strong attention in research as enabler of robust communication [3]. Starting point for the research of RLNC was the *butterfly* topology, where throughput could be increased significantly [4]. Practical applicability of *Network Coding* for wireless networks has been proven in [5].

Within the scope of this work, we leverage the Scalable Network Coding (ScalaNC) framework in order to enable reliable car-to-cloud communication. ScalaNC leverages RLNC and has been developed as part of the research project *SecInCoRe*[2].

[1]http://www.automat-project.eu/
[2]http://www.secincore.eu/

In the past, the focus of ScalaNC laid on providing connectivity for highly available safety-critical systems. However, data rates were not varying strongly. In order to adapt to the high frequent data rate changes in *Vehicle-to-X* (V2X) scenarios, the novel BBR Congestion Control Algorithm (CCA) is applied.

This paper is organized as follows: First, we analyze existing approaches of RLNC as well as competitive CCAs in section II. In section III we describe the system model, followed by an experimental evaluation in section IV.

II. Related Work

The improvement of car-to-cloud data transfer over mobile cellular networks has been on focus of many research projects in recent years. In order to fully exploit available bandwidth, congestion controls have been optimized to meet wireless network's requirements. The authors of [6] analyze the behavior of the five popular TCP CCAs CUBIC, NewReno, Westwood+, Illinois and CAIA Delayed Gradient in LTE networks. The study focuses especially on mobility scenario and provides a deep analysis on the slow start behavior. The work was picked up and refined in more detail in [7]. Both studies either investigate the cases that the User Equipment (UE) is either approaching, leaving or staying in constant distance to the eNodeB. Even though they use a realistic ns-3 simulation, it is difficult to judge if the results match real driving conditions. The authors of [8] analyze the novel BBR in comparison with CUBIC congestion control algorithm during a highway drive test. Even though BBR [9] was originally developed by Google for backends, it seems to be also well suited for LTE application. The study shows that BBR is able to "ramp up" faster than CUBIC as BBR aims at maintaining its self-inflicted Round-Trip Time (RTT) low. In addition, the analysis showed that BBR uses only very small queues in eNodeBs and makes BBR a good starting point for future cellular networks congestion control algorithms. BBR has also raised attention due to its exploitation in *QUIC* [10], the UDP-based secure and reliable transport protocol for HTTP/2. Therefore, BBR was chosen within the scope of this work as CCA for reliable and efficient car-to-cloud transfer with ScalaNC.

Network Coding has a wide field of applications and has been analyzed in different network environments. Showing that no excessive overhead is needed and that potential applications for *Network Coding* exceed the basic example of butterfly

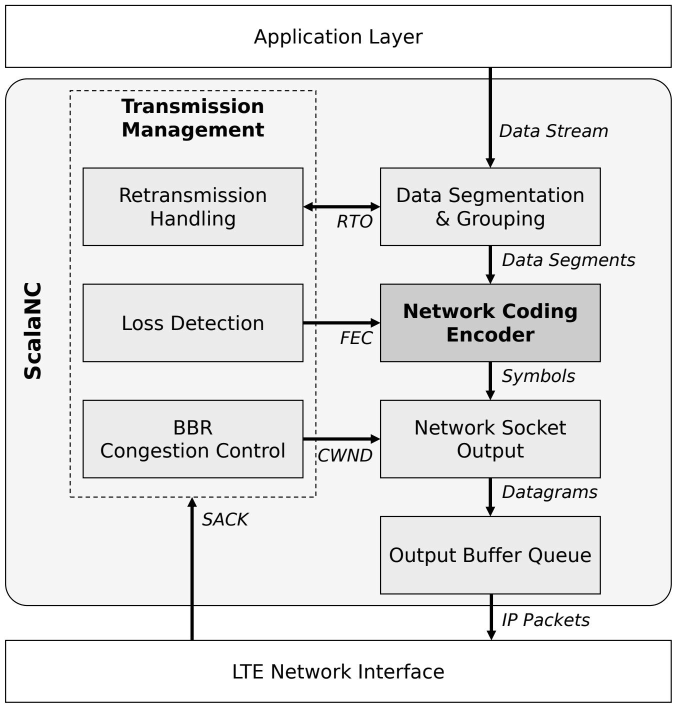

Fig. 1. Architecture model of the BBR-enabled ScalaNC solution approach

networks are provided in [11]. The applicability of *Network Coding* (NC) on transport layer via Transmission Control Protocol (TCP), accompanied with a detailed theoretic as well as practical implementation, has been demonstrated in [12] and [13]. Especially over wireless area network links the throughput can be improved significantly [5]. In [14], the usage of RLNC and User Datagram Protocol (UDP) for reliable transport service was introduced and leads to improved results.

III. ScalaNC System Model

Within the scope of this work, we extended the ScalaNC framework [15] by the novel and efficient BBR congestion control algorithm. The background of the development of the ScalaNC framework was the need for an efficient and robust communication of cloud-based emergency information systems. Its design focus lies on *high throughput*, *reliability* and *real-time capability*.

Fig. 1 shows the architecture of ScalaNC. Data is received as byte-stream from the application layer. In order to apply *Network Coding* the byte-stream first has to be segmented into smaller chunks $p_1, p_2, p_3, \dots$, each of length $d = 14500\ Bytes$. Afterwards, *Network Coding* encodes the chunks. ScalaNC leverages the *Kodo* [16] library to apply the Network Coding. Within the scope of the work, a generation size of $g = 10\ Symbols$ was used, which results in a transport layer payload of $1450\ Bytes$. This ensures that the maximum segment size of lower protocol layers is not exceeded and further packet fragmentation, e.g. on IP-layer, is avoided. The encoded data is packed into *UDP-Datagrams*. UDP has the advantage of being connectionless, which makes it beneficial in combination with network coding. In comparison to TCP it solves the problem of head-of line blocking. ScalaNC implements the novel *Bottleneck, Bandwidth and Round-Trip Time* [9] Congestion Control Algorithm in order to limit the send rate. BBR tries to optimize its congestion window by regulating the self-inflicted RTT as low as possible by not exceeding buffer limits, while at the same time adapting to the maximum bandwidth of the slowest link (the bottleneck).

Within the scope of this work, data is sent over LTE modem. The ScalaNC framework runs on a *Linux* operating system with kernel version $4.9.2$.

Within ScalaNC data is acknowledged using selective acknowledgments (SACK) on a per-generation basis. Acknowledgments are leveraged in order to determine the RTT for the BBR congestion control algorithm. Whenever the server has successfully received a full generation it is acknowledged. Lost data packages are detected by *a)* either receiving acknowledgments of subsequently sent data or when *b) Retransmission Timeout (RTO)* are triggered.

Upon retransmissions, a new generation symbol is created from the encoder. Due to the underlying RLNC algorithm it is only important for the receipient that the symbol is innovative and has no linear depdency to previously received symbols to be able to successfully decode the data. For comparison: in TCP an exact copy of the lost package needs to be transmitted. Therefore, this valueable network coding property is exploited for Forward Error Correction (FEC), which is described in full detail in the previous ScalaNC work [15]. When a certain packet loss rate is detected, additional symboles are encoded and sent, which provides the FEC.

IV. Methodology

For the evaluation and assessment of the ScalaNC, a Hardware-in-the-Loop (HIL)-experiment was conducted. In a HIL simulation, the device under test, in our case the ScalaNC is, is put into an controlled hardware setup, which imitates realistic conditions, but the full system is under control. In order to provide a realistic parametrization, channel-quality indicators were collected during a drive test covering both urban and highway situations. During the HIL experiment, the recorded values are replayed via a *Channel Emulator* and *LTE base station.*

A. Experiment Setup

Fig. 2 shows the HIL setup and system model for the car-to-cloud data transfer. Data is transferred from the client to the server. On the client side, the data aggregation from the vehicle is emulated in software. Subsequently, vehicle data is encoded using the ScalaNC encoder as described in the previous section III. Data is sent over *Sierra Wireless MC7455* LTE modem.

The Radio Frequency (RF) section of the HIL setup consists of a *Rohde & Schwarz CMW 500* LTE base station as well as an *Elektrobit Propsim C8* radio channel emulator. Using these devices in a laboratory together with the HIL simulation controller provides the advantage of a full system under control

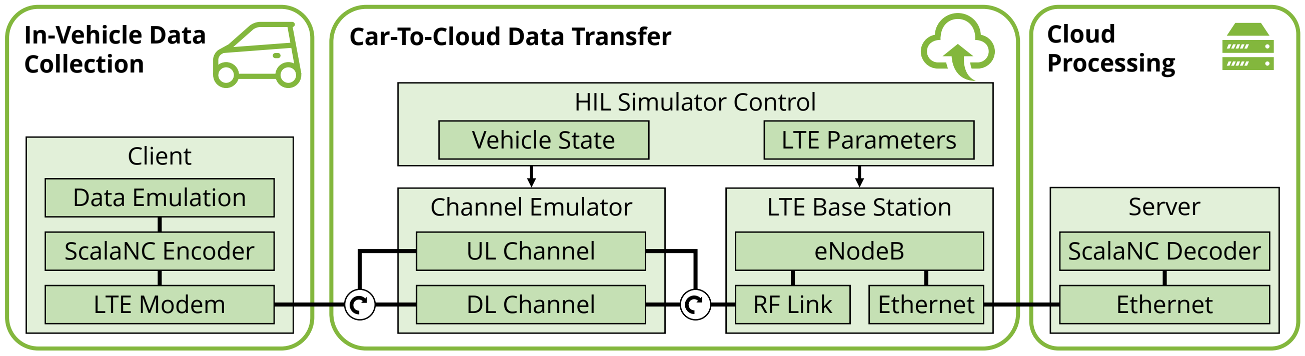

Fig. 2. Hardware-in-the-Loop (HIL) System Model for Evaluation of Network-Coding enabled Car-to-Cloud Communication

TABLE I
HIL PARAMETRIZATION FOR CAR-TO-CLOUD TRANSFER

User Equipment (UE)	**Value**
Modem	Sierra Wireless MC7455
UE Category	6
Supported LTE Release	11
Base Station	**Value**
Equipment	Rohde & Schwarz CMW 500
Carrier Frequency UL	806 MHz
Carrier Frequency DL	847 MHz
Channel Bandwidth	10 MHz
Duplexing Scheme	Frequency Division Duplex
Allocated Resource Blocks	50
MAC Scheduling	constant (single user scenario)
Modulation and Coding Schemes	TBS-IDs 1 to 19
Radio Link Control Mode	$AcknowledgedMode$
Antenna Scheme	$SISO$
Channel Emulator	**Value**
Equipment	Electrobit Propsim C8
Signal to Noise Ratio (SNR)	0 to 30 dB
Vehicle Speed	0 to 140 km/h
Fading Model	Extended Vehicular A [17]

in comparison to field tests. The LTE base station is connected via ethernet to the server, which decodes and evalutes the received data.

B. Parametrization

The car-to-cloud communication is evaluated using LTE. The *CMW 500* emulates the LTE network. It is operated in a single-user scenario on LTE Band 20 (Uplink 806 MHz, Downlink 847 MHz), which imitates one major German mobile network operator. The modulation and coding scheme and the transport block size ID (TBS-ID) selection aims at maximizing the throughput in dependency of the current channel quality indicators as described in detail in [18]. Table I provides an overview on all HIL parameters.

The *Propsim C8* channel emulator is able to control the channel conditions. Therefore, Uplink (UL) and Downlink (DL) channels are split using a RF circulator in order to emulate separate channel conditions. Afterwards, the SNR is modeled by adding Additive White Gaussian Noise (AWGN) noise until a targeted SNR is reached. Interference effects are applied using the *Extended Vehicular A model (EVA)* [17]. The channel emulator requires as input values the target SNR as well as the vehicle speed. In order to provide an realistic simulation those values were recorded in a drive test using the LTE modem. Fig. 3 shows the speed and SNR values, which are used in the HIL experiment.

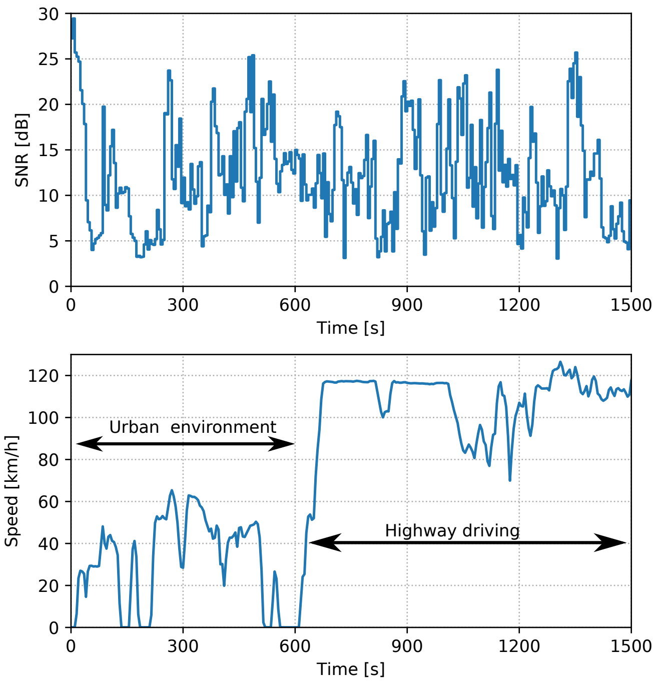

Fig. 3. Hardware-in-the-loop (HIL) parametrization for vehicle speed and Signal-to-Noise (SNR) demands, which were recorded during a real drive test.

V. CAR-TO-CLOUD DATA UPLOAD RESULTS

For the evaluation of the BBR congestion control implementation in ScalaNC the HIL experiment was conducted, as described in the previous section. Hereby, data was transferred

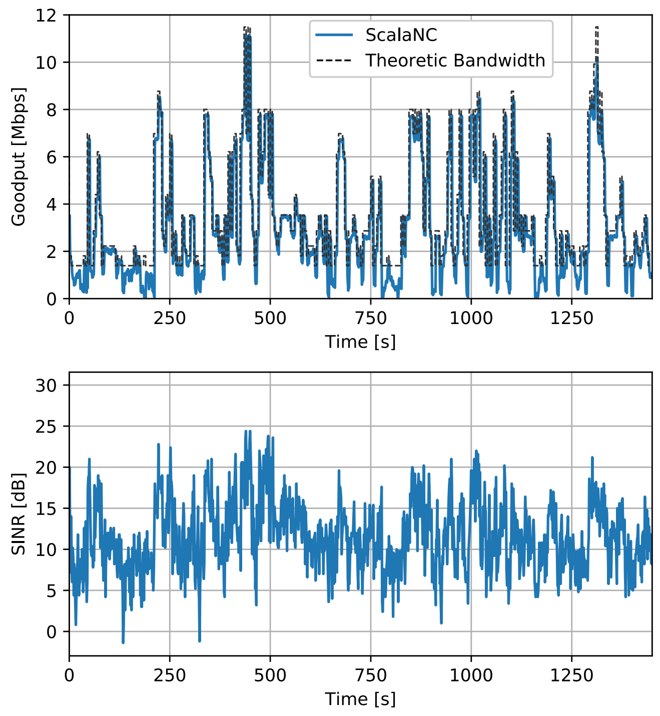

Fig. 4. Application layer throughput (*goodput*) and *Signal to Interference plus Noise Ratio (SINR)* results of the conducted Hardware-in-the-Loop (HIL) experiment

from the vehicle to the cloud. It was assumed that there is always data to be transferred.

Fig. 4 shows the resulting application layer throughput (goodput) over time. The theoretic bandwidth is plotted as a reference. The limit was derived from the leveraged modulation and coding scheme. The BBR congestion control adapts to the available congestion bandwidth very well. Especially, when there is nearly no packet loss, the channel bandwidth is fully exploited. In situations with bad SINR, e.g. due to shadowing or interferences caused from high vehicle speed, packet loss occurs and theoretic data rates diverge significantly from the actual goodput. In those cases, lost packets and thereby missing information for the network coding generations are identified due to timeouts and afterwards retransmitted.

A presentation of the initial adaption to the available channel bandwidth on smaller timescale is provided in Fig 5. The plot shows the slow start of ScalaNC's BBR implementation in comparison to TCP BBR. For both cases the data rate adapts within $0.2\ s$ to $90\ \%$ of the available bandwidth. The slight deviation at round $0.1\ s$ is caused by due to the RLNC generation based acknowledgments of ScalaNC. In comparison to TCP BBR, ScalaNC waits for multiple of $10 * 1470\ Byte$ in order to increase its congestion window, whereas TCP uses multiples of its maximum segment sizes. However, the deviation is negligible on a large scale ($> 1\ s$) and leaves room for future improvements.

Fig. 6 shows a ScalaNC's goodput in comparison to TCP Cubic and TCP BBR. The TCP experiments were conducted

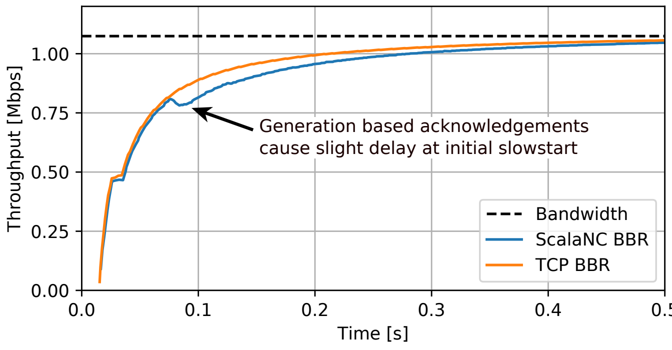

Fig. 5. Transport layer throughput of the initial slow start behavior of ScalaNC BBR in comparison to TCP BBR

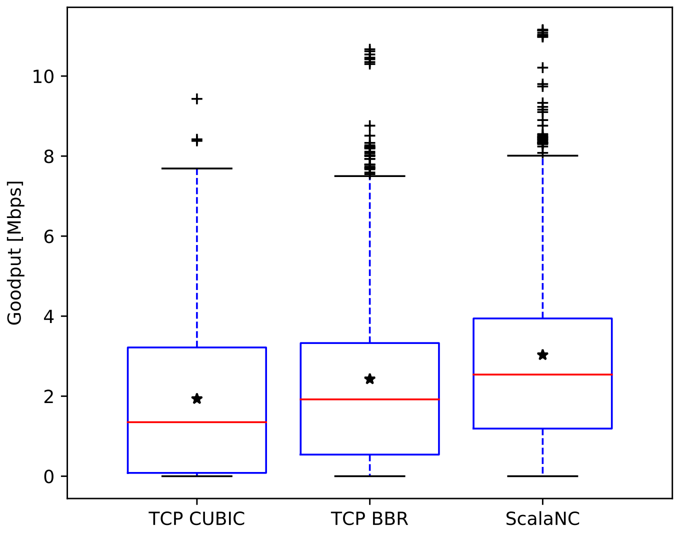

Fig. 6. Comparison of ScalaNC and TCP throughputs

without network coding with the same reproducible HIL experiment as described in the above sections. As reference measurement serves the Cubic CCA. Cubic is the most used CCA as it is default on Linux as well as android devices. In the HIL experiment an average goodput of $1.9 Mbps$ as achieved. When switching the CCA to the novel BBR the throughput improves to an average of $2.4\ Mbps$, which is an increase of $26.3\ \%$. The cubic algorithm assumes the loss of a packet as an indicator for a congestion event. Therefore, it reduces its congestion window and sends less data. Especially for wireless connections this leads to a bad performance, which is reflected by the results in figure 6. BBR performs better, because its congestion window is derived from an estimate of the communication link's bottleneck bandwidth. The estimate is derived from the amount of acknowledged data as well as the round-trip time. Therefore, BBR performs better, when packet loss occurs on the communication link.

The same can be observed for ScalaNC. ScalaNC implements BBR and therefore benefits from the algorithm. In addition it is able to apply forward error correction as well as network coding, when packet loss is expected or occured. As a result, ScalaNC achieves an average goodput of 3.0 $Mbps$. In comparison to TCP cubic this is an improvement of 57.9 %.

VI. Conclusion

Within the scope of this work, the *ScalaNC* framework was presented and evaluated. *ScalaNC* provides reliable and efficient car-to-cloud data transfer. Hereby, *ScalaNC* leverages *Random Linear Network Coding* in order compensate possible packet loss. In order to adapt as good as possible to the available car-to-cloud data rate the novel *BBR* congestion control algorithm has been implemented. In order to evaluate and assess the framework, a hardware-in-the-loop (HIL) experiment was conducted, which consisted of a LTE base station, a channel emulator and LTE cat. 6 modem. The HIL experiment was parametrized using a realistic channel conditions, which were previously recorded during a drive test. By using the *ScalaNC* framework, the throughput can be improved in comparison to a common TCP link.

Acknowledgment

This work has been conducted within the AutoMat (Automotive Big Data Marketplace for Innovative Cross-sectorial Vehicle Data Services) project, which received funding from the European Union's Horizon 2020 (H2020) research and innovation programme under the Grant Agreement no 644657, and has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 Providing Information by Resource-Constrained Analysis, project B4.

References

[1] M. Gerla, E. K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, March 2014, pp. 241–246.

[2] J. Pillmann, C. Wietfeld, A. Zarcula, T. Raugust, and D. C. Alonso, "Novel common vehicle information model (CVIM) for future automotive vehicle big data marketplaces," in *IEEE Intelligent Vehicles Symposium.*, jun 2017.

[3] R. Ahlswede, N. Cai, S. Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, Jul 2000.

[4] T. Ho and D. Lun, *Network coding: an introduction*. Cambridge University Press, 2008.

[5] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 497–510, June 2008.

[6] E. Atxutegi, F. Liberal, K. J. Grinnemo, A. Brunstrom, A. Arvidsson, and R. Robert, "TCP behaviour in LTE: Impact of flow start-up and mobility," in *2016 9th IFIP Wireless and Mobile Networking Conference (WMNC)*, July 2016, pp. 73–80.

[7] R. Robert, E. Atxutegi, A. Arvidsson, F. Liberal, A. Brunstrom, and K. J. Grinnemo, "Behaviour of common TCP variants over LTE," in *2016 IEEE Global Communications Conference (GLOBECOM)*, Dec 2016, pp. 1–7.

[8] F. Li, J. W. Chung, and X. Jiang, "Driving TCP congestion control algorithms on highway," *Proceedings of Netdev*, vol. 2, 2017.

[9] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Queue*, vol. 14, no. 5, pp. 50:20–50:53, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/3012426.3022184

[10] R. Hamilton, J. Iyengar, I. Swett, and A. Wilk, "Quic: A UDP-based secure and reliable transport for http/2," *IETF, draft-tsvwg-quic-protocol-02*, 2016.

[11] M. Medard, F. H. P. Fitzek, M. J. Montpetit, and C. Rosenberg, "Network coding mythbusting: Why it is not about butterflies anymore," *IEEE Communications Magazine*, vol. 52, no. 7, pp. 177–183, July 2014.

[12] J. K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, and J. Barros, "Network coding meets TCP: Theory and implementation," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 490–512, March 2011.

[13] C. Xu, P. Wang, C. Xiong, X. Wei, and G. M. Muntean, "Pipeline network coding-based multipath data transfer in heterogeneous wireless networks," *IEEE Transactions on Broadcasting*, vol. 63, no. 2, pp. 376–390, June 2017.

[14] P. Garrido, D. Gómez, J. Lanza, J. Serrat, and R. Aguero, "Providing reliable services over wireless networks using a low overhead random linear coding scheme," *Mobile Networks and Applications*, vol. 22, no. 6, pp. 1113–1123, Dec 2017. [Online]. Available: https://doi.org/10.1007/s11036-016-0731-7

[15] D. Behnke, M. Priebe, S. Rohde, K. Heimann, and C. Wietfeld, "ScalaNC - scalable heterogeneous link aggregation enabled by network coding," in *13th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2017) - Fourth International Workshop on Emergency Networks for Public Protection and Disaster Relief (EN4PPDR'17)*, oct 2017.

[16] M. V. Pedersen, J. Heide, and F. H. P. Fitzek, "Kodo: An open and research oriented network coding library," in *NETWORKING 2011 Workshops*, V. Casares-Giner, P. Manzoni, and A. Pont, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 145–152.

[17] ETSI, "LTE; evolved universal terrestrial radio access (E-UTRA); user equipment (UE) radio transmission and reception (3GPP TS 36.101 version 13.2.1 release 13), annex B."

[18] B. Dusza, C. Ide, P. B. Bök, and C. Wietfeld, "Optimized cross-layer protocol choices for LTE in high-speed vehicular environments," in *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, July 2013, pp. 1046–1051.