Analysis of Virtualized Turbo-Decoder Implementation for Cloud-RAN Systems

Dirk Wübben and Henning Paul
Department of Communications Engineering
University of Bremen, 28359 Bremen, Germany
Email: {wuebben, paul}@ant.uni-bremen.de

***Abstract*—The virtualization of radio access network functions using centralized cloud platforms will enable advanced joint processing approaches and offers the ability to improve the utilization efficiency of the computational resources. However, the tight timing constraints caused by the protocol stack makes the implementation of physical layer functionality on a cloud-platform a challenging task. As the Turbo decoder is the computational most demanding part of the LTE uplink processing, we analyze its implementation on a typical cloud platform, discuss the rate-complexity tradeoff, and analyze the benefit of computational aware link adaptation.**

I. Introduction

Future 5G mobile communication networks will need to support a dramatic increase in the density and rate demands of users. Currently, several novel technologies like massive MIMO, millimeter Wave communications, non-orthogonal waveforms, dense deployment of small-cells, and centralization of the Radio Access Network (RAN) are discussed for 5G [1], [2]. In particular, centralized processing allows for efficient coordinated resource allocation across multiple cells as well as joint physical layer processing for uplink detection and downlink transmission. In the frequently considered Centralized RAN (C-RAN) approach several remote radio heads (RRHs) are connected through optical fiber links (fronthaul (FH)) to a central baseband unit (BBU) where all baseband processing is performed, allowing for large centralization gains [3], [4]. To meet the timing requirements imposed by the RAN, these BBUs are built on specialized hardware platforms utilizing digital signal processors (DSPs) and field-programmable gate arrays (FPGAs) [5]. Furthermore, the strict requirement of fiber FH and the difficulty to support future requirements in 5G (e.g., extreme latency) prevent such completely centralized processing for all scenarios. Most likely, future networks will consist of a combination of distributed and centralized baseband deployments depending on the current needs and the availability of network resources like FH rates and processing power.

In [6], the Radio Access Network as a Service (RANaaS) concept has been introduced. It allows for a flexible assignment of RAN functionality between the radio access points (RAPs) and the central cloud processing center (CPC), features a tight integration of RAN, FH network, and CPC, and provides the deployment of commodity hardware at the CPC [6], [7]. Thus, it is a candidate for the Cloud-RAN concept, where cloud computing platforms are running on general purpose hardware (GP-HW) and resource virtualization is applied to match the computational resources to the actual needs. Beside its benefits, Cloud-RAN also imposes several challenges for implementing baseband processing on GP-HW mainly due to the tight timing constraints of the RAN.

When considering the physical layer (PHY), the processing load of the uplink is roughly 2.5 times the load of the downlink for a given modulation and coding scheme (MCS), and the computationally most demanding part of the uplink processing is the forward error correction (FEC) decoding [8]. In order to investigate the implications of virtualized implementation of RAN processing, we investigated in [9] the realization of the 3rd Generation Partnership Project (3GPP) Long-Term Evolution (LTE) Turbo decoder on a cloud-computing platform. In this paper, we extend this analysis by additional numerical results gained by the RANaaS testbed at the University of Bremen with a typical configuration for Cloud-RAN operator networks.

The remainder of this paper is organized as follows. In Section II the Cloud-RAN approach is presented, the implications of virtualized implementation are discussed, and the RANaaS testbed at the University of Bremen is described. The virtualized implementation of the LTE Turbo decoder is discussed in Subsections III-A and III-B.

II. Cloud-RAN

A. Functional Split

We consider the uplink (UL) operation of a Cloud-RAN (Cloud-RAN) network, where N_{RAP} RAPs are connected by FH links to a CPC. The RAPs implement the lower part of the protocol stack, whereas the cloud architecture hosts the remaining upper part of the protocol stack. In principle, several functional splits are possible as shown in Fig. 1 for UL transmission and discussed in [2], [10], [11].

One of the main benefits of the Cloud-RAN architecture is the ability to flexibly assign functionality to either the RAPs or the CPC. The actual functional split can be different for the various RAPs depending on both location and time according to, e.g., the traffic demand, FH technology or the deployment scenario. Of course, the actual split has implications on the processing needs for RAPs and the cloud platform, the reliability and latency requirements of the FH links, and the FH load. It also determines the principle gains due to centralization.

978-1-5090-3401-7/16/.00

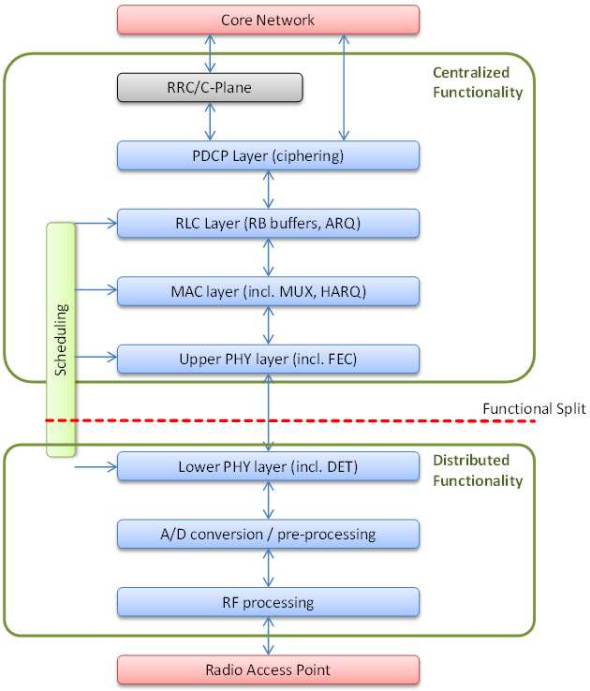

Fig. 1. Functional split between RAPs and CPC for UL transmission

In order to realize some kind of joint PHY-layer processing, the functional split needs to be placed on the PHY layer as illustrated in Fig.1. Thus, the FEC decoding needs to be executed in the CPC which requires the forwarding of soft values for the receive signals from the RAPs to the CPC and the execution of the most demanding PHY processing step on GP-HW.

B. RANaaS Testbed at University of Bremen

The use of GP-HW for the higher layer processing instead of proprietary, dedicated BBUs enables the transfer of the Infrastructure-as-a-Service (IaaS) paradigm to the RAN processing: Computational resources are allocated on demand, by spawning new instances of virtual machines (VMs) on a cloud-platform, which implement the RAN functionality above the functional split. This concept is well known from computer systems, e.g., for elastically scaling up web servers at times of high demand.

One of the main obstacles for virtualizing RAN functions on commodity information technology (IT) equipment is its processing performance. For this purpose, we performed a large-scale analysis of an LTE-compliant Turbo decoder running on a cloud-platform. Turbo decoding accounts for about 80% of the uplink processing load and is a highly stochastic process while the upper layer processing is more deterministic and less computationally intensive. The RANaaS testbed at the University of Bremen consists of commercial off-the-shelf (COTS) hardware by Hewlett Packard (2 blade servers, each equipped with 64GB RAM and 2 Intel Xeon E5-2630 processors at 2.6GHz, corresponding to 12 central processing unit (CPU) cores per blade). The testbed shown in Fig. 2 is running OpenStack Icehouse under stock Ubuntu 14.04. One blade server is a dedicated compute node, while the second one additionally acts as controller and storage node. This allows for the use of up to 20 virtual CPUs in parallel with 4 physical CPU cores exclusively allocated to management.

Fig. 2. RANaaS testbed at University of Bremen

III. Rate-Complexity Tradeoff

The tight constraints caused by the 3GPP LTE protocol stack makes the implementation of RAN functionality on a cloud-platform a challenging task. The most critical timer in LTE is associated to the hybrid automatic repeat-request (HARQ) process, which requires to finish the overall receive process within $3\,\mathrm{ms}$ to stay compliant with the 3GPP LTE timing. This timing includes the local processing of physical resource blocks (RBs) at the RAPs, the central processing at the CPC, and the round-trip time on the FH. Especially the Turbo decoding introduces a computational jitter as the decoding time varies significantly per RB. This computational jitter needs also to be considered in the overall processing delay.

A. Performance per User

In LTE, the mobile can use one out of 29 distinct MCSs which are characterized by different combinations of modulation scheme and code rate [12]. Subsequently, we present numerical results achieved for LTE uplink MCS $6 \leq I_{\mathrm{MCS}} \leq 28$ using the RANaaS testbed introduced above. The software implementation of the Turbo decoder was done in C++ using the GNU compiler (GCC), Ubuntu Linux 14.04, and multi-threading with one thread per user codeword (CW) [9] using a Qt QThreadPool [13]. The software performs LTE-compliant coding and decoding in the uplink with soft demapping and up to $N_{\mathrm{It}}^{\max} = 8$ iterations of the Turbo decoder. For each constituent decoder of the turbo loop, a double-precision Bahl-Cocke-Jelinek-Raviv (BCJR) implementation using the MAX-Log-MAP approximation is employed. No CPU-specific optimisation of the implementation was performed, which would potentially reduce the effective decoding time per codeword. Nevertheless, such optimization would not influence the computational jitter due to the number of iterations or the number of information bits per RB discussed subsequently.

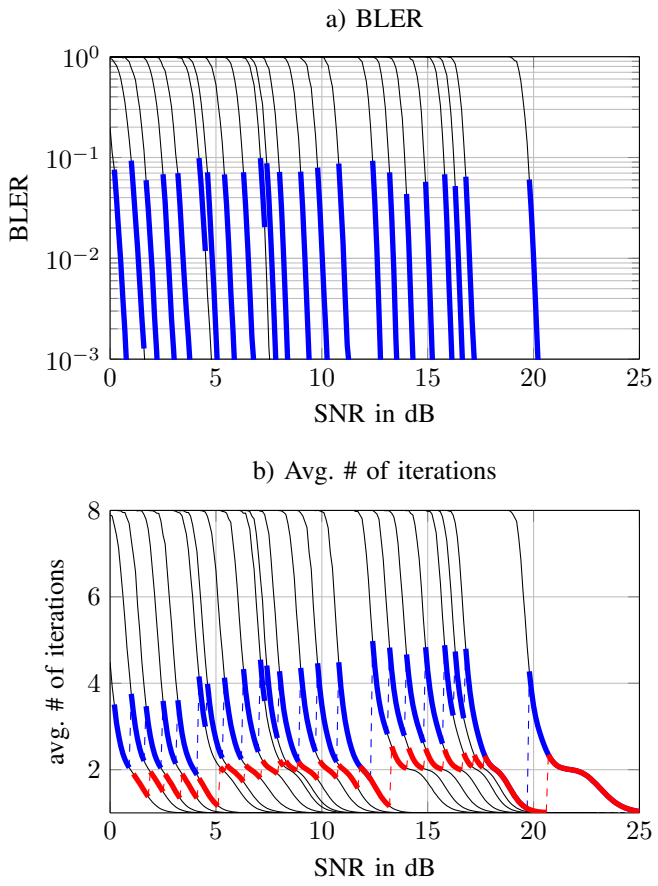

Fig. 3. a) BLER and b) average number of iterations per block for MCS $6 \leq I_{\mathrm{MCS}} \leq 28$, all MCS (—), active MCS without SNR margin (—), and active MCS with SNR margin $\Delta\gamma = 0.9$ (—)

Fig. 3 a) shows the resulting block error rates (BLERs) assuming maximum $N_{\mathrm{It}}^{\max} = 8$ Turbo decoding iterations for different MCSs versus the SNR of an additive white Gaussian noise (AWGN) transmission. The larger the MCS, the higher the necessary SNR required to achieve decreasing BLERs.

In LTE the radio link control (RLC) chooses for a given SNR the MCS such that a target block-error rate of 10% is not exceeded. Subsequently, γ_i denotes the SNR when the RLC switches from MCS $i-1$ to i. Correspondingly, the *effective* BLER of the active MCS per SNR has been highlighted in blue. In Fig. 3 b) the corresponding average number of decoder iterations per block for each MCS is depicted. Only if the SNR is high enough for a specific MCS, i.e., the chosen code rate R_{c} does not exceed the channel capacity, the Turbo decoder will converge within the maximum number of iterations (here $N_{\mathrm{It}}^{\max} = 8$). Furthermore, with an increasing SNR the required number of iterations per MCS reduces, leading to a reduction of computational complexity for an increasing SNR per MCS. Again, we have indicated the average number of iterations for the *active* MCS in blue. We can observe, that the *effective* average number of iterations varies between 5 and 2 and that peaks occur exactly when the RLC switches to the next higher MCS, i.e., at the switching SNRs γ_i.

In [9] it was suggested to reduce the processing complexity by introducing an SNR margin $\Delta\gamma$ in the RLC. With such margin, MCS i is active in the range of $\gamma_i + \Delta\gamma$ and $\gamma_{i+1} + \Delta\gamma$. Fig. 4 a) shows the achievable data rate for the case of no margin ($\Delta\gamma = 0\,\mathrm{dB}$) and a margin of $\Delta\gamma = 0.9\,\mathrm{dB}$, while Fig. 4 b) shows the measured decoding time on our demonstrator platform per CW. It is obvious that a margin of $\Delta\gamma = 0.9\,\mathrm{dB}$ only causes a small loss in data rate but reduces decoding time significantly, i.e., for all MCS up to 27, the decoding time is lower than 3 ms (relevant for HARQ).

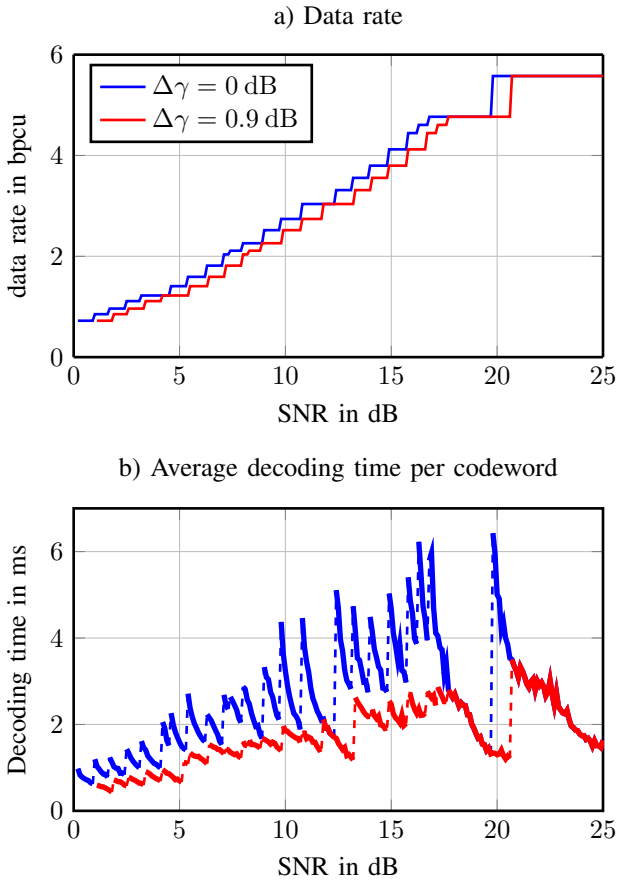

Fig. 4. a) Data rate and b) average decoding time per codeword for chosen MCS over SNR

We can observe two main overlaid effects for the decoding time. Firstly, we can observe a very peaky behaviour caused by the fact that the closer we operate to channel capacity, the more Turbo decoder iterations are necessary to decode a CW (as demonstrated in Fig. 3 b)). Secondly, the number of information bits increases with the SNR due to the higher MCSs. This causes a linear increase of the complexity and therefore processing time. Although, the absolute decoding time could be reduced by some optimized implementation, the computational jitter would remain due to these two impacts.

B. Multiuser Performance

The previous link-level (LL) analysis considered the processing effort per user for a given MCS and SNR. However,

the identified characteristic can also be used to analyze more complex scenarios present in the centralized Cloud-RAN communication system.

In a dense deployment of small-cells, the uplink processing of N_{RAP} RAPs is centralized in one cloud platform serving many users applying different MCSs based on the observed signal-to-interference-and-noise ratios (SINRs). In order to evaluate the actual processing complexity and the achievable throughput with a given number of available CPUs in the CPC, results from system-level (SL) evaluations have been used in combination with the RANaaS testbed. This joint LL/SL simulator serves the purpose of evaluating the *computational outage probability* and the *outage complexity* introduced in [14] and demonstrating the benefit of applying *computational aware* resource allocation schemes.

To this end, a 3GPP LTE compliant SL simulation including mobility was run offline beforehand and provides SINR traces with a resolution of 1 ms to the computational aware RLC, which is termed "joint Cloud-RAN scheduler". According to its allocation result, LL simulations with 1 frame containing a single CW per ms for a certain number of users N_{UE} are triggered. The measured decoding time performance for these CWs in turn serves as input to the scheduler. Please note that this joint LL/SL simulation is not running in real time, but extrapolates from the real decoding performance how many CPUs cores n_{CPU} would be required to achieve real-time decoding. E.g., for $N_{\mathrm{UE}} = 50$ users, the CPU occupation n_{CPU} depicted in blue in Fig. 5 is observed if no margin (i.e., $\Delta\gamma = 0\,\mathrm{dB}$) is considered in the RLC. Due to the fact that decoding takes more than 1 ms per CW, more than 50 cores would be simultaneously required for real-time decoding.

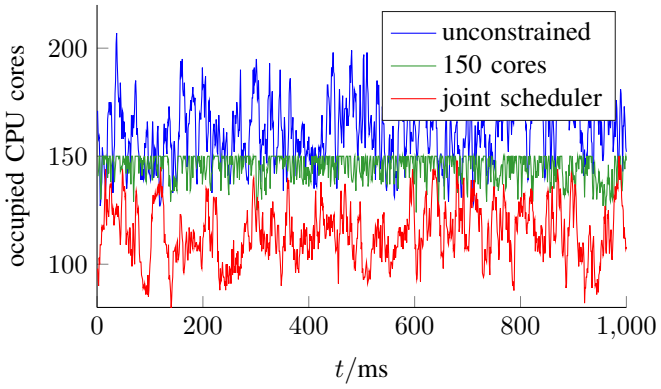

Fig. 5. Occupied CPU cores n_{CPU} over time for unconstrained case (—), hard limit of $N_{\mathrm{CPU}}^{\max} = 150$ CPU cores without joint scheduler (—) and constrained to $N_{\mathrm{CPU}}^{\max} = 150$ CPU cores with joint Cloud-RAN scheduler (—)

Due to the varying SINRs and thus, varying decoding time, the number of occupied CPU cores varies significantly over time. In practice, it is not economically reasonable to provision a system to its peak load, therefore, the number of available CPUs $N_{\mathrm{CPU}}^{\max}$ will be limited to a fixed amount smaller that the peak value. This will lead to computational outage events, since a user frame cannot be decoded in time for the HARQ acknowledgment and thus needs to be discarded. The green curve in Fig. 5 shows the limitation to an exemplary value of $N_{\mathrm{CPU}}^{\max} = 150$ CPU cores. The corresponding cumulative distribution functions (CDFs) of the normalized rates for scheduled user equipments (UEs) are shown in Fig. 6, with the average value indicated by a circle. It can be seen that compared to the unconstrained case, the average rate is reduced and the number of UEs with zero rate is increased.

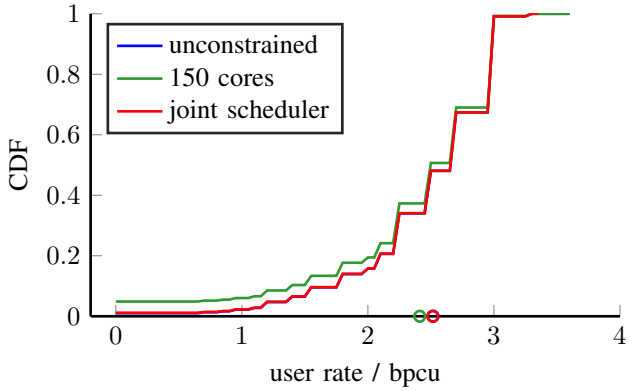

Fig. 6. CDF of scheduled user rates

In order to limit the computational load, the joint Cloud-RAN scheduler proposed in [14] assigns UEs at the lower SINR range of an MCS, i.e., close to γ_i, to the next lower MCS if an exceedance of the CPU limit is predicted. As visualized in Fig. 5 this results in a drastic reduction of the computational load, but only a negligible reduction of the average data rate even not observable in Fig. 6: While the instantaneous CPU occupation lies well below the limit of $N_{\mathrm{CPU}}^{\max} = 150$ CPU cores, the user rates are practically identical to the unconstrained case, e.g., the average rate decreases by 0.1% from 2.5168 to 2.5144.

C. Impact of virtualization on the decoding time

The measurement of decoding time in Fig. 4 has been performed in a very controlled fashion with a fixed number of CWs per SNR and the SNR iterated over a given interval. We will now contrast these measurements to values obtained by the joint LL/SL simulation. Here, SINRs provided by SL simulations are not uniformly distributed and do not adhere to discrete values. Fig. 7 shows the median decoding time over SINR binned into intervals of width 0.2 dB with error bars indicating 5% and 95% percentiles. The results obtained on the cloud-platform are shown in blue, while results obtained on a different, non-virtualized (i.e., "bare metal") system are shown in orange. It can be seen that for a conservatively provisioned cloud-platform, i.e., not assigning more virtual CPUs to VMs than physically available, the additional jitter introduced by virtualization can be ignored. However, if the VM is thin provisioned, i.e., the number of virtual CPUs allocated is not guaranteed to be physically available at all times, the decoding time per frame and in particular its spread is increased, as can be seen from the magenta plot in Fig. 7. In this case, 16

virtual CPUs were allocated while only 4 were guaranteed to be available, in contrast to that, the blue plot was obtained by allocating 4 virtual CPUs to the VM.

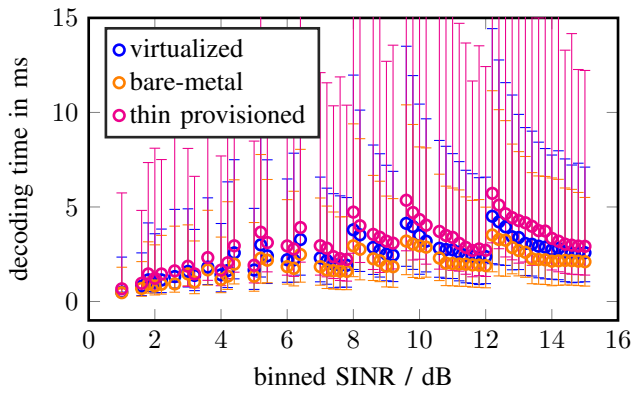

Fig. 7. Statistics of decoding time, median value with 5% and 95% percentiles, for conservatively provisioned virtualized system (—), bare-metal system (—) and thin provisioned virtualized system (—)

IV. Conclusions

Cloud-RAN combines the advantages of centralized processing with the benefits of improved utilization efficiency due to computational load balancing. For the computational most demanding function of the physical layer processing chain we demonstrate the feasibility of virtualized implementation. To this end, the rate-complexity tradeoff for the implementation of the LTE Turbo decoder on commodity hardware has been analyzed. The application of a joint Cloud-RAN scheduler strictly limits the computational load while virtually achieving the throughput of an computational unconstrained cloud processing center. Finally, the impact of virtualization of CPUs has been discussed.

References

[1] F. Boccardi, R. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Communications Magazine*, vol. 52, pp. 74–80, Feb. 2014.

[2] D. Wübben, P. Rost, J. Bartelt, M. Lalam, V. Savin, M. Gorgoglione, A. Dekorsy, and G. Fettweis, "Benefits and Impact of Cloud Computing on 5G Signal Processing," *Special Issue "The 5G Revolution" of the IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 35–44, Nov 2014.

[3] K. Chen, C. Cui, Y. Huang, and B. Huang, "C-RAN: A Green RAN Framework," in *Green Communications: Theoretical Fundamentals, Algorithms and Applications*, J. Wu, S. Rangan, and H. Zhang, Eds. CRC Press, 2013.

[4] NGMN, "Suggestions on Potential Solutions to C-RAN by NGMN Alliance," NGMN, Tech. Rep., Jan. 2013.

[5] G. Li, S. Zhang, X. Yang, F. Liao, T. Ngai, S. Zhang, and K. Chen, "Architecture of GPP based, scalable, large-scale C-RAN BBU pool," in *International Workshop on Cloud Base-Station and Large-Scale Cooperative Communications at IEEE Globecom 2012*, Anaheim, CA, USA, Dec. 2012.

[6] P. Rost, C. Bernardos, A. D. Domenico, M. D. Girolamo, M. Lalam, A. Maeder, D. Sabella, and D. Wübben, "Cloud Technologies for Flexible 5G Radio Access Networks," *IEEE Communications Magazine*, vol. 52, no. 5, May 2014.

[7] P. Rost, I. Gerberana, A. Maeder, H. Paul, V. Suryaprakash, M. Valenti, D. Wübben, A. Dekorsy, and G. Fettweis, "Benefits and Challenges of Virtualization in 5G Radio Access Networks," *IEEE Communications Magazine*, pp. 75–82, Dec. 2015.

[8] S. Bhaumik, S. P. Chandrabose, M. K. Jataprolu, G. Kumar, A. Muralidhar, P. Polakos, V. Srinivasan, and T. Woo, "CloudIQ: A Framework for Processing Base Stations in a Data Center," in *18th Annual Inter. Conf. on Mobile Computing and Networking (MobiCom)*, Istanbul, Turkey, Aug. 2012.

[9] H. Paul, D. Wübben, and P. Rost, "Implementation and analysis of forward error correction decoding for cloud-ran systems," in *Second International Workshop on Cloud-Processing in Heterogeneous Mobile Communication Networks (IWCPM 2015), co-located with IEEE ICC 2015*, London, GB, Jun 2015.

[10] J. Bartelt, P. Rost, D. Wübben, J. Lessmann, B. Melis, and G. Fettweis, "Fronthaul and backhaul requirements of flexibly centralized radio access networks," *Special Issue "Smart Backhauling and Fronthauling for 5G Networks" of the IEEE Wireless Communications Magazine*, vol. 22, no. 5, pp. 105–111, Oct 2015.

[11] J. Bartelt, D. Wübben, P. Rost, J. Lessmann, and G. Fettweis, "Fronthaul for a Flexible Centralization in Cloud Radio Access Networks," in *Backhauling / Fronthauling for Future Wireless Systems*. John Wiley & Sons Ltd, 2017.

[12] 3GPP, "Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures (Release 10)," Tech. Rep. 3GPP TS 36.213, Jun. 2015.

[13] The Qt Company, "Qt documentation of the QThreadPool class." [Online]. Available: http://doc.qt.io/qt-4.8/qthreadpool.html

[14] P. Rost, S. Talarico, and M. Valenti, "The Complexity-Rate Tradeoff of Centralized Radio Access Networks," *IEEE Transactions on Wireless Communications*, pp. 6164–6176, Nov. 2015.