Implementation and Analysis of Forward Error Correction Decoding for Cloud-RAN Systems

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

Peter Rost
NEC Laboratories Europe
Heidelberg, Germany
Email: peter.rost@neclab.eu

***Abstract*—In future 5G mobile networks, radio access network functions will be virtualized and implemented on centralized cloud platforms. In principle, this allows for more advanced algorithms of joint processing and offers the ability to balance the computational load. However, the shift of functionality on a cloud-platform also imposes challenges on the design of the applied algorithms. In this paper, we analyse the implementation of forward error correction decoding algorithms on general purpose hardware, which draws the main computational burden of signal processing in the uplink. Although the protocol stack introduces strict timing requirements we demonstrate by numerical results the feasibility of such implementation.**

I. Introduction

The evolution towards 5G mobile networks is characterized by an exponential growth of traffic caused by an increased number of user terminals and the more frequent usage of powerful internet-capable devices. Regionally and temporally fluctuating traffic patterns as well as an increasing diversity of terminal classes and services requires more scalability of mobile networks. Current mobile networks are not able to support this diversity efficiently but are designed for peak-provisioning and typical internet traffic. The use of very dense, low-power, small-cell networks with very high spatial reuse is a promising way to allow for handling future data rate demands [1], [2]. In small-cell networks, the distance between the radio access point (RAP) and terminals is reduced, and the spatial spectrum reuse is significantly increased. However, due to the density of the network, inter-cell interference increases and interference scenarios become more complex due to multi-tier interference.

Centralized processing allows for efficient interference avoidance and cancellation algorithms across multiple cells as well as joint detection algorithms. In addition, it permits to selectively turn RAPs on and off in order to address the spatial-temporal traffic fluctuations. Centralized RAN (C-RAN) recently attracted attention as one possible way to efficiently centralize Radio Access Network (RAN) processing [3]. In C-RAN, remote radio heads (RRHs) are connected through optical fiber links to a data center where all baseband processing is performed [4], [5]. Thus, by pooling baseband processing in baseband units (BBUs), centralization gains are achieved and e.g., joint downlink transmission becomes feasible [6], [7]. However, BBUs are based on specialized hardware platforms utilizing digital signal processors (DSPs) in order to meet the timing requirements imposed by the radio access network [8].

In the information technology (IT) community, cloud computing draws significant attention as it provides ubiquitous on-demand access to a shared pool of configurable computing resources. Commonly, cloud computing platforms are running on general purpose hardware (GP-HW) and by resource virtualization the computational resources are matched to the actual needs. Correspondingly, a Cloud-RAN system would allow for balancing out temporal and spatial traffic fluctuations. Furthermore, it will allow for deploying algorithms that scale with the current needs and leverage massive parallelism. On the other hand, Cloud-RAN also imposes challenges for implementing baseband processing on GP-HW due to the tight timing constraints of the radio access network.

In [9], the Radio Access Network as a Service (RANaaS) concept has been introduced. It addresses the deficiencies of C-RAN in order to allow for a centralization over heterogeneous backhaul. The main characteristics of RANaaS are the flexible assignment of RAN functionality between the RAPs and the central processor, the deployment of commodity hardware at the central processor, and the tight integration of RAN, backhaul network, and central processor [10], [11].

In this paper, we focus on implementing forward error correction (FEC) decoding on a cloud-computing platform as this is the most demanding part of the physical layer. We discuss the opportunities and the challenges, discuss implementation options, and provide numerical results gained by an implementation of a Long-Term Evolution (LTE) Turbo decoder on a Cloud-RAN test-bed. The achieved results indicate the feasibility of PHY layer processing on commodity hardware in Cloud-RAN systems.

The remainder of this paper is organized as follows. In Section II the flexible centralization by the Cloud-RAN approach is presented; Section III illustrates the particular requirements for virtualization of FEC decoding. After the introduction of the RANaaS test-bed in Section IV, simulative results are presented in Section V. The paper is concluded in Section VI.

II. Flexible Centralization through Cloud-RAN

A. Functional Split

For a centralized network the question arises, which functions are executed in the RAPs, which signals are forwarded

978-1-4673-6305-1/15/$31.00 ©2015 IEEE

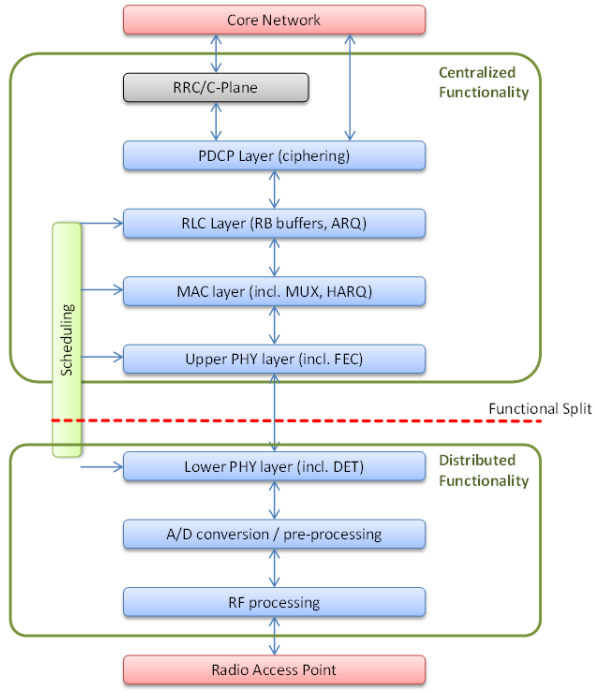

Fig. 1. Functional split between RAPs and cloud-platform for UL transmission

over the backhaul (BH) links, and which functions are executed on the cloud-platform. In principle, several functional split options are possible as shown in Fig. 1 and discussed in [12], [10]. By relying on general purpose processors (GPPs) instead of DSPs and through extensive use of function virtualization, the envisioned architecture allows to adapt the functional split flexibly in time (e. g., according to traffic demand) and location (e. g., depending on the density of the deployment).

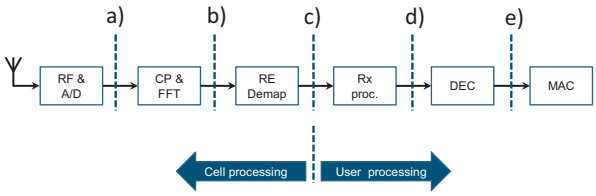

Fig. 2. Functional split between RAPs and cloud-platform for UL transmission

Fig. 2 illustrates the principle LTE signal processing chain of an uplink (UL) receiver and different options of placing a functional split. Note that similar shifts are also possible for downlink (DL) processing as considered, e. g., in the context of precoding for massive multiple-input multiple-output (MIMO) systems in [13].

If the functional split is placed on PHY layer as indicated in Fig. 1, FEC must be performed within the Cloud-RAN data center. Due to its complexity, FEC decoding is commonly performed on dedicated hardware and hence a centralized decoding on GPPs has not been considered in C-RAN. On the other hand, performing FEC locally at the RAPs according to split e) would terminate the possibility for joint PHY-layer processing and only cooperation on higher layers, e. g., joint scheduling, remains possible.

B. Opportunities of Cloud-RAN

Cloud-computing offers the ability of computational load balancing to RANs. This allows for spending more computational efforts to critical operations, e. g., in the case of interference scenarios or difficult channel conditions. In these scenarios, more advanced and computationally intense algorithms may be needed and could be executed in a cloud environment.

A flexible assignment of functionality will allow for shaping the signalling load on the BH connection. For instance, in the case of high-capacity low-latency BH, the central processor may process directly in-phase/quadrature (I/Q) samples. In the case of higher latency and lower bandwidth on the BH, the central processor may only perform upper layer functionality [10]. This will require changes to the operation of the backhaul, it will require changes to the signal processing platform, and it may require changes to the RAN protocol stack.

In cloud platforms the required computational resources are provided on-demand by introducing the concept of virtualization. This may be exploited in Cloud-RAN as well, which allows for deploying algorithms that scale with the current needs and leverage massive parallelism.

III. Virtualization of Forward Error Correction

A. Requirements

The difficulty of implementing RAN functionality in a cloud-platform lies in the tight constraints caused by the 3GPP LTE protocol stack. This implies that individual tasks need to finish within a pre-defined time window. Among all the timers defined in LTE, the one associated to the acknowledgment of a physical frame at the medium access control (MAC) layer is the most critical one. The reception status of any frame sent through the air interface needs to be fed back to the transmitter, in order to proceed to the transmission of a new frame (ACK) or to attempt a retransmission (NACK). This hybrid automatic repeat-request (HARQ) operation is performed at the MAC level, after all physical layer processing of a codeword is done (detection, demodulation, and FEC decoding).

In LTE, each frame sent at subframe n needs to be acknowledged (ACK or NACK) at subframe $n+4$ in both UL and DL direction, with a subframe lasting 1 ms [14]. Hence, the overall receive process has to finish within 3 ms to stay compliant with the 3GPP LTE HARQ timing. This timing includes the local processing of physical resource blocks at the RAPs, the central processing at the data center, and the round-trip time on the BH. However, some algorithms such as Turbo decoders underly a computational jitter which implies that the decoding time may significantly vary. Hence, it may happen that packets are re-transmitted even though they would have been decoded with more computational resources, i. e., either more time or

more parallel processors [15]. This computational jitter also adds up to the overall delay which needs to be considered.

The tight requirement of finishing the overall processing within $3\,\mathrm{ms}$ poses a significant challenge for executing FEC decoding within the cloud-platform. Usually, FEC decoders are implemented in specialized hardware, such as application-specific integrated circuit (ASIC) designs or field-programmable gate array (FPGA) implementations [16]. However, the introduction of many-core architectures opens new perspectives for massively parallel implementations. In order to meet the stringent requirements, cloud-based decoders will need to fully exploit the available parallelism of a cloud-computing platform. In this context, we investigate the implementation of LTE Turbo codes [17] in the next subsection.

Another approach to allow for centralized processing even in systems with non-ideal BH would be the adaptation of the LTE HARQ process. For instance, [12] proposed a suspension mode, which reduces the peak data rata by 50%. By contrast, the opportunistic HARQ approach proposed in [18] estimates locally the decoding error probability, performs the HARQ at the RAP, and then forwards the collected information to the central processor.

B. Approaches for Parallelization

From a high-level perspective, two main approaches can be used to exploit parallelism in multicore platforms as visualized in Fig. 3. The first approach parallelizes the decoder itself through decomposition of the decoding algorithms into multiple threads which run in parallel. The alternative approach parallelizes multiple codewords and assigns one processor to each codeword. The first approach decreases the latency per codeword but introduces more synchronization overhead across different threads. By contrast, the second approach uses less synchronization objects and therefore increases the parallelization gain. However, it may introduce a higher latency per codeword compared to the first approach.

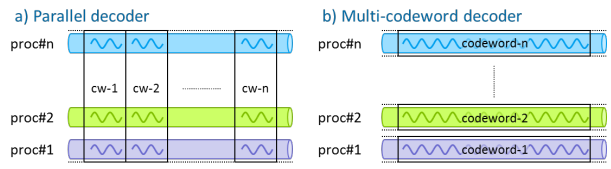

Fig. 3. Two approaches to exploit parallelism in multiprocessor platforms

C. LTE compliant Turbo Decoder

The 3GPP LTE Turbo encoding uses two parallel concatenated convolutional codes. Recursive systematic codes with generator polynomials $[1, \frac{1+D+D^3}{1+D^2+D^3}]$ are applied leading to a trellis with 8 states per constituent code and a mother code rate $1/3$. In addition, puncturing is applied in order to achieve a given effective code rate R_c.

At the receiver side, the receive signals are demapped and log likelihood ratios (LLRs) for the code bits are calculated. Then, two Bahl-Cocke-Jelinek-Raviv (BCJR) decoders are iteratively executed in order to perform a maximum a posteriori (MAP) decoding. The interaction of both decoders is done based on LLR values with double floating point precision.

IV. RANaaS Testbed at University of Bremen

One of the main obstacles for virtualizing RAN functions on commodity 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 and is a highly stochastic process while the upper layer processing is more deterministic and less computationally intensive. The demonstrator hardware consists of off-the-shelf 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 CPU cores per blade) shown in Fig. 4 running OpenStack Icehouse under stock Ubuntu 12.04. 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 central processing units (CPUs) in parallel with 4 physical CPU cores exclusively allocated to management.

The software implementation of the Turbo decoder was done in C++ using the GNU compiler (GCC), Ubuntu Linux 12.04, and multi-threading with one thread per user in order to account for operational overhead. The software performs LTE-compliant decoding in the uplink with soft demapping and up to eight iterations. For each constituent decoder of the turbo loop, a double-precision BCJR implementation using the max-log-MAP approximation is employed. No CPU-specific optimisation was performed.

Fig. 4. RANaaS test-bed at University of Bremen

V. Performance Evaluations

A. Evaluation Criteria and Performance Measures

In our testbed, we are using the block error rate (BLER) performance to calibrate our test-bed and validate the correctness of results. The actual performance measure applied is the required decoding time per codeword. Measurements of the decoding time have been obtained through the internal timekeeping mechanisms of the GNU/Linux kernel per thread. For a configurable number of blocks, the average decoding time and its variance are recorded.

B. Experimental Results for MCS 27

Simulations have been performed for LTE uplink modulation and coding scheme (MCS) 6 through 28. As an example, results for MCS 27 using 64 quadrature amplitude modulation (QAM), a block size of 3780 info bits encoded to 4752 code bits, resulting in an effective code rate of $R_c = 0.79$, will be presented. Fig. 5 a) shows the resulting BLERs for 1000 blocks assuming maximum 8 turbo decoding iterations. Due to the limited number of simulated blocks no block errors were observed for SNR larger than 17.2 dB for 8 iterations. In addition the BLER is visualized for the case that at maximum 2 turbo iterations are executed, leading to a loss of approximately 1.2 dB at a BLER of 10^{-1}.

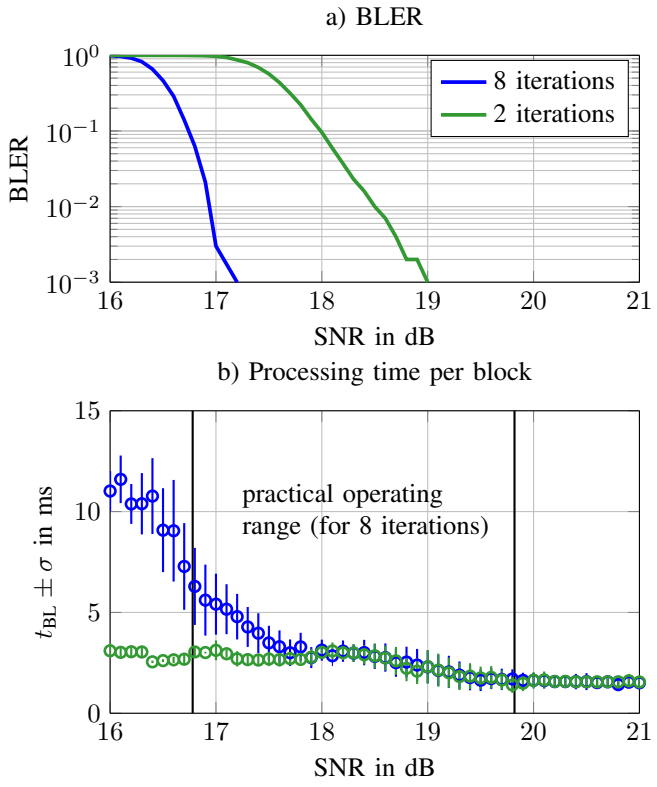

Fig. 5. a) BLER and b) average processing time ± standard deviation per block for MCS 27

The corresponding average required decoding time per block t_{BL} and its standard deviation are depicted in Fig. 5 b). This MCS achieves a BLER of less that 10% at SNRs above 16.8 dB. For the next higher MCS 28, this threshold lies at an SNR of 19.8 dB. This implies that the practically relevant signal-to-noise ratio (SNR) operation range of this particular MCS lies between 16.8 dB and 19.8 dB. It can be seen that at the lower end of this range, the decoding time is about 5-6 ms, with a significant variance. This is due to the fact that here the required number of iterations will vary heavily. This aspect is depicted in Fig. 6 which shows the statistics of the required number of iterations for the Turbo decoder where the maximum number was capped to 8 iterations.

If the maximum number of iterations is capped to 2, the

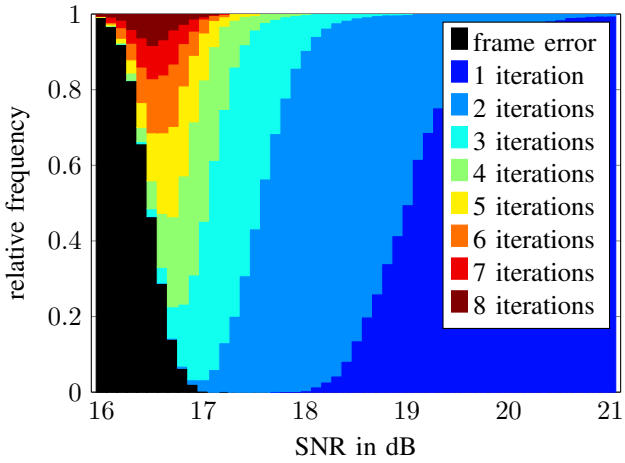

Fig. 6. Number of turbo decoder iterations for MCS 27

BLER depicted as green curve in Fig. 5 a) is obtained. We can observe that an SNR loss of approximately 1.2 dB is encountered, since the 10% BLER requirement is met at approximately 18 dB. However, inspecting the corresponding SNR range above this threshold in Fig. 5 b) for the processing time plotted in green, it can be seen that it lies below 4 ms. Furthermore, it is evident that both cases of 8 and 2 iterations show nearly identical behavior above 18 dB. The reason is illustrated by Fig. 6 which shows that above 18 dB, 2 turbo iterations are sufficient in more than 90% of the cases.

These results imply that providing a small back-off in the link adaptation performed by the radio link control (RLC), i.e., shifting the MCS selection thresholds slightly, will reduce the decoding time and thus the computational load significantly.

C. Experimental Results for different MCS

Based on further simulations of all MCS in the range of 0-25 dB for a suffiently large number of blocks, a table of SNR thresholds fulfilling the 10% BLER criterion was generated. Using this table, the RLC selects the appropriate MCS for a given SNR.

Fig. 7 a) shows the achievable data rate for the case of no back-off and a back-off of 0.9 dB, while Fig. 7 b) shows the measured decoding time on our demonstrator platform. It is obvious that a back-off of 0.9 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). We can observe two main overlaid effects for the decoding time. Firstly, we can observe a very peaky behaviour. This is caused by the fact that the closer we operate to channel capacity, the more turbo decoder iterations are necessary to decode a codeword. Secondly, the number of information bits increases with the SNR which causes a linear increase of the complexity and therefore processing time.

Fig. 8 compares the measurement results to the analytical model discussed in [15], which allows for quantifying the computational effort caused by FEC decoding. Using the appropriate parameterization of this model, we can observe

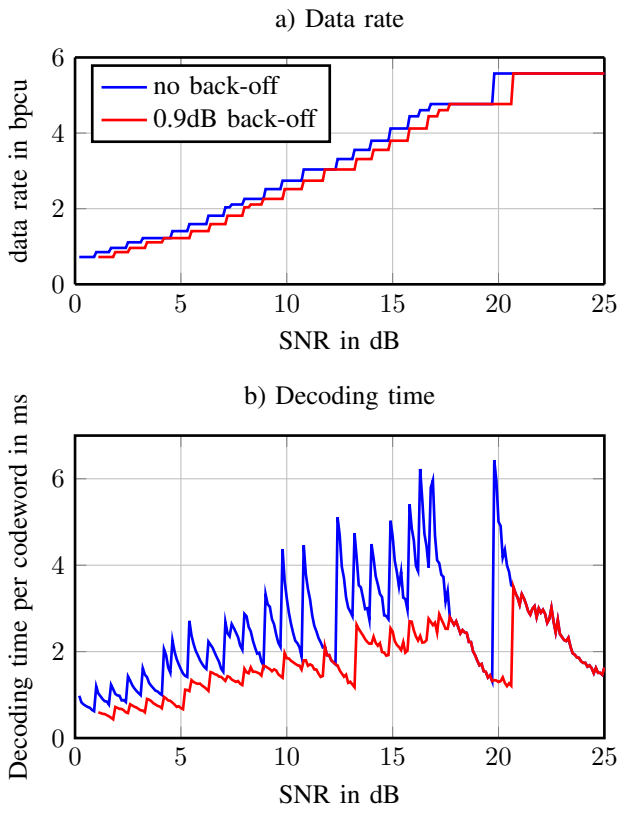

Fig. 7. a) Data rate and b) required decoding time per codeword for chosen MCS over SNR

a sufficient congruence of both results. Hence, we can utilize the model in [15] in order to efficiently control and manage the resource utilization in a Cloud-RAN data center.

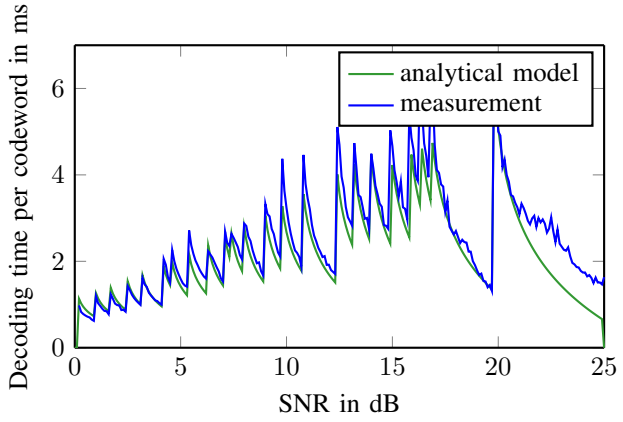

Fig. 8. Measured decoding time per codeword in comparison to analytical model

VI. Conclusions

Cloud-RAN offers massive parallel computing and allows for computational load balancing. We present throughput and complexity results for decoder implementations on commodity hardware and point out design criteria that allow for flexible load balancing.

VII. Acknowledgement

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 317941. The authors would like to acknowledge the contributions of their colleagues in iJOIN, although the views expressed are those of the authors and do not necessarily represent the project.

References

[1] M. Dohler, R. Heath, and A. Lozano, "Is the PHY layer dead?" *IEEE Communications Magazine*, vol. 49, no. 4, pp. 159–165, Apr. 2011.

[2] N. Bhushan, J. Li, D. Malladi, R. Gilmore, D. Brenner, A. Damnjanovic, R. Sukhavasi, C. Patel, and S. Geirhofer, "Network Densification: The Dominant Theme for Wireless Evolution into 5G," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 82–89, Feb. 2014.

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

[4] D. Wake, A. Nkansah, and N. Gomes, "Radio over fiber link design for next generation wireless systems," *IEEE/OSA Journal of Lightwave Technology*, vol. 28, no. 16, pp. 2456–2464, Aug. 2010.

[5] 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.

[6] Z. Ding and H. Poor, "The use of spatially random base stations in cloud radio access networks," *IEEE Signal Processing Letters*, vol. 20, no. 11, pp. 1138–1141, Nov. 2013.

[7] S. Luo, R. Zhang, and T. J. Lim, "Downlink and uplink energy minimization through user association and beamforming in c-ran," *IEEE Transactions on Wireless Communications*, vol. 14, no. 1, pp. 494–508, Jan. 2015.

[8] 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.

[9] 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.

[10] 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.

[11] 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*, Dec. 2014, submitted.

[12] U. Dötsch, M. Doll, H. P. Mayer, F. Schaich, J. Segel, and P. Sehier, "Quantitative Analysis of Split Base Station Processing and Determination of Advantageous Architectures for LTE," *Bell Labs Technical Journal*, vol. 18, no. 1, pp. 105–128, May 2013.

[13] S. Park, C.-B. Chae, and S. Bahk, "Before/After Precoded Massive MIMO in Cloud Radio Access Networks," *Journal of Communications and Networking*, vol. 15, no. 4, pp. 398–406, Aug. 2013.

[14] 3GPP, "Radio Access Network; Evolved Universal Terrestrial Radio Access; (E-UTRA); Physical channels and modulation (Release 10)," 3GPP, Tech. Rep., Dec. 2012.

[15] P. Rost, S. Talarico, and M. Valenti, "The Complexity-Rate Tradeoff of Centralized Radio Access Networks," *IEEE Transactions on Wireless Communications*, Oct. 2014, submitted.

[16] F. Kienle, N. Wehn, and H. Meyr, "On complexity, energy- and implementation-efficiency of channel decoders," *IEEE Trans. on Communications*, vol. 59, no. 12, pp. 3301–3310, 2011.

[17] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. on Communications*, vol. 44, no. 10, pp. 1261–1271, 1996.

[18] P. Rost and A. Prasad, "Opportunistic HARQ – Enabler of Cloud-RAN over Non-Ideal Backhaul," *IEEE Wireless Communications Letters*, vol. 3, no. 5, pp. 481–484, Jun. 2014.