This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Malm, Nicolas; Tirkkonen, Olav

Scheduling Latency of Midhaul-based Commodity Hardware C-RAN

Published in:
Proceedings - 2021 IEEE 4th 5G World Forum, 5GWF 2021

DOI:
[10.1109/5GWF52925.2021.00009](https://doi.org/10.1109/5GWF52925.2021.00009)

Published: 19/11/2021

Document Version
Peer reviewed version

Please cite the original version:
Malm, N., & Tirkkonen, O. (2021). Scheduling Latency of Midhaul-based Commodity Hardware C-RAN. In *Proceedings - 2021 IEEE 4th 5G World Forum, 5GWF 2021* (pp. 7-12). IEEE.
<https://doi.org/10.1109/5GWF52925.2021.00009>

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

© 2021 IEEE. This is the author's version of an article that has been published by IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Scheduling Latency of Midhaul-based Commodity Hardware C-RAN

Nicolas Malm
Aalto University, Finland
nicolas.malm@aalto.fi

Olav Tirkkonen
Aalto University, Finland
olav.tirkkonen@aalto.fi

Abstract—The impact of midhaul-induced latency on the performance of a wireless communication system must be understood in order to be appropriately accounted for. To this end, we study the impact of midhaul delay and jitter on the scheduling of a cloud radio access network. A soft-real-time commodity hardware testbed is used to gather data using up to 16 virtualized distributed units. The empirical data is used to develop models to analyze the midhaul latency's impact on spectral efficiency and outage probability. Modeling takes into account the burstiness of deadline misses in order to replicate outage duration in addition to frequency. The recorded data is then used to derive an estimate of system scaling in terms of shortest transmission time interval supported for a set number of distributed units for a given deadline miss rate. Results show that scaling behavior depends on the target rate. More stringent targets emphasize the impact of tail latencies. The testbed suffers less than 0.1 % spectral efficiency loss from deadline misses at a transmission time interval duration of 500 μ s with 16 distributed units.

Index Terms—C-RAN, SDR, midhaul, scheduling

I. INTRODUCTION

Fifth generation (5G) networks aim to support enhanced performance targets and new application categories [1] [2]. These applications present requirements not present in previous generations such as lower latency, improved mobility support and increased air interface flexibility. One of the means to achieve lower latency is the use of shorter transmission time intervals (TTI). Reducing the duration of TTIs shortens the time needed to wait for transmission opportunities by increasing the number of scheduling opportunities. Numerous methods for radio resource management (RRM) and scheduling have been proposed in the literature [3]. RRM methods provide the means to flexibly adapt the radio access network's (RAN) behavior to meet the needs of each application while maximizing the use of available resources.

Network architectures are evolving to support the needs imposed by the new requirements. One notable trend is the increased use of software-based designs [4] [5]. Software-defined approaches aim to offer benefits in terms of cost and flexibility compared to traditional hardware-centric designs. Hosting multiple virtualized software-based base stations (BS) on the same hardware enables cost reduction, as well as making inter-BS co-operation easier to implement. Use of

commodity hardware and general-purpose platforms further eases development as well as reduces costs. Cloud radio access networks (C-RAN) group virtualized instances into computing resource pools. Functionality is divided between a centralized pool and locations closer to the cell sites via a midhaul link. Joint computation and communication [6] and multi-access edge computing (MEC) [7] concepts considered for beyond 5G (B5G) networks also favor software-based implementations due to the possibility of integration with non-communication-related software.

Using software-based general-purpose platforms for C-RAN poses challenges with regards to timing-critical functions as they are not purpose-built for executing BS functions. Real-time use-cases divide into hard-real-time and soft-real-time [8]. The difference lies in the consequences of deadline misses. Hard-real-time systems fail completely while soft-real-time system may continue to operate albeit with degraded performance. A significant advantage of soft-real-time over hard-real-time approaches lies in the lack of need for challenging and time consuming verification [9] [10]. The lack of worst-case execution time guarantees in soft-real-time designs means large deviations can occur even though performance is good on average. Determining suitability for a particular use thus involves assessing the impact of the service time distribution's tail.

The concurrent diversification of application requirements and evolution of network architecture raises the question of achievable performance and suitable functional splits. In particular, the demands of B5G applications for lower latency puts pressure to shorten TTIs, which, in turn, requires faster RAN scheduling. This paper studies the impact of midhaul-induced latency on scheduling performance. A model is introduced to characterize the latency behavior of midhaul links. A dataset is obtained from a midhaul testbed to investigate the probability of deadline misses. The model is used to assess the impact of midhaul-induced late arrivals of scheduling decisions on spectral efficiency (SE) and the outage probability of user equipment (UE).

This paper is organized as follows. Section II presents the system model and metrics studied. Empirical data obtained using the testbed is then presented in Section III. The midhaul model is introduced and discussed in Section IV. Section V compares the model with the empirical data. Finally, Section VI concludes this paper.

This work was funded in part by the PriMO-5G project funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 815191.

Fig. 1. Midhaul connecting the layers of functionality placed in the CU with those in the DUs.

II. SYSTEM MODEL

The studied system architecture comprises centralized units (CU) and distributed units (DU) in an ultra-dense network (UDN) urban environment. A city is divided into areas of responsibility allocated to CUs. These areas of responsibility form the cells of the system. DUs are placed within each CU's cell according to network planning considerations.

A. RAN Architecture

CUs manage the resources within their cells. They do so based on information obtained from their attached DUs. DUs report the result of their measurements to the CU. The CU thus has centralized information with which to form an overall view of the situation within their cell. Resource allocation decisions made by the CU are sent to the DUs for execution. The division of tasks between CU and DU forms the functional split of the architecture [11]. The functional split studied in this work is Option 5 as defined in [12]. The MAC layer is split such that the CU handles inter-DU interference mitigation while DUs perform link adaptation and retransmissions.

Communication between CU and DU occurs over a midhaul link, as shown in Figure 1. The midhaul connects the functionality placed in the CU with the functionality in the DUs. Messages over the midhaul replace the exchange of information between protocol layers that takes place inside one physical node in conventional designs. Due to latency constraints imposed by the midhaul, DUs handle tasks requiring the fastest response times or which would result in the most information being sent to the CU.

Both CUs and DUs execute their allocated functionality as *soft-real-time* applications on commodity hardware with general-purpose operating system (GPOS). DUs on one server share the midhaul to the CU for cost reasons as deploying new fiber connections can be expensive [13].

Handovers between CUs are more costly than between DUs due to the need to transfer UE state and context from one CU to another. There might also be a need for the CUs to co-ordinate the resources used by the UE. It is therefore

advantageous to minimize inter-CU handovers by maximizing the number of DUs per CU. This, however, increases the contention on the shared midhaul link.

B. Midhaul Timing Accuracy

Base station disaggregation into physically separate CU and DU nodes introduces uncertainty in the timing of information exchange between layers of the RAN protocol stack. Timing diverges due to two reasons. The first is the time taken by each component to perform its task, including queueing in the case of shared resources. Queueing occurs due to the DUs' messages contending for the shared midhaul. This can be the case, for example, when a measurement report is scheduled at the same point in time for all DUs. Processing and queueing delays impact both commands from CU to DU as well as reports from DUs to the CU. The second error source is imperfect clock synchronization between CU and DU. Inaccuracy in clock synchronization affects scheduled tasks and timestamped information. Errors are introduced as the CU commands a certain action to take place at time T but the clocks in DUs differ causing the action to occur at $T + \delta$ instead.

The soft-real-time nature of the system impacts the distribution of the delay. Execution times have no upper bounds as soft-real-time systems do not undergo worst-case execution time analysis. Instead, they are engineered to cope with occasional deadline overruns. Consequently, the midhaul may occasionally experience latencies much larger than the typical case. A suitable midhaul latency model must thus capture these large deviations as they are the ones most likely to cause deadline misses. We define the proportion of TTIs with latencies greater than the threshold as the late rate denoted R_L . This work uses as thresholds the 5G New Radio (NR) [14] TTI durations of 125 μ s, 250 μ s, 500 μ s and 1000 μ s. As TTI instructions arriving late are discarded, the late rate indicates the ratio of lost transmission opportunities to the total.

C. Scheduling Metrics

Midhaul performance is assessed from the user and network perspectives. Empirically recorded CU-DU midhaul latencies are compared to those generated by simulation. From the raw latency results, further values are computed to analyze the impact of late scheduling instructions on the service received by the UEs and the efficiency of the RAN. Values are averaged over all DUs. For example, in the case of four DUs, R_L of DUs DU_1 - DU_4 are averaged to produce a single value representing the four DU case.

Relative spectral efficiency (SE) loss from late scheduling instructions quantifies the impact of midhaul delays on overall network performance. SE degrades for each instance of late transmission of scheduling information. This is because in the used model, all late instructions are considered completely outdated and will result in failed transmissions to UEs. Relative SE is

$$S_E = 1 - R_L \quad (1)$$

TTI	1	2	3	4	5	6	7	8
DU 1								
DU 2								
DU 3								
Count	0	1	0	2	1	0	1	1
Mean	0	.33	0	.66	.33	0	.33	.33

Fig. 2. Example latency results. There are six deadline misses: one for DU 1, three for DU 2 and two for DU 3.

Performance from the UE viewpoint is analyzed by studying the outage durations experienced by UEs. Consecutive deadline misses can lead to outages longer than the TTI duration. Figure 2 illustrates such a situation for DU 2 during TTIs 4 and 5. Any UE connected to DU 2 would have not received data transmissions for a period equal to two TTIs. The duration of an outage can be calculated by multiplying the number of consecutive TTIs in the outage by the duration of a TTI.

III. TESTBED DATA

The dataset contains message latencies between CU and DU for the number of DUs in the range $N_{DU} = 1, 2, 4, 8, 16$. For each value of N_{DU} , 5×10^7 message transfers were recorded. Messages contain 35 bytes of payload data in addition to Ethernet headers. Addressing is done using MAC addresses with no higher layer protocol. Messages are sent at 1 ms intervals. The testbed consists of two servers. The servers are equipped with AMD Epyc 7401P CPUs and Intel X710-DA4 network interface cards. One server operates as the CU and the other hosts the virtualized DU instances inside containers. The two servers are connected directly using two connections: one for the midhaul link and one for Precision Time Protocol (PTP) [15] clock synchronization. PTP enables the two servers to share a common notion of time over a standard Ethernet connection lacking built-in clock transfer or synchronization. The midhaul link is 10G Ethernet, while the PTP link is 1G Ethernet.

Measurement software was run on an unmodified Linux-based GPOS with real-time scheduling priority and pinned to specific CPU cores to reduce execution time variance. To further reduce latency, the Express Data Path (XDP) facility of the Linux kernel was used. XDP provides a means to bypass the kernel networking stack to reduce processing latency. Data was collected by recording the difference between the expected and actual time of message arrival from the CU's perspective. This means the recorded values include not only the latency encountered by the message on the midhaul but also the offset in clocks between DU and CU.

Table I gives the average latencies of midhaul messages for each N_{DU} tested. The values are averaged over all DUs for each case. It can be observed that mean latency scales with the number of DUs. Figure 3 presents the histogram of recorded latencies for the four DU case. Two regimes can be observed. The first contains the majority of recorded latencies below approximately 175 μ s. These values are grouped into four peaks, matching the number of DUs, and represent the on-average good performance of the system. The second part

TABLE I
EMPIRICAL AVERAGE LATENCY VALUES FOR EACH N_{DU} .

N_{DU}	1	2	4	8	16
Mean (μ s)	55.1	66.5	95.3	152.8	266.0

Fig. 3. Empirical latency histogram for 4 DUs.

of the histogram contains higher latencies forming a long tail. Figure 4 presents a similar latency histogram for the eight DU case. A similar structure to the four DU case can be seen. Values in the low-latency regime are characterized by log-normal distributions equal to the number of DUs. These likely result from the fact that all DUs' messages share a single midhaul and therefore experience queueing. The first DU served will see lower latencies than the others. Jitter comes from the soft-real-time nature of the platform as computations do not have constant duration. The high-latency regime tail shape varies according to the number of DUs. The transition point between low and high latency regime occurs at a value dependent on the number of DUs.

Table II gives the measured relative SE for each combination of N_{DU} and TTI length. It can be seen that in the majority of cases SE is only slightly degraded. Remaining cases, on the other hand, suffer significant SE degradation due to midhaul latency. Table III represents outage rate and length statistics for each of the considered TTIs. Durations were obtained by multiplying the average duration of an outage in TTIs by the TTI duration across all four DUs. The outage lengths are thus not necessarily multiples of TTI duration. Such values cannot occur in the actual system as either a TTI is fully late or fully on time, resulting in outages of durations that are integer

Fig. 4. Empirical latency histogram for 8 DUs.

TABLE II
MEASURED RELATIVE SE FOR EACH N_{DU} .

TTI N_{DU}	125 μ s	250 μ s	500 μ s	1000 μ s
1	0.9999659	0.9999787	0.9999886	1.000000000
2	0.9999650	0.9999789	0.9999811	1.000000000
4	0.7460000	0.9999736	0.99998930	0.99999655
8	0.3590000	0.9831000	0.99998130	0.999992400
16	0.1560000	0.4680000	0.99985500	0.999990140

TABLE III
MEASURED OUTAGE DATA FOR FOUR DUs. AVERAGE OVER FOUR DUs.

	125 μ s	250 μ s	500 μ s	1000 μ s
Average	162.50 μ s	570 μ s	695 μ s	1015 μ s
Maximum	38 700 μ s	1500 μ s	2750 μ s	1250 μ s
Rate	0.254	2.64e-05	1.068e-05	3.44e-07

multiples of the TTI.

Longer TTIs in Table III produce fewer outages but of longer average duration. For instance, the average outage duration for 250 μ s is higher than for 125 μ s but has a lower occurrence rate. It can thus be inferred that processing is fast enough on average but external processes can still occasionally cause interference. Typically, approximately two TTIs are late in a row per outage at 250 μ s. In contrast, for 125 μ s, there is a greater number of outages but more spread out in time. For the 1000 μ s case, the average outage duration matches the TTI duration. This means every outage lasted for one TTI. The system therefore had enough time during a single TTI to recover from the deadline miss.

IV. MIDHAUL DELAY MODEL

Understanding the characteristics of midhaul latency is important to assess its impact on the performance of a functionally split RAN design. A midhaul latency model provides a tool for studying the impact of task distribution and scaling during system design.

We apply a delay model which describes the difference in time between the sending and reception of a message across the midhaul. Modeling is performed at two levels of abstraction. The first is DU-level behavior. The aim is to model the probability of deadline misses and the pattern in which they occur. Late TTIs can cluster together because of interference from other software processes lasting longer than one TTI. The shorter the TTI duration, the more likely it is that the interfering process will impact multiple TTIs. Midhaul latency patterns will thus exhibit memory due to external factors. If one TTI is late, there is a higher probability of the next one also being late due to the cause of deadline miss persisting. Studying the burstiness of late TTIs is important as it reveals information concerning the duration of outages. If deadline misses cluster together, outages will be longer than if the late TTIs are spread out.

Late TTI distribution will vary from system to system. The characteristics of the hardware and software used will impact

outage patterns. It is, however, likely that many soft-real-time system configurations will exhibit similar clustering behavior, especially for shorter TTIs. A greater number of short TTIs will be affected for a given duration of interference from other software processes. Interference is a possibility when sharing resources, which must be done in order to realise cost benefits from co-location. Since soft-real-time systems do not have bounded execution times, software processes may occasionally encounter greater-than-average delays due to resource conflicts.

The relationship between midhaul latency and R_L is modeled as a discrete-time Markov chain to capture memory effects. Latencies are divided into classes because the precise value of the latency is not necessary to predict late TTIs. It is sufficient to know whether the value lies above or below the target threshold which determines whether a deadline miss occurs. The latency L for each TTI is represented as belonging to one of five classes. Class latency bounds were derived from the 5G NR TTI lengths used as targets to represent whether a given midhaul delay would have caused a deadline miss for a given TTI length. The classes used are:

- S_1 : $0 < L < 125 \mu$ s
- S_2 : $125 \leq L < 250 \mu$ s
- S_3 : $250 \leq L < 500 \mu$ s
- S_4 : $500 \leq L < 1000 \mu$ s
- S_5 : $L \geq 1000 \mu$ s

Use of a Markov chain provides information about midhaul latency in two ways. First, computing the steady state probabilities for each state allows for computation of R_L . For example, to obtain an estimate for the number of deadline misses for 250 μ s, the sum of S_3 , S_4 and S_5 is taken. Second, the state transition probabilities provide information as to how likely it is for consecutive TTIs to be above the threshold. This is valuable as it can model situations where deadline misses are more likely to occur in bursts. Such an event could occur due to a process being blocked due to the soft-real-time nature of the platform or when recovering from a deadline miss cannot be completed instantaneously.

The second level of modeling aims to predict scaling on the network level. Such a higher-level model helps in algorithm and network design. Design decisions depend on metrics describing the implementation's impact on communication performance. This work studies the shortest TTI duration the midhaul can support for a given R_L . Midhaul latency, and thus shortest TTI length supported, clearly scales with the number of DUs. This results in the probability of missing deadlines increasing for a fixed TTI duration as the number of DUs increases. Based on the empirical data of Table I, a linear model is selected. The model for the minimum TTI length that can be supported for a given target late rate R_L and N_{DU} is given by

$$f_{\text{TTI},R_L}(N_{DU}) = C_1 N_{DU} + C_2 \quad (2)$$

where N_{DU} is the number of DUs and C_i are constants for a given R_L . Each R_L will have its own set of C_i indicating

TABLE IV

MARKOV CHAIN STATE TRANSITION PROBABILITIES FOR FOUR DUS. EACH CLASS REPRESENTS A GIVEN LATENCY RANGE IN MICROSECONDS.

	S_1	S_2	S_3	S_4	S_5
S_1	0.734	0.266	4.94e-06	4.93e-06	1.66e-07
S_2	0.773	0.226	9.93e-06	7.56e-06	1.02e-07
S_3	0.257	0.166	0.443	0.128	0.007
S_4	0.286	0.177	0.261	0.267	0.009
S_5	0.312	0.190	0.066	0.418	0.014

how much N_{DU} affects minimum TTI length for that R_L .

A simulation was carried out to assess the ability of the proposed Markov chain approach to re-create the characteristics of the empirical data. Transition probabilities (Table IV) for each state were derived from a Markov chain model fit to the empirical data for four DUs using the R [16] package markovchain [17]. A total of 5×10^7 transitions were simulated to match the length of the empirical data. The number of occurrences of each state was recorded to compute the number of instances the TTI duration was exceeded for each TTI length. Doing so yields R_L , and thus the SE loss. Additionally, the length of sequences above the target threshold was recorded to obtain the length of outages.

V. RESULTS AND ANALYSIS

Comparing the modeled and empirical SE loss can be done using Tables II and V. Modeled data replicates the characteristics of empirical data accurately as the largest error is 0.003 for the case of 250 μ s with 16 DUs. The model captures the transition from negligible lates to a significant R_L at 125 μ s going from 2 to 4 DUs and at 250 μ s going from 8 to 16 DUs.

Outage results also match well between empirical (Table III) and simulated data (Table VI). Unlike with the SE loss metric, there is, however, one large divergence, namely the maximum outage duration for 125 μ s. The corresponding simulated value is approximately 20 times smaller. Total outage rate is similar. It thus appears likely that the model fails to capture the possibility of rare extreme deviations from the average performance present in the testbed. Similar very large maximum outages are observed for $N_{DU} = 8, 16$. This suggests that, at larger N_{DU} and shorter TTIs, the testbed suffers from slow recovery from deadline misses.

TABLE V

ESTIMATED RELATIVE SE CALCULATED FROM MARKOV CHAIN STEADY STATES PER TTI DURATION FOR EACH N_{DU} .

	125 μ s	250 μ s	500 μ s	1000 μ s
1	0.9999658	0.9999786	0.99999880	1.00000000
2	0.9999648	0.9999789	0.99999807	1.00000000
4	0.7450000	0.9999735	0.99998930	0.99999965
8	0.3580000	0.9812000	0.99998120	0.99999237
16	0.1540000	0.4650000	0.99982100	0.99999013

Fig. 5. Shortest TTI duration supported for various R_L per N_{DU} .

Longer TTI durations produce longer average outage durations but fewer instances. More frequent interruptions cause otherwise unnecessary retransmissions to occur and preclude the use of services requiring greater reliability. Longer average service interruptions can complicate the execution of handovers as one of the DUs will only get the CU's instructions for the UE after several retries. It should be noted that the trade-off between average duration and occurrence count is complicated by the fact that the shorter TTIs enable faster link adaptation to channel conditions. In an actual system, the longer TTIs may still result in more interruptions of service.

Figure 5 presents the minimum TTI duration required to obtain an R_L equal or lower than a given target as a function of the number of DUs. Behavior differs for 10^{-1} and 10^{-6} . Unlike other rates, 10^{-1} shows no upward trend. This reflects the fact that most latencies are low but the rest exhibits a long tail. By setting the tolerable R_L at 10^{-1} , the effects of the entire tail will be ignored. For the 10^{-6} rate, the minimum TTI duration is noticeably higher than for other rates. Furthermore, the relationship between the number of DUs and the TTI duration behaves less linearly. One possible explanation for this lies in interference from some other process on the testbed.

At lower target R_L , TTIs must be longer to absorb the extra delay caused. This suggests the need to over-dimension TTIs when employing soft-real-time platforms and targeting higher reliability.

Fitting a curve to the minimum TTI data allows for projecting beyond the recorded data in Figure 5. As an example, (2) is fit for $R_L = 10^{-2}, 10^{-5}, 10^{-6}$. These were chosen because $R_L = 10^{-6}$ has the highest minimum TTI duration overall while $R_L = 10^{-2}, 10^{-5}$ represent the lowest and highest cases for the middle group of curves in Figure 5. The case of

TABLE VI

SIMULATED OUTAGE DATA FOR FOUR DUS. AVERAGE OVER FOUR DUS.

	125 μ s	250 μ s	500 μ s	1000 μ s
Average	161.25 μ s	555 μ s	690 μ s	1000 μ s
Maximum	1500 μ s	1250 μ s	2500 μ s	1000 μ s
Rate	0.256	2.22e-05	7.3e-06	2.6e-07

Fig. 6. Maximum N_{DU} for $R_L = 10^{-2}, 10^{-5}, 10^{-6}$.

TABLE VII
 C_i FOR $R_L = 10^{-2}, 10^{-5}, 10^{-6}$.

	C_1	C_2
10^{-2}	24.91	34.58
10^{-5}	29.44	49.08
10^{-6}	42.67	360.60

$R_L = 10^{-1}$ is not considered because for the values of N_{DU} tested, no dependency on N_{DU} can be observed. Consequently, fitting a predictor function to the data would indicate the ability to maintain the same performance for an infinite number of DUs. This can clearly not be true. Fitting the model of (2) yields the values of C_i in Table VII. These coefficients are then used to estimate the scaling of midhaul performance in terms of N_{DU} .

Figure 6 presents the maximum supported N_{DU} for each considered TTI duration and R_L . Maximum N_{DU} for rates 10^{-2} and 10^{-5} scale faster than the TTI length. An 8-fold increase from 125 μ s to 1000 μ s results in an approximately 12-fold increase in maximum N_{DU} . The margin for extra deadline misses at a rate of 10^{-5} thus grows faster than the added variance from the additional DUs. Scaling for 10^{-6} behaves differently as the testbed is unable to support TTIs of 125 μ s and 250 μ s. This reflects the fact that the shortest TTI length suffering less than 10^{-6} deadline misses is longer than 250 μ s. The value of N_{DU} for 500 μ s is similar to the one that rates 10^{-2} and 10^{-5} experience at 125 μ s. The curve for 10^{-6} is shifted compared to the others as longer TTIs are required to provide enough buffer to reach the lower rate.

VI. CONCLUSION

Use of a midhaul link induces latency and variability to RAN functionality. This impacts communication performance and must be taken into account. A model was introduced to describe midhaul latency characteristics. Empirical data obtained using a commodity hardware GPOS testbed was used to assess the ability of the model to capture latency behavior. The testbed used provided midhaul latency performance sufficient to support 250 μ s at up to 8 DUs and 500 μ s at up to 16 DUs. While average performance was good, large

deviations were observed due to the soft-real-time nature of the implementation. The impact of these relatively rare large delays is minor so long as the average case remains below the target delay. Exceeding the capability of the platform results in significant degradation to SE and outage probability. The impact of the latency distribution's tail becomes more pronounced when targeting higher reliability levels. Modeling also enables assessment of scaling. A trade-off exists between using shorter TTIs offering more granular adaption and longer TTIs with lower probability of deadline misses. A similar trade-off exist at network level where potential efficiency gains increase as a single CU controls more DUs but the shortest TTI duration supported also increases.

REFERENCES

- [1] A. Ghosh, A. Maeder, M. Baker, and D. Chandramouli, "5G evolution: A view on 5G cellular technology beyond 3GPP Release 15," *IEEE Access*, vol. 7, pp. 127 639–127 651, 2019.
- [2] 3GPP, "Service requirements for the 5G system 15.8.0," 2019.
- [3] S. Manap, K. Dimiyati, M. N. Hindia, M. S. Abu Talib, and R. Tafazolli, "Survey of radio resource management in 5G heterogeneous networks," *IEEE Access*, vol. 8, pp. 131 202–131 223, 2020.
- [4] P. Rost, I. Berberana, 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 Commun. Mag.*, vol. 53, no. 12, pp. 75–82, Dec. 2015.
- [5] E. J. Kitindi, S. Fu, Y. Jia, A. Kabir, and Y. Wang, "Wireless network virtualization with SDN and C-RAN for 5G networks: Requirements, opportunities, and challenges," *IEEE Access*, vol. 5, pp. 19 099–19 115, 2017.
- [6] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Neww.*, pp. 1–9, 2019.
- [7] Y. Choi and N. Park, "Support for edge computing in the 5G network," in *Int. Conf. on Ubiquitous and Future Networks (ICUFN)*, 2018, pp. 592–596.
- [8] I. C. Bertolotti, "Real-time operating systems tutorial," in *IEEE Int. Symp. on Ind. Electron.*, 2010, pp. 4023–4120.
- [9] C. Maiza, R. Rihani, J. M. Rivas, J. Goossens, S. Altmeyer, and R. I. Davis, "A survey of timing verification techniques for multi-core real-time systems," *ACM Comput. Surv.*, Jun. 2019.
- [10] F. Reghenzani, G. Massari, and W. Fornaciari, "The real-time Linux kernel: A survey on PREEMPT_RT," *ACM Comput. Surv.*, vol. 52, no. 1, Feb. 2019.
- [11] L. M. P. Larsen, A. Checko, and H. L. Christiansen, "A survey of the functional splits proposed for 5G mobile crosshaul networks," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 146–172, 2019.
- [12] 3GPP, "Study on new radio access technology: Radio access architecture and interfaces (Release 14)," 2017.
- [13] I. A. Alimi, A. L. Teixeira, and P. P. Monteiro, "Toward an efficient C-RAN optical fronthaul for the future networks: A tutorial on technologies, requirements, challenges, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 708–769, 2018.
- [14] 3GPP, "NR and NG-RAN Overall Description 15.11.0," 2020.
- [15] IEEE, "IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008)*, pp. 1–499, 2020.
- [16] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020. [Online]. Available: <https://www.R-project.org/>
- [17] G. A. Spedicato, "Discrete time markov chains with R," *The R Journal*, 2017, R package version 0.8.5-3. [Online]. Available: <https://journal.r-project.org/archive/2017/RJ-2017-036/index.html>