Fine-Grain Slicing of Edge Servers for Radio Workloads

EmadElDin A. Mazied, Dimitrios S. Nikolopoulos, Scott F. Midkiff, and Kirk W. Cameron
Department of Computer Science, Virginia Tech
Blacksburg, VA 24061, USA
{emazied,dsn,midkiff,cameron}@vt.edu

Abstract

The next generation network employs virtualization to create on-demand virtual networks called network slices, ensuring different service categories meet their quality of service requirements. This, along with offloading schemes, requires small-scale data centers near radio base stations known as edge computing. These edge systems must effectively autoscale computing servers to handle fluctuating workloads. Radio access network (RAN) slicing involves containerized radio applications with compute-intensive functions at the physical and radio link control layers, imposing strict processing time requirements. To address delay-sensitive radio workloads, we propose a fine-grained autoscaling solution in the edge cloud. Our analytical approach dynamically tunes resource limits using stochastic decision processes to meet dynamic demands. Evaluations compare user-configured and soft-tuned resource limits, utilizing the Roofline model and Low Density Parity Check (LDPC) decoding algorithm for workload characterization. We establish processing time design constraints and discuss limitations and future research.

CCS Concepts: • **Networks** → **Network design principles; Network slicing**; • **Computer systems organization** → **Cloud computing; Edge computing**; • **Theory of computation** → *Integer programming; Stochastic optimization.*

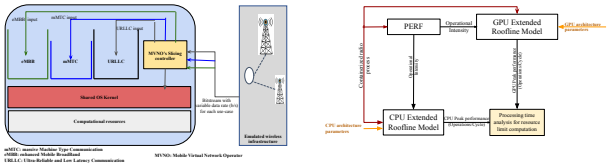
Keywords: Radio access network slicing, kubernetes-like systems, resource allocation, chance-constrained programming

ACM Reference Format:
EmadElDin A. Mazied, Dimitrios S. Nikolopoulos, Scott F. Midkiff, and Kirk W. Cameron. . Fine-Grain Slicing of Edge Servers for Radio Workloads. In *Proceedings of (HotInfra 2023).* ACM, New York, NY, USA, 3 pages.

1 Introduction

Cloud computing has emerged as a computing warehouse for diverse workloads. Unlike core cloud systems, edge computing addresses the need for offloading wireless traffic to servers situated at the edge of the radio access network (RAN). Edge servers process wireless data and signals in close proximity to radio towers. Additionally, next-generation wireless networks leverages virtualization to create on-demand virtual networks called network slices, tailored to specific wireless service categories with distinct quality of service requirements. To ensure the isolation for RAN slicing, the edge cloud processes virtualized radio functions within containers. Furthermore, autoscaling computational resources is a fundamental concept in cloud computing, allowing for dynamic allocation and release of resources based on workload fluctuations. However, existing autoscaling frameworks in Kubernetes-like systems have not adequately accounted for the characteristics of recent edge applications, particularly those with strict processing time requirements such as RAN slicing workloads. Current autoscaling approaches involve launching new containers when the application size exceeds predefined resource limits, leading to the scaling down of other containers operating below their configured limits. However, these methods are unsuitable for RAN slicing containers due to their narrow processing time requirements (0.2-25 milliseconds) and the significant time penalties incurred during container startup and scaling down operations (0.5-5 seconds). To circumvent this challenge, it has been suggested by previous studies [3, 5, 8] to remove resource limits from the container's configuration file. However, this approach proves impractical in resource-constrained edge computing environments with diverse computational service demands. In this paper, we propose a fine-grain autoscaling framework for RAN slicing workloads by dynamically adjusting containers' resource limits based on fluctuations in their radio workloads. We introduce an analytical framework that utilizes stochastic decision processes to tune resource limits, considering the stochastic characteristics of the wireless environment and the time-intensive Low Density Parity Check (LDPC) decoding radio function [1, 2]. We outline the methods used to prototype our proposed framework, present preliminary results utilizing the Roofline model and asymptotic analysis of the LDPC decoding algorithm to establish processing time design constraints. Additionally, we discuss the limitations of our model and provide insights into future improvements. Finally, we outline the next milestones and future research avenues in this domain.

HotInfra 2023, June 18, 2023, Orlando, FL



(a) System setup (b) Processing time estimation

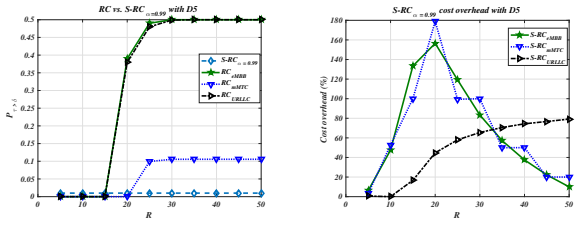
Figure 1. Fine-grain autoscaling framework for radio slices.

2 Methodology

The radio workload experiences rapid fluctuations as it processes data transmitted over time-varying stochastic wireless channels. Despite its unpredictable nature, successful wireless system deployment relies on harnessing the stochastic properties of wireless data. Therefore, we leverage these stochastic properties to dynamically scale edge cloud resources for RAN slicing workloads. Our system design, illustrated in Figure 1(a), encompasses three key components: (i) a wireless traffic generator; (ii) radio containers; and (iii) edge servers. We generate emulated wireless traffic to simulate real-world conditions and deploy radio containers at the edge, accompanied by a RAN slicing controller that utilizes decision processes to fine-tune the containers' resource limits.

To gain insights into the proposed system design, we develop an analytical model for resource control. We choose the LDPC decoding algorithm[1] as a representative case study for the radio process executed within each container due to its significant contribution to the execution time of radio tasks, accounting for approximately 60% of the uplink execution time [2]. We measure the LDPC's operational intensity by utilizing the LDPC asymptotic analysis presented in [6]. Moreover, to reflect the stochastic properties of the radio workload, we model the fluctuations in the LDPC as a Gaussian random number ($\tilde{R}$), which represents the number of iterations in the LDPC algorithm required to detect and correct the received messages transmitted over an Additive White Gaussian wireless channel. Furthermore, we adopt the Roofline model [10] to address peak performance considerations in the computing architecture, thereby encoding processing time through the definition of workload operational intensity and architecture peak performance, i.e., processing time $\tilde{\tau} = \frac{\tilde{R}.LDPC_{operations}}{l.min(\pi,\beta A)}$ where l is the resource limit variable that we need to determine, π is the processor peak performance, β is the memory bandwidth, and A is the LDPC's operational intensity. By integrating these elements, our aim is to fine-tune the resource limits of containers for RAN slicing workloads, e.g., LDPC decoding algorithm. In this context, the problem is formulated as stochastic integer optimization problem where we need to determine the optimum resource limit l_i of each container $i \in \mathcal{I}$ to minimize the total cost c_{ij} of running containers $i \in \mathcal{I}$ that utilize resources $j \in \mathcal{J}$ and guarantee the delay below predefined threshold δ_i. To determine l_i, we define a binary decision variable x_{ij} that equals 1 when a resource j is assigned to a container i and equals 0 otherwise. Thus, $l_i = \sum_{j \in \mathcal{J}} x_{ij} \; \forall i \in \mathcal{I}$. Accordingly, we aim to minimize $\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} x_{ij} c_{ij}$, such that processing delay τ_i is below threshold $\delta_i \; \forall i \in \mathcal{I}$ and total resource limits below the available resources $\sum_{i \in \mathcal{I}} l_i <= J$.

[1] LDPC is multi-threaded, with parallel tasks running on multiple cores.



(a) Probability of processing time violation (b) Cost overhead of S-RC

Figure 2. S-RC performance with dense input data size D_5

3 Preliminary Results

We approach the proposed problem using chance-constrained programming [4]. We conducted approximately 50 simulations to numerically evaluate the proposed optimization problem, referred to as stochastic resource control (S-RC). We compared S-RC with RC, where we hard-code the resource limit by solving the deterministic instance of the proposed stochastic formulation. We observed that the cost-overhead of S-RC is slightly higher than RC, as shown in Figure 2(b), although it satisfies the probabilistic constraint for the processing delay requirement, as depicted in Figure 2(a). The results, illustrated in Figure 2, indicate that the deployment of S-RC has a high likelihood of effectively auto-scaling the edge cloud for RAN slicing workloads. However, its deployment cost is high due to the use of the branch and cut algorithm [7, 11], which guarantees an optimal solution for the proposed S-RC.

4 Conclusion and Future Work

This research proposes a soft-tuning approach for resource limits of radio containers at the edge cloud to handle stochastic variations in RAN slicing workloads while meeting processing time constraints. The stochastic resource controller (S-RC) outperforms the legacy controller (RC) but incurs a notable overhead cost. Ongoing research focuses on designing an approximation algorithm to reduce deployment costs and studying the impact of wireless resource allocation on S-RC performance. Further development includes an online (reactive) S-RC algorithm to enhance the adaptability and responsiveness of the resource control mechanism. Central to this development is optimizing the scaling period, a crucial factor in ensuring the success of the online algorithm. Prototyping and validation will be conducted using a wireless data traffic generator in an emulated environment, such as the Colosseum testbed [9]. This research advances resource management in edge computing for efficient RAN slicing workloads.

References

[1] Jung Hyun Bae, Ahmed Abotabl, Hsien-Ping Lin, Kee-Bong Song, and Jungwon Lee. 2019. An overview of channel coding for 5G NR cellular communications. *APSIPA Transactions on Signal and Information Processing* 8 (2019).

[2] Xenofon Foukas and Bozidar Radunovic. 2021. Concordia: Teaching the 5G vRAN to share compute. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 580–596.

[3] Dina Henderson. 2022. *Kubernetes CPU Throttling: The Silent Killer of Response Time – and What to Do About It.* IBM. https://community.ibm.com/community/user/aiops/blogs/dina-henderson/2022/06/29/kubernetes-cpu-throttling-the-silent-killer-of-res

[4] P. Kall and S. W. Wallace. 1994. *Stochastic Programming*. John Wiley and Sons.

[5] Eric Khun. 2020. *Kubernetes: Make your services faster by removing CPU limits*. BUFFER. https://erickhun.com/posts/kubernetes-faster-services-no-cpu-limits/

[6] Federico Maguolo and Alessandra Mior. 2008. Analysis of complexity for the message passing algorithm. In *2008 16th International Conference on Software, Telecommunications and Computer Networks*. 295–299.

[7] John E Mitchell. 2002. Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization* 1, 1 (2002), 65–77.

[8] F. Musthafa. 2020. *CPU limits and aggressive throttling in Kubernetes*. OMIO Engineering. https://medium.com/omio-engineering/cpu-limits-and-aggressive-throttling-in-kubernetes-c5b20bd8a718

[9] NSF-PAWR. 2022. *COLOSSEUM Tetbed*. PAWR. https://colosseumneu.freshdesk.com/support/solutions

[10] Samuel Williams, Andrew Waterman, and David Patterson. 2009. Roofline: an insightful visual performance model for multicore architectures. *Commun. ACM* 52, 4 (2009), 65–76.

[11] Laurence A Wolsey. 2020. *Integer programming*. John Wiley & Sons.