Introducing Best-in-Class Service Level Agreement for Time-Sensitive Edge Computing

Subhadeep Sahoo(1), Sébastien Bigo(2), Nihel Benzaoui(2)

(1) University of California, Davis, CA, USA, 95616, subsahoo@ucdavis.edu.
(2) Nokia Bell Labs, 91620 Nozay, France.

Abstract To provide edge computing with pre-negotiated, guaranteed time budget, we propose an original joint optimization of compute and network resource allocations. For 20% time-sensitive applications, we show that we could guarantee 200-times smaller service latencies while reducing DC utilization efficiency by no more than 30%.

Introduction

5G Edge Cloud is an aggregate of small data centers (DC) located close to the edge to host time-sensitive applications. Part of these applications, e.g., those relying on data analytics, are compute hungry and require parallel processing, conventionally based on a partition-aggregate[1] workflow. Application requests are partitioned into smaller jobs to be executed simultaneously over a group of compute units (CU). The output data are then sent to a second group of CUs, where they get aggregated to deliver the final output. In this process, the workflow completion time is largely influenced by the delay for exchanging data over the network and possible buffering time at busy CUs.

Several solutions have been proposed to decrease workflow completion time through network[2][3] or compute resource allocation[4] – [6]. In [7], we proposed *Deterministic Dynamic Network (DDN)-based just-in-time-delivery*, a joint optimization of network and compute resource allocation. All these solutions aim at minimizing the total completion time for all workflows and maximize the edge-DC utilization efficiency. But none have addressed the challenge of providing pre-negotiated, guaranteed completion time.

In this paper, we propose to shift the paradigm of optimizing edge-DC utilization efficiency into offering best-in-class services for time-sensitive applications with predefined service level agreement (i.e. time budget), by enhancing the aforementioned DDN-based just-in-time approach with latency-awareness. We evaluate the effectiveness of the latency-aware approach to deliver best-in-class SLA. Naturally, guaranteeing service level agreement (SLA) for time-sensitive applications comes with a cost on the completion time for non-time sensitive workflows and on the edge-DC utilization efficiency, which we also assess.

Latency-aware just-in-time delivery

In a DDN network[8] (Fig. 1), bandwidth, latency (sub-ms) and jitter (sub-µs) are guaranteed through slot reservation. Slots of a few µs are allocated by a central scheduler in a dynamic fashion (sub-ms)[9]. Once slots are allocated for a given dataflow, no further switching is applied to that flow and its delivery delay is fixed, allowing for an accurate prediction of network latency at any time, on any path in the edge-DC. In [7], we proposed to leverage this predictable network latency to deliver dataflows just-in-time[10] to be processed by free compute resources. But this approach did not guarantee any SLA for the processed applications. Here, we propose to use the deterministic nature of DDN not only to reduce the total completion time of a group of workflows, but also to meet pre-agreed completion times per workflow for premium time sensitive applications. The new latency-aware approach works as follows:

1) We differentiate between the time-sensitive applications and the regular applications, by feeding them to two different queues; we prioritize bandwidth (network resources) and compute resources for time-sensitive requests.

2) For all workflows, we select the CUs performing the partition and aggregate phases, such that the load is balanced over the network.

3.a) For each non-time-sensitive workflow, we pre-calculate the delivery delay of all flows generated during the aggregate phase. The highest delivery delay among them is considered as the threshold delivery delay ($T_{threshold}$). Then the bandwidths of all the flows from this particular workflow are decreased such that their delivery delays all match $T_{threshold}$. The released slots are

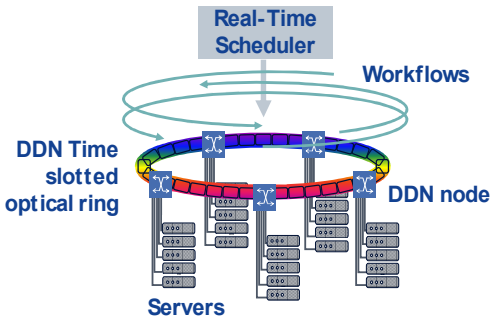

Fig.1: DDN-based edge data center (DC)

978-1-6654-3868-1/21/$31.00 ©2021 IEEE

used to allow for the simultaneous execution of other workflows in order to reduce the total completion time for the whole group of workflows.

3.b) For each time-sensitive workflow where a latency requirement (T_r) must be fulfilled, we set the threshold delivery delay as the required latency ($T_{threshold} \leq T_r$). Then for all flows generated during the aggregate phase of this particular workflow, we tune the bandwidth (B_r) by reserving as many slots as needed in the cyclic scheduling window, in order to obtain a network delivery latency smaller than the latency requirement T_r. Let's assume that T_s denotes the duration of each slot and that R is the capacity of the link. Therefore, each slot contains $B_r = R/T_s$ unit of bandwidth. To satisfy the requirement of the dataflow, we need to reserve N slots in a window period (W), given by $N = max\left(\frac{T_s * W}{T_r}, \frac{B_r}{B_s}\right)$. The B_s is guaranteed by $N * T_s$ bandwidth unit.

4) We map the flows over the CUs such that the expected delivery delay of each flow matches the remaining time before its target CU becomes available.

Evaluation

In the following simulations, we evaluate the effectiveness of the latency-aware approach. Our metric is the processing Completion Time (CT) per workflow. We also evaluate the impact of delivering such performance for a group of workflows on the remaining ones. Since the DDN-based approach tries to increase processing parallelization to reach the lowest total completion time for a group of workflows, it provides the allocation corresponding to the best possible utilization of resources. Hence, we consider the DDN-based just-in-time (JiT) approach as the reference for the ultimate performance that can be provided by the edge-DC for a given set of workflow requests and reference it under Optimal-JiT. Next, we compare the new latency-aware approach with Optimal-JiT . We also position the new approach w.r.t a baseline approach aiming at delivering data as-fast-as-possible (AFAP), for each flow – AFAP grabs the whole available bandwidth (slots) on the network path on a first-arrived, first-served basis. For benchmarking the three approaches, we use the Total Completion Time (Total-CT) for a group of workflows (time taken from the processing of the first to the last workflow). This metric is a good indicator of the completion time per workflow, and how many workflows are processed in parallel.

Our simulation environment is a DDN-based edge-DC (Fig.1), where 10 Gb/s servers (CUs) are connected using a 10-node DDN ring with four 10 Gb/s channels. For a fair comparison, we use the same topology for the latency-aware, Optimal-JiT and AFAP approaches. We generate workflows with an input data, randomly distributed between 2 to 5 GB. In the aggregate phase, we cap the data rate of all exchanged dataflows to 1 Gb/s.

We evaluate the cost of delivering a high SLA for time-sensitive applications on processing efficiency of the edge-DC. We consider a group of 500 workflows, where 50% are time-sensitive with an SLA, running on a 10 CUs edge-DC. Fig. 2 compares the Total-CT for all 500 workflows with varying SLAs of 100 ms, 10 ms, 100 µs and 10 µs latency, using the latency-aware, Optimal-JiT and AFAP approaches. Insets in Fig. 2 report the Completion Time Probability Density Functions (PDF) for time-sensitive workflows for each case. They show that using the latency-aware approach, we managed to obtain a significantly narrower Completion Time PDF, and strictly upper bounded by the pre-agreed latency. In Fig. 2 we report the evolution of the Total-CT when tightening the latency requirement. Compared to the Optimal-JiT approach, the latency-aware approach shows in all cases a higher Total-CT, but only by 10 to 20% for realistic latency requirements (≤ 100 µs). We find that for SLAs lower than the limit of 10 ms, the penalties affecting the coexisting non-time sensitive applications grow from +7% to +55% on Total-CT, as the pre-agreed latencies shorten. Penalties are also found to increase for large

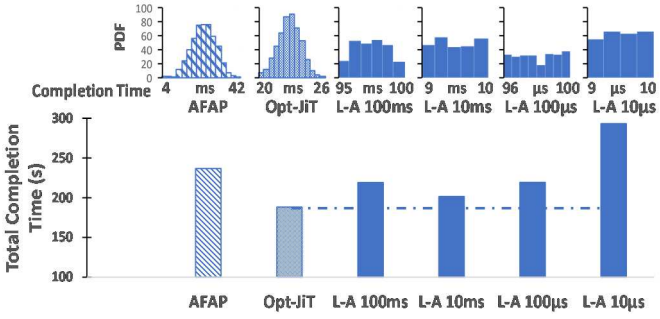

Fig. 2. Total-CT comparison using AFAP, Optimal-JiT and latency-aware (L-A) approaches. Insets: distributions of CT for time-sensitive workflows.

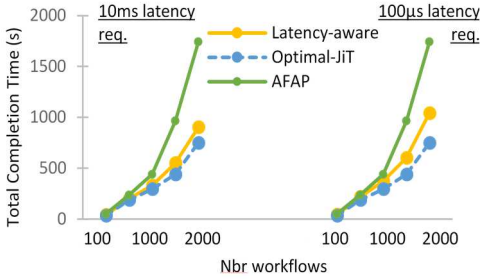

Fig. 3. Evolution of Total-CT with load, for 10m and 100µs pre-agreed latency requirements.

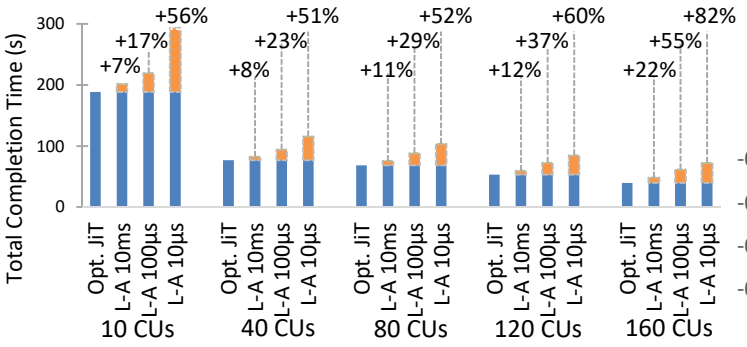

Fig. 4. Total-CT increase with number of CUs, using the latency-aware approach w.r.t to the Optimal-JiT one.

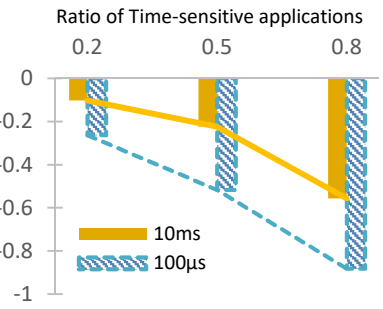

Fig. 5. Penalty on Total-CT for 2000 workflows and 80CUs scenario

SLAs, e.g. 100 ms. This is explained by the fact that we are forcing a latency constraint higher than what we could obtain without differentiating SLA across workflows, i.e., when using the Optimal-JiT approach, at ~23 ms according to the inset of Fig. 2. This value could be used as a threshold to define when it is better for the operator to let the network run free, without forcing an SLA. It can also be observed that the latency-aware approach performs better than the AFAP approach, when the SLA exceeds 100 µs. This is because the latency-aware approach maximizes parallel workflow processing, leading to a shorter Total-CT than AFAP in most situations.

To assess the dependence of the penalty on Total-CT with load, we vary the number of workflows and report the Total-CT for all three approaches for medium latency budgets (10 ms and 100 µs). Fig. 3 shows that the penalty increases super linearly with the number of workflows. When the edge-DC load increases, workflow parallelization becomes harder, leading to an increase of the per-workflow Completion Time and consequently of the Total-CT. Note that despite this penalty increase, Total-CT using the latency-aware approach is always well below Total-CT with AFAP.

To investigate the relationship between edge-DC computing capacity (processing parallelization capability), workflow load, and the penalty on the Total-CT, we now set the number of workflows to 500, and vary the number of CUs (from 40 to 160 with a step of 40) as well as the latency agreements (100 µs and 10 µs). Fig. 4 reports the Total-CT for the Optimal-JiT and latency-aware approaches. As expected, whatever the number of CUs, the penalty grows with the latency requirement. But counter-intuitively, increasing the number of CUs leads to a penalty increase. This means that the cost of delivering high SLA for time-sensitive applications increases with the size of the edge-DC. In this case, adopting alternative strategies, such as isolating the processing of time-sensitive workflows in a confined part of the edge-DC, should be considered.

To investigate the trade-off between the ratio of time-sensitive applications and the degradation of performance for remaining applications, we assess the Total-CT when varying the ratio of time-sensitive versus regular applications. We start from a ratio of 20% which we consider realistic and stretch it to 80%. Fig. 5 reports the Total-CT penalty of the latency-aware approach over the Optimal-JiT one, for a high load (2000 workflows) and a large edge-DC (80 CUs). Compared to the 50% time-sensitive application reference use-case, the 80% scenario shows a high impact on the Total-CT; almost doubling it. Shifting the cursor to a ratio of 20% we observe a low penalty never exceeding 30% while reducing completion time by a factor of 200 (23 ms obtained using the Optimal-JiT approach against 100 µs guaranteed with the latency-aware approach). Overall, the edge DC operator will have to set the cursor of time sensitive applications with respect to non-time-sensitive applications.

Conclusion

For edge computing, where applications can have strict latency constraints, we propose a latency-aware joint optimization of compute and network resource allocation to ensure a per-application computing time within a latency budget. We show that the cost of delivering such SLAs has a complex, sometimes counter-intuitive, relationship with the edge-DC load and size. We also show that knowing the baseline performance of the edge-DC is important, since forcing a higher SLA can be counter-productive. But overall, we showed that the proposed latency-aware approach open business opportunities to offer best-in-class service level agreement for time-sensitive applications, several orders of magnitude faster than over an optimally-configured edge DC, with minimum impact on the edge-DC efficiency.

References

[1] R. Nirmalan and K. Gokulakrishnan, “Survey on Map Reduce Scheduling Algorithms in Hadoop Heterogeneous Environments”, in Proc. 3rd International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, Nov. 2018.

[2] A. Chatzieleftheriou, S. Legtchenko, H. Williams, and Antony Rowstron, “Larry: Practical Network Reconfigurability in the Data Center,” In Proc. of the 15th USENIX Conference on Networked Systems Design and Implementation (NSDI'18), pp. 141-156, Apr. 2018.

[3] J. Perry, A. Ousterhout, H. BalakrishnanD. Shah and H. Fugal, “Fastpass: A Centralized "Zero-Queue" Datacenter Network,” SIGCOMM Comput. Commun. Rev, vol. 44, no. 4, pp. 307-318, Oct. 2014.

[4] N.- Kr. Sharma, C. Zhao, M. Liu, P.-G. Kannan, C. Kim, A. Krishnamurthy and A. Sivaraman, “Programmable Calendar Queues for High-speed Packet Scheduling,” In Proc. of the 17th USENIX Symposium on Networked Systems Design and Implementation, pp. 685-699, Feb. 2020.

[5] T. Levai, F. Nemeth, B. Raghavan and G. Retvari, “Batchy: Batchscheduling Data Flow Graphs with Service-level Objectives,” In Proc. of the 17th USENIX Symposium on Networked Systems Design and Implementation, pp. 633-649, Feb. 2020.

[6] Ks. Mahajan, A. Balasubramanian, A. Singhvi, S. Venkataraman, A. Akella, A. Phanishayee and Shuchi Chawla, “Themis: Fair and Efficient GPU Cluster Scheduling,” In Proc. of the 17th USENIX Symposium on Networked Systems Design and Implementation, pp. 289-304, Feb. 2020.

[7] S. Sahoo, N. -H. Bao, S. Bigo and N. Benzaoui, "Deterministic Dynamic Network-Based Just-in-Time Delivery for Distributed Edge Computing," 2020 European Conference on Optical Communications (ECOC), 2020, pp. 1-4.

[8] N. Benzaoui et al., “Deterministic Dynamic Networks (DDN)”, Journal of Lightwave Technology, vol. 37, no. 14, pp. 3465–3474, May. 2019.

[9] M. Szczerban et al., “Real-time Control and Management Plane for Edge-Cloud Deterministic and Dynamic Networks,” IEEE/OSA Journal of Optical Communication and Networking, vol. 12, no. 11, pp. 312–323, Aug. 2020.

[10] A. Ousterhout, A. Belay, and I. Zhang, “Just In Time Delivery: Leveraging Operating Systems Knowledge for Better Datacenter Congestion Control”, in Proc. 11th USENIX Conference on Hot Topics in Cloud Computing, Jul. 2019.