# Cover Letter

This is an extended version of our conference paper " Locality-aware Load-Balancing For Serverless Clusters" that appeared at ACM HPDC 2022. The preliminary conference version introduced a new locality-aware load-balancing algorithm for FaaS (Functions as a Service). **We make a significant *conceptual* extension to this problem, and introduce the notion of priorities and quality of service for function execution.** Compared to the conference version, we have made significant new contributions to algorithms, system design and implementation, and experimental evaluation. A detailed list of major contributions and experimental results is presented below.

Below, we provide a list of **major** new additions and extensions to the conference paper,

1. We perform a detailed analysis of real-world function workload traces and applications. We empicially analyze scenarios in which functions can be delayed. This is a major conceptual contribution and **extension of the problem-space.** We find that function prioritization can also help alleviate burstiness, which we show to be a major characteristic and challenge of function workloads.
2. **We develop a *new* algorithm** which considers different function priorities: k-CH-RLU. It uses some of the concepts from our previous algorithm CH-RLU, but under a completely new architecture (with partitioned cluster pools), and an overall new algorithm.
3. We conduct **extensive empirical evaluation of our new algorithm** by implementing it in OpenWhisk. We show several important new results:
    (a) Our new algorithm can provide service differentiation.
    (b) Introducing priorities can improve performance of both high and low-priority functions, when compared to the previous priority-agnostic approach.
    (c) Our approach can also help reduce the number of servers required and make FaaS more resource-efficient.

---

These major additions have resulted in the following additions to *all* sections of the paper:

1. Introduction: Rewritten to reflect the new problem of priority-aware load-balancing.
2. Background: Minor edits.
3. Challenges. New subsection 3.2 on the burstiness of function invocations. New figure (4) which analyzes the Azure function trace.
4. Function prioritization motivation (entirely new section). We describe which functions can be delayed and deprioritized, under what scenarios, and what impact that can have on serverless clusters.
5. Load and priority-based consistent hashing algorithm. We present the k-CH-RLU algorithm, which generalizes and extends our previous CH-RLU algorithm. The new algorithm is priority aware, and handles bursty workloads, locality, and server overloads is completely new ways. The architecture is different. Algorithm 2 is new. Section 5.3 is completely new. Figure 6 is new.

6. Implementation. We have shown how k-CH-RLU can be implemented, and made revisions to the earlier implementation, which was restricted to non-priority-aware execution.
7. Evaluation. We have a new subsection 7.3, which focuses on multi-pool (priority-aware) performance. This has three new graphs (Figures 12, 13, 14).
8. Related Work. We have added new sub-headings on differentiated services, and updated the related work to include the recent relevant papers on FaaS load-balancing and resource management.
9. Conclusion. Rewritten to reflect the importance of priorities.