Scheduling in the Presence of Data Intensive Compute Jobs

Amir Behrouzi-Far and Emina Soljanin
Department of Electrical and Computer Engineering, Rutgers University
Piscataway, New Jersey 08904, USA
{amir.behrouzifar,emina.soljanin}@rutgers.edu

Scheduling has been of interest since the emergence of computing systems. Although scheduling problems were studied for a wide variety of systems, most studies consider compute jobs with deterministic or exponentially distributed or heavy-tail distributed service times. In today's computer systems, a fraction of jobs are data intensive, and their the service time is on average several order of magnitude longer than that of the regular jobs in the system. Thus previously known scheduling results are not directly applicable in modern environments.

In this work, we studied a queuing system where most of the arriving jobs have (on average and relatively) modest service requirements, and there are sporadic job arrivals whose service is (on average and comparatively) very slow. We assume that jobs get scheduled upon arrival and the service requirement of jobs are not known upon arrival. In order to prevent fast jobs from being penalized in the queues with slow jobs, each job gets redundantly submitted to a few servers upon arrival. Once the first copy of a job started the service the other copies get cancelled. This way redundancy brings no extra cost to the system and it only helps a job to find shorter queues.

We compared several job scheduling policies in systems with redundancy for the job service model described above. Our first observation is that the performance of the scheduling policies is fundamentally different in systems with sporadic data intensive jobs, compared to their performance with classical service model. In particular, round-robin policy, which is known to be efficient for providing load balancing, performs worse than the random assignment policy. There is a subtle intuition behind this phenomenon. In classical job service time models, the load at each server could be measured approximately by the number of jobs queued at the server. With this measure of load, round-robin policy provides the best average load balancing. However, with sporadic data intensive compute jobs, the number of jobs is not a good measure of the actual load on the servers. For example, a server with only one data intensive job could have more compute load than a server with hundreds of fast jobs. In the case of systems with redundancy, if all the redundant copies of a job get queued behind the copies of a data intensive job then redundancy brings no benefit to performance. The round-robin policy increases the chance of this phenomenon [1].

This observation motivated us to think about new scheduling policies that perform better in the presence of sporadic data intensive jobs. We proposed a scheduling policy, based on combinatorial block designs, which chooses the server for consequent jobs in a way that the number of *overlapping* servers between consequent jobs is minimized. Among all, we focus on Balanced and Incomplete Block Designs (BIBD), and call the related policy *BIBD scheduling* policy.

BIBD scheduling policy is suitable for systems with sporadic data intensive compute jobs because in such systems, it is important to minimize the probability that copies of a fast job get queued behind the copies of a data intensive job on all servers. Round-robin scheduling can reduce this probability by choosing the groups of servers that it assigns to the arriving jobs in a cyclic fashion. BIBD policy, on the other hand, chooses the groups of servers in the best possible way by minimizing the number of overlaps a given group of servers have with the rest of the groups. Specifically, with $(\nu, \lambda, 1)$-BIBD policy, jobs get assigned to server groups of size λ, out of ν available servers, in a way that every pairs of group overlap only in one server.

To analyse the performance of scheduling policies, we developed an analogy to the classical Urns&Balls problem and derived performance indicators in the analogues model, which we believe are good predictors of scheduling policies' performance in the queuing system. Then, we implemented a queuing system and run extensive simulations to observe the performance of the scheduling policies. For the simulated scenarios, our performance indicators were indeed good predictors of the relative performance of the scheduling policies in the queuing system. Our performance indicators ranked the scheduling policies for systems with data intensive compute jobs as follows:

1) **BIBD Based** policy maximizes load balancing and diversity in redundancy, by cyclic assignment of jobs to the group of servers with minimum overlap.

2) **Random Assignment** policy maximizes the diversity of redundancy, but has inferior load balancing performance.

3) **Round-Robin** policy is inferior to the other policies in terms of redundancy diversity. It has good load balancing performance, which is not helpful with our job service model.

References

[1] A. Behrouzi-Far and E. Soljanin, "Redundancy scheduling in systems with bi-modal job service time distribution," *arXiv preprint arXiv:1908.02415*, 2019.