

# Network-Aware Scheduling for Data-Parallel Jobs: Plan When You Can – Public Review

Minlan Yu  
University of Southern California  
Los Angeles, CA  
minlanyu@usc.edu

In the big data era, job scheduling is an important and challenging problem. While there have been many solutions on improving job completion time, this paper makes the following interesting observations:

- Many jobs are recurring and have predictable characteristics such as input data size.
- We can coordinate the placement of input data and tasks to improve data locality and thus job completion time.
- We can place jobs within a few racks to reduce the cross rack bandwidth usage without affecting parallelism.

Based on these observations, this paper proposes Corral, a new scheduling framework that improves job completion time. Corral first builds analytical models (latency response function) that estimates jobs' running times under various resource allocations for recurring jobs, which is based on predicted input data sizes and resource demands. Based on the models, Corral determines the number of racks to be allocated to each job, searches the specific set of racks for the job, and determines when to run each job to minimize the average job completion time. Once the decision is made, Corral first pre-loads the input on the selected racks and then runs tasks in the same racks, which achieves the joint placement of data and tasks. Corral also helps adhoc jobs because it reduces the resource usage for recurring jobs and thus leaving more resources for other jobs.

Corral is implemented in Yarn and evaluated in a large-scale computing cluster with production workloads. The evaluation shows that Corral significantly reduces the cross-rack transfers and improves the job completion time.

The reviewers like the paper because Corral provides a novel view of job scheduling solution (combining data and task placement) and demonstrates its significant improvement over existing approaches. The paper also presents its intuitions and design in a clear way. Moreover, the paper shows the benefits of each design decision with individual strawman solutions such as the

original scheduler in Yarn, ShuffleWatcher which schedules jobs to reduce cross-rack traffic, localShuffle which uses Corral's task placement and HDFS data placement, and Varys which optimizes flow scheduling. There is also a good sensitivity analysis about the robustness of Corral against prediction errors and available network bandwidth.

The paper made several assumptions that may not be true for a diverse range of scenarios. First, Corral improves the job completion time more when the cross-rack bandwidth is limited. This may no longer be a constraint with better datacenter topologies. Second, Corral currently assumes that each job reads from its own dataset. However, in practice, multiple jobs may read from the same dataset. Lastly, the latency response function may not always be accurate for jobs in practice (e.g., jobs completion time may vary for different datasets or may not decrease significantly with more racks). While Corral does not require accurate latency estimation, it is still an open question on how to improve the latency estimation. Hopefully, this paper will inspire more future works that address these questions.