Masters Thesis: Learning-Based Restart Schemes for Reducing Big-Data Job Processing Times

Background

Today, big data applications are being processed at a massive scale on large computing clusters such as data centres. Highly efficient systems like Map-Reduce, Spark etc., have been successfully deployed in these clusters to manage resources and schedule applications with embarrassingly parallel workloads. At a high level these systems follow a master-slave architecture, where the workload/job is divided into tasks with each task processing an independent chunk of data. Despite knowing the statistics of these tasks (e.g. size of the data to be read, number of instructions to execute) a priori, some tasks have longer than expected processing times (straggling tasks) due to random factors of variation associated with the slave machines [1] on which they are scheduled. A few straggling tasks can introduce a significant delay in processing the job. To combat the straggling tasks problem *speculative execution* has been proposed. Two key techniques used in speculative execution are 1) *Restart* - kill a task on a slave machine and restart it on another slave machine and 2) *Redundancy* - schedule multiple copies of the same task on different slave machines and when a copy finishes then cancel all other copies.

Restarting a task saves resource usage time as it kills the straggling task on a machine and schedules it on another machine for a faster execution. However, it requires a "good" estimate for when to kill a task on a machine. In practice, such estimates are obtained by monitoring the progress of the task periodically. Since big data jobs have thousands of tasks, monitoring the progress of each task periodically may not be a scalable solution. Further, the estimates obtained can be error prone. In this project we take a new approach where we explore *learning-based* Restart schemes. To this end we exploit the fact that each job is allotted a finite number of machines from the cluster to process its tasks, and the number of machines can be far lower than the number of tasks. In this case one may observe the processing times of tasks that are finished and update the estimate for when to restart the unfinished tasks. Thus, a learning-based restart scheme does not require monitoring the progress of tasks, and the estimates become better with number of finished tasks.

Task

Part I (Mandatory): Understand the problem of stragglers and do a literature review on existing Restart schemes (cf. [2]). Propose learning-based restart schemes. To this end the student may formulate the problem of observing the processing times of the finished tasks and estimating the restart times for unfinished tasks as a multi-armed bandit problem or use similar approaches. Study the performance of the proposed schemes using trace driven simulation. For the trace driven simulation, statistics of jobs from the Google Cluster traces will be provided.

Part II: If the proposed schemes perform better than the existing alternatives, the student may try and prove performance bounds for the schemes. Further, the student may set-up a cluster of VMs on Amzon EC2 and implement the learning-based schemes in the scheduler by emulating the big data jobs.

Required Skills: The student must have basic knowledge of probability and random processes and have taken a course in Design and Analysis of Algorithms (can be a part of a programming course). A course in Reinforcement Learning is added plus. He/She should have strong coding skills in any one of the languages (e.g. C/C++/JAVA/Python). We use Pandas to analyze data from Google Cluster traces, so Python is an added advantage.

Contact

If you are interested in more details about the project please email to Jaya Prakash Champati at jpra@kth.se

References

[1] G. Ananthanarayanan, S. Kandula, A. Greenberg, I. Stoica, Y. Lu, B. Saha, and E. Harris, "Reining in the outliers in map-reduce clusters using mantri," in *9th USENIX Symposium on Operating Systems Design and Implementation (OSDI 10)*. Vancouver, BC: USENIX Association, 2010.

[2] M. Luby, A. Sinclair, and D. Zuckerman, "Optimal speedup of Las Vegas algorithms," *Information Processing Letters*, vol. 47, pp. 173–180, 1993.