IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Big Data Covey Routine with Emphatic Slow Starter Apprehension and Conceptive Enactment

Ms Mufliha Parveen C M[1], Smt Geetha N B[2]

M.Tech, Computer Science and Engineering, University BDT College of Engineering, Davanagere, Karnataka, India[1]

Assistant Professor, Department of Computer Science and Engineering, University BDT College of Engineering, Davanagere, Karnataka, India[2]

Abstract: When one of the many tasks is assigned to a congested machine, processing of these tasks becomes slow. This problem is named to be as straggler or slow starter tasks. To tackle this so-called straggler problem, most parallel processing frameworks such as Map Reduce adopts various strategies, among which one of that is speculatively launching additional copies of same task when extra idling resource is available. Processing clusters consists of different conditions of loading namely lightly loaded and heavily loaded the major focus is on lightly loaded condition. For lightly loaded case, a cloning scheme, namely, Smart Cloning Algorithm (SCA) is utilized which is based on maximizing overall system utility. For heavily loaded case, "Enhanced Speculative Execution (ESE)" is used which is an extension of the Microsoft Mantri scheme to mitigate stragglers. After simulating the results obtained show that SCA reduces the total job flow time that is the job delay/ response time by nearly 6% comparing to speculative execution strategy of Microsoft Mantri. In addition, ESE Algorithm outperforms the Mantri baseline scheme by 71% in terms of job flow time while consuming the same amount of computation resource.

Keywords: congested machine, parallel processing jobs, straggler problem, and smart cloning algorithm.

1. INTRODUCTION

EMPIRICAL performance studies of large-scale computing clusters have indicated that the completion time of a job is often slow due to existence of "stragglers" or straggling tasks, i.e., tasks which are unfortunately assigned to either a failing or overloaded server within a cluster of hundreds of thousands of commodity servers. To mitigate stragglers, recent big data frameworks such as the Map Reduce system or its variants have adopted various preventive or reactive speculation strategies under which the system launches extra (backup) copies of a task on alternative machines in a judicious manner. In particular, there exist two main classes of speculative execution strategies, namely, the Cloning approach and the Straggler-Detection-based one. Under the Cloning approach, extra copies of a task are scheduled in parallel with the initial task as long as the computation cost of the task is expected to be low and the system resource is available. For the Straggler-Detection based approach, the progress of each task is monitored by the system and backup copies are launched only when a straggler is detected. The cloning-based strategy is only suitable for a lightly loaded cluster as it launches the clones in a greedy, indiscriminately fashion.
On the other hand, the straggler-detection based strategy is applicable to both the lightly-loaded and heavily-loaded regimes but at the expense of extra system instrumentation and performance overhead as discussed in.
The situation is particularly challenging when the progress of a large number of tasks have to be tracked. However, previous works do not compare the performance between these two different speculation approaches.
Furthermore, most of the existing speculative execution schemes are based on simple heuristics and do not consider the optimization based on specific performance objectives.

2. LITERATURE SURVEY

Mitigating stragglers is an active topic and lot of work have been done in this area and there is lot to be improved also. Utilizing of correct scheduler plays an important role hence the discussion on this is given [1]. Lot of discussions are made in [3] based on improvement of Map Reduce progress. A technique named GRASS [5] was used which delicately maintains a balance amongst the immediately improved tasks and the tasks that is present in idle resources.

3. PROPOSED SYSTEM

Earlier in existing system to mitigate stragglers Microsoft Mantri scheme was used but the Mantri scheme showed poor performance as they miss outliers that happen early in the phase and by not knowing the true causes of outliers, the duplicates they schedule are mostly not useful hence each time it needed to restart for average phase

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

In proposed system model proposes a generalized optimization framework to maximize the overall system utility which is a combination of the overall job flow times and computation costs. In this paper cut-off workload threshold between the lightly-loaded and heavily-loaded operating regimes of a computing cluster is taken into consideration. Based on this workload threshold, the applicability of a speculative execution strategy can be analysed for different operating regimes. Specific Smart Cloning Algorithm (SCA), which serves as an approximate solution to our optimization framework in a lightly-load cluster.

4. IMPLEMENTATION

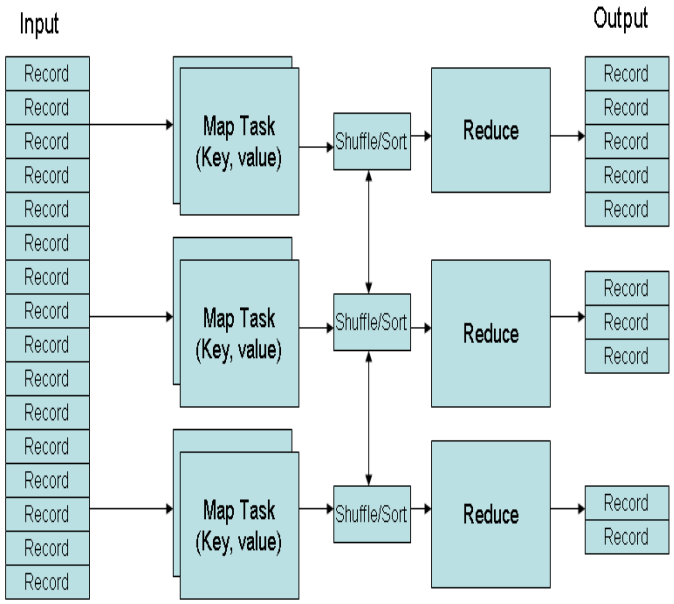

Fig 1: Architecture

4.1. JOB SCHEDULING IN A MAP REDUCE

In a big data processing cluster like Map Reduce and its variants or derivatives, different applications/ jobs need to share and compete for resources in the cluster. Thus, job scheduling plays a very important role.
Throughout the whole paper, the centralized scheduling paradigm is considered under which a global scheduler of the cluster manages all jobs where each job may consist of many small tasks. The scheduler allocates resources across jobs and also handles straggling tasks.

4.2. SPECULATIVE EXECUTION POLICIES

Several speculative execution strategies have been proposed for Map Reduce-like systems. The initial Google Map Reduce system only begins to launch backup tasks when a job is close to completion. It has been shown that speculative execution can decrease the job service time by nearly 44%. This scheme is easy to implement but it would unnecessarily launch backup copies for tasks of normal progress.
The speculative execution strategies in the initial versions of Hadoop and Microsoft Dryad closely follow that of the Google Map Reduce system.

However, a new strategy called LATE (Longest Approximate Time to End) was proposed for the Hadoop-0.21 implementation. It monitors the progress rate of each task and estimates their remaining time to completion.
Tasks with progress rate below certain threshold are chosen as backup candidates and the one with the longest remaining time is given the highest priority. The system also imposes a limit on the maximum number of backup tasks in the cluster.

4.3. SMART CLONING ALGORITHM (SCA)

Here integer part of the solutions is taken such that the number of copies allocated to each task is integer valued. However, following the analysis, there exists a case where there is no space for task cloning at the beginning of a particular time slot, i.e., P K i > N(l), though such a case seldom happens in a lightly-loaded cluster. In this scenario, it does not make sense to solve P2. Instead, an alternative scheduling scheme is designed to allocate the servers to these jobs by adopting a smallest remaining workload first scheme, which is extended from SRPT scheduler. Based on P2

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

and the extended SRPT scheduler, proposed the Smart Cloning Algorithm (SCA) below. m l i SCA consists of two separate parts. At the beginning of each time slot, we first schedule the remaining tasks of unfinished jobs and then check whether the condition P < N(l) is satisfied. If the condition is satisfied, we solve P2 to determine the number of clones for each task. Otherwise, i.e. i m l i P _ N(l), we sort _(l), i.e., the set of unscheduled jobs, according to the increasing order of the workload in each J i m l l i i and schedule the jobs based on this order and only one copy of each task is created. Note that the resultant workload is the product of m l i .

5. CONCLUSION

To obtain a redundant algorithm in big data processing cluster job scheduling and speculative executive schemes are combined together. Along with which to improve overall system utility an optimized framework is built in which an straggler problem is easily identified by "Smart Cloning Algorithm" which is based on cloning approach and classified under lightly loaded condition. Due to which the two major performance metrics average flow time and computational cost of the system is minimized to the great extent. As for future work can be of creating a speculative execution schemes that can easily handle more number of complex jobs.

REFERENCES

[1] Residual lives, hazard rates, and long tails.
[2] Apache. http://hadoop.apache.org, 2013.
[3] Capacity Scheduler. http://hadoop.apache.org/docs/r1.2.1/capacity scheduler.html, 2013.
[4] G. Ananthanarayanan, M. C.-C. Hung, X. Ren, and I. Stoica. Grass: Trimming stragglers in approximation analytics. In NSDI, April2014.
[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine Learning, 4:1–122, January 2011.