PADIMapNoReduce Platform

João Pinho
jpepinho@gmail.com

Diogo Rosa
diogomcrosa@hotmail.com

Cláudia Filipe
claudiapbfilipe@gmail.com

Instituto Superior Técnico
Middleware for Distributed Internet Applications
Lisbon, Portugal

Abstract

This project consists in the design and implementation of PADIMapNoReduce Platform, a simplified implementation of the MapReduce middleware and programming model. This platform extracts the input key/value pairs from input files and distributes the Map calls, called Jobs, across multiple machines, the Workers. Also, the platform ensures a good performance by monitoring jobs' progress, detecting faulty or slow machines and rescheduling their tasks on idle machines. That is assured by JobTrackers, which are distributed in this platform, in contrast to the original implementation of MapReduce, where they are centralised. This mechanism was implemented inspired by Facebook's solution Corona [1]. Additionally, in order to test the platform it was developed a PuppetMaster component which allows to control the platform, and also to induce some delays and faults to the system in order to perform some tests and evaluation.

1. Introduction

MapReduce was introduced by Google in 2004 [2] and is currently one of the most popular approaches for large scale data analytics - also thanks to the availability of high quality open-source implementations. When using the MapReduce paradigm, the computation takes a set of input key/value pairs, and produces a set of output key/value pairs. MapReduce users express the computation as two functions: Map and Reduce. This project focuses only the Map part, which uses a Map function given by the user and an input set of key/value pairs to produce a set of key/value pairs. In PADIMapNoReduce Platform the keys are the numbers of the line of the file being read and the values are the content of those lines.

The Map invocations, called Jobs, are distributed across multiple machines by automatically partitioning the input data into a set of splits of size S. The input splits can be processed in parallel by those machines, named Workers. The system ensures that for each job submitted, all the input data is processed with a good performance through the monitoring of jobs' progress, fault or slow machines detection and reschedule of idle machines' tasks. In the original MapReduce implementation these tasks are performed by the JobTracker which is a centralised component. If the JobTracker fails the system can't receive new jobs nor processing pending ones, which can be critical in systems that need high availability. Considering JobTracker as a single point of failure, it is necessary to replicate this component. As it would add complexity to the system and overhead, this project also introduces a new entity, the CoordinationManager, separating cluster resource management from job coordination, which allows the system to focus on make faster scheduling tasks.

2. Related work

3. Worker

Não sei se quero ter uma secção por cada componente mas por agora é isto.

4. JobTracker

5. CoordinationManager

6. DefaultJobScheduler

7. TaskTracker

8. TaskRunner

9. SlaveReplica

10. Distribution manner

11. Stuff and stuffy

Figure 1. Example of caption.

Figure 2. Example of long caption requiring more than one line. It is not typed centered but aligned on both sides and indented with an additional margin on both sides of 1 pica.

12. Puppet Master

Provavelmente faz mais sentido esta secção ser aqui, porque é o que vai permitir realizar os testes e avaliação.

12.1. Status

12.2. Create worker

12.3. Wait

12.4. Slow worker

12.5. Freeze/unfreeze worker

12.6. Freeze/unfreeze communication

13. Evaluation

The project's final report should also include some qualitative and quantitative evaluation of the implementation. The quantitative evaluation should be based on reference traces that will be provided at the project's web site, and focus:

- Job execution time, both in failure free scenarios and in presence of injected faults;
- Number of messages and size of messages exchanged among nodes;
- Size (in bytes) and number of the final key-value pairs emitted on average.

14. Future work

- se a replica mais prioritaria crashar
jpinho [2:15 PM] as outras ficam a aguardar...
jpinho [2:15 PM] deviam detectar e reorganizar-se

15. Conclusions

References

[1] A. Ching, R. Murthy, D. Molkov, R. Vadali, and P. Yang. Under the hood: Scheduling mapreduce jobs more efficiently with corona. *Facebook Engineering*, November 2012.

[2] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. *Communications ACM*, 51(1):107–113, January 2008.