Exploring QoS in MapReduce Task Scheduling

Pedro Alvarez-Tabio
Jesus Hernandez
Javier Villa
CS550

ILLINOIS INSTITUTE OF TECHNOLOGY

Index

- Background
- Motivation
- Overview of MapReduce task scheduling approaches
- Our approach: QoS
- System architecture
- QoS specifications and Service Levels
- Preliminary results
- Conclusions

ILLINOIS INSTITUTE OF TECHNOLOGY

Background

- MapReduce: data-driven programming model proposed by Google in 2004
  - Apache Hadoop: open-source version
- Especially well suited for distributed data analytics applications.
- Numerous companies and institutions focused on Big Data are relying in these technologies
  - To boost their data processing and computation performance in such data intensive jobs
  - Small tasks can be processed by the nodes in a parallel fashion
  - Example: LinkedIn uses Hadoop MapReduce for discovering People You May Know

ILLINOIS INSTITUTE OF TECHNOLOGY

Motivation

- Many companies require heavy data processing with MapReduce
- Outsourcing of MapReduce computation to other companies
  - Example: Cloudera
- Current task scheduling techniques in Hadoop do not take into account service guarantees

ILLINOIS INSTITUTE OF TECHNOLOGY

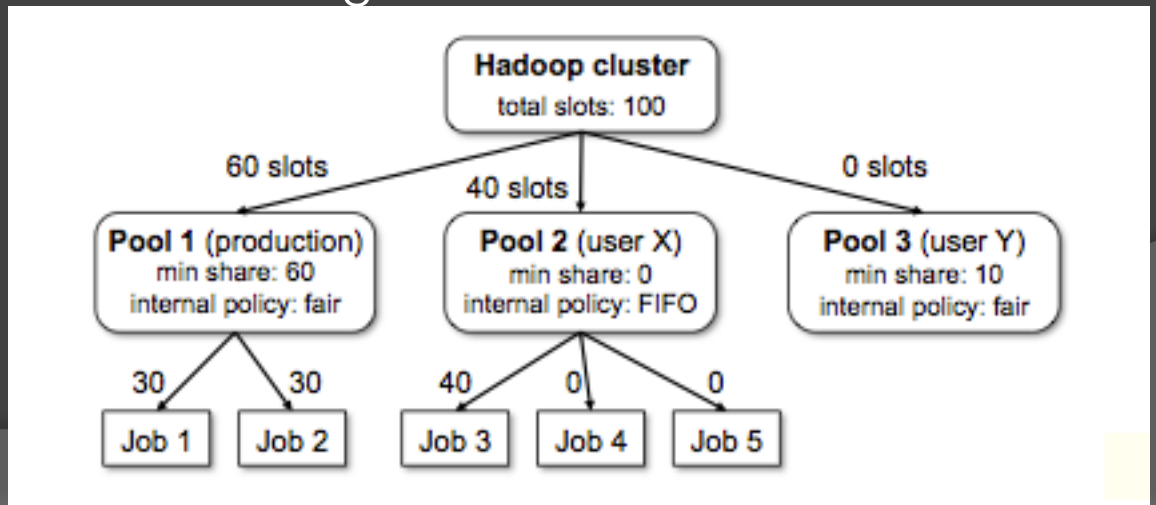
Overview of Hadoop task scheduling approaches

- FIFO scheduler
  - The original, primitive implementation of task scheduling
  - Included in Hadoop by default
  - Five priority levels, heartbeat-guided
  - Not fair because of its FIFO nature: short, high-priority jobs get stuck behind long, low-priority ones which have been scheduled before

ILLINOIS INSTITUTE OF TECHNOLOGY

Overview of Hadoop task scheduling approaches (II)

- Fair Scheduler
  - Takes into account both fairness and data locality to multiplex clusters efficiently
  - Two-level scheduling hierarchy
    - Top level: allocation of tasks across pools using weighted fair sharing
    - Lower level: each pool allocating slots (FIFO or fair sharing)
  - Prioritizes meeting minimum shares over fair shares



Overview of Hadoop task scheduling approaches (III)

- Capacity Scheduler
  - Based on arranging jobs into different queues
  - Each of them has an allocated fraction of the cluster capacity
  - The ordering of jobs can be performed either by a FIFO policy or by prioritizing jobs

ILLINOIS INSTITUTE OF TECHNOLOGY

Our approach: QoS

- Current approaches aim to specify fairness among jobs
- Neither of them say anything about guaranteed job/task processing
- Our approach: Quality of Service specification in Job and Task scheduler
  - Different client profiles for different performance requirements
  - Permits the fine tuning of the performance in each client
  - Allows the service provider to easily calculate the needed scale of the system in advance

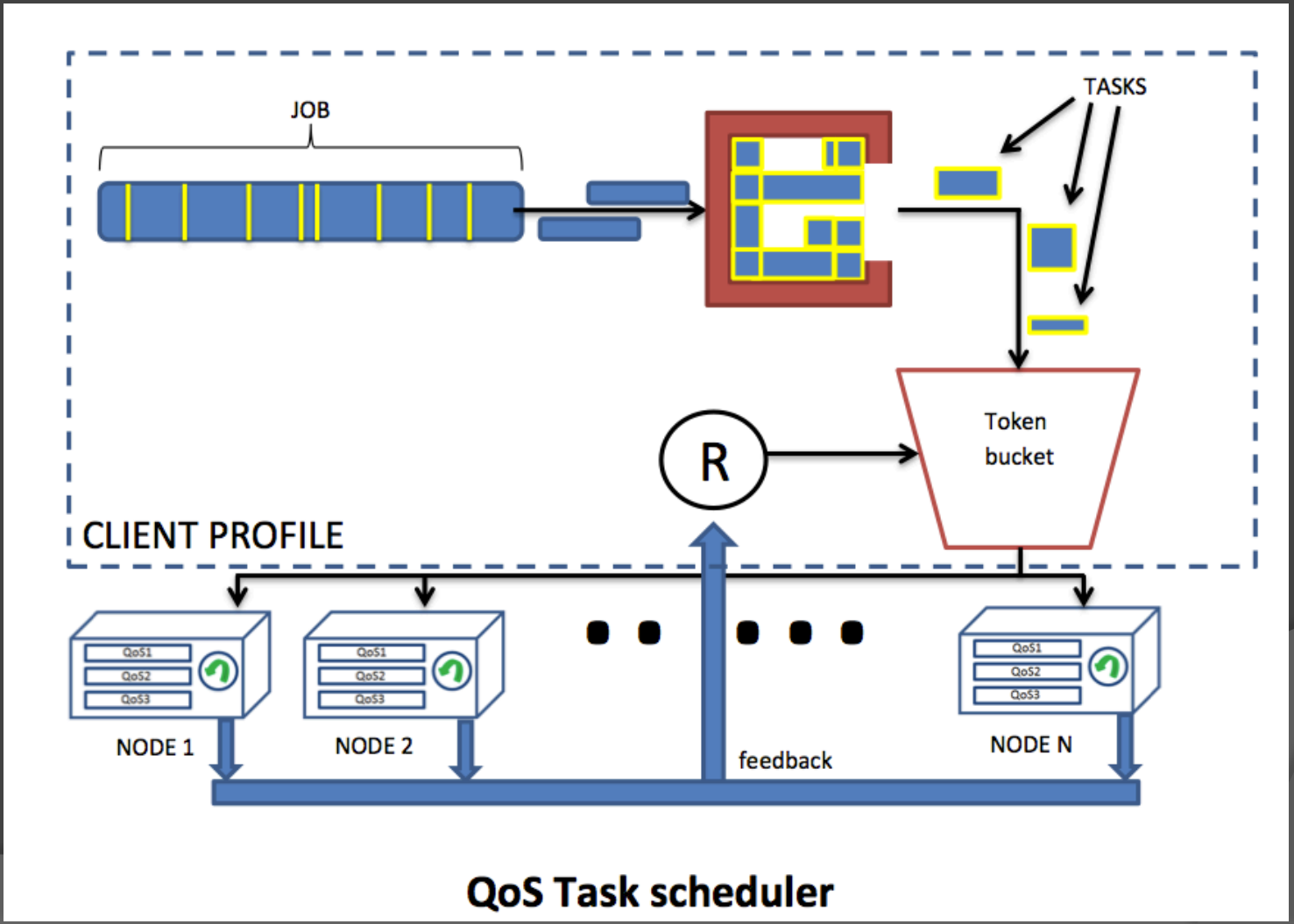
ILLINOIS INSTITUTE OF TECHNOLOGY

Our approach: main concepts

- We take algorithms from traffic control in computer networks and apply them to the special case of task scheduling
  - Token Bucket
    - Algorithm used in computer networks
    - Checks if data transmissions are conformant to limits in bandwidth and burstiness
  - Weighted Fair Queueing
    - Packet scheduling technique which allows different scheduling priorities (weights) to statistically multiplexed data flows
    - Can be implemented as a weighted round robin

ILLINOIS INSTITUTE OF TECHNOLOGY

System architecture



ILLINOIS INSTITUTE OF TECHNOLOGY

System architecture

- Queued Token Bucket
  - Tasks are queued before entering the bucket
  - Constant token arrival rate
  - Non-conformant tasks are sent lowest-priority
- Adaptive feedback to clients
  - 1 token = 1 ms of processing in any node
  - Weighted mean of the last durations gives Ntokens needed for next task submission
  - This way, we avoid greedy users submitting longer tasks take advantage of the system against other users

ILLINOIS INSTITUTE OF TECHNOLOGY

System architecture

- Weighted Fair Queueing
  - Each node has a queue for each user profile, with a specified priority
  - Weighted round-robin approach
    - Priorities are weights
    - Example: a queue with priority 10 will be polled 10 times a queue with priority 1, in case of great occupation
    - In case one of the queues is empty, the rest share the node's capacity
    - Non-conformant tasks get degraded to a lower-priority queue

ILLINOIS INSTITUTE OF TECHNOLOGY

QoS specification and Service Classes

- We have defined three Service Classes for the simulator:
  - Premium
    - Token Bucket: 1000 tokens/sec, 10000 tokens max
    - Priority: 10
  - Advanced
    - Token Bucket: 800 tokens/sec, 300 tokens max
    - Priority: 3
  - Basic
    - Token Bucket: 10 token/sec, 100 tokens max
    - Priority: 1

ILLINOIS INSTITUTE OF TECHNOLOGY

QoS metrics in simulator

- We have defined some metrics to evaluate the system's performance
  - Average job completion time
  - Task processing delay for each Class of Service
  - QoS : performance ratios
  - Percentage of non-conformant tasks

ILLINOIS INSTITUTE OF TECHNOLOGY

Conclusions

- Our design represents an initial attempt to make MapReduce “as a service” more flexible, scalable and cheap, both for the user and the service provider
- Final steps
  - Complete the metrics and curate the results of those metrics in our simulator

Questions?

ILLINOIS INSTITUTE OF TECHNOLOGY

THANK YOU