Implentation of an simplistic Interface for Big Data Workload Scheduler in Kubernetes

Implentation of an simplistic Interface for Big Data Workload Scheduler in Kubernetes

Lukas Schwerdtfeger

A thesis submitted to the
Faculty of Electrical Engineering and Computer Science
of the
Technical University of Berlin
in partial fulfillment of the requirements for the degree
Bachelor Technische Informatik

Berlin, Germany
December 22, 2015

Main supervisor:

Prof. Dr. habil. Odej Kao, Technical University of Berlin

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den

Zusammenfassung

Kurze Zusammenfassung der Arbeit in 250 Wörtern.

Abstract

Short version of the thesis in 250 words.

Acknowledgements

This chapter is optional. First of all, I would like to...

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

1.1.0.1 TODO:

- ◇ World is producing more and more data
- ◇ Business are storing more and more data
- ◇ Big Data Lakes with potential valuable information
- ◇ Requires Processing of Data
- ◇ Requires a lot of computational power, systems on single node are no longer enough to keep up with a stream of data
- ◇ Stream Processor frameworks like Spark and Flink
- ◇ Vastly different Runtime depending on Scheduling (Source?)
- ◇ Implementing/Testing new Scheduling Algorithms always require the Setup of an Cluster
- ◇ Unused Resources in a Production cluster due to inefficient Scheduling

1.1.0.2 Open:

- ◇ How much detail is required here?

1.2 Problem Description

1.2.0.1 TODO:

- ◇ Related Work to Scheduling Algorithms can not be applied to newer Cluster Management Software like Kubernetes
- ◇ Unfortunately setup takes a while to tryout new Scheduling Algorithms

1.2.0.2 Open:

- ◇ Should this section be already focusing on what i am actually trying to solve, or should this be more general and explain what is the Problem with scheduling

1.3 Goal of this Thesis

1.3.0.1 TODO:

- ◇ Implementation of easy to use Interface that would allow already Batch Job Scheduling Algorithms likes Hugo and Mary to be run with small changes, on the popular Cluster Management Software Kubernetes

1.4 Structure of this Thesis

1.4.0.1 TODO:

- ◇ Thesis starts by giving a brief background to Big Data Streaming Processing, Cluster Management Systems (Kubernetes), and Scheduling
- ◇ Discuss the Approach this thesis takes on tackling the Problem Description, by explaining how scheduling in Kubernetes works and what it takes to Extend Kubernetes (using the Operator Pattern)
- ◇ Implementation Details that a worth mentioning:
  - ○ An architectural Overview.
  - ○ The Process of designing an Interface
  - ○ The Operator that is used to extend Kubernetes
  - ○ Changes that had to be made to existing Algorithms (and their tests)
- ◇ How the work of thesis is evaluated, by testing it's functionality, comparing results from previous work and finally outlining its limitations
- ◇ Comparing the Work that was done to current State of the Art Technology like the Batch Scheduling Framework Volcano and comparing to Scheduling approaches that are not available on Kubernetes
- ◇ A final Conclusion, with a note on future work, that is missing from the current implementation or requires rethinking.

2
Background

2.1 Big Data Stream Processing

2.1.0.1 TODO:

- ◇ Explain why cluster computing is required to deal with the Big Data Problem
- ◇ Explain what makes distributing computation across a cluster hard
- ◇ Explain the Value of already existing Big Data Stream Frameworks like Spark and Flink
- ◇ Explain on a high level how these work
    - ○ Explaining the DAG is required in order to later differentiate between DAG-level scheduling and "Pod"-Level scheduling
    - ○ Driver and Executor Pods
- ◇ Mention the use on Kubernetes using the Spark and Flink Operator

2.2 Scheduling

2.2.0.1 TODO:

- ◇ Explain what Scheduling is
- ◇ Different kind of scheduling
    - ○ DAG Scheduling done by Spark (Not what this thesis is about)
    - ○ POD Scheduling done by the Cluster Resource Manager
        - ∗ Co-Location
        - ∗ Packing
- ◇ Explain why Scheduling is Important
    - ○ Co-Location Problem
    - ○ Low Resource Usage (Graph from Google)
    - ○ Results from Hugo/Mary Paper

2.2.0.2 Open:

- ◇ Should maybe start a bit less specific about this Thesis and find more Information about Scheduling in general or is it fine if the Background section starts of general and tailors towards the topic of my thesis?

2.3 Cluster Management Systems

2.3.0.1 TODO:

- ◇ Explain what a Cluster Resource Manager is doing
  - ○ Abstraction of using a single cluster as a single Machine
  - ○ Managing given resources making it scalable by adding more machines
- ◇ Show what are the differences between YARN and Kubernetes
  - ○ YARN: Emerging from Hadoop was design to Work with Batch Applications
  - ○ Kubernetes: All Round Cluster Manager, with a Big Community

2.3.0.2 OPEN:

- ◇ Where do I explain why I am using Kubernetes, this is already required to be part of the Problem Definition?

3

Approach

3.1 Scheduling in Kubernetes

3.2 Extending Kubernetes using the Operator Pattern

4

Implementation

4.1 Architecture

4.2 Designing the Interface

4.3 Operator

4.4 Changes to existing Algorithm

5
Evaluation

5.1 Testing

5.2 Comparing to baseline Runtime

5.3 Limitations

5.4 Discussion

6

State of the Art

6.1 Volcano

6.2 Non Kubernetes

7

Conclusion and Future Work

7.1 Conclusion

7.2 Future Work

Bibliography