Adaptive Optimal Control of MapReduce Performance, Availability and Costs
Sophie Cerf, Mihaly Berekmeri, Bogdan Robu, Nicolas Marchand, Sara Bouchenak

To cite this version:
Sophie Cerf, Mihaly Berekmeri, Bogdan Robu, Nicolas Marchand, Sara Bouchenak. Adaptive Optimal Control of MapReduce Performance, Availability and Costs. 11th International Workshop on Feedback Computing (Feedback Computing 2016), Jul 2016, Wurzburg, Germany. <hal-01331024>

HAL Id: hal-01331024
https://hal.archives-ouvertes.fr/hal-01331024
Submitted on 13 Jun 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Adaptive Optimal Control of MapReduce
Performance, Availability and Costs

Sophie Cerf
GIPSA-lab, CNRS
11 rue des Mathématiques
Grenoble, France
sophie.cerf@gipsa-lab.fr

Mihaly Berekmeri
GIPSA-lab, CNRS
11 rue des Mathématiques
Grenoble, France
mihaly.berekmeri@gipsa-lab.fr

Nicolas Marchand
GIPSA-lab, CNRS
11 rue des Mathématiques
Grenoble, France
nicolas.marchand@gipsa-lab.fr

Bogdan Robu
GIPSA-lab, CNRS
11 rue des Mathématiques
Grenoble, France
bogdan.robu@gipsa-lab.fr

Sara Bouchenak
LIRIS - INSA Lyon
20, Avenue Albert Einstein
Lyon, France
sara.bouchenak@insa-lyon.fr

ABSTRACT

MapReduce is a popular programming model for distributed data processing and Big Data applications running on clouds. Extensive research has been conducted either to improve the dependability or to increase performance of MapReduce, ranging from adaptive and/or a fee. Request permissions from permissions@acm.org.

Feedback Computing 2016 June 19, 2016, Wurzburg, Germany

DOI: 10.1145/1235
dustry to handle these large amounts of data. MapReduce is a popular programming model and execution environment for developing and executing distributed data-intensive and computer-intensive applications [9] now mostly used in its open source implementation Hadoop. It was developed in 2008 by Google, and broadly improved since, to automatically handle data partitioning, consistency and replication, as well as task distribution, scheduling, load balancing and fault tolerance, and nowadays it is backed by Big Data industry leaders such as Facebook, Yahoo or LinkedIn (see [17], [9] among others). Hence, the complexity of configuration of such a system is continuously increasing while the user expectations remain the same: continuous availability and fast response times are the required norm. Moreover, with the advent of cloud solutions, the environments where BigData systems need to run is becoming more and more dynamic. These environments are influenced, among many other, by input/output and network skews [15] and hardware and software failures [16]. Moreover, workloads variation over time [12, 1, 14] both in size and in nature are a major cause of the unpredictable changing behavior of MapReduce.

Furthermore, ensuring the performance and dependability of cloud services still poses several challenges. Extensive research has been conducted to improve dependability or performance of MapReduce by changing the behavior and/or algorithms of the framework itself (see [19] for instance). Although these solutions improve upon how MapReduce works, no guarantees are provided in term of performance and availability. Some solutions for performance modeling [18, 11] and control [5] can be found in the literature. However, to the best of our knowledge, there are no work to provide concurrent guarantees in terms of combined dependability and performance, while reducing utilization costs.

From this context we identify two objectives as being still open problems concerning MapReduce usage. This paper details a work-in-progress and is intended as a position (or challenge) paper. Further on the article highlights the main challenges that such approach has. First there is a need for solutions that can guarantee multiple objectives such as performance level or dependability while being parsimonious in resource usage. Second, these solutions need to be highly robust to variations of the Hadoop framework itself and to the dynamics of its environment. To tackle these issues we chose to use control theory, as it is a well established theory being promisingly applied to software systems [10]. In the following, we present an algorithm that generate the optimal configuration of a MapReduce system to meet performance, availability, costs

1. INTRODUCTION

The amount of data produced by everything from mobile phones, tablets, computers to smart watches brings novel challenges in data storage and analysis. Many solutions have arisen in research and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CCS Concepts

• Networks → Cloud computing; • Software and its engineering → Software configuration management and version control systems; • Computer systems organization → Dependable and fault-tolerant systems and networks;

Keywords

control of computing systems; cloud computing; adaptive control; optimal control
and power constraints taking into account the dynamics of the environment by scaling the resource cluster and realizing admission control. Moreover, this control algorithm is able to adapt itself to changes in Hadoop or its working environment, hence fulfilling robustness specifications.

2. STATE OF THE ART

2.1 MapReduce Dynamic Modeling

Capturing the complex behavior of a cloud service in varying environments which will be valid in all its configurations while being implementation independent is highly challenging. A simple model that captures quite well the dynamics of MapReduce systems and can be used to predict its performance and availability levels was already proposed in [3]. The model is built as a set of difference equations that describe the impact of changes in the inputs (control variables and disturbances) on system’s outputs, see Equation 1.

\[
\begin{align*}
R_t(k) &= a.R_t(k-1) + b(N(k-5) + N(k-6)) \\
R_{MC}(k) &= c.R_{MC}(k-1) + d(MC(k-8) + MC(k-9)) \\
Av(k) &= \alpha.Av(k-1) + \beta(MC(k) - C(k)) \\
R_t &= R_t(k) + R_{MC}(k) \\
\end{align*}
\]

Event though these techniques have proved their efficiency, to our knowledge there is very difficult to perform both performance and dependability control while reducing the cost of running the cluster and being robust to workload changes and environment variations. Indeed, the actions of both control systems for performance and availability are highly coupled, as the model of Equation (1) highlights. Moreover, these techniques are heuristic and based on reaction or prediction of events so they cannot ensure the decision taken is the optimal one. Consequently, we propose a Multi-Input-Multi-Output optimal controller to ensure multiple objectives in the best possible way.

3. PROPOSED SOLUTIONS

In order to meet the 2-fold objectives motivated in Section 1, we combine two fields of control theory:

- Adaptation of the model previously developed by [4] is performed on-line using a recursive algorithm;
- Control signals are optimally computed to achieve performance, availability and costs objectives, even when the objectives are contradictory.

3.1 Adaptive Modeling

When adapting the model, we keep the overall structure of Equation 1 unchanged and we re-compute its parameters on-line to better fit the real system. For that, we use a recursive least square estimator (RLSE) [2] to on-line optimize our model parameters. RLSE is a well known algorithm that has proved its stability, accuracy and simplicity for a long time [2]. RLSE recursively updates the model parameters by minimizing a least square cost function between the measured outputs and the estimated ones. To compute new parameters the estimator only needs current signals, past parameter estimations and the model structure. This model adaptation has been introduced in [6] on a Single-Input-Single-Output model between the cluster size and the response time. We further extended the estimator to our Multiple-Input-Multiple-Output (MIMO) model by introducing other variables: availability and the MC number.

![Figure 1: Comparison of adaptive and classic modeling](image)

Figure 1 shows the compared performance of both adaptive and non adaptive models for representatives configurations of MapReduce, it highlight the fitting improvement of the model, especially when system saturation is reached.
3.2 Optimal Control

The aim of our control algorithm proposed in [6] is to guarantee performance and availability levels using the minimal amount of resources, even when the system faces bursty variations in its workload. For that purpose, at each time step \( k \) we define a cost function \( J(k) \) (see Equation (2)) designed to be minimized and which has two components: the first one makes the response time (resp. the availability) match its reference values \( R_{\text{t max}} \) (resp. \( A_{\text{min}} \)), while the second component minimize the cluster size \( N \).

\[
J(k) = (R^k - R_{\text{t max}}) + N^k \quad \text{and} \quad J(k) = (A^k - A_{\text{min}}) + N^k
\]

Each of the above mentioned components are considered over a time horizon of \( H \) samples, called prediction horizon. At the \( k^{th} \) time instant, the response time, availability and cluster size respectively, are measured and the MapReduce model is used to predict the behavior of the system over the prediction horizon. Equation 3 is an example of the mathematical formulation of the predicted set for response time values, where \( R_t(k) \) is the measured one while the other elements are predicted by the model.

\[
R_t^k = \begin{pmatrix} R_t(k) \\ R_t(k+1) \\ \vdots \\ R_t(k+H) \end{pmatrix}
\]

The algorithm computes, and periodically updates, a profile for the control signals which minimizes the cost function \( J \) over the whole prediction horizon, that is to say it computes a control profile which guarantees performance and availability while minimizing the resource consumption. Furthermore, by adding weights (the \( Q \) and \( R \) matrices) to the different components of the cost function we can give more importance to particular specifications (response time, availability or costs): when the controller cannot meet all the constraints on the system, the weighting matrices allow him to make a trade-off, according to user specifications, between those which should be kept at any price an those for which we can afford relaxation.

Due to page restriction, we do not present here validation and implementation of the proposed control, for further details one can refer to [7].

4. VISION AND OPEN CHALLENGES: COMBINED ADAPTIVE AND OPTIMAL CONTROL

To achieve both goals (ensuring multiple objectives such as performance, availability and cost reduction while being robust to any kind of variation or disturbance) the presented solutions of adaptive modeling and optimal control should be combined. This merged solution consists in using the adaptive model for the output predictions used in the formulation of the optimal cost function (see Equation (3)). Figure 2 below sketches this principle.

Even if this idea seems promising, there is some theoretical and technical challenges that need to be tackled. The major ones are:

- **Stability** When updating the model, what kind of properties could we expect from the control signals? Can we ensure that the created closed loop will be stable? Indeed one can imagine that the consequence of applying a control profile designed at a certain operating point could be different than expected (for instance due to exogenous disturbances) and lead to oscillations or even instability. If absolute stability cannot be proved, can we find a condition on the control law that nevertheless leads to stability? Mathematical proof that the system will converge with a precedentely computed control signal even if the system itself changed should be given.

- **Comparing Dynamics** The dynamics of the model adaptation process is not necessary the same as the one of the control algorithm. When comparing computation complexity, the adaptation is expected to be faster than the control, however one can imagine intentionally slowing down the adaptation process. In the first case, the difference in dynamics can allow time for ensuring the convergence of the model parameters before modifying the system behavior, while in the second case it can enable the system to reach steady state before deciding if the system changed.

- **Event-Driven adaptation or control** Following this idea of asynchronous adaptation and control, one can imagine that model or control signal updates do not necessary need to be synchronous in time and could be driven by events generated by indicators such as the fit of the outputs estimation, the current operating point of the system or the level of specification satisfaction. The addition of such mechanisms could ensure stability for the system that were not reachable before. We believe that the mathematical proof of stability may be simplified, and, if found, will consist in one of the few existing solutions to guarantee adaptive optimal control stability [13].

5. CONCLUSION & FUTURE WORKS

In order to improve both performance and availability of a MapReduce deployments subject to a changing workload, we introduce a control strategy that, through mathematical optimization, scales the resources of the cluster and performs admission control. Furthermore, to ensure robustness to changes in the cloud environment or the MapReduce system itself, we develop an adaptation algorithm that updates on-line the prediction of the MapReduce system behavior using RLSE technique. Reaching those two objectives by combining the adaptive modeling with the optimal controller is a promising solution which however raises many challenges such as stability or synchronicity. All the technical keys have been developed to merge these theories, however there is still a theoretical gap to cross to properly use both these methods.

Future work will consist in solving these challenges through experimental validation as well as mathematical proofs. We aim to validate the stability and efficiency of the proposed optimal adaptive controller on different clusters (with different scales, deployments, etc.), with varying Hadoop distributions and configurations as well as with different workload profiles (data-intensive or computing-intensive jobs, different input data sizes, etc.).

![Figure 2: Adaptive optimal control schema](image)
6. REFERENCES


