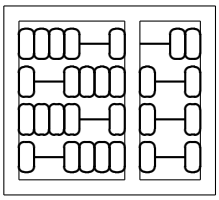
UNIVERSITY OF CAMPINAS

INSTITUTE OF COMPUTING



Dwsr: Predicting the memory usage of tensorial algorithms

Student: Daniel De Lucca Fonseca

Advisor: Prof. Edson Borin

Abstract: Lorem ipsum dolor

Contents

1 Introduction

Sit amet lorem

2 Background

Sit lorem dolor

3 Related Work

Lorem ipsum

4 Research proposal

In this research proposal, we aim to predict computer memory consumption from an algorithmic perspective. While there have been numerous studies on memory usage history analysis from the scheduler perspective, we believe that there is a significant gap in research when it comes to predicting memory usage of a specific algorithm.

4.1 Problem statement

The Discovery [1] laboratory, located at UNICAMP [2], is working in a seismic analysis project with Petrobras [3]. This project relies heavily on machine learning and seismic operators for its computing graphs. However, the input of these graphs is a massive seismic dataset that can contain terabytes of data. Even supercomputers do not have enough memory to handle the computation on a single node. Therefore, usually the execution is distributed by using data parallelism.

To facilitate this process, our laboratory implemented a framework called DASF [1] that simplifies the development of data-parallel computing graphs using DASK [2] under the hood. DASF [1] has an input parameter called **block_size** that defines the size of each seismic block used during data parallelism. However, setting this parameter can be challenging because it required finding the optimal relationship between it and the network overhead caused by it.

[1] `https://discovery.ic.unicamp.br/`

[2] `https://ic.unicamp.br/`

[3] `https://petrobras.com.br/`

To illustrate this challenge, I present image **??**sizefig:block_sizeh contains three computing graphs receiving input data from a seismic dataset. The first graph shows the input data as the entire dataset, which requires a significant amount of memory to execute. The second graph divides the data into thousands of small parts, reducing the memory requirement but adding network overhead. The third graph divides the data into a smaller number of parts, minimizing both network and memory requirements.

While executing the graph, the developer must manually set the block_size parameter. Since Petrobras use subcomputers to execute those graphs, setting a large number may lead to memory issues causing a signifcant delay due the trial-and-error nature of the usual execution. On the other hand, setting a small number may increase the execution time due to network overhead. Since Petrobras have a large number of graphs to execute, and each graph usually takes a long time to execute, I need to find a way to optimize the block_size parameter.

DASK [2] provides an automatic chunking feature, but it relies on the chunk_size parameter, which is a static parameter define prior to execution. Since the developer do need to define this parameter prior to the execution, they can't rely on DASK [2] auto chunking feature to automatically split the data, but they can rely on the automatic chunking if they figure out the ideal chunk_size prior to the execution.

Based on this assumption, if someone predicts the memory usage of the graph that person can use the DASK [2] auto chunking feature to automatically split the data into the ideal number of chunks. Therefore, this research aims to develop a memory usage prediction model that can help us determine the optimal block_size parameter during execution. This will help us optimize resource utilization and minimize waiting and execution time. The model will provide a comprehensive understanding of memory usage patterns for different block sizes and contribute to the development of more efficient resource allocation strategies in large-scale clusters.

4.2 Proposed solution

I see two possible solutions to optmize the block_size parameter: (i) I can create a model that would predict the ideal block_size of algorithm based on the input data or (ii) I can create a model that would predict the memory usage of an algorithm based on the input data, and use it to determine the optimal block_size.

Although appproach (i) is more straightforward, it is limited only to the block_size. Approach (ii), on the other hand, is broader and can be used not only to identify the block_size, but also as

an heuristic for other tasks like scheduling, resource allocation, and cloud cost estimates.

Based on this, I have decided to pursue approach (ii) and create a machine learning model that would predict the memory usage of an algorithm based on the input data. While there are already alternatives available to predict memory consumption from the scheduler's perspective, no research has been conducted to predict memory consumption from an algorithmic perspective.

The proposed solution is divided into two strategies:

4.2.1 Strategy One: Algorithm-Specific Model

The first and primary goal of this research is creating a machine learning model that can predict the memory consumption of a given algorithm given the input data. During this strategy, I will train a model for each algorithm, considering the seismic data's shape and size as the primary features for the prediction. Since seismic dataset are usually normalized, I may also extract other features to improve the prediction, but such features are going to be defined during the exploration phase. I will start the exploration with polynomial model, since I belive it is suitable for the prediction, and the model's output would help us determine the optimal block_size for each part of the graph.

This approach's primary limitation is that it requires a new model for each seismic operator, but it allows us to predict the output size and shape, enabling us to compose a graph prediction.

4.2.2 Strategy Two: Generic Algorithm Model

Our secondary goal is to develop a model that is algorithm-agnostic. If the results of the first approach are optimistic, I may extract features from the source code itself to predict memory consumption accurately. I believe that the algorithm could extract relevant information, such as how the code author deals with memory management. Although I do not have a clear picture of the features that I could extract, I believe that this is a possible experiment for this research. However, this objective is secondary and will be pursued only if the results of the first approach are positive.

References

[1] J. Faracco, O. Napoli, and E. Borin, *Dasf is an accelerated and scalable framework*, `https://github.com/discovery-unicamp/dasf-core`, 2023.

[2] M. e. a. Rocklin, *Dask*, `https://github.com/dask/dask`, 2023.