Max Planck Computing and Data Facility
Chair of Computer Architecture and Parallel Systems

Tensorflow

Seminar: Efficient Programming of HPC Systems
Frameworks and Algorithms

Efe Kamasoglu

May 29, 2023

Contents

Abstract

1 Introduction

TensorFlow is a free and open-source framework developed by Google Brain, which finds its application widely in the field of machine learning and artificial intelligence. It is used to build and train large-scale models according to the client's preferences and provided data sets. In order to train a model, TensorFlow carries out several computations by mapping them onto a variety of hardware, such as mobile devices or HPC clusters consisting of multiple computational units with hundreds of GPUs. Those computations are represented in a ”directed dataflow graph”[1] as a whole, which is then compiled statically or dynamically depending on the version of Tensorflow.

To construct such graph, a client needs to create a session which is an interface with methods of its own. Through those methods, the client can introduce his data set to Tensorflow's system, define his model and its specifications. A graph is typically composed of the following:

- *Node:* A node has zero or more inputs and zero or more outputs. It is an instantiation of an operation.
- *Edge:* Data flows through the edges from one node to another. There are also edges which are not for the dataflow, but for control dependencies between different nodes; e.g., execution order of the concerning operations.
- *Operation:* An operation represents a computation such as addition or matrix dot product.
- *Tensor:* Tensors are multidimensional arrays describing the data. They flow into the nodes as inputs through the edges of a graph.
- *Kernel:* A kernel is an implementation of an operation.
- *Device:* Devices are computational units on a machine that are utilized by TensorFlow's system to execute kernels. (e.g. GPUs)

Client can either execute all or a part of the graph with the help of the session, where he interacts with the processes that regulate access to devices. In addition to that, TensorFlow system itself has different implementations for execution mechanisms both for single- and multiple-device systems.

2 Execution of a Graph

Each implementation follows the same standard (Fig. 1): Client process calls the master process through the session interface to ignite the execution. An arbitrary number of worker processes, each of which executes a subgraph, are called by the master process. Master mostly distributes the operations as well

[1] Martin Abadi, Ashish Agarwal, ..., ”TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” November 9, 2015, p. 1, para. 2

as the data tensors to the workers. Each worker is responsible for one or more devices. The number of workers depends on the architecture of the system. As we are going to focus on efficiency, scalability and portability of TensorFlow in terms of HPC systems, we have to take a closer look at the distributed (multiple-device) implementation.

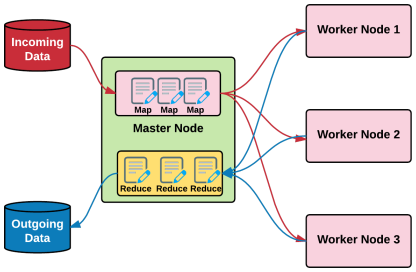

Figure 1: Master and Worker processes of a distributed system[2]

3 Distributed (Multiple-Device) Implementation

When it comes to a system with a single device and a memory, the execution is fairly simple: Place the data into the memory, map all the operations to the device and let it execute them sequentially.

[2] https://www.oreilly.com/api/v2/epubs/9781787281349/files/assets/Ch09-Fig11.png