Reading Assignment 1: TensorFlow/XLA and JIT

Sagar Jain
CS17BTECH11034

September 2, 2019

1. What is XLA? Why is it so crucial to the performance of TensorFlow?

   Ans. XLA stands for Accelerated Linear Algebra. It is the compiler for linear algebra for Tensorflow, an open source machine learning library.
   Almost any machine learning algorithm is implemented by constructing a computational graph and computing the values of the nodes (and gradients with respect to a set of nodes in case of back-propagation based algorithms). Computation in these graphs boils down to linear algebra i.e. vector/matrix operations, these operations provide lot of scope for parallelisation and almost all models are trained on GPUs as a result. As linear algebra occupies such an important role, the performance of any deep learning framework depends on how well it can optimise these calculations(which is done using XLA in linear algebra), therefore XLA is very crucial to the performance of TensorFlow.

2. What is JIT compilation? What other compilation modes are there in TensorFlow?

   Ans. JIT compilation stands for Just-in-time compilation, it is a way to run programs in which compilation is done at run-time i.e. not before execution starts. With additional information during runtime about the state, this kind of compilation offers more scope for optimization. TensorFlow also offers AOT compilation, i.e. Ahead of Time compilation, using this it is possible to generate binaries of tensorflow graphs during before execution of the program.

3. What are the performance metrics that are important that a compiler writer needs to focus on?

   Ans. A compiler writer need to focus on the following performance metrics:

   (a) Correctness of the program is perhaps the most important metric, any compiler must always produce correct target machine code.

   (b) Execution Time of the program i.e. faster the execution of the program, the better.

   (c) Speed of compilation, it should not take too long to compile programs.

(d) Memory Usage by the executable should be effecient since primary memory is a limited resource.

(e) The front-end, back-end of the compiler should be such that:

   i. For one target machine, the same back-end should be usable with different source languages.

   ii. Front-end for one language should be compatible with back-ends of different target machines.