Compilers-I
CS 3510 Spring 2019
Reading Assignment 1:
TensorFlow/XLA and JIT

INSTRUCTOR: DR. Ramakrishna Upadrasta

By:
Vijay Tadikamalla
CS17BTECH11040

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

1. **What is XLA? Why is it so crucial to the performance of TensorFlow?**
   XLA(Accelerated Linear Algebra) is a compiler for TensorFlow graphs that one can use to accelerate their TensorFlow ML models today with a minimal source code change.
   XLA is crucial to the performance of TensorFlow because
   - It is a domain-specific compiler for linear algebra that optimizes TensorFlow computations.
   - XLA comes with several optimizations and analysis passes that are target-independent, such as CSE(Common subexpression elimination), target-independent operation fusion, and buffer analysis for allocating runtime memory for the computation.
   - XLA GPU backend generally performs operation fusion beneficial specifically for the GPU programming models and determines how to partition the computation into streams.

2. **What is JIT compilation? What other compilation modes are there in TensorFlow?**
   Just-in-time (JIT) compilation is a way of executing computer code that involves compilation during execution of a program at run time rather than prior to execution. A JIT compiler runs after the program has started and compiles the code (usually bytecode) on the fly into a form that's usually faster. A JIT has access to dynamic runtime information whereas a standard compiler doesn't and can make better optimizations like inlining functions that are used frequently. This is in contrast to a traditional compiler that compiles all the code to machine language before the program ran.

   Compilation modes in TensorFlow:

   - JIT compilation
   - AOT(Ahead of time) compilation

3. **What are the performance metrics that are important that a compiler writer needs to focus on? (Hint: Why was XLA built?)**
   - **Execution speed:** The most important performance metric of a compiler is the amount of time consumed by the program. So ideal compiler improves the execution speed of the program with aggressive optimizations without changing the result of the program.

- **Memory usage:** A ideal compiler should analyze and schedule memory usage, in principle eliminating many intermediate storage buffers.
- **Power consumption:** This is a crucial metric for determining compiler performance which also depends on other factors like execution speed, memory usage of the program etc.
- **Compilation time:** A compiler should compile the code in the minimum time possible but must also maximize the optimizations without changing the result of the program.
- **Output file size:** The size of the output file matters a lot on a system with a low amount of available memory. Example, an embedded system etc.