Write-and-Do-Mini-Assignment#3: MLIR and LLVM

Sai Harsha Kottapalli
CS17BTECH11036

About MLIR in LLVM

MLIR is going to join LLVM foundation as a sub-project. This serves as a base compiler infrastructure to provide a unified, flexible and extensible intermediate representation which is language independent. LLVM IR is set as the main code generation target.
The integration would not pose too much of a hassle as MLIR relies heavily on design principles and practices developed by the LLVM community, hence, allowing contributors to easily get adapted.
LLVM-based languages like Swift had to develop their IRs (SIL), because many optimizations used in those languages cannot be done in LLVM. MLIR could provide a standard way to express those optimizations, which could in turn be re-used for other languages, especially in c++.
One of the main motivations which led to this that the IR can become a common format between ML models/frameworks and that the loss of information is mitigated through the integration of MLIR.
MLIR can also allow parallel compilation where it divides the code and works on segments of code in parallel.
MLIR has this notion of dialects which are defined operations with invariants. We can make multiple dialects to address different levels of the same model.
Some hardware might support some of those defined operations so instead of constructing it again from basic operations from LLVM IR this can be resolved in MLIR step itself.
Though there is the disadvantage of significant increase in compilation time.

Implications of MLIR for FORTRAN.

Flang is the compiler for fortran which gives IR for fortran after applying fortran specific optimizations which usually can’t be done via LLVM. Integration of MLIR into flang lets us use MLIR as a compiler infrastructure which makes it a lot easier to write optimizations and translation to LLVM IR. This will also reduce the compilation time as MLIR will be supported by LLVM and is very compatible. Here, MLIR’s “dialect” plays an important role as it allows easy integration of MLIR into flang.
Hence, currently though it will still be under development, after successful integration it would be beneficial for developers.