MLIR 编译框架的使用与探索

编译原理课程大作业

上海交通大学计算机系

1　背景简介

1.1　MLIR

MLIR 的全称为 Multi-Level Intermediate Representation，即“多层中间表示”。它本质上是一种全新的编译基础设施，提供了一种构建可重用和可扩展编译器基础结构的新方法。MLIR 的目标是解决软件碎片化的问题，降低异构计算的编译难度，降低领域专用编译器和加速器的构建成本，并协助将现有优秀的编译器相关工作连接在一起。近年来，越来越多基于 MLIR 的工作出现在计算机体系结构、系统和编译领域的顶会上。更多关于 MLIR 的具体信息可参考 MLIR 官方文档。

1.2　Pony 语言

Pony 语言是本次大作业的使用语言。它是一种自定义的、基于张量（tensor）的语言。Pony 语言支持少量功能，包括函数定义、基本数学运算以及打印功能。以下是一个具体实例：

```
def main() {
  var a = [[1, 2, 3], [4, 5, 6]];
  var b<2, 3> = [1, 2, 3, 4, 5, 6];
  print(transpose(a) * transpose(b));
}
```

图 1. Pony 语言实例

该实例中定义了两个 tensor 变量 a 和 b。a 的 shape 隐式定义为 <2,3>，可以由后面的赋值推断出来。b 的 shape 显示定义为 <2,3>，并同样进行了初始化赋值。可以看到，Pony 目前支持两种不同的对 tensor 的赋值方式。transpose() 函数执行矩阵转置运算，print() 函数进而输出这两个转置后的矩阵点乘（Element-wise Multiplication）的结果。两者都为 Pony 的内置函数，可以直接调用。目前 Pony 语言只支持一维和二维张量、64 位浮点数 (C 语言中的 double) 数据类型以及简单的加法和点乘运算。在实验过程中，同学们不必详细了解 Pony 语言的内置函数功能和具体实现，我们的重点在于从编译的角度去搭建一个小型编译器，从而解析这样一种语言。

2　大作业内容简介

在本次大作业中，我们将基于 MLIR 编译框架，搭建一个简单的编译器，使其可以编译基于 Pony 语言的代码。本次大作业共分为三个阶段：词法分析（6 分），语法分析（8 分），和代码优化与生成（6 分），合计 20 分。我们将分阶段发布各部分内容和要求，请大家独立完成并按要求提交。

（1）词法分析：在该阶段，我们要构建一个词法分析器来识别 Pony 语言中的各种词法单元 (Token)，包括关键字（如 var、def 和 return）、特殊符号、数字以及变量名/函数名等。我们要通过相关函数来获取 Token，判断其合法性，并针对非法格式输出错误信息。

（2）语法分析：在词法分析之后，我们需要构建一个语法分析器，将获得的词法单元序列构建成一棵抽象语法分析树 (AST)。具体包括解析函数的声明和调用，Tensor 变量的声明以及 Tensor 的二元运算表达式等，并针对非法格式输出错误信息。

（3）代码优化与生成：最后，我们将进一步实现代码优化和生成。该部分包括基础与进阶两个部分。基础部分共 2 分，要求以 Pony 语言内置函数 transpose() 为例，实现冗余代码的消除，并依据要求对优化效果进行验证。进阶部分共 4 分，要求在 Pony 语言中新增对普通矩阵乘法运算的支持，并对最终结果进行正确性验证。

3　实验环境及安装

1）首先通过 git 将大作业代码仓库克隆至本地

```
$ git clone https://github.com/Jason048/pony_compiler.git
```

由于 MLIR 与 LLVM 安装复杂，我们提供一个 Docker 镜像作为实验平台，同学们可以参考这个简单教程了解 Docker 的基本使用。

2）安装 Docker 后，通过以下命令拉取实验镜像，并启动相应容器，请注意将下面的 `REPO_PATH` 替换为本地大作业仓库的路径

```
$ docker pull jason0048/pony_compiler:2024
$ docker run -it -v REPO_PATH:/home/workspace/pony_compiler \
  --name pony jason0048/pony_compiler:2024
```

若由于关机等原因造成命令行退出，可以使用 `docker ps` 命令查看 `pony` 容器是否存在，如果存在，可以使用 `docker attach pony` 重新连接，如果不存在，可以使用 `docker start -ai pony` 重启容器。

3）通过下面的命令 build 大作业项目

```
$ cd /home/workspace/pony_compiler/
$ mkdir build && cd build
$ cmake .. -B .
$ cmake --build . --target pony
$ ./bin/pony --help
```