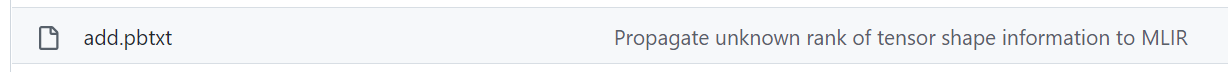
基于Tensorflow的mlir流程

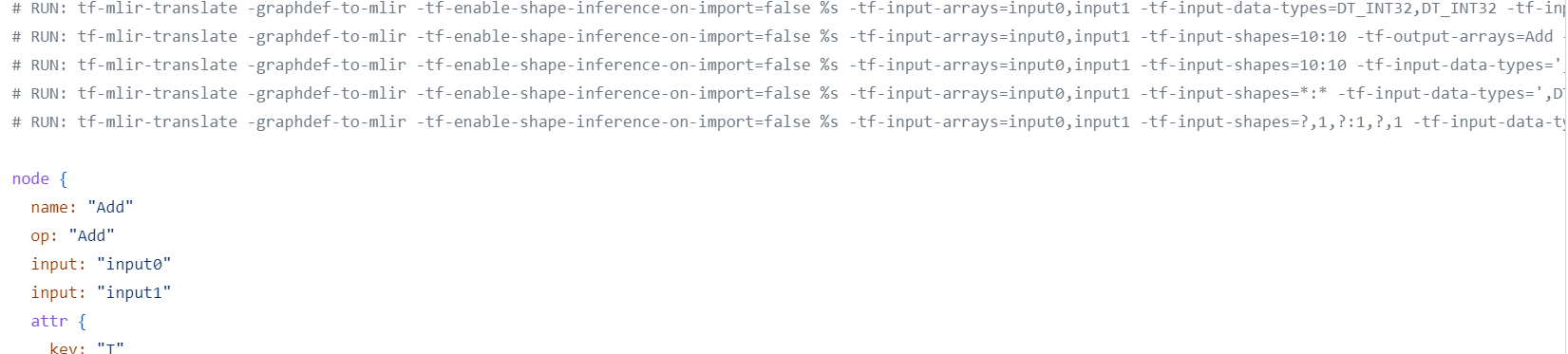
两种方式：第一种端到端，还有一种保留MLIR构建工具，修补其他链接工具

第一步：对于graph->mlir,这一步是将.pbtxt文件转化成mlir，但这个.mlir文件的方言不是MLIR基本的方言。

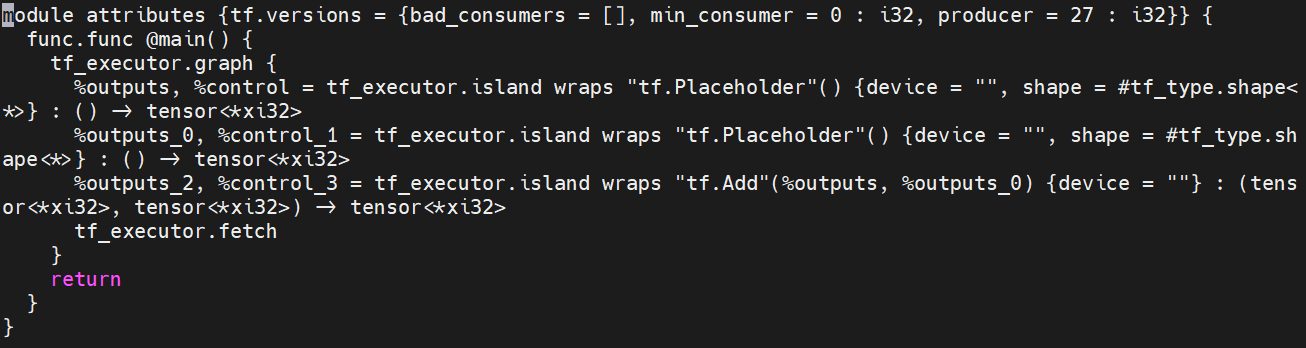
case的运行详情参阅

[https://github.com/tensorflow/tensorflow/tree/master/tensorflow/compiler/mlir/tensorflow/tests/graphdef2mlir](https://github.com/tensorflow/tensorflow/tree/master/tensorflow/compiler/mlir/tensorflow/tests/graphdef2mlir)

里面有详细的命令描述

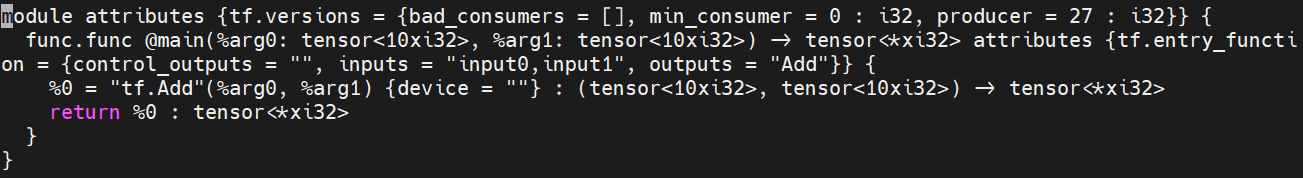


得到的mlir的结果如下：



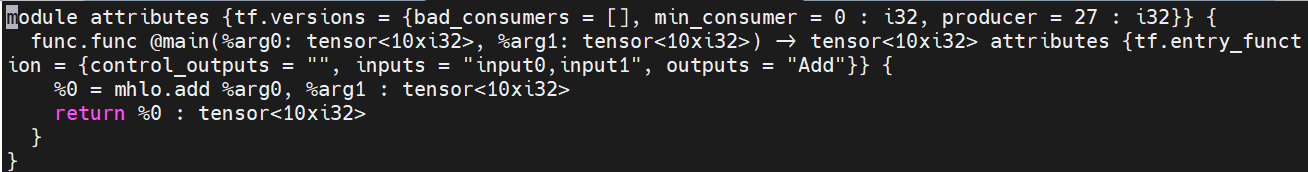
第二步：./tf-opt --tf-executor-to-functional-conversion add.mlir -o add-func.mlir，将基于图的ir，tf\_executor转化成mlir（三地址码）。

例子在<https://github.com/tensorflow/tensorflow/blob/master/tensorflow/compiler/mlir/tensorflow/tests/tf-executor-to-functional.mlir>



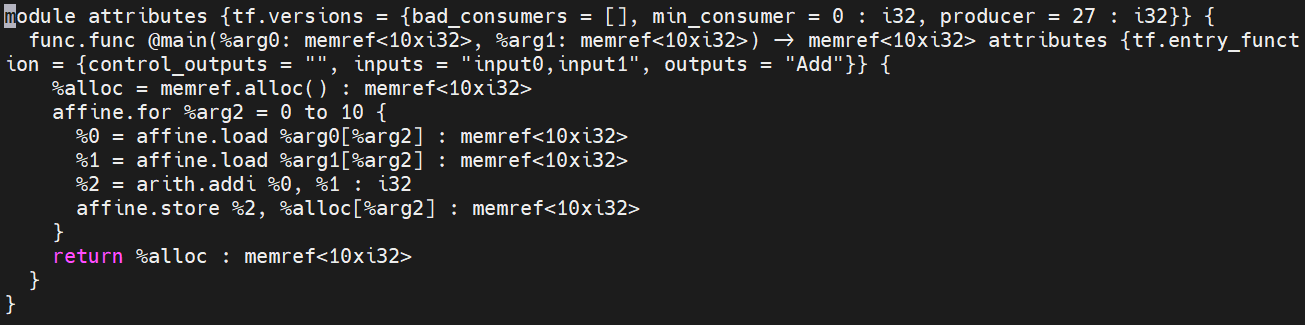
第三步：./tf-opt --tf-to-hlo-pipeline add-func.mlir -o add-mhlo.mlir，进一步下降，为了链接MLIR社区里面的mlir。

<https://github.com/tensorflow/tensorflow/blob/master/tensorflow/compiler/mlir/tensorflow/tests/tf_to_hlo_pipeline/sccp-post-shape-inference.mlir>



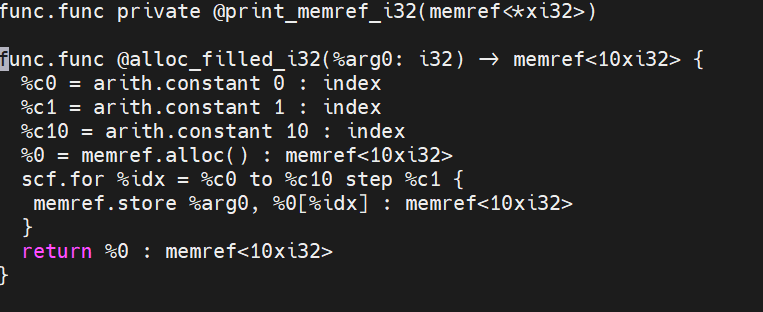
第四步：./mlir-hlo-opt --hlo-legalize-to-linalg 将mhlo转化成linalg dialect,

<https://github.com/tensorflow/mlir-hlo/blob/master/tests/Dialect/mhlo/hlo-legalize-to-linalg.mlir>



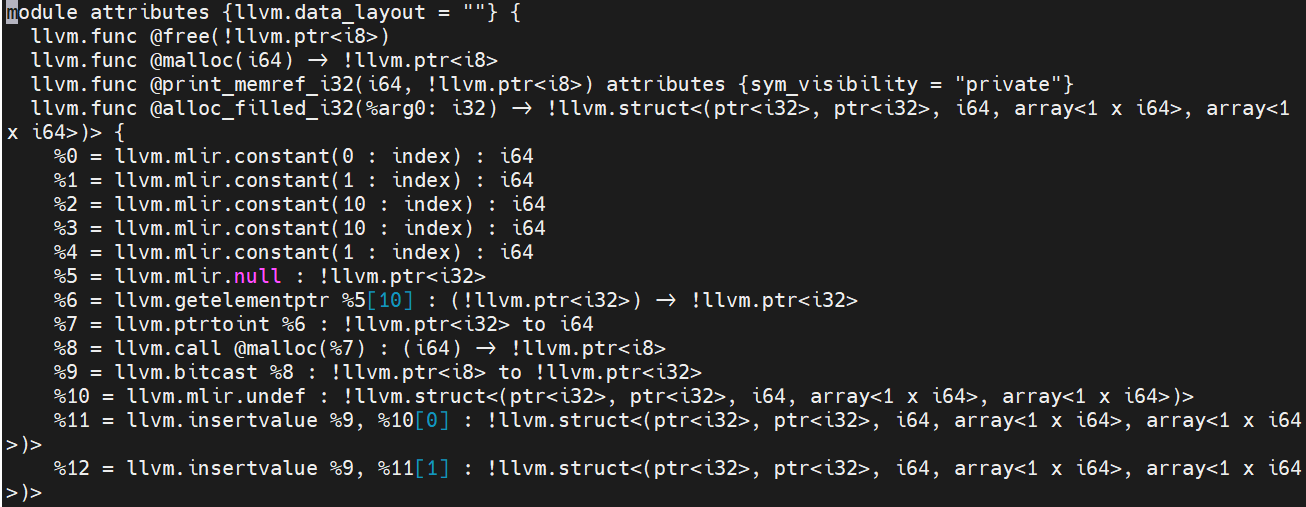
第五步，修改linalg mlir,添加输入模块，（建议构建llvm-project时用debug模式）

具体参考https://github.com/llvm/llvm-project/tree/main/mlir/test/Integration/Dialect/Linalg/CPU



第六步，./mlir-opt -convert-linalg-to-loops -convert-scf-to-cf -convert-linalg-to-llvm -convert-memref-to-llvm -convert-func-to-llvm -reconcile-unrealized-casts add-linalg.mlir -o add-llvm.mlir，这一步的目的是lowering到llvm mlir，

详情参阅：<https://github.com/llvm/llvm-project/blob/main/mlir/test/Integration/Dialect/Linalg/CPU/matmul-vs-matvec.mlir>



第七步，./mlir-cpu-runner -e main -entry-point-result=void -shared-libs=/home/zhaojie/llvm-project/build/lib/libmlir\_runner\_utils.so llvm.mlir，运行结果，出现问题

JIT session error: Symbols not found: [ print\_memref\_i32 ]  
Error: Failed to materialize symbols: { (main, { \_mlir\_main, alloc\_filled\_i32, \_mlir\_alloc\_filled\_i32, main })

基于tensorflow的iree构建流程：

安装链接：

https://iree-org.github.io/iree/getting-started/tensorflow/#support-matrix

第一步下载tensorflow工具

python -m pip install tf-nightly

第二步下载iree工具

python -m pip install \

iree-compiler \

iree-runtime \

iree-tools-tf

第三步，导入model并产生signature,

运行如下代码：

import tensorflow.compat.v2 as tf

loaded\_model = tf.saved\_model.load('/path/to/downloaded/model/')

print(list(loaded\_model.signatures.keys()))

编译savedmodel成mlir,

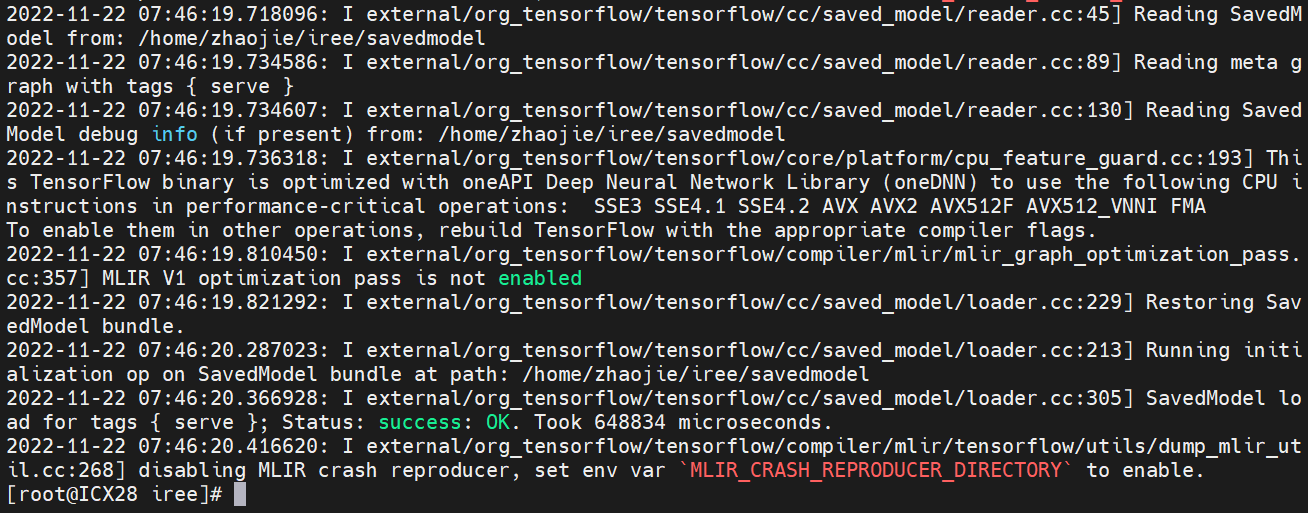
iree-import-tf

--tf-import-type=savedmodel\_v1 \

--tf-savedmodel-exported-names=predict \

/home/Zhaojie/iree/savedmodel -o iree\_input.mlir

在这一步，运行结果错误：



接下来的是没有运行的步骤

第四步，进一步编译iree\_input.mlir

iree-compile \

--iree-hal-target-backends=llvm-cpu \

--iree-input-type=mhlo \

iree\_input.mlir -o mobilenet\_cpu.vmfb

第五步，运行mobilenet\_cpu.vmfb

tools/iree-run-module \

--device=local-task \

--module\_file=mobilenet\_cpu.vmfb \

--entry\_function=predict \

--function\_input="1x224x224x3xf32=0"