Installation Instructions

1 TVMFUZZ

To release reviewers from laborious tasks of building experimental environments, we have created a docker image in which TVMfuzz, TVM0.7 and TVM0.8 are all deployed and pushed it to docker hub.

You can download the image and reproduce our experiments about TVMfuzz in the following steps:

1) Input the following commands:

```
docker pull mhypony/tvmfuzz:latest
docker run -it mhypony/tvmfuzz:latest /bin/bash
```

2) Now you are in our docker container, to eliminate unexpected situations that may corrupt our experiment, you should input the following two commands to activate the environment including TVM0.7

```
    source /etc/profile
    source activate
```

3) Now go to the folder called **DLCstudy** and run the program by the commands.

```
    python run.py
```

4) Finally, you can check the generated program (**program.py**) in the folder named **byproduct**.

If you want to compare execution message given by two TVM branches, you can follow the steps below:

1) Run `program.py` in `/DLCstudy/byproduct` under the default tvm version, 0.7.

2) Change the default tvm version specified in `/etc/profile` by replace `export TVM_HOME=/tvm0.7` to `export TVM_HOME=/tvm0.8` and type the following commands:

```
    source /etc/profile
    source activate
```

3) Run `program.py` in `/DLCstudy/byproduct` under the current default tvm version, 0.8 and compare two execution messages.

It's possible that in both two environments, nothing is outputted because our generated `program.py` is totally valid for both TVM branches. If two TVM branches output inconsistently, e.g. they provide different warning messages or one outputs something while the other does not, you may encounter a bug in one of the two branches. The 8 bugs we mentioned in our paper were found in this way, that is **searching detectable output inconsistency**. For instance, `1.py` in `buggyFile` folder was found because nothing was printed on screen under TVM0.8 but warning message1 was outputted under TVM0.7 and it was confirmed as a bug-triggered code snippet for TVM0.7.

To make it more clear, the program synthesized by TVMfuzz is not guaranteed to be logically correct. The major purpose of the synthesized program is to find the different behaviors among different versions of TVM. Thus, TVMfuzz may generate a logically incorrect program and expect the consistent warning messages given by several TVM branches. For instance, TVMfuzz intentionally requires the shapes before and after `reshape` operation are inconsistent and expect consistency among warning messages from different branches.

```
Traceback (most recent call last):
  File "/DLCstudy/buggyFile/1.py", line 8, in <module>
    xgupP=tvm.IRModule.from_expr(pDL7V)
  File "/tvm0.7/python/tvm/ir/module.py", line 237, in from_expr
    return _ffi_api.Module_FromExpr(expr, funcs, defs)
  File "/tvm0.7/python/tvm/_ffi/_ctypes/packed_func.py", line 237, in __call__
    raise get_last_ffi_error()
tvm._ffi.base.TVMError: Traceback (most recent call last):
  [bt] (8) /tvm0.7/build/libtvm.so(tvm::RelayExpr tvm::relay::TypeInferencer::Resolver::AttachCheckedType<tvm::relay::FunctionNode>(tvm::relay::FunctionNode const*)+0x16a) [0x7f7beff046ba]
  [bt] (7) /tvm0.7/build/libtvm.so(tvm::relay::ExprMutator::VisitExpr_(tvm::relay::FunctionNode const*)+0x59e) [0x7f7beffc4fee]
  [bt] (6) /tvm0.7/build/libtvm.so(tvm::relay::ExprMutator::VisitExpr(tvm::RelayExpr const&)+0x96) [0x7f7beffc7a76]
  [bt] (5) /tvm0.7/build/libtvm.so(tvm::relay::ExprFunctor<tvm::RelayExpr (tvm::RelayExpr const&)>::VisitExpr(tvm::RelayExpr const&)+0x89) [0x7f7befa53349]
  [bt] (4) /tvm0.7/build/libtvm.so(tvm::relay::ExprFunctor<tvm::RelayExpr (tvm::RelayExpr const&)>::InitVTable()::{lambda(tvm::runtime::ObjectRef const&, tvm::relay::ExprFunctor<tvm::RelayExpr (tvm::RelayExpr const&)>*)#3}::_FUN(tvm::runtime::ObjectRef const&, tvm::relay::ExprFunctor<tvm::RelayExpr (tvm::RelayExpr const&)>*)+0x27) [0x7f7befa4a427]
  [bt] (3) /tvm0.7/build/libtvm.so(tvm::relay::TypeInferencer::Resolver::VisitExpr_(tvm::relay::VarNode const*)+0x7f7) [0x7f7beff07a87]
  [bt] (2) /tvm0.7/build/libtvm.so(tvm::relay::TypeInferencer::Resolver::VisitVar(tvm::relay::Var const&)+0xc3) [0x7f7beff077d3]
  [bt] (1) /tvm0.7/build/libtvm.so(tvm::RelayExpr tvm::relay::TypeInferencer::Resolver::AttachCheckedType<tvm::relay::VarNode>(tvm::relay::VarNode const*)+0x14e) [0x7f7beff01cee]
  [bt] (0) /tvm0.7/build/libtvm.so(dmlc::LogMessageFatal::~LogMessageFatal()+0x61) [0x7f7bef497661]
  File "/tvm0.7/src/relay/transforms/type_infer.cc", line 611
TVMError: Check failed: checked_type.as<IncompleteTypeNode>() == nullptr: Cannot resolve type of Var(x) at (nullptr)
```

Figure 1: Warning message provided by TVM0.7

2 PLOTTING

To reproduce the figures based on the dataset in our paper, a drawing script is provided.

You can download the script and reproduce the figures in our paper in the following steps.

1) Download the **dataset** folder from our GitHub repository[1]. It contains a drawing script and the dataset file.

2) Before run the script, you need install related packages required by R language by the commend.

```
    install.packages(c("ggplot2", "readxl", "plyr", "
     patchwork", "corrplot"))
```

3) Go to the folder **dataset** and run the script (**drawing_script.R**) by R language IDE. RStudio(version 4.0.3) is recommended.

Notes:

1. The dataset file(dataset.xlsx) should be placed in the same directory as the drawing_script.R file.

2. If the running crash with a message "'path' does not exist: 'dataset.xlsx'", you need set the working directory to source file location.

3. Since not all the generated figures are saved locally, please run the script line by line so that you will see the generated figures in the IDE one by one.

[1] https://github.com/anonymousWork000/DLCstudy/tree/master/dataset