香港中文大學
The Chinese University of Hong Kong

CMSC5743 Lab 05
TVM Tutorial-1 Materials

Yang BAI
Department of Computer Science & Engineering
Chinese University of Hong Kong
ybai@cse.cuhk.edu.hk

November 26, 2021

Outline

TVM Installation

TVM Installation

- Recommended System: MAC OS or Linux.
- LLVM 9.0+
- git clone –recursive https://github.com/apache/tvm tvm
- mkdir build
- cp cmake/config.cmake build
- open the LLVM option
- cmake ..
- make -j10

TVM Environment Setup

- vim ~/.bashrc or ~/.bash_profile
- export TVM_HOME=/Users/baiyang/Documents/Project/tvm
- export PYTHONPATH=$TVM_HOME/python:$PYTHONPATH

Verify the correct installaltion

- import tvm
- tvm.__version__

Matrix Multiplication by TVM

Matrix Multiplication by TVM

- Defining the Matrix Multiplication
- Create the search task
- Set Parameters for Auto-Scheduler
- Run the search
- Inspecting the Optimized Schedule
- Check correctness and evaluate performance

Defining the Matrix Multiplication

To start, we define a matrix multiplication with a bias addition. Note that this uses standard operations available in TVMs Tensor Expression language. The major difference is the use of the auto_sceduler decorator at the top of the function definition. The function should return a list of input/output tensors. From these tensors, the auto-scheduler can get the whole computational graph.

Create the Search Task

With the function defined, we can now create the task for the auto_scheduler to search against. We specify the particular parameters for this matrix multiplication, in this case a multiplication of to square matricies of size 1024×1024. We then create a search task with N=L=M=1024 and dtype =“float32”.

Set Parameters for Auto-Scheduler

- num_measure_trials is the number of measurement trials we can use during the search. We only make 10 trials in this tutorial for a fast demonstration. In practice, 1000 is a good value for the search to converge. You can do more trials according to your time budget.
- In addition, we use RecordToFile to log measurement records into a file matmul.json. The measurement records can be used to query the history best, resume the search, and do more analyses later.
- see auto_scheduler.TuningOptions for more parameters

Inspecting the Optimized Schedule

We can lower the schedule to see the IR after auto-scheduling. The auto-scheduler correctly performs optimizations including multi-level tiling, layout transformation, parallelization, vectorization, unrolling, and operator fusion.

Run the Search

Now we get all inputs ready. Pretty simple, isn't it? We can kick off the search and let the auto-scheduler do its magic. After some measurement trials, we can load the best schedule from the log file and apply it.

Homework

Homwork

- Change the `N, L, M` dimensions of Matrix in /code/main.py
- Change the `num_measure_trials` for the search tasks
- Record the final prediction of your change and analysis the reason

THANK YOU!