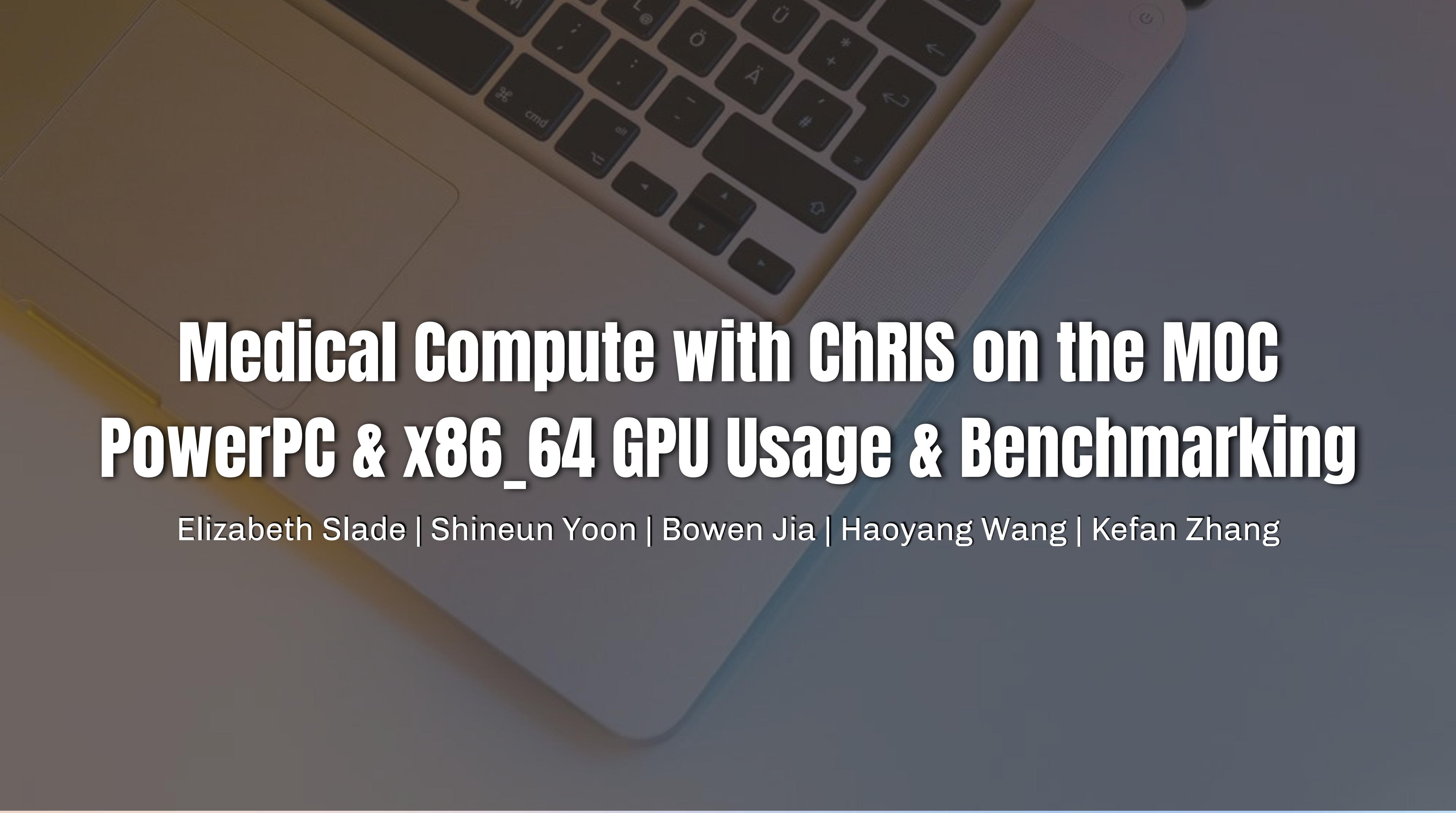


Medical Compute with ChRIS on the MOC PowerPC & x86_64 GPU Usage & Benchmarking

Elizabeth Slade | Shineun Yoon | Bowen Jia | Haoyang Wang | Kefan Zhang

What is ChRIS?

- An open source platform for medical analysis
- The goal is to democratize application development for medical analysis applications

How are we connected to ChRIS?

- Our project is to develop a ChRIS plugin that tests the performance on different architectures like x86 and PowerPC
- Target Users: ChRIS architects and developers of the platform to test their app

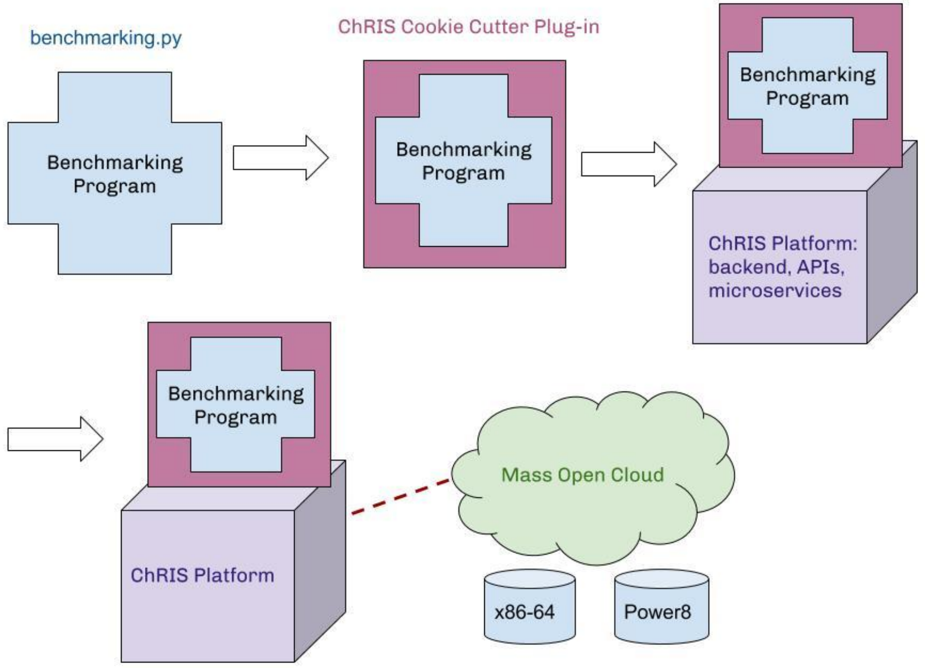
What we’ve Done

Plugin Implementation in progress

Setting up the CUBE environment

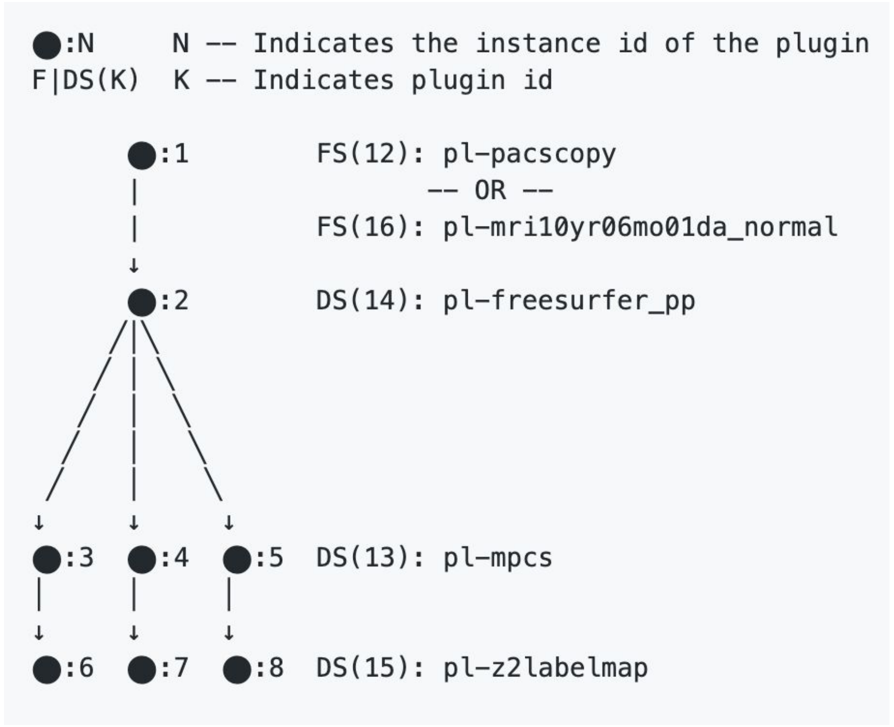
Ubuntu Virtual Machine deployed on OpenStack

The ChRIS Pipeline



Chris Cookie-Cutter Plug-in

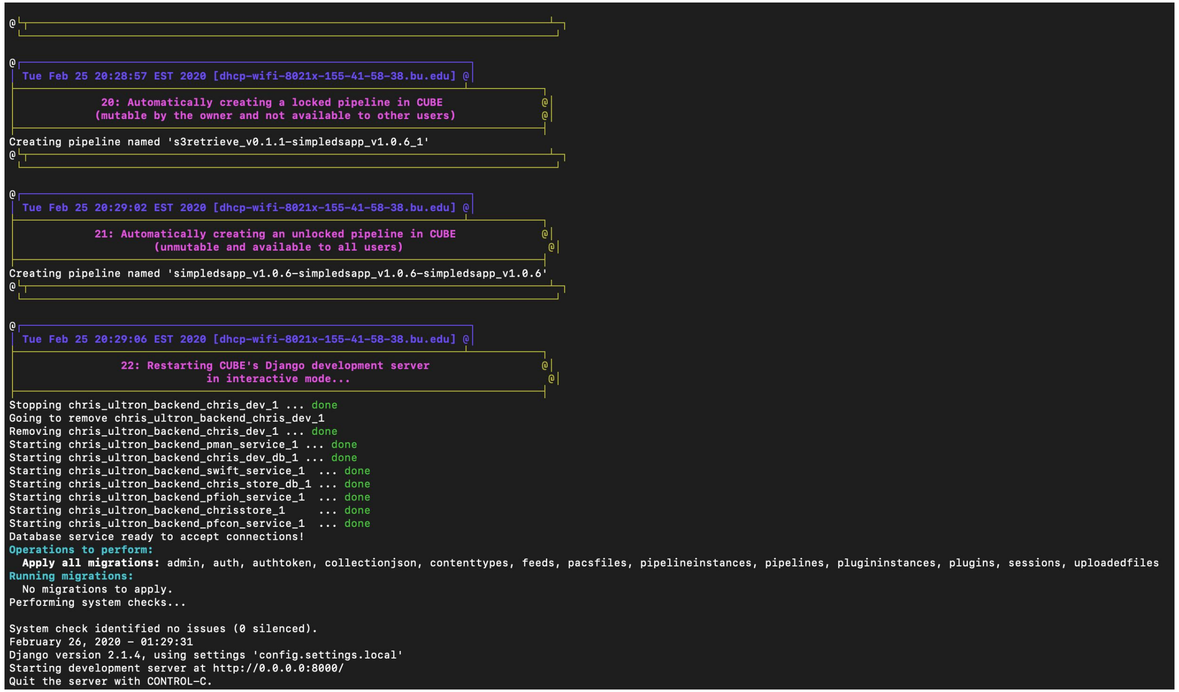
- Cookie-Cutter
- Two types of ChRIS Plugin
  - FS = Feed Synthesis
  - DS = Data Synthesis



```
●:N      N -- Indicates the instance id of the plugin
F|DS(K)  K -- Indicates plugin id

      ●:1         FS(12): pl-pacscopy
      |                     -- OR --
      |           FS(16): pl-mri10yr06mo01da_normal
      ↓
      ●:2         DS(14): pl-freesurfer_pp
     /|\
    / | \
   /  |  \
  /   |   \
 /    |    \
↓     ↓     ↓
●:3   ●:4   ●:5   DS(13): pl-mpcs
|     |     |
↓     ↓     ↓
●:6   ●:7   ●:8   DS(15): pl-z2labelmap
```

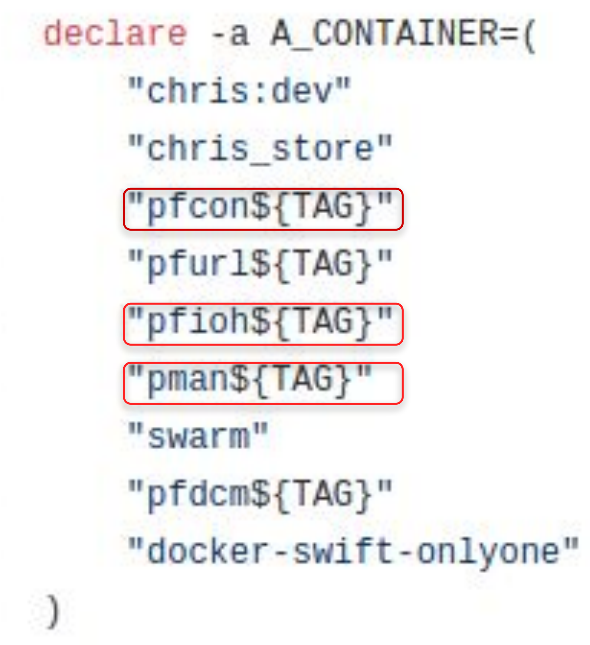
Running CUBE

- **CUBE** = **C**hRIS **U**ltron **B**ack**E**nd



```
@

@
 Tue Feb 25 20:28:57 EST 2020 [dhcp-wifi-8021x-155-41-58-38.bu.edu] @

              20: Automatically creating a locked pipeline in CUBE          @
             (mutable by the owner and not available to other users)        @

Creating pipeline named 's3retrieve_v0.1.1-simpledsapp_v1.0.6_1'
@

@
 Tue Feb 25 20:29:02 EST 2020 [dhcp-wifi-8021x-155-41-58-38.bu.edu] @

             21: Automatically creating an unlocked pipeline in CUBE        @
                     (unmutable and available to all users)                  @

Creating pipeline named 'simpledsapp_v1.0.6-simpledsapp_v1.0.6-simpledsapp_v1.0.6'
@

@
 Tue Feb 25 20:29:06 EST 2020 [dhcp-wifi-8021x-155-41-58-38.bu.edu] @

                 22: Restarting CUBE's Django development server            @
                              in interactive mode...                         @

Stopping chris_ultron_backend_chris_dev_1 ... done
Going to remove chris_ultron_backend_chris_dev_1
Removing chris_ultron_backend_chris_dev_1 ... done
Starting chris_ultron_backend_pman_service_1 ... done
Starting chris_ultron_backend_chris_dev_db_1 ... done
Starting chris_ultron_backend_swift_service_1  ... done
Starting chris_ultron_backend_chris_store_db_1 ... done
Starting chris_ultron_backend_pfioh_service_1  ... done
Starting chris_ultron_backend_chrisstore_1     ... done
Starting chris_ultron_backend_pfcon_service_1  ... done
Database service ready to accept connections!
Operations to perform:
  Apply all migrations: admin, auth, authtoken, collectionjson, contenttypes, feeds, pacsfiles, pipelineinstances, pipelines, plugininstances, plugins, sessions, uploadedfiles
Running migrations:
  No migrations to apply.
Performing system checks...

System check identified no issues (0 silenced).
February 26, 2020 - 01:29:31
Django version 2.1.4, using settings 'config.settings.local'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

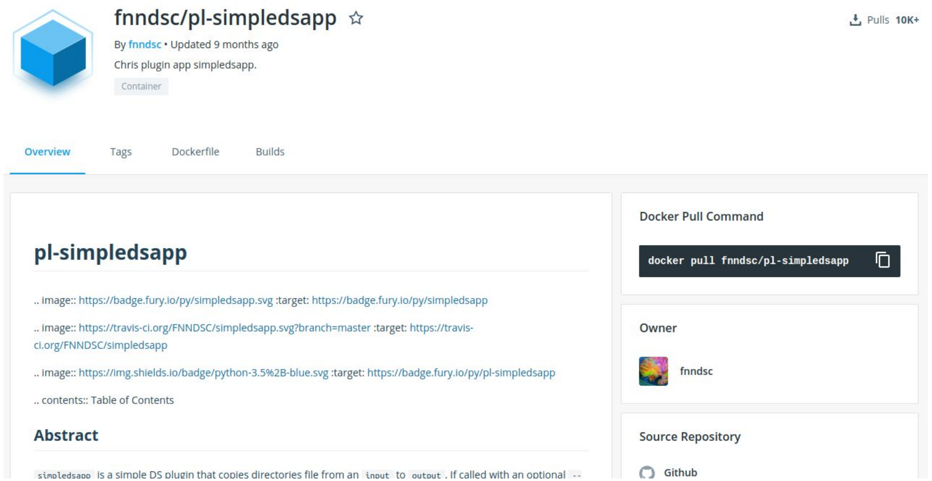
Plugin list



```
declare -a A_CONTAINER=(
    "chris:dev"
    "chris_store"
    "pfcon${TAG}"
    "pfurl${TAG}"
    "pfioh${TAG}"
    "pman${TAG}"
    "swarm"
    "pfdcm${TAG}"
    "docker-swift-onlyone"
)
```

Plugin Build

- Based on Cookie-cutter
- Successfully built
- Future work:
  - plugin code file
  - Requirements.txt
  - Setup.py
  - Dockerfile
- Test on Titan machine



fnndsc/pl-simpledsapp

By fnndsc • Updated 9 months ago

Chris plugin app simpledsapp.

Container

Pulls 10K+

Overview Tags Dockerfile Builds

pl-simpledsapp

.. image:: https://badge.fury.io/py/simpledsapp.svg :target: https://badge.fury.io/py/simpledsapp

.. image:: https://travis-ci.org/FNNDSC/simpledsapp.svg?branch=master :target: https://travis-ci.org/FNNDSC/simpledsapp

.. image:: https://img.shields.io/badge/python-3.5%2B-blue.svg :target: https://badge.fury.io/py/pl-simpledsapp

.. contents:: Table of Contents

Abstract

`simpledsapp` is a simple DS plugin that copies directories file from an `input` to `output`. If called with an optional --

Docker Pull Command

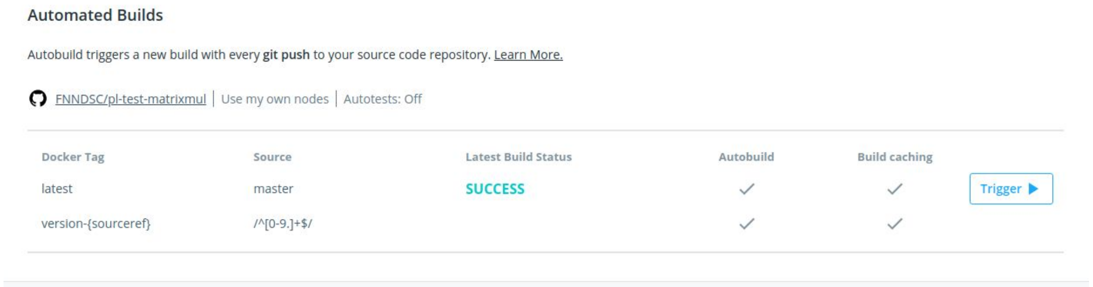
```
docker pull fnndsc/pl-simpledsapp
```

Owner

fnndsc

Source Repository

Github



Automated Builds

Autobuild triggers a new build with every **git push** to your source code repository. Learn More.

FNNDSC/pl-test-matrixmul | Use my own nodes | Autotests: Off

Docker Tag	Source	Latest Build Status	Autobuild	Build caching	
latest	master	SUCCESS	✓	✓	Trigger ▶
version-{sourceref}	/^[0-9.]+$/		✓	✓	

Mass Open Cloud ChRIS App Deployment

- RedHat Interactive Learning Portal
  - deployed a world map image with national park data as an overlay

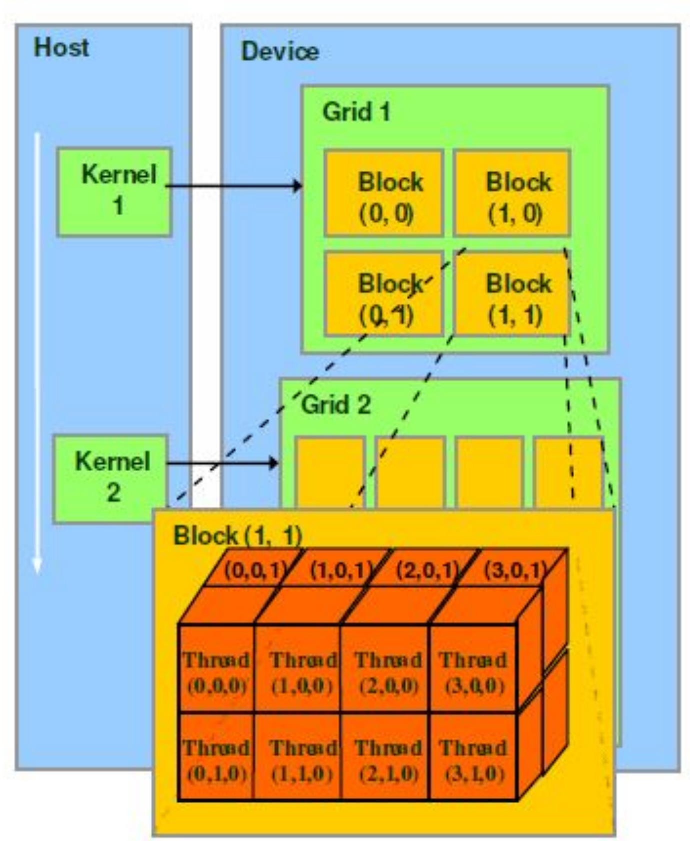
Mass Open Cloud ChRIS App Deployment

- Created a router, security group, key pair and Ubuntu VM on OpenStack
  - We can ssh into Ubuntu VM

Meeting with Red Hat Contact

- Best to create my own environment on our project instance
    - Deploy pman and pfioh on Openstack to communicate with ChRIS's CUBE

How GPU computing works



One Graphic Card is able to own 2~3 Grids

A Grid could be separated to many blocks.(Multi-processors)

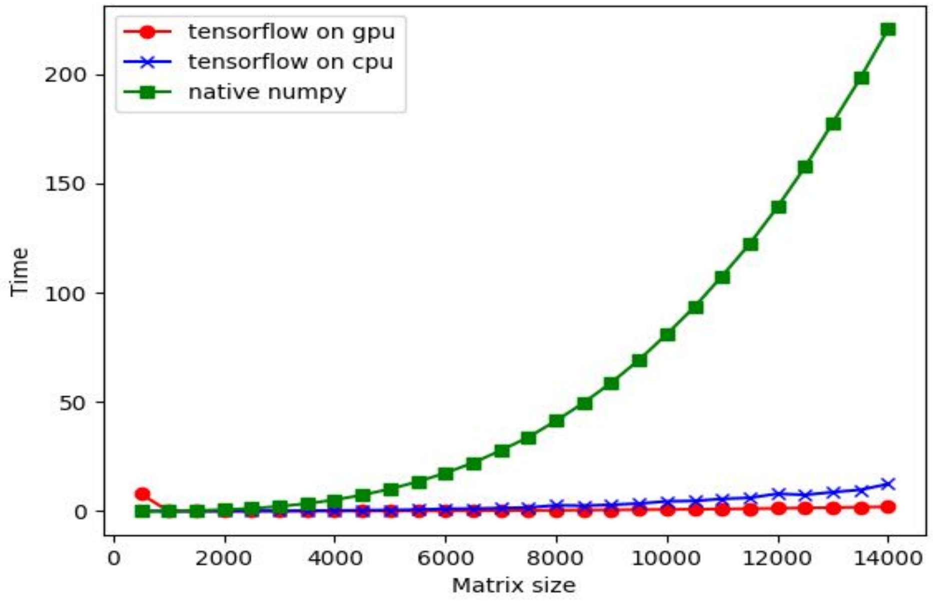
Every blocks has it shared memory (different from GPU memory).

Huge number of thread could run on one block.(multi-cores)

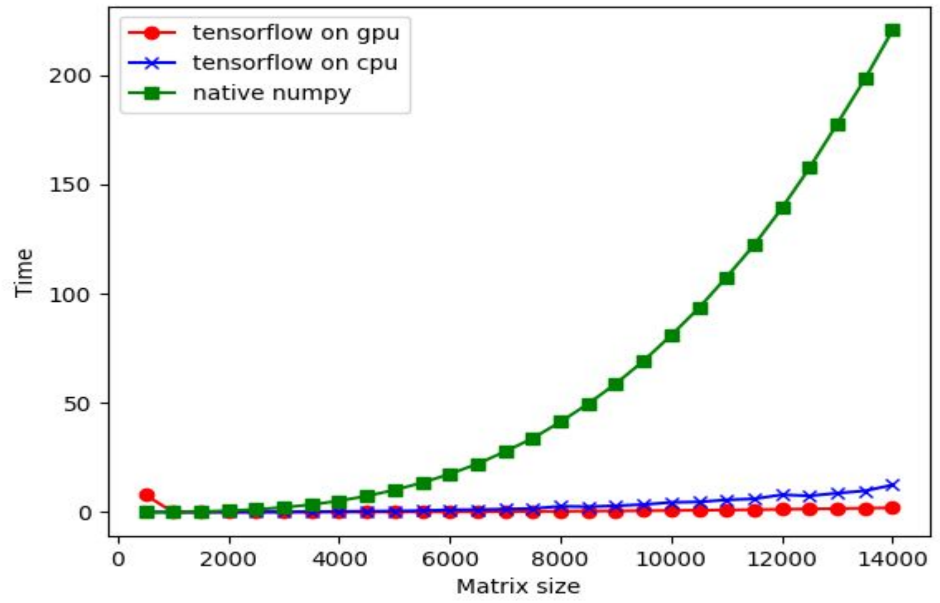
Different between Power8 and X86_64

	Power 8	X86_64
CPU connection to GPU	NvLink Supported	PCI-E 3.0 x16 (typical)
Bandwidth between GPUs	80GB/s	16 GB/s(typical)
Cores	1 Core 8 Threads	1 Core 2 Threads (typical)

Visualizing Results

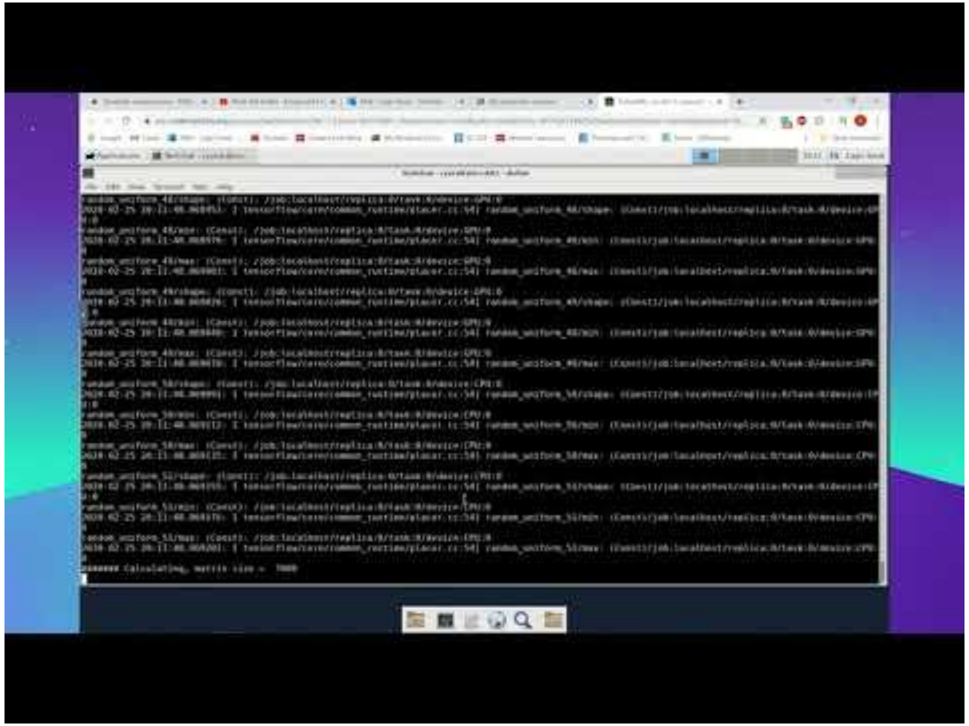


Visualizing Results



- Tested on x86 platform
- Collected executing time for GPU vs CPU
- Next step: Implement on PowerPC

Visualizing Results

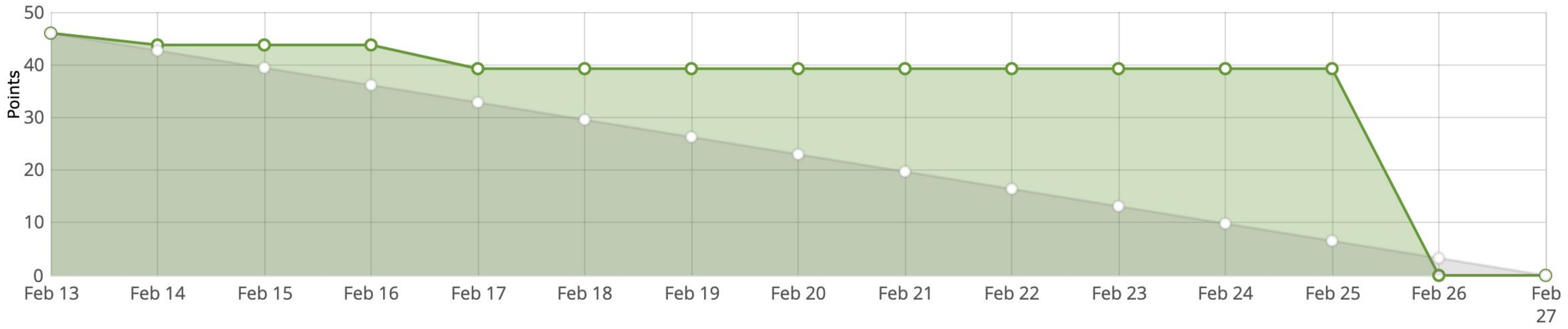


```
random_uniform_48/shape: (Const): /job:localhost/replica:0/task:0/device:GPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_48/shape: (Const)/job:localhost/replica:0/task:0/device:GPU:0
random_uniform_48/min: (Const): /job:localhost/replica:0/task:0/device:GPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_48/min: (Const)/job:localhost/replica:0/task:0/device:GPU:0
random_uniform_48/max: (Const): /job:localhost/replica:0/task:0/device:GPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_48/max: (Const)/job:localhost/replica:0/task:0/device:GPU:0
random_uniform_49/shape: (Const): /job:localhost/replica:0/task:0/device:GPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_49/shape: (Const)/job:localhost/replica:0/task:0/device:GPU:0
random_uniform_49/min: (Const): /job:localhost/replica:0/task:0/device:GPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_49/min: (Const)/job:localhost/replica:0/task:0/device:GPU:0
random_uniform_49/max: (Const): /job:localhost/replica:0/task:0/device:GPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_49/max: (Const)/job:localhost/replica:0/task:0/device:GPU:0
random_uniform_50/shape: (Const): /job:localhost/replica:0/task:0/device:CPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_50/shape: (Const)/job:localhost/replica:0/task:0/device:CPU:0
random_uniform_50/min: (Const): /job:localhost/replica:0/task:0/device:CPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_50/min: (Const)/job:localhost/replica:0/task:0/device:CPU:0
random_uniform_50/max: (Const): /job:localhost/replica:0/task:0/device:CPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_50/max: (Const)/job:localhost/replica:0/task:0/device:CPU:0
random_uniform_51/shape: (Const): /job:localhost/replica:0/task:0/device:CPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_51/shape: (Const)/job:localhost/replica:0/task:0/device:CPU:0
random_uniform_51/min: (Const): /job:localhost/replica:0/task:0/device:CPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_51/min: (Const)/job:localhost/replica:0/task:0/device:CPU:0
random_uniform_51/max: (Const): /job:localhost/replica:0/task:0/device:CPU:0
2020-02-25 20:11:48.[illegible]: I tensorflow/core/common_runtime/placer.cc:54] random_uniform_51/max: (Const)/job:localhost/replica:0/task:0/device:CPU:0
####### Calculating, matrix size = 7000
```

NEXT STEP

- Haoyang
    - Implement Object detection program
- Kefan
    - Implement matrix on PowerPC
    - Explore the possibilities of CNN testing
- Jeff
    - Finish building the benchmark program into a runnable docker image.
    - Keep working on the interaction with Chris backend(CUBE)
- Shineun
    - Run Matrix Multiplication App on ChRIS
    - Create Cookie-Cutter App for Object Detection
- Elizabeth
    - Deploy pman and pfioh to OpenStack
    - Run benchmarking program on OpenShift VM

BURNDOWN CHART



Our Team Specializations

1. Python Benchmarking Program - Haoyang, Kefan
2. ChRIS Cookie Cutter Plug-in - Shineun
3. The “CUBE”, ChRIS Ultron Backend - Jeff
4. Mass Open Cloud ChRIS App Deployment - Elizabeth