Medical Compute using ChRIS on the MOC

PowerPC & x86_64 GPU Usage & Benchmarking

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

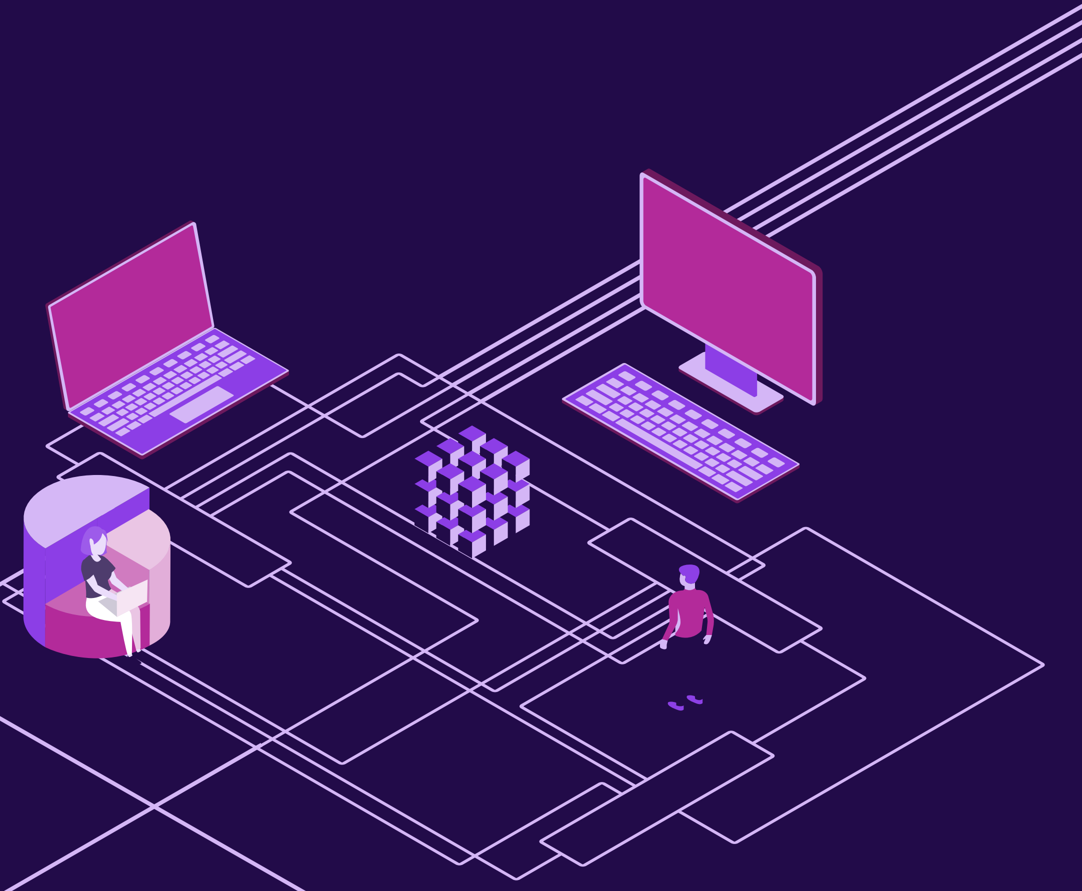


TABLE OF CONTENTS

01 **Project Overview**
Project Goals Description

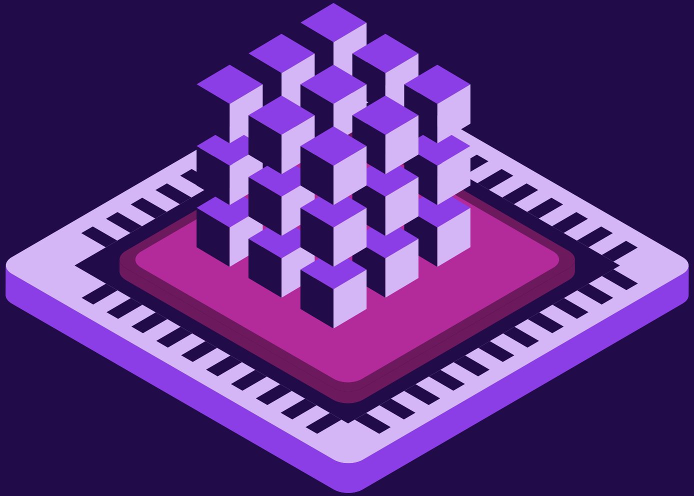
02 **Project Accomplishments**
What we have completed

03 **Project Summary**
What we have learned

04 **Limitations & Future Plan**
Future Perspective

01

Project Overview



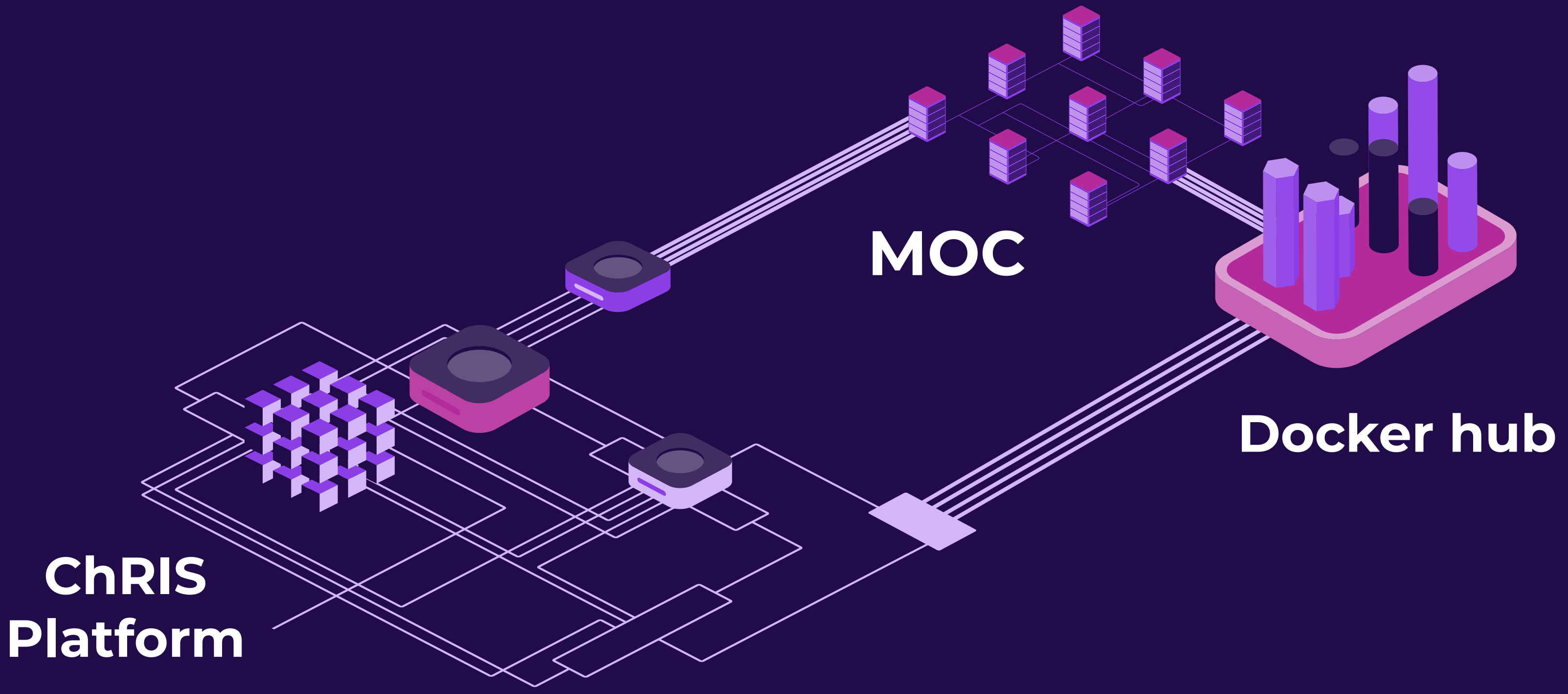
What is ChRIS?

An open source platform for medical analysis

Why ChRIS?

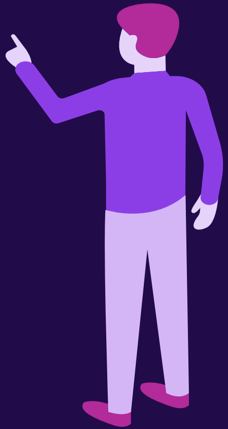
Technology in the medical industry is very *outdated*.
So, the Boston Children's Hospital team developed *ChRIS* to make *advanced technologies* like cloud computing and image analysis *more accessible* to doctors and researchers.

ChRIS ARCHITECTURE



USER STORY

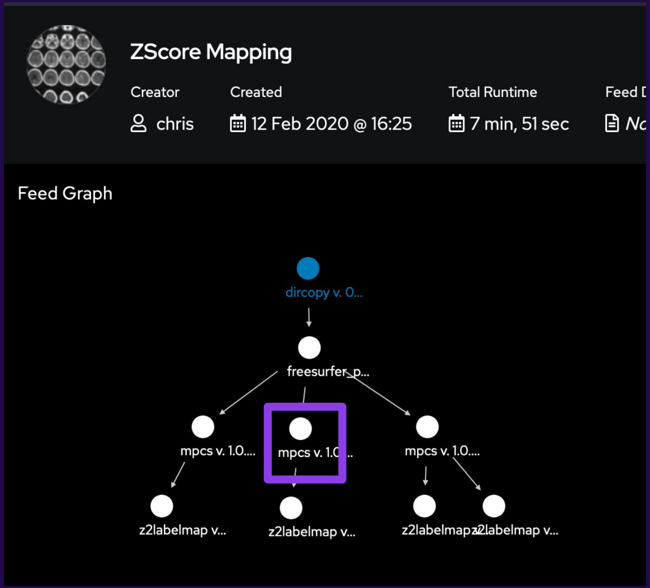
As a ChRIS developer / administrator,
I would like to have a way to test how my plugin performs on different architectures such as x86_64 vs PowerPC therefore I want a ChRIS plugin that performs benchmarking tests on these architectures



OUR PROJECT

Correctly develop
a runnable ChRIS plugin
that presents the
performance differences
between different platform
architectures

PROJECT GOALS



ZScore Mapping

Creator	Created	Total Runtime	Feed D
chris	12 Feb 2020 @ 16:25	7 min, 51 sec	No

Feed Graph

SCOPE & FEATURES

The Benchmarking Plugins should...

- Provide a series of tools to test the performance of the system
- Represent real workloads that may be deployed on the system
- Test functions should be small-sized and estimate the time that may be spent on running real computing tasks

02

Project Accomplishments

ACCOMPLISHMENTS

Docker Image 1

X86 Matrix Multiply Plug-in

Docker Image 2

PowerPC Matrix Multiply Plug-in

Docker Image 3

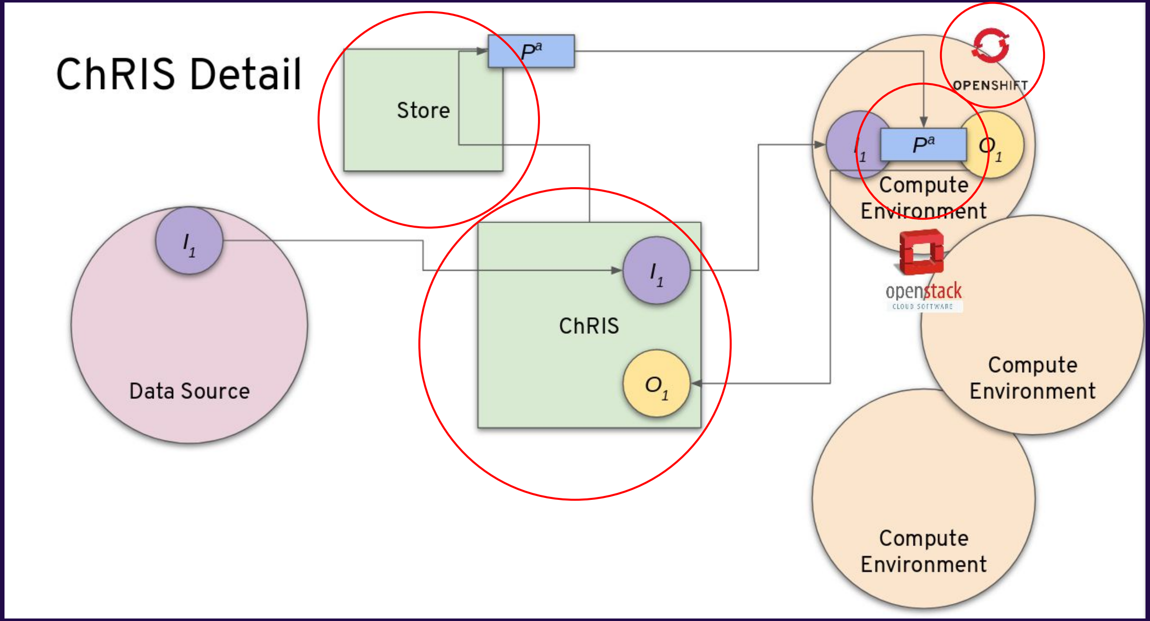
X86 Object Detection Plug-in

Docker Image 4

PowerPC Object Detection Plug-in

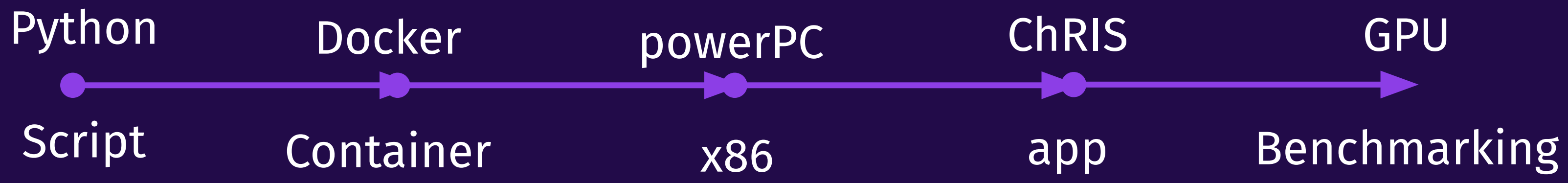
PowerPC pman and pfioh images on MOC are available to Boston Children’s team

ChRIS WORKFLOW



INDIVIDUAL APPS

Matrix Multiplication



Python Script

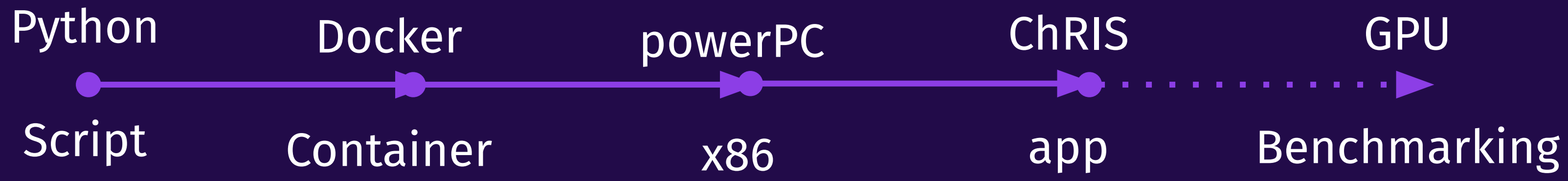
Docker Container

powerPC x86

ChRIS app

GPU Benchmarking

Object Detection



Python Script

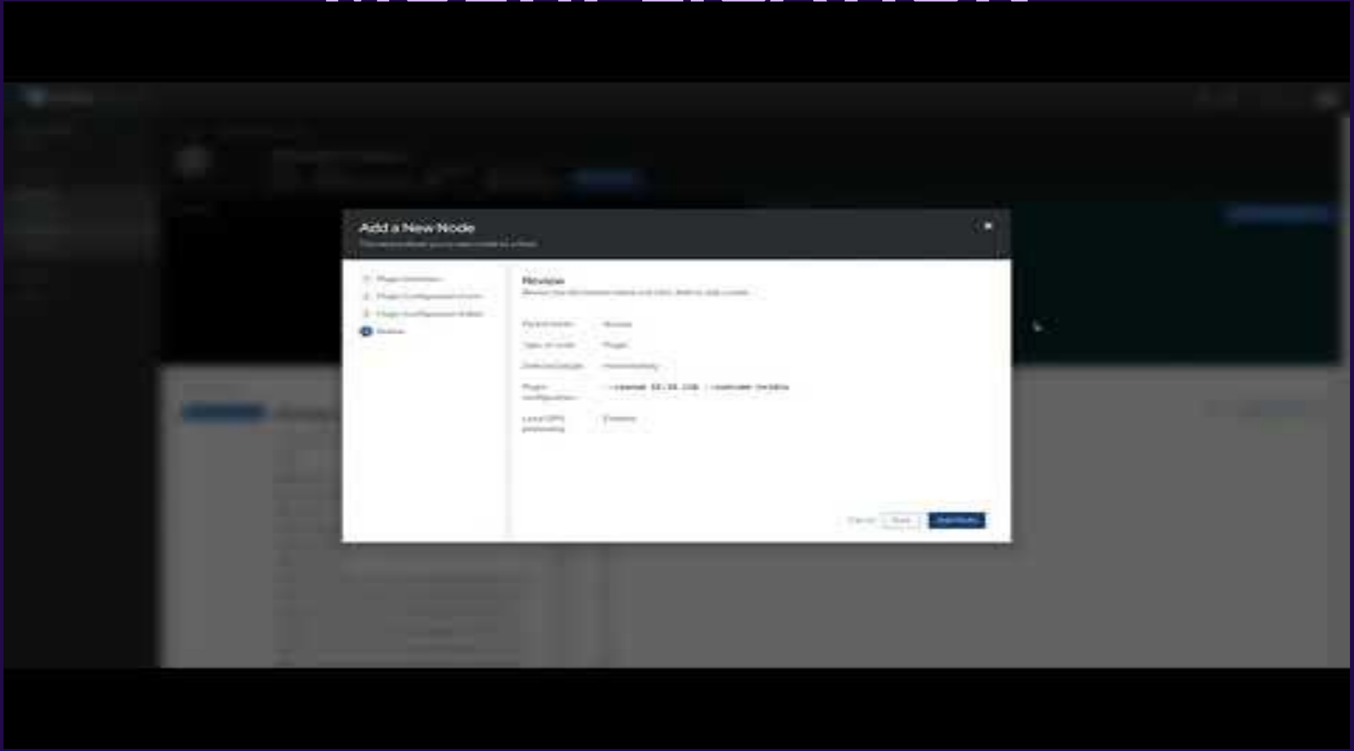
Docker Container

powerPC x86

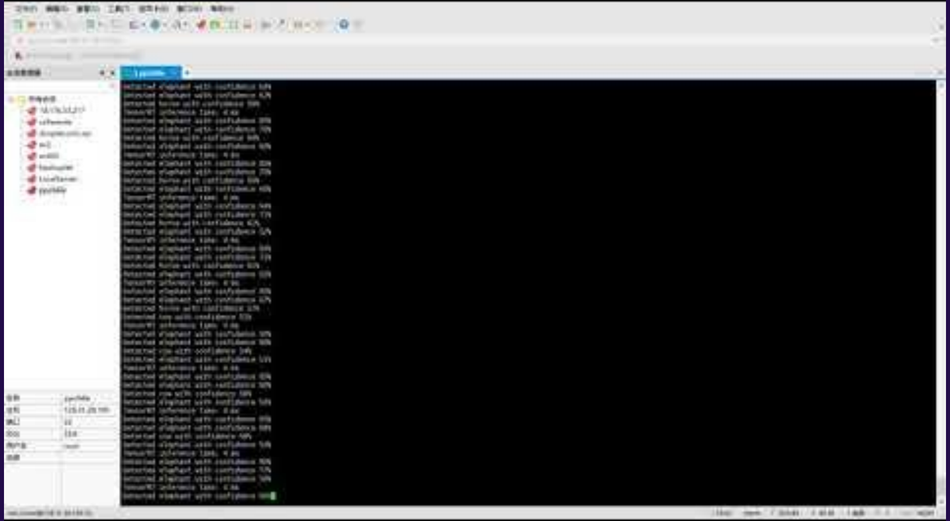
ChRIS app

GPU Benchmarking

MATRIX MULTIPLICATION



OBJECT DETECTION

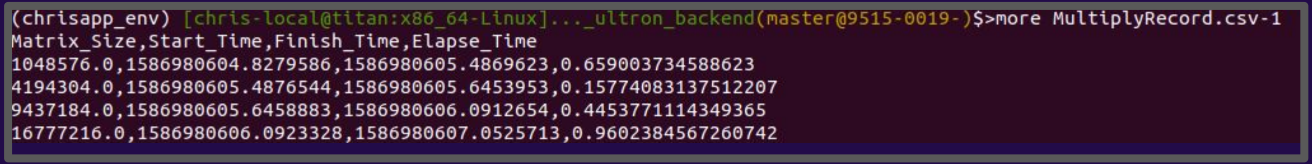


```
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected horse with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected horse with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected horse with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected horse with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected horse with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected horse with confidence [illegible]
Detected cow with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected cow with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected cow with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
Detected elephant with confidence [illegible]
TensorRT inference time: 4 ms
Detected elephant with confidence [illegible]
```

BENCHMARKING RESULT

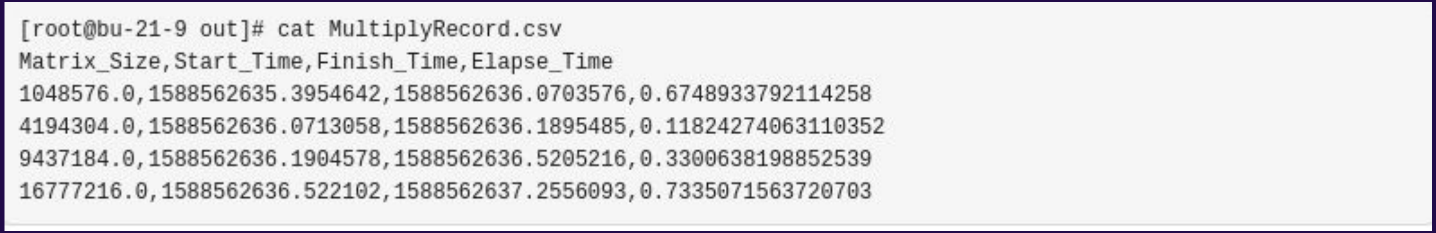
- Matrix Multiplication:

x86:



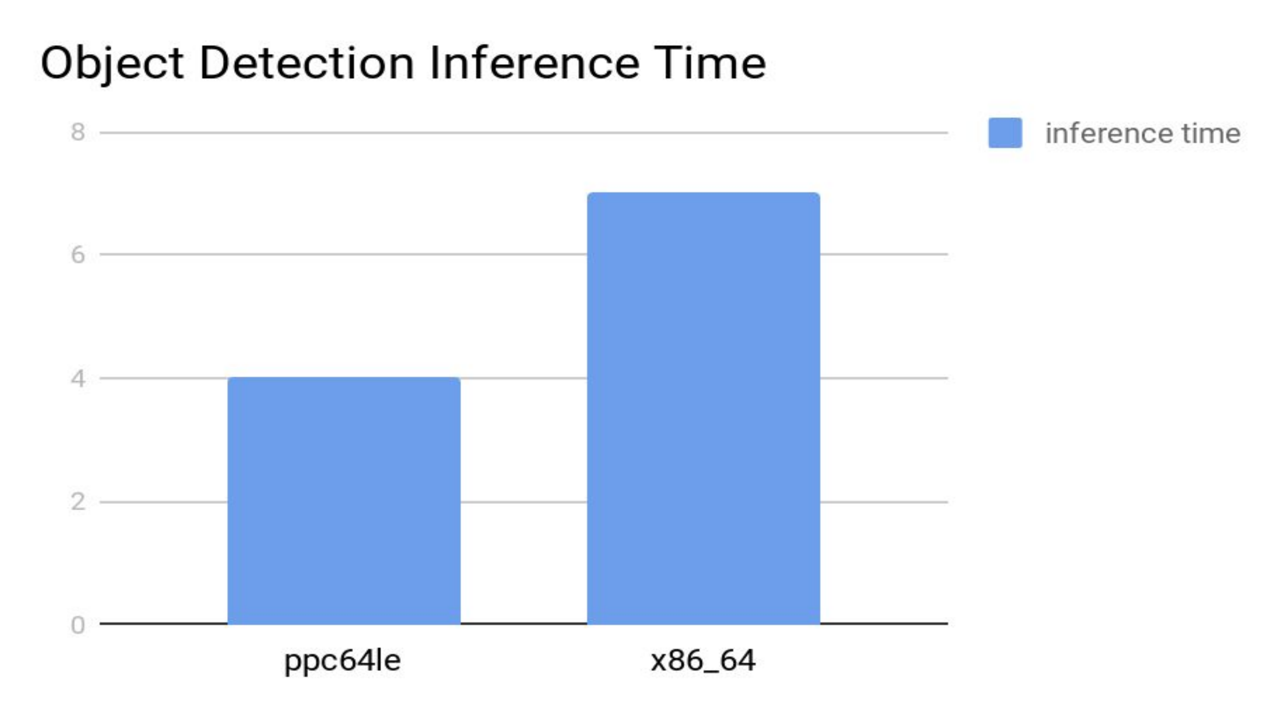
```
(chrisapp_env) [chris-local@titan:x86_64-Linux]..._ultron_backend(master@9515-0019-)$>more MultiplyRecord.csv-1
Matrix_Size,Start_Time,Finish_Time,Elapse_Time
1048576.0,1586980604.8279586,1586980605.4869623,0.659003734588623
4194304.0,1586980605.4876544,1586980605.6453953,0.15774083137512207
9437184.0,1586980605.6458883,1586980606.0912654,0.4453771114349365
16777216.0,1586980606.0923328,1586980607.0525713,0.9602384567260742
```

Power9:



```
[root@bu-21-9 out]# cat MultiplyRecord.csv
Matrix_Size,Start_Time,Finish_Time,Elapse_Time
1048576.0,1588562635.3954642,1588562636.0703576,0.6748933792114258
4194304.0,1588562636.0713058,1588562636.1895485,0.11824274063110352
9437184.0,1588562636.1904578,1588562636.5205216,0.3300638198852539
16777216.0,1588562636.522102,1588562637.2556093,0.733507156372070З
```

BENCHMARKING RESULT



MILESTONES

- ☑ Develop ChRIS plugins for x86 and PowerPC environments
- ☑ Deploy pman & pfioh on MOC both for x86 & PowerPC cluster
- ☑ Run ChRIS plugins through ChRIS GUI and CLI
- ☑ Get MicroBench results using our plugins

Bonus Task:

- □ Run ChRIS plugins on PowerPC MOC cluster through pfcon service

Demo for MOC, ChRIS app…

Like saying hello, whatever

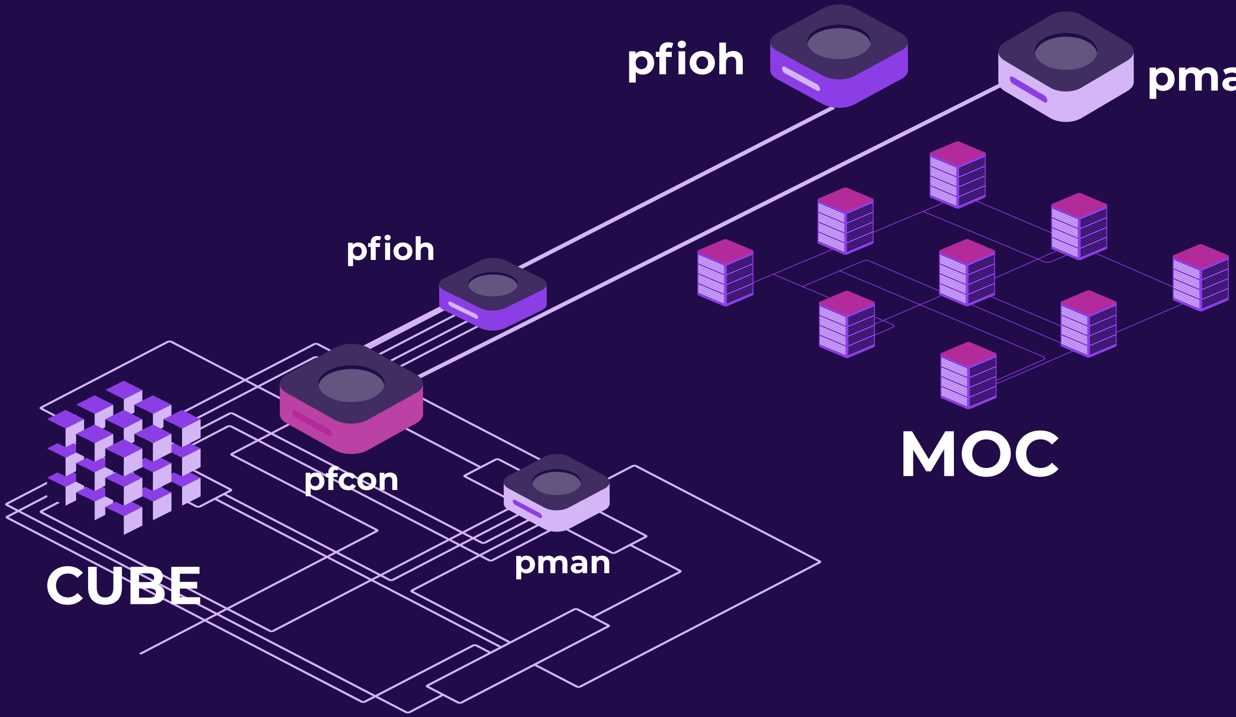
03

What we have learned

What we learned: Technical

- Concepts and usage of Docker and containers
- Deploying to MOC using Openshift
- Concepts of ChRIS architecture and building ChRIS apps
- GPU Programming (eg. parallel computing, GPU memory structure).
- Building PowerPC Images

Deploying PowerPC `pman` and `pfioh` on MOC



What we learned: Nontechnical

- Scrum team concept and sprints
- Managing and distributing tasks with Taiga.io
- Communicating technical challenges effectively

04

Limitations & Future Plan

LIMITATIONS

Benchmark purpose too general

Benchmark tests cannot tailor to different use cases

GPU Number Limitation

Only able to activate one GPU

FUTURE PLANS

Tool Box: More specific parameters

Parameter for focusing on optimizing iterative computations

- Machine learning plug-in, protein folding prediction

Parameter for focusing on memory read and writes speed

- File I/O, image recognition

GPU Number Limitation

Parameter for managing and processing big datasets

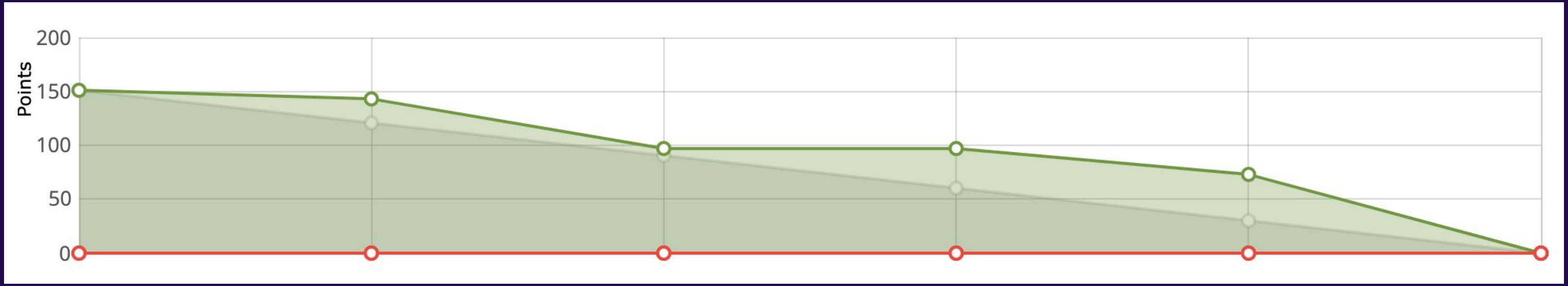
Brain scanning

Red Hat COVID-19

Computational projects that can help scientifically with COVID analysis

Moving project plugin to Power9 platform.

BURNDOWN CHART



THANK YOU