



Medical Compute with ChRIS on the MOC

PowerPC & x86_64 GPU Usage & Benchmarking

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

Quick Review

An open source platform for medical analysis

The goal is to democratize application development for medical analysis applications



Sprint4 Overview:

Matrix Multiplication App for x86 & PowerPC

Object Detection App for x86

Deployment on MOC

What we have learned:

How to run Plugin on PowerPC

Docker workflow

pfurl for pman & pfioh

ChRIS Instance



Last Sprint:

PROBLEMS NEED ATTENTION

- ✓ CUDA driver
- ✓ Hardware incompatible on VM
- ✓ Building pman and pfioh images on Power9 machine



Dockerized `Matmul` on Power9 & X86

Driver :

The ubuntu docker image provided by FNNDSC actually not contains a cuda driver.

-> We use the base docker image provided by nVidia.

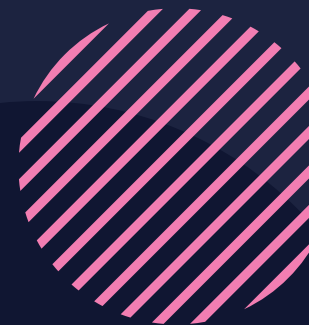
Hardware:

- ✓ Asked Power 9 admin for help to run a nvidia container
- ✓ Use `-e NVIDIA_VISIBLE_DEVICES=x` to specify the GPU id on x86



Power 9

Facts:

- All c/c++ based python libraries have to be compiled on Power 9 for `pip3 install` command.
 - Compile Environment.
 - Dependency issues.
- 

Power 9 version plugin

```
20 #
21 # docker run -ti -e HOST_IP=$(ip route | grep -v docker | awk '{if(NF==11) print $9}') --entrypoint /bin/bash local/pl-MatMultiply
22 #
23
24 FROM nvidia/cuda-ppc64le:10.1-devel
25 LABEL maintainer "NVIDIA CORPORATION <cuda@nvidia.com>"
26 ENV APPROOT="/usr/src/matmultiply"
27 COPY ["matmultiply", "${APPROOT}"]
28 COPY ["requirements.txt", "${APPROOT}"]
29 COPY ["dependencies.txt", "${APPROOT}"]
30 WORKDIR $APPROOT
31
32
33 RUN apt update
34 RUN apt install -y build-essential python3 llvm-7
35 RUN ln -s /usr/bin/llvm-config-7 /usr/bin/llvm-config
36 RUN apt install -y python3-pip
37 RUN pip3 install -r dependencies.txt
38 RUN pip3 install -r requirements.txt
39 ENTRYPOINT ["python3"]
40 CMD ["matmultiply.py", "--help"]
41
```

Compile Environment

2 Steps of pip install

Demo for Power 9

```
[root@bu-21-9 ~]# docker run --security-opt label=type:nvidia_container_t pupiltong/pl-matmultiply-ppc64le:latest matmultiply.py in out
0.6105351448059082
0.6105434894561768
[root@bu-21-9 ~]# nvidia-smi
Tue Mar 31 20:43:32 2020
```

NVIDIA-SMI 418.39 Driver Version: 418.39 CUDA Version: 10.1							
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile Uncorr. ECC		
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util	Compute M.	
0	Tesla V100-SXM2...	On	00000004:04:00.0	Off	0		
N/A	40C	P0	40W / 300W	0MiB / 16130MiB	0%	Default	
1	Tesla V100-SXM2...	On	00000004:05:00.0	Off	0		
N/A	44C	P0	39W / 300W	0MiB / 16130MiB	0%	Default	
2	Tesla V100-SXM2...	On	00000035:03:00.0	Off	0		
N/A	39C	P0	37W / 300W	0MiB / 16130MiB	0%	Default	
3	Tesla V100-SXM2...	On	00000035:04:00.0	Off	0		
N/A	43C	P0	38W / 300W	0MiB / 16130MiB	0%	Default	

Processes:					GPU Memory
GPU	PID	Type	Process name		Usage
No running processes found					

```
[root@bu-21-9 ~]# uname -a
Linux bu-21-9.infra.massopen.cloud 4.14.0-115.18.1.el7a.ppc64le #1 SMP Wed Jan 29 11:49:09 EST 2020 ppc64le ppc64le ppc64le GNU/Linux
[root@bu-21-9 ~]#
```

DEMO

Powerful Cluster

Ppc64le Arch

pman on MOC's OpenShift



APPLICATION

pman-ppc64le

<http://pman-ppc64le-benchmarking-chris-on-the-moc.p9-apps.osh.massopen.cloud/pman>



DEPLOYMENT CONFIG

pman-ppc64le, #8



CONTAINERS

pman-ppc64le

Image: [emslade/pman.ppc64le](#) d8ebe65 339.5 MiB
Ports: 5010/TCP



NETWORKING

Service - Internal Traffic

[pman-ppc64le](#)

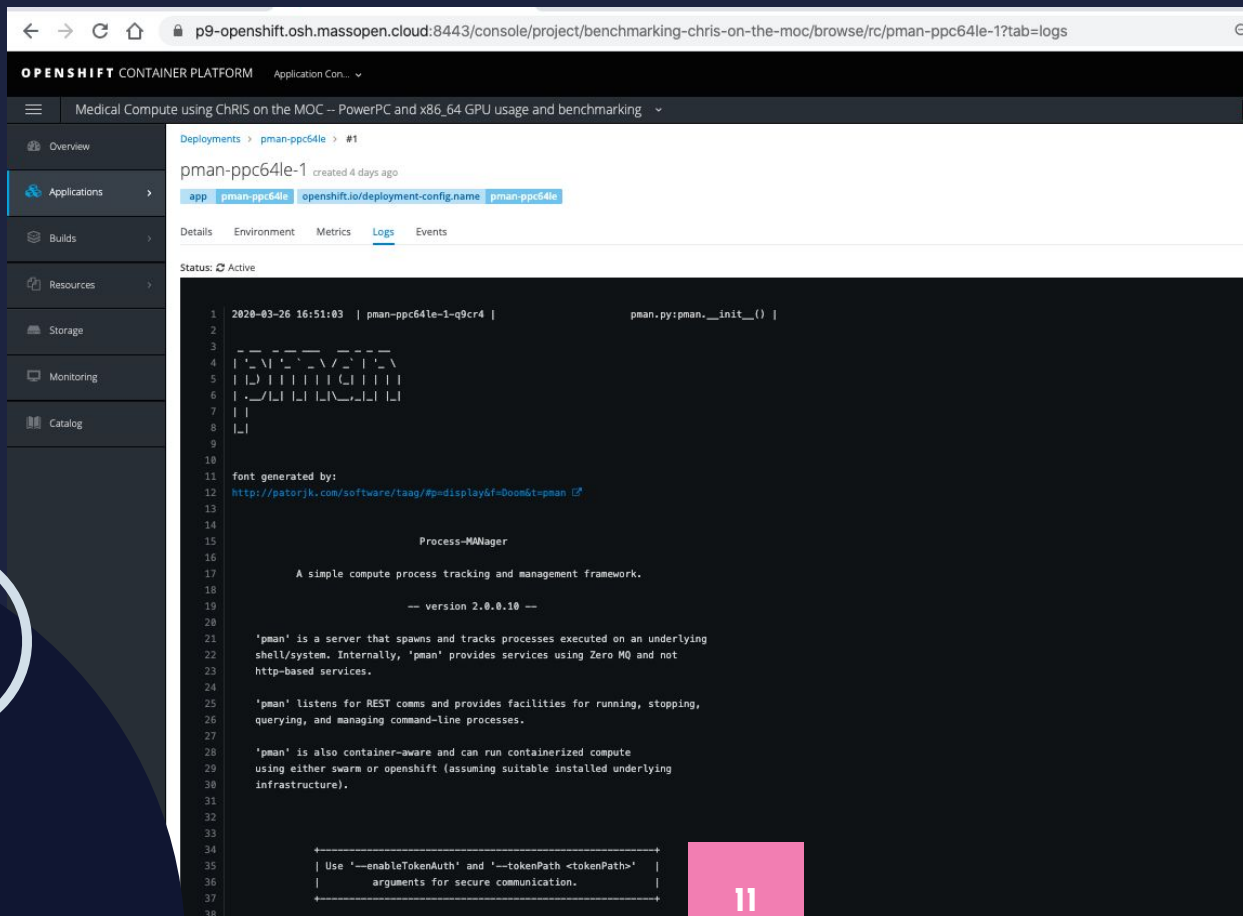
5010/TCP (5010-tcp) → 5010

Routes - External Traffic

<http://pman-ppc64le-benchmarking-chris-on-the-moc.p9-apps.osh.massopen.cloud/pman>

Route [pman-ppc64le](#), target port 5010-tcp

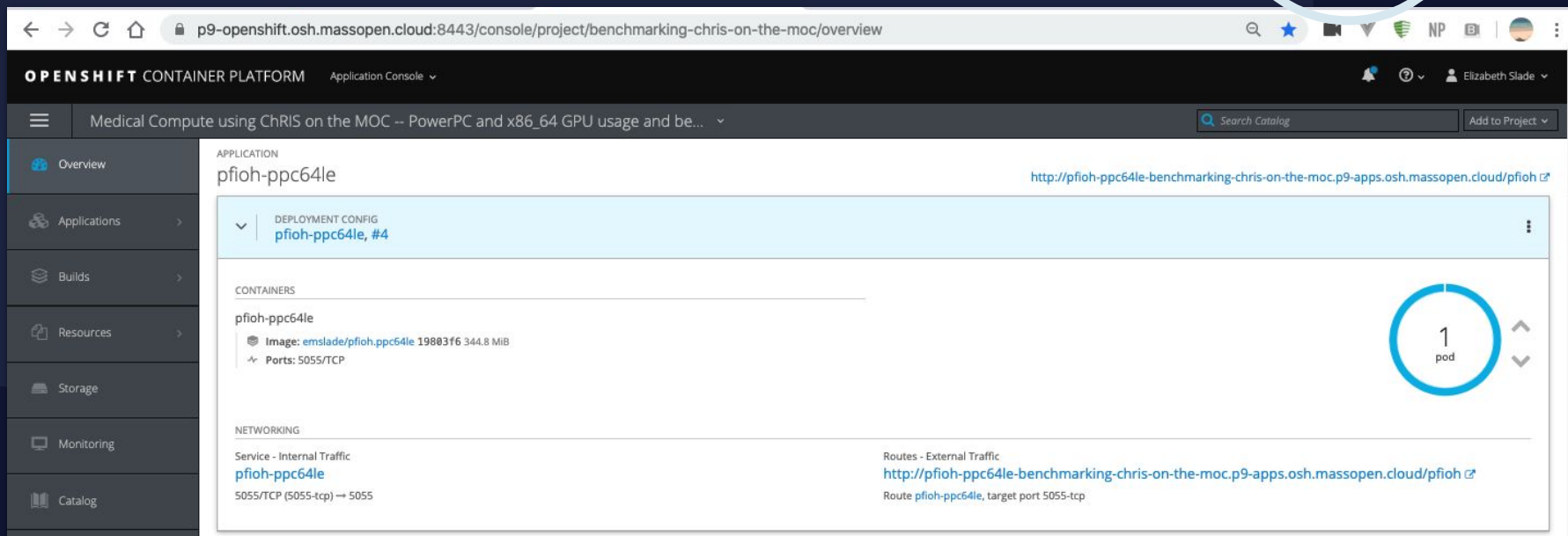
Process Manager, pman on OpenShift



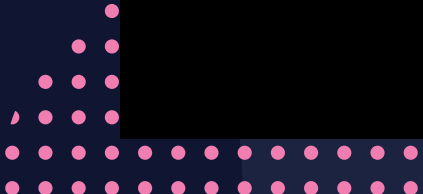
The screenshot shows the OpenShift Container Platform console interface. The top navigation bar indicates the current project is 'Medical Compute using CHRIS on the MOC - PowerPC and x86_64 GPU usage and benchmarking'. The left sidebar contains various navigation options: Overview, Applications, Builds, Resources, Storage, Monitoring, and Catalog. The main content area displays the details for a deployment named 'pman-ppc64le-1', which was created 4 days ago. The deployment is currently in an 'Active' state. The 'Logs' tab is selected, showing the output of the 'pman.py:pman.__init__() |' command. The log output includes a ASCII art logo for 'Process-MANager', a link to the source code, and a detailed description of the 'pman' tool.

```
1 2020-03-26 16:51:03 | pman-ppc64le-1-q9cr4 | pman.py:pman.__init__() |
2
3
4  _ _ _ _ _
5  | | | | | | | | | |
6  | _/ _/ _/ _/ _/ _/ _/
7  | |
8  | |
9
10
11 font generated by:
12 http://patorjk.com/software/taag/#p=display&f=Doom&st=pman
13
14
15 Process-MANager
16
17 A simple compute process tracking and management framework.
18
19 -- version 2.0.0.10 --
20
21 'pman' is a server that spawns and tracks processes executed on an underlying
22 shell/system. Internally, 'pman' provides services using Zero MQ and not
23 http-based services.
24
25 'pman' listens for REST comms and provides facilities for running, stopping,
26 querying, and managing command-line processes.
27
28 'pman' is also container-aware and can run containerized compute
29 using either swarm or openshift (assuming suitable installed underlying
30 infrastructure).
31
32
33
34 | Use '--enableTokenAuth' and '--tokenPath <tokenPath>' |
35 | arguments for secure communication. |
36
37
38
```





pfioh on MOC's OpenShift







The screenshot displays the OpenShift Container Platform console interface. The top navigation bar shows the URL `p9-openshift.osh.massopen.cloud:8443/console/project/benchmarking-chris-on-the-moc/overview`. The main header includes the OpenShift logo and the text "OPENSHIFT CONTAINER PLATFORM". The left sidebar contains a menu with options: Overview, Applications, Builds, Resources, Storage, Monitoring, and Catalog. The main content area is titled "APPLICATION pfioh-ppc64le" and includes a deployment configuration "pfioh-ppc64le, #4". Under the "CONTAINERS" section, it shows the container "pfioh-ppc64le" with details: Image: `emslade/pfioh.ppc64le` 19883f6 344.8 MiB, and Ports: 5055/TCP. A circular progress indicator shows "1 pod". The "NETWORKING" section displays the service "pfioh-ppc64le" with port 5055/TCP (5055-tcp) mapped to 5055. External traffic routes are also shown, including the URL `http://pfioh-ppc64le-benchmarking-chris-on-the-moc.p9-apps.osh.massopen.cloud/pfioh`.



Status of Plugin Development

Last Sprint	Object Detection	Matrix Multiplication
x86		
Power9		

Status of Plugin Development

This Sprint	Object Detection	Matrix Multiplication
x86		
Power9		

Modification: Parameter option

Examples

-c start, step, stop

```
docker run --runtime=nvidia \
    -e NVIDIA_VISIBLE_DEVICES=1 \
    -v $(pwd)/in:/incoming -v $(pwd)/out:/outgoing \
    fnndsc/pl-matrixmultiply \
    python3 matmultiply.py \
    -c 32,32,128 \
    /incoming /outgoing
```

Matrix size = (COE x TPB) ^2, TPB fixed as 32

Modification: Output as csv file

robertmorrislike — chris-local@titan (linux-gnu) pts/14 ~/kefan/kefan7/out — ssh -X -p 7639..

Matrix_Size	Start_Time	Finish_Time	Elapse_Time
1048576.0	1585700450.1756785	1585700450.8636434	0.687964916229248
4194304.0	1585700450.8645773	1585700451.027952	0.163374662399292
9437184.0	1585700451.028828	1585700451.4861693	0.45734143257141113
16777216.0	1585700451.487207	1585700452.4652367	0.978029727935791

(END)

$\text{Elapse} = \text{Finish} - \text{Start}$

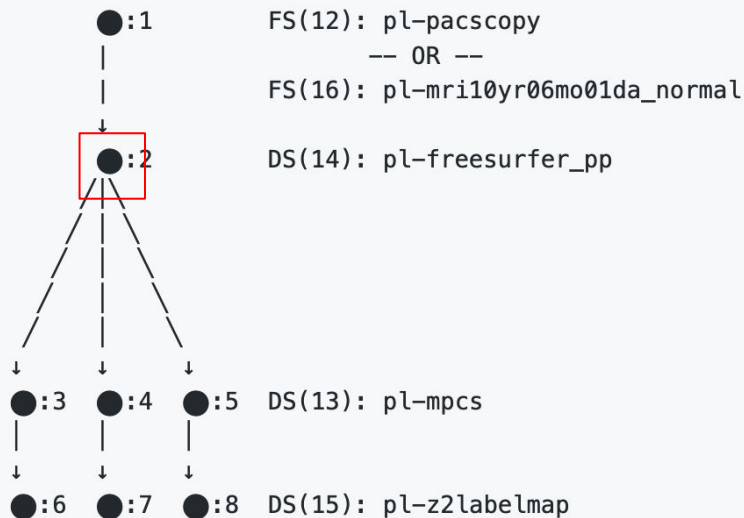
Now, we can benchmark with **Matrix multiplication plugin**

MatMul demo



Big Picture: a “node” on ChRIS

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



ZScore Mapping

Creator

Created

Total Runtime

Feed

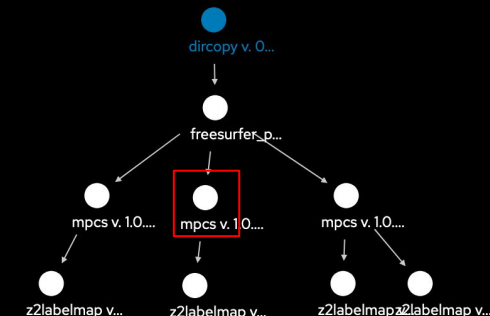
chris

12 Feb 2020 @ 16:25

7 min, 51 sec

Feed

Feed Graph



Recap: 2 ways to run a plugin

- Run it through Docker containers
 - Successfully build 3 images(local, x86_moc, ppc_moc).
 - maintained on Dockerhub under FNNDSC repo.
- Run it through Chris Instance
 - What does that means?

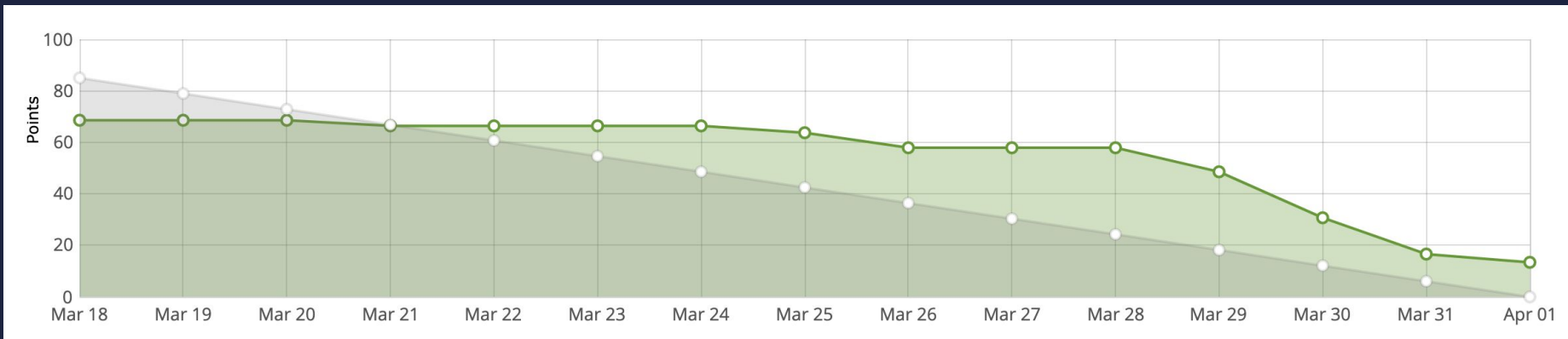
ChRIS Instance

- Register runnable plugin to Chris store.
- Fire up a ChRIS instance.
 - instantiated as Django-mysql project
 - offering a collection+json REST API.

Next Step

- Integrating **Object-Detection** to ChRis plugin
- Test plugins through **ChRIS instance**
- Test plugins through **ChRIS GUI**
- **Benchmarking** plugin result between x86 vs PowerPC

BURNDOWN CHART





THANKS

Any questions?