Kubernetes and Google Cloud

Ricardo Rocha

CERN IT-CM

CERN OpenLab Workshop
https://indico.cern.ch/event/1009424/

Summary and Goals

Ongoing collaboration since early 2019

Validate the use and scalability of the Google Cloud

- Evaluate its use for future computing models
- 2019: Cover for spikes with on-demand resources (CPUs)
- 2020: Access to a larger number of scarce resources (GPUs)
- 2020: Access to resources not available otherwise (TPUs)

Recent Achievements

Google Cloud customer story

Deployed Kubernetes based infrastructure to manage scale out use cases

Onboarding use cases into the new system

- Done: GitLab Runners and Kubeflow / Machine Learning
- Ongoing: JupyterHub, Binder, Dask, …

Cost analysis for usage of GPUs, TPUs and GPU vs TPU

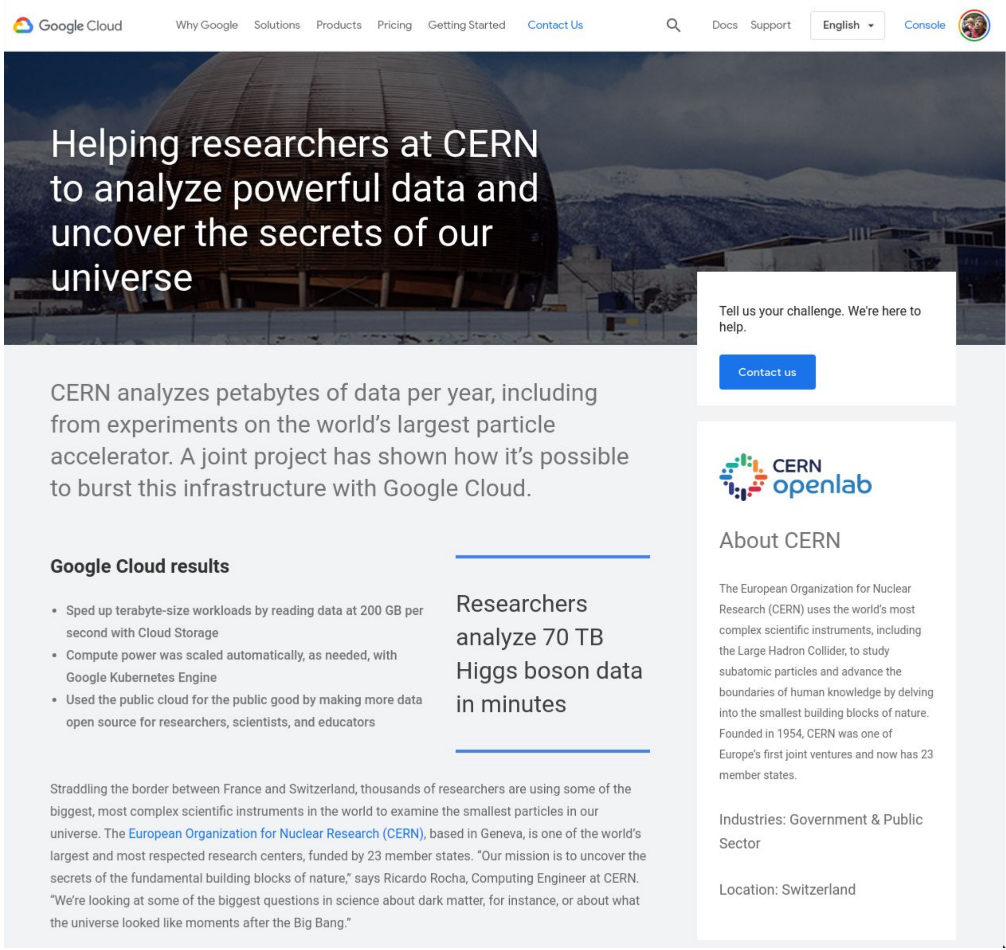
Moved to rely on Cloudbank EU / Broker Pilot resources

Customer Story

Focus on the results from 2019

“Reperforming a Nobel Prize Discovery on Kubernetes”

https://www.youtube.com/watch?v=CTfp2woVEkA



Google Cloud | Why Google | Solutions | Products | Pricing | Getting Started | Contact Us | Docs | Support | English | Console

Helping researchers at CERN to analyze powerful data and uncover the secrets of our universe

Tell us your challenge. We're here to help.

Contact us

CERN analyzes petabytes of data per year, including from experiments on the world’s largest particle accelerator. A joint project has shown how it’s possible to burst this infrastructure with Google Cloud.

Google Cloud results

- Sped up terabyte-size workloads by reading data at 200 GB per second with Cloud Storage
- Compute power was scaled automatically, as needed, with Google Kubernetes Engine
- Used the public cloud for the public good by making more data open source for researchers, scientists, and educators

Researchers analyze 70 TB Higgs boson data in minutes

Straddling the border between France and Switzerland, thousands of researchers are using some of the biggest, most complex scientific instruments in the world to examine the smallest particles in our universe. The European Organization for Nuclear Research (CERN), based in Geneva, is one of the world’s largest and most respected research centers, funded by 23 member states. “Our mission is to uncover the secrets of the fundamental building blocks of nature,” says Ricardo Rocha, Computing Engineer at CERN. “We’re looking at some of the biggest questions in science about dark matter, for instance, or about what the universe looked like moments after the Big Bang.”

CERN openlab

About CERN

The European Organization for Nuclear Research (CERN) uses the world’s most complex scientific instruments, including the Large Hadron Collider, to study subatomic particles and advance the boundaries of human knowledge by delving into the smallest building blocks of nature. Founded in 1954, CERN was one of Europe’s first joint ventures and now has 23 member states.

Industries: Government & Public Sector

Location: Switzerland

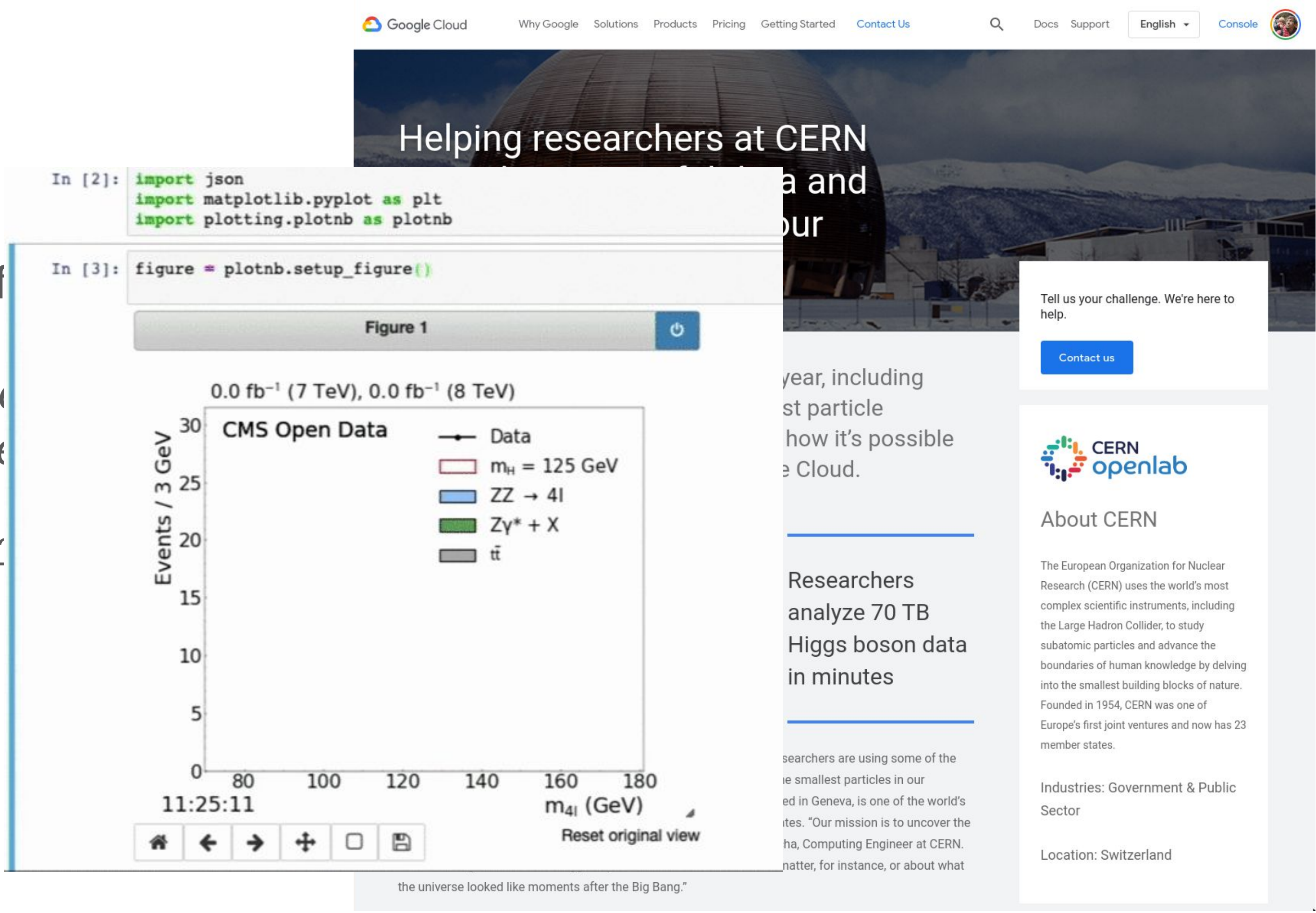
https://cloud.google.com/customers/cern

Customer Story

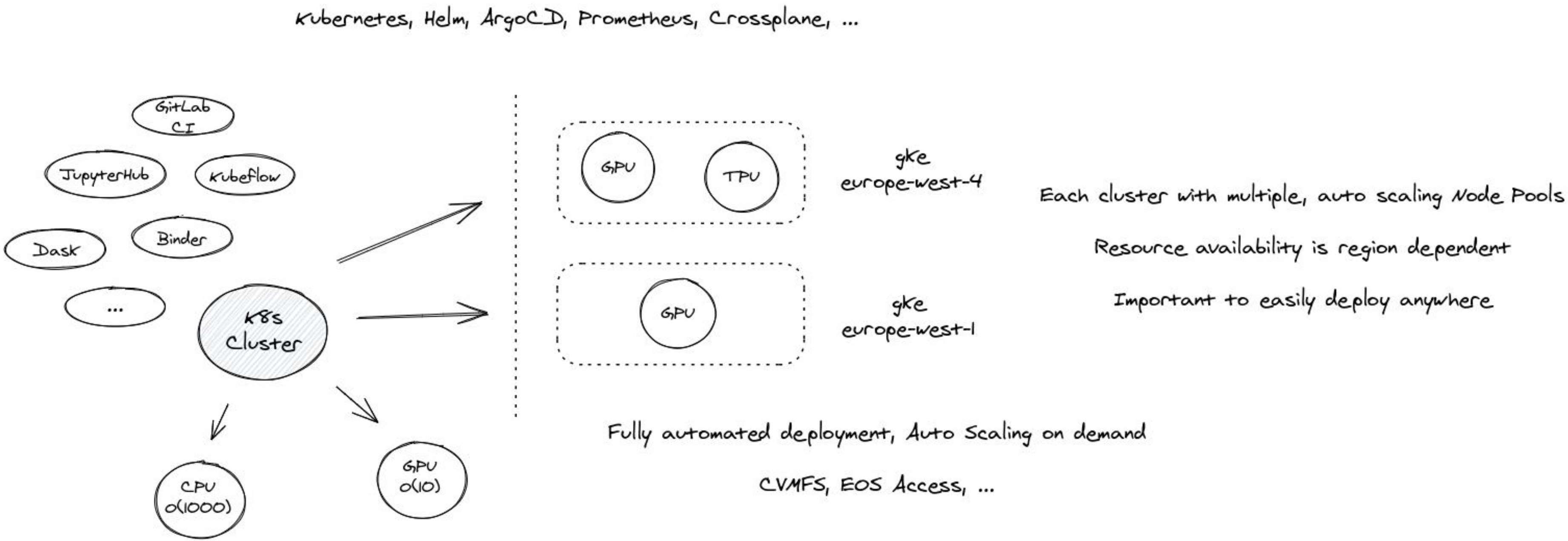
Focus on the results f

“Reperforming a Nob
Discovery on Kuberne

https://www.youtube.com/watch?



https://cloud.google.com/customers/cern

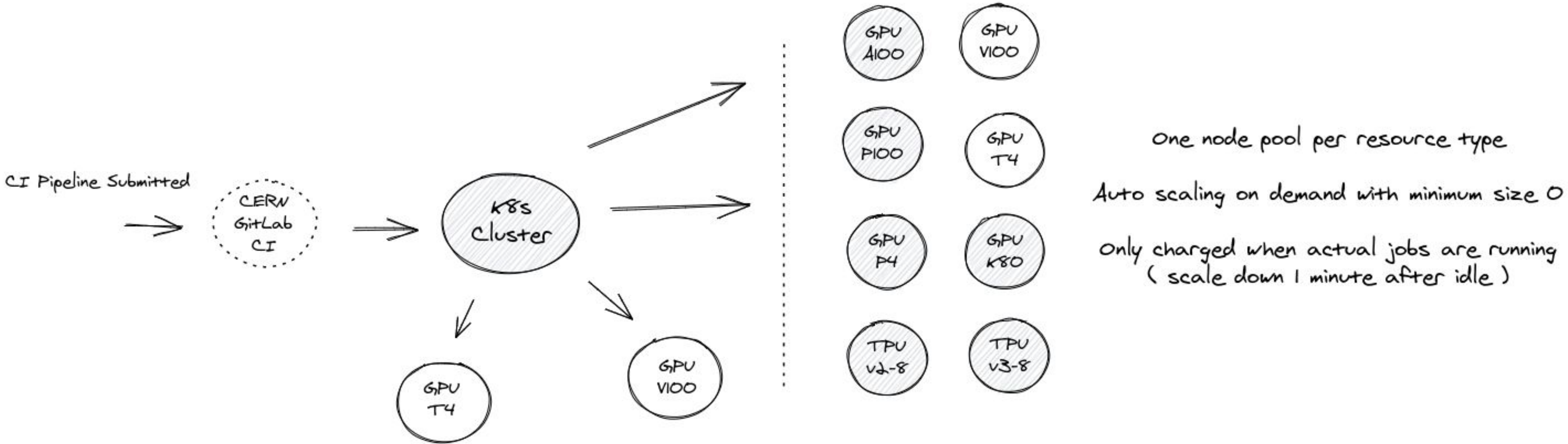

Kubernetes, Helm, ArgoCD, Prometheus, Crossplane, ...
GitLab CI
JupyterHub
Kubeflow
Dask
Binder
...
K8s Cluster
CPU O(1000)
GPU O(10)
GPU
TPU
gke europe-west-4
GPU
gke europe-west-1
Each cluster with multiple, auto scaling Node Pools
Resource availability is region dependent
Important to easily deploy anywhere
Fully automated deployment, Auto Scaling on demand
CVMFS, EOS Access, ...

GitLab CI Runners

Access to a larger pool of resources, including some not accessible on-premises

End users do not realise they are running on public cloud resources

https://gitlab.cern.ch/rbritoda/gitlab-runner-public



```
Waiting for pod gitlab-runner/runner-ka9reaze-project-108952-concurrent-1dgllp to be running, status is Pending
       ContainersNotReady: "containers with unready status: [build helper]"
       ContainersNotReady: "containers with unready status: [build helper]"
Running on runner-ka9reaze-project-108952-concurrent-1dgllp via gitlab-runner-gpu-a100-gitlab-runner-854f7d54b7-z9t9f...
Getting source from Git repository                                                                  00:01
Fetching changes with git depth set to 50...
Initialized empty Git repository in /builds/rbritoda/gitlab-runner-public/.git/
Created fresh repository.
Checking out 828259f8 as load...
Skipping Git submodules setup
Executing "step_script" stage of the job script                                                     00:00
$ nvidia-smi
Tue Feb  9 20:55:13 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.51.06    Driver Version: 450.51.06    CUDA Version: 11.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  A100-SXM4-40GB      Off  | 00000000:00:04.0 Off |                    0 |
| N/A   33C    P0    42W / 400W |      0MiB / 40537MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
Cleaning up file based variables                                                                    00:00
Job succeeded
```

```
Waiting for pod gitlab-runner/runner-hdqec7c-project-108952-concurrent-1fbmnv to be running, status is Pending
	ContainersNotReady: "containers with unready status: [build helper]"
	ContainersNotReady: "containers with unready status: [build helper]"
Running on runner-hdqec7c-project-108952-concurrent-1fbmnv via gitlab-runner-tpu-v3-8-gitlab-runner-5d54df77b-wcgq8...
Getting source from Git repository                                                                    00:01
Fetching changes with git depth set to 50...
Initialized empty Git repository in /builds/rbritoda/gitlab-runner-public/.git/
Created fresh repository.
Checking out 828259f8 as load...
Skipping Git submodules setup
Executing "step_script" stage of the job script                                                       00:12
```

```python
#!/usr/bin/python3
import tensorflow as tf
import os
from tensorflow.python.profiler import profiler_client

endpoint = os.environ['KUBE_GOOGLE_CLOUD_TPU_ENDPOINTS']
print("Connecting to TPU at %s\n" % endpoint)

tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
print('Running on TPU ', tpu.cluster_spec().as_dict()['worker'])

tf.config.experimental_connect_to_cluster(tpu)
tf.tpu.experimental.initialize_tpu_system(tpu)

print(profiler_client.monitor(endpoint.replace('8470', '8466'), 100, 2))
```

```
2021-02-09 21:01:12.702455: I tensorflow/core/distributed_runtime/rpc/grpc_server_lib.cc:390] Started server with target: grpc://localhost:30018
Connecting to TPU at grpc://10.116.18.178:8470
Running on TPU  ['10.116.18.178:8470']
  Timestamp: 21:01:22
  TPU type: TPU v3
  Utilization of TPU Matrix Units (higher is better): 0.000%
Cleaning up file based variables                                                                      00:00
Job succeeded
```

ML and Kubeflow

Kubeflow

https://www.kubeflow.org/

Scaling out a ML workload can be hard

- Access to often scarce resources
- Adapting the code, managing the deployment / infrastructure

Kubeflow is the machine learning toolkit for Kubernetes

- Manages the full ML lifecycle: from data preparation to serving
- Built-in constructs (operators) for distributing workloads
- Support for all popular frameworks: TensorFlow, Pytorch, MXNet, MPI, …

Hiding all the infra details so end users focus on their code

ML and Kubeflow: 3D GANs

With Renato Cardoso, Sofia Vallecorsa, Dejan Golubovic

Fast simulation with 3DGANs (see Renato’s talk earlier)

TF based, already adapted to use tf.distributed.Strategy

Can we scale this out to a large number of GPUs? And TPUs?

Steps

1. Wrap it in a TFJob
2. Evaluate optimal GPU layout (cards per node, number nodes), minimize contention
3. Evaluate optimal batch size
4. Train with a large number of resources, evaluate efficiency

ML and Kubeflow: 3D GANs

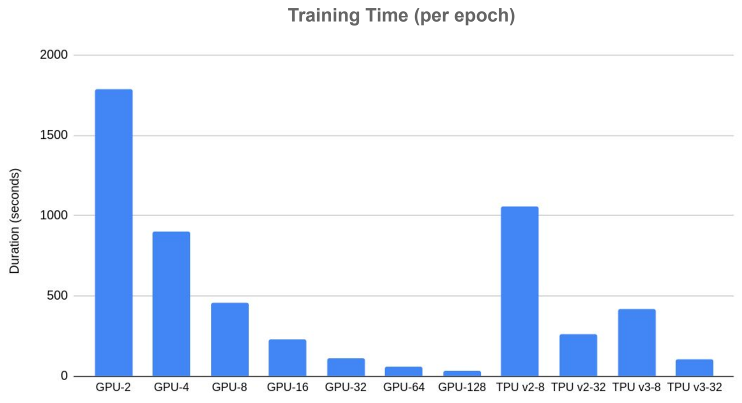
Settled on nodes with 8 GPUs each, total 128 GPUs

Training time reduced from 1780 secs to 34 secs

Close to linear scaling almost all the way
52x improvement with 128 GPUs vs 2 GPUs

TPU-v3 with slighter better results vs equivalent GPUs

ML and Kubeflow: 3D GANs



Settled on nodes with 8 GPUs each, total 128 GPUs

Training time reduced from 1780 secs to 34 secs

Close to linear scaling almost all the way
52x improvement with 128 GPUs vs 2 GPUs

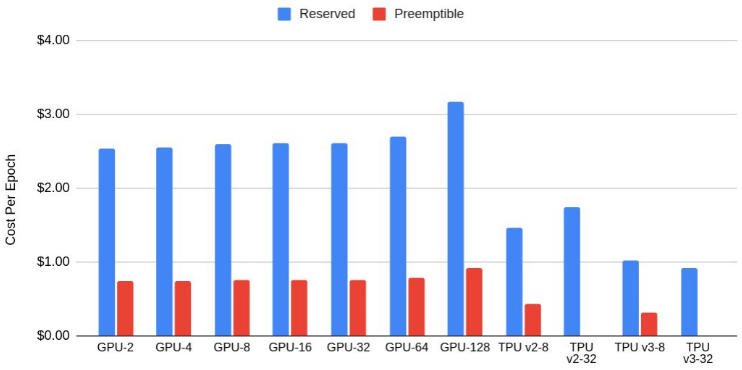
TPU-v3 with slighter better results vs equivalent GPUs

Close to flat cost when scaling out the number of GPUs

Preemptibles a major cost saving option when possible

TPUs offer 2.5x savings compared to equivalent GPU setup

Preemptible TPUs only available with 8 cores



Summary

Running a **multi-cloud Kubernetes** based infrastructure

- Auto scaling based on demand - grow and pay only when needed
- Transparent to end users, multiple service entrypoints

Achieved linear scaling of single workload up to 128 GPUs

- **52x times faster than 2 GPUs, with similar overall cost**

Demonstrated the public cloud can scale and be **cost effective**

- With potential benefits in compute time for end users
- **Preemptible GPUs** (as for CPUs) offer a huge gain in cost
- (Preemptible up to 8) **TPUs** even better if / when workloads can make use of them

Next Steps

Continue onboarding new use cases (do reach out - ricardo.rocha@cern.ch)

Continue pushing scale out workloads

- Ongoing test with up to 1024 GPUs for a single workload
- Similar test with much larger TPUs, up to v3-512 cores

Evolve usage accounting and billing

- Prototype deployed for (external) billing estimation - Prometheus based
- Explore options to link single cluster usage to multiple billing accounts