KubeCon | CloudNativeCon

North America 2017

How Kubernetes is Helpful for Accelerating Machine Learning Research and Engineering

Hitoshi Mitake, Software Engineer,
NTT Laboratories

Agenda

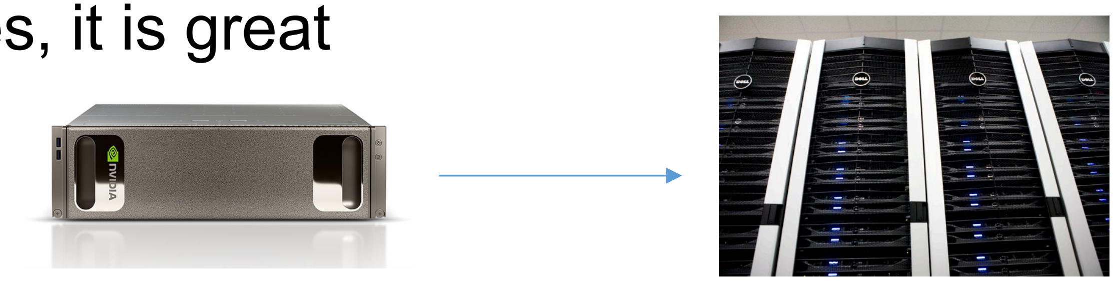
- Who I am
  - github.com/mitake
    - A maintainer of etcd: github.com/coreos/etcd
    - A contributor of kubernetes
- What I'll talk
  - A use case of k8s: a tool for managing distributed deep learning frameworks
    - What we needed to do and why k8s is useful for the purpose
      - Especially the case of custom scheduler
    - What we are doing, especially in the open source communities
      - Controlling RDMA resources from cgroups and scheduler stuff

K8s is widely used, of course

- For web applications, k8s is the de-fact standard
  - Especially for managing stateless services
- How about using k8s for managing deep learning frameworks?
  - e.g. executing TensorFlow or MXNet
  - Is this suitable?
  - Based on our experience, k8s is also good for the purpose

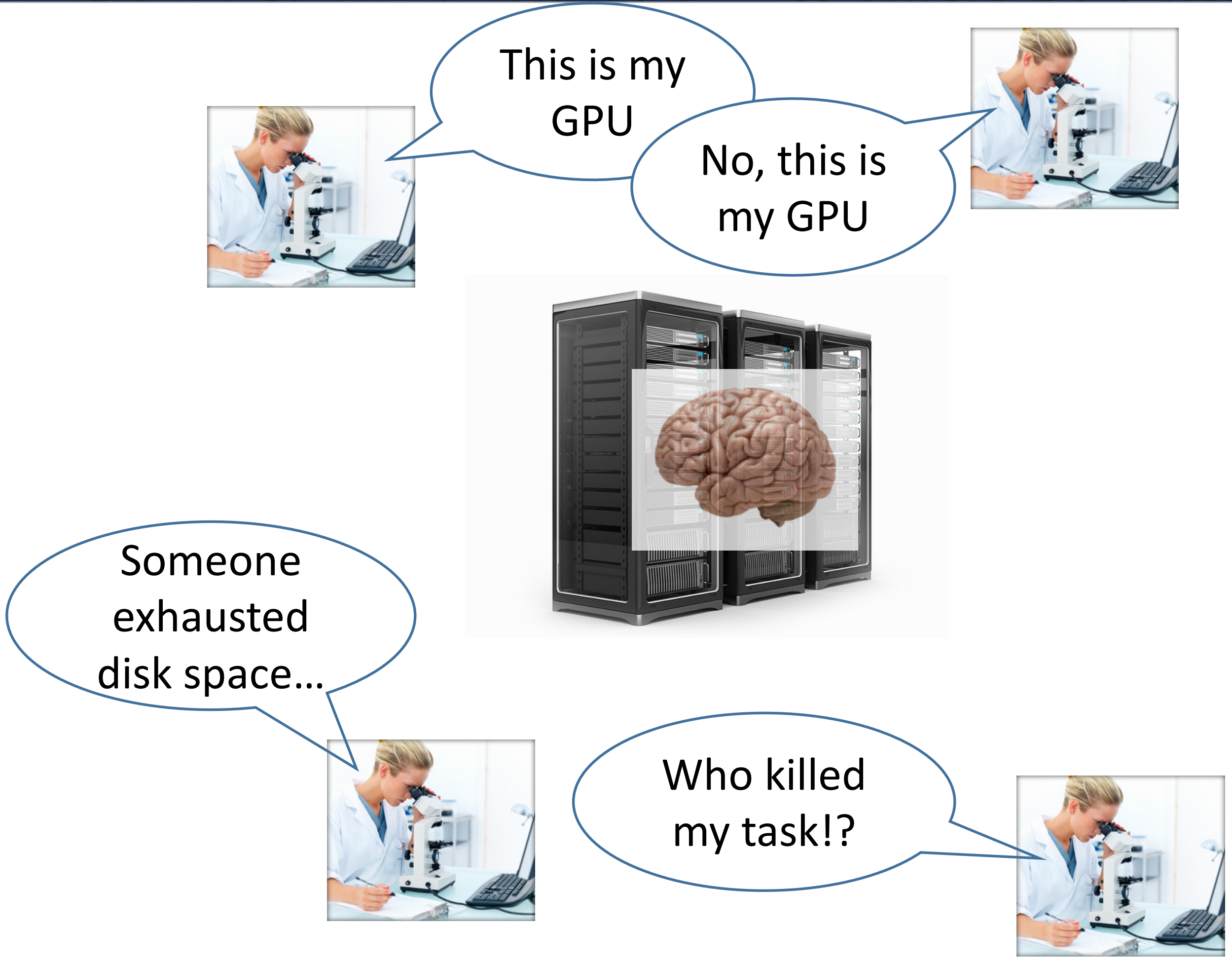
Motivation of having own clusters for DL research

- Why not use managed services e.g. Google cloud ML?
  - Researchers who work on improving optimization methods need to run their own modified versions of frameworks
- Why not pay lots of money to nvidia?
  - Using a single machine equipped with lots of GPUs is reasonable
    - e.g. DGX-1
  - But if we can achieve performance of a high end machine with a cluster of cheap GPUs or other devices, it is great
    - Cost efficient
    - Long term goal



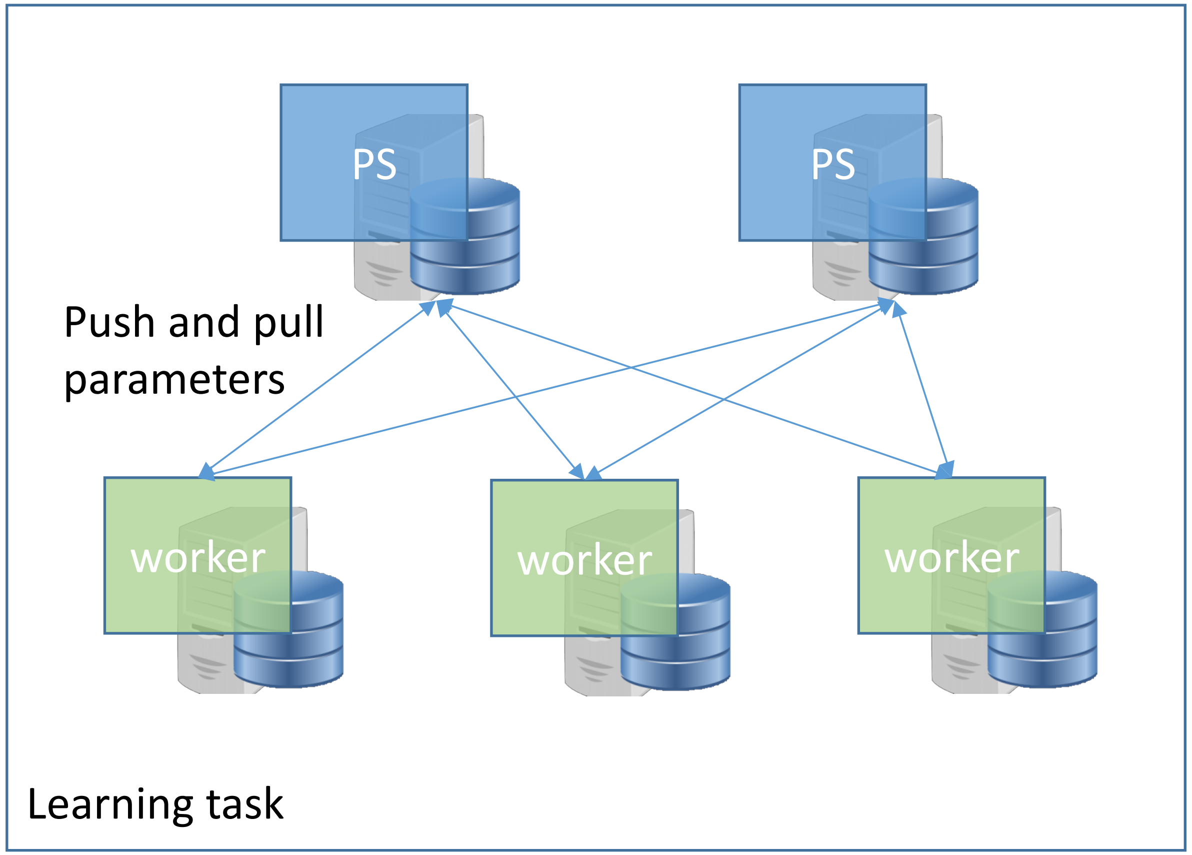
So using an in-house cluster is still reasonable

- But managing a cluster is a daunting task
  - GPUs are expensive, multiple researchers need to share a single cluster
  - Sharing machines is confusing and trouble prone
- Let’s use k8s
  - K8s makes lots of such tasks easier
  - But off-the-shelf k8s wasn’t enough



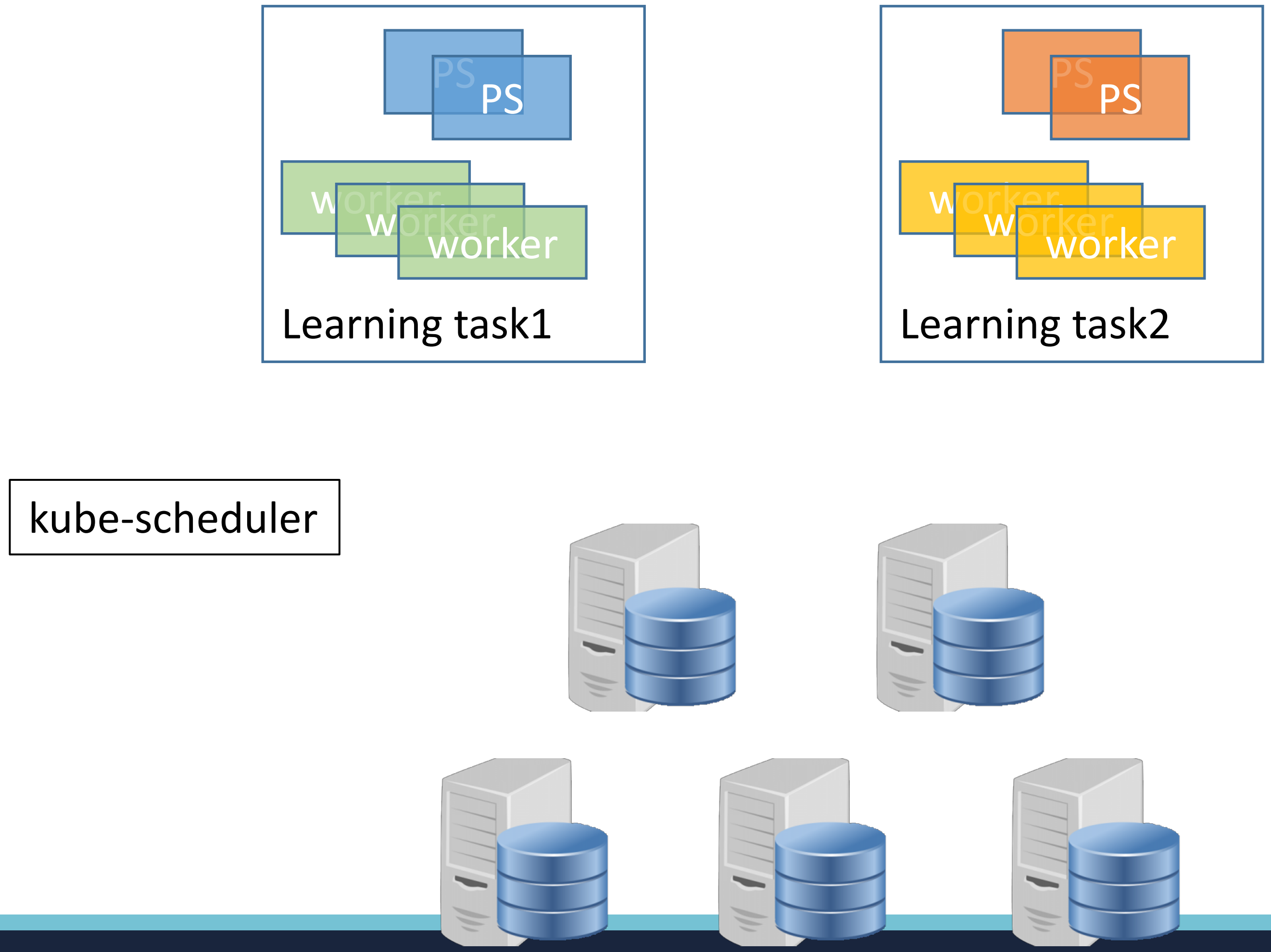
Typical distributed learning task of DNN

- Parameter server (PS) style
  - Each PS has a portion of parameters (weight and bias of NN)
  - Each worker trains NN and sync its result via PS
  - another popular style: all-reduce
  - How about just making a job which consists PSes and workers?



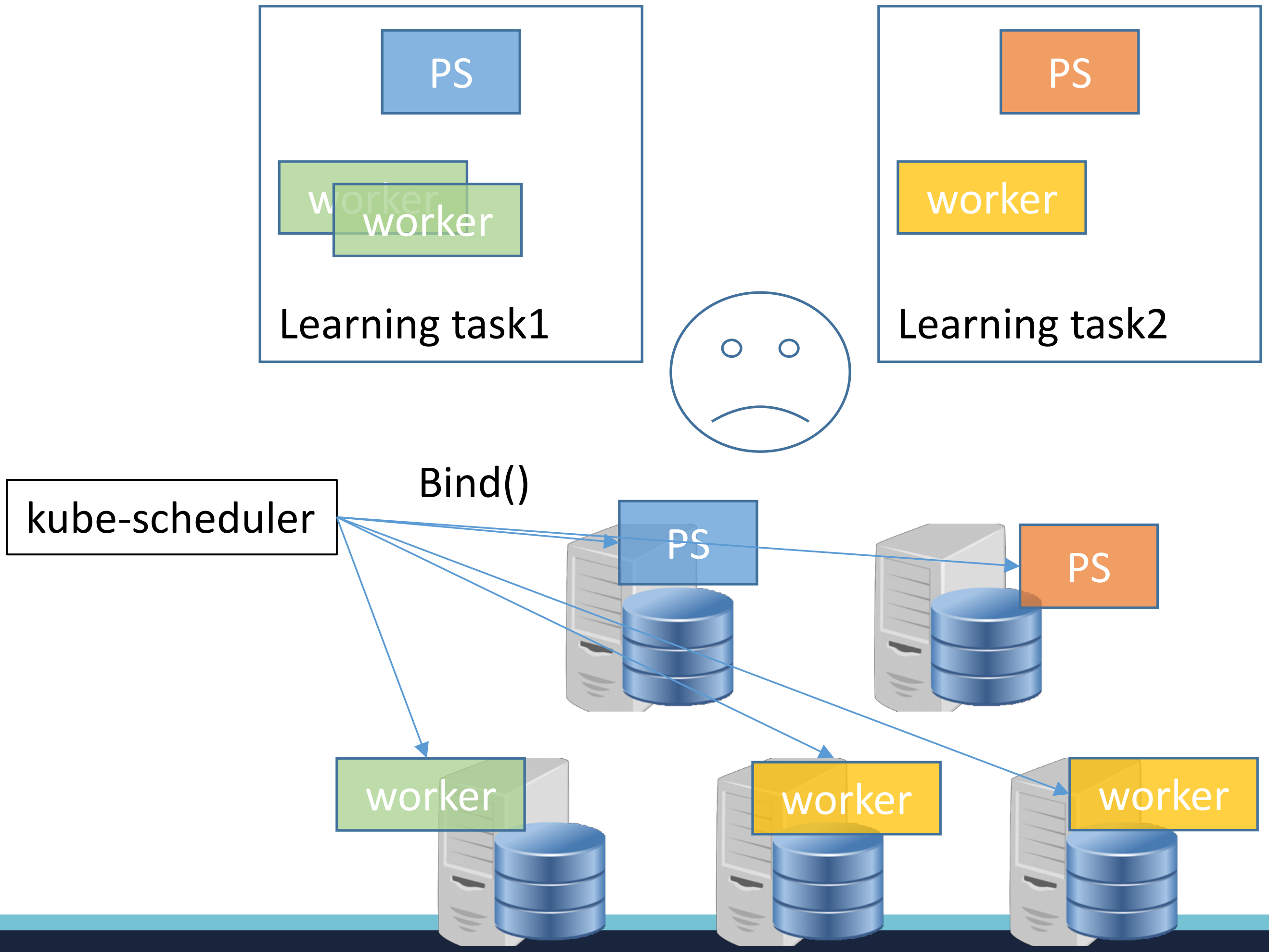
Job was not enough for the purpose

- 1st reason: resources required by PS and worker are quite different
  - worker is GPU hungry, PS doesn't require GPU
  - Their job objects are different
- 2nd reason: PSes and workers which belong to a single learning task must be scheduled at once
  - K8s's job doesn't care about the relation



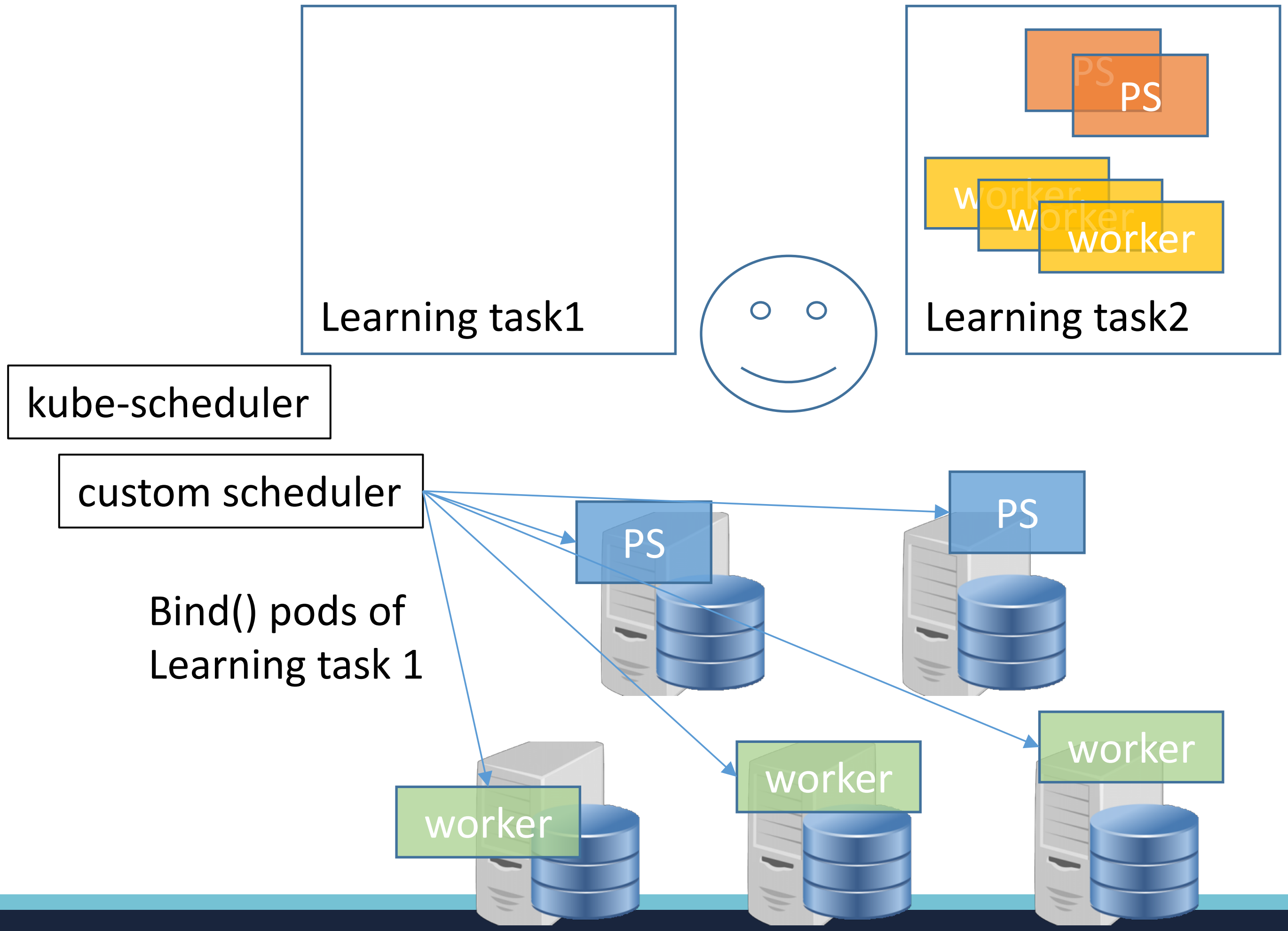
Job was not enough for the purpose

- 1st reason: resources required by PS and worker are quite different
  - worker is GPU hungry, PS doesn't require GPU
  - Their job objects are different
- 2nd reason: PSes and workers which belong to a single learning task must be scheduled at once
  - K8s's job doesn't care about the relation



Pluggable scheduler architecture is helpful

- So we've been developing a custom scheduler which is aware of the learning task concept
  - Multiple schedulers can coexist easily with specifying *PodSpec.SchedulerName*
  - A learning task of each pod can be described in *PodTemplateSpec.ObjectMeta.labels*
  - Thanks to the pluggable architecture, it was quite easy!



Our activities in OSS

- Supporting rdmacg from containers
  - RDMA is a promising interconnect protocol for distributing DL frameworks
    - But resources related to it isn't protected like other resources e.g. CPU, memory
  - These PRs are changes for using rdmacg, the Linux kernel's mechanism for protecting RDMA resources with cgroup, from runc
    - runc side: https://github.com/opencontainers/runc/pull/1612
    - OCI spec side: https://github.com/opencontainers/runc/pull/1612
- Multiple scheduler enhancement for k8s
  - https://github.com/kubernetes/kubernetes/pull/56035
    - A PR for making the default scheduler ResourceVersion aware

Thanks!