# Online Job Scheduling in Distributed Machine Learning Clusters

**Yixin Bao, Yanghua Peng, Chuan Wu, Zongpeng Li**

**The University of Hong Kong & University of Calgary**

# Distributed Machine Learning (ML) Systems





# Training ML Models

- Large ML clusters with hundreds of **GPU** servers
  - **Resource** intensive
  - **Time** consuming

| Model | Dataset | Supercomputer Server | Time |
|---|---|---|---|
| GoogLeNet model | ImageNet | 32 NVIDIA K20 GPUs | 23.4 hours |

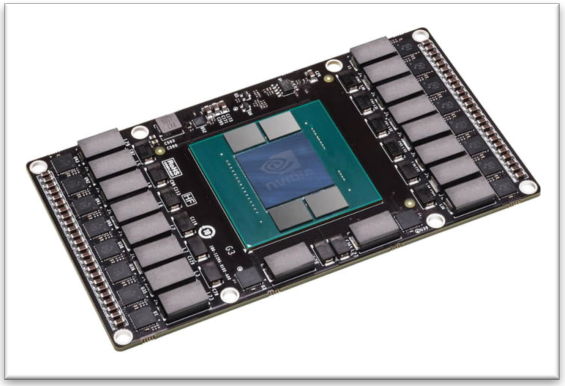
# Motivation

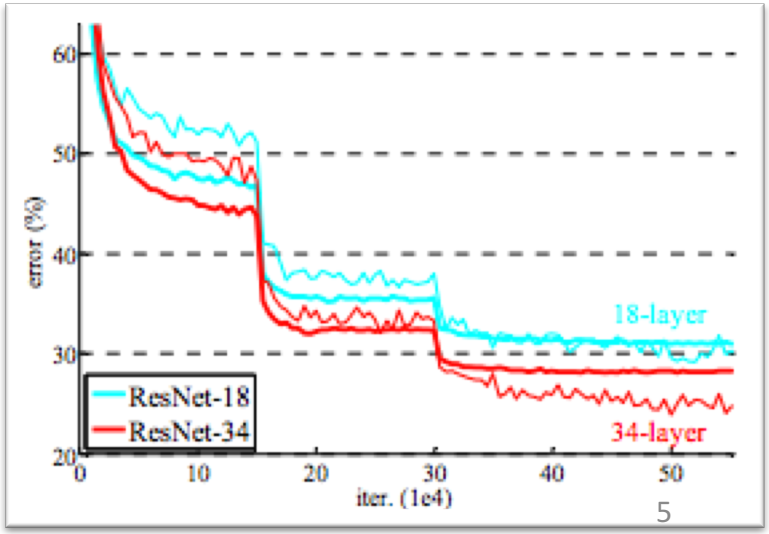
- Borg, YARN, Mesos: Static resource allocation
- Allow a varying number of concurrent workers in a training job
- To fully utilized ML cluster resources


In daytime: Scale out online services as more active users appear
During nights: Free more resources for deep learning experiments

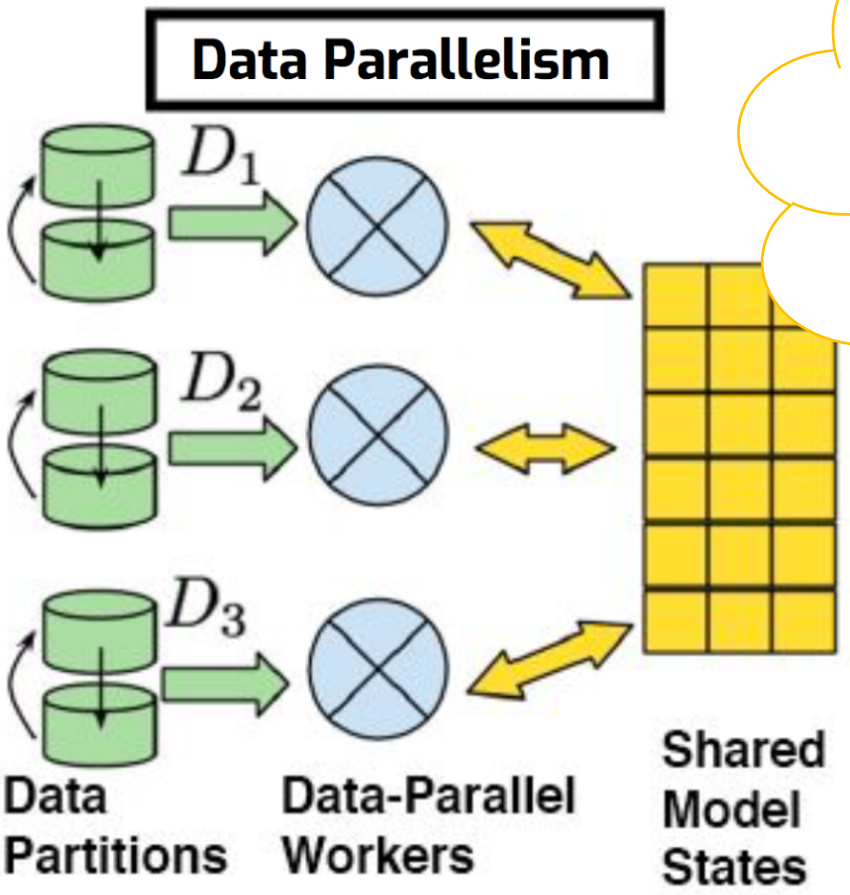
# Challenges

- Efficiently **schedule** submitted training jobs
  - Maximally exploit available server resources
  - Especially the expensive GPU cards
- Complete training in an **expedited** fashion



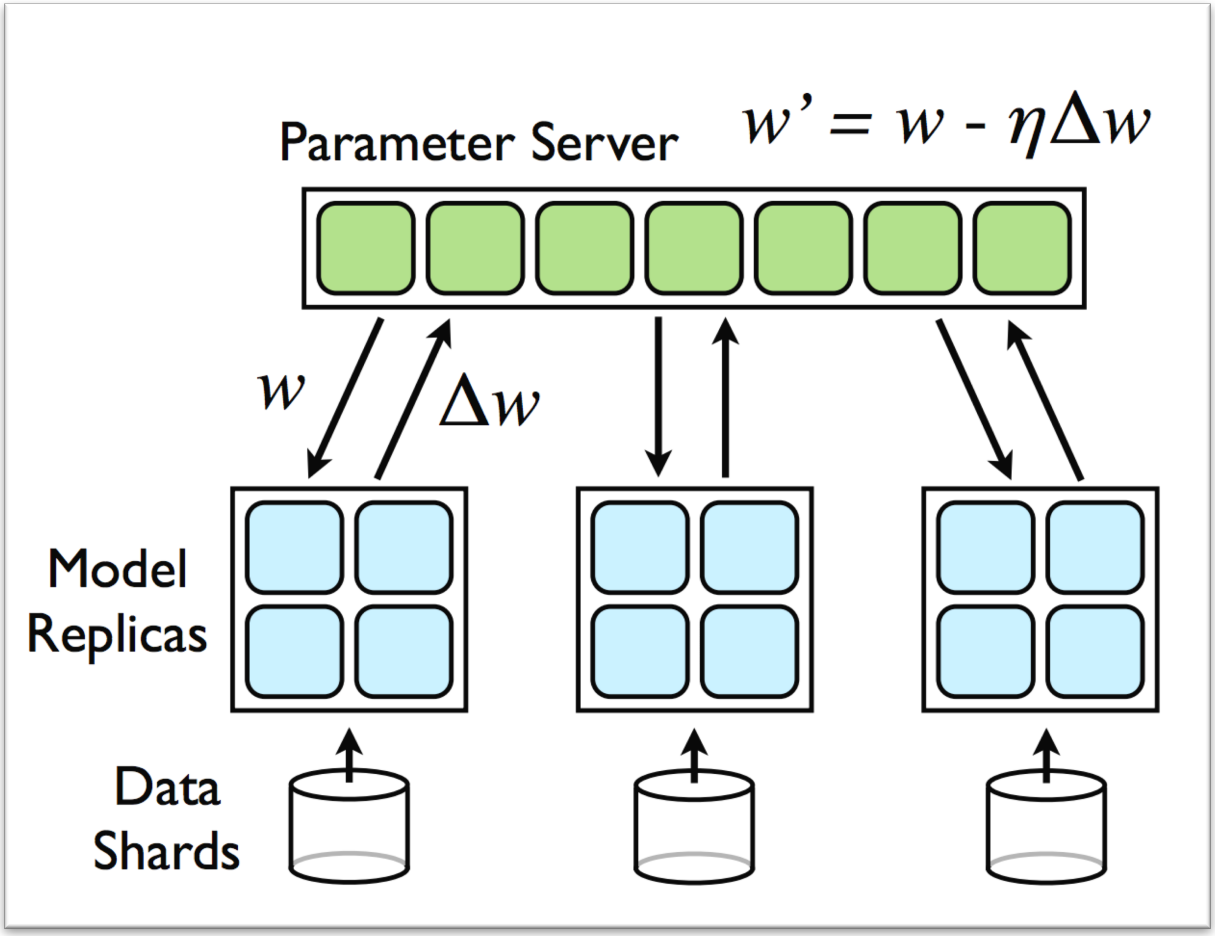


# Distributed Training

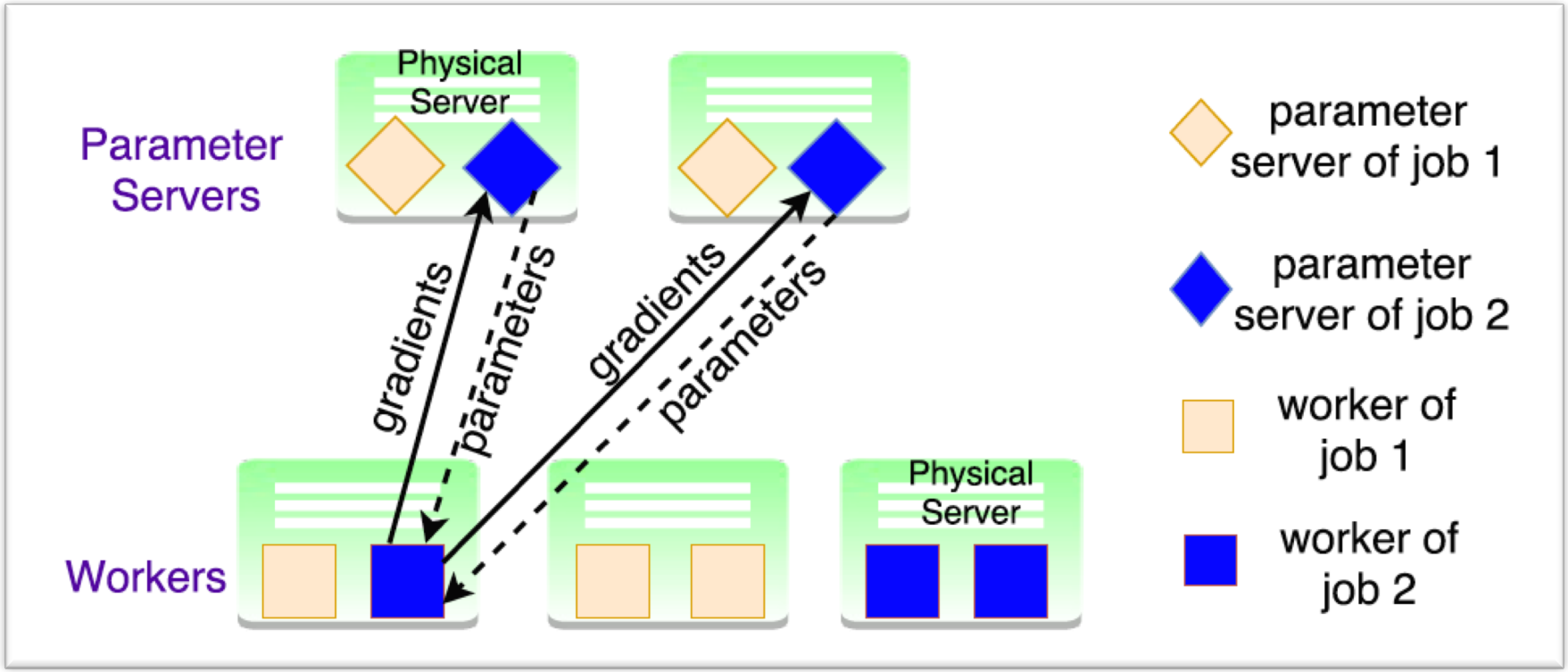



Most ML models can be entirely stored in the memory of modern GPUs

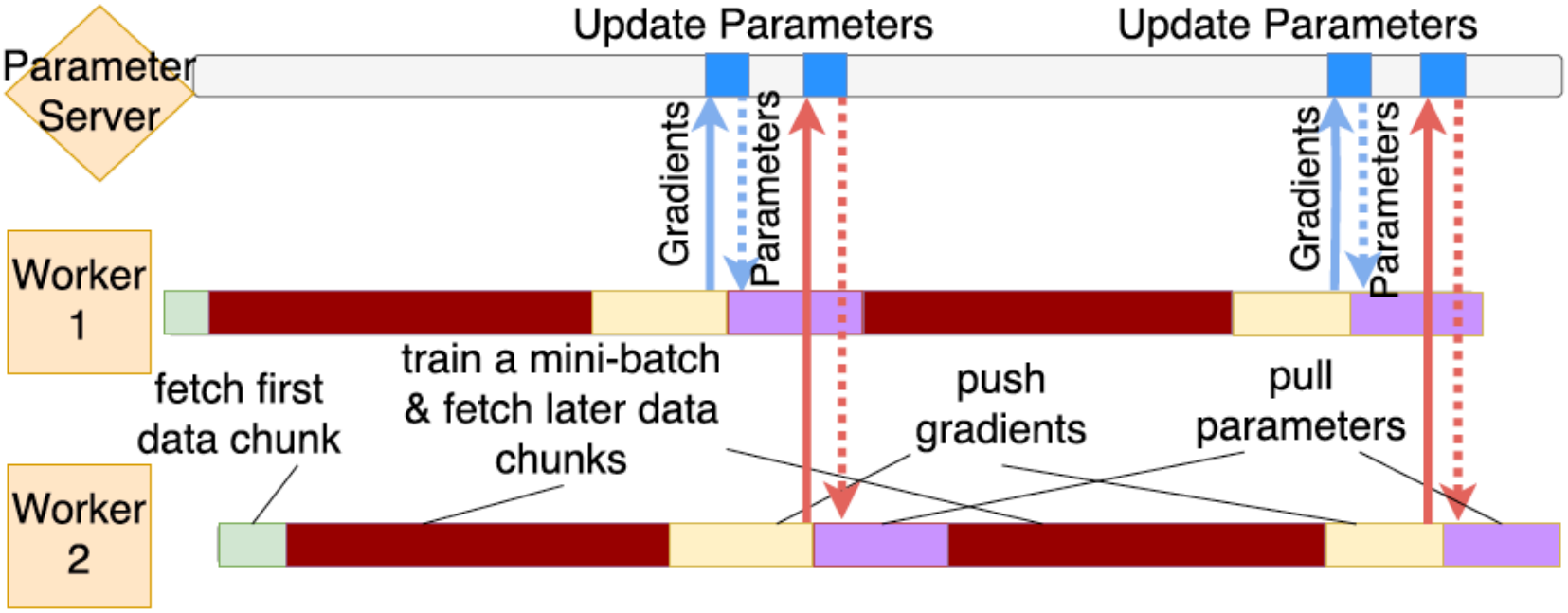
# Parameter Server Framework



# Distributed ML System



# Asynchronous Training Workflow



# An Online Approach

- **Maximizing overall job utility** in an ML cluster
  - Non-increasing with job completion time
- Which job should be accepted?
- How many workers and parameter servers should be allocated for each job in different time slots?
- Which physical server should workers and parameter servers be placed on?

# Problem Formulation

Accept job $i$ or not

$$\max \sum_{i \in [I]} x_i f_i(\hat{t}_i - a_i)$$

Utility of job $i$

**Online decision variables:**

$x_i$: accept job $i$ or not

$\hat{t}_i$: completion time of job $i$

# Problem Formulation

$$\sum_{t\in[T]}\sum_{h\in[H]} y_{ih}(t) \geq E_i N_i M_i(\tau_i + 2e_i/b_i)x_i, \forall i \in [I]$$

accomplish training for $E_i$ epochs

$$\sum_{h\in[H]} y_{ih}(t) \leq N_i x_i, \forall i \in [I], t \in [T] : t \geq a_i$$

limit concurrent number of workers

$$\sum_{i\in[I]} w_i^r y_{ih}(t) \leq c_h^r, \forall t \in [T], r \in [R], h \in [H]$$

$$\sum_{i\in[I]} s_i^r z_{ik}(t) \leq c_k^r, \forall t \in [T], r \in [R], k \in [K]$$

resource capacity constraints on physical machines

## Online decision variables:

$y_{ih}(t)$: # of workers of job $i$ deployed on server $h$ in $t$

$z_{ik}(t)$: # of parameter servers of $i$ deployed on server $k$ in $t$

# Problem Formulation

$$\sum_{h\in[H]} y_{ih}(t)b_i \leq \sum_{k\in[K]} z_{ik}(t)B_i, \forall i \in [I], t \in [T]$$

$$\sum_{k\in[K]} z_{ik}(t) \leq \sum_{h\in[H]} y_{ih}(t), \forall i \in [I], t \in [T]$$

$$\hat{t}_i = \arg\max_{t\in[T]}\{\sum_{h\in[H]} y_{ih}(t) > 0\}, \forall i \in [I]$$

**bandwidth guarantee**

limit the number of parameter servers

**completion time slot of job *i***

# Problem Reformulation - Primal

$$\max_{\mathbf{x}} \sum_{i\in[I]} \sum_{l\in\mathcal{L}_i} x_{il} f_i(t_{il} - a_i)$$

s.t.

$$\sum_{i\in[I]} \sum_{l:t\in l, h\in(t,l)} w_i^r y_{ih}^l(t) x_{il} \le c_h^r, \forall t \in [T], r \in [R], h \in [H]$$

$$\sum_{i\in[I]} \sum_{l:t\in l, k\in(t,l)} s_i^r z_{ik}^l(t) x_{il} \le c_k^r, \forall t \in [T], r \in [R], k \in [K]$$

resource capacity constraints

$$\sum_{l\in\mathcal{L}_i} x_{il} \le 1, \forall i \in [I]$$

$\mathcal{L}_i$: the set of feasible schedules for job $i$

**Online decision variables:**

$x_{il}$: whether job $i$ is admitted and scheduled according to schedule $l$

# Problem Reformulation - Dual

$$\min \sum_{i\in[I]} \mu_i + \sum_{t\in[T]} \sum_{h\in[H]} \sum_{r\in[R]} p_h^r(t) c_h^r + \sum_{t\in[T]} \sum_{k\in[K]} \sum_{r\in[R]} q_k^r(t) c_k^r$$

$$s.t.\ \mu_i \geq f_i(t_{il} - a_i) - \sum_{t\in l} \sum_{h\in(t,l)} \sum_{r\in[R]} p_h^r(t) w_i^r y_{ih}^l(t)$$

$$- \sum_{t\in l} \sum_{k\in(t,l)} \sum_{r\in[R]} q_k^r(t) s_i^r z_{ik}^l(t), \forall i \in [I], l \in \mathcal{L}_i$$

$$\mu_i \geq 0, \forall i \in [I]$$

**Dual variables:**

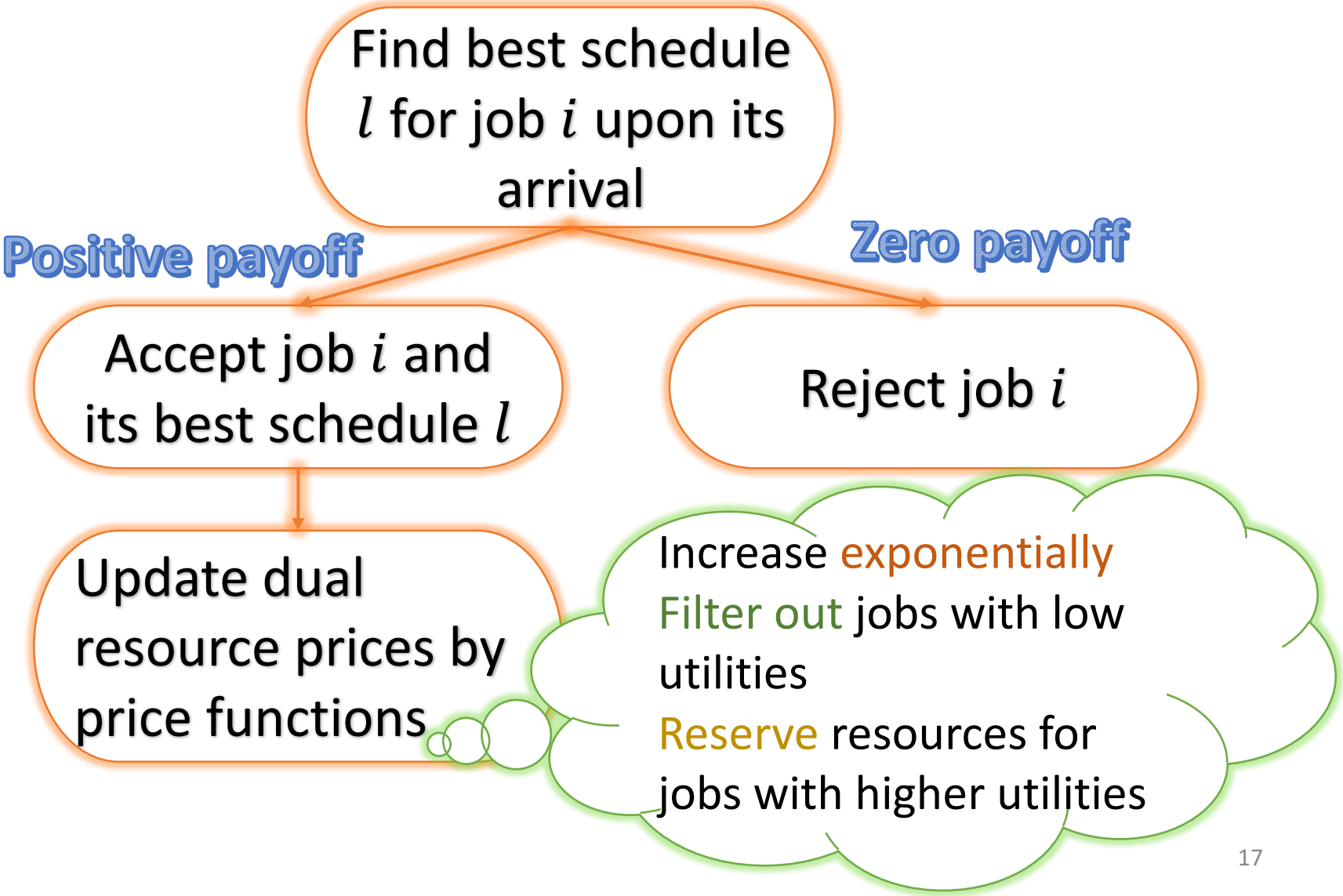
$p_h^r(t)$, $q_k^r(t)$: the unit cost for type-$r$ resource on the server $h$, $k$ in time $t$

$\mu_i$: the payoff of admitting job $i$ according to the best schedule $l$

# Online Algorithm

- An efficient approach for admission control
  - Online primal-dual framework
  - Accept jobs with positive payoff
- An efficient subroutine to find the best schedule for each job
  - Dynamic programming + Greedy method

# Online Algorithm



# Subroutine for Finding Best Job Schedule

- Minimize resource cost among all feasible schedules
- Finish training workload $E$ by time $t$ = $\min\{$Finish workload $d$ at time slot $t$ + Finish training workload $E - d$ by time $t - 1\}$, $0 \leq d \leq E$
- Fulfill workload $d$ at time slot $t$
  - Sort by resource cost
  - Select worker or parameter server greedily

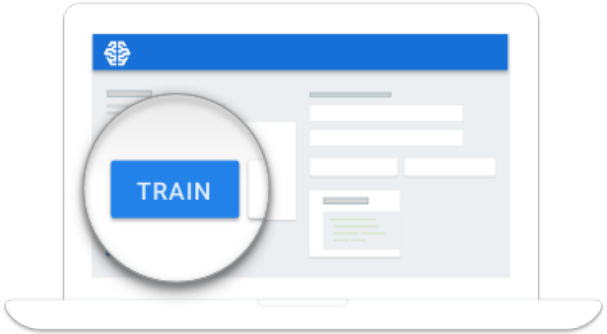
# Theoretical Analysis

- Optimality of Subroutine
- Correctness
- Polynomial Running Time
- Competitive Ratio $\alpha = \max_{r \in [R]}(1, \ln \frac{U_1^r}{L_1}, \ln \frac{U_2^r}{L_2})$

# Trace-driven Simulation

- Cluster configuration: Amazon EC2 CPU/GPU instances


- Job resource demand: Machine learning job trace from Dorm [cite]

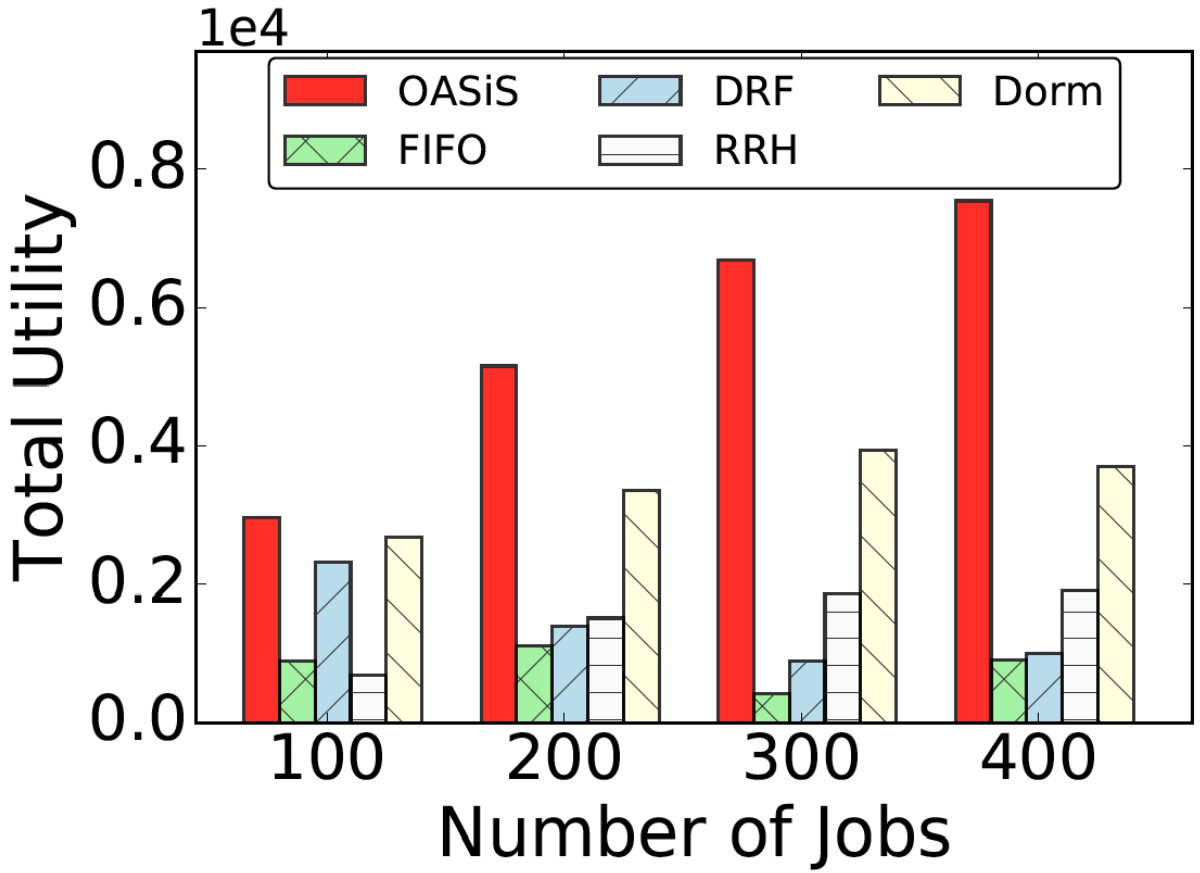



- Job arrival rate: Google cluster trace

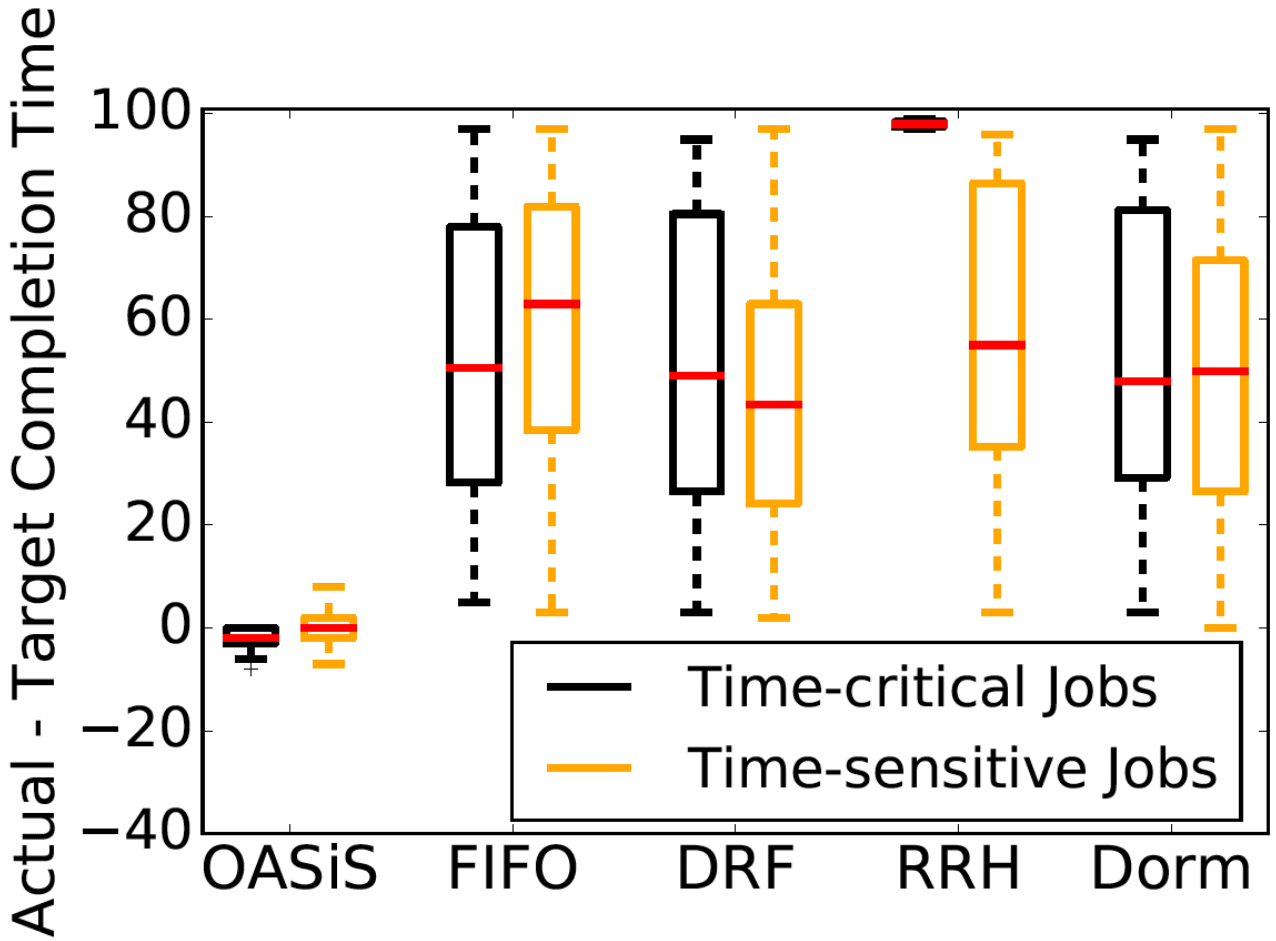
# Testbed Experiments


- Implement a distributed ML system based on MXNet with Kubernetes 1.6
- Modify MXNet to support dynamic adjustment of worker/parameter server numbers
- Implement custom schedulers to replace the default one in Kubernetes

# Comparison with heuristics



# Comparison with heuristics



thank you!