ML Training with Cloud GPU Shortages: Is Cross-Region the Answer?

Foteini Strati Paul Elvinger Tolga Kerimoglu Ana Klimovic

ETH zürich

Increased demand for GPUs

Recently, there have been huge advances in the ML field

➡ The size of models reaches trillions of parameters!

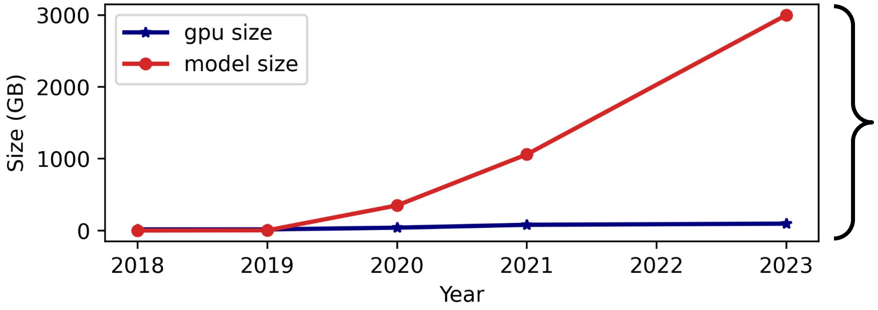
➡ However, the increase in memory per GPU is not proportional

Increased demand for GPUs

Recently, there have been huge advances in the ML field

➡ The size of models reaches trillions of parameters!

➡ However, the increase in memory per GPU is not proportional

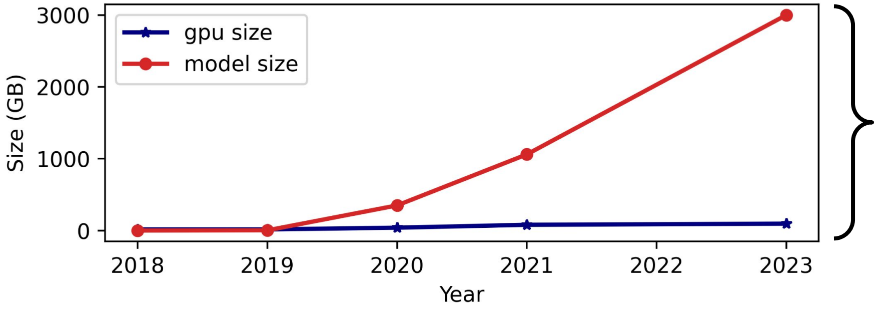


Increased demand for GPUs

Recently, there have been huge advances in the ML field

➡ The size of models reaches trillions of parameters!

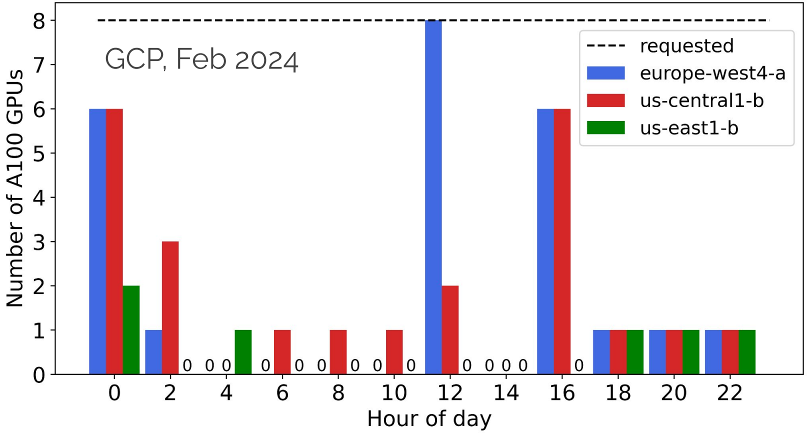
➡ However, the increase in memory per GPU is not proportional



More and more GPUs are needed to train these models

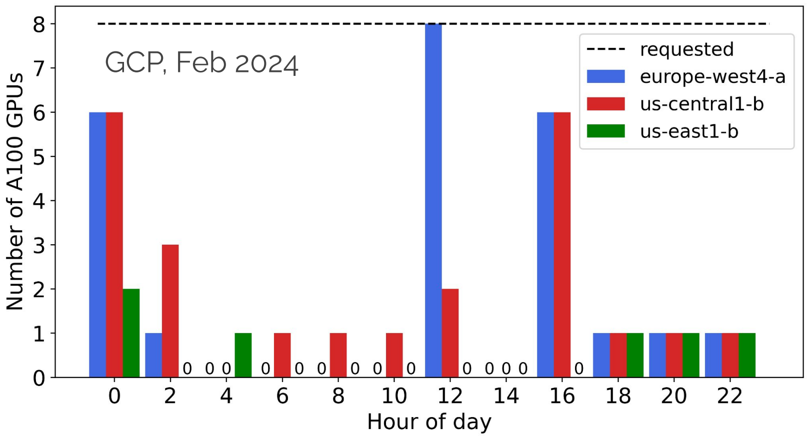
GPU scarcity in the public cloud

This increased GPU demand has led to a **GPU scarcity** in the public cloud



GPU scarcity in the public cloud

This increased GPU demand has led to a **GPU scarcity** in the public cloud



We found failover to be especially valuable for scarce resources (e.g., large CPU or GPU VMs). For example, depending on request timing, it took 3–5 and 2–7 location attempts to allocate 8 V100 and 8 T4 GPUs on AWS, respectively.

SkyPilot, NSDI'23

sioned GPU memory, and 9% of the failed jobs were induced by GPU out-of-memory events (OOMs). GPUs are becoming more scarce due to wide-spread learning demands [11, 22, 47]. For example, OpenAI is heavily GPU-limited at present and GPU shortage is delaying a lot of their short-term plans [22].

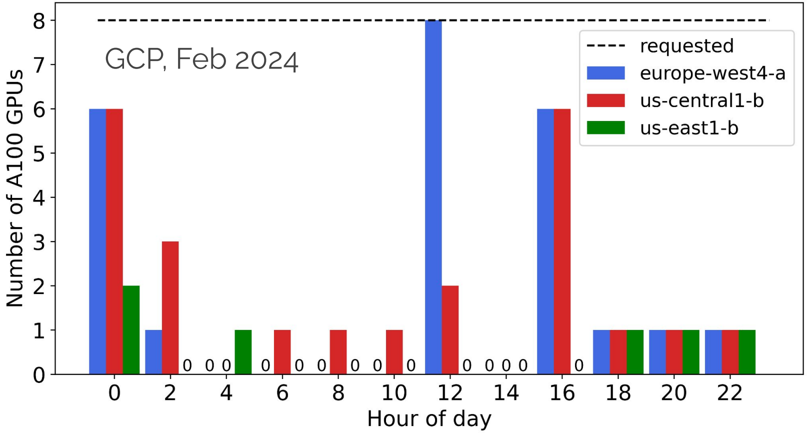
Cheng et al, SoCC'23

pools generally have lower utilization. We conjecture that this is due to the greater scarcity of GPUs in the public cloud.

Chugh et al, SoCC'23

GPU scarcity in the public cloud

This increased GPU demand has led to a **GPU scarcity** in the public cloud



We found failover to be especially valuable for scarce resources (e.g., large CPU or GPU VMs). For example, depending on request timing, it took 3–5 and 2–7 location attempts to allocate 8 V100 and 8 T4 GPUs on AWS, respectively.

[SkyPilot, NSDI'23]

sioned GPU memory, and 9% of the failed jobs were induced by GPU out-of-memory events (OOMs). GPUs are becoming more scarce due to wide-spread learning demands [11, 22, 47]. For example, OpenAI is heavily GPU-limited at present and GPU shortage is delaying a lot of their short-term plans [22].

[Cheng et al, SoCC'23]

pools generally have lower utilization. We conjecture that this is due to the greater scarcity of GPUs in the public cloud.

[Chugh et al, SoCC'23]

A natural way to acquire more GPUs would be to spread across cloud regions

Implications of geo-distributed training

However, when crossing the cloud zone boundaries there are two main implications:

Implications of geo-distributed training

However, when crossing the cloud zone boundaries there are two main implications:

1. **Reduced Network bandwidth and increased latency**

Traffic between	Bandwidth (GB/sec)	Latency (ms)
Same AZ (US)	1.45	< 1
Diff. AZ, same region (US)	1.42	0.9
Diff. regions (US)	0.63	31
Diff. continents (US/EU)	0.18	102

a2-highgpu-1g, GCP

Implications of geo-distributed training

However, when crossing the cloud zone boundaries there are two main implications:

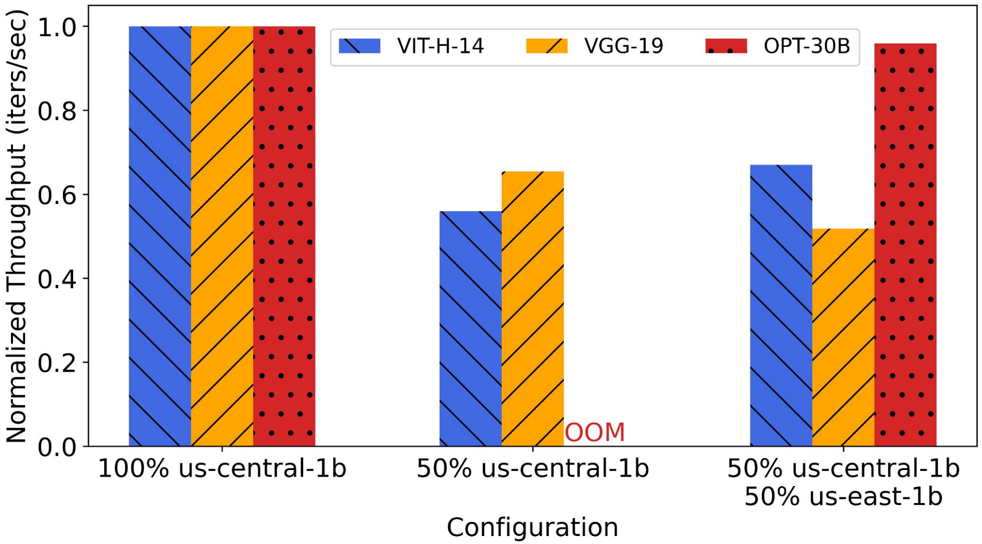
1. Reduced Network bandwidth and increased latency
2. **Charges for data exchange**

Traffic between	Bandwidth (GB/sec)	Latency (ms)	Cost/GB ($)
Same AZ (US)	1.45	< 1	0
Diff. AZ, same region (US)	1.42	0.9	0.01
Diff. regions (US)	0.63	31	0.02
Diff. continents (US/EU)	0.18	102	0.05

a2-highgpu-1g, GCP

Implications of geo-distributed training

The impact on training throughput and cost depends on the model and cluster configuration



Our work

We study **when and how it makes sense to use GPUs across zones and regions for large-scale, distributed training**

Methodology

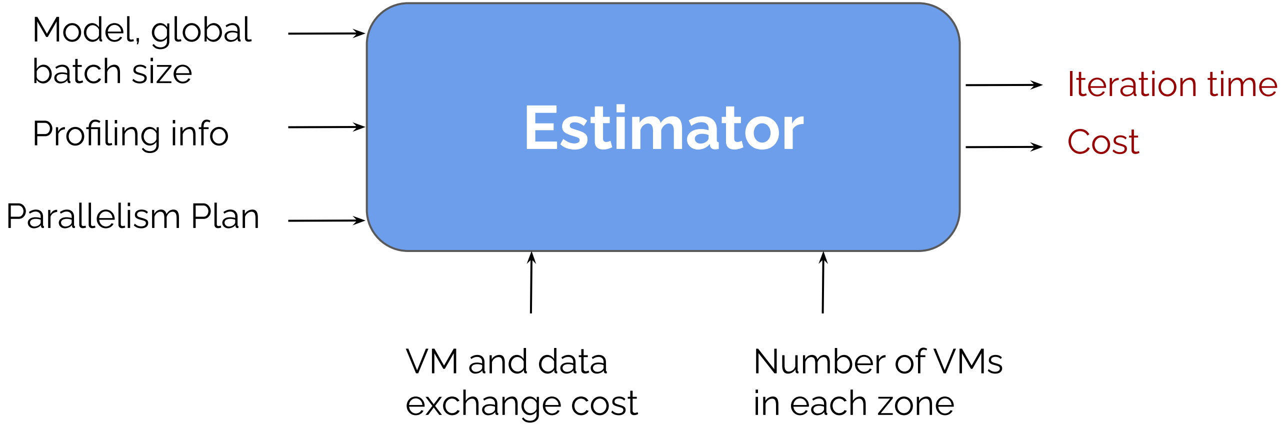
2-step approach:

1. **Profiling** (model + cloud-specific info)
2. **Throughput and cost estimation for data and pipeline parallelism**

Methodology

2-step approach:

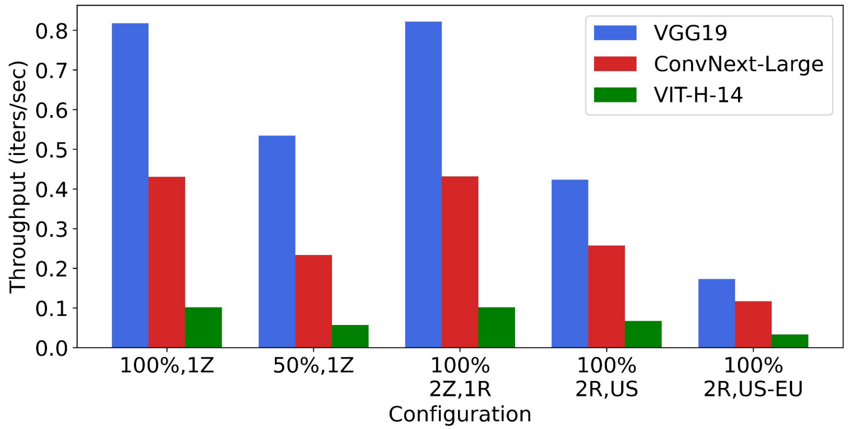
1. **Profiling** (model + cloud-specific info)
2. **Throughput and cost estimation for data and pipeline parallelism**

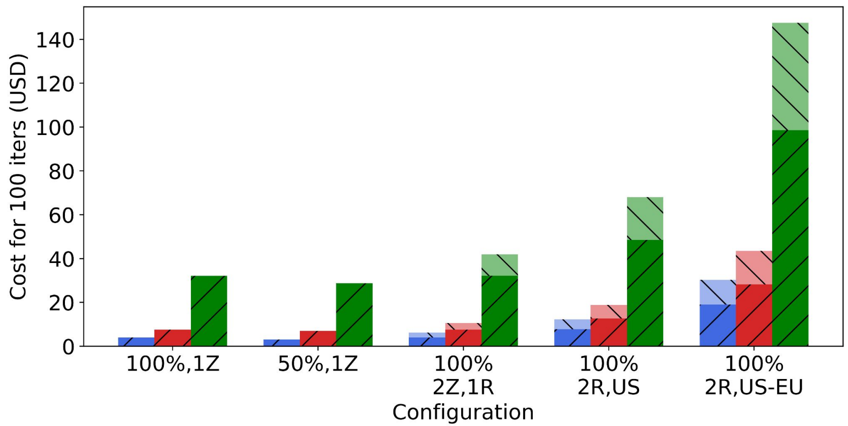


Evaluation and Key Insights

Data Parallelism

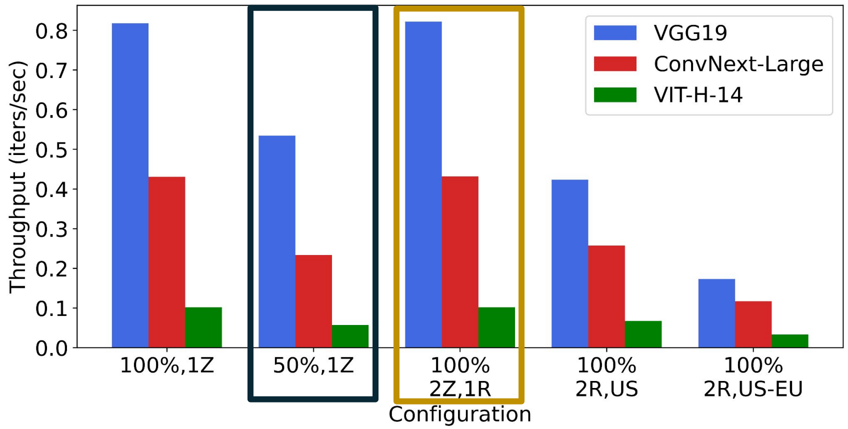
If only λ*N GPUs are available in one zone (e.g. λ=0.5), when to use GPUs from multiple zones?

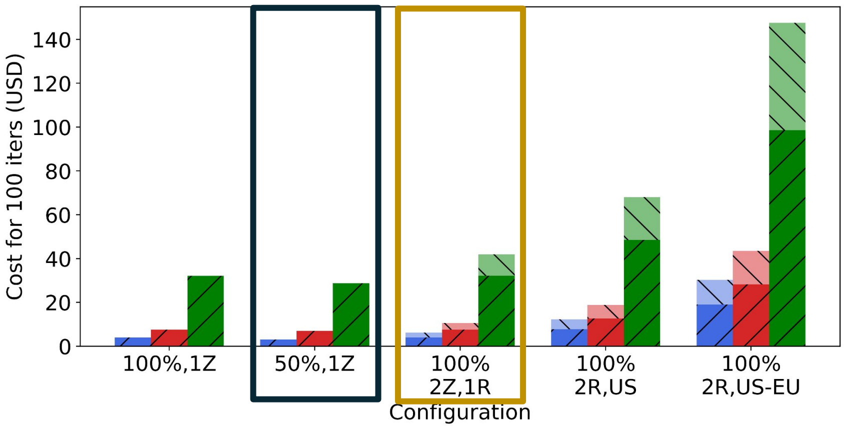




Data Parallelism

If only λ*N GPUs are available in one zone (λ=0.5 or λ=0.75), when to use GPUs from multiple zones?

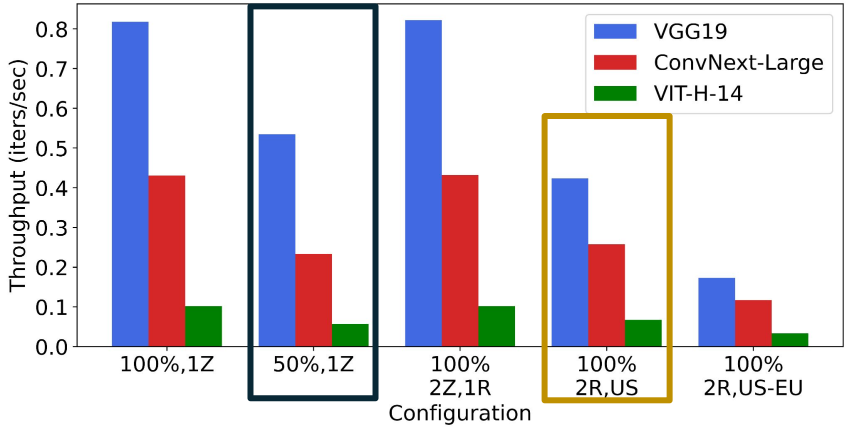


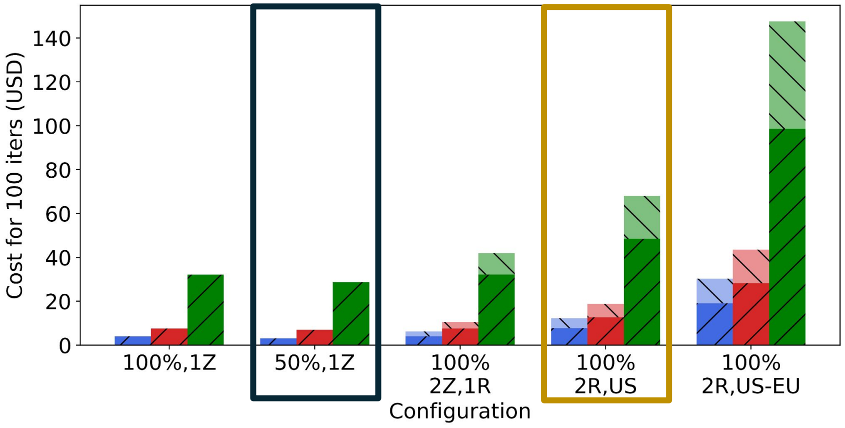


- **Using zones from the same region is always beneficial**

Data Parallelism

If only λ*N GPUs are available in one zone (λ=0.5 or λ=0.75), when to use GPUs from multiple zones?

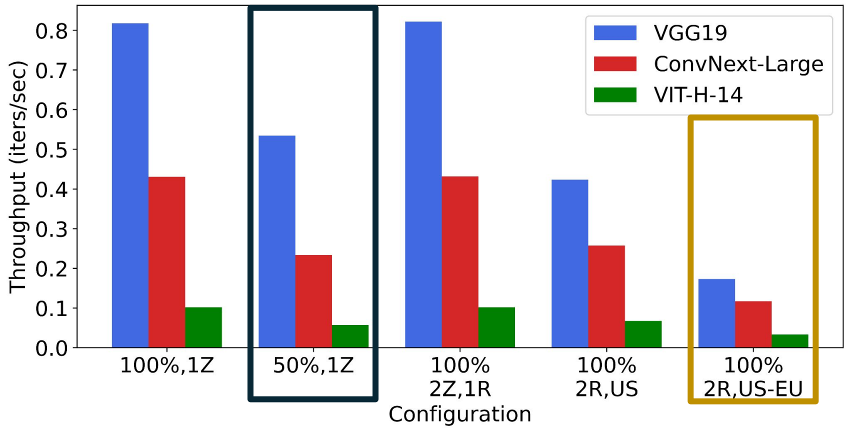


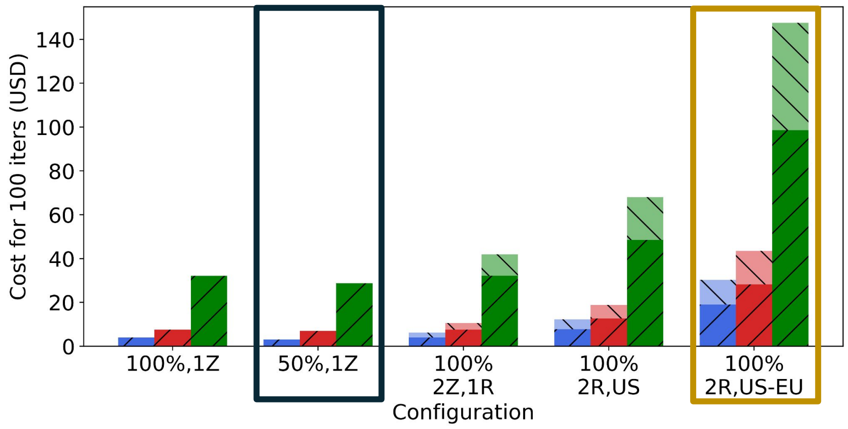


- Using zones from the same region is always beneficial
- **Multiple regions in the same continent (e.g. US) has varying effects depending on the model (ratio of compute to communication)**

Data Parallelism

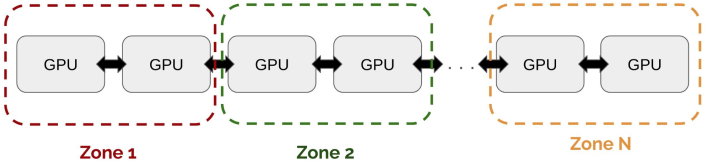
If only λ*N GPUs are available in one zone (λ=0.5 or λ=0.75), when to use GPUs from multiple zones?

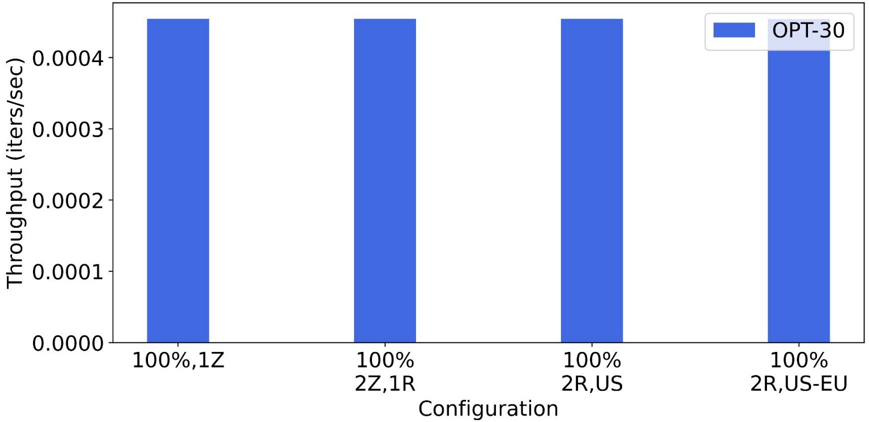


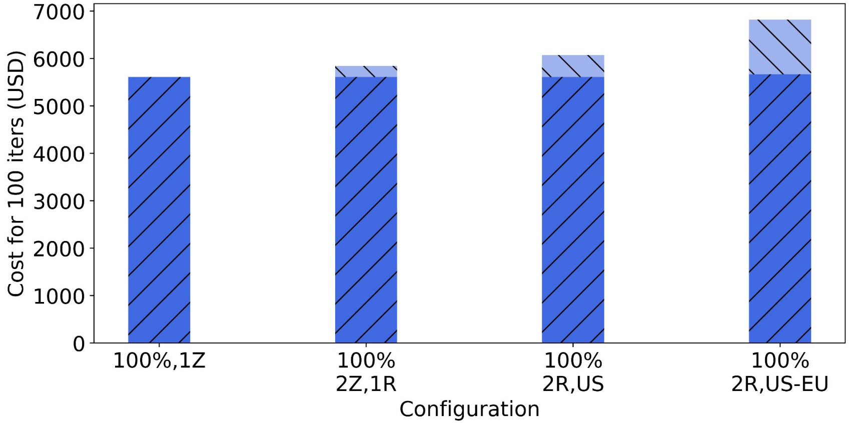


- Using zones from the same region is always beneficial
- Multiple regions in the same continent (e.g. US) has varying effects depending on the model (ratio of compute to communication)
- **Inter-continental training is detrimental to both throughput and cost**

Pipeline Parallelism

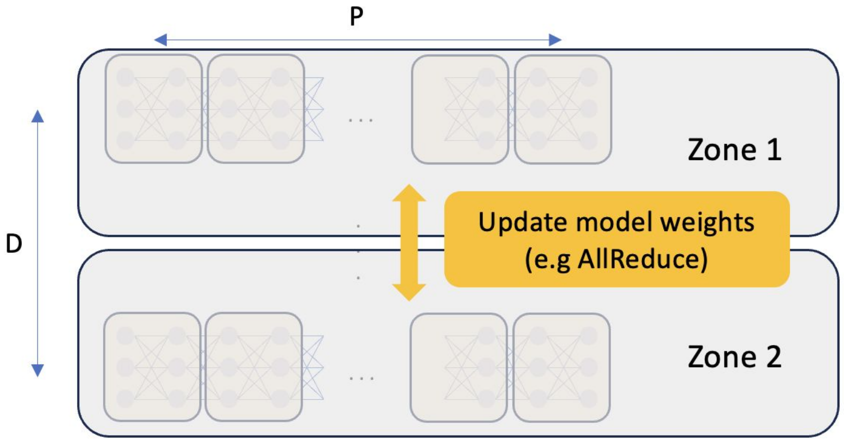




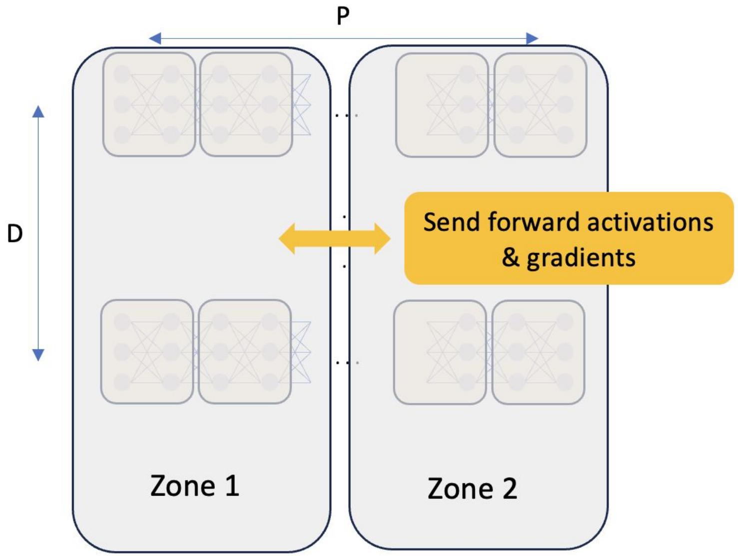


- Using multiple regions for pipeline parallelism has **marginal effects on throughput**:
    - Pipelining of computation with activation/gradient exchange
    - Small-sized per-layer activations even for big models
- However, the cost for multi-region or inter-continental setups is significant

Data + Pipeline Parallelism

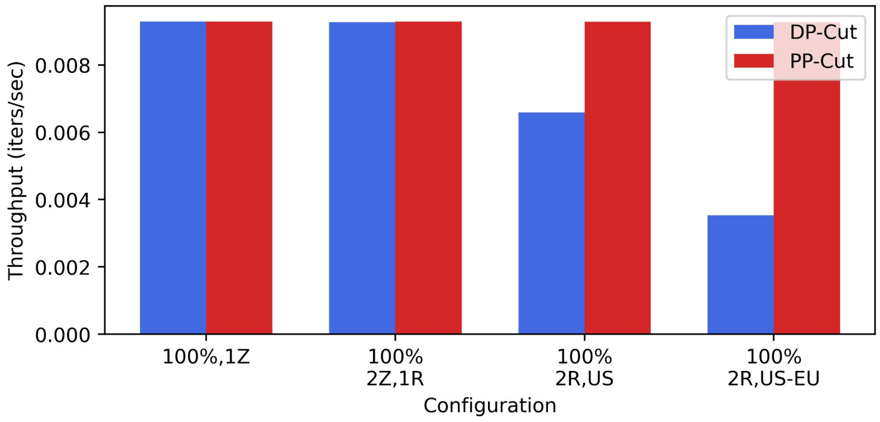


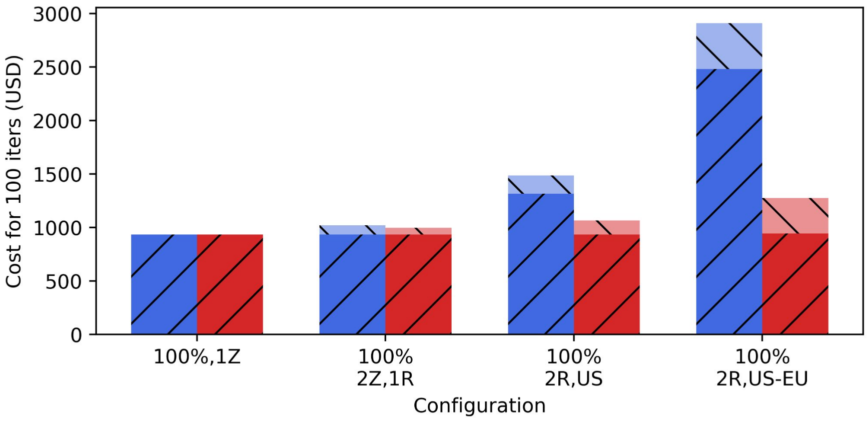
DP-Cut



PP-Cut

Data + Pipeline Parallelism



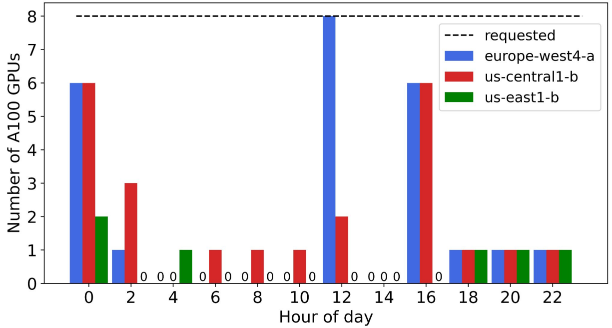


- **If we keep all the All-Reduce traffic in a single zone, multi-region training is beneficial**

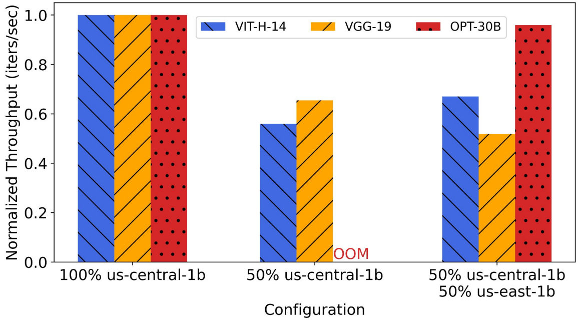
Key Insights

- Spreading training **across zones** in the same region has **minimal effects in training throughput**
- **Pipeline parallelism is much more tolerant to crossing zone boundaries compared to data parallelism**
- **Across-continental training is detrimental** to data parallelism
- **With 2D parallelism, maintain data parallel traffic within one region**

Advances in ML have led to GPU shortage in the public cloud

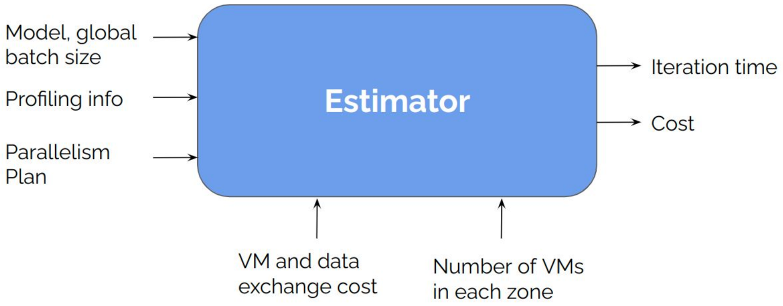


Using GPUs from multiple cloud regions can help the training of large models, but has 2 implications:
reduced network bandwidth and **data exchange charges**



The impact on training throughput and cost depends on the model and cluster configuration

We study **when and how does it make sense to use GPUs across zones and regions for large-scale, distributed training**



Key Insights:

- Spreading training across zones in the same region has minimal effects in training throughput
- Pipeline parallelism is much more tolerant to crossing zone boundaries compared to data parallelism
- Across-continental training is detrimental to data parallelism
- With 2D parallelism, maintain data parallel traffic within one region

Backup Slides

Iteration time estimation

- **Data parallelism**

$$T_i = (t_f + t_b) \cdot (ga - 1) + max((t_f + t_b), 2 \cdot (N - 1) \cdot \frac{M}{N \cdot b}) + t_u$$

Iteration time estimation

- **Data parallelism**

We consider:

- **Computation (with gradient accumulation)**
- All-reduce based synchronization
- Computation-Communication overlap

$$T_i = \boxed{(t_f + t_b) \cdot (ga - 1)}$$
$$+ max((t_f + t_b), 2 \cdot (N - 1) \cdot \frac{M}{N \cdot b}) + t_u$$

Iteration time estimation

- **Data parallelism**

We consider:

- Computation (with gradient accumulation)
- **All-reduce based synchronization**
- Computation-Communication overlap

$$T_i = (t_f + t_b) \cdot (ga - 1) + max((t_f + t_b), \boxed{2 \cdot (N - 1) \cdot \frac{M}{N \cdot b})} + t_u$$

Iteration time estimation

- **Data parallelism**

We consider:

- Computation (with gradient accumulation)
- All-reduce based synchronization
- **Computation-Communication overlap**

$$T_i = (t_f + t_b) \cdot (ga - 1) + \boxed{max((t_f + t_b), 2 \cdot (N - 1) \cdot \frac{M}{N \cdot b})} + t_u$$

Iteration time estimation

- **Pipeline parallelism**

We adapt the formula followed by [1] for 1F1B [2] microbatch scheduling

[1] Li et al, AMP: Automatically Finding Model Parallel Strategies with Heterogeneity Awareness, NeurIPS'22

[2] Narayanan et al, PipeDream: Generalized Pipeline Parallelism for DNN Training, SOSP'19

Iteration time estimation

- **Pipeline parallelism**

We adapt the formula followed by [1] for 1F1B [2] microbatch scheduling

- **Data + Pipeline parallelism**

We assume a grid of *D* pipelines, each with *P* stages

$$t_{sync} = 2 \cdot (D - 1) \cdot \frac{M}{D \cdot P \cdot b_{min}}$$

$$t_{iter} = t_{pp} + t_{sync}$$

[1] Li et al, AMP: Automatically Finding Model Parallel Strategies with Heterogeneity Awareness, NeurIPS'22

[2] Narayanan et al, PipeDream: Generalized Pipeline Parallelism for DNN Training, SOSP'19

Cost estimation

- **Computation cost:** $C_{comp} = t_{iter} * \sum_{\text{zone } i} (num\_vm_i \cdot cost_i)$
- **Communication cost:** $C_{comm} = \sum data_{ij} \cdot c_{ij}$
- **Total cost:** $C_{comp} + C_{comm}$

t_{iter}	Iteration time
$data_{ij}$	Data exchanged between workers i and j
c_{ij}	Cost of exchanging data between workers i and j
num_vm_i	Numbers of VMs at zone i
$cost_i$	Cost of a VM at zone i