ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

Alpa

Automating Inter- and Intra-Operator Parallelism for Distributed Deep Learning

Lianmin Zheng[1,*] **Zhuohan Li**[1,*] Hao Zhang[1,*] Yonghao Zhuang[4]
Zhifeng Chen[3] Yanping Huang[3] Yida Wang[2] Yuanzhong Xu[3] Danyang Zhuo[6]
Eric P. Xing[5] Joseph E. Gonzalez[1] Ion Stoica[1]

*[1]**UC Berkeley*** *[2]Amazon Web Services* *[3]Google* *[4]Shanghai Jiao Tong University*
[5]MBZUAI & Carnegie Mellon University *[6]Duke University*

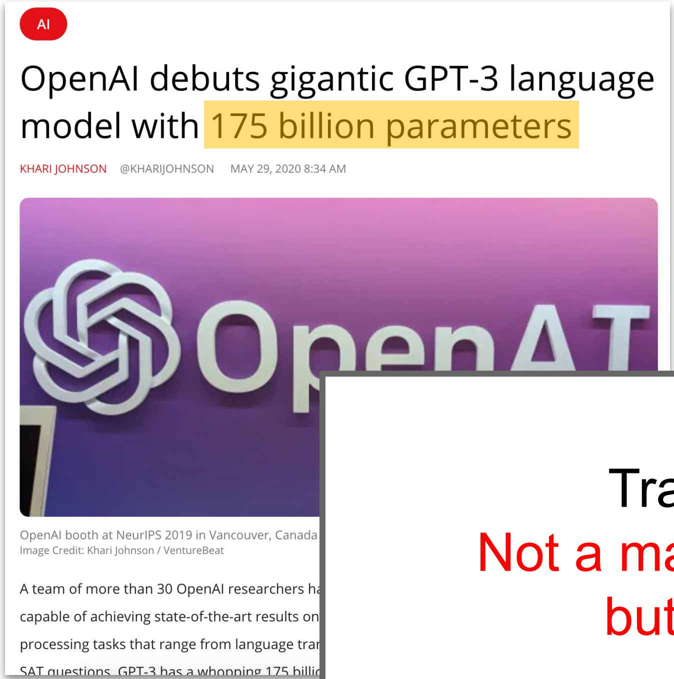
*Equal contribution

OSDI 22

Background

Alpa

Large Models Enable Breakthroughs in Machine Learning

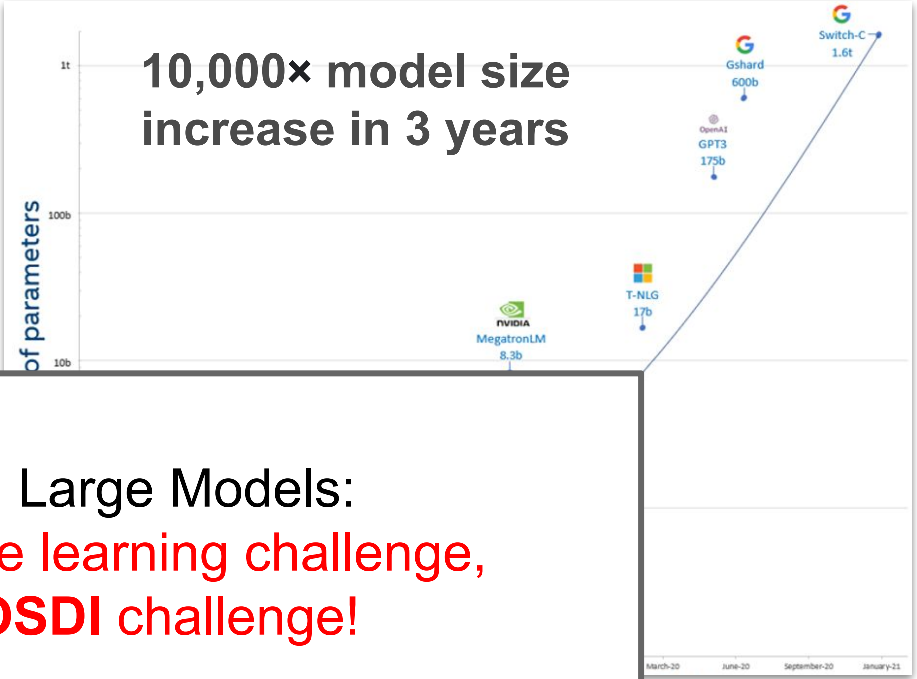

AI

OpenAI debuts gigantic GPT-3 language model with 175 billion parameters

KHARI JOHNSON @KHARIJOHNSON MAY 29, 2020 8:34 AM

OpenAI booth at NeurIPS 2019 in Vancouver, Canada
Image Credit: Khari Johnson / VentureBeat

A team of more than 30 OpenAI researchers ha
capable of achieving state-of-the-art results on
processing tasks that range from language tra
SAT questions. GPT-3 has a whopping 175 billi



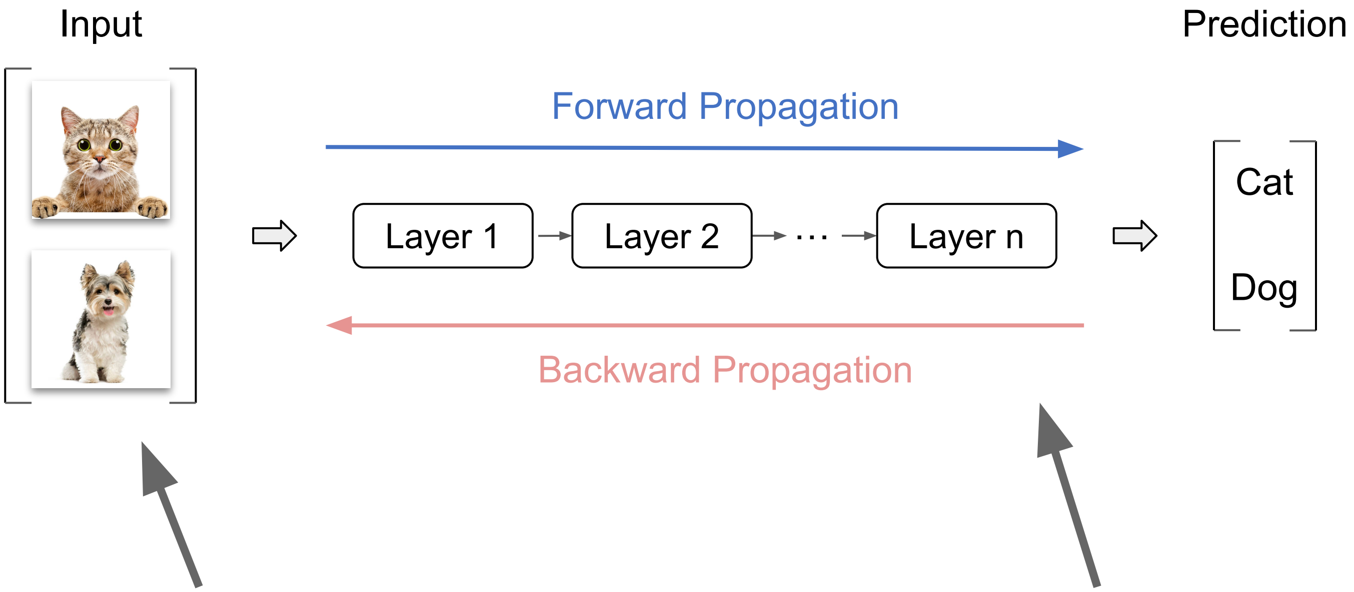
Training Large Models:

Not a machine learning challenge,

but an **OSDI** challenge!

Image source: towardsdatascience.com

What are System Challenges?



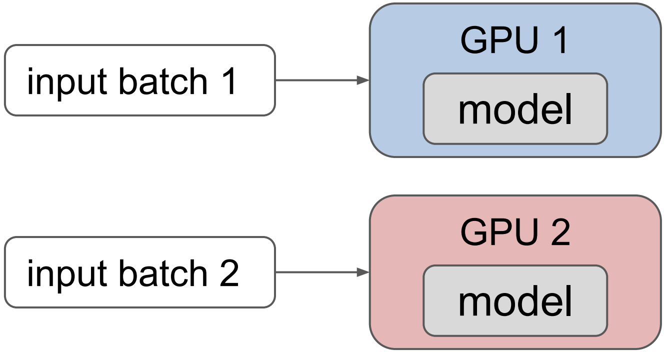
1. What if the **input dataset** is very large?

2. What if the **model** is very large?

What are System Challenges?

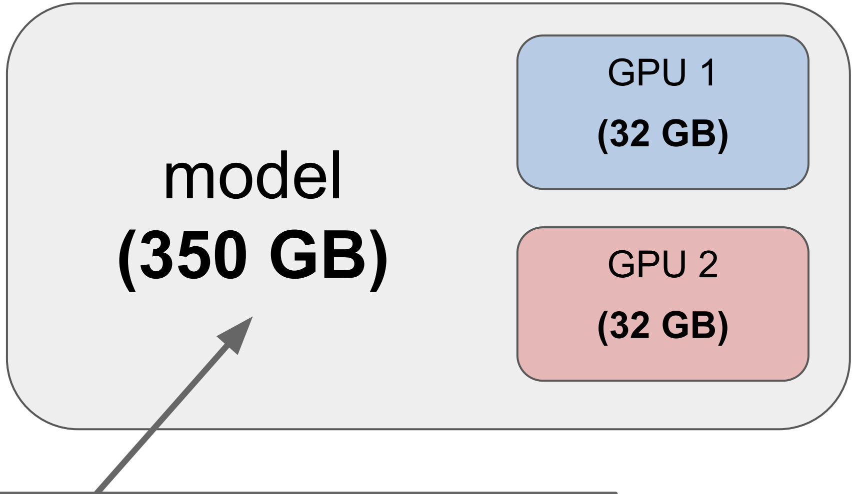
1. What if the input dataset is very large?

😃 **Easy.**
Use data parallelism: partition input data and replicate the model



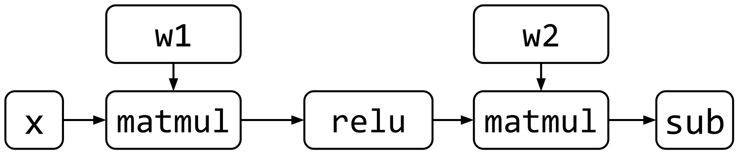
2. What if the model is very large?

😖 **Hard !!**



Challenge: How to partition a computational graph?

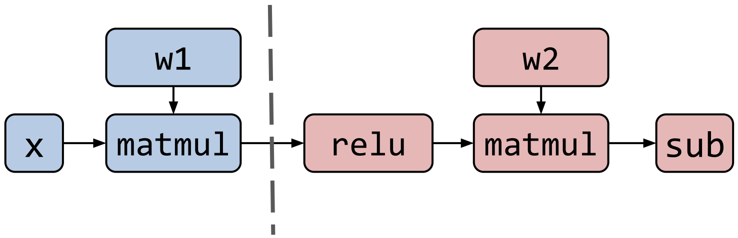
Partition Computational Graphs



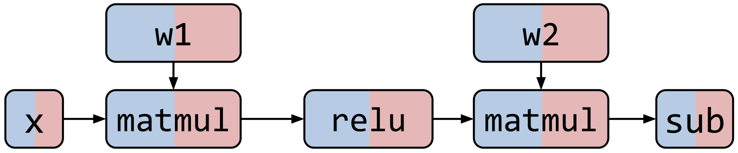
Device 1

Device 2

Strategy 1: Inter-operator Parallelism



Strategy 2: Intra-operator Parallelism

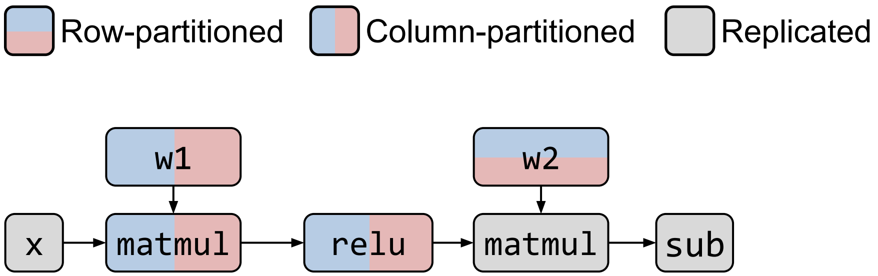


Trade-off

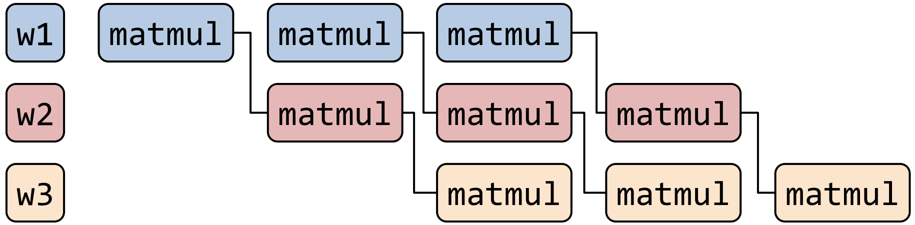
	Inter-operator Parallelism	Intra-operator Parallelism
Communication	Less	More
Device Idle Time	More	Less

Partition Computational Graphs

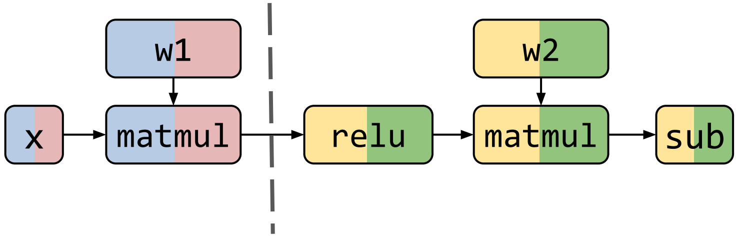
Multiple intra-op strategies for a single node

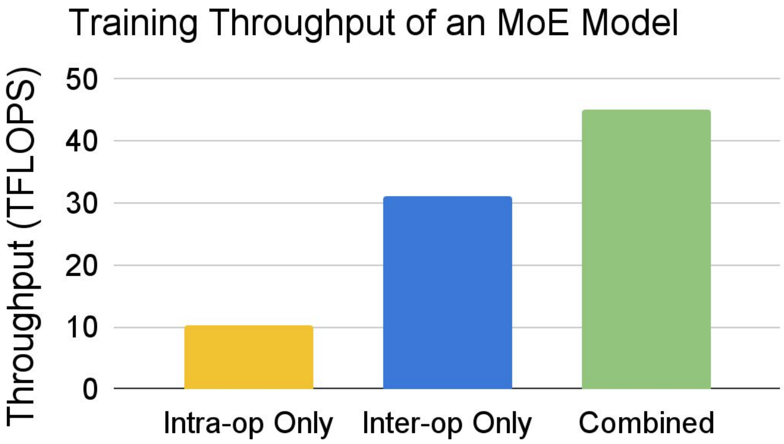


Pipeline the execution for inter-op parallelism

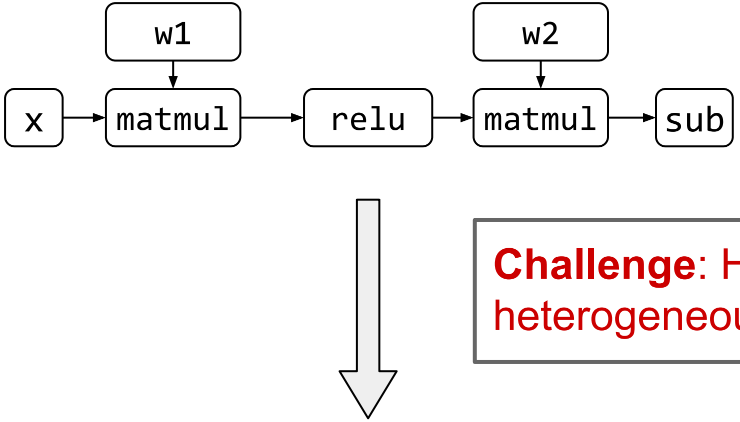


Combine Intra-op and Inter-op

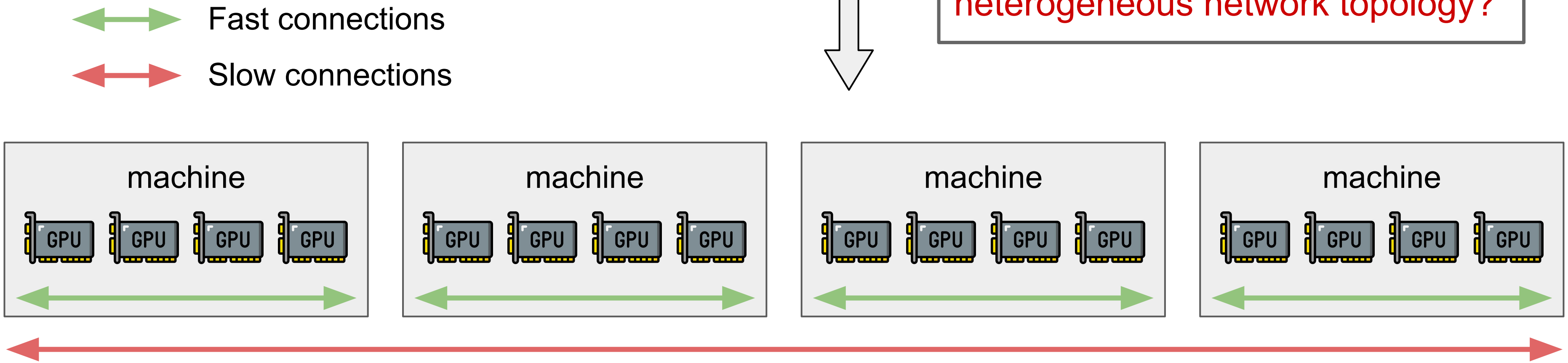




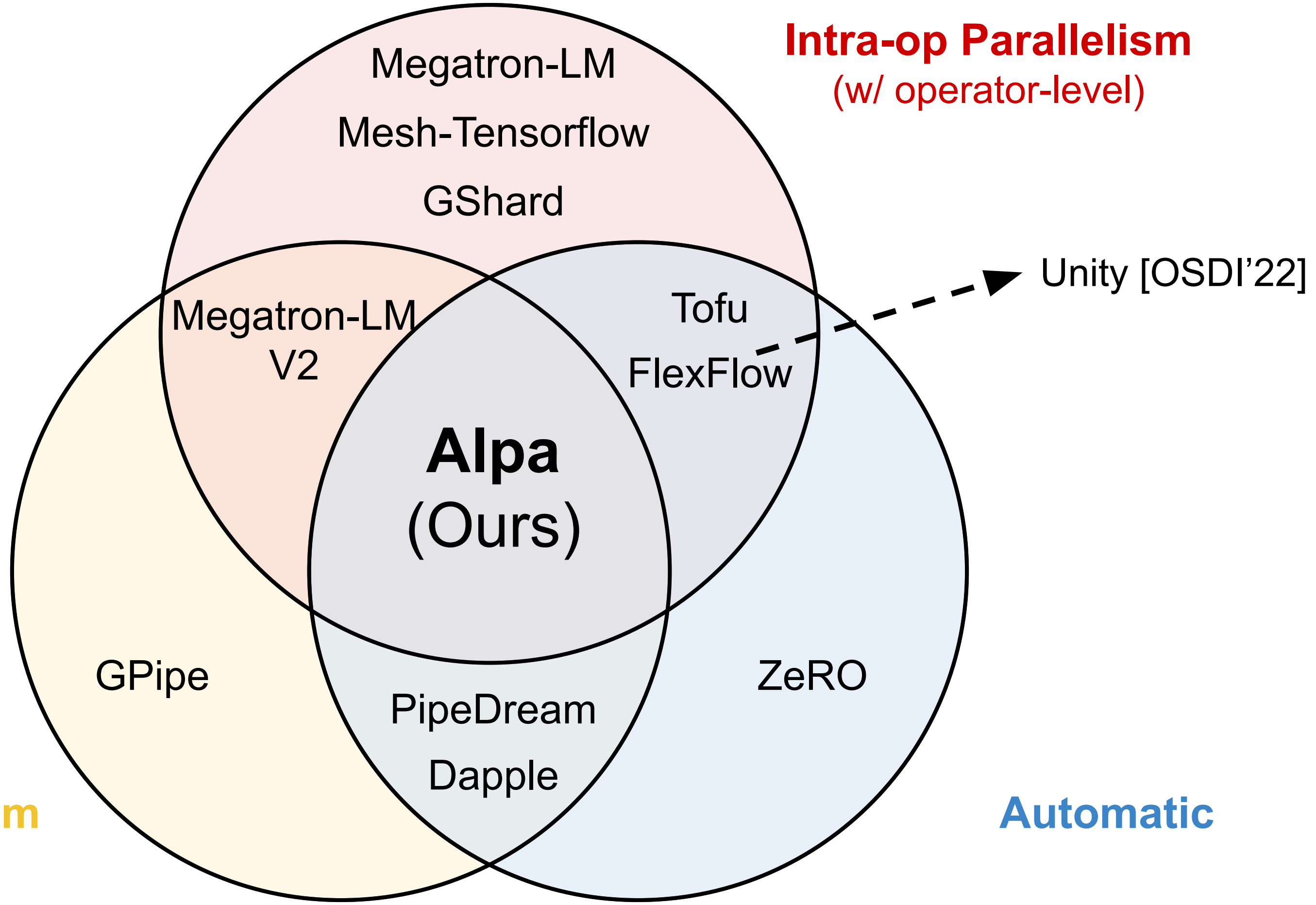
Network Topology



Challenge: How to handle heterogeneous network topology?



Prior Works



Intra-op Parallelism
(w/ operator-level)
Megatron-LM
Mesh-Tensorflow
GShard
Unity [OSDI'22]
Megatron-LM
V2
Tofu
FlexFlow
Alpa
(Ours)
GPipe
ZeRO
PipeDream
Dapple
Inter-op Parallelism
(w/ pipeline)
Automatic

Alpa Compiler

Alpa

A unified **compiler** that **automatically** finds and executes the best **Inter-op** and **Intra-op** parallelism for **large** deep learning models.

Alpa User API

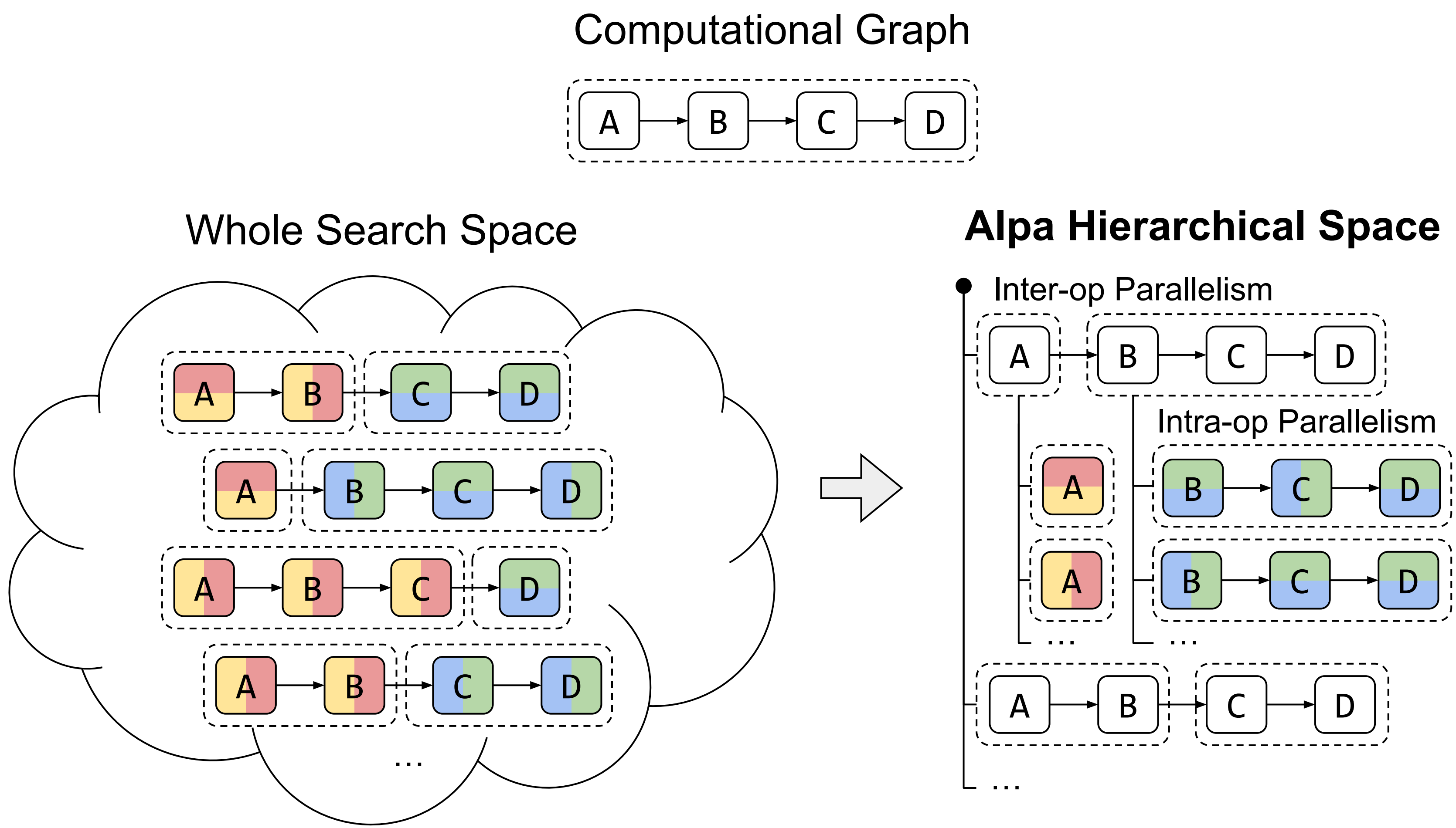
@alpa.parallelize

Distribute the training function with a simple decorator

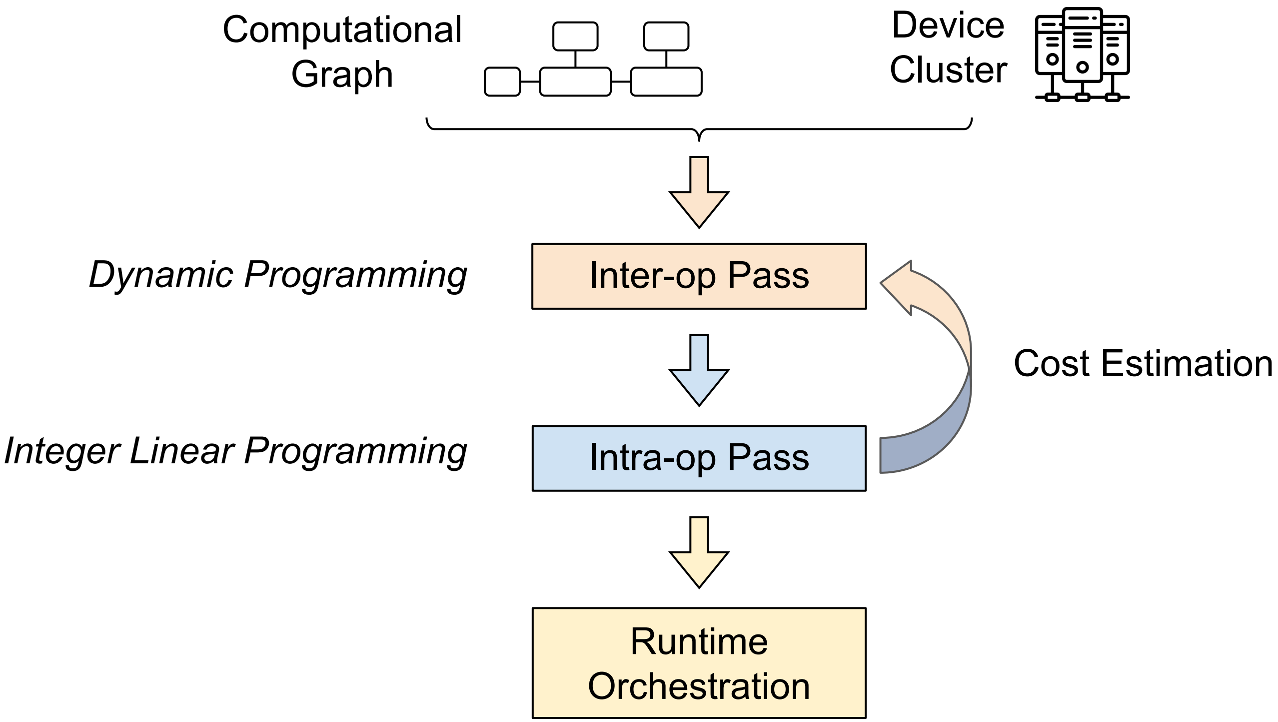
```python
@alpa.parallelize
def train_step(model_state, batch):
    def loss_func(params):
        out = model_state.forward(params, batch["x"])
        return np.mean((out - batch["y"]) ** 2)

    grads = grad(loss_func)(state.params)
    new_model_state = model_state.apply_gradient(grads)
    return new_model_state

# A typical JAX training loop
model_state = create_train_state()
for batch in data_loader:
    model_state = train_step(model_state, batch)
```

Alpa’s Main Contributions

- Two-level hierarchical space of parallelism techniques.
- Effective optimization algorithms at each level.
- Efficient compiler and runtime system implementation.

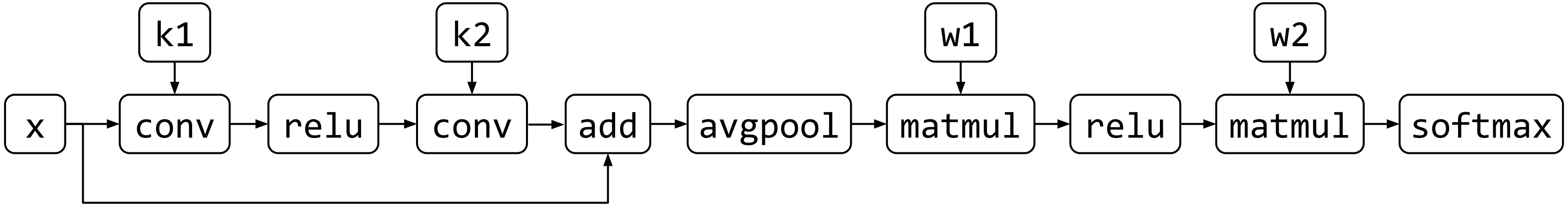

Computational Graph
A
B
C
D
Whole Search Space
...
Alpa Hierarchical Space
Inter-op Parallelism
Intra-op Parallelism
...

Alpa Compiler: Hierarchical Optimization

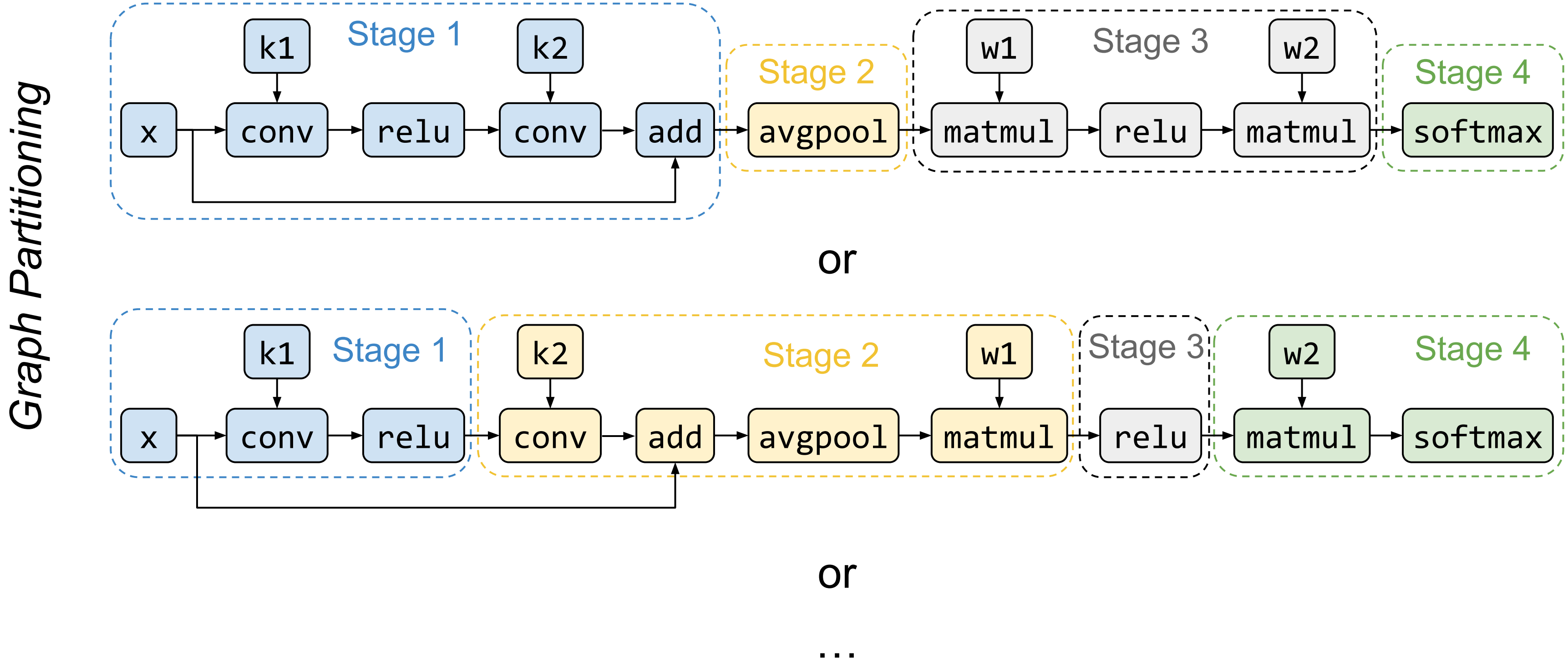


Inter-op Pass

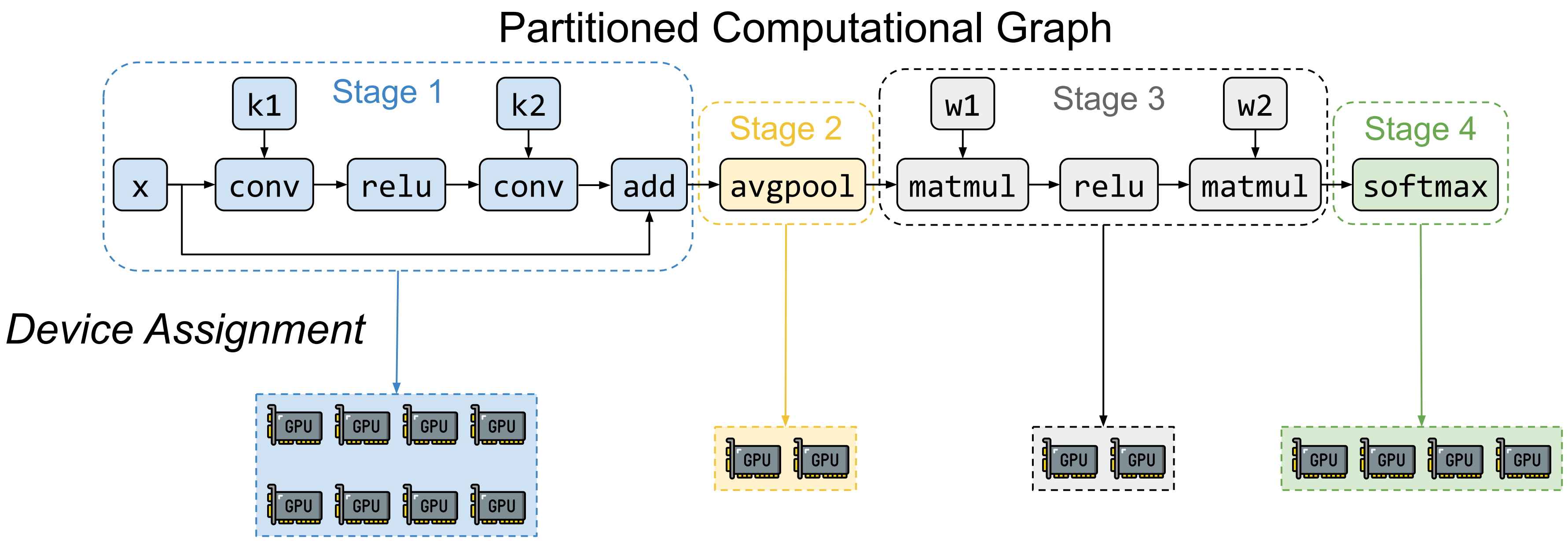
Computational Graph



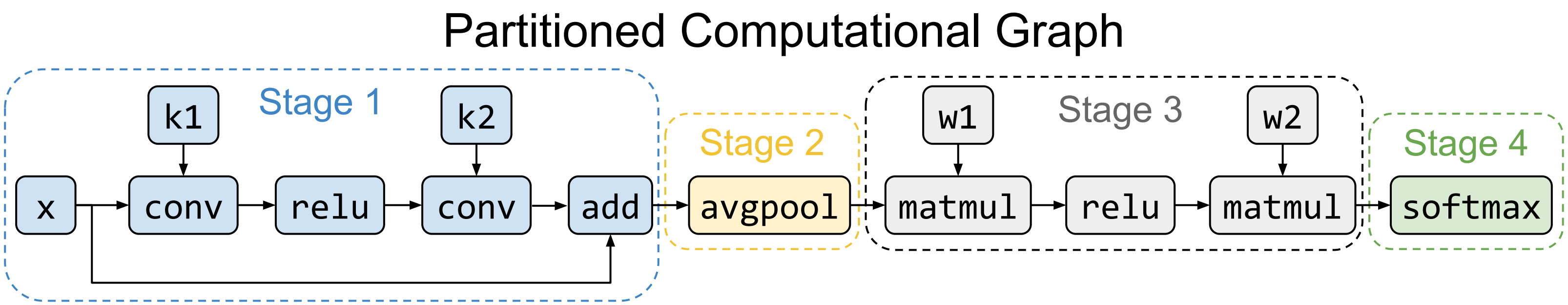
Inter-op Pass

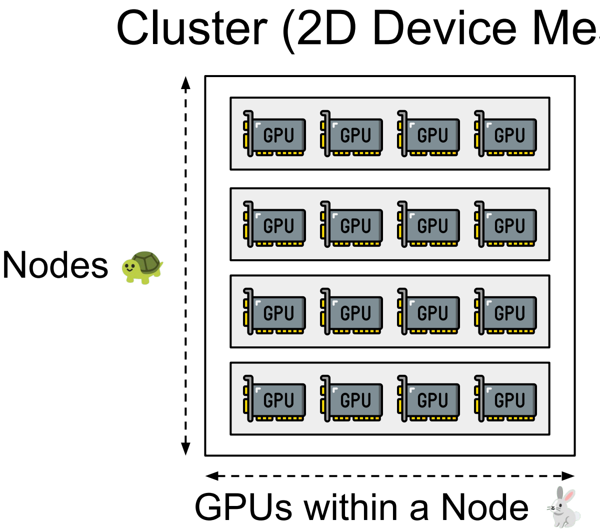


Inter-op Pass

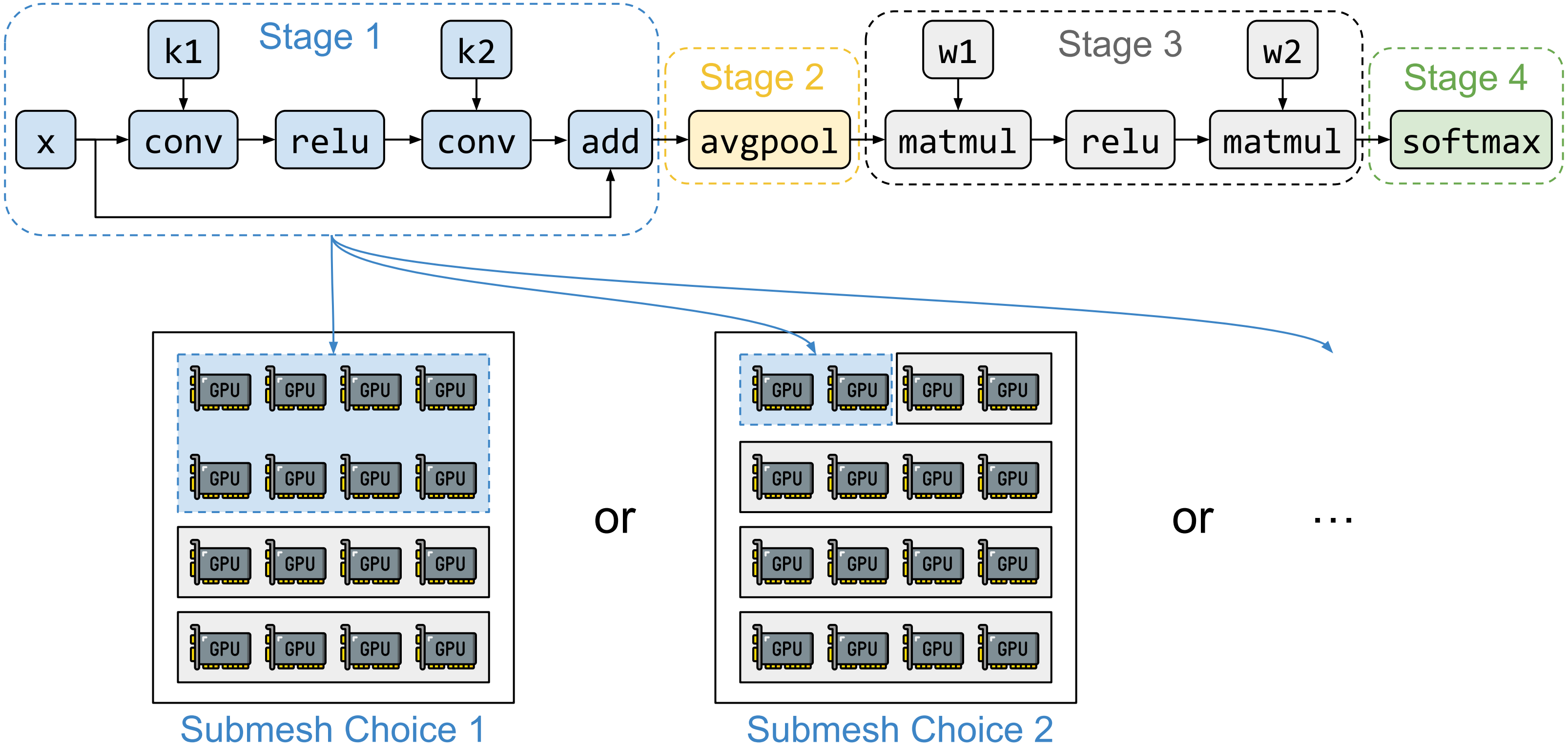


Inter-op Pass

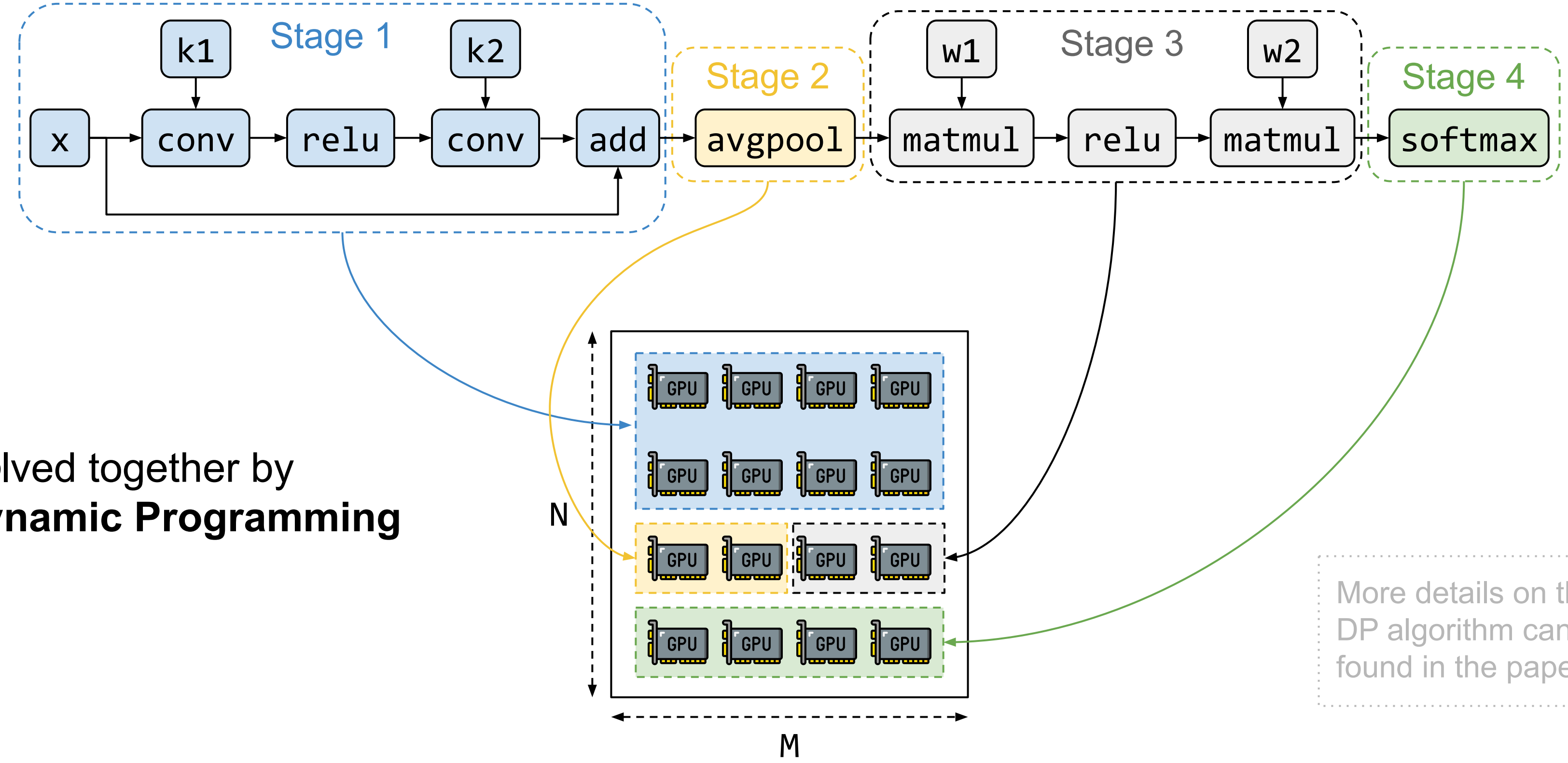




Inter-op Pass



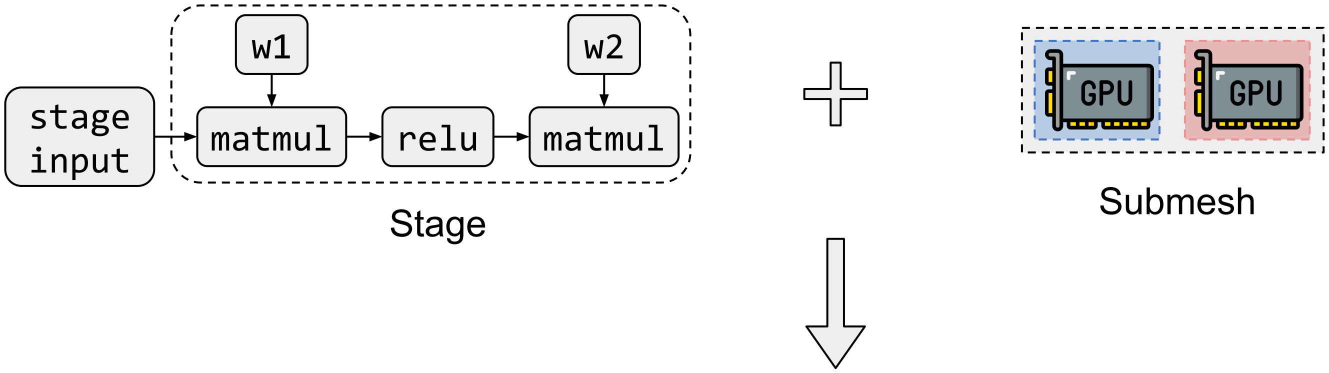
Inter-op Pass



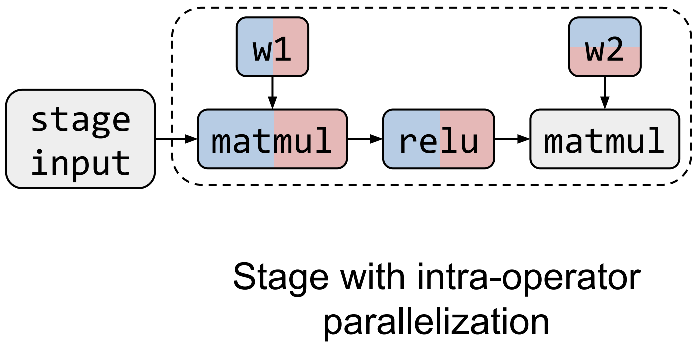
*S*olved together by
Dynamic Programming

More details on the DP algorithm can be found in the paper.

Intra-op Pass



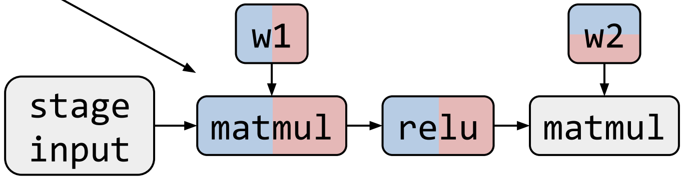
Solved by
Integer Linear Programming



Intra-op Pass

Integer Linear Programming Formulation

Decision vector
Parallel strategies of each operator



Minimize Computation cost + Communication cost

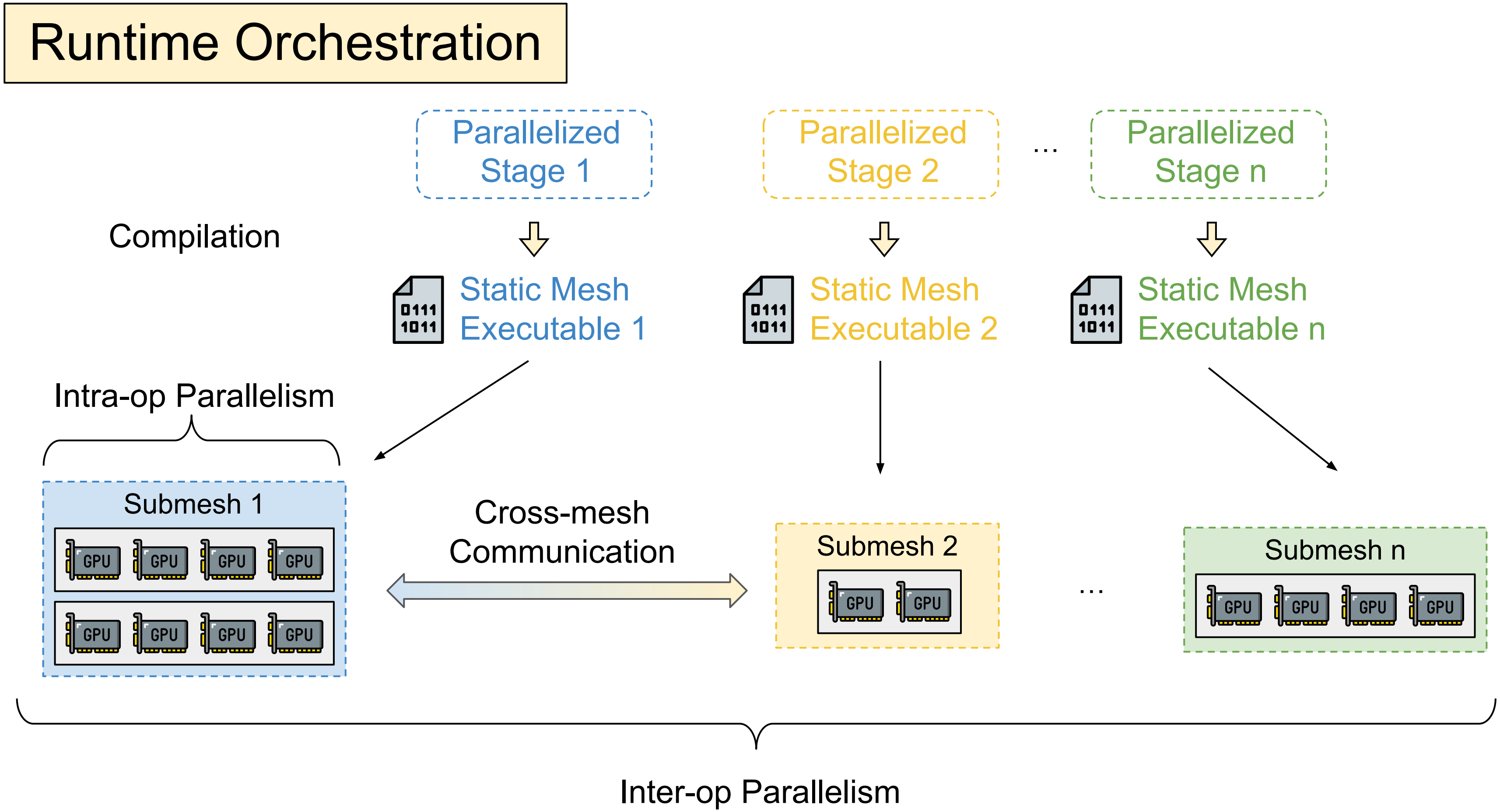
More details on the ILP algorithm can be found in the paper.

Compilation Time Optimization

Communication-aware operator clustering in ILP & DP	Early stopping in DP	Distributed Compilation

Alpa Compilation Time: < 40 min for the largest experiment.

- Can be further reduced by at least 50% with search space pruning.

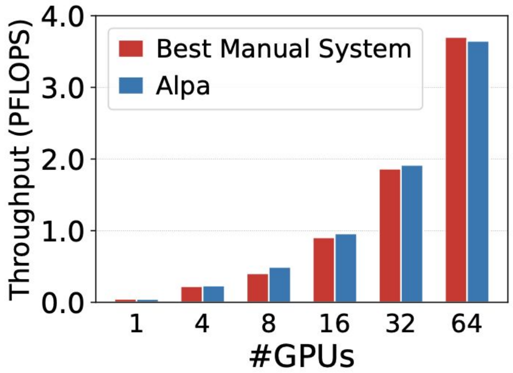

Runtime Orchestration
Parallelized Stage 1
Parallelized Stage 2
...
Parallelized Stage n
Compilation
Static Mesh Executable 1
Static Mesh Executable 2
Static Mesh Executable n
0111
1011
Intra-op Parallelism
Submesh 1
GPU
Cross-mesh Communication
Submesh 2
...
Submesh n
Inter-op Parallelism

Evaluation

Alpa

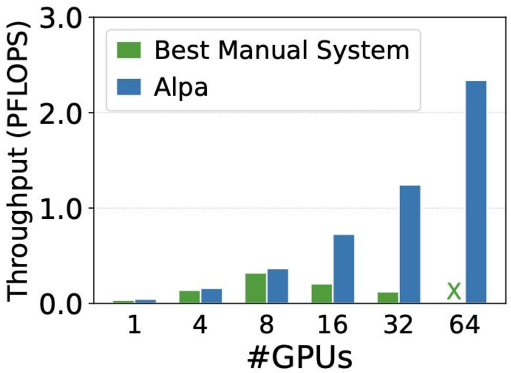
Evaluation: Comparing with Previous Works

GPT (up to 39B)



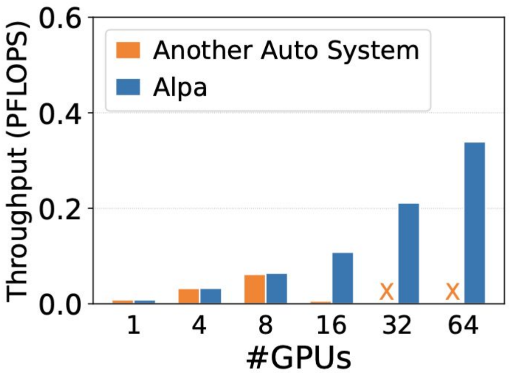
Match specialized manual systems.

GShard MoE (up to 70B)



Outperform the manual baseline by up to 8x.

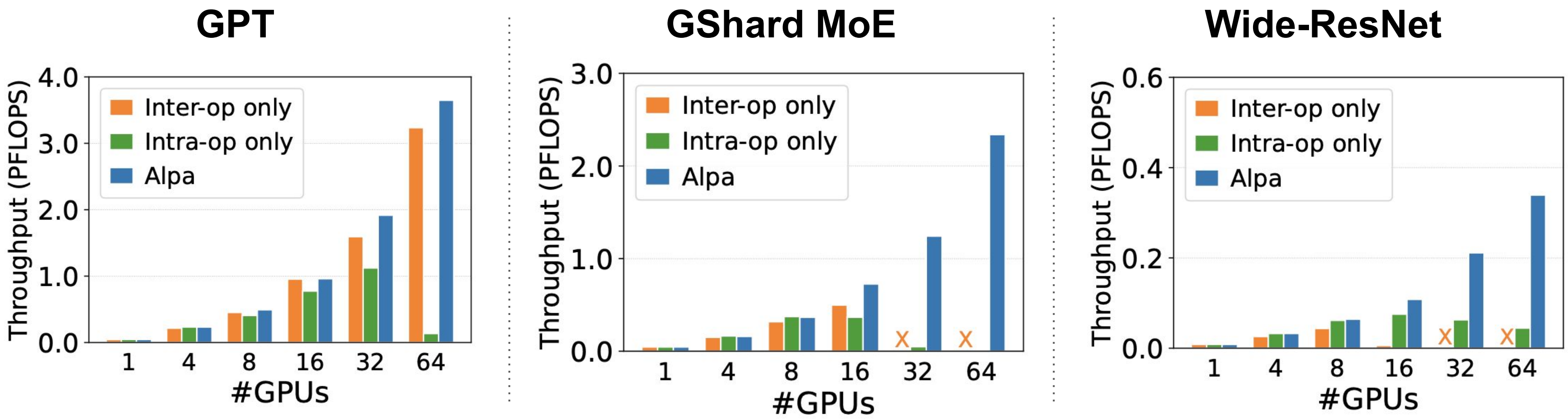
Wide-ResNet (up to 13B)



Generalize to models without manual plans.

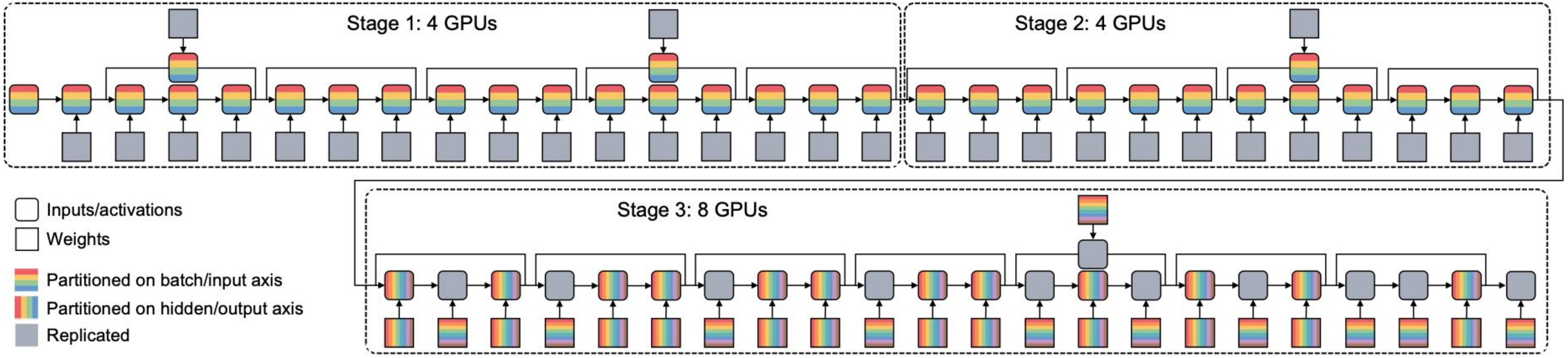
Weak scaling results where the model size grow with #GPUs.
Evaluated on 8 AWS EC2 p3.16xlarge nodes with 8 16GB V100s each (64 GPUs in total).

Evaluation: Ablation Study with Inter-op and Intra-op Only



Combining inter- and intra-operator parallelism scales to more devices.

Case Study: Wide-ResNet Partition on 16 GPUs.



`@alpa.parallelize:` automatic model-parallel training

- Hierarchical view: inter-op and intra-op
- Match or outperform specialized systems
- Generalizes to new models

Alpa

alpa.ai

Star 390