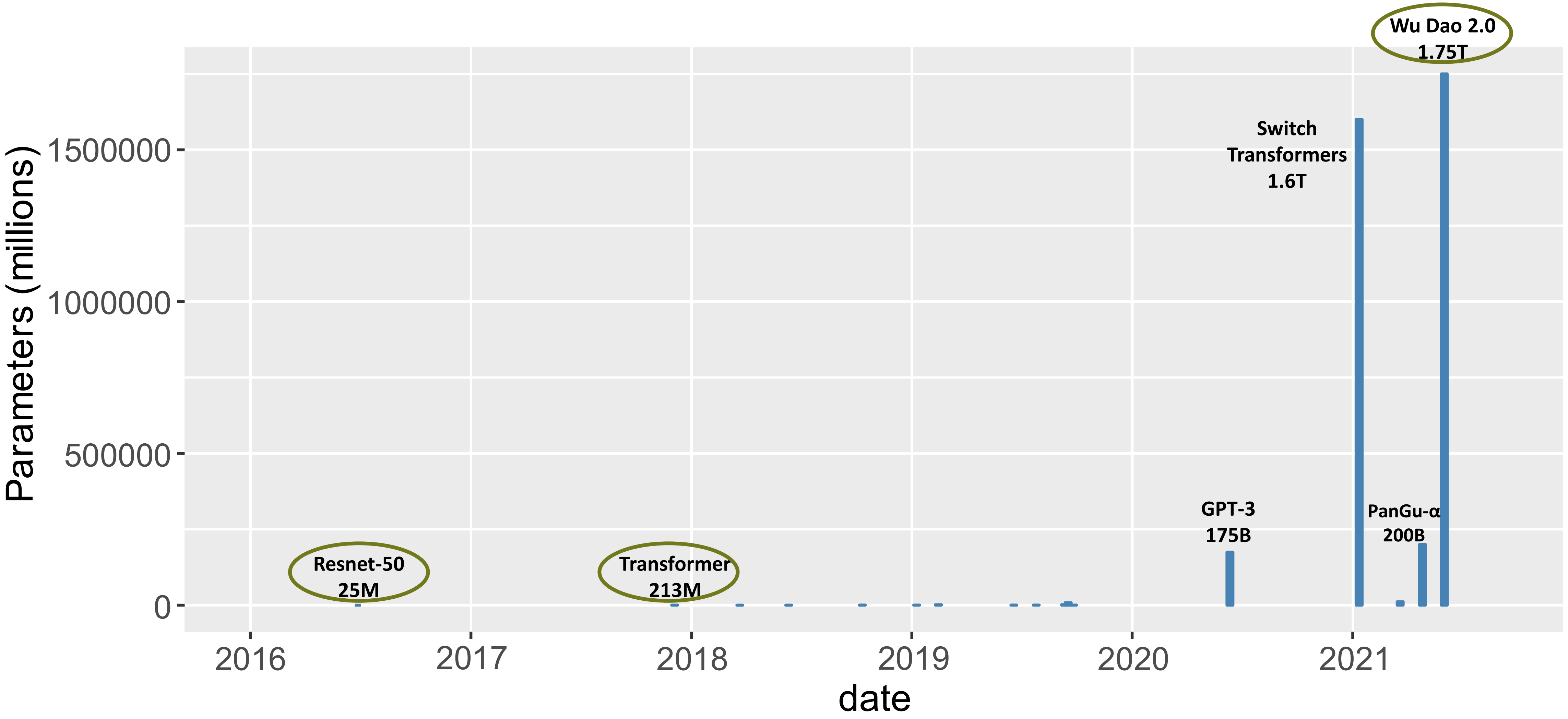
Chimera: Efficiently Training Large-Scale Neural Networks with Bidirectional Pipelines

Shigang Li, Torsten Hoefler
SPCL Lab, ETH Zurich

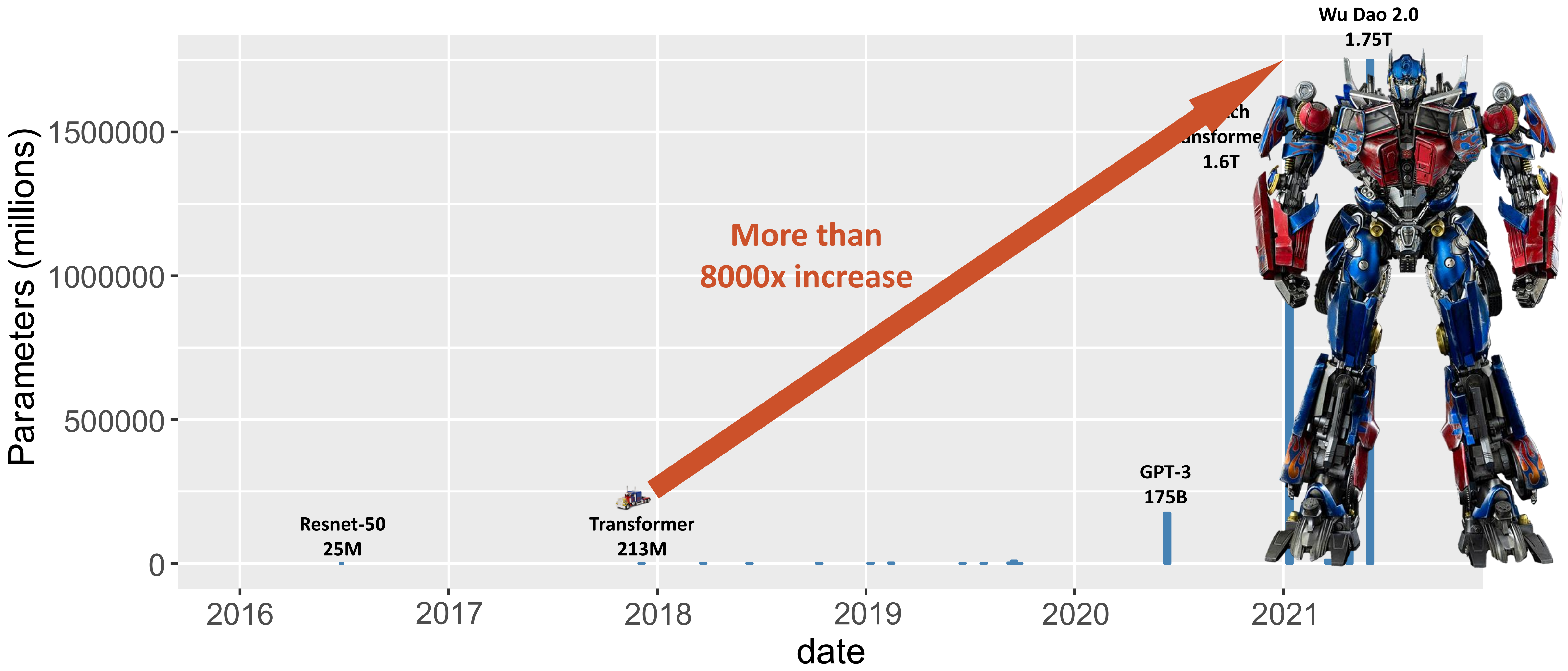
Contact email: *shigangli.cs@gmail.com*

Model size growing rapidly



Contact email: *shigangli.cs@gmail.com*

Model size growing rapidly

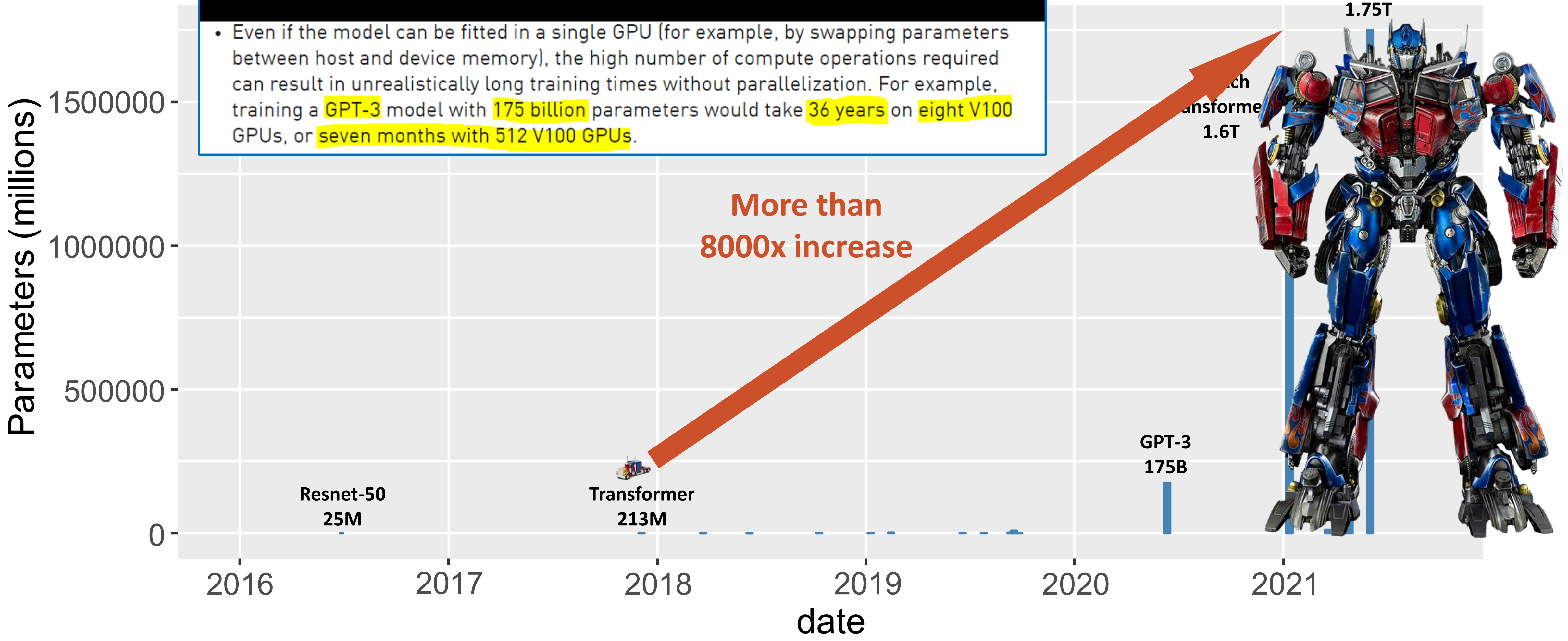


Contact email: *shigangli.cs@gmail.com*

Model size growing rapidly

NVIDIA DEVELOPER HOME BLOG FORUMS DOCS DOWNLOADS

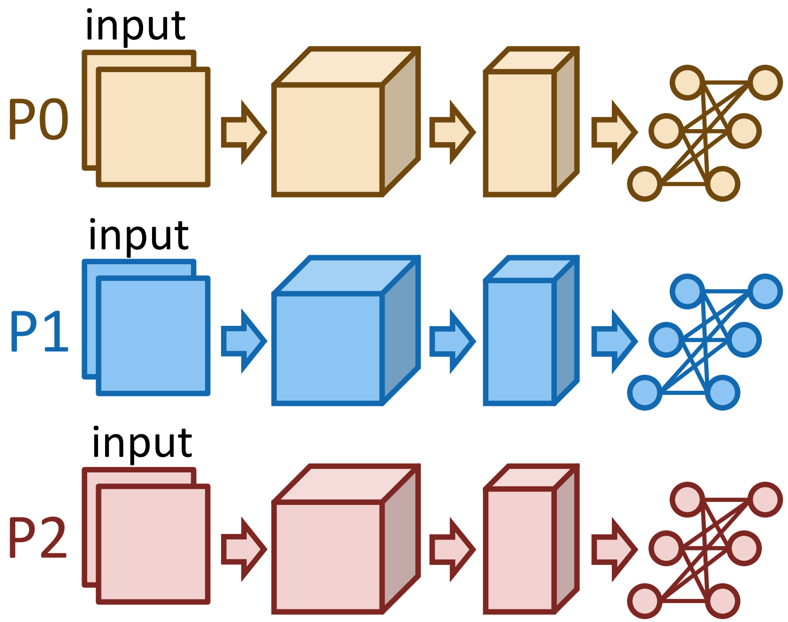
- Even if the model can be fitted in a single GPU (for example, by swapping parameters between host and device memory), the high number of compute operations required can result in unrealistically long training times without parallelization. For example, training a GPT-3 model with 175 billion parameters would take 36 years on eight V100 GPUs, or seven months with 512 V100 GPUs.



Contact email: *shigangli.cs@gmail.com*

Parallel and distributed training

Data parallelism



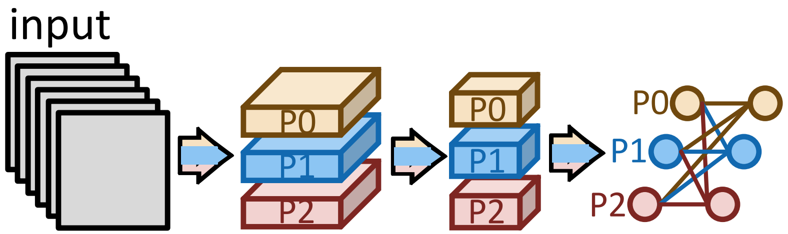
Pros:

a. Easy to realize

Cons:

a. Not work for large models
b. High allreduce overhead

Operator parallelism



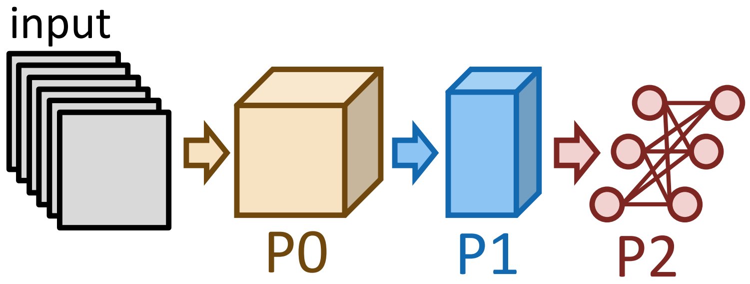
Pros:

a. Make large model training feasible

Cons:

b. Communication for each operator (or each layer)

Pipeline parallelism



Pros:

a. Make large model training feasible
b. No collective, only P2P between stages

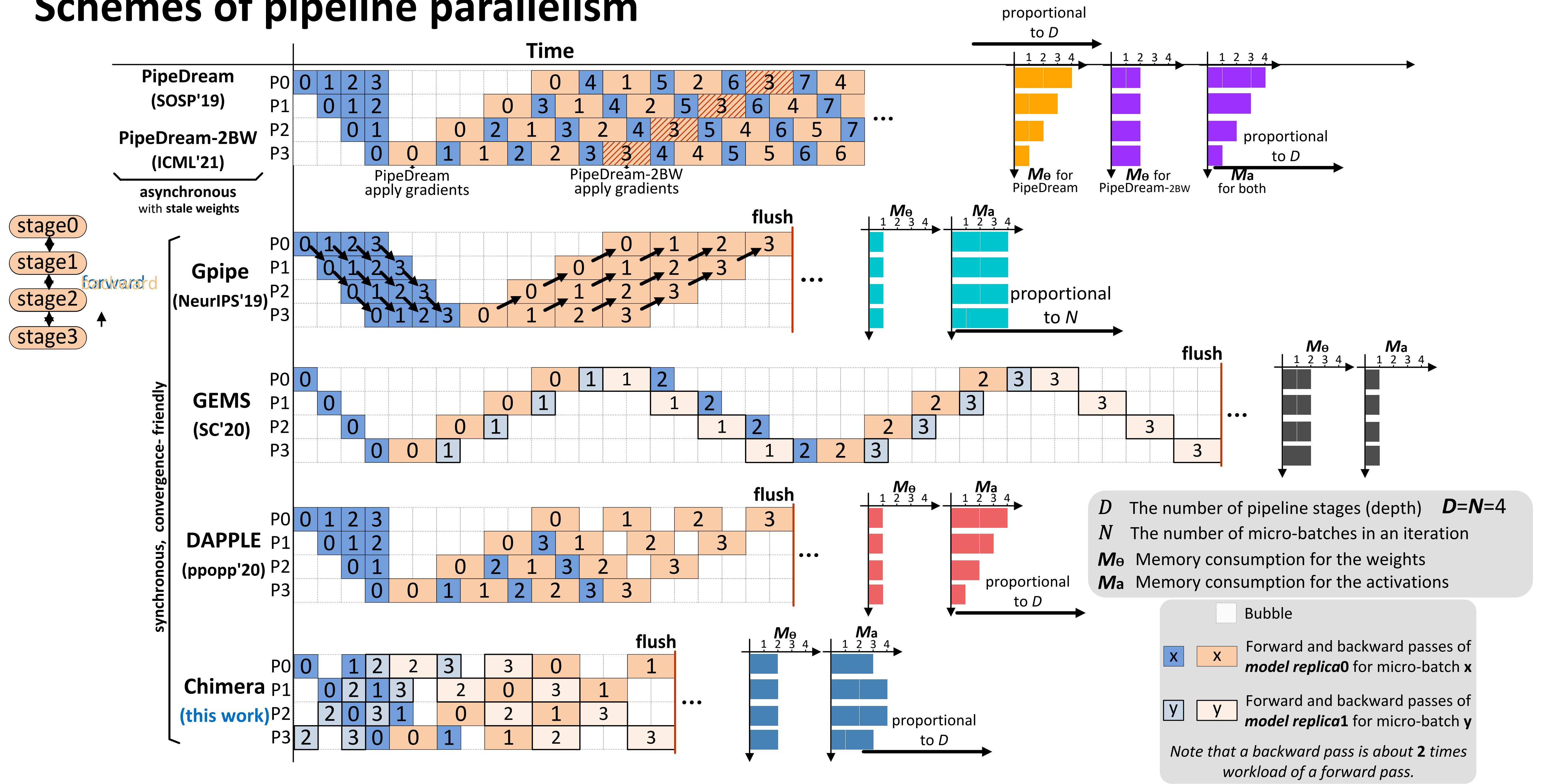
Cons:

a. Bubbles in pipeline
b. Removing bubbles leads to stale weights

Chimera (this work) aims to solve.

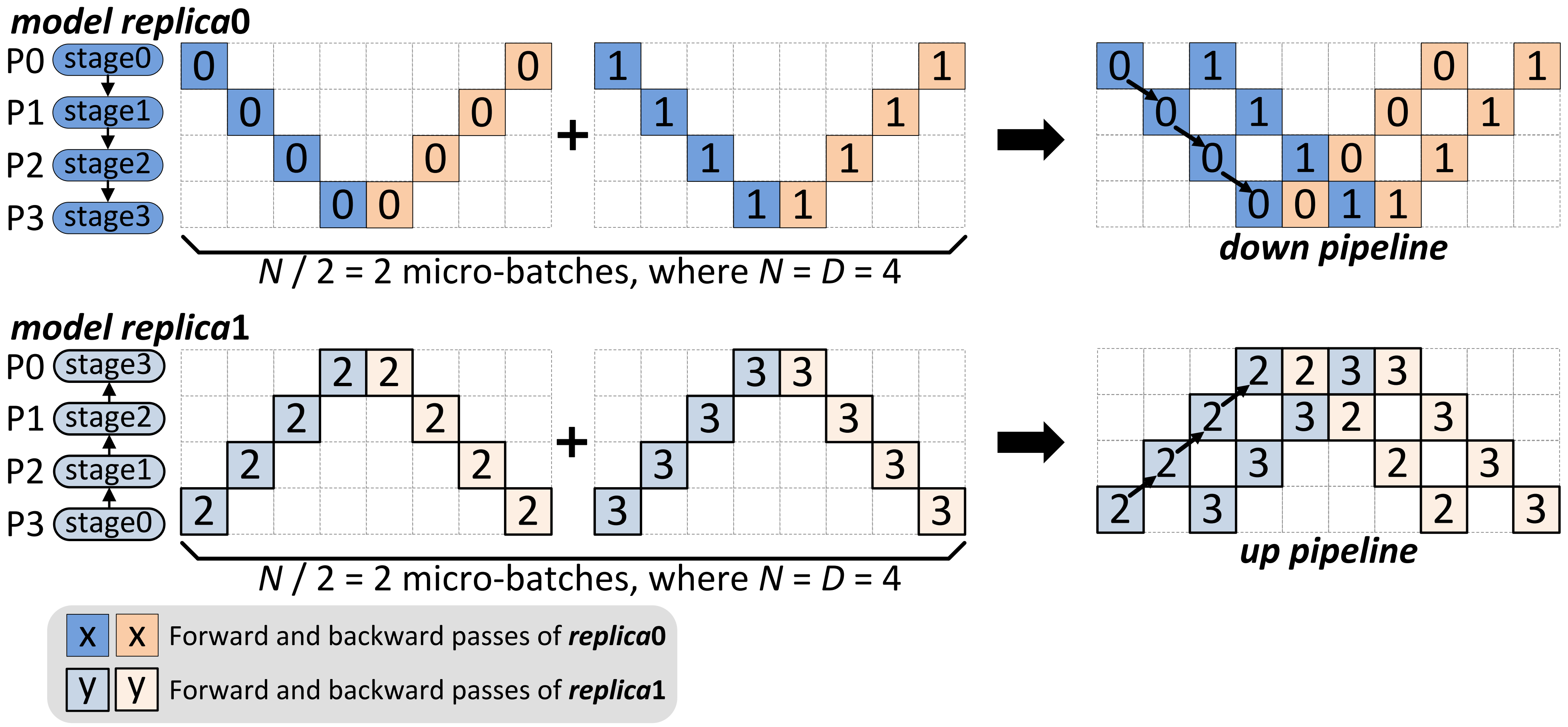
Contact email: *shigangli.cs@gmail.com*

Schemes of pipeline parallelism



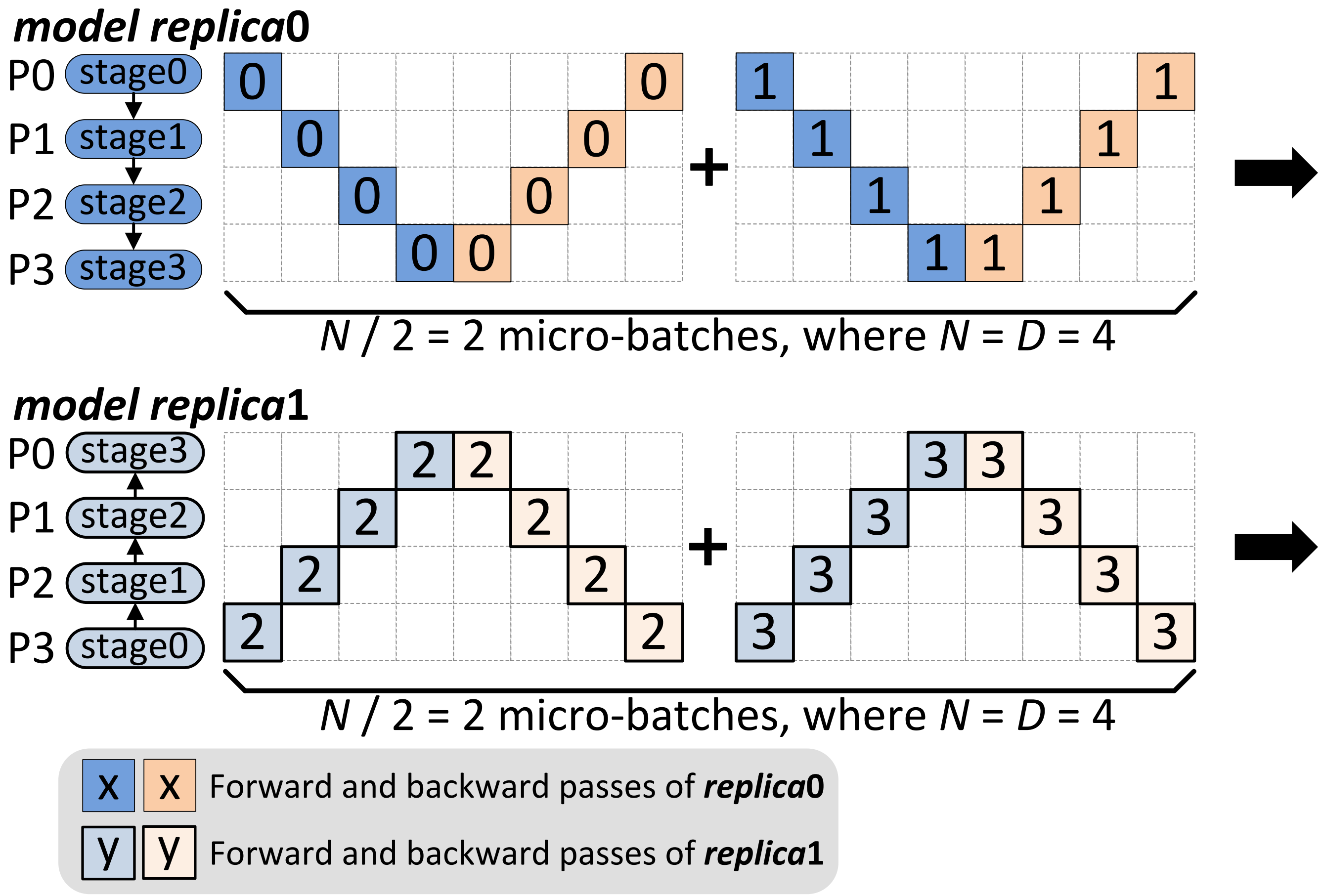
Contact email: *shigangli.cs@gmail.com*

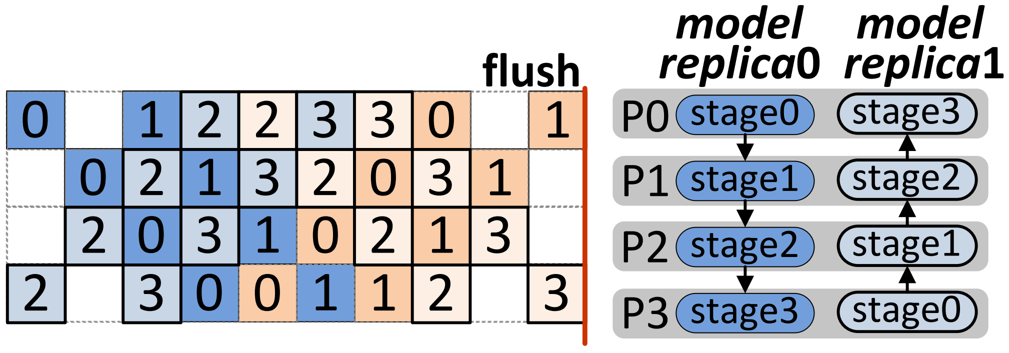
Bidirectional Pipelines



Contact email: *shigangli.cs@gmail.com*

Bidirectional Pipelines



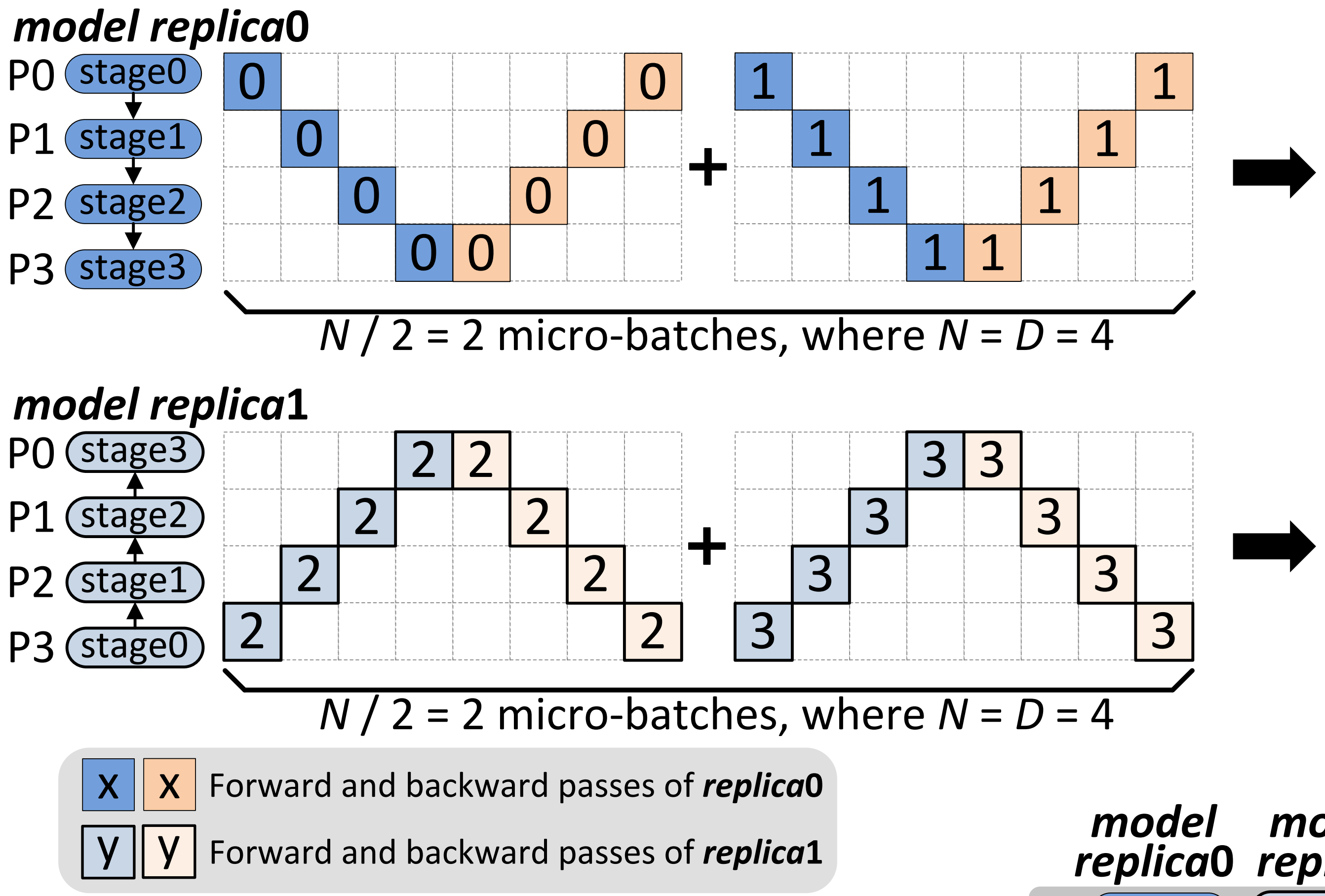


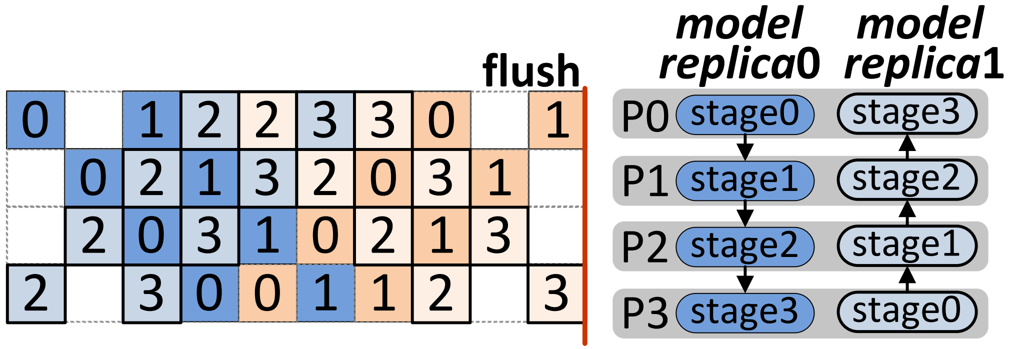
Chimera

from Blizzard War3

Contact email: *shigangli.cs@gmail.com*

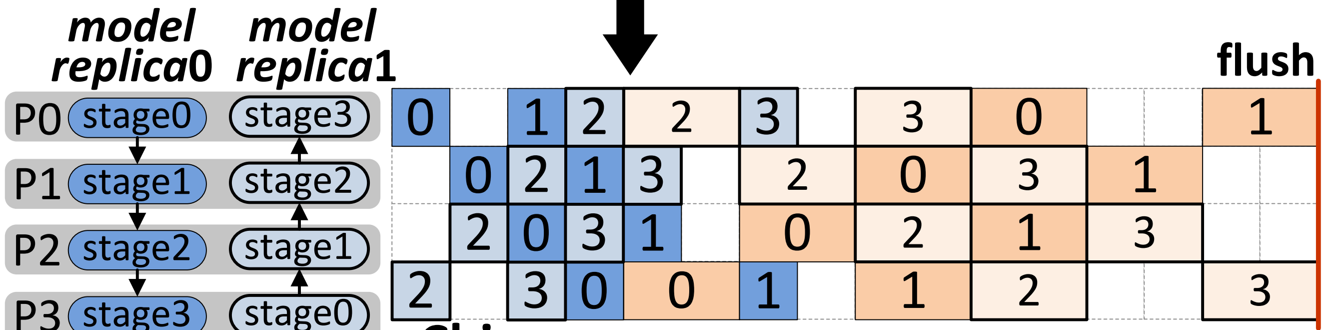
Bidirectional Pipelines





Chimera

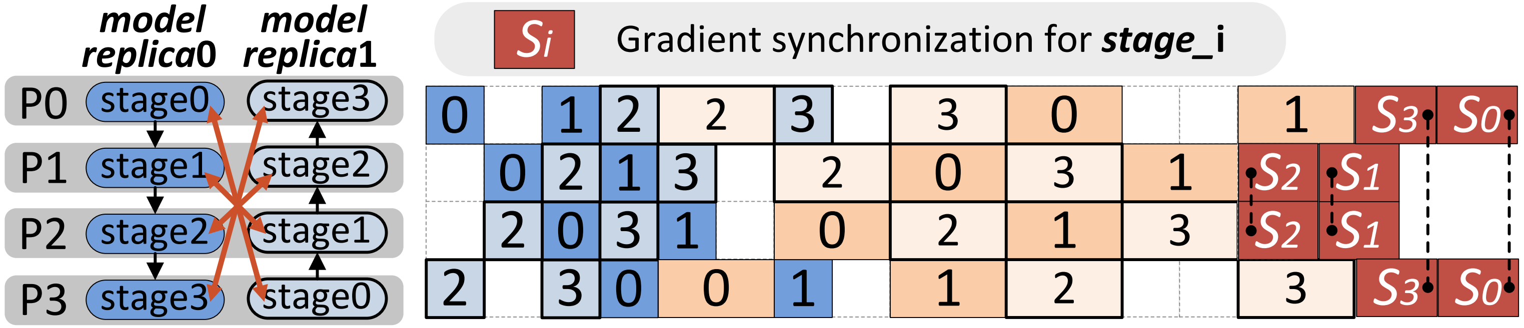
from Blizzard War3



Chimera (backward is **2×** workload of forward)

Contact email: *shigangli.cs@gmail.com*

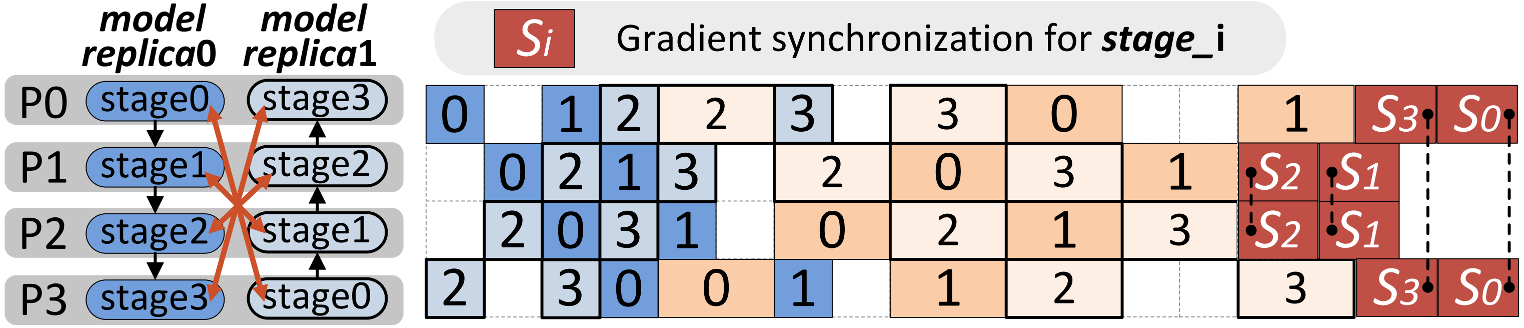
Gradient synchronization between model replicas



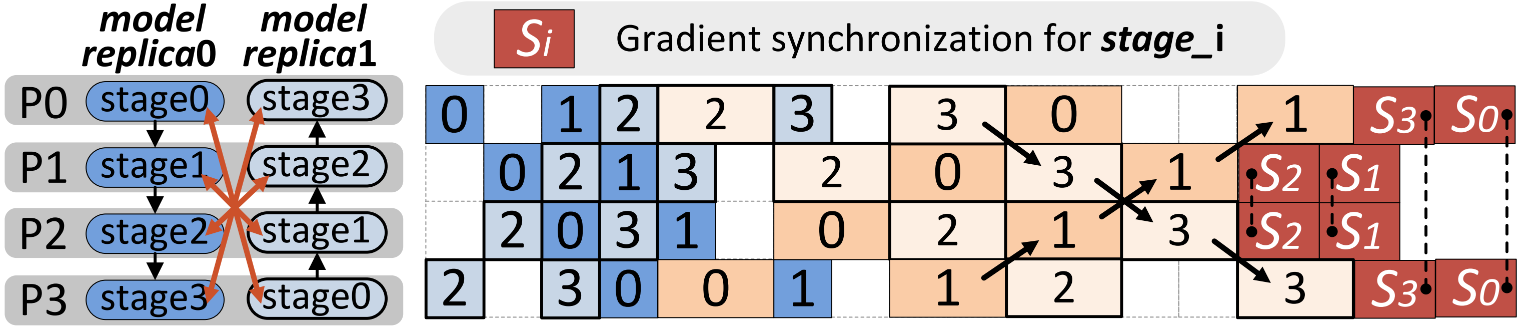
(a) Gradient synchronization after all local computation is finished

Contact email: *shigangli.cs@gmail.com*

Gradient synchronization between model replicas



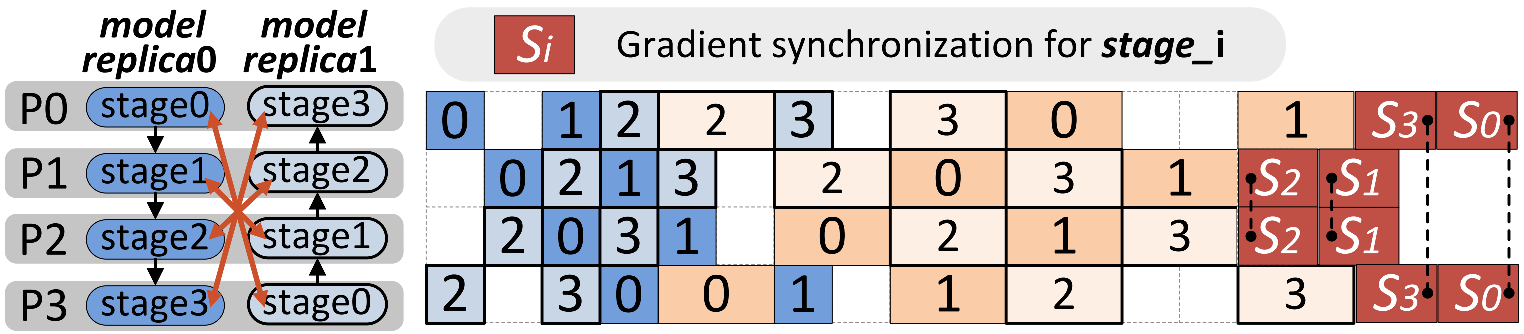
(a) Gradient synchronization after all local computation is finished



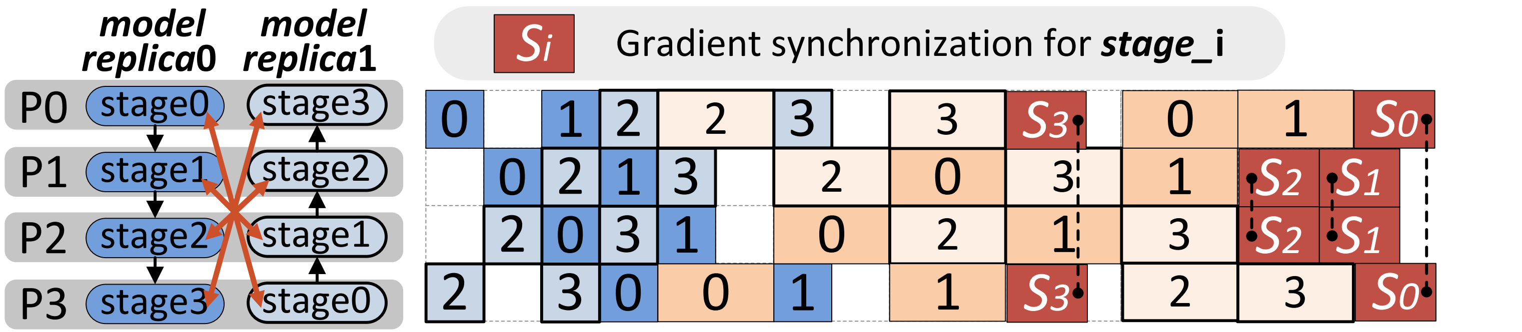
(b) Eager gradient synchronization for deeper overlapping

Contact email: *shigangli.cs@gmail.com*

Gradient synchronization between model replicas



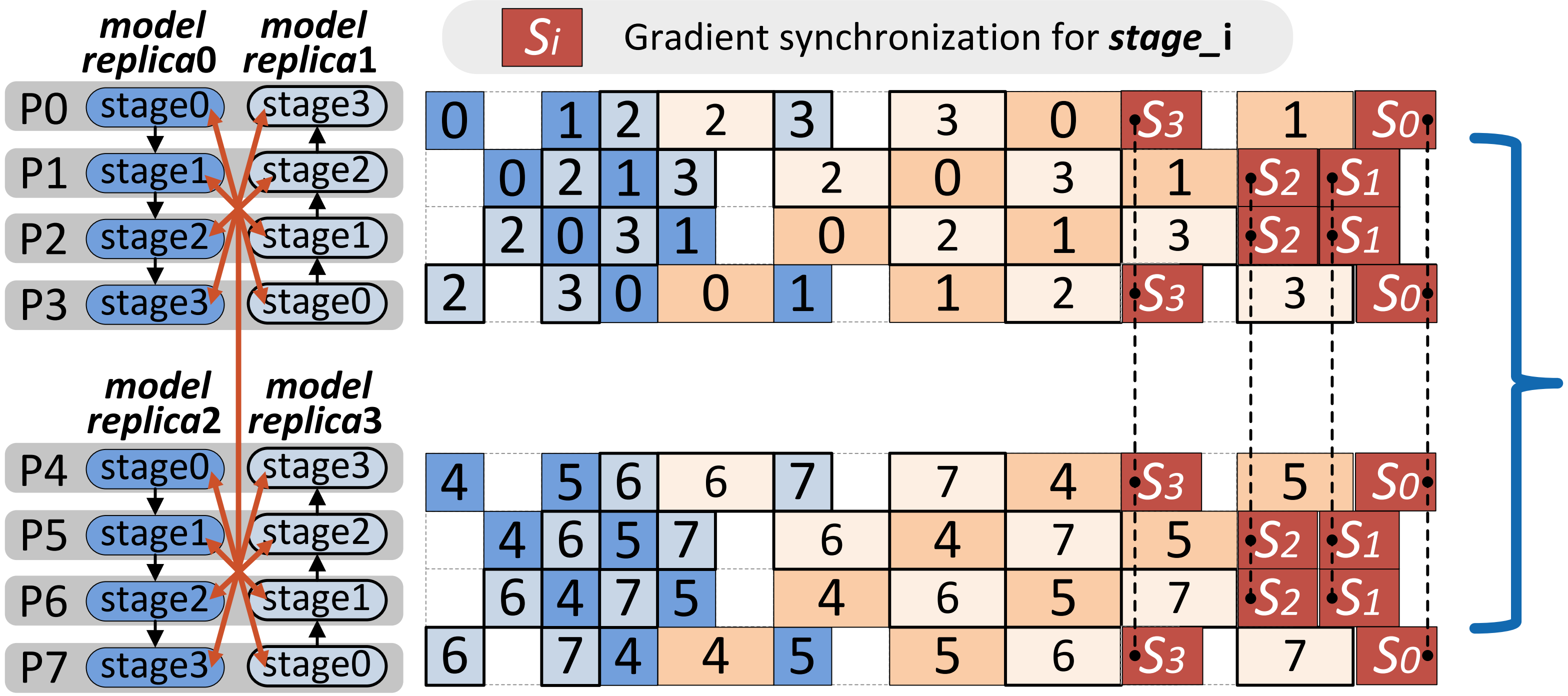
(a) Gradient synchronization after all local computation is finished



(b) Eager gradient synchronization for deeper overlapping

Contact email: *shigangli.cs@gmail.com*

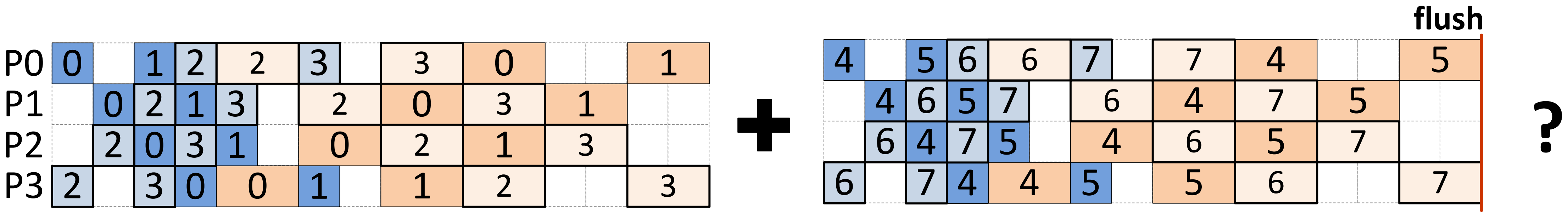
Hybrid of pipeline and data parallelism



2x pipelines,
i.e., ***W*** (width) **= 2**

Contact email: *shigangli.cs@gmail.com*

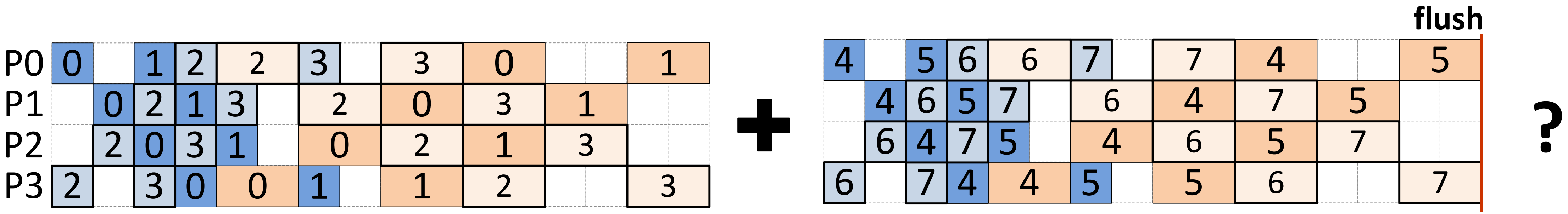
Scale to more micro-batches



N=2***D*** micro-batches, where ***D***=4

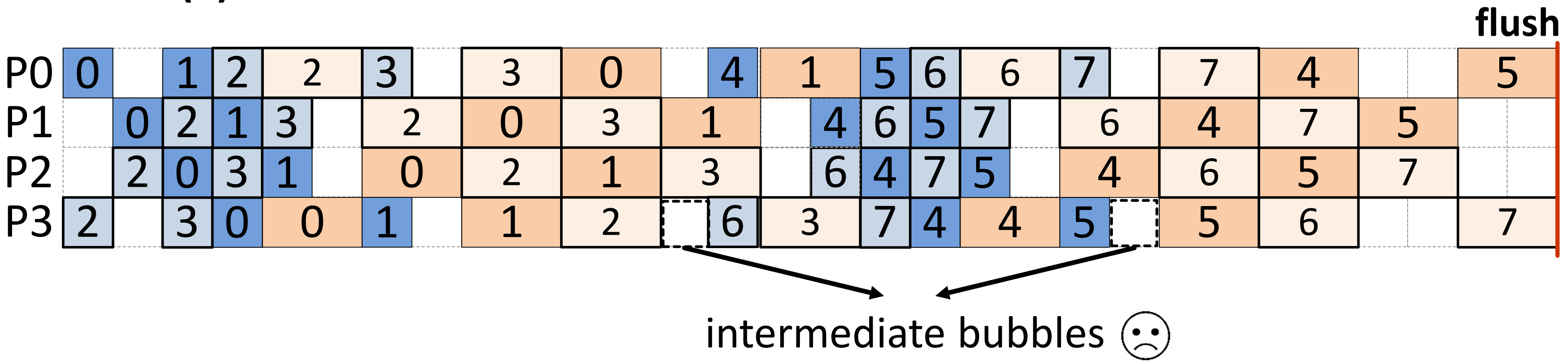
Contact email: *shigangli.cs@gmail.com*

Scale to more micro-batches



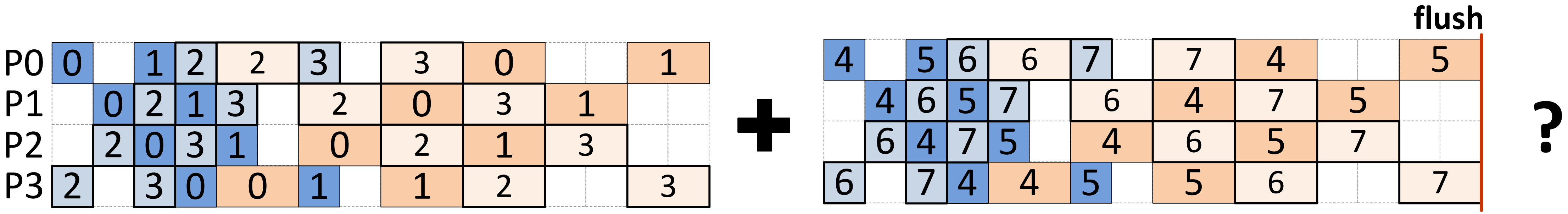
N=2***D*** micro-batches, where ***D***=4

Method (1): Direct concatenation



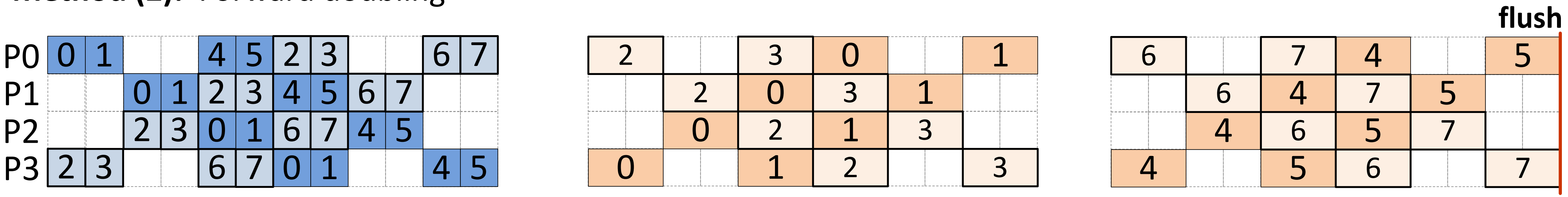
Contact email: *shigangli.cs@gmail.com*

Scale to more micro-batches



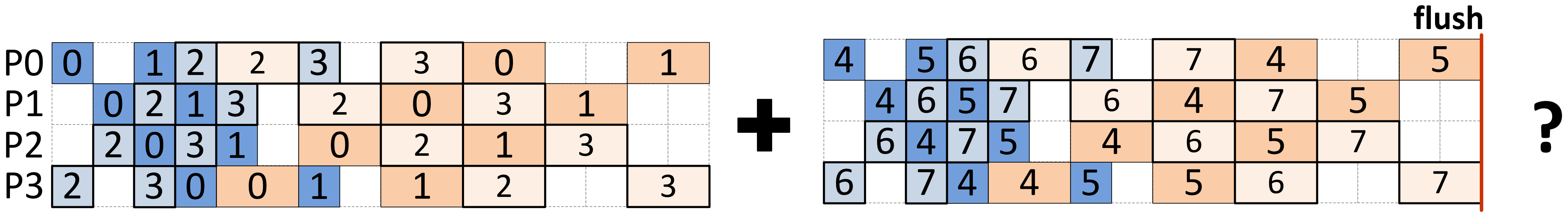
N=2***D*** micro-batches, where ***D***=4

Method (2): Forward doubling



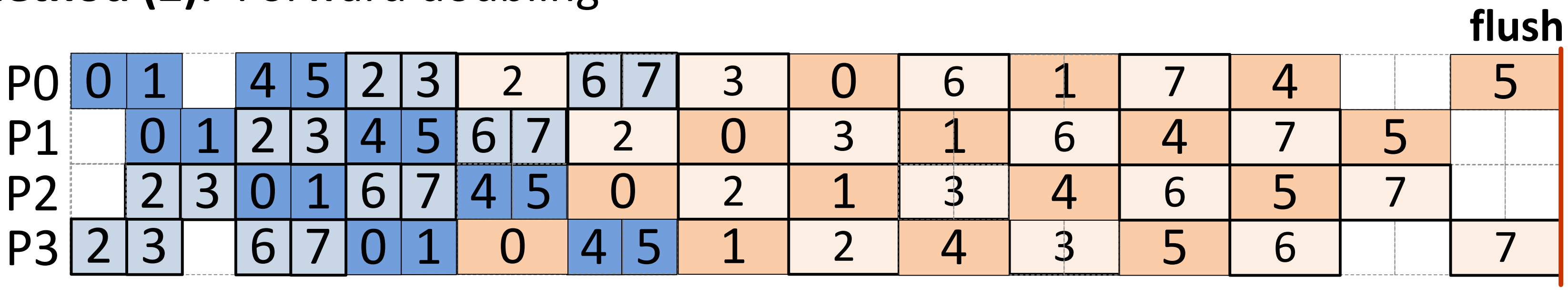
Contact email: *shigangli.cs@gmail.com*

Scale to more micro-batches



N=2***D*** micro-batches, where ***D***=4

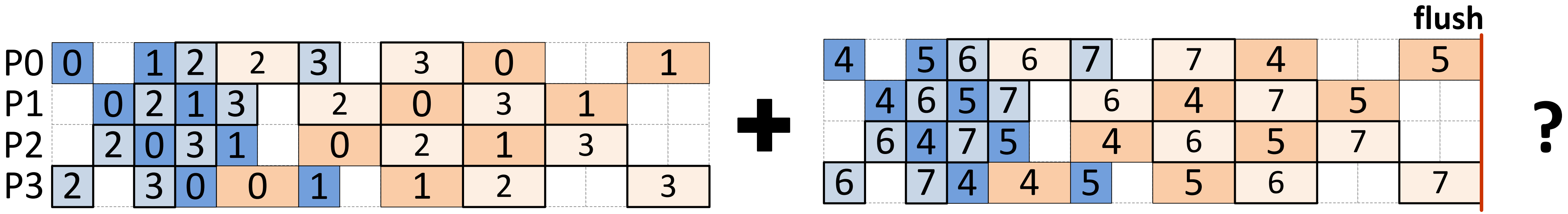
Method (2): Forward doubling



- **eliminate intermediate bubbles**
- **2x activation memory**

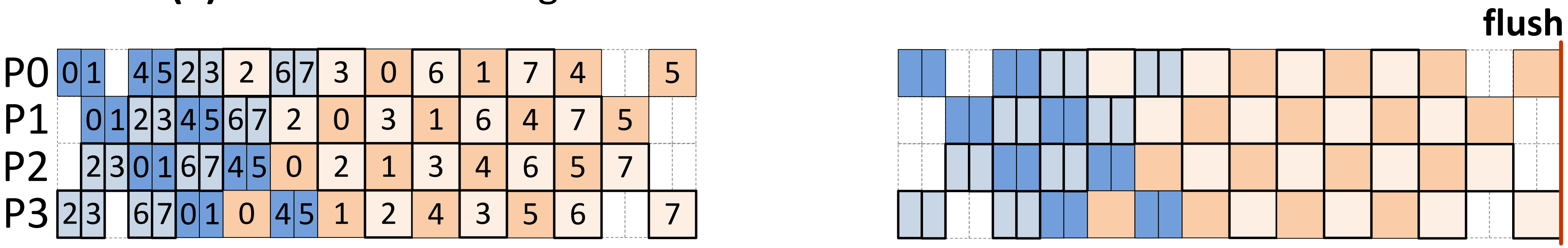
Contact email: *shigangli.cs@gmail.com*

Scale to more micro-batches



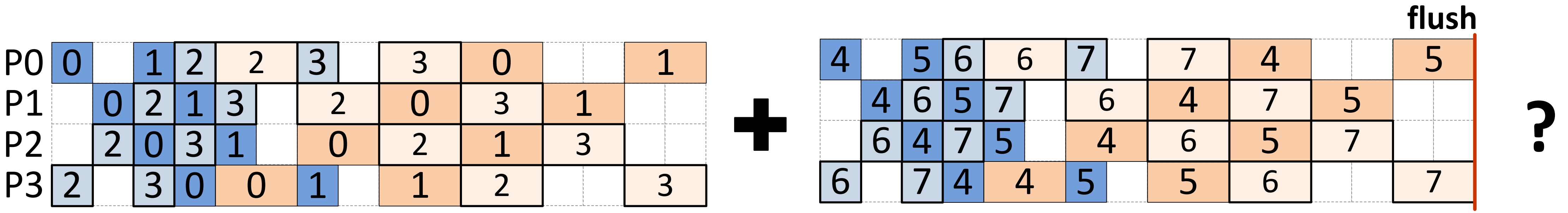
N=2***D*** micro-batches, where ***D***=4

Method (3): Backward halving



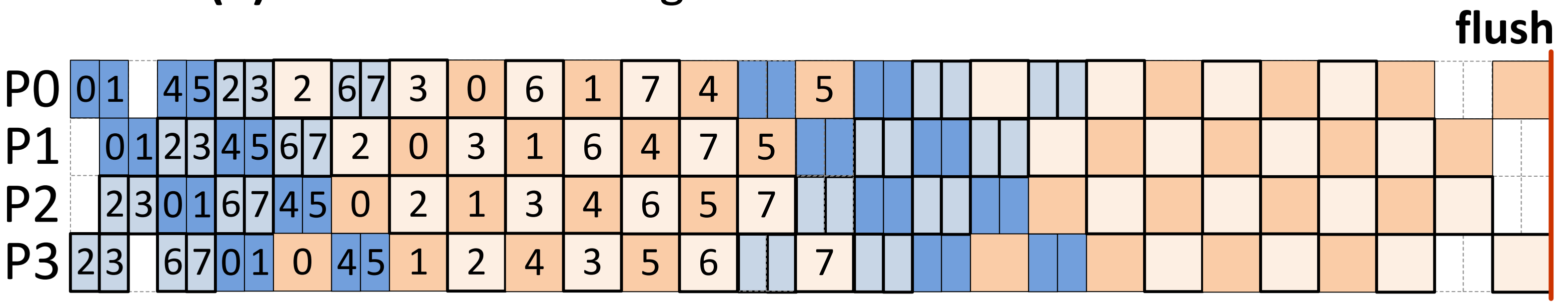
Contact email: *shigangli.cs@gmail.com*

Scale to more micro-batches



N=2***D*** micro-batches, where ***D***=4

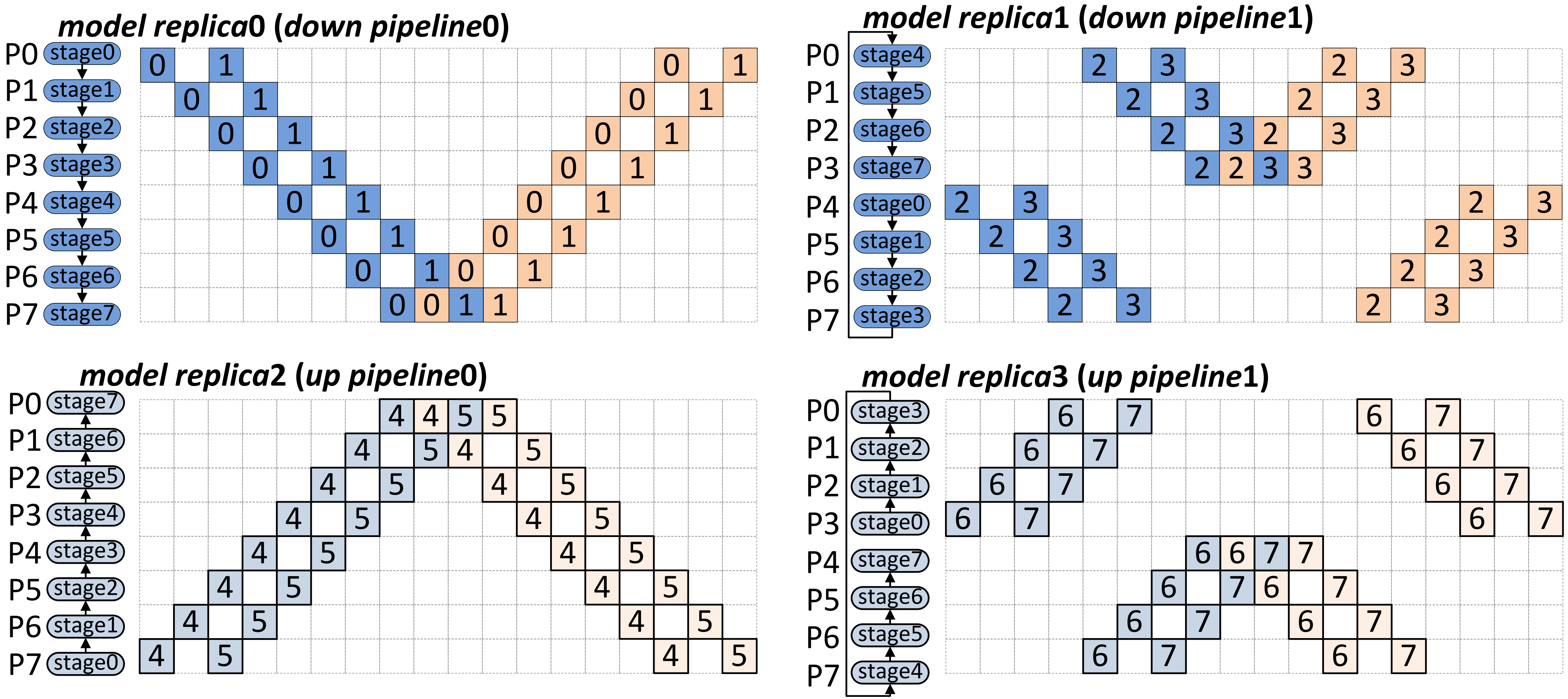
Method (3): Backward halving



- **eliminate intermediate bubbles**
- **halving micro-batch size**

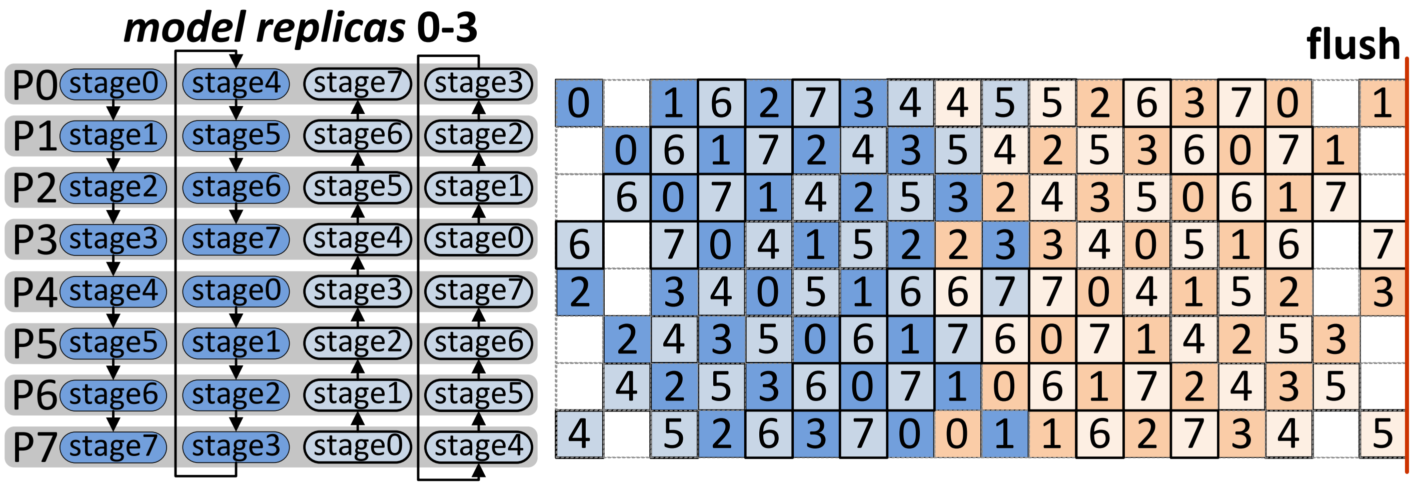
Contact email: *shigangli.cs@gmail.com*

Generalize to more pipelines



Contact email: *shigangli.cs@gmail.com*

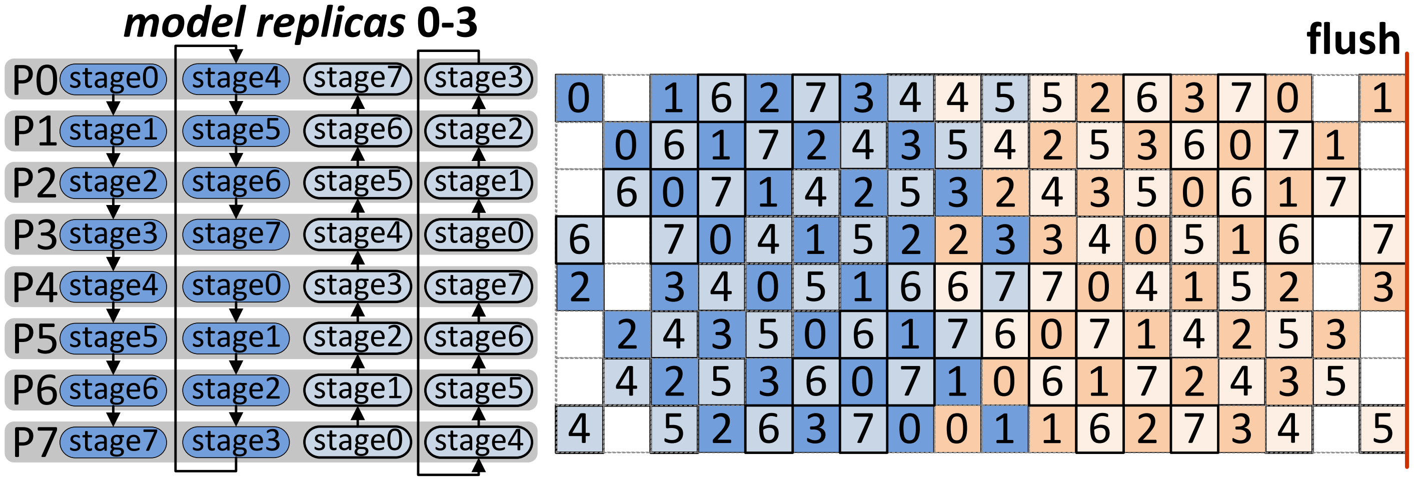
Generalize to more pipelines



Chimera (a combination of four pipelines)

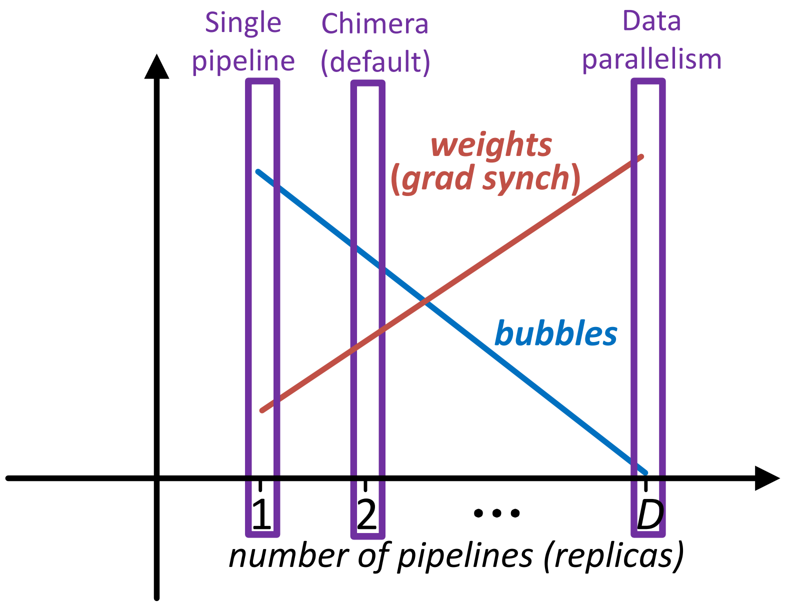
Contact email: *shigangli.cs@gmail.com*

Generalize to more pipelines



Chimera (a combination of four pipelines)

Chimera with ***r*** pipelines
(1 <= ***r*** <= ***D,*** if ***D*** is even)



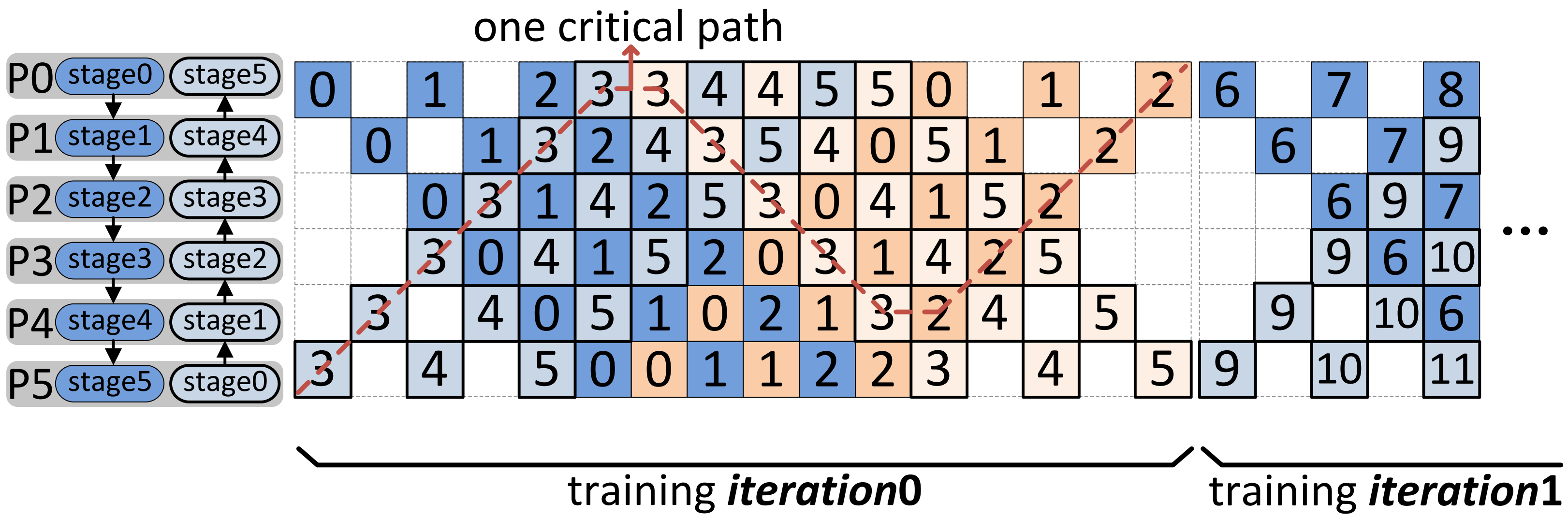
Contact email: *shigangli.cs@gmail.com*

Performance Modelling

Given a *P*, how to decide the best *D* (*depth*) and *W* (*width*)?

C_f – the number of forward passes in critical path $\quad C_f = 6$

C_b – the number of backward passes in critical path $\quad C_b = 10$



The runtime of a single training iteration is

$$T = (F_t + Comm_{p2p})C_f + (B_t + Comm_{p2p})C_b +$$

Contact email: *shigangli.cs@gmail.com*

Performance Modelling

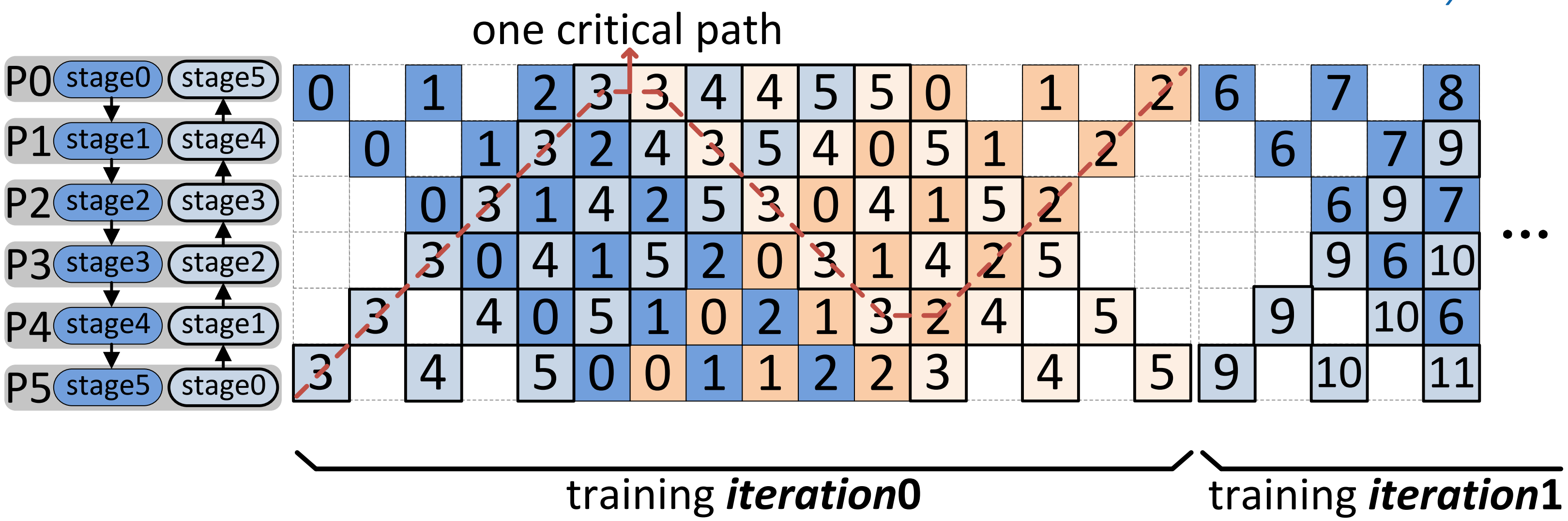
Given a ***P***, how to decide the best ***D*** (*depth*) and ***W*** (*width*)?

C_f – the number of forward passes in critical path $\quad C_f = 6$

C_b – the number of backward passes in critical path $\quad C_b = 10$

Rabenseifner's algorithm for allreduce:

$$Comm_{allreduce} = 2(\log_2 r)\underbrace{\alpha}_{latency} + 2(r-1)\underbrace{\beta}_{bandwidth} L/r$$



The runtime of a single training iteration is

$$T = (F_t + Comm_{p2p})C_f + (B_t + Comm_{p2p})C_b +$$

Contact email: *shigangli.cs@gmail.com*

Performance Modelling

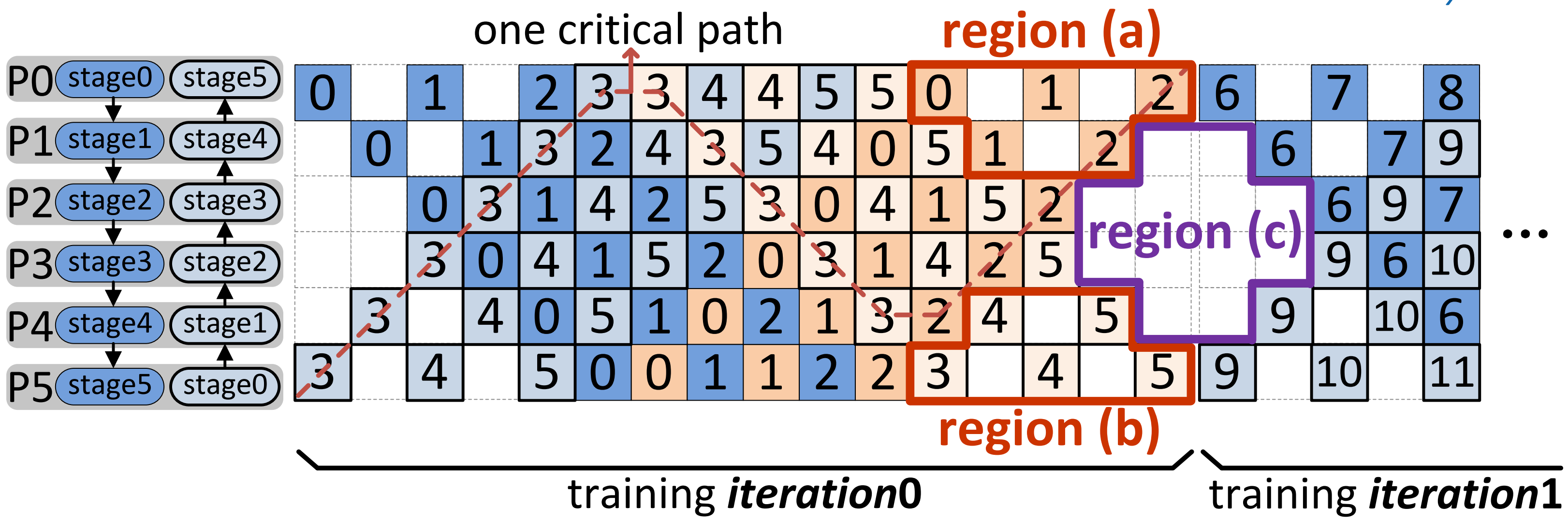
Given a *P*, how to decide the best *D* (*depth*) and *W* (*width*)?

C_f – the number of forward passes in critical path $\quad C_f = 6$

C_b – the number of backward passes in critical path $\quad C_b = 10$

Rabenseifner's algorithm for allreduce:

$$Comm_{allreduce} = 2(\log_2 r)\underbrace{\alpha}_{latency} + 2(r-1)\underbrace{\beta}_{bandwidth} L/r$$



The runtime of a single training iteration is

$$T = (F_t + Comm_{p2p})C_f + (B_t + Comm_{p2p})C_b + max\{Comm_{unoverlapped}(i) : i \in [0, D-1]\}.$$

Contact email: *shigangli.cs@gmail.com*

Evaluation

- **CSCS Piz Daint** supercomputer
  - Each node contains a 12-core Intel Xeon E5-2690 CPU, and one NVIDIA Tesla P100 GPU
  - Cray Aries interconnected network
- A small cluster with 32 NVIDIA Tesla V100 GPUs
  - 4x8 GPUs connected by NVLink and Infiniband

- Baselines include all schemes listed in Table 1: **GPipe**, **GEMS**, **DAPPLE**, **PipeDream**, **PipeDream-2BW**
- All schemes are implemented in **PyTorch** with **GLOO** distributed backend for both ***P2P*** and ***allreduce***

Table 3. Neural networks used for evaluation

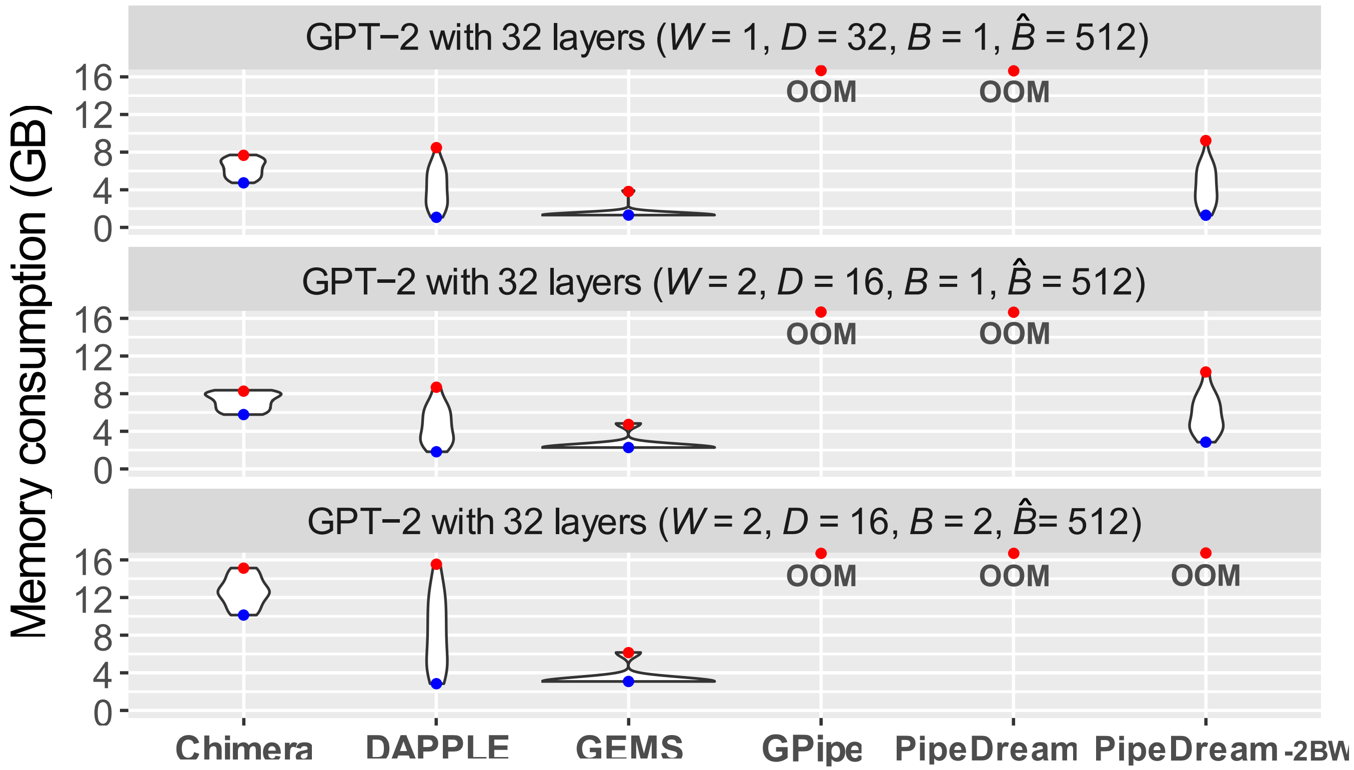
Networks	Layers	Parameters	Mini-batch size
Bert-48	48	669,790,012	>=256
GPT-2	64	1,389,327,360	>=512

Table 4. List of symbols

D	The number of pipeline stages (*depth*)
W	The number of replicated pipelines (*width*) for data parallelism
B	Micro-batch size
$\hat{B}$	Mini-batch size ($= B * N * W$)
R	Activation recomputation is required

Contact email: *shigangli.cs@gmail.com*

Memory consumption

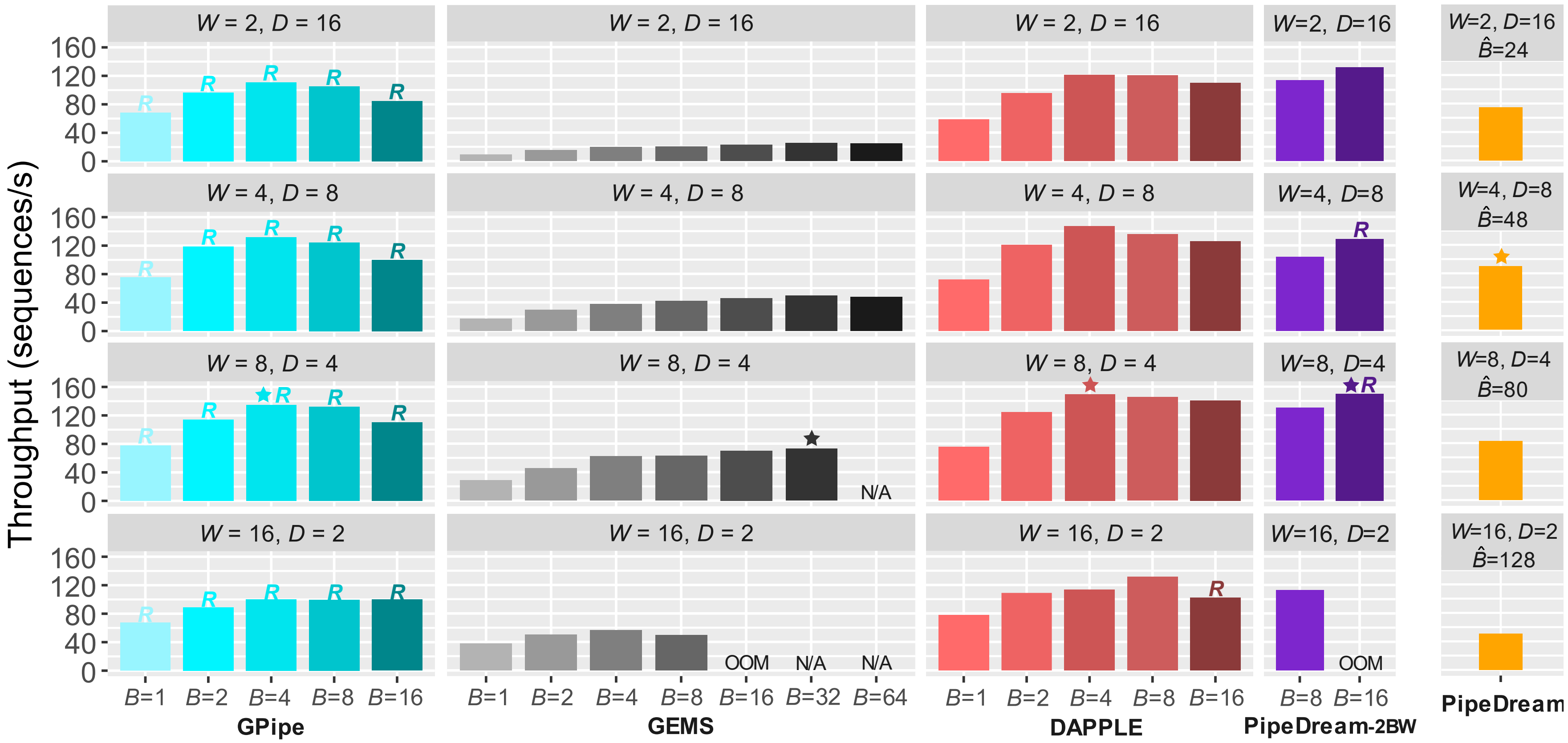


Pipeline Schemes	Weights Memory	Activations Memory
PipeDream [38]	$[M_\theta,\ D*M_\theta]$ [1]	$[M_a,\ D*M_a]$ [1]
PipeDream-2BW [39]	$2M_\theta$	$[M_a,\ D*M_a]$ [1]
GPipe [26]	M_θ	$N*M_a$
GEMS [28]	$2M_\theta$	M_a
DAPPLE [16]	M_θ	$[M_a,\ D*M_a]$ [1]
Chimera (this work)	$2M_\theta$	$[(D/2+1)M_a, D*M_a]$ [1] +

Contact email: *shigangli.cs@gmail.com*

Tuning for baselines

The parameters of ***D***, ***W***, and ***B*** affect the performance significantly.



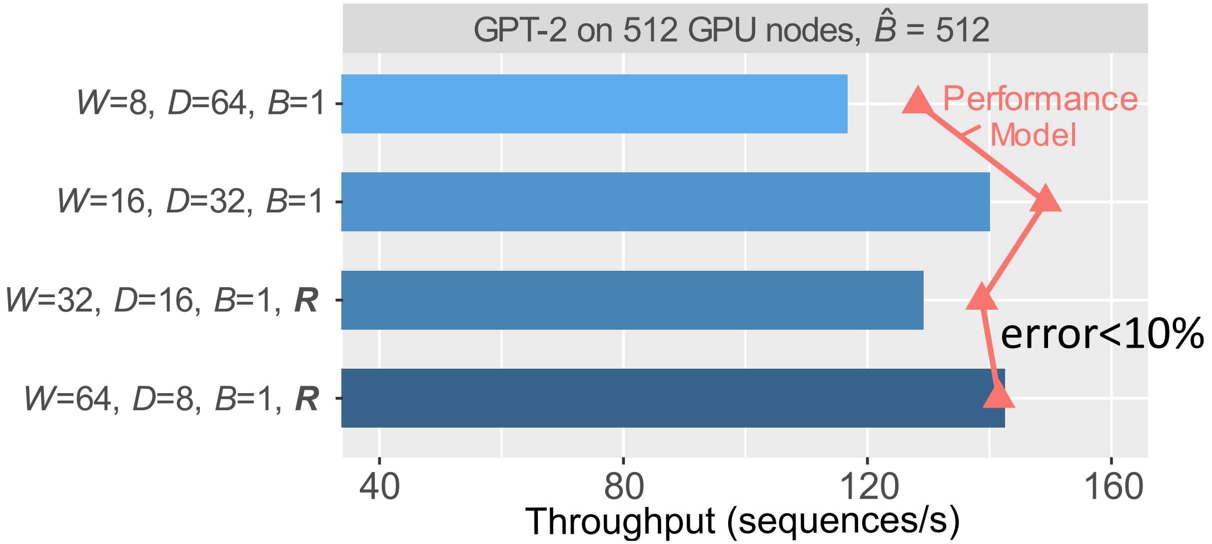
Performance tuning for the baselines for Bert-48 on 32 GPU nodes.
R denotes **activation recomputation** to avoid **OOM**. *Star* marks the best performance.

Contact email: *shigangli.cs@gmail.com*

Performance modelling of Chimera

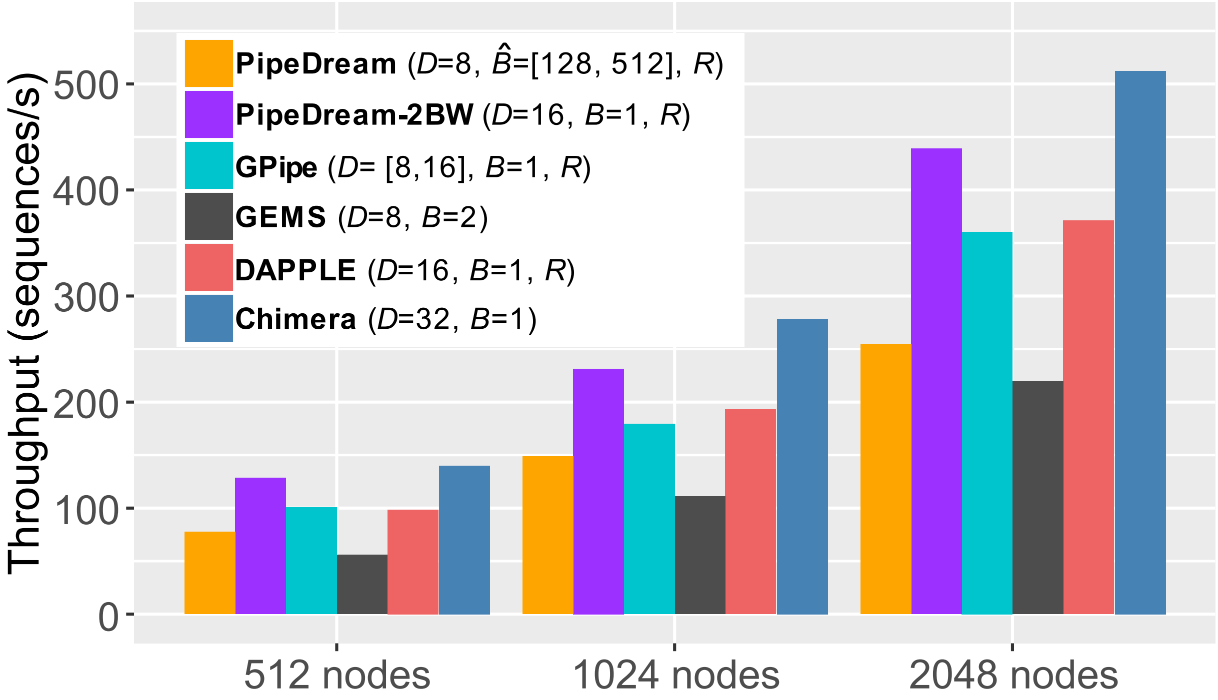
The runtime of a single training iteration is modelled as

$$T = (F_t + Comm_{p2p})C_f + (B_t + Comm_{p2p})C_b + max\{Comm_{unoverlapped}(i) : i \in [0, D-1]\}.$$



Contact email: *shigangli.cs@gmail.com*

Weak scaling

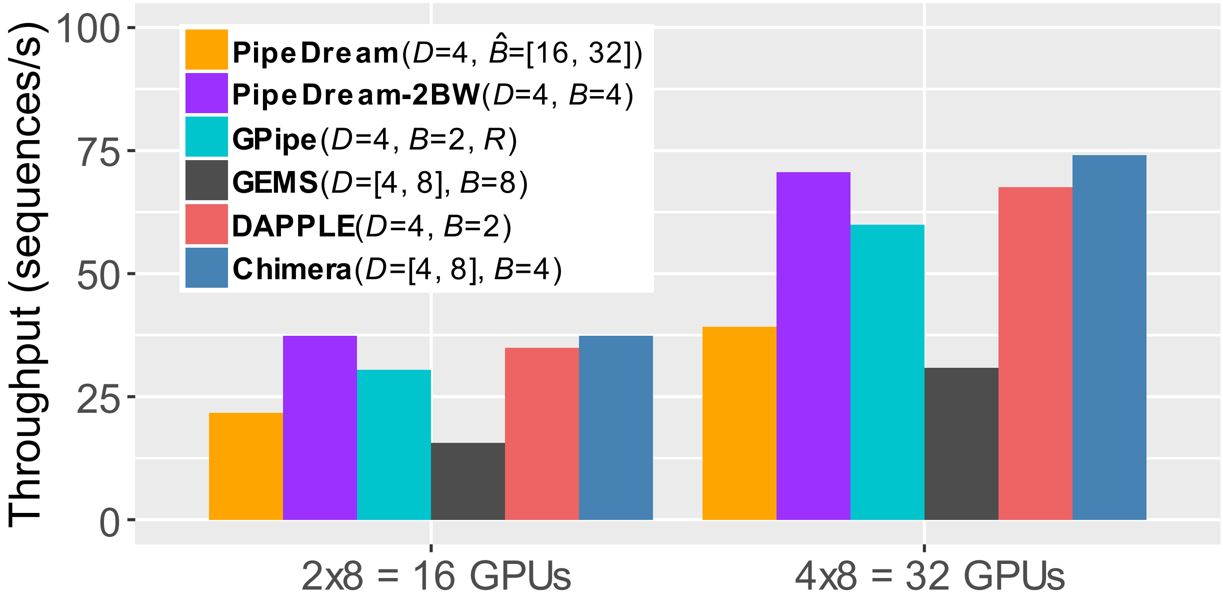


Weak scaling for GPT-2 on Piz Daint
(512 to 2048 GPU nodes)

- **1.38x - 2.34x speedup over synchronous approaches (GPipe, GEMS, DAPPLE)**
  - Less bubbles
  - More balanced memory thus no recomputation
- **1.16x - 2.01x speedup over asynchronous approaches (PipeDream-2BW, PipeDream)**
  - More balanced memory thus no recomputation
  - Gradient accumulation thus low synch frequency

Contact email: *shigangli.cs@gmail.com*

Weak scaling

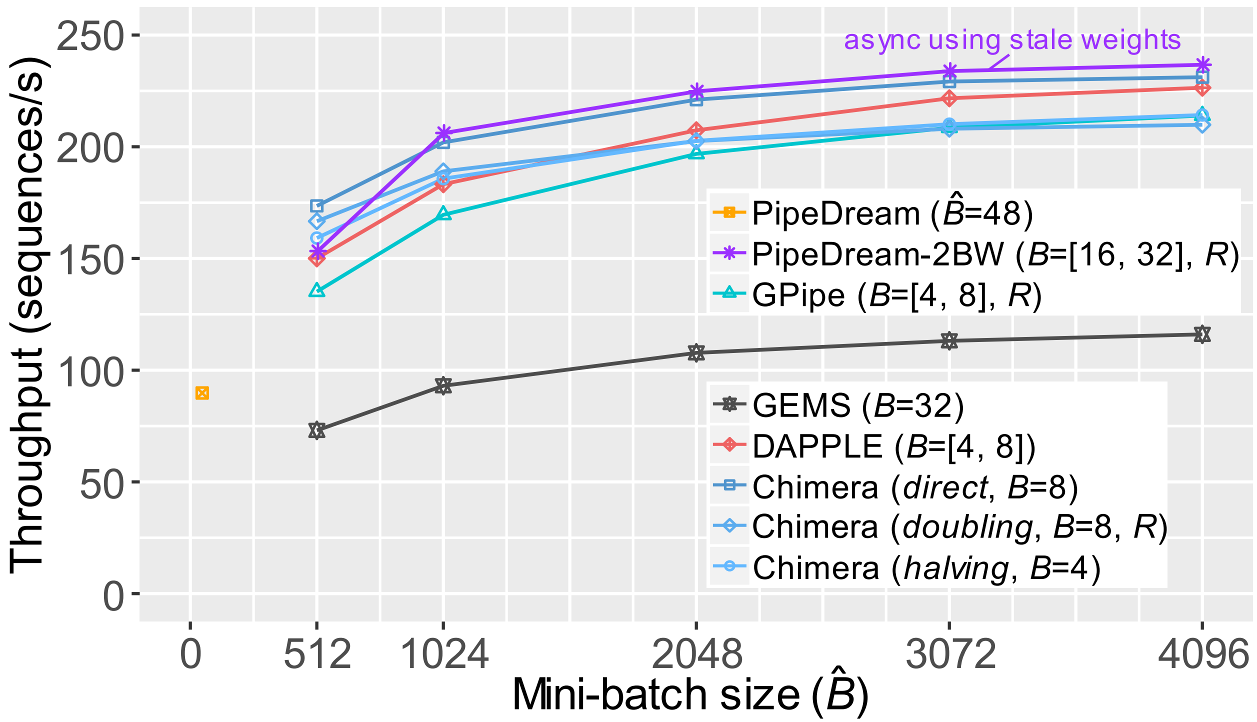


Weak scaling for Bert-48 on a cluster with 32 V100 GPUs, sequence length is 512.

- **Similar conclusion holds for BERT on the cluster with newer GPUs and heterogeneous interconnected networks.**

Contact email: *shigangli.cs@gmail.com*

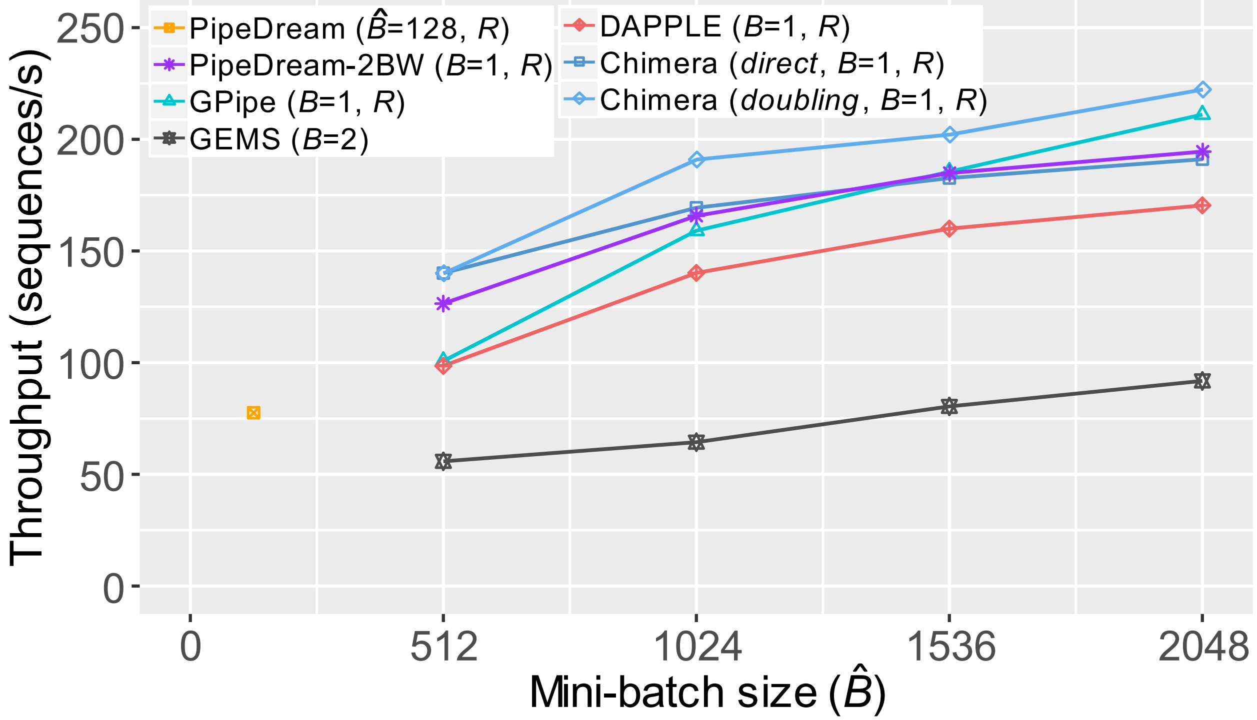
Scale to large mini-batches



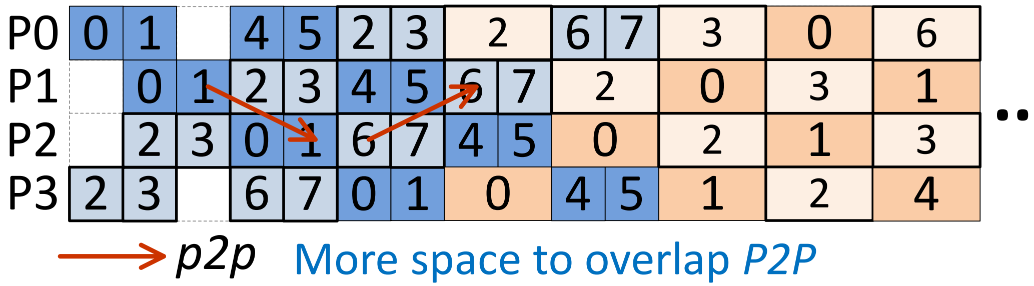
Scale to large mini-batch size for Bert-48
on 32 GPU nodes of Piz Daint.

Contact email: *shigangli.cs@gmail.com*

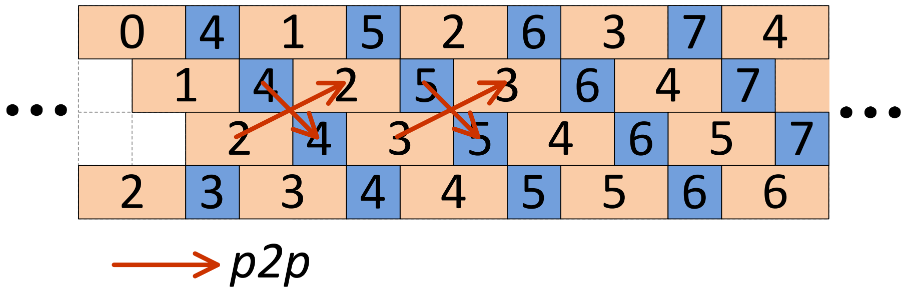
Scale to large mini-batches



Scale to large mini-batch size for GPT-2 on 512 GPU nodes of Piz Daint.



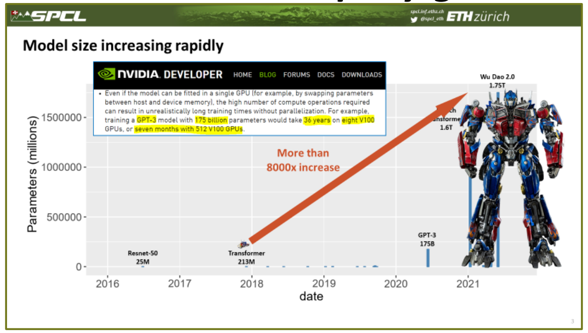
Chimera with *forward doubling*



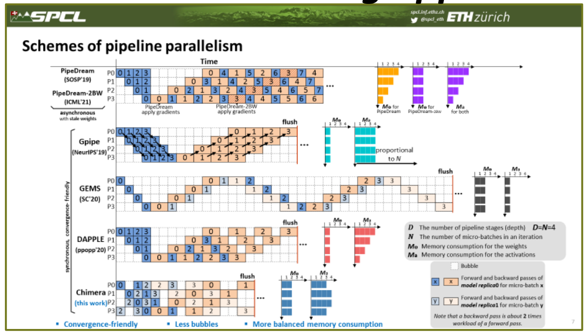
PipeDream-2BW

Conclusion

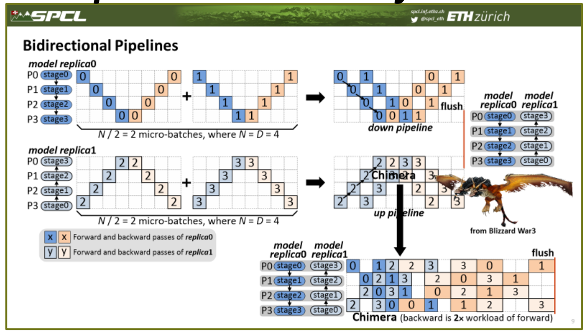
1. Model size rapidly grows



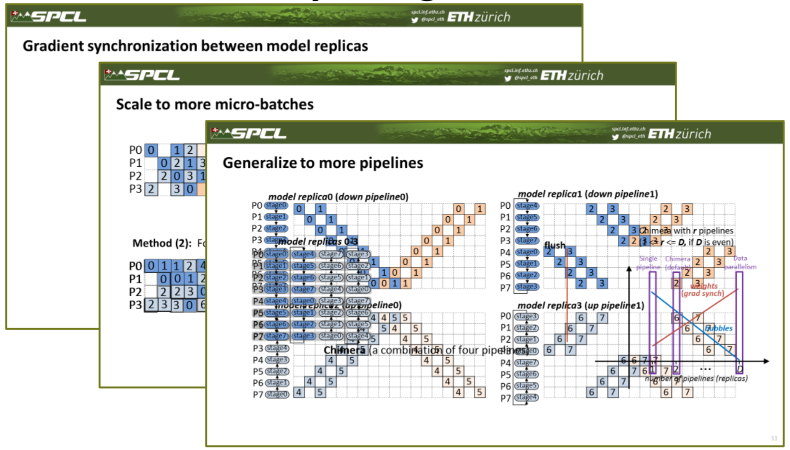
2. Distributed training approaches



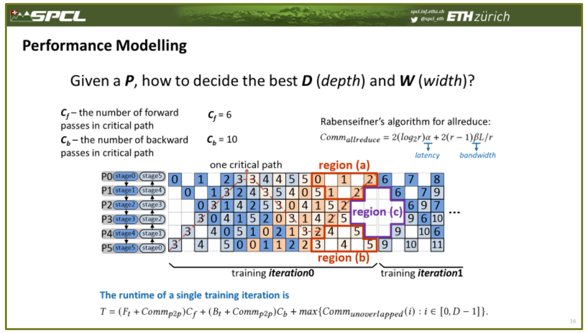
3. Pipeline schedule of Chimera



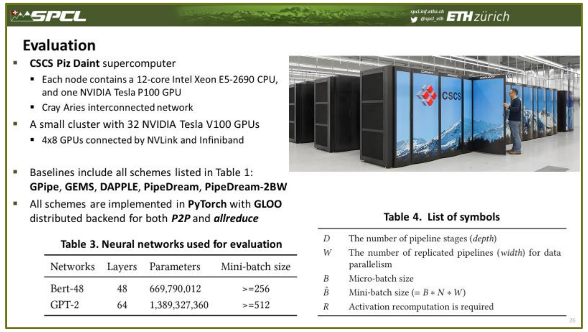
4. Scalability and generalization



5. Performance modelling



6. Evaluation on supercomputer



For any questions contact: ***shigangli.cs@gmail.com***