THE UNIVERSITY of EDINBURGH
informatics

Fundamentals for Distributed Machine Learning

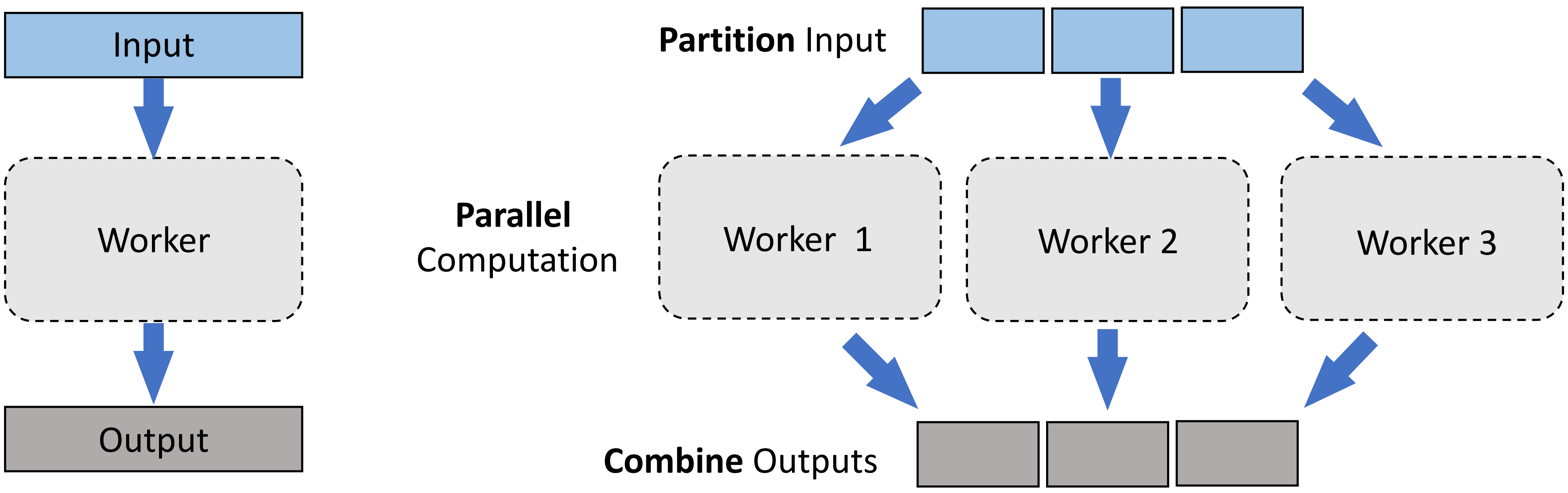
Luo Mai

University of Edinburgh

Motivation

Gap to be filled by **Distributed ML Systems**

Divide-and-Conquer



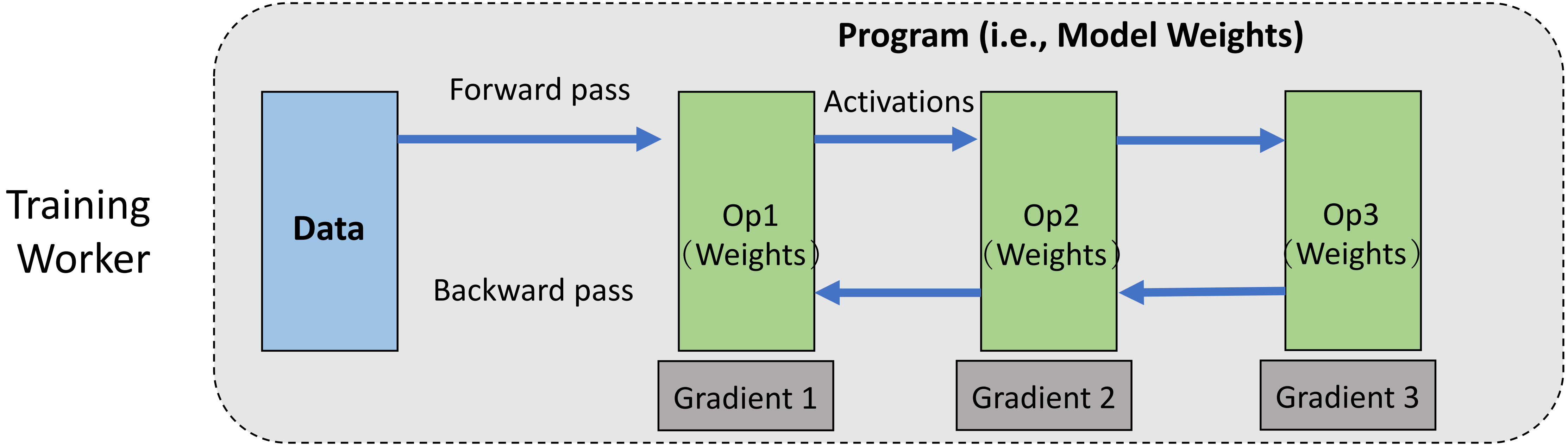
（a） Single-worker Execution

（b） Distributed Multi-worker Execution

Why distributed ML systems?

- Performance
  - Reducing the time to complete a data epoch
- Memory wall
- Economy
  - Multiple commodity servers, instead of a single expensive high-end server
- Hardware failure tolerance

Basic execution model



Workers must have sufficient memory to store data, weights, activations & gradients

- Otherwise, you get Out-Of-Memory (OOM) exception

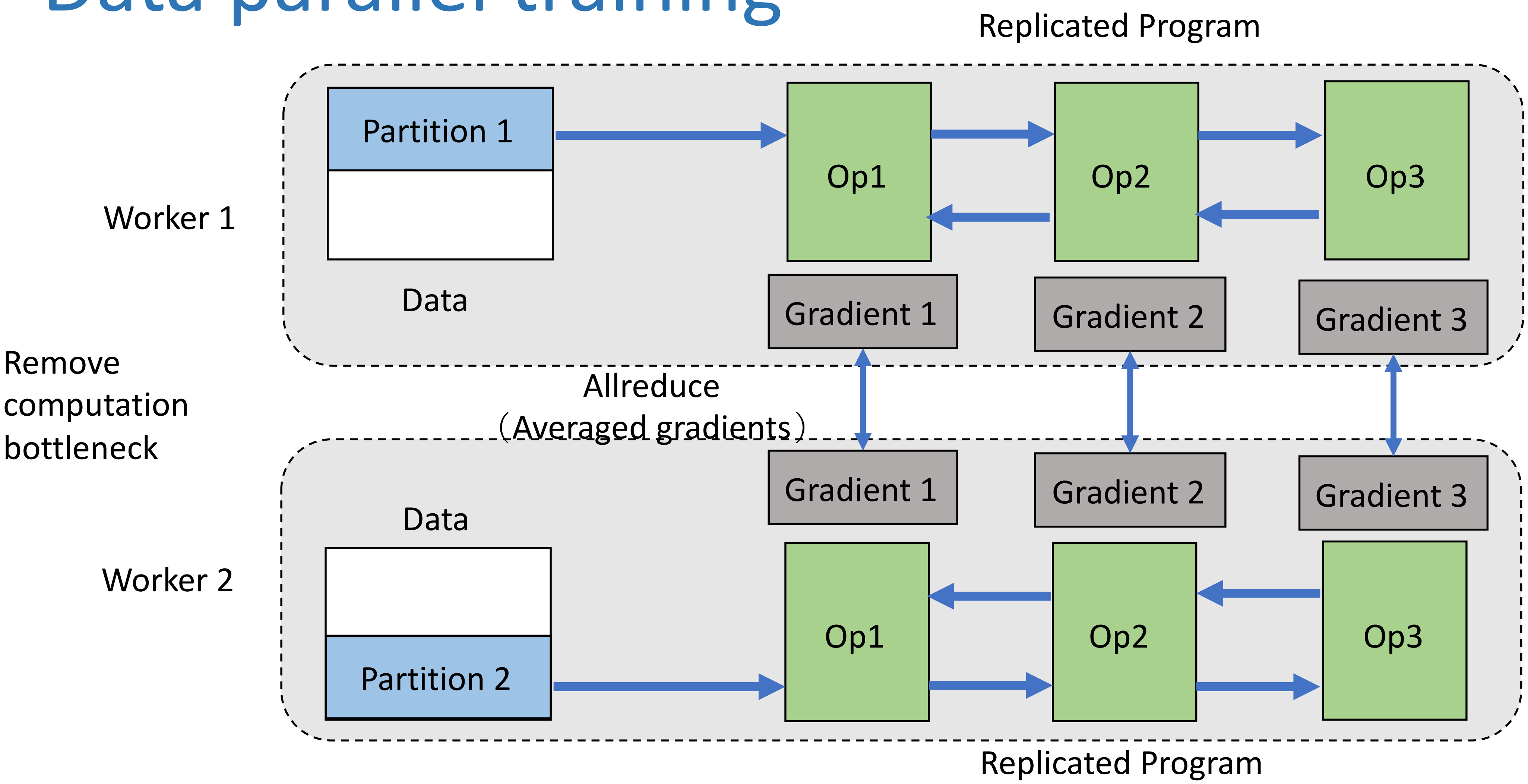
Questions?

Parallel training methods

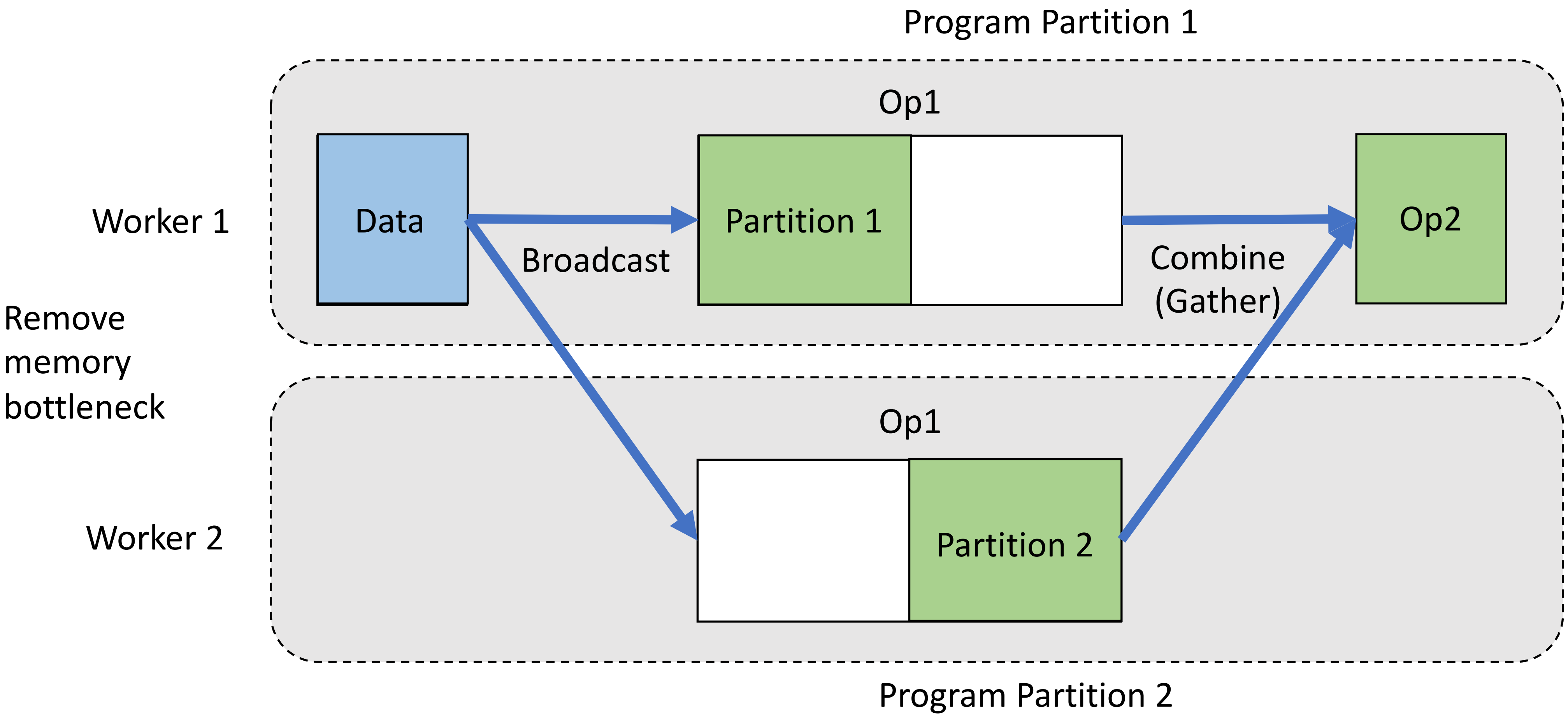
	Single Data	Multiple Data
Single Program	Single-Program-Single-Data (SPSD) **Single Worker Training**	Single-Program-Multiple-Data (SPMD) **Data Parallel Training**
Multiple Program	Multiple-Program-Single-Data (MPSD) **Model Parallel Training**	Multiple-Program-Multiple-Data (MPMD) **Hybrid Parallel Training**

![informatics logo]

Data parallel training

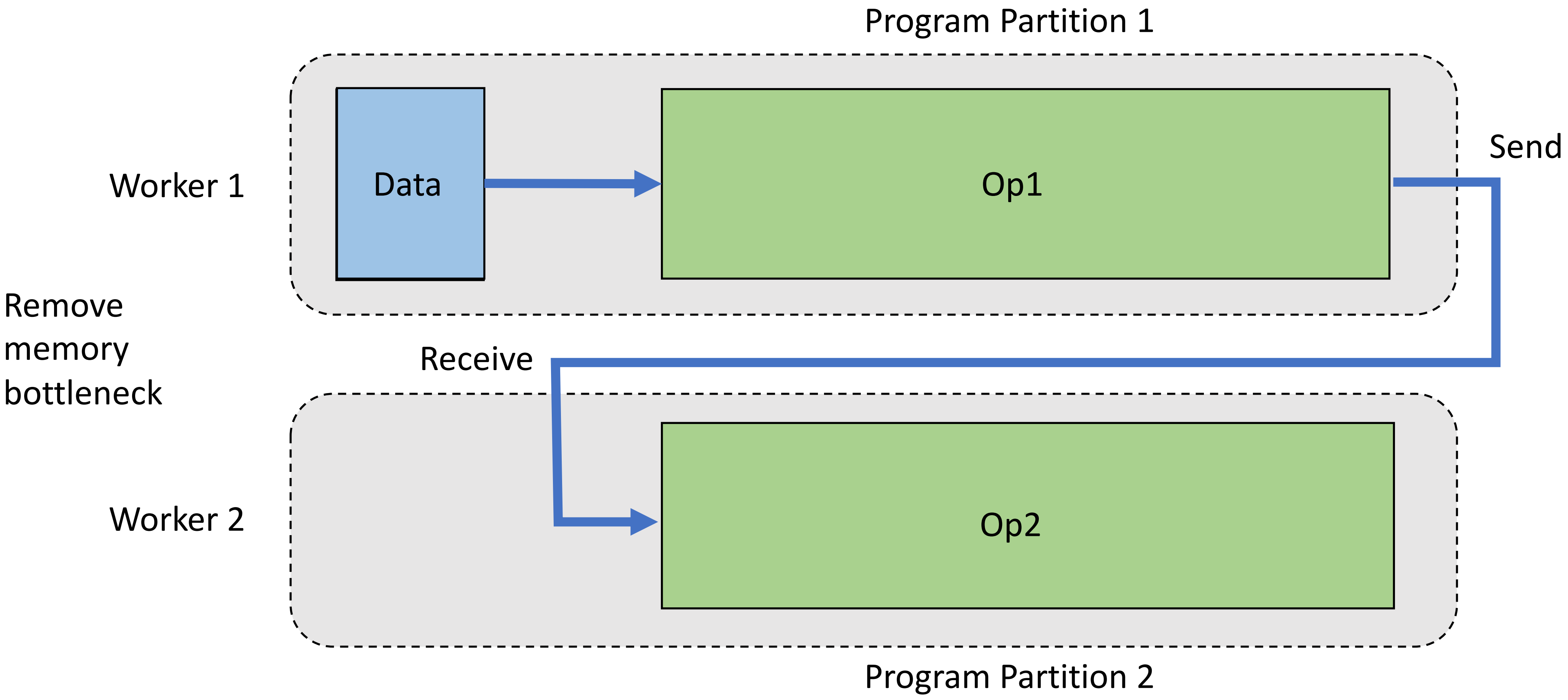


Model parallel training: Intra-operator

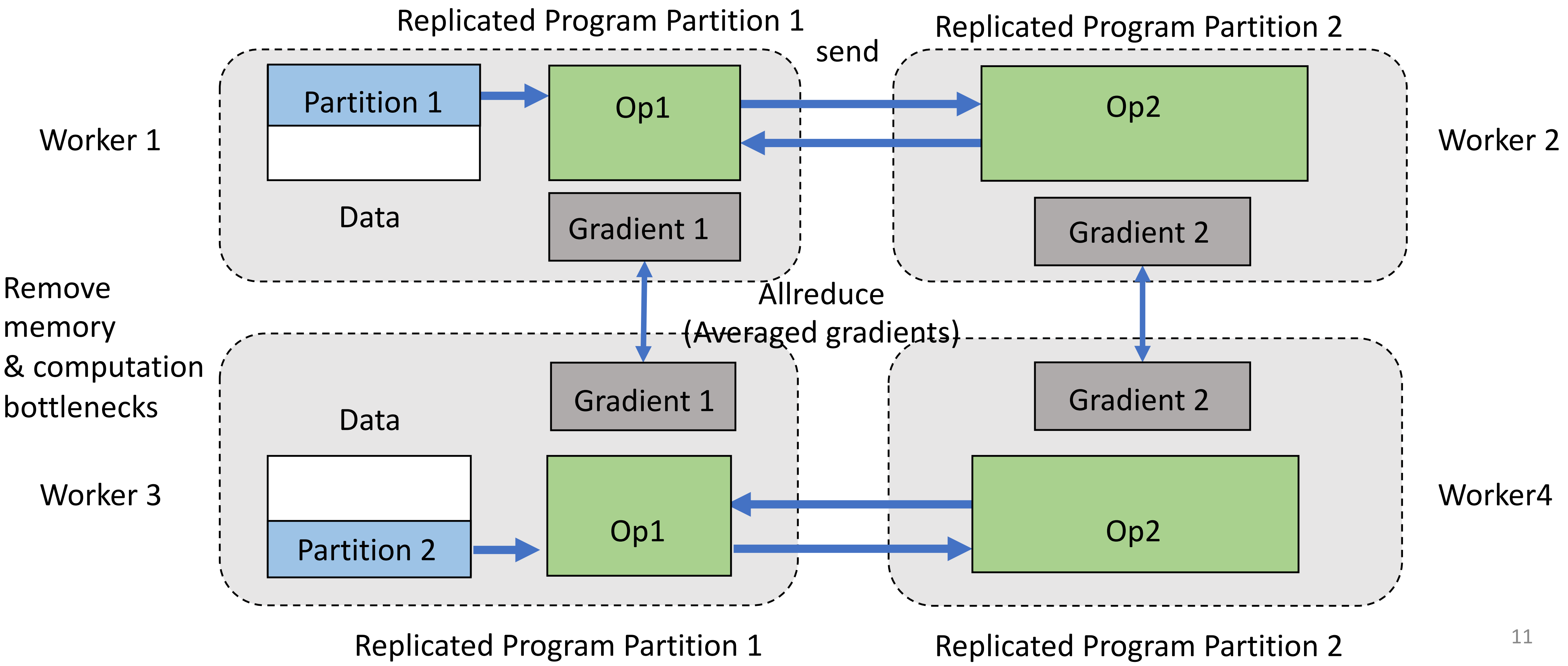


THE UNIVERSITY of EDINBURGH informatics

Model parallel training: Inter-operator



Hybrid parallel training



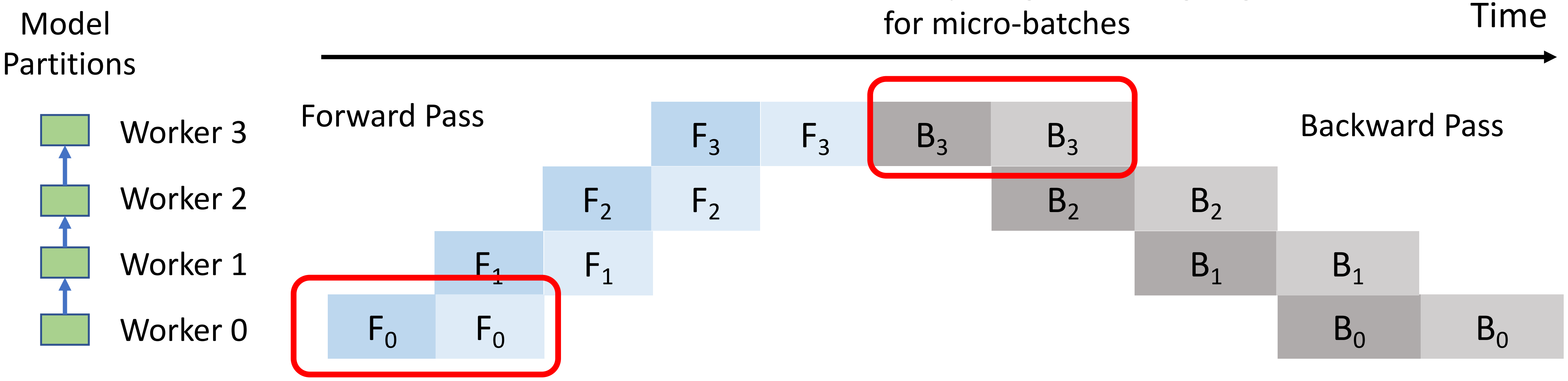
Questions?

How to choose parallelism methods?

- Empirical parallelism
  - TensorFlow Mesh
- Semi-automatic parallelism
  - Manually partition a few upstream operators and propagate the partitioning to downstream operators
- Automatic parallelism
  - Build a cost model for evaluating different parallel methods
  - Search for the best methods that incur minimal costs

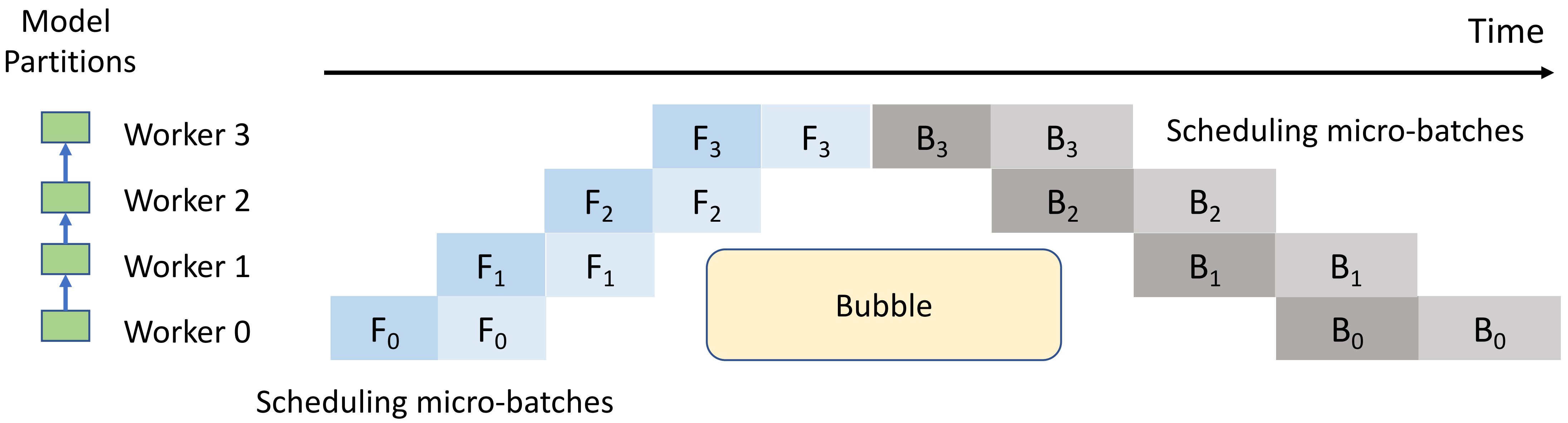
Pipeline parallelism

Computing the **averaged gradients** for micro-batches



Creating pipeline by dividing a data partition into **micro-batches**

Optimising micro-batch size



- **Small micro-batch** reduces bubble size; but incur large micro-batch scheduling overheads
- **Large micro-batch** incurs large bubble; but come with small micro-batch scheduling overheads
- Optimal micro-batch size must **balance bubble size and scheduling overheads**

Design aspects of distributed ML systems

- Cluster elasticity
  - Reserving a large number of GPU servers is prohibitively expensive
- Device roles in ML
  - CPUs for data processing, GPUs for training (PyTorch & TensorFlow)
  - This is causing problems in graph learning and reinforcement learning
- Mixed precision training
  - FP8, FP16, FP32, FP64

Summary

- Distributed ML systems are keys to tackle “End of Moore Law”
- Performance and economy benefits
- Spatial parallelism: Data-parallel, model-parallel and hybrid-parallel
- Temporal parallelism: Pipeline parallelism

Reading

- Optional reading
  - [PyTorch distributed overview]
  - [Google GPipe paper]

Questions?

Large-Scale System Software Group

- ML system projects (capstone project / final-year projects)
- Open-source project: the 1st open-sourced textbook for ML systems

Personal website: https://luomai.github.io
Email: luo.mai@ed.ac.uk

If interested, please send me your CV, transcript, and a description of your interest, and we can arrange a follow-up meeting