Efficient Flow Scheduling in Distributed Deep Learning Training with Echelon Formation

Rui Pan*, Yiming Lei*, Jialong Li, Zhiqiang Xie, Binhang Yuan, Yiting Xia

HotNets '22, Nov 14-15 2022, Austin, TX, USA

Deep Neural Networks (DNNs) are popular

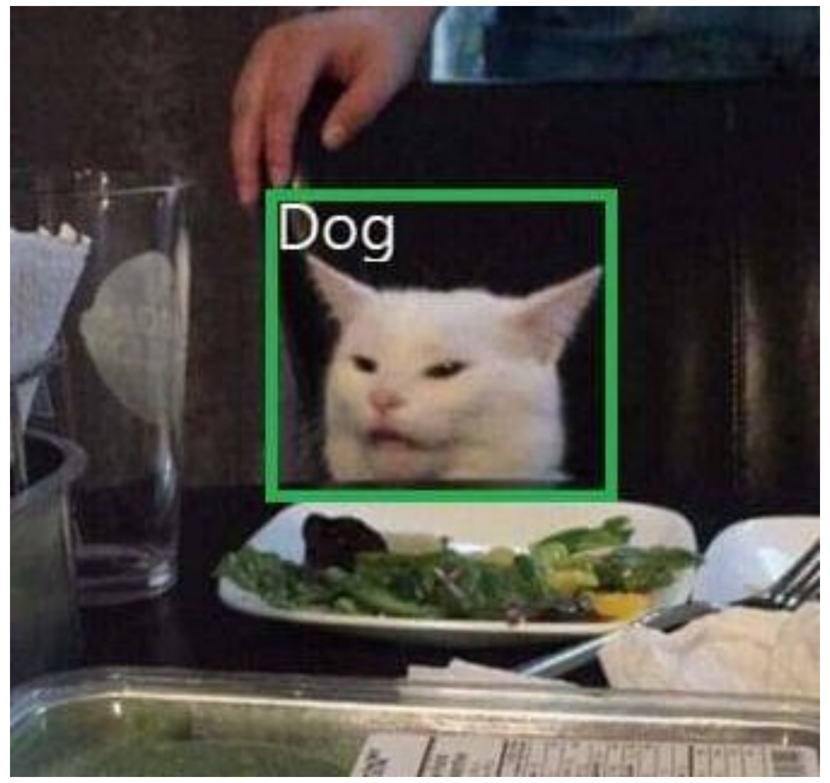


Image Classification

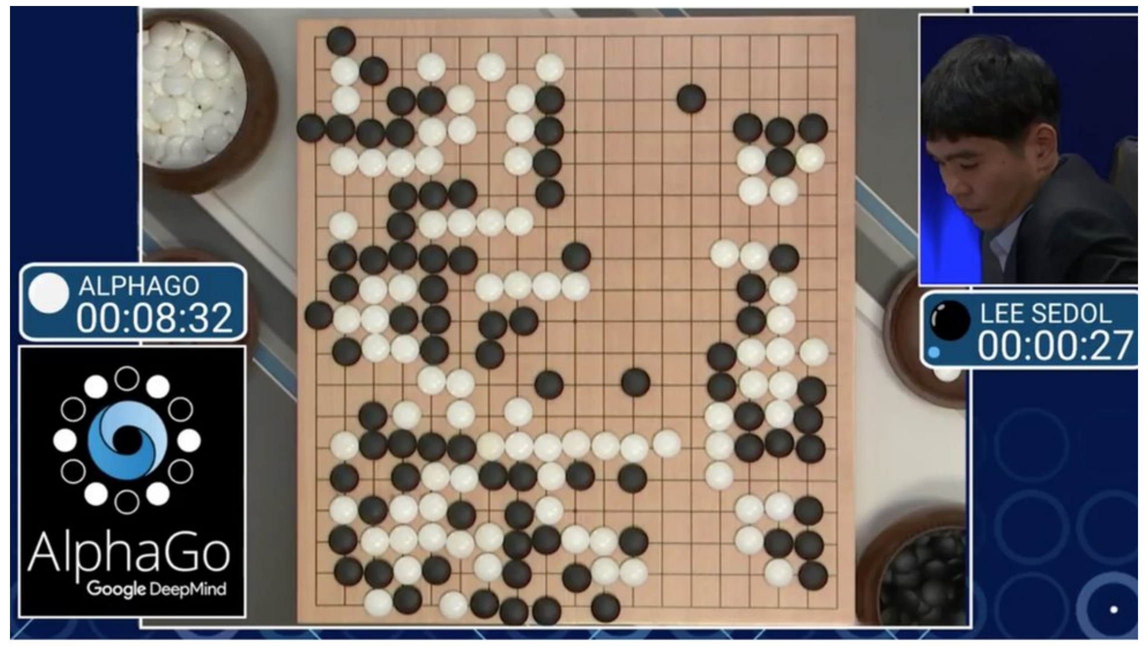
Machine Translation

Alexa

Google Assistant

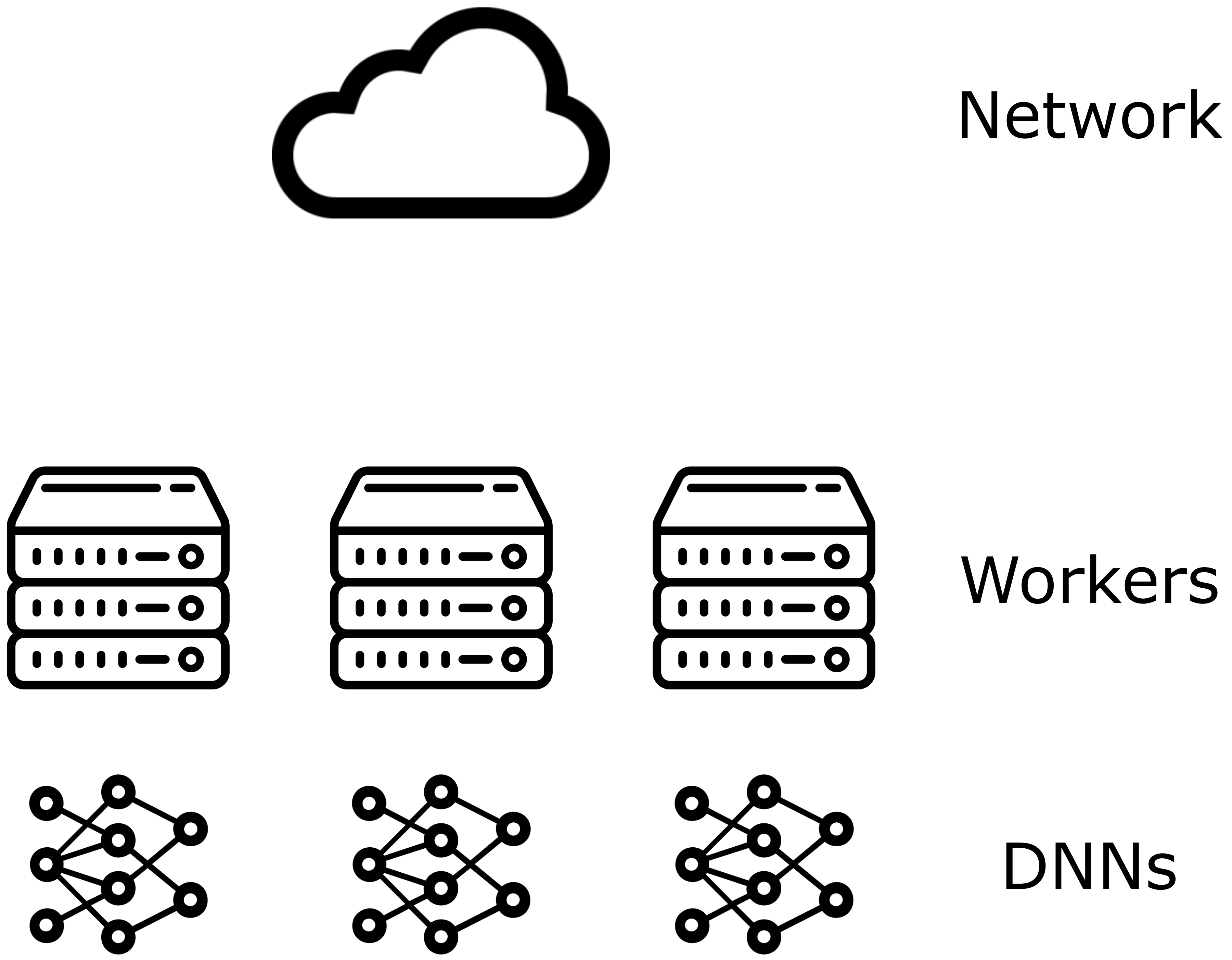
Siri

Speech-to-Text



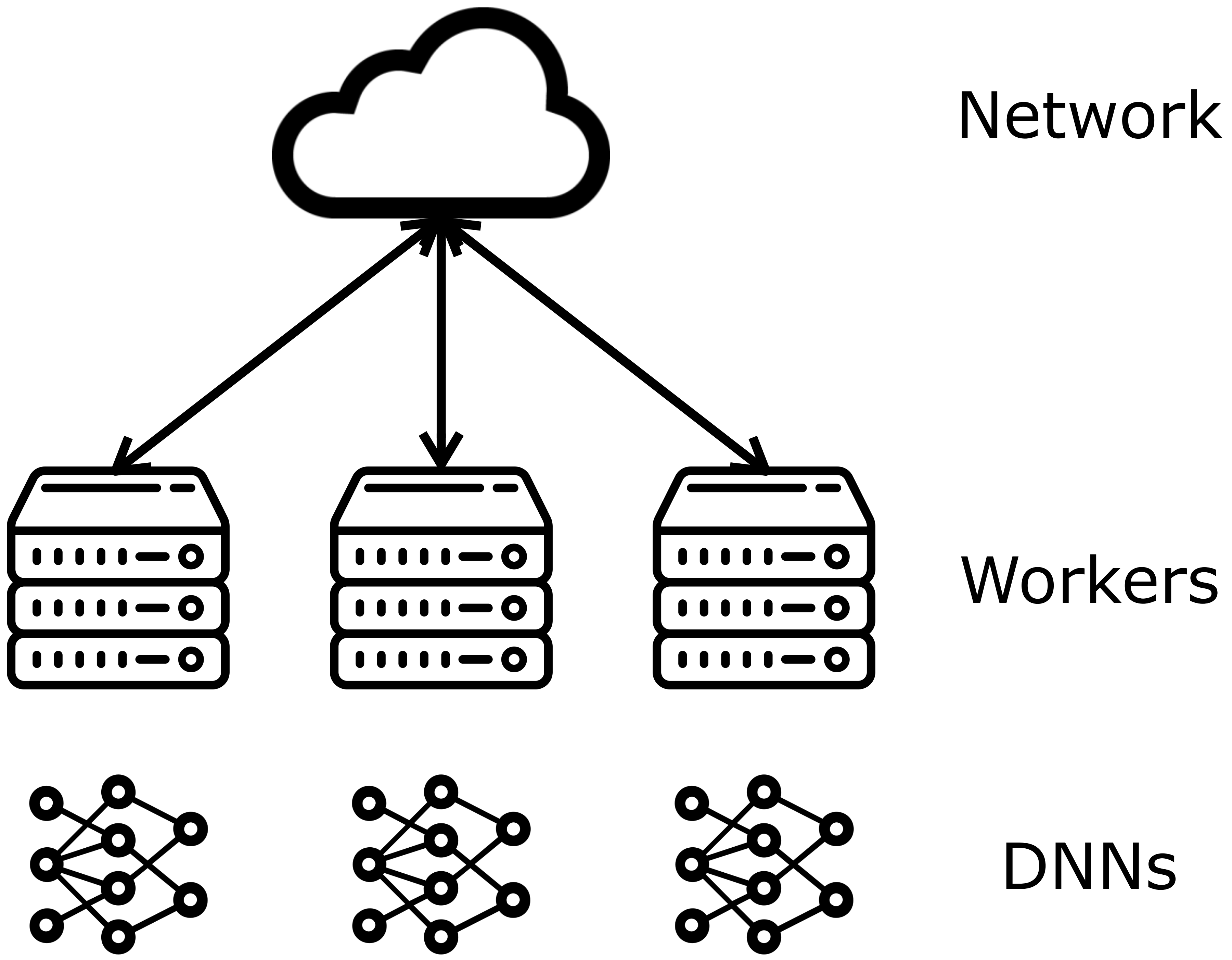
Game Playing

Communication is a bottleneck in distributed DNN training



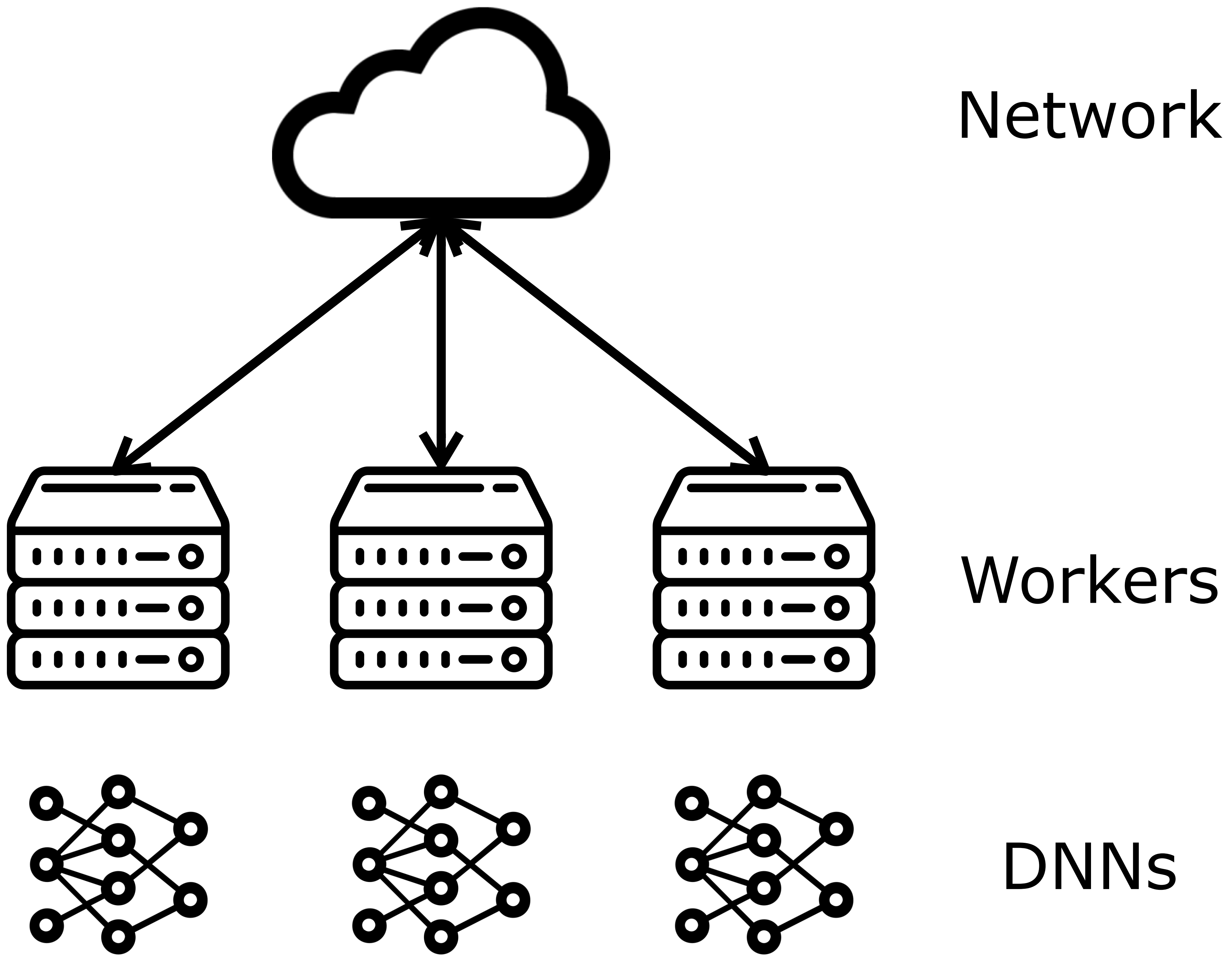
Distributed Training in GPU Clusters

Communication is a bottleneck in distributed DNN training

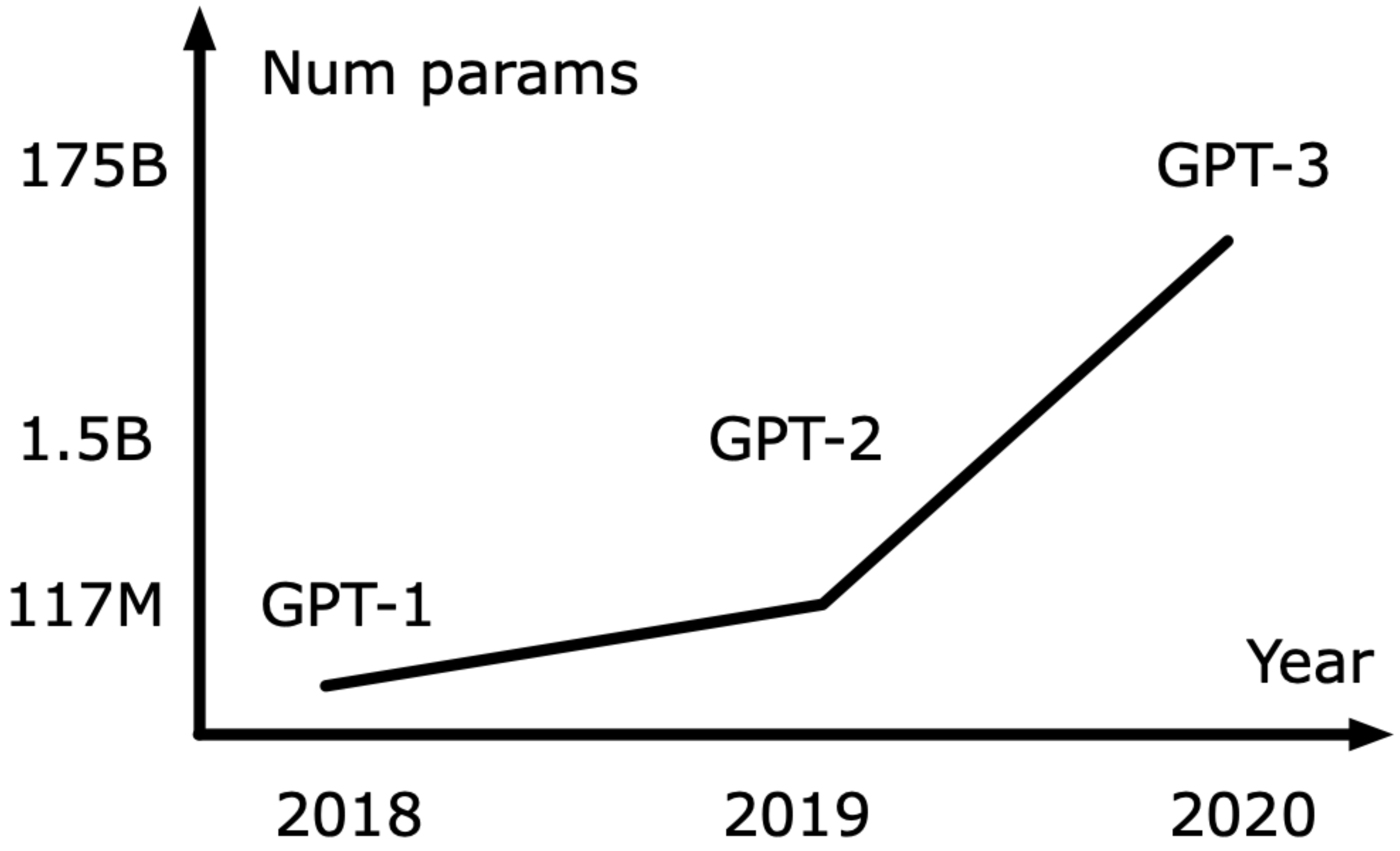


Distributed Training in GPU Clusters

Communication is a bottleneck in distributed DNN training

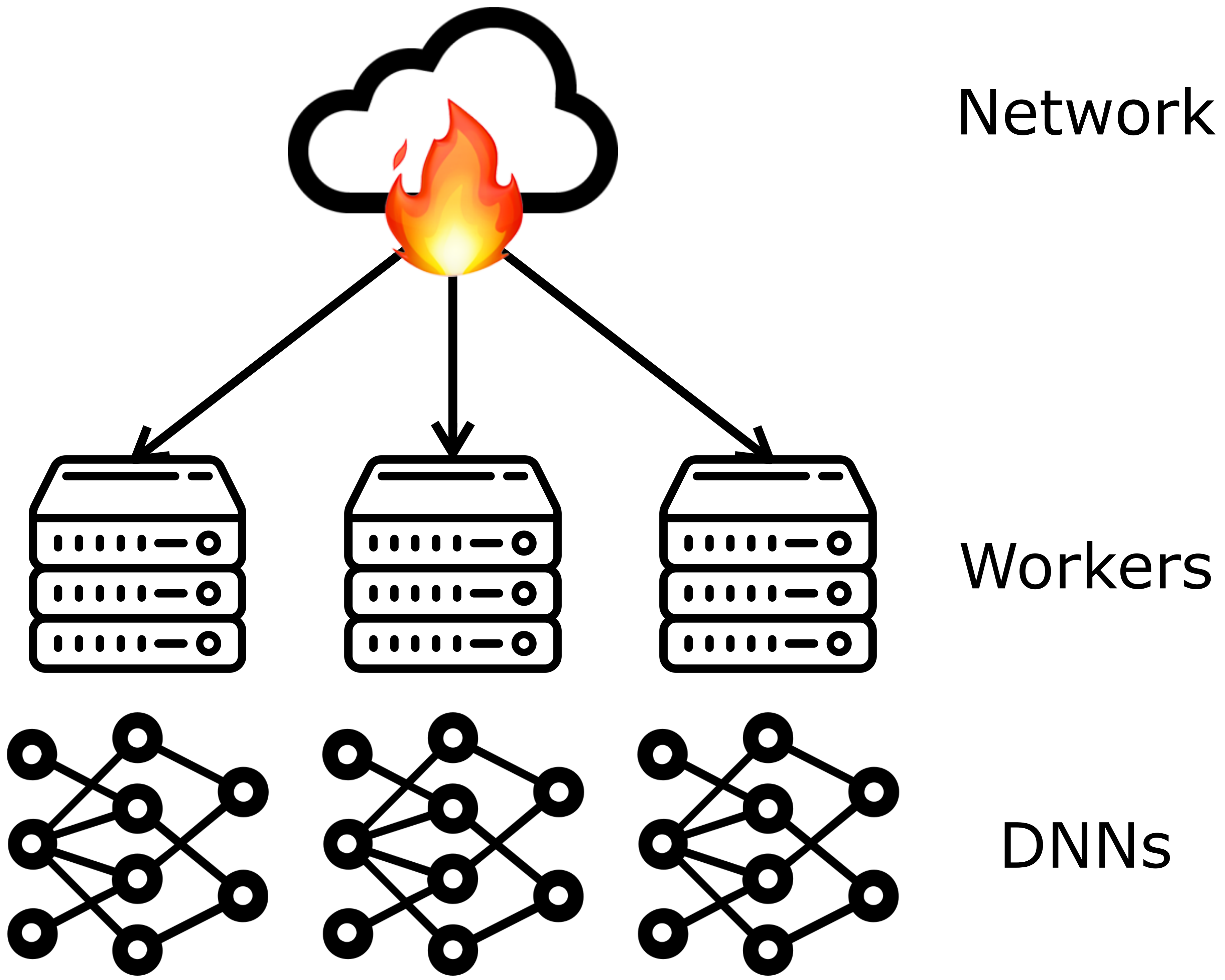


Distributed Training in GPU Clusters

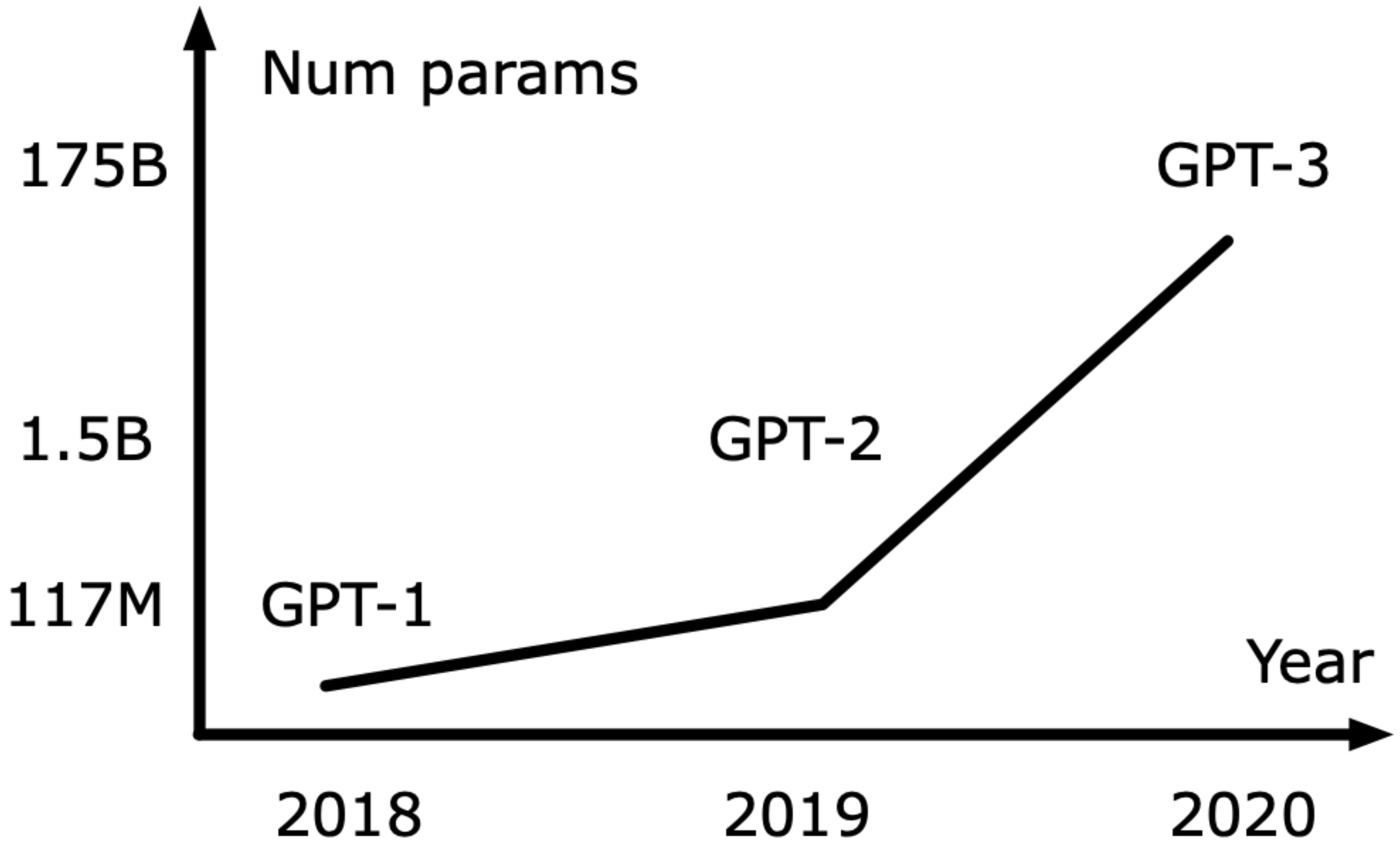


GPT Model Size Evolution

Communication is a bottleneck in distributed DNN training



Distributed Training in GPU Clusters



GPT Model Size Evolution

Flow scheduling mitigates the communication bottleneck

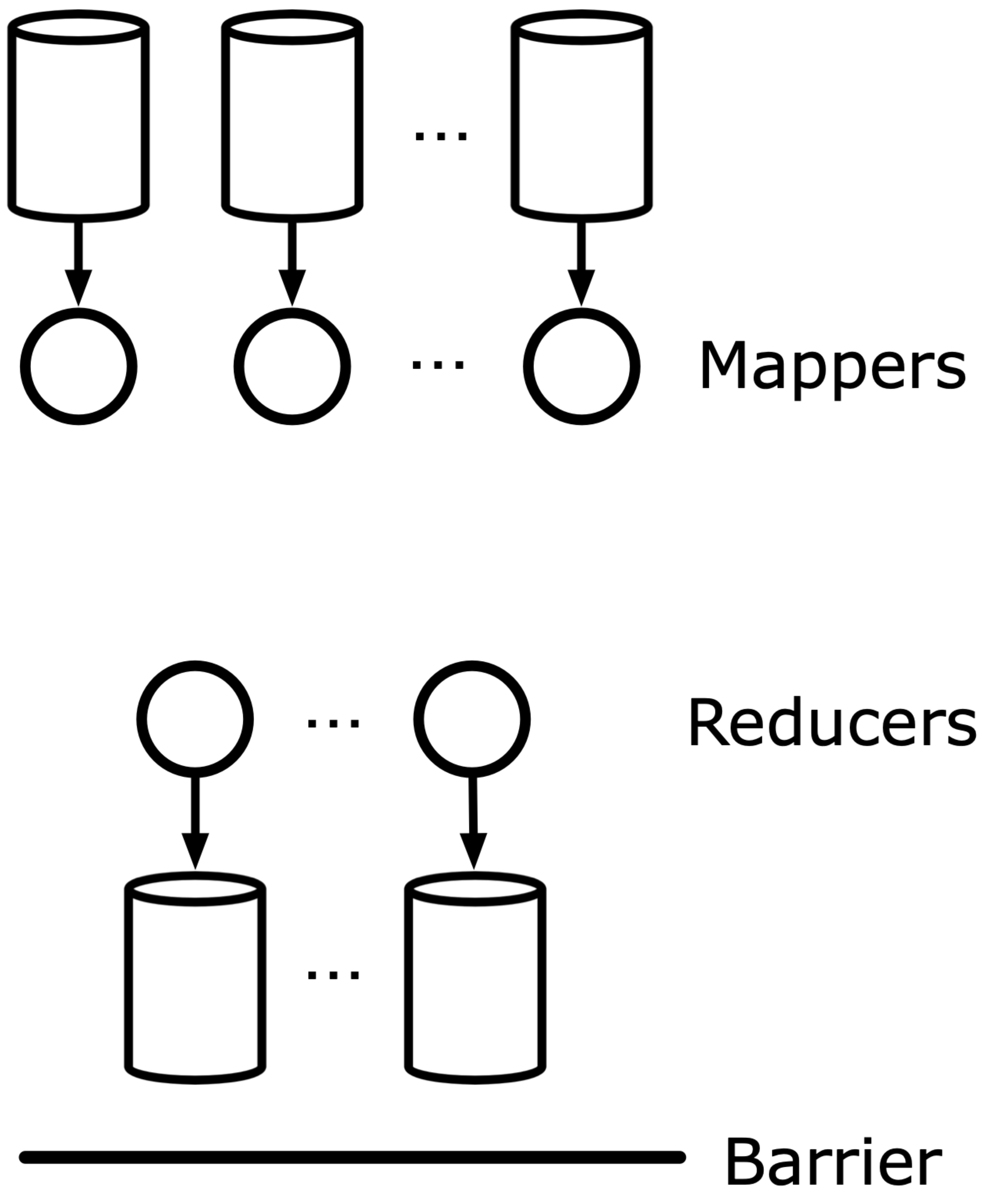
Flow scheduling mitigates the communication bottleneck

- CoFlow: network abstraction for flow scheduling

Flow scheduling mitigates the communication bottleneck

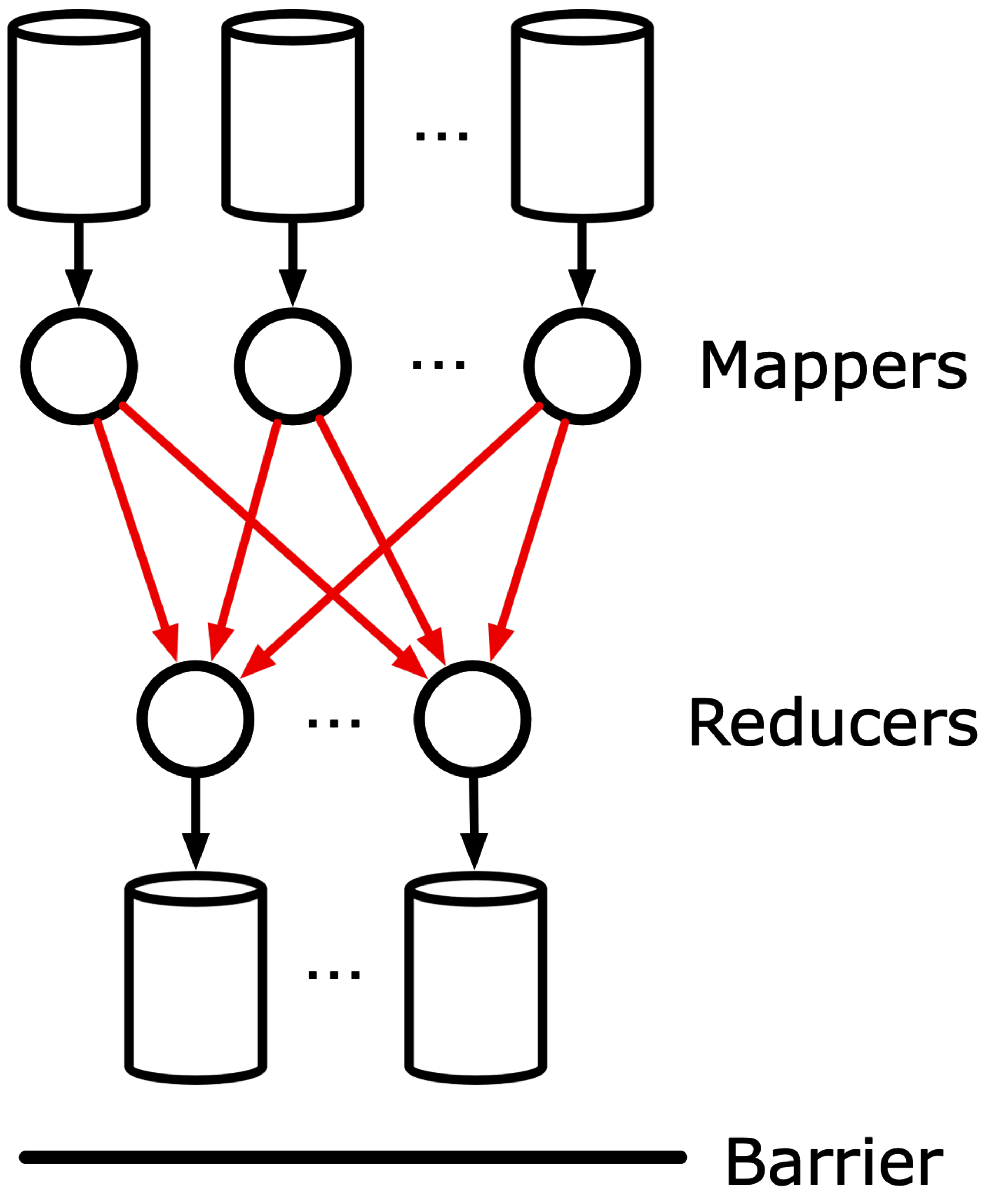
- CoFlow: network abstraction for flow scheduling
- A group of semantically-related flows, optimized to *finish at the same time*

Flow scheduling mitigates the communication bottleneck



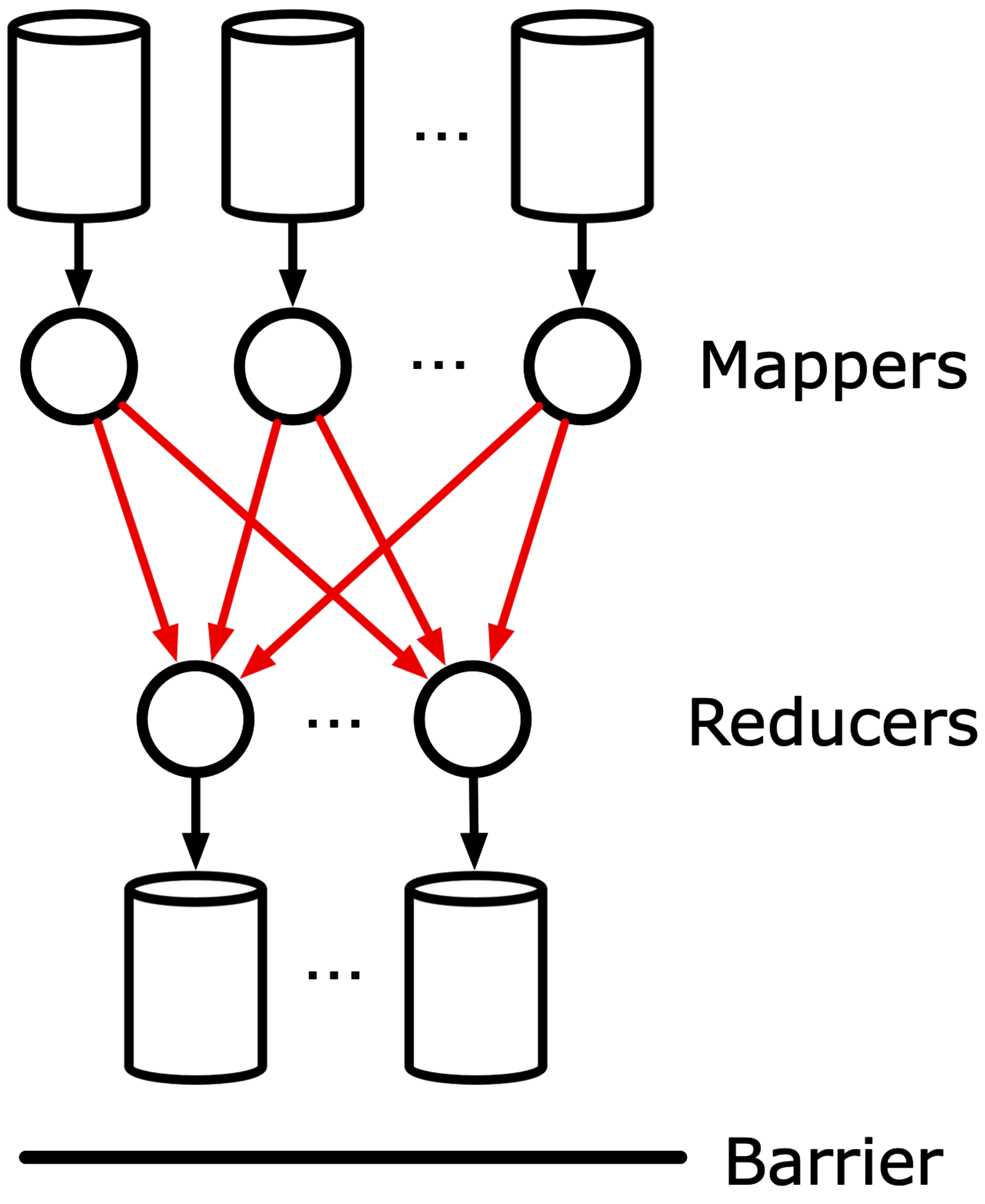
- CoFlow: network abstraction for flow scheduling
- A group of semantically-related flows, optimized to *finish at the same time*

Flow scheduling mitigates the communication bottleneck



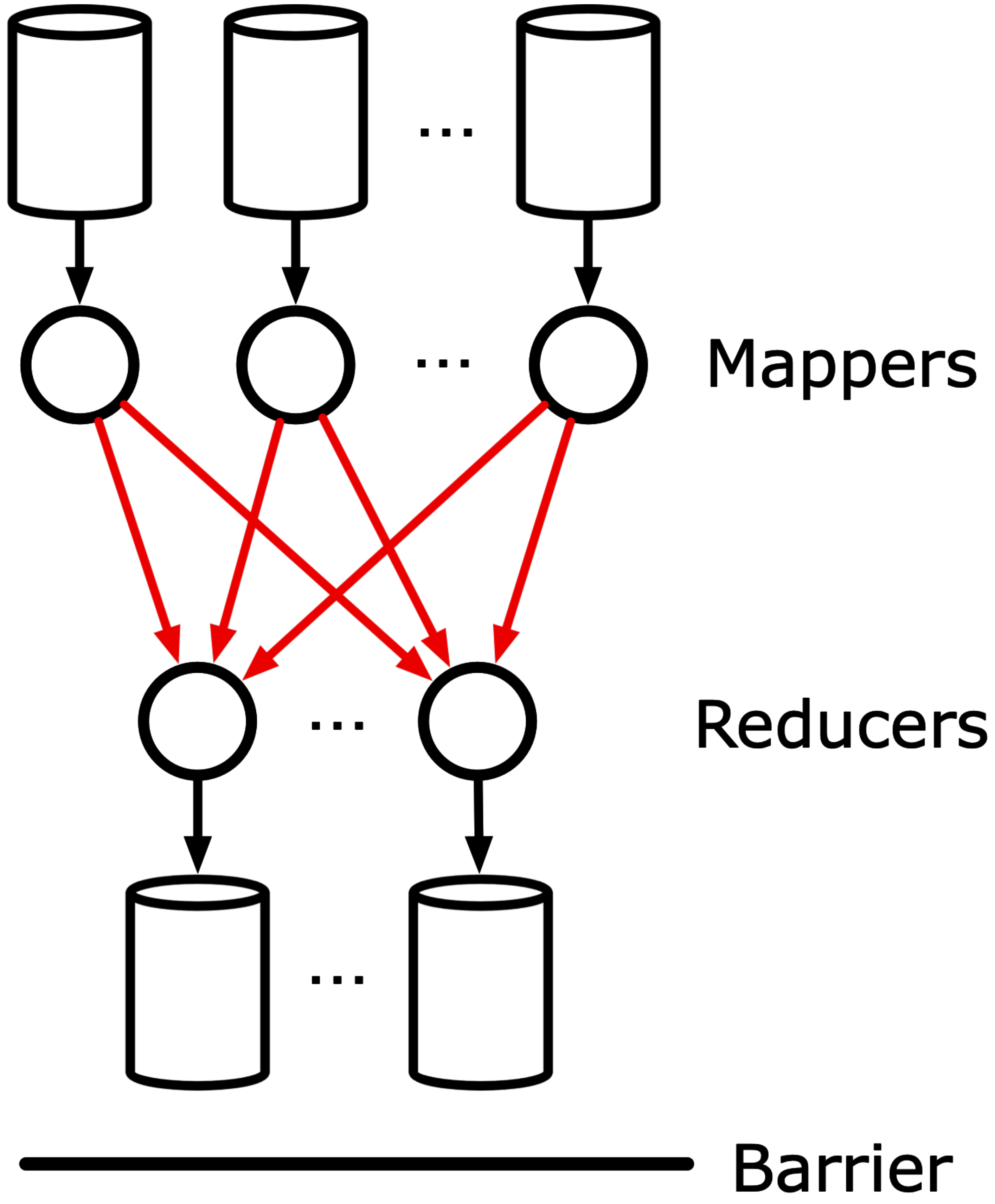
- CoFlow: network abstraction for flow scheduling
- A group of semantically-related flows, optimized to *finish at the same time*

Flow scheduling mitigates the communication bottleneck



- CoFlow: network abstraction for flow scheduling
- A group of semantically-related flows, optimized to *finish at the same time*
  - Big data frameworks 👍

Flow scheduling mitigates the communication bottleneck



- CoFlow: network abstraction for flow scheduling
- A group of semantically-related flows, optimized to *finish at the same time*
    - Big data frameworks 👍
    - ML training paradigms 👎

Why not flow scheduling for ML?

Why not flow scheduling for ML?

Training paradigm	Examples
Data Parallelism	AllReduce, Parameter Server
Pipeline Parallelism	GPipe, PipeDream
Tensor Parallelism	Megatron-LM
Fully Sharded Data Parallelism	ZeRO, FairScale

Why not flow scheduling for ML?

Training paradigm	Examples
Data Parallelism	AllReduce, Parameter Server
Pipeline Parallelism	GPipe, PipeDream
Tensor Parallelism	Megatron-LM
Fully Sharded Data Parallelism	ZeRO, FairScale

- Parallelization paradigms have different communication patterns

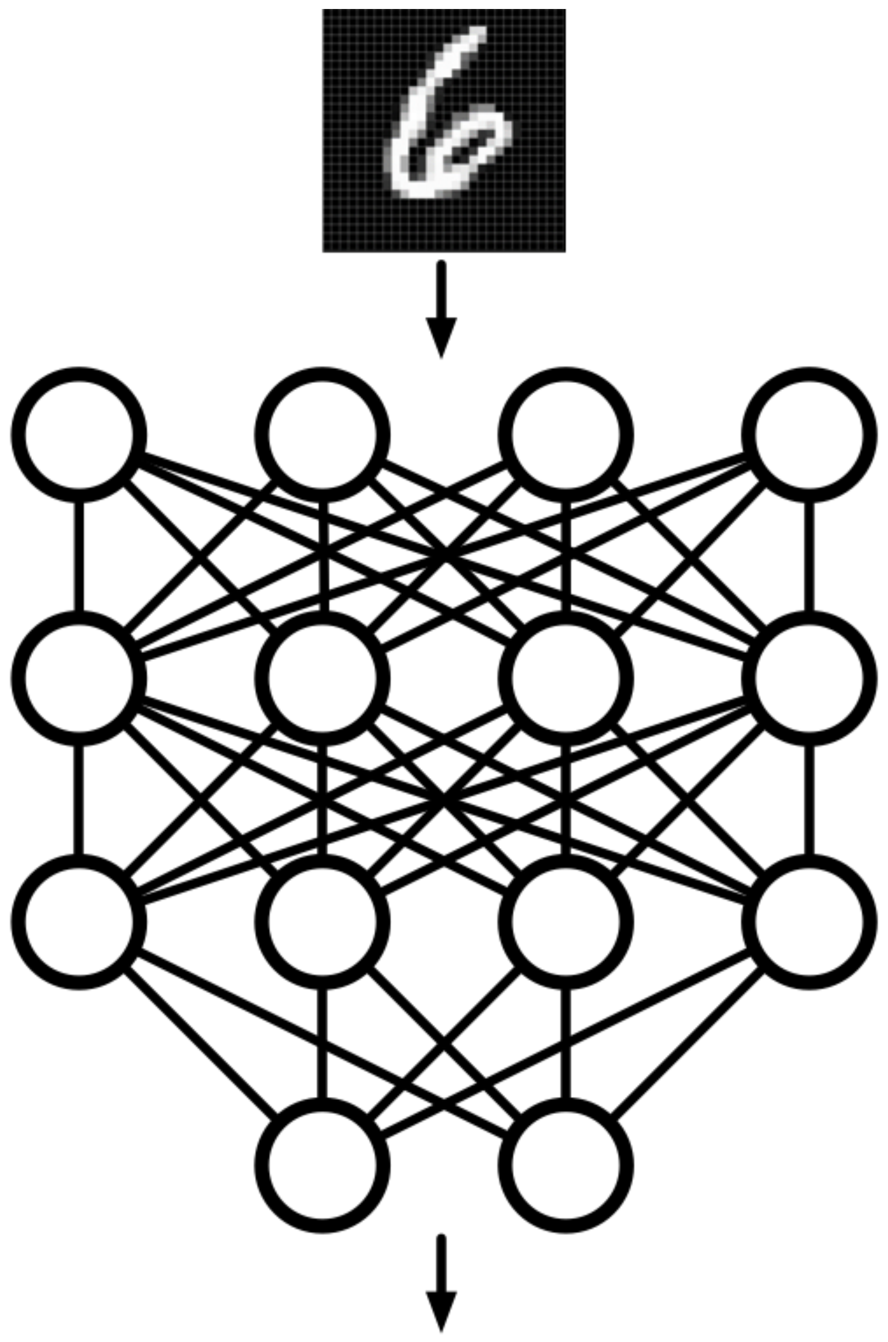
Why not flow scheduling for ML?

Training paradigm	Examples
Data Parallelism	AllReduce, Parameter Server
Pipeline Parallelism	GPipe, PipeDream
Tensor Parallelism	Megatron-LM
Fully Sharded Data Parallelism	ZeRO, FairScale

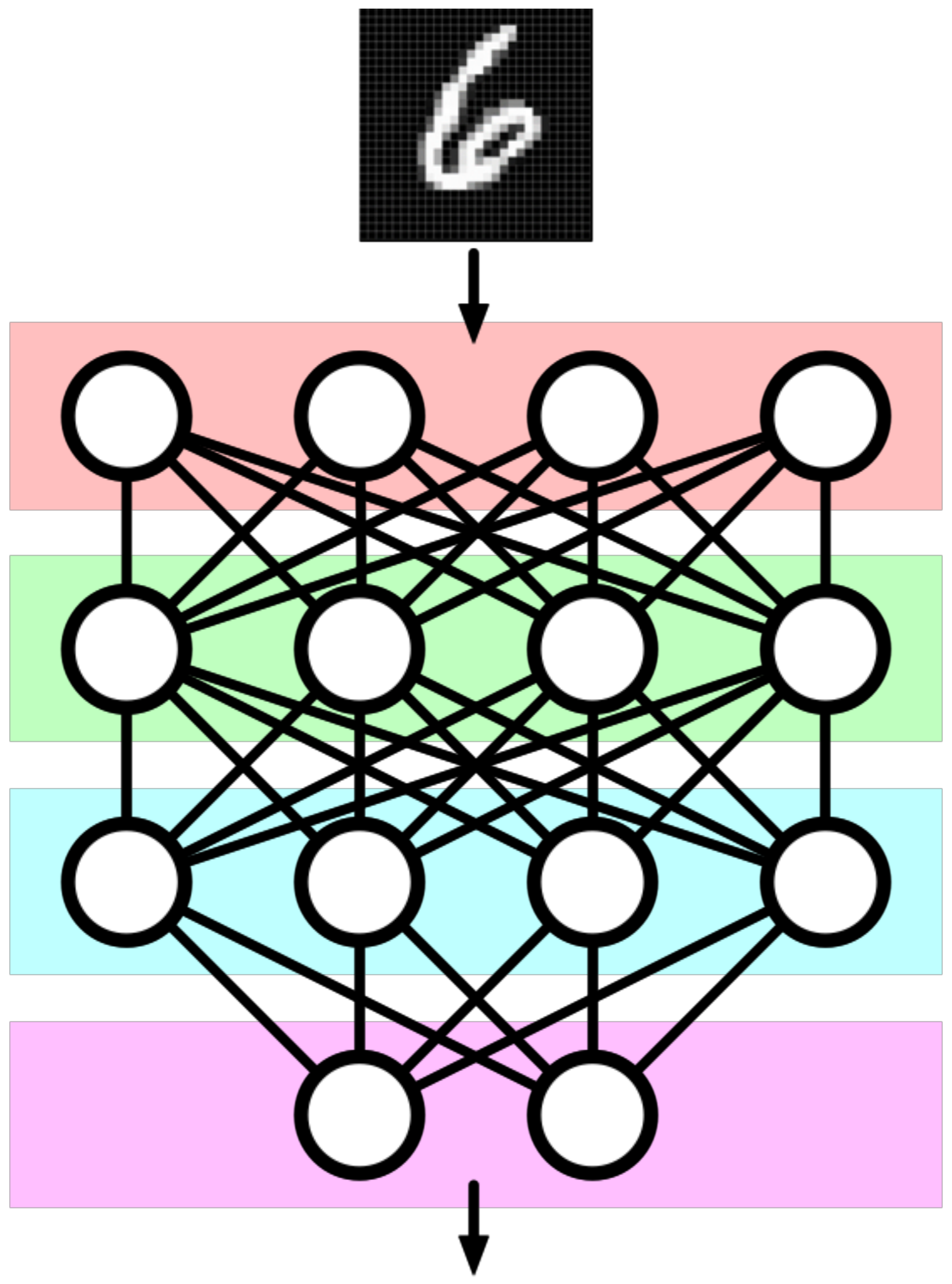
- Parallelization paradigms have different communication patterns
- Lack of good network abstraction

Example: pipeline parallelism

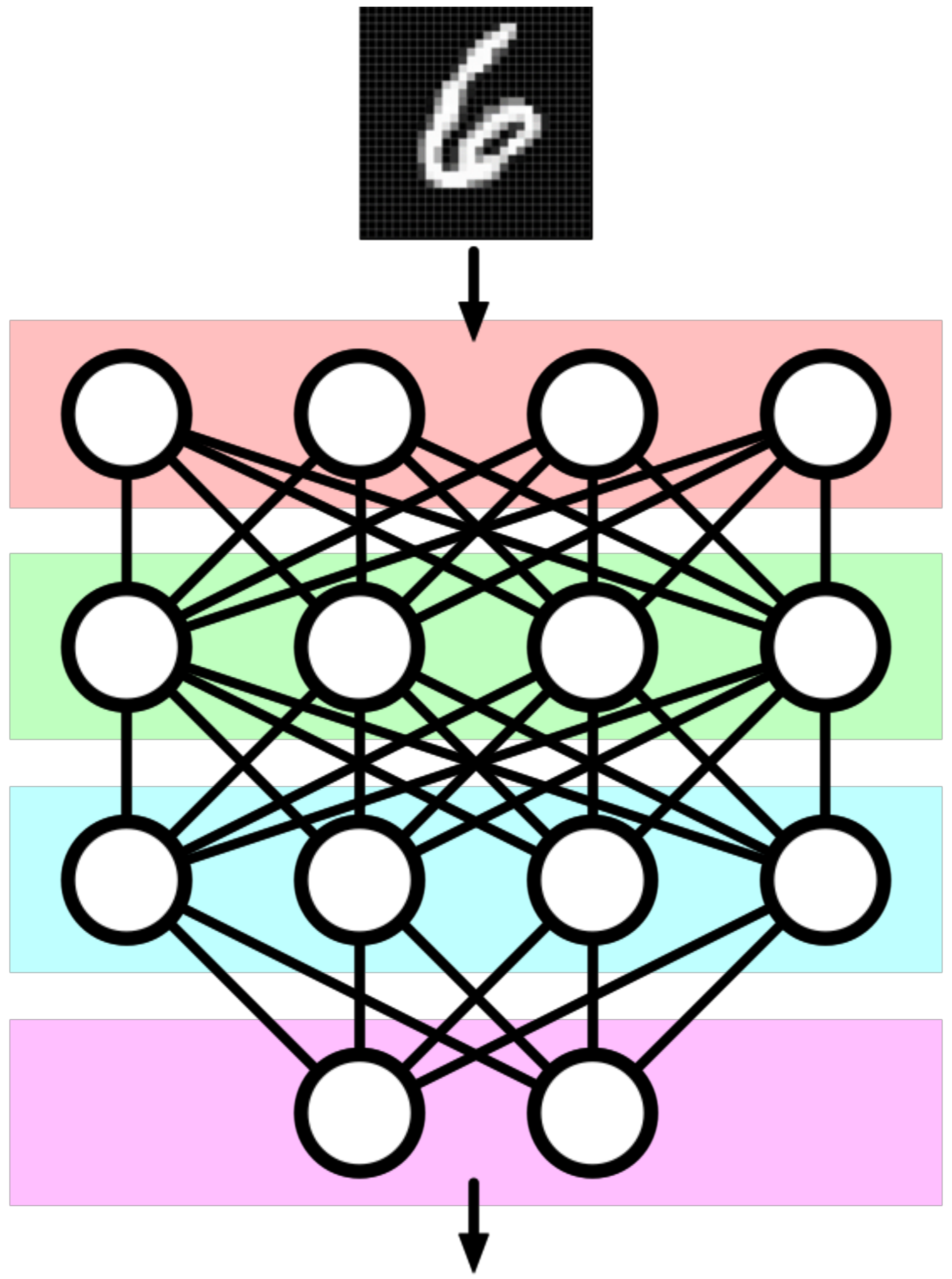
Example: pipeline parallelism

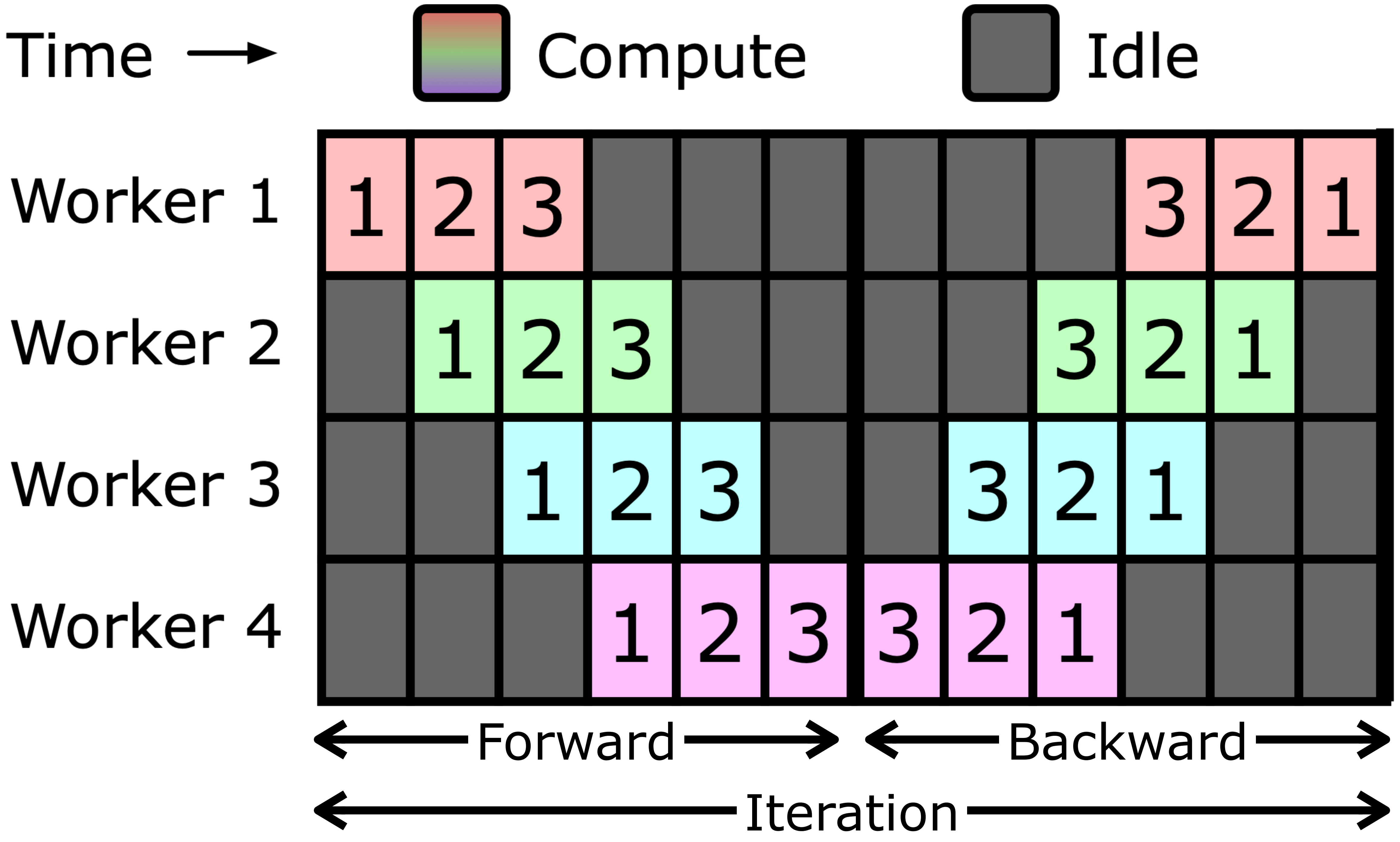


Example: pipeline parallelism

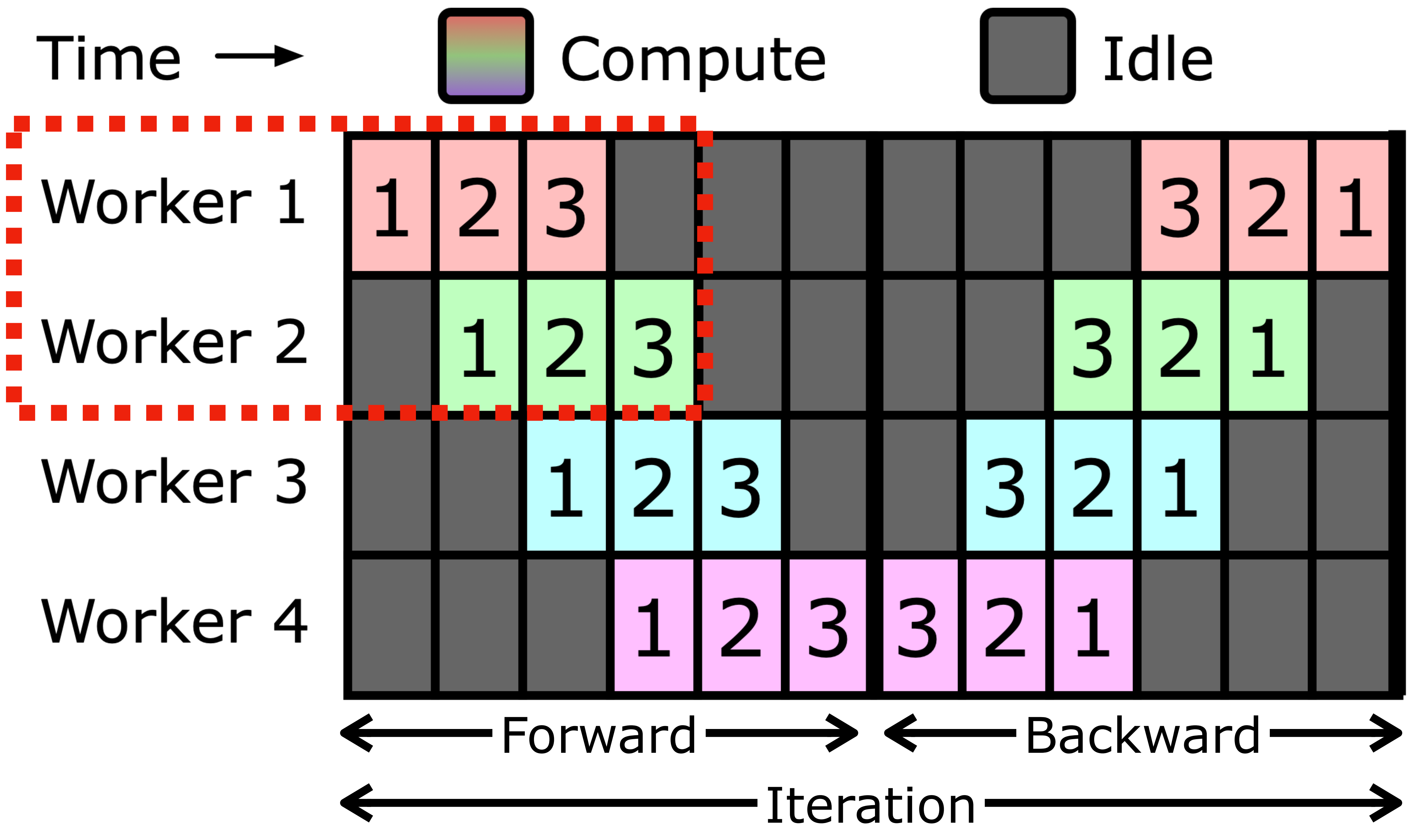


Example: pipeline parallelism

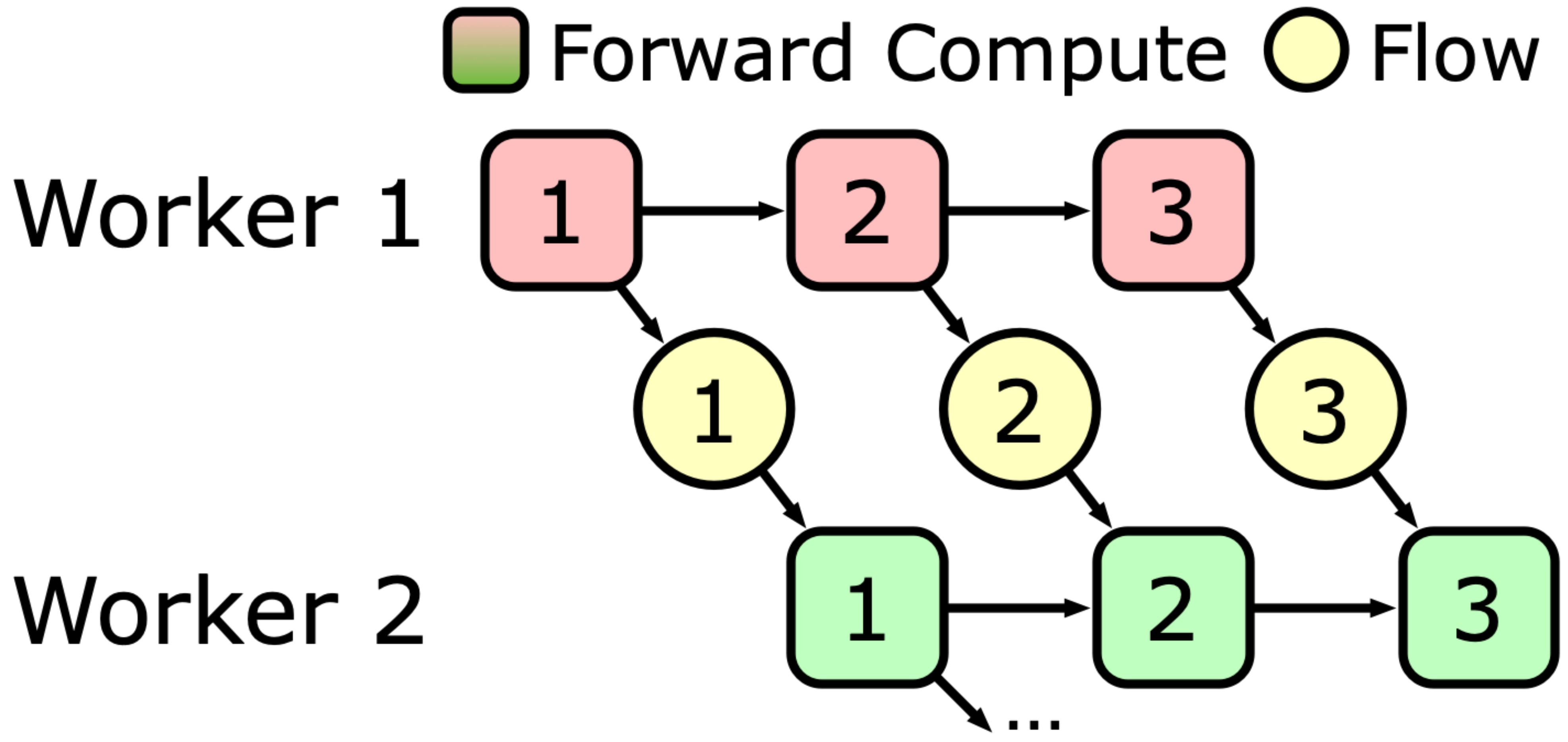


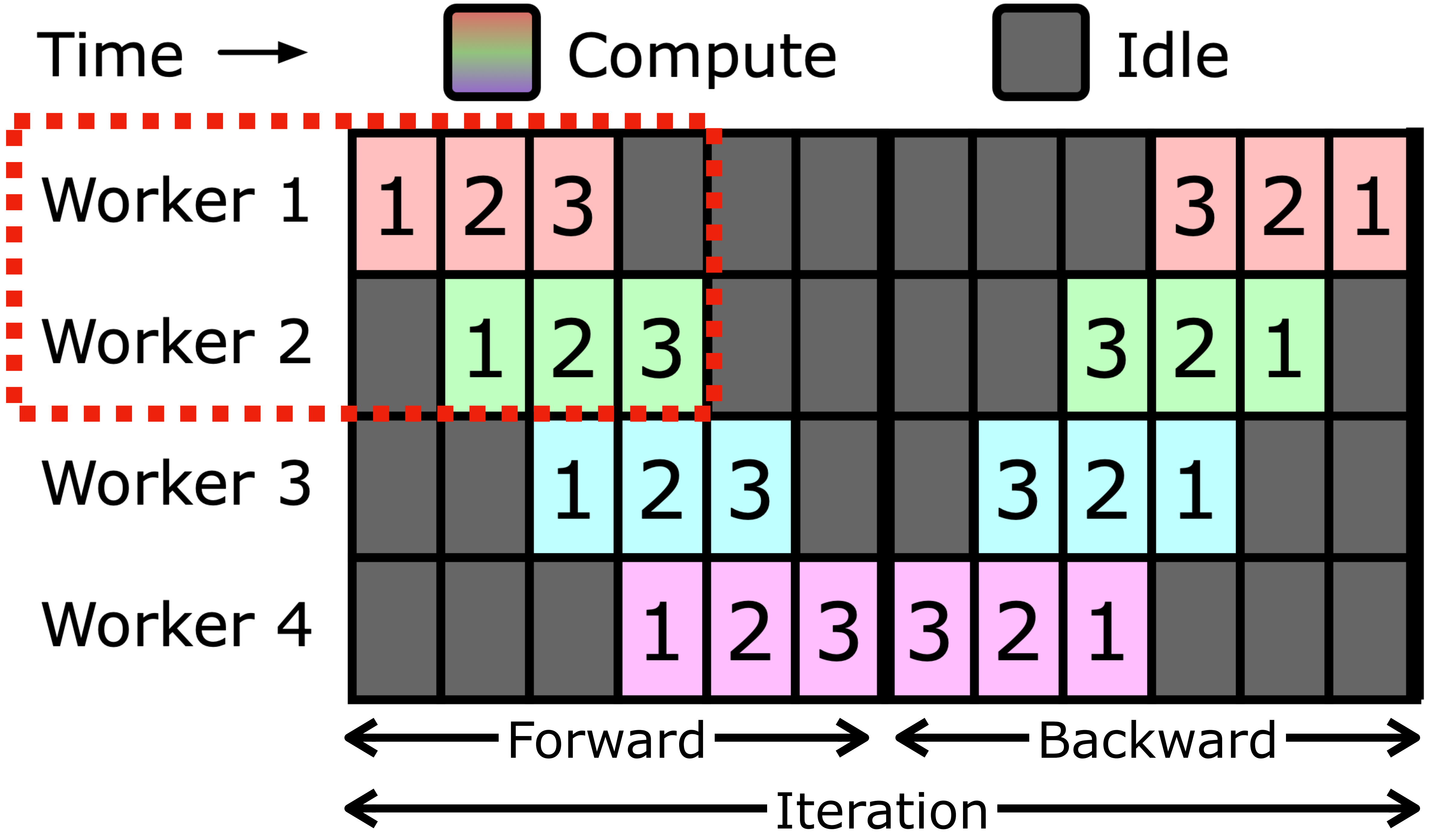


Example: pipeline parallelism

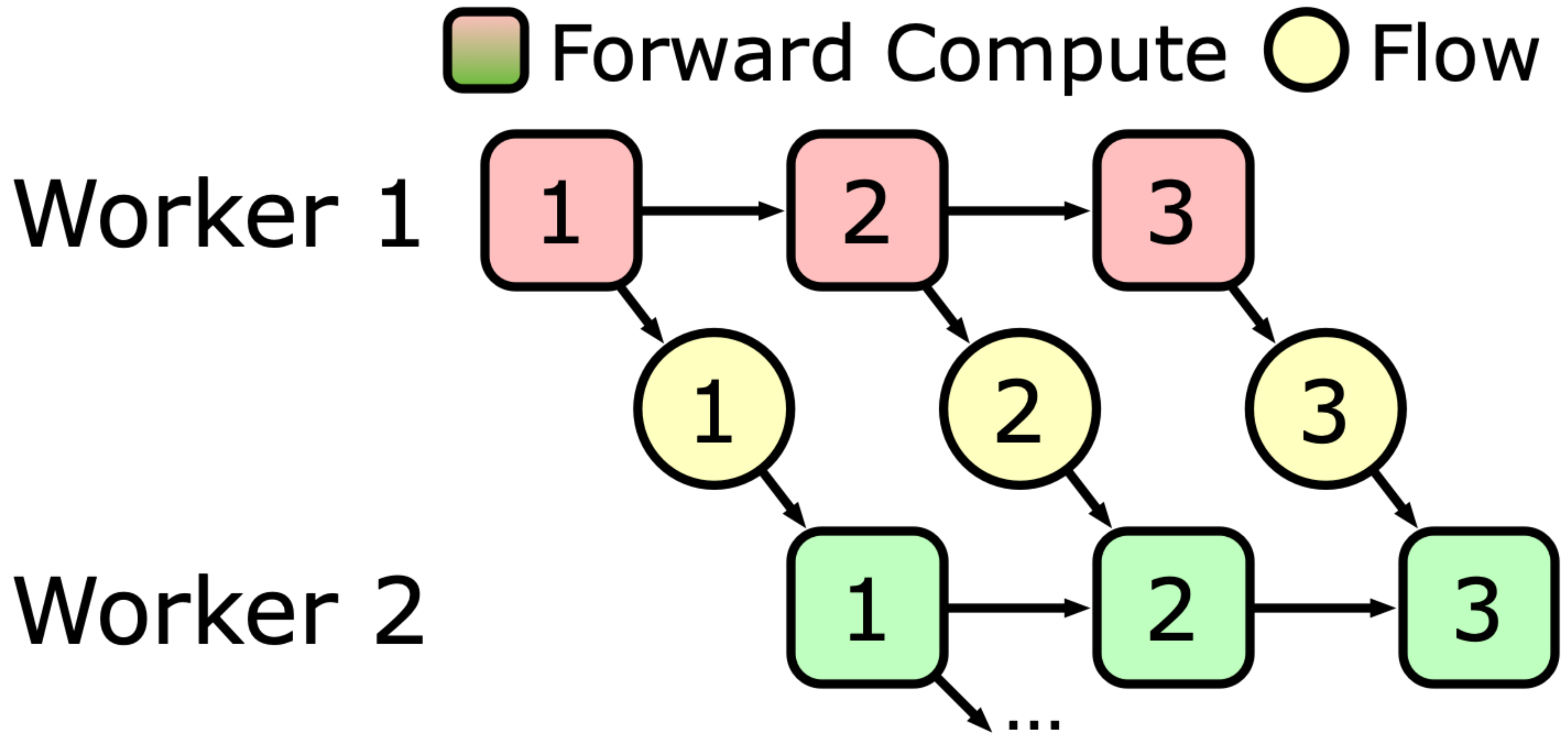


Example: pipeline parallelism

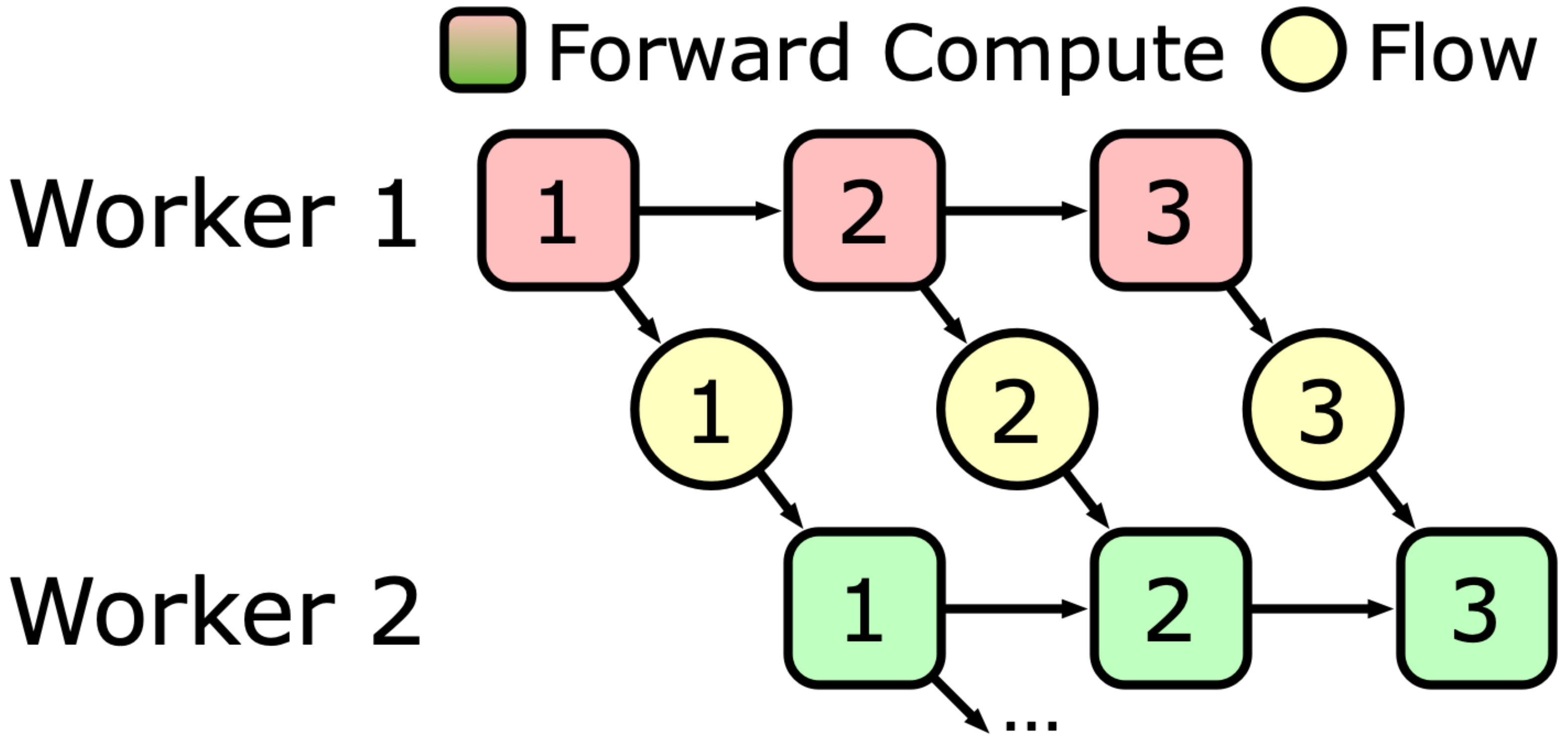




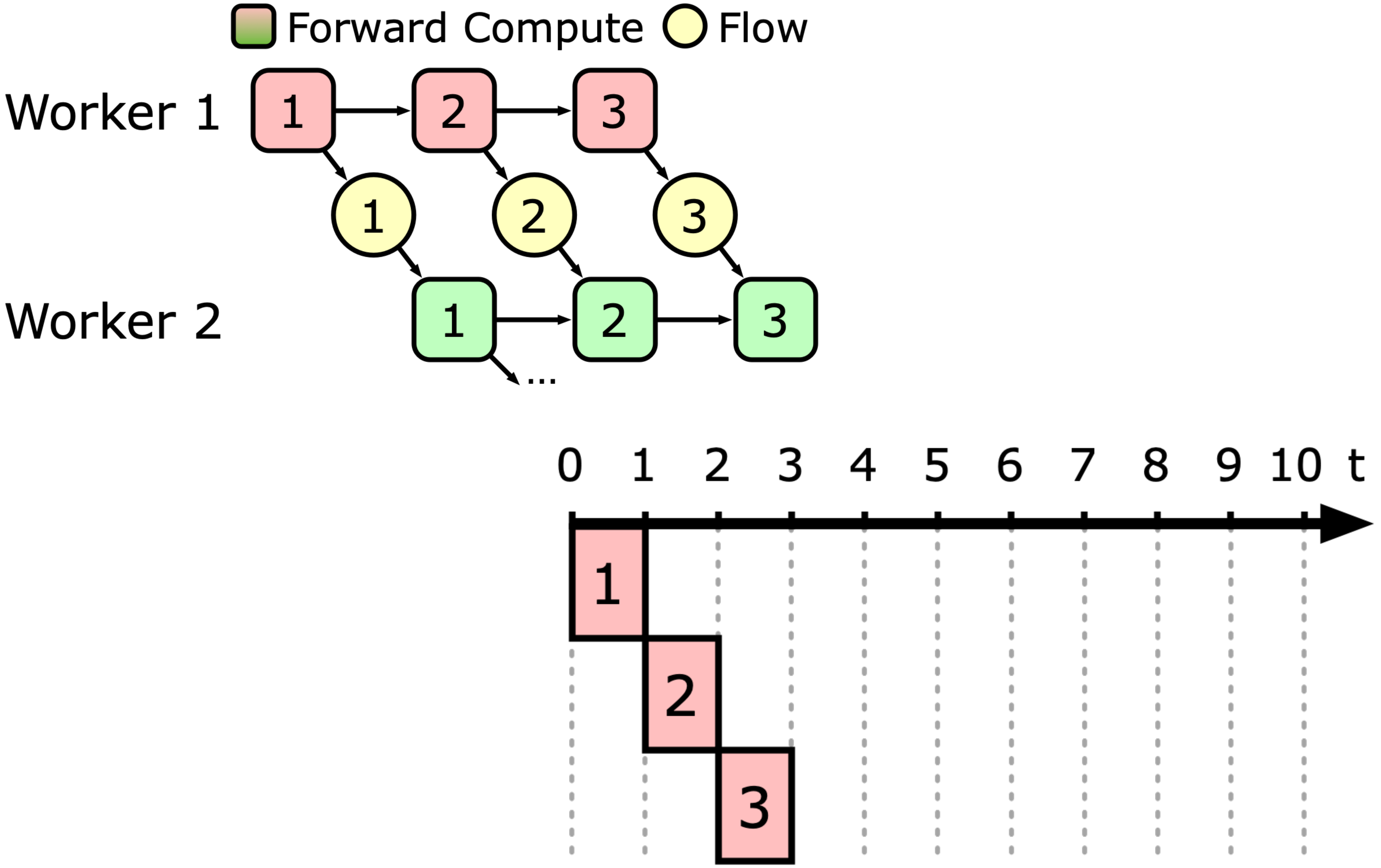
Example: pipeline parallelism



Example: pipeline parallelism

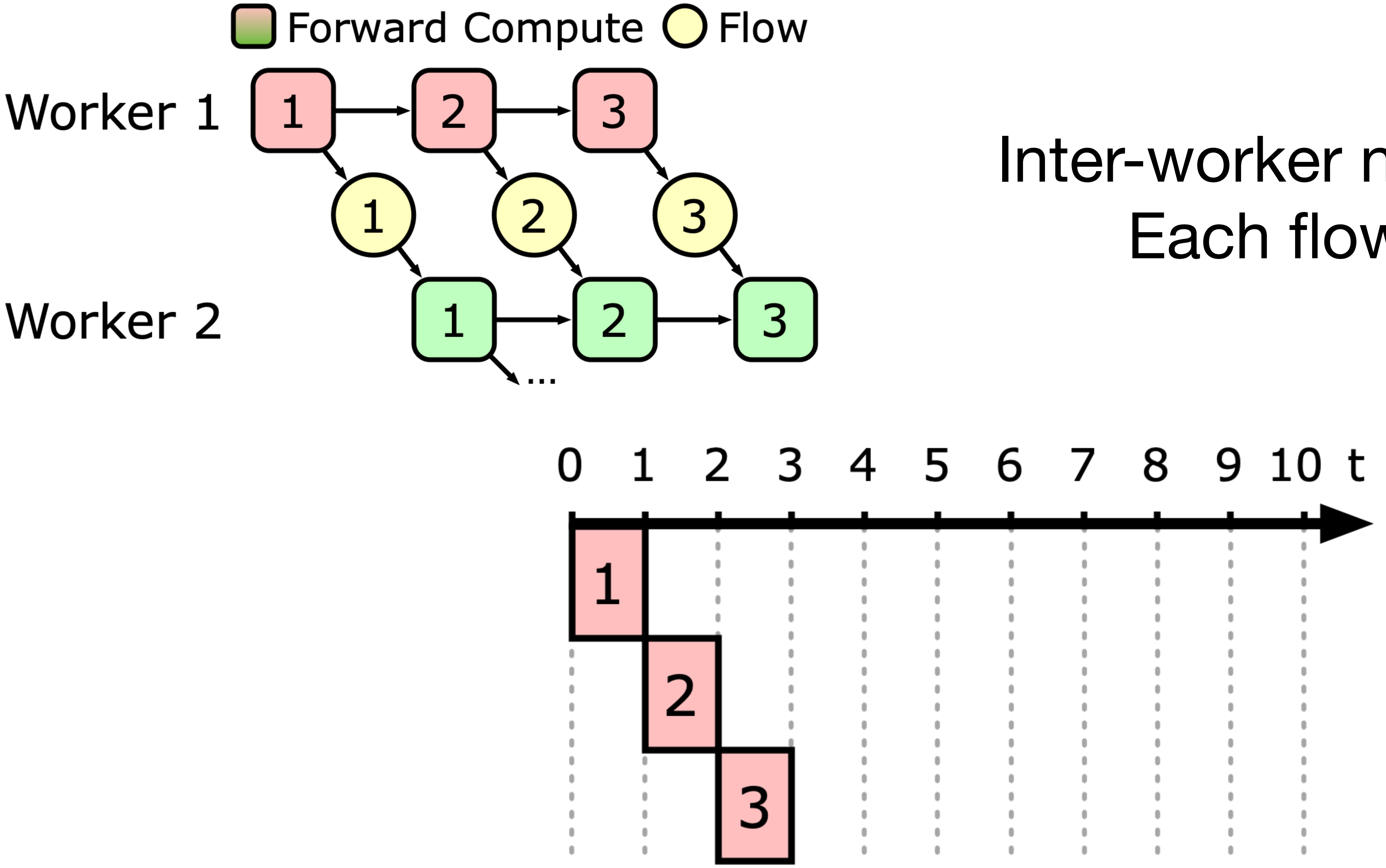


Example: pipeline parallelism



(b) Coflow Scheduling

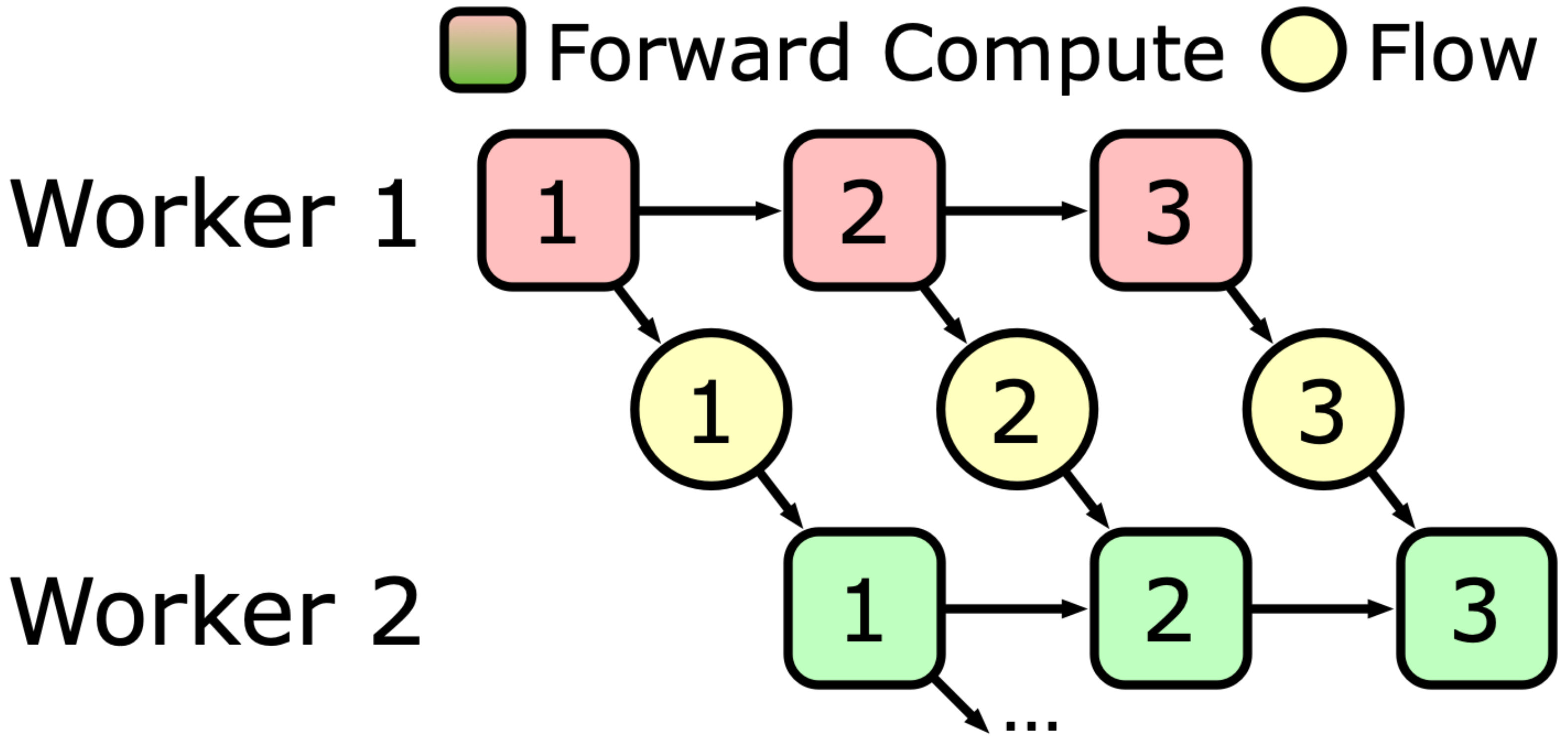
Example: pipeline parallelism



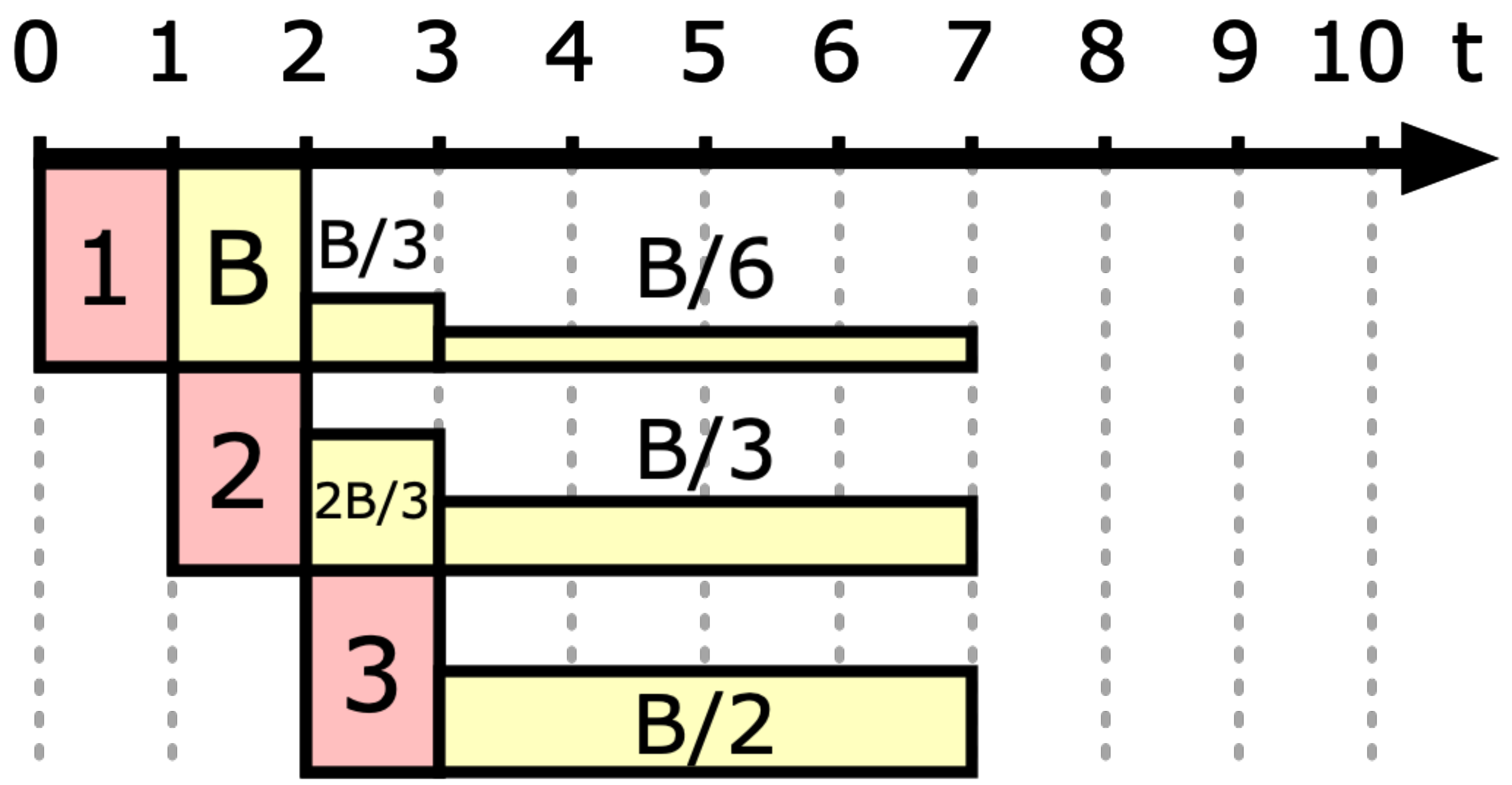
(b) Coflow Scheduling

Inter-worker network bandwidth: B
Each flow carries 2B data

Example: pipeline parallelism

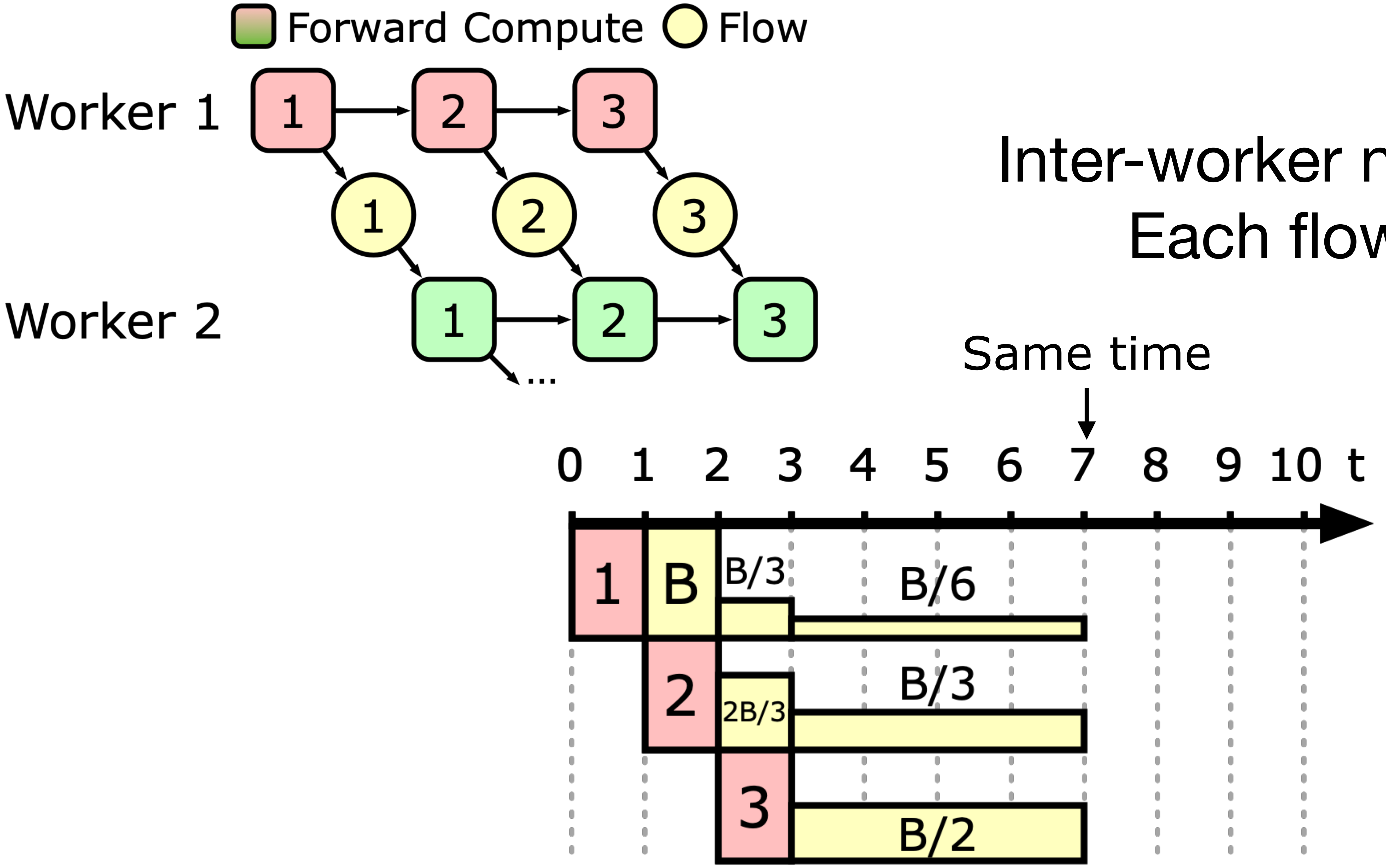


Inter-worker network bandwidth: B
Each flow carries 2B data



(b) Coflow Scheduling

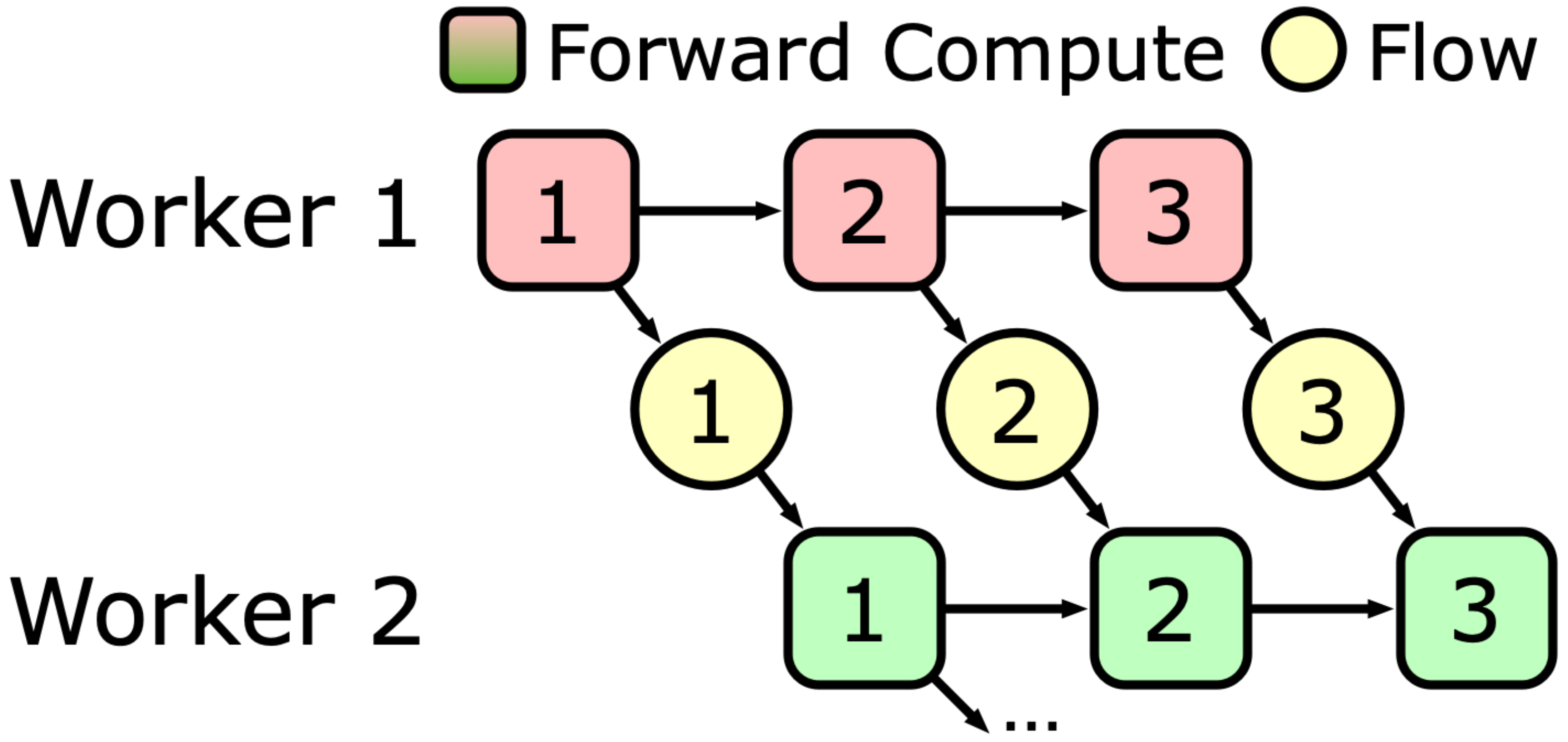
Example: pipeline parallelism



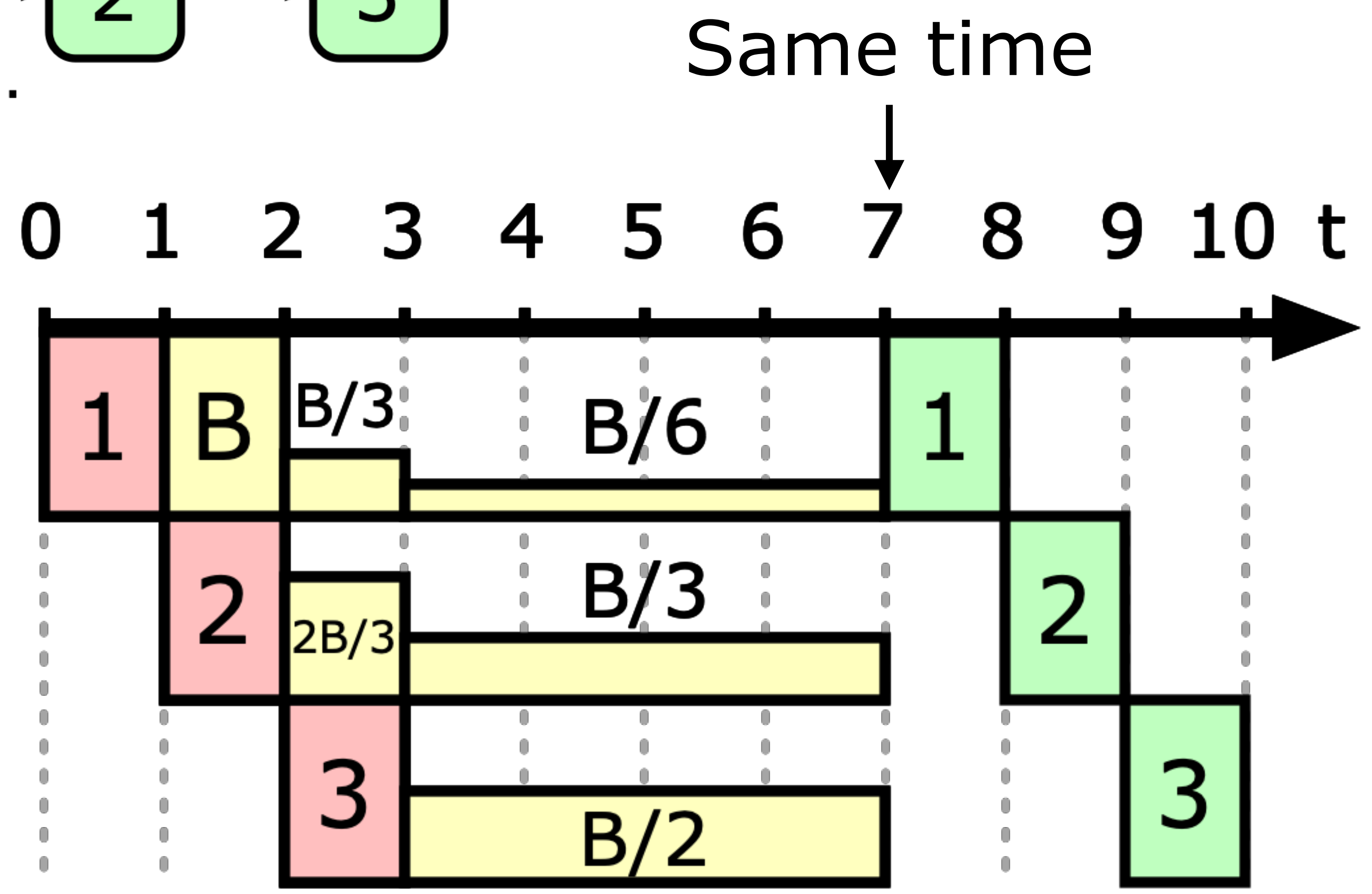
(b) Coflow Scheduling

Inter-worker network bandwidth: B
Each flow carries 2B data

Example: pipeline parallelism

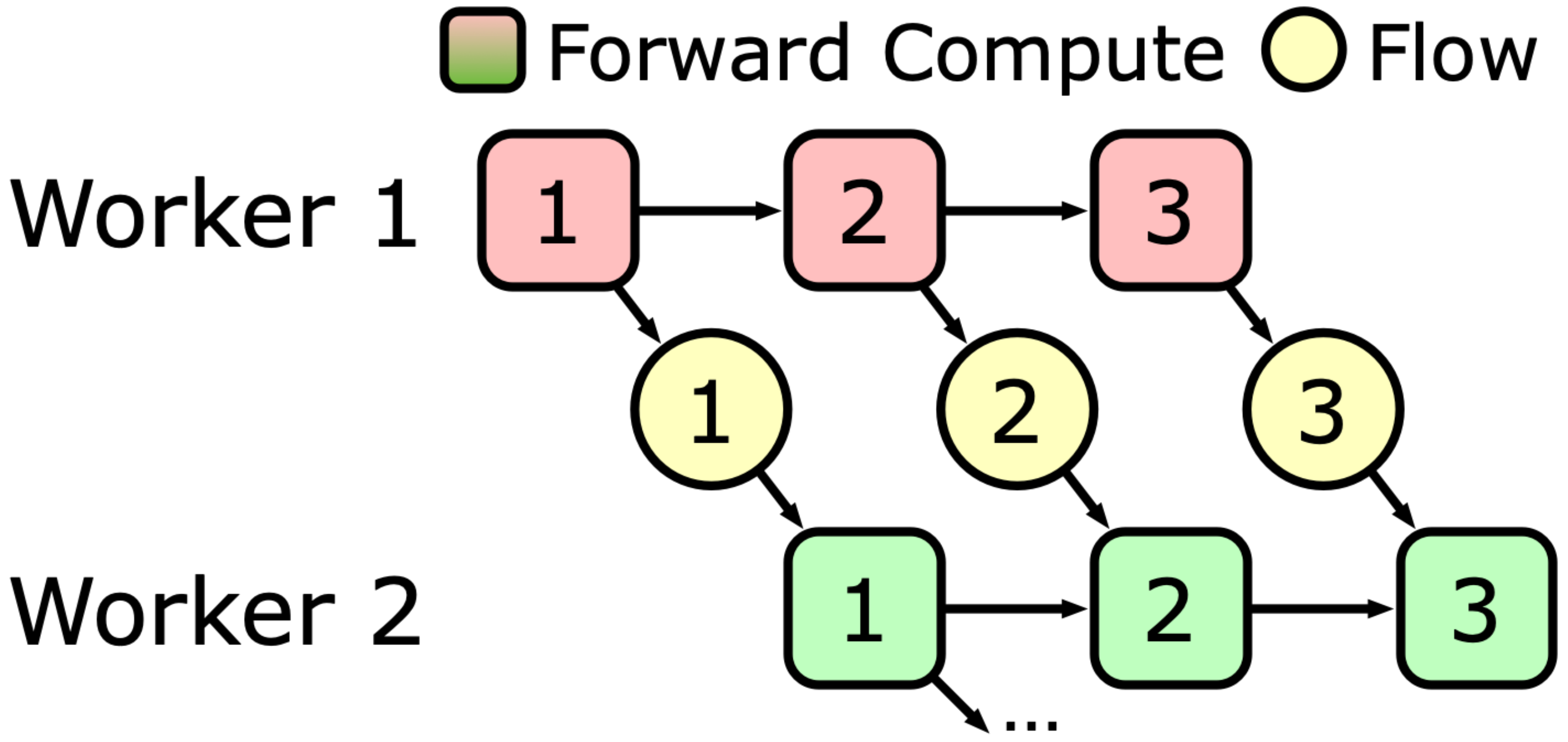


Inter-worker network bandwidth: B
Each flow carries 2B data

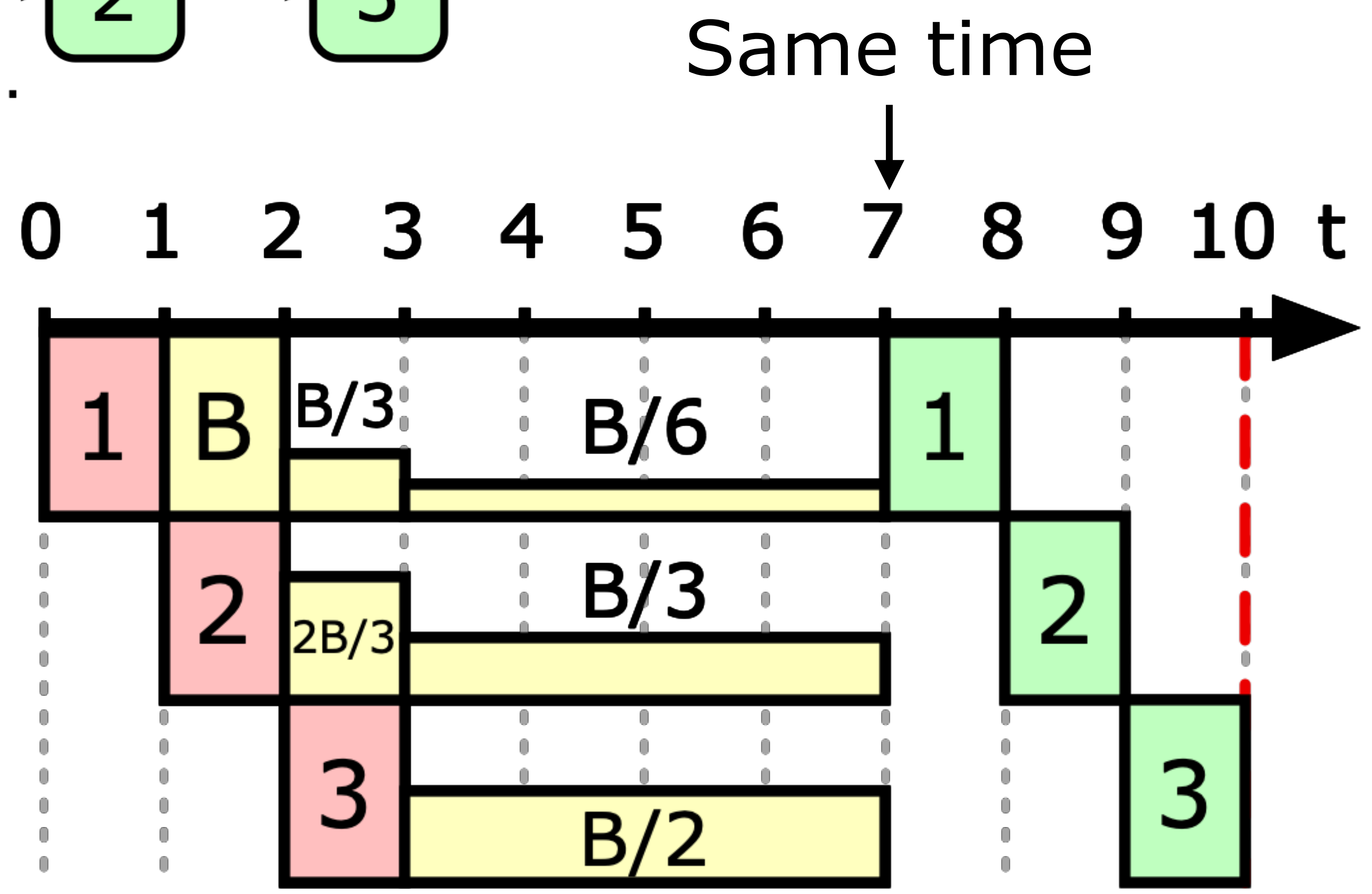


(b) Coflow Scheduling

Example: pipeline parallelism

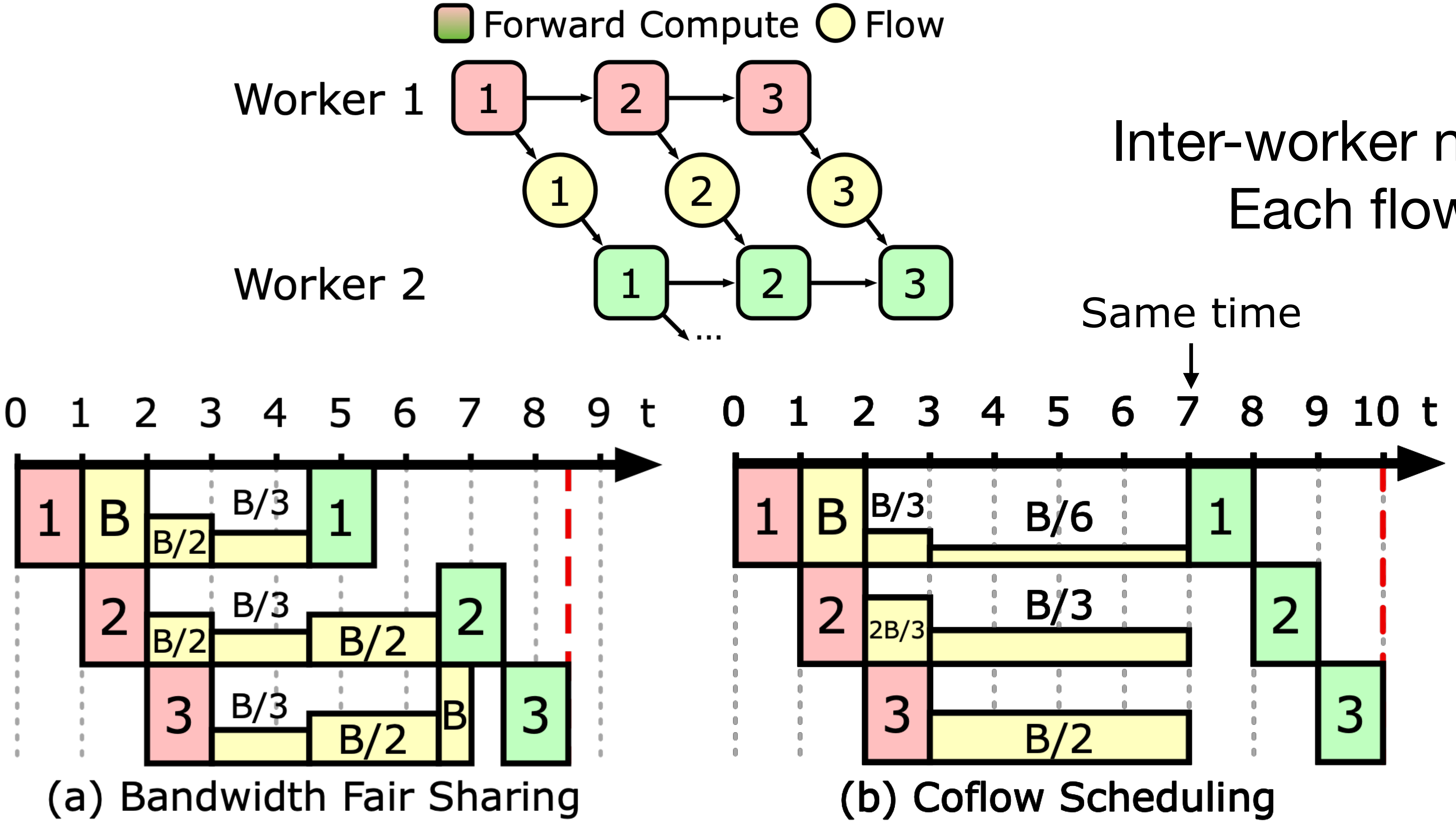


Inter-worker network bandwidth: B
Each flow carries 2B data



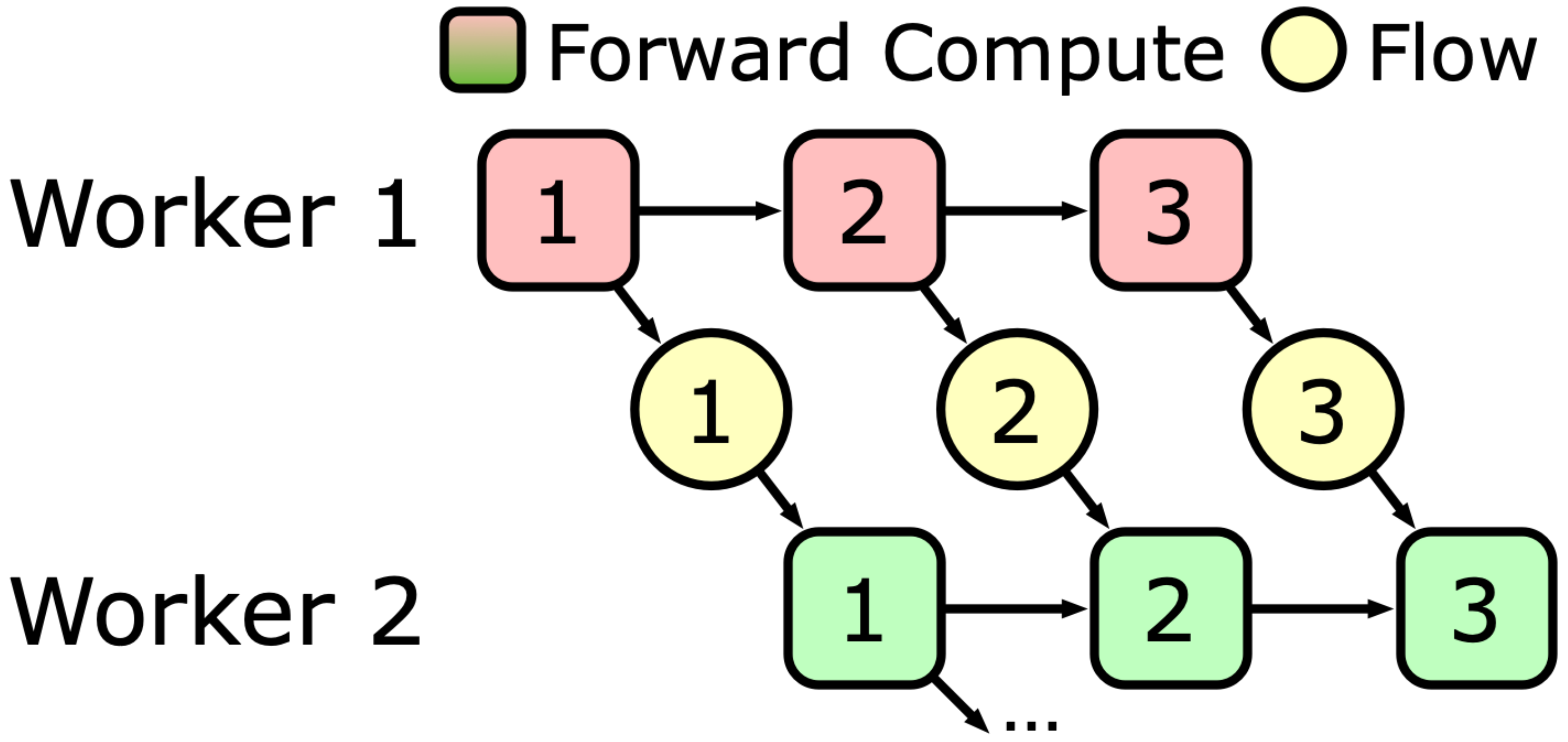
(b) Coflow Scheduling

Example: pipeline parallelism

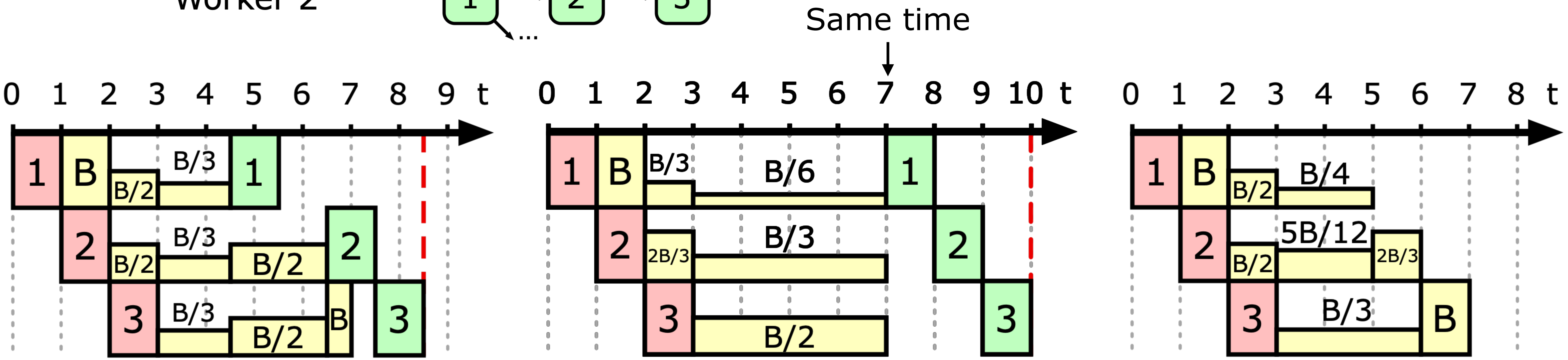


Inter-worker network bandwidth: B
Each flow carries 2B data

Example: pipeline parallelism

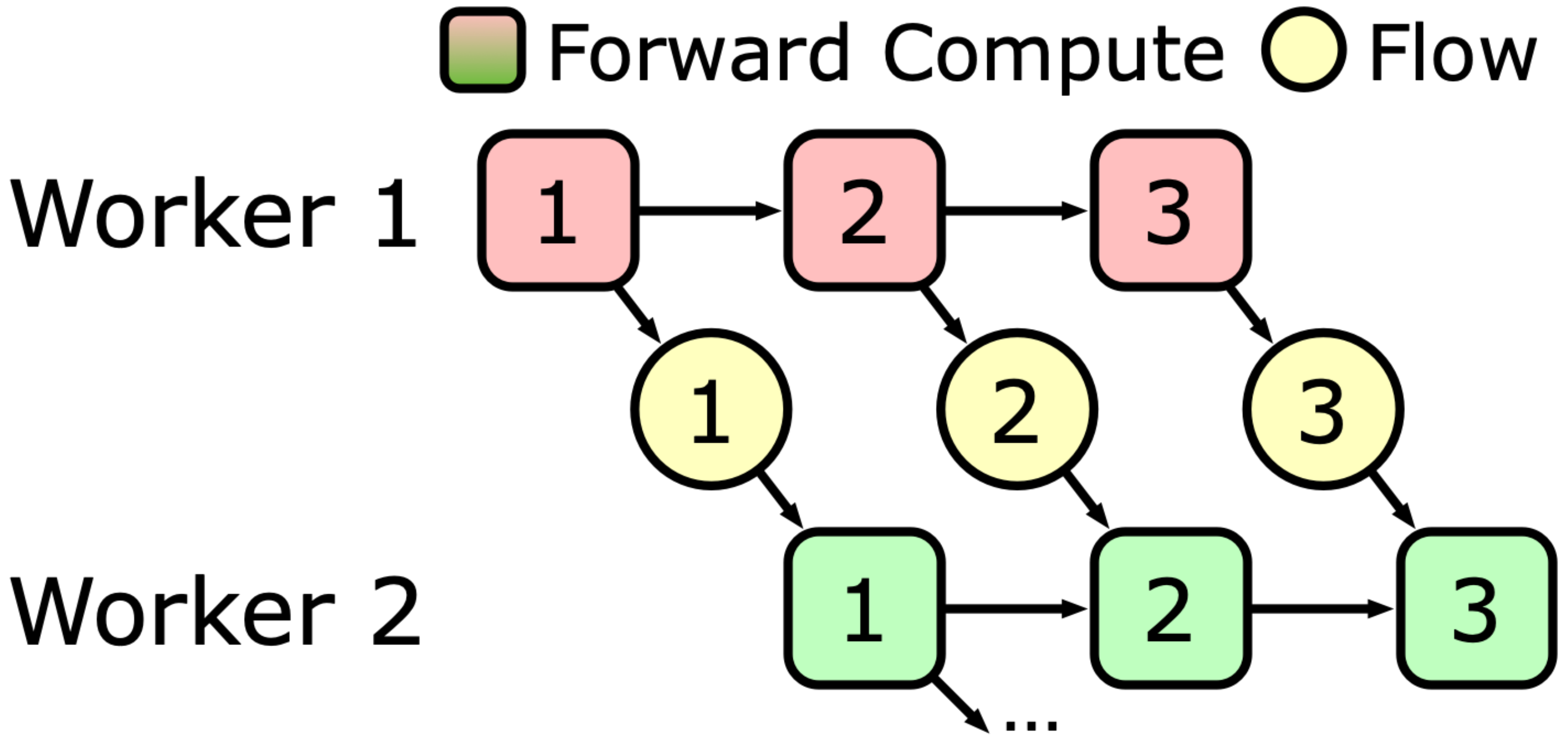


Inter-worker network bandwidth: B
Each flow carries 2B data

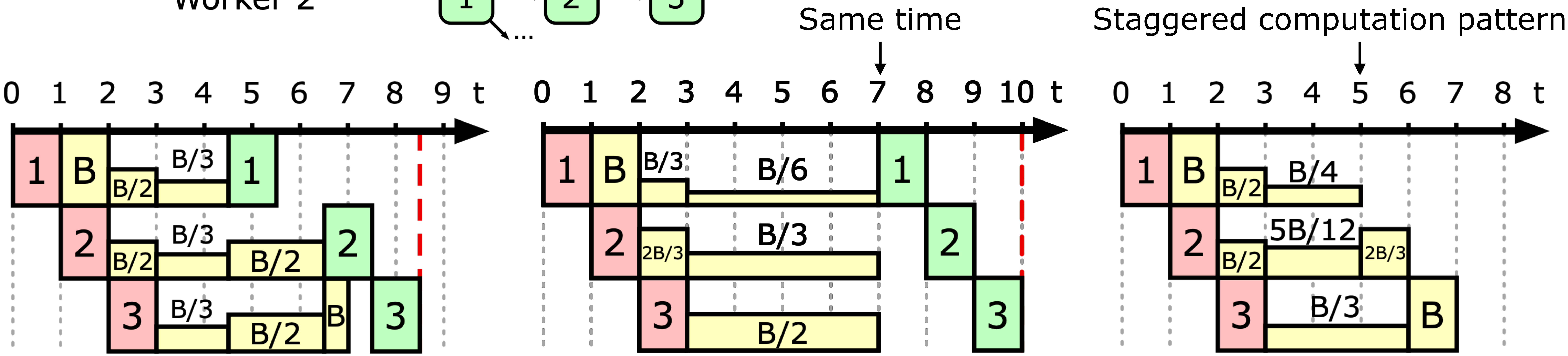


(a) Bandwidth Fair Sharing (b) Coflow Scheduling (c) EchelonFlow Scheduling

Example: pipeline parallelism



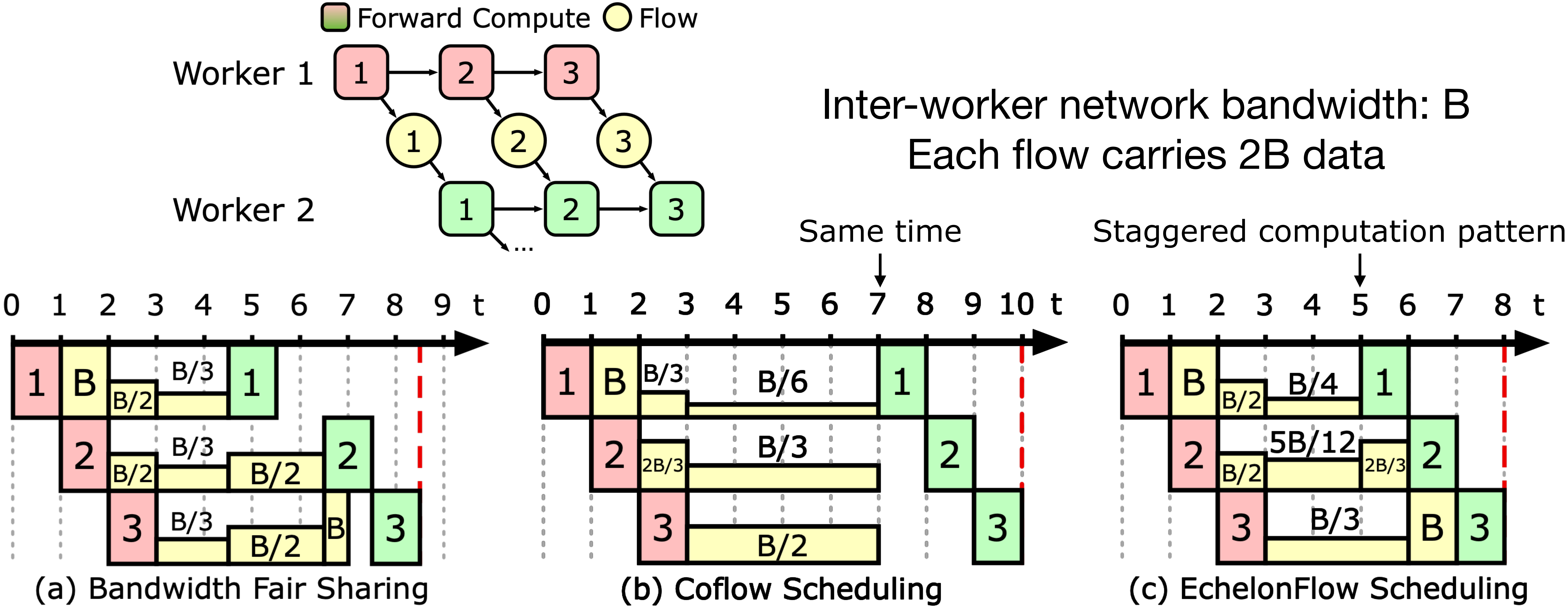
Inter-worker network bandwidth: B
Each flow carries 2B data



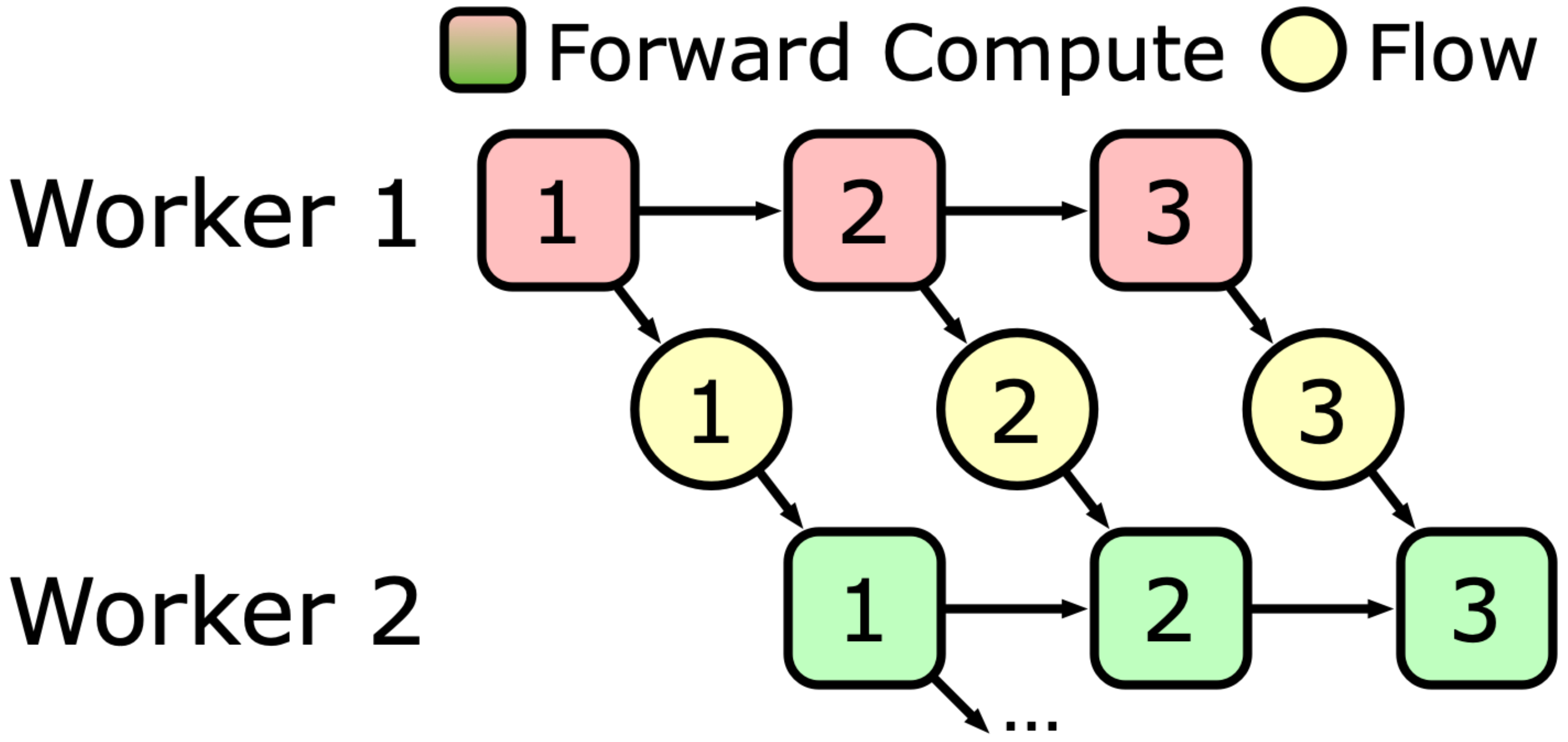
(a) Bandwidth Fair Sharing (b) Coflow Scheduling (c) EchelonFlow Scheduling

Example: pipeline parallelism

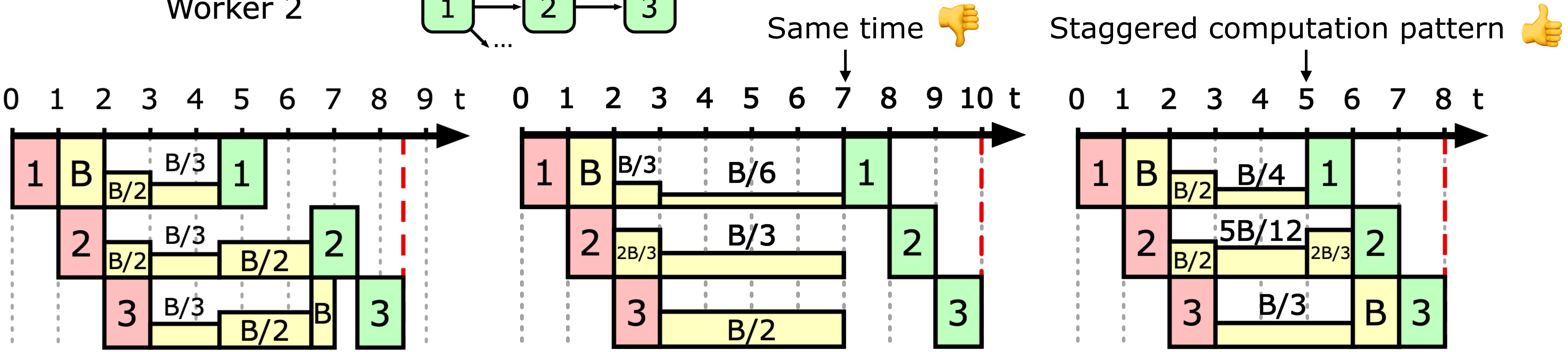
Inter-worker network bandwidth: B
Each flow carries 2B data



Example: pipeline parallelism



Inter-worker network bandwidth: B
Each flow carries 2B data



(a) Bandwidth Fair Sharing (b) Coflow Scheduling (c) EchelonFlow Scheduling

EchelonFlow insight

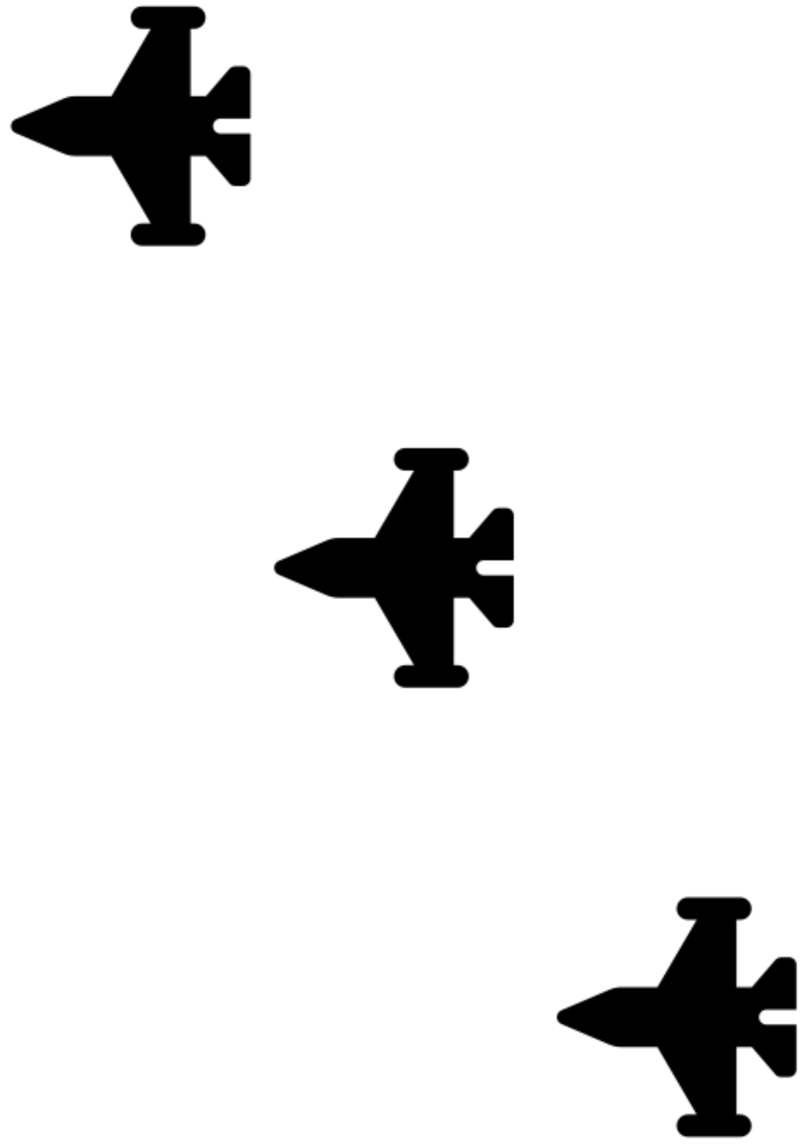
Flow finish times → computation pattern

EchelonFlow insight

Flow finish times → computation pattern

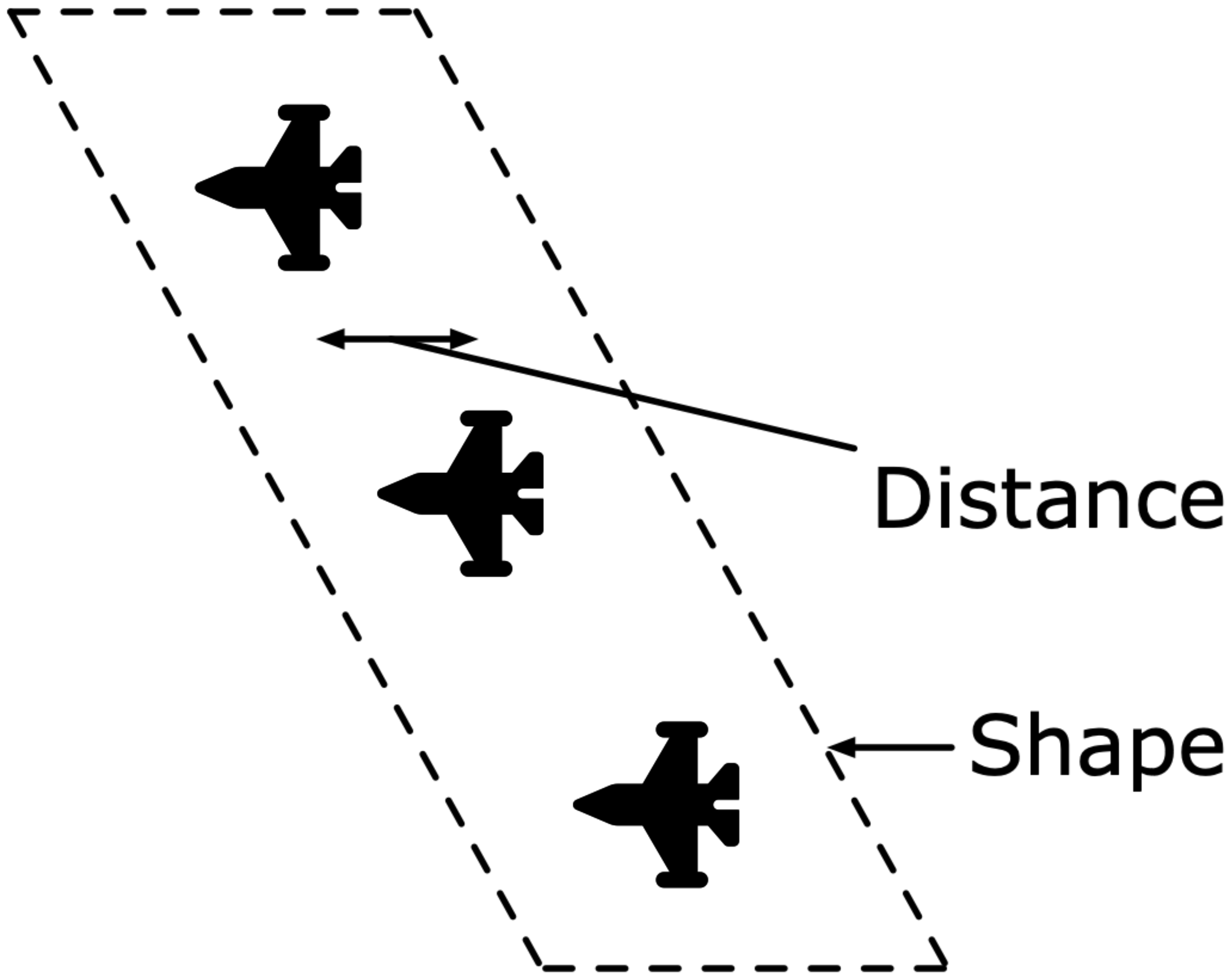
EchelonFlow insight

Flow finish times → computation pattern



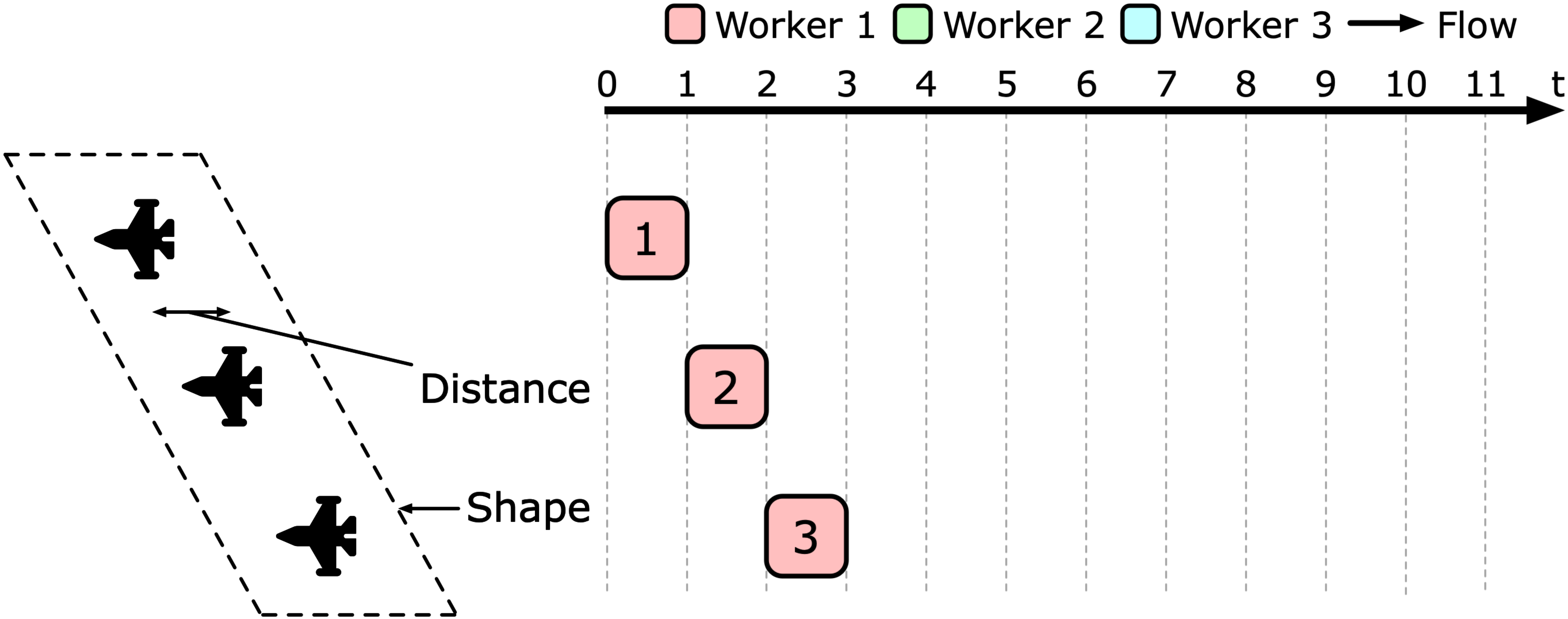
EchelonFlow insight

Flow finish times → computation pattern



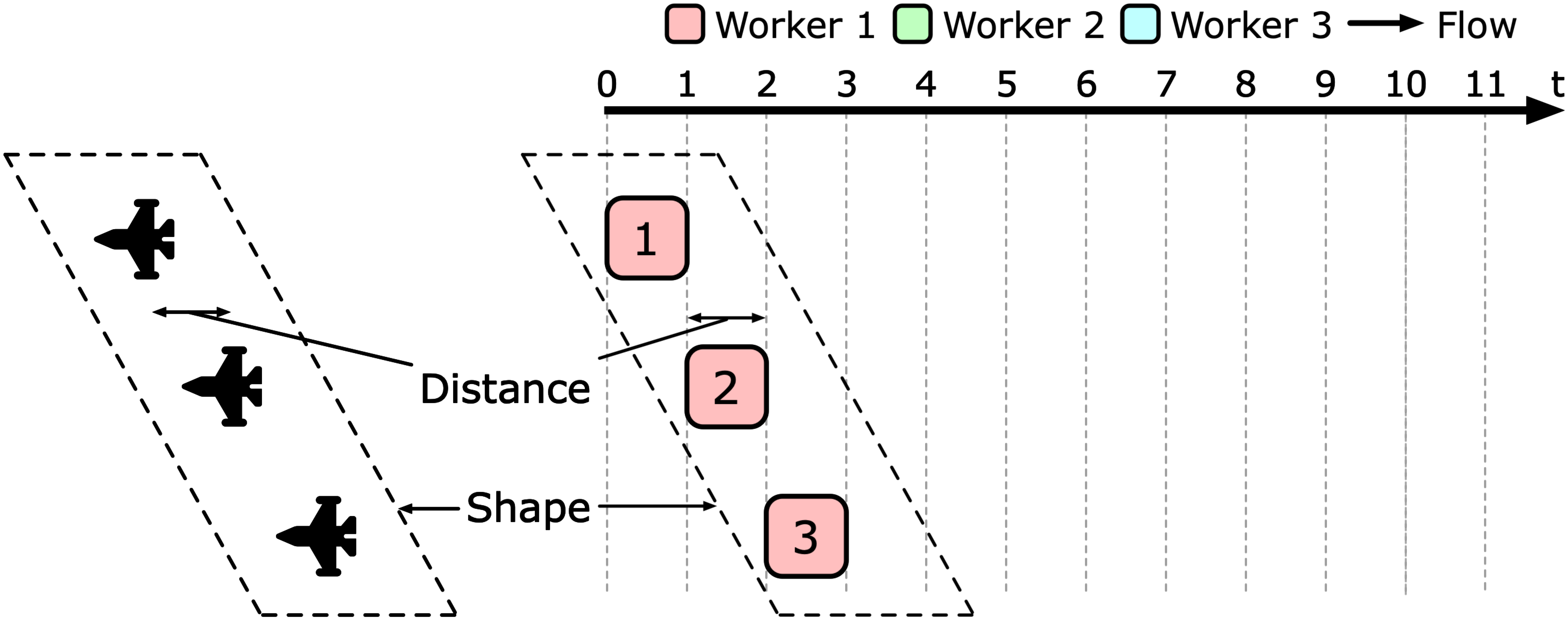
EchelonFlow insight

Flow finish times → computation pattern



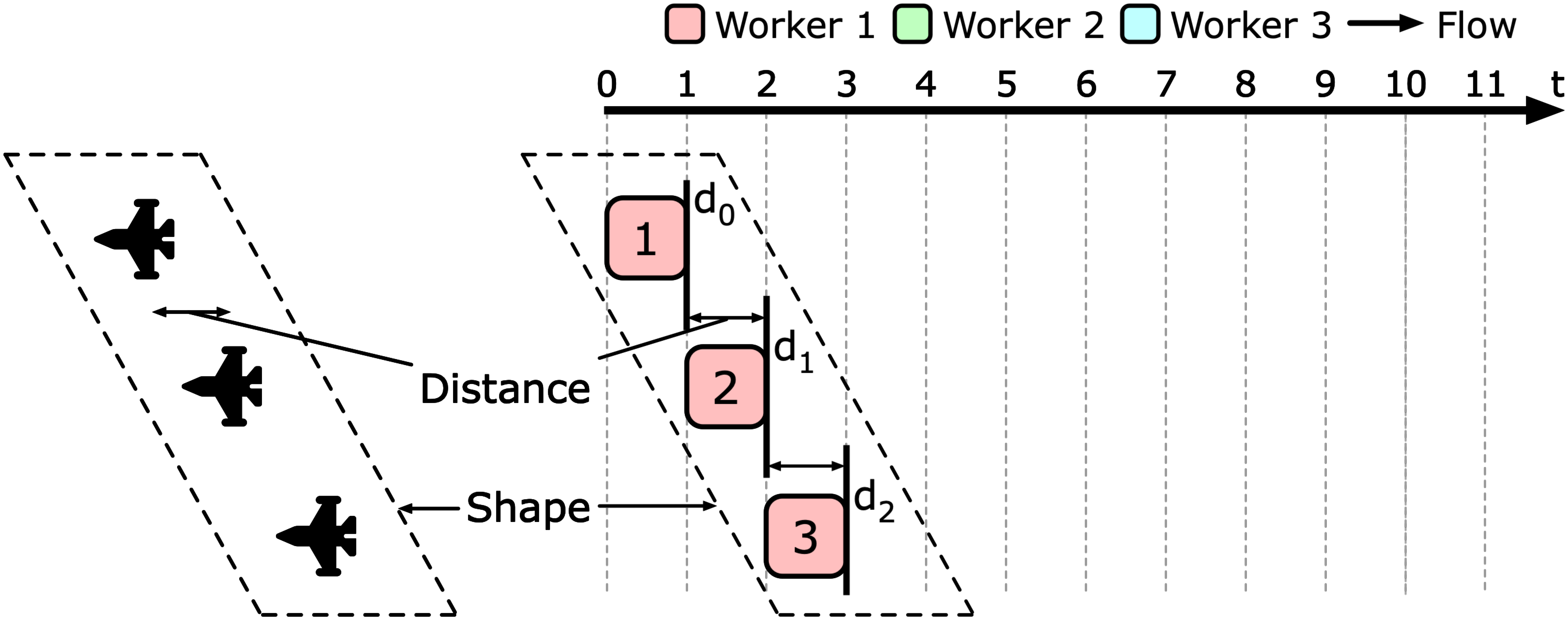
EchelonFlow insight

Flow finish times → computation pattern



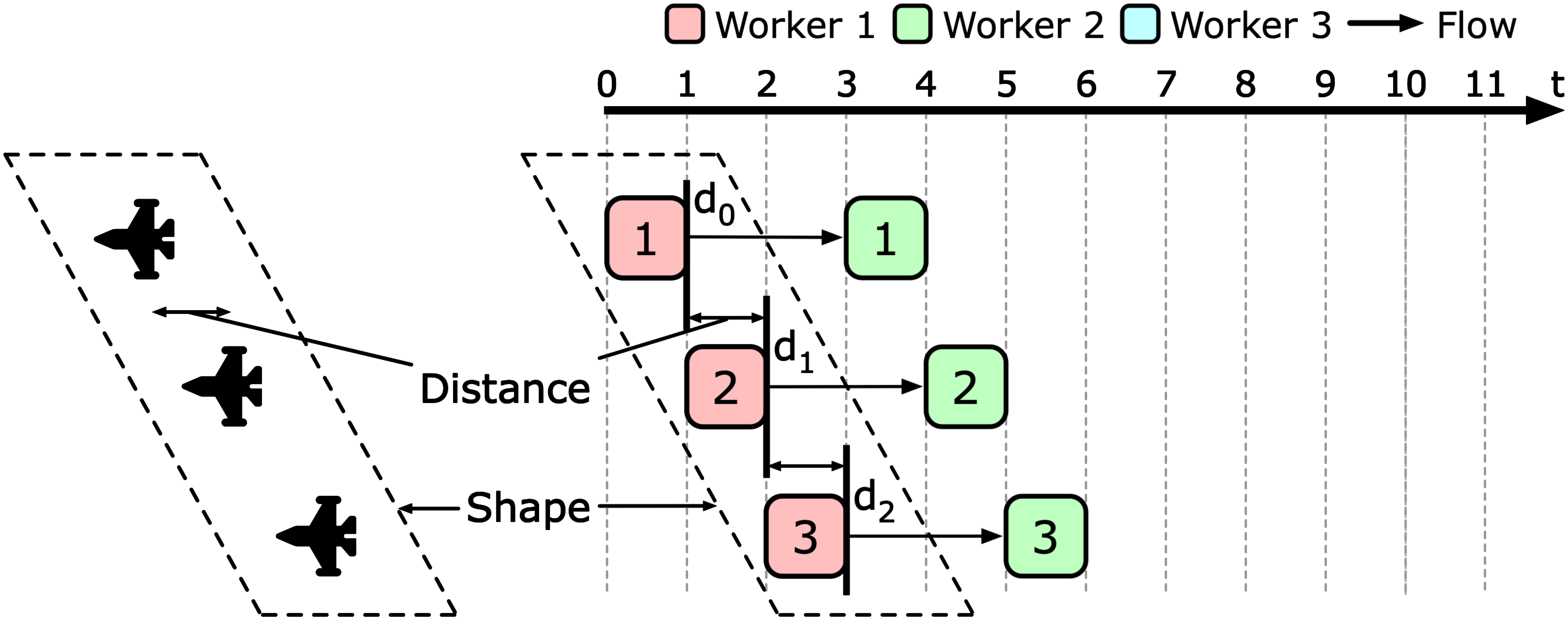
EchelonFlow insight

Flow finish times → computation pattern



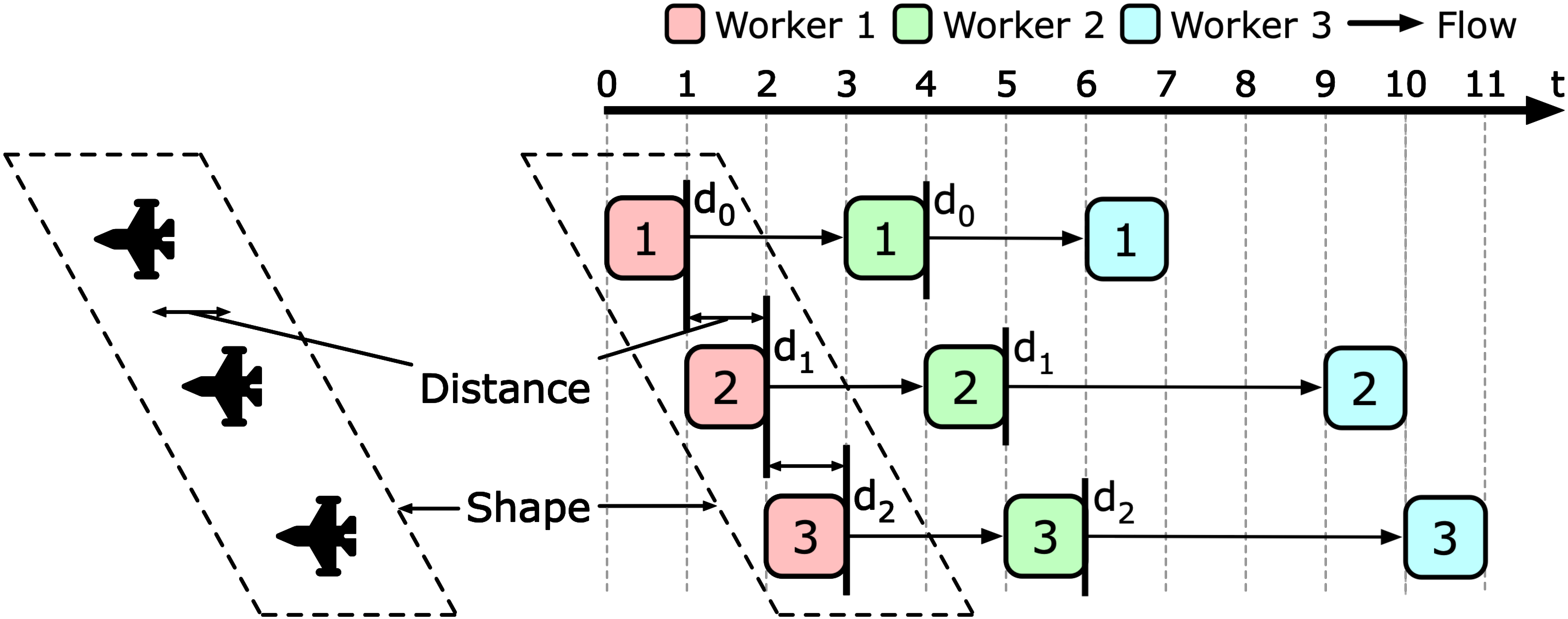
EchelonFlow insight

Flow finish times → computation pattern



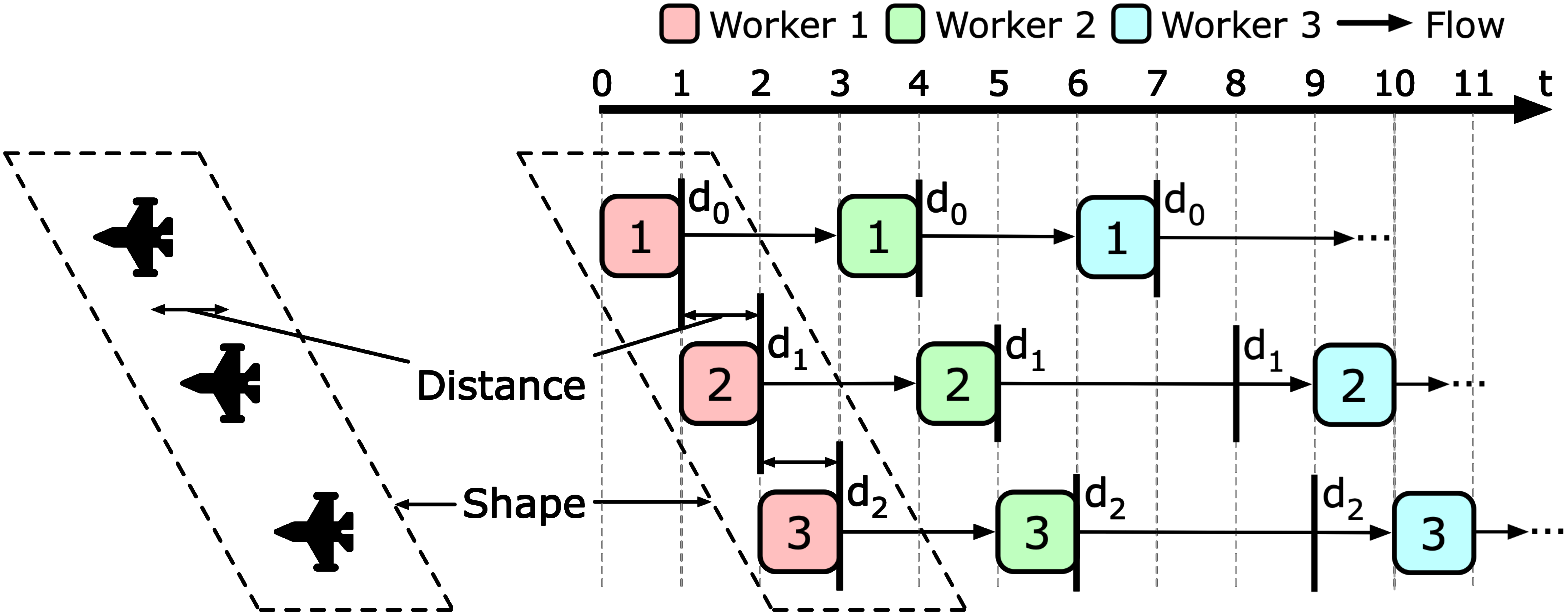
EchelonFlow insight

Flow finish times → computation pattern



EchelonFlow insight

Flow finish times → computation pattern



EchelonFlow

Definition

EchelonFlow

Definition

- EchelonFlow: a set of flows whose *ideal finish times* are related

EchelonFlow

Definition

- EchelonFlow: a set of flows whose *ideal finish times* are related
    - Not necessarily equal

EchelonFlow

Definition

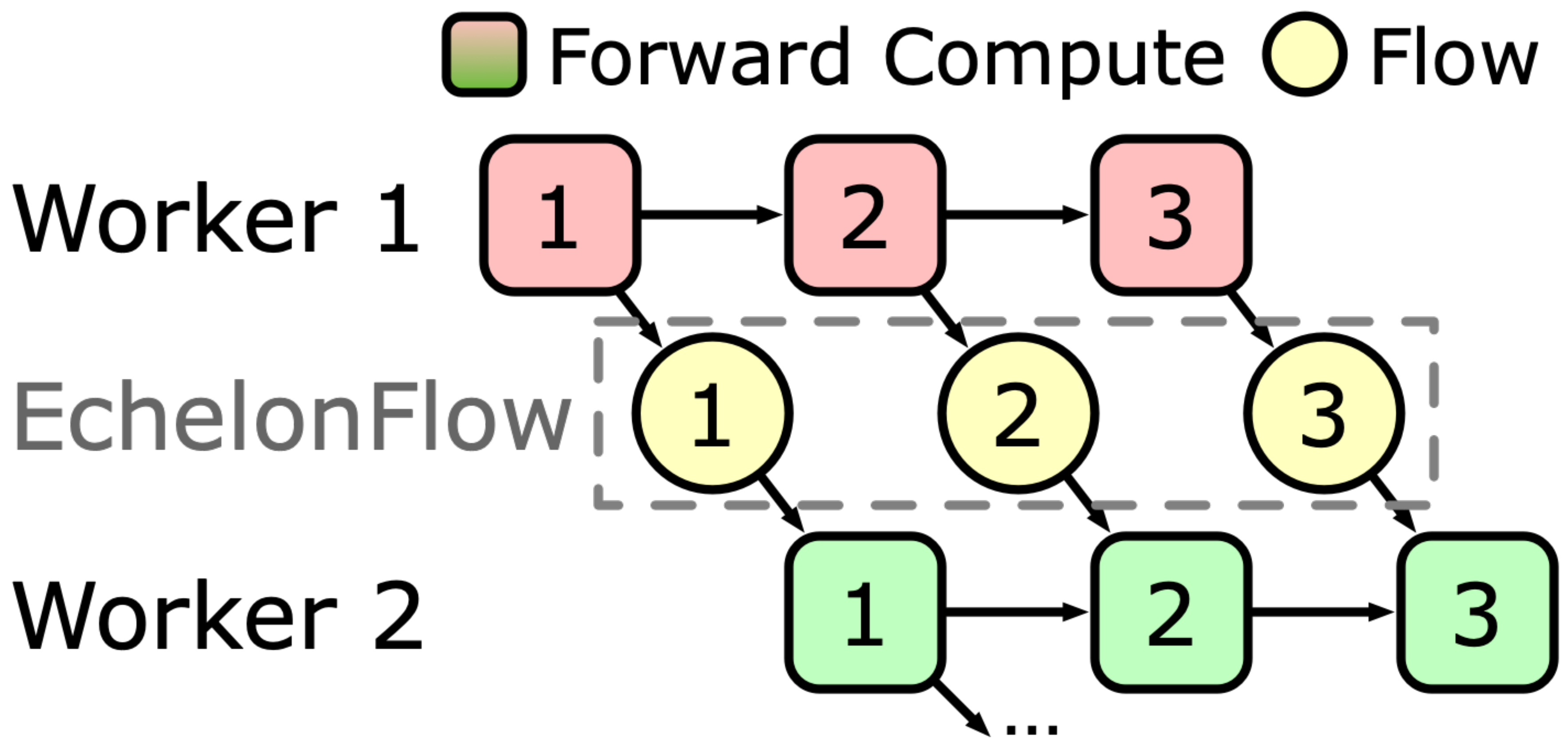
- EchelonFlow: a set of flows whose *ideal finish times* are related
  - Not necessarily equal
  - Can be represented by an **arrangement function**

EchelonFlow

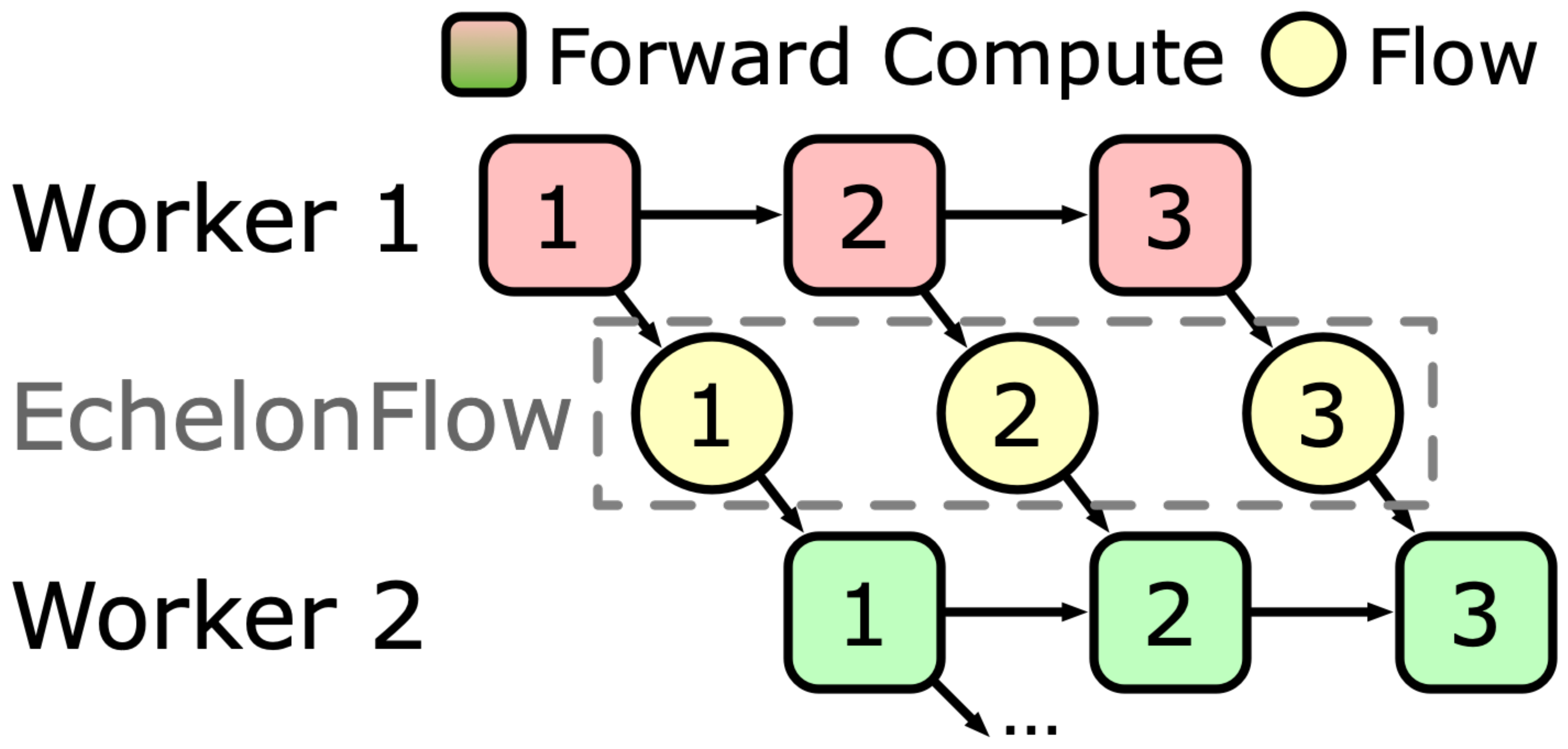
Expressiveness

Training paradigm	Examples	CoFlow compliance	EchelonFlow compliance
Data Parallelism	AllReduce, Parameter Server	✅	✅
Pipeline Parallelism	GPipe, PipeDream	❌	✅
Tensor Parallelism	Megatron-LM	✅	✅
Fully Sharded Data Parallelism	ZeRO, FairScale	❌	✅

Case study: pipeline parallelism



Case study: pipeline parallelism



Arrangement function:

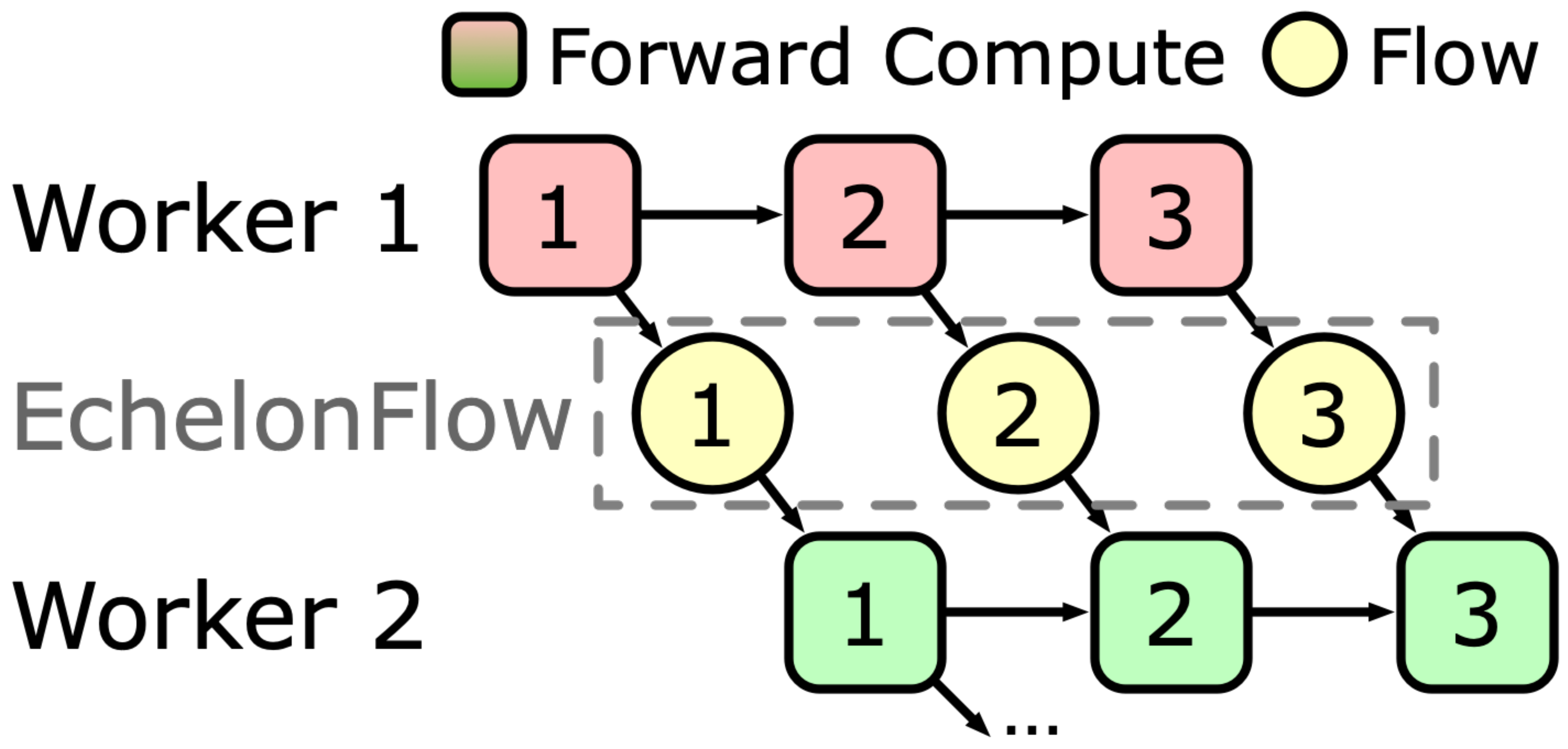
$$\boxed{d_j = d_{j-1} + T}$$

d_j: ideal finish time of flow j

T: time of one forward pass of one micro-batch

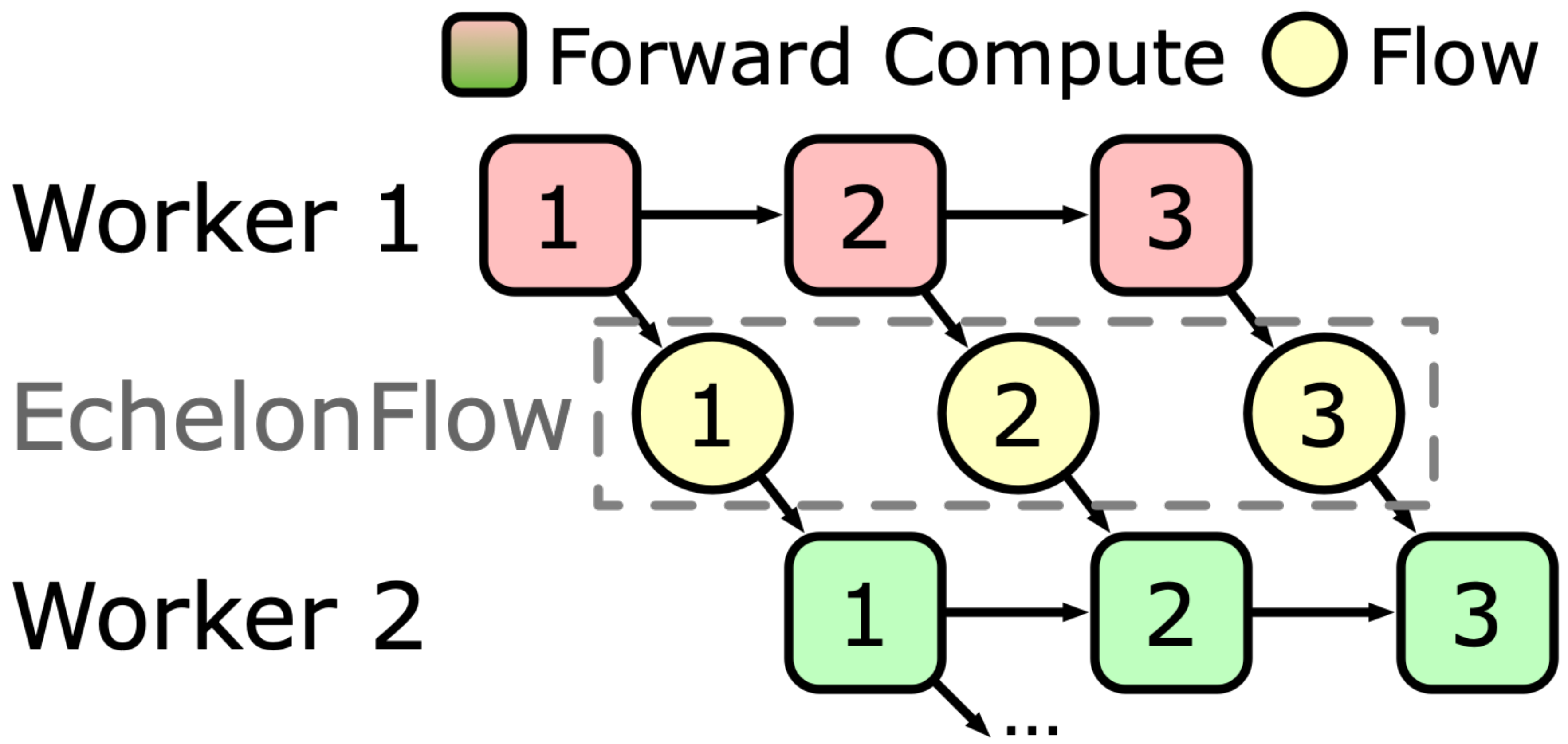
EchelonFlow

Properties



EchelonFlow

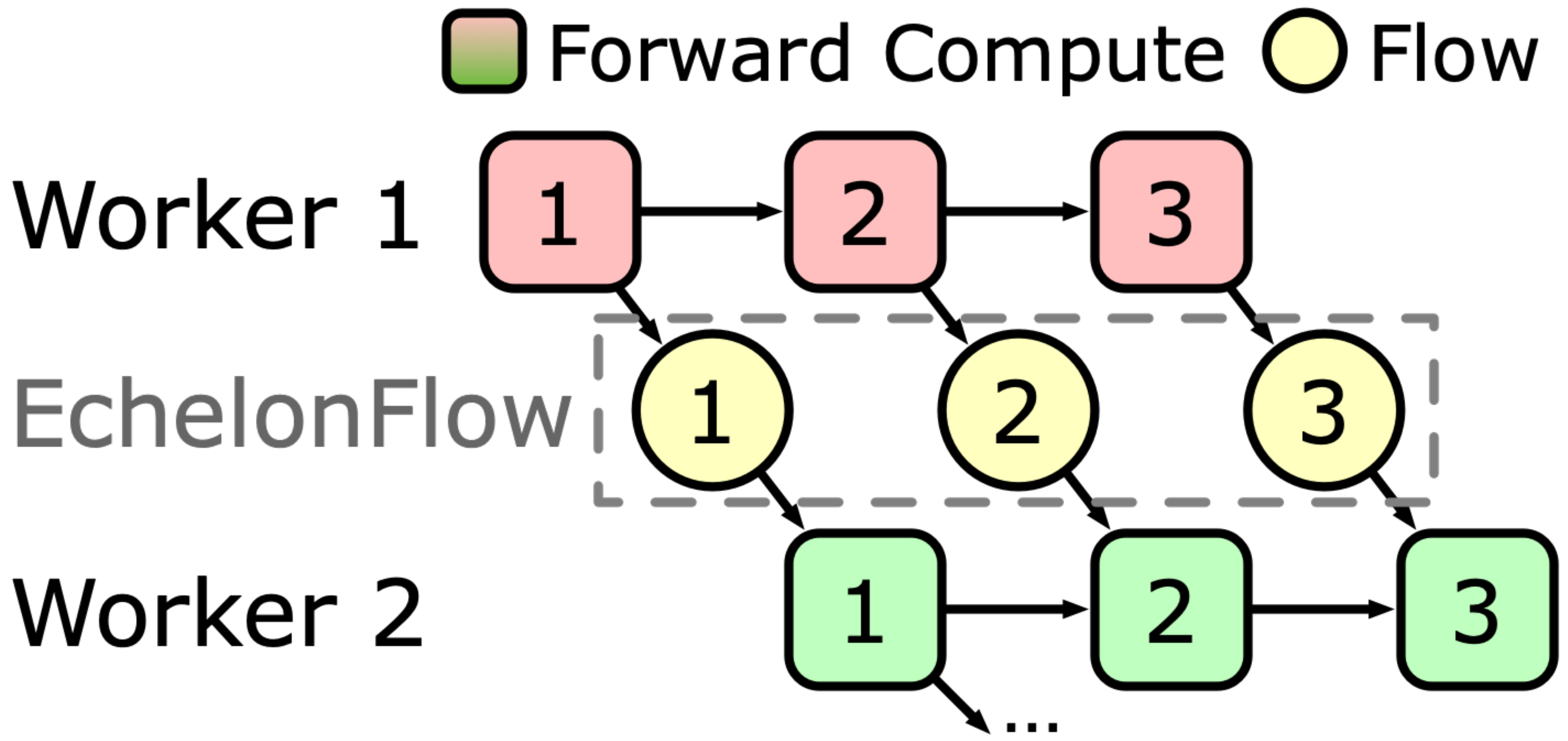
Properties



- General to diverse distributed training paradigms

EchelonFlow

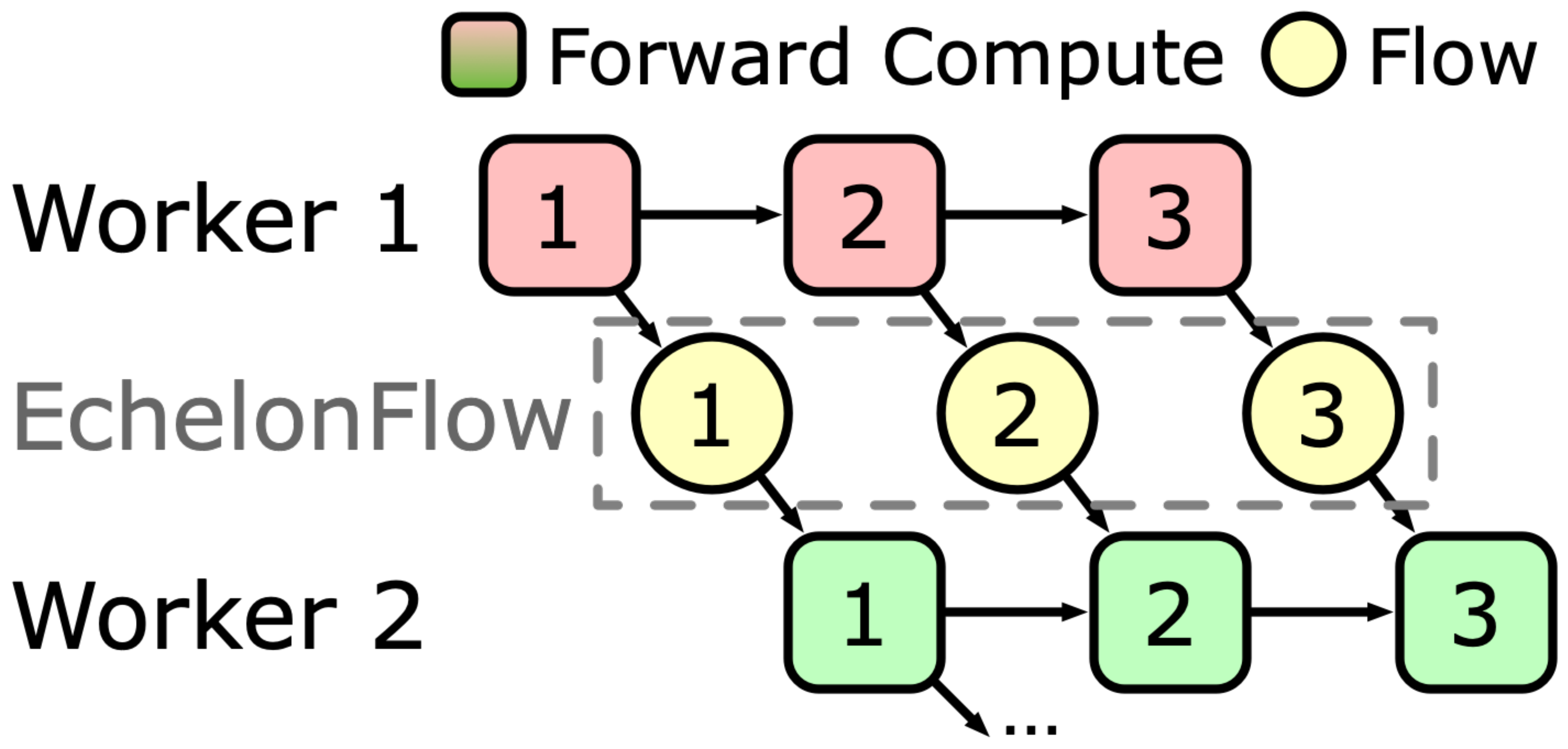
Properties



- General to diverse distributed training paradigms
- A superset of CoFlow

EchelonFlow

Properties



- General to diverse distributed training paradigms
- A superset of CoFlow
- Same complexity as CoFlow scheduling

EchelonFlow
Implementation

EchelonFlow

Implementation

- We know: flow size, computation time, and computation-communication dependencies

EchelonFlow

Implementation

- We know: flow size, computation time, and computation-communication dependencies
- Highly iterative → predictable

EchelonFlow

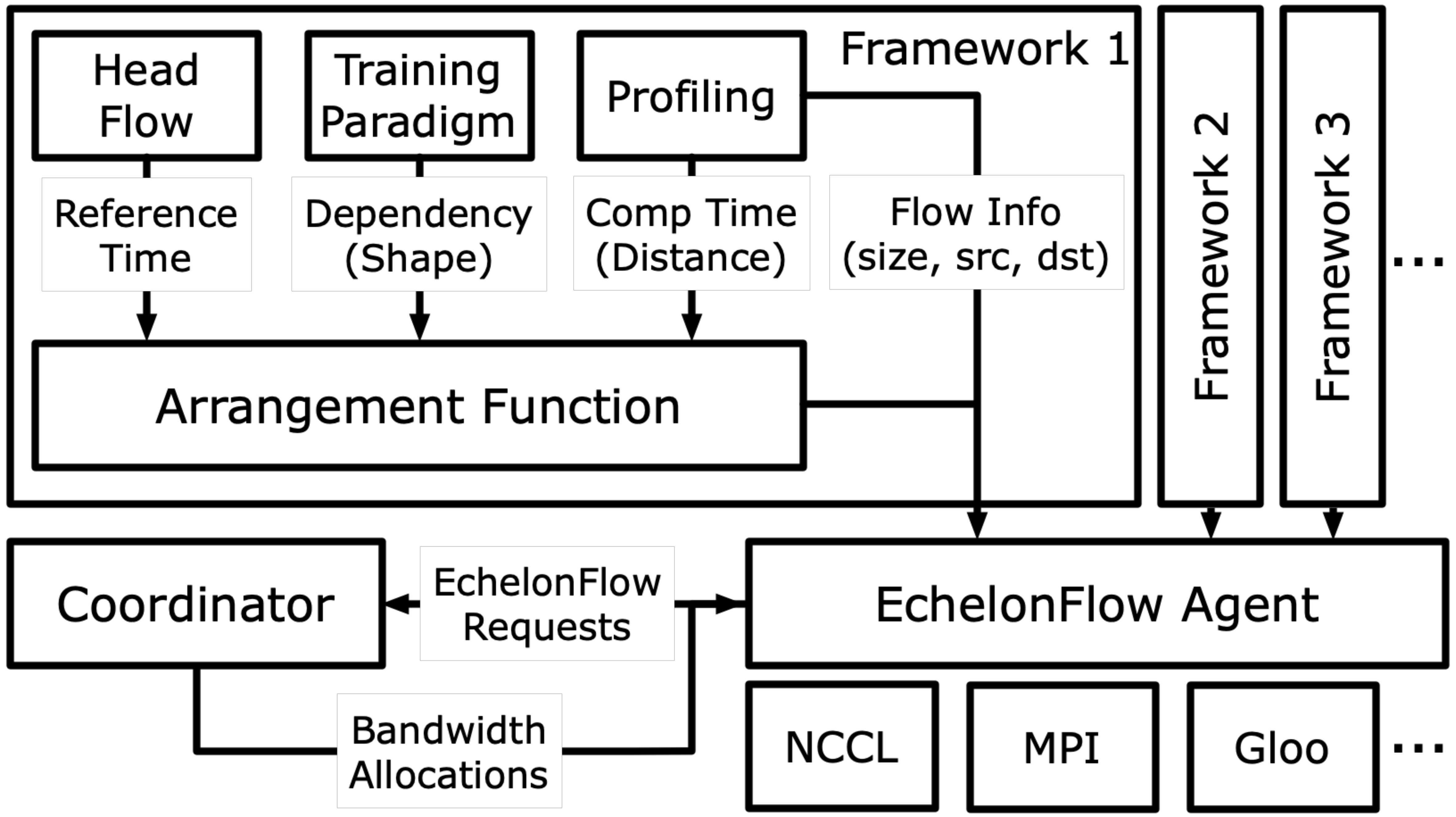
Implementation

- We know: flow size, computation time, and computation-communication dependencies
- Highly iterative → predictable
- Computation time and dependencies can be profiled a priori

EchelonFlow

Implementation

- We know: flow size, computation time, and computation-communication dependencies
- Highly iterative → predictable
- Computation time and dependencies can be profiled a priori



Conclusion and insights

Conclusion and insights

- EchelonFlow enables flow scheduling in distributed training

Conclusion and insights

- EchelonFlow enables flow scheduling in distributed training
- A new network abstraction

Conclusion and insights

- EchelonFlow enables flow scheduling in distributed training
- A new network abstraction
  - Flow finish times are not necessarily the same, but follow a pattern

Conclusion and insights

- EchelonFlow enables flow scheduling in distributed training
- A new network abstraction
  - Flow finish times are not necessarily the same, but follow a pattern
  - Arrangement function for high training throughput