

P. Barham, et al.: Pathways:
Asynchronous Distributed
Dataflow for ML

Background

- Most of today's state-of-the-art ML workloads use SPMD model
 - The limits are being pushed and developers have to resort to pipelining
 - Desire to better support computational sparsity
- Accelerators are becoming more heterogeneous
 - Giving exclusive access to homogeneous “islands” of compute is expensive
 - Pushing researchers towards MPMD computations
- Development of foundation models is getting more popular
 - They could allow a number of researchers to fine-tune the same shared model

Prior distributed ML systems

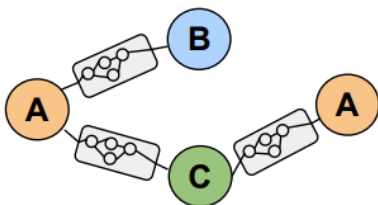
- Multi-controller architectures
 - For example PyTorch, Jax, and recent configurations of TensorFlow
 - Small communication costs
 - Makes pipelining or exploitation of computational sparsity hard
 - Typically assumes exclusive ownership of resources
- Single-controller systems
 - For example TensorFlow v1
 - Very general distributed dataflow model
 - Dispatch latency over DCN (data center network)
 - TF v1 is over-specialized for a single island of accelerators
 - Hard to achieve cross-host coordination


Pathways






- Adopts a single-controller model
 - Allows exploitation of computational sparsity and heterogeneity
 - Enables cluster management systems and better resource sharing
- Uses asynchronous dispatch
 - Matches the performance of multi-controller systems
- Supports scheduling with first-class support for gangs of SPMD computations
- Uses a sharded dataflow system

Architecture

Shared Dataflow Program (Plaque)



 Transfer subgraph

-  Host (many per island)
-  Resource Manager (global)
-  Scheduler (per island)
-  Executor (per device)
-  Collective operations

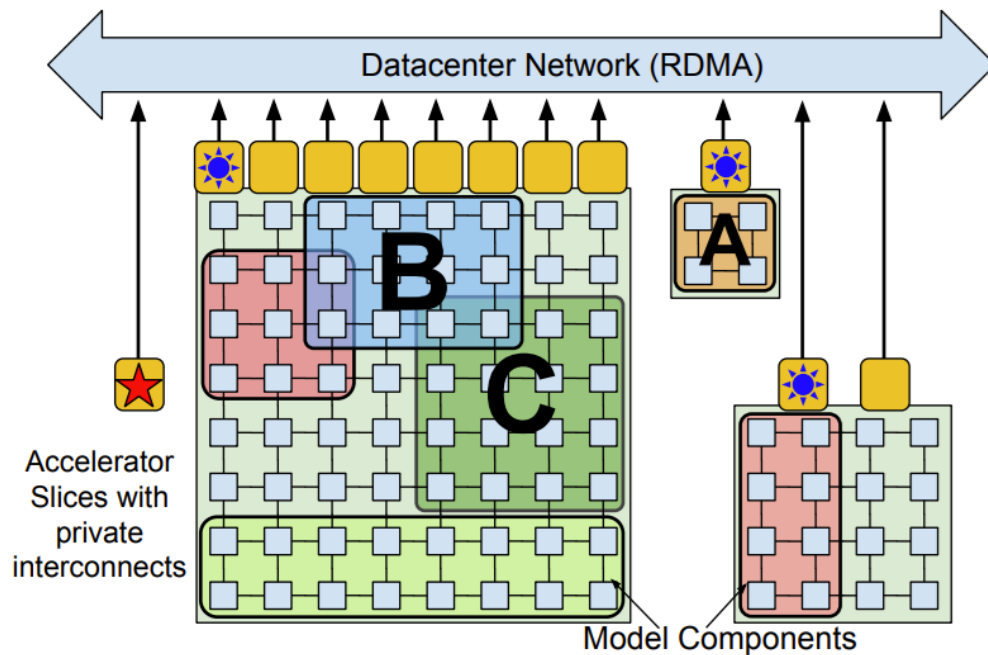


Diagram from the original paper

Coordination

- Each client program is
 - Assigned virtual devices
 - Converted to Pathways IR not containing device location information
 - This Pathways IR is progressively lowered to a low level representation including device locations
- Cross-host coordination using the DCN achieved via Plaque
 - Supports sparse data exchanges along sharded edges
 - Sends critical messages with low latency
 - Batches messages to the same host when high throughput required

Gang-scheduled dynamic dispatch

- Gang scheduling required for SPMD on shared set of accelerators
- Each island has its own centralized scheduler
- Plaques does the following:
 - enqueues local compiled functions at each accelerator, with buffer futures as inputs
 - enqueues network sends for the buffer futures output by function executions
 - communicates with the scheduler to find a consistent order of executions across the island

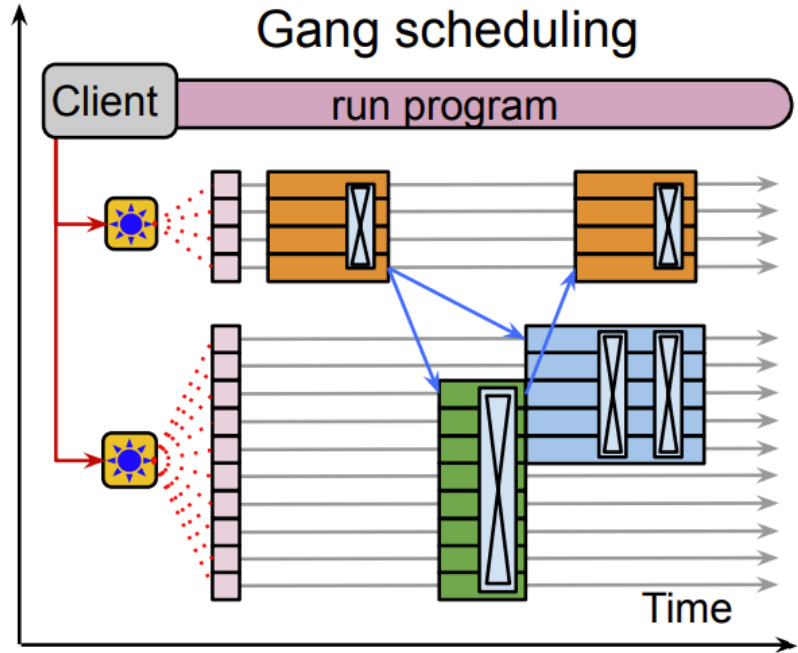
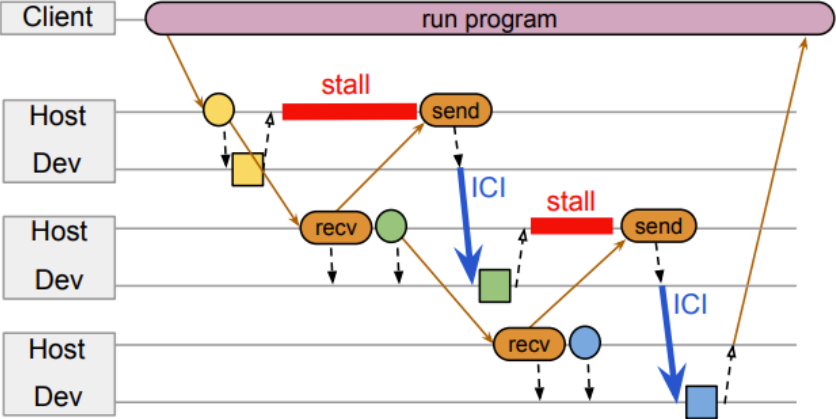
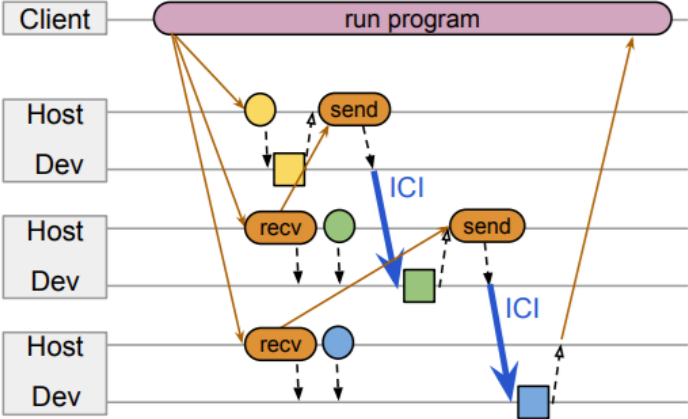


Diagram from the original paper

Parallel asynchronous dispatch



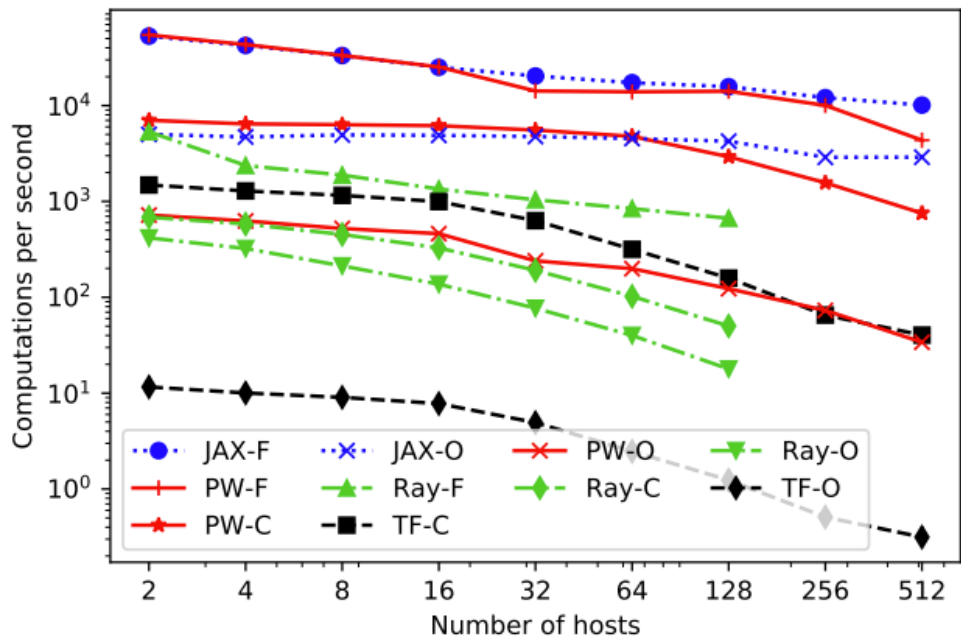
(a) Sequential dispatch



(b) Parallel dispatch

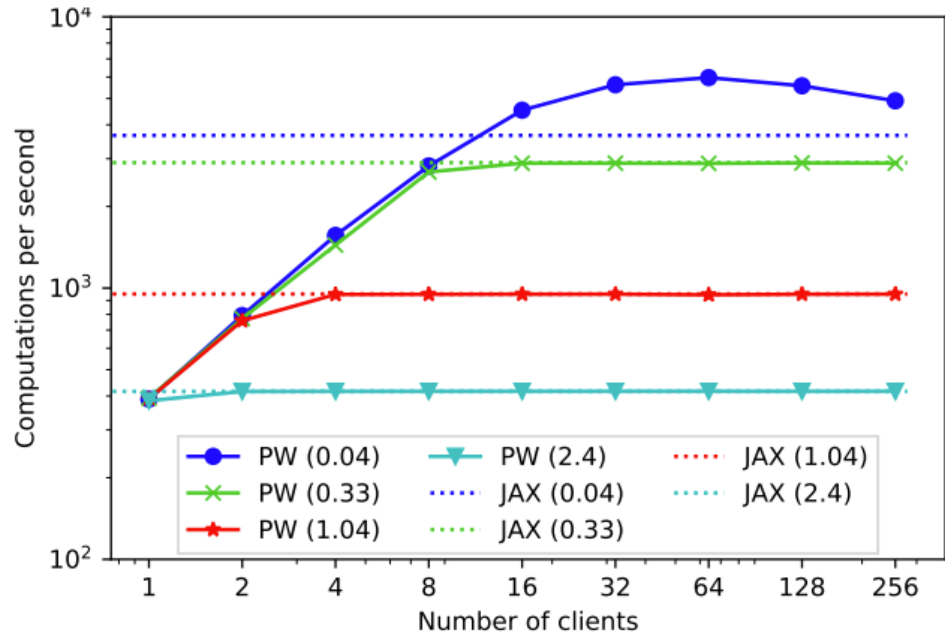
Diagram from the original paper

Results



Dispatch overheads compared
Diagram from the original paper

Results



Aggregate throughput of concurrent programs
Diagram from the original paper

Large scale model performance

- Numerical results on Pathways same as on JAX and TF running on their native systems
- Throughput of running a Transformer model with an Encoder-Decoder architecture matches JAX's

Model	Params	TPU cores	JAX	PATHWAYS
T5-Base	270M	32	618k	618k
T5-Large	770M	32	90.4k	90.4k
T5-3B	3B	512	282.8k	282.8k
T5-11B	11B	512	84.8k	84.8k

Training throughput (tokens/s)
Table from the original paper

Large scale model performance

- Training transformer-based language model with a Decoder-only architecture
 - Tested for different number of stages (S) and micro-batches (M)
- Pipelined matches SPMD
- Same throughput when cores partitioned into islands
- Throughput scales linearly with number of cores

Model configuration	TPU cores	PATHWAYS
Model-parallel (SPMD)	128	125.7k
Pipelining, S=4, M=16	128	133.7k
Pipelining, S=8, M=32	128	132.7k
Pipelining, S=16, M=64	128	131.4k
Pipelining, S=16, M=64	512	507.8k

Training throughput (tokens/s)

Table from the original paper

Impact

- PaLM (Pathways Language Model)
 - 540-billion parameter, dense decoder-only Transformer model trained with Pathways
- Minerva
 - Built on top of PaLM
- PaLM 2
- All of these models are closed

Opinion

- Seems promising, many large models developed on it already
- Focuses on TPUs not GPUs
- All models using it are closed, relies on Plaque which is closed-source too
- Evaluation of robustness? What if HBM not enough?

Discussion
