Network for AI Datacenters
IEEE802 NENDICA Meeting

José Duato

Royal Spanish Academy of Sciences

Motivation

- Understand what is an AI datacenter and how the emerging applications impact the network requirements.
- Encourage more discussions in this area to get consensus on what kind of network is needed.
- Standardize potential (new) technologies to build the large network on an open eco-system.

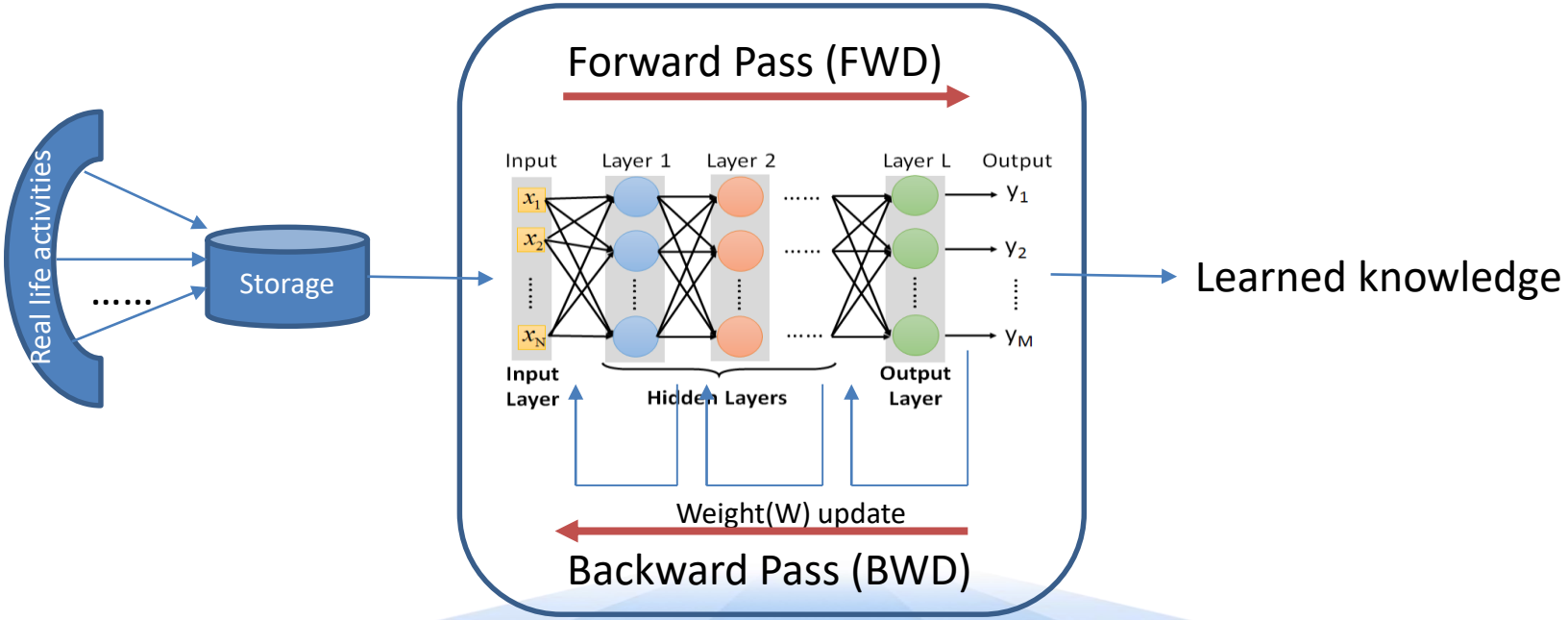
Expected Demand

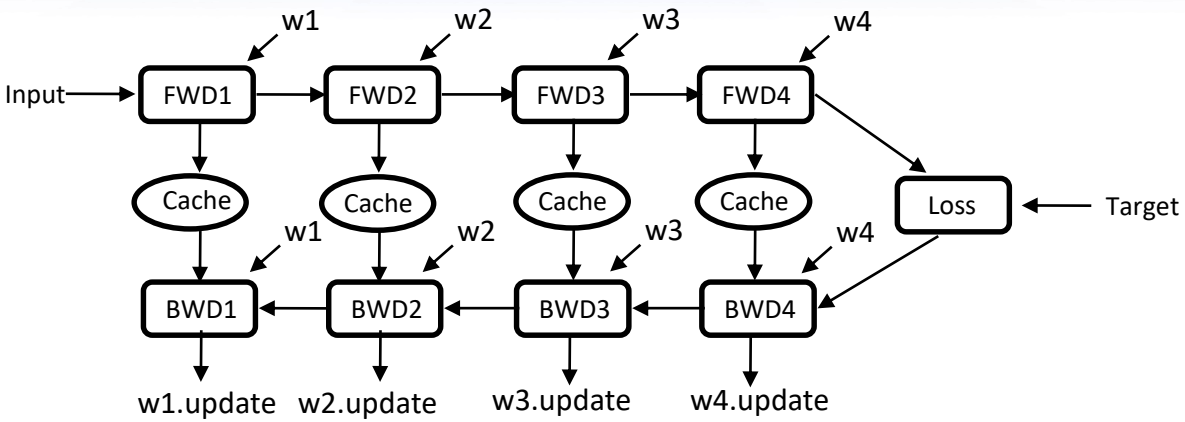
- The last decade has witnessed a very rapid expansion of many DNN-based AI solutions
- Regardless of where they are deployed, cloud datacenters are massively used for AI training
- The release of ChatGPT in Nov 2022 has garnered unprecedented attention, and triggered the recent boom of large language models (LLMs).

Model	Falcon_40B	GPT3_175B	GPT4_1.8T
Token Number	1 T	300 B	13 T
Training Time	2 months	34 days	100 days

- Huge datacenters are exclusively devoted to AI training and inference, and more are planned
- Expected size is on the order of 200K+ servers

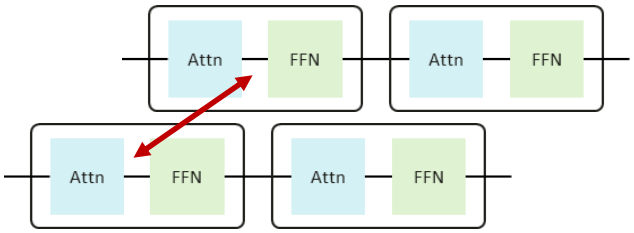
DNN-based AI Training



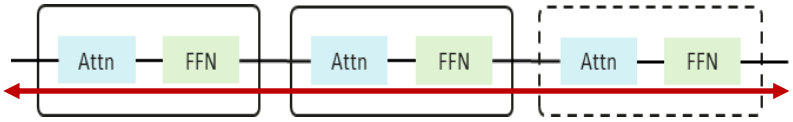


Parallelism in AI Training

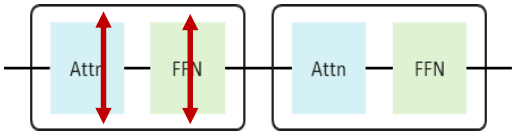
- Data parallelism
  - Massive parallelism: Batches are independent from each other
- Pipeline parallelism
  - Implemented when model does not fit into CPU/GPU memory
  - It is indeed two pipelines in opposite direction, where each pair of stages (one from each pipeline) need to share memory
  - Implemented with a multicore CPU/GPU with half the cores devoted to each of the pipelines
- Tensor parallelism
  - Samples processed in batches (matrix-matrix instead of matrix-vector)
  - Tensor parallelism is critical to maximize data reuse, increasing performance and energy efficiency
  - Benefits of tensor parallelism are maximized through scale-up technologies
- Expert parallelism
  - Multiple experts are used to expand AI model parameters. Normally only one of a few of them will be running.



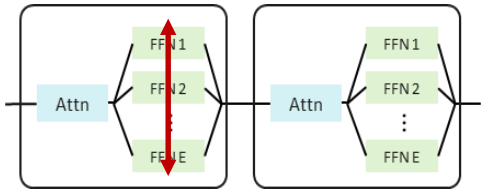
DP illustration in NN



PP illustration in NN



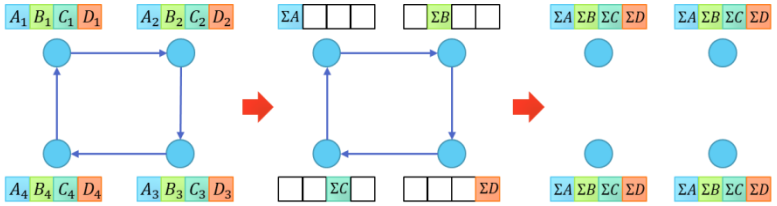
TP illustration in NN



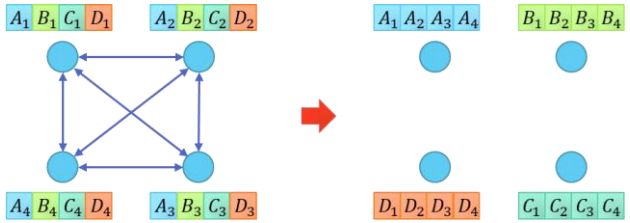
EP illustration in NN

Collective Communication in AI Training

- Collective communication is defined as communication that involves a group of processors. It used to be in MPI, including one to many, many to one or many to many communications.
- Modern distributed AI training relies on parallelism, that requires collective communication to achieve high performance.
- AllReduce and AlltoAll are typical collective communication operations in AI training.



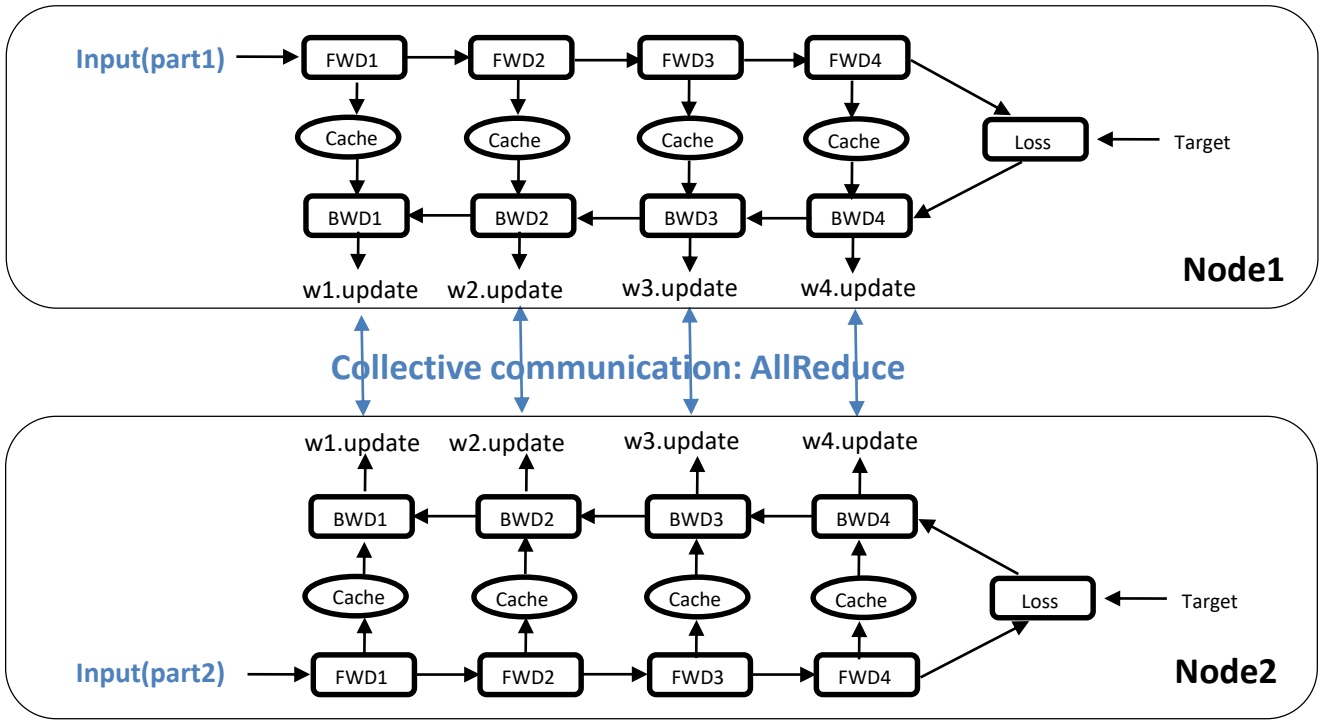
AllReduce



AlltoAll

Communication Requirements for DP

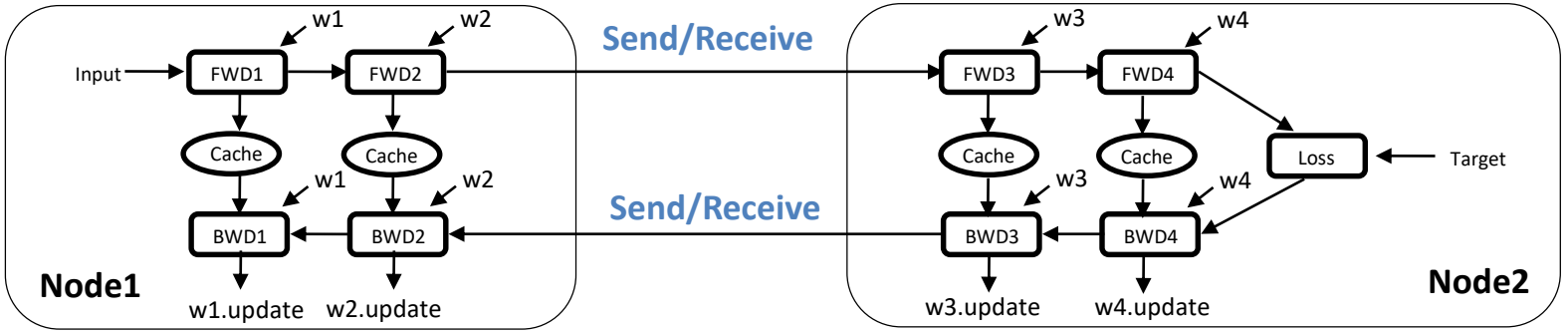
- Data parallelism
  - High bandwidth from secondary storage to CPUs/GPUs (for batch reading)
  - Achievable by connecting groups of SSDs and CPUs/GPUs to the same switch, in a balanced manner
  - Gradient exchanges among servers, mostly by means of calls to AllReduce
  - High bandwidth requirements. Latency also matters



Communication Requirements for MP

Depend on how the model is partitioned: Vertical slices is most usual

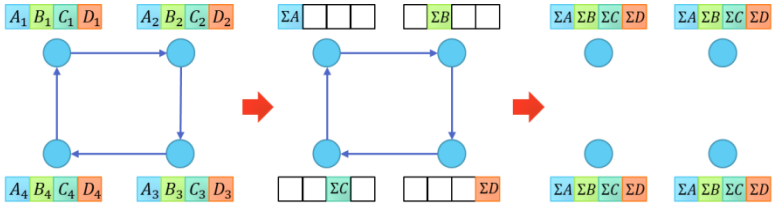
- Pipeline parallelism
  - Split the neural network into vertical slices (each slice containing several consecutive layers)
  - The results from each slice only need to be transmitted to the next pipeline stage
  - Just requires a ring topology (or a topology embedding it)
  - Huge bandwidth and very low latency are required
  - Latency can be partially hidden thanks to batch processing



- Tensor parallelism
  - Parallelism available within each pipeline stage (to be accurate, within each DNN layer)
  - When processing batches, each layer requires a matrix-matrix product (both in FWD and BWD)
  - This is where GPUs with tensor cores shine thanks to massive data reuse
  - Each matrix element is reused as many times as rows/columns exist in the other matrix

Viable implementations - Topology and Collective Communication Optimizations

- A ring can be simply embedded into a switch
  - Multi-port NICs or multiple NICs per server may be needed to achieve the required bandwidth
  - Attaching servers to the same switch also helps reducing latency (assuming that the required number of servers does not exceed the number of ports)
- The reduce phase of AllReduce can be implemented in software (possibly, with support in the NIC) in log time with a fat tree
  - Recursive reduce. A tree is required for each reduction, but many reductions occur in parallel
  - The communication is faster if different servers collect the results for different reductions



- The broadcast phase of AllReduce requires a topology with full bisection bandwidth (fat tree)
  - But a tree would suffice if hardware support for broadcast was available
- Topology is less important than server bandwidth and switch/NIC features (e.g. hardware-supported broadcast)

Realistic scenario

- The datacenter may not be exclusively devoted to AI training → application mix with very different communication requirements
- Task-to-server allocation and collective communication may not be fully optimized
- Most importantly, for 200K+ servers, components will frequently fail

Consequences

- Application mix
  - Not all traffic is collective communication
  - Network congestion and HoL blocking will occur
- Allocation and communication not optimized
  - Unbalanced resource utilization
  - Likely, network congestion and HoL blocking
- Components will frequently fail
  - Solutions are required: combination of hot swap, automatic path migration (APM), and checkpointing
  - Those solutions (especially APM) will unbalance traffic

Potential solutions

- Load balancing
  - Load-aware packet-level load balancing mechanisms will significantly help to eliminate bottlenecks and fully utilize existing bandwidth
  - Mandatory when implementing APM to balance traffic among the remaining healthy paths
- Adaptive routing with congestion control
  - Adaptive routing may be used together with load balancing to further alleviate in-network congestion, especially when produced by faulty components
  - Adaptive routing should only be used for in-network congestion, but never for incast congestion
  - Thus, incast congestion still requires congestion control
  - Incast congestion will likely occur during AllReduce

Why Two Similar Mechanisms(LB vs. AR)?

- Load balancing (LB) is better suited for regular, massive traffic
- Adaptive routing (AR) dynamically avoids congested network regions and it is best suited for very random or time-varying traffic
- Network load may vary so fast that load-aware LB may not adapt fast enough. In that case, AR achieves a very fast local response and quickly avoids rapidly arising congestion scenarios
- When a network component fails, the packets crossing the faulty component will be lost but will be re-transmitted
  - A fast response mechanism to guide the re-transmitted packets through an alternative healthy path is needed. AR takes into account local faulty components when selecting the switch output port
  - Faulty component information will later reach source NICs, and they will use it so that load- (and faulty-)aware LB makes the right decisions

Additional considerations

- Not all the mechanisms have the same response time
  - Adaptive routing reacts much faster than e2e congestion control and load-aware load balancing
  - Load balancing reacts faster than APM
- Not all the solutions have the same penalty
  - Backtracking to the last checkpoint has a high penalty
  - Avoid it whenever possible with APM plus load balancing, followed by retransmission of lost packets
  - Combine with AR for immediate response after failure detection

Summary & Next Steps

- Network requirements for AI datacenters have been introduced.
- Viable implementations in industry to meet AI training requirements have been shown.
- Realistic scenarios which request LB/AR/CC cooperation in addition to those viable implementations have been described.
- Next step:
  - Further analysis on topology/collective communication impact on datacenter network technologies.
  - Encourage more contributions in AI datacenters, and seek for potential standardization opportunities.