DRIVENETS

Utilizing Distributed Disaggregated Chassis for Back-End AI Networking Fabric

WHITE PAPER

The growing popularity of AI-based applications has propelled the buildup and usage of large high-performance computing (HPC) clusters dedicated for AI computation. Driven, predominantly, by generative AI applications - unlike other HPC applications - these applications are mainly used by hyperscalers for internal use or as a cloud-based service provided to enterprise customers. Within the networking (or fabric) part of AI clusters, back-end networking is a crucial element, impacting the overall performance of the clusters and the efficient utilization of their compute resources.

Networking for AI

AI workloads are applications that run over an array of servers, which typically host a dedicated computation engine (like GPGPU/FPGA accelerators) in addition to a CPU (that alone can act as a compute engine) and some storage capacity (typically a high-speed SSD). The AI application running on such servers is not running on a specific server but on multiple servers simultaneously. These can range from a few servers to thousands of machines all operating in sync and running the same application that is distributed among them.

The interconnect between these computation machines is referred to as back-end interconnect. This is opposed to front-end interconnect, a separate networking infrastructure that connects these machines to the public internet for queries and training data transfer. The back-end network must allow any-to-any connectivity between all machines running the same application. It also must cater for different traffic patterns that are associated with the type and stage of the application running.

A back-end interconnect solution for AI is different than a network built for connectivity to residential households or for a mobile network. It also is different than a network built for an array of servers purposed to answer queries from multiple users, like a typical data center.

Attributes of an AI Cluster

With the evolution of compute capacity, the ability to create a computing function that can factor in large data sets was created. The field of AI focuses on identifying such data sets and their resulting outcomes to educate the compute function with as many conclusion points as possible. The compute function is then required to identify patterns within these data sets to predict the outcome of new data sets not yet encountered.

Some example attributes of AI networking include:

- **Parallel computing:** AI workloads are a unified infrastructure of multiple machines running the same application and same computation task.
- **Size:** Such tasks can reach thousands of compute engines (e.g., GPGPUs, CPUs, FPGAs).
- **Job types:** Different tasks vary in their size, run duration, size and number of data sets considered, types of answers to be generated, and more. The different languages used to code the applications and the different types of hardware running the applications also contribute to a growing variance of traffic patterns within a network built for running AI workloads.
- **Bandwidth:** Large data sets require high-bandwidth traffic running in and out of servers for the applications to feed on. AI or other high-performance computing functions are reaching interface speeds of 400Gbps per every compute engine in modern deployments.
- **Latency and jitter:** Some AI workloads result in a response that is anticipated by a user. In such cases, the job completion time (JCT) is a key factor for the user experience, which makes latency an important factor. However, since such parallel workloads run over multiple machines, latency is dictated by the slowest machine to respond. This means that while latency is important, jitter (or latency variation) is in fact as much a contributing factor in achieving the required JCT.
- **Lossless behavior:** Following the previous point, a late arriving response delays the entire application. In a traditional data center, a dropped message results in retransmission (which often is not even noticed); in an AI workload, a dropped message means that the entire computation is either wrong or stuck. It is for this reason that AI-running networks require lossless behavior. IP networks are lossy by nature, so for an IP network to behave as lossless, certain additions need to be applied (which will be discussed later in this paper).

One important conclusion from these attributes is that a network purposed to run AI workloads differs from a traditional data center network in that it needs to operate in sync.

Industry solutions for AI and their drawbacks

There are several notable industry solutions for AI back-end networking. The following sections describe the key advantages and disadvantages of the main options now available for "in sync solutions."

Chassis-Based Solutions

Derived from telecom networking, a chassis-based router is built as a black box with all its internal connectivity concealed from users. It is often the case that the architecture used to implement the chassis utilizes line cards and fabric cards in a Clos-3 topology. As a result of this, the chassis behavior is predictable and reliable. It is in fact a lossless fabric wrapped in sheet metal, with only its network interfaces facing the user.

The disadvantage of a chassis in this case is its size. While a well-orchestrated fabric is a great fit for the network needs of AI workloads, its limited capacity of a few hundred ports for connecting to servers make this solution only fitting for very small deployments. In case chassis are used at a scale larger than the sum number of ports per single chassis, a Clos topology (in fact a non-balanced Clos-8 topology) is required, which breaks the fabric behavior of this model.

Standalone Ethernet Solutions

This type of solution derives from data center networking. While data center solutions are fast and can carry high-bandwidth traffic, they are based on standalone single-chip devices connected in a multi-tiered topology (typically Clos-5 or Clos-7). As long as traffic is only running within the same device in this topology, behavior of traffic flows will be close to uniform.

With the average number of interfaces per such device limited to the number of servers physically located in one rack, this single top-of-rack (ToR) device cannot satisfy the requirements of a large infrastructure. Expanding the network to higher tiers of the network also means that traffic patterns begin to alter, and application run-to-completion time is impacted. Furthermore, add-on mechanisms are mounted onto the network to turn the lossy network into a lossless one.

Another attribute of the traffic pattern of AI workloads is the uniformity of the traffic flows from the perspective of the packet header. This means that the different packets of the same flow will be identified by the data plane as the same traffic and be carried in the exact same path regardless of the network's congestion situation. This leaves parts of the Clos topology poorly utilized while other parts can be overloaded to a level of traffic loss.

Proprietary Locked Solutions

Additional solutions in this field are implemented as a dedicated interconnect for a specific array of servers. This is more common in the scientific domain of heavy compute workloads, such as research labs, national institutes, and universities. As proprietary solutions, they force the customer into one interconnect provider that serves the entire server array starting from the server itself and ending on all other servers in the array.

The nature of this industry is such where a one-time budget is allocated for building a "supercomputer." This means that the resulting compute array is not expected to grow further, only being replaced or surmounted by a newer model. This makes the vendor-lock of choosing a proprietary interconnect solution more tolerable. In AI implementation, however, this becomes problematic due to the dynamic and online nature of the infrastructure.

On the plus side, such solutions perform very well, and they can be found at the top of the list of the world's strongest supercomputers, using solutions from HPE (Slingshot), Intel (Omni-Path), Nvidia (InfiniBand) and more.

DDC as an AI Networking Distributed Synchronized Fabric (DSF)

As explained more fully in the DDC section below, Distributed Disaggregated Chassis (DDC) is a distributed formation of a chassis. As mentioned previously, the main disadvantage of a monolithic chassis is its size when building a network that serves AI workloads; a DDC can resolve this matter since it dismisses the limitations of the metal enclosure of the chassis.

When viewed from a telecom perspective, which is where DDC was defined, a DDC is a distribution of a chassis. When viewed from the perspective of data center networking, a DDC is a distributed synchronized fabric (DSF) solution that features the desired attributes of the back-end interconnect for AI workloads.

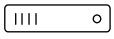
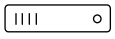
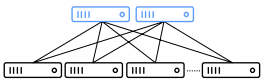
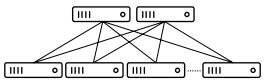
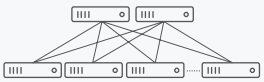
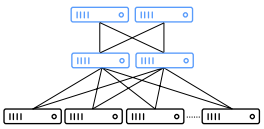
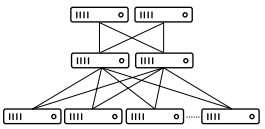
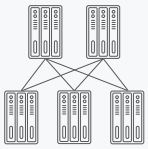
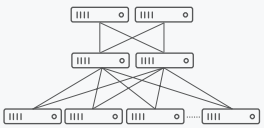
Such desired attributes include:

Standard - All external interfaces of the DSF are standard IP interfaces that can connect to any server type over standard Ethernet interfaces from any network interface card (NIC) vendor.

Lossless - Just like a black-box chassis, the DSF handles traffic between any group of interfaces on it at line rate and without dropping packets.

High-scale - A DSF is built like a Clos-3 without the limit of the metal enclosure, which allows a higher scale than the monolithic chassis. However, additional layers of fabric can be built into the distributed topology, transforming the Clos-3 topology into a uniform Clos-5 topology. This enlarges the maximum capacity of the DSF by an order of magnitude compared to Clos-3 topology.

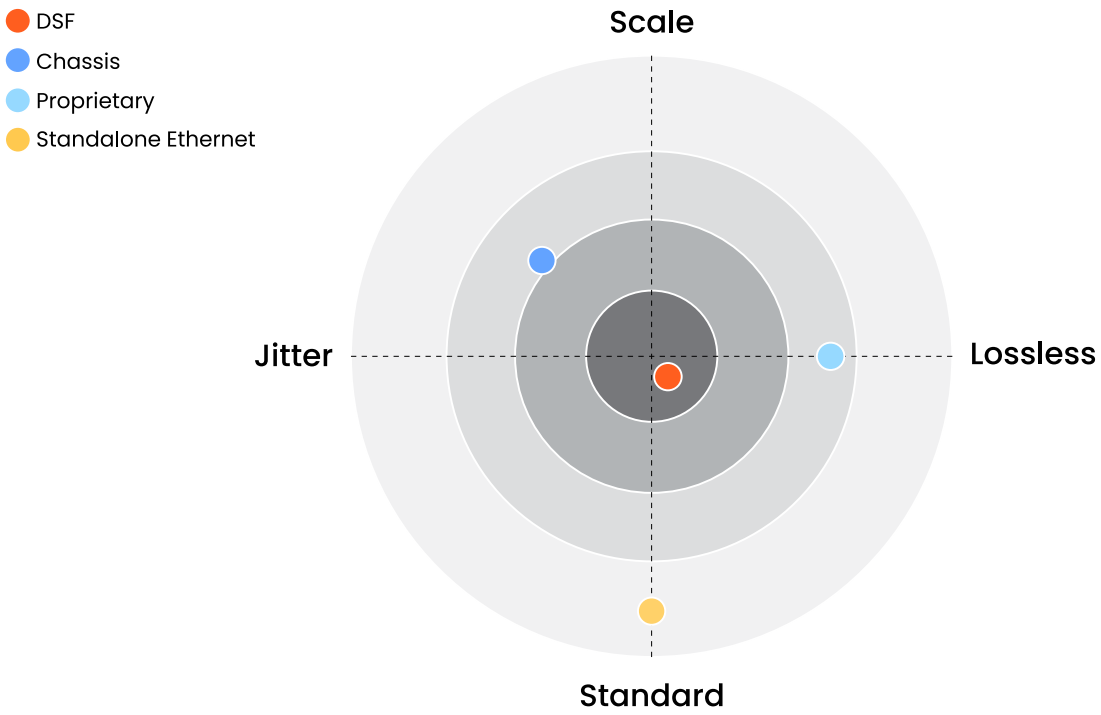
Figure 1: Scale comparison of DSF/Ethernet/chassis/proprietary networks

	DSF	Ethernet	Chassis	Proprietary
Standalone				
Network Hops / Standard Interfaces	**1/Tens**	**1/Tens**	**1/Tens**	**1/Tens**
2-tier				
Network Hops / Standard Interfaces	**1/Hundreds**	**3/Hundreds**	**1/Hundreds**	**0/0**
3-tier				
Network Hops / Standard Interfaces	**1/Thousands**	**5/Thousands**	**3/Thousands**	**0/0**

Low-jitter - With most chassis implementations, traffic between any two line cards is evenly distributed between all fabric cards. Similarly, the DSF implements a scheduling mechanism to balance traffic coming from distributed chassis packet-forwarder (DCP) devices between all DCF devices. This is done using a virtual output queue (VOQ) that is managed on the data plane (ASIC) level, allowing all fabric devices to get as close as possible to 100% utilization while none faces congestion scenarios. Such balancing implementation, which is blind to the packet header, avoids the impact of homogeneous and heterogeneous traffic patterns creating a varying latency for packets traversing the DSF.

Figure 2: Radar map of 4 solution types and their AI workload relevant attributes

Note: closer to center of the radar indicates a higher score.



Distributed Disaggregated Chassis (DDC)

DDC is an architecture that was originally defined by AT&T and contributed to the Open Compute Project (OCP) as an open architecture in September 2019. DDC defines the components and internal connectivity of a network element that is purposed to serve as a carrier-grade network router. As opposed to the monolithic chassis-based router, the DDC defines every component of the router as a standalone device.

Key definitions include the following:

- The line card of the chassis is defined as a distributed chassis packet-forwarder (DCP).
- The fabric card of the chassis is defined as a distributed chassis fabric (DCF).
- The routing stack of the chassis is defined as a distributed chassis controller (DCC).
- The management card of the chassis is defined as a distributed chassis manager (DCM).

All devices are physically connected to the DCM via standard 10GbE interfaces to establish a control and management plane. All DCPs are connected to all DCFs via 400G fabric interfaces in a Clos-3 topology to establish a scheduled, non-blocking data plane between all network ports in the DDC. A DCP hosts both fabric ports for connecting to DCFs and network ports for connecting to other network devices using standard Ethernet/IP protocols. A DCF does not host any network ports. The DCC is in fact a server and is used to run the main base operating system that defines DDC functionality.

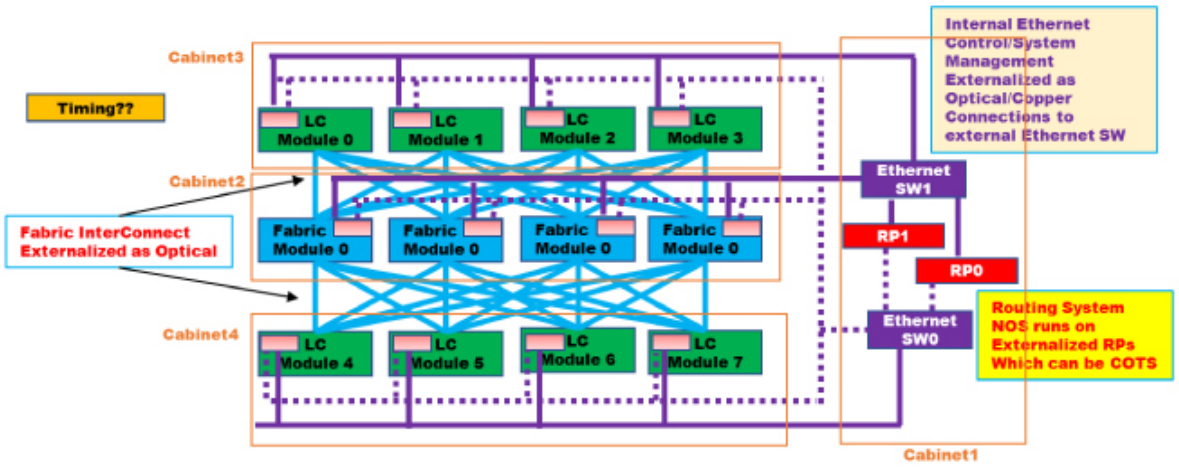
Advantages of the DDC include:

- **High-capacity:** Higher capacity is enabled since there is no metal chassis enclosure that must hold all these components into a single machine. This allows building a wider Clos-3 topology that expands beyond the boundaries of a single rack, making it possible for thousands of interfaces to coexist on the same network element (router).

- **Open:** DDC is an open standard definition that makes it possible for multiple vendors to implement its components. As a result, it is easier for the operator (telco) to establish a multi-source procurement methodology and stay in control of its network costs and supply chain as the network evolves.
- **Resiliency:** In this distributed array of components, each one can exist as a standalone as well as act as part of the DDC. This enables a very high level of resiliency to services running over a DDC-based router compared to services running over a chassis-based router.

AT&T announced it uses DDC to run its core MPLS, edge and peering IP networks, while operators worldwide also are using DDC for such functionality.

Figure 3: High-level connectivity structure of a DDC



Note: LC is defined as DCP above, fabric module is defined as DCF above, RP is defined as DCC above, and Ethernet SW is defined as DCM above.

Source: OCP DDC spec

DDC implements a concept of disaggregation. The decoupling of the control plane from the data plane enables sourcing software and hardware from different vendors and assembling them back into a unified network element when deployed. While this concept is rather new, it has had many successful deployments prior to its use as part of DDC.

Making an optimal performance AI back-end network a reality

AI back-end networking creates unique challenges that affect computational performance and compute resource utilization. Chassis-based, standalone Ethernet, and proprietary lock solutions all fail to resolve these challenges openly and efficiently. The introduction of a DDC-based solution, already proven as a robust and scalable solution in the telecom space, makes the DSF vision for an optimal performance AI back-end network a feasible reality.

DRIVENETS

DriveNets is a leader in cloud-native networking software and network disaggregation solutions. Founded in 2015 and based in Israel, DriveNets offers service providers and cloud providers a radical new way to build networks, substantially growing their profitability by changing their technological and economic models. DriveNets' solution – Network Cloud – adapts the architectural model of cloud to telco-grade networking. Network Cloud is a cloud-native software that runs over a shared physical infrastructure of standard white-boxes, radically simplifying the network's operations, offering telco-scale performance and elasticity at a much lower cost.

For more information, visit us at **www.drivenets.com**