

RTGWG Working Group
Internet Draft
Intended status: Informational
Expires: March 10, 2026

P. Huo
G. Chen
ByteDance
C. Lin
New H3C Technologies
Z. Jiang
ByteDance
September 10, 2025

Gap Analysis, Problem Statement, and Requirements in AI Networks
draft-hcl-rtgwg-ai-network-problem-03

Abstract

This document provides the gap analysis of AI networks, describes the fundamental problems, and defines the requirements for technical improvements.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 10, 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in

Table of Contents

1. Introduction.....	2
1.1. Requirements Language.....	4
1.2. Terminology.....	4
2. Existing Mechanisms.....	4
2.1. Load Balance.....	4
2.2. congestion control.....	7
2.3. Network reliability.....	8
3. Gap Analysis.....	9
3.1. Gap Analysis of Load Balancing.....	9
3.2. Gap Analysis of Congestion Control.....	10
3.3. Gap Analysis of Fast Failover.....	10
4. Problem Statement.....	11
5. Requirements for AI network Mechanisms.....	11
6. Security Considerations.....	12
7. IANA Considerations.....	12
8. References.....	12
8.1. Normative References.....	12
8.2. Informative References.....	12
Authors' Addresses.....	13

1. Introduction

Artificial Intelligence (AI), is a discipline and technology that studies how to enable machines to imitate and perform human intelligent activities. It involves simulating human thinking and decision-making processes, as well as analyzing and interpreting large amounts of data, allowing computer systems to learn, reason, judge, and predict automatically. The development of AI has achieved significant breakthroughs, including machine learning, deep learning, natural language processing, computer vision, and other fields. AI has a wide range of applications, covering areas such as healthcare, financial services, transportation, smart manufacturing, social media, and many more. In the future, AI will continue to advance and be applied, bringing more convenience and intelligent solutions to people's lives and work.

AI training network is a critical component in the field of artificial intelligence. It is a computer network system specifically designed for training and optimizing AI models. With large-scale datasets and optimization algorithms, AI training networks continuously drive the learning and evolution of AI models

to adapt to changing environments and demands. In the field of AI, training networks play a crucial role, providing strong support and foundations for technologies such as deep learning, machine learning, and neural networks. The development of AI training networks lays a solid foundation for the progress and application of AI technology, while also promoting its widespread use and development in various industries.

With the development of AI networks, the model parameters for AI training are becoming increasingly large. More than 100B parameters and are constructed with multiple layers. In order to improve training efficiency and make communication and computation parallel as much as possible, the current mainstream training frameworks all support mixed parallel strategies. To meet the demands of large-scale AI training, AI training networks typically adopt a distributed cluster approach, which brings forth the following new requirements:

- a. Ultra-high bandwidth demand: In AI training scenarios with large models, there will be a massive amount of communication data, which imposes higher bandwidth requirements on the network.
- b. Stability demand: Due to the long training time of large models, any failure during the training process can result in prolonged downtime, significantly affecting the efficiency of AI training. Therefore, it is necessary to quickly recover from failures and minimize their impact on AI training efficiency.
- c. Low Latency Demand: In large-scale AI training, which employs parallel distributed computing across multiple GPU nodes, each computation sub-process requires the completion of all participating GPUs. To improve training efficiency and parallelize communication and computation as much as possible, the current mainstream training frameworks support mixed parallel strategies.

Data Parallelism (DP): The training dataset is evenly distributed among all GPUs, with each GPU maintaining a replica of the entire model. In each iteration, all GPUs perform AllReduce to synchronize calculated gradients.

Pipeline Parallelism (PP): The model is divided into multiple stages, with each stage consisting of continuous layers of the model and handled by different GPUs. Each GPU in the pipeline receives input from the previous stage and sends output to the next stage.

Tensor Parallelism (TP): The entire model or each layer in PP can be further horizontally split, distributing each layer across a group of GPUs. In each iteration, GPUs in the same TP group perform

training networks, as it is often caused by network congestion.

the following features:

smaller number of flows, each with a substantial load.

However, in AI training for large models, individual flows have significant loads, resulting in bursts of high-intensity traffic.

1.1. Requirements Language

"OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

TBD

2. Existing Mechanisms

requirements of pretraining for large models are primarily reflected in terms of bandwidth, stability, and low latency. The following specifically illustrates the existing disparities in the actual capabilities of networks in these aspects.

2.1. Load Balance

N-tuple hash algorithm, which forwards traffic flow by flow.

distribute the load.

The diagram below illustrates a typical AI training network, utilizing a Spin-Leaf network architecture.

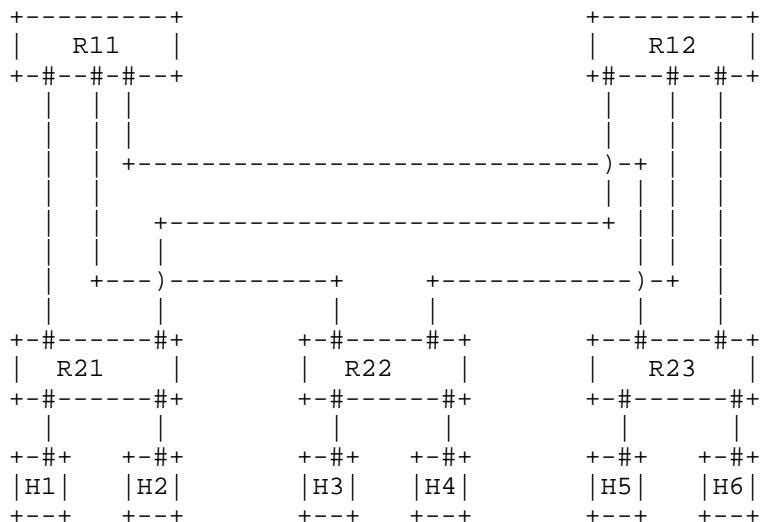


Figure 1: AI network diagram

In AI training networks, congestion is generally classified into three categories:

The first type is congestion from S-Leaf to Spin. For instance, if traffic flow 1 is from node H1 to node H5, and traffic flow 2 is from node H2 to node H6. Based on network bandwidth calculations, this should not cause network congestion. However, if the load balancing algorithm is inappropriate and leads to the selection of the same link from S-Leaf to Spin, it can result in congestion on the link from S-Leaf to Spine.

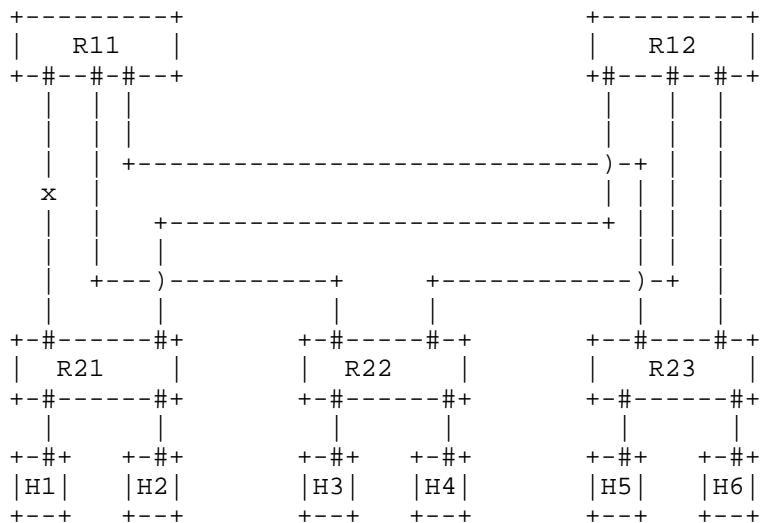


Figure : S-Leaf Spin

The second type is the link congestion from Spin to D-Leaf. For example, if traffic flow 1 is from H1 to H5, traffic flow 2 is from H2 to H6, traffic flow 3 is from H3 to H5, and traffic flow 4 is from H4 to H6. Based on network bandwidth calculations, this should not cause network congestion. However, due to inappropriate load balancing algorithms, if the traffic is directed to the same Spine, it can result in congestion on the link from Spin to D-Leaf.

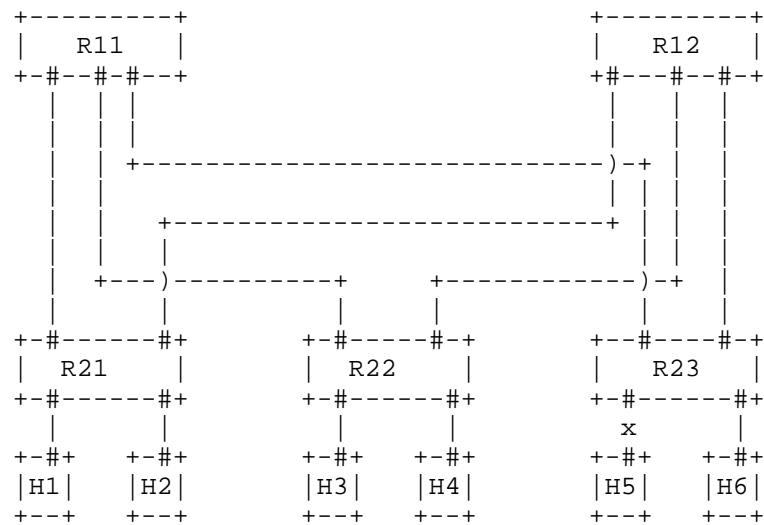


Figure : S-Leaf Spin

The third type is the congestion of network edge exit links. For example, if traffic flow 1 is from H1 to H5, traffic flow 2 is from H2 to H5, traffic flow 3 is from H3 to H6, and traffic flow 4 is from H4 to H6. Although there is no congestion within the network, due to uneven traffic planning, flow 1 and flow 2 occupy a large bandwidth, while flow 3 and flow 4 occupy a small bandwidth, resulting in congestion on the exit link from R23 to H5.

The above three scenarios illustrate that the flow-based load balancing strategy can easily lead to uneven load distribution, resulting in network congestion. While packet-based load balancing techniques can alleviate the uneven load distribution to some extent, they can cause packets of the same flow to arrive out of order due to different paths, necessitating network handling of packet reordering.

The inherent drawback of existing load balancing technologies is that they cannot perceive the actual utilization and congestion status of the network, thus leading to frequent congestion. Consequently, AI training networks require a more fine-grained load balancing capability to address these issues.

2.2. congestion control

The current mainstream network congestion control methods include ECN (Explicit Congestion Notification) and PFC (Priority-based Flow Control) technologies. These two techniques, while similar in principle, essentially represent a form of unidirectional congestion control. The underlying principle is to notify the sending end to reduce transmission speed when the receiving queue at the receiving

end reaches a threshold, thereby preventing congestion. In this context, ECN detects the state of the packet queue cache in the outgoing direction, while PFC detects the state of the packet queue cache in the incoming direction.

One issue with this method is the setting of queue thresholds. If the threshold is set too low, it can impact packet throughput and fail to effectively utilize the available bandwidth. On the other hand, setting the threshold too high may not effectively prevent network congestion.

Another issue relates to the extent of reduction in transmission speed when the sending end receives congestion notifications. Any significant reduction in speed can result in suboptimal network utilization, while a minimal reduction may not sufficiently address congestion.

Furthermore, when signaling the upstream sender to reduce transmission speed and adjust the network, this adjustment affects all traffic, rather than providing specific control for individual flows. Additionally, congestion can only gradually propagate upstream, leading to low adjustment efficiency.

Therefore, AI training networks require global congestion control mechanisms that can effectively manage congestion.

2.3. Network reliability

The methods for responding to local link faults and performing switchover.

Equal-Cost Multipath (ECMP): ECMP allows for fast fault switching by distributing traffic across multiple equal-cost paths. In the event of a failure on one path, traffic can be quickly redirected to an alternate path.

Fast Reroute (FRR): FRR is a mechanism that enables rapid switching to precomputed backup paths upon failure detection. It reduces the convergence time by bypassing the traditional control plane route convergence process.

The methods for responding to remote link faults and performing switchover.

BGP PIC (Prefix Independent Convergence): BGP PIC is a technique for fast iterative switching during network failures.

3. Gap Analysis

The training of large-scale AI models forms the foundation of artificial intelligence development. In comparison to small models, large models place stronger demands on large-scale distributed parallel training. On one hand, this is due to the sheer size of the models, which, limited by today's GPU memory, necessitates partitioning a single model across numerous GPUs for storage. On the other hand, training a larger number of parameters requires increased computational power, mandating the introduction of a larger scale of GPUs for acceleration. Consequently, there is a need for a significant increase in the quantity of GPUs.

Currently, training scale is generally denoted based on the number of GPU cards employed for a task. For instance, we refer to small-scale training for tasks involving fewer than a hundred cards, medium-scale for tasks involving a hundred to a thousand cards, and large-scale for tasks involving over a thousand cards. Models utilizing over ten thousand GPU cards are considered to be at an extremely large scale.

The large scale of AI networks gives rise to numerous challenges.

3.1. Gap Analysis of Load Balancing

As mentioned earlier, the current load balancing technologies primarily focus on per-flow hash-based forwarding and per-packet forwarding.

Technically, almost all network transmissions face an inherent issue: the need to avoid packet reordering within the network, as reordering triggers retransmission logic leading to reduced speeds at the receiving end. Consequently, when switches forward packets within the network, packets from the same connection are directed along a specific path, and the selection of this path relies on hash algorithms.

It is well-known that hash algorithms inevitably encounter collisions. If the distribution of hash algorithm is uneven, resulting in the majority of the traffic choosing the same link, congestion will occur on that link, while others remain underutilized. This issue is quite common in large-scale training scenarios.

Furthermore, due to the characteristics of traffic during AI training, bursts of high-bandwidth traffic often occur between the same connections. Consequently, selecting hash-based paths for these bursts of traffic can lead to severe hash conflicts, resulting in network congestion.

AI training networks require a new form of load balancing that can mitigate the impact of uneven loads caused by bursts of traffic. This new approach should aim to achieve load balancing as much as possible. For instance, breaking the assumption that packets from a single connection must be directed along a single path, allowing for out-of-order packet reception, thus fully utilizing the network's multipath forwarding capabilities.

3.2. Gap Analysis of Congestion Control

When a network experiences congestion, it is essential to promptly adjust traffic, directing it to paths with a larger available bandwidth while reducing the traffic on congested paths. This allows for the full utilization of the available bandwidth on idle paths.

Current congestion control methods mainly involve local congestion detection and adjustment at the congestion point, and they do not achieve global congestion control. While these methods have some effect, in certain specific situations, their efficiency is low. This is due to the need to wait for the congestion state to propagate upstream until it reaches a certain point in the upstream path, several hops away, before congestion control takes place to alleviate the current congestion. This results in inefficient congestion control.

Furthermore, this type of congestion control mechanism impacts the forwarding of all traffic and cannot achieve targeted congestion control.

For AI training networks, a new congestion control routing protocol is needed. When congestion is locally detected, it should be swiftly communicated, allowing for global congestion control. This approach is more efficient than local congestion control and requires a global end-to-end congestion control mechanism.

3.3. Gap Analysis of Fast Failover

Maintaining uninterrupted tasks for long periods is crucial for AI training of large models. However, hardware is prone to failures, and as the scale of training networks increases, the likelihood of network failures rises due to an increasing number of switches, network interface cards, and GPUs.

Therefore, AI training networks require the capability for rapid fault recovery. For instance, if a link in the network experiences a fault, packets transmitted through this link will be lost. It is essential to ensure that the duration of packet loss is shorter than the timeout period typically set by communication libraries to prevent task interruption. For AI training networks, fast fault

The current network fault switchover time includes fault detection, notification, and switchover time. For local fault scenarios with a backup link, fast switchover can achieve fault recovery at a millisecond level. However, for remote fault scenarios, the only option currently available is software convergence through routing protocols, resulting in fault recovery times in the seconds range, which does not meet the demand for rapid switchover in AI training networks.

For AI training networks, there is a need for a mechanism to handle remote fault points based on rapid fault detection, notification, and response to achieve fast fault recovery at a global level.

4. Problem Statement

The main issues in the current AI training scenarios include:

Load Imbalance: There is a lack of more appropriate load balancing mechanisms to handle hash imbalances and bursty traffic in AI training networks. Improved load balancing is needed to fully utilize the numerous links in AI networks, including optimal and non-optimal paths.

Reliability: There is a lack of fast handling mechanisms for remote network faults. A new global fast fault handling method is required, including fault detection, fault propagation, and routing protocol fast processing.

Fast Failover: The current fast failover mechanisms primarily respond to local faults and cannot achieve global fast response. Additionally, the performance of the failover mechanisms does not meet the requirements of AI training networks.

These issues need to be addressed to enhance the efficiency and reliability of AI training networks.

5. Requirements for AI network Mechanisms

For the existing AI training networks, new requirements include:
* **New Load Balancing Mechanisms:** Capable of performing load balancing based on data packets to avoid the imbalance caused by the relatively small number of flows and bursty traffic in AI training networks.

* New Congestion Control Mechanisms: To avoid the inflexibility of current congestion control mechanisms and achieve global, end-to-end congestion control.

* Fast Failover Mechanisms: The need for new fast failover mechanisms that can quickly detect faults, rapidly notify remote endpoints, and enable rapid global fault handling mechanisms.

6. Security Considerations

TBD.

7. IANA Considerations

This document does not request any IANA allocations.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

8.2. Informative References

TBD

PengFei Huo
ByteDance
China
Email: huopengfei@bytedance.com

Gang Chen
ByteDance
China
Email: chengang.gary@bytedance.com

Changwang Lin
New H3C Technologies
China

Email: linchangwang.04414@h3c.com

Zhuo Jiang
ByteDance
China
Email: jiangzhuo.cs@bytedance.com

