Network Management
Internet-Draft
Intended status: Informational
Expires: 7 January 2026

C. Liu
C. Guo
China Mobile

6 July 2025

Requirements Analysis of System and Network for Large Language Model Inference Service

draft-liu-nmrg-ai-llm-inference-requirements-01

Abstract

With the rise of ChatGPT, DeepSeek, and other Large Language Models, which is short for LLMs in the remaining part, as well as the proliferation of inference applications, inference serving oriented to large-scale users has become increasingly critical. However, due to the extreme demands on computing power and communication during inference, the large-scale service deployment of LLMs poses significant challenges. To address these challenges, different vendors have adopted diverse inference service architectures, such as vLLM, SGLang, Mooncake, etc. This paper investigates mainstream inference frameworks, summarizes their core design principle and research question, and analyzes the challenges and requirements they impose on network management. The goal is to lay a foundation for defining a unified LLM inference architecture in the future.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 7 January 2026.

Table of Contents

1. Introduction

Since the launch of ChatGPT in 2023, more and more product-level LLMs have emerged, with GPT-4o, Claude-Sonnet-3.5, Gemini, Kimi, and others leading the charge. In early 2025, DeepSeek-R1 reignited the LLM frenzy, and Musk's xAI recently unveiled the powerful Grok3. It is evident that LLMs will continue to reach new heights.

Major vendors, including OpenAI, Anthropic, DeepSeek, and Google, have deployed their LLM applications across mobile and web platforms. As the field grows, daily active users (DAUs) for these applications are expected to surge, potentially reaching hundreds of millions during peak periods. This presents significant challenges for large-scale inference services. For instance, up to now, DeepSeek still struggles with persistent "Service Busy" issues.

Existing large-scale inference service architectures primarily adopt two technical approaches: Prefill-Decoding (PD) Fusion and Prefill-Decoding Disaggregation, which is derived from the distinct computational characteristics of the Prefill (compute-intensive) and Decoding (memory-intensive) phases. Efficient network management and hardware coordination are essential to maximize system throughput and minimize user-perceived latency.

This document first introduces mainstream inference frameworks, then optimization metrics, and finally elaborates on the network and system requirements for deploying large-scale LLM inference services.

2. Service-Oriented Inference Frameworks

At present, there are two main technical routes of the mainstream LLM service systems, namely PD Fusion and PD Disaggregation. Prefill, which is to simultaneously compute all of tokens of user requests, also known as prompts, is characterized as computational intensive, computing-bound, with extremely high computing force requirements. Decoding generates user-required content based on the KV Cache and first token generated by Prefill phase. Due to the reuse of KV Cache of the tokens prior to the current token, it is characterized as memory-intensive and memory-bound, with higher requirements for memory in decoding phase. A complete LLM inference procedure is shown in Figure 1. Based on whether to decouple two stages with obviously different computing requirements, two technical routes of LLM inference serving system emerge, namely, PD Fusion and decoupled PD Disaggregation. The rest of this section describes in detail about the two technical architectures.

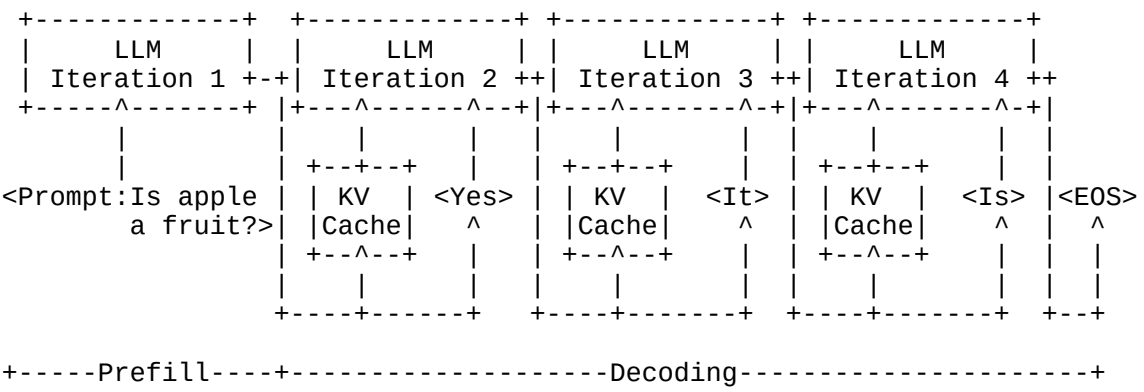

Figure 1: LLM Inference Process

Prefill: Processes all tokens in user prompts (Parallelizable, compute-bound, requiring high computing power).

Decoding: Generates output tokens sequentially based on the KV Cache from Prefill (Memory-bound, requiring high GPU memory).

2.1. PD Fusion Architecture

In PD Fusion, LLM instances are deployed within a single cluster, managed by a global scheduler responsible for load balancing, KV Cache management, and resource allocation. Most frameworks adopt vLLM[vLLM]’s paged KV Cache mechanism, inspired by OS virtual memory management. This approach stores KV Cache into non-contiguous physical blocks across nodes and uses a scheduler to map logical blocks to physical memory. Additionally, prefix-sharing strategies are employed to reuse KV Cache for prompts with identical prefixes, reducing redundant computations. Remote KV Cache replication across nodes is also required to reduce duplicated computing of KV Cache of same tokens. The architecture is shown in Figure 2.

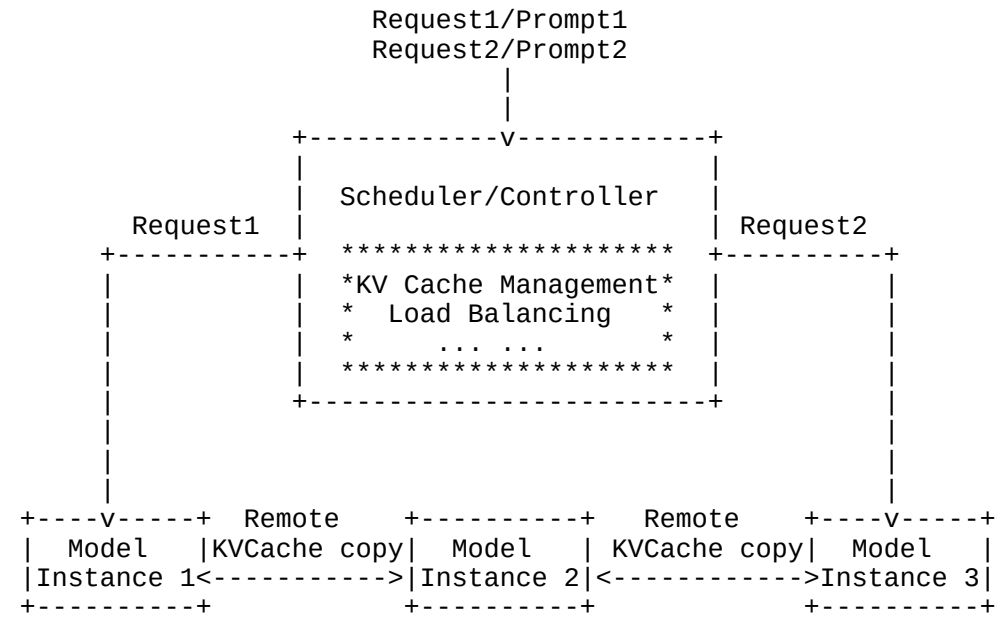

Figure 2: PD Fusion Architecture

2.2. PD Disaggregation Architecture

In PD Disaggregation, Prefill and Decoding are decoupled into separate instances to optimize hardware utilization. After Prefill computes the full KV Cache for a prompt, the data is transferred to Decoding instances for text generation. This architecture demands efficient coordination between Prefill and Decoding instances, as well as reliable high-speed data transmission. The workflow is illustrated in Figure 3.

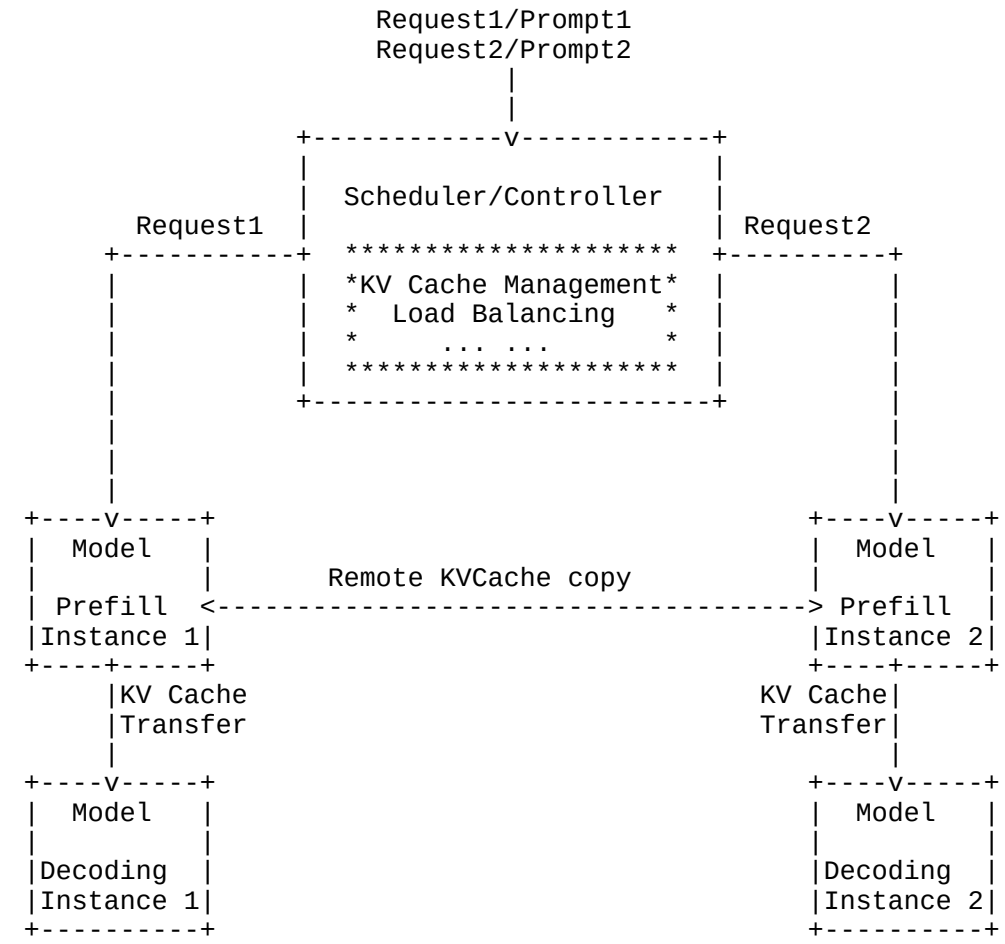

Figure 3: PD Disaggregation Architecture

3. Inference-related Metrics

The ultimate goals of an inference system are to maximize system goodput which reflects the serving volume of user requests and minimize user-perceived latency at low cost. For both PD Fusion and PD Disaggregation architectures, three kind of specific key metrics are defined as follows:

Latency-related Metrics:

TTFT (Time to First Token): The time taken by the Prefill phase to generate the first token.

TBT (Time Between Tokens): The interval between consecutive token generations in the Decoding phase.

E2E Latency (End to End Inference Latency): The latency of the request’s entire inference process, including TTFT and TBT.

Throughput-related Metrics:

TPS (Tokens Per Second): The number of tokens generated per second by the inference system.

RPS (Requests Per Second): The number of requests processed per second by the inference system.

Cost-related Metrics:

Cost Per Token: How much it costs for inference system to generate each token.

4. Research Question for Service-Oriented Inference Frameworks

From the user’s perspective, end-to-end inference latency generated during a complete inference process is a key metric directly impacting quality of service and user experience. From the perspective of llm deployment, the most critical challenge is how to deploy llm services at the lowest possible cost while meeting basic user requirements for inference latency, minimizing idle rates of computational resources, and ensuring systems operate at near-saturation levels. Industry commonly adopts decoupled architectures (e.g., PD Disaggregation) or other decoupled structures to deploy and independently optimize distinct model components on heterogeneous clusters. However, maximizing system throughput and reducing inference deployment costs inevitably sacrifices user end-to-end inference latency. In other words, latency, throughput, and cost form an impossible trinity. Synthesizing the needs and objectives of both users and service providers, the primary challenge for inference systems is: How to maximize system throughput and minimize deployment costs while adhering to Service Level Objective (SLO) constraints for inference services?

5. Challenges for Service-Oriented Inference Frameworks

To address the above key issue, current industry practices (e.g., Mooncake[Mooncake], DeepSeek, vLLM[vLLM], SGLang[SGLang]) employ methods such as KV Cache prefix matching, PD Disaggregation deployment optimization, efficient KV Cache memory management, and flexible load balancing scheduling to enhance resource utilization. These approaches also aim to leverage fragmented, low-cost and idle

resources, thereby increasing user throughput and reducing inference service deployment costs. However, this approach faces several significant challenges:

5.1. Challenge 1

Whether using PD Disaggregation or PD Fusion deployment, the industry widely adopts KV Cache prefix matching for large-scale inference deployment optimization. This technique aims to reduce redundant computation and storage of prefix KV Cache across different inference requests, thereby saving computational resources and ultimately improving system throughput while lowering deployment costs. How to systematically manage these KV Cache prefixes within the inference network— including storage mechanisms, placement strategies, scheduling schemes, and replacement policies— remains a critical and urgent challenge.

5.2. Challenge 2

To optimize the trade-offs between inference deployment cost, service performance, and system throughput, implementing more efficient resource- and KV Cache-aware mechanisms within the inference network’s management and control plane, alongside hardware- and network-aware load balancing scheduling, presents an extremely crucial challenge.

5.3. Challenge 3

If a PD Disaggregation deployment approach is adopted, ensuring efficient transmission of KV Cache between the separated, heterogeneous Prefilling and Decoding clusters becomes a vital challenge. This transmission must be designed to either avoid impacting end-to-end inference latency entirely or minimize its impact as much as possible.

6. Network Management Requirements for Service-Oriented Inference Frameworks

To achieve large-scale LLM service deployment, frameworks MUST meet the following requirements in both control plane and data plane.

6.1. Efficient Load Balancing

Both PD Fusion and PD Disaggregation architectures require dynamic load balancing to prevent server overload. For PD Disaggregation, schedulers MUST consider compute constraints (Prefill) and memory constraints (Decoding) when distributing requests.

6.2. KV Cache Management

Effective KV Cache management is critical. Most frameworks adopt vLLM’s paged KV Cache mechanism, schedulers are REQUIRED to handle memory allocation, cross-request KV cache sharing, and KV cache replacement policies. Future optimizations must address exponential user growth and ensure efficient cache synchronization across clusters or nodes.

6.3. KV Cache Transmission

PD Disaggregation architectures demand high-speed, reliable transmission of KV Cache data between Prefill and Decoding instances. The network MUST provide low-latency, high-bandwidth channels to ensure seamless coordination.

7. Security Considerations

TBD.

8. IANA Considerations

TBD.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

[Mooncake] Qin, R., "Mooncake: A KVCache-centric Disaggregated Architecture for LLM Serving", 9 July 2024.

[SGLang] Zheng, L., "SGLang: Efficient Execution of Structured Language Model Programs", 6 June 2024.

[vLLM] Kwon, W., "Efficient Memory Management for Large Language Model Serving with PagedAttention", 2023.

Authors’ Addresses

Chang Liu
China Mobile
Beijing
100053
China
Email: liuchangjc@chinamobile.com

Chuyi Guo
China Mobile
Beijing
100053
China
Email: guochuyi@chinamobile.com