cats
Internet-Draft
Intended status: Standards Track
Expires: 17 September 2026

N. Zhang, Ed.
M. Han, Ed.
X. Yi, Ed.
China Unicom

16 March 2026

A token-aware traffic steering solution for agent service
draft-zhang-cats-token-aware-ts-00

Abstract

This document proposes a token-aware traffic steering mechanism. By parsing estimated token length, task type, semantic urgency, and comprehensively incorporating network link status, model capabilities, and compute resource states into routing decisions, this mechanism achieves joint optimal scheduling of resources.

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 17 September 2026.

Table of Contents

1. Introduction

In the era of thriving AI applications, agents have become a critical link connecting users and cloud services, where their response speed directly determines user experience. Collaboration between terminal and the cloud can enhance resource utilization efficiency and promote the popularization of intelligent services. However, challenges exist when agents invoke large model services on the cloud:

1. Data processing on cloud (especially inference computation) accounts for about 85% of the end-to-end total latency. Simply relying on network path optimization or bandwidth guarantees cannot fundamentally resolve latency issues caused by computing queuing or model overload.

2. Requests initiated by agents show significant differences. Short-context tasks (such as command control) and long-context tasks (such as complex inference) have different requirements for latency sensitivity and computing resources. Traditional "best-effort" scheduling modes cannot meet these differentiated service demands.

3. The network lacks awareness of the token characteristics of upper-layer applications, leading to resource mismatches and user experience bottlenecks.

To solve these problems, the network should act as a collaborative scheduling medium connecting agents and cloud computing. By perceiving the token features of requests, the network can perform global optimal scheduling combined with network status, model capabilities, and computing load. While Compute-Aware Traffic Steering (CATS) has proposed joint routing and scheduling based on network and computing status[I-D.ietf-cats-framework], existing CATS solutions lack awareness of model capabilities and token features. This draft proposes a token-aware traffic steering method to improve the end-to-end service experience for agents.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "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.

Abbreviations and definitions used in this document:

*Token: The fundamental unit for Large Language Models (LLMs) to process text. *TTFT: Time to First Token, a key metric for measuring the response speed of inference services. *SLA: Service Level Agreement.

3. Use Cases

3.1. Low-latency interactive control

When agents respond to immediate user commands (e.g., "turn on the light", "play music"), they are expected to provide an extremely fast interactive experience. However, such tasks usually involve few tokens and simple logic, if blocked by longer tasks, it can lead to severe response delays. Therefore, the network needs to identify these short-token featuress and schedule them onto low-latency path and edge computing to ensure millisecond-level response times.

3.2. Complex infer3ence and content generation

When agents process complex tasks like code generation or long document summarization, they are expected to provide deep intellectual assistance to users. However, such tasks involve hundreds or even thousands of Tokens for computation, requiring high GPU memory and stable computing. Insufficient resources may cause task timeouts or interruptions. Therefore, long-token tasks need to be precise identified and guided to center computing nodes with ample computing to ensure service continuity.

3.3. Resource contention under mixed load

During peak business hours, when a large number of simple requests and a small number of complex inference tasks concurrently flood into the cloud cluster, it is expected to maximize the utilization of computing resources. However, traditional First-In-First-Out queue mechanisms often result in low-latency and short tasks being blocked by long-computation tasks, significantly degrading overall user experience. Therefore, the network needs to identify task priority based on token features, dynamically scheduling short tasks to lightly loaded nodes or dedicated channels, achieving isolated operation and differentiated assurance for tasks of varying complexities.

4. Requirements

To achieve the aforementioned goals, the system needs to meet the following key requirements.

4.1. Token features awareness capability

Terminal agent can extract key token fearures, including: estimated token length, task type, semantic urgency.

Terminal agents or agent gateways support transferring these features to a Network ID and embedding it into network packets, enabling identification by network devices.

4.2. Real-time multi-dimensional state monitoring

Network can monitor real-time network state such as link latency, jitter, available bandwidth, and packet loss rate.

Computing can obtain real-time information on each inference node’s GPU load, memory usage, current queue length, and estimated inference time (TTFT).

4.3. Dynamic joint scheduling decision

Policy-based routing algorithms are supported, capable of dynamically selecting optimal network paths (e.g., low-delay dedicated line vs. public internet) and optimal computing nodes (e.g., edge small model vs. cloud large model) based on token features and real-time states.

Priority queue management capabilities are possessed to ensure high-priority short tasks are not blocked by long tasks.

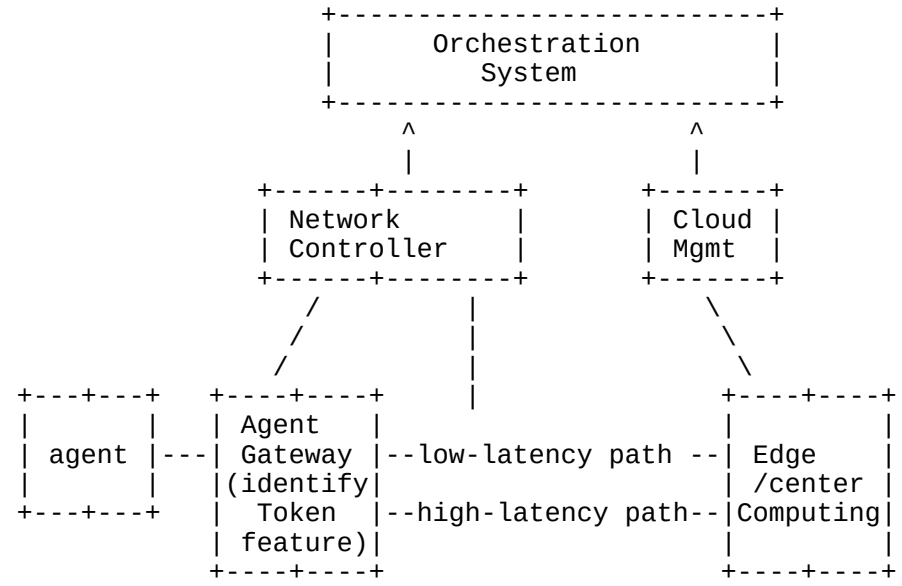
4.4. Standardized Interfaces and Protocol Extensions

A standard format for token feature descriptions needs to be defined.

Signaling interaction mechanisms between the network controller and orchestration system, and between the cloud management (Mgmt) and orchestration system.

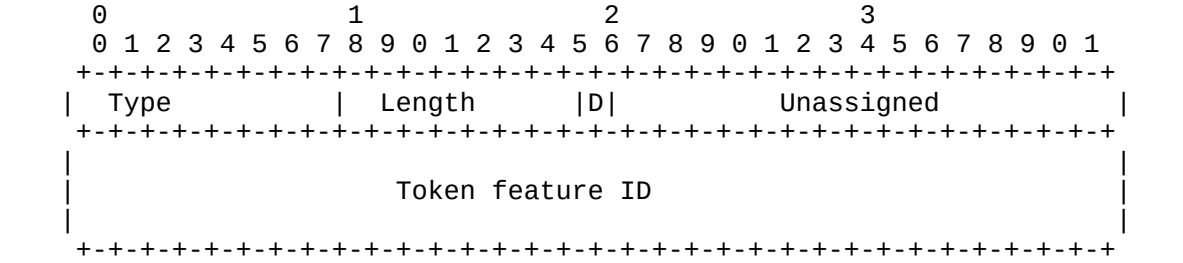
5. Token-aware traffic steering deployment solution

Figure 1 shows the architectural diagram of this deployment solution.



The specific implementation process is as follows:

a. Token feature extraction: When a request is initiated, the agent or agent gateway instantly analyzes the prompt content, estimates the number of tokens, and identifies the task type (e.g., "immediate control" or "complex inference"). Subsequently, the feature is encoded as a token feature ID and encapsulated in the packet header using the following format:



Length: 8-bit unsigned integer that indicates the total number of the octets of the value field.

D: 1-bit field used to indicate whether the current data transaction is directed.

Unassigned: 15-bit field reserved for future use. They MUST be set to 0 on transmission and MUST be ignored on receipt.

Token feature ID: 64-bit group ID of token feature.

b. Real-time state awareness: The orchestration system continuously collects full-network link quality (bandwidth, delay, jitter) and real-time status of each compute node (GPU utilization, queue length, estimated TTFT).

c. Dynamic joint routing decision: The orchestration system matches request features with real-time states:

Short/urgent tasks: Prioritize routing to the nearest edge node with the lowest network latency.

Long/heavy tasks: Prioritize routing to a central cloud node with ample computing and shorter queues.

d. Execution and feedback: The network forwards according to the specified path, and the computing processes the queue according to priority. After the task is completed, actual latency data is fed back to optimize subsequent scheduling strategies.

6. Deployment Effect

By employing a refined task priority identification mode, it eliminates long-task blocking and resource contention, improving the accuracy of service guarantees for critical business (e.g., interactive control).

This solution can simultaneously achieve acceleration for low-latency short tasks and stability guarantees for computationally intensive tasks by jointly monitoring network link latency and node TTFT.

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>.

[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>.

9.2. Informative References

[I-D.ietf-cats-framework] Li, C., Du, Z., Boucadair, M., Contreras, L. M., and J. Drake, "A Framework for Computing-Aware Traffic Steering (CATS)", Work in Progress, Internet-Draft, draft-ietf-cats-framework-22, 14 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-cats-framework-22>.

Authors' Addresses

Naihan Zhang (editor)
China Unicom
Beijing
China
Email: zhangnh12@chinaunicom.cn

Mengyao Han (editor)
China Unicom
Beijing
China
Email: hanmy12@chinaunicom.cn

Xinxin Yi (editor)
China Unicom
Beijing
China
Email: yixx3@chinaunicom.cn