High-speed Railway 1 (2023) 141–146

Contents lists available at ScienceDirect

High-speed Railway

journal homepage: www.keaipublishing.com/en/journals/high-speed-railway

Research article

TCP-LTE/5 G Cross-layer performance analysis tool for high mobility data networking and a case study on high-speed railway

Ruihan Li, Yueyang Pan, Xiangtian Ma, Haotian Xu, Chenren Xu*

School of Computer Science, Peking University, Beijing 100871, China

ARTICLE INFO

Keywords:
LTE
5 G
Cellular networks
High mobility
TCP
Network diagnostics
High-speed railway

ABSTRACT

Nowadays, high mobility scenarios have become increasingly common. The widespread adoption of High-speed Rail (HSR) in China exemplifies this trend, while more promising use cases, such as vehicle-to-everything, continue to emerge. However, the Internet access provided in high mobility environments still struggles to achieve seamless connectivity. The next generation of wireless cellular technology 5 G further poses more requirements on the end-to-end evolution to fully utilize its ultra-high band-width, while existing network diagnostic tools focus on above-IP layers or below-IP layers only. We then propose HiMoDiag, which enables flexible online analysis of the network performance in a cross-layer manner, i.e., from the top (application layer) to the bottom (physical layer). We believe HiMoDiag could greatly simplify the process of pinpointing the deficiencies of the Internet access delivery on HSR, lead to more timely optimization and ultimately help to improve the network performance.

1. Introduction

High mobility scenarios have become more and more prevalent in modern society, with transportation systems evolving to accommodate fast-paced lifestyles. High-speed Rail (HSR) exemplifies this trend, particularly in China, which boasts over 37,900 km of HSR – accounting for more than two-thirds of the world's total [1]. HSR services offer competitive advantages in terms of reliability, frequency, and comfort for short to medium-distance trips [2]. In 2019, HSR transported more than 2.2 billion passengers [3]. Despite the efficiency of HSR compared to traditional transportation, passengers often spend hours on trains [2]. To enhance the passenger experience, it is crucial to provide a seamless Internet access that supports work, study, and leisure activities, even under high mobility conditions.

Nonetheless, unstable Internet connections have been a persistent issue for HSR passengers, and seamless high mobility Internet access remains a challenge. High mobility data networking, a relatively new field, has seen its difficulties explored from multiple perspectives [4–7]. The primary source of network instability is attributed to the complex interactions between below-IP (i.e., physical and data link layer) and above-IP (i.e., transport and application layer) layers. Existing tools like Wireshark [8] and MobileInsight [9] focus on either above-IP or below-IP protocols, leaving a gap in the understanding of the interactions between these layers.

Current state-of-the-art methodologies for pinpointing deficiencies across below-IP and above-IP layers involve capturing extensive traces and writing case-by-case scripts to combine analysis results offline [4,7]. While this approach can be effective, it demands substantial domain knowledge and can be inefficient and overly complex for network operators. This complexity hinders the timely identification and resolution of HSR network issues, obstructing prompt optimizations.

In light of the existing limitations, we propose HiMoDiag, a novel TCP-LTE/5 G cross-layer performance analysis tool specifically designed for high mobility data networking. Hi-MoDiag has an experimental platform and provides with a suite of online visualization features. The experimental platform enables users to flexibly control mobile phones as clients, establish connections with servers, and initiate various experiments, including transport-layer bandwidth testing and application-layer video streaming. Building upon the experimental platform, HiMoDiag efficiently extracts and visualizes information from both clients and servers in real-time, which covers an extensive range of performance indicators spanning both below-IP and above-IP layers. Table 1 shows an incomplete list of performance indicators that Hi-MoDiag currently supports for online visualization.

Designing and implementing HiMoDiag involves a series challenges, such as dealing with non-negligible clock differences across layers and endpoints, large numbers of packets and events due to 5 G's high bandwidth, and potential interference caused by sending server-side

---

* Corresponding author.
E-mail address: chenren@pku.edu.cn (C. Xu).

https://doi.org/10.1016/j.hspr.2023.05.002
Received 4 April 2023; Received in revised form 5 May 2023; Accepted 8 May 2023

Table 1
Performance indicators.

	Client	Server
Above-IP	throughput, buffer length, …	congestion window, minRTT, BltBw …
Below-IP	signal strength, resource allocation, handover statistics, …	

performance indicators to the clients during experiments. To address these challenges, we synchronize clocks to clearly demonstrate the relationship of events occurring in different layers and endpoints, implement early filtering and parallel processing for efficient handling of packets and events, and encode performance indicators within the experiment payload to prevent interference with bandwidth-intensive experiments.

To demonstrate the effectiveness of HiMoDiag, we present its user interfaces and showcase real-world examples, illustrating its capability to facilitate the real-time visualization of cross-layer interactions in high mobility scenarios by drawing comprehensive graphs that evolve as experiments progress. We conclude that HiMoDiag empowers network operators with a more efficient means to identify and resolve bottlenecks and inefficiencies, ultimately enhancing network performance.

2. Background and motivation

The public network access for passengers on high-speed trains is enabled by the track-side broadband radio (e.g., LTE, 5 G). To be in-service, mobile clients have to set up connections through the wireless channel to one or more base stations. However, even though base stations are densely deployed along the track, data networking still suffers from a lot of performance degradation because of high mobility. There are two major reasons — poor link quality and frequent handovers.

Link quality becomes poorer and more variable in high mobility. The wireless channel between mobile clients and the base station is unstable since the relative location between the train and the serving cell varies a lot. Meanwhile, the increased mobility level brings large and varying Doppler spread, causing indeterminate carrier frequency offset and inter-carrier interference [10]. Poorer link quality can force the base station to choose lower modulation and coding rate, throttling the upper layers' throughput. Meanwhile, higher coding errors can lead to more packet retransmission at the data link layer, possibly misguiding the upper layers' decisions.

The handover occurs when the mobile client goes out of the cell served by the current base station. Handovers happen frequently, about every 10 s with a speed of 350 km/h, and they become more likely to fail in high mobility. And it may fail in multiple ways: unreliable handover control information transmission, tight time budget for handover completion due to limited overlapped region of cells served by adjacent base stations, and more. Data disruption, caused by handover failure, can pose a long time of disconnection at the transport layer as the effect could be amplified when passed through the network stack.

The problems of high mobility data networking stem not only from low-level instabilities but also from suboptimal responses by upper layers, both of which contribute to a degraded user experience. Previous research has carefully examined this by capturing extensive traces over several months, leveraging a comprehensive understanding of cellular and transport protocols to extract key events and study the relationships between events across different layers. This approach has successfully identified a number of deficiencies at each layer, such as extended data disruptions caused by TCP overreacting to handovers. However, the methodology is inefficient for two reasons: 1) It relies on offline analysis, which, if issues arise, necessitates additional experiments in new iterations, significantly prolonging the diagnostic process, often lasting several months. 2) It demands deep expertize in the full network stack, as events must be manually extracted and correlated. In particular, understanding the LTE/5 G signaling processes themselves require a significant investment of time in learning thousands of pages of 3GPP standards, not to mention analyzing their impact on performance. The need for an out-of-box, real-time, and cross-layer performance analysis tool becomes evident, as it can streamline the diagnostic process and provide valuable insights into the complex interactions between below-IP and above-IP layers.

The advent of 5 G networks introduces further complexity to the high mobility data networking landscape. Rapid deployment of 5 G has been facilitated by reusing LTE's control plane and core network (EPC) in the non-standalone (NSA) deployment scheme. This approach, however, combines LTE as the control plane with 5 G as the data plane, resulting in coexisting LTE and 5 G handovers. Consequently, handover failures could trigger the release of 5 G connections, causing drastic fluctuations in network performance. Alternatively, the 5 G standalone (SA) deployment scheme does not rely on LTE components but demands more effort for implementation. Although 5 G purports to support mobility up to 500 km/h, its real-world performance lags behind the requirements specified in its standards [7]. Given these complexities and the fact that the end-to-end utilization of 5 G's ultra-high bandwidth has yet to be fully realized, the need for a full-ledged real-time performance analysis tool becomes increasingly imperative in order to optimize network performance and user experience.

3. HiMoDiag overview

As depicted in Fig. 1, HiMoDiag is built upon an experimental platform and offers a series of visualization features. In this section, we provide a high-level overview of HiMoDiag and highlight several design challenges.

The experimental platform consists of a controller, clients (mobile phones), and servers (hosted in the cloud). Typically, a laptop serves as

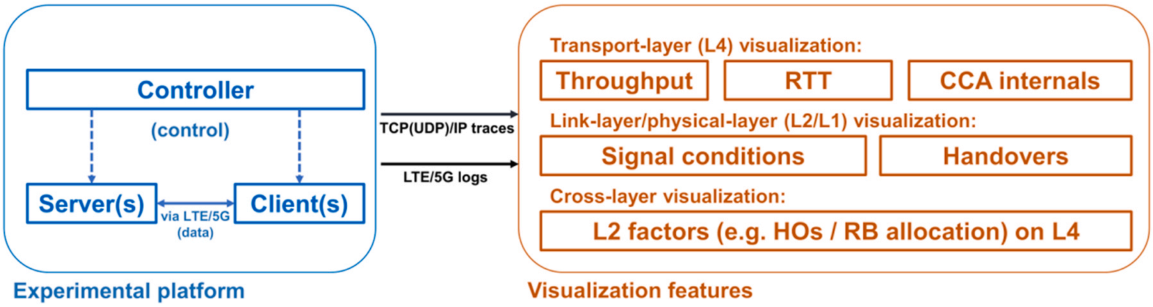


Fig. 1. HiMoDiag Overview. The controller (typically served by a laptop) schedules experiments between clients (mobile phones) and servers (hosted in the cloud), which produces traces and logs to feed a series of visualization features.

the controller. When an experiment is to be conducted, the controller sends control commands to configure clients and servers, enabling clients to establish connections with servers and perform experiments of specified types. Throughout the experiments, packet traces and event logs from both above-IP and below-IP layers on both endpoints are collected, which is done in real-time. HiMoDiag further decodes those traces and logs, extracts relevant performance indicators, and displays them in a cross-layer, cross-endpoint, and online manner. Once the experiments have finished, Hi-MoDiag supports trace replay to see what has happened again. While conceptually straightforward, designing and implementing HiMoDiag involves several challenges:

1) Challenge 1 (C1): The collected packet traces and event logs utilize three distinct clocks for timestamping: the server-side clock for server-side traces, the client-side application processor's clock for client-side above-IP packet traces, and the client-side baseband processor's clock for client-side below-IP event logs. Unsynchronized clocks between above-IP/below-IP layers and client-side/server-side packets make it difficult to coherently display cross-layer and cross-endpoint interactions in a single figure.
2) Challenge 2 (C2): The extremely high bandwidth brought by 5 G results in a substantial number of captured packets and recorded events. Coupled with numerous visualization features covering different layers, endpoints, and performance indicators, this creates a significant processing burden for parsing and decoding packets and events. A naive solution could easily lead to processing times exceeding the experiment duration, violating real-time requirements.
3) Challenge 3 (C3): To perform real-time visualization, it is crucial to obtain all relevant data on the local laptop in a timely manner. While acquiring client-side traces and logs is relatively straightforward, server-side traces and logs contain essential performance indicators, such as the internal states of congestion control algorithms. A simplistic solution for transmitting server-side traces and logs to the client-side during experiments might cause interference between the transmission and experiments, particularly when the experiments are bandwidth-intensive. This interference could result in either delayed indicator visualization or degraded experimental bandwidth.

4. HiMoDiag design

The design of HiMoDiag is illustrated in Fig. 2. In this section, we will elaborate on the design and demonstrate how it addresses the aforementioned challenges.

HiMoDiag interacts with mobile devices using adb through USB cables and communicates with cloud servers via ssh through the Internet (①). Previous work [4], [5] suggests conducting experiments on the laptop by sharing the cellular network through USB tethering, which circumvents restrictions imposed by Android's strict security policies. However, HiMoDiag employs an alternative approach, scheduling experiments directly on our Android devices using the Ubuntu installation provided by Linux Deploy [11] within Android. This method eliminates potential accuracy loss due to USB tethering and prevents USB tethering from becoming a bottleneck when conducting ultra-high bandwidth experiments, as facilitated by 5 G technology.

In client devices (②), HiMoDiag employs tcpdump to capture client-side above-IP packet traces. Considering the significant performance impact arising from below-IP events, we develop diag_logcat and utilize it to communicate with the baseband processor and record client-side below-IP logs. As clients are connected to the laptop via USB cables, all recorded information can be immediately transmitted to the laptop for visualization, incurring minimal or no overhead.

On the server side (③), HiMoDiag uses tcpdump to capture server-side above-IP packet traces. As the sender, the server proactively implements Congestion Control Algorithms (CCAs) to adjust the sending rate and strike a balance between bandwidth and latency. Traditional CCAs are primarily designed for wired networks and often make incorrect decisions in mobile networks. To diagnose CCA-related issues in high mobility networks, we employ modified kernels that log internal parameters of CCAs.

Transmitting the server-side traces and logs to the clients in real-time without interfering with ongoing experiments presents a challenge (C3). Our insight lies in the fact that we typically do not care about the payload of the experiments. For example, during bandwidth testing experiments, the server sends packets with as much randomly generated content as possible to the client to measure the maximum bandwidth. In cases where we need to transmit useful information, we can simply use it as the packet payload. By encoding server-side traces and logs in the experiment payload (④), we avoid potential interference with the running experiments.

Upon receiving these traces and logs on the laptop, Hi-MoDiag features a storage manager (⑤) that organizes them on the local disk. A wide range of data can be generated from various types of experiments, encompassing different access technologies, deployment modes, multiple layers, and both endpoints. As a result, the storage manager is designed to manage these files. Its current implementation employs a path hierarchy to record experiment properties. Alternative implementations, such as leveraging databases, are also possible.

Once the data become ready, HiMoDiag notifies its backend, namely the translation module (⑥), to parse, decode, and analyze the traces and logs. In other words, this module translates many files in heavily-encoded binary formats (e.g., pcap files for captured traces) into concrete datasheets for each relevant performance indicator (e.g., the bandwidth) and high-level events (e.g., handovers). However, the large volume of data presents a challenge for real-time processing (C2). To address this issue, HiMoDiag employs two methods. First, it installs necessary filters during packet capturing and log recording to exclude unnecessary information as early as possible, which minimizes the amount of data being processed. Second, it runs different types of data processing in multiple threads, accelerating the process by leveraging multiple processors. Additionally, the processed datasheets are cached on the disk, enabling near-instant visualization for offline replays.

Another challenge lies in the unsynchronized clocks between the client-side application and baseband processors (between the client and server sides), which impedes precise visualization of cross-layer (cross-endpoint) interactions (C1). Previous work [5] suggests using data-driven alignment, which not only requires extensive processing but also assumes all data is readily available beforehand, rendering it unsuitable for our online processing scenario. Instead of adopting this approach, HiMoDiag proactively synchronizes timestamps in a best-effort manner. On the one hand, HiMoDiag employs the NTP protocol to synchronize client-side and server-side timestamps before initiating experiments. On the other hand, we extend diag_logcat to record the application processor timestamp when it first encounters a baseband log record, and estimate the event time using the recorded application processor timestamp. Although these methods may not be perfectly precise, it is sufficiently accurate for practical use.

As the last step of HiMoDiag's design process, the processed datasheets representing performance indicators or events are delivered to their respective visualization modules (⑦). These visualization modules dynamically update the displayed figures in real-time as the experiments progress. This ensures that users can track changes and make timely adjustments to experiment settings during the course of the experiments. By providing continuous updates to the visualizations, HiMoDiag allows users to monitor the evolving network performance and gain insights into the ongoing interactions between various layers and endpoints.

5. HiMoDiag evaluation

In this section, we showcase the tool's comprehensive performance diagnosis abilities by demonstrating its functionality across three key

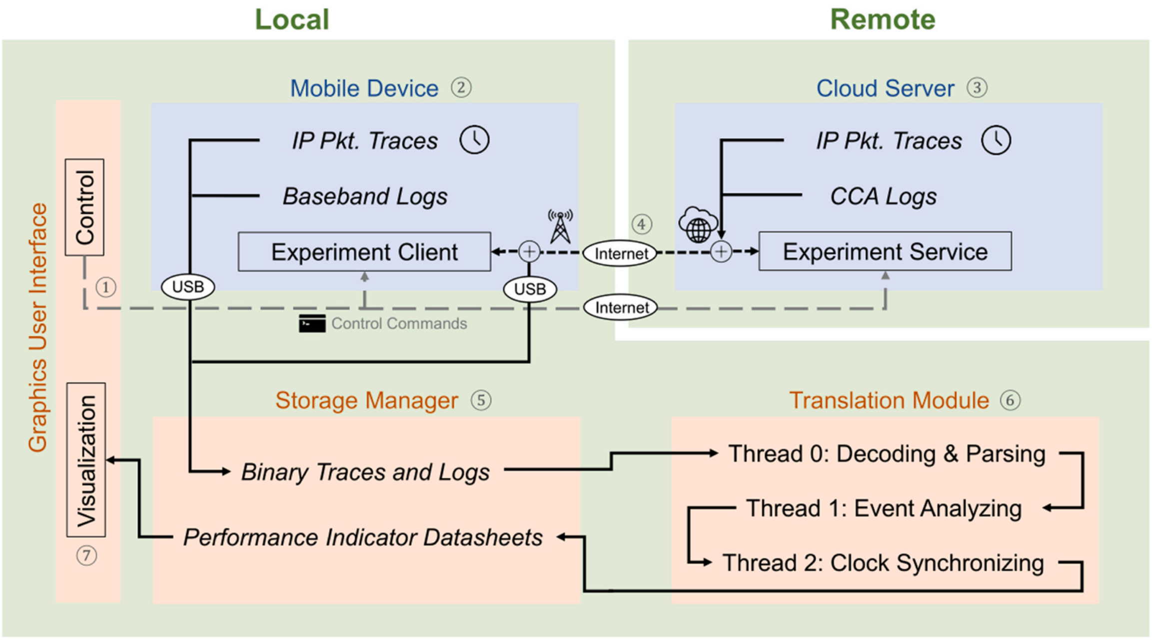


Fig. 2. HiMoDiag Design. The black line with the arrow shows how traces and logs are created, transferred, processed, and finally visualized.

aspects: cross-layer (§V-A), cross-endpoint (§V-B), and application-specific (§V-C) visualization. Subsequently, we assess the performance of HiMoDiag by determining its capability for real-time visualization (§V-D). For trace replays, we use the dataset from [7] (avaliable in https://soar.group/projects/hsrnet/dataset.html).

5.1. *Cross-layer visualization*

The cross-layer visualization component of HiMoDiag offers a comprehensive view of network performance by displaying timeseries figures on the right panel and statistical values on the left panel. In Fig. 3, an example is shown where the experiment has already been completed. Users can examine these figures by dragging the horizontal scrollbar to select a specific time window.

Fig. 3(a) presents throughput timeseries on the right side, alongside key statistics such as average throughput, disruption times, and disruption durations on the left side. Fig. 3(b) demonstrates signal conditions by depicting how Reference Signal Received Power (RSRP) and Signal-to-Noise Ratio (SNR) vary over time, with handover statistics like the number of failed handovers and duration of handovers listed on the left. Lastly, Fig. 3(c) describes channel utilization through Resource Block (RB) utilization and Modulation and Coding Scheme (MCS), along with statistics such as the time ratio of each MCS and the average RB utilization for each MCS. In all three figures, handover events are denoted by marking the corresponding time window between the start and end of handovers in green for successful handovers and red for failed handovers. This helps users understand how signal conditions and resource utilization vary across handovers and how throughput degrades due to handovers. These visualizations together enable a better understanding of the interactions between various layers and facilitate more informed network performance analyses.

5.2. *Cross-endpoint visualization*

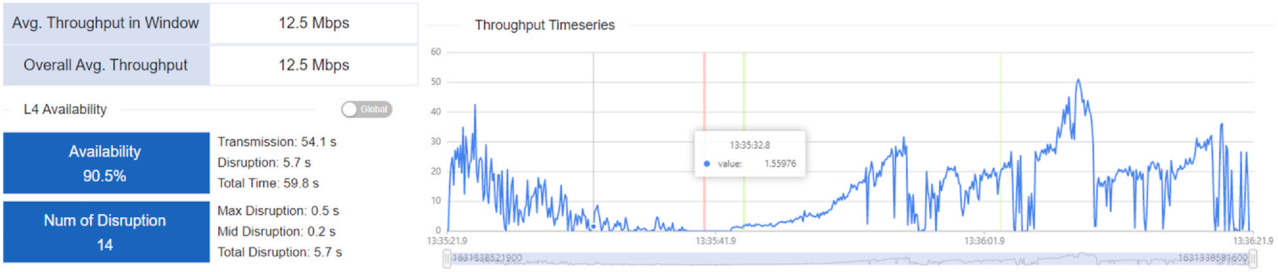
The cross-endpoint visualization component of HiMoDiag offers an integrated perspective of network performance by incorporating server-side performance indicators. In Fig. 4, an example is showcased where an experiment is in progress. With a layout similar to the cross-layer visualization, it displays the timeseries figure within a time window of the most recent 10 s, as the experiment is ongoing. On the right side, server-side performance indicators, such as the congestion window and RTT measured by the server, are depicted alongside their variations. The statistical data on the left side includes the congestion window, bottleneck band-width, and RTTs, revealing the inner workings of congestion control algorithms. By combining these server-side indicators together with client-side indicators (as shown earlier in §V-A), HiMoDiag helps to perform more comprehensive diagnostics, bridging the gap between client and server endpoints.

5.3. *Application-specific visualization*

The application-specific visualization component of Hi-MoDiag is tailored to display performance indicators for specific application-layer experiments. Currently, HiMoDiag supports video streaming experiments, with potential future extensions to other application types. As shown in Fig. 5, similar to the cross-layer visualization, this component displays a completed experiment with pertinent performance indicators.The right panel presents the variations of application-specific performance indicators, including buffer length and video bitrate. Meanwhile, the left panel displays statistics about video quality and rebuffering, such as average bitrate, smoothness, and rebuffering duration. These visualizations enable users to assess application-specific performance and identify potential bottlenecks, ensuring optimal end-to-end user experience in high mobility data networking scenarios.

5.4. *Real-time visualization*

Ideally, achieving strict real-time visualization requires the time spent on data decoding, analyzing, and visualizing (referred to as visualization time hereafter) to be less than the experiment duration. However, in practice, it is still acceptable for the visualization time to be slightly longer than the experiment duration without significantly impacting the diagnosis flexibility. In the case of HiMoDiag, we tested 60-second experiments, and the median visualization time was 77.3 s. The 17.3-second delay is due to the large amount of captured data and limited processing capability. Though HiMoDiag has not yet achieved strict real-time performance, this result demonstrates that it is capable



Avg. Throughput in Window	12.5 Mbps
Overall Avg. Throughput	12.5 Mbps

L4 Availability

Availability 90.5%	Transmission: 54.1 s Disruption: 5.7 s Total Time: 59.8 s
Num of Disruption 14	Max Disruption: 0.5 s Mid Disruption: 0.2 s Total Disruption: 5.7 s

(a) Throughput.

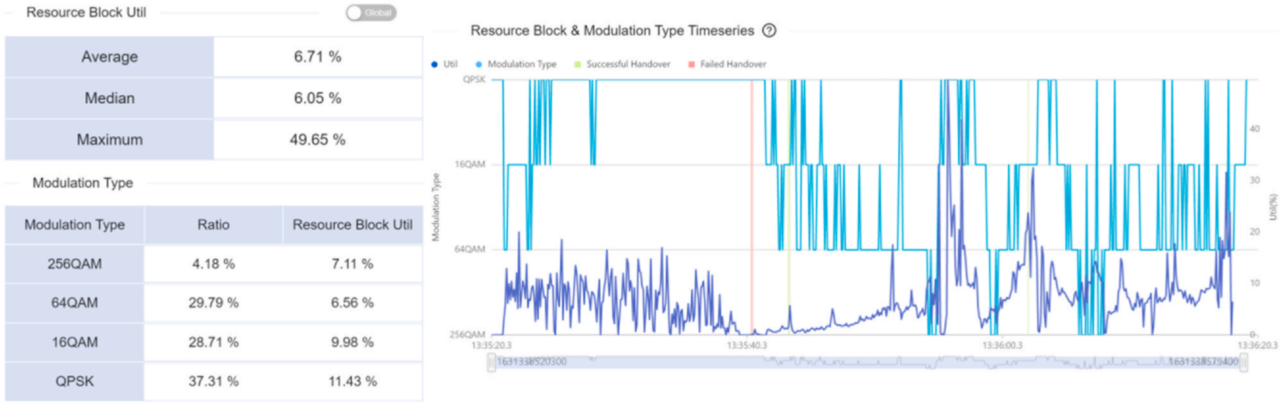
L2 Availability

LTE Connection Time	0 s
5G Connection Time	59.2 s
Total Connection Time	59.2 s

Handover

Frequency 3	# of Successes: 2 # of Failures: 1 # of RLFs: 1
Total Duration 0.219 s	Max Duration: 0.05 s Mid Duration: 0.052 s Min Duration: 0.117 s

(b) Signal conditions.



Resource Block Util

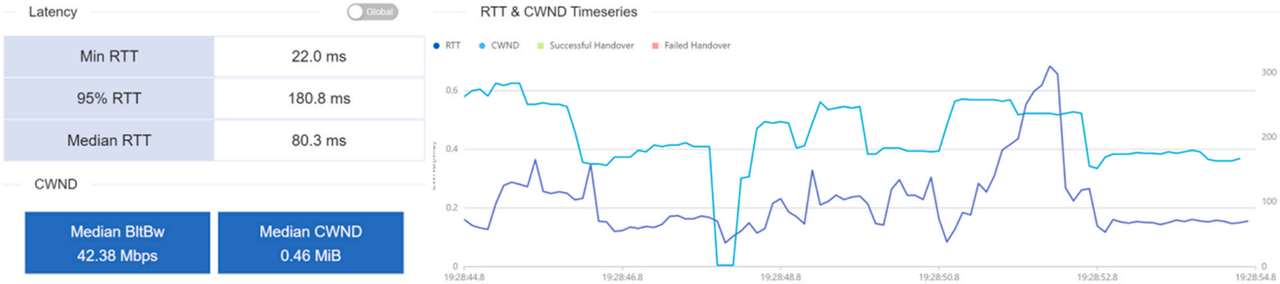
Average	6.71 %
Median	6.05 %
Maximum	49.65 %

Modulation Type

Modulation Type	Ratio	Resource Block Util
256QAM	4.18 %	7.11 %
64QAM	29.79 %	6.56 %
16QAM	28.71 %	9.98 %
QPSK	37.31 %	11.43 %

(c) Channel utilization.

Fig. 3. Cross-layer visualization.



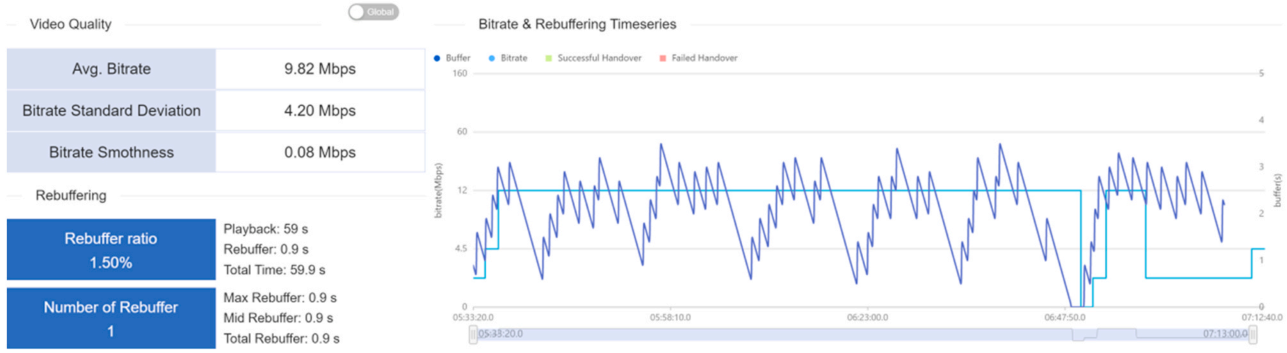
Latency

Min RTT	22.0 ms
95% RTT	180.8 ms
Median RTT	80.3 ms

CWND

Median BltBw 42.38 Mbps	Median CWND 0.46 MiB

Fig. 4. Cross-endpoint visualization.



Video Quality

Avg. Bitrate	9.82 Mbps
Bitrate Standard Deviation	4.20 Mbps
Bitrate Smothness	0.08 Mbps

Rebuffering

Rebuffer ratio 1.50%	Playback: 59 s Rebuffer: 0.9 s Total Time: 59.9 s
Number of Rebuffer 1	Max Rebuffer: 0.9 s Mid Rebuffer: 0.9 s Total Rebuffer: 0.9 s

Fig. 5. Application-specific visualization.

of near real-time visualization in practice, which is generally sufficient for practical applications in common scenarios.

6. Conclusion

In this paper, we present HiMoDiag, a TCP-LTE/5 G cross-layer performance analysis tool for high mobility data networking. Integrated with an experimental platform, it enables the execution of experiments in an automatic, reliable, and efficient manner. HiMoDiag is equipped with the following real-time analysis functions: 1) cross-layer visualization to understand performance degradation due to poor signal conditions, low channel utilization, and frequent handovers; 2) cross-endpoint visualization to examine performance degradation arising from suboptimal decisions made by server-side congestion control algorithms, which may not be designed to account for the unique network characteristics of high mobility data networking; 3) application-specific visualization to assess how end-to-end performance is impacted by application-specific decisions. We believe HiMoDiag significantly contributes to network optimization in HSR contexts by streamlining the network diagnostic process.

Acknowledgment

This work was supported by National Key Research and Development Plan, China (Grant No. 2020YFB1710900) and National Natural Science Foundation of China (Grant No. 62022005 and 62172008).

References

[1] Wikipedia, High-speed rail - wikipedia. 2021. [Online]. Available:https://en.wikipedia.org/wiki/High-speed\rail.
[2] The World Band, China's high-speed railway development. 2019. [Online]. Available: ⟨https://documents1.worldbank.org/curated/en/697391567777005065/pdf/Chinas-High-Speed-Rail-Development.pdf⟩.
[3] Statista, China: passenger transport volume of high speed rail. 2019. [Online]. Available: ⟨https://www.statista.com/statistics/1120071/china-passenger-transport-volume-of-high-speed-rail/⟩.
[4] J. Wang, Y. Zheng, Y. Ni, et al., An active-passive measurement study oftcp performance over lte on high-speed rails. In: Proceedings of the the 25th Annual International Conference on Mobile Computing and Networking, (2019) 1–16.
[5] C. Xu, J. Wang, Z. Ma, et al., A first look at disconnection-centric tcp performance on high-speed railways, IEEE J. Sel. Areas Commun. 38 (12) (2020) 2723–2733.
[6] D. Xu, A. Zhou, X. Zhang, et al., Understanding operational 5G: A first measurement study on its coverage, performance and energy consumption. In: Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication, (2020) 479–494.
[7] Y. Pan, R. Li, C. Xu, The first 5G-lte comparative study in extreme mobility, Proceedings of the ACM on Measurement and Analysis of Computing Systems (2022).
[8] The Wireshark team, "Wireshark," 2021. [Online]. Available: ⟨https://www.wireshark.org/⟩.
[9] Y. Li, C. Peng, Z. Yuan, et al., Mobileinsight: Extracting and analyzing cellular network information on smartphones. In: Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking, (2016) 202–215.
[10] Y. Yang, P. Fan, Y. Huang, Doppler frequency offsets estimation and diversity reception scheme of high speed railway with multiple antennas on separated carriages. In: Proceedings of the 2012 International Conference on Wireless Communications and Signal Processing (WCSP), (2012) 1–6.
[11] meefik, Linux deploy 2021. [Online]. Available: ⟨https://github.com/meefik/linuxdeploy⟩.