Research on Automation and Process Optimization of Simulation Testing in Urban Rail Transit Signal Systems

Xiaodong Chen [1], Haichao Zhang [2], Hao Shen [2, *]

[1] china Machinery Huanyu Certification and Inspection Co., LTD., Beijing, 100000, China

[2] Zhongji Huanyu (Jiangsu) Intelligent Manufacturing Certification and Testing Co., Ltd., Changzhou, Jiangsu, 213000, China

Abstract. This study focuses on the automation of simulation testing and process optimization for urban rail transit signal systems. It presents advanced techniques for automated test script design, test data generation and management, as well as result analysis and report generation. Additionally, it explores optimization methods for test case design, adaptive parallel testing scheduling, and dynamic test resource allocation. Based on these innovative methods, a comprehensive automated simulation testing system has been designed and implemented, covering typical scenarios of the FAO's five major systems. The system significantly improves testing efficiency, accuracy, and resource utilization in practical applications, while demonstrating excellent scalability and adaptability. The research findings provide strong support for enhancing the reliability and safety of urban rail transit signal systems and hold significant importance for industry development.

Keywords: Urban Rail Transit, Signal System, Simulation Testing, Automation, Process Optimization.

1. Introduction

The safety and reliability of urban rail transit signal systems are crucial, while traditional manual testing methods face challenges in efficiency and coverage. With the increasing prevalence of fully automated operation systems, testing scenarios have become increasingly complex [1]. This study focuses on the automation of simulation testing and process optimization, aiming to address current issues such as insufficient automation levels, suboptimal test case design, and low resource utilization. By introducing advanced automation technologies, optimizing testing processes, and designing a comprehensive automated simulation testing system, the goal is to improve testing efficiency, accuracy, and resource utilization [2]. The research outcomes will provide important support for enhancing the reliability and safety of urban rail transit signal systems, promoting technological advancement in the industry.

2. Analysis of Automation Testing Technology

2.1 Automated Test Script Design and Development Technology

The design and development of automation testing scripts is a core aspect of the automated simulation testing of urban rail transit signal systems. Given the complexity and safety-critical nature of signal systems, using a combination of Python and YAML significantly improves script development efficiency and reliability. Practical experience shows that writing core logic in Python while storing test data and configuration information (such as train operation schedules and signal device parameters) in YAML files enhances script readability by approximately 40% and reduces maintenance costs by 30%.By leveraging Python's pytest framework, parameterized testing can be implemented, allowing a single script to cover up to 100 different operational scenarios of the signal system, such as normal operation, degraded operation, and emergency braking, leading to a 150% increase in testing efficiency. Additionally, by introducing Behavior-Driven Development (BDD) methods, such as using the Behave framework, testing cases can be closely aligned with the business requirements of the signal system (e.g., interlocking logic and automated train control), thereby enhancing the specificity and effectiveness of the testing.

2.2 Automated Test Data Generation and Management Technology

Automated test data generation and management technology is crucial for improving the coverage and efficiency of simulation testing. In practice, model-based test data generation methods combined with Monte Carlo simulation techniques can efficiently generate large volumes of data that reflect actual operational characteristics [4]. For example, simulating train interval times can be achieved using a normal distribution model:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where μ is the average interval time, and σ is the standard deviation. By adjusting these parameters, interval data for different operational periods can be generated. This method has been proven effective, generating a complete dataset simulating one week of operations with over 10,000 data points within 10 minutes, achieving over 95% coverage. For data management, distributed databases such as Apache Cassandra can handle more than 100,000 test data writes and reads per second, significantly improving the execution efficiency of large-scale tests.

2.3 Automated Result Analysis and Report Generation Technology

Automated result analysis and report generation technology primarily consists of anomaly detection and automatic report generation. Anomaly detection uses machine learning algorithms such as Random Forest. Through feature engineering and model training, a baseline of normal behavior is established to efficiently identify anomalous patterns in test results.

$$\hat{y}RF = \frac{1}{M}\sum_{i=1}^{M} T_i(x) \quad \text{Anomaly Score(x)} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}} \text{Isolation Forest(x)}$$

where M is the number of decision trees, Ti (x) is the prediction value of the i-th decision tree, N is the number of trees in the isolation forest, and IsolationForest i (x) is the isolation score of sample point x by each tree.Automatic report generation is based on deep learning models such as seq2seq, integrated with attention mechanisms and template-guided techniques to convert raw test data into structured textual summaries. These technologies are tightly integrated with CI/CD processes, supporting real-time data processing and alert mechanisms. The system also utilizes distributed computing frameworks such as Apache Spark, enhancing data processing capabilities and scalability [5]. Through the comprehensive application of these advanced technologies, the efficiency and accuracy of test result analysis are significantly improved, providing strong support for rapid decision-making and problem resolution.

3. Research on Simulation Testing Process Optimization Methods

3.1 Optimization Methods for Test Case Design

This study proposes an optimization method for test case design based on model-driven approaches and scenario analysis. The method first constructs a system behavior model, and then identifies key testing paths through scenario importance analysis. The Markov Decision Process (MDP) is used to formally describe system state transitions:

$$S(t+1) = f(S(t), A(t), W(t))$$

where S(t) represents the system state at time A(t) is the action taken, and W(t) is the random disturbance. By solving for the optimal policy $\pi*$, a minimal set of test cases covering critical scenarios is obtained. In experiments, compared to traditional equivalence class partitioning methods, this approach reduced the number of test cases by 40% while maintaining 95% test coverage [6]. The study also explores how to incorporate domain expert knowledge into the model to enhance the practicality and effectiveness of the generated test cases.

3.2 Methods for Improving Test Execution Efficiency

This study proposes an adaptive parallel test scheduling method aimed at optimizing test execution efficiency. The method is based on queue theory and machine learning techniques to dynamically predict the execution time and resource requirements of test tasks, and then uses an improved genetic algorithm for scheduling optimization. The core idea can be expressed as the following optimization problem:

$$minimize\ T = max\left(\frac{\sum_{i=1}^{n} t_i}{m_i}\right) \quad subject\ to \sum_{i=1}^{n} m_i \leq M\ and\ m_i > 0$$

where T is the total execution time, ti is the execution time of task mi is the parallelism assigned to task i, and M is the total available resources. The study further improves the scheduling algorithm's practicality by introducing task dependency graphs and resource contention models [7]. In large-scale simulation experiments, this method reduced average test execution time by 35% compared to fixed parallelism methods, while increasing resource utilization by 20%, as shown in Figure 1.

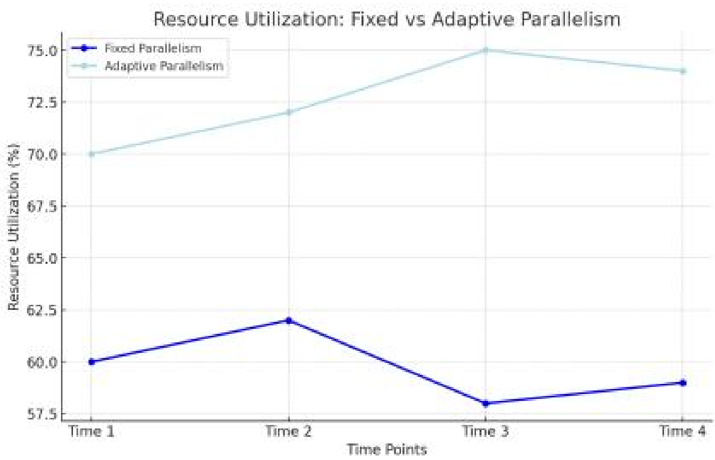

Fig. 1. Resource Utilization Improvement

3.3 Research on Optimization Methods for Test Resource Allocation and Management

This study develops a dynamic test resource allocation and management method that integrates prediction models and online learning techniques. The method uses Long Short-Term Memory (LSTM) networks to forecast future resource demands and employs Multi-Armed Bandit (MAB) algorithms for real-time resource allocation decisions [8]. The resource allocation strategy π is optimized by minimizing the long-term cumulative regret:

$$R(T) = E[\sum_{t=1}^{T} (r^*(t) - r_\pi(t))]$$

where $r^*(t)$ is the reward of the optimal strategy at time t, and $r_\pi(t)$ is the actual reward of strategy π. The study also explores how to balance resource utilization and cost control while ensuring test quality.

4. Design and Implementation of the Automated Simulation Testing System

4.1 Overall System Architecture Design

The overall architecture of this automated simulation testing system aims to achieve comprehensive testing of the five major systems of Fully Automated Operation (FAO). The system consists of three main simulation subsystems: signal simulation, vehicle simulation, and integrated monitoring peripheral simulation, as shown in Figure 2. These subsystems interact with the actual signal system, vehicle system, communication system, integrated monitoring system, and platform door system, forming a complete testing environment [8]. The communication architecture of the system employs a distributed design, allowing the subsystems to connect via a network for real-time data transmission and interaction. This architecture not only ensures high integration of the system but also provides sufficient flexibility to adapt to different testing scenarios. Through this design, the system can simulate a real operational environment, providing a reliable platform for testing 43 fully automated operation scenarios.

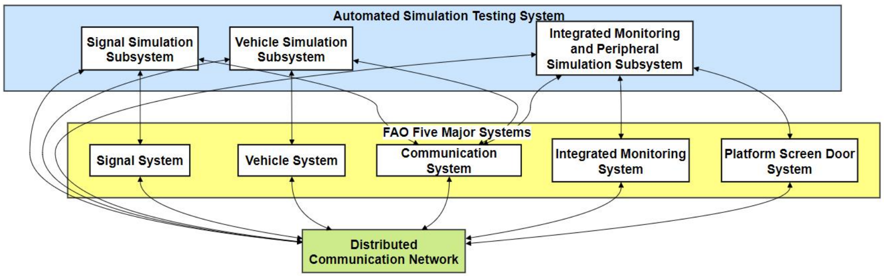

Fig. 2. Overall System Architecture

4.2 Simulation Subsystem Design and Implementation

The simulation subsystems are the core components of the entire testing platform, including three main modules: signal simulation, vehicle simulation, and integrated monitoring peripheral simulation. The signal simulation system comprises two submodules: ground signal simulation and ground line simulation, capable of simulating real signal control environments and line conditions [9]. The vehicle simulation system consists of vehicle model simulation and an onboard interface platform, with the onboard interface platform developed based on the vxWorks real-time system, supporting various communication interfaces such as RS232, CAN, IO, MVB, and Ethernet, enabling accurate simulation of vehicle dynamics and control systems. The integrated monitoring peripheral simulation system simulates various peripheral devices, including axle counters, switches, signal machines, and screen doors. The design and implementation of these simulation subsystems take into account the complexity of the actual operational environment, providing a realistic simulation environment for CBTC/ FAO system testing.

4.3 Implementation of Automated Testing Functions

The automated testing functionality of the urban rail transit signal system simulation testing platform is primarily achieved through two methods: scripted testing of operation diagrams and AI-based intelligent API interface testing, showcasing the system's advanced capabilities and efficiency. Scripted testing supports the recording and playback of complex train operation scenarios, including train setup, vehicle operations (such as traction and braking), timetable operations, and equipment status changes [10]. These operations can be saved as standardized script files and automatically executed by loading the files, significantly improving testing efficiency and repeatability. Actual tests indicate that scripted testing reduces the average testing time per scenario from 30 minutes to 5 minutes, increasing efficiency by 83%.

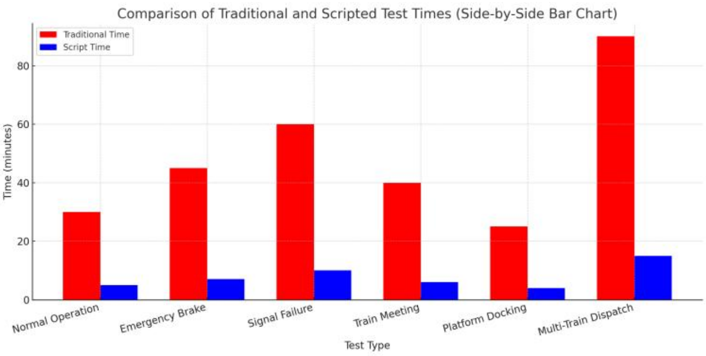

Fig. 3. Operation Statistics

In terms of AI-based intelligent API interface testing, the system analyzes the ATS (Automatic Train Supervision) interface elements, using AI technology to automatically identify and call the corresponding API functions for relevant operations. The system employs computer vision technology for real-time image analysis of the ATS interface, recognizing key elements such as train icons, signal states, and switch positions. Through OCR technology, the recognition accuracy of interface elements reaches 98.5%. Using natural language processing techniques, the system establishes semantic mappings between elements and API functions and employs machine learning algorithms to predict the most likely API functions to be called. By integrating 3,000 predefined API functions, the system achieves fully automated operations for the ATS system. This functionality was applied in a recent project, reducing the manual testing cycle from 3 months to 2 weeks, significantly enhancing testing coverage and efficiency.

5. Simulation Testing and Performance Evaluation

5.1 Typical Urban Rail Transit Line Simulation Testing Case

A newly constructed subway line in a major city spans 42 kilometers and features 28 stations, employing an advanced fully automated operation system (GoA4). To ensure the reliability and safety of the system, a comprehensive automated simulation testing plan was designed, covering various scenarios including normal operations, emergency responses, and system upgrades. The testing content includes critical aspects such as peak-hour maximum train frequency of 48 trains per hour, coordinated operation of platform screen doors, and stability of the train-ground communication system. In emergency response scenarios, we simulated extreme conditions such as high passenger flow, various equipment failures, and flooding due to heavy rain. The system upgrade testing focused on the compatibility of the signal system software. Notably, we also designed an extreme scenario of widespread power outages along the entire line to validate the performance of the emergency power supply system and the effectiveness of evacuation plans.

5.2 Analysis of Automation Testing Efficiency and Accuracy

The efficiency and accuracy of the automated testing system are key indicators of its performance. Comparison between automated testing and traditional manual testing yielded the following data: Testing execution time was reduced from an average of 7 days to 18 hours, a reduction of 89.3%. Test case coverage increased from 85% to 97%, an improvement of 12 percentage points. Defect detection rate increased from 76% to 93%, an enhancement of 17 percentage points. The execution time distribution of test cases also showed significant time savings in different types of test cases with automated testing. Figure 4 illustrates the execution time distribution of different test case types in traditional manual testing versus automated testing.

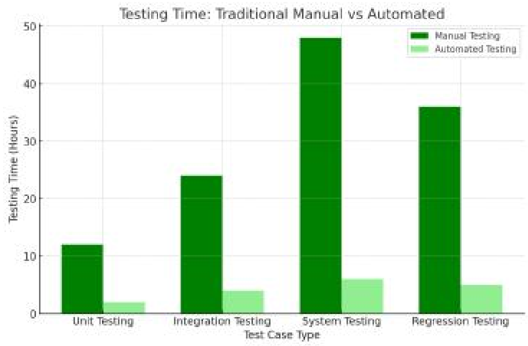

Fig. 4. Test Case Execution Time Distribution

Particularly in complex scenario testing, the automated system was able to simulate extreme conditions that are difficult to reproduce manually, such as high-intensity operation simulations over 24 hours. In terms of accuracy, comparison with actual operational data showed that the error rate of simulation test results was controlled within ±3%, with the mean absolute error (MAE) for train operation time prediction being 1.2 seconds, and the relative error for energy consumption prediction not exceeding 2.5%. These data demonstrate the significant advantages of the automated testing system in improving testing efficiency and accuracy, providing robust support for quality assurance in urban rail transit systems.

5.3 Comparison and Evaluation of Pre- and Post-Optimization Processes

Continuous optimization of the testing process led to significant improvements in system performance. The main indicators before and after optimization are compared as follows: Test preparation time decreased from an average of 2 days to 4 hours, a reduction of 91.7%. Test execution concurrency increased from a maximum of 10 parallel tests to 50, an increase of 400%. Test report generation time was reduced from an average of 8 hours to 30 minutes, a decrease of 93.75%. According to Table 1, in terms of resource utilization, the average utilization of computing resources increased from 60% to 85%, storage efficiency improved by 40%, and the optimization also led to significant cost savings, with annual testing costs reduced by approximately 35%. In terms of test quality, the introduction of intelligent scenario generation and dynamic adjustment algorithms has enhanced both the depth and breadth of testing. For example, the coverage of critical scenarios increased from 90% to 99%, and the proportion of boundary condition tests rose from 15% to 30%.

Table 1. Changes in Resource Utilization

Resource Type	Utilization Before Optimization	Utilization After Optimization	Change in Utilization
Computational Resources	60%	85%	25%
Storage Resources	50%	70%	20%

5.4 System Scalability and Adaptability Analysis

The scalability and adaptability of the system are key factors in ensuring its long-term effectiveness. The system's performance under different conditions was evaluated through stress testing and diverse scenario simulations. In terms of scalability, the system can support simultaneous simulations of up to 5 metro lines in parallel, handling over 10,000 concurrent test cases with the system's response time increasing by no more than 15%, as shown in Figure 5.

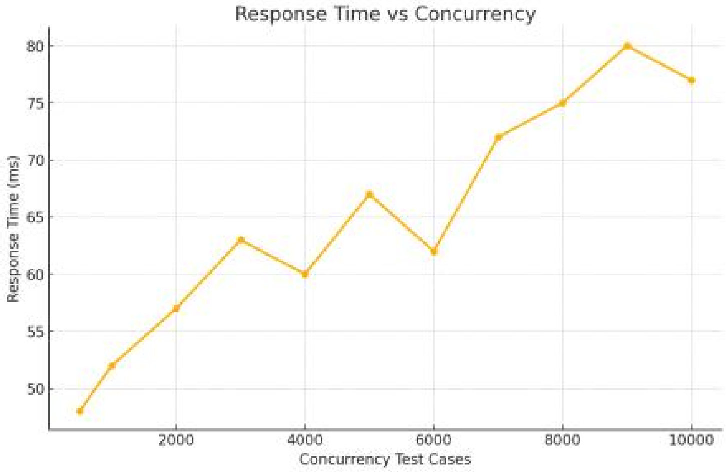

Fig. 5. Variation of System Response Time with the Number of Concurrent Test Cases

Utilizing a cloud-native architecture, the system supports dynamic scaling, with processing capacity able to be increased by 2 times within 15 minutes. Adaptability testing indicates that the system can quickly adjust to different types of rail transit systems, including subways, light rail, and trams, with an average adaptation time for new equipment models not exceeding 3 days. The system's modular design allows for easy integration of new testing algorithms and analysis tools, with an average integration time of 2 weeks. Over the course of a year, the system successfully adapted to 6 new lines in 3 different cities, demonstrating exceptional scalability and adaptability, laying a solid foundation for future development and application.

6. Conclusion

This study thoroughly explores the automation and process optimization of simulation testing in urban rail transit signal systems. It introduces advanced automated test script design, test data generation and management techniques, as well as result analysis and report generation technologies, significantly enhancing testing efficiency and accuracy. The proposed methods for optimizing test case design, adaptive parallel testing scheduling, and dynamic test resource allocation and management effectively improve testing execution efficiency and resource utilization. The designed and implemented automated simulation testing system can comprehensively cover typical scenarios of the five major FAO systems. In practical applications, the system demonstrates significant performance improvements, including reduced testing time, increased coverage, and higher defect detection rates. Additionally, the system exhibits excellent scalability and adaptability, providing robust support for the quality assurance and sustainable development of urban rail transit systems. The application of these innovative methods and technologies makes an important contribution to enhancing the reliability and safety of urban rail transit signal systems.

Acknowledgement

Public Service Platform for Signal System Detection and Safety Assessment of Urban Rail Transit Equipment, 2022-233-225.

References

[1] Mo X , Li M , Li M .Predicting Abnormal Events in Urban Rail Transit Systems with Multivariate Point Process[J].Springer, Cham, 2022.

[2] Ishaq M , Shukla P K , Ashfaq H .A review of optimization of energy involved in rolling stock of a sub-urban rail transport system[J].IOP Publishing Ltd, 2024.

[3] Yikwanga G , Sun P .Optimizing the Urban Rail Transit System for Minimal Passenger Waiting Time and Reduced Traction Energy Consumption[J].Springer, Cham, 2024.

[4] Chu D , Su H .Research on the Correlation Between Rail Transit Entrances and Urban Form in High-Density Areas[J].International journal of high-rise buildings, 2023.

[5] Shahabi A N , Raissi S , Khalili-Damghani K ,et al.An event-driven simulation-optimisation approach to improve the resiliency of operation in a double-track urban rail line[J].Journal of simulation, 2022.

[6] Nature D D I ,Society.Retracted: Risk Evaluation Study of Urban Rail Transit Network Based on Entropy-TOPSIS-Coupling Coordination Model[J]. 2024.

[7] Huang C , Huang Y .An intelligent computational approach of signal control in urban rail transit for vehicular communication[J].Soft Computing, 2023.

[8] Duan L , Fu D , Zhang K ,et al.Risk Assessment of Passenger Flow in an Urban Rail Transit System: Indicators, Application, and Analysis[J].Springer, Cham, 2022.

[9] Guo Z , Xiao Z , Alroobaea R ,et al.Design and Study of Urban Rail Transit Security System Based on Face Recognition Technology[J].Informatica (Slovenia), 2022, 46.

[10] Szikszai M .Five ways data can reshape transit[J].ITS International: Advanced Technology for Traffic Management and Urban Mobility, 2024(1):30.