

## 城市轨道交通 CBTC 系统实时故障定位方法\*

白广争 郑泽熙 冯浩楠 许展璵 刘海祥 范 楷

(中国铁道科学研究院集团有限公司通信信号研究所, 100081, 北京)

**摘 要** [目的] 为了实现城市轨道交通 CBTC(基于通信的列车控制)系统的快速故障诊断和定位,针对系统网络监测数据,提出一种基于时间约束的实时故障监测定位方法。

[方法] 介绍了所提 CBTC 系统实时故障定位方法的设计思路;提出了基于时间约束的故障定位算法;构建了 CBTC 数据监测系统,并进行了算法验证。[结果及结论] 通过定义故障匹配模式,构建故障模式库,描述模式匹配规则并定义模式相似度;考虑 CBTC 系统中各子系统的通信周期、故障响应时间等功能需求方面的差异性,以及多系统之间的数据处理存在同步问题等,加入时间约束因素,时间约束以外的字符不能参与故障模式匹配;利用每次循环匹配失败点的位置,通过继承已匹配成功字符的信息,提高下一次匹配的效率,避免算法的回溯搜索,提高模式匹配效率。算法验证结果表明,所提算法能够快速、准确地定位系统中实时发生的故障。

**关键词** 城市轨道交通; CBTC 系统; 实时故障定位

**中图分类号** U284.48

DOI:10.16037/j.1007-869x.2024.02.021

## Real-time Fault Locating Method for Urban Rail Transit CBTC System

BAI Guangzheng, ZHENG Zexi, FENG Haonan, XU Zhanying, LIU Haixiang, FAN Kai

(Signal and Communication Research Institute of China Academy of Railway Sciences Group Co., Ltd., 100081, Beijing, China)

**Abstract** [Objective] To carry out rapid fault diagnosis and locating for urban rail transit CBTC (communication-based train control) system, a real-time fault monitoring and locating method based on time constraints is proposed, focusing on system network monitoring data. [Method] The design concept of the proposed real-time fault locating method for the CBTC system is introduced. An algorithm based on time constraints for fault locating is proposed. A CBTC data monitoring system is constructed, and the algorithm is validated. [Result & Conclusion] By defining fault matching patterns, a fault pattern library is constructed, pattern matching rules are described, and

pattern similarity degree is defined. Considering the differences in functional requirements, such as communication cycles and fault response time among various sub-systems in the CBTC system, and the synchronization issues in data processing among multiple systems, the time constraint factors are introduced, and characters beyond the specified time constraints are excluded from participating in fault pattern marching. Through utilizing the position of each failed matching point in the cycle and inheriting information of characters from successful matches, the efficiency of the next match is enhanced and backtracking searches by the algorithm are avoided, improving overall pattern matching efficiency. The algorithm validation results demonstrate that the proposed algorithm can rapidly and accurately locate real-time faults occurring in the system.

**Key words** urban rail transit; CBTC system; real-time fault locating

CBTC(基于通信的列车控制)系统由 ATS(列车自动监控系统)、CI(计算机联锁)系统、ZC(区域控制器)系统、VOBC(车载控制器)系统和 DCS(数据通信子系统)共同构成,系统构成相对复杂。在运行过程中,若有某一模块发生故障,往往会有多个子系统报警,仅从系统表现出来的故障现象较难确定故障问题来源。不同子系统通过自身逻辑判断提供一定的报警信息,此时需要有经验的维护人员归纳这些信息,同时结合子系统之间实时传输的接口数据信息进行综合判断,才能确定故障发生的来源。这不仅需要维护人员耗费较长的时间进行问题分析,而且其自身也需具有较高的技术水平。因此,设计一套针对 CBTC 系统进行实时故障诊断定位的分析系统具有重要的应用价值。

将模式匹配应用在故障定位领域是一种有效的方法。文献[1]研究了在设备资源受限情况下,基于云服务器辅助计算策略,完成对模式端和数据库端两方的安全模式匹配协议设计,并通过设计功

\* 中国铁道科学研究院集团有限公司科技研究开发计划项目(2021YJ100)

能函数实现对模式的检测。文献[2]通过对文本关键信息进行模糊化预处理,基于WM(Wu-Manber)算法进行中文信息的初步匹配,最后进行多模式匹配。文献[3]将模糊理论应用于网络故障的实时故障监测,结合模糊理论与数据挖掘方法,建立关联规则知识库,再利用模糊匹配对告警信息进行模糊推理,以达到定位故障的目的。

在进行故障模式匹配过程中,现有文献研究均没有考虑时间约束对故障模式匹配的影响。而在实际应用中,从通信网络上采集到的多方实时数据需要考虑其时间要求,即满足时间要求的数据才能参与模式匹配,传统的字符串模糊匹配算法难以解决该问题。鉴于此,针对系统网络监测数据,本文提出一种基于时间约束的实时故障监测定位方法,以实现城市轨道交通基于CBTC系统的快速故障诊断和定位。本文研究可为CBTC系统的集成测试、故障运维等提供借鉴。

## 1 CBTC系统实时故障定位方法设计思路

假设在某一时刻,CBTC系统突发故障,针对各子系统的报警信息分别构建数据序列,例如ZC报警数据序列、VOBC报警数据序列、CI报警数据序列、各子系统之间实时通信数据序列等。当特定故障发生时,故障诊断系统结合预先制定好的故障模式库中的不同故障模式,利用故障识别定位算法进行故障模式匹配。在设计模式匹配算法过程中,需要考虑时间约束的问题:

1) 同一数据序列内信息之间的时间约束性。对于同一数据序列,将其中的数据信息与故障的对应子模式进行匹配时,模式中各字符的出现时机与顺序只有满足给定的时间约束,才被认为该故障子模式匹配成功;不满足时间约束的字符,认为其并不处于同一故障状态下,则不满足模式匹配规则。通信周期越长、通信数据量越大,则待匹配字符之间的干扰字符数量越大,数据序列与故障模式之间的ED(编辑距离)越大<sup>[4-5]</sup>;反之,则数据序列与故障模式之间的ED越小。

2) 不同数据序列之间相关信息的同步性。对于CBTC系统,当从不同子系统分别获得数据信息后,会形成多个数据序列。不同数据序列内的信息在参与某一特定故障模式匹配时,只有同一时期内出现的信息才能进行匹配。不同时期产生的信息,即使能够匹配成功,但不具有实际意义。“同一时

期”的定义是由系统的响应周期、通信周期等参数共同决定的。例如,当其中一个子系统发生故障后,另一个子系统的响应时间为 $t$ ,则进行该故障模式匹配时, $t$ 时间域内的信息均属于“同一时期”。

根据以上分析,设计基于时间约束的故障识别定位算法,以提高CBTC系统的实时故障快速监测定位与现场恢复能力。

## 2 基于时间约束的故障定位算法

### 2.1 模式定义及匹配规则

根据实际问题,对模式定义及匹配规则进行说明。

#### 2.1.1 总模式和子模式

1次完整的匹配对应1个总模式,总模式匹配成功,即完成了模式对象的匹配过程。1个总模式由 $K$ 个子模式组合而成,第 $n$ 个总模式的第 $j$ 个子模式称为 $s_{n,j}$ ( $1 \leq n \leq N, 1 \leq j \leq K$ ),该子模式对应一个实时数据序列,每个总模式的 $K$ 个子模式对应 $K$ 个实时数据序列。子模式 $s_{n,j}$ 中的字符在数据序列中出现时均具有时间属性,出现的时机和顺序受时间约束。多个总模式共同构成模式库。

#### 2.1.2 子模式 $s_{n,j}$ 匹配规则

子模式 $s_{n,j}$ 匹配规则为:① $s_{n,j}$ 由 $x_{n,j}$ 个报警信息组成;② $s_{n,j}$ 中的各报警信息在数据序列中可以不相邻;③ $s_{n,j}$ 中的各报警信息在数据序列中必须按顺序且全部出现;④ $s_{n,j}$ 中的第 $m$ 个报警信息 $s_{n,j,m}$ 与第 $m+1$ 个报警信息 $s_{n,j,m+1}$ 之间的时间间隔不超过规定时间 $t_0$ ,则有:

$$\begin{aligned} t_{n,j,m+1} - t_{n,j,m} &\leq t_0 \\ 1 &\leq m < x_{n,j} \end{aligned} \quad (1)$$

式中:

$t_{n,j,m+1}$ ——报警信息 $s_{n,j,m+1}$ 出现的时刻。

同时满足上述4个规则,该子模式 $s_{n,j}$ 匹配成功。

#### 2.1.3 总模式匹配规则

总模式匹配规则为:①对于某一确定的总模式 $n$ ,其所包含的各个子模式 $s_{n,j}$ 均匹配成功;②对于总模式 $n$ , $s_{n,j}$ 的第1个字符 $s_{n,j,1}$ 出现的时刻必须分布在某一确定的时间围栏 $T$ 内,即 $t_{n,j,1} \in T$ ( $t_{n,j,1}$ 为 $s_{n,j}$ 的第1个报警信息 $s_{n,j,1}$ 的出现时刻)。时间围栏 $T = [t_{n,j_0,1} - t_1, t_{n,j_0,1} + t_1]$ ,其中: $t_{n,j_0,1}$ 为总模式 $n$ 第 $j_0$ 个子模式 $s_{n,j_0}$ 的第1个报警信息 $s_{n,j_0,1}$ 的出现时刻; $t_1$ 为设定常数。

同时满足上述 2 个规则,总模式  $n$  匹配成功。

#### 2.1.4 相似度

假设总模式  $n_1$  中所包含的报警字符在总模式  $n_2$  中均按顺序出现,则称  $n_1$  包含于  $n_2$ ,记为  $n_1 \subseteq n_2$ 。通过定义相似度  $R$  提高故障模式的识别准确度。故障模式  $n$  发生的相似度  $R_n$  表示总模式  $n$  在与实时报警字符序列进行匹配时,所能够实现匹配的字符数。根据定义,若在实时获取的报警字符序列中同时出现了故障模式  $n_1$  和  $n_2$ ,且  $R_{n_2} \geq R_{n_1}$ ,即所发生的故障与模式  $n_2$  的相似度更高,故认为发生的是故障模式  $n_2$ 。

#### 2.2 故障定位算法设计

根据模式定义及匹配规则,设计模式匹配算法。

步骤 1 将故障模式库中所有总模式的相似度初始化为 0;选定第 1 个子模式对应的数据序列作为第 1 个待匹配对象;选取该数据序列的第 1 个字符,假定该字符为  $\gamma_{\text{str},1}$ ,则该字符的发生时刻为  $t_{1,1}$ 。

步骤 2 根据  $\gamma_{\text{str},1}$ ,在模式库中搜索所有总模式中第 1 个子模式  $s_{n,1}$  的首字符,选出  $s_{n,1}$  的首字符  $s_{n,1,1}$  与  $\gamma_{\text{str},1}$  匹配成功的所有总模式,进而初步筛选出第 1 个子模式的首字符匹配成功的总模式集合  $A$ 。若集合  $A$  非空,  $A = \{a_1, a_2, \dots, a_p\} (1 \leq p \leq N)$ ,则返回该集合,并转至步骤 3;否则,删除步骤 1 中所选数据序列中的第 1 个报警信息,并转至步骤 1。

步骤 3 根据  $t_{1,1}$  计算时间围栏  $T, T = [t_{1,1} - t_1, t_{1,1} + t_1]$ 。

步骤 4 在时间围栏  $T$  内,针对集合  $A$  中的总模式集合,任意选取一个总模式  $a_i (1 \leq i \leq p)$ ,在除第 1 个子模式外的  $a_i$  中的其他各子模式所对应的数据序列中逐个搜索  $s_{a_i,j,1} (1 \leq j \leq K \text{ 且 } j \neq 1)$ ,其中  $s_{a_i,j,1}$  为总模式  $a_i$  的第  $j$  个子模式的首字符。若在时间围栏  $T$  内,存在任意  $j$  使得  $s_{a_i,j,1}$  搜索失败,则从集合  $A$  中删除  $a_i$ ;否则,将  $a_i$  加入集合  $B$ 。遍历集合  $A$  中所有  $a_i$ 。

步骤 5 若集合  $B$  非空,  $B = \{b_1, b_2, \dots, b_q\} (1 \leq q \leq p)$ ,则返回该集合,并转至步骤 6;否则,删除步骤 1 所选的数据序列中的第 1 个字符,并转至步骤 1。

步骤 6 对集合  $B$  中的总模式  $b_1$  按以下过程进行匹配:在  $[t_{b_1,j,m-1}, t_{b_1,j,m-1} + t_0] (1 < m \leq x_{b_1,j}, x_{b_1,j}$  为总模式  $b_1$  的第  $j$  个子模式中包含的数据序列总数)时间范围内,对总模式  $b_1$  中第  $j$  个子模式  $s_{b_1,j}$  对应的数据序列搜索  $s_{b_1,j,m-1}$ 。若存在某一确定的

$j_1 (1 \leq j_1 \leq K), m_1 (1 < m_1 \leq x_{b_1,j_1})$ ,使得  $s_{b_1,j_1,m_1}$  未被搜索到,则  $b_1$  匹配失败,返回  $j_1$  和  $m_1$  值;否则,  $b_1$  匹配成功,  $j_1 = K, m_1 = x_{b_1,K}$  ( $x_{b_1,K}$  为总模式  $b_1$  的第  $K$  个子模式中包含的数据序列总数),返回  $j_1$  和  $m_1$  值。若  $b_1$  匹配成功,则将  $b_1$  加入空集合  $C$  中,同时计算  $b_1$  的相似度  $R_{b_1}$ 。

步骤 7 在集合  $B$  中继续搜索“包含”步骤 6 中已完成匹配字符  $s_{b_1,j,m} (1 \leq j \leq j_1, 1 \leq m < m_1)$  的总模式,这里的“包含”指与已完成匹配字符的位置和字符值均相同。若搜索到符合条件的总模式,设为  $b_z$ ,即  $s_{b_1,j,m} = s_{b_z,j,m} (1 \leq j \leq j_1, 1 \leq m < m_1)$ ,则在  $s_{b_z,j_1,m_1}$  处继续对  $b_z$  进行剩余字符的匹配,直到再次发生字符匹配失败为止,返回失败点为  $j_2$  和  $m_2$ 。然后,继续在集合  $B$  中搜索包含已完成匹配字符  $s_{b_z,j,m} (1 \leq j \leq j_2, 1 \leq m < m_2)$  的总模式,如此循环直至遍历集合  $B$ 。若存在某一总模式匹配成功,则将该总模式加入集合  $C$ ,并计算该总模式的相似度。

步骤 8 若  $C$  不为空集,则选取其中相似度最大的故障模式作为匹配成功的模式进行报警,并将各实时数据序列中已参与匹配的字符删除,转至步骤 1;否则,表示没有完成匹配,则删除步骤 1 所选的数据序列中的第 1 个字符,并转至步骤 1。

在步骤 6—步骤 8 中,通过有效利用每一次匹配成功部分的信息,避免了算法的回溯搜索。当模式库中的一个总模式  $n$  与数据序列中的数据进行匹配时,若匹配过程中的某个字符  $s_{n,j,m_l} (1 \leq n \leq N, 1 \leq j \leq K, 1 \leq m_l < x_{n,j})$  匹配成功,但在匹配  $s_{n,j,m_l+1}$  时失败,此时若直接删除该条匹配失败的总模式,并对模式库中下一个总模式进行匹配,就会损失已进行比较且匹配成功了的字符信息。为此,可通过记录该总模式  $n$  中已匹配成功字符的位置,即  $j, m_l$  值,在模式库中进行搜索,查找出包含已匹配成功字符的总模式,再将这些查找出的总模式分别与数据序列中的数据进行匹配,初始匹配位置为  $s_{n,j,m_l+1}$ 。如此遍历模式库,直至完成一个总模式的完全匹配后停止。

### 3 CBTC 数据监测系统构建与算法验证

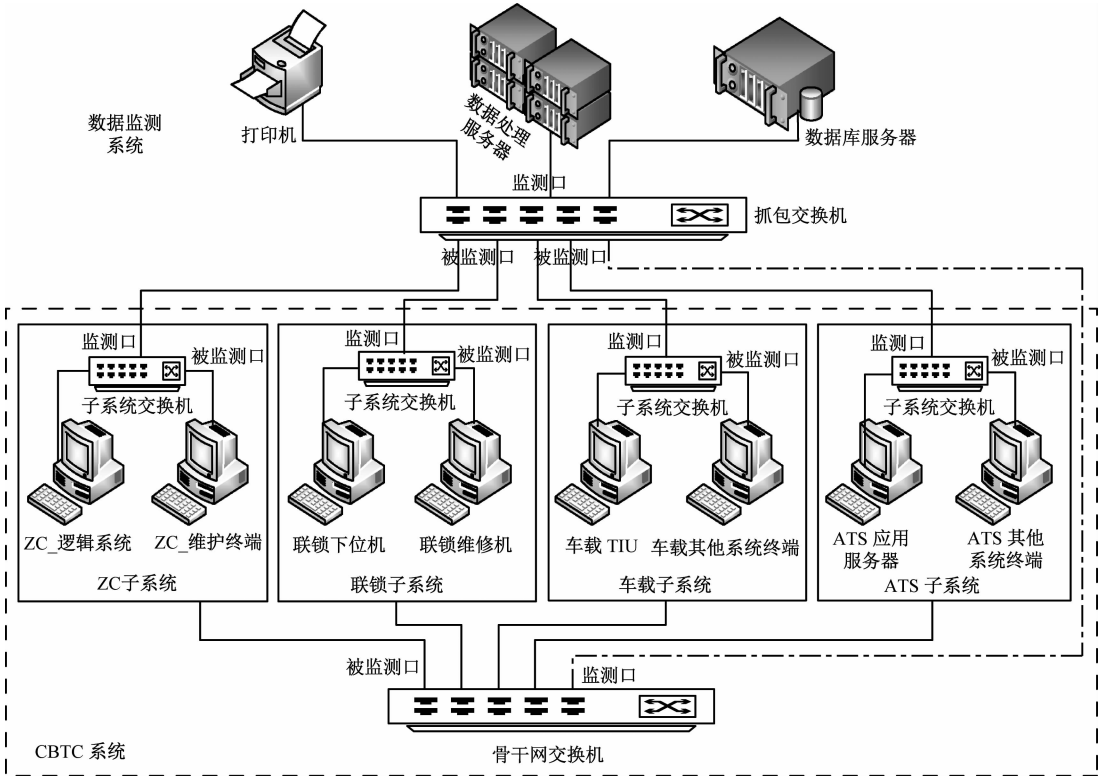
#### 3.1 CBTC 数据监测系统构建

CBTC 数据监测系统架构示意图如图 1 所示。CBTC 系统的各子系统之间通过骨干网交换机交互接口应用数据,各子系统内的不同模块(或终端)之



间,通过子系统交换机实时交互报警数据等信息。利用交换机的端口镜像技术,设计交换机级联方案,将骨干网交换机内的数据和各子系统内部交换

机上的通信数据分别镜像到抓包交换机,再对抓包交换机做端口镜像,将所有数据汇总至上层数据处理服务器,并以此作为故障检测模型的原始输入数据。



注:图中将 CBTC 系统的所有终端均以 PC(个人计算机)终端作为示意代替;ATS 为列车自动监控;TIU 为车载接口单元。

图 1 CBTC 数据监测系统架构示意图

Fig. 1 Diagram of CBTC data monitoring system architecture

CBTC 数据监测系统架构设计从物理上保证了数据获取通道的畅通性。在此基础上,设计 CBTC 数据监测系统的软件部分,包括抓包模块<sup>[6]</sup>和故障识别模块。故障识别模块识别流程示意图如图 2 所示。

3.2 算法验证

利用所提故障定位算法及 CBTC 数据监测系统,在重庆轨道交通 9 号线的 CBTC 系统上进行算法验证。所提故障定位算法在 CPU(中央处理器)为 Intel® Xeon® Silver 4116,主频为 2.2 GHz,24 线程 12 核心,内存为 16 GiB 的服务器上连续运行,内存及 CPU 占用率始终维持在较低水平。

CBTC 数据监测系统的故障标签库规模为 2 046 条,随机选取故障标签库中的 5 条故障,作为待测案例。通过人工对 CBTC 数据监测系统植入相应的故障状态,使得所提故障定位算法能够实时识别出系统内发生的故障,并提供正确的故障定位报警提示。故障定位耗时记录如表 1 所示。

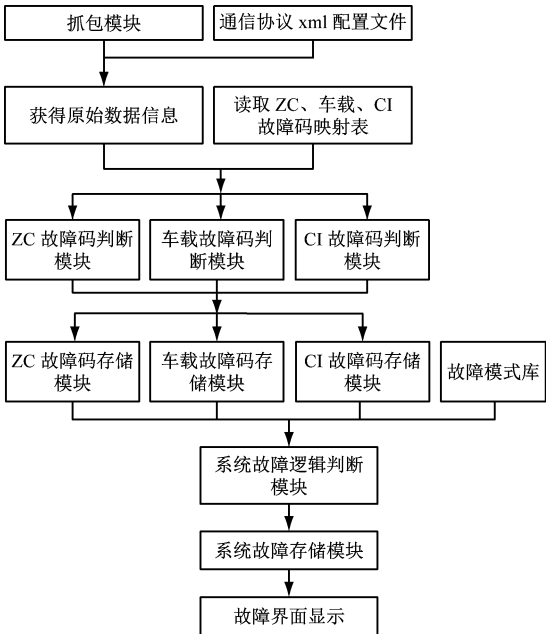


图 2 故障识别模块识别流程示意图

Fig. 2 Identification process diagram of fault identification module

表 1 故障定位耗时记录

Tab. 1 Records of fault locating time consumption

序号	故障名称	模式库中位置	故障定位算法平均耗时/ms
1	CTC(调度集中)车连续丢失两个应答器降级	第 1991 条	81.4
2	CTC 车通信超时降级	第 1533 条	193.6
3	ZC 与 CI 通信中断	第 1047 条	163.7
4	CTC 车头筛丢失降级	第 670 条	75.3
5	CTC 车闯红灯降级	第 362 条	139.4

## 4 结语

根据 CBTC 系统的系统架构、应用数据传输原理及系统参数,结合测试经验,设计了一套能够进行实时快速故障定位的方法。所提 CBTC 系统实时故障定位方法在针对具有时间约束的故障诊断定位方面具有普适性,尤其适用于大型分布式软件多点监测控制系统的故障检测及智能故障诊断。根据模式匹配规则设计了故障定位算法,运用了已完成有效匹配信息的继承方法,能够有效避免故障模式的回溯搜索过程。经 CBTC 数据监测系统验证,所提算法能够快速完成 CBTC 系统的故障定位功能。所提 CBTC 系统实时故障定位方法无论在测试阶段还是在现场运营阶段均能发挥重要作用,其不仅降低了系统测试和故障运维的技术难度,通过快速故障定位,大大减少了故障恢复时间。

## 参考文献

- [1] 魏晓超,徐琳,郑志华,等. 适用于智能环境的高效安全云辅助模式匹配协议[J]. 计算机研究与发展, 2019, 56(11): 2365.  
WEI Xiaochao, XU Lin, ZHENG Zhihua, et al. Efficient and secure cloud-assisted pattern matching protocol for intelligent envi-

ronment[J]. Journal of Computer Research and Development, 2019, 56(11): 2365.

- [2] 秦建,孙秀锋,吴春明. “垃圾短信”监控的中文多模式模糊匹配算法[J]. 西南大学学报(自然科学版), 2013, 35(3): 168.  
QIN Jian, SUN Xiufeng, WU Chunming. A research of Chinese multi-pattern fuzzy matching algorithm for SMS Spam monitoring[J]. Journal of Southwest University (Natural Science Edition), 2013, 35(3): 168.
- [3] 冷晓杰,李兴明. 模糊理论在网络故障实时诊断中的应用[J]. 计算机技术与发展, 2013, 23(6): 170.  
LENG Xiaojie, LI Xingming. Application of fuzzy theory in real-time diagnosis of network fault[J]. Computer Technology and Development, 2013, 23(6): 170.
- [4] 姜华,韩安琪,王美佳,等. 基于改进编辑距离的字符串相似度求解算法[J]. 计算机工程, 2014, 40(1): 222.  
JIANG Hua, HAN Anqi, WANG Meijia, et al. Solution algorithm of string similarity based on improved levenshtein distance[J]. Computer Engineering, 2014, 40(1): 222.
- [5] 赵晓,何立风,王鑫,等. 一种高效的模式串匹配算法[J]. 陕西科技大学学报(自然科学版), 2017, 35(1): 183.  
ZHAO Xiao, HE Lifeng, WANG Xin, et al. An efficient pattern matching algorithm for string searching[J]. Journal of Shaanxi University of Science & Technology (Natural Science Edition), 2017, 35(1): 183.
- [6] 白广争,李亮,白广良. 基于通信的列车控制系统数据采集与解析软件的设计与实现[J]. 城市轨道交通研究, 2018, 21(10): 119.  
BAI Guangzheng, LI Liang, BAI Guangliang. Design and implementation of data collection and analysis software based on CBTC system[J]. Urban Mass Transit, 2018, 21(10): 119.

· 收稿日期:2023-02-27 修回日期:2023-04-24 出版日期:2024-02-10  
Received:2023-02-27 Revised:2023-04-24 Published:2024-02-10

· 第一作者:白广争,助理研究员,562623634@qq.com  
通信作者:冯浩楠,副研究员,562623634@qq.com

· ©《城市轨道交通研究》杂志社,开放获取 CC BY-NC-ND 协议  
© Urban Mass Transit Magazine Press. This is an open access article under the CC BY-NC-ND license

(上接第 111 页)

- [5] 贾明杰. 计轴设备在轨道交通信号领域的应用[J]. 城市建设理论研究(电子版), 2015, 5(12): 1375.  
JIA Mingjie. Application of axle counting equipment in rail transit signal field[J]. Urban Construction Theory Research, 2015, 5(12): 1375.

Received:2021-07-22 Revised:2021-07-22 Published:2024-02-10

· 第一作者:杨荣兵,高级工程师,3574275944@qq.com  
通信作者:梁建彬,高级工程师,811019146@qq.com

· ©《城市轨道交通研究》杂志社,开放获取 CC BY-NC-ND 协议  
© Urban Mass Transit Magazine Press. This is an open access article under the CC BY-NC-ND license

· 收稿日期:2021-07-22 修回日期:2021-07-22 出版日期:2024-02-10