软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(8):1713−1728 [doi: 10.13328/j.cnki.jos.004666]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于有限状态机的 RFID 流数据过滤与清理技术*

罗元剑[1,2,4], 姜建国[1,2], 王思叶[1,3,4], 景 翔[1,2,4], 丁 昶[1,2,4], 张珠君[1,2,4], 张艳芳[1,2,4]

[1](中国科学院 信息工程研究所,北京 100093)

[2](中国科学院大学,北京 100049)

[3](北京交通大学 计算机与信息技术学院,北京 100044)

[4](物联网信息安全技术北京市重点实验室(中国科学院 信息工程研究所),北京 100093)

通讯作者: 王思叶, E-mail: wangsiye@iie.ac.cn, http://www.iie.cas.cn/

摘 要: 为了实现从大量不可靠、冗余的 RFID(radio frequency identification)流数据中提取有效信息,提高 RFID 系统中数据的质量,提出了一种基于有限状态机的 RFID 流数据过滤与清理方法.实验结果表明:该方法能够有效过滤系统外标签数据,清理系统内部冗余标签数据,筛选有效标签数据,并能够降低漏读、误读带来的风险.最后,利用地理信息系统的可视化技术,将过滤与清理结果展示在地图上.

关键词: RFID(radio frequency identification);有限状态机;流数据过滤与清理;地理信息系统

中图法分类号: TP393

中文引用格式: 罗元剑,姜建国,王思叶,景翔,丁昶,张珠君,张艳芳.基于有限状态机的RFID流数据过滤与清理技术.软件学报, 2014,25(8):1713−1728. http://www.jos.org.cn/1000-9825/4666.htm

英文引用格式: Luo YJ, Jiang JG, Wang SY, Jing X, Ding C, Zhang ZJ, Zhang YF. Filtering and cleaning for RFID streaming data technology based on finite state machine. Ruan Jian Xue Bao/Journal of Software, 2014,25(8):1713−1728 (in Chinese). http://www.jos.org.cn/1000-9825/4666.htm

Filtering and Cleaning for RFID Streaming Data Technology Based on Finite State Machine

LUO Yuan-Jian[1,2,4], JIANG Jian-Guo[1,2], WANG Si-Ye[1,3,4], JING Xiang[1,2,4], DING Chang[1,2,4], ZHANG Zhu-Jun[1,2,4], ZHANG Yan-Fang[1,2,4]

[1](Institute of Information Engineering, The Chinese Academy of Sciences, Beijing 100093, China)

[2](University of Chinese Academy of Sciences, Beijing 100049, China)

[3](School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China)

[4](Beijing Key Laboratory of IOT Information Security (Institute of Information Engineering, The Chinese Academy of Sciences), Beijing 100093, China)

Corresponding author: WANG Si-Ye, E-mail: wangsiye@iie.ac.cn, http://www.iie.cas.cn/

Abstract: In order to extract useful information from a large number of redundant and unreliable RFID (radio frequency identification) streaming data and improve the data quality of the RFID system, a filtering and cleaning method based on finite state machine for RFID streaming data is proposed. According to the experiment, suppressing interference tag data, filtering redundant tag data and extracting effective tag data can be realized in this method with reduced risk of system omission and false positives. Finally, using GIS technologies the monitoring and tracking results are displayed on the world map.

Key words: RFID (radio frequency identification); finite state machine; filtering and cleaning streaming data; GIS (geographic information system)

* 基金项目: 国家高技术研究发展计划(863)(2013AA014002)
收稿时间: 2013-12-30; 修改时间: 2014-04-29; 定稿时间: 2014-05-29

RFID(radio frequency identification)无线射频技术具有自动识别、快速检测、降低人力资源成本等优点,已经成熟地应用到很多的行业中,例如物流行业、制药和保健行业、图书馆、非接触式 ID 卡和门票、资产管理、制造业、服装业、汽车工业、动物标识、交通管理、航空航天、军队等[1,2].多行业的广泛应用也带来了对 RFID 数据的更高要求.

由于 RFID 信号穿透水或金属能力弱、功耗低、无线信号多途径效应等原因,造成了数据的不可靠性.此外,大量 RFID 标签数据被系统读取也会造成 RFID 原始数据的冗余.不可靠性主要指系统外标签数据的读入、系统内标签数据漏读、误读.冗余性主要指大量标签数据重复、无效[3].如果将 RFID 原始数据直接在实际环境中应用,得到的结果往往没有价值,因此,系统要对 RFID 原始数据进行过滤与清理,以提高 RFID 数据的质量.

本文针对基于 RFID 技术的监控和跟踪定位应用场景,提出了一种基于有限状态机的 RFID 流数据过滤与清理方法.实验结果表明:该方法能够有效排除系统外标签数据、清理系统内部冗余标签数据,筛选有效标签数据,同时降低系统漏读、误读带来的风险.

本文第 1 节介绍本文的研究背景和目前 RFID 数据过滤与清理方面的相关研究工作.第 2 节介绍基于监控和跟踪定位应用场景定义的 RFID 三元组数据模型和流数据处理架构.第 3 节针对 RFID 原始数据不可靠、冗余的问题,提出数据过滤与清理模型及算法.第 4 节对本文提出的方法的正确性、有效性进行实验证明.第 5 节对全文进行总结.

1 研究背景和相关研究工作

1.1 本文研究背景

本文所进行的研究主要针对基于 RFID 技术的监控和跟踪定位的应用场景.我们根据应用需求对实验场景进行了部署,如图 1 所示,把实验室划分为 4 个监控区域,分别标记为 area0,area1,area2,area3;在每个监控区域及边界部署多个检测天线和读写器,用于获取粘贴在被监控对象上的 RFID 标签信息.

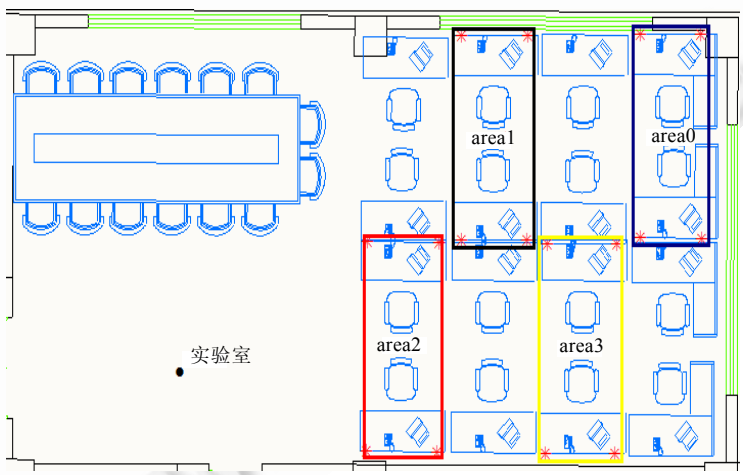


Fig.1 Illustration of application scenario

图 1 应用场景示意图

- 应用需求 1:监控被监控对象是否在监控区域,例如被监控对象在监控区域(area1),那么,监控系统需要每隔某个时间段(例如 1 小时)或者在某几个时间点(例如,9:00am,18:00pm,23:00pm)提示管理员,被监控对象还在监控区域.
- 应用需求 2:跟踪定位被监控对象的位置信息,例如被监控对象最初在监控区域(area0),当被移动到其他监控区域(area1 或者其他监控区域)时,监控系统应该立刻通知管理员,被监控对象从 area0 区域被带

出,并被带入监控区域 area1,并根据报警信息画出被监控对象的移动轨迹路线.

根据应用需求可知,监控系统具有实时性的特点,监控数据具有有序性的特点.怎样从大量不可靠、冗余的 RFID 流数据中提取有效信息,提高监控系统中的数据质量,对提高基于 RFID 技术的监控系统的监控效果起到关键作用.

1.2 相关研究工作

根据对相关文献的调研和研究,目前,提高 RFID 标签数据质量的解决方案主要有 3 种:第 1 种为物理层解决方案,提高 RFID 系统硬件来提高读写器的读取效率,以减少系统漏读;第 2 种为中间件层解决方案,把 RFID 数据存入数据库之前,利用一些数据清理的算法来提高数据质量;第 3 种为应用层解决方案,在数据库中,使用智能合并处理技术以达到数据清理的效果[4].

在物理层解决方案中,尽管采用 C1G2 协议[5]能够提高标签的识别率,较少数据的缺失率,但是标签与读写器的配置、多路径问题以及不可预测的干扰,会造成大量无效与冗余的数据[6].文献[6]中提出了一种结合数据分析(data analysis)和非单调推理的方法(non-monotonic)对漏读的数据进行恢复,但是该方法主要针对主动式 RFID 标签,在被动式 RFID 标签中并不能很好地应用;同时,在处理多个读写器接受到的 RFID 标签信息时,具有非时序的特性,对非单调性推理影响很大.

文献[7,8]针对在清理一定区域静态(非移动)RFID 标签数据时,无法准确还原 RFID 标签数据的位置问题,分别采用了集合论和贝叶斯推导的方法.但是在实时性要求高、监控移动 RFID 标签的监控系统中,并不能得到很好的应用.

文献[9]提出了一种流水线数据清理的框架.该框架称为可扩展的传感器数据流处理(ESP).它根据阅读器所获得的数据在时间和空间上具有相关性对数据进行处理.数据清理过程是由点处理、平滑处理、合并处理、判决处理和虚化处理 5 个阶段组成的数据清理流水线,但是该框架处理流程比较固定,交换其中任意两个阶段的顺序,都会影响数据处理的效果.在平滑处理阶段,确定某个对象 O 在时刻 t 是否在某个位置上,不仅取决于阅读时刻 t,而且依赖于预先设定的时间窗口 W,但是阅读时刻 t 随机性比较大,时间窗口 W 难以设定.在合并处理阶段,将监控同一个逻辑区域的多个阅读器的阅读数据组合起来,以判断对象 O 在某个逻辑区域中的位置,这样一来,读写器的增加必定会产生大量冗余的数据.

文献[3]在时间窗口的基础上提出了两种 RFID 冗余数据的融合算法:一种为基本融合算法(BaselineMerge),另一种为哈希融合算法(HashtabelMerge).两种算法都假设所处理的 RFID 数据来源自于同一个读写器,即该算法只能在时间维度上对 RFID 冗余数据进行融合和过滤,提取有效数据,而不能提取多监控区域(多读写器)产生的有效 RFID 数据.同时,该方法只能设定 1 个时间窗口,不能满足监控系统中,任意时间段与某几个时间点上,监控系统对被监控对象是否在监控区域的提示.

针对文献[3,9]中难以设定适合的时间窗口的问题,文献[10]基于数理统计的方法提出了一种自适应的不可靠 RFID 数据统计平滑方法(statistical smoothing for unreliable RFID data,简称 SMURF).基于数据的统计特性,连续采用平滑策略的统计方式,提出了二项分布标签数据清理模型和基于参量的多标签清理模型.但该方法主要针对单读写器情况下的数据清洗,对于多读写器的清洗工作,即多监控区域,只给出了一个初步设想,目前没有进一步的相关研究.

文献[11]在读写器识别标签时进行冗余数据处理,提出了一种 RRE 的解决方法.该方法可以使每一个 RFID 标签只能被唯一的读写器识别,从而解决了多个阅读器同时识别同一个 RFID 标签数据的问题.但是该方法在执行过程中,RFID 标签将不断重写由读写器发送过来的读写器覆盖的标签数量以及读写器的标识符信息,不仅使传输时间变长,而且标签的重写操作需要更多的时间,对不能进行重写的 RFID 标签,该方法不能得到应用;同时,该方法并没意识到一段时间内读写器读取的静态标签数据中包含了很多冗余数据的问题.

文献[12]提出了两种基于数据仓库冗余数据清理模型.第 1 种模型为(EPC,location,time_in,time_out),其中,EPC(electronic product code)表示 RFID 标签编号.该模型假设相同标签在某一时间段内产生的数据都是冗余数据.第 2 种模型为(EPC list,location,time_in,time_out).该模型假设 RFID 对象在某一时间段倾向于一起移动和停

留,因此可根据它们的位置来分组,相同组的数据可抽取一条,其他的为冗余数据.但是该方法在清除冗余数据时粒度过大,一些有效信息可能被清理.在实时性要求高的监控系统中,这种基于数据仓库的方法并不适用.

2 数据模型与系统架构

2.1 RFID三元组数据模型

根据应用场景需求,定义 RFID 三元组数据模型:{EPC,Position,Timestamp},其中,EPC 表示 RFID 标签编号,可以唯一标识一个 RFID 标签;Position 表示标签被读取的位置,由读写器 IP 与端口组成;Timestamp 表示标签数据被读写器读取的时间,如下所示:

```
〈sensor〉
  〈epc〉
    E2008181810101061160A01E
  〈/epc〉
  〈ip〉
    192.168.0.100
  〈/ip〉
  〈port〉1〈/port〉
  〈timestamp〉
    2013-11-13 22:28:06
  〈/timestamp〉
〈/sensor〉
```

2.2 流数据处理架构

在传统基于数据库或者数据仓库的数据处理技术中,系统响应用户提交的 SQL 查询语句,返回查询结果.当数据规模很大时,数据往往以磁盘或者磁带为存储介质,因此,执行查询操作需要大量的 I/O 交换,效率低下;同时,用户需要主动发起查询,查询到的数据已经成为历史数据,不能适应监控系统实时性高的要求[13].本文基于流数据处理技术设计了 RFID 流数据过滤与清理架构.系统架构如图 2 所示.

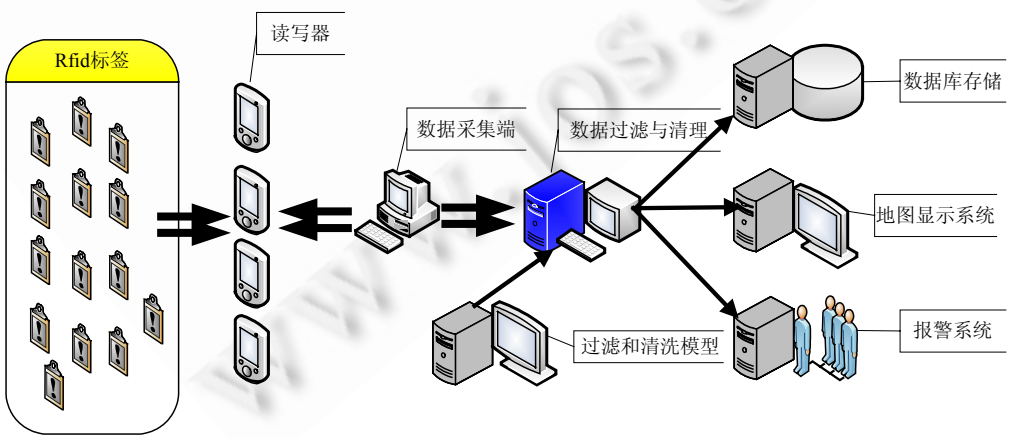


Fig.2 Architecture of filtering and cleaning for RFID streaming data system

图 2 RFID 流数据过滤与清理系统架构

监控系统主要分为数据采集前端,数据过滤与清理中间件,数据存储、展示、报警系统这 3 个部分.使用 RFID

设备中间件在数据采集前端不间断收集 RFID 编号 EPC,读取读写器 IP 地址、天线编号、时间等信息;收集到的 RFID 数据将被传输到数据过滤与清理中间件中,过滤和清洗模型可以动态添加到数据过滤与清理中间件中,中间件根据过滤与清理模型对 RFID 流数据进行处理;处理后的 RFID 数据将主动推送到数据库服务器、GIS 服务器、报警系统服务器,分别进行存储、展示、报警.

3 数据过滤与清洗模型及算法

3.1 有限状态机模型

根据应用场景和实际需求,本文提出了基于有限状态机的 RFID 流数据过滤与清理方法.下面根据应用场景,阐述有限状态机模型.有限状态机模型由一个五元组构成,记作 $M=(S,\Sigma,f,S_0,Z)$,其中,

- S={状态 i},S 是一个有限集,其中每一个元素称为一个状态.

在该实验场景中,有两种状态集:一种为位置(position)状态,例如场景中的(area0,area1,area2,area3);另一种为时间状态,例如监控时间段{time0(9,10],time1(10,11],time2(11,12],time3(13,14]}.为了规范,定义 RFID 标签的状态为(state0,state1,state2,state3,state4,…,state(n−1),state(n)).在该实验场景中,状态信息被保存在 state.xml 配置文件中,内容如下所示:

```
〈?xml version="1.0" encoding="UTF-8"?〉
〈root〉
〈!--状态 0--〉
〈state〉
  〈id〉state0〈/id〉
  〈position〉
    〈node〉Reader1:port1〈/node〉
    〈node〉Reader1:port2〈/node〉
    〈node〉Reader1:port3〈/node〉
    〈node〉Reader1:port4〈/node〉
  〈/position〉
  〈time〉
    〈node〉
      〈begin〉09:00〈/begin〉
      〈end〉10:00〈/end〉
    〈/node〉
  〈/time〉
〈/state〉
〈!--状态 1 与状态 0 时间状态不一致--〉
〈state〉
  〈id〉state1〈/id〉
  〈position〉
    〈node〉Reader2:port1〈/node〉
    〈node〉Reader2:port2〈/node〉
    〈node〉Reader2:port3〈/node〉
    〈node〉Reader2:port4〈/node〉
  〈/position〉
  〈time〉
```

```
        〈node〉
          〈begin〉10:00〈/begin〉
          〈end〉11:00〈/end〉
        〈/node〉
      〈/time〉
    〈/state〉
    〈!--状态 2 与状态 0、状态 1 地理位置不一致--〉
    〈state〉
      〈id〉state2〈/id〉
      〈position〉
        〈node〉Reader2:port1〈/node〉
        〈node〉Reader2:port2〈/node〉
        〈node〉Reader2:port3〈/node〉
        〈node〉Reader2:port4〈/node〉
      〈/position〉
      〈!-状态 2 与状态 1 时间状态不一致-〉
      〈time〉
        〈node〉
          〈begin〉09:00〈/begin〉
          〈end〉10:00〈/end〉
        〈/node〉
      〈/time〉
    〈/state〉
    …
    〈/root〉
```

- Σ={输入字符 i},Σ是一个有穷字母表,它的每一个元素称为一个输入字符.

在该实验室场景中,输入字符为新进入数据过滤与清理中间件的每一条标签信息,即(EPC,Position, TimeStamp).其中,每一条标签信息都包含 RFID 标签号、位置状态信息与时间状态信息.

- F:$S\times\Sigma\Rightarrow S$ 是状态转换函数,表示某状态接受某个新输入字符后所转变的状态.

在该实验场景中,状态转移函数为:如果当前一状态与当前输入的标签数据的位置状态、时间状态不相同,那么该 RFID 标签的状态更改为当前状态;如果相同,则保存原有状态.

- $S_0\in S$,S_0 是 S 中的一个元素,是唯一的一个初态.

在该实验环境中,S_0 表示,系统给被监控 RFID 标签预设定的初始状态,该状态可以为任意值.

- $Z\subseteq S$,且 $Z\neq\varnothing$,Z 是 S 的一个子集,是一个终态集,或者称为结束集.

在该实验环境中,Z 表示,被监控 RFID 标签最后一次状态改变后的状态.

3.2 相关概念定义

根据应用场景,下面定义本文应用到的一些概念.

定义 1(系统外标签数据). 当某个不在监控范围内的对象(RFID 标签)也被监控系统中的读写器读取到时,该数据称为系统外标签数据.

定义 2(标签数据漏读). 当被监控对象(RFID 标签)停留在监控区域或者经过监控区域时,没有被监控区域中的读写器读取到,这种情况下称为标签数据漏读.

定义 3(标签数据误读). 当被监控对象(RFID 标签)停留在某监控区域或者经过某监控区域时,在被该监控

区域的读写器读取到,同时也被其他监控区域的读写器读取到该标签数据,这种情况下称为标签数据误读.

定义 4(标签数据有效性). 在监控与跟踪定位系统中,只有在被监控对象(RFID 标签)从某个监控区域移动到另一个监控区域时,监控时间从某个时间段跳转到下一时间段时,即有限状态机模型中的状态改变时,读写器获取的 RFID 标签数据才是有效的,如图 3 中的实心部分所示.设经过系统过滤与清理后的有效数据的数目为 *EO*,理论有效数据的数目为 *ER*,*ER* 等于系统内标签状态改变的次数之和.

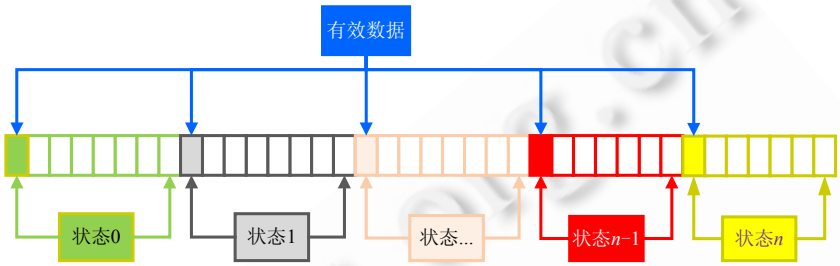


Fig.3 Distinction between valid data and redundant data

图 3 有效数据与冗余数据区分图

理论有效数据 *ER* 的计算公式如下:

$$ER=TotalTimes_{statechange}\times TotalCounts_{tags},$$

其中,*TotalTimes* 表示状态转移的总数,*TotalCounts* 表示被监控的标签总数.

定义 5(标签数据冗余). 在监控与跟踪定位系统中,读写器在同一状态下读取的 RFID 标签数据为冗余数据,即图 3 中的空心部分.在该实验场景中,主要有两种数据的冗余:物理空间位置冗余与时间冗余.物理空间位置冗余为同一监控区域中获取的被监控标签的数据,时间冗余为同一监控时间段内获取的被监控标签的数据.

定义 6(数据检测率). 在 1 次或者多次被监控对象(RFID 标签)经过监控区域时,RFID 标签数据经过系统处理过后,获取的有效标签数据数目 *EO* 与理论有效标签数据数目 *ER* 的比,称为数据检测率,公式如下:

$$detection\ ratio=(EO/ER)\times 100\%.$$

定义 7(数据压缩率). 在 1 次或者多次被监控对象(RFID 标签)经过监控区域时,系统采集到的所有 RFID 原始数据 *T* 与有效数据 *EO* 的比,称为数据压缩率,公式如下:

$$compression\ ratio=T/EO.$$

3.3 数据过滤与清洗算法

为了消除系统外标签数据,在系统配置文件中,配置被监控对象(RFID 标签)的标签号 EPC 与对应的初始状态,即有限状态机中的初态 S_0.在监控的过程中,如果读取的 RFID 标签号 EPC 不在配置文件中,则认为该 RFID 标签数据为监控系统外标签数据,只需把该数据丢弃.

为了减少标签数据漏读的情况,在每个监控区域逐渐增加天线个数,以提高数据检测率(detection ratio),直到检测率到达预设值为止,例如 100%.

为了消除标签数据误读的情况,首先设定一个标签数据阈值 *valueThreshold* 与时间阈值 *timeThreshold*,如果某两个监控区域或者某几个监控区域在该时间阈值内监控到同一 RFID 标签原始数据 *T* 大于 *valueThreshold*,则这两个监控区域或者该几个监控区域的物理空间位置是邻近的,即为同一个物理空间位置状态,应合并为一个监控区域;如果监控区域监控到的同一 RFID 标签原始数据 *T* 小于 *valueThreshold*,则该 RFID 标签监控区域的物理空间位置不邻近,即,不在同一个物理空间位置状态.如图 4 所示.

在图 4 中有 3 个监控区域,分别为监控区域 1、监控区域 2、监控区域 3.每一区域内都部署有相同数目和型号的 RFID 天线以及读写器,在监控区域 1 中有一个 RFID 标签,时间阈值 *timeThreshold* 假定为 1s,标签数据阈值 *valueThreshold* 假定为 3,监控区域 1 与监控区域 2 检测到的该 RFID 标签原始数据 *T* 超过了 3,则监控区域 1 和监控区域 2 在物理空间位置上是邻近的,应该把它们合并为同一物理空间位置,即,同一物理空间位置状

态;在监控区域 3 中,监控到的 RFID 标签原始标签数据 T 低于阈值 3,则物理空间位置不是邻近的,是不同的物理空间位置状态.

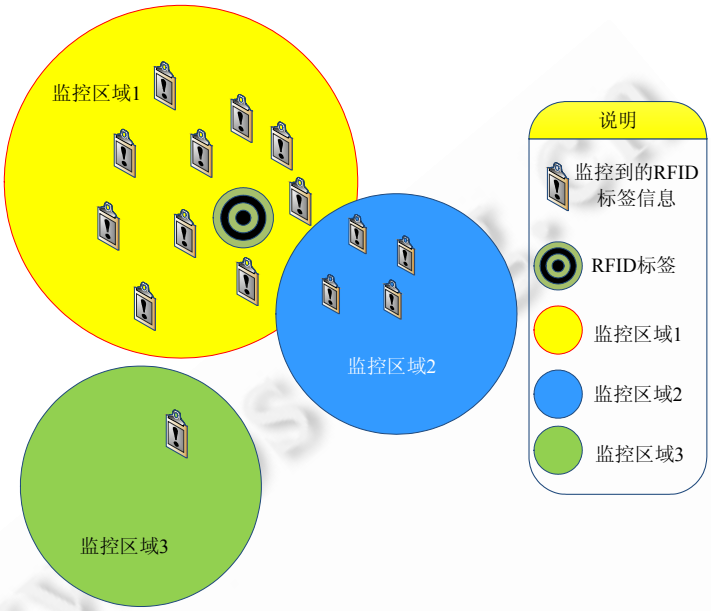


Fig.4 Illustration of combining the adjacent monitoring areas

图 4 合并相邻物理空间位置示意图

算法描述如下:

Initialize:

从配置文件 state.xml 中读取物理空间位置状态信息,并保存在 Vector 中(stateVector).Vector 中的元素为物理空间位置状态信息和是否被测试(checked)

设置一个待测 RFID 标签 testRFID

设置时间阈值 *timeThreshold* 与标签数量阈值 *valueThreshold*

Begin

//设置状态迭代器

stateIterator=*stateVector.iterator*(·)

//判断迭代器是否还有下一个元素

While (*stateIterator.hasnext*(·))

//获取下一个未测试物理空间位置状态信息

If (*stateVector.next*(·).*checked*==true)

continue;

End If

testStatePosition=*stateVector.next*(·)

//标记该位置已经被测试

testStatePosition.checked=true

//获取待测 RFID 标签在区域 *testStatePosition* 下,*timeThreshold* 时间内

//被其他 *checked* 为假的检测区域检测到的数量,并记录到一个 hash 表中

//hash 表的 *key* 值为物理空间位置状态,*value* 为该物理空间位置内检测到的 RFID 标签数量

```
rfidHashTable=getRFIDData(testStatePosition.Position,timeThreshold)
//如果 rfidHashTable 的 size 大于 0,则说明,其他监控区域也检测到了该标签
If (rfidHashTable.size(·)>0)
  //遍历整个物理空间位置状态
  For (i=0; i<stateVector.size(·); i++)
    //判断 rfidHashTable 是否包括了物理空间位置状态 i
    If (rfidHashTable.containsKey(stateVector.get(i).Position))
    //判断该监控区域内获取的标签数据的数目是否大于 valueThreshold
    //如果大于 valueThreshold,合并物理空间位置状态
    If (rfidHashTable.get(stateVector.get(i).Position).Value>valueThreshold)
        //合并物理空间位置状态
        mergePosition(testStatePosition,stateVector.get(i))
        //同时标记该位置已经被测试
        stateVector.get(i).checked=true
      End If
    End If
  END For
 End If
End While
//把合并后的物理空间位置状态写回 state.xml 中
writeBackXML(stateVector)
End
```

- 空间复杂度

该算法需要保存物理空间位置状态,因此空间代价为$\Theta(n)$.用 HashTable 存储检测区域检测到的被检测标签的数量,空间代价为$\Theta(n)$,则该算法空间复杂度为$\Theta(n)$.

- 时间复杂度

假设每次某一检测区间获取被检测标签的数量花费的时间为 a,每次合并物理空间位置状态花费的时间 b.

- 最坏情况下:物理空间位置两两相互独立(检测到的数量都小于阈值),那么花费时间代价为

$$T(n)=na+(n-1)a+\ldots+2a+a=\frac{n(n+1)}{2}a=\frac{n(n+1)}{2}a+0b;$$

- 最佳情况下:物理空间位置两两相邻,即第 1 次检测时,物理空间位置合并为一个检测区域,那么时间复杂度为

$$T(n)=na+(n-1)b,$$

则该算法的平均时间复杂度为$\Theta(n^2)$.

为了清理监控系统内的标签数据冗余,获取有效数据,根据有限状态机理论,只需要记录被监控对象(RFID 标签)的上一状态的信息.当读写器读取新的 RFID 标签数据(EPC,Position,Timestamp)后,与该 RFID 标签的上一状态进行比较:如果状态一致,即标签数据是冗余的,就丢弃该信息;如果状态不一致,即数据是有效的,就更新该 RFID 标签的状态信息,并把该 RFID 标签数据分发到数据库服务器、GIS 服务器、报警系统服务器,分别进行存储、展示、报警,如图 5 所示.

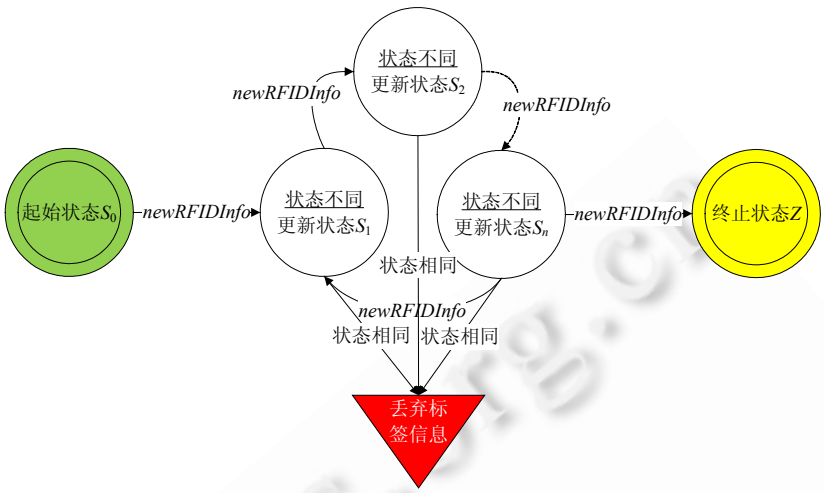


Fig.5 Illustration of filtering the redundant data

图 5 冗余数据过滤示意图

算法描述如下:

Input: *newRFIDInfo*(*EPC*,*Position*,*Timestamp*).

Initialize:从配置文件 state.xml 中读取状态信息,保存在 hash 表中(*stateHashTable*),初始化另一个 hash 表(*rfidHashTable*),存放被检测的 RFID 标签的的标签号与初始状态,*key* 值为被检测的 RFID 标签号,*value* 为初始状态,可以为任意状态值.

```
Begin
//在 stateHashTable 中,根据 newRFIDInfo 中的 Position 与 Timestamp 信息,查找该标签对应的状态,并
//把该状态保存到一个临时变量中,newState
newState=findNewState(stateHashTable,newRFIDInfo)
  //如果该标签数据,它的物理空间位置状态与时间状态都与 state.xml 中所规定的状态信息不符合,
  //即不再规定范围内,此时,newState 的值为空
  If (newState==NULL)
    //非监控内标签(非法数据),丢弃,并跳过后面的检查操作,执行下一条新到来的数据
    continue
  End if
  //在 rfidHashTable 中,根据 newRFIDInfo 中的 EPC,查找该标签对应的上一状态信息,如果查找失败
  //返回 NULL,查找成功返回状态信息,并保持在 oldState 中
  oldState=findOldState(rfidHashTable,newRFIDInfo)
  If (oldState!=NULL)
    //oldState 和 newState 进行匹配,如果相等,则丢弃该信息,
    //如果不相等,则更新 rfidHashTable 中该标签对应的状态
    If ((oldState.Position==newState.Position) && (oldState.Position==newState.Position))
      //丢弃该信息
    Else
      //在 rfidHashTable 中,更新该标签对应的状态
      updateRFIDState(rfidHashTable,newState)
      //并把该信息保存到数据库中、传输到报警系统中、显示在地图上
```

```
        dispatchRFIDInfo(newRFIDInfo)
      End if
    Else
      //系统外标签,不做处理
    End if
End
```

- 空间复杂度

存储状态信息需要$\Theta(n)$,存储被检测标签的数量$\Theta(n)$,则该算法空间复杂度为$\Theta(n)$.

- 时间复杂度

查找状态信息需要 $O(1)$,更新被检测标签的时间复杂度需要 $O(1)$,所以该算法的时间复杂度为$\Theta(n)$.

4 实 验

在每个监控区域内分别部署 1 台 UHF 频段无源读写器,每台读写器连接 4 支检测天线,分别放在监控区域的 4 个角.每个读写器的功率衰减为 60DBM,监控区域之间的距离可调,监控区域 0,1 与监控区域 2,3 之间用非金属挡板隔离.用于实验的读写器是 Alien 9800,配套天线是圆极化天线 Alien 9600BC,实验标签是 Alien H3 标签.Alien 9800 与服务器之间用网线连接,每个读写器都有自己的固定 IP 地址,每个读写器的 4 支天线接口分别用 port0,port1,port2,port3 标识.计算机实验环境见表 1.

Table 1 Experimental envionment

表 1 实验环境

硬件配置	CPU	Intel(R) Core(TM) i5-3210M CPU @2.50GHz
	硬盘	500GB
	内存	4GB
	网卡	Eth0: Intel® 82579LM Gigabit Network Connection
软件环境	操作系统	Windows 7 Ultimate (64bit)

实验中,令穿过监控区域的测试人员的步行速度保持在 4m/s 左右;采用两个 Alien H3 类型的标签作为系统内被监控对象,采用一个 Alien H3 类型的标签作为系统外标签;并规定 area0→area1→area2→area3 为一次路过,area3→area2→area1→area0 也为一次路过,每次路过,物理空间位置状态发生 3 次改变,即有效标签数据为 3 条.

实验 1. 针对标签数据漏读的情况,在每一个监控区域,逐渐增加天线个数来提高数据检测率,以降低标签数据漏读的情况.实验为 3 组,分别部署 1 对天线、2 对天线、4 对天线.测试人员携带 2 个被监控标签,路过监控区域 30 次,理论有效数据应为 182 条数据,实验结果如图 6 所示.

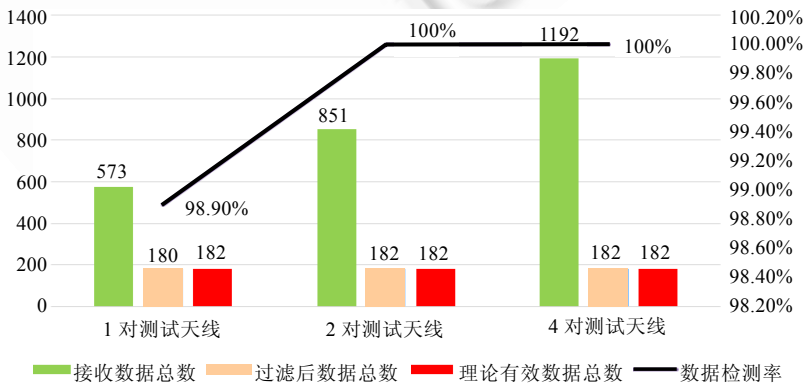


Fig.6 Result of processing negative data

图 6 处理漏报数据结果

实验 1 表明:在同一个区域增加 RFID 检测天线的数量,可以减少 RFID 标签数据的漏报,当增加的天线数目达到一定量时,检测率可以达到 100%;同时,随着天线数量的增加,获取的 RFID 标签总数也在增加,即冗余数在增加,压缩率也在增加.

实验 2. 针对标签数据误读的问题,设定时间阈值为 2s,标签数量阈值为 0,最后通过标签数据误读消除算法,合并相邻的物理空间位置状态,确定了当监控区域之间的距离为 2m 时,系统误报率为 0.实验 2 通过合并相邻的物理空间位置状态可以消除标签数据误读的情况.

为了验证本文提出的算法(FiniteStateMerge)对过滤与清理冗余数据和提取有效数据的有效性,在实验 1 和实验 2 结果的基础上,确定在每个监控区域分别部署 2 对天线,监控区域之间的距离为 2m,做了关于物理空间位置状态冗余数据过滤与清理和有效数据提取的实验 3 和关于时间状态冗余数据过滤与清理和有效数据提取的实验 4.

实验 3. 物理空间位置状态冗余数据的过滤与清理和有效数据的提取.设监控时间段为[00:00:00,23:59:59],以保障有限状态机模型中的时间状态为同一状态,在前 3 组实验中,逐渐增加物理空间位置状态的个数,同时设定在每一个物理空间位置状态内停留的时间为 3s,测试人员携带 2 个被监控标签,路过监控区域 10 次.第 1 组的物理空间位置为[area0,area1],理论有效数据位 ER=11×2=22(条);第 2 组的物理空间位置为[area0,area1,area2],理论有效数据为 ER=21×2=42(条);第 3 组的物理空间位置为[area0,area1,area2,area3],理论有效数据为 ER=31×2=62(条);在第 4 组实验中,物理空间位置为[area0,area1,area2,area3],但停留时间变为 5s,理论有效数据 ER=31×2=62(条).实验结果如图 7~图 10 所示.

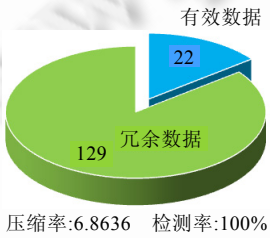


Fig.7 Two monitoring areas

图 7 2 个监控区域

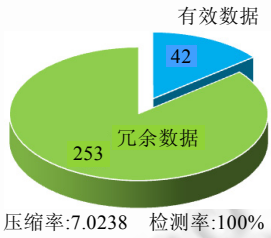


Fig.8 Three monitoring areas

图 8 3 个监控区域

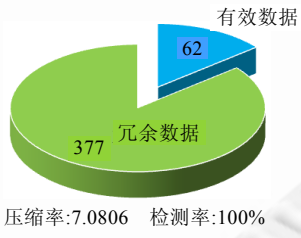


Fig.9 Four monitoring areas for 3s

图 9 4 个监控区域停留 3s

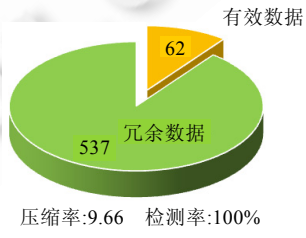


Fig.10 Four monitoring areas for 5s

图 10 4 个监控区域停留 5s

实验 3 在确保系统漏读数与误读数为 0 与时间状态相同的前提下,系统过滤出的有效数据 EO 和理论有效数据 ER 一致,即,检测率为 100%.这表明本文提出的算法能有效过滤与清理相同物理空间位置状态下的冗余数据,提取有效数据.前 3 组实验表明:随着物理空间位置状态的个数的增加,系统的压缩率基本保持不变,即监控区域的个数与压缩率无关.第 3 组与第 4 组实验表明:压缩率和标签在物理空间位置中停留的时间相关,停留时间越长,压缩率越大.

实验 4. 时间状态冗余数据的过滤与清理和有效数据的提取.放置 2 个被监控标签在 area1 中,以保障有限

状态机模型中的物理空间位置状态为同一状态,

在前 3 组实验中,每组的时间区间为 $time_i[beginTime+timeLength\times i,beginTime+timeLength(i+1)]$,时间间隔(*timeLength*)相同,都设为 5s,第 1 组 *i* 的取值[0,5],理论有效数据为 *ER*=5×2=10 条;第 2 组 *i* 的取值为[0,10],理论有效数据为 *ER*=10×2=20 条;第 3 组 *i* 的取值为[0,20],理论有效数据为 *ER*=20×2=40 次;第 4 组 *i* 的取值为[0,5],但时间间隔 *timeLength*=10s,理论有效数据 *ER*=5×2=10 条.实验结果如图 11~图 14 所示.

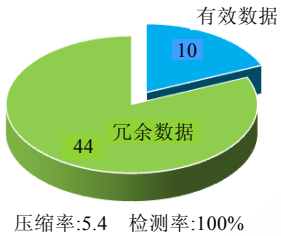


Fig.11 5 counts of time interval

图 11 5 个时间区间

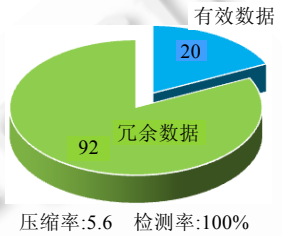


Fig.12 10 counts of time interval

图 12 10 个时间区间

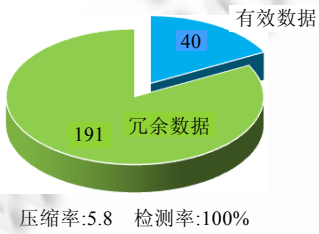


Fig.13 20 counts of time interval

图 13 20 个时间区间

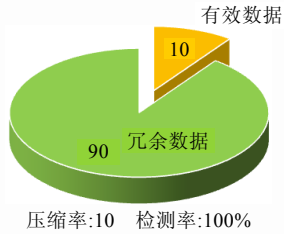


Fig.14 5 counts of time interval

图 14 5 个时间区间

实验 4 在确保系统漏读数与误读数为 0 与物理空间位置状态相同的前提下,系统过滤出的有效数据 *EO* 和理论有效数据 *ER* 一致,即检测率为 100%.这表明该算法能够有效过滤与清理相同时间状态下的冗余数据,提取有效数据.前 3 组实验表明:随着时间区间的个数的增加,压缩率基本保持不变,即时间区间个数与压缩率无关.第 1 组实验和第 4 组实验表明:压缩率和时间状态的时间间隔相关,时间间隔越长,压缩率越大.

实验 5. 在实验 1 和实验 2 的基础上,用本文提出的冗余数据过滤与清理方法(FiniteStateMerge)和文献[3]中提到的 BaselineMerge 与 HashtableMerge 方法做了如下的对比实验:

实验用 3 个 Alien H3 标签,ID 分别为 tag1,tag2 与 tag3,tag1,tag2 为系统内标签,tag3 为系统外标签.实验模拟 3 000 次路过,每个标签在每一个监控位置产生 5 条标签信息,则系统产生的标签数据总数为

3000(路过次数)×3(1 次路过经过的监控区域的个数)×3(标签总数)×

5(每次产生的标签数)+3×5(刚进入系统时标签数)=135015(条).

同时,设定从一个监控区域移动到另一个监控区域的时间间隔 1s.文献[3]中提到的两种方法都是基于时间窗口的方式进行冗余数据的过滤与清理.为了与实验场景保持一致,设定时间窗口为 3s,即一次路过的时间.理论有效数据为 *ER*=(3001)×2=6002,其中,3 001 为时间窗口总个数,2 为系统内标签数据.为了与文献[3]中的方法进行比较,设定每隔 3s 为一个时间状态,那么时间状态改变总数为 3 001.同时,物理空间状态改变总数为 3000×3+1=9001.因此,理论有效数据位 *ER*=(3001+9001)×2=24004(条),实验结果如图 15 所示.

由实验结果可知:本文提出的过滤与清理冗余数据提取有效数据的方法(FiniteStateMerge)过滤出的有效数据 *EO* 与理论有效数据 *ER* 一致,即检测率为 100%.这表明该算法能够有效过滤与清理相同状态下的冗余数据,提取有效数据.同时,文献[3]中的方法不能有效地识别系统外标签 tag3,只能从时间维度上对冗余数据进行清

洗,时间窗口一旦设置就不能变化,且大小不易设置.此外,一些有效的数据也被清理掉了,例如监控系统需要在某几个时刻通知管理员时产生的数据和被监控对象从一个监控位置移动到列一个监控位置时产生的数据.从时间性能上看,本文提出的方法(finiteStateMerge)需要一点额外的时间从配置文件中读取状态信息与系统内标签,总体上与 HashtableMerge 用时一致.

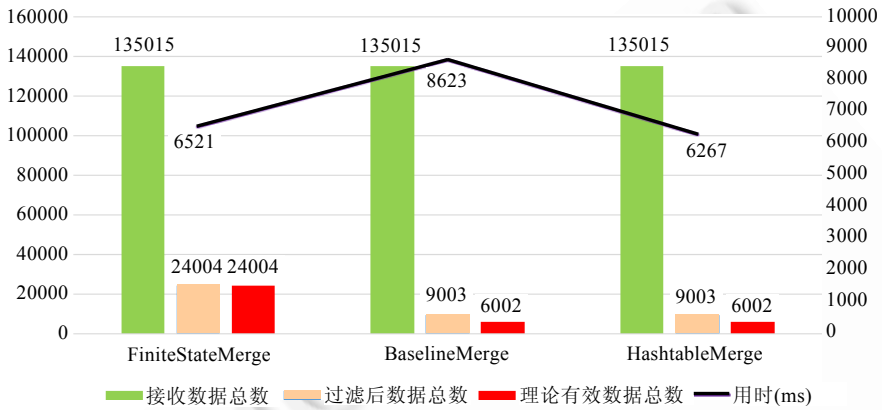


Fig.15 Three methods of duplicate elimination: FiniteStateMerge, BaselineMerge, HashtableMerge

图 15 3 种冗余消除算法:FiniteStateMerge,BaselineMerge,HashtableMerge

最后,结合地理信息系统技术,把测试人员携带两个被测标签路过一次实验场景的数据,不经过本文提出的方法处理与经过本文提出的方法处理后,分别显示在地图上,如图 16、图 17 所示.

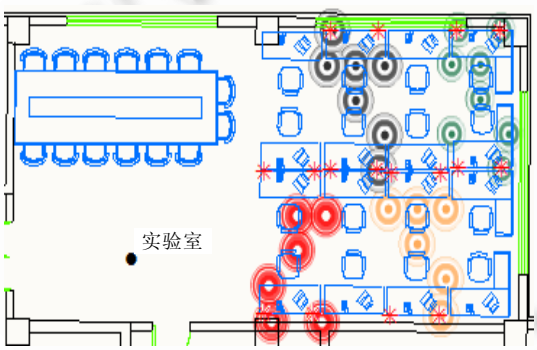


Fig.16 Display RFID data without processing method

图 16 显示未处理的 RFID 数据

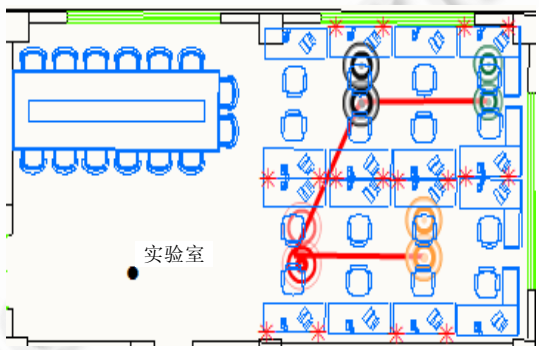


Fig.17 Display RFID data with processing method

图 17 显示处理后的 RFID 数据

实验总结:

在实验 1 中,不断增加天线数量,可以减少对被监控标签漏读的情况.当天线数量增加到一定量时,漏读标签数可以为 0.与文献[6]相比,该方法通过增加冗余数来减少系统对被监控标签的漏读,再通过本文提出的过滤与清理冗余数据的方法提取有效数据,在硬件相对便宜的情况下,该解决方案有一定的实用价值,同时实用于主动式与被动式 RFID 标签、多读写器的应用场景.

在实验 2 中,不断合并相邻物理空间位置状态,可以减少被监控标签误读的情况.当物理空间位置相隔较远时,误报数可以为 0.本文获取的被监控标签数据为三元组模型数据,即{EPC,Position,Timestamp},再通过合并相邻物理空间位置,排除误读情况,就能判断被监控标签在哪个监控区域.与文献[7,8]相比,该方法可操作性更强,效果也更佳.

在实验 1 和实验 2 的前提下,实验 3~实验 5 表明:本文提出的基于有限状态机的过滤与清理方法能够有效地过滤与清理同一状态下的 RFID 标签数据,提取有效监控数据.与传统的基于时间窗口的 RFID 流数据过滤方法(文献[3,9,10])相比,该方法不仅能够从时间维度上对数据进行过滤与清理,而且能够提取特定监控时间段内的数据,同时可以从物理空间位置维度上对数据进行过滤与清理,提取有效的监控数据,并过滤掉系统外标签数据,即干扰数据.文献[11]在读取标签数据时、文献[12]在标签数据存入数据仓库后对冗余标签数据进行过滤与清理,与之相比,本文提出的方法实时性更高,可操作性更强,代价相对较小.此外,该方法也能实用与快速移动的被监控对象,但前提是被监控对象能被监控系统所识别.

5 结束语

本文针对在基于 RFID 技术的监控和跟踪定位应用场景实现过程中要求实时性较强的特点,提出了一种流数据处理框架,被监控数据通过数据过滤与清理中间件处理后,实时的主动推送到数据库服务器、GIS 服务器、报警系统服务器;然后,针对在实现过程中产生的大量不可靠、冗余的 RFID 数据的问题,提出了一种基于有限状态机的 RFID 流数据过滤与清理方法.该方法利用 RFID 标签数据的三元组模型(EPC,Position,Timestamp)与有限状态机理论,一方面根据信号读取位置建立等效的物理空间位置状态模型,对相邻物理空间位置状态进行合并,以降低系统对被监控对象(RFID 标签)的误读;另一方面,通过对 RFID 流数据的分析,对同一物理空间位置状态与时间状态内的冗余数据进行过滤与清理.实验结果表明:该方法能够排除系统外标签数据、清理系统内部冗余标签数据,筛选有效标签数据,并降低漏报、误报带来的风险,在基于 RFID 技术的监控与跟踪定位系统上有一定的实用价值.

致谢 在此,谨向对本文的工作给予支持和建议的同行,尤其是中国科学院信息工程研究所的黄伟庆研究员、朱大立研究员及组内的其他老师表示感谢.

References:

[1] Derakhshan R, Orlowska ME, Li X. RFID data management: Challenges and opportunities. In: Proc. of the 2007 IEEE Int'l Conf. on RFID. IEEE, 2007. 175−182 [doi: 10.1109/RFID.2007.346166]

[2] Ahsan K, Shah H, Kingston P. RFID applications: An introductory and exploratory study. IJCSI Int'l Jornal of Computer Science Issues, 2010,7(1):1−7.

[3] Bai YJ, Wang FS, Liu PY. Efficiently filtering RFID data streams. In: Proc. of the CleanDB. 2006.

[4] Darcy P, Stantic B, Sattar A. A fusion of data analysis and non-monotonic reasoning to restore missed RFID readings. In: Proc. of the 2009 Int'l Conf. on Intell. Sensors, Sens. Networks Inf. Processing. IEEE, 2009. 313−318. [doi: 10.1109/ISSNIP.2009.5416745]

[5] EPCglobal Inc. EPC^{TM} Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz−960 MHz. Specification for RFID Air Interface, 2008.

[6] Massawe LV, Vermaak H, Kinyua JDM. An adaptive data cleaning scheme for reducing false negative reads in RFID data streams. In: Proc. of the 2012 IEEE Int'l Conf. on RFID. IEEE, 2012. 157−164. [doi: 10.1109/RFID.2012.6193044]

[7] Pupunwiwat P, Stantic B. Location filtering and duplication elimination for RFID data streams. Int'l Journal of Principles and Applications of Information Science and Technology, 2007,1(1):29−43.

[8] Chen HQ, Ku WS, Wang HX, Sun MT. Leveraging spatio-temporal redundancy for RFID data cleansing. In: Proc. of the 2010 Int'l Conf. on Management of Data-SIGMOD. 2010. 51. [doi: 10.1145/1807167.1807176]

[9] Jeffery SR, Alonso C, Franklin MJ, Hong W, Widom J. A pipelined framework for online cleaning of sensor data streams. Technical Report, UCB/CSD-5-1413, UC Berkeley, 2005.

[10] Jeffery SR, Franklin MJ, Garofalakis M. An adaptive RFID middleware for supporting metaphysical data independence. The VLDB Journal, 2008,17(2):265−289. [doi: 10.1007/s00778-007-0084-8]

[11] Bilenko M, Mooney R. Adaptive name matching in information integration. In: Proc. of the IEEE Intelligent Systems. 2003. 16−23. [doi: 10.1109/MIS.2003.1234765]

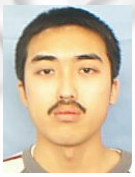
[12] Gonzalez H, Han JW, Li XL, Klabjan D. Warehousing and analyzing massive RFID data sets. In: Proc. of the 22nd Int'l Conf. on Data Engineering. IEEE, 2006. 83−94. [doi: 10.1109/ICDE.2006.171]

[13] Jin CQ, Qian WN, Zhou AY. Analysis and management of streaming data: A survey. Ruan Jian Xue Bao/Journal of Software, 2004, 15(8):1172−1181 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/1172.htm

附中文参考文献:

[13] 金澈清,钱卫宁,周傲英.流数据分析与管理综述.软件学报,2004,15(8):1172−1181. http://www.jos.org.cn/1000-9825/15/1172.htm

罗元剑(1988－),男,四川凉山人,硕士生,主要研究领域为信息安全,物联网.
E-mail: luoyuanjian@iie.ac.cn



丁昶(1990－),男,硕士生,CCF 会员,主要研究领域为信息安全,物联网.
E-mail: dingchang@iie.ac.cn

姜建国(1964－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为信息安全,保密科学技术.
E-mail: jiangjianguo@iie.ac.cn

张珠君(1987－),女,硕士,助理工程师,主要研究领域为信息安全,物联网.
E-mail: zhangzhujun@iie.ac.cn

王思叶(1981－),女,博士生,工程师,主要研究领域为信息安全,物联网.
E-mail: wangsiye@iie.ac.cn

张艳芳(1987－),女,硕士,助理工程师,主要研究领域为信息安全,物联网.
E-mail: zhangyanfang@iie.ac.cn

景翔(1979－),男,博士生,CCF 学生会员,主要研究领域为信息安全,智能计算.
E-mail: jingxiang@iie.ac.cn