基于布隆过滤器的 RFID 数据冗余处理算法研究

黄伟庆[1,2,3], 张艳芳[2,3], 曹籽文[2,3], 王思叶[1,2,3]

[1] 北京交通大学, 计算机与信息技术学院, 北京 中国 100044
[2] 中国科学院信息工程研究所, 北京 中国 100093
[3] 中国科学院大学网络空间安全学院, 北京 中国 100093

摘要 RFID 技术作为物联网领域的关键技术, 具有广阔的应用前景。然而 RFID 设备在读取标签信息时会产生大量冗余数据。因此, RFID 数据冗余处理的研究对于减少 RFID 中间件系统负荷、快速检测出入标签有着重要的意义。之前针对 RFID 数据冗余过滤的研究往往是单维度、静态场景的简单过滤, 无法实现复杂场景下标签的出入检测。因此, 本文提出一种名为时间距离布隆过滤器(TDBF)的算法, 该算法从时间和空间两个维度进行冗余过滤。与常用的时间布隆过滤器相比, 该算法兼顾了 RFID 标签的读取时间和读取距离, 极大的降低了数据的冗余问题。在保证漏读率较低的情况下, 极大的降低了数据的误读率。同时该算法支持动态场景中移动标签的冗余过滤, 能够较好的满足出入监控需求。

关键词 布隆过滤器; 冗余过滤; 数据清洗; 射频识别
中图法分类号 TP399 **DOI 号** 10.19363/J.cnki.cn10-1380/tn.2019.05.07

Redundant RFID Data Filtering Algorithm Research Based on Bloom Filter

HUANG Weiqing[1,2,3], ZHANG Yanfang[2,3], CAO Ziwen[2,3], WANG Siye[1,2,3]

[1] School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044 China
[2] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[3] School of Cyber Security, University of Chinese Academy of Sciences Beijing 100093, China

Abstract As a key technology in the field of Internet of Things, RFID technology has broad application prospects. However, RFID devices generate a large amount of redundant data when reading tag information. Therefore , the research on RFID data redundancy processing is of great significance for reducing the load of RFID middleware system and quickly detecting incoming and outgoing tags. Previous studies on the redundancy filtering of RFID data are often simple filtering of single-dimensional and static scenes, and it is impossible to detect the ingress and egress of tags in complex scenarios. Therefore, this paper proposes an algorithm called Time Distance Bloom Filter (TDBF), which performs redundant filtering from both time and space. Compared with the Time Bloom filter, this algorithm takes into account the reading time and reading distance of the RFID tags, which greatly reduces the redundancy of the data. In the case of ensuring a low miss rate, the false-positive rate of the data is greatly reduced. At the same time, the algorithm supports redundant filtering of mobile tags in dynamic scenarios, which can better meet the requirements of access control.

Key words bloom filter; redundant filtering; data cleaning; Radio frequency identification

1 引言

随着物联网(Internet of Things, IoT)技术的快速发展, 物联网已经改变了日常监控、交通出行等生活方式。作为物联网的关键技术之一, 无线射频识别(Radio Frequency Identification, RFID)技术经过多年的发展, 如今已经应用于生活中的各个领域, 尤其是结合保密行业以及重要单位的实际管控需求, 被广泛应用在供应链管理、图书管理、交通管理、文件追踪、文件防伪和门禁管理等[1]。

一般来说, RFID 系统主要由读写器(也叫阅读器, reader)、天线、标签以及管理软件组成, 如图 1 所示。读写器用于读写电子标签的数据, 内含信号发射模块及解调模块等; 标签内含芯片和天线, 保存一定

通讯作者: 王思叶, 硕士, 高级工程师, wangsiye@iie.ac.cn。
本课题得到物品管控系统安全方案设计及系统测试(No.Y7V0131104)资助。
收稿日期: 2019-02-25; 修改日期: 2019-04-19; 定稿日期: 2019-05-13

编码格式的数据, 用于唯一标识一个标签和物品; 天线作为读写器和标签之间的传输数据的发送以及接收装置。最后管理软件通过计算机网络与读写器连接, 负责对读取到数据信息进行存储、管理和控制。一般而言, 管理软件可以表现为 RFID 中间件。

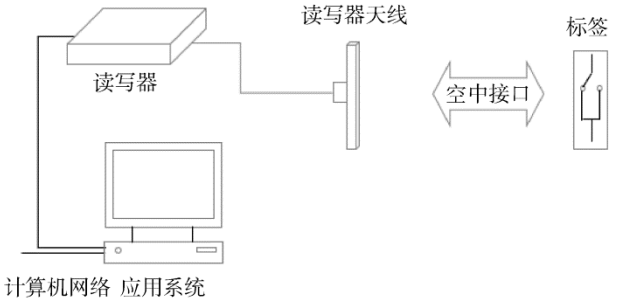

图 1 RFID 系统组成结构
Figure 1 RFID System Structure

RFID 技术由于其非接触性、快速识别等技术优势, 已经被用在智能识别、出入检测等场合, 用于识别出入的人员、物品等信息。尤其是针对出入检测的场景, 比如常见的包括会议文件监控场景、档案监控场景、仓储监控场景等, 对数据冗余过滤提出了更高的要求, 将监控区域以外动态或静态目标的数据进行有效过滤。除此之外, 读写器在对监控区域以内的 RFID 数据流采集时也会产生海量的冗余流数据。因此, 如何实时有效的对 RFID 冗余数据进行处理, 使我们需要解决的难题。

为了有效进行数据冗余过滤, 首先我们分析一下产生 RFID 数据冗余的原因: 一方面是因为多个临近读写器之间存在交叉读写区域, 从而同一个标签被反复读取。另一方面, 同一个标签长时间停留在某一固定读写器的范围内, 被多次读取所以产生了大量的重复、无效的数据, 也可能是为了提高精度, 所以对相同的物品贴了多个标签[2]。在中间件系统中, 需要对这些海量的冗余数据进行实时的处理, 一方面, 这些数据往往占据了大量系统存储资源以及网络带宽资源, 增加功耗并增大时延, 使得中间件系统的运行效率大大的降低。在一些高机密场景中, 往往需要对可疑人员的出入进行快速的判断, 否则存在该可疑人员潜逃的风险。若对冗余数据逐条处理, 毫无疑问会降低系统的运行效率, 导致不能快速判断。另一方面, 冗余数据在很多场景中往往缺少实际应用价值, 是浪费资源的无效数据。门禁系统中, 我们只对一定时间或者范围内的特定数据感兴趣, 冗余数据在特定场景中就变成了干扰数据。因此, 如何快读、有效的对冗余数据进行处理十分关键。

与此同时, RFID 数据作为一种流数据, 与传统关系数据库和数据仓库的数据相比, 在实时性上要求更高。在面对上千上万个标签同时工作时, 一方面由于内存的限制, 很多在静态数据上有效的清洗算法对 RFID 数据流并不适用; 另一方面, 流数据的特质使得可见记录的范围只能限于局部, 这使得过滤结果的准确性更低。因此, 如何在处理精度与处理速度上找到一个适用于 RFID 数据流过滤方法的平衡点十分重要。

现有的方法中存在着对冗余处理不及时以及对动态场景下冗余过滤效果不佳等诸多问题, 面对以上需求, 为了能过实时过滤检测区域以外的误读数据以及检测区域内的冗余读数据, 本文提出了一种基于布隆过滤器的 RFID 冗余数据处理算法, 针对出入检测的应用场景进行有效的过滤。首先, 分析 RFID 信号特性及数据结构, 经过大量实验及 RFID 信号采集分析, 明确了 RFID 的信号传播特性, 选择不同的属性数据用于 RFID 数据建模, 构建出 RFID 数据的时空特性; 其次, 在对 RFID 数据建模的基础上, 结合实际的应用场景, 通过时空关联特性设计时间距离的布隆过滤算法(Time-Distance Bloom Filter, TDBF), 通过实验对比分析算法的优劣。通过对 RFID 冗余过滤算法的研究, 大幅度的降低冗余数据, 减轻了数据处理的压力。本文的主要贡献包括: 一、提出了一种基于布隆过滤器的数据冗余处理算法, 该算法兼顾了 RFID 的时间特性与空间特性; 二、冗余处理算法不仅考虑到静态标签的冗余过滤, 同时可以对运动中的标签进行有效的过滤; 三、经过实验证明, 该算法能够有效应用在实际场景中, 不仅存在学术价值, 也有很高的应用价值。

论文剩余部分组织如下: 第二部分首先介绍了 RFID 的数据格式特点, 并对当前数据冗余过滤策略进行概述; 第三部分针对布隆过滤方法进行详细介绍, 并设计了名为时间距离布隆过滤器的算法; 第四部分对我们提出的算法从多个维度进行实验验证分析; 最后一部分总结并得出结论。

2 相关工作

RFID 数据流相对于其他流数据有着不同的数据格式与特点, 本小节中, 首先我们概括了 RFID 的数据特征, 然后针对 RFID 数据流的特征, 我们在表 1 中总结了大量国内外相关的过滤算法研究[3-4]。

2.1 RFID 数据特征

为了进行 RFID 冗余数据过滤, 首先分析 RFID

数据流的特点, 和传统数据流不同, RFID 数据流有着以下特征[5,6]:

1) **数据格式简单**: RFID 数据流的格式往往是*(tagid, location, time)*的三元组形式, 其中 *tagid* 表示的是标签唯一编码, *location* 是RFID读写器读取标签的位置, *time* 是发生读取行为的时间。除此之外, RFID 数据流还包括了一系列不常用的数据格式如接收信号强度值等, 但总体我们可以发现 RFID 数据流相对其他数据流携带了较为简单的信息。

2) **海量数据**: RFID 读写器不间断的读取标签信息, 在短时间内便会产生大量的 RFID 数据流, 在仓库这类场景中, 有着成千上万的标签, 且各类读写器之间还存在密集部署的问题, RFID 数据流将会十分庞大。

3) **不确定性**: 限制 RFID 技术广泛应用的重要原因之一就是 RFID 的数据流具有不确定性, 由于射频干扰以及读取原理等多种原因, RFID 数据能被正确读取的概率大约是 60%~70%。RFID 数据流不确定性主要体现在这些读取的数据中还存在着多读(false positive)、漏读(false negative)、冗余读取的问题, 多读是指读写器读到了阅读范围之外的标签, 漏读是指对范围内的标签数据存在读取不全的现象, 冗余读即在一个规定的时间段内, 对相同的标签采集到了多条数据。

4) **实时性**: 读写器在获取 RFID 标签信息时, 会记录读取该记录的时间戳, 连续的时间戳标识了该标签数据的可见时间范围, 这样的时序关系表示了 RFID 标签的动态性。

5) **关联性**: RFID 数据一般来说不是单独存在的, 而是相互关联的。RFID 数据具有时空特性, 同一标签的时间戳反映了其在时间上的时序关系, 而记录的位置与状态反应了标签在空间上的变化过程, 时空关联又共同反映了该标签所标识物品对象的有关事件。

因此, 在进行 RFID 数据流冗余去除时, 要考虑以上 RFID 数据流的特点。RFID 的数据质量直接决定了上层软件的处理效率, 这些不可靠的数据往往对系统的效率产生了不良影响。

2.2 RFID 冗余数据过滤方法

为了有效消除 RFID 数据流中的冗余数据, 传统数据冗余处理技术的思想是将所有数据都存储到数据仓库或者数据库中, 再根据数据库查询语句返回查询结果[7]。然而随着数据规模的扩大, 查询效率便会下降且不满足实时性的要求, 此时便产生了基于窗口的过滤方法, 它不需要保存所有的数据, 而是仅需要维护一个远小于其规模的窗口大小的数据[8], 从而对流数据不间断的实时处理。滑动窗口模型关注的是最新到达的数据, 随着新的数据流的到达, 窗口内的数据不断更新, 期满数据溢出, 新数据再插入。然而滑动窗口法依赖于属性字段的选择且窗口的大小不易确定, 过大的窗口可能导致时间复杂度较高, 较小的窗口可能在冗余去除上不够彻底。

冗余过滤的目的根本上还是提取流数据中的有效信息, 因此, 配合数据挖掘, 流数据有了更多的学习算法[9, 10], 利用在线学习算法可以及时学习流数据特征, 从而进行冗余判断。然而这些模型必须满足一系列要求: 例如每个对象在训练过程中只能被处理一次以及处理每个实例的计算复杂度必须尽可能小, 否则处理速度会过慢。集成学习同样是数据流挖掘的一种流行的方法。该算法通常是对最近到达的数据(通常以块的形式收集)训练新的分类器并将其添加到整个模型中。通过训练, 迭代每个分类器的权重, 从而对数据进行更好的过滤与预测。数据挖掘的方法能够一定程度上有效解决数据冗余的问题, 但往往需要大量数据支撑且处理速度有待提高。

此外, 还有很多针对 RFID 数据流的特征设计的冗余处理算法。文献[11, 12]各自提出了一种流水线型数据清理的框架——ESP 和 PDC.两种清洗框架都由一定的清洗步骤构成, 每个步骤有各自的清洗任务。根据读写器所获得的数据在时间和空间上具有相关性进行处理, 全面的对漏读、多读、冗余读数据进行清洗过滤。

文献[13, 14]都是从读写器层面考虑 RFID 冗余数据的过滤, 其中 CBF 算法是基于布隆过滤器的改进算法, 针对现有冗余处理算法中往往采用大量 Hash 函数导致冗余处理效率低下的问题, 该作者仅采用单布隆过滤器就实现了对冗余读写器进行有效的过滤, 文献[14]中提出了一种阈值选择算法(TSA), 以消除大规模分布式 RFID 网络中的冗余读取器。该算法是基于由期望标签覆盖范围所确定的阈值序列迭代地选择有效读取器。两种算法都考虑了分布式环境下多读写器的去除, 从而有效的对 RFID 数据冗余进行过滤。

在机器学习背景下, 文献[15]提出了一种基于动态贝叶斯网络的大量 RFID 数据集清洗方法。该方法引入了一个清洗框架, 来产生一个全局最优的清洗方案从而使得代价最小化, 还介绍了一种基于决策树清洗方法。但是 DBN 清洗方法是从历史数据中得到预测值和观察值的关系, 不能动态更新, 动态标签的清洗效果不明显。此基础上, 文献[16]分别提出

了基于静态贝叶斯网络(SBN)和动态贝叶斯网络(DBN)的传感器数据的冗余预处理方法, DBN 是 SBN 在时域上的扩展, 从而可以实时消除冗余传感器数据。但是该算法对历史数据的依赖性较强, 冗余决策速度较慢。

文献[17]利用有限状态机模型进行冗余数据清洗, 有限状态机方法的重点是将有效数据定义为状态发生改变的数据。应用场景往往是门禁、监控之类需要大量移动的场景。实际场景中, 状态机的数量会受到限制, 且根据场景的不同, 需要制定不同的状态模型, 不适用于复杂的环境中。

表 1 RFID 冗余数据过滤方法

Table 1 RFID redundant data filtering approaches

RFID 冗余数据过滤方法	缺点
RFID 数据仓库[7]	不能满足数据流的实时处理
滑动窗口[8]	依赖于属性字段的选择、窗口大小不易确定
CBF[13]和 TSA[14]	仅在读写器层面进行冗余过滤, 没有考虑流数据特征
ESP[11]和 PDC[12]	清洗过程复杂, 不适用于动态场景下的实时过滤
贝叶斯网络 [15, 16]	算法复杂、动态标签清洗效果不明显
有限状态机[17]	状态机的个数受限且受环境因素影响较大
布隆过滤器[18]	不能删除过期数据, 过滤维度单一

以上方法虽然能够有效去除冗余, 但是对于 RFID 数据冗余过滤的实时性不能兼顾。因此基于哈希的布隆过滤器算法被提出。布隆过滤器能够快速对数据冗余进行过滤, 然而布隆布隆过滤器本身存在维度单一的缺陷, 且不能剔除已插入到过滤器中的数据, 所以存在一定的局限性, 下一节中, 将详细介绍布隆过滤器在 RFID 数据流过滤中的应用以及改进算法。

3 时间距离布隆过滤器算法

3.1 布隆过滤器相关研究

布隆过滤器自七十年代提出, 广泛地应用于网络、数据流等领域[19], 考虑到布隆过滤器快速、高效的过滤特性, 本文将研究如何利用改进的布隆过滤器方法对冗余数据进行高效的实时过滤。

布隆过滤器是以哈希函数为基础的数据冗余去重算法[18,20]。该算法由 m 位初始化为 0 的数组和 k 个独立的哈希函数$h_1, h_2, \cdots, h_k$组成,其值域范围为$\{1,2,\cdots,\mathrm{m}\}$。设集合$S=\{x_1, x_2, \cdots, x_n\}$有 n 个元素, 新的元素经由每一个哈希函数分别映射到位数组中的不同位置, 如果对应位置本身是 0, 则置为 1, 如果对应位置本身是 1, 此时就不改变位数组中的值, 即对每个$x \in S$, $BF_{h_i x}=1$，$1 \leqslant i \leqslant k$。如果映射的所有的哈希函数值在数组中的对应位置都为 1, 那么这个数据表示曾经在集合S中出现过, 此时可以被判定该数据为冗余数据。如果存在一个或多个映射值为 0, 此时认为该数据是新的集合元素数据。

布隆过滤器的具体操作如图 2 所示。图 2(a)中的“TagID1”是新到达的数据。数据经由布隆过滤器, 分别映射到了数组的第 0、3、5 位, 此时位数组由 0 变为 1, 图 2(b)在检查“TagID2”是否重复时, 第 1 位和第 6 位映射值为 0, 意味着“TagID2”尚未插入位数组中。

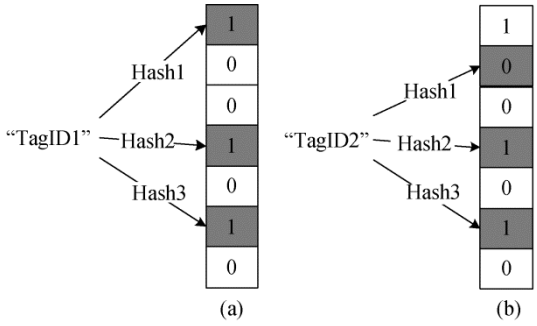

图 2 布隆过滤器基本原理

Figure 2 Basic principle of Bloom filter

RFID 数据流往往表示为<*TagID, ReaderID, Time*>的三元组模型, 因此基于布隆过滤器的 RFID 冗余去除往往是针对经典三元组模型信息的冗余度进行冗余去除, 表 2 展示了布隆过滤器在 RFID 数据流冗余过滤的应用算法。

表 2 布隆过滤器在 RFID 数据流的应用算法

Table 2 Application algorithm of Bloom filter in RFID data stream

RFID 数据流冗余去除	特点
时间/时间间隔布隆过滤器 TBF/TIBF[21]	存储数据的读取时间戳, 从而判断判断是否是冗余数据(TIBF 存储开始和结束的时间戳)
比较布隆过滤器 CBF[22]	过滤单元内存储一段时间内被检测到次数, 来判断该标签数据属于哪个读写器
时空布隆过滤器 TSBF[23]	考虑 RFID 冗余清洗的时间空间特性, 判断标签的移动性
近似概率合成布隆过滤器 PSBF[24]	针对 RFID 数据流的移动性、流特性、不确定性设计的有概率特性的布隆过滤器

其中时间布隆过滤器是针对 RFID 数据流的特点, 将位数组修改成了整型数组, 数组内保存最新数据的时间戳, 通过时间的更新来保证过滤器中的元组不会因全部存满而导致错误率急剧增加。当需

要判断一个数据 x 是否为冗余数据时, 首先将标签 ID 分别映射到 m 个独立的数组单元中: $h_1(x.Tid), h_2(x.Tid), \cdots, h_k(x.Tid)$,如果存在至少一个单元满足$x.Time - M[h_i[(Tid)] > \tau$(时间阈值), 则认为数据 x 不是冗余的, 如果全部的k个单元都满足$x.Time - M[h_i[(Tid)] \leqslant \tau$, 那么认为该数据在时间上是冗余的。在此基础上, 作者又根据 RFID 数据可能存在的假阳性提出了改进的时间布隆过滤器——时间间隔布隆过滤器, 该过滤器采用了二维布隆过滤器结构, 分别存储数据的开始时间和结束时间, 通过判断一组哈希值对应的过滤单元里的开始时间到结束时间这一时间段内是否存在相交时间, 如果为空, 则可确定所包含对象的数据未在时间间隔内到达, 即不是冗余数据。否则, 为冗余数据。该算法在时间层面处理标签的冗余问题。

比较布隆过滤器在过滤器单元里存储标签一段时间内被检测到的次数, 通过比较检测次数来判断标签属于哪个读写器, 并去除其他读写器关于该标签的数据, 该算法是在空间层面处理标签冗余信息, 能够判别标签所属问题, 但可能产生误删操作。

时空布隆过滤器将一维布隆过滤器拓展为二维, 其中一个维度存储数据的位置信息, 一旦发现标签的位置信息发生改变, 就更新过滤器数据单元。另一个维度存储数据的时间信息, 只有一定时间内到达的数据才是有效数据。该算法考虑了标签的移动性, 但是基于的假设是读写器不存在相交, 很难在实际场景中应用。

近似概率合成布隆过滤器将基本的 RFID 三元组形式转变成了<*tid, loc, ts, p*>的四元组模式, 其中 *tid* 表示的是标签编码, *loc* 是 RFID 读写器读取标签的位置, *ts* 是读取数据时间戳, *p* 表示 RFID 数据的读取概率因子, 由于距离读写器的更近的标签有更高的数据读取率, 所以空间过滤时只保留读取概率最大的标签, 这样就解决了 TSBF 中读写器范围不能相交的问题。该模型主要是针对不确定 RFID 数据流进行冗余处理。

根据上述布隆过滤器在 RFID 数据流的冗余过滤策略, 我们应该找到一种兼顾时间、空间的过滤方法, 从而对 RFID 数据流进行全面、高效的过滤。

3.2 相关定义

现有的冗余去除算法中, 对冗余的定义往往是将时间维度上相近且标签 ID 相同的数据定义为冗余数据, 这样能很大程度的进行冗余过滤, 但是针对本次重点研究的出入检测场景, 监控对象存在有运动和静止两种状态, 针对不同的状态数据冗余的定义也不同, 需要我们针对冗余探究其不同的含义。例如在某些会议场景或者文件仓库中, 门口往往存在一些 RFID 读写器, 这些读写器可以在短时间内读取到大量的 RFID 数据流, 然而, 这些数据大部分都是毫无意义的, 我们所关心的数据是那些距离读写器较近的标签数据, 这些数据往往存在着现实含义, 例如是否有人想要携带某份文件离开该出口, 而距离读写器较远, 却又采集到的数据往往是没有意义的, 那些数据只能增加系统运行的负荷。

因此, 除了如上一节所述在时间维度上的采用的布隆过滤器算法, 在空间上我们也应该对数据的冗余定义进行思考[25,26]。在此, 我们在原有 RFID 三元组*(TagID, ReaderID, Time)*的基础上, 引入了一个新的采集数据值 RSSI, RSSI 值用于表示信号强度的大小, 目前行业里面已经应用 RSSI 值来粗略的判断标签距离读写器的远近了[27]。接收到的 RSSI 值越大, 可以表明标签距离读写器越近, 越小则表示越远。然而, RSSI 信号值受环境影响很大, 会出现一定程度的波动[28]。因此, RSSI 适用于对标签位置精度要求不是太高的场景。文献[29]采用了读取速率表示读取的距离远近, 并进行了数据清洗, 考虑到对数据的实时处理, 这里直接采用读取的 RSSI 值表示距离。且利用改进算法降低 RSSI 值波动带来的影响。

在此, 结合时间和空间两个维度的考虑, 我们给出本算法中 RFID 数据流冗余的一些定义:

定义 3.1: 用 S 表示一串 RFID 数据流, $S = \{s_1, s_2, \ldots, s_n\}$,其中$s_i$表示一个 RFID 三元组<*Tid, Time, RSSI*>, *Tid* 是标签的唯一标识, Time 是读取 RFID 数据的时间戳, 表示标签数据被读写器读取的时间, RSSI 是采集的信号强度值。和经典 RFID 三元组 <*TagID, ReaderID, Time*> 不同, 我们忽略了 *ReaderID* 的值, 在这里仅考虑了单个读写器环境下的数据冗余问题。

定义 3.2: (时间冗余)设 w 为数据流 S 中的一段时间窗口, 若在 w 中存在两个标签数据分别满足 $x \in S, y \in S$, 如果 $x.Tid = y.Tid, y.Time - x.Time \leqslant \tau$且$y.Time > x.Time$, 其中$\tau$为设定的时间阈值, 此时认为数据 y 相对于数据 x 在时间上冗余。

定义 3.3: (距离冗余)若存在$x \in S, y \in S$, 标签数据 x 和 y 分别满足 $x.Tid = y.Tid$,$y.Time - x.Time > \tau$且 $y.RSSI < \varepsilon$。其中ε为 RSSI 的距离阈值, 小于该阈值表明标签距离较远, 所以可以认为数据 y 是无意义的冗余数据。

定义 3.4: 若分别存在$x \in S, y \in S$, 标签数据 x 和 y 分别满足 $x.Tid = y.Tid$,$y.Time > x.$

$Time, y.RSSI > \varepsilon$且$y.RSSI > x.RSSI$, 此时我们认为数据 y 不是冗余数据。当新来的数据 y, 有着更大的 RSSI 值时, 表明该数据有着接近 RFID 读写器的趋势, 因此此时的数据, 不能简单的根据时间或距离的限制判定为冗余数据。

3.3 时间距离布隆过滤器算法

3.3.1 时间距离布隆过滤器算法基本原理

时间距离布隆过滤器的算法(TDBF)考虑 RFID 数据的时空特性, 在时间布隆过滤器的基础上, 添加了对空间维度的思考, 实现了在一定时间范围及空间范围内对 RFID 冗余数据去重的功能。

TDBF 的结构如图 3 所示: TDBF 数组大小初始化为 m, 每个单元表示了一个二维整型数组, 其中第一列存储了读取标签的时间戳信息, 第二列存储了读取标签的接收信号强度值, 其中第 i 个单元的数据可以表示为 $M_i[Time][RSSI]$, $RSSI$ 值总是以负数的形式呈现, 因此在初始化过滤器时, 第一列初始化为 0, 第二列初始化为距离阈值ε。

当有新的 RFID 数据到达时, 对能标识该数据的 Tid 编码使用 k 次独立的哈希函数, 同时利用存储在映射单元中的时间和接收信号强度信息来判断该数据是否为冗余数据, 冗余判断原理如下:

(1) 对到达的新数据 x, 对其唯一标识 ID 进行 k 次独立哈希映射到对应的数组单元中。

(2) 如果存在数据 $i \in \{1,2,\cdots,k\}, M_i[Time] = 0, x.\mathrm{RSSI} > \varepsilon$, 那么表明该数据是新到达的距离范围内的数据, 并非冗余数据, 此时更新 k 个数组单元里的时间信息和信号强度信息, 即$M_i[Time] = x.Time, M_i[RSSI] = x.RSSI$。

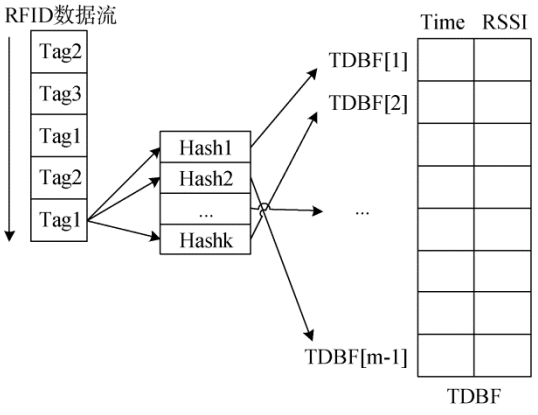

图 3 TDBF 数据结构图

Figure 3 TDBF data structure diagram

(3) 如果存在数据 $i \in \{1,2,\ldots,k\}, x.Time - M_i[Time] > \tau, x.RSSI > \varepsilon$,表明新到达的数据在规定的时间间隔外, 且在距离范围内, 并非冗余数据, 此时更新 k 个数组单元里的时间信息和信号强度信息, 即$M_i[Time] = x.Time, M_i[RSSI] = x.RSSI$。

(4) 如果存在数据 $i \in \{1,2,\cdots,k\}, x.RSSI > M_i[RSSI] \geqslant \varepsilon, x.Time > M_i[Time]$,此时新到达数据的 RSSI 在距离范围内, 且检测值变大, 我们有理由认为该数据存在接近读写器的趋势, 因此不论此时该数据是否在规定的时间间隔内, 我们都认为该数据是非冗余数据, 需要及时过滤。此时更新 k 个数组单元里的时间信息和信号强度信息, 即$M_i[Time] = x.Time, M_i[Rssi] = x.Rssi$。

(5) 否则认为该数据是冗余或者说是无效的数据, 直接丢弃。

结合之前所介绍的冗余定义和 TDBF 的基本原理, TDBF 的伪代码表述如算法 1 所示:

算法 1. TDBF 算法

输入: RFID 数据 x: x.Tid, $x.Time$, $x.RSSI$.

输出: x 是否为冗余数据

```
BEGIN
1:    INIT TDBF[Time]=0
2:    INIT TDBF[RSSI]=w
3:    FOR ( i = 1 ; i ≤ k ; i++)
4:        THEN p[i]=Hash_i(x.Tid)
5:    FOR ( i= 1 ; i ≤ k ; i++)
6:        IF TDBF_p[i][Time]=0 and x.RSSI>w
7:            THEN Update TDBF(x.Time, x.RSSI)
8:            Send x to the event
9:            BREAK
10:       ELSE IF x.time-TDBF_p[i][Time]>t and
x.RSSI>w
11:           THEN Update TDBF(x.Time, x.RSSI)
13:           Send x to the event
14:           BREAK
15:       ELSE IF x.time>TDBF_p[i][Time] and
x.RSSI>TDBF_p[i][RSSI]
16:           THEN Update TDBF(x.Time, x.RSSI)
17:           Send x to the event
18:           BREAK
19:       END IF
20:   END FOR
21:   Drop x
END
```

以上算法说明了 TDBF 的工作流程, 为了更加清晰展示 TDBF 算法, 我们假定监测环境中存在 Tag1 和 Tag2 这两种标签, 设定的时间阈值为 2 秒, RSSI 阈值为–60, 标签数据依次以三元组的形式到达, 下面依次对每个到达的数据进行分析:

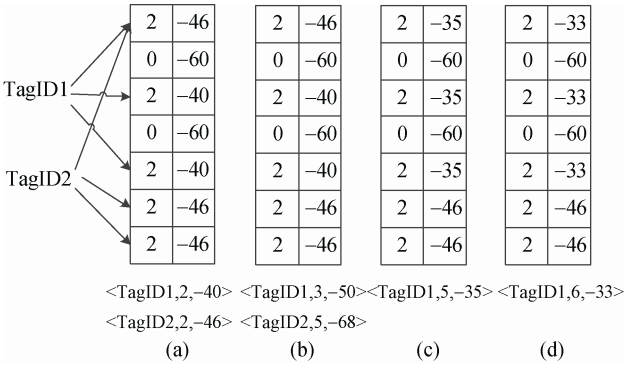

图 4　TDBF 算法实例图

Figure 4　TDBF Algorithm example diagram

(1) 数据<TagID1,2,–40>到达 TDBF 中, 根据标签 ID, 将该数据依次映射到数组的 0、2、4 号存储单元, 由于数组内初始时间为 0, 判断出该数据为新数据又根据其 RSSI 值大于–60, 所以判断出该数据在检测范围内, 此时更新数组单元内的时间和 RSSI 信息。数据<TagID2,2, –46>重复上述操作。更新结果如图 4-(a)所示。

(2) 数据<TagID1,3, –50>到达时, 由于其 ID 和之前存在数据的 ID 相同, 所以映射到了数组的相同位置, 但由于新到达数据的时间戳与存储时间戳的差值小于阈值, 所以判定该数据为冗余数据, 数据<TagID2,5, –68>到达时, 虽然该数据的阈值满足了非冗余条件, 但是设定的 RSSI 值小于了设定的距离阈值, 所以仍判定该数据为冗余数据。

(3) 数据<TagID1,5, –35>到达时, 对映射单元内的时间和 RSSI 信息进行判断, 该数据的时间差大于设定的时间阈值且 RSSI 值在距离阈值内, 所以更新 TDBF 数组, 更新结果如图 4(c)所示。

(4) 数据<TagID1,6, –33>到达时, 对映射单元内的时间和 RSSI 信息进行判断, 该数据的时间差虽然小于设定的时间阈值, 但是存在 RSSI 值大于最新存储的 RSSI 值, 因此我们认为该数据有可能是移动靠近读写器的重要数据, 因此该数据不是冗余数据, 所以更新 TDBF 数组, 更新结果如图 4(d)所示。

3.3.2　时间距离布隆过滤器算法流程设计

TDBF 的流程图如图 5 所示, 首先我们提取 RFID 数据中的三元组信息, 包括标签序列号、采集时间以及采集的 RSSI 值。并对缓存三元组数据的数组进行初始化, 包括时间和 RSSI 值的初始化。再利用冗余判断规则对数据进行冗余判断, 冗余判断规则一方面是利用时间和标签序列号信息针对静态数据进行冗余判断, 另一方面是利用时间和 RSSI 值针对动态数据进行冗余过滤。根据最终判断结果, 将冗余的数据直接丢弃, 而非冗余数据则作为有效信息递交给上层应用软件。上层软件可以根据过滤后数据对来访人员的出入进行迅速的访问控制判断, 从而对可疑人员进行有效、快速的监控。

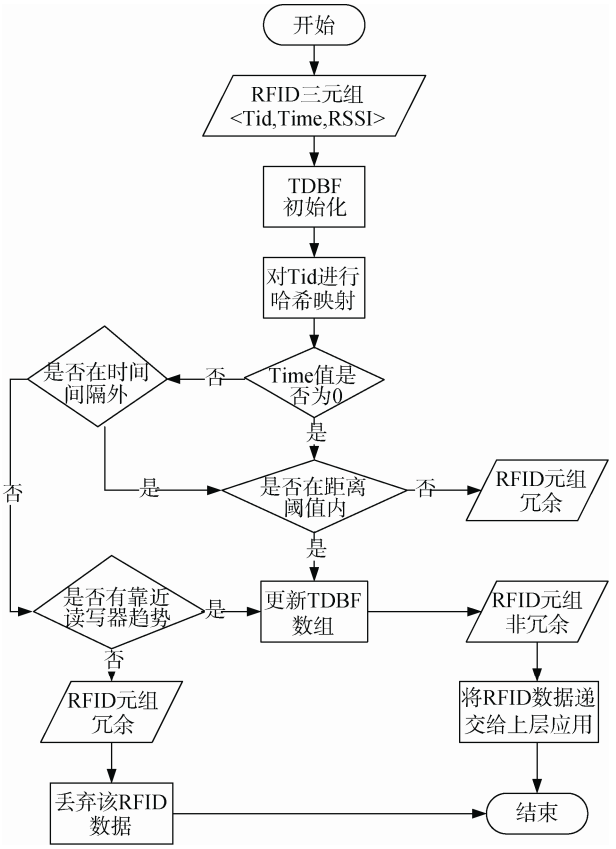

图 5　时间距离布隆过滤器算法流程设计

Figure 5　Time-distance Bloom filter algorithm design

4　算法分析验证

4.1　实验方案设计与分析

本实验的硬件采用了 Impinj revolution 系列无源读写器以及若干个配套标签。实验中, 将读写器部署在了门禁出入口位置。为了验证算法的有效性, 实验环境均为没有杂物干扰的空旷房间, 对标签进行数据采集时, 均模拟实际环境, 将标签放置于与读写器同等高度的位置上。读写器与计算机之间利用网线连接, 计算机的实验环境具体见表 3。

表 3　实验环境

Table 3　Experimental environment

硬件配置	CPU	Intel(R) Core(TM) i5-3320M CPU
	硬盘	256GB
	内存	4GB
	网卡	Eth0: Intel@82579LM Gigabit Network Connection
软件环境	操作系统	Windows 7 Ultimate(64 bit)
	软件版本	Visual Studio 2017

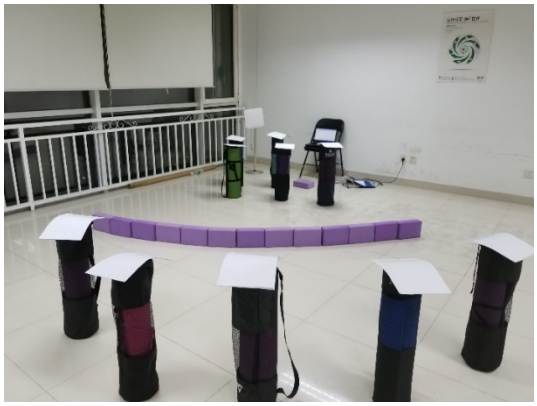

图 6　实验环境

Figure 6　Experiment environment

4.1.1　静态场景实验

静态实验场景部署如图 7 所示, 为了保证没有障碍物的影响, 所以选择了一个空旷的房间。实验中, 针对同一数据流, 分别采集其经过基本布隆过滤器(BF)、时间布隆过滤器(TBF)以及时间距离布隆过滤器(TDBF)过滤后的数据, 并进行分析对比。图中所示的 RSSI 距离阈值由一条曲线大致表示, 实际场景中, 由于 RSSI 值具有一定的波动性而且相同位置的标签产生的 RSSI 值会略有差别, 因此 RSSI 的距离阈值应该表示为一个距离范围。这里我们选取的 RSSI 阈值对应于实际场景中大约距离读写器 3m 的位置。考虑到 RFID 数据流的冗余过滤效率, 所以将时间阈值设置为 2 秒。为了验证静态环境下算法的过滤效率, 我们采取了如下实验。

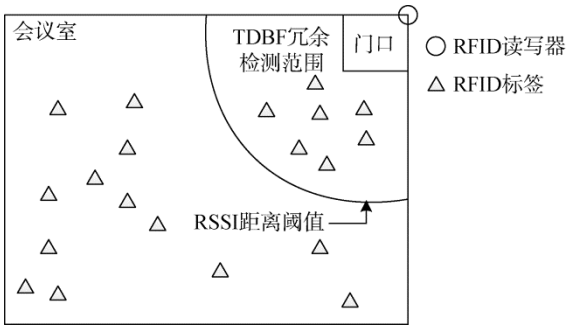

图 7　静态场景部署

Figure 7　Static scenario deployment

实验 1. 我们将 10 个标签分别静止放在 TDBF 冗余检测范围内和冗余检测范围外, 分别采集在 1 分钟的读取时间内经过 BF、TBF 与 TDBF 算法过滤后的剩余数据总量, 为了减少 RSSI 值波动所造成的影响, 我们在摆放标签时, 皆放置在了与所设定距离阈值保持了约 1m 距离的位置外。该场景刻意摆放标签是为了比较在理想场景中, 这三种算法的数据压缩率, 即采集到的原始标签数量与过滤后的标签数量的比值, 并分析 TDBF 算法相比于 TBF 算法及 BF 算法的漏读率与误读率。实验的数据直方图如图 8 所示:

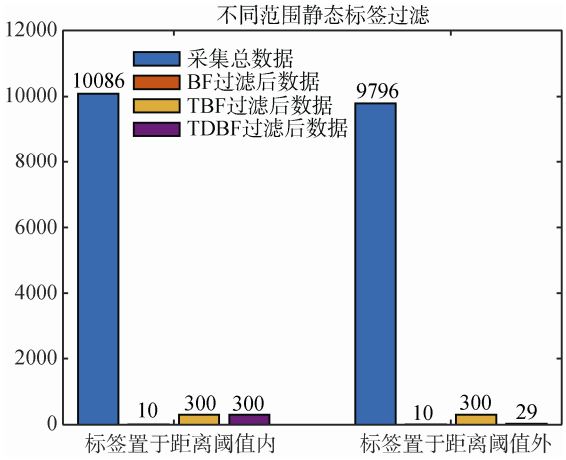

图 8　不同范围静态标签过滤

Figure 8　Different range static label filtering

由图可知, 在相同时间内, 采集到的距离阈值外标签数据量略低于标签阈值内标签的总量, 但数据总量基本相似, 证明在该实验环境中, 外部干扰较小, 能够有效的采集到大部分的数据信息, 但标签的距离远近会一定程度影响数据读取率。

在经过 BF、TBF 和 TDBF 算法后, 冗余标签都得到了良好的过滤。

当标签全部静态置于距离阈值内时, 我们观测发现原始数据的 RSSI 值非常稳定。理论上此时的有效标签个数为 300, 此时 TDBF 算法与 TBF 算法均达到了理论值, 保持了相同的数据压缩率。这说明 TDBF 算法在距离阈值范围内时, 基本上不会产生标签的漏读现象。而 BF 算法, 因为没有考虑到标签的时间与空间信息, 仅仅针对标签 ID 进行了粗粒度的过滤, 虽然有较高的数据压缩率, 但同时也丧失了很多门禁出入处有价值的信息。

当标签全部静态置于距离阈值外时, 此时理论有效标签个数为 0, 因为阈值外的标签数据都可以认为冗余数据, TBF 算法只考虑了 RFID 数据流的时间特性, 所以此时产生了 300 条误读数据, 但经过 TDBF 过滤后, 误读数据的数量减少到了 29 条。其中产生的数据误判率的原因一方面是可能由于 RSSI 值的波动产生, 另一方面可能是由 BF 算法本身的误判率造成的。但是我们观察到 BF 算法此时读取到了理论上的读取量, 因此证明是 RSSI 值的波动导致了 TDBF 算法产生了微小的偏差。

实验证明, TDBF 算法与 TBF 算法在标签阈值内时, 保持着同样的零标签漏读率, 但是在标签阈值外时, TDBF 算法的误判率大大小于了 TBF 算法的误判率。因此, TDBF 算法在静态理想环境中能够实现零漏读率、低误读率以及高数据压缩率。

实验 2. 我们将 50 个标签分别随机静止散落在 TDBF 冗余检测范围内和冗余检测范围外, 分别采集在 10 分钟的读取时间内经过 BF、TBF 与 TDBF 算法过滤后的剩余数据总量, 该场景增加了标签个数与数据采集的时间, 从而验证该算法能否在复杂场景中保持良好的数据压缩率。实验采集的数据如表 4 所示。

表 4 静态标签过滤数据

Table 4 Static tag filtering data

过滤算法	阈值内数据总数	阈值外数据总数
无	380820	347293
BF	50	50
TBF	12232	11094
TDBF	11402	2386

由表可知, 相对于较少标签的理想环境, 存在大量标签的复杂场景下, 由于标签数据间存在的相互干扰, 所以读取速率会有所下降。在这种情况中, BF 算法皆读取到了所有 RFID 标签数据, 证明没有产生标签 ID 误判现象。此时, 理论上读取 TBF 过滤后数据量为 15000, 实际过滤后阈值内数据量为 12232, 阈值外的数据总量为 11094, 漏读的数据主要是因为存在标签间的电磁相互干扰, 属于系统误差。理论上距离阈值内读取 TDBF 过滤后数据量为 15000, 而距离阈值内实际采集数据量为 11402, 距离阈值外的数据总量理想值为 0, 而此时产生了 2386 条数据信息。相较于 TBF 过滤后数据量, 由于此时标签在距离阈值内散落摆放, 所以靠近距离阈值处的标签可能存在较大的 RSSI 值的波动, 造成 TDBF 过滤时进一步产生了误判现象。

综上所述, 在复杂场景中, TDBF 算法较 TBF 算法产生了低漏读率, 但仍大大降低了误读率, 实现了漏读率与误读率的性能平衡, 平均数据压缩率得到了大大的提高, 能够有效减少冗余数据, 将系统专注于有效数据的采集。

4.1.2 动态场景实验

动态实验场景部署如图 9 所示, 实验中, 由实验操作人员携带若干标签, 从图中所示的距离阈值外位置, 缓慢靠近门禁出入口位置的读写器。采用时间布隆过滤器(TBF)以及时间距离布隆过滤器(TDBF)对采集数据进行过滤, 并进行分析对比。选取的 RSSI 阈值对应于实际场景中大约距离读写器 2m 的位置。考虑到人从距离阈值处到门禁读写器位置期间至少需要采集到一个数据, 且不能产生数据漏读, 还要兼顾 RFID 数据流的冗余过滤效率, 所以将时间阈值设置为 1 秒。为了验证动态环境下算法的过滤效率, 我们采取了如下实验。

实验 3. 我们首先将标签全部置于距离阈值外, 然后由实验者携带 5 个标签, 沿着如图 9 所示的运动轨迹, 缓慢进入 TDBF 检测范围, 采集经过 TDBF 与 TBF 过滤后标签数据, 查看 TDBF 能否过滤出有效数据。

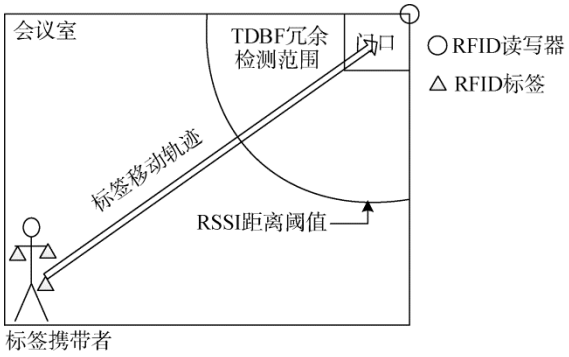

图 9 动态场景部署

Figure 9 Dynamic scenario deployment

如图 10 所示, 随着时间的推移, 携带标签的实验人员不断沿着轨迹缓慢靠近 RFID 读写器, 缓慢移动的目的在于使标签近似于静态状态, 从而模拟理想环境中的冗余数据过滤效率。此时, TBF 不考虑空间特性, 过滤出的数据量总体呈线性增长趋势, 其中偏离预期值的数据往往是由于在实验人员移动时, 标签不够稳定从而导致的漏读或误读现象。当实验人员在距离阈值较远处时, TDBF 几乎不会产生误判数据, 随着实验人员接近距离阈值, RSSI 值逐步增加且有一定的波动, 产生了较低的数据误判率, 在第约 16 秒进入距离阈值后, RSSI 的波动使得 TDBF 存在一定程度的漏读率, 但在 20 秒完全进入距离阈值范围内后, 此时 TDBF 保持了和 TBF 相似的标签读取率。

实验反映出距离阈值的附近存在着 RSSI 的波动范围, 在这个范围内, TDBF 算法会由于 RSSI 值的不稳定, 从而产生一定程度的漏读率与误读率, 但离开了这段波动范围, TDBF 能产生良好的过滤效果, 在距离阈值内, 能够产生近似 TBF 算法的过滤效果。从而实现了利用距离信息对 RFID 冗余数据进行空间粒度上的进一步过滤。

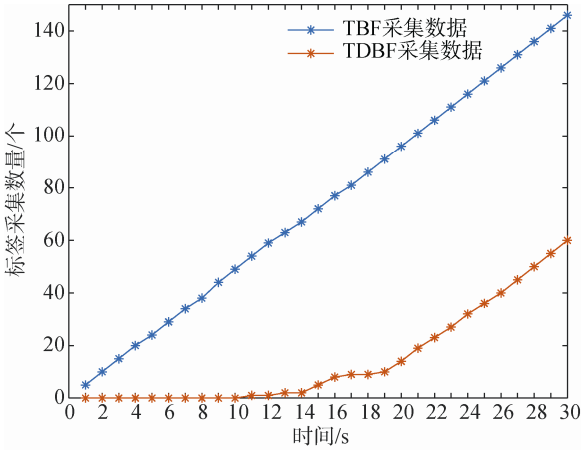

图 10　用 TBF 和 TDBF 过滤实时场景中移动标签
Figure 10　Filtering mobile tags in live scenes with TBF and TDBF

实验 4. TDBF 算法中, 为了进行更高效的数据过滤, 在距离阈值内, 我们更关心的数据是存在接近读写器的数据, 这样的数据往往更加能够体现出标签携带者有携带标签离开该区域的趋势, 而在实验 3 中, 标签的移动轨迹为一条不断接近读写器的直线,不能模拟真实环境中人物的行动轨迹, 因此, 在实验 4 中, 我们将人物的运动轨迹近似表示为图 11 所示的移动轨迹。

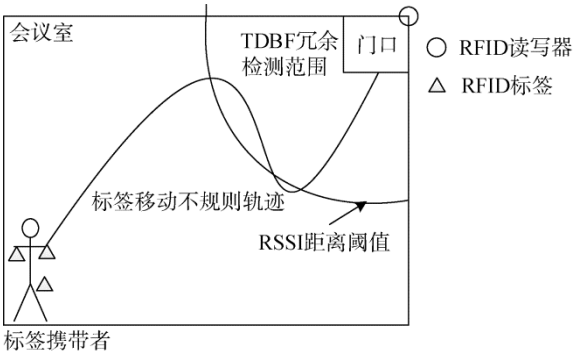

图 11　不规则移动轨迹
Figure 11　Irregular movement track

实验结果如图 12 所示, 根据图像可以观察到, 经过 TBF 过滤的数据因为不受空间条件的约束仍保持着线性增长的趋势, 而经过 TDBF 过滤的数据, 在前 10s 近似为 0, 表示该标签仍处于距离阈值外, 直接被过滤。12~17 秒期间, 能够观察到过滤后数据处于增长趋势, 表明该标签处于距离阈值内, 且向读写器靠近中, 18~23 秒期间, 过滤数据处于较为平缓的状态, 表明没有产生新的数据过滤信息, 此时该标签携带者可能处于远离读写器的状态, 而在 24~30 秒期间, 过滤后的数据又重新出现线性增长的趋势, 说明改标签携带人员又重新向读写器靠近, 有疑似携带标签离开会议场所的风险。边缘节点可以重新根据过滤后信息, 产生报警信息, 从而提示管理员有人员疑似携带标签离开重要会议、涉密等场所。

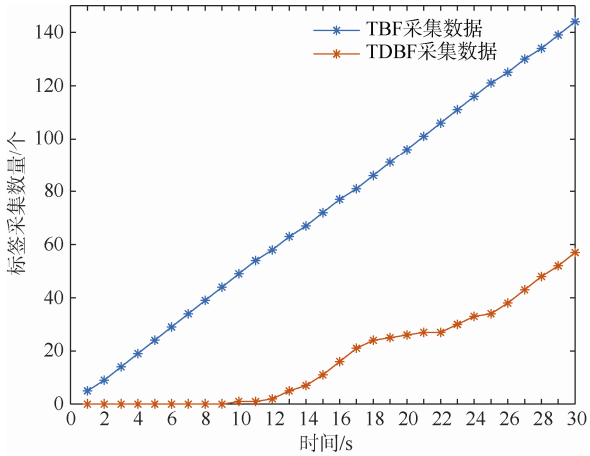

图 12　用 TBF 和 TDBF 过滤不规则移动标签
Figure 12　Filter irregular moving labels with TBF and TDBF

综上所述, TDBF 算法不论在静态环境还是动态环境中, 均能保持低漏读率、低误读率以及高数据压缩率。在动态移动场景中, 相对比 TBF 算法, 更能根据实际场景需求, 过滤出有效的报警数据。因此, 该算法能够实际应用于出入检测中。

4.2　算法与性能分析

RFID 数据流的不确定性, 使其存在着难以避免的多读、漏读现象。TDBF 中的误读现象是指将那些本来不属于冗余数据的 RFID 标签误判为冗余数据。TDBF 的误读率F_{fp}由三部分组成:

(1) 假定存在RFID数据流$S = \{s_1, s_2, \cdots, s_n\}$,$s_1 = < TagID1, Time1, RSSI >$, $s_2 = < TagID2, Time2, RSSI >$, $s_3 = < TagID3, Time2, RSSI >$, 其中 RSSI 值大于距离阈值, Time2 和 Time1 的差值大于时间阈值, TagID1 被映射到大小为 m 的数组第 k 位, 例如第 r、s 位, 则通过哈希函数插入后, 没有被置为 1 的概率为:

$$P_i = 1 - \frac{1}{m} \tag{1}$$

一共有 k 个哈希函数分别进行映射, 则 k 个哈希函数中没有一个对其置位的概率为:

$$P_k = \left(1 - \frac{1}{m}\right)^k \tag{2}$$

如果插入了 n 个元素, 但都未将其置位的概率为:

$$P_{kn} = \left(1 - \frac{1}{m}\right)^{kn} \tag{3}$$

TagID2 被映射到数组第 s、t 位, TagID3 被映射到第 r、t 位, 此时虽然 TagID3 在标签 ID 上不同, 应该判断为非冗余数据, 但是因为映射的位置上的时间和距离信息均满足非冗余条件, 所以产生了误判, 此时某一位的重复概率为:

$$P_{kn'} = 1 - \left(1 - \frac{1}{m}\right)^{kn} \tag{4}$$

又因为共映射了 k 位, 所以误判概率:

$$f_1 = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^{k} \tag{5}$$

产生该误判概率的原因是布隆过滤器方法固有的误判原因。因此, 基于布隆过滤器的改进算法都存在该误判率。

(2) 假定存在 RFID 数据流$S = \{s_1, s_2, \dots, s_n\}$,$s_1 = < TagID1,2,-40 >, s_2 =< TagID2,3,-50 >, s_3 =< TagID1,4,-45 >$,其中 RSSI 距离阈值为–60, 时间阈值为 3, TagID1 被映射到大小为 m 的数组的第 r、s 位, TagID2 被映射到数组第 s、t 位, 根据 TDBF 算法, 此时数据s_3由于存在 RSSI 值为–45 大于数组第 s 位中存储的 RSSI 值–50, 所以被判定为非冗余数据, 但在实际场景中, 该标签虽然在距离范围内, 但在时间阈值范围内且没有接近读写器, 所以应该认定为冗余数据。此时的误读率为f_2。

(3) RSSI 值本身存在着一定的波动性, 理想中距离读写器越近, RSSI 值越大, 但由于实际环境的复杂性, 可能存在虽然标签在靠近读写器, 但是其 RSSI 值变小的情况, 此时的误读率的大小是依据环境而定的f_3。

TDBF 中的漏读现象是指将那些本来属于冗余数据 RFID 标签误判为非冗余数据。TDBF 的漏读率由 RSSI 值本身的不可靠性产生的, 由于实际环境的复杂性, 标签远离读写器时, 其 RSSI 值可能变小, 因此将冗余数据误判为非冗余数据。在实际场景中, 应该尽量降低标签的漏读率, 因为很可能丧失了数据采集中的关键信息。

TDBF 实现了时间、空间效率与准确性的平衡。空间上, TDBF 采用了二维整型数组, 其大小为 m, 所以空间复杂度为$\theta(m)$, 其大小是时间布隆过滤器 TBF 的一倍。时间上, 新数据到达时, 最快只比较一次就能得出是否冗余, 处理 n 个数据只需要比较 n 次, 最好的时间复杂度为$\Omega(n)$, 最坏的情况的是针对每个新数据, 需要比较 k 次, 其中 k 为哈希函数的个数, n 个数据一共需要比较 $k*n$ 次, 此时最坏的时间复杂度$O(n)$, 因此平均时间复杂度为$\theta(n)$。

5 结论

RFID 数据流中包含了大量的冗余数据, 这些冗余数据往往没有实际的价值并且降低了系统的运行效率, 传统的冗余处理算法很难同时在时间上和空间上对数据进行精确的过滤, 为了兼顾数据的实时处理效率, 本文采用了布隆过滤器算法, 并基于时空的考虑, 提出了一种名为时间距离布隆过滤器(TDBF)的算法, 该算法兼顾了 RFID 标签的读取时间和读取距离, 从时间粒度的冗余和空间粒度的冗余两个角度进行冗余去除, 极大的降低了 RFID 数据的冗余度。通过实验, 将 TDBF 算法同时间布隆过滤器算法进行对比, 发现该算法在保证漏读率较低的情况下, 大大降低了数据的误读率, 实现了性能的提高。同时, 对移动中的标签进行过滤, 发现该算法能够实际应用于出入标签的冗余过滤。

然而 TDBF 算法仍存在一定的局限性, 首先, 该算法目前的设计和验证针对的单个读写器的情况[30,31], 不适用于大规模应用场景, 因此需要改进该算法, 以适用于不同的应用场景; 其次, 本文中采用的 RSSI 值存在一定的不确定性, 且受环境、标签质量的影响较大, 如何利用算法进一步的降低 RSSI 值波动带来的误差影响或者利用新的元组模型代替 RSSI 值进行距离的大致评测成为下一步工作的重点[32]。

参考文献

[1] Derakhshan. R, M. E. Orlowska, and X. Li, “RFID Data Management: Challenges and Opportunities.” *IEEE International Conference on Rfid*. pp. 175-182, 2007.

[2] Bai. Y, Wang. F, Liu. P, “Efficiently filtering RFID data streams” Proc *Cleandb Workshop*, pp. 50-57, 2006.

[3] Y. Zhang, Yibin. Tang, and Xufei. Li, “Overview of RFID Data Cleaning Algorithm”, *Micro Processing*. Vol. 31, pp:32-36, 2016. (张燕, 汤一彬, 李旭斐, “RFID 数据清洗算法概述”, *微处理机*, 2016, 37(1): 32-36).

[4] Wang, Jinlin, et al, “A survey on data cleaning methods in cyberspace.” *2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, pp. 74-81, 2017.

[5] Tian. W, Xue. R, Dong. X, and Wang. H, “An Approach to Design and Implement RFID Middleware System over Cloud Computing.” *International Journal of Distributed Sensor Networks*. pp. 1-13, 2013.

[6] Vahdati, Farahnaz , R. Javidan , and A. Farrahi, “A new method for data redundancy reduction in RFID middleware.” *International Symposium on Telecommunications*. pp. 175-180, 2010.

[7] Gonzalez. H, Han. J, Li. X, and et al, “Warehousing and analyzing massive RFID data sets.” *International Conference on Data Engineering IEEE (ICDE)*, vol. 6, pp. 83-94, 2006.

[8] Che. Qing. Jin, Q. Wei. Ning, and Zao. Ying, “Analysis and Management of Streaming Data: A Survey.” *Journal of Software*, vol.

15, pp. 1172-1181, 2004.

[9] Iyer. Vasanth, S. S. Iyengar, and Niki Pissinou, "Ensemble stream model for data-cleaning in sensor networks." *AI Matters*. pp:29-32, 2015.

[10] Ramírez-Gallego, Sergio, et al, "A survey on data preprocessing for data stream mining: Current status and future directions." *Neuro computing*, pp:39-57, 2017.

[11] Le. Z, "Research and development of RFID middleware data processing[Ph.D.dissertation]," Shanghai Jiao Tong University, 2008 (张乐. "RFID 中间件数据处理研究与开发[D]", 上海交通大学, 2008.)

[12] Jeffery, Shawn R , Alonso G, and Franklin M J, "A Pipelined Framework for Online Cleaning of Sensor Data Streams." *International Conference on Data Engineering IEEE*, pp. 140-140, 2006.

[13] Kamaludin, Hazalila, Hairulnizam Mahdin, and Jemal H. Abawajy, "Filtering redundant data from RFID data streams." *Journal of Sensors,* vol. 2016, pp. 1-7, 2016.

[14] Ma. Meng, Ping. Wang, and Chao-Hsien Chu, "Redundant reader elimination in large-scale distributed RFID networks." *IEEE Internet of Things Journal* vol. 5, no. 2, pp:884-894, 2018.

[15] Gonzalez. Hector, J. Han, and X. Shen . "Cost-Conscious Cleaning of Massive RFID Data Sets." *IEEE International Conference on Data Engineering*, pp. 1268-1272, 2007.

[16] Qiaomin. Lin, and et al, "A method of cleaning RFID data streams based on Naive Bayes classifier." *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 21, no.4, pp. 237-244, 2016.

[17] Luo. YJ, Jiang. JG, Wang. SY, Jing X, Ding C, Zhang ZJ, and Zhang YF, "Filtering and cleaning for RFID streaming data technology based on finite state machine." *Journal of Software*, vol. 25, no. 8, pp. 1713-1728, 2014.
(罗元剑, 姜建国, 王思叶,等. "基于有限状态机的 RFID 流数据过滤与清理技术". *软件学报*, 2014(8):1713-1728.)

[18] Mahdin. Hairulnizam, "A Review on Bloom Filter Based Approaches for RFID Data Cleaning." *Lecture Notes in Electrical Engineering,* vol. 285, pp. 79-86, 2014.

[19] Zhijian. Y, Yingwen. C, Jiajia. Y, Yan. J and Shuqiang Y, "Research on typical Bloom filters and their data flow applications." *Computer Engineering*, vol. 35, no. 7, pp.5-7, 2009.
(袁志坚, 陈颖文, 缪嘉嘉, 贾焰, 杨树强, "典型 Bloom 过滤器的研究及其数据流应用." *计算机工程,* 2009, 35(7): 5-7.)

[20] Bloom. Burton, "Space/Time Tradeoffs in Hash Coding with Allowable Errors". *Ipsj Magazine*, vol. 13, pp. 422-426, 1970.

[21] Lee. Chun Hee, and C. W. Chung, "An approximate duplicate elimination in RFID data streams." *Data & Knowledge Engineering*, vol. 285, no. 12, pp. 1070-1087, 2011.

[22] Mahdin. H, Abawajy. J, "An approach for removing redundant data from RFID data streams." *Sensors.* vol. 11, no. 10, pp. 9863-9877, 2011.

[23] Rui. Wu, L. Guoqiong, and D. Guoqiang, "Filtering Redundant RFID Data Based on Sliding Windows." International Conference on Management of E-commerce & E-government, pp. 187-191, 2014.

[24] Guoqiong. L, Jun. Z, Ni. H, Xiaomei. H, Zhiwei. H, and Changxuan, W, "Approximately Filtering Redundant Data for Uncertain RFID Data Streams." IEEE International Conference on Mobile Data Management, pp. 56-61, 2017.

[25] Yongli. W, Chuan. W, and Xiaohui. J, "RFID redundant data cleaning algorithm based on space-time Bloom filter" *Journal of Nanjing University of Science and Technology*, vol. 39, no. 3, pp. 253-259, 2015.
(王永利, 王川, 蒋效会, "基于时空布隆过滤器的 RFID 冗余数据清洗算法." *南京理工大学学报*, 2015, 39(3): 253-259.)

[26] Rui. W, "Research on RFID redundant data filtering based on sliding window[M.S. dissertation]." Jiangxi University of Finance and Economics, 2014.
(吴锐. "基于滑动窗口的 RFID 冗余数据滤重研究[D]", 江西财经大学, 2014.)

[27] Ali G Ç, ˙ Ik-Gerber, Oktepe A B. , Li, S. , & Li, N, "Analysis of the variability of RSSI values for active RFID-based indoor applications." *Turkish Journal of Engineering & Environmental Sciences*. vol. 37, no. 2, pp. 186-211, 2013.

[28] W. Duan, Chunjiang. Liu, Yueshan. Wu, "Application of RSSI in RFID Reader." *Computer Engineering*. vol. 36, no. 22, pp. 289-290, 2010.
(段璞, 刘春江, 武岳山, "RSSI 在 RFID 读写器中的应用", *计算机工程*, 2010, 36(22): 289-290.)

[29] H. Hadj. M, R. Touhami; S. Tedjini, "Cleansing RFID data based on RSSI estimation." *International Conference on Ubiquitous Wireless Broadband (ICUWB)*, pp. 1-4, 2017.

[30] Yu. Wei, Liang. F, He. X, Hatcher. W. G, Lu. C, and Lin. J, "A Survey on the Edge Computing for the Internet of Things." *IEEE Access*. vol. 6, pp:6900-6912, 2018.

[31] Shi. W, Cao, J, Zhang. Q, Li. Y, and Xu. L, "Edge Computing: Vision and Challenges." *IEEE Internet of Things Journal*. vol. 3, no. 5, pp: 637-646, 2016.

[32] Jing. Su, and C. Guoqiang, "A new RFID middleware architecture design." *International Conference on Computer & Automation Engineering IEEE*, pp: 637-639, 2010.

黄伟庆 于 2005 年北京邮电大学计算机应用技术专业获得工学硕士学位。现在北京交通大学攻读博士学位。中国科学院信息工程研究所正高级工程师, 博士生导师, 第四研究室主任。主要研究方向为信号处理理论与技术、电磁声光检测与防护、物联网安全等。Email: huangweiqing@iie.ac.cn

张艳芳 于 2012 年在北京邮电大学电路与系统专业获得硕士学位。现在中国科学院大学信息安全专业攻读博士学位。现任中国科学院信息工程研究所工程师。研究领域为信息安全、物联网。Email: zhangyanfang@iie.ac.cn

曹籽文 于2018年在南京邮电大学物联网专业获得学士学位。现在中国科学院大学网络空间安全专业攻读硕士学位。研究领域为物联网、边缘计算。研究兴趣包括: 边缘计算、数据挖掘、物联网安全。Email: caoziwen@iie.ac.cn

王思叶 于 2006 年上海交通大学信息与信号系统专业获得硕士学位。现在北京交通大学信息安全专业攻读博士学位。现任中国科学院信息工程研究所高级工程师。研究领域为信息安全、物联网等。Email: wangsiye@iie.ac.cn