基于时隙补偿机制的 RFID 混合查询树算法(AHQT)

谢晓辉, 林正浩

(同济大学 电子信息工程学院,上海 201804)

摘 要: 标签防碰撞算法的优劣决定了 RFID 系统性能的好坏,针对标签识别的 RFID 系统前人已经提出了许多算法,但都有明显的缺点,包括识别速度慢、不稳定等. 已有的算法主要分为两大体系:基于 ALOHA 的算法、查询树算法. 基于混合查询树算法,引入时隙补偿机制和采用特定编码方式(曼彻斯特编码),由标签中每三位中 1 的个数,决定标签响应时隙,大大降低了碰撞时隙,提高了识别效率.

关键词: 射频识别; 混合查询树; 标签; 阅读器; 防碰撞

中图分类号: TN 911 **文献标识码:** A **文章编号:** 1000-5137(2013)06-0599-06

0 引 言

RFID(Radio FrequencyIdentification)射频识别技术是一种利用无线电射频信号进行通信的非接触式自动识别技术. 有源标签的出现和 RFID 技术在高速移动物体中的应用,迫切需要读写器在有限时间内高效快速地识别大量标签. 标签防碰撞算法就是要解决在读写器有效通信范围内,多个标签同时与读写器进行通信的问题. 一个典型的 RFID 系统主要由读写器和标签两大部分组成.

现阶段 RFID 标签在仓库管理、食品安全、二代身份证、运动计时和入侵检测等领域都开始投入使用. 目前,国际上有三大标准体系,分别是 ISO 标准体系、EPC Global 标准体系和 Ubiquitous ID 标准体系. 在欧美,TI、Intel、Philips、STMicroelectronics 等公司都在 RFID 技术上投入了巨资进行实用芯片的研发;IBM、Microsoft 和 HP 等也在积极开发相应的软件来支持 RFID 的应用. 目前在欧美,RFID 已经逐渐深入到生活的道路交通、车辆管理、身份识别等. 物流方面诸如沃尔玛、吉列、强生、宝洁等已经开始实用 RFID 标签.

当前 RFID 系统标签防碰撞算法主要可以分为两大类:基于时隙的 ALOHA 算法、查询树算法. 基于时隙的 ALOHA 算法主要包括纯 ALOHA、时隙 ALOHA、帧时隙 ALOHA、动态帧时隙 ALOHA. 其中纯 ALOHA 和时隙 ALOHA 算法的吞吐率分别为 $G \cdot e^{(-2G)}$ 和 $G \cdot e^{(-G)}$. 经计算两者最大吞吐率分别为 18.4% 和 36.8%. ALOHA 算法简单,易于实现,但不稳定性高,可能会出现标签长时间无法识别而导致标签饿死现象. 查询树算法主要有二进制搜索树和查询树算法(QT)以及基于补偿时隙算法的 HQT 算法. 二进制搜索树能确保标签百分百识别但其查询深度大,空闲时隙多,延迟较大,所以本文作者基于 HQT 提出的 AHQT 可以彻底改善空闲时隙问题,提高查询效率.

收稿日期: 2013-09-09

作者简介: 谢晓辉(1988 -),女,同济大学电子信息工程学院硕士研究生;林正浩(1957 -),男,同济大学电子信息工程学院教授.

1 改进算法关键技术分析

1.1 查询树算法(QT)

QT 算法即二叉树查询,将碰撞的标签分为两组,阅读器以“0”或“1”开始查询过程,如果发生碰撞则这两组分别在原有的前缀后面分别扩展一位“0”和一位“1”,得到新的查询前缀重复查询过程,直到识别出所有标签为止. 查询树的内部节点为碰撞周期,叶子节点为空闲周期或者成功周期. 查询树的缺点为空闲周期多,查询时间长.

1.2 曼彻斯特编码

曼彻斯特编码可以轻松识别碰撞发生的位置. 采用 IEEE 802.4(令牌总线)和低速版的 IEEE 802.3(以太网)中的规定,低 - 高电平跳变表示 1,高 - 低的电平跳变表示 0. 如有标签 T1(010111)和标签 T2(011001),曼彻斯特编码如图 1 所示,其中,2、3、4 位为碰撞位.

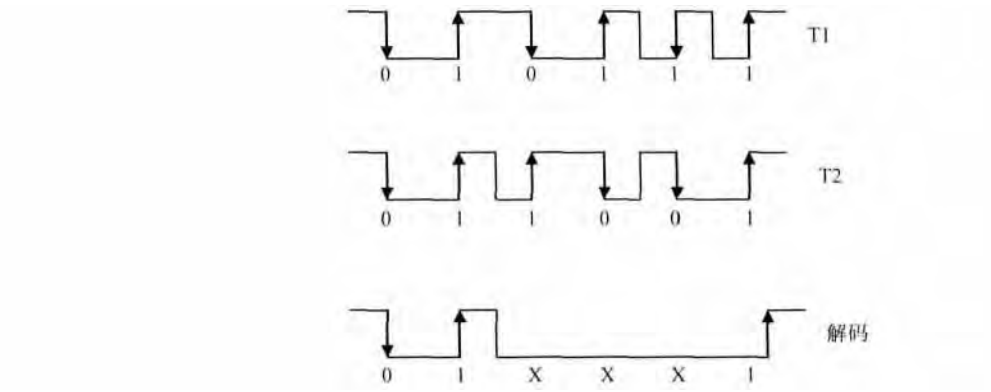

图 1 曼彻斯特编码

1.3 时隙补偿机制

时隙补偿机制能消除多叉树机制带来的空闲周期增加的问题. 在 QT 算法中,匹配前缀的标签立即响应阅读器的反馈. HQT 算法中标签会延时 0 ~ 3 个时隙(slot)再反馈阅读器. 补偿时隙的大小是由标签要被扩展的 2 位比特来决定的. 假定 2 位比特位 YY,YY 和补偿时隙 slot 的关系如表 1 所示.

表 1 补偿时隙算法

YY	00	01	10	11
slot	0	1	2	3

2 位比特位与时隙的对应关系属人为规定,本文作者只是按常规方法规定不同的比特位在不同的时隙响应,只要定义了一种时隙补偿的策略,能到达标签可以错开时隙响应的目的即可. 为了实现补偿机制必须在每个标签上设置一个计数器. 假设阅读器范围内有 3 个标签 T1(0101),T2(0110),T3(0111),阅读器发送一个查询前缀 01,则 T1,T2,T3 按表 1 得出对应的补偿时隙分别为 1,2,3. 当标签计数器的值等于 slot 时,标签反馈自己的数据给阅读器. 采用时隙补偿机制可以减少一定的不必要的空闲周期,减少阅读器的识别时延.

1.4 混合查询树算法(HQT)

HQT 算法结合了 QT 算法和时隙补偿机制,用四叉树机制代替 QT 中的二叉树机制,通过减少碰撞周期和空闲周期来减少标签识别过程中的延时. HQT 中如果标签发生碰撞,会在之前的查询前缀的基础上增加 2 位查询码再进行询问,减少了碰撞周期但也增加了空闲周期. 因此引入时隙补偿机制,来减少空闲周期. 当标签 ID 的前缀与阅读器的询问串匹配时,标签并不像二叉询问树算法那样直接响应阅

读器,而是延时 w($w=0\sim3$)个时隙再响应. 每个标签的补偿时隙由标签匹配前缀的后 2 位决定. 对 HQT 算法的分析都将依照本研究之前介绍的时隙补偿算法为基础.

依然假设阅读器范围内有 3 个标签 T1(0101),T2(0110),T3(0111),阅读器发送一个查询前缀 01,则 T1,T2,T3 按表 1 得出对应的补偿时隙分别为 1,2,3. 当标签计数器的值等于 *slot* 时,标签反馈自己的数据给阅读器. 为了实现时隙补偿功能,标签需要增加一个定时器在固定的时隙发送信息给阅读器.

HQT 性能大大优于 QT,而 HQT 中阅读器是根据信道忙周期计算传输的比特位数,从而计算子树的数目,HQT 的缺点就在于此. 如图 2 所示,如果没有 T2 发送到阅读器,信道忙周期依然为 T,即说明阅读器认为查询子树依然为 01,10 和 11. 所以在实际没有 T2 存在的情况下会产生 1 个空闲周期. 如果查询子树为 00 和 11,此情况会引入 2 个空闲周期. 本算法 AHQT 就致力于解决该问题.

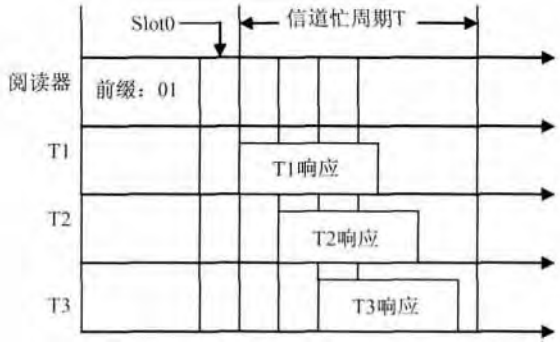

图 2　信道忙周期示意图

2　改进的混合查询树算法(AHQT)

2.1　AHQT 算法基本原理

本算法在 HQT 的基础上将依然继续沿用曼彻斯特编码方式来识别碰撞位,将四叉树查询变为八叉树查询,同时优化时隙补偿机制得到算法 AHQT(advanced hybrid query tree). 以下详细说明改进点:

(1) 搜索后匹配前缀增加 3 位而不是 2 位,这样可以减少碰撞周期.

(2) 采用曼彻斯特编码和改进的补偿时隙法. 改进的补偿时隙法中是根据匹配前缀后 3 位比特中“1”的个数来决定延迟几个时隙响应. 假如查询前辍(prefix)后 3 位比特为 YYY,改进的时隙补偿机制的响应时隙如下表 2 所示.

表 2　改进补偿时隙算法

YYY	000	001	010	101	011	101	110	111
Slot	0	1			2			3

(3) 标签端有一加法器,可计算 prefix 的长度 L,标签响应只需要发送标签的 $L+1$ 位至 N 位到阅读器.

2.2　AHQT 算法流程

将算法流程分为阅读器端和标签端分别阐述.

令 N 为标签总长度,*slot* 为当前的时隙数.

2.2.1　标签端的工作流程

Ⅰ 接收 prefix,判断其是否为“#”,如果是,转到Ⅳ,否则继续;

Ⅱ 计算 prefix 位数 L,如果自身 ID 前 L 位如果与 prefix 不同,此次不响应,否则继续;

Ⅲ 统计自身 ID 的 $L+1$ 到 $L+3$ 位“1”的个数 C;

Ⅳ 在时隙 C 响应阅读器;

Ⅴ将第 $L+1$ 到标签末尾的值发送给阅读器. 转到Ⅰ.

2.2.2 阅读器端的工作流程

Ⅰ 判断 queue 是否为空,如果是空,转到Ⅵ,否则继续;

Ⅱ 读取并删除栈顶的 prefix,将其发送给标签;

Ⅲ 每个时隙内扫描标签发送的高 3 位信息,将其存入 buffer;

Ⅳ 根据 buffer 中存储的曼彻斯特码和当前时隙判断发生碰撞的比特位,扩展 prefix 将其放入栈底;

Ⅴ当前 prefix 查询完毕继续发送新的 prefix,直到所有标签被识别,此时堆栈为空;

Ⅵ 结束识别.

2.3 AHQT 算法实例分析

现有 6 个 9 位的标签 ID,如下:

T1:000 010 001　　T2:001 010 010　　T3:010 111 011

T4:101 010 101　　T5:101 100 100　　T6:111 111 110

标签的识别过程如下:

(1) 阅读器初始化清空堆栈并发送请求命令#. 此时阅读器范围内的所有标签均响应. 标签根据自己高 3 位"1"的个数在不同的时隙响应阅读器. Slot0:只有标签 T1 在此时隙响应,无碰撞,成功识别. Slot1:T2,T3 均在此时隙响应,出现碰撞. Slot2:T4,T5 同时在此时隙响应,也有碰撞产生. Slot3:只有 T6 在此时隙响应,无碰撞成功识别.

(2) 阅读器接收到标签 ID 的前 3 位,在 Slot1 解码得到 0XX,故将 001 和 010 放入 prefix 堆栈. 同时在 Slot2 接收到的 ID 前 3 位解码得 101,说明碰撞位发生在第三位之后,继续扫描接下来的 3 位,解码得 XX0,故又将 101010 和 101100 放入栈底. 此时 prefix 堆栈共有 4 个成员.

(3) 阅读器发送新的 prefix "001",只有 T2 响应,识别. 阅读器继续发送栈顶的 prefix "010",T3 被识别. 接下来依次发送"101010"和"101100",T4 和 T5 也相继被识别.

(4) 最后 prefix 堆栈为空,所有标签均被识别.

整个识别过程如图 3 所示.

2.4 AHQT 算法仿真结果

标签 ID 采用 EPC96 编码的标签. 该标签由 8 位的标头,28 位的厂商识别代码,24 位的对象分类代码及 36 位的序列号构成. 仿真软件为 Matlab. 在相同的标签数量的前提下比较算法 QT、HQT 和 AHQT 的阅读器询问次数,即发送的前缀数. 仿真分两种情况考虑,一种为顺序产生的标签,即标签分布从 96 位全 0 开始到 96 位全 1 为止. 另一种情况为随机产生 96 位的标签. 图 4 为随机产生标签不同算法查询次数的对比结果,图 5 为顺序标签不同算法查询次数的对比结果. 图 4、5 以标签数作为横坐标,查询次数作为纵坐标. 图 4、5 中方形节点曲线代表 AHQT,圆形节点曲线代表 HQT,菱形节点曲线代表 QT. 由图 4 和图 5 可看出 AHQT 算法无论是在标签随机产生还是顺序产生的前提下,相同标签数目所需的查询次数都远少于 QT 和 HQT,均为图 4、5 内最下方的线条.

在 QT 算法中,每次遇到碰撞的时候,算法就使前缀扩展 1 位,而在 HQT 中,算法使用扩展 2 位来加快了树的分裂,减少了碰撞前缀,但同时也产生了比 QT 算法更多的空闲前缀. 本文作者提出的改进型 HQT 算法 AHQT 很好地解决了这些问题. AHQT 算法中,标签从阅读器收到前缀,如果前缀与自己相同,再根据 C 值选择一个时隙对阅读器进行响应,并发送匹配前缀后剩余的所有位. 如果有碰撞发生,阅读器将用 buffer 和 slot 的信息对标签发送的情况进行准确的识别,这样就能避免空闲前缀的产生. 对比于其他两种算法,AHQT 大大地减少了发送前缀的数量,减少了阅读器的问询次数和识别时间,降低了发送的数据量.

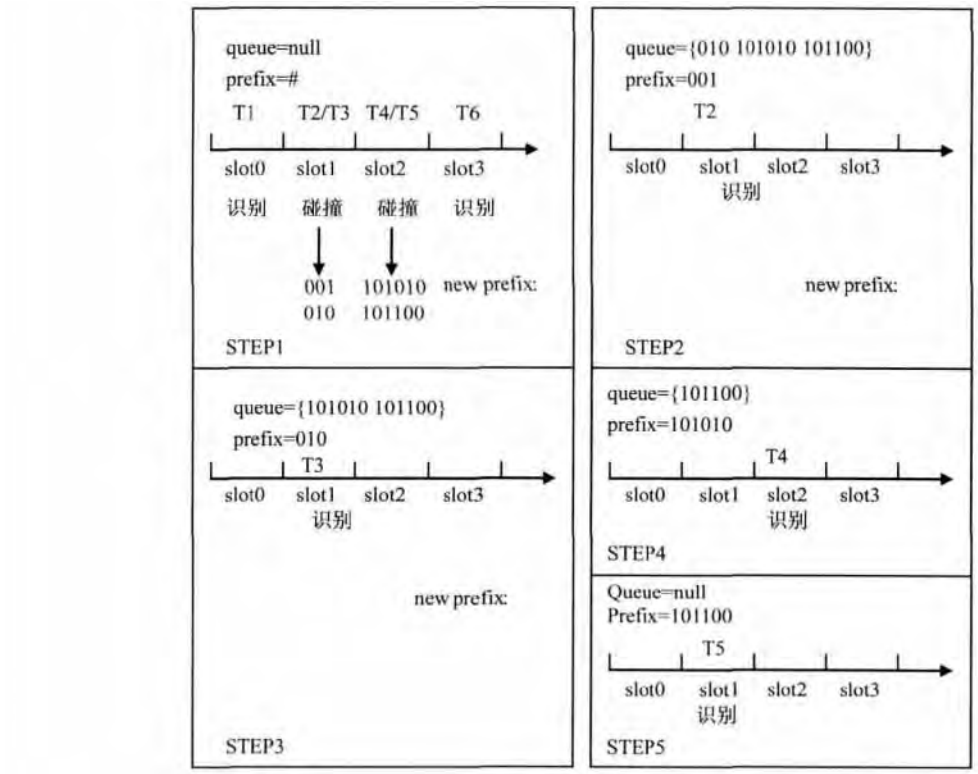

图 3　AHQT 算法实例图

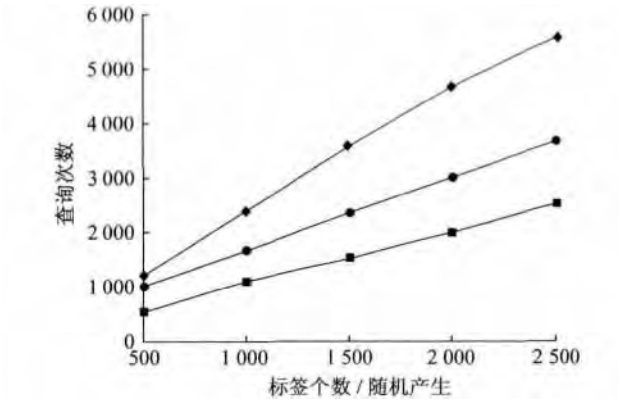

图 4　标签随机产生时
QT、HQT、AHQT 算法查询次数比较图

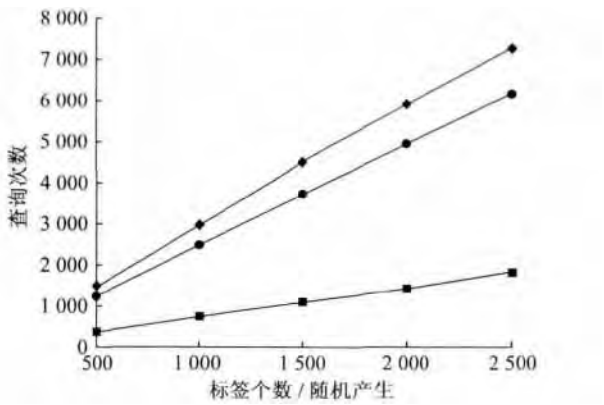

图 5　标签个数顺序产生时
QT、HQT、AHQT 算法查询次数比较图

3　结　语

本文作者在找寻优化算法时分别尝试了基于八叉树和十六叉树的改进算法，经计算十六叉树仅在查询速度方面有微弱的优势，但是会引入更多的空闲时隙，所以最终采用了八叉树的改进算法. 改进的时隙补偿机制可以让标签的发送有的放矢，不会导致大量的冲突产生，而且阅读器也不会有空闲的发送周期产生. 曼彻斯特编码可以有效地识别碰撞位，也给算法提供了极大的帮助.

参考文献:

[1] 丁治国. RFID 关键技术研究与实现[D]. 合肥:中国科学技术大学,2009.
[2] 叶传玲. RFID 标签防碰撞算法研究[D]. 南京:南京邮电大学,2013.
[3] 崔士津. RFID 系统关键技术研究[D]. 哈尔滨:哈尔滨工程大学计算机科学与技术学院,2008.
[4] 徐东升. RFID 系统防冲突算法的研究与实现[D]. 哈尔滨:哈尔滨工程大学计算机科学与技术学院,2009.
[5] 廖传书,付泰. 射频识别系统的防碰撞算法研究[J]. 国外电子元器件,2008,16(9):6-8.
[6] CHIO J H,LEE D,LEE H. Query tree-based reservation for efficient RFID tag anti-collision[J]. IEEE Communications Letters,2007,11(1):85-87.
[7] MATHYS P,FLAJOLET P. Q-ary collision resolution algorithm sin random-access systems with free or blocked channel access[J]. IEEE Trans Inform,1985,31(4):217-243.

Advanced hybrid query tree algorithm based on slotted backoff mechanism in RFID

XIE Xiaohui, LIN Zhenghao
(Electronical Information Engineering College,Tongji University,Shanghai 201804,China)

Abstract: The merits of performance quality for a RFID system are determined by the effectiveness of tag anti-collision algorithm. Many algorithms for RFID system of tag identification have been proposed,but they all have obvious weaknesses,such as slow speed of identification,unstable and so on. The existing algorithms can be divided into two groups,one is based on ALOHA and another is based on query tree. This article is based on the hybrid query tree algorithm,combined with a slotted backoff mechanism and a specific encoding (Manchester encoding). The number of value "1" in every three consecutive bits of tags is used to determine the tag response time slots,which will greatly reduce the time slot of the collision and improve the recognition efficiency.
Key words: RFID; hybrid query tree (QT); tag; reader; anti-collision

(责任编辑:包震宇)