物联网中 RFID 位匹配防碰撞算法*

江 雨，马满福
（西北师范大学 数学与信息科学学院，甘肃省物联网工程研究中心，兰州 730070）

摘 要： 在对现有的二进制搜索算法、4 线树形查询算法及混合查询树算法进行理论分析的基础上，提出了一种基于碰撞位匹配的自适应混合树防碰撞算法。新算法是根据检测标签 EPC 编码最高碰撞位连续个数的匹配信息，在二叉树和四叉树中进行动态自适应地选择分叉数，引入堆栈和后退策略，使得搜索性能进一步改善。通过对算法的分析和仿真实验结果，新算法有效地减少了识别总次数，缩短了识别时间，大幅提高了搜索效率和吞吐率。

关键词： 物联网；射频识别；位匹配；混合树；防碰撞
中图分类号： TP301.6；TP393 **文献标志码：** A **文章编号：** 1001-3695(2012)01-0088-04
doi:10.3969/j.issn.1001-3695.2012.01.024

RFID bit match anti-collision algorithm in Internet of things

JIANG Yu, MA Man-fu
(*College of Mathematics & Information Science, Northwest Normal University, IoT Center of Gansu, Lanzhou* 730070, *China*)

Abstract: This paper analyzed the existing binary query tree algorithm, 4-ary query tree algorithm and hybrid query tree algorithm. Then it presented the adaptive hybrid collision-bit tree which was based on the matching of anti-collision algorithm. New algorithm was based on the highest collision detection code tag EPC number of consecutive matching information, then adaptively selected the bifurcation in the binary tree and the quadtree. The search performance of the algorithm was further improved by introducing the stack and back strategies. Through analysis of the algorithm, the simulation results show that the new algorithm effectively reduces the total number of identified, shortens the identification time, and significantly improves the search efficiency and throughput.

Key words: Internet of things (IoT); radio frequency identification(RFID); bit match; mixed tree; anti-collision

0 引言

射频识别(RFID)技术是一种利用无线射频方式在阅读器和电子标签之间进行非接触双向数据通信，以达到信息识别的目的[1]。与传统的条形码识别技术相比，由于该技术具有穿透性强、识别速度快、识别距离远、安全性高、多目标识别等诸多优点，现已广泛应用于物流、交通、食品安全、军事等领域。在 RFID 系统中，每一个标签都有唯一的一个电子产品代码(electric product code，EPC)[2]来标记物品在供应链中移动的相关动态信息，标签中存储了需要被识别和交互的数据，阅读器与标签之间是通过射频信号进行耦合的。借助于 RFID 技术可使得处在这个体系下的任何物体都可以利用 EPC 编码的唯一性进行标志，从而成为网络中的一个节点，实现物体与网络的连接、交互、追踪和准确定位。

当 RFID 系统运行时，可能有很多的标签处于阅读器的作用范围之内，当这些标签在同一时刻向阅读器传输信息时，就不可避免地会出现相互干扰的现象，这种现象称为碰撞。防碰撞算法就是要提出相应的策略和方法使得阅读器对各个标签都能够准确识别。高效的防碰撞算法是改善 RFID 系统中标签识别吞吐率一个主要因素[3]，是当前物联网领域的研究热点。

现有的防碰撞算法主要分为两类：基于时隙的 Aloha 协议随机性算法；基于二叉树搜索的确定性算法[4]。这两类算法都属于时分多路，它们分别存在各自的优缺点。Aloha 协议算法的复杂度及对标签的要求低，但由于该类算法的时隙是随机分配的，这就意味着存在某些特定的标签在相当长的一段时间内无法被识别出来的可能性，一般包括时隙 Aloha 算法(slotted Aloha，SA)[4]、动态时隙 Aloha 算法(dynamic slot allocation，DSA)[4]。二叉树搜索算法虽然比较复杂，识别时间相对较长，但标签的识别率可达到 100%，所以这类算法被称为确定性算法，如跳跃式动态树(jumping and dynamic searching，JDS)冲突树的 RFID 自适应防碰撞(adaptive anti-collision arithmetic base on collision-tree，ACT)算法[5]、堆栈存储的 RFID 防冲突(reduced dynamic binary algorithm based on stack，RDS)算法[6]等。考虑到在选择二叉树搜索算法进行标签识别时，当待识别的标签数量较多，就会频繁出现碰撞，而在每次发生碰撞后，仅仅只产生两个分支，所以搜索效率低下。而基于四叉树搜索算法的混合搜索(hybrid query tree，HQT)算法[7]是使用时隙延时机制来减少查询过程中的闲置时期，而延时器的引入影响了算法的性能。因此，本文在二叉树和四叉树搜索算法的思想上，

收稿日期：2011-06-04；修回日期：2011-07-24 基金项目：国家教育部科学技术研究重点资助项目(208148)；甘肃省科技攻关项目(2GS064-A52-035-03)

作者简介：江雨(1983-)，男，安徽桐城人，硕士研究生，主要研究方向为并行计算、网格计算、物联网(jyu1983@163.com)；马满福(1968-)，男，副教授，博士(后)，主要研究方向为计算机系统结构、网格计算、物联网.

结合 HQT 算法和自适应多叉树防碰撞(adaptive multi-tree search anti-collision algorithm, AMS) 算法[8] 的优点提出了一种根据检测最高碰撞位连续个数来动态自适应地确定搜索树的分叉数,即在某分支下检测出最高碰撞位连续的个数大于或等于 2,则选择四叉树搜索,否则选择二叉树搜索。

1　防碰撞算法原理

目前,很多 RFID 系统都采用比特位冲突监测协议作为碰撞检测的方案。ISO14443 规定: 阅读器到标签的命令传输采用密勒码,标签到阅读器的数据传输采用曼彻斯特编码。在曼彻斯特编码中,每一位的中间有一跳变,位中间的跳变既作时钟信号,又作数据信号; 从低到高跳变表示“0”,从高到低跳变表示“1”,若无状态跳变,则被视为非法数据。当多个同时返回的标签编码发生碰撞后,那么在碰撞位上的上升沿和下降沿就会相互抵消,从而丧失曼彻斯特编码格式,所以,这种编码格式能够很准确地检测出碰撞位。假设有三个标签,其 EPC 编码分别为 tag1: 10110101、tag2: 11001001 和 tag3: 11000011。当这三个标签同时发送自己的 EPC 编码信息时,利用曼彻斯特编码识别碰撞位位置示意如图 1 所示。

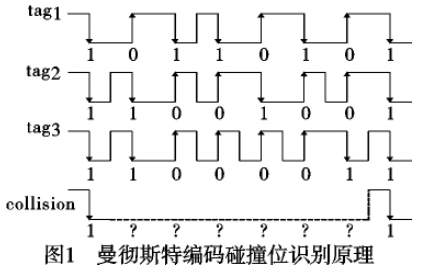

图1　曼彻斯特编码碰撞位识别原理

图 1 中,标签发送 EPC 编码的序列为 D_7、D_6、D_5、D_4、D_3、D_2、D_1、D_0,而阅读器无法识别 D_6、D_5、D_4、D_3、D_2 和 D_1 位,可知在该阅读器作用的范围内存在多个标签,并发生碰撞。

2　碰撞位匹配的自适应混合树防碰撞算法

2.1　算法原理

阅读器发送命令,那些被选中的标签向阅读器传输其自身的 EPC 编码信息,阅读器在接收到有反馈信息时,检测本次所请求的响应标签的碰撞情况。在检测到有碰撞发生时,根据碰撞位的连续信息分裂该组标签,即在本组标签中,如果检测出自最高碰撞位起连续发生 2 位以及以上的 EPC 编码碰撞,则将该组标签分裂为“00”“01”“10”“11”四个标签子组。如果最高碰撞位与次高碰撞位不连续时就意味着自最高碰撞位起连续发生 1 位编码碰撞,这种情况下,根据最高碰撞位位置将该组标签分裂为“0”和“1”两个标签子组,并将已知标签编码入栈。阅读器持续探测和分裂子标签,直到标签信息不发生碰撞为止,此时,该标签就能够被准确地识别。算法采用后退策略,直到所有标签识别完成。所谓后退,就是阅读器在成功识别出一个标签或空分支后,返回至前一次发生碰撞的位置。由于新的算法是根据检测标签 EPC 编码最高碰撞位连续个数的匹配信息,在二叉树和四叉树中进行动态自适应地选择分叉数的这样一种思想,新的算法称之为基于碰撞位匹配的自适应混合树防碰撞算法(adaptive hybrid query tree anti-collision algorithm based on collision-bit match, ABM) 。算法的流程框图如图 2 所示。

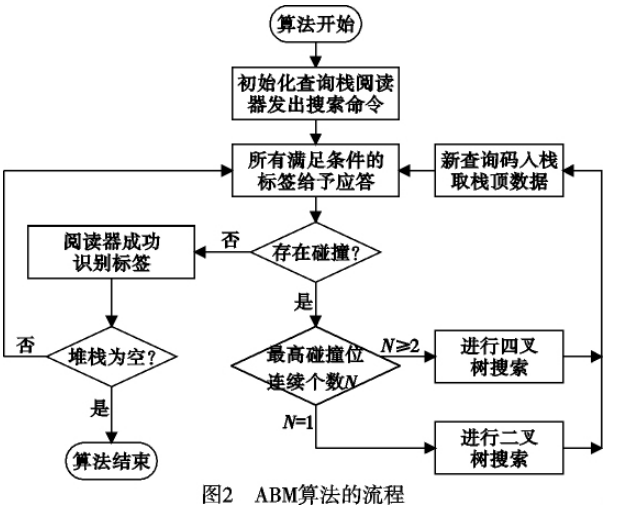

图2　ABM算法的流程

标签识别过程如图 3 所示。识别完所有的标签,共需要 11 个时隙。其中,碰撞时隙 4 个、空闲时隙 1 个、可读时隙 6 个。定义 N 为成功识别出所有标签所花费的总时隙,N_{read} 为可读时隙的个数,那么,在 ABM 算法中,识别效率 $\eta = N_{read}/N = 6/11 \approx 0.5455$; 而对于同样的标签,在 DBS 算法中,识别效率 $\eta = N_{read}/N = 6/13 \approx 0.4615$; 在 DFS 算法中,识别效率仅为 $\eta = N_{read}/N = 6/17 \approx 0.3529$。

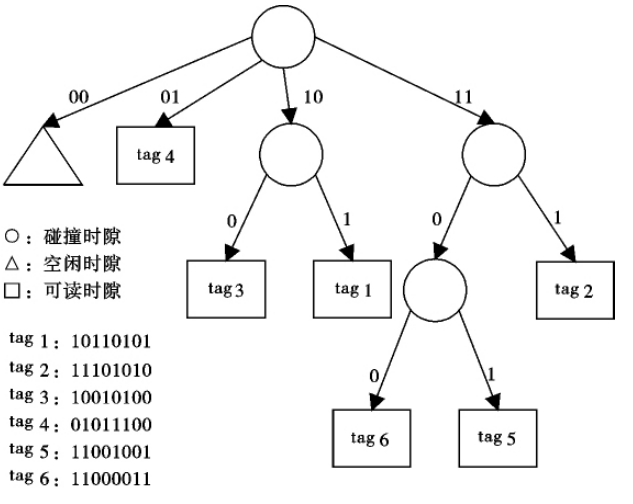

图3　自适应混合树算法的搜索示意图

由上述例子可见,如果防碰撞算法能够自动根据其待识别的分支内的标签碰撞信息自适应地选择搜索树的分叉数,就可以有效提高算法的执行效率。

2.2　算法约定

为了便于描述,引用以下五个指令:

a) 请求命令 request。命令的形式为 request(NULL) 、request(x,m) 和 request(x,y,m) 。

Request(NULL) 的含义为第一次查询时,由于没有碰撞位信息,发送此命令后,阅读器作用范围内的所有待命标签均给予响应,用以检测碰撞位信息。

Request(x,m) 的含义为参数 x 为 1 位的二进制数,m 为检测到的冲突位的最高位,读写器向其作用的区域内的待命标签发送该命令,待命标签将对比自己 EPC 值的第 m 位是否与 x 的数值相同。如果相同,则对阅读器应答,并返回冲突位及相关信息; 否则就将该标签转为休眠态,并将相应的休眠计数器置“1”,对那些已经处于休眠态的标签,令其休眠计数器值加 1。

Request(x,y,m) 的含义为参数 x,y 分别为 1 位的二进制数,m 为检测到的冲突位的最高位,当检测出最高碰撞位和次高碰撞位连续时,调用此命令。读写器向其作用区域内的待命标签发送该命令,待命标签将对比自己 EPC 值的第 m 位是否与 x 的数值相同,以及 $m-1$ 位是否与 y 的数值相同。如果都相同,则对阅读器应答,并返回冲突位及相关信息;否则就将该标签转为休眠态,并将相应的休眠计数器置“1”,对那些已经处于休眠态的标签,令其休眠计数器值加 1。

b) 选择命令 select。命令形式为 select(ID) ,参数 ID 为电子标签的 EPC 编码,该命令通过此 ID 来识别所要选择的标签;即 ID 值与标签的 EPC 编码相同的标签作出响应。

c) 读取命令 read。命令形式为 read。被读取的标签首先要通过 select(ID) 命令选中该标签;在该命令执行时,被选中的标签将自身的信息数据发送给阅读器。

d) 激活命令 active。命令形式为 active(I) 。参数 I 为整数,代表本次操作节点和将要返回的节点之间的数的层次差,该命令激活正处于休眠状态的标签,只有那些正处于休眠态的标签才能响应。该命令执行后,所有休眠态的标签将其休眠计数器的值减去 I,如果操作后的计数器的值变为“0”,标志着该标签已处于待命态,此时可重新响应 request 命令。

e) 灭活命令 unselect。命令形式为 unselect (ID) 。参数 ID 为电子标签的 EPC 编码,该命令的执行是在 read 命令读取数据之后,读写器发送给电子标签,电子标签接收到该命令后立即处于“无声”状态,此状态表明该标签已成功识别。这种状态下,标签对 request 命令不作任何响应。只有在该标签离开读写器的作用范围重新进入后,此标签才能被重新激活。

2.3 算法工作原理

每个标签都有唯一的一个识别码,现假设标签的编码为 8 位,在阅读器作用的范围内有五个标签,各个标签的编码分别为: 10110101,11101010,10010100,01011100, 11000011。其识别过程如图 4 所示。

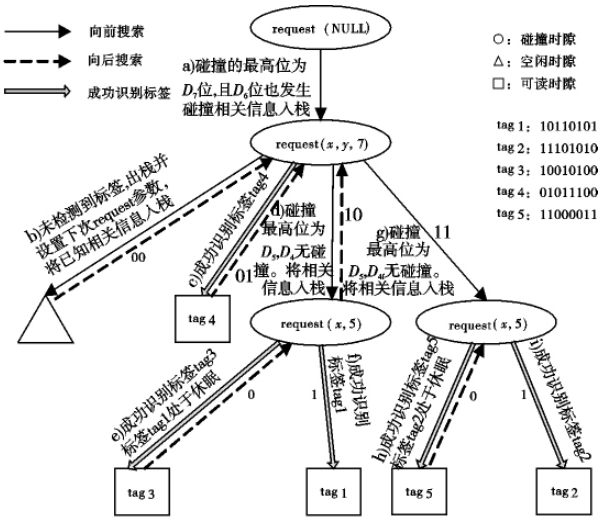

图4 碰撞位匹配的自适应混合树防碰撞算法工作流程

a) 阅读器发送 request(NULL) 命令,阅读器作用范围内的所有待命标签均给予应答,根据曼彻斯特编码原理,解码后得到所有的碰撞位信息,解码后的 EPC 数据为????????,即 $D_7D_6D_5D_4D_3D_2D_1D_0$ 都发生碰撞,碰撞的最高位为 D_7 位,即第 7 位。算法作以下的处理: 检测相邻的第 6 位即 D_6 是否也发生碰撞,本例中,连续两位最高位都发生了碰撞,m 取 7(最高碰撞位的下标) ,这样就获得下一次的请求命令 request(x,y,m) 所需要的数据。同时,设置所有的标签处于激活状态,即将休眠计数器置“0”,这样,标签可以响应 request 命令。用 push 命令将已知标签编码和碰撞位最高位等信息压入堆栈。

b) 阅读器发送 request(0,0,7) 命令,即选择 D_7D_6 为“00”的待命态标签,同时,将 D_7D_6 为“01” “10” “11”的标签转为休眠态,其余的标签将各自的休眠计数器值加 1,这里是所有的标签。而在此分支下,阅读器没有接收到任何标签反馈回来的 EPC 编码信息,这种情况下,意味着此分支下没有任何节点。在识别出该空分支后,由于当前子树的深度为“1”,阅读器发送 active(1) 命令,各休眠标签的休眠计数器值自动减去 1。阅读器调用 pop 命令,从堆栈中读取栈顶信息,设置 $x=0$,$y=1$,$m=7$,即从该节点的父节点获得下一次 request 命令参数,用 push 命令将该分支已知标签编码和碰撞位最高位等信息压入堆栈。

c) 阅读器发送 request(0,1,7) 命令,选择第 D_7D_6 位分别为“01”的待命态标签,将 D_7D_6 位为“10”和“11”的标签转为休眠态,此时只有标签 tag4 满足条件,不存在冲突问题。所以,阅读器就对该标签执行相应的操作(如 select,read 等) ,处理结束后,执行 unselect 命令,对标签进行灭活操作,使之处于“无声”状态。在正确识别一个标签后,阅读器发送激活命令 active(1) 。

d) 阅读器发送 request(1,0,7) 命令,即选择第 D_7D_6 位分别为“01”的待命态标签,此时区域内将会有标签 tag1、tag3 给予应答。从该分支下标签信息碰撞位 $D_5D_4D_3D_2D_1D_0$ 的返回信息中检测出新的碰撞,其最高冲突位为 D_5 位,而 D_4 位没有发生冲突,这样就获得下一次的请求命令 request(x,m) 。与此同时,将 D_7D_6 为“10” “11”的标签的休眠计数器值加“1”。并用 push 命令将该子分支下已知标签编码和碰撞位最高位等信息压入堆栈。

e) 阅读器发送 request(0,5) 命令,第 5 位为“1”的标签转为休眠态,这里是标签 tag1,将其休眠计数器置“1”,也就是标签 tag3 给予应答而标签 tag1 处于休眠,并将处于休眠态的标签 tag2 和 tag5 的休眠计数器值自动加 1,此时 tag2、tag5 的休眠计数器值为 2。由于没有发生任何冲突,标签 tag3 被正确识别。这时阅读器就可对该标签执行相应的操作。操作结束后,执行 unselect 命令,对该标签进行灭活操作,操作完成后执行激活命令 active(1) ,各休眠标签的休眠计数器值自动减去 1。然后,阅读器调用 pop 命令,从堆栈中读取栈顶信息,设置 $x=1$,$m=5$。

f) 阅读器发送 request(1,5) 命令,此分支下只有标签 1 作出应答,不存在冲突问题,因而视为正确识别此标签。根据 request 约定,对处于休眠态的标签 tag2、tag5 的休眠计数器值自动加 1,此刻,tag2、tag5 的休眠计数器值为“2”。在完成对该标签处理后,执行 unselect 命令进行灭活操作。由于本次被操作的节点是子树的最右叶子节点,根据堆栈中信息,可以得知本次操作节点和将要返回的节点之间的数的层次差为“2”,故执行 active(2) 命令,所以,tag2、tag5 的休眠计数器值重新为“0”,即处于待命态。调用 pop 命令,从堆栈中读取栈顶信息,设置 $x=1$,$y=1$,$m=7$。

g) 阅读器发送 request(1,1,7) 命令,即选择第 D_7D_6 位分别为“11”的待命态标签,此时标签 tag2、tag5 均作出响应。从

该分支下的标签信息碰撞位 $D_5D_4D_3D_2D_1D_0$ 的返回信息中检测出新的碰撞,其最高冲突位为 D_5 位,而 D_4 位没有发生冲突,这样就获得下一次的请求命令 request(x,m) ,并用 push 命令将该分支已知标签编码和碰撞位最高位等信息压入堆栈。

h) 阅读器发送 request(0,5) 命令,此时标签 tag5 给予应答而标签 tag2 休眠。由于没有发生冲突,标签 tag5 被正确识别。操作完成后执行激活命令 active(1) ,各休眠标签的休眠计数器值自动减去 1,此时标签 tag2 的休眠计数器值为 "0"。然后,阅读器调用 pop 命令,从堆栈中读取栈顶信息,设置 x = 1, m = 5,即从该节点的父节点获得下一次 request 命令参数。

i) 阅读器发送 request(1,5) 命令,只有标签 tag2 应答,这样,所有的标签已全部识别完成,阅读器在其作用的范围内识别为空,栈也为空,说明算法已成功识别每一个标签。

3　算法仿真及结果分析

为了验证本文中所提出的算法的有效性,下面在计算机上通过大量数据对上述算法的读写过程进行仿真。目前 EPC 编码体系有 EPC、64, EPC-96 和 EPC-256 三种,使用较多的是 EPC-64 编码体系,而新一代 EPC 标签将采用 EPC-96 编码体系[9] 。所以,仿真实验是在 VC + + 平台上,以 96 位 EPC 编码的电子标签为例,对每一组数据取 20 次运行结果的平均值,并分别对 5 种算法作了一个比较,结果如图 5 ~ 7 所示。

3.1　时隙及吞吐率分析

对识别一定数量标签的过程中所需要的总的识别次数和吞吐率作分析对比。总识别次数是衡量算法优越性的一个重要指标,它能直接体现出在一定量的标签识别中所花费时间的多少。实验结果显示,与其他几种算法相比较,本文算法在识别同样多标签的情况下,需要更少的总次数、更短的识别时间。经计算,ABM 算法在待识别标签总数在 200 的情况下,识别总次数比 ACT 和 RDS 少 18% ,比 HQT 少 21.46% ,而比 4-ary QT 算法少了近 37% 。如图 5 所示,ABM 在待识别标签数量越多的情况下,所体现出来的优越性越强。从图 6 可以看出,本文的算法在待识别标签数量增加的情况下,吞吐率始终保持在 0.55 左右,与目前比较好的算法如 ACT、RDS 相比也提高了 10% 左右,比其他算法有了更进一步的提高。

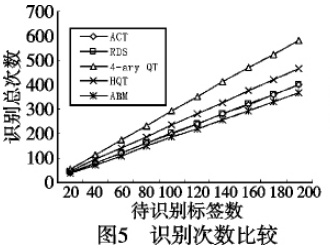

图5　识别次数比较

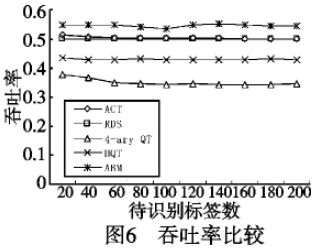

图6　吞吐率比较

3.2　传输数据量分析

传输数据量也是衡量防碰撞算法性能的一个指标,它和标签识别的总次数以及每次传输的有效位数都有关系。标签的识别时间除了与总识别次数有关外,还与识别时隙内的数据通信量有关[10] 。如图 7 所示,由于 4-ary QT 算法采用的是四叉树搜索机制,虽然减少了一定的碰撞次数,但却增加了很多空闲时隙,所以总传输位数最多。HQT 是 Aloha 算法和树形(tree-based) 算法的混合,传输数据量有所降低,由于引入时隙延迟机制,所以标签的识别时间不理想。ACT 和 RDS 算法虽然在总识别次数上基本相同,但是在 request 命令中的参数传递方式不一样,ACT 算法显得更加合理,所以总传输位数相比较而言有了进一步的改善。本文提出的新算法,在减少碰撞发生的同时减少了总的识别次数,降低了总传输数据量。经计算,在标签数为 200 时,ABM 算法识别总传输位数比 ACT 算法少 10% ,比 RDS 少 25.6% ,比 HQT 少 29.3% ,而比 4-ary QT 算法少了近 38.3% 。

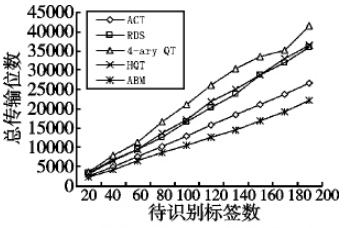

图7　传输数据量比较

4　结束语

本文提出了一种基于碰撞位匹配的自适应混合树防碰撞算法,用于解决 RFID 系统中标签碰撞问题。新算法是在考虑到二叉搜索在标签识别中的频繁碰撞和四叉搜素的空闲时隙多的情况下提出的一种新思路。根据检测标签 EPC 编码最高碰撞位的连续个数匹配信息,动态自适应在二叉树和四叉树中进行叉数选择。而碰撞堆栈的引入,更进一步减少了标签识别过程中的总消耗时隙数和通信数据量并提高了时隙间的吞吐率。通过对算法的分析和仿真实验,ABM 算法与其他常见的防碰撞算法相比,有效地减少了识别总次数和识别时间,大幅提高了搜索效率和吞吐率。

参考文献:

[1] CHAWLA V,HA D S. An overview of passive RFID [J] . IEEE Communications Magazine,2007,45(9) : 11-17.

[2] EPCglobal. EPCglobal tag data standards version 1.4 [S/OL] . (2008-06-11) . http: //www. epcglobalinc. org/standards/tds/tds_1_4-standard-200806-11. pdf.

[3] YEH K H,LO N W,LI Ying-jun,*et al*. An adaptive *n*-resolution anti-collision algorithm for RFID tag identification [C] //Proc of the 24th IEEE International Conference on Advanced Information Networking and Applications Workshops. 2010: 335-338.

[4] FINKENZELLER K. RFID handbook: fundamentals and applications in contactless smart cards and identification [K] . Hoboken: Wiley, 2003.

[5] 陈天娥,程载和. 基于冲突树的 RFID 自适应防碰撞算法 [J] . 计算机应用,2010,30(7) : 1728-1730.

[6] 陈炳才,徐东升,顾国昌,等. 一种基于堆栈存储的 RFID 防冲突算法 [J] . 计算机应用, 2009,29(6) : 1483-1486.

[7] RYU J,LEE H,SEOK Y,*et al*. A hybrid query tree protocol for tag collision arbitration in RFID systems [C] // Proc of IEEE International Conference on Communications. 2007: 5981-5986.

[8] 丁志国,朱学永,郭立,等. 自适应多叉树防碰撞算法研究 [J] . 自动化学报,2010,36(2) : 237-241.

[9] EPCglobal. EPC tag data standards version 1.1 rev. 1.24 [S/OL] . (2004-04-20) . http: //www. epcglobalinc. org/standards_technology/EPCTagDataSpecification11-rev124. pdf.

[10] SEOL J H,KIM S W. Collision-resilient multi-state query tree protocol for fast RFID tag identification [M] . Computational Intelligence and Security. Berlin: Springer-Verlag,2006: 1159-1162.