一种基于混合树防碰撞算法的改进算法

张秀艳 吴 丹* 顾婉莹
(东北石油大学电气信息工程学院 黑龙江 大庆 163318)

摘 要 由于无线射频识别技术 RFID(Radio Frequency Identification)现广泛应用于各个领域中,标签的碰撞问题也成为一个待解决的重要问题。根据现有多叉树防碰撞算法提出一种动态自适应多叉树防碰撞算法(DIHQT)。该算法根据碰撞位最高三位的连续性自行调节算法的搜索叉数,在没有附加查询的条件下,动态自适应地选择二叉树、四叉树或八叉树来查询标签 ID 编码。通过对算法性能分析和仿真实验结果可以表明,DIHQT 算法在时间复杂度上有约 200 次的减少,以及识别效率上较其他算法都有约 5% 提高。

关键词 防碰撞算法 混合树 射频识别技术

中图分类号 TP301 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2017.02.053

AN IMPROVED ANTI-COLLISION ALGORITHM BASED ON HYBRID QUERY TREE

Zhang Xiuyan Wu Dan* Gu Wanying
(*School of Electrical Engineering and Information, Northeast Petroleum University, Daqing 163318, Heilongjiang, China*)

Abstract The Radio Frequency Identification system, which is widely used in various fields nowadays, leads that the tags collision problem an important problem to be solved. A new dynamic adaptive anti-collision algorithm (DIHQT) is proposed based on the existing multi-tree search anti-collision algorithm. According to the features of the highest three collision bit, this algorithm self-adjusts the search tree branches in the absence of additional query, and chooses binary tree, quad tree or octree to query the label codeautomatically. Experimental results of the performance analysis and simulation show that the DIHQT algorithm has 200 times decrease in complexity communication complexity and a 5% increase in recognition efficiency than those of other multi-tree algorithms.

Keywords Anti-collision algorithm Hybrid tree Radio frequency identification

0 引 言

射频识别技术(RFID),作为现今物联网的核心技术之一,有效地利用射频信号及空间耦合的双向传输特性,对静止或动态对象实现非接触式的自动识别[1]。该技术的读写速度快,对于多个移动非可视物体的识别准确性高,标签信息覆盖量大等特点。

射频识别系统共分三个部分:阅读器、应用系统以及电子标签,其中应用系统主要包括中间件和应用系统软件,如图 1 所示[2]。

在一个 RFID 射频识别系统中,当多个标签同时出现在一个阅读器的工作范围内时,两个或两个以上

图 1 无线射频识别系统

标签同一时隙向阅读器发送信息,造成阅读器识别错误,即发生了碰撞。为了提高系统识别效率,减少碰撞的发生,防碰撞算法的研究也就成为无线射频识别领域的关键技术之一。防碰撞算法主要分为两个分支,非确定性 ALOHA 算法以及确定性搜索树算法。ALOHA 算法采用随机的时分多址方法,操作简单且便于实际应用,但标签“饿死”的情况也十分明显。树形搜索算法是由二叉树搜索算法演变而来,逐步缩小搜索范围,直到找到有且仅有的满足指定条件的标

收稿日期:2015 - 10 - 09。**张秀艳**,教授,主研领域:无线通信技术。**吴丹**,硕士生。**顾婉莹**,硕士生。

签。树形搜索算法可以将所以标签无差别识别出来,但它也具有识别时延长以及传输数据量大等缺点。

1 研究现状

现阶段使用频率大且成熟度较高的防碰撞算法主要包括 DFSA 算法和树形搜索算法。DFSA 算法程序简单,操作简便,但存在标签漏读以及吞吐率低等问题。树形算法通过 Manchester 编码方式给每一个标签匹配唯一的 ID 号,解决了标签"饿死"现象,但同时也降低了系统工作效率。

文献[3]分别提到通过动态二叉树算法 DBS、四叉树算法 DFS、查询树算法 QT 来处理标签防碰撞问题。文献[4]中提及在传统二叉树的基础上增加了锁位后退功能的锁位后退防碰撞算法 BLBO,阅读器译码后,根据结果锁定碰撞位来解决标签碰撞的产生。文献[7]中 Shakiba 采用生日悖论的算法,即在 23 人中两人的生日是同日同时的概率大于 50% 这一概念,来计算空闲时隙、碰撞时隙以及成功时隙的发生概率。文献[6]综合了码分多址 CDMA 与 DFSA 算法的特点更大程度上减小了标签发生碰撞的情况。文献[10]通过调整盘存帧长的方法同时跳过空闲时隙以增加系统的防碰撞机能。

本文针对现有 DFSA、优化的树型防碰撞算法中,提出一种基于 RFID 系统的动态自适应混合树算法。该算法有效地增加有效时隙、减少空闲时隙,合理有效利用时隙。

2 动态自适应混合树算法

2.1 曼彻斯特编码

在 QT 算法和 HQT 算法中,主要通过标签的 ID 来确定碰撞发生的位置,从而产生新的查询前缀,在这里我们引入了曼彻斯特编码的方式。由编码中的"0"、"1"来判断脉冲的变化。在编码的译制过程中,如果信号没有发生跳变,则表示发生碰撞。阅读器通过这种"无变化"确定碰撞发生的位置。

假设有两个标签的 ID 是 10110010 与 10101010,根据图 2 可识别出第四和第五位为碰撞位[8]。

2.2 算法描述

算法的执行分为两个部分完成:确定碰撞位以及处理碰撞位。

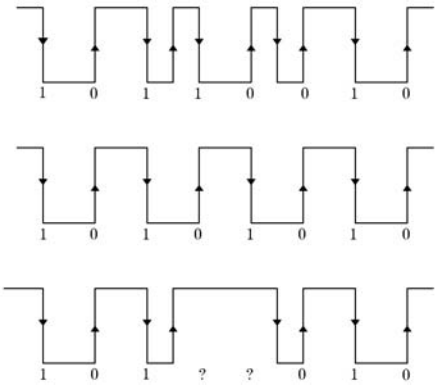

图 2 Manchester 编码原理

查询阶段:阅读器先发声,同一时刻向有效识别范围内的标签发送问询命令(Req),标签接收到命令信号后向阅读器回送自身的 ID 号。

防碰撞阶段:阅读器根据 Manchester 编码原理计算出发生碰撞的准确位置,参照碰撞发生的连续性来自适应决定搜索叉数。

在树形算法中一般会产生三种类型的时隙:碰撞时隙、成功时隙和空闲时隙[9]。文献[9]中提出多叉树搜索需要的总时隙数为:

$$\bar{t}(m) = 1 + B\sum_{L=0}^{\infty} B^{L}[1-(1-B^{-L})^{m}] - mB^{-L}(1-B^{-L})^{m-1} \quad (1)$$

式中 B 代表树结构的分叉数,m 代表标签总量,L 为目前层数。可以看出 $t(m)$ 与 B 的取值成正相关。由式(1)可以看出,为使系统效率最高即总时隙数 $t(m)$ 最小,应采用三叉树查询,而 Manchester 编码由二进制数构成,因此只能选用树形的几种搜索方式。本文提出的算法的主要思想是:在不增加新的查询的基础上,阅读器根据碰撞位的连续性,自适应地调整搜索叉数。当碰撞位为独立一位时,使用二叉树查询,如碰撞比特发生在连续两位时,使用四叉树查询,当碰撞发生在连续三位时,才用八叉树查询。由此可见,产生一个碰撞比特的位置,使用二叉树查询,能减少空闲时隙;在连续两位碰撞位置使用四叉树,能够降低碰撞发生的概率[10];在连续三位碰撞位的发生位置使用八叉树查询,能够减少查询深度,提高系统效率。

2.3 算法流程

加强型自适应混合树算法的主要特征是通过碰撞位的连续性对待识别标签采取成二叉树查询,四叉树查询或八叉树查询。在每一轮询问过程中,阅读器先发送查询前缀,标签进行自查,如果含有发送的前缀,标签对阅读器的查询进行回应。当只有一个标签响应时,识别成功;如有两个或两个以上标签同时响应,产生碰撞,识别失败。根据 Manchester 编码检测对应的碰撞位的连续特征,重新设置查询前缀。对于阅读器

发出的查询 $q_1q_2\cdots q_k$ 和标签对应的 $r_1r_2\cdots r_{n-1}r_nr_{n+1}$，如果最高碰撞位是 r_{n-1} 同时 r_n 并未发生碰撞，则使用二叉树查询；当最高碰撞位 r_{n-1} 与次高位 r_n 都是碰撞位且 r_{n+1} 为未发生碰撞，则采用四叉树查询方法；如 $r_1r_{2\ldots}r_{n-1}r_nr_{n+1}$ 皆为碰撞位，则采用八叉树查询方法。

算法执行中，将前缀保存在堆栈中，在下一次查询时使用。首先将一个空字符串压入栈，之后，阅读器每次查询后将新的前缀压入栈中，直到堆栈内无内容，则标签完成识别。

阅读器操作步骤：

1）空字符串存入初始化后堆栈。

2）检查栈内，为空跳至步骤 7。

3）将查询前缀 q 广播给各个标签。

4）等待标签响应。如果标签没有回应，执行步骤 7，如果标签有回应，判断是否有碰撞发生；没有则执行步骤 6，如果有碰撞发生，判断最高三位碰撞位是否连续不间断：

最高位与次高位碰撞发生不连续：$q0$、$q1$ 压入栈，$PUSH(q0,q1)$；

最高碰撞位位次高位碰撞位连续发生碰撞，且下一位未发生碰撞：将 $q00$、$q01$、$q11$、$q10$ 压入栈内，$PUSH(q00,q01,q11,q10)$；

最高三位为连续碰撞位：将 $q000$、$q001$、$q011$、$q111$、$q010$、$q011$、$q100$、$q111$ 压入栈 $PUSH(q000,q001,q011,q111,q010,q011,q100,q101,q111)$。

5）重复步骤 2 - 步骤 4。

6）识别标签。

7）程序结束。

3　性能分析与仿真结果

3.1　性能分析

3.1.1　时间复杂度

时间复杂度是指在所有标签成功识别过程中所用到的所有时隙的总数，相对应于混合树中所有叶子的节点的总数。在纯二叉树搜索算法中，如果待识别标签的总量为 n，时间复杂度即总时隙数为 $2n-1$；在纯四叉树中时间复杂度在 $[(4n-1)/3,2n-3]$ [9] 内；在八叉树中，时间复杂度 $[(8n-1)/7,8n-7]$ 之间。

证明：由于碰撞位存在三位连续的情况，因此使用八叉树查询优于四叉树和二叉树查询。纯八叉树的内除根节点外所有的节点的分支都是 8，叶子节点的分支为 0，将 n_8 定义分支为 8 的节点，n_0 表示度为 0 的叶子节点，因此八叉树的总节点 $N=n_0+n_8$，此外度为 8 的节点有 8 个分支，度为 0 的节点没有分支，根节点上方没有根，$N=0\times n_0+8\times n_8+1$，则 $8\times n_8+1=n_0+n_8$，因此推出叶子的节点总数为：

$$n_0=7\times n_8+1 \tag{2}$$

当 n 表示标签数量，n_i 表示空闲时隙时：

$$n+n_i=7\times n_8+1 \tag{3}$$

在八叉树搜索算法中，用叶子节点 n_0 表示所有的可读时隙和空闲时隙，在八叉树中产生三位连续碰撞位，对于一个发生碰撞的时隙中，最多的情况会出现 6 个空闲时隙，最少的情况不会产生空闲时隙[6]。

当一个碰撞时隙产生 6 个空闲时隙时，$n_i=6n_8$，代入式(3)可得 $n_8=n-1$，因此，八叉树的最大时间复杂度为：

$$N=n+n_i+n_8=8n-7 \tag{4}$$

当一个碰撞发生的同时并不产生空闲时隙，叶子节点数 n_0 等于标签数量 n，由式(2)得 $n=7\times n_8+1$，$n_8=(n-1)/7$ 所以八叉树的最小时间复杂度：

$$N=n+n_8=(8n-1)/7 \tag{5}$$

因此，八叉树的时间复杂度在 $(8n-1)/7$ 与 $8n-7$ 之间。

为了便于分析，这里对 DIHAT 算法的时间复杂度取均值为 $(32n-25)/7$，再参考二叉树与四叉树的时间复杂度[8]，得出 DIHQT 算法的平均时间复杂度为：

$$\overline{N}=(269N-206)/84 \tag{6}$$

3.1.2　识别效率

识别效率为有效时隙与总时隙的比值，对于 n 个待识别标签，则有效时隙为 n，因此该算法的系统识别效率为[12]：

$$\eta=\frac{n}{\overline{N}}=\frac{84n}{269n-206} \tag{7}$$

3.1.3　通信复杂度

通信复杂度即系统识别过程中传输的总比特数[13]。DIHQT 算法的通信复杂度为阅读器的通信复杂度与标签的通信复杂度之和，即待标签识别过程中总的传输比特数[11]。DIHQT 算法的通信复杂度用 $B(n)$ 来表示，阅读器的通信复杂度用 $B_R(n)$ 表示，标签的通信复杂度用 $B_T(n)$ 表示：

$$B(n)=B_R(n)+B_T(n) \tag{8}$$

在 DIHQT 算法执行过程中，阅读器的查询前缀的长度随着标签相应的位长减少而逐步增加，阅读器第 i 次查询的前缀长度用 $L_1(i)$ 表示，L 为标签长度，用 $L_{2(i)}$ 来表示第 i 次标签应答时的前缀长度[14]。则式(8)也可以表示为：

$$B(n)=\sum_{i=1}^{\overline{N}}(L_1(i)+L_2(i)) \tag{9}$$

由于 $L = L_1(i) + L_2(i)$，根据式(6)，式(9)可以表示为：

$$B(n) = \sum_{i=1}^{\bar{N}} (L) = (269n - 206)/84 \times L \quad (10)$$

3.2 仿真结果

本实验通过软件进行仿真，选择理想信道，随机生成 96 bit 的标签 ID，通信速率为 100 bit/s[15]。在相同实验条件下采用 10 次实验平均值，标签数量从 0 逐步增加到 1000。如图分别对查询树算法 QT、碰撞树算法 CT、与本算法性能进行比较，DIHQT 算法在时间复杂度与三者相比有所减少；通信复杂度由查询前缀长度决定，前缀数越少，阅读器与标签之间的信息交互就越少，传输的数据量也相应减少。如图 3 - 图 5 所示。

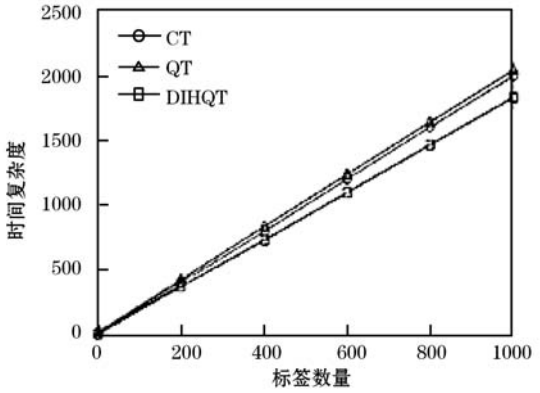

图 3 CT 算法、QT 算法以及 DIHQT 算法时间复杂度的比较

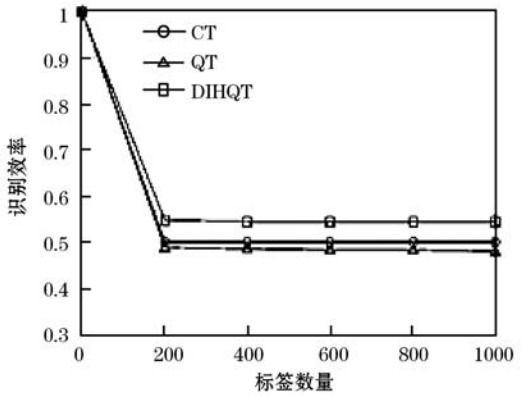

图 4 CT 算法、QT 算法以及 DIHQT 算法识别效率的比较

从图中能看出，当标签数量逐步增加时，DIHQT 算法在时间复杂度上一直优于 QT、CT 算法，当标签数量达到最大时，DIHQT 算法相较于其余两种算法在总的查询次数上减少约 200 次。而 DIHQT 算法在系统效率上有约 5% 的提高。通过在三种算法的通信复杂度、系统效率以及时间复杂度三个方面比较，DIHQT 算法优于其余两种算法。

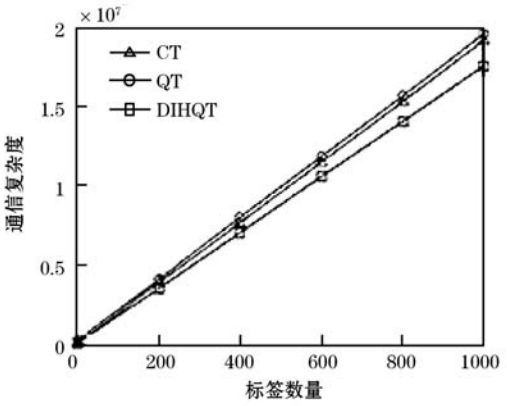

图 5 CT 算法、QT 算法以及 DIHQT 算法通信复杂度的比较

4 结 语

基于树形搜索算法的标签无“饿死”现象的优势，在原有二叉树算法的基础上，通过改进，以及四叉树，八叉树优势的结合，提出了加强型自调整混合树算法。该算法根据碰撞发生的特征自行选择搜索叉数，当最高碰撞的三位为连续不间断时，采用八叉树查询以增加有效时隙；当最高碰撞发生在连续两位时，改为四叉树的搜索方式来减少碰撞的发生；当最高碰撞发生在单独一位时，采用二叉树搜索方式以减少空闲时隙的产生。对算法性能的分析以及仿真实验的结果表明，该算法在通信复杂度、系统效率和时间复杂度上有明显优势。

参 考 文 献

[1] 张学军，蔡文琦，王锁萍. 改进型自适应多叉树防碰撞算法研究[J]. 电子学报，2012，40(1)：193 - 198.

[2] 周艳聪，孙晓晨，顾军华. 一种改进二进制防碰撞算法研究[J]. 计算机应用研究，2012，29(1)：256 - 259，262.

[3] 王春华，刘迟时，徐浩，等. 一种改进的基于二叉树的防碰撞算法[J]. 湖南大学学报(自然科学版)，2013，40(8)：97 - 101.

[4] Sun Y，Hawrylak P J，Mickle M H. Application of ICA in collision resolution for passive RFID communication[C]// Proceedings of the 2009 World Congress on Engineering and Computer Science，2009：1292 - 1297.

[5] Yang X，Wu H，Zeng Y，et al. Capture-aware estimation for the number of RFID tags with lower complexity[J]. IEEE Communications Letters，2013，17(10)：1873 - 1876.

[6] 刘森. 基于 RFID 的物联网感知层查询树防碰撞算法研究[D]. 长春：吉林大学，2013.

(下转第 333 页)

nel rootkits with VMM-based memory shadowing[C]//11th International Symposium on Recent Advances in Intrusion Detection. Springer,2008:1-20.

[5] Hund R, Holz T, Freiling F C. Return-oriented rootkits: bypassing kernel code integrity protection mechanisms[C]// Proceedings of the 18th Conference on USENIX Security Symposium,2009:383-398.

[6] PAX. Homepage of The PaX Team[OL]. http://pax.grsecurity.net.

[7] 王曼丽,翟高寿. 基于编译器插件的轻量级内核重构加固方法研究[J]. 软件,2015,36(3):1-9.

[8] Chen S, Xu J, Sezer E C, et al. Non-control-data attacks are realistic threats[C]//Proceedings of the 14th Conference on USENIX Security Symposium. Berkeley, CA, USA: USENIX Association,2005:12.

[9] Garfinkel T, Rosenblum M. A virtual machine introspection based architecture for intrusion detection[C]//Proceedings of the 2003 Network and Distributed Systems Security Symposium,2003:191-206.

[10] Petroni N L, Fraser T, Molina J, et al. Copilot-a coprocessor-based kernel runtime integrity monitor[C]//Proceedings of the 13th Conference on USENIX Security Symposium, 2004:13.

[11] Chkrootkit: locally checks for signs of a rootkit[OL]. http://www.chkrootkit.org.

[12] RkHunter: protect your machine[OL]. http://www.rootkit.nl/projects/rootkit_ hunter.html.

[13] Petroni N L, Hicks M. Automated detection of persistent kernel control-flow attacks[C]//Proceedings of the 14th ACM Conference on Computer and Communications Security, 2007:103-115.

[14] Petroni N L, Fraser T, Walters A, et al. An architecture for specification-based detection of semantic integrity violations in kernel dynamic data[C]//Proceedings of the 15th Conference on USENIX Security Symposium,2006:289-304.

[15] Hofmann O S, Dunn A, Kim S, et al. Ensuring operating system kernel integrity with OSck[C]//Proceedings of the 16th International Conference on Architectural Support for Programming Languages and Operating Systems, 2011: 279-290.

[16] Riley R. A framework for prototyping and testing data-only rootkit attacks[J]. Computers & Security,2013,37:62-71.

[17] Baliga A, Kamat P, Iftode L. Lurking in the shadows: identifying systemic threats to kernel data[C]//Proceedings of the 2007 IEEE Symposium on Security and Privacy,2007:246-251.

(上接第 298 页)

[7] Shakiba M, Singh M J, Sundararajan E, et al. Extending Birthday Paradox Theory to Estimate the Number of Tags in RFID Systems[J]. PLoS One, 2014, 9(4):e95425.

[8] Yeh M K, Jiang J R. Parallel splitting for RFID tag anti-collision[J]. International Journal of Ad Hoc and Ubiquitous Computing, 2011, 8(4):249-260.

[9] 王雪, 钱志鸿, 胡正超, 等. 基于二叉树的 RFID 防碰撞算法的研究[J]. 通信学报, 2010, 31(6):49-57.

[10] 郭志涛, 程林林, 周艳聪, 等. 动态帧时隙 ALOHA 算法的改进[J]. 计算机应用研究, 2012,29(3):907-909.

[11] Zhang W, Guo Y, Tang X, et al. An efficient adaptivez anti-collision algorithm based on 4-ary pruning query tree[J]. International Journal of Distributed Sensor Networks, 2013, 2013:848746.

[12] 丁治国, 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.

[13] 王必胜, 张其善. 可并行识别的超高频 RFID 系统防碰撞性能研究[J]. 通信学报, 2009, 30(6):108-113.

[14] Shin J, Jeon B, Yang D. Multiple RFID tags identification with Mary query tree scheme[J]. IEEE Communications Letters, 2013,17(3):604-607.

[15] 吴博, 周铜, 王栋. RFID 防碰撞算法分析与研究[J]. 微电子学与计算机, 2009, 26(8):237-239,242.

(上接第 318 页)

[5] Android 系统各版本市场份额进化图[OL]. http://www.199it.com/archives/311622.html.

[6] 巧艳. Android 远程控制恶意软件兴起 可窃取用户信息[OL]. http://www.newhua.com/2013/0718/224074.shtml.

[7] Hennessy S D, Lauer G D, Zunic N, et al. Data-centric security: Intergrating data privacy and data security[J]. IBM Journal of Research and Development,2009,53(2):208-224.

[8] 郑磊,马兆丰,顾明. 基于文件系统过滤驱动的安全增强型加密系统技术研究[J]. 小型微型计算机系统,2007,28(7):1181-1184.

[9] 刘岸,吴琨,仲海骏,等. 基于策略机制的分布式文件保护系统 PFICS[J]. 计算机工程,2004,30(18):119-121.

[10] 廉喆. 手机文档保护系统的设计与实现[D]. 北京邮电大学,2010.

[11] 周巧扣,倪红军. 基于 Android 的文件加密系统的设计与实现[J]. 计算机光盘软件与应用,2013(16):245-246,248.

[12] 朱筱赟,胡爱群,邢月秀,等. 基于 Android 平台的移动办公安全方案综述[J]. 信息网络安全,2015(1):76-83.

[13] CVE 2009-1185[OL]. https://launchpad.net/bugs/cve/2009-1185.

[14] Yu X, Wen Q, Yan T. A novel solution to document protection on mobile platform[M]//Future Wireless Networks and Information Systems. Springer,2012:447-455.