基于标签识别码分组的连续识别防碰撞算法研究

张学军* 王 娟 王锁萍
(南京邮电大学电子科学与工程学院 南京 210003)

摘 要：标签碰撞增加了射频识别(RFID)系统的时间开销和无源标签的能量消耗，降低了识别速率。该文提出了一种适用于标签识别码连续的防碰撞算法——UIG 算法，该算法首先根据公司编码和产品编码将所有标签分组，再由产品序列号的碰撞信息生成每组的两个初始标签识别码。最后，通过对初始标签识别码分别连续减 1 和加 1 识别出所有标签。性能分析和仿真结果显示，该算法在时间复杂度和通信复杂度上都有很大改善，吞吐率得到了大大的提高。

关键词：射频识别技术；防碰撞算法；时间复杂度；通信复杂度
中图分类号：TN92 **文献标识码**：A **文章编号**：1009-5896(2011)05-1159-07
DOI: 10.3724/SP.J.1146.2010.00940

An Uninterrupted Anti-collision Algorithm with ID-based Grouping for RFID System

Zhang Xue-jun Wang Juan Wang Suo-ping
(*School of Electronic Science and Engineering*, *Nanjing University of Posts and Telecommunications*, *Nanjing* 210003, *China*)

Abstract: Tag collision in RFID system increases the time overhead and energy consumption of passive tags, reduces the recognition rate. An Uninterrupted anti-collision algorithm with ID-based Grouping (UIG) is proposed. Firstly, it separates tags into different groups resort to the company code and product code. Then, the algorithm generates two initial tag identification codes of each group by the collision information of products' serial number. Finally, it identifies all tags via add or decrease 1 of the initial tags. Analysis of performance and the results of simulation show that the proposed algorithm improves the time complexity and communication complexity, the throughput is also greatly improved.

Key words: Radio Frequency IDentification (RFID) technique; Anti-collision algorithm; Time complexity; Communication complexity

1 引言

射频识别(Radio Frequency Identification, RFID)是一种非接触式自动识别技术。典型的RFID系统由射频标签和阅读器组成[1]，标签具有唯一的识别码(ID)，被贴附到待识别的物体上。阅读器向标签发送能够提供能量和命令的信号，标签则将其识别码回传给阅读器，阅读器根据接收到的识别码再查阅外部的数据库，以获取目标物体的相关信息[2]。RFID技术具有非接触、存储数据量大等优点，被越来越多地应用于交通控制、资源回收(waste collection)、电子物品防盗门禁(Electronic Article Surveillance gate)、电子监视、供应链管理以及图书馆、超市等允许使用射频无线识别的场合[3]。

在RFID系统中，一个突出的问题是标签碰撞。标签碰撞是指在一个阅读器的识别范围内，有多个标签同时传送它们的识别码信息给阅读器时，信息在共享的无线信道上发生碰撞，使得阅读器无法成功识别任何一个标签的现象[4]。标签碰撞不仅降低了RFID系统的识别速度，而且增加了无源标签的能量消耗。因此，快速简单的防碰撞算法对于无源标签的有效识别非常重要。

现有的防碰撞算法可以分为两类，基于ALOHA的防碰撞算法和基于树的防碰撞算法[5]。基于ALOHA的防碰撞算法[6–9]中，标签随机选择一个时隙传送其识别码，以减少标签碰撞的概率。基于树的防碰撞算法[2–4,10–14]中，当标签发生碰撞时，利用标签识别码或随机二进制数0和1将一个标签集分成两个子集，阅读器继续对子集进行分解，直到识别全部标签。基于ALOHA的算法比较简单，易于实现，且标签数较少时具有很好的性能。但是，由于无源标签能量有限，帧长是受限的[9]，而随着标签数

2010-08-31 收到，2011-02-28 改回
国家自然科学基金(60806027，61001077)，江苏省高校自然科学基金(08KJB510015)，华为高校科技基金(YJCB2008039WL)和南京邮电大学引进人才项目(NY208048)资助课题
*通信作者：张学军 xjzhang@njupt.edu.cn

量的增加，识别所有标签所需的时隙数将呈指数上升[7]，系统的性能将急剧下降，并且该类算法存在一个严重的缺陷——“标签饥饿”[14]，即可能存在一个标签在很长一段时间内都不能被识别。基于树的算法不存在“标签饥饿”的问题，但是对包含所有标签的标签集进行分解的过程会造成很长的识别时延。

现有的大多数算法都是基于标签或 RFID 系统的特点设计的，并没有充分考虑算法应用场合的特点。例如，在产品单一的供应链中，同一个公司、同一类产品的公司编码和产品编码是相同的，产品的序列号也是连续分布的。利用这些信息可以得到效率更高的防碰撞算法，倒置识别码的查询树(Query Tree with Reversed IDs, QTR)防碰撞算法[6]就是充分考虑了识别码的连续性提出的一种新的算法。本文提出了一种适用于产品种类有限、标签识别码连续的防碰撞算法——基于标签识别码分组的连续识别防碰撞算法(Uninterrupted anti-collision algorithm with ID-based Grouping, UIG)。算法的基本思想是，首先根据公司编码和产品编码将所有标签分组，再利用连续序列号的碰撞信息确定每组中首先识别的两个初始标签识别码，然后通过分别减 1 和加 1 连续识别出所有标签。性能分析和仿真结果显示，该算法在时间复杂度和通信复杂度上都有很大改善，吞吐率得到了大大的提高。

本文其它部分的安排如下：第2节简要介绍了基本查询树算法和倒置识别码的查询树算法，第3节详细介绍了UIG算法步骤，第4节从时间复杂度和通信复杂度两个方面对UIG算法的性能进行了分析，第5节通过仿真与其它算法的性能进行了比较，第6节为本文的结束语。

2 相关防碰撞算法介绍

基于树的防碰撞算法可以分为确定性的查询树算法和不确定性的二进制树算法。前者利用标签的识别码对标签集进行分解，后者则利用标签内部的随机数发生器随机产生二进制数0或1分解标签集。查询树算法是一种无记忆算法，对标签计算能力的唯一要求就是将它的识别码与查询命令中的二进制序列进行比较[16]。所以这类算法可以降低标签的成本，更适于大量、广泛的应用。本节主要介绍基本的查询树算法和倒置标签识别码的查询树算法。

2.1 基本查询树算法[2]

在基本查询树 (Query Tree, QT)算法中，阅读器发送由二进制序列组成的查询命令(前缀)，与该前缀匹配的标签响应，并发送自身的识别码到阅读器。该算法的基本识别步骤如下。

(1)阅读器依次发送查询命令集合$Q=\{q_i,\cdots,q_j\}$中的查询命令$q_i=q_1,\cdots,q_x$，并更新集合$Q=Q-\{q_i\}$；

(2)与该查询命令的二进制序列相匹配的标签都响应。若只有一个标签响应，则识别该标签；若有多于一个的标签响应，说明发生了碰撞，则分别在查询序列后面加0和1形成两个新的查询命令加入到集合$Q=Q\cup\{q_1,\cdots,q_x0,q_1,\cdots,q_x1\}$，这样也就把发生碰撞的标签集分成了两个子集；若没有标签响应，则不进行任何操作；

(3)重复步骤(1)和步骤(2)，直到Q为空。

一般情况下，阅读器首先发送值为空的查询命令或者直接发送查询序列0和1。该算法易于实现，但是当标签的识别码有很多位相同时，会有大量的碰撞周期存在。碰撞跟踪树 (Collision Tracking Tree, CTT) 防碰撞算法[17]在这方面做了一些改进，其核心思想是只有在阅读器检测到碰撞时才在碰撞位置处加0和1，更新查询前缀，碰撞位前的部分识别码正常接收。该算法避免了识别码相同部分引起的碰撞，提高了算法的识别效率。

2.2 倒置识别码的查询树(QTR)算法[15]

QTR算法是一种在基本查询树算法基础上考虑了识别码连续性的防碰撞算法。算法将标签的识别码倒置后得到新的识别码，从而将识别码中相同的部分置于后面，而将连续变化的序列号部分放在了新的识别码的前面，然后运用QT算法进行查询。该算法利用连续的序列号即可将所有标签识别，避免了识别码中相同部分引起的碰撞。通过算法分析和仿真可知，该算法与碰撞跟踪树算法具有相近的效率。

QT算法对识别码的每一位都进行查询，每一次查询标签都以完整的识别码响应阅读器，不仅产生了不必要的碰撞周期和空闲周期，还增加了标签的能量消耗。CTT算法是对QT算法的改进，避免了对识别码相同部分的查询，减少了碰撞周期和空闲周期。QTR算法通过倒置识别码达到了与CTT算法相近的识别效率，但这两种算法都没有充分利用二进制序列连续的特性。对于连续的二进制序列，根据碰撞位信息可以直接确定部分的二进制序列，然后再根据这些序列来识别其相邻的序列，减少了碰撞与空闲时隙，UIG算法就是充分利用了识别码的这一特点。

3 UIG算法

标签识别码通常分成几个区。例如，通用标识符 GID-96[18]定义的 96 位的标签识别码包含 4 个字

段：标头(header, 8 位)、通用管理者代码(general manager number, 28 位)、对象分类代码(object class, 24 位)和序列号(serial number, 36 位)。标头定义了编码的长度和结构，对于确定的应用，标头通常是固定的；通用管理者代码标识一个组织实体(一个公司、管理者或其他管理者)；对象分类代码被 EPC 管理实体使用来识别一个物品的种类；序列号在每一个对象分类之内是唯一的。如果一个公司将采用 EPC 编码的标签应用到供应链管理中，这个公司将被赋予一个唯一的公司识别码。这个公司生产的每一种类型的产品都被赋予一个唯一的产品识别码，同一类产品，即具有相同产品识别码的产品将分配不同的序列号[15]。这样，每一个产品拥有一个唯一的识别码，同一类产品的序列号一般是连续分布的。对于序列号连续的同类产品的标签识别码可以根据碰撞的位数来选择多叉树的叉数。

如果连续的序列号发生碰撞，那么碰撞位一定在右边低位部分。理想情况下，如果有两位发生碰撞，可以分成四叉，若三位发生碰撞就可以分成八叉，依此类推，可以高效快捷地识别所有的标签。但实际运用中，标签的起始序列号并不确定，不加考虑地应用这种方法可能反而会产生大量的空闲周期。例如，4 个标签的序列号分别为 0111 1111、1000 0000、1000 0001、1000 0010。在这种情况下，4 个标签同时响应，有 8 位发生碰撞。如果直接采用上述方法，需要对标签进行 2^8 叉树分解，所需周期是标签数的 64 倍，效率反而降低了。

为了防止上述情况的发生，先根据序列号的最高碰撞位对标签进行分解，上例被分解为{0111 1111}，{1000 0000；1000 0001；1000 0010}两个子集。第 1 组可以成功识别，第 2 组进行四叉树分解会产生一个空闲周期。也就是说，即便根据最高碰撞位对标签进行了分解，避免了上述情况的发生，还是会有空闲周期存在，而且随着序列号的增大空闲周期数也会增多。由上面的分组发现，两个子集中序列号连续的标签 0111 1111 和 1000 0000 可以根据碰撞位和连续性确定。对应的两个标签成功识别后，如果分别将识别的标签识别码减 1 和加 1，就可以得到另外两个标签的识别码。由此提出的新算法称为基于标签识别码分组的连续识别防碰撞算法(UIG)，算法流程图如图 1 所示。

UIG 算法是一种适用于标签识别码连续的查询树算法。算法步骤分为两部分，第 1 部分采用 CTT 算法，根据标签的公司和产品编码部分的碰撞信息将标签进行分组，并通过堆栈保存每个分组的两个初始查询序列；第 2 部分，在每个标签分组中，通过将两个初始查询序列分别加 1 和减 1 来识别标签，直到产生一个空闲时隙时结束此查询序列的查询。

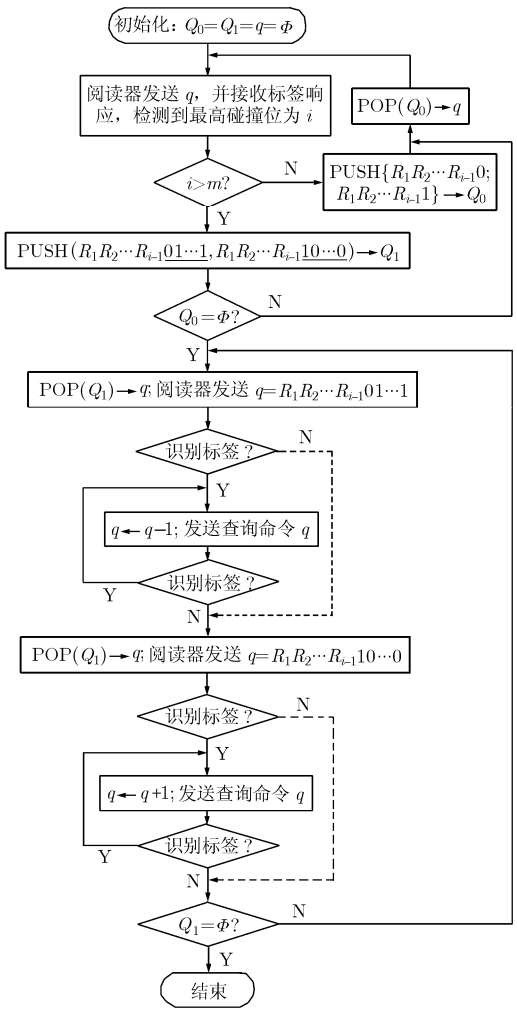

图 1 UIG 算法流程图

假设产品序列号连续的待识别标签数为 N，标签 ID 长度为 n，定义其左边为高位，位序号自左向右为：1,2,3,⋯,n。公司编码和产品编码长度为 m($m<n$)。Q_0,Q_1 为阅读器中的两个先入先出 (First In First Out，FIFO)堆栈存储器，Q_0 存储标签分组的查询序列，Q_1 存储每一个标签分组的初始查询序列，UIG 算法步骤为

第 1 部分　根据公司编码和产品编码部分的碰撞信息对标签进行分组

(1)初始化 $Q_0=Q_1=q=\Phi$；

(2)阅读器发送查询命令 q，标签响应，阅读器接收标签响应，并检测碰撞位。设最高碰撞位为 i，若 $i\leqslant m$，转步骤(3)；若 $i>m$，转步骤(5)；

(3)分别在最高碰撞位 i 之前的 ID 右边添 0 和 1 后，对 Q_0 压栈，即 PUSH($R_1R_2\cdots R_{i-1}0$,$R_1R_2\cdots R_{i-1}1$)→ Q_0；

(4)Q_0出栈到 q，即 $\mathrm{POP}(Q_0)\to q$，转步骤(2)；

(5)分别在最高碰撞位 i 之前的 ID 右边添 01…1(共 $n-i+1$ 位)和 10…0(共 $n-i+1$ 位)后，对 Q_1 压栈，即 $\mathrm{PUSH}(R_1R_2\cdots R_{i-1}01\cdots1, R_1R_2\cdots R_{i-1}10\cdots0)\to Q_1$；若 Q_0 不为空，则转步骤(4)，否则转步骤(6)；

第 2 部分　对每一个分组的标签进行识别

(6)Q_1出栈到 q，即 $\mathrm{POP}(Q_1)\to q$，阅读器发送 $q=R_1R_2\cdots R_{i-1}01\cdots1$。若成功识别一个标签，则转步骤(7)，否则转步骤(8)(注：由于标签产品序列号是连续的，所以本步骤至少有一个标签被识别)；

(7)$q\leftarrow q-1$，阅读器发送 q。若成功识别一个标签，则重复步骤(7)，否则转步骤(8)；

(8)Q_1出栈到 q，即 $\mathrm{POP}(Q_1)\to q$，阅读器发送 $q=R_1R_2\cdots R_{i-1}10\cdots0$。若成功识别一个标签，则转步骤(9)，否则转步骤(10)(注：同步骤(6))；

(9)$q\leftarrow q+1$，阅读器发送 q。若成功识别一个标签，则重复步骤(9)，否则转步骤(10)；

(10)若 Q_1 不为空，则转步骤(6)，否则结束标签识别过程。

在算法第 1 部分，标签以完整的识别码响应阅读器，在第 2 部分中，由于阅读器已经根据碰撞信息确定可能存在的识别码，所以当阅读器发送识别码作为查询命令时，被识别的标签只需返回一位脉冲序列确认即可。

下面举例说明 UIG 算法识别标签的步骤。设有 8 个识别码连续的待识别标签，标签识别码分别为 a：1000 0001, b：1000 0010, c：0110 0110, d：0110 0111, e：0110 1000, f：0110 1001, g：0110 1010, h：0110 1011，设公司编码和产品编码长度为 2，标签序列号长度为 6。阅读器发送查询命令和标签的响应过程如表 1 所示，对应的查询树如图 2 所示。

4　算法性能分析

衡量 RFID 系统防碰撞算法性能主要有两个指标[19]，一是识别一个完整的标签集所需的总时间，另一个是在整个识别过程中标签所消耗的能量。识别所有标签所需的总时间可以用识别过程所需时隙或者查询周期数来表示；标签消耗的能量与识别过程中标签响应的次数，或者标签所发送的数据总量有关。对应于上述两个指标，本节从时间复杂度和通信复杂度[16]两个方面对 UIG 算法的性能进行分析。

4.1 时间复杂度(即识别标签所需的总时隙数)

定理　若 N 个产品序列号连续的待识别标签根据公司和产品编码有 k 个分组，则识别所有标签所需的总时隙数为 $T=N+4k-1$。

表 1 阅读器发送的命令和标签的响应情况列表

	阅读器发送的查询命令 q	响应的标签	碰撞信息	堆栈查询序列状态
第 1 部分	空序列	a, b, c, d, e, f, g, h	i=1<2	Q_0={0,1}; $Q_1=\Phi$
	0	c, d, e, f, g, h	i=5>2	Q_0={1};Q_1={01100111,01101000}
	1	a, b	i=7>2	$Q_0=\Phi$; Q_1={01100111,01101000, 10000001,10000010}
第 2 部分	0110 0111	d	识别	$Q_0=\Phi$;Q_1={01101000, 10000001,10000010}
	0110 0110	c	识别	不变
	0110 0101	没有	空闲	$Q_0=\Phi$;Q_1={10000001,10000010}
	0110 1000	e	识别	不变
	0110 1001	f	识别	不变
	0110 1010	g	识别	不变
	0110 1011	h	识别	不变
	0110 1100	没有	空闲	$Q_0=\Phi$;Q_1={10000010}
	1000 0001	a	识别	不变
	1000 0000	没有	空闲	$Q_0=\Phi$;$Q_1=\Phi$
	1000 0010	b	识别	不变
	1000 0011	没有	空闲	$Q_0=\Phi$;$Q_1=\Phi$，结束识别过程

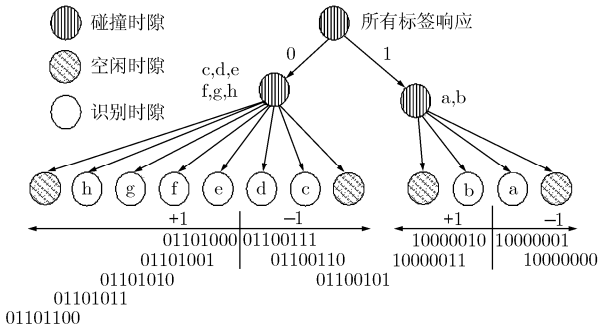

图 2 UIG 算法查询树结构

证明　根据文献[19]分析，QT 算法识别 k 个标签所需的总时隙数为

$$\overline{T}(k)=1+2\sum_{L=0}^{\infty}2^L[1-(1-2^L)^k-k2^L(1-2^L)^{k-1}] \quad (1)$$

碰撞时隙数

$$\overline{C}(k)=\frac{1}{2}(\overline{T}(k)-1) \quad (2)$$

空闲时隙数

$$\overline{Z}(k)=\frac{1}{2}\overline{T}(k)-k+\frac{1}{2} \quad (3)$$

其中 L 为查询树的深度。

CTT 算法是在 QT 算法基础上避免了相同识别码部分产生的碰撞，即 CTT 算法不仅消除了 QT 算法中的多余的碰撞时隙，同时也去掉了与那些碰撞时隙相对应的空闲时隙。所以，CTT 算法识别 k 个标签所需的时隙数为

$$\overline{T}_{\mathrm{CTT}}(k)=\overline{T}(k)-2\overline{Z}(k)=2k-1 \quad (4)$$

在 UIG 算法步骤的第 1 部分，当最高碰撞位小于 m 时，说明碰撞发生在公司编码和产品编码部分，此时只是将标签进行分组。将标签分成 k 组就相当于 CTT 算法识别 k 个标签，所以第 1 部分中将 N 个标签分解为 k 个分组需要的时隙数为

$$T_1=\overline{T}_{\mathrm{CTT}}(k)=2k-1 \quad (5)$$

在 UIG 算法步骤的第 2 部分，每个标签分组中有两个初始查询序列，而每个查询序列都是在产生一个空闲时隙时才结束此查询序列的查询。因此，在第 2 部分识别每一个分组内的标签需要的时隙数等于该分组内的标签数加 2 个空闲时隙，即第 2 部分所需的时隙数为

$$T_2=N+2k \quad (6)$$

综上分析可知，UIG 算法识别 N 个标签需要的总时隙数为

$$T_{\mathrm{UIG}}=T_1+T_2=2k-1+N+2k=N+4k-1 \quad (7)$$

系统吞吐率为

$$S_{\mathrm{UIG}}=\frac{N}{T_{\mathrm{UIG}}}=\frac{N}{N+4k-1} \quad (8)$$

4.2 通信复杂度

通信复杂度包括阅读器通信复杂度和标签通信复杂度。阅读器通信复杂度是指阅读器所发送的比特数，标签通信复杂度是指标签所发送的比特数。标签的通信复杂尤为重要，它决定了标签的功耗[15]。

假设标签识别码的长度为 n，其中 m 位为公司编码以及产品编码，对阅读器和标签的通信复杂度分析如下。

(1)阅读器通信复杂度　在 UIG 算法步骤的第 1 部分，利用 $2k-1$ 个时隙来识别 k 个分组，发送的查询序列的长度为 m，通信量为 $C_{\mathrm{UIG\_R1}}=(2k-1)m$；在 UIG 算法步骤的第 2 部分，利用 $N+2k$ 个时隙来识别所有分组中的 N 个标签，每次发送长度为 n 的查询序列，通信量为 $C_{\mathrm{UIG\_R2}}=(N+2k)n$。所以，阅读器的总通信量为

$$C_{\mathrm{UIG\_R}}=C_{\mathrm{UIG\_R1}}+C_{\mathrm{UIG\_R2}}=(2k-1)m+(N+2k)n \quad (9)$$

(2)标签通信复杂度　当采用 CTT 算法识别 k 个标签时，每一个标签经历的碰撞时隙等于 QT 算法的碰撞时隙数减去消去的碰撞时隙，而 CTT 算法消去的碰撞时隙数等于 QT 算法的空闲时隙数。所以，CTT 算法的碰撞时隙数 $\overline{C}_{\mathrm{CTT}}(k)=\overline{C}(k)-\overline{Z}(k)=k-1$。UIG 算法中，在 UIG 算法步骤的第 1 部分的 $2k-1$ 个时隙中，一个标签平均经历 $k-1$ 次碰撞后被成功分组，且在每个时隙标签都以完整的识别码响应阅读器，此过程的通信量为 $C_{\mathrm{UIG\_T1}}=(k-1+1)n=kn$；在 UIG 算法步骤的第 2 部分的 $N+2k$ 个时隙中，一个标签只响应一次，发送一个单脉冲作为响应，通信量为 $C_{\mathrm{UIG\_T2}}=1$。

所以，UIG 算法中一个标签的总通信量为

$$C_{\mathrm{UIG\_T}}=C_{\mathrm{UIG\_T1}}+C_{\mathrm{UIG\_T2}}=kn+1 \quad (10)$$

算法的性能除了考虑时间复杂度和通信复杂度两个主要指标外，标签的存储和计算能力也是影响标签成本的重要因素。在 UIG 算法中，标签只需存储识别码，不需要额外的数据存储。在标签识别过程中，标签每接收到一次阅读器的查询前缀就会将查询前缀与其识别码进行一次比较运算。在第 1 阶段对标签分组需要 $2k$–1 个时隙，标签需要 $2k$–1 次的比较运算，在第 2 阶段中，标签最多需要 N+$2k$ 次比较运算。可见，UIG 算法中标签进行比较运算的次数最多等于识别所有标签所需的时隙数，而该算法识别同样的标签所需的时隙数少于文中其他防碰撞算法，所以标签进行比较运算次数也少于其他算法。另外，UIG 算法减少了不必要的空闲时隙和碰撞时隙，标签的响应次数也大大减少。在第 1 阶段，一个标签平均响应的次数为 k，在第 2 阶段标

签只需要响应一次就可以被成功识别。所以一个标签被识别平均响应的次数为 k+1。

5 算法仿真及分析

UIG 算法与 QT 算法、CTT 算法和 QTR 算法时间复杂度比较如图 3 所示。仿真时选择了 100 个标签，标签识别码分为 k 组，且每一组的标签码随机连续分布。对 UIG 算法仿真时分组数 k 分别取 1，2，4。由图 3 可见，UIG 算法所需的时隙数随着标签数量的增加线性增加，当标签数量较大，分组较少时，分组对时隙总数的影响很小，算法有较高的效率。当总标签数一定，标签的分组数 k 越大，识别所有标签所需的时隙数也越多。由式(5)和式(7)可知：当 k=N/4 时，UIG 算法和 CTT 算法识别 N 个标签需要相同的时隙数。在这几种算法中，当 k ＜N/4 时，UIG 算法所需的时隙数最少，CTT 算法和 QTR 算法的时隙数相差不大，QT 算法所需时隙数随标签数呈线性增加，与其它算法相比 QT 增长速率最快，所需的总时隙数也最多。

图 4 和图 5 分别将 UIG 算法和 QT 算法、CTT 算法、QTR 算法的阅读器通信复杂度和标签通信复杂度进行了比较(注：CTT 算法采用最优近似)。仿真时根据通用标识符 GID-96 标准，设标签识别码长度 n=96，识别码标头、公司编码和产品编码长度为 m=60，序列号长度为 36。

如图 4 所示，由于 QT 算法中阅读器每发送一次查询前缀，符合条件的标签都以完整的识别码响应，在所有响应中只有一次是有效的，该算法的通信复杂度较高，效率较低。QTR 算法的阅读器通信复杂度最低，因为该算法通过倒置识别码，使得阅读器发送的查询前缀长度减小了，但标签响应的数据长度没有减少。UIG 算法的阅读器通信复杂度并不是最优的，因为该算法中大量数据发送都是由阅读器承担的，但由图 5 可知，UIG 算法的标签通信复杂度是最优的，对于无源、低功耗的标签而言是十分有利的。

6 结束语

本文提出的UIG算法是一种适用于识别码连续场合的查询树防碰撞算法。算法首先对种类有限的产品的标签进行分组，在每一组内部利用连续性及碰撞信息，通过对初始查询序列分别加 1 和减 1 来识别序号相邻的标签。该算法大大提高了阅读器识别标签的速度，当产品种类单一时(即 k=1)算法吞吐率达到最高 $S_{\mathrm{UIG}}=N/(N+3)$。算法对标签通信复杂度的改善尤其明显，而且没有提高对标签的要求，标签只需要具备最基本的存储和计算能力。

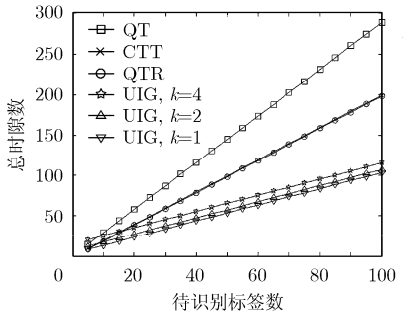

图 3 QT 算法、CTT 算法、QTR 算法和 UIG 算法所需时隙数比较

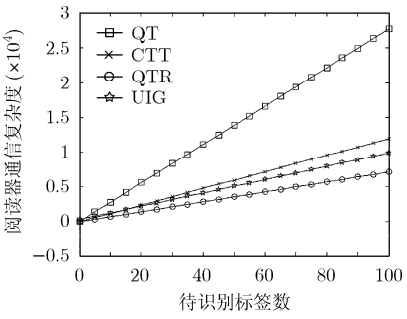

图 4 QT 算法、CTT 算法、QTR 算法和 UIG 算法阅读器通信复杂度比较(CTT 算法采用最优近似)

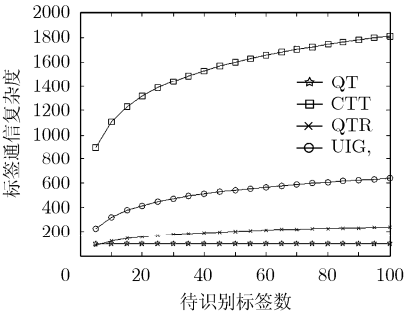

图 5 QT 算法、CTT 算法、QTR 算法和 UIG 算法标签通信复杂度比较(CTT 算法采用最优近似)

参 考 文 献

[1] Ali K, Hassanein H, and Taha A E M. RFID anti-collision protocol for dense passive tag environments [C]. IEEE Conference on Local Computer Networks, Dublin, Ireland, 2007: 819–824.

[2] Myung J, Lee W, and Srivastava J. Tag-splitting: adaptive collision arbitration protocols for RFID tag identification [J]. *IEEE Transactions on Parallel and Distributed Systems*, 2007, 18(6): 763–775.

[3] Kim Y H, Kim S S, and Lee S J. An anti-collision algorithm without idle cycle using 4-ary tree in RFID system[C]. ICUIMC-09, Suwon, S. Korea, 2007: 592–596.

[4] Yeh M K and Jiang J R. A counter-based RFID anti-collision protocol using parallel splitting[EB/OL]. http://www.csie.ncu.edu.tw/~jrjiang/pervasive2009/, 2009.

[5] 丁治国, 郭立, 朱学永等. 基于二叉树分解的自适应防碰撞算法[J]. 电子与信息学报, 2009, 31(6): 1395–1398.
Ding Zhi-guo, Guo Li, and Zhu Xue-yong, *et al.*. An adaptive anti-collision algorithm based on binary-tree disassembly[J].

Journal of Electronics & Information Technology, 2009, 31(6): 1395–1398.

[6] Zhen B, Kobayashi M, and Shimizu M. Framed ALOHA for multiple RFID objects identification[J]. *IEICE Transactions on Communications*, 2005, E88-B(3): 991–999.

[7] Lee S R, Joo S D, and Lee C W. An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification[C]. IEEE MobiQuitous, San Diego, California, 2005: 166–172.

[8] Eom D F and Lee T J. Accurate tag estimation for dynamic framed-slotted ALOHA in RFID systems[J]. *IEEE Communications Letters*, 2010, 14(1): 60–62.

[9] Vogt H. Efficient object identification with passive RFID tags[C]. International Conference on Pervasive Computing, Zurich, 2002: 98–113.

[10] Choi J H, Lee D, and Jeon H. Enhanced binary search with time-divided responses for efficient RFID tag anti-collision[C]. IEEE International Conference on Communications, Glasgow, Scotland, 2007: 3853–3858.

[11] Kim Y H, Kim S S, and Lee S J. Improved 4-ary query tree algorithm for anti-collision in RFID system[C]. International Conference on Advanced Information Networking and Applications, Bradford, United Kingdom, 2009: 699–704.

[12] Chen Y H, Horng S J, and Tun R S. A novel anti-collision algorithm in RFID systems for identifying passive tags[J]. *IEEE Transactions on Industrial Informatics*, 2010, 6(1): 105–121.

[13] Lai Y C and Lin C C. Two blocking algorithms on adaptive binary splitting: single and pair resolutions for RFID tag identification[J]. *IEEE/ACM Transactions on Networking*, 2009, 17(3): 962–975.

[14] Myung J, Lee W, and Shih T K. An adaptive memory-less protocol for RFID tag collision arbitration [J]. *IEEE Transactions on Multimedia*, 2006, 8(3): 1096–1101.

[15] Cho J S, Shin J D, and Kim S K. RFID tag anti-collision protocol: query tree with reversed IDs[C]. International Conference Advanced Communication Technology, Seoul, Korea, 2008: 225–230.

[16] Law C, Lee K, and Siu K Y. Efficient memory-less protocol for tag identification[EB/OL]. http://www.autoidlabs.org/single-view/dir/article/6/91/page.html, 2000: 1–22.

[17] 熊伟, 腾培俊, 梁青. 一种基于冲突跟踪的RFID防冲突算法[J]. 空军工程大学学报, 2009, 10(3): 68–72.
Xiong Wei, Teng Pei-jun, and Liang Qing. Research of an anti-collision algorithm based on collision tracking of RFID system[J]. *Journal of Air Force Engineering University*, 2009, 10(3): 68–72.

[18] EPCglobal tag data standards version 1.3[EB/OL]. http://www.epcglobalinc.org/standards/tds/TDS13-Standard Ratified-20060308.pdf, 2006.

[19] Hush D R and Wood C. Analysis of tree algorithms for RFID arbitration[C]. Proceedings of IEEE Symposium on Information Theory, Cambridge, MA, USA, 1998: 107–116.

张学军：　男，1969 年生，副教授，硕士生导师，研究方向为无线射频识别技术、通信网络的性能分析、流量控制、QoS 理论与技术等.

王　娟：　女，1982 年生，硕士生，研究方向为无线射频识别技术、通信网络的性能分析等.

王锁萍：　男，1946年生，教授，博士生导师，研究方向为通信网络的性能分析、流量控制、QoS理论与技术、单播组播路由算法、通信及网络安全理论与技术和信源编码、信道编码的理论与技术.