doi: 10.7690/bgzdh.2016.07.023

一种适用于超高频 RFID 系统的低复杂度防碰撞算法

李　川
（绵阳职业技术学院信息工程系，四川 绵阳 621000）

摘要：为提高 EPCglobal C1 Gen2 射频识别系统的多标签识别性能，提出一种快速防碰撞算法。该算法仅在一帧中的某个监测点估计标签数量和调整帧长，分析研究了多个监测点，对不同监测点的性能进行比较以寻找最佳检测点，并利用 Matlab2010 软件，采用蒙特卡罗仿真方法对其进行仿真验证。仿真结果表明：在该算法下，系统吞吐率可以达到 0.34，非常接近 EPCglobal C1 Gen2 防碰撞算法的理论最大值 0.368。同现有算法相比，该方法能极大地降低计算复杂度，提高识别性能。

关键词：射频识别；防碰撞；早期调整；系统吞吐率
中图分类号：TP391.4　　**文献标志码**：A

An Anti-collision Algorithm with Low Computation Complexity for UHF RFID System

Li Chuan
(*Department of Information Engineering*, *Mianyang Polytechnic*, *Mianyang* 621000, *China*)

Abstract: A fast anti-collision algorithm has been proposed to improve the identification performance of the EPC C1 Gen2 standard. The algorithm estimates the tag quantity and adjusts the frame size at a check point of a frame for each identification round. The Monte Carlo simulation is carried out to verify the performance of the proposed algorithm by using MATLAB. Simulation results show that the system throughput of our method can reach a value of up to 34%, which is very close to the theoretical maximum of 36.1% for the EPC C1 Gen2 anti-collision scheme. As compared to existing methods, the algorithm significantly reduce the complexity and improve the performance.

Keywords: RFID; anti-collision; early adjustment; system throughput

0　引言

近年来，超高频(ultra high frequency，UHF)射频识别(radio-frequency identification，RFID)技术以其读写距离长、标签成本低等特点，被广泛应用于供应链管理、智能交通和食品溯源等领域。RFID 系统的一个主要挑战就是在有限的时间内识别大量目标物体。在实际应用中，每秒读取超过 300 个物体是目前工业界的普遍需求[1]；因此，防碰撞算法的效率在识别多个标签上扮演着重要角色。

为了处理多路访问问题，超高频 RFID 标准 EPCglobal C1 Gen2[2]规定了一种被称为 Q-算法[3]的防碰撞算法。该算法基于时分的思想并允许读写器在任意时隙通过改变参数 Q 来调整帧长，其中帧长等于 2^Q。在 Q-算法中，读写器通过浮点变量 Q_{fp} 和调整步长 w 来控制 Q 值的变化，从而实现帧内调整。

Q-算法的优势在于便于实现。然而 Q-算法只给出了 0.1～0.5 的推荐范围，并未详细规定 w 值的调整策略。作为 Q-算法的演进，动态帧时隙 Aloha 算法(dynamic framed slotted Aloha，DFSA)[4-6]被广泛用于解决多标签碰撞问题。为了提高未读标签数估计的准确性，这些算法需要较高的浮点运算成本(floating pointing operation，FLPO)或计算复杂度。尽管准确的估计方法可以提高防碰撞算法的性能，但是所需的计算成本和计算复杂度也随之增加，不能应用于计算能力受限的读写器上。

文献[7]提出了一种便于实现的防碰撞算法(feasible easy-to-implement anti-collision，FEIA)。作者在该文献中指出，未读标签数估计误差并不是导致识别性能恶化的主要因素，18%的估计误差仅造成了 0.7%的性能恶化。该方法提出了一种有效的帧长调整原则并可以实现 0.33 左右的吞吐率；然而，对于未读标签数的估计和帧长的调节，需要在每个时隙内执行，这无疑会导致计算负载的增加。大多数 RFID 读写器都只装配单核处理器，例如 8 bit 的 8051 单片机，它们的计算能力都非常有限；因此，笔者提出一种基于快速帧内调整(fast in-frame

收稿日期：2016-04-02；修回日期：2016-05-13
基金项目：四川省教育厅自然科学重点项目(15ZA0369)
作者简介：李　川(1972—)，男，四川人，硕士，副教授，从事信号处理与传输研究。

adjustment，FIFA)的防碰撞算法。

1　FIFA 算法

基于帧内早期调整的防碰撞算法主要基于 DFSA。此类算法允许读写器在一帧内的任意时隙终止当前识别过程，并调整一个新的帧长来进入一轮新的识别；因此，读写器必须判断当前帧是否合适，从而进一步判断该帧是否需要进行调整。

对于读写器而言，一个重要参数就是需要识别的标签数。通常来说，标签数量对于读写器来说是未知的；因此，需要估计标签数量。在识别过程中，当读写器读到第 i 个时隙时，对于整个标签数的估计可以通过下式[6]得到：

$$n_{\text{est}} = \left(S_i + 2.39 \times C_i\right) \times \frac{F}{i}。\tag{1}$$

其中：S_i 为前 i 个时隙，成功读取的标签数；C_i 为前 i 个时隙统计出的碰撞时隙数；F 为当前帧长。当估计完标签数后，读写器应该判断当前帧长是否合适。假设总的标签数为 n，每个时隙内的标签数分布服从二项分布，系统吞吐率可以计算为

$$U = \frac{n}{F}\left(1 - \frac{1}{F}\right)^{n-1}。\tag{2}$$

对于一个给定的标签数，可以通过式 (2) 计算出相应的最佳帧长。针对不同的标签数范围，表 1 给出了其最佳的帧长设置。值得注意的是，距离 n_{est} 最近的帧长 F 不一定是最佳帧长。例如，对于 n=90，其最近的帧长为 64，然而它的最佳帧长为 128。

表 1　不同标签数范围的最佳帧长设置

Q	帧长(F=2^Q)	标签范围 n_1～n_2	Q	帧长 (F=2^Q)	标签范围 n_1～n_2
1	2	1～3	6	64	45～89
2	4	4～6	7	128	90～177
3	8	6～11	8	256	178～355
4	16	12～22	9	512	356～710
5	32	23～44	10	1 024	711～1 420

读写器通过式 (1) 估计出的标签数量如果不在当前帧的最佳范围内，就说明当前的帧不合适，需要重新调整帧长。读写器调整帧长的策略也是基于剩余标签数

$$n_{\text{backlog}} = n_{\text{est}} - S_i。\tag{3}$$

有了剩余标签数，读写器就可以根据表 1 来调整新的帧长。例如，当前剩余标签数为 87，那么新的帧长就应该为 64。在 FEIA 算法中，读写器需要在每个时隙都判断当前帧是否需要调整。在 FIFA 中，读写器在每一帧中仅判断一次。文中提出的 FIFA 算法的最大优势在于，它可以极大地降低计算复杂度。在 FIFA 算法中，有个重要参数 i，该参数为帧内调整的检测点，即在一帧中的第 i 个时隙进行帧长检测。直观地，在识别过程中一个完整帧 F 可以被 3 个检测点(0.25F、0.5F 和 0.75F)分为 4 个部分。比较这 3 个检测点的吞吐率性能，并找出性能最佳的检测点。

归一化的系统吞吐率可以表示为

$$\eta = \frac{E_{\text{S}}}{E_{\text{S}} + E_{\text{C}} + E_{\text{E}}}。\tag{4}$$

其中 E_{S}, E_{C}, E_{E} 分别表示平均的成功时隙数、碰撞时隙数和空闲时隙数。值得注意的是，DFSA 算法的理论最大吞吐率为 0.368；但是在 EPCglobal C1 Gen2 标准中，帧长只能为 2 的整数次幂，因此最大吞吐率变成 0.361[7]。

FIFA 算法的流程可以归纳如下：

```
1. 初始化 F=F_ini
2.   S=0, E=0, C=0;
3.   while (未读标数不等于 0)
4.      slot_index=1;
5.      while slot_index<=F
6.        接收标签响应
7.        if 响应数为 1
8.        S++; slot_index++   //成功时隙
9.        else if 响应数为 0
10.       E++; slot_index++   //空闲时隙
11.       else
12.       C++; slot_index++ //碰撞时隙
13.       end
14.       if slot_index= =i
15.         根据式(1)和(3)来估计剩余标签数
16.         if 剩余标签数在当前 F 的最佳范围内
17.               继续本轮识别过程
18.         else
19.                通过表 1 更新帧长
20.                slot_index=L;
21.         end
22.       else
23.                slot_index++;
24.       end
25.   end
26. end
```

2　仿真结果

笔者利用 Matlab2010 软件，对文中提出的 FIFA 算法采用蒙特卡罗仿真方法进行仿真验证。为了确保实验结果的收敛性，笔者对每个实验进行了 5 000 次仿真，首先比较了不同检测点 i 下的吞吐率，初

始帧长设为 64。从图 1 可以观察到，当标签数 n 大于 200，i 等于 0.25F 时，可以实现更高的吞吐率。作为比较，当标签数 n 小于 200 时，i=0.25F 的性能低于 0.5F 和 0.75F，但是差距也不明显，图中同样显示了 FEIA 算法的性能。平均来看，0.25F 的平均吞吐率可以达到 0.336 9，0.5F、0.75F、FEIA 和 Q-算法分别可以达到 0.328 2、0.321 2、0.331 7 和 0.298 0。结果显示，早期调整策略可以提高系统吞吐率，特别是标签数量超过 200 的时候。

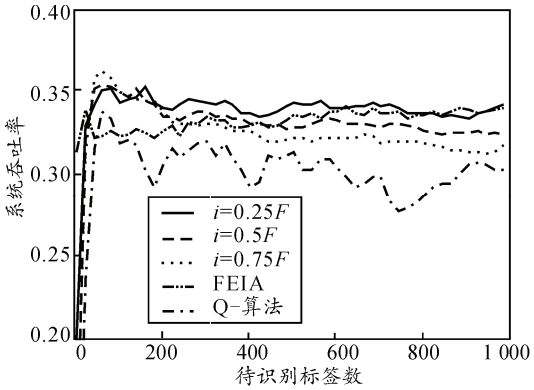


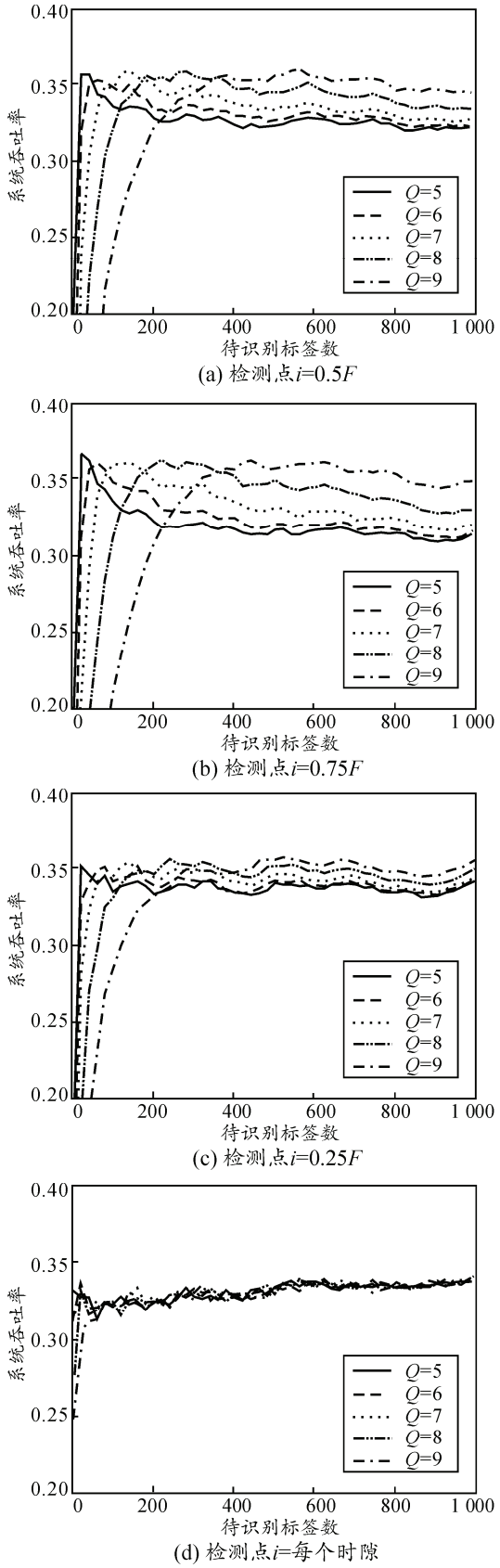
图 1 不同检测点的吞吐率性能比较

如上文所述，笔者提出的 FIFA 算法可以降低系统的计算复杂度。这一优势主要来源于每轮识别过程检测次数的降低。表 2 列出了标签数从 5～1 000 之间变化，不同检测点所需的检测次数。FEIA 算法在每次时隙都执行一次检测，所需的检测次数大约为 FIFA 算法的 26 倍。每次检测主要包括未读标签数估计和帧长调整；所以，FIFA 算法的计算复杂度明显低于 FEIA 算法。

表 2 不同检测点所需的检测次数对比

性能指标	吞吐率	检测次数	性能指标	吞吐率	检测次数
0.25F	0.336 9	20	FEIA（每个时隙都检测）	0.331 7	511
0.5F	0.328 2	18	Q-algorithm	0.298 0	8
0.75F	0.321 2	16			

此外，DFSA 算法的性能会受到初始帧长的影响。例如，当初始帧长远小于未读标签数时，会产生大量碰撞时隙，从而导致性能降低；因此，需要衡量初始帧长对吞吐率性能的影响。图 2 为提出的算法在不同初始帧长下的性能，如图(a)和(b)所示，当 i=0.5F 和 i=0.75F 时，吞吐率性能受初始帧长影响较大。作为对比，当标签数大于 200 时，i=0.25F 的吞吐率性能几乎独立于初始帧长。尽管 FIFA 算法的稳定性要次于 FEIA 算法，但是 FIFA 算法的平均性能和计算复杂度都优于 FEIA 算法。通过图 1 和图 2 的比较可知：文中提出的 FIFA 算法在吞吐率、计算复杂度和稳定性方面，都具有良好性能。



(a) 检测点i=0.5F

(b) 检测点i=0.75F

(c) 检测点i=0.25F

(d) 检测点i=每个时隙

图 2 初始帧长对性能的影响

（下转第 92 页）