Analysis of Tree Algorithms for RFID Arbitration

Don R. Hush* and Cliff Wood**
*EECE Dept, Univ.of New Mexico, Albuquerque, NM 87131, hush@eece.unm.edu
**Micron Communications, 3176 Denver Way, Boise, ID 83707

Abstract — **The *Tree Search* (or *splitting*) method introduced by Capetanakis for use in conventional multiaccess systems [1] can also be applied to RFID arbitration. This paper performs a transient analysis of this, and related methods.**

I. Introduction

Radio Frequency Identification (RFID) systems have emerged as a viable and affordable alternative for tagging and/or labeling small to large quantities of items. The arbitration problem is that of identifying all 'tags' present in the current environment (i.e reading the labels). This is a multiaccess communication problem in which contention for the communication medium is transient in nature. Important measures of performance include the time required to identify the tags, and the power consumed by the tags. The first of these is proportional to the number of time slots needed to complete the arbitration process, and the second is proportional to the total number of times the tags are required to transmit during this process. We analyze a family of *Tree Search* algorithms with respect to these two measures. Previous analysis on the average number of time slots [1–3] reveals a linear dependence on the number of tags, m. We extend this work by developing expressions for the decomposition of these slots into three basic types: zero, one and multiple reply slots. Original expressions are also developed for the total number of tag replies.

II. Results

More specifically, this paper develops expressions for the following quantities:

$\bar{t}_{TS}(m)$ = Average (total) number of Time Slots
$\bar{c}_{TS}(m)$ = Average number of Collisions
$\bar{z}_{TS}(m)$ = Average number of Zero replies
$\bar{r}_{TS}(m)$ = Average Total number of tag replies

The first of these can be decomposed into the second two as follows,

$$\bar{t}_{TS}(m) = \bar{c}_{TS}(m) + \bar{z}_{TS}(m) + m \qquad (1)$$

where the '$+m$' accounts for the m time slots corresponding to Single replies.

Although the traditional *splitting method* was developed using a binary tree (i.e. $B = 2$) [1], it can in principle carry out a $B-ary$ tree search for any integer $B > 1$. The results for the first few values of B are shown in the two tables below.

B	$\bar{t}_{TS}(m)$	$\bar{c}_{TS}(m)$	$\bar{z}_{TS}(m)$
2	$2.885m$	$1.443m$	$0.442m$
3	$2.730m$	$0.910m$	$0.820m$
4	$2.882m$	$0.720m$	$1.162m$
5	$3.113m$	$0.622m$	$1.491m$

B	$\bar{r}_{TS}(m)$
2	$m\log_2 m + 2.333m = 1.44m\ln m + 2.333m$
3	$m\log_3 m + 2.030m = 0.91m\ln m + 2.030m$
4	$m\log_4 m + 1.921m = 0.72m\ln m + 1.921m$
5	$m\log_5 m + 1.867m = 0.62m\ln m + 1.867m$

If the number of tags, m, is known then a more efficient tree search can be employed. Conceptually it works by dividing the search space into m equal intervals, and then employing a $B-ary$ tree search over each of these intervals. The results for this approach are shown in the next two tables.

B	$\bar{t}_{OP}(m)$	$\bar{c}_{OP}(m)$	$\bar{z}_{OP}(m)$
2	$2.337m$	$0.669m$	$0.669m$
3	$2.431m$	$0.477m$	$0.954m$
4	$2.642m$	$0.411m$	$1.231m$
5	$2.881m$	$0.376m$	$1.505m$

B	$\bar{r}_{OP}(m)$
2	$2.49m$
3	$2.08m$
4	$1.93m$
5	$1.86m$

Not only does this approach reduce the average number of time slots, it also reduces the total number of tag replies from $\Theta(m\log m)$ to $\Theta(m)$.

Acknowledgements

We gratefully acknowledge the support of Micron Communications, Inc. in Boise, Idaho.

References

[1] J.I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," *IEEE Transactions on Information Theory*, IT–25, No. 5, pp. 505–515, 1979.

[2] J.L. Massey, "Collision resolution algorithms and random–access communications," *Multi–User Communication Systems*, Ed. G. Longa, Springer–Verlag, New York, pp. 73–137, 1981.

[3] M.A. Kaplan and E. Gulko, "Analytic properties of multiple–access trees," *IEEE Transactions on Information Theory*, IT–31, No. 2, pp. 255–263, 1985.