An Efficient Collision Resolution Scheme for Wireless Multiple Access *

Po-Ning Chen❖, Chiung-Shien Wu† and Gin-Kou Ma†

❖Department of Communication Engineering,
National Chiao Tung University, Taiwan
E-mail: poning@cc.nctu.edu.tw

†Computer and Communication Research Labs.
ITRI, Hsinchu, Taiwan 310, R.O.C.
E-mail: cwu@e0sun3.ccl.itri.org.tw

Abstract - *This paper presents a contention resolution scheme for multiple random access based on tree algorithms. The proposed scheme, called Dynamic Collision Resolution (DCR), is a variation of the tree algorithm. Given that plural users are transmitting packets to a shared communication medium, the tree algorithm will divide the transmitting users into n parts upon detecting a collision condition and, in the later retransmission, collisions will only occur in between the users that fall into the same part. It was well known that the optimal value of n is 3 under the condition that n is fixed and the number of users is infinite. The proposed DCR scheme adopts a dynamically change on the value of n for a better contention resolution. The performance is evaluated through the simulation over a finite number of users and the result shows that the DCR scheme gives a non-trivial improvement on the tree algorithms under a moderate number of users.*

1. Introduction

The contention resolution technique has been considered as an essential part for distributed control of multiple access to a shared channel in a communication system. Plural transmitters attached to the shared channel are time-synchronized and start their transmission of packets only at the beginning of a slot. When more than one transmitters send their packets to the same slot, a collision occurs and this status indicating COLLISION will be feedback to all the transmitters for starting later retransmissions based on a random waiting time. If, fortunately, only a transmitter has sent its packet on a slot, then a status indicating NO-COLLISION will also feedback to inform a successful transmission.

Channel access algorithms using to the above mechanism are called *Random-Access Algorithms (RAA)* and the slotted-ALOHA is the most original work [1]. However, the maximum throughput is only 0.36. Another algorithm called, *Collision Resolution Algorithm (CRA),* proposed by Capetanakis [2] and Tsybakov, Mikhailov [3] has adopted the tree resolution concept and the maximum throughput is improved up to 0.407 [4]. A further improvement to 0.4878 which uses ternary feedback (i.e., IDLE, COLLISION, NO-COLLISION) and bias splitting was made by Mosely [5]. All the above works were analyzed by assuming infinite number of transmitters and the upper bound of the maximum throughput has been shown to be no greater than 0.587 [6].

A great improvement on RAA can also be achieved by using carrier sensing. This type of improvement is even more significant than the tree algorithms and thus many realistic networks such as Ethernet has adopted this method [7]. However, the carrier sensing function is sometimes hard or impractical to be implemented in communication medium such as wireless links, broadband coaxial cable, ... etc. And also because that the channel is usually shared by a small number of users instead of infinite number of users, we propose to improve the tree algorithm under the condition that no carrier sensing function will be used and the number of users is not big. We also make a simple requirement that the channel status is binary feedback (COLLISION, and NO-COLLISION) instead of ternary feedback (IDLE, COLLISION, and NO-COLLISION).

2. Tree Algorithm

According to [4], there are two types of tree algorithms, namely the Free-Access Algorithm and the Blocked-Access Algorithm, described as follows.

1. **Free-Access Algorithm.** Newly arrived packets are transmitted immediately at the beginning of the next slot following their arrival.
2. **Blocked-Access Algorithm.** Newly arrived packets are transmitted in the first slot after all previous conflicts are resolved, i.e., new packets are blocked at their transmitters until all the other transmitters with collided packets have successfully retransmitted these packets.

The basic tree algorithm is described as follows.

3. **Basic *n-ary* Tree Algorithm.** After a collision, each transmitter that has sent a packet would flip an *"n-sided coin"* with value *1, 2, 3, ..., n*. This will split the contending transmitters into *n* subsets. For those transmitters in the subset with value *i* ($1 \leqq i \leqq n$), we assign each of these transmitters an index *i*. Then, in the very next slot, transmitters with index 1 are permitted to send. If the outcome is NO-COLLISION (only one transmitter is with

* † This work is currently supported by the *R.O.C. Ministry of Economic Affairs* under the project *No. 3P12200 conducted by ITRI.*

❖ * This paper is a partial result supported by *National Science Council* under project No. *NSC 87-2213-E-009-139-.*

0-7803-4320-4/98/$5.00 © 1998 IEEE

index 1), then each of the other transmitters with index greater than 1 will decrease the index by 1. If the outcome is COLLISION, then each of the other transmitters with index greater than 1 will increase the index by $n-1$ and the colliding transmitters with index 1 will once again flip the *"n-sided coin"* to resolve the contention.

The basic n-ary tree algorithm can be combined with the Free-Access Algorithm or the Blocked-Access Algorithm to form different protocol variations. Thus, the combination of 1. and 3. will produce the Free-Access Tree algorithm (FAT). The combination of 2. and 3. will produce the Blocked-Access Tree algorithm (BAT). Note that the best throughput is 0.403 for FAT and 0.368 for BAT when $n = 3$ and the number of transmitters is infinite.

3. The DCR Scheme

Given that a plurality of users are transmitting packets to a shared communication medium, the tree algorithm will divide the transmitting users into n parts upon detecting a collision condition and, in the later retransmission, collisions will only occur in between the users that fall into the same part. In FAT and BAT algorithms, the number of groups that the system is split into is fixed at each detected collision, for example, $n = 2$, or $n = 3$. However, we note that the number of collided transmitters in each slot may be dynamically changing. It is possibly helpful to let the value of n change at different point of collisions. For example, we let $n = 3$ in the first collision and $n = 2$ in the next collision. We refer to this scheme as the *Dynamic Collision Resolution (DCR)* or *Hybrid-n* in that the value of n is not fixed. Here a *Transmission Splitting Vector,* $TSV = \{n_1, n_2, n_3, \ldots, n_k\}$, is used for the purpose of recording the changing sequence of the different value of n. For simplicity, we also refer to the original FAT and BAT algorithms with fixed value of n as the *Fixed-n* scheme. The parameter k in the *Hybrid-n* scheme is a positive integer. When $k = 1$, the scheme works as the *Fixed-n* scheme does. When $k > 1$, all the collided transmitters will be split into n_1 subgroups at the first collision, n_2 subgroups at the second collision, and n_i subgroups at the i^{th} collision. For the $(k+1)^{th}$ collision, the system will rotate back to the first value of n, i.e., n_1.

4. Simulations Results

We observe the performance of the proposed scheme by simulation. We assume that the system has finite number of transmitters and the packet arrival rate on each transmitter is assumed as the Poisson arrival process.

In the first simulation, we let $TSV = \{3, 3, 3, 2\}$ for 10 transmitters. As shown in Figure 1, the performance for BAT is improved a little by using the *Hybrid-n* scheme. However, the FAT scheme is not improved by *Hybrid-n*.

In the second simulation, we let $TSV = \{2, 4, 8, 2, 2, 2, 2, 2\}$ for 10 transmitters. In Figure 2, the performance for BAT is improved significantly by the *Hybrid-n* scheme. And the FAT scheme is also improved by *Hybrid-n*, although just a little.

In the third simulation, we let $TSV = \{2, 4, 8, 16, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2\}$ for 10 transmitters. In Figure 3, the performance for both BAT and FAT is improved non-trivially by the *Hybrid-n* scheme.

We also observe that the performance of the Hybrid-n scheme varies with the number of transmitters. In Figures 4 and 5, we set the number of transmitters to 50 and 5 respectively. The result shows that under a small number of transmitters, the *Hybrid-n* scheme will work better than the *Fixed-n* scheme. In Figure 6, the average throughput for BAT using *Hybrid-n* under different number of transmitters is shown. The result also conforms to that observed in the previous figures.

References

[1] N. Abramson, "The ALOHA system – Another Alternative for Computer Communications", in Proc. *AFIPS Conf. Fall Joint Comp. Conf.,* 1970, vol .37, pp. 281-285.

[2] J. Capetanakis, "The Multiple Access Broadcast Channel: Protocol and Capacity Considerations", *Ph.D. dissertation, Massachusetts Institute of Technology., Cambridge,* 1977.

[3] B.S. Tsybakov, and V.A. Mikhailov, "Free Synchronous Packet Access in a Broadcast Channel with Feedback", pp. 259-280, *in Plenum Press translation of article in Problemy Peredachi Informatsii,* vol. 14, no. 4, pp. 32-59, Oct.-Dec. 1978.

[4] P. Mathys, and P. Flajolet, "Q-ary Collision Resolution Algorithms in Random-Access Systems with Free or Blocked Channel Access", IEEE Transactions on Information Theory, vol. IT-31, no. 2, pp. 218-243., March 1985.

[5] J. Mosely, and P.A. Humblet, "A Class of Efficient Collision Resolution Algorithms for Multiple Access Channels", IEEE Transactions on Communications, vol. COM-33, no. 2, pp. 145-151, Feb. 1985.

[6] V.A. Mikhailove, and B.S. Tsybakov, "Upper Bound for the Capacity of a Random Multiple Access System", *in Problemy Peredachi Informatsii (USSR),* 17:90-95, 1981.

[7] D. Bertsekas, and R. Gallager, "Data Networks", second edition, *Prentice Hall, Inc.,* 1992.

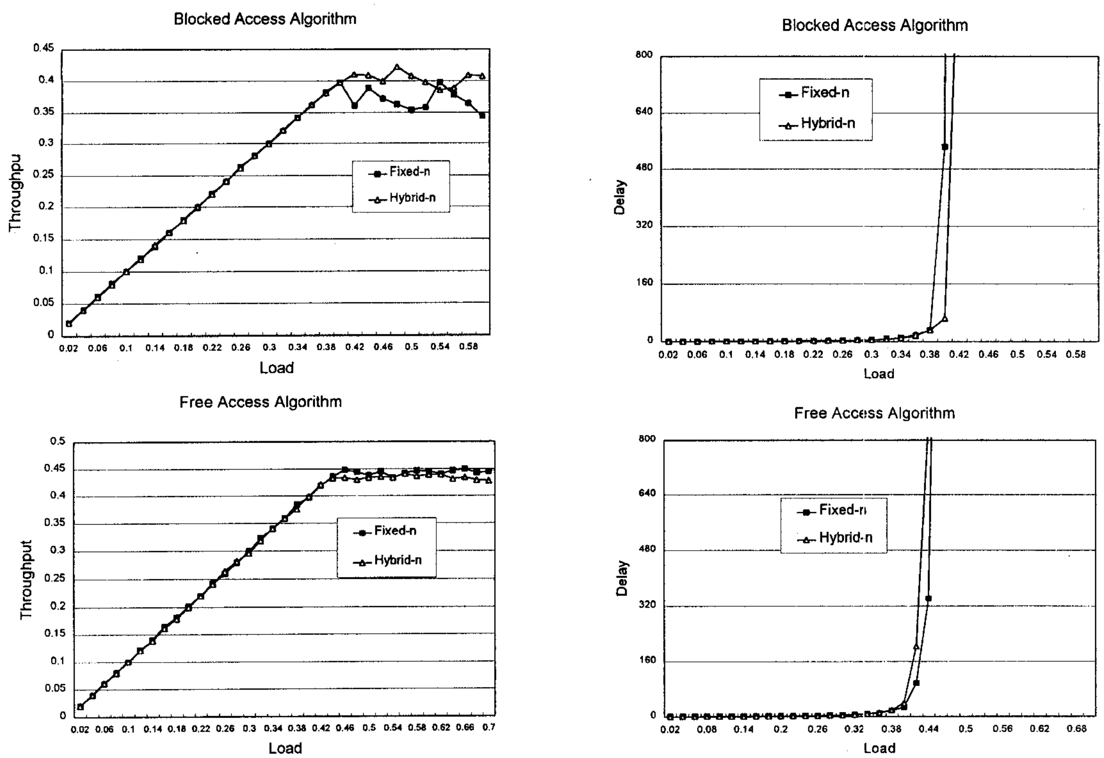

Figure 1. Simulation result for *TSV* = *{3, 3, 3, 2}*, 10 transmitters.

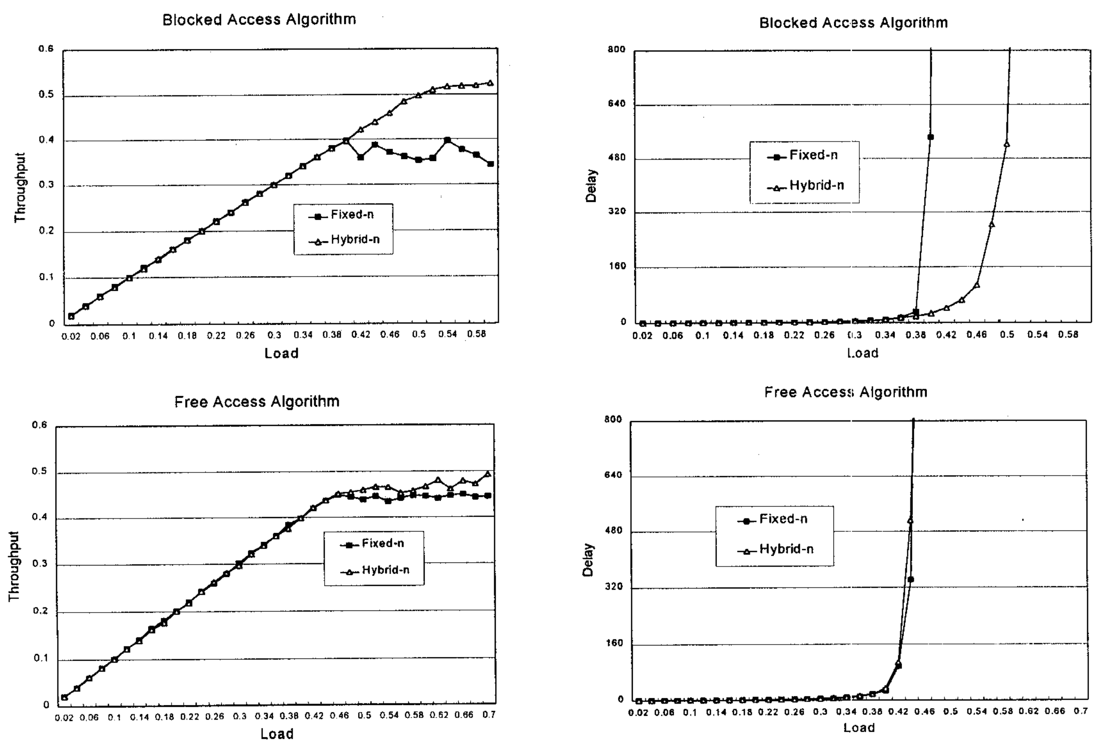

Figure 2. Simulation result for *TSV* = *{2, 4, 8, 2, 2, 2, 2, 2}*, 10 transmitters.

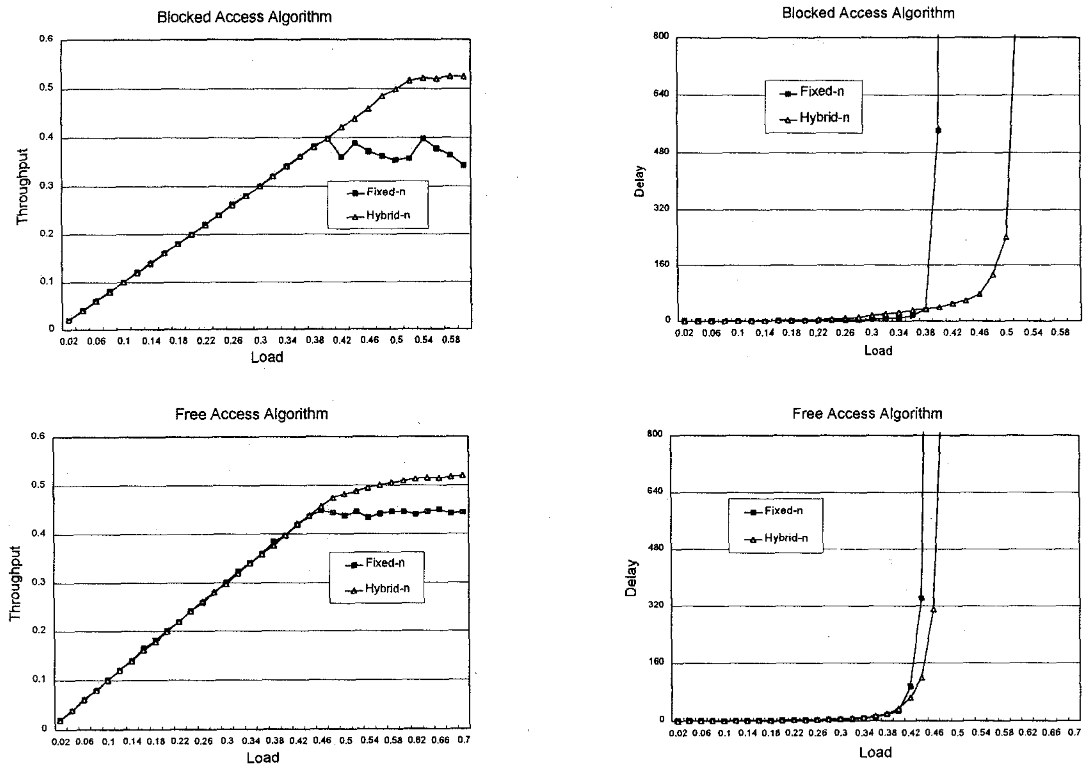

Figure 3. Simulation result for *TSV = {2, 4, 8, 16, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2}*, 10 transmitters.

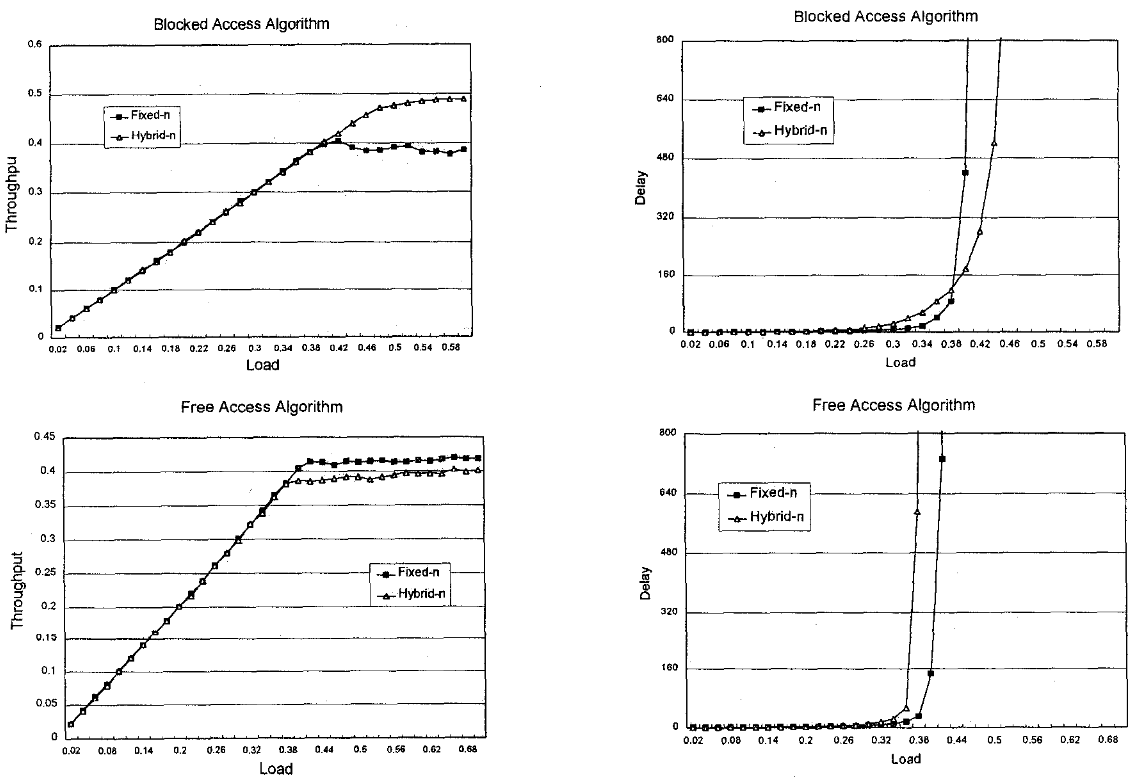

Figure 4. Simulation result for *TSV = {2, 4, 8, 16, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2}*, 50 transmitters.

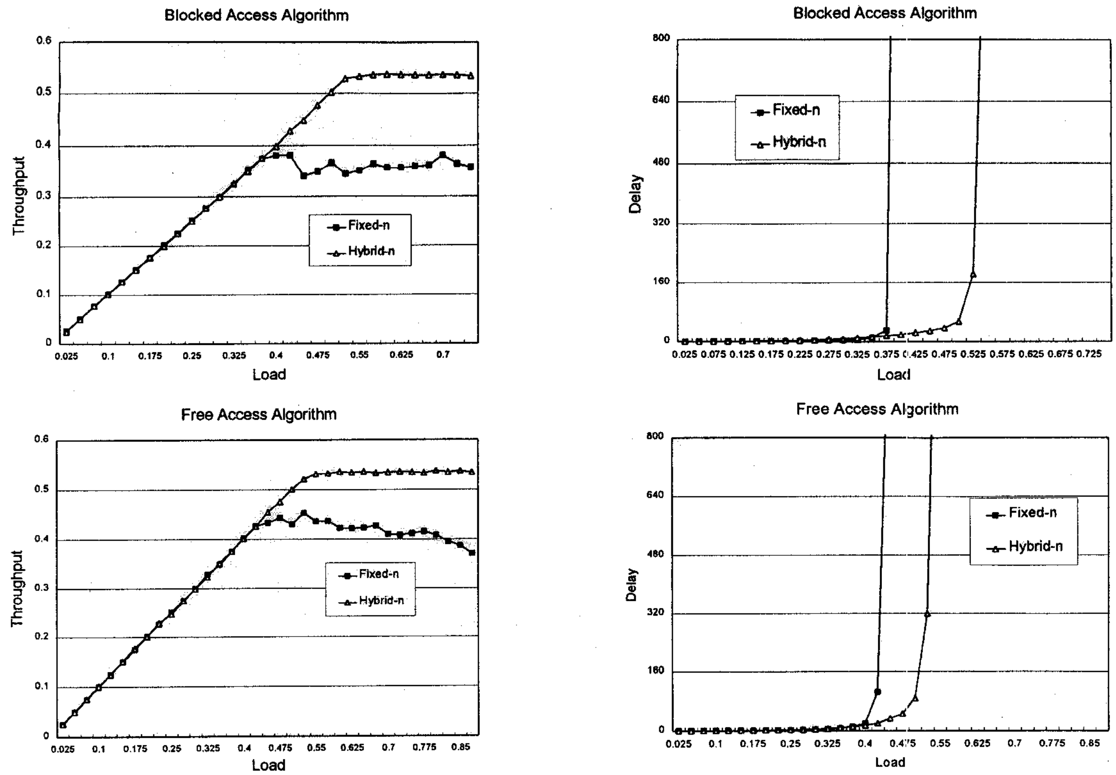

Figure 5. Simulation result for *TSV = {2, 4, 8, 16, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2}*, 5 transmitters.

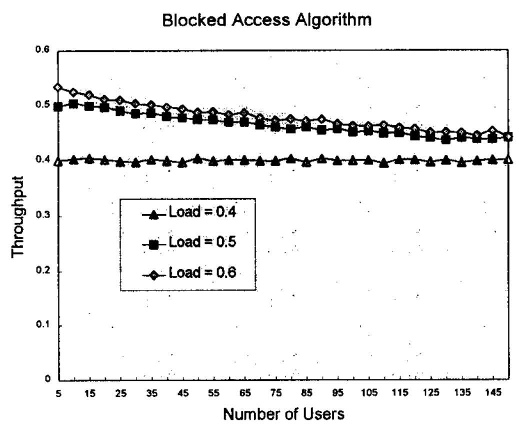

Figure 6. Result for different number of transmitters.