Abstract

We consider the random multiple access of a slotted broadcast communication channel. Packet arrivals for such channels are often modeled as Poisson processes because the latter have attractive theoretical properties and are well understood, even though a number of traffic studies have shown that packet interarrival times are not always exponentially distributed. For example, recent studies argue convincingly that traffic from a variety of working packet networks (LANs, WANs, etc.) is much better modeled using statistically self-similar processes to characterize the packet interarrival times of the aggregate traffic, which have much different properties than Poisson processes. Because of the great influence multiple access algorithms (used in the channel access subsystem) will have on the performance of third-generation digital wireless mobile communication systems, we study in this article the problem of random multiple access under packet traffic generated by interactive data applications (e.g., reading/composing short e-mail messages, responding to paging type messages, transferring files between the mobile and a fixed computer, and querying of a database). We examine and demonstrate the performance robustness of the above RAAs when they are driven by a strongly bursty packet arrival process, commonly found in interactive data applications.

Random Multiple Access of Broadcast Channels with Pareto Distributed Packet Interarrival Times

ZAHAROULA HARPANTIDOU AND MICHAEL PATERAKIS
TECHNICAL UNIVERSITY OF CRETE

In this article we study the random multiple access of a slotted broadcast communication channel loaded with data packet traffic generated by interactive applications. The analytic traffic model used in our study is taken from [1–4], and it fits measured Ethernet LAN network traffic [3], wide-area TCP live network traces of interactive Telnet applications [1], and wide-area network traffic measurements of applications such as Web and multimedia backbone (MBone) [3]. Although the operation environments for wide area and local area networks may be different from the context of wireless mobile communication systems, the model used is important since it describes the user behavior associated with interactive data applications [2]. Such applications will gain in importance in the wireless communications arena as small, portable, inexpensive computing devices proliferate.

The above-mentioned empirical studies of traffic measurements from a variety of packet networks have convincingly demonstrated that actual network traffic is *self-similar* in nature (i.e., bursty over a wide range of timescales). Self-similar traffic exhibits correlations over a wide range of timescales and therefore has *long-range dependence*, while traditional traffic models (e.g., Poisson packet arrivals) are short-range dependent in nature (i.e., they typically focus on a very limited range of timescales). In [4, 5] the authors show that the superposition of many alternating independent and identically distributed ON/OFF sources whose ON and OFF periods have high variability or infinite variance results in self-similar aggregate traffic. As the mathematical vehicle for modeling such phenomena they used *heavy-tailed* distributions with infinite variance (e.g., *Pareto*).

Therefore, in this article the packet interarrival times are assumed independent, identically distributed according to a Pareto distribution. The Pareto distribution is a distribution with memory, heavy tail, and strong burstiness. Depending on the value of one of its parameters it can have finite mean and infinite variance. The strong burstiness of such a packet arrival process leads to severe channel multiple access operational conditions capable of providing us with indications on the performance robustness of the random multiple access algorithms we study to the characteristics of the input packet arrival process.

We chose to study the throughput-delay trade-off of the following three random access algorithms (RAAs):

- Free access, stable, ideal, controlled ALOHA
- Free access, stable, implementable controlled ALOHA
- The limited feedback sensing free access m-ary stack algorithm [9, 10]

All three algorithms allow a channel user with a newly generated packet to transmit in the very next time slot following the packet generation instant (free access). The ideal and implementable ALOHA algorithms were chosen as representatives of the ALOHA family of RAAs. This family of algorithms constitute the most popular family of RAAs. They are straightforward to implement, and have enjoyed wide commercial employment [11]. The ALOHA algorithm, in its original form, dictates that packet-transmitting users be interested only in the channel outcome (i.e., channel feedback) of their own transmissions. Therefore, a network user does not need to monitor channel activity except for the time slots in which it is transmitting. Unfortunately, ALOHA algorithms using the above channel feedback sensing (also known as *purely acknowledgment type*) are notoriously unstable. It has been proven analytically and verified through simulations that for a large user population purely acknowledgment type ALOHA algorithms are unstable for any positive arrival rate of new packets [7, 8, 11]. Stable ALOHA "look-alike" RAAs have been devised [7, 8], with maximum throughput equal to e^{-1}.

1070-9916/98/$10.00 © 1998 IEEE

These algorithms require channel users to continuously monitor the channel feedback provided to them at the end of each time slot (this feedback sensing mechanism is known as *continuous feedback sensing*), and accordingly adjust the retransmission probability for the collided packets (if any). The channel feedback can be either binary (collision versus noncollision, C/NC), or ternary (empty versus success versus collision, E/S/C).

The ideal ALOHA algorithm we chose to study in this article assumes that users, based on their continuous observation of the channel feedback, know at the beginning of each time slot the exact total number of packets which have experienced at least one collision so far (backlogged packets), and the packet arrival rate to the system. Clearly, this is not easy to realize in practice (although elaborate estimation algorithms have appeared in the literature [8]). For this reason the algorithm is characterized as ideal. The throughput (packet delays) of the algorithm will provide an upper (lower) bound(s) on the throughputs (packet delays) achieved by all implementable stable ALOHA "look-alike" algorithms.

In practice, we cannot accurately estimate the backlog size. The next algorithm we study is the implementable controlled ALOHA which, in contrast to the ideal, can be realized in practice. This algorithm assumes that users, based on their continuous observation of the channel feedback (E/S/C), can estimate the backlog size at the beginning of each slot without knowing the exact number of backlogged users [7], and the packet arrival rate to the system, and accordingly adjust the retransmission probability for the collided packets.

The third algorithm we study was chosen as a representative of the large family of inherently stable random access collision resolution algorithms [7, 8, 11]. These algorithms achieve throughputs higher than e^{-1} without the need for elaborate estimation procedures. The chosen algorithm is very appealing from a practical point of view since it operates with limited channel feedback sensing (in which a user is required to monitor the channel feedback only when it has a packet to transmit).

The remainder of the article is organized as follows. In the second section we introduce the system and arrival models and the performance metrics we will be using. The examined algorithms are briefly described in the third section. Representative results of an extensive simulation study used to evaluate the throughput-delay trade-off for each algorithm and to facilitate comparisons are presented and discussed in the fourth section. Finally, the article is concluded in the fifth section.

The System and Arrival Models

The System Model

The idealized slotted multi-access model of [7, pp. 275–76], with a large number of identical channel users (theoretically infinite) and an aggregate packet arrival stream (with mean packet arrival rate denoted by λ) characterized by independent, identically distributed Pareto interarrival times, is adopted. Each newly arriving packet arrives at a new user. The channel is assumed time-slotted, all transmitted packets have the same length, and a packet transmission requires one time slot. If two or more channel users transmit a packet in a given time slot, there is a *collision* and the common receiver obtains no information about the contents or source of the colliding packets. If only one user transmits a packet in a given time slot, the packet is assumed to be correctly received by the receiver. Each packet involved in a collision must be retransmitted in some later slot until it is successfully received. At the end of each slot each channel user obtains *feedback* from the receiver specifying whether there was a collision or not in that slot (binary C/NC).

The important algorithmic performance metrics we consider are *channel throughput* and *packet delay*. We define packet delay as the time it takes from the generation of a packet until the end of the slot that contains the successful transmission of the packet. We look into the moments of the packet delay process because we know from earlier experience that RAAs with Poisson packet arrivals incur packet delay distributions with heavy tails and medians much larger than mean values [12, 13].

The Arrival Model

The packet interarrival times are assumed independent, identically distributed according to a Pareto distribution with shape parameter α and location parameter k ([1, 2]):

$$P(T \le t) = 1 - \left(\frac{k}{t}\right)^{\alpha} \quad k, \alpha \ge 0, t \ge k \tag{1}$$

The location parameter k is the minimum interarrival time between packets. It can easily be shown that if $\alpha \le 2$ the distribution has infinite variance, while if $\alpha \le 1$ the mean is infinite as well. Therefore, the Pareto distribution is heavy-tailed with infinite mean and variance (a distribution is said to be heavy-tailed if the ratio

$$\frac{P(T \ge t)}{t^{-\alpha}}$$

tends to 1 as $t \to \infty$, $\alpha \ge 0$). A more general definition of heavy-tailed distributions defines a distribution as heavy-tailed if the mean conditional exceedance of the random variable T, $E[T - t | T \ge t]$, is an increasing function of t [6]. For a Pareto distributed random variable T with $\alpha \ge 1$ (i.e., with finite mean), the mean conditional exceedance time is a linear function of t [6, p. 70]. Using this second definition, consider a Pareto distributed random variable that represents the waiting time of a customer in a service facility. Then the longer the customer has waited in the service system, the longer is its expected future waiting time. Contrast this behavior with waiting times distributed according to a light-tailed distribution (such as the uniform for which the mean conditional exceedance is a decreasing function of t) or a medium-tailed distribution (such as the memoryless exponential for which the mean conditional exceedance is independent of t, the waiting time so far) [1]. The above-mentioned mathematical properties account for the strong burstiness of the packet arrival process characterized by Pareto distributed packet interarrival times.

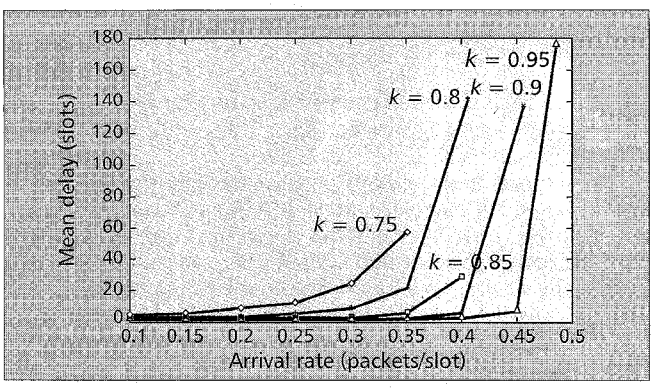
The Random Multiple Access Algorithms

The Stable Ideal Controlled ALOHA

For more on this, see [7, 8]. The backlogged packet retransmission probability at the beginning of a time slot is chosen to be

$$p = \frac{d - \lambda}{n} \tag{2}$$

where d is a constant, $\lambda < d \le 1$, to be optimally selected so that the average packet delay is minimized, λ is the current aggregate packet arrival rate, and n is the current number of backlogged packets. The new packets which have been generated during a slot are transmitted by the corresponding channel users during the next time lost (free access of newly generated packets). If a collision occurs, the newly generated



■ **Figure 1.** *Optimum mean packet delay as a function of the packet arrival rate λ, for different values of the location parameter k of the Pareto distribution.*

packets involved join the backlog set and remain there until they are successfully transmitted. Channel users with backlogged packets retransmit during a time slot according to the probability p. Note that the algorithm requires only C/NC channel feedback information.

The Implementable Stable Controlled ALOHA [7]

The algorithm operates by requiring each user to maintain an estimate $\hat{n}$ of the backlog n at the beginning of each slot. Each backlogged packet is then transmitted (independently) with probability:

$$p = \min\left\{1, \frac{1-\lambda}{\hat{n}}\right\} \tag{3}$$

The minimum operation in Eq. 3 limits p to at most 1. Subject to this limitation, the algorithm tries to maintain the average number of the newly generated and backlogged packets transmitted during a slot at one. The estimated backlog size at the beginning of slot $i + 1$ is based on the backlog estimate at the beginning of slot i, and the feedback of slot i according to the following rule (due to Rivest, [7]):

$$\hat{n}_{k+1} = \begin{cases} \max\{\lambda, \hat{n}_k + \lambda - 1\}, & \text{for empty or success} \\ \hat{n}_k + \lambda + (e-2)^{-1}, & \text{for collision.} \end{cases} \tag{4}$$

The addition of λ on the right side of Eq. 4 accounts for new arrivals, and the max operation ensures that the backlog size estimate is never less than the contribution from new arrivals. On successful transmissions, 1 is subtracted from the backlog to account for the successful departure. Finally, subtracting 1 from the backlog on idle slots and adding $(e - 2)^{-1}$ on collisions has the effect of decreasing $\hat{n}$ when too many idles occur and increasing $\hat{n}$ when too many collision slots occur. For large backlog size, if $\hat{n} = n$, each of the n backlogged packets is independently transmitted in the current slot with probability

$$p = \frac{1-\lambda}{n}.$$

Thus, the total packet transmission rate, $G(n)$, $G(n) = np + \lambda$, is 1, and by the Poisson approximation [7, p. 218], idles occur with probability $1/e$ and collisions with probability

$$\frac{(e-2)}{e},$$

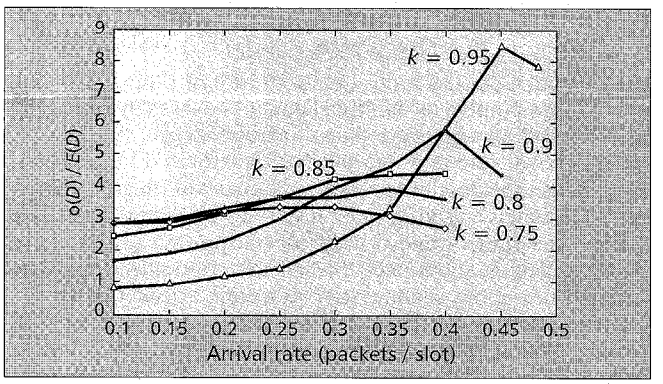
so that decreasing $\hat{n}$ by 1 on idles and increasing $\hat{n}$ by $(e - 2)^{-1}$ on collisions maintains on the average the balance between n and $\hat{n}$. We use this backlog size estimation, despite the fact that the packet arrivals in our case are characterized by Pareto distributed interarrival times instead of exponentially distributed (Poisson arrivals).

The m-ary Stack Collision Resolution [7, 8, 11]

The operation of this algorithm can be described (and visualized) with the help of a *virtual stack* with an infinite number of cells. We emphasize that the stack is a creation of human imagination, is used for illustration purposes, and is not part of the algorithm or realized by any channel user. Newly generated packets during a slot enter the bottom cell of the stack (known as the transmission cell) at the beginning of the next slot. Packets occupying the transmission cell at the beginning of a slot are transmitted during the slot. If the channel feedback is NC (noncollision), the transmission cell contained either zero or one packets (in the latter case, the single packet was successfully transmitted). The content of the stack moves one cell downward and the process repeats in the next slot. If the channel feedback is C (collision), the transmission cell of the stack contains at least two packets. The collided group of users probabilistically splits into m subgroups (the splitting mechanism is controlled by $m - 1$ probabilities which can be determined so that the average packet delay is minimized). The packets which have not transmitted in the current slot (occupying higher cells in the stack), move m cells upward in the stack to create m empty cells at the bottom of the stack for the m subgroups discussed above. At the beginning of the next slot, the newly generated packets (if any) enter the transmission cell of the stack and the process repeats. Like the stable ideal controlled ALOHA this algorithm also requires only binary C/NC channel feedback information. Each channel user is required to follow the channel feedback only during the time period it has a packet to transmit. This is in contrast to the case of the Stable Controlled ALOHA RAAs, where a user needs to follow the channel feedback continuously. In this work, we consider the case where $m = 3$. The selection of $m = 3$ is motivated by existing analytical results for the Poisson packet arrival process, which show that the throughput and delay performance of the algorithm is optimized when $m = 3$ [9].

Simulation Results

Simulation was chosen as the method for evaluating the performance of the algorithms. The fact that the Pareto distributed packet interarrival time is a random variable with memory, greatly complicates the throughput and delay analyses of the algorithms compared to the corresponding analyses when the packet arrival process is Poisson. Simulations were executed on a desktop UNIX workstation (SUN Sparcstation 5). Each run simulates the successful transmission of $N = 1$ million



■ **Figure 2.** *The ratio of the standard deviation to the mean of the packet delay as a function of the packet arrival rate λ, for different values of the location parameter* k *of the Pareto distribution.*

packets. In many cases, we simulated lower and higher N values (e.g., 500,000 and 2,000,000 packets, respectively). In all of these cases, we observed that by simulating 1,000,000 successfully transmitted packets, we were able to satisfactorily estimate the steady-state algorithmic performance metrics.

Notice that despite the fact that the assumed total number of channel users is infinite, in the simulation we do not need to simulate the behavior of the infinite number of users. We only need to generate the consecutive packet interarrival times, since according to the adopted system model, packets arrive one at a time and its newly arriving packet is assumed to arrive at a new user.

p	$k = 0.75$	$k = 0.8$	$k = 0.85$	$k = 0.9$	$k = 0.95$
$\lambda = 0.1$	$(0.54 - \lambda)/n$	$(0.55 - \lambda)/n$	$(0.49 - \lambda)/n$	$(0.5 - \lambda)/n$	$(0.73 - l\lambda)/n$
$\lambda = 0.15$	$(0.58 - \lambda)/n$	$(0.53 - \lambda)/n$	$(0.48 - \lambda)/n$	$(0.52 - \lambda)/n$	$(0.6 - \lambda)/n$
$\lambda = 0.2$	$(0.55 - \lambda)/n$	$(0.55 - \lambda)/n$	$(0.49 - \lambda)/n$	$(0.54 - \lambda)/n$	$(0.6 - \lambda)/n$
$\lambda = 0.25$	$(0.55 - \lambda)/n$	$(0.54 - \lambda)/n$	$(0.52 - \lambda)/n$	$(0.54 - \lambda)/n$	$(0.59 - \lambda)/n$
$\lambda = 0.3$	$(0.59 - \lambda)/n$	$(0.61 - \lambda)/n$	$(0.55 - \lambda)/n$	$(0.52 - \lambda)/n$	$(0.54 - \lambda)/n$
$\lambda = 0.35$	$(0.67 - \lambda)/n$	$(0.59 - \lambda)/n$	$(0.6 - \lambda)/n$	$(0.61 - \lambda)/n$	$(0.61 - \lambda)/n$
$\lambda = 0.4$	$(0.79 - \lambda)/n$	$(0.67 - \lambda)/n$	$(0.65 - \lambda)/n$	$(0.56 - \lambda)/n$	$(0.53 - \lambda)/n$
$\lambda = 0.45$	—	—	—	$(0.62 - \lambda)/n$	$(0.57 - \lambda)/n$
$\lambda = 0.4774$	—	—	—	—	$(0.54 - \lambda)/n$

■ **Table 1.** *The optimum backlogged packet retransmission probability (used in Figs. 1 and 2).*

Our simulation study determines the stability region of each algorithm (i.e., the values of the parameters k and α of the Pareto packet interarrival time distribution for which the corresponding random multiple access algorithm is stable). In addition, we study through the simulation the packet delay behavior of each algorithm. The mean and the ratio of standard deviation to the mean of the packet delays are plotted versus the packet arrival rate λ, for different values of the location parameter k of the Pareto distribution. Notice that the shape parameter a of the Pareto distribution is also involved in the obtained results, although not mentioned in the graphs, since its value depends on the values of λ and $k(\lambda = 1/E(T) = (\alpha - 1)/\alpha k)$. Therefore, when the value of the location parameter k remains constant and the value of the arrival rate λ changes, the value of the shape parameter α changes as well according to the above mentioned formula.

Finally, the reader is made aware that the vertical and horizontal axes ranges in Figs. 1–9 are different.

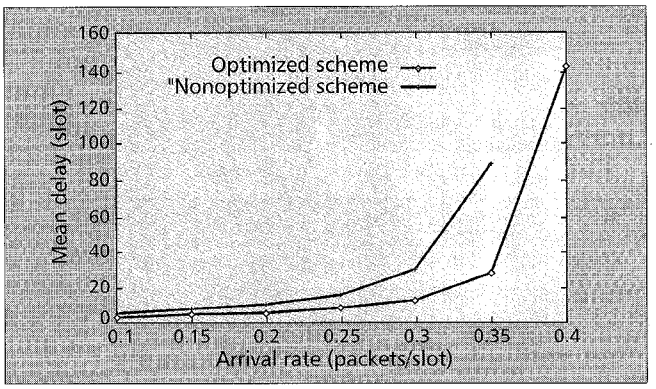
Simulation Results for the Stable Ideal Controlled ALOHA

The system was simulated extensively (mainly because the behavior of the Stable Ideal Controlled ALOHA RAA for Pareto distributed packet interarrival times was unknown, since to the best of our knowledge the problem was never studied before). The algorithm was found stable when the location parameter of the Pareto distribution is $k \geq 0.7$ slots, and provided that the shape parameter belongs to the interval [1, 1.83]. The highest algorithmic throughput is equal to 0.4774 (approximately 30 percent higher than the throughput of the same algorithm with Poisson packet arrivals), achieved when the location parameter k is equal to 0.95. The algorithmic throughputs for lower k values are lower than the throughput for $k = 0.95$. For example, when $k = 0.85$ the algorithmic throughput is between 0.4 and 0.45, while for $k = 0.75$ it is between 0.35 and 0.4.

In Figs. 1 and 2, we present results of the packet delay behavior of the algorithm. We define the optimum average packet delay as the average packet delay optimized with respect to the parameter d of the backlog retransmission probability for given λ and k values. Given the values of the parameters λ and k, the optimal average packet delay was determined through an exhaustive simulation search over all the possible values of the variable d, $(\lambda < d \leq 1)$, in the numerator of Eq. 2 giving the backlogged packet retransmission probability. In Fig. 1, the optimum average packet delay parameterized on the location parameter k is plotted versus the aggregate packet arrival rate λ. Each point in this plot was produced from a simulation run of 1 million successfully transmitted packets. From Fig. 1 we observe that the increase of the optimum average packet delay as λ increases becomes steeper with increasing k. When $k = 0.95$, the optimum average packet delay is maintained well below 20 slots for arrival rate values less than 0.45, and is almost negligible when $\lambda \leq 0.4$, while for the lowest k value we examined ($k = 0.75$, recall that $k \geq 0.7$ for stable algorithmic operation) the value of the optimum average packet delay exceeds 20 slots for λ values above 0.3. Furthermore, we observe that for a given λ value, the optimum average packet delay is a decreasing function of k. This is intuitively pleasing, since k is the minimum packet interarrival time. Therefore, as k approaches one from below the likelihood of packet collisions decreases. Notice that the mean delay curves for $k = 0.75$ and $k = 0.85$ are missing their final data point. This is because the algorithm is approaching its corresponding maximum throughput, and the mean delay values at this point are increasing dramatically and are outside the range of the vertical axis in the figure. The same logic explains missing final data points in the remaining figures throughout the article.

For the reasons we explained in the last sentence of the second section, we estimate via simulation the ratio of the standard deviation to the average of the packet delay parameterized on the location parameter of the Pareto distribution k, as a function of the packet arrival rate λ. The estimator used to estimate the standard deviation of the packet delay is the standard unbiased one [14, p. 288]. Figure 2 shows the fraction of the standard deviation to the average of the packet delay process as a function of λ. We observe that for relatively large λ values ($\lambda > 0.35$), as k increases the ratio increases as well. For example, when $k = 0.75$ the ratio is maintained around 3, while when $k = 0.95$ the ratio varies between 1 and 8. The latter result basically means that the delay of a randomly selected packet may be as high as 8 x the average packet delay value. This observation reaffirms what we already know about the characteristics of the packet delay process incurred by RAAs [12, 13]. Knowledge of the average packet delay value alone does not suffice to predict the delay of a randomly selected packet.

The optimal backlog retransmission probabilities for given λ and k values are shown in Table 1. They were produced through an exhaustive simulation search over all the possible values of the variable d, $(1 < d \leq 1)$, in the numerator of Eq. 2 giving the backlogged packet retransmission probability. The



■ **Figure 3.** *Mean packet delay versus packet arrival rate λ for the nonoptimized and optimized schemes for* k = *0.8.*

search was done very carefully, because we discovered that the optimum average packet delay is extremely sensitive even to small changes in the value of the variable d [15]. The dashed boxes correspond to unstable cases (i.e., there is no d value, $\lambda < d \leq 1$, which yields a stable algorithmic operation). From Table 1 we observe that the optimal d values depend on both λ and k, and that it does not seem to exist as a unique pattern exhibited by the optimal d values. Finally, it is interesting to find how far the average packet delay can be from its optimum value when using $d = 1$ in Eq. 2 for all λ values. The above packet retransmission policy does not optimize the average packet delay with respect to the parameter d; however, it is simpler to implement.

From Fig. 3, we observe that when the location parameter of the Pareto distribution is equal to $k = 0.8$, the average packet delay in the non-optimized case is higher than the corresponding delay in the optimized case for all λ values as it was expected (the difference becomes significant for λ values larger than 0.3). Furthermore, from Fig. 3 we observe that the optimized scheme achieves higher throughput. Similar conclusions are drawn for different values of k. The corresponding results are not shown here due to lack of space, the interested reader is referred to [15].

Simulation Results for the Stable Implementable Controlled ALOHA

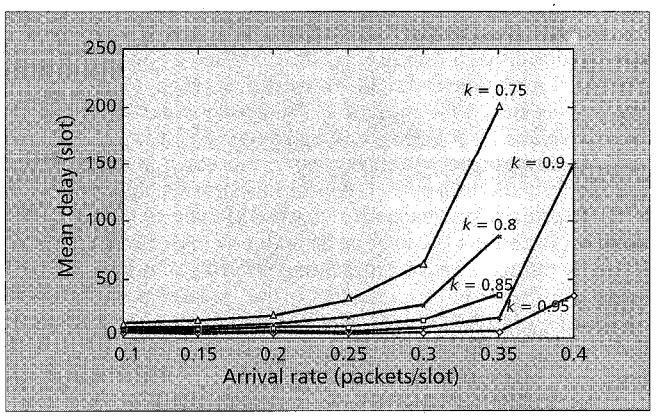
The algorithm was found stable when the location parameter of the Pareto distribution is $k \geq 0.7$ slots, and provided that the shape parameter α belongs to the interval [1,1.68]. The maximum algorithmic throughput is equal to 0.426, achieved when $k = 0.95$. In Figs. 4 and 5, we present results of the packet delay behavior of the algorithm.

In Fig. 4, the average packet delay, parameterized on the location parameter k, is plotted versus the aggregate packet arrival rate λ. Each point in this plot was produced from a simulation run of 1 million successfully transmitted packets. When $k = 0.95$, the average packet delay is maintained well below 35 slots for arrival rate values less than 0.4, and is almost negligible when $\lambda \leq 0.35$, while for the lowest k value we examined ($k = 0.75$, recall that $k \geq 0.7$ for stable algorithmic operation) the value of the average packet delay exceeds 30 slots for λ values above 0.25. The case $k = 0.75$ is unstable for $\lambda = 0.4$, while the mean packet delays for $k = 0.8, 0.85$ are outside the range of the vertical axis in the figure. Furthermore, we observe that for a given λ value, the average packet delay is a decreasing function of k. This is intuitively pleasing, for the reason we explained previously. Figure 5 shows the fraction of the standard deviation to the average of the packet delay process as a function of λ. We observe that for relatively large λ values ($\lambda > 0.3$), as k increases, the ratio increases as well. For example, when $k = 0.75$ the ratio is maintained near 3, while when $k = 0.95$ the ratio varies between 1 and 6.5. The latter result basically means that the delay of a randomly selected packet may be as high as 6.5 x the average packet delay value. This behavior of the packet delay process is similar to what we already know about the characteristics of the packet delay process incurred by RAAs [12, 13], as we also noticed for the case of the ideal ALOHA algorithm.

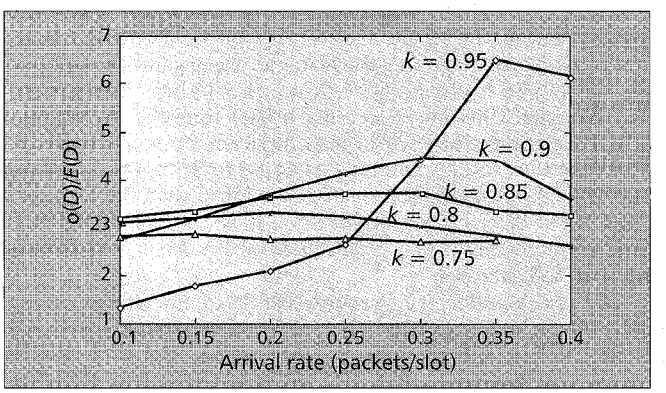
To fairly compare this implementable algorithm with the ideal ALOHA, we have simulated the ideal algorithm with retransmission probability

$$p = \frac{1-\lambda}{n} \qquad (5)$$

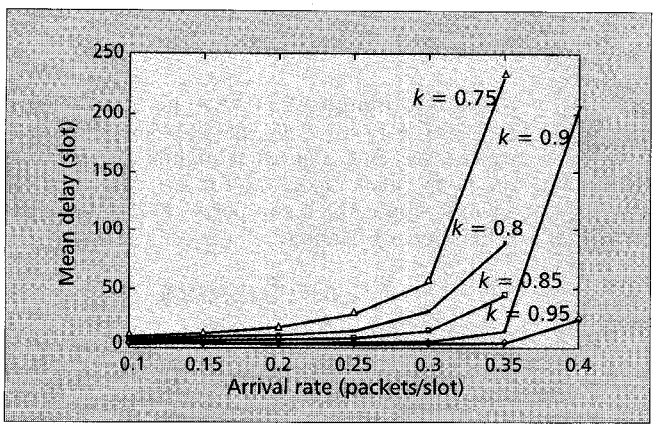
(i.e., we set $d = 1$ in Eq. 2), and we present the results in Figs. 6 and 7. (In Fig. 6, the case $k = 0.75$ is unstable for $\lambda = 0.4$, while the mean packet delays for $k = 0.8, 0.85$ and $\lambda = 0.4$ are outside the range of the vertical axis in the figure.) Comparing the results in Figs. 4 and 6, we observe that the average packet delays for the ideal controlled ALOHA algorithm are close to those of the implementable controlled ALOHA algorithm. For arrival rates $\lambda \leq 0.3$ the average packet delays for the two algorithms are almost the same. Only for $\lambda = 0.35$ and $k = 0.75$, the average packet delay for the ideal algorithm



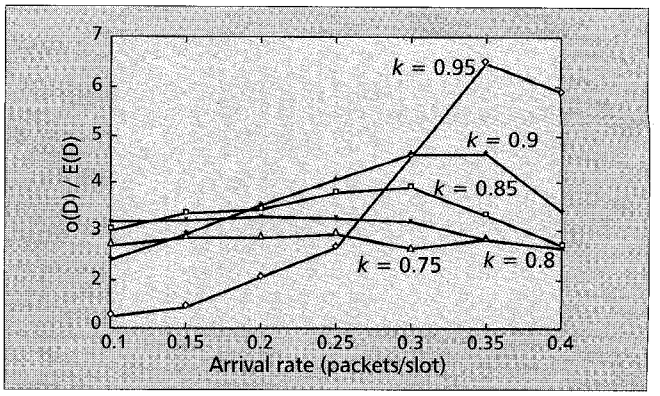
■ **Figure 4.** *Mean packet delay as a function of the packet arrival rate λ, for different values of the location parameter* k *of the Pareto distribution, with estimated backlog and retransmission probability of the backlogged packets* p = *1* – λ/n̂.



■ **Figure 5.** *The ratio of the standard deviation to the mean of the packet delay as a function of the packet arrival rate λ, for different values of the location parameter* k *of the Pareto distribution, with estimated backlog and retransmission probability of the backlogged packets* p = *1* – λ/n̂.



■ **Figure 6.** *Mean packet delay as a function of the packet arrival rate* λ *for different values of the location parameter* k *of the Pareto distribution and retransmission probability of the backlogged packets* p = *(1 – λ)/n.*



■ **Figure 7.** *The ratio of the standard deviation to the mean of the packet delay as a function of the packet arrival rate* λ *for different values of the location parameter* k *of the Pareto distribution and retransmission probability of the backlogged packets* p = *(1 – λ)/n.*

is higher than the corresponding delay for the implementable algorithm (by almost 30 slots).

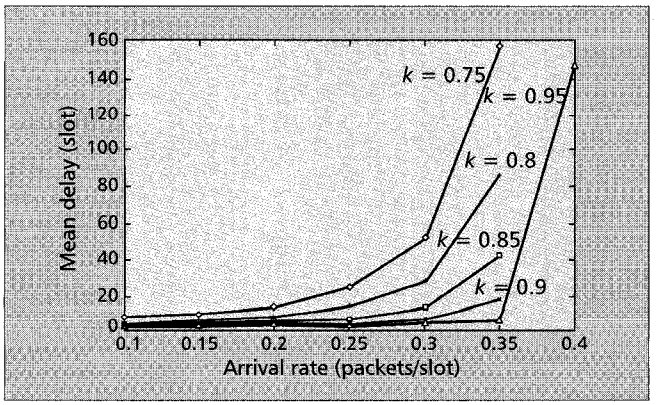
From these results we conjecture that the backlog estimator in Eq. 4 must be very accurate. To confirm this, we have estimated via simulation the average value of the difference between the estimated and actual backlog. From the obtained results, we noticed that for a given λ value the mean of the absolute value of the above difference is an increasing function of *k*. Also, the same mean is an increasing function of the arrival rate λ. This explains the small difference between the average packet delay values for λ < 0.35 in Figs. 4 and 6.

Simulation Results for the m*-ary Stack Algorithm*

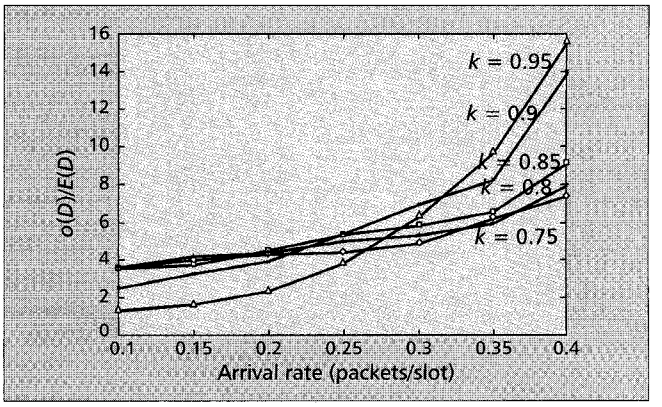
Here we present and discuss results for the $m = 3$ case only. The reason is that the $m = 2$ case yields higher average packet delays and lower throughputs than the $m = 3$ case. Representative results for the $m = 2$ case can be found in [15]. The algorithm was found stable when the location parameter of the Pareto distribution is $k \geq 0.7$ slots, and provided that the shape parameter a belongs to the interval [1, 1.646]. The maximum algorithmic throughput is equal to 0.41315 (slightly higher than the corresponding for Poisson packet traffic which is equal to 0.4), achieved when $k = 0.95$.

In Fig. 8, we show the optimum average packet delay (corresponding to the appropriate splitting probabilities, the selection of which will be explained shortly), parameterized on the location parameter *k*, versus the aggregate packet arrival rate λ. Each point in this plot was produced from a simulation run of 1 million successfully transmitted packets. We observe a similar behavior with the one observed in the case of the stable ideal controlled ALOHA RAA (shown in Fig. 1). When $k = 0.95$, the optimum average packet delay is maintained below 4 slots for arrival rate values λ < 0.35, while for the lowest *k* value we examined ($k = 0.75$) the optimum average packet delay value varies between 7 slots (for λ = 0.1) and 155 slots (for λ = 0.35), and it exceeds 20 slots for λ ≥ 0.25. Unlike what was found for the other two algorithms examined in the article, the case $k = 0.75$ is stable here for λ = 0.4. This is because when *k* is small we have more packet collisions, in which case stack-type RAAs perform better than ALOHA-type RAAs. Furthermore, we observe from Fig. 8 that for a given λ value the optimum average packet delay decreases significantly with increasing *k*. Figure 9 shows the ratio of the standard deviation to the average of the packet delay parameterized on the location parameter of the Pareto distribution *k* as a function of the packet arrival rate λ. Once again we observe that for relatively large λ values (λ > 0.35), as *k* increases so does the variability of the packet delay around its mean. For example, when $k = 0.75$ we observe that the ratio in Fig. 9 varies between 4 and 7, while for $k = 0.95$ the ratio is maintained between 1.5 and 6 for λ ≤ 0.3, and it takes the value 15 (!) when λ = 0.4. The latter observation reaffirms that knowledge of the average packet delay value alone does not suffice to predict the delay of a randomly selected packet.

Finally, the optimal splitting probabilistic mechanism



■ **Figure 8.** *Optimum mean packet delay as a function of the packet arrival rate* λ*, for different values of the location parameter* k *of the Pareto distribution (*m = *3).*



■ **Figure 9.** *The ratio of the standard deviation to the mean delay as a function of the packet arrival rate* λ*, for different values of the location parameter* k *of the Pareto distribution (*m = *2)*

p	k = 0.75	k = 0.8	k = 0.85	k = 0.9	k = 0.95
λ = 0.1	0.31/0.67	0.31/0.62	0.3/0.63	0.33/0.6	0.31/0.66
λ = 0.15	0.32/0.65	0.29/0.61	0.3/0.63	0.3/0.59	0.3/0.64
λ = 0.2	0.32/0.64	0.3/0.59	0.3/0.63	0.28/0.63	0.28/0.66
λ = 0.25	0.29/0.65	0.28/0.64	0.3/0.6	0.32/0.65	0.33/0.63
λ = 0.3	0.31/0.63	0.28/0.6	0.28/0.63	0.29/0.64	0.32/0.62
λ = 0.35	0.28/0.64	0.29/0.59	0.28/0.59	0.28/0.62	0.3/0.59
λ = 0.4	0.32/0.62	0.3/0.65	0.31/0.64	0.28/0.6	0.28/0.6

Table 2. *The optimum splitting threshold parameters (used in Figs. 4 and 5.)*

which controls the splitting of a collided group of packets into $m = 3$ subgroups is shown in Table 2. The results have been produced through exhaustive search over all possible values of the threshold parameters P_1 and P_2 ($0 < P_1 < P_2 < 1$) via series of simulation runs. The meaning of the parameters P_1 and P_2 is made clear next. According to the algorithm, a colliding packet decides to stay in the transmission cell (bottom cell of the virtual stack, or cell 0) with probability P_1, decides to move to cell 1 of the virtual stack with probability $P_2 - P_1$, and decides to move to cell 2 of the virtual stack with probability $1 - P_2$. From the results shown in Table 2 we observe that the optimal splitting threshold parameters P_1 and P_2 depend on both λ and k, but otherwise do not exhibit any specific pattern (other than that their values are close to 1/3 and 2/3, respectively). Based on the latter observation one can conjecture that the average packet delays when $P_1 = 1/3$, and $P_2 = 2/3$, will not be much higher than the corresponding optimal average packet delay values. This conjecture has been confirmed through computer simulations [15].

Mean Packet Delay Comparison of the Algorithms with the Same Location Parameter

In Figs. 10 and 11, we plot the mean packet delay versus the arrival rate λ when the location parameter k takes the values 0.75, and 0.85, respectively. From the results presented in these figures, we observe that the ideal controlled ALOHA algorithm achieves the best performance. For low values of the location parameter k (e.g., $k = 0.75$ in Fig. 10), the m-ary stack algorithm comes second. This is because when k is small we have more collisions, in which case stack-type RAAs perform better than ALOHA-type RAAs. As k increases (e.g., $k = 0.85$), the implementable controlled ALOHA algorithm slightly outperforms the m-ary stack algorithm for $\lambda \geq 0.3$, while for lower λ values the mean delays of the two algorithms are very similar.

Conclusions

In this work we have considered the problem of the random multiple access of a packet broadcast slotted channel with Pareto distributed, independent, packet interarrival times. The Pareto distribution is a distribution with memory, strong burstiness and heavy tail. The problem considered is interesting because the traffic model used was found to fit well interactive data applications in working wide-area and local-area packet networks. Such data applications are expected to be a critical component of future wireless communications networks.

The multiple random access algorithms we examine provide free and immediate access to channel users with newly arriving packets. Their performance can be optimized provided that the parameters of the corresponding packet retransmission mechanisms for the colliding packets are appropriately selected.

Through an extensive simulation study we have shown that the examined RAAs are stable, and depending on the location parameter k of the Pareto packet interarrival time distribution, they achieve higher throughputs than those under Poisson packet arrivals. One of the reasons is that for a Pareto distributed packet interarrival time T, $T \geq k$ where k is the location parameter of the Pareto distribution, while for Poisson packet arrivals (exponentially distributed packet interarrival times) $T \geq 0$.

The behavior of the packet delay distribution of the above RAAs resembles the corresponding behavior under Poisson packet arrivals. The packet delay distributions are characterized by low mean values and by standard deviations which are multiples of the corresponding mean values (the same behavior applies under Poisson packet traffic). Only when the cumulative packet arrival rate λ approaches the maximum throughput of the algorithm do the mean and the ratio of the standard deviation to the mean of the packet delay increase dramatically.

Of the three RAAs we examined, the first is ideal (nonimplementable), while the other two are implementable. The

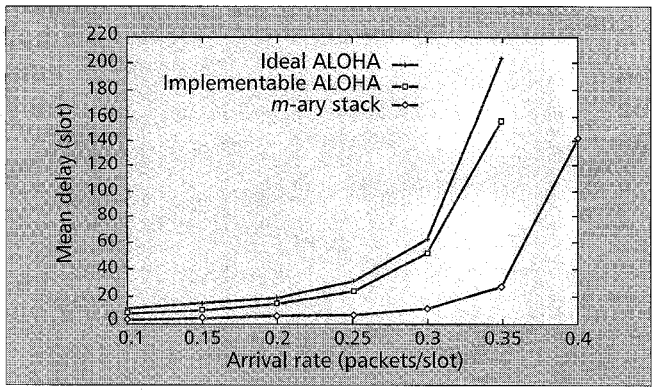


Figure 10. *Mean packet delay as a function of packet arrival rate λ for the ideal controlled ALOHA, implementable controlled ALOHA, and* m*-ary stack algorithms, for the same value of location parameter* k *= 0.75.*

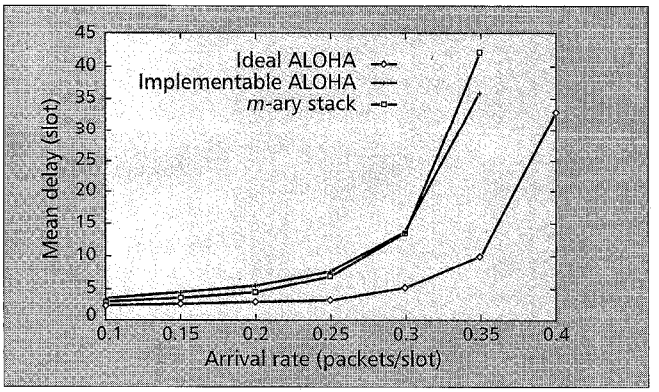


Figure 11. *Mean packet delay as a function of packet arrival rate λ for the ideal controlled ALOHA, implementable controlled ALOHA, and* m*-ary stack algorithms, for the same value of location parameter* k *= 0.85.*

nonimplementable RAA is the stable ideal controlled ALOHA, and it was examined because it provides an upper (lower) bound on the throughput (packet delays) achieved by all implementable stable ALOHA RAAs.

The second is a representative of the implementable stable controlled ALOHA family of RAAs. From the results found previously, we conclude that the implementable stable ALOHA RAA achieves throughput and delay performance similar to its ideal counterpart, despite the fact that the backlogged packet retransmission mechanism of the former algorithm has been designed with Poisson packet traffic in mind. The latter observation leads us to the conclusion that the backlog estimator (due to Rivest) presented earlier in Eq. 4, is *robust* with respect to the nature of the packet interarrival time distribution. Therefore, Rivest's backlog estimation algorithm appears to be a particularly simple and effective way to stabilize the ALOHA RAA *irrespective of the burstiness* of the packet arrival process.

The third is an inherently stable stack-type collision resolution RAA. It is very appealing from a practical point of view, since in contrast to the previous two algorithms it operates with limited channel feedback sensing and is simpler to implement because it does not use elaborate backlog estimation mechanisms. The stack RAA achieves lower maximum throughputs than the ideal and implementable ALOHA RAAs because it uses limited channel feedback sensing. However, for small values of location parameter k and large values of packet arrival rate λ, the algorithm outperforms the ALOHA RAAs in terms of mean delay (see Figs. 4, 6, and 8 for $k = 0.75$). This is because when k is small we have more collisions, in which case the stack-type RAAs perform better than the ALOHA RAAs.

In conclusion, our work has shown that RAAs designed and evaluated under the Poisson packet arrival process assumption can also efficiently operate and be optimized under the extremely bursty packet arrival process characterized by Pareto distributed packet interarrival times.

References

[1] V. Paxson and S. Floyd, "Wide Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Trans. Networking*, vol. 3, no. 3, June 1995, pp. 226–44.

[2] G. Anastassi *et al.*, "A Bandwidth Reservation Protocol for Speech/Data Integration in TDMA-Based Advanced Mobile Systems," *Proc. 1996 INFOCOM*, pp. 722–29.

[3] W. Willinger *et al.*, "Self-Similarity through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level," *IEEE/ACM Trans. Networking*, vol. 5, no. 1, 1997, pp. 71–86.

[4] W. Leland *et al.*, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Trans. Networking*, vol. 2, no. 1, 1994, pp. 1–16.

[5] M. Taqqu, W. Willinger, and R. Sherman, "Proof of a Fundamental Result in Self-Similar Traffic Modeling," *Comp. Commun. Rev.*, vol. 27, 1997, pp. 5–23.

[6] B. Arnold, *Pareto Distributions*, Baltimore, MD: International Cooperative, 1983.

[7] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed., Prentice Hall, 1992.

[8] R. Rom and M. Sidi, *Multiple Access Protocols: Performance Analysis*, Springer-Verlag, 1990.

[9] P. Mathys and P. Flajolet, "Q-ary Collision Resolution Algorithms in Random Access Systems with Free or Blocked Channel Access," *IEEE Trans. Info. Theory*, vol. IT-31, no. 2, Mar. 1985, pp. 217–43.

[10] L. Merakos and C. Bisdikian, "Delay Analysis of the n-ary Stack Algorithm for a Random Access Broadcast Channel," *IEEE Trans. Info. Theory*, vol. IT-34, no. 5, Sept. 1988.

[11] C. Bisdikian, "A Review of Random Access Algorithms," IBM res. rep. RC 20348, Jan. 1996.

[12] L. Georgiadis, and M. Paterakis, "Bounds on the Delay Distribution of Window Random-Access Algorithms," *IEEE Trans. Commun.*, vol. COM-41, no. 5, May 1993, pp. 683–93.

[13] G. Polyzos and M. Molle, "A Queueing Theoretic Approach to the Delay Analysis for the FCFS 0.487 Conflict Resolution Algorithm," *IEEE Trans. Info. Theory*, vol. IT-39, no. 6, Nov. 1993, pp. 1887–1906.

[14] A. Law and W. Kelton, *Simulation Modeling and Analysis*, 2nd ed., New York: McGraw Hill, 1991.

[15] Z. Harpantidou, "Random Multiple Access with Pareto Distributed Packet Interarrival Times," Diploma thesis, Dept. of Electronics and Computer Eng., Tech. Univ. of Crete, Greece, 1996 (in Greek).

Biographies

ZAHAROULA HARPANTIDOU received her five-year diploma degree in electronics and computer engineering from the Technical University of Crete in 1996. During the period September 1996–February 1997, she was a research assistant with CNUCE-CNR (Consiglio Nationale delle Ricerche) in Pisa, Italy. She is currently with INTRACOM, Athens, Greece, in the Network Management Systems department. She is also pursuing her Master's degree at the Technical University of Crete, on multimedia applications in computer networks. Her research interests are in the areas of computer network protocols, wireless communication networks, and real-time network applications.

MICHAEL PATERAKIS [SM] (pateraki@telecom.tuc.gr) received his Diploma degree from the National Technical University of Athens, Greece, his M.Sc. degree from the University of Connecticut, and his Ph.D. degree from the University of Virginia in 1984, 1986, and 1988, respectively, all in electrical engineering. Since 1995 he has been an associate professor in the Department of Electronic and Computer Engineering at the Technical University of Crete, Greece. He was an associate professor in the Department of Computer and Information Sciences at the University of Delaware, where he started in September 1988. His research interests include computer communication networks with emphasis on protocol design, modeling, and performance evaluation of broadband high-speed multimedia networks, multiple access wireless cellular communication systems, and packet radio networks; and queuing and applied probability theory and their application to computer communication networks. He has published extensively in archival journals, refereed conference proceedings, and edited books in the above-mentioned technical areas. He served on the Technical Program Committees of the 1991 International Conference on Distributed Computing Systems, 1992 and 1994 IEEE INFOCOMs, and the 1997 IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems (HPCS'97). Professor Paterakis is a member of the IEEE Technical Committee on Computer Communications, the Greek Chamber of Professional Engineers, and the Greek Association of Electrical and Electronic Engineers.