A Game Theoretic Analysis of Protocols Based on Fountain Codes *

Luis López
LADyR
Universidad Rey Juan Carlos
28933 , Móstoles, Madrid (Spain)
luis.lopez@urjc.es

Antonio Fernández
LADyR
Universidad Rey Juan Carlos
28933 , Móstoles, Madrid (Spain)
antonio.fernandez@urjc.es

Vicent Cholvi
Universitat Jaume I
12071, Castelló (Spain)

Abstract

In this paper we analyze a novel paradigm of reliable communications which is not based on the traditional timeout-and-retransmit mechanism of TCP. Our approach, which we call FBP (Fountain Based Protocol), consists on using a digital fountain encoding which guarantees that duplicate packets are not possible. Using Game Theory, we analyze the behavior of TCP and FBP in the presence of congestion. We show that hosts using TCP have an incentive to switch to an FBP approach obtaining a higher throughput. Furthermore, we also show that a Nash equilibrium takes place when all hosts use FBP. At this equilibrium, the performance of the network is similar to the performance obtained when all hosts comply with TCP.

1. Introduction

Congestion control in communication systems has been an important and largely studied issue. Since many communication systems in our days are based on the principle of sharing common resources (e.g., routers, communication links) among different users, one of the main objectives of congestion control schemes is to establish rules to guarantee that the common resources are used optimally and shared fairly among users. However, most of these schemes require end-users to behave in a cooperative way.

Nevertheless, it is currently impossible to guarantee that end-users will not act in a selfish manner. If they use TCP, this means that they will never reduce their sending rates even in the presence of congestion. As it has been shown in [1, 10], if this happens and users overload the network, the total throughput of the network drops. This happens since most Internet routers use a drop-tail FIFO (First In First Out) scheduling discipline, and users can obtain more network bandwidth by transmitting more packets per unit of time. (With this policy, the more packets a user sends the more resources it gets.) Thus, the optimal strategy for each user is strongly suboptimal for the network as a whole.

Among the different techniques that can be used to evaluate the impact of selfish users, one of the most popular is *Game Theory* [3, 11]. Game theory is a tool for analyzing the interaction of decision makers with conflicting interests. Roughly speaking, a *game* has three components: a set of players, a set of possible actions for each player, and a set of utility functions mapping action profiles into real numbers. In our case, the *game players* are the users and the congestion control schemes establish the *game rules*. Each player has a strategy, which establishes the traffic that it injects into the network.

The behavior of the TCP protocol has already been addressed with a game-theoretic approach by several authors. Some of the most remarkable works in this field are the ones carried out by Nagle [9, 10], and Garg et al. [4]. Both of them show that evil (selfish) behavior leads to disaster and propose solutions based on creating incentive structures in the systems that discourage this behavior.

Another interesting work based on slightly different ideas is the one carried out by Akella et al. [1]. They show that a novel stateless buffer scheduling discipline called CHOKE [12], which does not require per-packet processing, may be useful in restoring the Nash equilibrium efficiency [1].

*This work has been partially supported by URJC under Grant PPR-2004-42, by MCyT under Grant No. TSI2004-02940 and by Bancaixa under Grant No. P1-1B2003-37.

[1] An important concept in game theory is the Nash equilibrium. In our context, a Nash equilibrium is a scenario where no selfish user has incentive to unilaterally deviate from its current state. Clearly, being in a Nash

IEEE COMPUTER SOCIETY

The abovementioned problem has a closer analogue with the, so called, Tragedy of the Commons [5] problem in economics. To understand precisely what a Tragedy of the Commons is, we need first to observe that, in the context of Game Theory, players choose their strategy in a selfish way trying to maximize their benefit. If the system gets into a state in which no player has an incentive to unilaterally change its strategy we say that the system has reached the Nash equilibrium. In this context, a game is a Tragedy of the Commons when (i) there is always an incentive for a new player to become evil (this guarantees that the Nash equilibrium is reached when all players are evil) and (ii) the final benefit for evil players in the Nash equilibrium is under the initial benefit of fair players when all players collaborate. Hence, all players loose.

In the context of network protocols, it has been observed by several authors [10, 1, 6] that when hosts behave in a selfish manner and do not comply with the TCP congestion control mechanisms (for example, by using lower timeouts), a Tragedy of the Commons arises and the network throughput drops due to the presence of duplicate packets.

In this paper we compare, from a game theoretic point of view, TCP with a protocol based on digital fountain codes [7, 15], which we call Fountain Based Protocol (FBP). This protocol is similar to UDP but uses fountain codes to avoid the presence of duplicate packets. Because of this, it does not require any type of packet retransmission mechanism. We show that the Nash equilibrium of a network with a mixture of hosts using TCP and hosts using FBP is reached when all hosts behave in a selfish manner (by using FBP instead of TCP), but that this does not drive the network to a collapse. Moreover, we demonstrate that, in general, it does not even lead to a Tragedy of the Commons, since the throughput of hosts, even in the case where all of them act in a selfishly way, is no less than the throughput obtained when all host comply with the TCP protocol.

In the next section we present the details of the protocol FBP. In Section 3 we present the network model we use, with some analytical results under that model. In Section 4 we present simulations of the same network and compare them with the previous analysis. Finally, in Section 5 we present some concluding remarks.

2. Protocols Based on Fountain Codes

The basic principle behind the use of *digital fountain codes* [7, 2, 15] is conceptually simple. Roughly speaking, it consists of sending a stream of different encoded packets into the network, from which a receiver can reconstruct the source data. The key property is that the source data can be reconstructed from any subset of the encoded packets of (roughly) the same size as the source data. Such a concept is similar to ideas found in the seminal works of Maxemchuk [8] and Rabin [13].

A first approach to the codes we need are classical erasure codes. Erasure codes generate additional redundant packets from the original k packets of the source data. Then, they guarantee that the source data can be recovered from any subset of $(1+\varepsilon)k$ packets ($1+\varepsilon$ is called the *decoding inefficiency*). Hence, they allow to tolerate packet losses during transmissions. For instance, one can use Reed-Solomon erasure codes [14], since they have the property that a decoder at the receiver can reconstruct the original source data whenever it receives any k of the transmitted packets (i.e., their decoding inefficiency is 1). However, the encoding and decoding processing times for such a class of codes are prohibitive.

Fortunately, digital fountain codes have been recently proposed [7, 15]. These codes have very fast enconding and decoding. Furthermore, the number of encoded packets that can be generated from the source data by using these codes is potentially limitless and can be determined on the fly. That allows a digital fountain to take source data consisting of k packets and produce as many encoded packets as needed to meet the user demand. The only drawback is that these codes have a decoding inefficiency a little larger than 1 (i.e., $\varepsilon > 0$).

Now, we consider the protocol FBP, that is based on the digital fountain concept. By using FBP, whenever a file has to be transferred, we use an appropriate digital fountain encoder (such as those described above) that continuously generates encoded packets. These packets are injected into the network, by the sender, at a given rate. On its turn, when the receiver has enough packets to reconstruct the source data, it sends a `stop` message to the sender which, on the reception of such a message, stops injecting new packets. That is, the FBP does not implement any kind of congestion control mechanism. Furthermore, it does not make use of packet retransmissions. The only "overhead" are the packets injected in the time interval since the receiver sends the `stop` message until the sender receives it.

For simplicity, in the following sections we will assume that the decoding inefficiency of the used codes is 1 (i.e., we assume that ε is zero). Current implementations of digital fountain codes can guarantee an inefficiency of about 1.054 [2] and even less than that [7, 15] (up to 1.02). We will also assume that the rate at which senders inject packets is constant (i.e., it is CBR).

equilibrium means that we are in a stable state in the presence of selfish users.

IEEE COMPUTER SOCIETY

3. A model for the interaction between TCP and FBP

To understand the interaction between TCP and FBP we will use the traditional single-bottleneck problem [6], in which a communication line is shared between N different hosts. All the hosts try to communicate with a remote destination D, which is reachable though a common router. All communication lines are assumed to have the same capacity C and all hosts are assumed to be greedy, meaning that they always wish to send new packets to the destination. The router is assumed to have a finite buffer of size M. Then, when congestion occurs and the buffer is full, new incoming packets will be dropped.

3.1. The ordering protocol

The ordering protocol we use tries to emulate the two main characteristics of TCP: congestion control and resource sharing. However, and for the sake of simplicity, we establish a set of assumptions. First, we assume that time is discrete, and we call a discrete time unit a *slot*. We consider that the capacity of all links is fixed to one packet per slot, and that all packets have the same size. We also assume time structured as a sequence of consecutive rounds, where each round is a groups of $S \geq N$ consecutive slots. The underlying ordering protocol assigns one fixed exclusive slot to each host within each round, in which the host is allocated to send packets. Then, when all hosts are fair, they transmit one packet per round, and this packet does not compete with any other to enter the router buffer, because the protocol guarantees that the slot is exclusively assigned to that host.

When congestion occurs (i.e. when the buffer is full), packets can be lost. For this reason, our simplified TCP ordering protocol needs to implement a basic timeout-and-retransmit mechanism. With this purpose, we introduce an acknowledgment scheme: whenever the destination receives a fair packet, it immediately generates an `ack`, which is sent back to the corresponding host in the subsequent time slot. (Note that `acks` can not suffer congestion; we also assume that they can not be lost.) If an `ack` for a given packet is not received after some time (a timeout), the packet is retransmited.

3.2. The disordered protocol

Regarding the disordered protocol, we consider that evil hosts use FBP and transmit on all slots of the round with a given probability p, which depends of the degree of selfishness of the evil host (a high value of p indicates that the host is very selfish because it is trying to monopolize the shared resource). For simplicity, we assume all evil hosts share the same value of p. Thus, evil hosts use some kind of digital fountain encoding which guarantees that duplicates are not possible and all packets reaching the destination are useful. As we presented previously, a single `stop` message is sent at the end of the whole file transfer to indicate the sender that the transmission has ended (i.e. it is not necessary to take into account any particular timeout-and-retransmit mechanism). As a first approximation, we consider that the size of the files being exchanged is very large (tends to infinite), and hence we disregard `stop` messages.

3.3. Analysis

From the above, our communication scheme is based on rounds of S slots, with two kind of slots. First, N_f fair slots (F-slots), where one fair host always transmits and N_e evil hosts transmit with probability p. Second, $N_e = S - N_f$ evil slots (E-slots) where N_e evil hosts transmit with probability p.

In the analysis of the rest of this section we assume that the system is congested, i.e., the router buffer is continously full. This is clearly not the case, in general. However, is has been shown in [6] that the probability of the buffer being full is very high even when there is only one evil host. Then, taking into account that, in the congested situation, only one packet can enter the buffer in each time slot (because there is only one free position: the one left by the packet being sent by the router in that slot), it is easy to prove that the probability of a given evil host with selfishness degree p to get a packet in the congested buffer in an E-slot is:

$$p_E^e(N_e, p) = \sum_{i=1}^{N_e} \frac{1}{i} \binom{N_e - 1}{i - 1} p^i (1-p)^{N_e - i}. \quad (1)$$

In the same way, the probability of an evil host to get a packet into the buffer in an F-slot is given by

$$p_F^e(N_e, p) = \sum_{i=1}^{N_e} \frac{1}{i+1} \binom{N_e - 1}{i - 1} p^i (1-p)^{N_e - i}. \quad (2)$$

Analogously, the probability of a fair host to get its packet into the buffer in its F-slot is

$$p_F^f(N_e, p) = \sum_{i=0}^{N_e} \frac{1}{i+1} \binom{N_e}{i} p^i (1-p)^{N_e - i}. \quad (3)$$

With these results, we can easily evaluate the transmission rate for evil hosts R_e. Since we assume evil hosts use FBP, then all packets arriving to the destination (all packets getting into the router buffer) are useful. Note that, in each round, we must take into account all the packets entering the buffer in F and E-slots. Then, it can be obtained that

$$R_e(N, N_e, p, M) = \frac{(S - N_f) p_E^e(N_e, p) + N_f p_F^e(N_e, p)}{S}. \quad (4)$$

IEEE COMPUTER SOCIETY

For fair hosts the result is similar. Although we do not use any particular encoding mechanism, the ordering protocol and the choice of S guarantees that all packets that arrive to the destination are useful (duplicates are not possible). So, the rate for fair hosts R_f is

$$R_f(N, N_e, p, M) = \frac{p_F^f(N_e, p)}{S}. \quad (5)$$

The most interesting conclusion we may obtain from this expressions comes when observing the two extreme situations: when all hosts are fair ($N_e = 0$) and when all hosts are evil ($N_e = N$). In the former case $R_f(N, 0, p, M) = \frac{1}{S}$, and in the latter $R_e(N, N, p, M) = \frac{1}{N} - \frac{(1-p)^N}{N}$. Observe that if $S = N$ then $R_f(N, 0, p, M) \geq R_e(N, N, p, M)$, which means that the best throughput obtained when all hosts use an ordered protocol controlling the congestion is over the one obtained when they try to access the common resource in an unordered way. Nevertheless, the interesting point is to remark that, if $p = 1$ then $R_f(N, 0, 1, M) = R_e(N, N, 1, M) = \frac{1}{N}$. That is, an optimal protocol controlling the congestion is just as good as letting all hosts to send their FBP packets as fast as possible trying to occupy the shared resource in a selfish manner and without any kind of control.

The key issue to understand why this happens is to observe that, when using FBP, all packets arriving to the destination are useful and duplicates are not possible. Many authors have remarked [1, 9, 6] that when using a timeout-and-retransmit based approach (as the one of TCP), if congestion and flow control algorithms are not respected by the hosts, the global throughput of the network drops due to the presence of duplicates, which are retransmitted when timeouts occur. Nevertheless, when using FBP, no duplicates are present and no timeouts are needed to ensure that the network is not collapsed by them.

Another interesting feature of the model, than can be easily proven using the results on [6], is that $R_e(N, N_e + 1, p, M) > R_f(N, N_e, p, M)$ for all $N_e \in \{0, ..., N-1\}$. That is, in any given situation, a fair host always has an incentive to become evil. This guarantees that the Nash equilibrium of the game is reached when all hosts are evil.

Before continuing, we wish to remark that, even though our model is not a totally realistic scenario where TCP and FBP could be competing, it is optimistic when estimating the fair (TCP based) yield, and pessimistic for the evaluation of the evil rates. The optimistic behavior occurs since our simplified ordering protocol does not react in any way when packets are lost, while current TCP implementations react to congestion by decreasing its offered load. In turn, the pessimistic behavior of FBP occurs since the decrease in the offered load of the TCP-based hosts would imply a higher probability for evil packets to get into the router buffer. This can be observed in the results of the simulations in the following section.

4. Simulations for the TCP-FBP interaction

The model we have just present allows understanding some key issues in the interaction between TCP and FBP. Nevertheless, the obtained throughput cannot be considered to be exact because of the simplifying assumptions we have made. For this reason, to gain a deeper insight into the TCP/FBP competition, we have carried out a number of simulations of the single-bottle-neck problem using a slightly modified version of the NS2 simulator.

In our simulations all communication lines have a fixed capacity of $C = 100$ Kbps, with delays of 1 ms and a router buffer of 10 packets. Fair hosts have been modeled using standard one-way TCP agents (modified to be able to obtain the throughput). FBP hosts have been implemented using modified UDP agents which simply compute the addition of all received bytes. In both cases we use the throughput (including headers) as the measurement of the information transmitted by each player. The traffic of the TCP hosts has been implemented using the usual FTP application of NS2 (which assumes that the file being transmitted is infinite). Fountain traffic has been implemented with CBR generators. The buffer management policy is drop-tail and the scheduling discipline is FIFO. All the simulation results presented in this paper have been averaged for 50 executions of the simulation scenario. Each execution has been run for a simulated time of 30000 seconds.

Note that it is possible to establish a direct parallelism between the TCP based hosts of the simulations and the fair hosts of the analytical model because both comply with a set of ordered rules which try of optimize the utilization of the shared resource avoiding congestion. In the same way, the evil hosts of the analytical model can be assimilated as the FBP (CBR-UDP) hosts of the simulations. In this case, the selfishness probability p can be easily calculated as the utilization of the corresponding line (the ratio between the offered load of the evil CBR source, λ_e, and the total capacity of the communication line C). For instance, since $C = 100$ Kbps, an evil hosts with $p = 0.5$ would correspond to an FBP agent using a CBR source of $\lambda_e = 50$ Kbps.

The results of the simulations, as well as the predictions of the simplified mathematical model presented above, have been depicted in Fig. 1. The first thing we can observe is that our observations about the analytical model are correct. That is, the theoretical curve for fair hosts is optimistic and it remains always over the real throughput of the TCP hosts, and the one of evil hosts is pessimistic and stays all the time under the real FBP results. Furthermore, we can see that the TCP (fair) rate when N_e hosts are evil (for any value of

IEEE COMPUTER SOCIETY

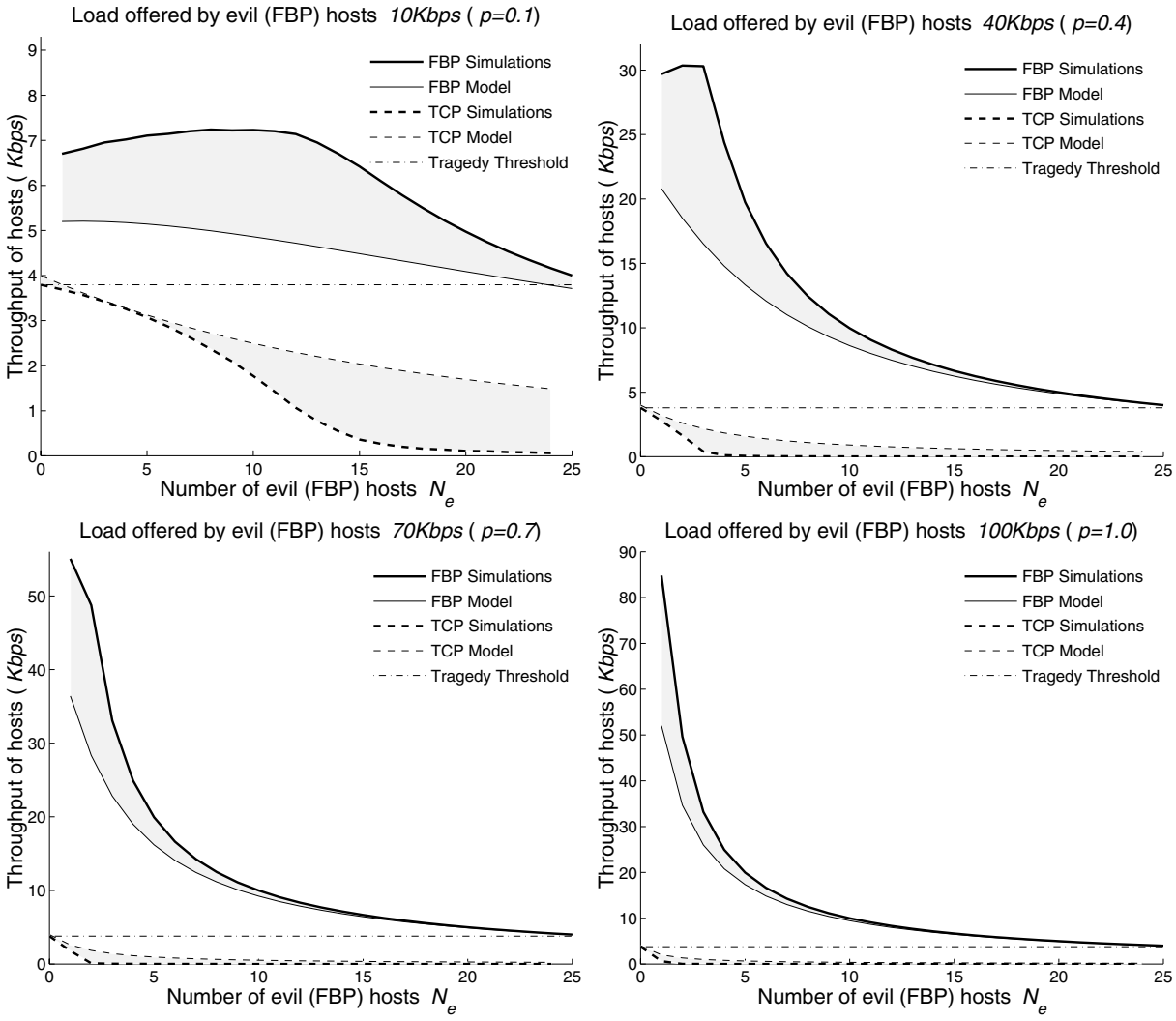

Figure 1. The figure represents the throughput of evil (FBP) and fair (TCP) hosts as a function of the number of evil hosts N_e for four different values of evil selfishness. The shaded region represents the error between the theoretical model and the simulations. The horizontal dashed line indicates the simultated throughput of the TCP hosts when no evil players are present. All pictures have been calculated for $N = S = 25$.

N_e) is always under the FBP (evil) rate when one more host becomesg evil. As we explained previously using the mathematical model, this means that fair hosts always have an incentive to become evil, because in any possible situation the most rational strategy is to use FBP.

Regarding the Nash equilibrium, we have again that it is reached when all hosts are evil ($N_e = N$). In such a case, in the simulations the throughput is always over the throughput obtained when all hosts comply with the TCP protocol. This feature can be observed more clearly in Fig. 2, where we have represented the simulated Nash equilibrium throughput and the simulated cooperative throughput for 10 different values of the load injected by the FBP hosts (10 different values of p). This means that, in this particular game, the selfish equilibrium is slightly more efficient than the global cooperation of TCP. Hence, we can claim that the Tragedy of the Commons is not present, at least under the assumptions we have accepted.

IEEE COMPUTER SOCIETY

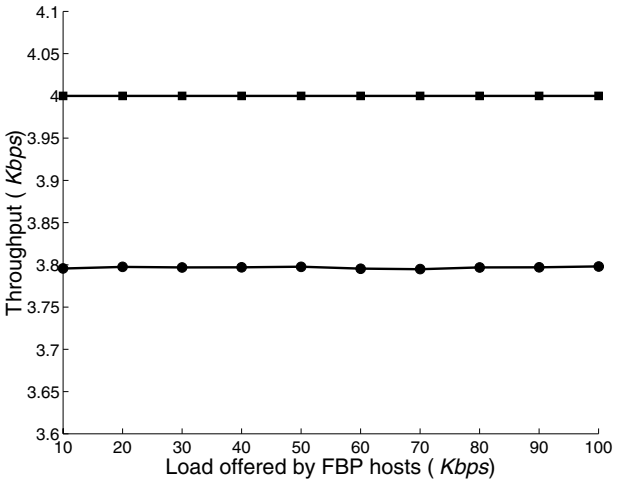

Figure 2. This figure represents the simulated FBP Nash equilibrium throughput (solid line with squares) and the simulated TCP cooperative throughput (solid line with circles) for 10 **different values of the load offered by FBP hosts. The simulation conditions are identical to the ones described in Fig. 1.**

5. Concluding Remarks

In this paper we have analyzed a novel paradigm of reliable communication which is not based on the traditional timeout-and-retransmit mechanism of TCP. Our approach, which we call FBP (Fountain Based Protocol), consists of using a fountain encoding which guarantees that all recived packets are useful. Using Game Theory, we analyze the behavior of TCP and FBP in the presence of congestion and show that two main characteristics arise. First, in this scenario, any given host using TCP has an incentive to switch to an FBP approach obtaining a higher throughput. This guarantees the Nash equilibrium to be reached when all hosts use FBP. Second, we show that, at this equilibrium, the performance of the network is similar (may be slightly over or slightly under) the performance obtained when all hosts comply with TCP.

Although these results seem promising, we wish to note that the FBP approach presents several aspects that should be taken into consideration. First, the analysis we have carried out has been done on the basis of large file transfers. However, this scenario can substantially change when considering other kind of communications requiring more interaction between the sender and the receiver. For example, several real time or multimedia applications (like Telnet) require small units to be continuously transfered. This scenario makes the FBP approach less practical, because, although duplicate packets cannot exists, it is possible that useless packets not containing additional information could flood the network before the appropriate stop message issued by the receiver arrives to the sender.

References

[1] A. Akella, S. Seshan, R. Karp, S. S. , and C. Papadimitriou. Selfish behavior and stability of the internet: A game-theoretic analysis of TCP. In *Proceedings of ACM SIGCOMM*, pages 117–132. ACM Press New York, 2002.

[2] J. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to the reliable distribution of bulk data. *IEEE Journal on Selected Areas in Communications*, 20:8:1528–1540, 2002.

[3] D. Fudenberg and J.Tirol. *Game Theory*. MIT Press, 1991.

[4] R. Garg, A. Kamra, and V. Khurana. A game-theoretic approach towards congestion control in communication networks. *Computer Communication Review*, 32(3):47–61, 2002.

[5] G. Hardin. The tragedy of the commons. *Science*, 162:1243–1248, 1968.

[6] L. López, G. Rey, A. Fernández, and S. Paquelet. A mathematical model for the TCP tragedy of the commons. *Theoretical Computer Science*, 2005 in press.

[7] M. Luby. LT codes. In *43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[8] N. F. Maxemchuk. Dispersity routing. In *Proceedings of ICC*, pages 41–10, 41–13, San Francisco CA, 1975.

[9] J. Nagle. Congestion control in IP/TCP internetworks. *IETF Request for Comments 896*, 1984.

[10] J. Nagle. On packet switches with infinite storage. *IEEE Transactions on Communications*, 35(4):435–438, 1987.

[11] G. Owen. *Game Theory*. Academic Press, 1995.

[12] R. Pan, B. Prabhakar, and K. Psounis. CHOKE, A stateless active queue management scheme for approximating fair bandwidth allocation. In *Proceedings of the IEEE INFOCOM*, pages 942–951. IEEE Los Alamitos, 2000.

[13] M. O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the Association for Computing Machinery*, 36(2):335–348, 1989.

[14] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[15] A. Shokrollahi. Raptor codes. *Unpublished manuscript*, 2003.

IEEE COMPUTER SOCIETY