Improving network performance with Prioritized Dispersal[†]

Yitzhak Birk and Noam Bloch[‡]
Electrical Engineering Department
Technion — Israel Inst. of Technology, Haifa 32000, Israel
birk@ee.technion.ac.il, noam@mellanox.co.il

Abstract— **Redundant traffic dispersal exploits the topological redundancy of networks and improves load balancing by replicating each message or partitioning it into several "data" packets and generating several "redundant" ones; all are then sent over different paths to the destination. The redundancy overcomes the "weakest link" problem, but increases the load. This paper introduces "prioritized dispersal", whereby "redundant" packets receive lower priority than the "data" ones. Moreover, the use of non-FCFS queuing policies for the redundant packets leads to the timely arrival of at least a fraction of them even under heavy load. Queuing-theoretic analysis shows the new schemes to substantially outperform non-prioritized ones in terms of both the blocking probability and that of delay exceeding a specified limit. One possible use of prioritized dispersal, which is discussed in this paper, is to improve the quality of service for best-effort traffic in ATM networks with multiple paths between nodes. Another is in conjunction with ad hoc path trunking. Additional likely uses include parallel access to mirrored data sites and reliable multicast.**

Keywords— **information dispersal; prioritized dispersal; selective exploitation of redundancy; QoS; ATM; ad hoc trunking.**

I. Introduction

TOPOLOGICAL redundancy, namely multiple paths from a source to a destination, permits fault-tolerance as well as performance enhancements. For example, a source and a destination can combine multiple paths connecting them to form an ad hoc trunk, thereby increasing the throughput between them or reducing latency. This paper explores performance enhancements attained through the concurrent use of multiple paths (as opposed to conventional routing, which selects among paths), focusing on the intra-message granularity. Its main contribution is the selective exploitation of redundant data in order to minimize the overload brought about by the redundant traffic while retaining the benefits of redundancy.

A. Dispersity routing

In [1], Maxemchuk proposed *dispersity routing*, whereby a message is partitioned into several (m) packets, which are then sent to the destination over different paths in a store-and-forward network (see Fig. 1). Dispersal assists in balancing the load among network links. Also, multiple packets can be transmitted and propagated concurrently, thereby reducing transmission time. However, all packets must be received before the message can be reconstructed. This may increase the blocking probability, and the delay of successful messages is determined by the slowest path. Finally, as depicted in Fig. 1, any intra-message dispersal involves *dispersal overhead* in the form of header replication.

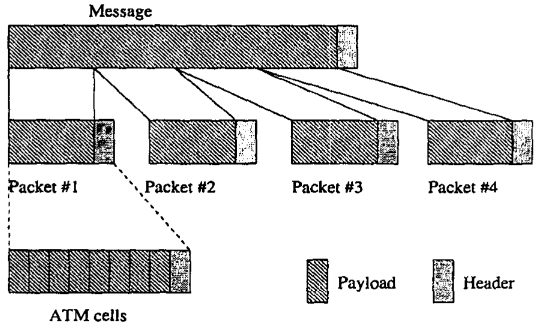

Fig. 1. Dispersal of a message comprising a header and a payload into 4 packets. The message is dispersed into packets in the application level. Later, each packet may be split into ATM cells.

To overcome the "slowest path" sensitivity of dispersity routing, Maxemchuk also suggested *redundant dispersity routing*: $m + r$ packets are derived from the m original ones using "maximum distance separable" (MDS) error correcting codes, such that reception of any m packets suffices for reconstruction of the original message. (Replication is considered a special case of redundant dispersity.) Various error-correcting codes can be used for this purpose, e.g., [2][3][4].

Traffic dispersal has received increasing attention over the years. Rabin proposed the *Information Dispersal Algorithm (IDA)* [5] for communication inside parallel computers. More recently, focus has been on high speed communication networks such as ATM. For a recent literature survey of traffic dispersal, see [6]. In [7][8][9], the applicability of non-redundant traffic dispersal to ATM networks is discussed. In [10], it is proposed to expedite the connection-setup phase by pursuing it along multiple paths (replication); subsequently, the resources of all but one path are released. [11],[12] and [13] suggested to chop a traffic stream into strings (packets) of consecutive ATM cells, and to distribute the strings over parallel links in a round robin or random manner. With their schemes, a guide cell is added to each packet (see the lower part of Fig. 1). Packet length represents a trade-off between the quality of load balancing and dispersal overhead. [14][15] apply redundancy to such a scheme, and discuss application to different service classes of ATM under light and heavy load. [16] suggests to use traffic dispersal for fault tolerance. Redundant dispersal has also been suggested for parallel access to multiple networked data servers [17][18].

[†] This research was supported in part by the High-Bandwidth Communications Consortium, administered by the office of the Chief Scientist of the Israeli Ministry of Industry and Commerce, and by the Ollendorff Research Center at the Technion.

[‡] Presently with Mellanox Technologies, Inc.

0-7803-5880-5/00/$10.00 (c) 2000 IEEE

In [19], Maxemchuk suggested to use dispersity routing for the transmission of very large medical images over virtual-circuit networks: bandwidth is reserved over multiple paths before transmitting the image, and is released once transmission ends. He noted that there is no apparent performance reduction due to the inclusion of redundancy as long as all users behave as expected. Moreover, when unexpected imbalance occurs, redundant dispersal outperforms the non-redundant dispersal since it can tolerate occasional loaded paths. When applied to ATM networks with small bursts and on-the-fly bandwidth reservation [20], whereby bandwidth may be wasted by unsuccessful transmissions, Maxemchuk reports a substantial performance reduction in a balanced situation due to the inclusion of redundancy.

The use of dispersal, redundant or not, thus appears to be a mixed blessing. In order to mitigate the shortcomings of non-redundant dispersal, Maxemchuk suggested that non-uniform load be handled by adaptive source routing. The main contribution of this paper is *Prioritized Dispersal* (PD), which mitigates the negative effects of the extra load brought about by redundant dispersal while preserving its benefits. One advantage of this approach over adaptive source routing is that it does not require the source to know the dynamic network state.

B. Selective exploitation of redundancy

In [21][22], which discuss the use of redundancy for performance enhancement of video servers, an important distinction is made between the cost of including redundancy and the cost of exploiting it. For example, when a parity block is stored for every four blocks of data, the storage overhead is only 25 percent; however, when one of the blocks is requested and the redundancy is exploited in order to avoid reading the requested block, four blocks must be read, resulting in a four-fold increase in disk accesses! This has lead to the idea of *selective exploitation of redundancy*. In this paper, we apply this idea to networks.

In communication networks, the bulk of the cost of exploitation is incurred within the network in the form of extra load, which results in higher packet delays and/or loss probabilities due to buffer overflow. Unlike centralized storage systems, in which system state can be known to a central controller, the inherently distributed nature of high-speed communication networks and their rapidly-changing state limit the ability of source nodes to make intelligent decisions.

Our focus in this paper is on ways of permitting some automatic adaptation of the level of redundancy-exploitation within the network based on its dynamic state, even after data has been transmitted. We propose *"prioritized dispersal"* (PD) schemes, whereby low priorities are assigned to the redundant packets, as a novel improvement over conventional redundant-dispersal schemes: the overload caused by the redundant packets is mitigated, and packet losses are more uniformly distributed among messages. (Uniform packet-loss distribution among messages is best when the goal is to make a good situation very good. This is our focus.) Using a model that resembles statistically-multiplexed ATM networks as an example, we show PD to substantially outperform non-prioritized schemes in the case of unexpected load imbalances. Even in balanced situations, there is almost no penalty due to the inclusion of the prioritized redundancy. (In some cases there is even substantial benefit.)

The work of [23] and [24] is related to our work. They considered ATM networks with inter-cell Forward Error Correction (FEC) to reduce packet loss, and suggested to assign low priority to redundant cells using a threshold priority mechanism. Because they do not use spatial dispersal, however, the fates of cells of any given packet are highly correlated. Such correlations have been shown to dramatically reduce the beneficial effect of FEC [25]. Also, dispersal does not contribute to delay reduction when a single path is used for all packets of a given message.

The main resource considered in this paper is communication bandwidth; the computation required for generation of redundant information and for reconstruction of the original information will be ignored. The main performance measures considered are blocking probability and that of exceeding a specified delay.

The remainder of the paper is organized as follows. In section II, we present a family of prioritized-dispersal schemes. In section III, we provide an approximate queuing model and assess its closeness through simulation. Sections IV and V present derivations of the distribution of the delay and the blocking probability, respectively, for a prioritized-dispersal scheme under the queuing model. Section VI presents numerical results and a comparison among schemes. Section VII extends the discussion to multi-hop paths, Section VIII discusses the applicability of PD to ATM networks, and section IX offers concluding remarks.

II. Prioritized-dispersal schemes

The novelty of PD is that "redundant" packets are assigned lower priorities than the original ones. For the performance measures discussed in this paper, it makes little or no difference which packets are labeled as "redundant". We therefore conveniently speak of m original packets, accompanied by r redundant ones.

In designing a prioritized dispersal scheme, one must make several decisions, which span a family of prioritized-dispersal schemes:

Degree of splitting and redundancy (m,r). The degree of splitting presents a trade-off between the quality of load balancing and header overhead. With a perfectly-preemptive priority discipline, redundancy cannot cause any harm, but in practice it may. Also, the total number of packets per message is limited by the topological redundancy. In this paper, we experiment with several sensible values to assess the approach.

The priority assigned to each packet. We assign high priority to the m "original" packets of any given message, and low priority to its r redundant packets. The rationale is that this helps in spreading packet losses and long delays more uniformly among messages. Also, by assigning a low priority to all the redundant packets, we prevent them from interfering with the "original" traffic. Nonetheless, this policy is not necessarily optimal and warrants further research, including consideration of more than two priority levels.

Priority discipline. This determines whether the service of a low-priority packet is preempted by the arrival of a high-priority packet. We consider *preemptive-resume* (PR) at the packet level, which closely corresponds to ATM networks with a non-preemptive policy at the cell level, as well as *non-preemptive* (NP), which represents the case of "atomic" packets.

Queuing discipline
This refers to the order of service to same-priority packets. It is reasonable to assume that, prior to the addition of redundancy, the quality of service for high-priority packets is generally good, and the redundancy is aimed primarily at solving occasional problems in order to achieve a very high QoS at the message level. With a preemptive priority discipline, this is not altered by the introduction of additional, low-priority traffic. It therefore makes sense to strive for a small variance in the delay of high-priority packets, leading to the use of a "first come, first served" (FCFS) queuing discipline for those. ("Earliest deadline first" would also make sense.)

For low-priority packets, the situation is different: even when the arrival rate of the original packets does not exceed the service rate, applying redundancy may cause overload. When the system is overloaded, the number of redundant (and thus lower-priority) packets increases without bound, and at least a fraction of them never get served. With an FCFS queuing discipline, the expected delay for all low-priority packets is infinite in such a case. In this situation, it is beneficial to increase the variance of the quality of service seen by these packets, thereby increasing the (temporal) relevance of the packets that do get served. This can be achieved by non-FCFS queuing disciplines.

A non-FCFS queuing discipline may be based on rejection as well as on order-changing. Rejection mechanisms entail pushing out (discarding) the oldest packets when the buffer is full, or timing out (discarding) packets whose waiting time exceeds a predefined threshold even if the buffer is not full. Order-changing mechanisms entail serving newly generated packets before older ones. If packets are queued only once along the path to the destination, this is simply a "Last-Come First-Served" (LCFS) policy. With multi-hop paths, however, LCFS may give rise to "oscillatory" phenomena: a packet that overtakes an earlier one in a queue may be overtaken by it in a subsequent queue. Instead, "Last-Generated First-Served" or "Latest deadline first" can be used. Rejection mechanisms should best exploit knowledge regarding the sensitivity of users to delay in order to decide which packets can be considered old. Order changing mechanisms, in contrast, do not need such knowledge. For a discussion of non-prioritized schemes for overload control, see [26]. In this paper, we only consider order-changing mechanisms.

III. QUEUING MODEL

In this section, we present a simple queuing model for statistically multiplexed networks. The model captures the key elements of dispersal in general, and prioritized dispersal in particular. It is subsequently used for a comparison among a variety of schemes.

The $(m+r)$ disjoint paths used by an (m,r) dispersal scheme for sending data from a given source to a given destination are represented by $(m+r)$ parallel queues. The $(m+r)$ packets jointly making up a given message are assigned randomly to these queues. Each queue is shared among multiple, possibly different, (source, destination) pairs; therefore, there is substantial interfering cross traffic, which is not the same for the different queues. In view of this, we model the states of the queues as i.i.d.; with this approximation, it suffices to analyze a single queue. Fig. 2 depicts the queuing model for $m=3$, $r=2$. This model corresponds most closely to two-hop (single-queue) paths. It is extended to longer paths and studied via simulation in Section VII.

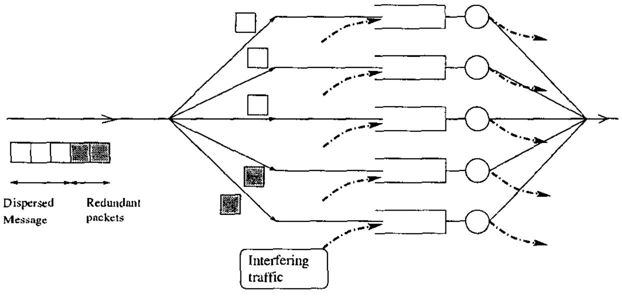

Fig. 2. Approximate queuing model for a $(3,2)$ dispersal system. Each message is dispersed into $m=3$ packets, and $r=2$ redundant packets are constructed. The 5 packets are randomly allocated to 5 independent queues with equally distributed service times. Dotted arcs denote the cross traffic through the queues.

The order of service in the queues is FCFS for the high priority packets; for low-priority packets, FCFS and LCFS are analyzed and compared. Both PR and NP priority disciplines (among packets with different priorities) are considered. In the derivation of delay for a preemptive-resume policy, such a policy is also applied among low priority packets. In blocking-probability analysis, low priority packets do not preempt one another. This is done mainly for facility of analysis, and simulations show that it has a very minor effect on the results.

The arrival process of H-P packets ("original" packets in a dispersal-routing system) to any single queue is assumed to be Poisson with rate λ_h, creating a load of $\rho_h \equiv \frac{\lambda_h}{\mu}$, where $\frac{1}{\mu}$ is the mean service time of a packet, including both payload and header. The arrival process of L-P packets (representing redundant packets) is Poisson with rate λ_l, and $\rho_l \equiv \frac{\lambda_l}{\mu}$. Also, $\frac{\lambda_h}{\lambda_l} = \frac{m}{r}$. The aggregate offered load is $\rho \equiv \frac{\lambda}{\mu}$, where $\lambda \equiv \lambda_h + \lambda_l$. We also assume that the arrival processes are independent. The service time for each (low- or high-priority) packet is assumed to be independent from packet to packet and generally-distributed with Laplace transform $B^*(s)$.

The two performance measures studied in this paper are the blocking probability of a **message**, defined as the probability that at least $r+1$ of its packets are lost due to full queues, and its delay, defined as the time until at least m of its packets reach the destination. The two are studied for the case of finite and infinite queues, respectively. The analysis incorporates the dispersal header overhead, and the case of unexpected load imbalances is considered in addition to that of balanced load.

Evaluation of the independence approximation

In practice, the arrival processes to the $m+r$ queues representing the paths used by a given message are correlated. However, if different subsets of those paths are shared among many different (source, destination) pairs, only a small fraction of the traffic through different queues is correlated. Consequently, we conjecture that performance with the independence assumption closely approximates the performance of dispersal routing systems in which each path is shared among many (source, destination) pairs. [27] considered a somewhat analogous system comprising a very large number of queues and a single source of traffic. It was assumed that messages are split, upon generation, into packets that are sent to $m+r$ randomly selected queues. For infinitely large systems, the queues were shown to be independent M/M/1 queues. We have confirmed through simulations that our model also closely approximates more realistic situations, especially when it comes to the relative performance of different schemes. (In the simulations, queue state is tracked and the packets comprising any given message are equisized.) Therefore, we present numerical results derived from analysis of the approximate model.

Since, with our model, all queues are independent and statistically identical, it suffices to analyze a single queue. In the following sections, we consider a single priority queue with FCFS service for H-P packets and both FCFS and LCFS for L-P packets. Both PR and NP priority disciplines are considered. In section IV, we derive the delay distribution for an unlimited-capacity priority queue. In section V, we derive the blocking probabilities for a limited-capacity priority queue with generally distributed service time. A detailed version of these derivations appears in [28].

IV. Delay Distribution

In this section, we sketch the derivation of the delay distribution. A more detailed derivation appears in [28].

Let $D_h(t)$ and $D_l(t)$ be the delay distributions of high- and low-priority packets, respectively. Let $D(t)$ be the delay distribution of a dispersed message. Finally, let $D_h^*(s)$, $D_l^*(s)$, and $D^*(s)$ be the respective Laplace transforms. With the independence assumption, the $m+r$ queues accessed by the $m+r$ packets of a dispersed message are i.i.d. Therefore, the dispersed message delay is equal to the m^{th} out of $m+r$ order statistics from a parent population equal to the distribution of a single queue. Accordingly,

$$D(t) \equiv Pr(delay \leq t) =$$
$$\sum_{i_m=(m-r)^+}^{m} \left(\binom{m}{i_m} (D_h(t))^{i_m} (1-D_h(t))^{m-i_m} \cdot \sum_{i_r=m-i_m}^{r} \binom{r}{i_r} (D_l(t))^{i_r} (1-D_h(t))^{r-i_r} \right). \qquad (1)$$

In the following subsections, we derive $D_h(t)$ and $D_l(t)$ for both PR and NP priority disciplines. Substituting them in (1) yields the message delay distribution $D(t)$.

A. Delay distribution with preemptive-resume discipline

The delay distribution of the high priority packets is exactly the same as for a non-priority M/G/1 queue to which only high priority packets arrive. Its Laplace transform is

$$D_h^*(s) = B^*(s) \frac{s(1-\rho_h)}{s - \lambda_h + \lambda_h B^*(s)},$$

where $B^*(s)$ is the Laplace transform of the packet service time [29].

The delay distribution of an L-P subtask is the same as the distribution of busy period duration in an M/G/1 queue with arrival rate λ, in which there is exceptional first service ($B_f^*(s)$). (For more details and for the derivation of the delay with FCFS, see [28].) Therefore,

$$D_l^*(s) = B_f^*(s + \lambda - \lambda Y_0^*(s)), \qquad (2)$$

where

$$Y_0^*(s) = B^*(s + \lambda - \lambda Y_0^*(s)) \qquad (3)$$

and

$$B_f^*(s) = \frac{s(1-\rho_h)}{s - \lambda_h + \lambda_h B^*(s)} \cdot B^*(s). \qquad (4)$$

(For exponentially distributed service time, there is an explicit expression for Y_0^* [29].)

B. Delay distribution with non-preemptive discipline

For L-P packets in a non-preemptive system, we use the results of [30]. For H-P packets in a non-preemptive system with $\rho = (\rho_h + \rho_l) < 1$, we use the results of [31], section 4.6.1. For $\rho_h + \rho_l \geq 1$, these results do not hold and require modification. We base the modification on the observation that for $\rho \geq 1$, the queue never becomes empty. Accordingly, whenever an H-P packet arrives to a system with no H-P packets, it finds an L-P packet in service. Consequently, from the point of view of the H-P packets, the system can be modeled as a queue with arrival rate λ_h and with a setup time at the beginning of busy periods [31]. The setup time is the residual service time of the L-P packet in service, whose distribution is

$$SU^*(s) = \frac{1 - B^*(s)}{s/\mu}.$$

Alternatively, we can use the observation that, as long as $\rho \geq 1$ and for fixed λ_h, the value of λ_l has no effect on the sojourn time of the H-P packets. Consequently, we can obtain very accurate numerical results using the equations of [31] by changing λ_l to

$$\lambda_l = \mu - \lambda_h - \varepsilon, \quad \varepsilon << 1$$

without changing λ_h.

V. Blocking Probability

For the blocking probability of H-P and L-P packets with preemptive resume priority discipline, we quote the results of [32]. For non-preemptive discipline, we derive the blocking probability of H-P packets from the aggregate blocking probability and the blocking probability of L-P packets. The derivation of the latter is more challenging. Van Doremalen [33] presented a recursion and an explicit formula for the calculation of the blocking probabilities with exponentially distributed service. We present a different recursion, and use it for a generally-distributed service time system. The complete derivation of the blocking probabilities of messages is omitted for brevity. For the detailed derivation, see [28].

Throughout the analysis, single-packet buffers are taken as atomic units. In other words, if a packet occupies any portion of a buffer, that buffer cannot be used by any other packet. We also assume that each buffer is sufficiently large to hold a maximum-length packet.

Remark. When comparing schemes, buffer size will be measured in terms of full messages, reflecting the fact that packet size is an artifact of the degree of splitting, and buffer size will be denoted by K. Nonetheless, buffer space is allocated in single-packet units.

Blocking probabilities for L-P packets with non preemptive discipline

The orders of service and discarding within each class do not affect the blocking probabilities. Therefore, without loss of generality, we conveniently assume FCFS (and "last come first discard") within each class. Thus, a queued low-priority packet is not affected by later arrivals of L-P packets.

We next calculate recursively the blocking probability of an L-P packet that sees, upon arrival, k (high or low priority) packets in the queue. Using the probabilities for k packets in the queue, we derive the blocking probability of an arbitrary packet. Let $P_{B_l}(k)$ be the blocking probability of a tagged low priority packet that sees, upon arrival, k packets (high or low priority) in the queue.

Clearly,

$$P_{B_l}(K) = 1, \quad P_{B_l}(0) = 0.$$

The time from the arrival of the tagged L-P packet until the next departure is the residual service time of the head-of-line packet. If $j < (K-k)$ H-P packets arrive during this time period, the tagged L-P packet will be at the $k+j-1$ position at the end of this period. (j arrivals and one departure.) Note that now, the time until the following departure is a full service time. Let $\hat{P}_{B_l}(i)$ be the blocking probability of an L-P packet that is in the $(i+1)_{th}$ position (there are i packets in front of it) when the service of the packet at the head of the queue is just beginning.

Clearly,

$$\hat{P}_{B_l}(K) = 1, \quad \hat{P}_{B_l}(0) = 0.$$

Let $P^R(j)$ be the probability of j arrivals of H-P packets during the residual service time of the head of the line packet, and $P^B(j)$ — the probability for j arrivals of H-P packets during a full service time. Then, for $1 \leq i \leq K-1$,

$$\begin{aligned} P_{B_l}(K-i) &= \qquad (5)\\ &= \sum_{j=0}^{i-1} P^R(j) \cdot \hat{P}_{B_l}(K-i-1+j) + \sum_{j=i}^{\infty} P^R(j) \cdot 1 \\ &= 1 - \sum_{j=0}^{i-1} P^R(j) \left(1 - \hat{P}_{B_l}(K-i-1+j)\right), \end{aligned}$$

where, for $1 \leq i \leq K-1$,

$$\begin{aligned} \hat{P}_{B_l}(K-i) &= \qquad (6)\\ &= \sum_{j=0}^{i-1} P^B(j) \cdot \hat{P}_{B_l}(K-i-1+j) + \sum_{j=i}^{\infty} P^B(j) \cdot 1 \\ &= 1 - \sum_{j=0}^{i-1} P^B(j) \left(1 - \hat{P}_{B_l}(K-i-1+j)\right). \end{aligned}$$

The first term of (5) is for all cases in which $j < i$ H-P packets arrived between the arrival of the tagged L-P packet and the departure of the packet in service. In all those cases, the tagged L-P packet is not discarded during this period; rather, it is pushed back $j-1$ positions in the queue. If $j \geq i$ packets arrived during the residual service time, the tagged packet is discarded (2nd term of (5)).

The explanation of (6) is almost the same. The only difference is that for $\hat{P}_{B_l}(K-i)$, the time until the departure is the full service time of the packet at the head of the queue instead of the residual service time.

$P^B(j)$ ($P^R(j)$) can be evaluated as the inverse Z-transform of the Laplace transform of the (residual) service time evaluated at point $s = \lambda_h(1-z)$ [29]:

$$P^B(j) = Z^{-1}\{B^*(\lambda_h(1-z))\}; \qquad (7)$$

$$P^R(j) = Z^{-1}\left\{\frac{1-B^*(\lambda_h(1-z))}{\lambda_h(1-z)B^{*'}(0)}\right\}. \qquad (8)$$

The steady state probabilities for k (high and low priority) packets in the queue, P_k, are the same as for a limited capacity non-priority queue with arrival rate λ. Those probabilities can be calculated using the scaling relation of Keilson and Servi. See [32] for details.

Finally, the blocking probability of an L-P packet is

$$P_{B_l} = \sum_{k=0}^{K} P_k P_{B_l}(k). \qquad (9)$$

VI. Numerical Results and Comparison

In this section, we numerically compare the performance of several schemes: redundant dispersal with and without priority, non-redundant dispersal, and non-dispersal systems. To do so, we use the model that was presented earlier and the analytical results that were derived for it. Specifically, we check the ability of different schemes to tolerate unexpected load in one of the paths and to mitigate the dispersal overhead. Note that interfering cross traffic is implicitly reflected by the total load. In fact, it is assumed to exist as part of the justification for the independence assumption.

We consider a communication network with 5 paths between any given (source, destination) pair. Messages are generated at the source according to a Poisson process with rate λ. When dispersal overhead is considered, it is assumed that each message originally consists of 96% data and 4% header. Messages are dispersed into $m \leq 5$ packets, and $r \leq (5-m)$ redundant packets are constructed, each with its own header. This is referred to as an (m,r) scheme. Prioritized dispersal schemes are denoted by (m,r)PR-LCFS, (m,r)NP-LCFS (m,r)PR-FCFS or (m,r)NP-FCFS, depending on the priority discipline and the order of service for L-P packets. (For brevity, we also use (m,r)PR, (m,r)NP to refer to the LCFS case.) With $(4,1)$ and $(4,0)$ schemes, for example, the header constitutes 16% of each packet. When considering unexpected imbalances, the load on one of the paths is assumed to be heavier by 0.2 than the load on the other paths. The transmission (service) time of packets (including data and header) is assumed to be exponentially distributed in all cases (this is done mainly for the simplicity of the computations); however, based on simulations, the relative results are similar for fixed packet lengths. The mean transmission time of an undispersed message (including the header) is assumed to be $\frac{1}{\mu} = 1$. Consequently, the mean service time of a packet with, for example, $(4,1)$ or $(4,0)$ dispersal , including the header, is 0.28.

The performance measures are blocking probability (of the message) with a given buffer capacity, and the probability of exceeding a given (overall message) delay. Those are derived for various "net" loads, defined as the load of non-dispersed messages $(\frac{\lambda}{\mu})$. The former is derived for a buffer capacity of 3 messages (it is assumed that $3m$ packets can be buffered in each queue) and the latter — for a delay threshold of 5 times the mean transmission time of an undispersed message. (Header overhead in buffers has been neglected for facility of analysis.) Results for other parameter values lead to the same conclusions.

We first compare the performance of PD with different levels of splitting. Then we compare its performance with those of non-prioritized schemes.

Fig. 3 depicts the performance of PD-PR for different levels of dispersal as well as for $(4,1)$-NP while considering dispersal overhead. It shows that there is only a small difference between $(4,1)$PR and $(4,1)$NP for those performance measures. This also holds for other buffer capacities and delay-threshold values, but the relevant figures are omitted for brevity. As was the case with non-prioritized dispersal, dispersing messages into a sufficiently large number of packets ($(4,1)$PD in this example) yields the best performance at light loads. For a heavy load, however, the header-overhead caused by dispersal dominates and $(2,3)$PR exhibits the highest performance. Nonetheless, we use only $(4,1)$PR for comparison with non-prioritized schemes.

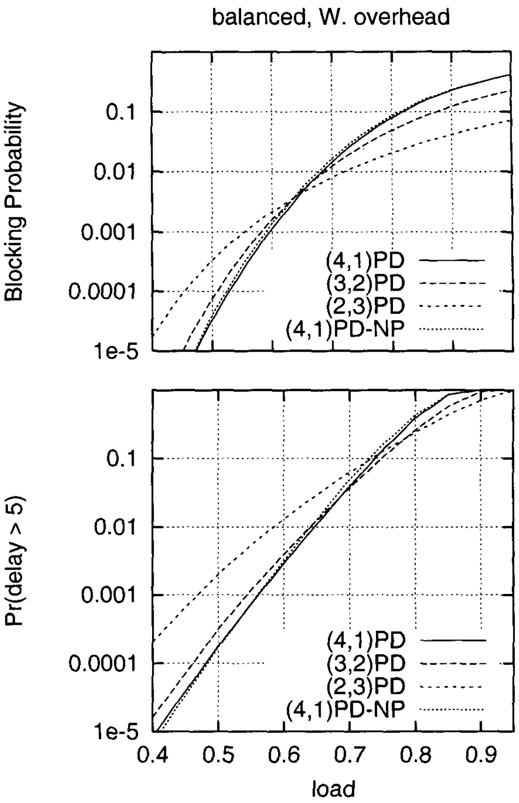

Fig. 3. Prioritized dispersal. The probability that a dispersed message's delay exceeds 5 full-message transmission times (left) and the blocking probability with 3-full-message buffers (right) for different values of (m,r).

Fig. 4 and 5 compare the performance of the prioritized-dispersal scheme with those of non-prioritized schemes. The following systems are compared: $(1,0)$ non-dispersal; $(5,0)$ non-redundant dispersal; $(4,1)$ redundant dispersal without a priority mechanism, and $(4,1)$PD. When dispersal overhead is ignored and all paths are equally loaded, the probability for the delay to exceed the threshold with prioritized dispersal (with LCFS for L-P packets) is close to that of the $(5,0)$ non-redundant dispersal. With FCFS, the probability for the delay to exceed the threshold is slightly higher with PD. The blocking probability with PD is lower than with non-prioritized schemes.

When considering dispersal overhead and/or unexpected imbalance, prioritized dispersal (either with LCFS or FCFS for L-P packets) outperforms non-prioritized schemes across the entire load range in terms of both delay and blocking. With dispersal overhead, when the load equals 0.7, for example, the probability for the delay to exceed the threshold is 3.4 times smaller

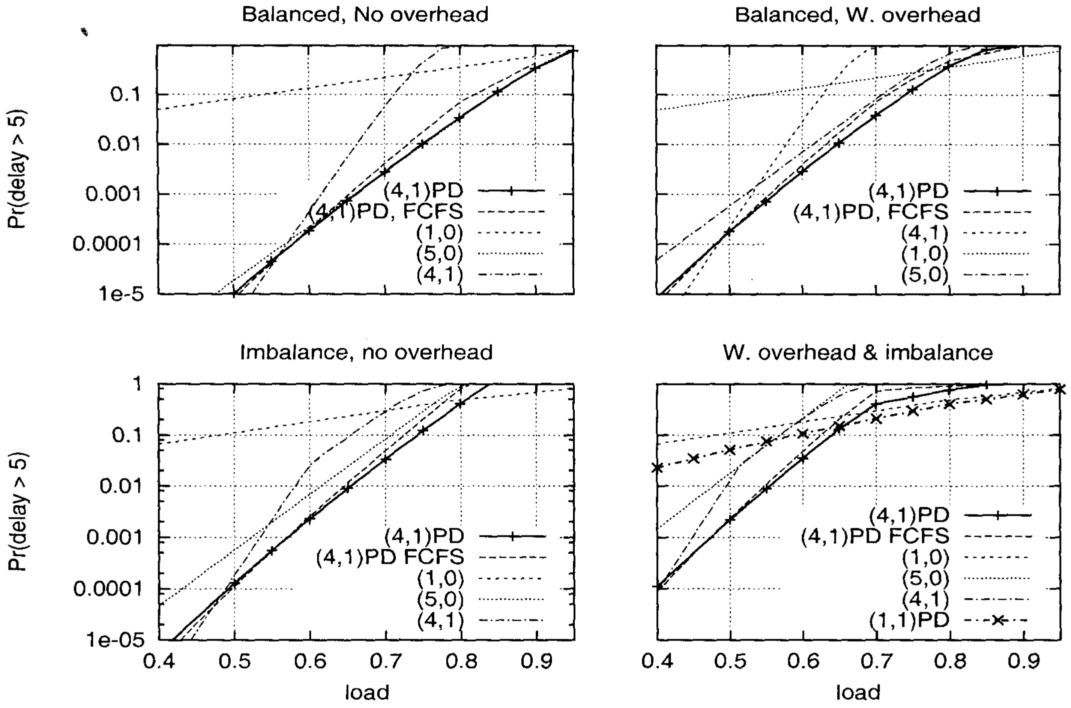

Fig. 4. The probability that a message's delay exceeds 5 full-message service times.

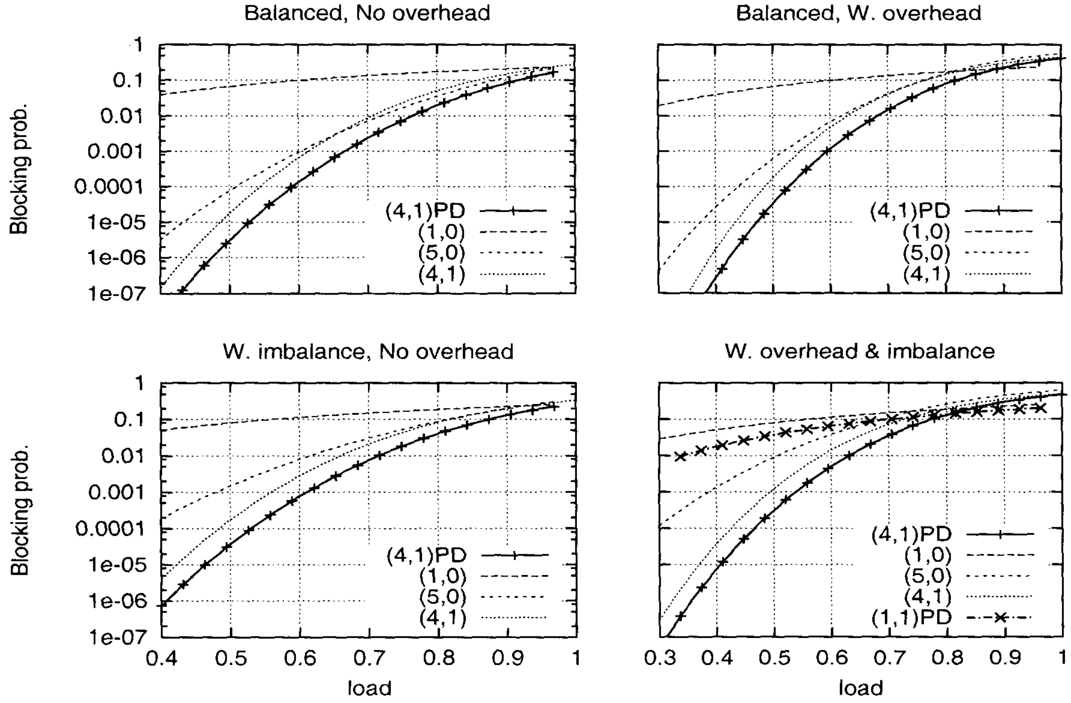

Fig. 5. Message blocking probability with a 3-full-message-buffer ($3m$ packets).

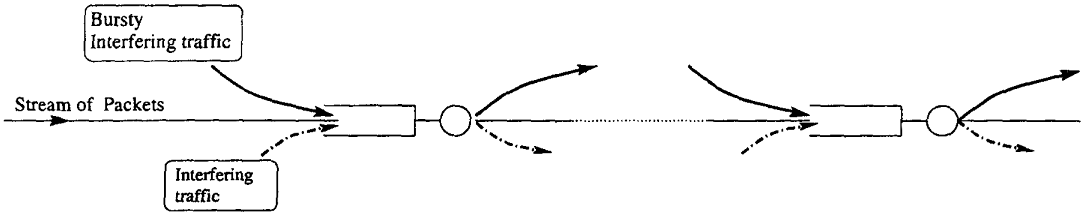

Fig. 6. Queuing model for one path of the $m+r$ used by a message. Dotted arcs denote the cross-traffic through the queues. Bold arcs denote interfering bursty traffic

with $(4,1)$-PR-LCFS and 2.4 times smaller with $(4,1)$PR-FCFS than with $(5,0)$, which is the best non-prioritized scheme at this load. Blocking probability at this load is 2.8 times smaller with $(4,1)$PD than with $(5,0)$. Although a $(1,0)$ non-dispersal scheme may outperform maximum dispersal schemes ($(5,0),(4,1)$PD) at heavy load in the presence of dispersal overhead and unexpected imbalances, it is easy to see that any $(1,r)$PD scheme outperforms it. This is due to the fact that redundant L-P packets do not interfere with the H-P ones and there is no dispersal overhead for $(1,r)$, which is simply replication.

We repeat the comparison among those schemes for a fixed load, for various threshold levels and buffer capacities. This comparison shows that superiority of PD over non-prioritized schemes is sustained for different delay thresholds and buffer capacities.

PD outperforms non-prioritized schemes across most of the load range. However, at very low load and, in some cases, with a very high delay threshold, the probability for the delay to exceed the threshold with $(4,1)$ non prioritized dispersal is smaller than with $(4,1)$PD. In these cases, even with the redundancy, the probability for timely arrivals of all packets is very high. Assigning low priority to redundant packets increases the probability for their delay to exceed the threshold while only slightly decreasing this probability for the original packets. For this reason, PD-NP may slightly outperform PD-PR at very low loads.

Prioritized dispersal exhibits a lower H-P dispersal (header) overhead than non-redundant schemes for an equal number of packets per message $(m+r)$. This is because the number of high-priority headers per message is only m. $(4,1)$PD also tolerates the loaded path better than $(5,0)$ non-redundant dispersal and mitigates the overload caused by the redundant packets better than $(4,1)$ redundant dispersal.

VII. Extension to multiple-hop paths

We consider the following model (see Fig. 6): each path goes through of several nodes, each modeled as a queue. The traffic arriving to each queue comprises Poisson traffic from the source to the destination as well as interfering Poisson cross traffic and bursts of traffic that arrive from time to time. The traffic originating from the tagged source is a fraction α of the aggregate Poisson traffic. The number of packets in a burst is geometrically distributed with mean 50. The service time of each such packet is exponentially distributed with mean 0.25 (as the service time of a packet with $(4,1)$ schemes). The time between packets in a burst is exponentially distributed with mean 0.5. The time between bursts of cross traffic arriving to a given node is exponentially distributed with mean 1000. The arrival processes of bursts to nodes are independent among nodes. Analysis of such a system appears extremely difficult, so we resort to simulation. (In the simulation, packet lengths are iid. However, the results with equisized packets of any given message are nearly identical.)

Fig. 7 compares the probability of delay exceeding a given threshold with various schemes for single- and 3-queue paths as a function of load. In all cases, Prioritized Dispersal outperforms the non-prioritized schemes.

Fig. 8 depicts a similar comparison for a fixed load as a function of the delay threshold T. For very small values of T ($T=1,2$), $(5,0)$ slightly outperforms $(4,1)$PD. For reasonable values of T, PD substantially outperforms the non-prioritized schemes.

VIII. Applying Prioritized-Dispersal to ATM Networks

In ATM networks, communication is connection oriented. In order to effectively exploit network resources, traffic of different connections is usually *statistically multiplexed* onto common links. However, transmission of a large burst may be carried out in a non-multiplexed manner by using dynamic bandwidth reservation: resources for the transmission of the burst can be reserved for the duration of the burst transmission and then released [19][20]. The applicability of non-prioritized dispersal to ATM networks in the application level was discussed in detail in [16]. We next explore ways of applying PD to both the case of statistical multiplexing and that of dynamic bandwidth reservation. Our analysis in this paper, however, is only applicable to statistical multiplexing.

Statistical multiplexing.

Prioritized dispersal can be applied as follows: a source sets up $m+r$ (preferably) path-disjoint connections to the destination; m of them are "high priority connections" and are assigned a high service class (e.g., VBR), whereas the remaining r connections are "low priority" connections and are assigned a low

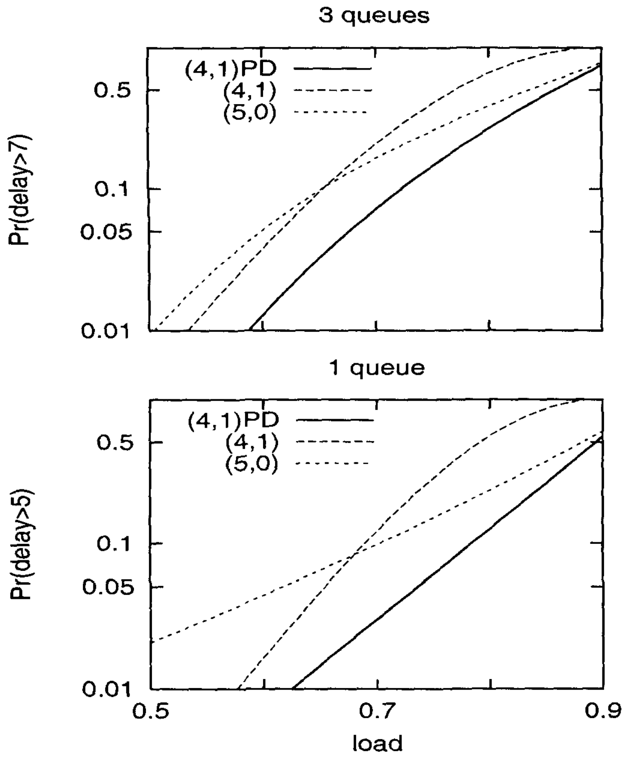

Fig. 7. The probability that a dispersed message's delay exceeds 5 and 7 full-message transmission times with 1- and 3-queue paths, respectively.

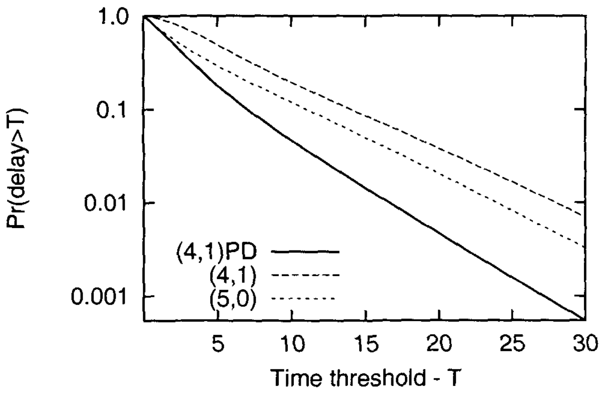

Fig. 8. The probability that a dispersed message's delay exceeds a threshold of T full-message transmission times with 3-queue paths. $\rho = 0.7$.

service class (e.g., ABR). Whenever a source has a message to transmit, it disperses it (at the application level) into m packets, encodes them into $m+r$ packets, and sends them through the $m+r$ connections. The first m to arrive are decoded by the destination. The priority discipline would be implemented automatically by ATM: cells of H-P packets are served before those of L-P packets in a non-preemptive manner. This non-preemptive priority discipline at the cell level closely corresponds to preemptive resume at the packet level. The approximation is close if a packet contains a sufficient number of cells.

An alternative way to assign low priorities to redundant packets, on cell-by-cell basis, is by using the Cell Loss Priority (CLP) bit in the cell header. The queuing and priority disciplines commonly associated with the use of CLP (e.g. [34]) are different from those discussed in this paper because they permit some processing of low-priority cells even in the presence of high-priority ones. The analysis of PD with such priority disciplines is left for future research.

Implementation of an order-changing queuing policy among L-P packets may require some effort, but existing ATM mechanisms for early packet discarding [35] that capture the notion of packets serve as a good basis. These, as well as the ability to randomly access cells, have been implemented in products, e.g. the MMC 2000SATM chip set [36], albeit for other purposes.

Dynamic bandwidth reservation.
Prioritized dispersal can also be applied to per-burst bandwidth reservation: bandwidth for transmitting a sub-burst is reserved before transmission or on-the-fly. Priority may be applied to such systems through "threshold" or "push-out" disciplines. With a "threshold" priority discipline, a bandwidth reservation request for an L-P , redundant subburst would be denied if more than some threshold fraction of the bandwidth of the path is already reserved (even though there may be sufficient bandwidth for the L-P subburst). Such a scheme is simple and easy to implement. With a "push-out" priority discipline, bandwidth reservation requests are accepted whenever there is sufficient available bandwidth. However, high-priority (H-P) requests may cut off ongoing L-P subburst transmissions if they need the bandwidth. This scheme seems to be harder for implementation but is expected to yield better performance. Analysis of prioritized per-burst bandwidth reservation is left for future research.

Finally, we note that our purpose in this section was not to address all the details of implementing the proposed schemes in ATM. Rather, it was merely to show that there is no fundamental contradiction between the use of these schemes and ATM.

IX. Conclusions

In this paper, we presented and analyzed the "Prioritized Dispersal" family of schemes, along with matching queuing policies, for selective exploitation of redundancy in multi-path networks. Numerical results for exponentially-distributed as well as for fixed packet lengths show that PD outperforms non-prioritized schemes. This is due to the combination of dispersal, redundancy and the ability to prevent redundant traffic from overloading the network. The advantage is most pronounced in the face of unexpected load imbalances.

Prioritized dispersal has numerous applications. For example, it can be used in sending requests to multiple candidate servers: one copy of any given request would be assigned high priority while additional ones would be assigned low priority.

Another possible application is redundant, distributed storage systems: the request for the parity block would be sent in all cases but with a low priority.

The distinction made with prioritized dispersal between redundant and non-redundant packets moreover enables us to combine prioritized dispersal with "Join the shortest queue" allocation schemes [37], in which knowledge of the state of system resources is used to select the least loaded one. In the absence of perfect knowledge, one could send additional low-priority requests to other resources. Such *JSQ-PD* schemes are explored in [38].

In summary, prioritized dispersal appears to be an attractive technique for improving performance of distributed, redundant-resource systems, and serves as yet another example of the merit of selective exploitation of redundancy. It may find use in numerous applications, both in isolation and in conjunction with other approaches, warranting further investigation of its possibilities and merits.

Acknowledgments. The authors would like to thank Moshe Sidi for providing the code that was used to compute the inverse Laplace transform. Also, they are grateful to Scientific and Engineering Software Inc. for providing the Workbench simulation package that was used in obtaining the simulation results.

REFERENCES

[1] N. Maxemchuk, "Dispersity routing," in *Proc Int. Commun. Conf.*, pp. 41.10–41.13, 1975.

[2] A. McAuley, "Reliable broadband communications using a burst erasure correcting code," in *ACM SIGCOMM'90*, pp. 297–306, 1990.

[3] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures," *IEEE Trans. on Computers*, vol. 44, pp. 192–202, Feb. 1995.

[4] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proc. 29th ACM Symposium on Theory of Computing*, 1997.

[5] M. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *J. ACM*, vol. 36, pp 335–3488, April 1989.

[6] E. Gustafsson and G. Karlson, "A literature survey on traffic dispersion," *IEEE Network*, vol. 11, pp. 28–36, March/April 1997.

[7] R. Krishnan and J. Silvester, "Choice of allocation granularity in multipath source routing schemes," in *Proc. IEEE INFOCOM'93*, vol. 1, pp. 322–29, Mar. 1993.

[8] H Suzuki and F. Tobagi, "Fast bandwidth reservation scheme with multilink & multipath routing in ATM networks," in *Proc. IEEE INFOCOM'92*, vol. 3, pp. 2233–40, May 1992.

[9] T.-H. Cheng, "Bandwidth allocation in B-ISDN," *Comp Networks and ISDN Sys.*, vol. 26, no 9, pp. 1129–42, 1994.

[10] I Cidon, R. Rom, and Y. Shavitt, "Multi-path routing combined with resource reservation," in *Proc. IEEE INFOCOM'97*, 1997.

[11] J. Dejean, L Dittmann, and C. Lorenzen, "Performance inprovement of an ATM network by introducing string mode," in *Proc IEEE INFOCOM'91*, vol. 3, pp. 1394–1402, April 1991.

[12] J. Dejean, L. Dittmann, and C. Lorenzez, "String mode - a new concept for performance improvement of ATM networks," *IEEE JSAC*, vol. 9, pp. 1452–60, December 1991.

[13] A. Jagd, S. Myken, C. Phillips, O Theologou, J. Giamniadakis, and G Migliarina, "Implementation of string mode: a multi-link broadband network," in *Proc. 2nd Int'l Conf. Broadband Services, Sys. and Networks*, Nov 1993.

[14] T. Lee and S. Liew, "Parallel communications for ATM network control and management," in *Proc. IEEE GLOBECOM'93*, pp 442–46, Dec. 1993.

[15] Q.-L. Ding and S. Liew, "A performance analysis of a parallel communication scheme for ATM networks," in *Proc. IEEE GLOBECOM'95*, vol 2, pp. 898–902, Nov. 1995.

[16] A. Banerjea, "On the use of dispersity routing for fault tolerant real-time channels," *European Transactions on Telecommunications*, vol. 8, pp. 393–407, July/August 1997.

[17] Q. Malluhi and W. Johnston, "Coding for high avilability of a distributed-parallel storage system," *IEEE Trans. Parallel and Distributed Systems*, vol. 9, pp. 1237–1252, Dec. 1998.

[18] J. Byers, M. Luby, and M. Mitzenmacher, "Accessing multiple mirror sites in parallel: Using tornado codes to speed up downloads," tech. rep., The International Computer Science Institute (ICSI), 1998.

[19] N. Maxemchuk, "Dispersity routing in high speed networks," *Comp. Networks and ISDN Sys.*, vol. 25, no. 6, pp. 641–61, 1993.

[20] N. Maxemchuk, "Dispersity routing on ATM networks," in *Proc. IEEE INFOCOM'93*, vol. 1, pp. 347–57, Mar 1993.

[21] Y. Birk, "Method and apparatus for supplying data streams." U.S. Patent No. 5,592,612, 1995.

[22] Y. Birk, "Random RAIDs with selective exploitation of redundancy for high performance video servers," in *NOSSDAV'97, St. Louis, MO*, May 1997.

[23] M. Murata, Y. Kikuchi, and H. Miyahara, "Performance of high speed data transfer using a FEC method in a multimedia network environment," *Trans Inst. of Electronics, Info. and Commun. Engineers B I.*, vol. J78B-I, pp. 325–334, Aug. 1995.

[24] J. Wu and C. Peng, "Simulation study of prioritized forward error correction in ATM networks," *Computer Communications*, 1997.

[25] I. Cidon, A. Khamisy, and M. Sidi, "Analysis of packet loss processes in high speed networks," *IEEE Transactions on Information Theory*, vol. IT-39, pp. 98–108, Jan. 1993.

[26] B. Doshi and H. Heffes, "Overload performance of several processor queueing disciplines for the M/M/1 queue," *IEEE Trans. on Commun.*, vol. COM-34, pp. 538–546, June 1986.

[27] N. Vvedenskaya, "Large queueing systems where messages are trasmitted via several routes," *Problems of information trasmission*, vol. 34, no. 2, pp 180–189, 1998.

[28] Y. Birk and N. Bloch, "Prioritized dispersal. improving network performance through selective exploitation of redundancy," tech. rep., Technion - Israel Institute of Technology, 1999.

[29] L. Kleinrock, *Queueing Systems*, vol. 1. Wiley, 1976.

[30] B. Doshi and E. Lipper, "The throughput performance of a prioritized LIFO service discipline," *Operations Research Letters*, vol. 3, pp. 75–80, June 1984.

[31] J. Daigle, *Queueing Theory for Telecommunications* Addison–Welsey, 1992.

[32] J. Keilson and L. Servi, "The M/G/1/K blocking formula and its generalizations to state dependent vacation systems and priority systems," *Queueing Systems*, vol. 14, pp. 111–123, 1993.

[33] J. V. Doremalen, "A note on "analysis of a finite capacity nonpreemptive queue," *Computers and Operations Research*, vol. 13, no. 4, pp. 525–526, 1986.

[34] S. Suri, d. Tipper, and G. Meempat, "A comparative evaluation of space priority strategies in ATM networks," in *Proc. IEEE INFOCOM'94*, pp. 516–523, 1994.

[35] A. Romanow and S. Floyd, "Dynamics of TCP traffic over ATM networks," *IEEE JSAC*, vol. 13, pp. 633–641, May 1995.

[36] WWW URL: http://www.mmcnet com.

[37] F. Haight, "Two queues in parallel," *Biometrika*, no 45, pp. 401–410, 1958.

[38] Y. Birk and N. Bloch, "Mitigating the effects of uncertainty in join-the-shortest-queue resource allocation schemes through prioritized dispersal," in *Proc. 36th Allerton Conf. on Commun., Control and Comp* , Sep. 1998.