Bandwidth problems in high-speed networks

by I. A. Saroit Ismail

High-speed networks are capable of carrying many types of services such as voice, data, images, and video. These services have different requirements in terms of bandwidth, cell loss, delay, etc. The goal is to maximize the quality of service offered during periods of stress, as viewed by both the network provider and the customer. Many problems are created by these different requirements. This paper illustrates four bandwidth problems in high-speed networks, then describes several solutions to them. The first problem is topology design and bandwidth allocation, and it is concerned with the ability to dynamically reconfigure a network in order to efficiently benefit from network resources. The second problem is concerned with flow control and congestion avoidance. Bandwidth management (BWM) protocols are used to prevent congestion, essentially by accepting or refusing a new-arrival cell. The third problem, which is the most critical one, is bandwidth allocation, which is concerned with successful integration of link capacities through the different types of services. Given that a virtual path is a logical direct link, composed of a number of virtual circuits, between any two nodes, the last problem is concerned with how to assign bandwidth to each virtual path in the network, in order to optimize performance for all users. This paper may be a good guide to researchers concerned with high-speed networks in general.

1. Introduction

Service integration has been one of the most researched areas in communication networks, and it is known that high-speed networks such as the asynchronous transfer mode (ATM) network offer a powerful method to integrate network services. These networks must carry traffic with widely varying characteristics such as voice, data, image, and video. They must be designed to efficiently manage a wide range of information bit rates and various types of traffic of differing static nature.

This paper illustrates four bandwidth problems and several solutions to them. The first deals with topology design and bandwidth allocation, the second with bandwidth management and congestion. The third is another bandwidth allocation problem, and the last considers virtual-path setup.

The topology design and bandwidth allocation problem is concerned with the ability to dynamically reconfigure a network in order to obtain best performance. In this paper, we are not concerned with the configuration of the transparent circuit-switched network, but with a packet-switched network built on top of the facility network. The

0018-8646/00/$5.00 © 2000 IBM

aim is to exploit the flexibility of the digital cross-connect system to obtain more efficient design and operation of the packet-switched network [1–5].

Bandwidth management (BWM) protocols address congestion control. Congestion occurs when various sources compete for network resources, and these resources cannot handle the demand. This may happen when logical channels request bandwidth that cannot be supported, or when the network admits more calls than the links can handle, or at any node due to buffer shortage. BWM protocols prevent congestion essentially by accepting or refusing a new-arrival call. These protocols have five objectives: protection of shared network resources, fairness, robustness, simplicity, and flexibility. This paper illustrates several bandwidth management methods that can be used to avoid congestion based on call admission or refusal [6–23].

Broadband high-speed networks (BISDN) are expected to support a wide variety of services, so various quality-of-service (QOS) levels and bandwidths are needed. Accordingly, efficient bandwidth allocation methods must be implemented to meet the needed QOS while maintaining high utilization of the bandwidth. Several bandwidth allocation methods exist, such as simple channel strategy [24], slot assignment methods [25, 26], virtual finishing-time methods [27–30], integrated access and cross-connected systems [10, 31], resource allocation analysis programs [32–34], multilevel control methods [35, 36] and others [37–56]. Expert systems, neural networks, and fuzzy logic are also used to deal with the bandwidth allocation problem [48, 49, 57].

The virtual path is a logical direct link between two nodes; it includes a number of virtual circuits. Each virtual path has a bandwidth that defines the maximum number of virtual circuits it can carry. Virtual paths are set up to optimize performance for all users. This paper presents three methods for setting up the virtual paths [7, 52, 58, 59].

The rest of the paper is divided into five sections, each concerned with one problem, and these are divided into many subsections, each illustrating a solution. Section 2 presents topology design and bandwidth allocation; Section 3 discusses bandwidth management and congestion. Section 4 is concerned with another bandwidth allocation problem. Section 5 presents the virtual-path set-up problem. Finally, Section 6 is a summary of the paper.

2. Topology design and bandwidth allocation

In emerging network technologies designed to support a variety of services, it is common to find that the packet-switching service is implemented on top of a facility network (as in narrowband ISDN and some private networks). The ability to reconfigure a customer network dynamically is a well-known advantage of digital cross-connect systems.

In broadband high-speed networks, switches are connected with multiple digital cross-connect system (DCS) trunks. The ability to reconfigure the network dynamically is very desirable in order to benefit efficiently from network resources, especially during variable flow.

Most previous studies have been based on networks that provide circuit-switched channels of various rates between pairs of user sites. In this section we are concerned with packet-switched networks. The aim is to exploit the flexibility of DCS to obtain more efficient design and operation of the embedded network [1–5].

The backbone topology comprises a number of switches and trunks, while the embedded topology comprises a set of pipes, each having a specific capacity. The sum of the capacities of the embedded links multiplexed on the backbone must not exceed the capacity of the trunk itself [1]. **Figure 1(a)** shows a backbone network topology. Several embedded topologies can be derived from this topology; two examples are shown in **Figure 1(b)**.

We have assumed that the backbone facility network has already been defined (number and locations of switches, fiber trunks, etc.). This facility will be partitioned among many services, public and private, operational and experimental (telephony, video distribution, private P/S nets, ISDN, BISDN, etc.).

To design the embedded-network topology, the problem is formulated as a network optimization problem in which a congestion measure based on the average packet delay is minimized subject to capacity constraints.

Assuming that the number and locations of switches and trunks in the backbone network are already defined, we address the problem of mapping the embedded topology. This mapping may have to be revised very frequently in order to overcome the traffic fluctuation.

Two types of reconfiguration procedures exist. The first one may involve a major change in embedded-network topology and may not be transparent to users (i.e., some connections may be interrupted and later resumed); this type of reconfiguration may take place with a frequency of months or weeks. The second one involves only minor perturbations in the embedded topology, and should be user-transparent; it could be carried out on an hourly basis. The proposed methodology is the same for both types.

- *Problem definition*

Suppose that an initial embedded topology is defined, and the embedded link bandwidth and packet routing are chosen such that

- The average packet delay is minimized.
- The sum of the aggregate bandwidth of all pipes using the trunk does not exceed the capacity of this trunk.

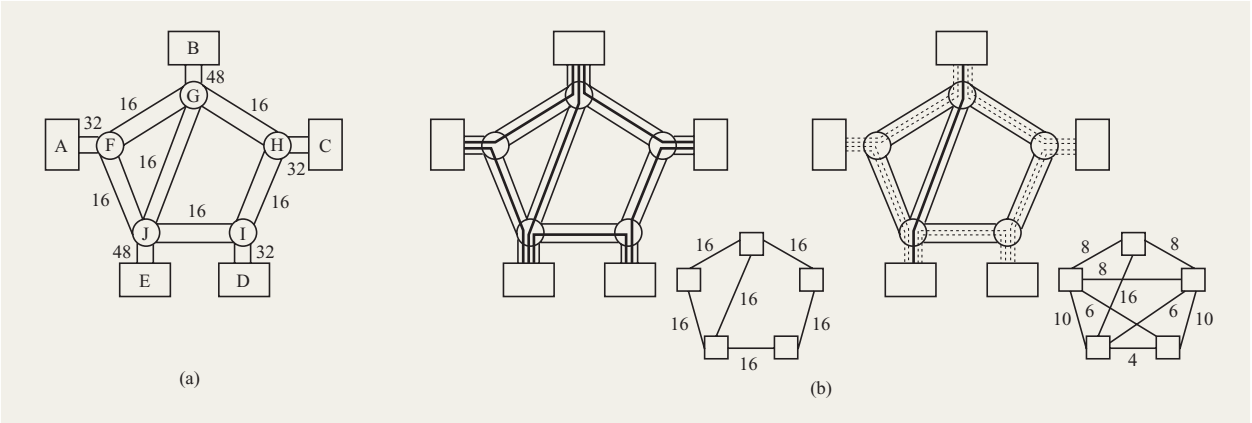

Figure 1

(a) Backbone topology for a packet-switched network; (b) two possible embedded topologies.

- All traffic requirements are satisfied without exceeding the pipe bandwidth.

If both backbone and embedded networks are modeled using graphs, C_i is the capacity of arc i, f_i is the flow of arc i, commodity com(i, j) is the flow of packets from source i destined for j, X_i is the flow of each commodity i, and Z is the average packet delay, the problem can be formulated as follows.

Given:

- Topology and arc capacities of the links of the backbone network.
- Topology of the embedded topology. (Note that each arc in the embedded topology contains a number of pipes.)
- Offered traffic.

Objective:

- Minimize the average packet delay Z.

Variables:

- Capacities of the arcs of the embedded topology C.
- Routing on the embedded topology.

Constraints:

1. The aggregate capacity $\bar{C}$ of all arcs in the embedded topology must not exceed the capacity of the backbone topology arcs C,

$$\sum_{i=1}^{\bar{M}} \bar{C}_i P_i \le C, \tag{1}$$

where $\bar{M}$ is the number of arcs in the embedded topology and P is the arc path vector (0s or 1s).

2. The flow $\bar{x}$ of each commodity k carried on all of the pipes must be equal to the flow r_k of commodity k offered to its source,

$$\sum_{n=1}^{N_k} \bar{x}(k; n) = r_k \qquad \forall\, k, \tag{2}$$

where N is the number of pipes and n is the pipe number.

3. The aggregate flow $\bar{f}$ on each arc in the embedded topology must be equal to the sum of all commodities flowing on all of the pipes using this arc,

$$\sum_{k=1}^{Q} \sum_{n=1}^{N_k} \bar{x}(k; n)\bar{p}(k; n) = \bar{f}, \tag{3}$$

where $\bar{Q}$ is the number of commodities and $\bar{p}(k; n)$ is the flow commodity vector (0s or 1s).

4. The aggregate flow on each embedded arc must not exceed its capacity,

$$\bar{f} \le \bar{C}. \tag{4}$$

5. $(a)\ \bar{C} \ge 0,$ (5)

$(b)\ \bar{f} \ge 0, \qquad \bar{x} \ge 0.$ (6)

- *Algorithm*

First, a random initial embedded topology is found. One choice is to include all possible $M(M - 1)$ links (fully connected graph), and then let an optimization procedure pick the most effective ones. Another choice is to insert direct links between node pairs with the highest throughput

requirements [1]. Remember that C_i is the capacity of arc i, f_i is the flow of arc i, commodity com(i, j) is the flow of packets from source i destined for j, X_i is the flow of each commodity i, and Z is the average packet delay.

The problem can be formulated as follows.

Given:

- Topology and arc capacities of the backbone topology.
- Arc vectors of the embedded topology (0 or 1).
- Traffic requirements.
- Initial feasible solution $(\bar{C}^0, \bar{f}^0)$.
- Tolerance t.

Objective:

- Find vectors (C, f, X) that minimize the average packet delay Z (suppose M/M/1 model) [60]:

$$Z = \frac{1}{\lambda} \sum_{m=1}^{\bar{M}} \frac{\bar{f}_m}{\bar{C}_m - \bar{f}_m}, \tag{7}$$

where $\bar{C}_m$ is the value of the capacity of link m, $\bar{f}_m$ is the aggregate flow on link m, and λ is the overall flow.

Steps:

1. Set the iteration number k to zero, and delay to ∞ $(k = 0, Z^{\text{old}} = \infty)$.
2. Increment the iteration number k by 1 $(k = k + 1)$.
   Find the vector $C^{\#}$ satisfying constraints 1 and 5 and minimize: min $\nabla_{\bar{c}} Z(\bar{C}^{k-1}, \bar{f}^{k-1}) \cdot \bar{C}$. This may be done using the revised simplex method [61].
   Find the vector $f^{\#}$ satisfying constraints 2 to 6 and minimize: min $\nabla_{\bar{f}} Z(\bar{C}^{k-1}, \bar{f}^{k-1}) \cdot \bar{f}$. This may be done by constructing a minimum path solution $(\bar{f}^{\#}, \bar{x}^{\#})$ for the embedded topology where the cost of each arc u is calculated as
   $$\frac{\partial Z(\bar{C}^{k-1}, \bar{f}^{k-1})}{\partial \bar{f}_u}. \tag{8}$$
3. Find the value α^* that minimizes the delay
   $$Z[\alpha(\bar{C}^{\#}, \bar{f}^{\#}) + (1 - \alpha)(\bar{C}^{k-1}, \bar{f}^{k-1})]. \tag{9}$$
   This optimum may be obtained by any linear search method, such as the golden section technique [62].
4. Set $(\bar{C}^k, \bar{f}^k) = \alpha^*(\bar{C}^{\#}, \bar{f}^{\#}) + (1 - \alpha^*)(\bar{C}^{k-1}, \bar{f}^{k-1})$.
   If $Z(\bar{C}^{k-1}, \bar{f}^{k-1}) - Z(\bar{C}^k, \bar{f}^k) \le t$,
   then go to step 5 (a potential local minimum has been found);
   else go to step 2.
5. If $Z^{\text{old}} - Z(\bar{C}^k, \bar{f}^k) \ge t$,
   then go to step 6;
   else if $Z^{\text{old}} \le Z(\bar{C}^k, \bar{f}^k)$,
   then set $(\bar{C}^*, \bar{f}^*) = (\bar{C}^{\text{old}}, \bar{f}^{\text{old}})$;
   else set $(\bar{C}^*, \bar{f}^*) = (\bar{C}^k, \bar{f}^k)$.
   Stop (the local minimum has been found).
6. Set $(\bar{C}^{\text{old}}, \bar{f}^{\text{old}}) = (\bar{C}^k, \bar{f}^k)$ and $Z^{\text{old}} = Z(\bar{C}^{\text{old}}, \bar{f}^{\text{old}})$.
   Using $\bar{C}^{k-1}$, find the vector $\bar{C}^a$ that solves
   $$\min Z = \frac{1}{\lambda} \sum_{i=1}^{\bar{M}} \frac{\bar{f}_i^{k-1}}{\bar{C}_i - \bar{f}_i^{k-1}}$$
   such that $\sum_{i=1}^{\bar{M}} \bar{C}_i P_i \le C$ and $\bar{C} \le \bar{f}^{k-1}$.
   Set $(\bar{C}^k, \bar{f}^k) = (\bar{C}^a, \bar{f}^{k-1})$.
   Go to 2.

Output:

The local minimum is $(\bar{C}^*, \bar{f}^*, \bar{x}^*)$, which satisfies all of the constraints.

If R is the ratio between the delay of the embedded topology and that of the original (backbone) topology, experimental results show that R decreases with the increase in traffic, implying that using an optimized embedded topology instead of the backbone topology leads to improvement in the average delay.

3. Bandwidth management

This section discusses network bandwidth management (BWM) and congestion control. A congestion-control mechanism must be designed according to the nature of the carried traffic, and must satisfy the wide variety of different requirements in terms of bandwidth, cell loss, number of connections, etc.

Bandwidth management, used to prevent congestion, is essentially done by accepting or refusing a new-arrival cell. These BWM protocols must be simple, distributed, and flexible, and must have three objectives: protection of shared resources, fairness, and robustness. Many BWM protocols exist [6–23, 31], and each of the following subsections represents one.

- *Leaky-bucket method*

In this method, the new traffic entering the network from each end device is monitored in real time. The algorithm uses the following steps [6–8]:

- Initialize a counter X to zero.
- If a new cell arrives,
  decrement X; $X = \max(X - cT, 0)$,
  where c is a specified parameter and T is the elapsed time since the last cell arrival.
  If $X \ge M$, where M is a prescribed value,
  then the cell is passed tagged;
  else the cell is passed untagged.
  Increment X; $X = X + \alpha L$,
  where α is a specified parameter and L is the cell length in octets.
- Tagged cells are discarded whenever they encounter a moderate threshold at any cell queue along the virtual circuit.

One can select α so that $\alpha L = 1$; then the throughput is c cells per unit time. M specifies the cell-burst size that is allowed following a substantial idle period.

This algorithm allows a sustained, untagged throughput rate of c/α octets per unit, and a burstiness that is controllable (for a given value of c and α) through selection of the value of M.

Let FVT represent the fraction of violated traffic and FPR represent the fraction of reduction in peakedness (ratio of non-violation-tagged peakedness to traffic output peakedness). τ is the traffic-shaping parameter (idle time between successive ATM cells), and M is the leaky-burst parameter.

Results show that both FVT and FPR decrease with an increase in M or τ, meaning that utilization also decreases with an increase in M or τ. In general, we can say that using the leaky-bucket method with good selection of M and τ reduces effective network utilization (for non-violated traffic), and that this traffic could be handled very easily by the network.

- *Peak-rate allocation*

In this method, the user specifies both the maximum rate of cell arrival and the network virtual circuits. As shown in **Figure 2**, each link has a peak-rate monitor in order to guarantee that the sum of the peak rates of all virtual circuits using that link does not exceed its maximum cell rate [9, 18]. Each virtual circuit (VC_i) must have the following parameters:

- Minimum time between cells, d_i.
- Time the most recent cell was transmitted, t_i.

The following steps are taken at any new cell arrival:

When a new cell arrives for VC_i at time T:

If $T - t_i < d_i$,
  then if at the link queue the total peak rate < link rate,
    then the cell enters the network;
    else one of two decisions may be taken:
      discard the cell, or
      mark the cell (marked cell can be discarded in case of congestion);
  else wait until time reaches $t_i + d_i$.

This method is easy for users to understand and specify, and it permits a straightforward implementation. Results prove that the peak-rate allocation method offers very strong performance in the case of uniform traffic, but in the case of burst traffic, it makes poor use of network bandwidth.

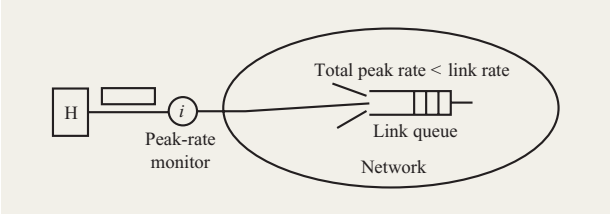

Figure 2

Peak-rate allocation.

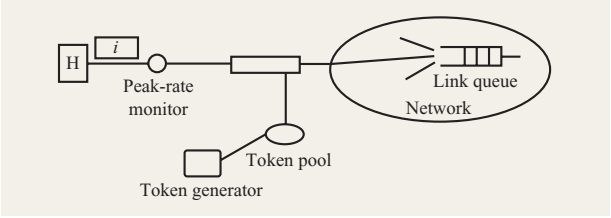

Figure 3

Peak-rate allocation with burst specification.

- *Bursty traffic specification and allocation*

In this approach, the user specifies a peak cell rate, an expected average cell rate, and a maximum burst size. As shown in **Figure 3**, each link has a peak-rate monitor and a token generator. The peak-rate monitor acts as in the previous method, while the token generator generates a number of tokens to be used by the cells passing from the peak-rate monitor. The maximum number of tokens in the pool is limited by the specified maximum burst size [9, 18].

The algorithm is divided into two parts; the first part is the same as the peak-rate method. However, if a cell passes from the peak-rate monitor, the second part takes place:

If there are any token(s) in the pool,
then one token is consumed, and the cell enters the network;
else the cell is marked and passes. (Marked cells are discarded and returned to the user if the network encounters congestion.)

The network must be able to decide whether a given set of virtual circuits can be safely multiplexed together according to the acceptable cell loss rate.

This method is more difficult for users to understand, specify, and implement than the previous one, but it allows performance guarantees to be traded off

against link efficiency and traffic burstiness. The main disadvantage of this approach is that there are currently no computationally effective ways to decide when a new virtual circuit can be safely multiplexed with other virtual circuits specified in this manner.

- *Minimum-throughput allocation*

In this method, the network allocates slots in the link buffers of the virtual circuits, in proportion to the bandwidth they require from this link. Virtual circuit routing ensures that the sum of the minimum required throughputs does not exceed the link bandwidth [9, 18].

During overload periods, each virtual circuit can use only its own slots, and any excess cells are discarded. During the unloaded periods, the following steps are taken:

When a new cell for virtual circuit VC_i arrives,
  if free slots exist for VC_i,
    then if the queue of the link is not full,
      then store the new cell unmarked
      else discard one marked cell in the link queue
        and store the new cell unmarked;
    else if there are no free slots for VC_i,
      then if the queue of the link is not full,
      then store the new cell marked (it may be
        discarded);
      else discard this new cell.

This method is easy to specify and implement. It can provide high efficiency in the case of uniform traffic, but uniformity is not always guaranteed, since the network cannot promise anything about throughput in excess of that requested. Users with bursty traffic streams, but needing all or almost all of their data to get through regardless of other traffic, can specify a high required throughput. However, this will lead to peak-rate allocation.

- *Fast buffer reservation*

In this method, each virtual circuit has one of two states, idle or active. Any active virtual circuit is assigned a number of buffer slots according to the bandwidth it requires (as in the preceding subsection). Transition between an active and an idle state occurs upon reception of a marked cell, denoting either the start or the end of a burst. A maximum time-out period exists; after this period the virtual circuit must change from the active to the idle state. This period is determined by the delay variation in the network. The choice of this period puts a bound on the peak rate of virtual circuits [9, 18].

Suppose that three types of cells exist: start S, middle M, and end E. Traffic may be of the form SSS . . . SSSE, SMMM . . . MMME, or SMM . . . MSM . . . MSM . . . E.

Any virtual circuit VC_i must provide the following information:

- Number of buffer slots for VC_i when active: B_i.
- Number of buffer slots already used by VC_i: b_i ($b_i \leq B_i$).
- State of virtual circuit VC_i (active or idle).

The number of unallocated buffer slots B in the queue link must also be known.

This algorithm takes the following steps:

- If a virtual circuit i (VC_i) receives a start cell:
  if VC_i is idle,
    then if $B - B_i < 0$,
      then discard the cell;
      else set VC_i state to active,
        set timer of VC_i,
        $B = B - B_i$,
        if $b_i = B_i$,
          then store the cell in the buffer marked;
          else $b_i = b_i + 1$ ($b_i < B_i$);
            store the cell in the buffer unmarked.
- If a virtual circuit i (VC_i) receives a start or middle cell:
  if VC_i is active,
    then queue the cell;
      reset the timer of i;
      if $b_i = B_i$,
        then store the cell in the buffer marked;
        else $b_i = b_i + 1$ ($b_i < B_i$);
          the cell is left unmarked.
- If a virtual circuit i (VC_i) receives a middle or end cell:
  if VC_i is idle,
    then discard the cell.
- If a virtual circuit i (VC_i) receives an end cell:
  if VC_i is active or its timer expires,
    then set VC_i state to idle;
      $B = B - B_i$.
- If a virtual circuit i (VC_i) receives a loner (low priority cells):
    then store the cell in the buffer marked.

Note that whenever a cell is removed from the buffer of the link, its appropriate b_i is decremented. It is clear that this method requires two bits of the ATM cell header to encode cell type (loner, start, middle, end).

The buffer reservation mechanism can be applied directly to constant-rate virtual circuits (uniform traffic), as well as to bursty virtual circuits (bursty traffic). Such virtual circuits would simply be activated initially and would remain active all the time. The software making the routing decisions must be chosen to ensure a very small probability that the instantaneous demand for buffer slots will exceed the buffer capacity.

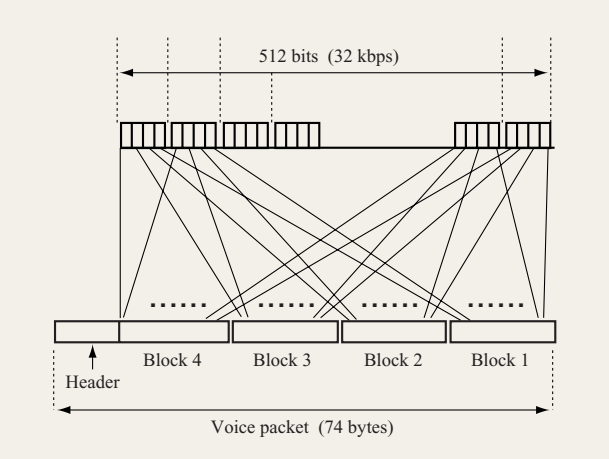

Figure 4

Encoded voice packet.

• *Congestion control by block dropping on voice*

This method considers only voice, voice-band data, and facsimile coding. It uses the concept that the loss of less significant bits in voice (compared to whole cells) causes negligible degradation of quality. This loss may reach 10% of the whole cell [10, 11].

Each voice source is sampled at 8 kHz and encoded using an embedded scheme at 32 kbps. Over 16 ms, 128 samples are collected and organized into four-block cells of equal length (note that all cells are also of equal length).

Each cell is divided into four blocks. All of the least significant bits (first quarter of the cell) from the 128 samples are contained in the first block of the cell, the next bits (next quarter) are contained in the second block, etc.

As shown in **Figure 4**, a header is used to incorporate a range of information about the packet (such as destination). One bit in the header may be designated to indicate whether or not a packet is "bit-droppable." The system must distinguish a voice-band data (VBD) cell from a voice cell, and can set this bit to prevent bit dropping on a VBD cell.

A congestion measure is defined in terms of the number of voice and data cells currently in wait state. Four thresholds are also defined. If the congestion measure exceeds the first threshold, the first block from the cell about to enter transmission is dropped. If the measure exceeds the second threshold, the second block from the cell about to enter transmission is dropped, etc.

The cell header contains a field indicating the number of droppable blocks in the cell. In order to protect voice-band data and facsimile cells from block dropping, this field is always set to zero.

Results in [11] show that, in general, the bit-dropping method reduces the mean delay, especially with heavy loads, where threshold values have great effect on network performance. Low threshold values lead to severe degradation in the mean number of bits per sample, and also to increased packet loss probability at high load. High threshold values may result in a slight improvement in mean bits per sample or packet loss, but at the same time the mean packet delay increases.

Equation (10) offers a prudent choice of threshold values such that delay performance is optimized while performance objectives are met:

$$b = \min\left[4, \frac{1}{S}\left(\frac{C}{N_r} - H\right)\right], \tag{10}$$

where N is the number of voice sources, S is the number of voice samples per packet, H is the number of bits in the header, r is the packet arrival rate per voice source, and C is the transmission capacity.

In general, results prove that bit dropping smooths the superposition-packet voice process by speeding up the packet service rate during critical periods of congestion in the queue.

4. Bandwidth allocation methods

Integrated services such as data, voice, graphics, and video have different requirements in terms of performance and availability. The challenge is to satisfy these requirements while maximizing the use of shared network resources. This requires a flexible, controlled bandwidth allocation technique.

In this section the problem of adaptively managing the bandwidth of an integrated network is considered. Several methods exist for dealing with this problem, such as simple channel strategy [24], slot assignment methods [25, 26], virtual finishing time methods [27–30], integrated access with a cross-connected system [10, 31], resource allocation analysis programs [32–34], multilevel control methods [35, 36], network access port control methods [37], filtered input rate methods [38], statistical multiplexing methods [39], synchronous bandwidth allocation methods [40], and others [31, 41–47, 50–56]. Expert systems, neural networks, and fuzzy logic are also used to deal with the bandwidth allocation problem [48, 49, 57]. This section is divided into a number of subsections, each illustrating one method of efficiently allocating the bandwidth.

• *Simple dynamic channel strategy*

In this method, each service has its own channels. If a call arrives from a service without sufficient free channels, it

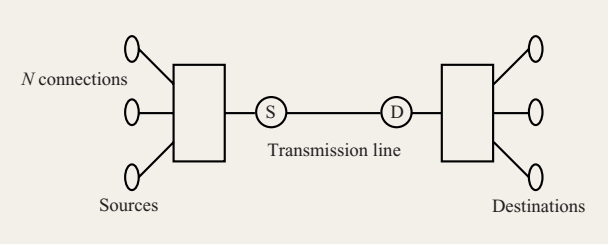

Figure 5

Two-node architecture.

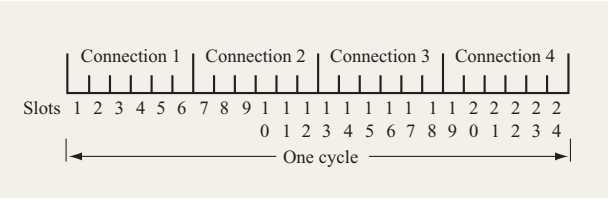

Figure 6

Division of a cycle among four connections.

can use channels from another service, but only if the current number of free channels at this service exceeds a certain threshold; otherwise the call is blocked.

Sometimes it is possible, especially in the case of telephone traffic, to determine the expected number of new calls arriving in a certain period of time by using a Poisson distribution. Then it is possible to maintain a specific blocking probability by keeping a reserve of free channels based on the expected number of calls in this period. This number may also help to set the threshold [24].

Results show that dynamic channel allocation affords more efficient resource utilization for all types of services. The throughput is increased with this method over that for the case in which each service uses its own channels. This increase is small, however, when the system carries primarily traffic from a single type of service. The percentage increase in throughput decreases as the number of channels assigned for each service increases. The throughput also decreases with any decrease in block probability or mean time between call initiations.

- *Slot assignment methods*

In the model shown in **Figure 5**, N connections are multiplexed in the source. Each source has N buffers, with a number of cells stored on each [25, 26].

In slot assignment algorithms, time is partitioned into cycles, each cycle is partitioned into a number of slots, and each slot is used to transmit one cell per connection. Five bandwidth allocation methods can be built using the slot assignment concept [25].

1. Static slot assignment method

In this method, both the duration of each cycle and the duration of each slot are constant, so the number of slots in each cycle is fixed. These slots are divided equally among the connections.

Figure 6 shows an example in which the cycle is divided into 24 equal slots, and the slots are divided equally among the four existing connections so that each connection has six slots to use, regardless of the number of cells it has. It is clear that this method leads to wastage of cells, since no connection can use slots other than its own, even if it has exhausted its own slots and the others have not used theirs.

2. Buffer-population-based dynamic slot assignment method

In this method also, both the duration of each cycle and the duration of each slot are constant, so the number of slots in each cycle is fixed. But in this method, the number of slots allocated to each connection is not constant, but is proportional to the number of cells in the buffer belonging to the connection at the time of allocation. However, there is a maximum limit on the number of cells allocated to a connection.

Due to this maximum limit, some slots may be unallocated in a cycle, leading to wastage of cells. There are two ways to overcome this disadvantage: The first is to ignore this maximum limit, and the second is to allocate these excess slots to the connection with the largest number of cells in its buffer.

3. Adaptive slot assignment method

In this method the cycle size is variable, and all connections are served in order. At its turn, any connection sends all of the cells stored in its buffer at that time, with a maximum limit m, so if the number of cells in the connection exceeds this limit, it sends only the first m cells and must wait for its next turn to send the rest (or another m cells).

4. First-come, first-served slot assignment method

In this method, no cycles exist. Each cell is transmitted on a first-come, first-served basis, regardless of the connection to which it belongs.

This method is extremely inefficient, since if a connection generates many cells because of an error, it will use a large portion of the network bandwidth, and this will slow the other connections with correct cells.

5. Rate-based dynamic slot assignment method
In this method, the size of each cycle and the number of slots allocated to a connection in a cycle are variable. With the bit rate for each connection being known, the number of slots allocated to a connection per cycle changes with this bit rate, but each connection must have at least one slot per cycle. Thus, for each period of time, the cycle size and the number of slots allocated to a connection per cycle both change. This method decreases the number of wasted slots, but some may still exist.

Results prove that Methods 2, 3, and 4 yield much lower average delay than Methods 1 and 5:

- As stated previously, the first-come first-served method has no way to prevent a connection from generating abnormally large numbers of cells and consequently using a large portion of the bandwidth.
- Methods 3 and 4 are superior, since slots are never wasted.
- In Methods 1 and 5 slots may be unused. This occurs when there is no cell from the connection to assign to these slots, while there are cells waiting from other sources.

Comparing average delay to line utilization, it has been found that

- The average delay for Method 4 is nearly equal to that for Method 3 for the entire range of line utilization.
- The average delay for Method 2 is slightly higher than that for Method 4 for low to moderate line utilization.
- The average delay for Method 2 is nearly equal to that for Method 4 for high line utilization.

- *Virtual finishing time methods*

High-speed networks are capable of carrying many types of traffic (voice, graphics, video, etc.), each with a different required quality of service (QOS). In virtual finishing methods (**Figure 7**), each node has a number of buffers, each type of service is stored in one buffer, and these buffers are themselves FIFO (the head of each buffer is served first). Since all of the buffers share one link, a scheduling method must be used.

This section discusses algorithms that determine how these buffers share the link bandwidth, using the expected departure time of calls, or virtual finishing time (VFT). Several methods for calculating the VFT [27–30] allow a guaranteed rate and average delay for each service, independently of the behavior of the other services. Note that c_i^n is the cell i arriving at a buffer n, and VFT F_i^n is the expected departure time of the cell i from buffer n.

1. Packetized generalized processor sharing (PGPS)
This method distributes the idle bandwidth at any time t, in proportion to a partitioning parameter ϕ of the non-

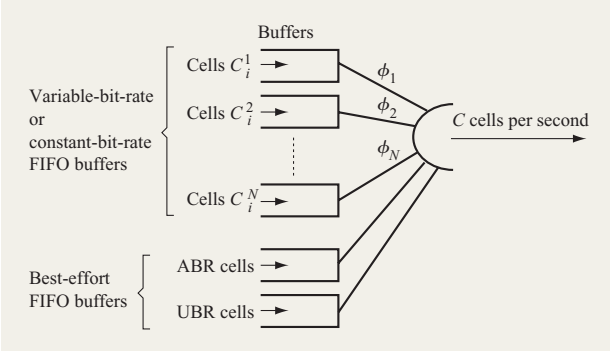

Figure 7

Processor-sharing mode. (ABR = available bit rate; UBR = unspecified bit rate.)

empty buffers [27–29]. Each c_i^n is given a VFT F_i^n. The cell having the least VFT is served first; if multiple cells, each at the head of a buffer, have the same VFT, they are served in arbitrary order:

$$VFT = F_i^n = \max\,[F_{i-1}^n, V(a_i^n)] + \frac{1}{\phi_n} \qquad (F_0^n = 0), \tag{11}$$

where ϕ_n is the bandwidth-partitioning parameter of the buffer n ($\phi_1 + \phi_2 + \phi_3 + \cdots + \phi_N \le 1$), a_i^n is the arrival time of c_i^n, and V is the virtual time function of the node.

The virtual time $V(t)$ is defined to be zero for all times at which the link is idle. For a busy period, the beginning time is set to zero [$V(0) = 0$]; then $V(t)$ evolves as follows [28]:

$$V(t_{j-1} + \tau) = V(t_{j-1}) + \frac{\tau}{\sum_{i \in B_j} \phi_i} \qquad \tau \le t_j - t_{j-1},$$

$$j = 2, 3, \cdots, \tag{12}$$

where B_j is the set of sessions that are busy in the interval (t_{j-1}, t_j).

Results prove that under PGPS, the delay of a packet from service i can be bounded in terms of the service i queue size seen by that packet upon arrival, even in the absence of any rate control. This enables services to take advantage of lightly loaded network conditions.

2. Self-clocked fair queuing (SCFQ)
This method uses the same concept as the packetized generalized processor sharing method, but with different $V(t)$. $V(t)$ is the VFT of the last non-best-effort cell served before time t. If no such cell exists, then $V(t) = 0$ [27, 29]:

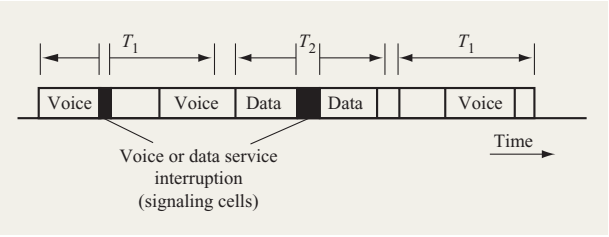

Figure 8

Time line for T1–T2 scheme.

$$V(t) = \begin{Bmatrix} 0 & \text{if no cell has been served before time } t \\ F_i^n & \text{otherwise} \end{Bmatrix}. \quad (13)$$

This method puts a bound on the delay of any cell passing through a node. It leads to higher average delay than the previous method, but also provides fairness in bandwidth distribution. Also, its VFT calculation is easier than that for the PGPS method [27].

3. Non-idling virtual clock

This method uses the same concept as the packetized generalized processor sharing method, but with different VFT [27, 30]:

$$VFT = F_i^n = \max\,(F_{i-1}^n, a_i^n) + \frac{1}{\phi_n} \qquad (F_0^n = 0). \quad (14)$$

This method produces less average packet delay than the previous two methods. Its main advantage is that the computation of the virtual time is negligible, since it is simply extracted from each packet as it goes into service.

4. Idling virtual clock

In this method, all cells at the heads of the buffers are considered for service at each departure epoch. This method uses the same concept as the PGPS and the same VFT equation as the previous method, but with different treatment of congested and best-effort cells. Buffers are considered to be congested if they exceed some threshold.

Two methods are offered; the first one favors congested cells with no regard for best-effort cells, while the second also favors congested cells, but with best-effort cells favored in the absence of congestion [27].

A. Favoring congested buffers This method favors congested cells. At any time t, the VFT for all buffers is calculated:

- If no buffers are congested,
  then a cell from buffer i of the smallest VFT (F_i^n) is served first.

Table 1 Service classification for the T1–T2 scheme.

Cell type	*Characteristics*	*Examples*
Type 1A Delay-sensitive, isochronous high-bandwidth services	Real-time services Bandwidth must be guaranteed at call setup Isochronous: Fixed high bandwidth is required for duration of call	Services needing high bandwidth with constant bit rate (CBR), such as CBR conference video Real-time high-CBR services
Type 1B Delay-sensitive, non-isochronous high-bandwidth services	Real-time services, with high bandwidth Non-isochronous: Each call alternates between active and inactive periods and generates high-bandwidth data stream during active periods	Variable-bit-rate video LAN-to-LAN calls
Type 2 Delay-insensitive high-bandwidth services	Non-real-time services User may specify tolerable delay Usually transported during slack periods	Document Image Video delivery services (non-real-time)
Type 3 Low-bandwidth statistically multiplexed services	Delay-sensitive, with end-to-end delay ranging from 10 s to 100 ms	Packetized voice Low-speed data Enquiry-response messages

- If at least one buffer is congested,
  then find the congested buffer j with the smallest VFT (F_j^m);
  find the non-empty buffer i (among all buffers) with the smallest VFT (F_i^n);
  if ($t \geq F_i^n$ and $F_i^n < F_j^m$),
  then c_i^n is served;
  else c_j^m is served.
- Best-effort cells are served iff all N buffers (congested or not) are idle.

B. Favoring congested buffers, then best-effort traffic This method also favors congested cells, but best-effort buffer cells are favored if no congested buffers exist. At any time t, the VFT for all buffers is calculated:

- If no buffers are congested and the best-effort buffer is empty,
  then the cell from buffer i of the smallest VFT (F_i^n) is served first.
- If no buffers are congested and the best-effort buffer is not empty,
  then find the buffer i with the smallest VFT (F_i^n);
  if ($F_i^n \leq t$),
  then c_i^n is served;
  else a best-effort cell is served.
- If at least one buffer is congested,
  then find the congested buffer j with the smallest VFT (F_j^m).
  find the non-empty buffer i (among all buffers) with the smallest VFT (F_i^n);
  if ($t \geq F_i^n$ and $F_i^n < F_j^m$),
  then c_i^n is served;
  else c_j^m is served.

The idling virtual clock method handles congested cells as well as best-effort cells (in the second scheme); this leads to minimum average delay for the overall packets.

- *Integrated access and cross-connect system (IACS) multiplexing method*

In this section, the T1–T2 scheme using only voice and data is considered; then a generalization of this method using all services is explained. We first classify all of the services, and then explain how they are treated.

T1–T2 scheme

In this method, only three types of cells are considered: signaling cells, voice cells (voice, voice-band data, and facsimile), and digital data cells. Each type is queued in a single buffer, and cells are served according to the T1–T2 scheme [10]:

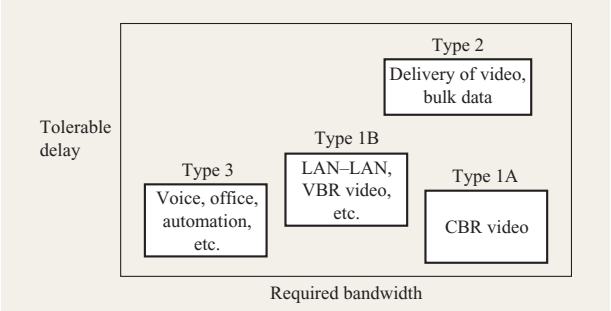

Figure 9

Delay bandwidth requirements for various traffic types.

- Waiting voice cells are served until a maximum of T_1 ms has elapsed *or* until the voice queue is exhausted.
- Waiting data cells are served until a maximum of T_2 ms has elapsed *or* until the data queue is exhausted.
- Signaling cells have high priority; they may interrupt both voice and data cells.

Figure 8 represents an example of the T1–T2 scheme. Because signaling traffic intensity is very small compared to others, we can say that the link bandwidth is allocated to voice and data with the ratio of T_1 to T_2, respectively:

$$\text{Bandwidth for voice} \approx \frac{T_1}{T_1 + T_2} B;$$

$$\text{Bandwidth for data} \approx \frac{T_2}{T_1 + T_2} B, \quad (15)$$

where B is the overall link bandwidth. Values of T_1 and T_2 can be chosen either to reserve minimum bandwidth proportions for both types, or to adjust the required delays for both cell types.

Dynamic time slice (DTS) scheme

This method is a generalization of the T1–T2 scheme, but with all broadband services. Cells are classified into four types; **Table 1** shows the characteristics of each type and gives examples of each one [10, 31]. **Figure 9** shows the traffic types and their delay bandwidth requirement.

Each type of service is assigned a separate queue. The DTS server services all of the queues by cyclically visiting each queue and allocating it a time slice. This time slice is proportional to the bandwidth required by the queue.

In any instance, if n is the number of non-empty queues in the DTS, one queue (with number 0) is added for signaling. To calculate the time slice for the $n + 1$ queue, the following equation must be used:

$$f_i = \frac{T_i}{\sum_{i=0}^{n} T_i} \qquad 0 \le i \le n;$$

$$\sum_{i=1}^{n} f_i \le 1 - f_0;$$

$$\sum_{i=0}^{n} T_i \le D_c, \tag{16}$$

where f_i is the fraction of the bandwidth for queue i, T_i is the time slice required for queue i, D_c is the DTS service cycle time (≈1 to 2 ms),

$$D_c = M_c \tau, \tag{17}$$

M_c is the number of cells in a DTS service cycle, τ is the cell transmission time on the link, and f_0 is the bandwidth reserved for the signaling traffic on each link. Different schemes are used for each traffic type.

Servicing scheme for Type 1A Whenever a user makes a call-set request to virtually allocate the required bandwidth, the network sends the request to all nodes en route to the destination. The user is guaranteed the allocated bandwidth for the duration of the call. A routing scheme is used to find the appropriate path from source to destination.

Servicing scheme for Type 1B With this type, calls within each class are statistically multiplexed, but each class is assigned a guaranteed bandwidth. Each class maintains a traffic table (bandwidth versus capacity) to provide the amount of bandwidth required to multiplex a given number of calls in the class queue. A new call is admitted iff spare bandwidth is available on the link, in which case the bandwidth for this call class is increased in the table by increasing the time slice. When the call completes, the bandwidth for this call class is updated by decreasing the time slice for this class. Because of statistical fluctuations, unused bandwidth may exist; any such unused bandwidth is available to other traffic in the system. A routing scheme is used to find the appropriate path for each call.

Servicing schemes for Type 2 Three schemes exist for this type; the needs specified by the call requests determine which should be used.

1. In this scheme, the user sets up a connection with the nearest service node, which in turn accepts the bulk information and stores it in its memory. After the destination is notified, the connection with the user is terminated. During slack periods of traffic, the service node negotiates with other nodes en route to the destination to allocate the appropriate bandwidth, then transports the data. When the data is delivered, the call is cleared, and the destination sends a service-completion message to the source.
2. In this scheme, the user specifies the transaction size, delay tolerance, and bandwidth required. The network itself requests the call setup, and the information awaits transmission at the user terminal, not in the access node. During a slack period, the call is set up, but within the deadline specified by the user; then information is delivered as in the first scheme.
3. In this scheme, each link is assigned a low-priority queue with no bandwidth guarantee. It uses only spare bandwidth (idle cell slots) in each DTS cycle. If the length of this buffer exceeds a certain threshold, STOP/SEND signals are sent to neighbors to prevent congestion.

Servicing scheme for Type 3 Serving this type is similar to serving Type 1B, except for call set-up acceptance and routing. Each Type 3 call class (e.g., voice, low-speed data, interactive image), has its own queue. Statistical multiplexing efficiency can be very high compared to Type 1B. Results prove that the DTS scheme has the following advantages:

- It guarantees the desired bandwidth to connections which require a fixed wide bandwidth, and thus facilitates setting up virtual circuits.
- For each class, call admission is based on capacity versus bandwidth tables for that class. These tables are updated as technology changes.
- It is an efficient way of combining isochronous high-bandwidth services with variable-bit-rate, statistically multiplexed connections.
- Placing each call class in a separate queue facilitates serving it according to its required QOS.
- Any bandwidth unused by a call class is available to other traffic present in the multiplexor.
- The call-admission procedures are call-class-based, so the usage cost for network resources can be estimated separately for each call class, and a simple call-class billing system can be built.

The main disadvantage of the DTS scheme is that users may be penalized by buffer overflow if they exceed the contracted bandwidth, and there is no spare bandwidth on the link.

- *Resource Allocation Analysis Program (RAAP)*

The Resource Allocation Analysis Program [32] is particularly useful for satellite communication. Communication by satellite requires both bandwidth and power resources. A single request is actually two

subrequests, one forward and one backward, and backward subrequests may have different characteristics from forward ones. A request is not satisfied unless sufficient quantities of both power and bandwidth are available for both subrequests. Resource requests may be negotiated at the time of the request.

Each subrequest has the following information:

- *Arrival time* (time at which the request is received by the resource allocation process).
- *Start time* (time at which resources will actually begin to be used).
- Duration.
- Data rate (D).
- Modulation (M).
- Request negotiation.
- Forward error correction code rate (FEC).
- Geographical zone.

Note that $M = 0.5$ for QPSK (forward), and $M = 1$ for BPSK (backward). Requests may be immediate or booked (in advance).

From this information, bandwidth (B) for each subrequest (direction) is computed using the formula

$$B = \frac{KDM}{FEC}, \tag{18}$$

where K is a bandwidth factor to provide margin for adjacent-channel interfaces, etc. ($K \approx 1.5$). Two possible algorithms for allocating the resources exist:

- *Complete partitioning (CP):* All available resources are partitioned from the beginning among the different service classes, and each request can use only its own channel [33].
- *Complete sharing (CS):* All available resources are available to all classes [34].

RAAP allows the following:

- Partitioning of bandwidth into pools (each pool has a number of channels).
- Sharing among channel pools.
- Designation of a pool of raw bandwidth for common use.

Each time period, statistics are taken in order to assign bandwidth to each channel pool. For each pair of pools x and y, a table exists to specify the maximum number of channels that pool x can take from pool y (a maximum threshold may exist).

For any request, both the start time and the duration are known, so we can visualize a "request window" rectangle, with its left side at the start time and its right side at the start time + duration. Height represents the required resource, and any modification of start time or duration will change the position of the whole rectangle.

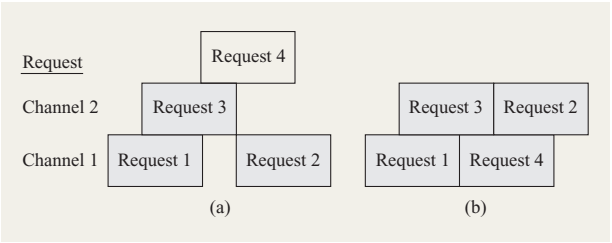

Figure 10

A two-channel request pool.

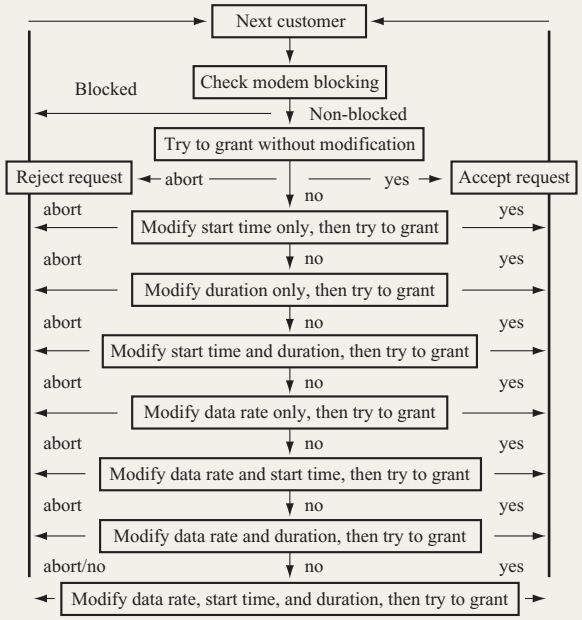

Figure 11

Request modification sequence.

Figure 10 represents a pool with only two channels, into which four requests are made. In Figure 10(a) Request 4 is lost, since no channel is free at its arrival time, while in Figure 10(b) requests are organized such that no request is lost. Research is still needed on means for allocating the requests among channels to obtain maximum performance.

As shown in **Figure 11**, when requests arrive at the resource controller, the controller checks the resource

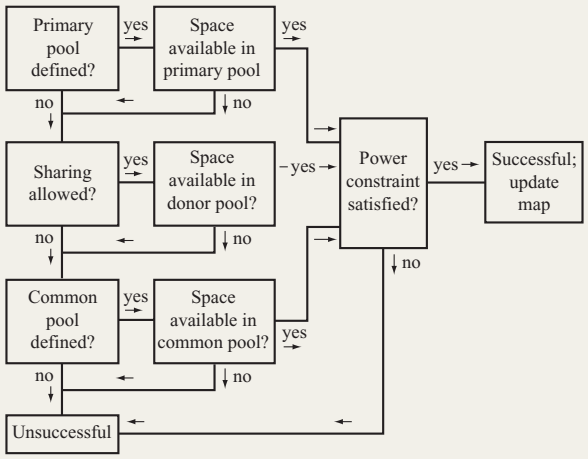

Figure 12

Resource allocation sequence.

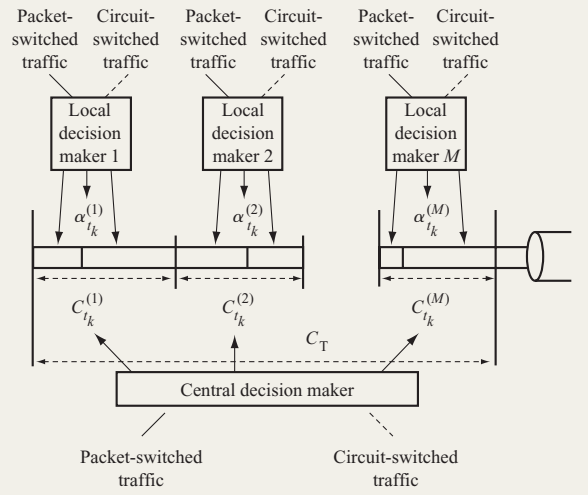

Figure 13

Decision-making structure for TDM multiplexor.

maps and attempts to grant requests without any modification. If this is not possible, it will then try to grant requests with various combinations of modifications (start time, duration, data rate, or any combination of them). Three answers are possible for any attempt:

- *Yes:* The request is granted and the customer accepts any modifications.
- *No:* The request is not granted because the modifications cannot be satisfied.
- *Abort:* The request can be granted but the customer refuses the modifications.

Note that a request may be blocked at the outset if there are no idle modems available at the resource controller to accept the data call.

If a subrequest with modification is granted, the resource allocation sequence shown in **Figure 12** will be followed. This algorithm is used for considering various levels of sharing.

The primary channel pool is the pool whose data rate, modulation, and FFC values match those of the subrequest itself. If this pool does not exist or has insufficient space, sharing among channel pools is attempted. If this also fails, the controller attempts to place the subrequest in the common bandwidth pool if it is defined. The satellite power constraints must also be checked.

The usefulness of the RAAP program was illustrated in [32] using two data-rate pairs. Parameters of these examples are given in **Table 2**.

Note that an initial block occurs whenever a request is not granted with its original values (start time, duration, data), while an ultimate block occurs whenever a request does not receive service of any kind.

- *Hierarchical (multilevel) control method*

This section addresses a bandwidth allocation problem in the access area to a TDM network carrying two basic traffic types, circuit-switched and packet-switched. Each user is assigned a portion of the total bandwidth (slots per frame), and this portion is dynamically allocated to the two traffic types at the user premises, using a two-layer hierarchical optimization scheme [31, 35, 36].

Consider a TDM multiplexor that carries the traffic of M user sites on a link with total capacity of C_T slots per frame. Suppose each user i has the two traffic types, circuit switching with an average call arrival rate of $\lambda_1^{(i)}$ calls per second and an average holding time of $1/\mu^{(i)}$ second, and packet-switching traffic with an arrival rate of $\lambda_2^{(i)}$ packets per second, with a fixed packet length of one slot.

Using a discrete time model with frame duration of b seconds, the following formula must hold:

$$\sum_{i=1}^{M} (\lambda_1^{(i)}/\mu^{(i)} + \lambda_1^{(i)} b) < C_T . \qquad (19)$$

The hierarchical control has a decision-making structure (**Figure 13**) comprising the following:

Table 2 Results and statistics from application of the RAAP program [32].

Application	*High-speed image/data transfer*	*Medium-speed data transfer*
Data-rate pair (kb/s) (Traffic intensity per number of channels)	768/64	256/64
Average no. of requests	35	30
Average total requests	65	65
Minimum total requests	55	55
Maximum total requests	75	75
Duration (minutes)	$15 \rightarrow 90$	$30 \rightarrow 150$
Start time	50% 8 a.m.–2 p.m. 25% 12–8 p.m. 25% 2–12 p.m.	All day
Average arrival rate (requests per minute)	49.44	84.58
Average holding time for a channel (minutes)	0.04381	0.02084
Average service delay (minutes)		
Booked	0.2	0.3
Immediate	0.3	0.8
Average blocking probability (%)		
Initial	2.1	3.9
Ultimate	1.3	2.8

Satellite bandwidth utilization = 0.129.
Satellite power utilization = 0.126.

- One central decision maker, which assigns each user i a capacity of $c_{t_k}^{(i)}$ slots per frame at decision instants t_k, $k = 0, 1, \cdots$. This assignment lasts over the frames $t_k, t_k + 1, \cdots, t_{k+1} - 1$.
- M local decision makers, which, at the same instant, choose a parameter $\alpha_{t_k}^{(i)}$ which influences the local admission control rules that each decision maker exerts at the beginning of each frame on the incoming call requests for circuit-switched traffic.

At any time, if a new frame t arrives from any user i, it is either accepted (by the local decision maker) with probability $B_t^{(i)}$, or blocked with probability $1 - B_t^{(i)}$. If $r_t^{(i)}$ is the number of calls requiring continuation of service in the next frame, then

$$B_t^{(i)} = \begin{cases} 1 - \left[\dfrac{3(r_t^{(i)})^2}{(c_{t_k}^{(i)} - 1)^2} - \dfrac{2(r_t^{(i)})^3}{(c_{t_k}^{(i)} - 1)^3} \right]^{\alpha_{t_k}^{(i)}} & \text{if } c_{t_k}^{(i)} > r_t^{(i)} + 1; \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where $t = t_k, t_k + 1, \cdots, t_{k+1} - 1$; $B_t^{(i)}$ is continuously differentiable; and $0 \le B_t^{(i)} \le 1$.

For every $c_{t_k}^{(i)} > 0$, $\alpha_{t_k}^{(i)} > 0$ is the parameter to be chosen by the local decision maker at time t_k. From Equation (20), we conclude that as $\alpha_{t_k}^{(i)}$ increases, $B_t^{(i)}$ increases (as $\alpha_{t_k}^{(i)} \rightarrow \infty$, $B_t^{(i)} \rightarrow 1$).

If each process $r_t^{(i)}$, $i = 1, 2, \cdots, M$, is a Markov chain with transition probability, then

$$p_{jk}^{(i)} = P(r_{t+1}^{(i)} = k | r_t^{(i)} = j)$$

$$= \begin{cases} \mu^{(i)} b_j & k = j - 1; \\ 1 - \mu^{(i)} b_j - \lambda_1^{(i)} b \beta_t^{(i)} & k = j; \\ \lambda_1^{(i)} b \beta_t^{(i)} & k = j + 1; \\ 0 & \text{otherwise,} \end{cases} \tag{21}$$

where $P_{t_k}^{(i)}$ is the transition probability matrix of $r_t^{(i)}$, $t \in (t_k, t_{k+1} - 1)$. Hence, we can define the following parametric optimization problem.

At the beginning of each new period $(t_k, t_{k+1} - 1)$, $k = 0, 1, \cdots$, on the basis of the knowledge of all prior information and of $r_{t_k}^{(i)}$, $i = 1, 2, \cdots, M$, and also with the following constraints:

$$\sum_{i=1}^{M} c_{t_k}^{(i)} = C_{\mathrm{T}} \qquad k = 0, 1, \cdots, \tag{22}$$

$$c_{t_k}^{(i)} \ge r_{t_k}^{(i)} \qquad \begin{matrix} i = 1, \cdots, M, \\ k = 0, 1, \cdots, \end{matrix} \tag{23}$$

$$\alpha_{t_k}^{(i)} > 0 \qquad \begin{matrix} i = 1, \cdots, M, \\ k = 0, 1, \cdots, \end{matrix} \tag{24}$$

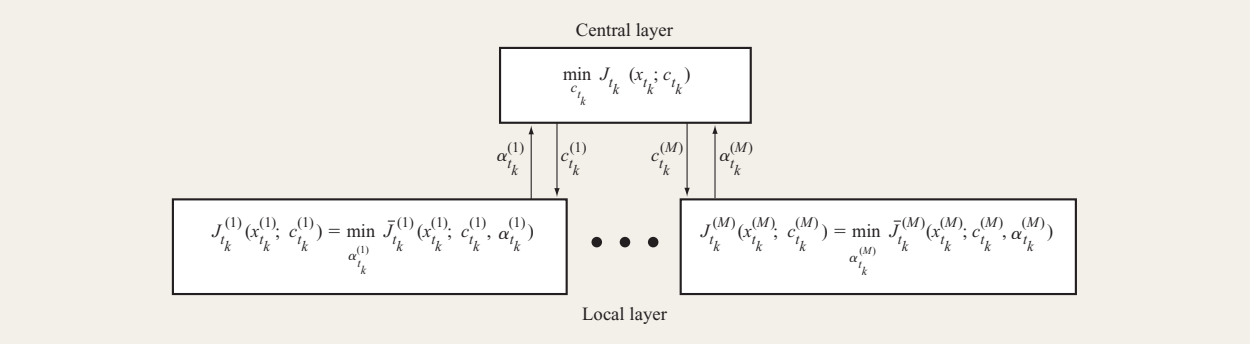

Figure 14

Decomposition and coordination structure.

find capacity vector $c_{t_k} \equiv \text{col}[c_{t_k}^{(1)}, c_{t_k}^{(2)}, \cdots, c_{t_k}^{(M)})$ and parameter vector $\alpha_{t_k} \equiv \text{col}[\alpha_{t_k}^{(1)}, \alpha_{t_k}^{(2)}, \cdots, \alpha_{t_k}^{(M)})$, which minimize

$$\bar{J}_{t_k}(X_{t_k}; c_{t_k}, \alpha_{t_k}) = \sum_{i=1}^{M} \bar{J}_{t_k}^{(i)}(x_{t_k}^{(i)}; c_{t_k}^{(i)}, \alpha_{t_k}^{(i)}), \tag{25}$$

where

$$\bar{J}_{t_k}^{(i)}(x_{t_k}^{(i)}; c_{t_k}^{(i)}, \alpha_{t_k}^{(i)}) = \sum_{t=t_k}^{t_{k+1}-1} \left\{ \sum_{s=0}^{C_T} [(1 - \beta^{(i)}(s))\lambda_1^{(i)} b] \cdot \Pi_t^{(i)}(s) \right\}$$
$$+ \sigma^{(i)} \sum_{t=t_k}^{t_{k+1}-1} \max \left[0; \frac{L_{t_k}^{(i)}}{G} + \lambda_2^{(i)} b - \sum_{s=0}^{C_T} (c_{t_k}^{(i)} - s) \cdot \Pi_t^{(i)}(s) \right], \tag{26}$$

- $\sigma^{(i)}$ is a weighting coefficient,
- $L_{t_k}^{(i)}$ is the number of packets in the packet queue of user i at time t_k,
- $X_{t_k} \equiv \text{col}[x_{t_k}^{(1)}, x_{t_k}^{(2)}, \cdots, x_{t_k}^{(M)}]$; $\quad x_{t_k}^{(i)} \equiv \text{col}[r_{t_k}^{(i)}, L_{t_k}^{(i)}]$,
- $\beta^{(i)}(s) \equiv \beta_t^{(i)}$ for $r_t^{(i)} = s$,
- $G \equiv t_{k+1} - t_k$ (control horizon in frames),
- $\Pi_t^{(i)}(s) = \Pr\{r_t^{(i)} = s\}$,
- $\pi_t^{(i)} = [\Pi_t^{(i)}(0), \Pi_t^{(i)}(1), \cdots, \Pi_t^{(i)}(C_T)]^T$, and
- $\pi_{t+1}^{(i)} = [P_{t_k}^{(i)}]^T, \quad t \in [t_k, t_{k+1} - 1]$.

The first term in J represents the average number of blocked calls per frame at user i (circuit-switching traffic). In the second term, the nonzero argument of $\max(0; \cdots)$ represents an approximation of the difference between the average input flow into and the average output flow out of the packet queue at user i.

A multilevel decomposition of the problem can be done (**Figure 14**). Having M local decision makers and one central, this central assigns each user i a capacity c_{t_k} at decision instants t_k, while the local decision makers decide the parameter α_{t_k}.

The following two steps are repeated until convergence or for a suitable number of iterations:

- For fixed values of variables $c_{t_k}^{(i)}$, all decision makers independently solve their local problems in order to decide their own $\alpha_{t_k}^{(i)}$. Vector α_{t_k} is passed to the central decision maker.
- Using vector α_{t_k}, the central decision maker finds the appropriate c_{t_k} considering all mentioned constraints [Equations (22)–(24)]. Vector c_{t_k} is passed to a local decision maker.

Note that the local optimization problem is numerically straightforward (unidirectional search), while the central optimization can be effected by means of the gradient projection method [31, 35]. Some simplification is done so that the gradient projection takes place only on the constraint: $c_{t_k}^{(i)} \geq r_{t_k}^{(i)} \ \forall \ i \neq m$. *If* w is the iteration number of the optimization procedure and $c_{t_k}^{w} \equiv \text{col}[c_{t_k}^{(1),w}, c_{t_k}^{(2),w}, \cdots, c_{t_k}^{(M),w}]$, the descent step is given by

$$c_{t_k}^{(i),w+1} = \max \left\{ r_{t_k}^{(i)}; c_{t_k}^{(i),w} - \gamma^w \left[\frac{\partial \tilde{J}_{t_k}}{\partial c_{t_k}^{(i)}} - \frac{\partial \tilde{J}_{t_k}}{\partial c_{t_k}^{(m)}} \right] c_{t_k}^{w} \right\}$$
$$\forall \ i \neq m, \tag{27}$$

where γ^w is the step size.

This method not only avoids congestion, but also achieves efficient resource utilization.

The hierarchical control method labeled as "optimum distributed" was compared with another one representing

centralized optimization. Centralized optimization has the following characteristics:

- There is only one parameter affecting the user's admission control function: the assigned capacity.
- The optimization problem is carried out only on the capacity variables by a central decision maker.
- α_{t_k} is initially fixed and kept constant.

Percentage of refused calls and average queue length are used as performance parameters in the comparison between the hierarchical control method and centralized optimization. With respect to call refusal, distributed optimization always yields better results than centralized optimization. With respect to the average packet queue, in most cases (especially at maximum load), distributed optimization also gives better results than centralized optimization.

5. Bandwidth and virtual path

In this section, the virtual path concept used in broadband networks is considered [7, 58, 59, 63–68]. The virtual path is defined as a labeled path (bundles of multiplexed circuits) terminated by virtual path terminators which can identify each connection from or to the network elements (terminals) that are included in the segment network to which the virtual path terminals belong.

A simpler definition is illustrated:

- A virtual path is a logical direct link between two nodes.
- A predefined route is defined for each virtual path in the physical facilities network.
- A virtual path includes a number of virtual circuits.
- Each virtual path has a bandwidth, which defines the maximum number of virtual circuits it can carry.
- Virtual paths are multiplexed on physical transmission links in a cell-multiplexing manner.

In **Figure 15**, virtual path 1 (VP1) is set between nodes 1 and 4, virtual path 2 (VP2) is set between nodes 1 and 5, and virtual path 3 (VP3) is set between nodes 4 and 5.

When a customer asks the network to set up a call with a known bandwidth, the following steps are taken. If the request bandwidth is less than the total idle bandwidth, the network accepts the call, and the customer may communicate at any arbitrary bandwidth less than the allocated one. This is done by selecting the most appropriate virtual path from the end nodes terminating the virtual path. Otherwise the call is blocked, but the customer may request a new setup with reduced bandwidth.

Many control schemes exist to set up individual virtual path bandwidth in order to optimize the performance measures fairly for all users [7, 58, 59, 63–68].

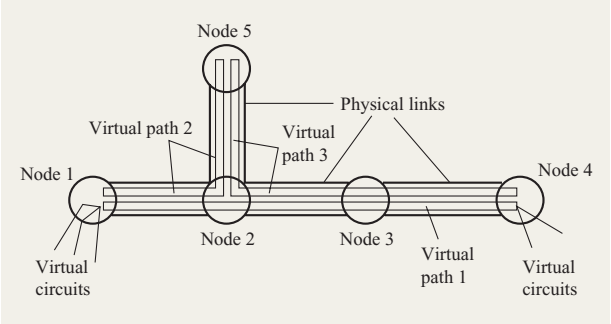

Figure 15

Virtual path concept.

- *Flexible assigning method*

In this method, when the connections on a virtual path decrease, the remaining link capacity on that virtual path is assigned to the busiest virtual path so that each virtual path is well utilized.

Results prove that this method increases transmission efficiency, but it also increases the processing load [52, 58].

- *Facility (or group transit) switching method*

In this method, a number of circuit groups exist. Each circuit groups a number of virtual paths along a specific route. The number of circuits in the group corresponds to the virtual path and its bandwidth. Circuit groups must be renewed at transit nodes as well as at the end nodes. The number of circuit groups is also variable. A packing algorithm is necessary to increase the utilization of the groups.

This method also increases the processing load [52, 58, 66].

- *Dynamic control method*

In this method, a routing procedure at call setup is used at the end nodes to select the most appropriate virtual path between the two nodes. No routing procedure is used at the transit nodes.

Bandwidth allocation at call setup is done by comparing the bandwidth of the request call to the unused bandwidth of the virtual path at the end nodes. The algorithm takes the following steps [3, 52, 58]: If the current bandwidth is less than the bandwidth needed for connecting this call to the end nodes, then try to increase the bandwidth by a specified step. If the increase is allowed, then increase the bandwidth and set up a virtual path for the call. Otherwise, maintain the current bandwidth and refuse the call.

Any increase or decrease in the bandwidth by a specified step, *if possible*, is according to the virtual path utilization condition. The performance of the network depends on the step size of the bandwidth change. If i_k is the set of concurrent connections of virtual circuits in virtual path k, then $W_k(i_k)$ is the normalized bandwidth of virtual path k. The term $u_{p,k}$ is a non-negative integer satisfying the following formula:

$$W_k(i_k) = u_{p+1,k}, \qquad u_{p,k} < i_k \le u_{p+1,k}, \qquad p = 0, 1, 2, 3, \cdots. \quad (28)$$

This formula implies that the unused bandwidth of virtual path k is zero when there are $u_{p,k}$ concurrent connections. This bandwidth is increased to $u_{p+1,k}$ if a new call arrives.

If the bandwidth change step is S, then

$$u_{p,k} = pS \qquad \text{for} \quad \begin{matrix} k = 1, 2, 3, \cdots, n, \\ p = 0, 1, 2, \cdots, \end{matrix} \quad (29)$$

where n is the number of virtual paths multiplexed on a link.

Results prove that dynamic control of the virtual path provides an effective means for constructing dynamically reconfigurable networks. It enhances the adaptability and flexibility of the network and improves the transmission link capacity utilization, but also increases the necessary processing [66].

Mathematical analysis in [58] proves that the smallest step (4) greatly increases efficiency even in the case of low traffic, while a step of 8 or 12 increases efficiency only in high traffic.

6. Summary

This paper describes four bandwidth problems in high-speed networks and presents several possible solutions for them. These problems are 1) the topology design and bandwidth allocation problem, 2) the bandwidth management and congestion problem, 3) the bandwidth allocation problem, and 4) the virtual-path setup assignment problem.

The topology design and bandwidth allocation problem is concerned with the ability to dynamically reconfigure a packet-switched network in order to obtain a more efficient design that achieves the best operation of the network.

Bandwidth management (BWM) protocols are concerned with congestion, which occurs when various sources compete for network resources while these resources cannot meet all of the demands. This may happen when logical channels request bandwidth that cannot be supported, or when the network admits more calls than its links can handle, or at any node due to buffer shortage. BWM methods have five objectives: protection of shared network resources, fairness, robustness, simplicity, and flexibility. Five methods are illustrated in this paper, all based on call admission or refusal.

Bandwidth allocation methods are built to adaptively manage the transmission bandwidth of an integrated network by implementing optimal partitioning policies for that bandwidth. This paper illustrates six methods to deal with this problem, each using a different concept.

The virtual path is a logical direct link between two nodes which includes a number of virtual circuits. Each virtual path has a bandwidth that defines the maximum number of virtual circuits it can carry. Three methods are given to set up such virtual paths in order to optimize the performance parameters for all users in a fair manner.

References

1. Mario Gerla, Jose Augusto, Suruagy Monteiro, and Rodolfo Pazos, "Topology Design and Bandwidth Allocation in ATM Nets," *IEEE J. Selected Areas in Commun.* **7,** No. 8, 1253–1262 (1989).
2. Achille Pattavina, "Multichannel Bandwidth Allocation in Broadband Packet Switch," *IEEE J. Selected Areas in Commun.* **6,** No. 9, 1489–1499 (1988).
3. Eric Livermore, Richard P. Skillen, Maged Beshai, and Mark Wernik, "Architecture and Control of an Adaptive High-Capacity Flat Network," *IEEE Communications Magazine* **36,** No. 5, 106–112 (1998).
4. Bulent Yener, Yoram Ofek, and Moti Yung, "Combinational Design of Congestion-Free Networks," *IEEE/ACM Trans. Networking* **5,** No. 6, 989–1000 (1997).
5. Andre Girard and Brunile Sanso, "Multicommodity Flow Models, Failure Propagation, and Reliable Loss Network Design," *IEEE/ACM Trans. Networking* **6,** No. 1, 82–93 (1998).
6. A. E. Eckberg, D. T. Luan, and D. M. Lucantoni, "Bandwidth Management: A Congestion Control Strategy for Broadband Packet Networks Characterizing the Throughput-Burstiness Filter," *Computer Network & ISDN Syst.* **20,** 415–423 (1990).
7. Youichi Sato and Ken-Ichi Sato, "Virtual Path and Link Capacity Design for ATM Networks," *IEEE J. Selected Areas in Commun.* **9,** No. 1, 104–110 (1991).
8. Levent Gun and Rock Guerin, "Bandwidth Management and Congestion Control Framework of the Broadband Network Architecture," *Computer Network & ISDN Syst.* **26,** 61–78 (1993).
9. Jonathan S. Turner, "Managing Bandwidth in ATM Networks with Bursty Traffic," *IEEE Network Magazine* **6,** No. 5, 50–58 (1992).
10. K. Sriram, "Methodologies for Bandwidth Allocation Transmission Scheduling, and Congestion Avoidance in Broadband ATM Networks," *Computer Network & ISDN Syst.* **26,** 43–59 (1993).
11. Kotikalapudi Sriram and David M. Lucantoni, "Traffic Smoothing Effects of Bit Dropping in a Packet Voice Multiplexer," *IEEE Trans. Commun.* **37,** No. 7, 703–712 (1989).
12. David W. Petr, Luiz A. Dasilva, Jr., and Victor S. Frost, "Priority Discarding of Speech in Integrated Packet Networks," *IEEE J. Selected Areas in Commun.* **7,** No. 5, 644–656 (1989).
13. Gillian M. Woodruff and Rungroj Kositpaiboon, "Multimedia Traffic Management Principles for Guaranteed ATM Network Performance," *IEEE J. Selected Areas in Commun.* **8,** No. 3, 437–446 (1990).

14. Israel Cidon, Inder Gopal, and Roch Guerin, "Bandwidth Management and Congestion Control in plaNET," *IEEE Communications Magazine* **29,** No. 10, 54–64 (1991).
15. S. Jamaloddin Golestani, "Congestion-Free Communication in High-Speed Packet Networks," *IEEE Trans. Commun.* **39,** No. 12, 1802–1812 (1991).
16. Aurel A. Lazar and Giovanni Pacifici, "Control of Resources in Broadband Networks with Quality of Service Guarantees," *IEEE Communications Magazine* **29,** No. 10, 66–73 (1991).
17. Nasir Ghani, Soracha Nananukul, and Sudir Dixit, "ATM Traffic Management Considerations for Facilitating Broadband Access," *IEEE Communications Magazine* **36,** No. 11, 98–105 (1998).
18. Kuyan Liu, David W. Petr, Victor S. Frost, Hongbo Zhu, Cameron Braun, and William L. Edwards, "Design and Analysis of a Bandwidth Management Framework for ATM-Based Broadband ISDN," *IEEE Communications Magazine* **35,** No. 5, 138–145 (1997).
19. Bing Zheng and Mohammed Atiquzzaman, "Traffic Management of Multimedia over Satellite ATM Networks," *IEEE Communications Magazine* **37,** No. 1, 33–39 (1999).
20. Rohit Goyal, Raj Jain, Mukul Goyal, Sonia Fahmy, Bobby Vandalore, and Sastri Kota, "Traffic Management for TCP/IP over Satellite ATM Networks," *IEEE Communications Magazine* **37,** No. 5, 56–61 (1999).
21. Zbigniew Dziong, Marek Juda, and Lorne G. Mason, "A Framework for Bandwidth Management in ATM Networks—Aggregate Equivalent Bandwidth Estimation Approach," *IEEE/ACM Trans. Networking* **5,** No. 1, 134–147 (1997).
22. Shivakant Mishra and Lei Wu, "An Evaluation of Flow Control in Group Communication," *IEEE/ACM Trans. Networking* **6,** No. 5, 571–587 (1998).
23. Kohei Shiomoto, Shinichiro Chaki, and Naoaki Yamanaka, "A Simple Bandwidth Management Strategy Based on Measurements of Instantaneous Virtual Path Utilization in ATM Networks," *IEEE/ACM Trans. Networking* **6,** No. 5, 625–634 (1998).
24. Chris J. Powell and Victor C. M. Leung, "Demand Assignment Multiple Access and Dynamic Channel Allocation Strategies for Integrating Radio Dispatch and Telephone Services over Mobile Satellite Systems," *IEEE J. Selected Areas in Commun.* **10,** No. 6, 1020–1029 (1992).
25. Shyamal Chowdhury and Kazem Sohraby, "Bandwidth Allocation Algorithms for Packet Video in ATM Networks," *Computer Network & ISDN Syst.* **26,** 1215–1223 (1994).
26. Prodip Sen, Basil Marglaris, Nasser-Eddine Rikli, and Dimitris Anastassiou, "Models for Packet Switching of Variable-Bit-Rate Video Sources," *IEEE J. Selected Areas in Commun.* **7,** No. 5, 865–869 (1989).
27. Anthony Hung and George Kesidis, "Bandwidth Scheduling for Wide-Area ATM Networks Using Virtual Finishing Times," *IEEE/ACM Trans. Networking* **4,** No. 1, 49–54 (1996).
28. Abhay Parekh and Robert G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case," *IEEE/ACM Trans. Networking* **1,** No. 3, 344–357 (1993).
29. Mark W. Garrett, "A Service Architecture for ATM: From Applications to Scheduling," *IEEE Network Magazine* **10,** No. 3, 14 (1996).
30. Lixia Zhang, "Virtual Clock: A New Traffic Control Algorithm for Packet-Switched Networks," *ACM Trans. Computer Syst.* **9,** No. 2, 101–124 (1991).
31. Raffaele Bolla, Franco Davoli, and Mario Marchese, "Bandwidth Allocation and Admission Control in ATM Networks with Service Separation," *IEEE Communications Magazine* **35,** No. 5, 130–137 (1997).
32. David W. Petr, K. M. S. Murty, Victor S. Frost, and Lyn A. Neir, "Modeling and Simulation of the Resource Allocation Process in a Bandwidth-on-Demand Satellite Communications Network," *IEEE J. Selected Areas in Commun.* **10,** No. 2, 465–477 (1992).
33. G. J. Foschini, A. Gopinath, and J. F. Hayes, "Optimum Allocation of Servers of Competing Customers," *IEEE Trans. Commun.* **29,** No. 7, 1051–1055 (1981).
34. Farouk Kamoun and Leonard Kleinrock, "Analysis of Shared Finite Storage in a Computer Network Node Environment Under General Traffic Conditions," *IEEE Trans. Commun.* **28,** No. 7, 992–1003 (1980).
35. R. Bolla and F. Davoli, "Dynamic Hierarchical Control of Resource Allocation in an Integrated Services Broadband Network," *Computer Network & ISDN Syst.* **25,** 1079–1087 (1993).
36. Janusz Filipiak, "Structured Systems Analysis Methodology for Design of an ATM Network Architecture," *IEEE J. Selected Areas in Commun.* **7,** No. 8, 1263–1273 (1989).
37. Gopalakrishnan Meempat and Malur K. Sundareshan, "Optimal Channel Allocation Policies for Access Control of Circuit-Switched Traffic in ISDN Environments," *IEEE Trans. Commun.* **41,** No. 2, 338–350 (1993).
38. San-Qi Li, Song Chong, and Chia-Lin Hwang, "Link Capacity Allocation and Network Control by Filtered Input Rate in High-Speed Networks," *IEEE/ACM Trans. Networking* **3,** No. 1, 10–25 (1995).
39. Hyong W. Lee and Jon W. Mark, "Capacity Allocation in Statistical Multiplexing of ATM Sources," *IEEE/ACM Trans. Networking* **3,** No. 2, 139–151 (1995).
40. Sijing Zhang and Alan Burns, "An Optimal Synchronous Bandwidth Allocation Scheme for Guaranteeing Synchronous Message Deadlines with the Timed-Token MAC Protocol," *IEEE/ACM Trans. Networking* **3,** No. 6, 729–741 (1995).
41. George Kesidis, Jean Warland, and Cheng-Shang Chang, "Effective Bandwidths for Multiclass Markov Fluids and Other ATM Sources," *IEEE/ACM Trans. Networking* **1,** No. 4, 424–428 (1993).
42. K. J. Maly, E. C. Foudriat, R. Mukkamala, C. M. Overstreet, and D. Game, "Dynamic Allocation of Bandwidth in Multichannel Metropolitan Area Networks," *Computer Network & ISDN Syst.* **25,** 203–233 (1992).
43. Zbigniew Dziong, Liao Ke-Qiang, and Lorne Mason, "Effective Bandwidth Allocation and Buffer Dimensioning in ATM-Based Networks with Priorities," *Computer Network & ISDN Syst.* **25,** 1065–1078 (1993).
44. Gerard R. Pieris and Galen H. Sasaki, "Scheduling Transmissions in WDM Broadcast-and-Select Networks," *IEEE/ACM Trans. Networking* **2,** No. 2, 105–110 (1994).
45. David R. Irvin, "The Role of Customer-Premises Bandwidth Management," *IEEE Network Magazine* **8,** No. 3, 18–25 (1994).
46. Sally Floyd and Van Jacobson, "Link-Sharing and Resource Management Models for Packet Networks," *IEEE/ACM Trans. Networking* **3,** No. 4, 365–386 (1995).
47. Hua Jiang and Stephen S. Rappaport, "Prioritized Channel Borrowing Without Locking: A Channel Sharing Strategy for Cellular Communications," *IEEE/ACM Trans. Networking* **4,** No. 2, 163–172 (1996).
48. Shervin Erfani, Manu Malek, and Harvi Sachar, "An Expert System-Based Approach to Capacity Allocation in a Multiservice Application Environment," *IEEE Network Magazine* **4,** No. 2, 7–12 (1991).
49. Atsurshi Hiramatsu, "Integration of ATM Call Admission Control and Link Capacity Control by Distributed Neural Networks," *IEEE J. Selected Areas in Commun.* **9,** No. 7, 1131–1138 (1991).
50. Lampros Kalampoukas, Anujan Varma, and K. K. Ramakrishnan, "Two-Way TCP Traffic over Rate

Controlled Channels: Effects and Analysis," *IEEE/ACM Trans. Networking* **6,** No. 6, 729–743 (1998).
51. Erol Gelenbe, Xiaowen Mang, and Raif Onvural, "Bandwidth Allocation and Call Admission Control in High-Speed Networks," *IEEE Communications Magazine* **35,** No. 5, 123–129 (1997).
52. Hiroshi Saito, "Dynamic Resource Allocation in ATM Networks," *IEEE Communications Magazine* **35,** No. 5, 146–153 (1997).
53. Marwan Krunz, "Bandwidth Allocation Strategies for Transporting Variable-Bit-Rate Video Traffic," *IEEE Communications Magazine* **37,** No. 1, 40–46 (1999).
54. Jaime Sanchez, Ralph Martinez, and Michael W. Marcellin, "A Survey of MAC Protocols Proposed for Wireless ATM," *IEEE Network Magazine* **11,** No. 6, 52–62 (1997).
55. George M. Stamatelos and Vassilios N. Koukoulidis, "Reservation-Based Bandwidth Allocation in a Radio ATM Network," *IEEE/ACM Trans. Networking* **5,** No. 3, 420–428 (1997).
56. Brian L. Mark and Gopalakrishnan Ramamurthy, "Real-Time Estimation and Dynamic Renegotiation of UPC Parameters for Arbitrary Traffic Sources in ATM Networks," *IEEE/ACM Trans. Networking* **6,** No. 6, 811–827 (1998).
57. Christos Douligeris and George Develekos, "Neuro-Fuzzy Control in ATM Networks," *IEEE Communications Magazine* **35,** 154–162 (1997).
58. Satoru Ohta and Ken-Ichi Sato, "Dynamic Bandwidth Control of the Virtual Path in an Asynchronous Transfer Mode Network," *IEEE Trans. Commun.* **40,** No. 7, 1239–1247 (1992).
59. Nikolaos Anerousis and Aurel A. Lazar, "Virtual Path Control for ATM Networks with Call Level Quality of Service Guarantees," *IEEE/ACM Trans. Networking* **6,** No. 2, 222–236 (1998).
60. Leonard Kleinrock, *Queueing Systems*, *Vol. II: Computer Applications*, Wiley-Interscience, New York, 1976.
61. Hamdy A. Taha, *Operations Research*, *An Introduction*, Macmillan Publishing Co., New York, 1992.
62. Moktar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty, *Nonlinear Programming Theory and Algorithms*, John Wiley, New York, 1993.
63. Jean-Yves Le Boudec, "The Asynchronous Transfer Mode: A Tutorial," *Computer Network & ISDN Syst.* **24,** 279–309 (1992).
64. Rainer Handel, Manfred N. Huber, and Stefan Schroder, *ATM Networks, Concepts, Protocols, Applications*, Addison-Wesley Publishing Co., Reading, MA, 1994.
65. P. Adam and J. P. Coudreuse, "Typical Network Applications of ATM," *L'echo des RECHERCHES*, English issue, pp. 73–81 (1991).
66. Ken-Ichi Sato, Satoru Ohto, and Ikuo Tokizawa, "Broad-Band ATM Network Architecture Based on Virtual Paths," *IEEE Trans. Commun.* **38,** No. 8, 1212–1222 (1990).
67. Joseph Y. Hui, "Resource Allocation for Broadband Networks," *IEEE J. Selected Areas in Commun.* **6,** No. 9, 1598–1608 (1988).
68. Aurel A. Lazar, Ariel Orda, and Dimitrios Pendarakis, "Virtual Path Bandwidth Allocation in Multiuser Networks," *IEEE/ACM Trans. Networking* **5,** No. 6, 861–871 (1997).

Received May 17, 1998; accepted for publication July 19, 2000

Imane Aly Saroit Ismail *11 El Falaky Street, El Mobtadayane, El Saida Zenab, Cairo, Egypt (permanent address); c/o Tarek Mohamed Ibrahim, Banque du Caire, Mailbox 533, Abu Dhabi, United Arab Emirates (current contact address) (iasi@emirates.net.ae).* Dr. Saroit Ismail is a member of the Faculty of Computers and Information in the Department of Information Technology at Cairo University. She received the B.Sc. in electronics and communication in 1985, and the M.Sc. in 1990 and the Ph.D. in 1994, both in computer science and computer networks at Cairo University. Her research interests include requirements analysis, resource allocation and utilization, switching, self healing, multicasting, security, and performance evaluation in high-speed computer communication networks (particularly ATM).