ELSEVIER

Computer Communications 24 (2001) 811–821

computer communications

www.elsevier.com/locate/comcom

QoS-aware routing in ATM and IP-over-ATM

R. Bolla[a], F. Davoli[a,*], M. Marchese[b], M. Perrando[a]

[a]*Department of Communications, Computer and System Science (DIST), University of Genoa, Via Opera Pia 13, 16145 Genova, Italy*
[b]*Italian National Consortium for Telecommunications (CNIT), University of Genoa Research Unit, Via Opera Pia 13, 16145 Genova, Italy*

Received 6 November 2000; accepted 6 November 2000

Abstract

This paper analyzes the performance of some dynamic routing algorithms for ATM and IP-over-ATM networks. The main algorithm, named AR-DLCP (Alternate Routing — Distributed Least Congested Path), is based on a distributed computation, where the "best" route is chosen node-by-node in the call set-up phase, by taking decisions on the basis of a cost function, composed of a local part and an aggregate part. The local cost is constructed for each outgoing link on the basis of the knowledge of the current and the maximum number of connections that the link can support, while still ensuring the required Quality-of-Service at the cell level. The aggregate cost is aimed at reflecting the congestion situation of a node and is computed through an information exchange mechanism among adjacent nodes. In a fully connected core network, only direct and two-hop paths are considered; in the general meshed topology case, paths are organized in a two-level hierarchy. Static and dynamic trunk reservation schemes that guarantee enough bandwidth to direct paths are discussed. The performance of the two alternatives is evaluated by simulation under various traffic load situations. In particular, AR-DLCP is compared with other algorithms already in the literature as DLCP, Learning Automata and RTNR (Real Time Network Routing). In the general topology case, the algorithm is also combined with a bandwidth allocation mechanism and IP routing, by taking into account the presence of best-effort traffic, in an IP-over-ATM context.

Keywords: Quality-of-Service; Routing; Asynchronous transfer mode

1. Introduction

Routing is one of the fundamental issues in telecommunication networks and, as such, has deserved a wide attention, in the context of both packet-switched and circuit-switched networks [3,12]. Though similarities exist, the two environments are characterized by specific algorithms and techniques; in both cases, however, some hierarchical structure is defined for very large networks. One specific aspect of routing in circuit-switched networks is the possible presence, within a certain hierarchical level, of high connectivity areas, where a direct path between switching nodes is chosen whenever possible and a two-hop alternate route is attempted only as a second choice.

The Asynchronous Transfer Mode (ATM) network inherits some features of both environments. The statistical multiplexing nature of ATM is an element in common with packet switching; however, provision of Quality-of-Service (QoS) and the related resource allocation mechanisms are (in some broad sense) more akin to the circuit-switching world. This aspect (see, for example, Ref. [16]) somehow characterizes the issue of routing in ATM, which has a more recent history. Moreover, the hierarchical structuring with different levels of aggregation is also present [2,11].

The routing problem in ATM is further connected with several other control schemes, among which Call Admission Control (CAC) and bandwidth allocation play a central role [14–17]. Another related issue is network dimensioning both from the physical and the virtual topology points of view.

For the purpose of the present work, we will not be concerned with the latter issue. Actually, we will consider fixed structures where we distinguish a fully connected core network (this does not imply necessarily physical connectivity, but just the presence of a Virtual Path (VP) between any two nodes) and an access network with generic meshed topology. In this context, however, the VP concept is not used as an additional routing option, but rather as a fixed given structure; in fact, we will not be concerned with finding the "best" virtual topology over the physical one. On the other hand, it will be possible to change the amount of bandwidth allocated to a given VP. In more detail, the

* Corresponding author. Tel.: +39-010-353-2732; fax: +39-010-353-2154.
E-mail addresses: lelus@dist.unige.it (R. Bolla), franco@dist.unige.it (F. Davoli), mario.marchese@cnit.it (M. Marchese), perr@dist.unige.it (M. Perrando).

0140-3664/01/$ - see front matter
PII: S0140-3664(00)00354-6

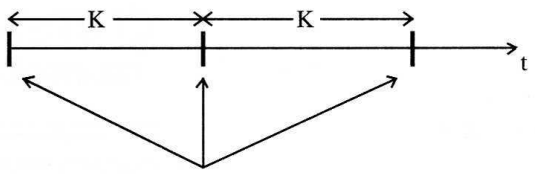

Fig. 1. Timing of the reallocation mechanism.

approach followed with respect to services with different statistical and performance characteristics (namely, Service Separation, as will be outlined in the following) will take us to dynamically redistributing the bandwidth among VPs dedicated to specific services. In the following, the term "link" will be used to indicate a generic Virtual Path Connection (VPC) between two switching nodes.

The routing strategies considered in the paper belong to the family of DLCP (Distributed Least Congested Path), previously defined by the authors (see, for example, Ref. [4]). A common feature of DLCP algorithms is that the "best" route is chosen step-by-step by a messenger packet upon which decisions are taken at each node traversed by using a function composed of a local cost and an aggregate cost. The local cost is constructed for each outgoing link on the basis of the knowledge of the current and the maximum number of connections that the link can support while still ensuring the required QoS at the cell level. The aggregate cost is aimed at reflecting the congestion situation of a node and of its neighboring area and it is computed through an information exchange mechanism among adjacent nodes.

The scope of the present work embraces both backbone and access networks. In the first category, it is currently customary to use routing strategies belonging to the family of Alternate Routing, where a direct path is attempted first, and an alternate route is chosen, according to some criterion, only if no direct path is available [12,16]. Therefore, we have decided to embed a DLCP algorithm in an alternate routing framework (AR-DLCP), a strategy already examined in Ref. [8]. On the other hand, in the access area, we suppose to use similar routing strategies, both for connection-oriented, QoS-aware traffic at the ATM call-level (including "long-lived" IP flows) and for connectionless "short-lived" IP flows. To do so, we suppose switching nodes to possess both IP and ATM switching capabilities. ATM VP/VC routing is used for traffic of the first type whereas datagrams of the second type are segmented into cells and transferred over ATM VPs between IP routers, where the datagram is reconstructed and routed accordingly. An adaptive capacity allocation between ATM and pure IP traffic is introduced in this case.

In applying AR-DLCP to the fully connected network, only direct and two-hop paths are considered, as in most parts of the literature (see, for example, Ref. [13]). This implies that the above-mentioned path-finding mechanism is applied only whenever a direct path is not available; moreover, the constraint on the number of hops greatly simplifies the task of the algorithm and avoids the use of the signaling procedure through the messenger packet. A further feature of AR-DLCP is that a sort of trunk reservation scheme is introduced in order to guarantee enough bandwidth to direct paths; the trunk reservation can be based on static or dynamic thresholds. In the case of a generic topology, AR-DLCP introduces a hierarchical choice of paths (and, correspondingly, of the links outgoing from a node), possibly with a limitation on the number of hops.

The description of the basic algorithm is given in Section 2 for both generic and full connectivity. The choice of a threshold value for trunk reservation is discussed in Section 3. The performance of AR-DLCP is evaluated by simulation in Section 4 under various traffic load situations and with two alternatives for trunk reservation. In this section, AR-DLCP is compared with other algorithms already in the literature such as DLCP [4], Learning Automata [10] and RTNR (Real Time Network Routing, [1]); moreover, an example of the performance of the algorithm in the general meshed situation with the presence of best-effort IP flows is provided. Section 5 contains the conclusions.

2. The routing algorithm

2.1. General framework and bandwidth allocation

As regards ATM flows requiring QoS (specifically, CBR and VBR services), the traffic is divided into H classes. Each class differs from the others for the required QoS, in terms of cell loss and delayed cell rate, and for statistical properties, such as average (or sustainable) and peak cell rate. Time is slotted with the slot equal to a cell transmission time. The source model for bursty sources (obviously including continuous-rate as a special case) used in the computations is an Interrupted Bernoulli Process (IBP), and the superposition of IBP sources is considered to derive QoS requirements at the cell level analytically (cell loss rate and delayed cell rate). The latter provide the basis for the Admission Control and bandwidth allocation rules. The actual source generation process used in the simulation runs is a (more general) superposition of Talkspurt–Silence models with deterministic generation of cells within a talkspurt [4]. The distribution of the connection time is assumed to be exponential.

Each class has a dedicated buffer and is assigned a fixed amount of bandwidth over each link as in a Complete Partitioning scheme under Service Separation [16]. However, we suppose that a dynamic fair assignment of the bandwidth partitions among the various classes of traffic is maintained by link bandwidth allocators [5,6], which periodically adjust the class bandwidth assignments to be kept over a fixed period of time. This means that a maximum number of acceptable calls can be chosen for each link and for each traffic class and it can be maintained until the next reallocation instant. The bandwidth reallocation timing

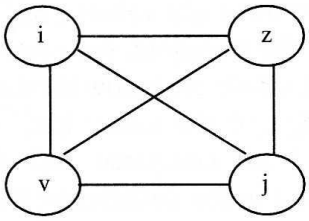

Fig. 2. A simple topology.

mechanism is periodic (with period K [slots] as in Fig. 1) in this approach, but this is not mandatory. We might as well have an asynchronous mechanism that works whenever necessary (e.g. when the ratio of blocked to offered calls for a certain class, measured over a time window, falls below a given threshold).

In the present work, we assume that the bandwidth is reallocated on direct links only if at least one connection having the link (VPC) endpoints as source and destination, respectively, is rejected in the previous K-slot period. In more detail and by limiting our consideration to the fully connected situation for simplicity, if there is a connection request between nodes i and j, the routing algorithm, as will be seen in Section 2.2, tries addressing the call on the direct link; if this is not possible, it chooses the "less saturated" two-hop path; if no two-hop route is available, the connection is rejected. Only in this case, the bandwidth on the link (VPC) ij will be reallocated. It is important to note that the bandwidth is not changed on two-hop (or more) paths. An example should help to better explain this situation. Let us suppose that the simple four-node network in Fig. 2 is used. At the proper instant, we allocate the new bandwidth partitions for each link on the basis of the behavior in the previous interval. Consider, for instance, link ij to be the one under analysis. The bandwidth is re-computed only if we have at least one connection with source i and destination j that is rejected. It has to be noted that, if link ij blocks connections as a part of a larger route (an alternative two-hop path, as zij, vij, ijv or ijz), this behavior does not affect the reallocation mechanism on link ij.

2.2. AR-DLCP in a fully connected network

We start our description of AR-DLCP (Alternate Routing — Distributed Least Congested Path) from the fully connected core network case.

In this situation, if there is a connection request of traffic class h between two generic nodes (for instance, source i and destination j), the scheme to decide the "best" route is structured in two steps:

(i) The number of connections in progress of traffic class h at instant (slot) k on link ij, namely $N_{ij}^{(h)}(k)$, is compared with the maximum number of calls acceptable on that link for traffic class h, decided at the last reallocation instant $s < k$, and denoted as $N_{\max,ij}^{(h)}(s)$. If the number of connections in progress on link ij for class h is lower than the maximum number of acceptable calls, the connection is accepted; otherwise we go to point. (ii) The value of the maximum number of acceptable calls on the link ij at the instant s ($N_{\max,ij}^{(h)}(s)$) depends on the reallocation method used. In any case, the "point" in the Call Space of the various classes represented by such values must be within a region, where QoS constraints at the cell level, expressed in terms of cells lost and delayed beyond a given threshold, are satisfied (Feasibility Region, FR). Section 2.4 contains some possible proposals. Anyway, the routing strategy is not dependent on the specific computation of $N_{\max,ij}^{(h)}(s)$.

(ii) *A two-hop path is chosen.* The choice of the alternative path is ruled by the measure of congestion of the route under analysis and by the utilization of a threshold (fixed or variable), which represents a sort of trunk reservation mechanism aimed at reserving a part of the bandwidth for direct links.

The congestion of each route is measured by using a cost, divided into two terms: a "local" one, weighting the run-time status of each link, concerning a specific traffic class, and an "aggregate" one, weighting the average congestion of a node and of its neighborhood.

Let i be the source and j the destination node; let q be the generic intermediate node between the source and the destination. The cost of the path from i to j (through q) for traffic class h at instant k is measured by

$$w_{iq}^{(h)}(k) = w_{iq,\mathrm{loc}}^{(h)}(k) + \alpha_q w_q^{(h)}(x) \tag{1}$$

where $w_{iq,\mathrm{loc}}^{(h)}(k)$ is the "local" cost and $w_q^{(h)}(x)$ is the "aggregate", non-real-time term, computed at instant $x < k$ and received by node i from node q. It is important to note that the cost is independent of the destination as should be clear from the definition of each term, reported in the following. α_q is a weighting coefficient to measure the "importance" of the delayed aggregate information; we have chosen to keep $\alpha_q < 1$, in order to limit the importance of the delayed information.

The local metric is defined as

$$w_{iq,\mathrm{loc}}^{(h)}(k) = \begin{cases} \dfrac{1}{N_{\max,iq}^{(h)}(s) - N_{iq}^{(h)}(k)} & \text{if } N_{iq}^{(h)}(k) < N_{\max,iq}^{(h)}(s) \\ Z & \text{if } N_{iq}^{(h)}(k) = N_{\max,iq}^{(h)}(s) \end{cases} \tag{2}$$

where Z is a very large value (Z should be large enough to ensure that no saturated link will be chosen if non-congested links are available; in the numerical results below, we have chosen $Z = 10$). Thus, the local cost is inversely proportional to the "available space" (in terms of the number of acceptable connections on the link) and it is Z when there is no more bandwidth available (i.e. when link iq is completely saturated, as regards class h traffic).

Regarding the aggregate cost of node q, we have defined

$$w_q^{(h)}(x) = \frac{1}{L_q} \sum_{n \in \mathrm{Succ}(q)} w_{qn,\mathrm{loc}}^{(h)}(x) \tag{3}$$

where Succ(q) is the set of links outgoing from node q, independent of the destination and Lq is its cardinality. Averaging over the L_q links is used to obtain an aggregate value of the costs attributed to the nodes (irrespective of the number of outgoing links and of the destinations). The aggregate cost is updated and passed among adjacent nodes every T seconds.

The complete independence of the destination might seem not to be so precise, but the operation of distinction among destinations is very heavy from a computational viewpoint and would negatively affect the scalability of the algorithm, especially in the generic topology case, to be discussed below. The use of a different cost for each destination would mean to have the same amount of information as a Real-Time Network Routing (RTNR), where a distinction among the various destinations is performed [1]. It is worth noting, however, that a modification in this sense (i.e. by distinguishing per-destination aggregate costs) might be introduced in the case of small networks in order to increase the precision of the algorithm.

After defining the metrics to measure the congestion, we have to define the strategy to choose the "best" two-hop path from i to j, i.e. an intermediate node between the source and the destination. To this aim, we introduce now a threshold mechanism, which in conjunction with the cost, plays the role of a trunk reservation strategy.

Let $\mathrm{thr}_{iq}^{(h)}(k)$ be a threshold value for each link ij and traffic class h, at a generic instant k. The threshold is not chosen for each source–destination since this would affect the scalability of the algorithm. Its choice follows the same philosophy as the one of the cost: as $w_{iq}^{(h)}(k)$ weighs the traffic conditions from i to all possible destinations through node q (for class h, instant k), in the same way, $\mathrm{thr}_{iq}^{(h)}(k)$ fixes a threshold value for every path from i to any destination through q. The value is dependent on the time and the traffic class (possible choices will be discussed in the next section).

Now, let Int(i,j) be the set of intermediate nodes between i and j and $S_k(i,j) = \{y : y \in \mathrm{Int}(i,j) \text{ AND } w_{iy}^{(h)}(k) \leq \mathrm{thr}_{iy}^{(h)}(k)\}$. Then, the routing scheme works as follows:

$$\begin{aligned} &\text{if } w_{iq}^{(h)}(k) \geq \mathrm{thr}_{iq}^{(h)}(k), \forall q \in \mathrm{Int}(i,j) \\ &\qquad \Rightarrow \text{the connection is rejected} \\ &\text{otherwise, } q = \underset{y \in S_k(i,j)}{\mathrm{argmax}} \left[\mathrm{thr}_{iy}^{(h)}(k) - w_{iy}^{(h)}(k)\right] \text{ is chosen} \end{aligned} \tag{4}$$

For instance, let us consider the same simple network as in Fig. 2. If there is a call request from i to destination j, for class h, at instant k, the direct route is checked first; if it is congested, there are two other possibilities to get to the destination: through node z (route izj) and through node v (route ivj). The cost values ($w_{iz}^{(h)}(k)$ and $w_{iv}^{(h)}(k)$) are compared with the threshold values ($\mathrm{thr}_{iz}^{(h)}(k)$ and $\mathrm{thr}_{iv}^{(h)}(k)$). If both are above the threshold, the connection is rejected. On the other hand, if both are below the threshold (i.e. both z and v belong to $S_k(i,j)$), the values $[\mathrm{thr}_{iz}^{(h)}(k) - w_{iz}^{(h)}(k)]$ and $[\mathrm{thr}_{iv}^{(h)}(k) - w_{iv}^{(h)}(k)]$ are compared and the largest value (the one whose cost is the farthermost from the threshold) is chosen.

So, even if there is some room left on a two-hop path to go from i to j, not necessarily the bandwidth is utilized; a trunk reservation scheme, based on the threshold value, is in fact applied to reserve the bandwidth for direct connections.

It should anyway be observed that even if the route is chosen from the source, a connection request could be blocked at the intermediate node. The traffic conditions may be changed because the aggregate information is not run-time or they may not have been properly estimated by Eq. (3), which does not distinguish among the destinations. For these reasons, the link chosen might be saturated (i.e. the number of connections in progress might equal the maximum number of acceptable calls); in this case, the call request is rejected at the intermediate node.

2.3. AR-DLCP in a generic topology

When the network is not completely connected (in terms of VPs), a pair of nodes may not be joined by a direct link. AR-DLCP routing is still applicable in this situation with the same admission control rules outlined previously, provided a hierarchy of paths is introduced. More specifically, we may distinguish two levels: (i) first choice paths, with the minimum number of hops; (ii) second choice paths (all others). The distinction should be operated at each node for each destination. It requires a certain amount of additional topological information with respect to the previous situation:

(i) The number of hops of the shortest path from source to destination.
(ii) The number of hops already done.
(iii) The minimum number of hops to reach the destination from a neighboring node in the set of eligible ones (i.e. those nodes through which the destination can be reached) by excluding paths that contain nodes already traversed.

The algorithm operates as follows:

- All first choice links (the outgoing links of a first choice path) are checked and the minimum cost link that allows to allocate the necessary resources (if any) is chosen;
- If no first choice link is available, the minimum cost link among the remaining ones is chosen;
- If no second choice link is available, the call is rejected.

In the above steps, further constraints may be introduced, in order to avoid paths with too many hops; besides fixing an upper bound, one can impose that, once a second choice

path has been taken, all successive selections must be limited to first choice ones. Obviously, the number of hierarchical levels may be increased to more than two.

As regards the expression of the aggregate cost, in this case it may be worth introducing an additional term in Eq. (3), trying to capture the congestion situation of distant nodes more effectively. More specifically, also the aggregate costs of neighboring nodes may be taken into account, by using the following expression (which is the general DLCP aggregate cost definition), instead of Eq. (3):

$$w_q^{(h)}(x) = \frac{1}{L_q} \sum_{n \in \mathrm{Succ}(q)} w_{qn,\mathrm{loc}}^{(h)}(x) + \frac{\beta_q}{L_q} \sum_{n \in \mathrm{Succ}(q)} w_n^{(h)}(x-1) \tag{5}$$

Here, β_q is another weighting coefficient; obviously, the above expression reduces to Eq. (3) if $\beta_q = 0$. Note that, if the maximum number of hops is two, the additional term makes little sense and therefore it has not been considered in Eq. (3). In any case, the rationale behind expressions (3) and (5) is to try to capture (with some unavoidable delay) the information on downstream congestion, which is propagated by the periodic exchange of the aggregate costs in a similar fashion to the propagation of shortest distances in distance vector routing algorithms. Expression (5) introduces a recursion in the calculation of the aggregate costs. The conditions for its convergence in a static situation (i.e. temporarily fixed values of the connections in the whole system) are derived in Appendix A.

2.4. Bandwidth allocation strategies

The definition of the maximum number of acceptable calls on a link at a specific instant of time depends on the bandwidth allocation strategy chosen. Ref. [6] contains some possible proposals. The strategy varies in dependence of the aim.

The results reported in this paper refer to a method called Dynamic Reallocation Scheme (DRS), concerning the backbone network, i.e. the results when no IP traffic is involved (Section 4.1). The choice is not mandatory. Any other method may be used. Other strategies have been tested but they did not provide results deeply different from DRS, at least concerning the parameters adopted. For the sake of simplicity, just one strategy has been presented.

The DRS is based on a Complete Partitioning strategy. Two controls are applied, in a two-level hierarchical structure, one of them acting on the scheduler that serves the link buffer and the other on the admission of connection requests. At the higher level of the hierarchy, the bandwidth allocation controller periodically reassigns (by minimizing a suitable cost function) bandwidth partitions to the traffic classes, whose sum amounts to the total link bandwidth (the partitions may be interpreted as Virtual Paths, one for each traffic class). The scheduler receives the values of the partitions and must ensure that each traffic class is assigned the necessary slots accordingly. At the lower level, independent access controllers for each class decide upon the acceptance of incoming connection requests, by computing the maximum number of calls that the class can support, given the assigned bandwidth.

The allocation strategy when also best-effort connectionless IP traffic is involved deserves a particular attention. It derives from the studies in Refs. [7,9]. The essential contents are summarized in the following.

The bandwidth partitions now include a portion reserved for the IP traffic in order to still guarantee a certain level of QoS to the ATM connection-oriented flows and, at the same time, to assure a minimum quality also to the best-effort IP flow, which is supposed to be offered ABR or even UBR service. The presence of the additional portion of reserved bandwidth requires a change in the strategy adopted by the bandwidth allocation controller. More specifically, an upper bound C_{co} may now be computed on the bandwidth assigned to connection-oriented, QoS-aware traffic. As in a movable boundary scheme, the connectionless packets will utilize whatever bandwidth is available, but will nevertheless be guaranteed the minimum amount deriving from the difference between the link bandwidth and the upper bound. For each possible value of such "basic" partition, two cost functions can be defined: $J_1(C_{\mathrm{co}})$, referring to connection-oriented traffic and $J_2(C_{\mathrm{co}})$, referring to connectionless traffic, respectively. Function $J_1(C_{\mathrm{co}})$ can represent a specific allocation criterion within the class of Complete Partitioning strategies considered in Ref. [6]; we have chosen here the maximum blocking probability over the connection-oriented classes (as in the previous case, the particular choice of the allocation strategy has the value of an example). It is worth noting that, within the Feasibility Region, owing to the Service Separation assumption, the blocking probability for each class can be simply represented by the Erlang B formula [16]. Thus, given a value of C_{co}, the maximum number of acceptable connections for each class can be derived by the minimization of J_1. On the other hand, for the same value of C_{co}, an expression for function J_2 can be constructed, to reflect the loss probability of the cells generated by the segmentation of the IP packets, which are supposed to feed a common buffer, served by the residual capacity. An analytical approximation of this loss probability can be constructed by using results from a self-similar model of IP traffic [7].

At this point, a global cost function, to be minimized for the "optimal" choice of C_{co} can be defined as

$$J(C_{\mathrm{co}}) = J_1(C_{\mathrm{co}}) + \sigma J_2(C_{\mathrm{co}}) \tag{6}$$

The influence of the weighting coefficient σ will be evidenced in Section 4.2. It must be noted that in this case, the strategy applied to route the data flows should try to avoid oscillatory behaviors, caused by the dynamic interaction between the routing and reallocation algorithms. In the present work, we have adopted a very straightforward solution to this problem. We use a shortest path algorithm for the IP data flows with a fixed metric (namely, minimum

Table 1
Choice of the dynamic threshold

Value of the estimated cost	Threshold
$0 < \hat{w}_{iq}^{(h)}(k) \leq 0.0625 + \alpha_q \times 0.0625$	20
$0.0625 + \alpha_q \cdot 0.0625 < \hat{w}_{iq}^{(h)}(k) \leq 0.125 + \alpha_q \times 0.125$	10
$0.125 + \alpha_q \times 0.125 < \hat{w}_{iq}^{(h)}(k) \leq 0.25 + \alpha_q \times 3.0$	0.7
$0.25 + \alpha_q \times 3.0 < \hat{w}_{iq}^{(h)}(k) \leq 0.5 + \alpha_q \times 3.0$	0.3
$\hat{w}_{iq}^{(h)}(k) > 0.5 + \alpha_q \times 3.0$	0.2

number of hops) and we let the reallocation procedure ensure enough bandwidth over the links by driving connection-oriented traffic in other directions. In summary, the bandwidth allocation procedure works on-line by dynamically adjusting (with the same timing procedure described in Section 2.1) the parameter C_{co}, as well as the bandwidth partitions among QoS classes. In doing so, it influences the behavior of two different routing algorithms, operating at the ATM and the IP level, respectively.

3. "Trunk reservation" thresholds

The choice of the thresholds that have been introduced above may heavily affect the performance of the routing algorithm. In particular, the threshold may be static or dynamic. Concerning the first case, the choice can be made only on the basis of practical experience. A good choice depends on the type and distribution of the traffic and if we have no precise information about the network behavior, a threshold well suited for any application is difficult to find. If a very low threshold is chosen, we reserve much bandwidth for direct links, so obtaining good performance only at high-traffic load; on the other hand, with a high threshold value, we do not reserve room for direct connections and we get satisfying performance at average load. In general, the choice may be more sensitive in our case than in a real trunk reservation mechanism [16], as the bandwidth is not reserved directly, but rather implicitly through the cost, which has dynamically varying components. However, due to the nature of our routing metrics, which is partially based on the "instantaneous" occupation of the links, we have preferred to embed the protection of the direct paths within the routing algorithm itself.

As far as the dynamic value is concerned, several alternatives can be conceived; a possible proposal is reported below and its performance will be investigated in Section 4.

In this case, an estimation procedure of the number of connection requests in the future for a certain link and traffic class is necessary. Let $\hat{N}_{iq}^{(h)}(k)$ be the expected traffic load (in Erlangs) over the average length of a connection, for link ij and traffic class h, measured at instant k. The details of the estimation algorithm of $\hat{N}_{iq}^{(h)}(k)$ are not reported here. In general, the algorithm can be based on the measure of the average connections in progress and of the call blocking rate, over the previous reallocation interval. From $\hat{N}_{iq}^{(h)}(k)$, it is possible to get an estimated cost $\hat{w}_{iq}^{(h)}(k)$, which can be interpreted as in Eq. (1):

$$\hat{w}_{iq}^{(h)}(k) = \hat{w}_{iq,\mathrm{loc}}^{(h)}(k) + \alpha_q \hat{w}_q^{(h)}(x) \tag{7}$$

where the quantities in Eq. (7) are obtained from Eqs. (2) and (3), by substituting $N_{iq}^{(h)}(k)$ with $\hat{N}_{iq}^{(h)}(k)$.

The computation of the estimated cost, repeated at each connection request for the paths under analysis, allows to fix the threshold, whose numerical values (Table 1) have been chosen from simulation results on a five-node completely connected network under various load conditions. The choice is reported in Table 1.

In this way, if the estimated traffic load is very high, most bandwidth is reserved for direct links and a very low threshold is chosen; on the other hand, if a low traffic is forecast, a high value of the threshold is decided.

The choice of the threshold described is obviously a heuristic one. Anyway, it has a precise physical meaning. If we match the numerical values with the cost structure, we can note that value 0.125 for the local cost means that eight connections is the free space left on that link for that traffic class, i.e. the difference between the maximum number of acceptable calls and the number of connections in progress.

Table 2
Parameter values

Traffic class: h	$h = 1$	$h = 2$
Peak bandwidth: $p^{(h)}$	1 Mbit/s	2 Mbit/s
Burstiness: $b^{(h)}$	2	5
Average burst length: $B^{(h)}$	100 cells	500 cells
Average connection duration	20 s	15 s
P_{loss} upper bound: $\epsilon^{(h)}$	0.0001	0.0001
P_{delay} upper bond: $\delta^{(h)}$	0.001	0.001
Delay constraint: $D^{(h)}$	400 slots	200 slots
Buffer length: $Q^{(h)}$	20 cells	15 cells

4. Numerical simulation results

4.1. Backbone network

In this section, we report some simulation results on several network configurations. Only the connection-oriented traffic is considered. We use a simple five-node test network in order to obtain some indications on the performance of the proposed routing scheme and to compare them with other possible solutions. Two traffic classes ($H = 2$), a "reallocation interval" $K = 8 \times 10^7$ cells, an updating time $T = K/10$ and bi-directional links of capacity $C = 50$ Mbit/s, have been used in this case. The quantities $\rho^{(h)}$ [Erlangs], $h = 1, \ldots, H$, represent the global average traffic intensities offered to the network;

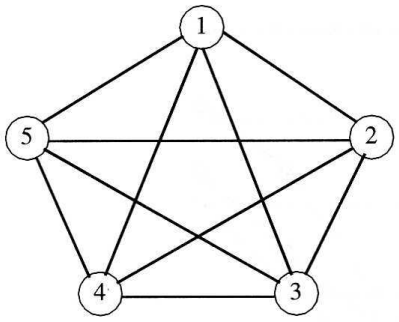

Fig. 3. Topology of the first test network.

the call arrival processes follow independent Poisson distributions. Moreover, all connections are supposed to be full duplex with the same characteristics in both directions. All other parameter values are shown in Table 2, where P_{loss} and P_{delay} represent the loss probability and the probability of exceeding a given delay threshold (D), respectively, in the link buffers.

We refer to a "reference" traffic flow generated by the above data as an offered load 1 when $\rho^{(1)} = 180$, $\rho^{(2)} = 160$. An offered load "x" corresponds to the same data, except for the traffic intensities $\rho^{(h)}$, $h = 1, 2$, which are multiplied by x. The coefficients α_i $i = 0, \ldots, 6$, are considered to be the same for each node, i.e. $\alpha_i = \alpha$, $\forall i$; the coefficients β_i, $\forall i$, are set to 0, being the network completely connected.

The topology of the network that has been used in the simulations is shown in Fig. 3; it contains five nodes, completely connected (i.e. there is a direct link between each pair of nodes).

Two different scenarios are considered in the following: (i) the balanced load, where every node in the network generates the same portion of external traffic and the destination is chosen among all the nodes from a uniform distribution; (ii) the unbalanced load, where again every node in the network generates the same portion of external traffic but 50% of the generated traffic is directed to node 4, while the remaining one is uniformly distributed among the other nodes. All simulations extend over a time period such that the 95% confidence interval is less than 3% of the value to be estimated. The results obtained are compared with three other routing strategies already in the literature: a simple Hot-Potato, which may be regarded as a performance lower bound, a Learning Automata [10] and the Real Time Network Routing (RTNR) [1] strategy. Even if present in the literature, the last two strategies may deserve a short explanation. In any case, all these routing algorithms are applied in a connection-oriented context during the call set-up phase.

Learning Automata is a strategy where, as in our algorithm, the direct link is checked firstly. If it is not available, a two-hop path is sought. A probabilistic value is associated to each intermediate node between the source and the destination (i.e. to each alternative path). The updating of the probabilities is ruled by the behavior of the path in the past: If a call is accepted, the probability that this route will be chosen in the future increases; otherwise the probability of being chosen decreases.

Also in the RTNR case, a two-hop path is sought only if the direct path is not free. When a two-hop path is necessary, the source node requires a list containing the status of all links from the destination. The source knows the status of all its outgoing links. Thus, performing a logic operation, the congestion of each source–destination path is verified and the best path chosen. Various levels of "saturation" may be selected to measure the congestion. This choice is made by estimating some quantities (like the number of connections rejected and accepted) for each node and each specific destination. The RTNR routing is very effective because continuously updated information is available but, being dependent on the destination chosen, it is not efficiently scalable and very difficult to implement in large networks. In this case, the amount of information that moves throughout the network grows more and more with the network

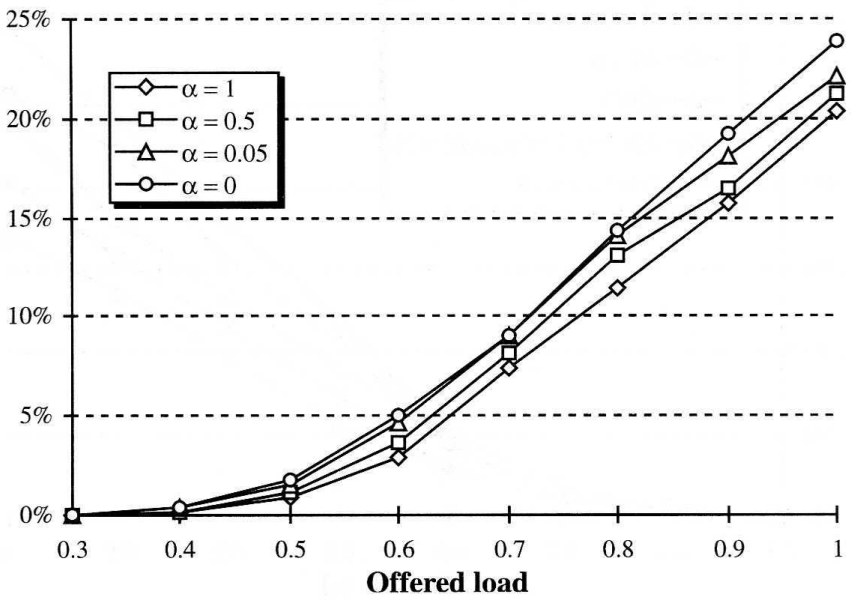

Fig. 4. Percentage of blocked calls versus traffic load, DT-AR-DLCP.

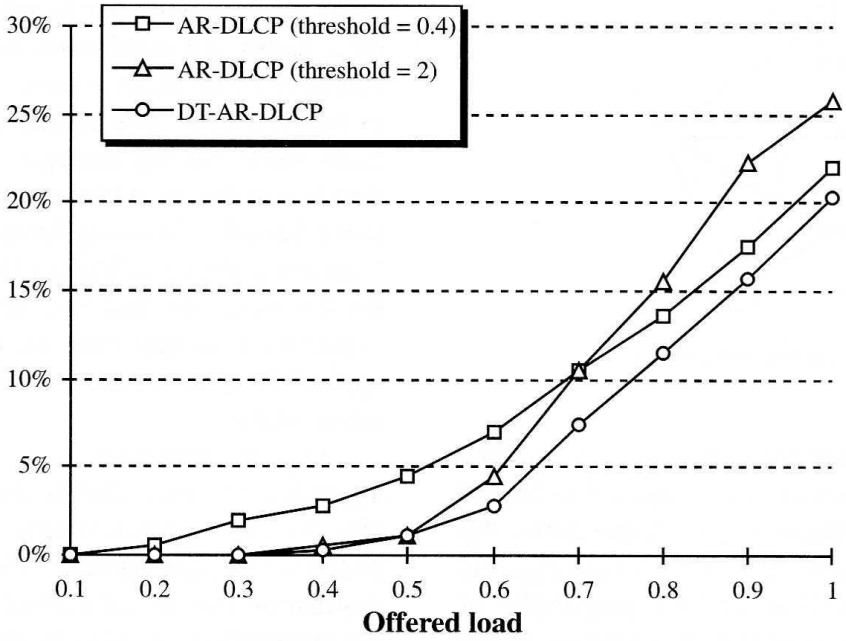

Fig. 5. Percentage of blocked calls versus traffic load, $\alpha = 1$, balanced load scenario.

dimension as well as the transmission delay problems when the status list is required by the source. There are some methods to reduce the complexity of RTNR, but in general, it is not really applicable in large high-speed networks. Anyway the results it provides may be regarded as a performance upper bound for routing strategies in ATM.

All the graphs in the following report the overall percentage of blocked calls (for both traffic classes) versus offered traffic load. Fig. 4 shows the Dynamic Threshold AR-DLCP (DT-AR-DLCP) behavior for some values of α, in the balanced load scenario. It appears that the most effective value is $\alpha = 1$, which corresponds to giving the same weight to the local and the aggregate cost parts.

Fig. 5 reports a comparison among different types of AR-DLCP. More in detail, the AR-DLCP with two different fixed thresholds (namely, 0.4 and 2) and the dynamic threshold version are compared. The graph shows that, by using a fixed threshold, for large load values the largest threshold works better, while below the load value 0.7 the smaller one is the best choice. On the other hand, the DT-AR-DLCP, by adapting its parameters dynamically, is able to obtain always the best results.

Figs. 6 and 7 show the performance of AR-DLCP (with a fixed threshold of 2), Dynamic Threshold AR-DLCP (DT-AR-DLCP), the Real Time Network Routing (RTNR), the Learning Automata (AUTO) and the Hot-Potato in the balanced and unbalanced load scenario, respectively. In the balanced situation, the DT-AR-DLCP shows a behavior not far from that of RTNR and clearly better than the others. In the unbalanced load scenario, except for the hot-potato, which always has the worst performance, all algorithms have the same behavior owing to the heavy use of two-hop routes.

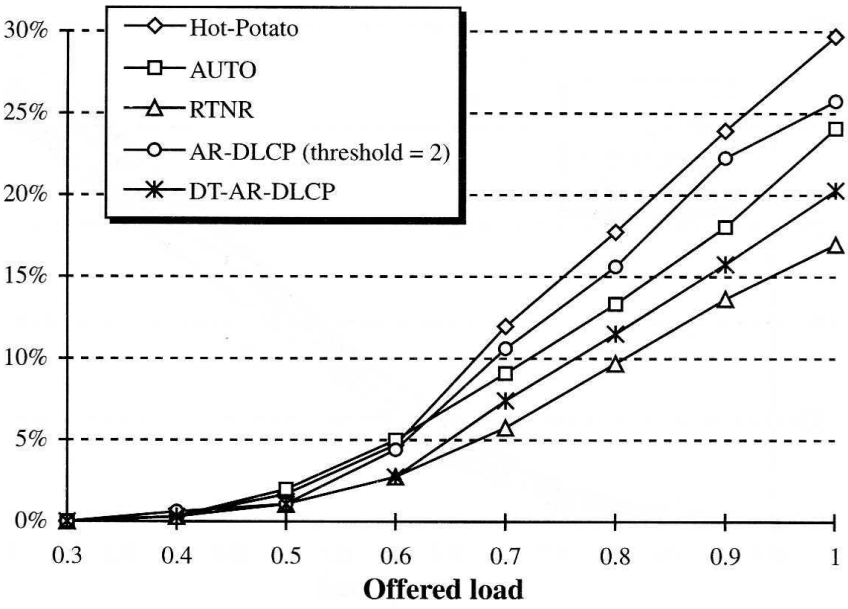

Fig. 6. Percentage of blocked calls versus traffic load, $\alpha = 1$, balanced load scenario.

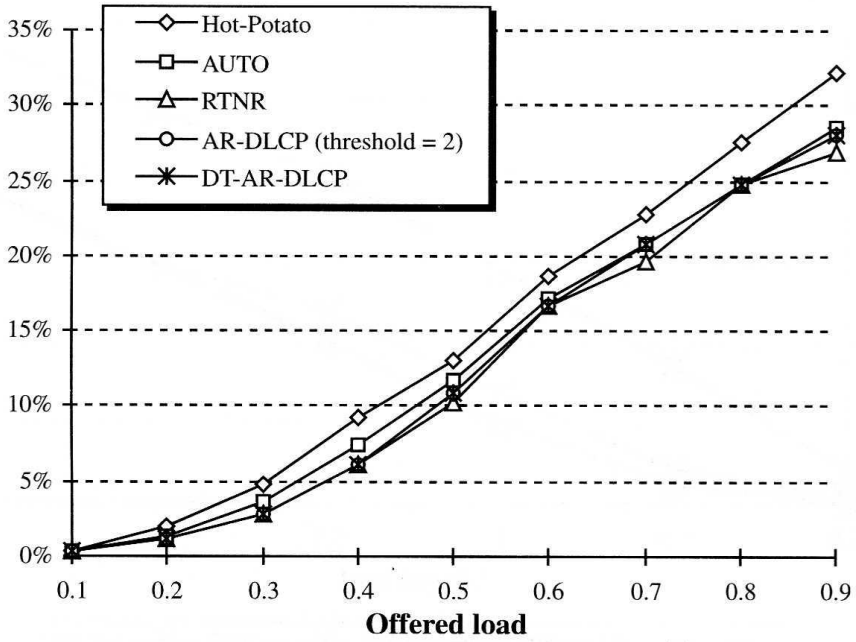

Fig. 7. Percentage of blocked calls versus traffic load, $\alpha = 1$, unbalanced load scenario.

4.2. Access network

The behavior of the algorithm in a non-completely connected topology and the presence of best-effort IP flows have been tested with the network of Fig. 8. All coefficients have been chosen equal ($\alpha_i = \beta_i = 0.5\ \forall i$). Only one traffic class (CBR calls with 1 Mbit/s bit rate) is present and the threshold is not used. The traffic distribution in this network has been chosen to be particularly unbalanced and a bottleneck has been created. More specifically, nodes 0, 1 and 10 are the only ones generating connection-oriented traffic. Nodes 0 and 1 generate 80% of the total load (20% between each other and 80% toward node 10); the remaining 20% of the connection-oriented traffic is generated from node 10, and it is equally destined to nodes 0 and 1. This distribution aims at reproducing a situation where most traffic is concentrated toward a single destination and the network presents a single primary path plus a number of equivalent secondary ones. The best-effort traffic flows from node 0 to node 10. All links have a capacity of 150 Mbit/s, except the ones evidenced in Fig. 8 (links 6–9, 6–8, 9–8 and 8–9), where a 50 Mbit/s capacity simulates a local congestion. In this way, most of node 0's secondary channels are free, whereas node 1 finds most of its secondary channels congested.

This section is dedicated to analyze the percentage of blocked calls and the percentage of best-effort traffic packet loss by varying the weighting coefficient σ, which governs,

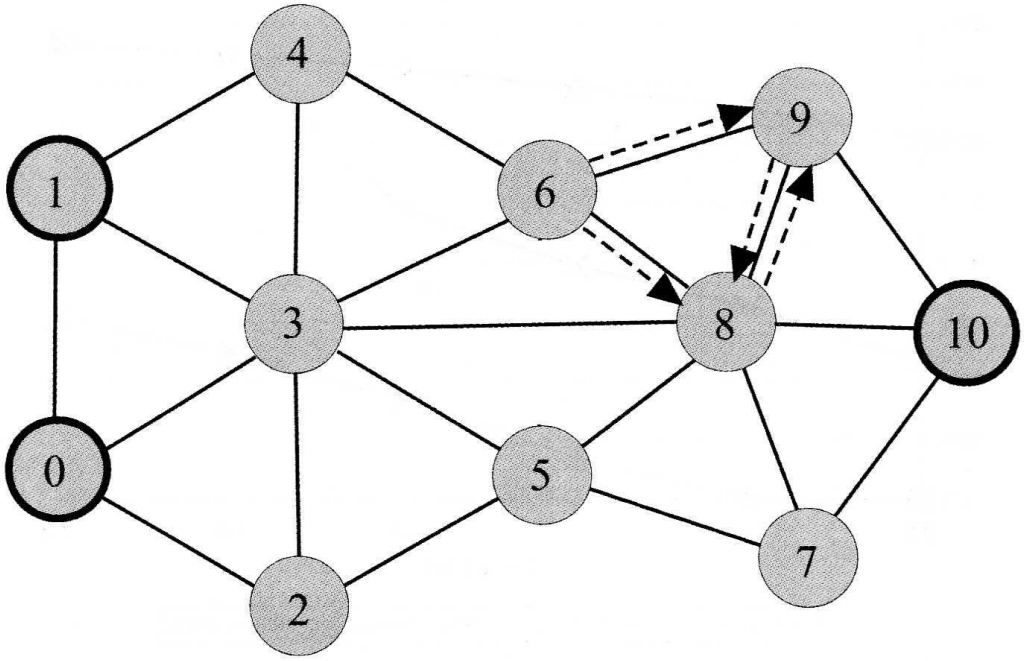

Fig. 8. Generic test network.

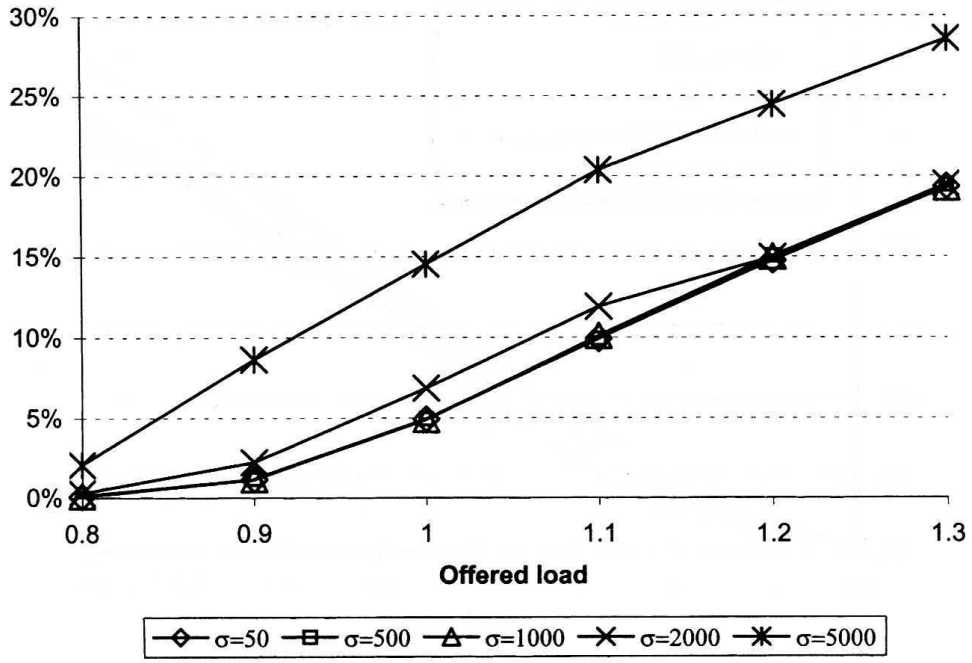

Fig. 9. Percentage of blocked calls versus traffic load, access network.

as underlined in Section 2.4, the importance of the best-effort traffic in the global allocation mechanism.

Fig. 9 contains the percentage of blocked calls of the ATM traffic. Fig. 10 reports the percentage of packet loss. Both graphs are depicted versus an increasing load in the network. The value 1 of the load refers to an overall load of 600 Erlangs for the connection-oriented traffic and of 50 Mbit/s for the best-effort flow; for the other values of network load, the two flows are scaled in the same proportions. The constraint on the precision of the simulation results is the same imposed before. The model used for the IP traffic is a self-similar one, derived from the superposition of ON–OFF flows with Pareto-distributed ON time [7].

Clearly, the blocked calls increase for larger values of σ, while the lost packets decrease. The interesting aspect of the two figures is that the effect of σ is not relevant up to a value of 1000. Only larger values of the weighting coefficient have a real impact over the performance. In more detail, it is interesting to observe the results deriving from $\sigma = 2000$, which assures a low percentage of blocked calls (only slightly higher than the percentage guaranteed by the lower values of σ) together with a sensible decrease of the packet loss for the best-effort flow.

5. Conclusions

A routing algorithm for ATM networks has been defined

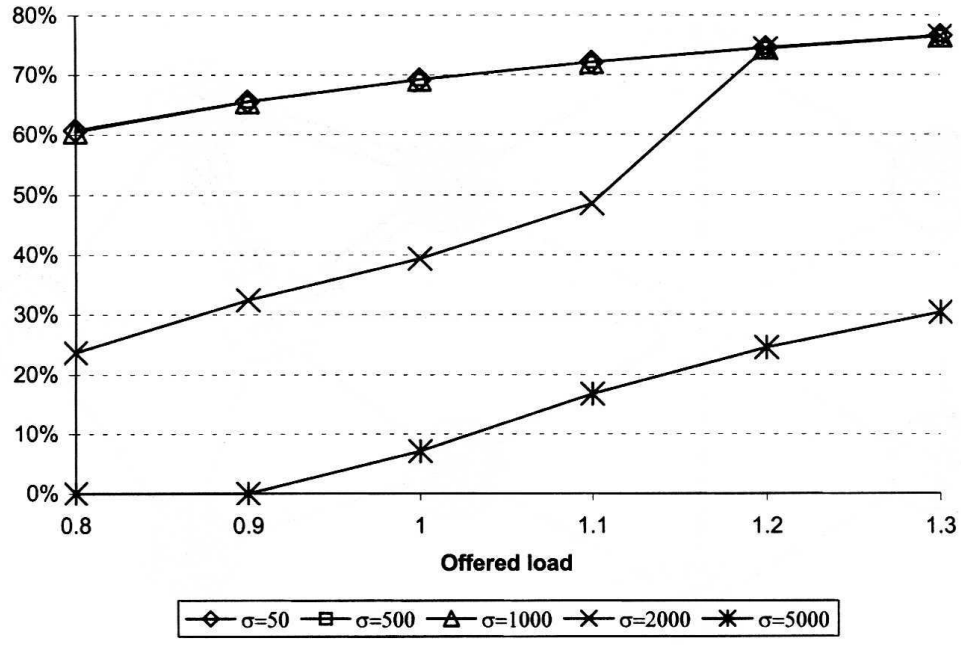

Fig. 10. Percentage of lost packets versus traffic load, access network.

in the paper. The traffic is divided into classes, which are assigned a certain amount of bandwidth on each link over a given time period. In the completely connected situation (a VPC between any pair of switching nodes), the algorithm considers only direct and two-hop routes. The preferred choice is always the direct path; a two-hop path is chosen only when not enough bandwidth is available on the direct link to accept the call and maintain the necessary QoS. The choice of the two-hop route is made by ranking the various possibilities according to a dynamic cost value, which is attached to the originating and the intermediate node pair. The cost itself is made up by two terms: an "instantaneous" local one and an "aggregate" one, which is passed along by the nodes periodically. In order to protect the direct links from saturation, a threshold is set on the usage of the alternate routes, which can be chosen statically or dynamically. In the case of networks with general meshed topology, the algorithm ranks the possible paths hierarchically as first and second choice and adds a term to the aggregate cost, whose aim is to better reflect the congestion state of distant nodes (in other words, to improve the "visibility" of DLCP).

Simulation results have been reported and discussed, which are aimed at suggesting the best choice for some parameters of the scheme and at comparing it with different ones. The results have been divided into two parts. The first one concerns a fully connected backbone network. Only the ATM-guaranteed traffic has been considered in this case. The overall behavior appears to be quite satisfactory and very close to that of the RTNR routing scheme. The second part refers to the access network. The guaranteed and the best-effort portions share the network. The "importance" of the best-effort traffic is ruled by a weighting coefficient, whose tuning is the object of the part dedicated to the access network. The most meaningful result is that it is possible to find particular values of the weighting coefficients that assure a low percentage of blocked calls as well as a low percentage of lost packets concerning the best-effort portion.

Appendix A

We want to derive the conditions under which the recursion underlined by Eq. (5), i.e.

$$w_q^{(h)}(x) = \frac{1}{L_q} \sum_{n \in \mathrm{Succ}(q)} w_{qn,\mathrm{loc}}^{(h)}(x) + \frac{\beta_q}{L_q} \sum_{n \in \mathrm{Succ}(q)} w_n^{(h)}(x-1),$$

$$q = 1, \ldots, N \qquad \text{(A1)}$$

(where N is the total number of nodes in the network) is stable, in the sense that it converges to a finite value if the number of connections does not change for a sufficiently long time.

To this aim, let us drop the index h for simplicity, and let $W_{\mathrm{loc}} = (w_{qn,\mathrm{loc}})$ be the matrix of the local costs (supposed to have 0 qn element if nodes q and n are not connected) and $W_{\mathrm{agg}} = \mathrm{col}[w_1, \ldots, w_N]$ the vector of the aggregate costs. Moreover, let

$$L = \mathrm{diag}[1/L_1, \ldots, 1/L_N] \qquad L_\beta = \mathrm{diag}[\beta_1/L_1, \ldots, \beta_N/L_N] \qquad \text{(A2)}$$

be two diagonal matrices and C the network connectivity matrix. Then, we can write Eq. (A1) in compact form as

$$W_{\mathrm{agg}}(x) = L_\beta C W_{\mathrm{agg}}(x-1) + LW\underline{1} \qquad \text{(A3)}$$

where $\underline{1}$ is the vector of all 1's. Eq. (A3) represents a discrete time linear system with a constant input, which converges to a constant value if the eigenvalues of the $L_\beta C$ matrix are within the unit circle.

References

[1] G.R. Ash, Real-time network routing in a dynamic class of service network, Proc. ITC 13, Copenhagen, Denmark, 1991.

[2] ATM Forum, PNNI SWG 94-0471R13, ATM Forum PNNI Draft Specification, 1996.

[3] D. Bertsekas, R. Gallager, Data Networks, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[4] R. Bolla, A. Dalal'ah, F. Davoli, M. Marchese, Dynamic route selection at call set-up level in ATM networks, in: D.D. Kouvatsos (Ed.), ATM Networks Performance Modeling and Analysis, vol. 3, Chapman and Hall, London, 1997, pp. 121–140.

[5] R. Bolla, F. Davoli, M. Marchese, Bandwidth allocation and admission control in ATM networks with service separation, IEEE Commun. Mag. 35 (5) (1997) 130–137.

[6] R. Bolla, F. Davoli, M. Marchese, Distributed dynamic routing of virtual circuits in ATM networks under different admission control and bandwidth allocation policies, Int. J. Parallel Distrib. Syst. Netw. (special issue on ATM switching/networking architectures and performance) 2 (4) (1999) 225–234.

[7] R. Bolla, F. Davoli, S. Ricciardi, A hierarchical control structure for multimedia access networks, Proc. IEEE Int. Conf. Commun. (ICC'99), 1999, pp. 1320–1325.

[8] R. Bolla, P. Castelli, F. Davoli, M. Marchese, Analysis of a distributed alternate routing strategy for ATM networks, Proc. Symp. Perf. Eval. Comput. Telecommun. Syst. (SPECTS'2000), Vancouver, BC, Canada, 2000, pp. 44–51.

[9] R. Bolla, F. Davoli, M. Marchese, M. Perrando, Call admission control and routing of QoS-aware and best-effort flows in an IP-over-ATM networking environment, to be presented at International Workshop on QoS in Multiservice IP Networks (QoS-IP 2001), Rome, Italy, 2001.

[10] A.E. Economides, P.A. Ioannou, J.A. Silvester, Decentralized adaptive routing for virtual circuit networks using stochastic learning automata, Proc. Infocom'88, 1988, pp. 613–622.

[11] E. Felstaine, R. Cohen, On the distribution of routing computation in hierarchical ATM networks, IEEE/ACM Trans. Netw. 7 (6) (1999) 906–916.

[12] A. Girard, Routing and Dimensioning in Circuit-Switched Networks, Addison-Wesley, Reading, MA, 1990.

[13] S. Gupta, P.P. Gandhi, Dynamic routing in multi-class non-hierarchical networks, Proc. Int. Conf. Commun., 1994, pp. 1390–1394.

[14] D.E. McDysan, D.L. Spohn, ATM Theory and Applications, McGraw-Hill, New York, NY, 1999.

[15] J.W. Roberts (Ed.), COST 224 — Performance Evaluation and Design of Multiservice Networks, Commission of the European Communities, Brussels, 1992.

[16] K.W. Ross, Multiservice Loss Models For Broadband Telecommunication Networks, Springer-Verlag, London, 1995.

[17] M. Schwartz, Broadband Integrated Networks, Prentice-Hall, Upper Saddle River, NJ, 1996.