Analytical/Simulation Optimization System for Access Control and Bandwidth Allocation in IP Networks with QoS

R. Bolla, R. Bruschi, F. Davoli, M. Repetto
Department of Communications, Computer and Systems Science (DIST)
University of Genoa
Via Opera Pia 13, I-16145 Genova, Italy
Email: lelus,franco,repetto,roberto.bruschi@dist.unige.it

Abstract— **A multiservice IP network based on the DiffServ paradigm is considered, composed by Edge Routers (ER) and Core Routers (CR), forming a domain that is supervised by a Bandwidth Broker (BB). The traffic in the network belongs to three basic categories: Expedited Forwarding (EF), Assured Forwarding (AF) and Best-Effort (BE). Consistently with the DiffServ environment, CRs only treat aggregate flows; on the other hand, ERs keep per-flow information (from external sources or other network Domains), and convey it to the BB, which knows at each time instant the number (and the bandwidth requirements) of flows in progress within the domain for both EF and AF traffic categories. A global strategy for admission control, bandwidth allocation and routing within the domain is introduced and discussed in the paper. The aim is to minimize blocking of the "guaranteed" flows (EF and AF), while at the same time providing some resources also to BE traffic. As the main objective is to apply the above mentioned control actions on-line, a computational structure is sought, which allows a relatively fast implementation of the overall strategy. In particular, a mix of analytical and simulation tools is applied jointly: "locally optimal" bandwidth partitions for the traffic classes are determined over each link (which set the parameters of the link schedulers in the CRs, the routing metrics and admission control thresholds); a "high level" simulation (involving no packet-level operation) is performed, to determine the per-link traffic flows that would be induced by the given allocation; then, a new analytical computation is carried out, followed by a simulation run, until the allocations do not significantly change. The possible convergence of the scheme (under a fixed traffic pattern) is investigated and the results of its application under different traffic loads are studied on a test network.**

Keywords- IP DiffServ; QoS-IP; Resource Allocation.

I. INTRODUCTION

Today's Internet is a widely deployed network with hundreds of millions of hosts, yet growing in size [1], connected by a network architecture belonging to different organizations. This results in a hierarchical organization, with the ASs (*Autonomous Systems*) at the top and interconnected to each other to form the backbone of the net. The internal structure of ASs can be also hierarchical, reducing the number of peers in every level, with many management benefits, such as simplifying IP addresses assignment to secondary Internet Service Providers (ISP) which also reduce IP routing table size, limiting control traffic to smaller areas, and so on. This is very simple in IPv6 [2], where the increased size of the address field allows great flexibility in assigning and partitioning the whole space. As an example of how a huge network can affect performance, we cite inter-domain (among ASs) routing, where BGP often fails to assure route stability and routing convergence [3]-[5].

For these reasons, we choose a hierarchical structure for our approach, and we assume to have few nodes on each level. At the bottom level, nodes correspond to hosts; at higher levels, nodes correspond to aggregates of nodes at lower levels. Such architecture is very flexible: the same procedure can be used at every level, with the only difference in the aggregation size of the underlying network traffic. From now on, we therefore deal only with the intra-level architecture.

Basically, there are two complementary philosophies in the Internet to achieve Quality of Service (QoS) [6]: *Integrated Services* (IntServ) and *Differentiated Services* (DiffServ). IntServ tries to meet QoS requirements by reserving resources for every single flow along its path across the network, so that it can guarantee per-flow QoS. This is done by the RSVP signaling protocol ([7,8]), and it becomes very heavy in the presence of many flows. DiffServ is quite different, as it identifies only several QoS classes, with different packet treatment; it aims at satisfying common-class QoS requirements, by controlling network resources at the nodes (*Per Hop Behavior, PHB* [9]). For example, this is done by setting suitable queue service disciplines, by limiting the amount of traffic in the network, by assigning different priorities, and by performing Call Admission Control (CAC). A central controller, denoted as *Bandwidth Broker* (BB), is dedicated to perform resource reservation, as well as to collect aggregate information for the upper levels. The BB can run on a dedicated machine or on a generic router and exchanges messages with the network nodes by some signaling protocols, in the same fashion as RSVP. We will analyze in detail in the following what kind of information the BB needs to exchange with the nodes in our proposal.

We have chosen to implement three QoS classes: *Expedited Forwarding* (EF), *Assured Forwarding* (AF) and *Best Effort* (BE). EF is the maximum priority traffic, and it should experience very low loss, latency and jitter while traversing the network [*]; such a service appears to the endpoints like a point-to-point connection, or a leased line. AF is an intermediate class, with less pressing requirements on bandwidth, delay and jitter [10]. BE is the current Internet traffic with no guarantees on throughput, delay and jitter.

A global strategy for admission control, bandwidth allocation and routing within the domain is introduced and discussed in the paper. The aim is to minimize blocking of the "guaranteed" flows (EF and AF), while at the same time providing some resources also to BE traffic. As the main objective is to apply the above mentioned control actions on-line, a computational structure is sought, which allows a relatively fast implementation of the overall strategy. In particular, a mix of analytical and simulation tools is applied jointly: "locally optimal" bandwidth partitions for the traffic classes are determined over each link (which set the parameters of the link schedulers in the CRs, the routing metrics and admission control thresholds); a "high level" simulation (involving no packet-level operation) is performed, to determine the per-link traffic flows that would be induced by the given allocation; then, a new analytical computation is carried out, followed by a simulation run, until the allocations do not significantly change.

II. Resource allocation issues

We have previously sketched our DiffServ network architecture; we now need to explain how resources are shared among competitive traffic classes and how to achieve some form of fairness among the different traffic categories.

First of all, a suitable queuing discipline is needed to achieve the required performance for every class: we have chosen to use different queues on each link, one for each kind of traffic. The EF queue is given priority over all others; if we limit the amount of EF traffic in the network, this results in very low delay and jitter experienced by these packets. To meet this requirement, EF traffic is assigned a priority queue, which is served by the scheduler as long as it is non-empty. Users requiring this kind of service are asked to declare their peak rate, which (given that this service is presumed to be very expensive) we assume coinciding with their mean rate. Thus, EF users are classified in a limited number of subclasses, according to their peak rates. Classifiers and traffic conditioners (meters, markers, shapers and droppers) are required at the network edge to perform this task, as well as to verify the compliance of the users' traffic to the agreement, but this issue is beyond our scope.

Once the EF queue is empty, AF high priority traffic is served, while low priority packets are served only if there are still unused AF reserved resources. Finally, when all these queues are empty, BE traffic is served. An AF user declares its mean transmission rate, from which we assign a maximum peak rate. The mean transmission rate is always guaranteed to users; better performance can be achieved if the network load is low. Thus, at the edge nodes, packets will be marked as high priority AF until they do not exceed the mean rate; surplus traffic will be marked as low priority AF. Also in this case, users are classified in a fixed number of subclasses, based on their mean rate value.

Obviously, the performance experienced by the packets depends on the amount of traffic flows of each category admitted into the network: a CAC is necessary to achieve the QoS promised by the DiffServ architecture. With such a discipline, every queue is served at the maximum speed allowed by the link; our aim is to control the bandwidth reserved for each class on every link. Given the peak rate for EF traffic, we know in advance how many connections can be accepted not to exceed the reserved bandwidth. A similar consideration holds for AF, except that mean rates are taken into consideration instead of peak rates. Briefly, we fix two bandwidth thresholds on each link: one for EF and the other for AF; rates of flows traversing the link for each category can never exceed these thresholds. BE traffic can exploit all the unused link bandwidth at any time instant. This results in a *complete partitioning* CAC [11] for both EF and AF, but not for BE, whose upper bound is not fixed, but depends dynamically on the bandwidth left free from the other traffic flows. However, BE is guaranteed the link bandwidth exceeding the two thresholds, but the queuing disciplines do not assure any performance on delay and jitter.

Regarding routing, as EF and AF flows require QoS, we think that this kind of traffic needs a fixed route from source to destination, which appears to them as a virtual leased line [12]. Routing of both AF and EF connections is a priori fixed from the BB, so that it can perform a CAC on the bandwidth available on each traversed link and possibly deny access to the network. Thus, we have already identified a first task for our BB.

Since we wish to maximize the network throughput, we use a simple min-hop metrics for both AF and EF PHB, to reduce the number of links traversed. Otherwise, the whole architecture is suitable for more advanced routing metrics [13], especially those based on aggregation and multi-parameter metrics [14]. For BE traffic routing, instead, we choose a cost equal to the reciprocal of the residual available bandwidth left from the connections with a guaranteed level of QoS. In this case, paths vary dynamically as network load changes over time.

The EF, AF and BE traffic is generated at the nodes of the network as aggregates: this means that the same node can generate more EF or AF connections or BE flows. In such a way, we can easily view the node either as a host on the network or as a border router connected to a hierarchical low-level network (as mentioned in section I).

Now, the issue is how to share link bandwidth among the three competitive classes. Obviously, QoS traffic is critical, because it is thought of as a high revenue one. But we would like to give some form of "protection" to BE, as well. To do this, we assign a cost to each class (as explained later), and try to minimize some given cost function. Nodes collect data on the incoming traffic load, and then the BB gathers all these data and computes a new allocation for all links. Finally, it sends the EF and AF thresholds to each node for its output links. This is the second task demanded to the BB.

To complete the description of our architecture, let us note that a third task of the BB is to collect information about domain ingoing/outgoing traffic from the nodes, and to act as a node for the BB of an upper level.

III. BB ARCHITECTURE AND ALLOCATION ALGORITHM

Allocating resources over the whole network is a very complex task, especially if many links are present. Furthermore, resources to be shared reside on each link.

As previously mentioned, the BB knows from every single node the input traffic matrix for each flow; so, knowing also the different allocations of resources for every link, the BB would be able to simulate the behavior of the whole network in terms of routing, mean link utilizations and blocking probabilities. So, the BB is certainly the best network control point to perform a centralized task for the optimization of the whole system. However, when the input traffic matrix changes enough to make new allocations necessary, in order to cope with the variations and maintain a high level of utilization, the task performing dynamic resource allocation has to be launched on the BB and should complete its calculations as soon as possible.

In order to reduce the computational complexity of this allocation task, we have decided to calculate the new thresholds independently for each link, (in a "locally optimal" way) and to use a very fast network simulator to collect, link by link, all the data necessary to perform this operation. More specifically, the independent link allocations, calculated in this way, have to be tested with a new simulation process to understand if the new network settings have produced routing changes, undesired link utilizations or high blocking probability. In particular, it has to be noted that our BE routing metric depends on the link EF and AF allocations and on the ensuing traffic distribution; but the allocation itself, in its turn, depends on the paths selected by the routing algorithm. A congested link at a certain time instant can become an empty one after allocation owing, for example, to the discovery of a new optimal route for BE traffic. This mechanism should then be applied iteratively, until a fixed point is hopefully reached, if the traffic patterns remain stationary for a sufficiently long time interval.

Therefore, the mechanism is composed by the iteration of two different modules: a fast network simulator (a numeric fluid model that works at aggregate flow level) and an allocation algorithm, based on an analytical model (a stochastic knapsack) and on the output of the simulator, which calculates the resource reservations independently for each link.

Summarizing the resource allocation architecture, the BB alternates simulation of the model and allocation steps. Through the model we generate network traffic and evaluate offered load on every link; successive allocations try to share the available link bandwidth among the different traffic classes. If allocation converges, network resources are optimized, and the BB can set new thresholds on all nodes. When variations on the incoming traffic are learned at the nodes, the BB is notified and starts a new allocation procedure.

A. *Traffic model.*

As noted before, in this kind of environment we need a traffic model that could yield valuable performance in describing the behavior of aggregate flows, with a very low computational effort. Moreover, the aim of the traffic model is not to describe the dynamics at the IP packet level, but to extract some parameters, like link bandwidth occupations or blocking probabilities, with a good approximation,. It is obvious that for this particular purpose a packet-level simulator, like NS2 [14], is not the best suitable choice.

Therefore, we have chosen to implement a traffic model based on fluid approximation, which can efficiently describe the behavior of aggregate flows, ignoring details of the packet level or about the state of each connection.

More in particular, our model can be regarded as a "fast" simulation model, i.e., it is based on a sort of simulation (event generations and performance measures), and it is quite fast, because it operates on aggregates, by using a fluid approximation. Actually, the computational time it requires is comparable to that of a relatively complex analytical model.

In a DiffServ network we can reasonably consider that the traffic behavior is dependent on the different QoS level: in fact it is obvious to think that AF and EF will be used to carry real time flows, like multimedia streams, whereas BE will be used primarily for data transport (i.e., web, mail, file sharing or other). Moreover, in a DiffServ environment, both admission control and traffic shaping have to be applied to QoS flows. Therefore, in our model we had to take into account the significant differences between services with assured QoS level and the common BE ones.

As we have mentioned in the previous section, EF and AF flows use a shortest path route from source to destination, which appears to them as a virtual leased line. For this reason, from now on we will indicate these two categories as *Connection Oriented* (CO) traffic. This hypothesis would be very realistic if we think that, actually, this kind of IP traffic is frequently transported with switching technologies that simply assure the desired QoS level (e.g., MPLS [15]).

An interesting element in our model is the CO traffic source. In fact, we approximate the incoming process of the CO flows with a sort of aggregate source. This source is thought to generate an aggregate of CO flows, which is composed by all the AF (or EF) connections that traverse the same path between two network nodes (like FECs in MPLS). These connections appear directly available at the incoming Edge Router, which can indifferently represent the access node for the hosts or a gateway to another AS. CO flows are generated at each source with exponentially distributed and independent inter-arrival and service times.

To reduce the computational effort of the traffic model, we decided to describe both AF and EF flows as CBR traffic. For EF flows, the constant rate is the maximum assured bandwidth, whereas it is thought of as the minimum nominal bandwidth, which must be guaranteed in every condition, for AF flows. If there are sufficient resources, the AF flows can increase their rates up to a fixed percentage.

Regarding BE traffic, we suppose it to be composed of TCP connections, characterized by their "elastic" behavior. We have adopted a very simple model of TCP, which is described and analyzed in detail in a companion paper [16]. Our goal is not to model the individual TCP flows, but rather to capture the aggregate behavior of connections between an ingress-egress router or source-destination pair for control purpose. Basically, we take into account that connection durations in a simulation depend on network load. We believe it is more realistic to generate the amount of data to be transferred over a connection, instead of connection durations (sharing the same idea of other Internet researchers [17]). The effective duration of a connection is actually determined by the congestion control mechanisms of TCP and overall network conditions: in our architecture, the numbers of CO connections of every class determine the bandwidth left for BE traffic, and the TCP congestion control algorithm influences the way the remaining bandwidth is shared among the TCP connections (or, better, among our aggregate flows). Because of this, we think of each BE peer as an infinite queue, where session requests arrive; each session represents a given amount of data to be transferred. It can come from a network host (as a single user request) or from a lower-level network (as an aggregate of traffic). Inter-arrival times between sessions are exponentially distributed; the size p of each request follows a Pareto distribution, with shape and location parameters α and β, respectively. Such arrivals correspond to the amount of data to be transferred; sessions are served in parallel and the queue output rate is computed accordingly to network load. Basically, in order to model the behavior of the aggregate flows of all peers in sharing the network bandwidth, so as to avoid a detailed simulation of the TCP characteristics, we suppose the aggregate flows to distribute according to a *max-min fair* rule [16].

To obtain steady and meaningful average values, like BE link utilizations or CO blocking probabilities from the flow-level simulation model, the simulation lengths are extracted by a stabilization criterion, which allows to achieve a measured mean value of the parameters under study, which can differ from the statistical mean within a 5% confidence interval, with a confidence level equal to 95%.

B. Analytical Resource Allocation

As the BB knows the input traffic matrix for each flow from the single node, by executing a simplified and fast simulation at the flow level, it can collect data about offered load, in terms of number of flows trying to cross a given link (CO) and average usage of BE bandwidth on the link. From this point on, it begins an independent allocation for each link.

We choose a bandwidth step *bw_step*, equal to the minimum admissible rate for CO connections. Then, the algorithm starts calculating the costs for all possible bandwidth assignments, which are generated as follows.

CO bandwidth is initially set to zero, then it is increased by *bw_step*, until the link bandwidth is reached;

For each of these values, CO bandwidth is shared between EF and AF in a similar way: the EF threshold is set initially to zero and the AF threshold to the total CO assigned bandwidth. All possible assignments are generated by incrementing EF and decreasing AF thresholds by *bw_step*.

1) Cost calculation

In this particular environment, CO traffic looks like traditional circuit switched traffic: there are incoming connections of a given size (peak rate for EF and mean rate for AF), competing for a limited resource. This is the so-called *Stochastic Knapsack* problem; the blocking probabilities can be easily calculated with an extended *Erlang Loss* formula [11].

Let K denote the number of EF rates (for example) and C the resource units. Connections of the k-th class are distinguished by their size, b_k, their arrival rate, λ_k, and their mean duration, $1/\mu_k$. Let $\rho_k = \lambda_k / \mu_k$ and let us denote with n_k the number of active connections of class k and with $\mathcal{S}$ the space of admissible states:

$$\mathcal{S} = \left\{ \mathbf{n} \in \mathbf{I}^K : \mathbf{b} \cdot \mathbf{n} \le C \right\} \tag{1}$$

where I^K denotes the K-dimensional set of non-negative integers, $\mathbf{b}=(b_1,b_2,...b_K)$ and $\mathbf{n}=(n_1,n_2,...n_K)$.

By defining $\mathcal{S}_k$ as the subset of states in which the knapsack admits an arriving class-k object, that is

$$\boldsymbol{\mathcal{S}}_k = \left\{ \mathbf{n} \in \boldsymbol{\mathcal{S}} : \mathbf{b} \cdot \mathbf{n} \le C - b_k \right\} \tag{2}$$

the blocking probability for a class-k connection is:

$$B_k = 1 - \frac{\sum_{n \in \mathbf{S}_k} \prod_{j=1}^{K} \rho_j^{n_j} / n_j!}{\sum_{n \in \mathbf{S}} \prod_{j=1}^{K} \rho_j^{n_j} / n_j!} \tag{3}$$

The blocking probability is computed separately for EF and AF, since each category has its own reserved bandwidth and incoming traffic, independently of the others.

The link cost is derived from the blocking probabilities of the different rate classes:

$$J_{CO} = \max_k \left\{ B_k \right\} \tag{4}$$

and it holds for both EF and AF.

For BE traffic, we have a single aggregate: it is difficult to say if it has enough space or not, because TCP aims at occupying all the available bandwidth. A possibility consists of deciding that the bandwidth available to BE on a link can be considered sufficient if its average utilization by the BE traffic is within a given threshold. To fix this kind of threshold analytically is a very complex task; instead, we have chosen to run several NS2 simulations to extract information about the average behavior of an individual connection, with respect to different values of aggregate utilization ratio. In Figs. 1 and 2, we report two of the several results obtained about the individual TCP connection's performance; for both cases, we have a quick deterioration of the average rate per connection as the aggregate rate approaches 80% of the bandwidth.

Thus, we can choose to impose 80% as maximum threshold for the utilization ratio.

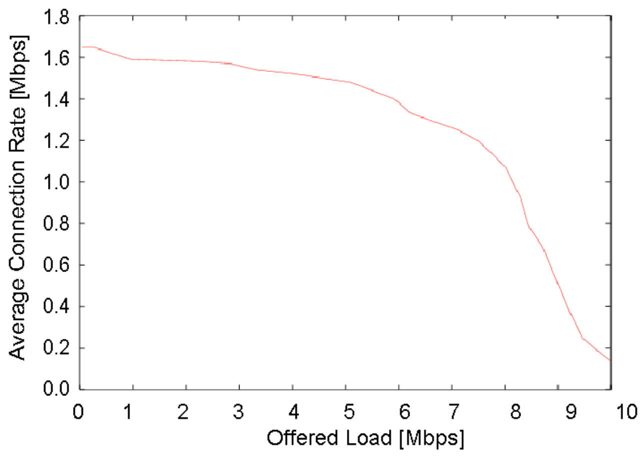

Fig. 1. Average rate per connection, with respect to different values of aggregate utilization of a single link, whose bandwidth is 10 Mbps.

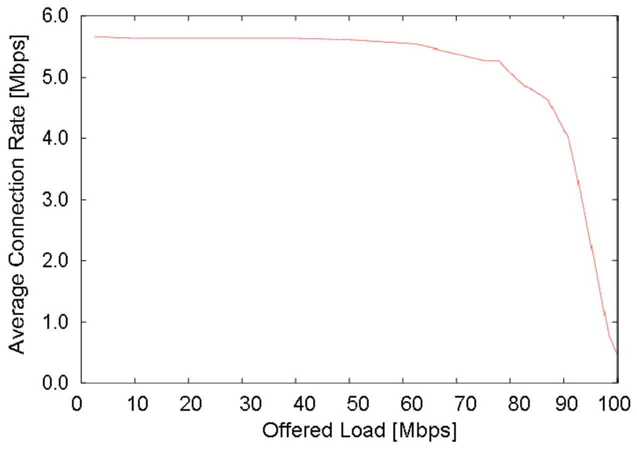

Fig. 2. Average rate per connection, with respect to different values of aggregate utilization of a single link, whose bandwidth is 100 Mbps.

To compute the utilization, let *q(c)* be the probability to have *c* resources occupied in the system by EF and AF connections (which can be calculated from the knapsack model, given EF and AF thresholds); the average quantity *UTIL* of occupied resources results:

$$UTIL = cq(c) \tag{5}$$

Thus, for a total link bandwidth $LINKBW$, the mean available bandwidth for BE is:

$$AVLB_{BE} = LINKBW \square UTIL_{EF} \square UTIL_{AF} \tag{6}$$

The BB keeps track of the BE mean occupied bandwidth on the link, *OCCBW*, during the simulation, so we now are able to define the utilization ratio *UT* as:

$$UT = \frac{OCCBW}{AVLB_{BE}} \tag{7}$$

Starting from this, we expect that connections on the link are given enough bandwidth if this ratio is less than a given value □=0.8.

2) Optimization function

Several functions can be selected to optimize the bandwidth sharing, depending on which goals one is pursuing. Our aim is to equalize the costs of the traffic classes, so that we can vary the system's fairness simply by changing some weight parameters.

However, we want to give some kind of priority to AF and EF traffic classes, with respect to the BE one. Thus, we have chosen a particular equalization cost function *J*, which gives us the possibility to smoothly impose two different constraints: the first one on the maximum blocking probability J_{CO}, and the second one on the BE aggregate utilization ratio.

To take these constraints into account we have defined a specific BE cost function , namely

$$J_{BE} = \frac{\tan\{\square[UT \square\square + \frac{\square}{2}]\}}{\square \cdot \tanh(UT \cdot \square)} \tag{8}$$

where □ is the desired utilization ratio value and □ and □ are two shape parameters; moreover, we have introduced a penalty function for the CO traffic, that is

$$\tilde{J}_{co} = \frac{\arctan\{\square \cdot [J_{CO} \square\square]\} \square \arctan[\square\square\square]}{\arctan\{\square[1 \square\square]\} \square \arctan[\square\square\square]} \tag{9}$$

where □ is a shape parameter and □ is the maximum acceptable blocking probability.

With these new cost components we are now ready to write the final equalization function as:

$$J_{TOT} = J_{BE} + J_{CO} + \tilde{J}_{CO} \square J_{BE}\tilde{J}_{CO} \tag{10}$$

so that the equalization of costs results in minimizing

$$\min_{T_{EF},T_{AF}} \{J_{TOT}\} \tag{11}$$

with respect to T_{EF} and T_{AF}, which are the thresholds for EF and AF traffic, respectively.

A graph of the chosen cost function (10) is reported in Fig. 3, where *□=0.77, □=50, □=0.7,* □=500 and □=0.05. As it can be seen, this function could be roughly divided into three different regions, which are smoothly interconnected:

- the first region corresponds to values of *UT* and J_{CO} that are both below the thresholds; namely, both constraints are satisfied, and the associated global cost is very low, as the link is not congested;
- the second region corresponds to an excessive value of *UT* by BE traffic, while the blocking probabilities remain below their threshold: in this case, the cost is quite high, as there are insufficient resources on the link to carry both BE and CO traffic efficiently, and the allocator decides to penalize BE to respect the constraint on CO blocking probabilities;
- finally, the third region corresponds to the case where the CO constraint is not respected: the cost equalization function takes on higher values because, in heavy congestion conditions, network optimization attempts to respect the CO constraint first, and then, if possible, to protect the BE traffic.

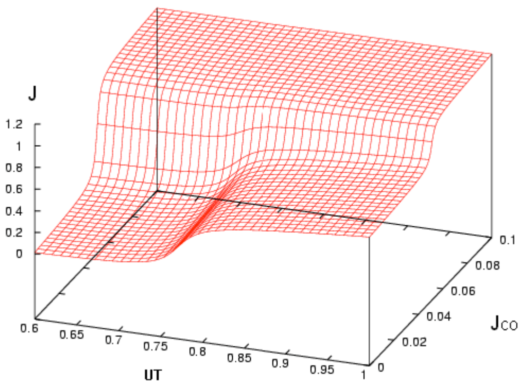

Fig. 3. Graph of the cost equalization function used in link optimization.

Obviously, the optimization problem is not defined for all values of *UT* and J_{CO}, but only in a limited domain, which depends on both BE and CO traffic load on the considered link, with fixed bandwidth. Since the reserved bandwidths are calculated with a minimum granularity equal to the lowest admissible rate for CO flows, the domain of this problem is also discrete. As an example, Fig. 4 reports one of the computed domains, which was obtained by varying all the possible reserved bandwidth thresholds for AF and EF flows, under fixed link capacity and traffic load, and by calculating the resulting *UT* and J_{CO} values.

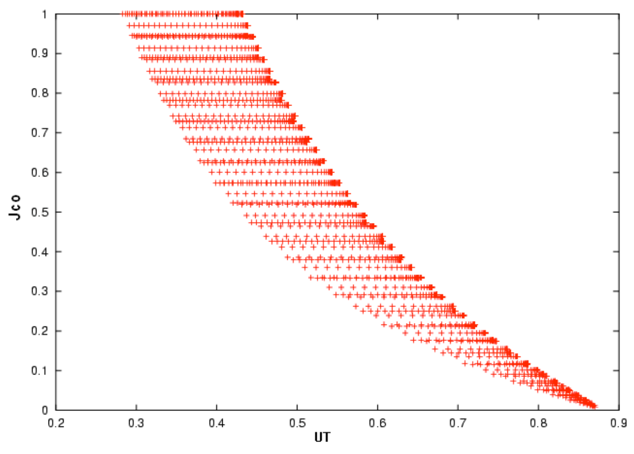

Fig. 4. Example of a domain for the link resource allocation problem.

IV. ALLOCATION RESULTS

In this section, we report and comment some of the results obtained about the performance of the resource allocator. We are going to illustrate first the validation tests for our dynamic allocation mechanism in a DiffServ environment and, then, to analyze the performance. Some preliminary results were already shown in [18].

As a first goal, to understand whether, in order to calculate a correct new allocation of resources, the BB receives useful data on the real state of all the network links, we start analyzing how the fluid model simulator works. For this purpose, we compare our flow level simulation model with the packet level Network Simulator 2 (NS2).

As first test, we have used the simple network in Fig 5, where the bandwidth for every link is 100 Mbps, except for L3 and L6, which are the bottleneck links of the network, and have 50 Mbps of bandwidth. Table 1 contains the mean offered load values of all the aggregate flows used in this test, and Table 2 reports the assigned bandwidth values.

Some basic parameter values are kept constant throughout all tests in the paper: the mean lifetime of EF connections is fixed to 200 s, the one of AF connections to 800 s, and the mean burst size of BE traffic is 0.122 MBytes. With the network topology of Fig. 5, we compare the results of the fluid model to the ones of NS2, first for a Complete Partitioning (CP) policy, utilizing the link bandwidth allocation in Table 2, and then under a Complete Sharing (CS) policy.

Flow	Tx Node	Rx Node	Offered Load [Mbps]	[conn./s]		Average Burst Length [MBytes]	Class Rate [Mbps]
EF1	0	5	33	0.0825	0.005	-	2
EF2	1	6	17	0.085	0.005	-	1
AF0	0	6	33	0.04125	0.00125	-	1
AF1	1	6	17	0.010625	0.00125	-	2
BE1	2	5	27	27	-	0.122	-
BE2	4	6	13	13	-	0.122	-

Table 1. parameters of the flows used in the first validation test.

Link	EF reserved bandwidth [Mbps]	AF reserved bandwidth [Mbps]	Unreserved bandwidth [Mbps]
L1	48	44	8
L2	44	39	17
L3	0	45	5
L4	0	0	100
L5	0	0	100
L6	23	26	1
L7	26	30	4
L8	0	0	100

Table 2. Reserved bandwidths in the links of the network in Fig. 5 during the first CP test.

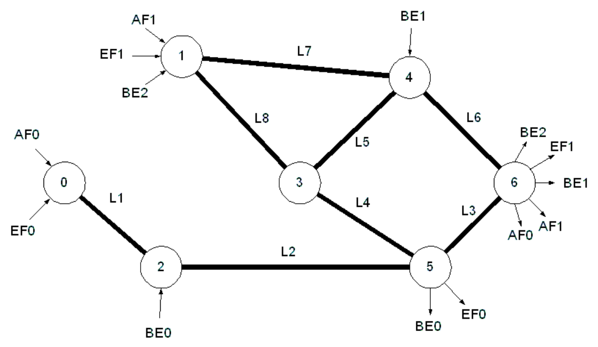

Fig. 5. The first studied network topology.

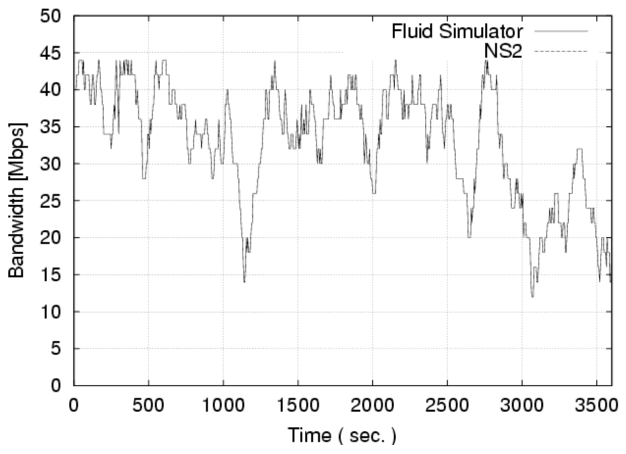

Fig. 6. Bandwidth occupation on L2 of the EF0 aggregate flow, with reference to Fig. 5, in the CP validation test.

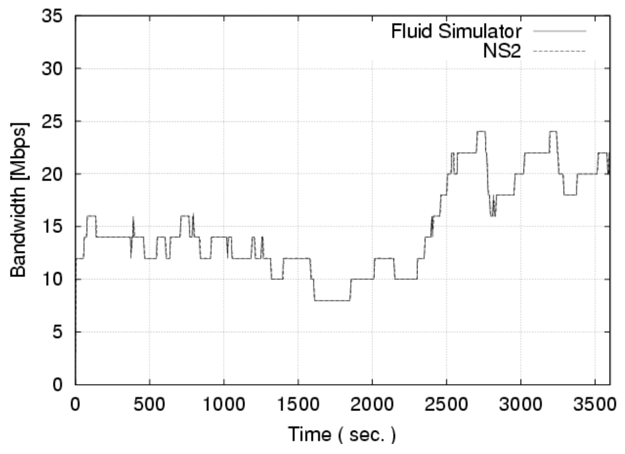

Fig. 7. Bandwidth occupation on L3 of the AF1 aggregate flow, with reference to Fig. 5, in the CP validation test.

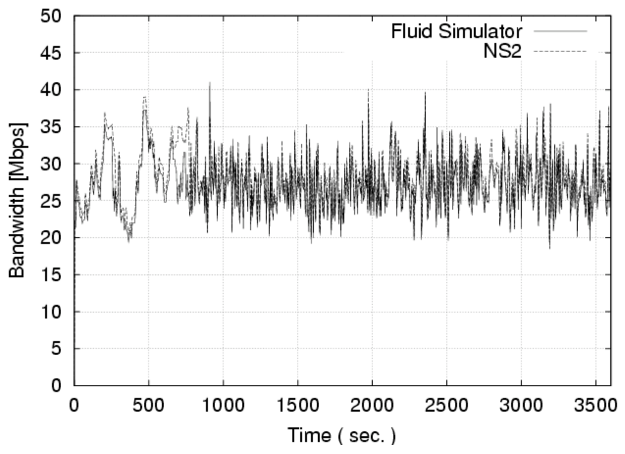

Fig. 8. Bandwidth occupation on L2 of the BE0 aggregate flow, with reference to Fig. 5, in the CP validation test.

Some results of the comparison between NS2 and the fluid simulator are reported in Figs 6-8. These figures show the bandwidth occupation over time, whereas Fig. 9 plots the blocking probability for EF0, calculated in both the fluid model and NS2, by using the same traffic matrix for aggregate flows. It can be observed that, for CO flows, the fluid simulator yields results very close to the NS2 ones. As regards BE traffic, our model and NS2 tend to agree on longer time scales: the model, designed to have a small computational effort, does not consider the behavior of each individual TCP connection, and works on the mean of all aggregate flows; as such, it cannot match exactly the real TCP dynamics.

Nevertheless, the application of the fluid model in the allocation mechanism is not aimed at understanding and predicting the detailed traffic dynamics, but rather at extracting some mean values, like the average bandwidth occupation on each network link or the average blocking probabilities for the CO aggregate flows. In this respect, by analyzing the results obtained in this environment, we can reasonably conclude that the fluid models provide excellent performance.

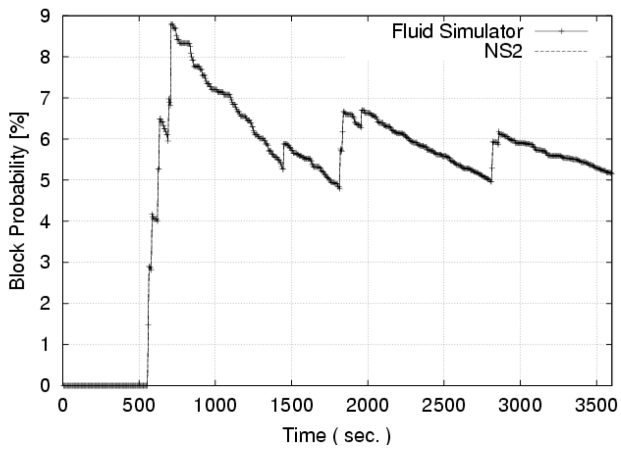

Fig. 9. Blocking probability in percentage of aggregate flow EF0 for both fluid model and NS2 during the CP validation test.

As regards the CS policy, we report some results similar to the previous ones: we used the network topology in Fig. 5 and the traffic matrix reported in Table 1. As with the CP policy, we obtained satisfactory performance with the fluid model: Figs 10-13 show the comparison between the throughputs of some aggregate flows calculated in NS2 and the ones corresponding to the fluid model.

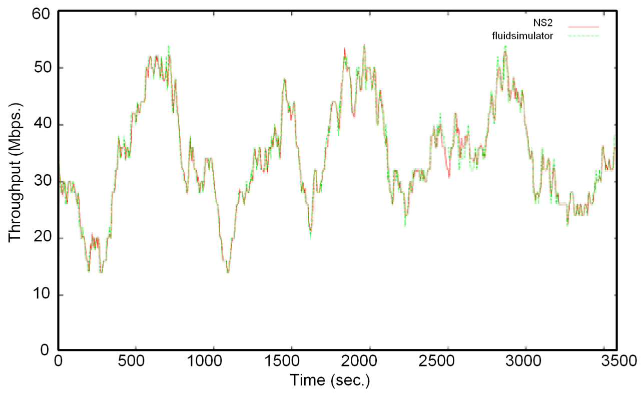

Fig. 10. Bandwidth occupation on L6 of the EF1 aggregate flow, with reference to Fig. 5, in the CS validation test.

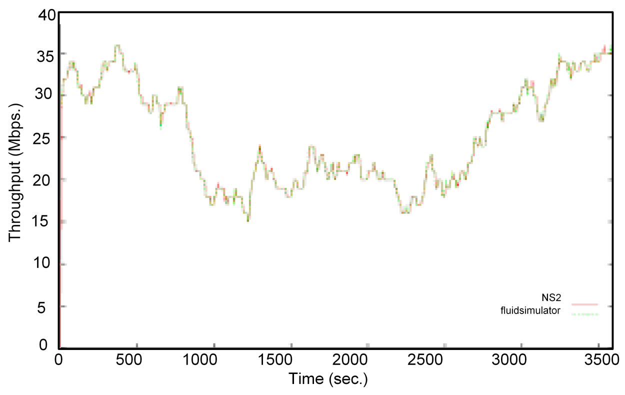

Fig. 11. Bandwidth occupation on L2 of the AF0 aggregate flow, with reference to Fig. 5, in the CS validation test.

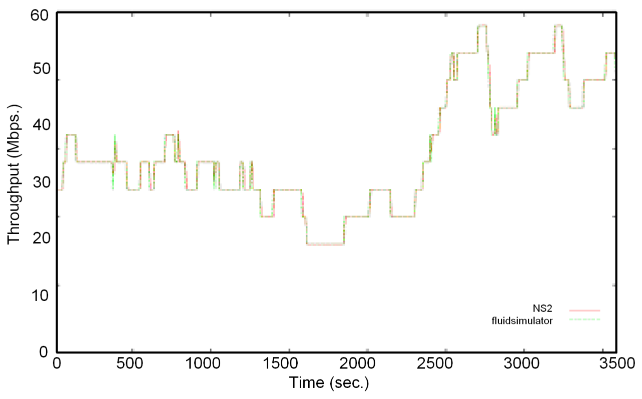

Fig. 12. Bandwidth occupation on L6 of the AF1 aggregate flow, with reference to Fig. 5, in the CS validation test.

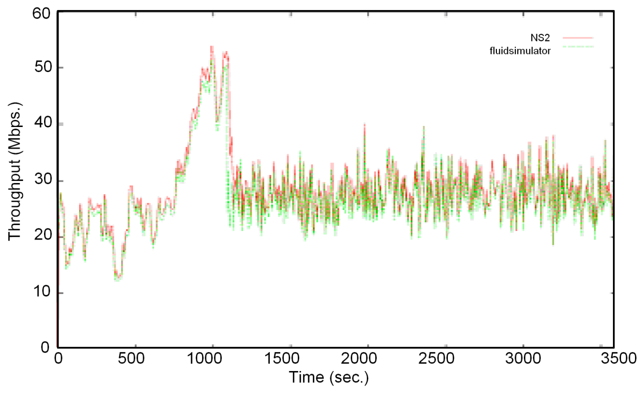

Fig. 13. Bandwidth occupation on L2 of the BE0 aggregate flow, with reference to Fig. 5, in the CS validation test.

By analyzing the performance of the fluid model in this kind of test, we can reasonably conclude that it correctly describes the average performance of both CO and BE traffic. Therefore, it will be used in the following to test the allocator's performance.

In the second test session we have verified the invariance of the allocation results, with respect to different initial network configurations. Thus, we have used the network topology in Fig. 5, and we have executed some allocations, starting with different configurations (21) of reserved bandwidth in all the network links. The initial conditions, used in each of the 21 tests, are reported in Table 3. From the resulting link bandwidth thresholds reported in Table 4, we can clearly conclude that the new optimal network allocation does not depend on the previous network configuration, but only on the traffic matrix.

	Link 1		Link 2		Link 3		Link 6		Link 7	
	EF	AF	EF	AF	EF	AF	EF	AF	EF	AF
Sim 1	0	0	0	0	0	0	0	0	0	0
Sim 2	0	20	0	20	0	10	0	10	0	20
Sim 3	0	40	0	40	0	20	0	20	0	40
Sim 4	0	60	0	60	0	30	0	30	0	60
Sim 5	0	80	0	80	0	40	0	20	0	20
Sim 6	0	100	0	100	0	50	0	50	0	100
Sim 7	20	0	20	0	10	0	10	0	20	0
Sim 8	20	20	20	20	10	10	10	10	20	20
Sim 9	20	40	20	40	10	20	10	20	20	40
Sim 10	20	60	20	60	10	30	10	30	20	60
Sim 11	20	80	20	80	10	40	10	40	20	80
Sim 12	40	0	40	0	20	0	20	0	40	0
Sim 13	40	20	40	20	20	10	20	10	40	20
Sim 14	40	40	40	40	20	20	20	20	40	40
Sim 15	40	60	40	60	20	30	20	30	40	60
Sim 16	60	0	60	0	30	0	30	0	60	0
Sim 17	60	20	60	20	30	10	30	10	60	20
Sim 18	60	40	60	40	30	20	30	20	60	40
Sim 19	80	0	80	0	40	0	40	0	80	0
Sim 20	80	20	80	20	40	10	40	10	80	20
Sim 21	100	0	100	0	50	0	50	0	100	0

Table 3. Initial network configurations for the invariance test. Note that all the threshold values are in Mbps.

	Link1		Link2		Link3		Link6		Link7	
	EF	AF	EF	AF	EF	AF	EF	AF	EF	AF
Sim 1	50	42	44	39	0	45	23	26	24	30
Sim 2	48	42	44	39	0	45	23	27	24	28
Sim 3	48	42	44	40	0	46	23	27	24	28
Sim 4	48	42	44	40	0	46	23	27	24	28
Sim 5	48	42	44	40	0	46	23	27	24	30
Sim 6	48	42	44	39	0	46	23	27	24	30
Sim 7	48	42	44	40	0	45	22	28	24	30
Sim 8	48	42	44	39	0	45	22	27	23	28
Sim 9	50	42	44	39	0	46	22	26	24	28
Sim 10	48	42	44	39	0	45	22	26	24	28
Sim 11	48	42	44	39	0	45	22	26	24	28
Sim 12	48	42	44	39	0	44	22	28	24	28
Sim 13	48	42	44	39	0	45	22	27	23	28
Sim 14	48	42	44	39	0	46	22	26	24	28
Sim 15	48	42	44	39	0	44	22	28	24	28
Sim 16	48	42	44	40	0	45	22	28	24	30
Sim 17	48	42	44	40	0	46	23	27	24	28
Sim 18	48	42	44	39	0	45	22	27	23	28
Sim 19	48	42	44	40	0	45	22	28	24	30
Sim 20	48	42	44	39	0	46	22	26	24	28
Sim 21	48	42	44	39	0	45	22	27	23	28

Table 4. Optimal network configurations calculated by the allocator in the invariance test. Note that all the threshold values are in Mbps.

Finally, we report two of the numerous test sessions aimed at determining the resource allocator's performance, obtained with different networks. For each network analyzed, we have chosen to study how the allocator's efficiency changes according to the traffic load. Moreover, we have decided to compare the results achieved to those of a CS based network: with this kind of policy we expect to obtain (with respect to a CP network with the allocation mechanism) no protection for BE traffic but, especially for high network loads, better (overall) CO performance, because in this case the only thresholds for the CAC are the link bandwidths.

In this context, as a first test network, we utilized once again the one in Fig. 5, which is not too complex and allows us understand easily how the mechanism works: in fact, this simple network presents only three bottleneck links (with reference to Fig. 5, L2, L3 and L6), shared by the aggregate flows in Table 5. Thus, we present the results of 13 allocations, which were obtained under different traffic loads that are reported in Table 5. Tables 6 and 7 show the achieved link thresholds for AF and EF traffic, respectively.

The average throughputs of all aggregate flows used in this test session are reported in Figs. 14-20, while the blocking probabilities for EF and AF flows are in Figs. 21-24.

	EF 0	EF 1	AF 0	AF 1	BE 0	BE 1	BE 2
Sim 1	28	14	28	15	24	11.5	11.5
Sim 2	29	15	29	15	24	11.5	11.5
Sim 3	30	16	30	15	24	11.5	11.5
Sim 4	31	17	31	15	24	11.5	11.5
Sim 5	32	18	32	15	24	11.5	11.5
Sim 6	33	19	33	15	24	11.5	11.5
Sim 7	34	20	34	15	24	11.5	11.5
Sim 8	35	21	35	15	24	11.5	11.5
Sim 9	36	22	36	15	24	11.5	11.5
Sim 10	37	23	37	15	24	11.5	11.5
Sim 11	38	24	38	15	24	11.5	11.5
Sim 12	39	25	39	15	24	11.5	11.5
Sim 13	40	26	40	15	24	11.5	11.5

Table 5. Average offered loads (in Mbps) for the aggregate flows in the performance tests with the network in Fig 5.

	Link 1	Link 2	Link 6	Link 7
Sim 1	42	42	21	21
Sim 2	44	44	21	22
Sim 3	44	44	22	23
Sim 4	46	46	23	24
Sim 5	46	46	24	26
Sim 6	48	46	25	27
Sim 7	48	46	26	28
Sim 8	50	46	26	29
Sim 9	50	50	27	31
Sim 10	52	52	28	32
Sim 11	54	54	28	32
Sim 12	52	52	28	34
Sim 13	52	52	28	35

Table 6. Resource thresholds (in Mbps) allocated for EF traffic in the links of the network of Fig 5 during the performance tests. Note that the links not shown have no reserved resources.

	Link 1	Link 2	Link 3	Link 6	Link 7
Sim 1	36	36	36	26	26
Sim 2	38	38	38	24	28
Sim 3	39	39	39	24	26
Sim 4	39	39	39	24	28
Sim 5	42	42	41	24	26
Sim 6	42	42	42	24	26
Sim 7	44	41	43	24	26
Sim 8	45	41	45	24	26
Sim 9	46	44	45	22	26
Sim 10	47	47	46	22	26
Sim 11	46	46	47	22	28
Sim 12	47	47	48	22	26
Sim 13	47	47	49	22	26

Tab. 7. Resource thresholds (in Mbps) allocated for AF traffic in the links of network of Fig R1 during the performance tests. Note that the links not shown have no reserved resources.

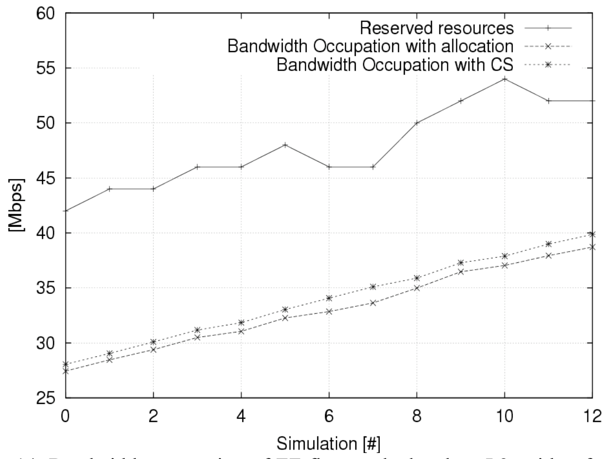

Fig. 14. Bandwidth occupation of EF flows calculated on L2, with reference to Fig. 5, in the case of allocation and of CS policy.

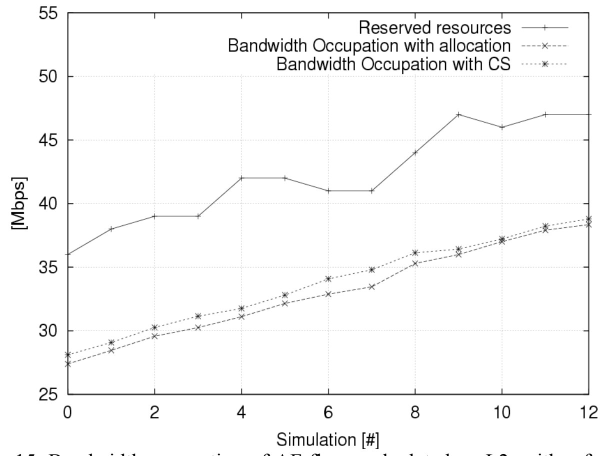

Fig. 15. Bandwidth occupation of AF flows calculated on L2, with reference to Fig. 5, in the case of allocation and of CS policy.

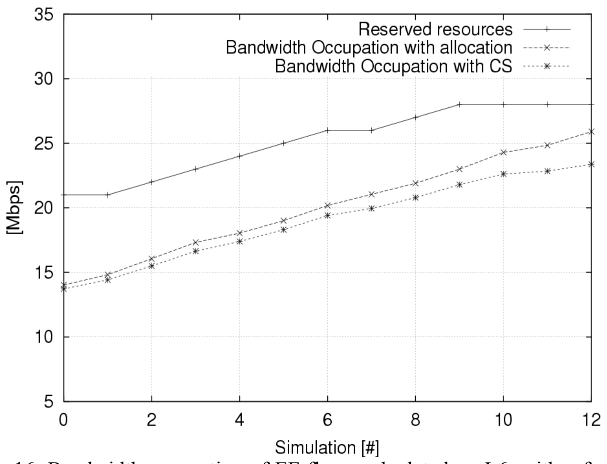

Fig. 16. Bandwidth occupation of EF flows calculated on L6, with reference to Fig. 5, in the case of allocation and of CS policy.

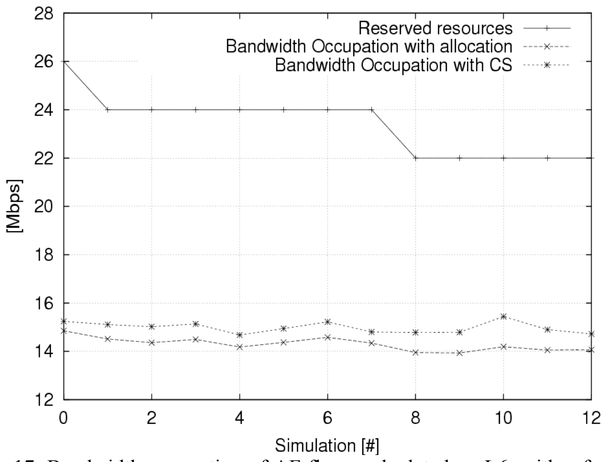

Fig. 17. Bandwidth occupation of AF flows calculated on L6, with reference to Fig. 5, in the case of allocation and of CS policy.

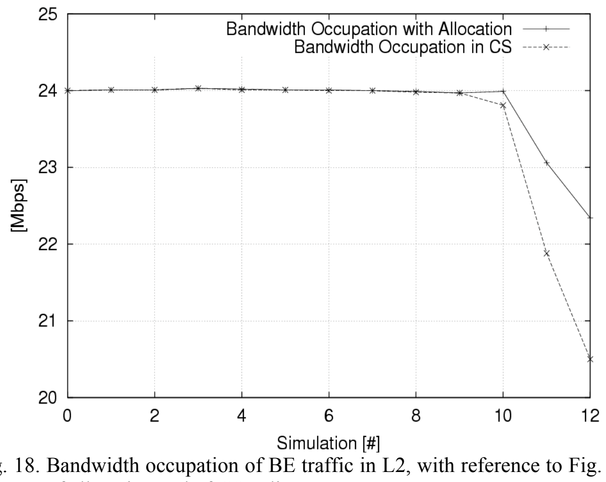

Fig. 18. Bandwidth occupation of BE traffic in L2, with reference to Fig. 5, in the case of allocation and of CS policy.

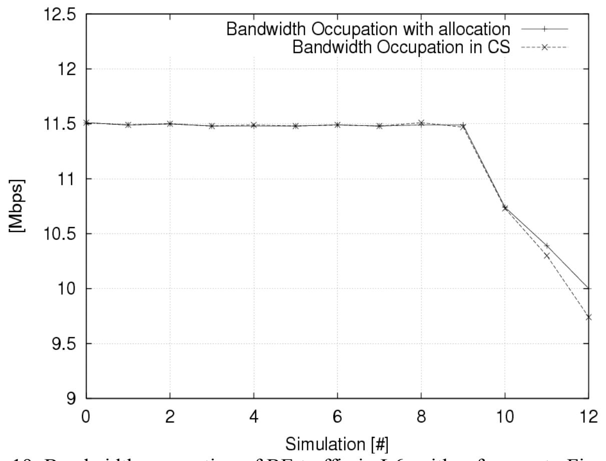

Fig. 19. Bandwidth occupation of BE traffic in L6, with reference to Fig. 5, in the case of allocation and of CS policy.

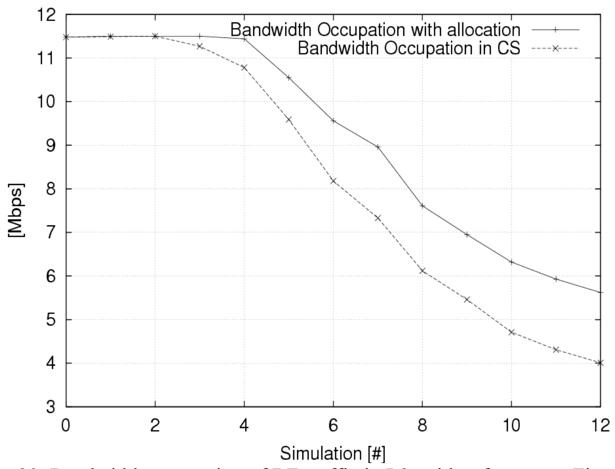

Fig. 20. Bandwidth occupation of BE traffic in L3, with reference to Fig. 5, in the case of allocation and of CS policy.

From all these results (Figs. 14-20) we can remark how the BE aggregates are "protected", as far as possible, by CO allocated thresholds, as the network congestion level rises. On the contrary, we can stress how a CS policy (without BE protection) could obviously improve the CO performance in terms of higher average throughputs and lower blocking probabilities.

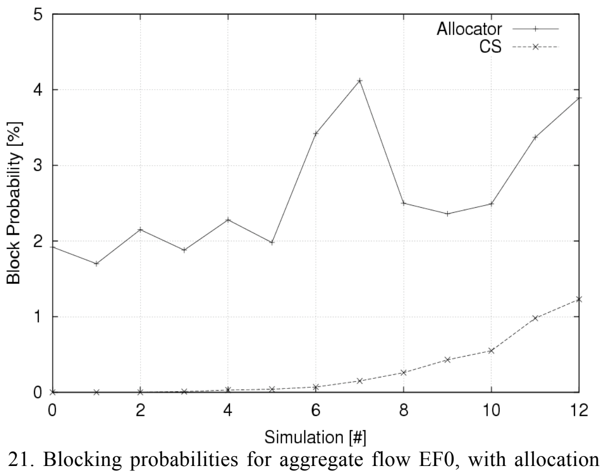

Fig. 21. Blocking probabilities for aggregate flow EF0, with allocation and with CS policy.

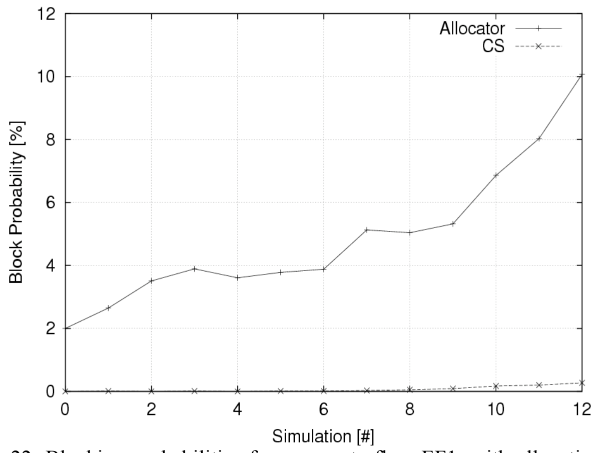

Fig. 22. Blocking probabilities for aggregate flow EF1, with allocation and with CS policy.

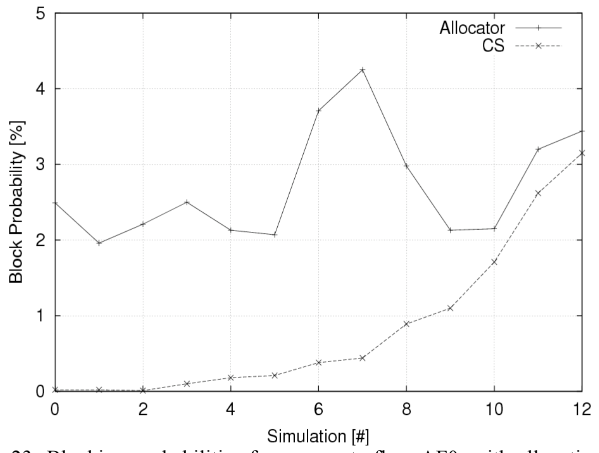

Fig. 23. Blocking probabilities for aggregate flow AF0, with allocation and with CS policy.

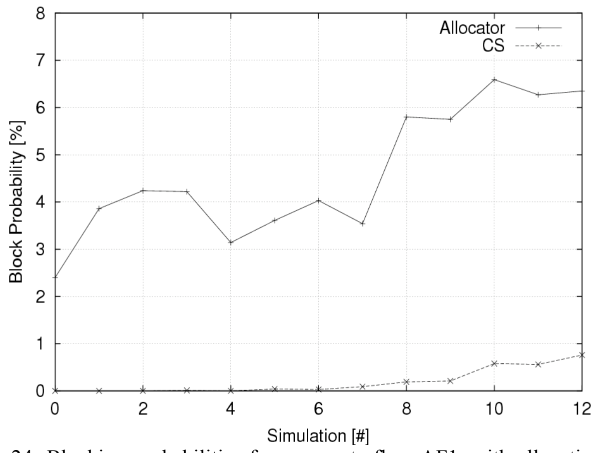

Fig. 24. Blocking probabilities for aggregate flow AF1, with allocation and with CS policy.

To better understand how the allocation decisions are adaptively extracted from the congestion level of the links, we show in Figs. 25-27, for each bottleneck link of network in Fig. 5, how the cost of the "locally optimal" link resource configuration changes according to the increase in network offered load. To do so, we have reported all the allocations calculated during the test on the cost functions of the link that better reflects the specific behavior we want to highlight.

In Fig. 25, we can observe how, on bottleneck link L2 in Fig. 5, the allocator tends to satisfy all the desired constraints, as long as the network congestion is low; when the traffic load increases and link resources are no longer sufficient, the allocator decides to maintain the CO blocking probability constraint and to sacrifice BE performance, accepting a value of utilization ratio higher than the desired one. Moreover, the values of the cost function parameters are *□=0.77, □=50, □=0.7,* □=500 and □=0.05, which means that we try to maintain the CO blocking probability under the 5% and the utilization ratio for BE under the 77%. On the other hand, Fig. 26 highlights the shift in the optimal configuration during the increase of the traffic load for link L3: in this case, the resources remain sufficient to satisfy both the CO constraint and the BE one, and so all the allocations appear mapped onto the lower part of the cost function. Finally, as we can analyze for link L6 in Fig. 27, first, when the traffic load rises slightly, the allocator has not enough resources to assure the BE ratio constraint; afterwards, while traffic load continues to increase further, there are no more resources to satisfy the CO constraint, as well. Note that in this last case the BE ratio values of the calculated allocation do not diverge, because BE traffic continues to use all the reserved bandwidth which the AF and EF flows do not temporarily utilize.

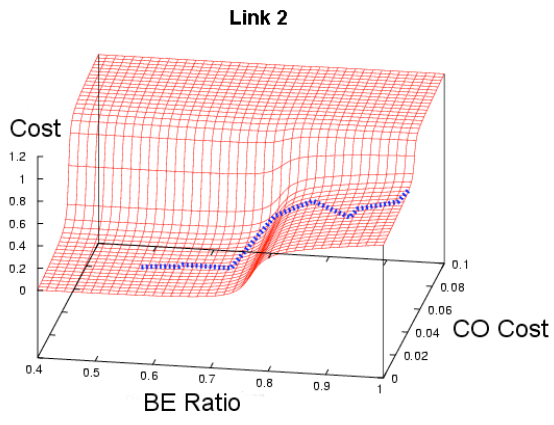

Fig. 25. Mapping on the cost function profile of the change in resource allocation on link L2 in Fig. 5, while the traffic load increases.

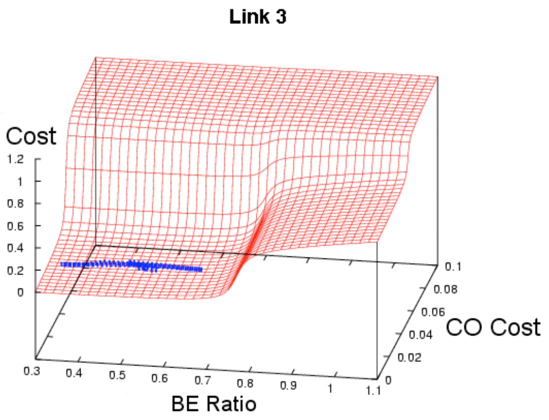

Fig. 26. Mapping on the cost function profile of the change in resource allocation on link L3 in Fig. 5, while the traffic load increases.

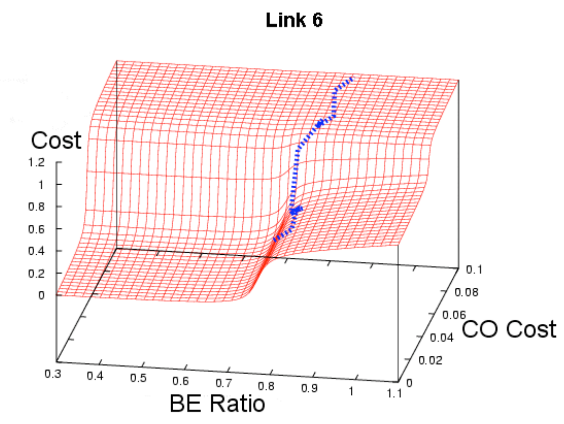

Fig. 27. Mapping on the cost function profile of the change in resource allocation on link L6 in Fig. 5, while the traffic load increases.

Finally, we show the results of a second test network, more complex than the previous one. The network, represented in Fig. 28, is composed by eleven links and seven nodes. The bandwidth of all links is fixed to 80 Mbps and the propagation delay time to 5 ms. In this new test session we used six EF aggregate flows, five AF and six BE ones, whose parameter are described in Tables 8 and 9, where we can observe how the traffic load changes in all the different simulations.

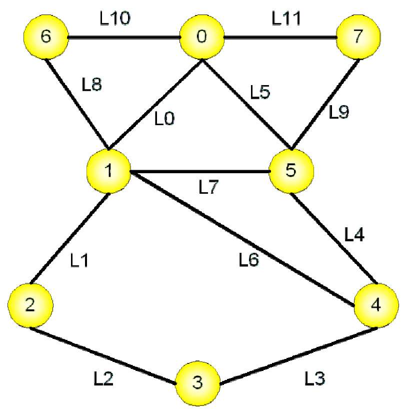

Fig. 28. A second network topology used for the allocator's performance test.

EF aggregate flows			**AF aggregate flows**			**BE aggregate flows**		
Flow id	Tx node	Rx node	Flow id	Tx node	Rx node	Flow id	Tx node	Rx node
0	0	3	0	7	3	0	6	2
1	6	4	1	0	3	1	6	4
2	7	4	2	6	3	2	6	0
3	1	3	3	0	4	3	7	2
4	5	3	4	5	4	4	7	4
5	0	4				5	7	0

Table 8. Flow ID, Tx node and Rx node for each aggregate flow used for the performance test with the network in Fig. 28.

	EF 0	EF 1	EF 2	EF 3	EF 4	EF 5	AF 0	AF 1	AF 2	AF 3	AF 4	BE 0	BE 1	BE 2	BE 3	BE 4	BE 5
Sim 0	7	10	8	7	5	8	4	4	5	8	10	8	14	20	10	14	20
Sim 1	8	11	9	8	6	9	5	5	6	9	11	8	14	20	10	14	20
Sim 2	9	12	10	9	7	10	6	6	7	10	12	8	14	20	10	14	20
Sim 3	10	13	11	10	8	11	7	7	8	11	13	8	14	20	10	14	20
Sim 4	11	14	12	11	9	12	8	8	9	12	14	8	14	20	10	14	20
Sim 5	12	15	13	12	10	13	9	9	10	13	15	8	14	20	10	14	20
Sim 6	13	16	14	13	11	14	10	10	11	14	16	8	14	20	10	14	20
Sim 7	14	17	15	14	12	15	11	11	12	15	17	8	14	20	10	14	20
Sim 8	15	18	16	15	13	16	12	12	13	16	18	8	14	20	10	14	20
Sim 9	16	19	17	16	14	17	13	13	14	17	19	8	14	20	10	14	20
Sim10	17	20	18	17	15	18	14	14	15	18	20	8	14	20	10	14	20
Sim11	18	21	19	18	16	19	15	15	16	19	21	8	14	20	10	14	20
Sim12	19	22	20	19	17	20	16	16	17	20	22	8	14	20	10	14	20
Sim13	20	23	21	20	18	21	17	17	18	21	23	8	14	20	10	14	20
Sim14	21	24	22	21	19	22	18	18	19	22	24	8	14	20	10	14	20

Table 9. Offered load, in Mbps, of the aggregate flows in all the simulations of performance test for the network in Fig. 28.

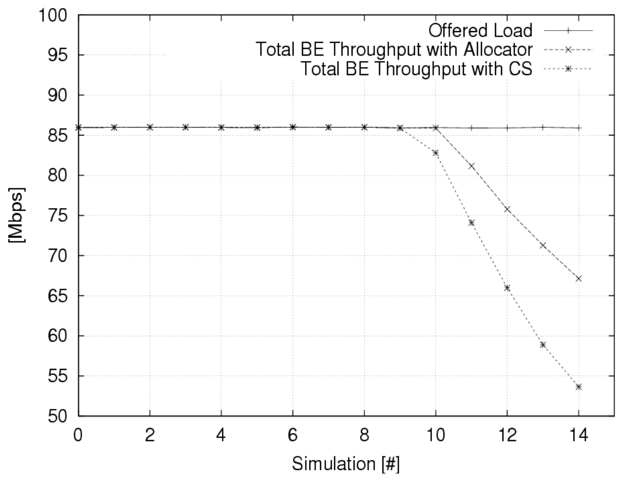

Fig. 29. Total offered load and sum of the throughputs of all the BE aggregate flows with allocation and with CS policy during the simulation tests with the network in Fig. 28.

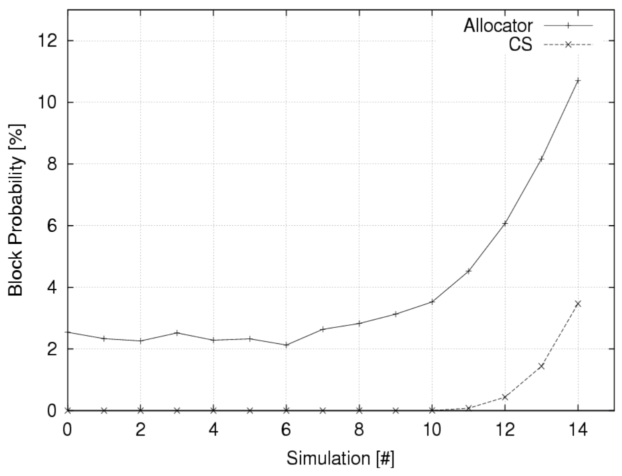

Fig. 30. Maximum blocking probability value for all the EF flows with allocation and with CS policy during the simulation tests with the network in Fig. 28.

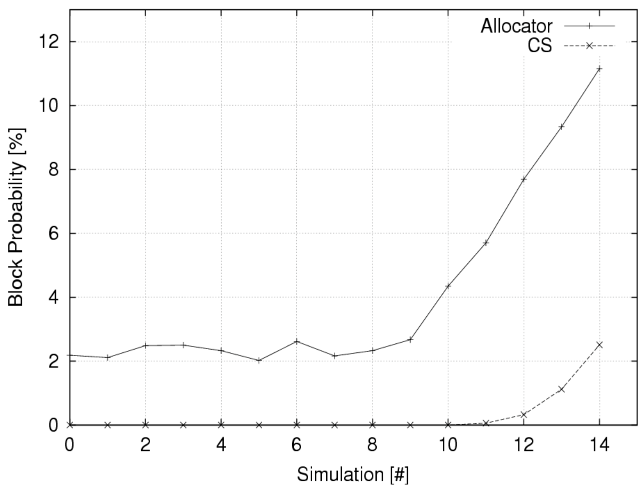

Fig. 31. Maximum blocking probability value for all the AF flows with allocation and with CS policy during the simulation tests with the network in Fig. 28.

Figs. 29, 30 and 31 report the results obtained with the network topology in Fig. 28. More in particular, Fig. 29 shows the total BE offered load and the sum of the throughputs of all the BE aggregate flows for both the allocation with our mechanism and the CS policy. Figs. 30 and 31 show the maximum blocking probability value for EF and AF flows, still in both allocations with our mechanism and the CS policy. As we can observe from these results, with considerable congestion levels the allocation mechanism, limiting the reserved resources to the CO, assures better performance to the BE traffic class than the CS policy.

V. CONCLUSIONS

A multiservice IP network based on the DiffServ paradigm has been considered, composed by Edge Routers (ER) and Core Routers (CR), forming a domain that is supervised by a Bandwidth Broker (BB). The traffic in the network belongs to three basic categories: Expedited Forwarding (EF), Assured Forwarding (AF) and Best-Effort (BE). A global strategy for admission control, bandwidth allocation and routing within the domain has been introduced and discussed. The aim is to minimize blocking of the "guaranteed" flows (EF and AF) within a fixed constraint on the maximum blocking value, while at the same time providing some resources also to BE traffic in terms of “utilization ratio”. As the main objective is to apply the above mentioned control actions on-line, a computational structure has been devised, which allows a relatively fast implementation of the overall strategy. In particular, a mix of analytical and simulation tools is applied jointly: "locally optimal" bandwidth partitions for the traffic classes are determined over each link (which set the parameters of the link schedulers in the CRs, the routing metrics and admission control thresholds); a "high level" simulation (involving no packet-level operation) is performed, to determine the per-link traffic flows that would be induced by the given allocation; then, a new analytical computation is carried out, followed by a simulation run, until the allocations do not significantly change. The possible convergence of the scheme (under a fixed traffic pattern) has been investigated and the results of its application under different traffic loads has been studied on a test network. The results show the capability of the proposed method of controlling the performance of the network and of maintaining a good level of global efficiency.

REFERENCES

[1] Telcordia Technologies, WWW document. URL: *http://www.netsizer.com*

[2] S. Deering, R. Hinden. RFC 2460: *Internet Protocol, Version 6 (IPv6) Specifications*. Available online: URL *http://www.ietf.org/rfc/rfc2460.txt*. December 1998.

[3] R. Govindan and A. Reddy, *An Analysis of Internet Inter-Domain Topology and Route Stability*. Proceedings of the IEEE INFOCOM 1997, Kobe, Japan.

[4] C. Labovitz, A. Ahuja, A. Bose and F. Jahanian. *Delayed Internet Routing Convergence*. IEEE/ACM Transactions on Networking, vol. 9, no. 3, June 2001, pp. 293-306

[5] V. Paxson. *End-to-End Routing Behavior in the Internet*. IEEE/ACM Transactions on Networking, Volume 7, no.3, June 1999, pp. 277-292

[6] X. Xiao and L. M. Ni. *Internet QoS: A Big Picture*. IEEE Network, March/April 1999, pp. 8-18.

[7] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin. RFC 2205: *Resource ReSerVation Protocol – Version 1 Functional Specification*. Available online: URL: *http://www.ietf.org/rfc/rfc2205.txt*. September 1997.

[8] S. Hergoz. RFC 2750: *RSVP Extensions for Policy Control*. Available online: URL *http://www.ietf.org/rfc/rfc2750.txt*. January 2000.

[9] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss. RFC 2475: *An Architecture for Differentiated Services*. Available online: URL *http://www.ietf.org/rfc/rfc2475.txt*. December 1998.

[10] J. Heinanen, T. Finland, F. Baker, W. Weiss and J. Wroclawski. RFC 2597: *Assured Forwarding PHB Group*. Available online: URL: *http://www.ietf.org/rfc/rfc2597.txt*. June 1999.

[11] K. W. Ross. *Multiservice Loss Models for Broadband Telecommunication Networks*. Springer, 1995.

[12] V. Jacobson, K. Nichols and K. Poduri. RFC 2598: *An Expedited Forwarding PHB*. Available online: URL *http://www.ietf.org/rfc/rfc2598.txt*. June 1999.

[13] S. Chen, K. Nahrstedt. *An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions*. IEEE Network, Special Issue on Transmission and Distribution of Digital Video, Nov./Dec. 1998.

[14] The Network Simulator – ns2. Documentation and source code from the home page: *http://www.isi.edu/nsnam/ns/*.

[15] D. Awduche, J. Malcom, J. Agogbua, M. O'Dell, J. McManus. RFC 2702: *Requirements for Traffic Engineering Over MPLS*. Available online: URL *http://www.ietf.org/rfc/rfc2702.txt*. September 1999.

[16] R. Bolla, R. Bruschi, M. Repetto, "A Fluid Model for Aggregate TCP Connections", TANGO project report, Madonna di Campiglio, January 2003.

[17] S. Floyd, V. Paxson, "Difficulties in Simulating the Internet", IEEE/ACM Trans. Networking, vol. 9, no. 4, 2001.

[18] R. Bolla, F. Davoli, M. Repetto, "A control architecture for Quality of Service and resource allocation in multiservice IP networks", *Proc. Internat. Workshop on Architectures for Quality of Service in the Internet* (Art-QoS 2003), Warsaw, Poland, March 2003; *Lecture Notes in Computer Science*, 2698, Springer-Verlag, Berlin, 2003, pp. 49-63.