Proactive Two-Tier Bandwidth Brokerage for on-demand Policy-based Resource Allocation in Stateless IP Networks

Kamel Haddadou[1], Yacine Ghamri-Doudane[2], Samir Ghamri-Doudane[1], and Nazim Agoulmine[2]

[1] LIP6, Pierre & Marie Curie University, 8, rue du Capitaine Scott, 75015 Paris – France
Kamel.Haddadou@lip6.fr, Samir.Ghamri-Doudane@lip6.fr
[2] Networks and Multimedia Systems Research Group, IIE/University of Evry joint group, 18 allée Jean Rostand, 91025 Evry Cedex – France
Ghamri@iie.cnam.fr, Nazim.Agoulmine@iup.univ-evry.fr

Abstract. In order to improve the scalability of the IETF's PBM architecture, we previously proposed an extension to this architecture. The aim of our proposal was to facilitate the support of on-demand resource allocation in stateless IP networks. The proposed schema distribute part of the decision making process but keeps centralized the bandwidth brokerage operation as this latter uses a critical resources (traffic matrix). Our current work aims to render the PBM architecture completely distributed by designing a scalable scheme for bandwidth brokerage. To that end, we propose using a proactive two-tier scheme in order to improve the scalability of the resource management procedures. In this paper, we present the proposed scheme, its design features and the set of experimentations we realized in order to demonstrate its performances.

1 Introduction

A major challenge in emerging multi-service and QoS-enabled IP networks is to offer access to a wide range of services anytime and ideally anywhere. Both of network operators and end-users are willing to respectively offer and use these services with a large range of QoS-guarantees. To achieve this aim, efficient and dynamic control of network resources is submitted to be the key issues in the ongoing all-IP realm. To that end, we proposed in a previous work [1] a scalable on-demand policy based resource allocation framework.

The proposed framework is based on the combination of per-session QoS signaling and Policy-Based Management (PBM) [2]. It suggests distributing part of the decision making while keeping centralized the critical operations which use critical resources on the management system. In our previous work, we identified the bandwidth brokerage operation as the only one that uses critical resources (the traffic matrix). Indeed, two different resource requests initiated by different traffic sources may concern the same link in the network. This operation has then been maintained centralize. Moreover, the Bandwidth Broker (BB) has been designed in such a way that it avoids paral-

lel access to the traffic matrix. The analytical and experimental results, we obtained when distributing some of the decision making operations in the PBM system [1], showed that our architecture is highly scalable compared to the PBM architecture proposed by the Internet Engineering Task Force (IETF) [3]. However, the system-throughput, that represents the rate of QoS request treated by the management system, has always an upper bound. We proved analytically that this upper bound is only due to the operations that remained centralization (i.e. the Bandwidth Broker). Indeed, we demonstrated analytically that the system-throughput is inversely proportional to the average computational time of the Bandwidth Broker element. The maximum number of active users in the network was also proven to be upper bounded.

In order to render our on-demand policy-based resource allocation framework completely insensitive to both the rate of requests and the number of active users, we suggest to instantiate several bandwidth brokers in the domain. We propose that each BB will be responsible of managing a part of the network resources. To do so, the idea is to manage network-resources using two granularity levels: a link-level and a path-level. A path is defined as the set of links between two edge-routers. Then, the idea behind instantiating multiple BBs is to allocate to each path a part of the resources of each of its links. Note however that in order to optimize network-resource usage, this part should not be static. Hence, we will have two hierarchical levels for bandwidth management:

1. A centralized Bandwidth Broker (cBB) aiming to manage link resources as a traffic matrix.
2. Multiple edge Bandwidth Brokers (eBBs) aiming to both: (i) realizing a per-session path-oriented admission control procedure, and (ii) proactively managing the amount of resources assigned to the subset of network-paths managed by the eBB. The proactive allocation/de-allocation of link-resources to paths is realized in order to automatically adapt path-resources according to the customers' resource-request distribution.

The idea of distributing the bandwidth allocation among multiple bandwidth brokers by using the two granularity levels (link-level and path-level) is not new. This idea was previously presented in [4]. In our work, we follow the same idea but we propose a different way to manage the bandwidth assignment to paths. Indeed, in order to minimize the delay to respond to a resource allocation request, we suggest to proactively increase (respectively decrease) the amount of resources assigned to a particular path depending on its actual usage. The objectives of this paper are to present, discuss the design feature, and realize a complete experimental evaluation of our Proactive Two Tier Bandwidth Brokerage (PTT-BB) Architecture.

The rest of this paper is organized as follows. IETF's PBM architecture and its scalability limitations are presented in Section 2. Then, we briefly review our previously proposed scalable on-demand policy-based resource allocation framework and its performances. In Section 4, we present the bandwidth brokerage operation and the problem statement. Following section presents the PTT-BB architecture and its components. The test-bed description and the empirical results targeting several testing scenarios are presented in Section 6. Finally, Section 7 concludes the paper.

2 Scalability limitation of the IETF's PBM Architecture

The IETF Resource Allocation Protocol (RAP) Working Group has specified a complete framework for policy definition and administration [3]. This framework introduces a set of components to enable policy rules definition, saving and enforcing: the Policy Decision Point (PDP), the Policy Enforcement Point (PEP), and the Policy Repository. PEP components are policy decision enforcers located in network and system equipments. The PDP is the component responsible for high-level decision-making process. This process consists of retrieving and interpreting policies, and implementing the decision in the network through the set of PEPs. The policy repository contains policy rules that are used by the PDP.

In order to exchange policy information and/or decisions, the PDP interacts with the PEPs using one of the several protocols specified or extended for this purpose. Among them, the Common Open Policy Service (COPS) protocol [3] is the one which was specifically designed by the IETF to realize this interaction.

Initially, the COPS protocol was designed mainly for resource allocation in an Internet backbone. In order to make such allocation, two models within the COPS protocol were proposed: the Outsourcing model and the Provisioning model. In the former, policy-requests are triggered by particular events and forwarded to the PDP for policy-decisions. In contrary, in the provisioning model, policy-decisions are installed in the PEP prior to the arrival of the concerned flows. In both cases, the used policies are supposed deduced from static customer's contracts which are called Service Level Agreements (SLAs).

The main problem with the IETF PBM architecture is that it does not take into account the scalability problem. In fact, the client-server architecture as defined by the IETF is not scalable as it stands. In the case of a large network, the PDP become a bottleneck leading to serious problems while handling a potentially high number of policy requests. Hence, this architecture does not handle explicitly dynamic changes in the users' Service Level Agreements (SLA). The SLA is treated in a static manner and the customer should re-negotiate completely her/his SLA in the case of changes in her/his requirements. This lack of dynamicity is a curb to the development of punctual access and usage of services such as Voice over IP (VoIP) and Video on Demand (VoD). This is also harmful for the optimization of network-resource usage.

3 Scalable on-demand Policy-based Resource Allocation

It appears nowadays that management systems following the PBM architecture are neither responding to operators' scalability issues nor to customers needs. In fact, customers are willing to dynamically request network-resources depending on their instantaneous needs and without having to contract a SLA for long period of time. However, from the operator perspective, the integration of dynamic resource allocation to the existing IETF's PBM architecture is not feasible in a large scale.

In order to overcome these limitations, we proposed in a previous work [1] a novel solution for on-demand policy-based resource allocation in IP networks. This solution aims to distribute the decision making operations among several distributed PDPs.

Therefore, the PBM architecture has been decomposed into a set of functional components. The idea of this decomposition is to identify which components represent critical sections in the decision-making process. Once this phase achieved, the solution consists on proposing a new instantiation model where non-critical components are distributed according to none functional requirements (such as performance objectives, network size, etc.). Hence, the impact of critical operations on the overall management system performances is minimized. To maintain the consistency of the decision-making process, critical operations are kept centralized. These operations are identified as those operations that need to access to critical resources (shared information, common databases, etc.) in the system.

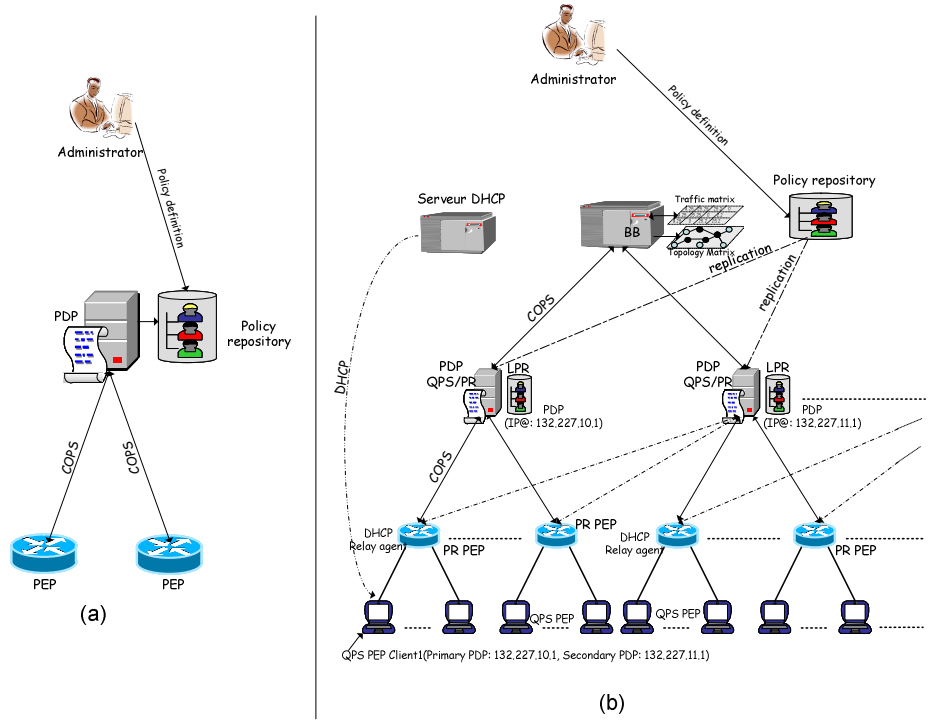

Fig. 1. Policy-based Management: (a) the IETF framework, and (b) our framework.

The critical operations identified in our framework are identified as those related to the bandwidth brokerage. All other operations related to decision making appeared as replicable. Based on these statements verified in our previous work [1], we propose to keep centralized the Bandwidth Brokerage while distributing all other functional components. Fig. 1 presents in details our framework (Fig. 1(b)) and shows its differences with the IETF's PBM framework (Fig. 1(a)).

As our objective is to demonstrate its scalability features, we both realized a complete implementation and a detailed analytical analysis of our proposed framework. The practical experiments highlighted the scalability property of our approach. These experiments also permitted to identify the effect of each component of the framework on the overall performance of the management system. The obtained practical results

demonstrated that the overall-system delay is always below the ITU-T recommended signaling-delay limit [5] and that the system throughput is higher than the 200 req/s as recommended by the ITU-T [6]. However, the system throughput has an upper bound. The analytical study confirmed that the bottleneck of such framework is the bandwidth broker. Undeniably, we demonstrated throughout our analytical model that the system throughput is inversely proportional to the average computational time of the BB element. This parameter is recognized as having a major influence on the size of a domain (number of customers). Hence, we concluded that the performances of the BB element will always drive the performance of the framework we proposed.

4 Bandwidth Brokerage and Problem Statement

Bandwidth Brokers uses generally traffic matrices in order to store and manage network resource usage. Let's first explain how these traffic matrices are organized. In fact, the traffic matrix representation depends on the network technology used, such as the use of the Multi-Protocol Label Switching (MPLS). In our work, we are interested in the most general case for traffic matrix representation. As proposed in [4], a traffic matrix of a particular administrative domain contains several management information bases (MIBs). Among them, one can find a topology information base and a link-QoS information base. The topology information base helps to identify the set of links forming a particular path while the link-QoS information base contains information about the available resources within each link. Using these two specific MIBs, the BB element can achieve admission control and update the amount of available resources on all path-links accordingly. The admission control procedure involves checking each link on the path to verify whether or not it has sufficient resources to satisfy a particular resource request. The traffic matrix is updated whenever a new resource allocation is granted or a resource release is achieved. The traffic matrix described here is clearly a critical section. It can not be accessed simultaneously by several requests as some inconsistent states can appear leading to deadlock situations. As the BB in our on-demand policy-based resource allocation framework uses this operational mode, it appeared as the bottleneck of the overall management system.

One solution to overcome this limitation is to instantiate several bandwidth brokers in the domain. In this case, each BB will be responsible for managing a part of the network resources. In order to avoid conflicts between BB, the idea is to allocate distinct parts of link-resources to the various bandwidth brokers. The authors in [4] have proposed an interesting approach to achieve this. They proposed to use a hierarchical scheme for link-resource distribution over multiple BBs. In this scheme, a centralized Bandwidth Broker (cBB) is in charge of managing link-resources while a set of edge Bandwidth Brokers (eBBs) are responsible for managing the resources assigned to a mutually exclusive subset of paths. As a link can be shared by multiple paths, then the cBB will be in charge of allocating/de-allocating the link-bandwidth to paths. This is realized on an on-demand basis. Hence, the cBB is in charge of link-based resource management and has a micro view of the resource management problem when eBBs are in charge of path-based resource management and has a macro view of the resource management problem.

In our work we will use the same concept of hierarchical bandwidth brokerage. We will however implement this concept differently to achieve better performances.

5 PTT-BB: Proactive Two Tier Bandwidth Brokerage

The proposal in [4] is to virtually divide the bandwidth of each link into *quotas*. Quotas are allocated/de-allocated to paths on an on-demand basis with one quota at a time. Hence, in the case where a particular path is unable to handle an arriving resource request then the corresponding eBB requests the cBB to increase the bandwidth allocated to the corresponding path (quota request). Note also that when an eBB has a quota in excess for a particular path it returns it to the cBB (quota release). If the quota request is granted by the cBB, the eBB accepts the resource request (normal mode). Otherwise (critical mode), two different behavior models are considered: a *non-lossy-path* model and a *lossy-path* model. In the former the cBB keep a part of link-resources shared and centralizes its management. Consequently, when the quota request is not granted by the cBB, the eBB forwards the per-session resource request to the cBB which tries to satisfy it thanks to the shared resource part. Contrarily to this first model, in the *lossy-path* model when a quota request fails, the eBB simply rejects the per-session resource request, instead of passing it to the cBB. In this latter model, the cBB distributes all link-resources to paths and does not maintain a shared part. It then only performs quota management.

Even if the resource usage is not optimized when using the lossy-path model, this latter clearly decreases the processing overhead of the cBB. This is very important as the cBB is the bottleneck of the bandwidth management system. The scalability of the system remains however highly limited by the fact that a quota request is sent to the cBB each time a resource request targeting a critical path is received by the eBB. Indeed, from our previous experience [1], we argue that the quota request rate that can be handled by the cBB is upper bounded. In fact, the cBB can be analytically modeled as a single server queue (as it processes quota-requests sequentially). Let's assume that the cBB service law is approximated by an exponential distribution with a mean of μB. Then, the maximum quota request rate that can be handled by the cBB within the critical mode can be computed using the formula: $1/\mu B$. Furthermore, this assumes that no quota releases are generated towards the cBB for the mean time. Otherwise, the maximum quota request rate will be smaller. We can then conclude that in large scale networks and when the resource usage is very high (i.e. several paths are running under the critical mode), the resource request rate can be very high and therefore the cBB will not be able to manage all the quota-requests. For these reasons, we propose to use a proactive scheme for quota management rather than the on-demand quota management scheme introduced in [4].

In the scheme we propose, quota requests are set-up proactively once the available resources for a particular path are below a certain threshold (T_{REQ}). Similarly, quota releases are triggered proactively once the available resources for a particular path are above a certain threshold (T_{REL}). Quota requests and releases can only be achieved in periodical cut-off times. As the quota request rate that can be handled by the cBB is upper bounded, the periodical cut-off times should be chosen in accordance to the

number of eBB and to the processing capabilities of the cBB. In addition to the fact that our proposal have a better scalability features than the one proposed in [4], it also minimizes the time needed for handling resource allocation request. Indeed, if there are not sufficient resources on the corresponding path once a resource request is received, this latter is immediately rejected by the eBB without referring to the cBB.

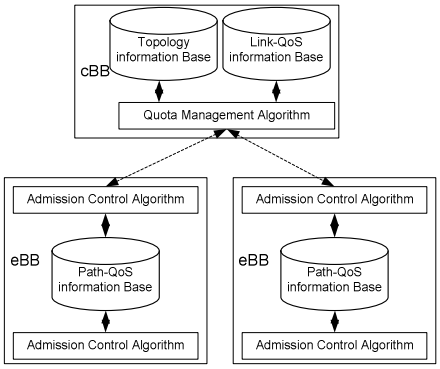

Fig. 2. PTT-BB Architecture.

Following the description of the basic design features of our Proactive Two Tier Bandwidth Brokerage (PTT-BB) architecture, let's now give a more formal and detailed description. Fig. 2 highlights the set of functional components of PTT-BB. Hence, as introduced before, we use a hierarchical scheme composed of a centralized Bandwidth Broker (cBB) and a set of edge Bandwidth Brokers (eBB). The cBB uses a topology information base and a link-QoS information base in order to realize per-path quota allocation/de-allocation. The quota management algorithm is depicted in Fig. 4. Each eBB is assigned a mutually exclusive subset of paths which it stores within a path-QoS information base. Each eBB performs two different resource management operations: the Proactive Path-Bandwidth Adaptation (PPBA) algorithm and the path-oriented admission control algorithm. Fig. 5 and Fig. 6 present in details these two algorithms. Fig. 3 outlines the notations used in the different algorithms.

As depicted in Fig. 6, the PPBA algorithm periodically checks the resource availability of all the paths managed by the eBB. If the available resources level is under a fixed threshold T_{REQ} (respectively above a fixed threshold T_{REL}) for a particular path then the PPBA algorithm requests the cBB to increase (respectively decrease) the amount of resources allocated to this one. In both cases, the eBB asks the cBB to increase (respectively decrease) its path-resource availability in order to reach T_{REQ}+N*$Quota$, respectively T_{REL}-N*$Quota$. This is undertaken in order to always maintain the resource availability level between the following values [T_{REQ} , T_{REL}] for each path. Note that an appropriate choice of T_{REQ} , T_{REL} and N permits to optimize the resource usage while minimizing the call blocking probability in the network. However, the optimization of these parameters is outside the scope of this paper and will be the subject of a future work.

- Req (s,d).Bw and Rel (s,d).Bw : resource allocation and release requests of Bw between two edge routers identified by their interface addresses s and d.
- Q_{BW} : bandwidth unit (or quota) size in bits per second.
- P(s,d) : path allowing to handling the communications between two addresses s and d.
- P(s,d).A_{BW} : Available bandwidth along the path P(s,d).
- P(s,d).Q : the number of quotas allocated to the path P(s,d)
- L(i,j) : link between two routers identified by their interface addresses I and j.
- L(i,j).A_{BW} : Available bandwidth along the link L(i,j).
- PTT-BB.Time : Adaptation Time Interval

Fig. 3. Notations.

```
1:  If Message[i] == msg(Req, P(s,d), ReqQuota) {
2:          If all L(i,j) ∈ P(s,d)) verify
    (L(i,j).A_BW>ReqQuots
3:                  For each link L(i,j) ∈ P(s,d)
4:                          L(i,j).A_BW - ReqQuota;
5:                  Msg (Req, P(s,d), granted);}
6:  If Message[i] == msg(Rel, P(s,d), ReqQuota) {
7:          For each link L(i,j) ∈ P(s,d)
8:                  L(i,j).A_BW + ReqQuota;}
```

Fig. 4. Quota management algorithm.

```
1:   Arrival of a Req (s,d).Bw
2:   If  (Req (s,d).Bw < P(s,d).A_BW) {
3:          P(s,d).A_BW = P(s,d).A_BW - Req (s,d).Bw;
4:          Accept Req (s,d).Bw;}
5:   Else Reject  Req (s,d).Bw;
6:
7:   Arrival of a Rel (s,d).Bw
8:   P(s,d).A_BW = P(s,d).A_BW + Rel (s,d).Bw;
```

Fig. 5. Path oriented admission control algorithm.

```
1:  While (true) {
2:      For each path P(s,d).Bw in eBB {
3:          NbQuota = P(s,d).A_BW / Q_BW;
4:          If (NbQuota < T_REQ)
a.              Msg.add (Req, P(s,d), (T_REQ – NbQuota)+N*Q_BW);
5:          If (NbQuota > T_REL)
a.              Msg.add (Rel, P(s,d), (NbQuota – T_REL)+N*Q_BW); }
6:      Send (Message)
7:      Sleep (PTT-BB.Time); }
```

Fig. 6. Proactive path-bandwidth adaptation (PPBA) algorithm.

6 Performance Evaluation

In order to analyse the performances of our PTT-BB, a test-bed has been implemented and a set of intensive experimentations have been carried out. The objective of these experiments is to compare the performances of PTT-BB to those obtained using a single BB. Both these solutions have been implemented and integrated to the test-bed.

Two performance parameters are analyzed: the call blocking probability and the scalability gain.

6.1 Call Blocking Probability

The experimentations have been carried out using two different topologies (Fig. 7). These topologies are referred to as the peer-to-peer topology (Fig. 7.a) and the chain topology (Fig. 7.b). In the case of PTT-BB and for each topology, the paths to manage have been distributed among the 3 eBBs used in our test-bed.

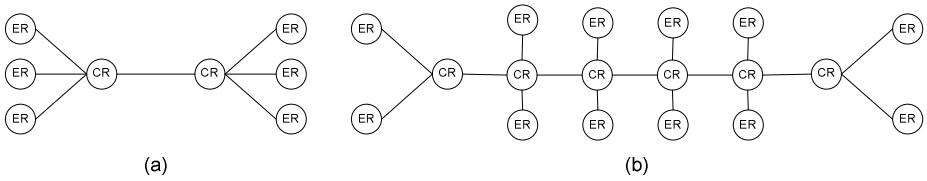

Fig. 7. Experimentation topologies: (a) the peer-to-peer topology, and (b) the chain topology.

Resource-requests are generated by a Poisson process with a rate λ equal to 300 requests/s. Each session lifetime is exponentially distributed and its average duration is μ^{-1} = 30 (in seconds). Session throughputs are chosen randomly in the interval [0.5Mbps, 1.2Mbps]. Note that each link capacity is set to 1 Gbps. The path is also chosen randomly according to the topology. These values have been chosen in order to simulate a highly load network. Several experimentations have been carried out by changing one of the two parameters: the adaptation time interval and the bandwidth unit (or Quota) size. Each experimentation-duration is 200s.

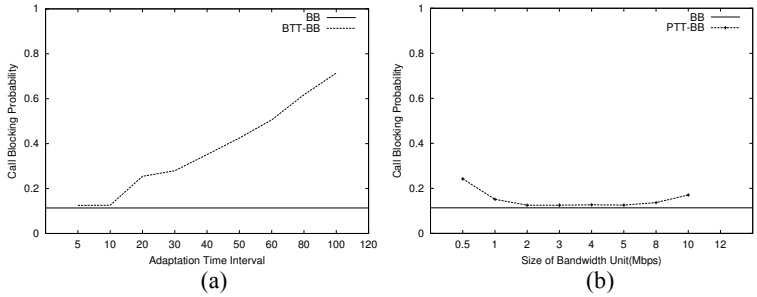

Fig. 8. Call blocking probability, in the peer-to-peer topology, as a function of: (a) adaptation time interval, and (b) quota size.

In all our experimentations, T_{REQ}, T_{REL} and N were chosen as being multiple of the bandwidth unit (*Quota*) size. Hence, T_{REQ} was chosen equal to *5*Quota*, T_{REL} to *10*Quota* and N to *2*Quota*.

The obtained results throughout the experimentations carried out using the peer-to-peer topology are depicted in Fig. 8. Those obtained using the chain topology are depicted in Fig. 9. Both curves depicted in Fig. 8 (respectively Fig. 9) shows the evolution of the call blocking probability while changing the adaptation time interval and

the bandwidth unit (or quota) size. In the former the quota size is set to 5Mbps while in the latter the adaptation time interval is set to 10s. The evolution of the call blocking probability for PTT-BB is also compared to the one obtained with a single BB.

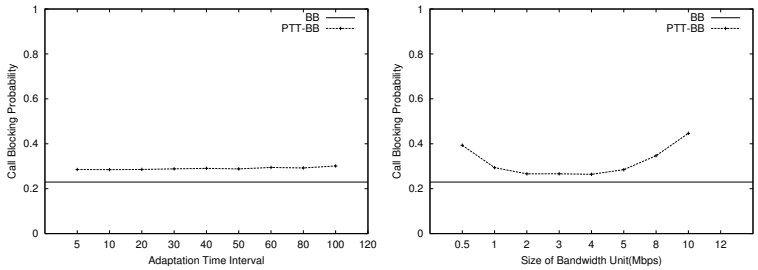

Fig. 9. Call blocking probability, in the chain topology, as a function of: (a) adaptation time interval, and (b) quota size.

From Fig. 8.a we can see that the call blocking probability obtained with PTT-BB is slightly higher than the one obtained with a single BB for adaptation time interval less than 10s. Even if 10s corresponds already to a low adaptation frequency (as the cBB can support even higher loads), we have also performed experimentations for a higher adaptation time intervals (i.e. lower adaptation frequencies). With the decrease of the adaptation frequency, we have noticed an important increase in the call blocking probability. This is not surprising, as the request rate to be handled by PTT-BB is very high and the evolution of the bandwidth allocation to paths is very slow. Consequently, the resources allocated to a path are rapidly consumed and the number of unsatisfied requests becomes significant. This is mainly due to the fact that the number of paths in the peer-to-peer topology is very small (30 paths). Thus, the per path request rate is very high. In parallel, we can see from Fig. 9.a that the call blocking probability remains always stable. In this case, the number of paths (132 paths) is higher than in the case of the peer-to-peer topology. In this second case, the per-path request rate is smaller. The adaptation time interval should then be chosen carefully in accordance to the per-path expected session arrival rate.

Fig. 8.b and Fig. 9.b show that the call blocking probability obtained with PTT-BB stabilizes for quota sizes between 2Mbps and 5Mbps. Furthermore, for this interval, the call blocking probability is slightly higher than the one obtained with a single BB. However, for both cases: quota sizes smaller than 2Mbps or quota sizes higher than 5Mbps, the call blocking probability increases:

1. In the first case, this is due to the fact that PBBA tries to maintain the amount of available bandwidth within the [*5*Quota*, *10*Quota*] interval. This interval is very small when the quota size is small. Consequently, the available resources for each path are rapidly consumed. The number of rejected requests will then be elevated.
2. In the second case, elevated quota sizes implies that an important part of the bandwidth is maintained unused by certain paths. Indeed, in this case also PBBA tries to maintain the amount of available bandwidth within the [*5*Quota*, *10*Quota*] interval. Therefore, all the link-bandwidth can be allo-

cated by the cBB to its paths. However, some paths will have a higher available bandwidth while others are running under the critical mode (i.e. rejecting resource request).

Hence, the quota size should also be appropriately dimensioned. It should be chosen according to the average bandwidth requirement of typical flows.

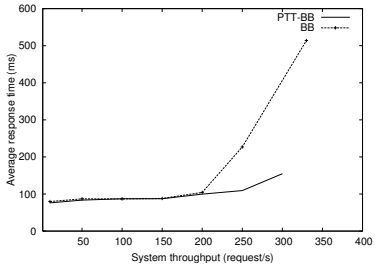

Fig. 10. Average response time to a request as a function of the system throughput.

6.2 Scalability Gain

After demonstrating the performances of PTT-BB in terms of call blocking probability, let's now analyze its behaviour in term of scalability gain. Fig. 10 summarizes the results obtained in our scalability experimentations. It highlights the evolution of the time needed to handle a resource request obtained for different resource request rates (system throughput). This time is referred to as overall-system delay in the following. Note that for these experimentations, PTT-BB uses two eBBs, the quota size is set to 5Mbps while the adaptation time interval is set to 10s. We also used the same values for T_{REQ}, T_{REL} and N as in the previous experiments.

From Fig. 10, one can note that for rates over 200 requests per second, the slope of the response time obtained with a single BB increases drastically. Indeed, the BB element has to process one resource allocation request (REQ) or resource release request (REL) at a time (sequential treatments). Moreover, one should note the load of the BB is doubled compared to the system throughput, i.e. the BB has to process sequentially 400 requests per second (REQ or REL) when the system throughput is 200 requests per second. On the other hand, the slope of the response time obtained when using PTT-BB remains almost stable over time. This is due to three main reasons. The first one is that a per-path resource management at the eBB level involves smaller processing overhead than a per-link resource management as realised by the single BB. Furthermore, two resource requests involving two different paths are treated in parallel as they do not use the same "resource". This can not be the case with a single BB as we have also to check if the two set of links involved by the two paths are mutually exclusive or not. Last but not least, the bottleneck in the system, which is the cBB, is heavily loaded as each eBB sends path-quota requests each 10s. The fact that PTT-BB is more scalable than a single BB is therefore not a surprising conclusion.

7 Conclusion and Future Work

In this paper, we presented a novel approach to achieve a scalable resource management within the scope of the on-demand policy based resource allocation process. The scalability property of the proposed solution (PTT-BB) has been achieved through the distribution of the bandwidth brokerage operation among several mutually exclusive components. PTT-BB is based on a hierarchical management of network-resources. This management involves several edge-BB that realize a per-path resource management and a centralized-BB which proactively allocates/de-allocates link-resources to paths. Hence, a set of algorithms have been designed in order to: (i) realized a per-path admission control within the eBB, (ii) proactively adapt the amount of resources allocated to each path, and finally (iii) realize a per-link resource allocation/de-allocation to paths within the cBB.

In order to demonstrate the performance features of PTT-BB, an intensive set of experimentations have been carried out and the performance results have been compared to those obtained when a single BB is used. These experimentations have demonstrated that a system using PTT-BB is more scalable than a system using a single BB. Indeed, the average response time to a request remains almost stable with PTT-BB and this remains true in spite of the system throughput increase. This is not the case when using a single BB. Furthermore, we depicted throughout our experimentations that the call blocking probability of PTT-BB can be similar to the one obtained with a single BB. However, in the case of PTT-BB, two parameters have to be carefully chosen in order to control the call blocking probability: the adaptation time interval and the bandwidth unit (or Quota) size.

As a perspective to the work presented in this paper, we target to investigate the dynamic optimization of the parameters used in the proactive path bandwidth adaptation algorithm. The objective of this future work is to minimize the call blocking probability while optimizing network resource usage.

References

1. Haddadou, K., Ghamri-Doudane, S., Ghamri-Doudane, Y., and Agoulmine, N.: Designing Scalable on-demand Policy-based Resource Allocation in IP Networks. Technical Report under submission (2005).
2. Verma, D.C.: Policy-Based Networking–Architecture and Algorithms. New Riders Publishing, Indianapolis (2000).
3. Boyle, J., *et al*: The COPS (Common Open Policy Service) Protocol. RFC 2748 (2000).
4. Zhang, Z.-L., Duan, Z., and Hou, Y. T.: On Scalable Design of Bandwidth Brokers. IEICE Transactions on Communications, Vol. E84-B, No. 8 (2001).
5. ITU-T Recommendation No. E.721: Network grade of service parameters and target values for circuit-switched services in the evolving ISDN. (1999).
6. ITU-T Recommendation No. E.500: Traffic intensity measurement principles. (1998).