POLITECNICO DI TORINO
Repository ISTITUZIONALE

Transparent Bandwidth Aggregation for Residential Access Networks

Original
Transparent Bandwidth Aggregation for Residential Access Networks / Duan, Y., Giaccone, P., Castrogiovanni, P., Mana, D., Borean, C., Rossi, C.. - In: TRANSACTIONS ON EMERGING TELECOMMUNICATIONS TECHNOLOGIES. - ISSN 2161-3915. - ELETTRONICO. - 28:6(2017). [10.1002/ett.3138]

Availability:
This version is available at: 11583/2656264 since: 2017-11-13T18:13:04Z

Publisher:
John Wiley & Sons, Ltd

Published
DOI:10.1002/ett.3138

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

06 September 2026

Transparent Bandwidth Aggregation for Residential Access Networks

Yufeng Duan[1], Paolo Giaccone[1*], Pino Castrogiovanni[2], Dario Mana[2], Claudio Borean[2], Claudio Rossi[3].

[1]Dept. of Electronics and Telecommunications, Politecnico di Torino [2]Telecom Italia [3]Istituto Superiore Mario Boella (ISMB)

ABSTRACT

We propose, implement and evaluate a bandwidth aggregation service for residential users that allows to improve the upload throughput of the ADSL connection by leveraging the unused bandwidth of neighboring users. The residential access gateway adopts the 802.11 radio interface to simultaneously serve the local home users and to share the broadband connectivity with neighboring access gateways. Differently from previous works, our aggregation scheme is transparent both for local users, who are not required to modify their applications or device drivers, and for neighboring users, who do not experience any meaningful performance degradation. In order to evaluate the achievable performance and tune the parameters driving the traffic balancing, we developed a fluid model which was shown experimentally to be very accurate. Our proposed scheme is amenable to efficient implementation on Linux networking stack. Indeed, we implemented it and tested in some realistic scenarios, showing an efficient exploitation of the whole available bandwidth, also for legacy cloud storage applications.
Copyright © 2017 John Wiley & Sons, Ltd.

*Correspondence
E-mail: paolo.giaccone@polito.it

1. INTRODUCTION

The growing popularity of high-speed Wi-Fi technologies (as IEEE 802.11n) deployed in residential networks has exacerbated the inequality between the bandwidth available in local domestic networks and the bandwidth available to access the Internet Service Provider (ISP) network. Notwithstanding EU plans for fast broadband foresee 100% coverage of 20 Mbps (or more) access connections for EU citizens by 2020, the majority of member states are still on the way to support basic broadband access connections, based on ADSL technology. Basic ADSL provides a download bandwidth (around 1-10 Mbps) and an upload bandwidth much smaller (around 0.1-1 Mbps), typically lower than local area networks based on Ethernet and Wi-Fi. Thus, ADSL link constitutes often the performance bottleneck for residential users accessing Internet-based applications in which upload traffic is dominant (as cloud storage and computing). Consequently, several methods have been recently proposed to increase the performance perceived by the domestic users through aggregating the available bandwidth of neighboring Wi-Fi Access Points (APs).

This approach is practically enabled by the high density of APs in residential areas, which provides overlapping radio coverage. A recent study [1] showed that, in a metropolitan area, a generic AP can see up to 52 neighbors, with a median value around 7. Furthermore, despite the strong correlation of the residential daily Internet traffic at the aggregated level, which follows nicely a day-night sinusoidal traffic shape, the traffic correlation among neighboring users is typically small due to different habits and behaviors of each individual person. Thus the instantaneous traffic of neighboring ADSL connections is often uncorrelated, enabling statistical multiplexing of the traffic among neighboring users. State-of-art solutions for access bandwidth aggregation require modifications at client side, such as custom drivers or specific applications to be installed on users' devices. This makes their deployment not commercially viable due to chipsets variety, to operating systems diversity, and to additional constraints imposed by smartphones and tablets development environments.

In this paper we propose Beyond One's Bandwidth (BOB), a distributed gateway-centric system exploiting the collaboration between multiple Access Gateways (AGs) to provide a higher Internet connection speed

to residential users without any software or hardware modification at the client side. The AG is a standard access device that integrates a broadband modem, a network-layer router, and a Wi-Fi AP. Residential devices such as laptops, smart phones and TVs can access the Internet by associating to the AP or by connecting through Ethernet. We design BOB to meet the transparency requirements of a real commercial deployment. Each AG in BOB is responsible of constructing a dedicated wireless topology with other nearby AGs, forwarding portion of traffic to neighbors while guaranteeing that each user is able to fully exploit his own broadband connection bandwidth, without observing any meaningful performance degradation due to the cooperative sharing scheme.

We provide the following contributions. We propose BOB architecture, based on integrating a novel flow-balancing scheme with a traffic scheduler, both amenable for implementation in Linux networking stack. We develop a fluid model to estimate the achievable performance in terms of throughput and fairness based on the traffic distribution policy adopted in BOB. This model allows to compute the optimal parameters of the architecture to maximize the performance, given the knowledge of the user traffic rate. Finally, we implement and test BOB in an operational scenario with several typical clients. The results show that BOB can provide a substantial increase of the throughput with negligible overhead under synthetic traffic and real applications. Note that a preliminary version of our work was published in [2].

The paper is organized as follows. In Sec. 2 we discuss the relevant previous work. In Sec. 3 we describe the proposed architecture, whose detailed implementation is discussed in Sec. 4. In Sec. 5 we present the fluid model, which is evaluated numerically and validated experimentally in Sec. 6. Finally, in Sec. 7 we assess experimentally the performance of BOB and draw our conclusions in Sec. 8.

2. RELATED WORK

In recent years, many works have been focused on the bandwidth aggregation problem in the context of the access network. FatVAP [3] is one of the first works on aggregating the access broadband bandwidth of multiple APs. FatVAP has two main contributions: an 802.11 driver that enables a client to connect to multiple APs using a single radio wireless chip; a scheduler that enables a client to decide to which APs to connect to maximize the throughput. Unfortunately, the proposed approach is not suitable for domestic users of ISPs, as targeted by our work, since it is not transparent for the user, who is required to install a new driver that is not integrated in an off-the-shelf operating system and maybe not supported by the available chipset of the wireless card. In addition to that, [3] does not involve an authentication procedure when connecting to an AP and thus cannot be adopted in a residential access network. The work in [4] extends [3] and addresses specifically the security issue when connecting to multiple APs. It proposes a fast authentication mechanism, but it is not compatible with legacy 802.11 security protocols. Furthermore, clients connect directly to each other using a virtual interface connected in ad-hoc mode, limiting severely the portability of the approach. Finally, whenever a client shuts down, it cannot share anymore the backhaul bandwidth to others, even if its access gateway is still working. In [5] a new scheduling scheme is proposed that guarantees a fair access among multiple clients, which overcomes the drawback of FatVAP that only maximizes the performance of a single client, neglecting the potential unfairness among all the clients. Finally, [6] proposes to aggregate the access bandwidth by exploiting the transmission on overlapping channels, but it relies completely on a non-standard communication technology. In [7] a new opportunistic approach is proposed to aggregate the backhaul bandwidth. The main idea is that the client sends a packet in broadcast towards all the APs and each AP runs a scheduler that decides whether to forward this packet. As a result, it requires a strong cooperation among the clients, APs, and the servers, only achievable with customized devices. Furthermore, broadcast communications in 802.11 are inefficient in terms of bandwidth, since occurring always at the minimum data rate.

One of the most relevant work to ours is SmartAP [8]. Evolving from [5], SmartAP develops an AP-based scheduling algorithm that tries to maximize the overall throughput of all the clients. SmartAP is transparent for the user, since it does not require modification on the applications, the wireless card driver, or the operating system. An optimization algorithm coordinates the traffic flows among neighboring APs in order to maximize the bandwidth. To achieve some transparent behavior for the local user of an AP, each AP must estimate its own available bandwidth and communicate to a central controller running the optimization algorithm. Due to the latencies for the bandwidth estimation and the centralized control, the approach slowly reacts to fast varying loads. Unlike SmartAP, our approach is completely distributed, since each Access Gateway runs a traffic control scheme independently from the others; in addition, our scheme preserves the bandwidth available for the local traffic. Another relevant work is 3GOL [9], which shares the same motivation of our work, i.e. boosting the speed of ADSL users. Instead of aggregating the bandwidth of multiple APs, [9] proposes to exploit a parallel cellular connection to increase the speed of a home user. The proposed approach is not transparent for the user, which must install a dedicated scheduler in her access devices.

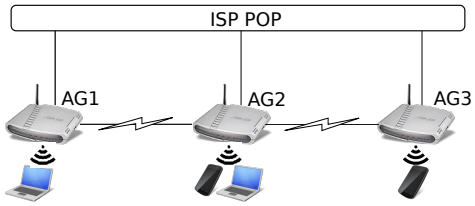

Figure 1. Example of a scenario with three cooperating AGs running BOB

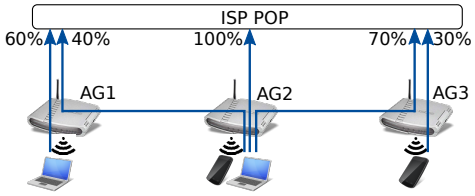

Figure 2. Example of a bandwidth sharing scenario achieved by BOB

3. COOPERATIVE BANDWIDTH AGGREGATION

We consider a residential access network of an ISP, in which each household is equipped with an AG. The AG is connected through an ADSL line to the ISP's POP and provides connectivity to users' devices through an 802.11 interface. Fig. 1 shows a basic example of 3 AGs that cooperate to implement our proposed BOB bandwidth aggregation scheme. The wireless interface of each AG is connected to the *local* devices but also to the *neighboring* AGs. In the example, AG2 is connected to both AG1 and AG3. BOB is designed to exploit the unused ADSL bandwidth of the neighboring AGs to boost the performance of the local devices. Fig. 2 shows an example in which the local devices of AG1 and AG3 are currently exploiting 60% and 30% of their own ADSL upload bandwidth, respectively. Thanks to the cooperation of the two neighboring AGs, the local devices of AG2 exploit not only their local ADSL bandwidth, but also the unused ADSL bandwidth of AG2 and AG3 to/from the POP. Assuming all the ADSL link rates being the same, a 2.1× gain would be experienced on the overall uplink bandwidth.

We designed the cooperative bandwidth aggregation system by considering the following constraints. *(1) Transparent performance*: the users' QoS should not be affected negatively by the sharing scheme. This means that the local user should be able to fully exploit his local ADSL bandwidth, independently from the neighbors' behavior. If some performance degradation is experienced, it should be negligible. *(2) Transparent deployment*: the system should not require any modification of the applications and of the Wi-Fi interface drivers at the users' devices, and should be compatible with the most common existing Internet transport protocols and with standard IP routing. *(3) Single wireless interface*: to reduce costs, the AG should exploit a single Wi-Fi physical interface to connect both the local users and the neighboring AGs. *(4) Self-configuring scheme*: the cooperation scheme must be enabled in a distributed way without the need of a central control. All the previous constraints are dictated by an ISP who wishes to scale the approach to a large population of users, by providing a proprietary low-cost AG to its subscribers.

To meet all the previous design constraints, we combined many techniques. First, to achieve transparent performance, a traffic scheduler running in the AG regulates the traffic flows contending for the ADSL bandwidth. This guarantees that most of the ADSL bandwidth is devoted to the local devices, whereas a small bandwidth (negligible for the local user) is given to the neighboring AGs to avoid the starvation of active TCP flows. Sec. 3.3 will be devoted to describe the details of such traffic scheduler. Second, to achieve transparent deployment, the AG is entirely responsible to route the traffic flows. An internal flow-based load balancing scheme, described in Sec. 3.2, route part of the incoming TCP/UDP flows to the neighboring AGs, seamlessly for the local devices and for the whole ISP network. Thus, neither hardware or software modification is required in the users' devices and the bandwidth sharing scheme can work with all the different 802.11 standards currently available. We wish to emphasize that such *transparent deployment* is the main difference of our approach with respect to the state of art [3, 4, 5]. The constraint about the single wireless interface poses some natural limitation regarding the scalability of the approach, since a common channel must be shared across all the cooperating AGs. Finally, to allow the cooperation among the AGs, we build a logical interconnection topology among the neighboring AGs, as detailed in the following Sec. 3.1.

3.1. Communication topology

The topology connecting the AGs and their local devices requires multiple connections at MAC layer for each AG. Since the network interface is single, this configuration can be achieved by defining many virtual interfaces on the Wi-Fi physical interface. One virtual interface, referred as "*private AP*", is devoted to the local user (more precisely, to all the domestic Wi-Fi devices) and acts as a standard 802.11 AP establishing a BSS for the local devices. Another virtual interface, denoted as "*public BOB AP*", provides the connectivity to the neighboring AGs acting as access point. Finally, one or more virtual interfaces, referred as "*BOB station*", allows the AG to connect to the public BOB AP interfaces of other cooperating AGs. Each virtual interface is configured with a MAC address chosen automatically in increasing order with respect to the original MAC of the physical interface, to avoid conflicts at MAC layer. Fig. 3 shows an example of a

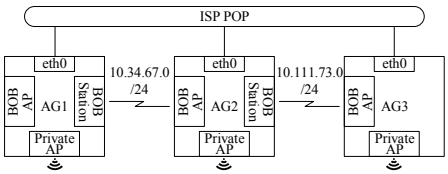

Figure 3. Layer-2 topology highlighting the role of each kind of virtual interface.

topology achievable by the three virtual interfaces present in each AG.

The formation of the layer-two topology is achieved by a simple distributed algorithm, as follows. Each AG periodically scans for public BOB AP interfaces. Whenever it finds a new one, if not yet connected to it, it associates to it and establishes the bidirectional cooperation among the two AGs. To enable such process, each public BOB AP is identified by a BSSID composed by combining the unique identifier of the AG (obtained by the native public AP BSSID) and the string "BOB". This allows to understand whether one AG is already associated to another AG and to avoid double associations (as when a single AG acts as both public AP and station towards another AG). After establishing the layer-two connectivity among the AGs, the public AP runs a DHCP server to provide the IP address to the BOB-station interfaces. To avoid conflicts of IP network addresses, the public AP is set in the range 10.X.Y.0/24, where X.Y are chosen according to a 16-bit hash function applied to the string of native AG BSSID. In addition to the above association scheme, the cooperation between two AGs is actually established only if the RSSI is above a given threshold. This guarantees that only neighboring AGs cooperate when they have a reasonable good connectivity.

3.2. Flow-level balancing

All the incoming traffic on the private AP interface is processed by a flow balancer, to eventually distribute the traffic across different neighboring AGs. The balancer works at transport layer and identifies the traffic based on the standard pair of transport ports and network addresses. When the first packet of a new flow reaches the balancer, a load balancing algorithm selects the local destination, that is either the local interface of the ADSL connection, or the BOB interfaces (BOB-station or BOB-AP). Note that, since all the traffic is routed through a NAT to reach Internet, the first packet of a flow is always generated by a local device. Furthermore, given that an unique public IP address is associated to each AG, all the following packets of the same flow need to be forwarded along the same path of the initial packet. Note that this choice also avoids out-of-sequence packets within a flow, which are very poorly tolerated by TCP/UDP protocols, and it is compatible with standard IP routing also for the backward path.

We propose two possible flow-balancing schemes. The first scheme is based on a classical *Weighted Round Robin* (WRR) in which the number of flows sent on each BOB-interface and to the ADSL connection is proportional to the maximum bandwidth available in each interface. Such value can be estimated approximatively by the association data rate of each wireless interface and by the sync rate of the ADSL link. The AG must maintain an Interface Rate Table (IRT) with the updated data rates of each interface. The second scheme is denoted as *Pending Flow Balancing* (PFB) and is designed to distribute each new flow to the specific local interface (either BOB or ADSL) with the minimum number of pending active flows. A TCP flow is defined as "pending active" during the period between the initial SYN handshake and the final FIN handshake. An UDP flow is similarly defined after the initial packet and until a timeout expiration after the last observed packet. The AG must maintain a Pending Flow Table (PFT) that keeps track of the numbers of active flows on each interface.

If we assume equal flow sizes, WRR tends to balance the load across the BOB interfaces obliviously of the actual bandwidth that different paths would experience, which depends on the local congestion in each neighboring AG. In a worst-case scenario, all elephant flows will be directed to lower bandwidth paths whereas mice flows to the higher bandwidth paths, with severe throughput degradation. Instead, PFB self-adapts the load on each path according to its actual available bandwidth, since it concentrates the flows on the paths with the higher throughput, on which the number of pending flows tends to decrease faster. Whenever a sudden reduction of the available bandwidth is experienced (due for traffic fluctuations and to the implemented traffic scheduler), the corresponding flows will start to experience some temporary starvation and the number of pending flows along the path will not decrease, thus raising the probability that new flows will be routed along alternative paths with better available bandwidth. Thus, PFB is also preferred for asymmetric links, as it balances the load according to the actual available bandwidth of each link. In Sec. 7.2 we will show an experimental comparison between the performance of the two algorithms, and discuss the performance for asymmetric links.

Note that a flow-level balancing could be inefficient in the case of very few concurrent flows. This is true in general, but in practice the number of active flows by a user is quite large, since many bandwidth-hungry applications open more than one flows, as described later in the evaluation part of the paper (see Sec. 7). An alternative solution to optimize the routing would be to balance the traffic at packet level. In this case, the only option would be to implement a MPTCP [10] proxy within the AG, but this approach requires to install a reverse MPTCP proxy within the POP. Thus, this option is not transparent for the ISP and violates one of our design constraints.

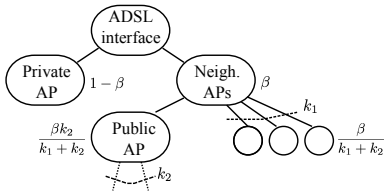

Figure 4. Traffic classes in HTB scheduler with corresponding minimum rates $r_{\min}$, normalized to the ADSL link speed

3.3. Traffic scheduling

A work-conserving traffic scheduler regulates the access to the uplink ADSL interface, and manage the set of output queues associated to such interface. This scheme is crucial to guarantee transparent performance for the local devices of an AG. The flows are scheduled at a fine granularity, i.e. at packet level, according to a Hierarchical Token Bucket (HTB) [11, 12]. HTB can be described by a multilayer tree, in which each node corresponds to a traffic class and in particular the root node corresponds to the main interface towards which all the traffic is sent. Each class is controlled by an internal token bucket and the multilayer topology is used to aggregate multiple classes (i.e. each node defines a new class aggregating all the children classes) and to specify detailed scheduling rules on such aggregation. More in details, each class is assigned with a pair of parameters $(r_{\min}, r_{\max})$, where $r_{\min}$ is the minimum average rate and $r_{\max}$ is the maximum rate that cannot be exceed by the traffic within the class. In a nutshell, when the corresponding queues are backlogged, the class traffic will receive at least $r_{\min}$ bandwidth, but no more than $r_{\max}$. The hierarchy allows one child class to increase its rate by borrowing the unused bandwidth from the ancestor class, providing high flexibility to set the minimum and maximum rates for single and groups of traffic classes.

Fig. 4 shows the hierarchy among the traffic classes defined in BOB. Consider a generic AG to which $k_1 + k_2$ neighboring AGs are associated; k_1 are associated as stations to the public BOB-AP interface, whereas k_2 are associated through multiple BOB-station interfaces. We set $r_{\max} = 1$ (normalized to the ADSL link speed) for all the traffic classes, in order to exploit all the available bandwidth: the scheduler is indeed fully work-conserving. This choice guarantees also that the unused bandwidth of the local ADSL connection can be fully exploited by the neighboring nodes. In addition to this, we impose that $1 - \beta$ (with a small value of $\beta > 0$) fraction of the local ADSL bandwidth must be guaranteed to the local devices, whenever they are actively sending packets. A small β fraction of the ADSL bandwidth is instead devoted to guarantee that the flows from the neighboring nodes will not starve. This minimum bandwidth is divided evenly across all the $k_1 + k_2$ neighboring AGs. Note that this allocation provides a reasonable level of fairness among neighboring AGs that use different data rates to connect wirelessly to the local AG.

4. IMPLEMENTATION

We implemented BOB on Linux and tested it on several laptops with several kernels, namely from 2.6.38 to 3.16. The wireless card was an Atheros AR9285 for which Linux provides a built-in driver (ath9k). We also successfully imported BOB into Arduino YUN with RTL8188CUS wireless card with the default driver provided by Realtek. In the following sections, we describe some relevant design issues for the Linux implementation.

4.1. Communication topology

We implemented the topology formation using a Python script that uses the subprocess module to call external Linux commands. In particular, to create multiple virtual interfaces we used `iw` tool.

4.2. Flow-level balancing

We implemented the algorithms for the flow-level balancer in Python and run it as a daemon. It chooses the path for each flow, according to the WRR or the PFB scheme. When using the PFB scheme, we use the `conntrack` tool to get the information of the active pending flows.

The main implementation issue to address is how to route in a transparent way the packets according to the flow-balancer decision. To solve this, we adopt the sequence of operations described in Fig. 5. Whenever a packet is received by the AG through the local private AP interface, we have two cases. (1) If the packet is the first of a TCP/UDP flow, the flow-level balancer chooses the path (i.e. the local broadband connection or one of the neighboring AG) to route the packet. The packet is marked through an appropriate `iptables` rule based on the chosen path and this is stored in a marking table. The marking allows to use a specific routing table when exploiting a neighboring AG, according to which the default gateway has been set equal to the IP address of the BOB interface present in the neighboring AG. In addition to this, NAT modifies the source IP address to support the routing back on the reverse direction. (2) If the packet is not the first one of a flow, it is marked according to the marking table to choose the same path of the first packet of the flow, and thus the packet is processed by the desired routing table.

4.3. Traffic scheduler

We implemented the traffic scheduler using the native HTB scheduler available in the traffic control `tc` tool [13] available in Linux. Referring to Fig. 4, we set $\beta = 0.01$ to reserve just 1% of the bandwidth for the flows arriving from the neighboring AGs. To be able to classify correctly the traffic entering the scheduler, we exploited the same

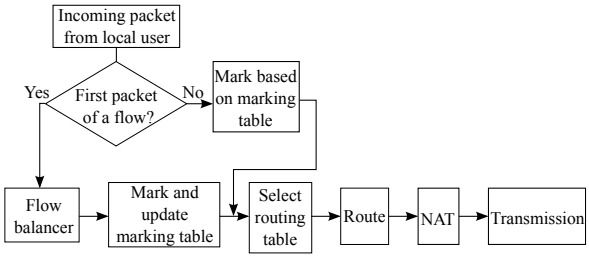

Figure 5. Procedure followed by the flow-level balancer.

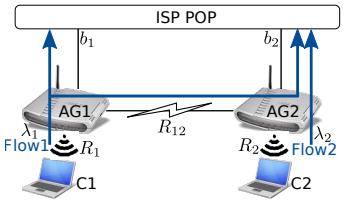

Figure 6. The topology of the single-hop scenario with 2 AGs.

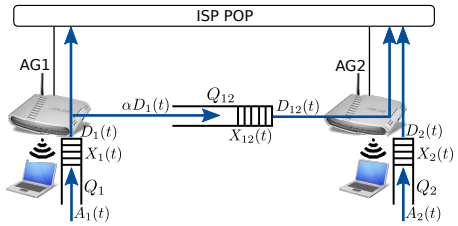

Figure 7. The queueing for the fluid model for the single-hop scenario of Fig. 6.

marking capabilities described before to mark a packet based on the incoming interface.

5. A FLUID FORMULATION OF THE BANDWIDTH SHARING SCHEME

In order to assess the performance gain achievable by the proposed scheme, we consider a distributed upload scenario in which the amount of traffic that must be forwarded to the neighboring nodes must be optimally chosen. Notably, forwarding the local traffic to the neighbor increases also the radio contention for the channel and thus reduces the transmission opportunities for the local user. We will show that it is possible to find the optimal fraction of traffic to be forwarded from the local AG to the neighboring ones.

In the following, we describe a fluid queueing model to compute the optimal fraction of traffic to be forwarded to the neighboring nodes. The model can be numerically solved in an efficient way, and the results apply to a broad family of ingress traffic, since the traffic rate is required just to satisfy the law of large numbers (i.e. an average rate can be defined). The results are coarser than a standard queuing model based on Markov chains, but actually we will show its great accuracy in Sec. 6.2 where we will validate the model in an experimental scenario.

5.1. Single-hop formulation

We start by formulating the model of a basic scenario with two AGs, as shown in Fig. 6, coherent with the network topology described in Sec. 3.1. Let AG1 and AG2 denote the two AGs and let C1 and C2 be the two local users. The ADSL uplink bandwidth of the two AGs are denoted by b_1 and b_2.

We model all the traffic in this scenario with a first-order fluid model. We assume that the users send the traffic at constant rates, λ_1 and λ_2 respectively, towards the POP. Without loss of generality, we consider the case in which C1 is allowed to exploit the ADSL bandwidth of AG2 for the upload traffic. To model the transmission contention on the shared wireless channel, we define three transmission queues, Q_1, Q_{12} and Q_2, which are associated to C1, the interface towards AG2 at AG1, and C2 respectively, as shown in Fig. 7. The traffic generated by C1 firstly enters Q_1. As we allow C1 to use the available bandwidth of its neighbor, part of the traffic departed from Q_1 enters Q_{12}, traverses Q_{12} and goes out through the ADSL link of AG2, while the rest goes out through AG1 ADSL link directly. Since C2 does not exploit the ADSL bandwidth of its neighbor, the traffic generated at C2 just passes Q_2 and exits through its own ADSL link. In our model we do not consider the queues associated to the ADSL uplinks since not affecting the contention for the wireless channel.

According to the 802.11b/g/n protocol, the wireless interfaces adapts the transmission data rate based on the channel condition [14]. Let R_1 and R_2 denote the data rate used from C1 to send traffic to AG1 and from C2 to AG2, respectively. Let R_{12} be the data rate between the two AGs. Notably, the data rate (between 6 and 54 Mbit/s) is not the actual achievable throughput, because of the time sharing among the different transmitters.

We denote the *forwarding factor* as α, with $\alpha \in [0, 1]$, which indicates the fraction of the traffic generated by C1 that AG1 forwards to AG2. We assume α fixed and not changing with the time, since it can be shown that this choice allows optimal bandwidth allocation in the stationary traffic conditions considered in our fluid model.

Consider now queue Q_i. Let $A_i(t)$ denote the cumulative amount of traffic entered the queue from time 0 to time t and $D_i(t)$ the cumulative amount of traffic transmitted from time 0 to time t. Let $X_i(t)$ denote the queue length of Q_i at time t. AG1 forwards a fraction α of the traffic that exits from Q_1, thus we can impose that $A_{12}(t) = \alpha D_1(t)$, for any t. We assume initial null conditions: $A_i(0) = D_i(0) = X_i(0) = 0$, for $i \in \{1, 12, 2\}$. The evolution of the occupancy of the 3 transmission queues can be described by the standard

Lyndley's equations:

$$X_1(t) = A_1(t) - D_1(t) \quad (1)$$
$$X_{12}(t) = \alpha D_1(t) - D_{12}(t) \quad (2)$$
$$X_2(t) = A_2(t) - D_2(t) \quad (3)$$

We now compute the first-order derivative of (1)-(3). Since the arrival rate is constant, thus $\dot{A}_1(t)$ and $\dot{A}_2(t)$ are the average offered load of C1 (i.e. λ_1) and of C2 (i.e. λ_2), respectively. Thus, we can claim:

$$\dot{X}_1(t) = \lambda_1 - \dot{D}_1(t) \quad (4)$$
$$\dot{X}_{12}(t) = \alpha \dot{D}_1(t) - \dot{D}_{12}(t) \quad (5)$$
$$\dot{X}_2(t) = \lambda_2 - \dot{D}_2(t) \quad (6)$$

We now characterize the departure rates $\dot{D}_1(t)$, $\dot{D}_{12}(t)$ and $\dot{D}_3(t)$ based on a standard and basic model for 802.11b [15], which can be adopted also for 802.11g [16, 17]. Even if the model is very simple, in Sec. 6.2 we will show to provide accurate results also in experimental scenarios. All the Wi-Fi interfaces contend for the same channel, but we assume that the relevant traffic is just from C1 to AG1 and from C2 to AG2. All the traffic in the reverse direction (e.g. TCP ACKs) is assumed to be negligible. Thus, we can consider the radio contention occurring from the traffic from C1 to AG1, from C2 to AG2 and from AG1 to AG2 (which is a given fraction of the traffic generated from C1).

Observe that $\dot{D}_1(t)$, $\dot{D}_{12}(t)$ and $\dot{D}_3(t)$ corresponds to the instantaneous service rate of the corresponding interface, which depends on the actual number of contending interfaces on the same channel. Note that an interface is *active* and competes for the channel if and only if its transmission queue is not empty. For the sake of simplicity, we assume that all the hosts transmit frames of length L. The transmission time t_i of a frame at interface i operating at rate R_i is, by definition, L/R_i. We assume an ideal MAC protocol that would allow a perfect fair access to the channel, i.e. a perfect round-robin for the transmissions among all the interfaces with non-empty transmission queues (i.e. for all j such that $X_j(t) > 0$). We can now approximate the duration $T_{\text{tot}}(t)$ of a cycle of transmissions (according to a round-robin) among all the active interfaces at time t as follows:

$$T_{\text{tot}}(t) = \sum_{j \in \{1,12,2\}} \frac{L}{R_i} \mathbb{1}_{X_j(t)>0} \quad (7)$$

where $\mathbb{1}_A$ is the standard binary indicator function, i.e. equal to one iff event A holds. Hence, the instantaneous throughput $\dot{D}_i(t)$ for an active interface at time t corresponds to one frame transmission over a cycle duration T_{tot}:

$$\dot{D}_i(t) = L/T_{\text{tot}}(t) \quad (8)$$

By combining (7) with (8), we can claim

$$\dot{D}_i(t) = \begin{cases} \frac{1}{\sum_{j \in \{1,12,2\}} [\mathbb{1}_{X_j>0}/R_j]} & \text{if } X_i(t) > 0 \\ 0 & \text{if } X_i(t) = 0 \end{cases} \quad (9)$$

In summary, the evolution of the fluid model describing the dynamics of all the queues involved in the wireless channel contention is obtained by solving the set of equations: (4)-(6) (for the queue evolutions) and (9) (for the instantaneous throughput at each queue).

Given the dynamics of the queues described above, assuming a long enough observation time τ, the average throughput T_1 of AG1 and T_2 of AG2 on the ADSL uplinks are the following:

$$T_1 = \min\left\{b_1, \frac{(1-\alpha)D_1(\tau)}{\tau}\right\} \quad (10)$$
$$T_2 = \min\left\{b_2, \frac{\alpha D_1(\tau) + D_{12}(\tau)}{\tau}\right\} \quad (11)$$

Indeed, by referring to Fig. 7, AG1's ADSL throughput (10) is obtained by considering just the fraction of local traffic sent from Q_1 to the ADSL link, but clipped to the maximum ADSL rate b_1. Similarly, the AG2's throughput (11) is obtained by summing the fraction of throughput arriving from AG1 plus the local user, and finally clipped to b_2. Notably, the actual throughput of all the queues depends on the channel contention level, captured by (9).

The aim of the proposed traffic balancing scheme is to find the optimal forwarding factor α^* to maximize the throughput. Formally:

$$\alpha^* = \max_{\alpha \in [0,1]} T_1 + T_2$$

Given the throughput observed at each ADSL link, it is possible to compute the throughput per each user, taking into account that each local user will get highest priority in the traffic to access the ADSL link, as forced by the traffic scheduler implemented in BOB.

Let T_i^u be the throughput obtained by user i associated at its local AGi. Because of the priority of the local user at AG2:

$$T_2^u = \min\left\{b_2, \frac{D_2(\tau)}{\tau}\right\} \quad (12)$$

The remaining ADSL bandwidth $b_2 - T_2^u$ at AG2 is exploited by user 1 and thus his throughput is:

$$T_1^u = T_1 + \min\{b_2 - T_2^u, \alpha D_1(\tau)\} \quad (13)$$

Later in Sec. 6.1 we will solve numerically the fluid model for the single-hop scenario to get the optimal forwarding factor.

5.2. Multi-hop formulation

We extend the single-hop fluid model to the scenario in which 3 AGs cooperate and thus the traffic may traverse

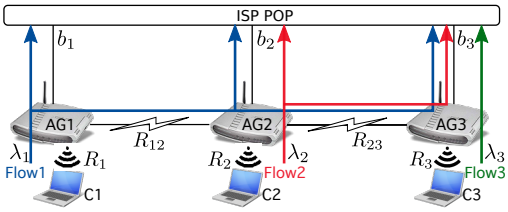

Figure 8. The topology of the multi-hop scenario with 3 AGs.

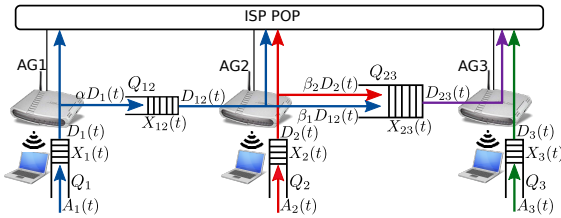

Figure 9. The queueing in the fluid model for the multi-hop scenario of Fig. 8

up to 2 hops before reaching the ISP POP. In addition to identifying the optimal way to distribute the traffic in order to maximize the throughput, we wish to investigate the achieved fairness among different users.

We consider an access network with three AGs, denoted by AG1, AG2 and AG3 respectively, as shown in Fig. 8. Each AG is associated with a single user (C1, C2 and C3, respectively) generating upstream traffic. Furthermore, we have AG1 connected to AG2, and AG2 connected to AG3 through their Wi-Fi interfaces. Finally, as before, all the three AGs are connected to the ISP through ADSL links of which the uplink bandwidth are b_1, b_2 and b_3 respectively. For simplicity, in this scenario C1 exploits the ADSL bandwidth of AG1, AG2 and/or AG3; C2 exploits the one of AG2 and/or AG3; C3 exploits the bandwidth of only its own AG (i.e. AG3). Furthermore, the available bandwidth of an ADSL link is shared equally among the neighboring users (i.e. C1 and C2 have the same priority to access AG3's ADSL bandwidth), coherently with the traffic scheduler adopted in BOB. The users generate the traffic at constant rates, equal to λ_1, λ_2, and λ_3. Using a notation similar to the single-hop scenario, let R_1, R_2 and R_3 denote the Wi-Fi data rate between each user and its corresponding AG respectively. We also have R_{12} denote the Wi-Fi data rate from AG1 to AG2, and R_{23} from AG2 to AG3.

Similarly to the single-hop scenario, we have five transmission queues defined for the wireless interfaces, as shown in Fig. 9: Q_1, the queue at C1; Q_{12}, the queue associated to the interface of AG1 towards AG2; Q_2, the queue at C2; Q_{23}, the queue at AG2 on the interface towards AG3; Q_3, the queue at C3. The traffic at rate λ_1 generated by C1 leaves its transmission queue Q_1. A fraction α of it (with $\alpha \in [0,1]$) is forwarded by AG1 to AG2, passing through Q_{12}. The rest just goes out through the ADSL link of AG1 directly. When the traffic from AG1 reaches AG2, a β_1 fraction of it (with $\beta_1 \in [0,1]$) is sent to AG3 through Q_{23}, while the rest is sent to the ADSL link. AG2 forwards also a fraction β_2 of the traffic generated by C2 (with $\beta_2 \in [0,1]$), to AG3 through Q_{23}. Finally, AG3 gathers the traffic from Q_{23} and the traffic sent from Q_3 (generated by C3), and sends them out through its own ADSL link.

We start to formulate this scenario by modeling the temporal evolution of the queues. Similarly to the single-hop scenario, $A_{12}(t) = \alpha D_1(t)$ and $A_{23}(t) = \beta_1 D_{12}(t) + \beta_2 D_{23}(t)$ because of the forwarding behavior of AG1 and AG2. Assuming $A_i(0) = D_i(0) = X_i(0) = 0, \forall i$, we have:

$$X_1(t) = A_1(t) - D_1(t) \tag{14}$$
$$X_{12}(t) = \alpha D_1(t) - D_{12}(t) \tag{15}$$
$$X_2(t) = A_2(t) - D_2(t) \tag{16}$$
$$X_{23}(t) = \beta_1 D_{12}(t) + \beta_2 D_2(t) - D_{23}(t) \tag{17}$$
$$X_3(t) = A_3(t) - D_3(t) \tag{18}$$

By taking the first-order derivative of (14)-(18), we have:

$$\dot{X_1}(t) = \lambda_1 - \dot{D_1}(t) \tag{19}$$
$$\dot{X_{12}}(t) = \alpha \dot{D_1}(t) - \dot{D_{12}}(t) \tag{20}$$
$$\dot{X_2}(t) = \lambda_2 - \dot{D_2}(t) \tag{21}$$
$$\dot{X_{23}}(t) = \beta_1 \dot{D_{12}}(t) + \beta_2 \dot{D_2}(t) - \dot{D_{23}}(t) \tag{22}$$
$$\dot{X_3}(t) = \lambda_3 - \dot{D_3}(t) \tag{23}$$

Similarly to (9), the instantaneous service rate can be evaluated as follows:

$$\dot{D_i}(t) = \begin{cases} \dfrac{1}{\sum_{j \in \{1,12,2,23,3\}} [\mathbb{1}_{X_j > 0} / R_j]} & \text{if } X_i(t) > 0 \\ 0 & \text{if } X_i(t) = 0 \end{cases} \tag{24}$$

where $i \in \{1, 12, 2, 23, 3\}$.

By solving the equation set (19)-(24), we can obtain the evolution of all the queues in our fluid model and compute the overall throughput and the throughput of each flow. Let τ be a long enough observation time; the corresponding throughput T_i of the ADSL link at AGi is:

$$T_1 = \min \left\{ b_1, \frac{(1-\alpha) D_1(\tau)}{\tau} \right\} \tag{25}$$
$$T_2 = \min \left\{ b_2, \frac{(1-\beta_1) D_{12}(\tau) + (1-\beta_2) D_2(\tau)}{\tau} \right\} \tag{26}$$
$$T_3 = \min \left\{ b_3, \frac{D_{23}(\tau) + D_3(\tau)}{\tau} \right\} \tag{27}$$

Indeed, in (25) the throughput on AG1 ADSL link of AG1 is a fraction of the traffic from Q_1. In (26), the throughput on AG2 ADSL link is the sum of two components: the traffic from Q_{12} which is not further forwarded to AG3 and the traffic from Q_2 which is not redirected to AG3.

Similarly, in (27) the total traffic arrived at AG3 ADSL link is composed of two parts: the traffic generated by C1, forwarded by AG1 and AG2 to AG3, plus the traffic generated by C2 and redirected by AG2 to AG3; the traffic generated by C3.

Our object is to find the optimal forwarding factors α^*, β_1^* and β_2^* that maximize the overall throughput:

$$(\alpha^*, \beta_1^*, \beta_2^*) = \max_{\alpha,\beta_1,\beta_2 \in [0,1]} T_1 + T_2 + T_3$$

To evaluate the fairness achieved among users, we can compute the throughput for each user, obtained by summing the throughput of all his traffic flows and sent out through all the ADSL links. Let T'_{ji} denote the throughput of user j obtained on the ADSL link of AGi. As the bandwidth of AG1's ADSL link is only exploited by C1:

$$T'_{11} = T_1 \tag{28}$$

Based on the adopted traffic scheduler in BOB, on AG2 ADSL link, the throughput contribution for C1 and C2 are the following:

$$T'_{22} = \min\left\{b_2, \frac{(1-\beta_2)D_2(\tau)}{\tau}\right\} \tag{29}$$

$$T'_{12} = \min\left\{\frac{(1-\beta_1)D_{12}(\tau)}{\tau}, b_2 - T'_{22}\right\} \tag{30}$$

Note that (30) depends on the fact that C1 has lower priority than C2 in AG2 ADSL link and thus C1 can use only the available bandwidth, if any.

When considering AG3, since the traffic of C3 has the highest priority on the ADSL link:

$$T'_{33} = \min\left\{b_3, \frac{D_3(\tau)}{\tau}\right\} \tag{31}$$

To compute the contribution of the throughput due to C1 and C2 on AG3 ADSL, we must recall the scheduler behavior of BOB. Due to the round-robin selection for neighboring APs (see Fig. 4), AG2 will serve the flows directed to AG3 (i.e. traversing Q_{23}) proportionally to the incoming traffic, i.e. β_1 fraction of C1 traffic and β_2 fraction of C2 traffic to AG3. Let λ'_{13} and λ'_{23} be the offered load of C1 and C2 for AG3 ADSL link. Note that from the point of view of Q_{23}, λ'_{13} and λ'_{23} are the throughput of C1 and C2. Hence we can obtain

$$\lambda'_{13} = \frac{\beta_1 D_{12}(\tau)}{\beta_1 D_{12}(\tau) + \beta_2 D_2(\tau)} \frac{D_{23}(\tau)}{\tau} \tag{32}$$

$$\lambda'_{23} = \frac{\beta_2 D_2(\tau)}{\beta_1 D_{12}(\tau) + \beta_2 D_2(\tau)} \frac{D_{23}(\tau)}{\tau} \tag{33}$$

According to BOB scheduler, C1 and C2 have the same priority to access the available bandwidth of AG3 ADSL link, i.e., $b_3 - T'_{33}$. Thus let $b'_3 = (b_3 - T'_{33})/2$, we have four cases for the combination of T'_{13} and T'_{23}:

$$(T'_{13}, T'_{23}) = \begin{cases} (\lambda'_{13}, \lambda'_{23}) & \text{if } \lambda'_{13} \le b'_3, \lambda'_{23} \le b'_3 \\ (\lambda'_{13}, b'_3 - \lambda'_{13}) & \text{if } \lambda'_{13} \le b'_3, \lambda'_{23} > b'_3 \\ (b'_3 - \lambda'_{23}, \lambda'_{23}) & \text{if } \lambda'_{13} > b'_3, \lambda'_{23} \le b'_3 \\ (b'_3, b'_3) & \text{if } \lambda'_{13} > b'_3, \lambda'_{23} > b'_3 \end{cases}$$

	Wi-Fi			ADSL		User
	R_1	R_{12}	R_2	b_1	b_2	λ_2
Rate [Mbit/s]	54	6	54	2	2	0.5

Table I. Parameter settings for the single-hop scenario.

Finally, we obtain the throughput of each user by summing the throughput contributed by each ADSL link:

$$T_1^u = T'_{11} + T'_{12} + T'_{13}$$
$$T_2^u = T'_{22} + T'_{23}$$
$$T_3^u = T'_{33}$$

6. EVALUATION AND VALIDATION OF THE FLUID MODEL

In this section we will discuss the numerical results obtained by solving the fluid model and then we will validate our theoretical findings in an experimental testbed.

6.1. Numerical results

We developed a C-language solver for the set of equations describing the fluid model presented in Sec. 5.1 and Sec. 5.2, to study the effect of the different input parameters on the model.

6.1.1. Single-hop scenario

We start by considering a single-hop scenario, as shown in Fig. 7. The parameters for the scenario are shown in Table I. The data rates of users C1 and C2 are assumed to be 54 Mbit/s which is the maximum data rate that can be achieved in 802.11g. Data rate of AG1 is assumed to be 6 Mbit/s, i.e. the minimum allowed in 802.11g, to describe a realistic scenario in which neighboring APs adopt low data rates due to the obstacles and the high distance. In order to investigate scenarios with available bandwidth to share, we set the offered load of C2 to 0.5 Mbit/s.

Fig. 10 shows the overall throughput when varying the offered load of C1 and the forwarding factor α, respectively. When the offered load increases, the overall throughput increases and then saturates at around 4 Mbit/s, which is the maximum achievable in this setting. We now consider the effect of α. We notice that for any load it is possible to find at least one value of α to maximize the performance.

Interestingly, the performance degrades when α is too large, since too much traffic of C1 is redirected to AG2 and the ADSL link of AG1 is not fully utilized, which prevents the overall throughput from being maximum. To better highlight the effect of α, Fig. 11 shows the performance in function of α. When the offered load is small, i.e. $\lambda_1 \le 1$ Mbit/s, α has no effect on the overall throughput, since the bandwidth provided by a single ADSL link is already enough to sustain the offered load. When the offered load is large, we have many choices for

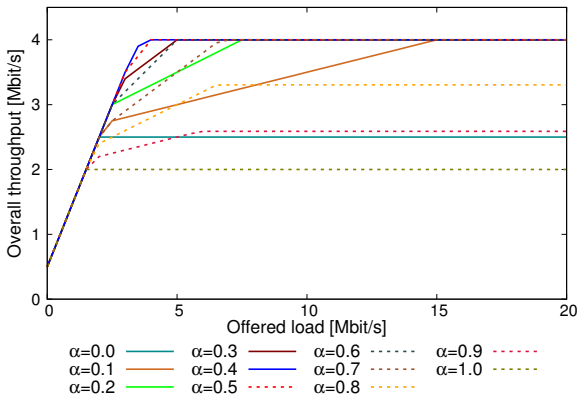

Figure 10. Throughput in the single-hop scenario for different λ_1

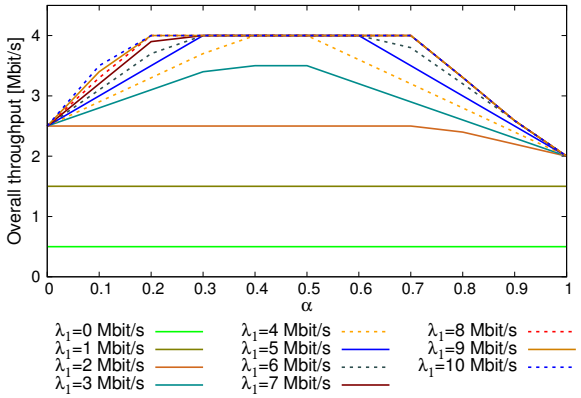

Figure 11. Throughput in the single-hop scenario for different α

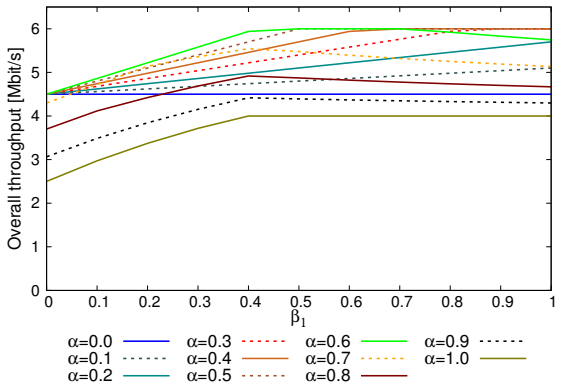

Figure 12. Overall throughput in MH-1 scenario in function of β_1.

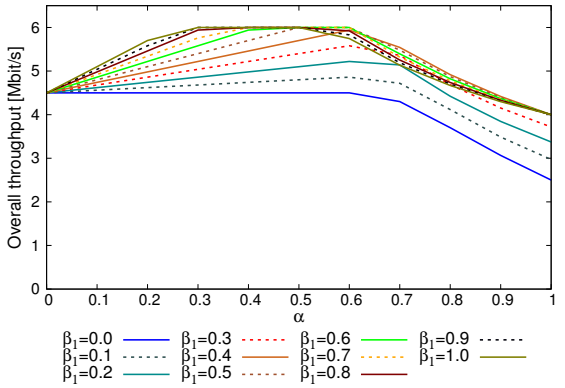

Figure 13. Overall throughput in MH-1 scenario in function of α.

the optimal value of α, all of them guaranteeing that the ADSL links are saturated. Note that the throughput is a concave function of α and this allows local gradient-based searching algorithms to get the optimal solution for the optimization problem.

6.1.2. Multi-hop scenario

After investigating the single-hop scenario, we now consider the multi-hop case shown in Fig. 9, with the parameter setting of Table II. We set $\lambda_3 = 0.5$ Mbit/s in order to leave some spare bandwidth on its ADSL link and thus to enable multi-hop communications.

In the first scenario MH-1, we set $\lambda_1 = 6$ Mbit/s and $\lambda_2 = 2$ Mbit/s. This configuration allows to saturate the local ADSL links of AG1 and AG2. We set $\beta_2 = 0$ since C2 must exploit only its only ADSL link and saturate it. Thus, only the traffic of C1 is sent eventually to AG3. Thus we can easily observe the multi-hop behavior for C1 traffic. We varied α and β_1 in the full interval $[0, 1]$ with steps equal to 0.1 to get all the possible combinations of these two factors. In all the possible settings, the throughput of C2 and C3 are maximum: $T_2^u = \lambda_2$ and $T_3^u = \lambda_3$, thus the ADSL bandwidth is always guaranteed to the local user, as expected. Fig. 12 shows the result of the overall throughput $T_1^u + T_2^u + T_3^u$. As in the single-hop scenario, many combinations of α and β_1 allow to achieve the maximum overall throughput. It is also worth noticing that when β_1 is less than 0.4, C1 traffic is not able to fully exploit the link between AG2 and AG3 and thus the throughput is not maximum. Interestingly, the throughput of C1 is maximized when exploiting also AG3 ADSL (occurring when $\beta_1 > 0$). Fig. 13 shows the overall throughput when varying α. When $\alpha < 0.3$, the overall throughput is not maximum, since the traffic forwarded to AG2 is too small, which further makes it impossible to fully exploit the ADSL link of AG3. Similarly, when $\alpha > 0.6$, AG1 forwards too much traffic to the other AGs and thus its own ADSL is not fully utilized. Observing both Figs. 12 and 13, the overall throughput appears as a concave function of β_1 and α, thus enabling, also in this case, local gradient-based algorithms to find the optimal parameters.

We now consider MH-2 scenario, in which, differently from MH-1, we set $\lambda_2 = 4$ Mbit/s to exceed the bandwidth of the local ADSL link. We evaluated the results for all

Parameter		Scenario			
		MH-1	MH-2	MH-3	MH-4
Wi-Fi data rates [Mbit/s]	R_1, R_2, R_3	54	54	54	54
	R_{12}, R_{23}	6	6	6	6
ADSL bandwidth [Mbit/s]	b_1, b_2, b_3	2	2	2	2
Forwarding factors	α	[0,1]	0.4	0.4	0.2
	β_1	[0,1]	[0,1]	[0,1]	1.0
	β_2	0	0.5	0.2	[0,1]
User offered load [Mbit/s]	λ_1	6	6	6	2.5
	λ_2	2	4	2.5	6
	λ_3	0.5	0.5	0.5	0.5

Table II. Parameter settting for the multi-hop scenarios.

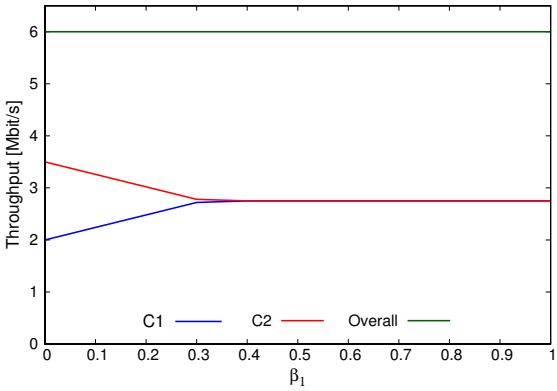

Figure 14. Throughput in MH-2 scenario.

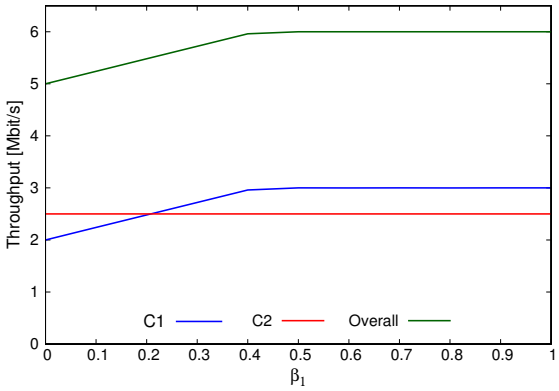

Figure 15. Throughput in MH-3 scenario.

the combination of the parameters, but for the sake of space we report only the results for $\beta_2 = 0.5$ and $\alpha = 0.4$. Fig. 14 shows the throughput of C1 (i.e. T_1^u) and of C2 (i.e. T_2^u) and the overall throughput when varying β_1. When β_1 increases, while the overall throughput is the maximum, the throughput of C1 increases accordingly. This is reasonable as larger β_1 increases the utilization of the bandwidth of C3 by C1. This result proves again that it is worth exploiting multi-hop cooperation in terms of the performance of C1, while not affecting the overall throughput. As the total available bandwidth of AG3 is fixed, the throughput of C2 decreases accordingly. For $\beta_1 = 0.3$, C1 and C2 throughputs reach the same value and then remain the same for $\beta_1 > 0.3$. This is due to the scheduling mechanism which provides the same priority to the two traffic flows when accessing AG3 ADSL link, and this confirms fairness of the adopted scheme.

As a comparison, we consider the scenario MH-3, where we set $\lambda_2 = 2.5$ Mbit/s and $\beta_2 = 0.2$. The results are shown in Fig. 15. When β_1 increases, i.e., more traffic of C1 is forwarded to AG3, the throughput of C1 grows. However it does not affect the performance of C2 even when β_1 is extremely large. This implies that we can preserve the performance of the flow with the smallest offered load, thus guaranteeing max-min fairness among the contending flows.

In the last scenario MH-4, we set $\lambda_1 = 2.5$ Mbit/s and $\lambda_2 = 6$ Mbit/s. We run the simulations with all the combinations of the three forwarding factors and for the sake of space we report here only the results for $\alpha = 0.2$ and $\beta_1 = 1.0$. In Fig. 16 we observe that when $\beta_2 \in [0.2, 0.6]$, the overall throughput is maximum corresponding to the cases in which C2 fully exploits the available bandwidth of C3. As β_2 increases, C2 takes higher bandwidth on AG3 ADSL link, which is also used by C1, and thus the throughput of C1 does not degrade. This is achieved by the scheduling mechanism on the AG3 ADSL link where we assign the same priority to the traffic coming from C1 and C2. Note that for $\beta > 0.8$, C2 is not able to fully exploit its own ADSL link.

As a summary, our results show that it is beneficial to exploit the multi-hop bandwidth sharing approach, by setting the forwarding factors properly. Moreover, our scheme guarantees higher priority to the local user at an AG and achieves max-min fairness among all the traffic flows coming from the neighboring AGs. Furthermore, the throughput is a concave function of α, β_1 and β_2, and this enables the use of simple gradient-based approaches to find the optimal parameter settings.

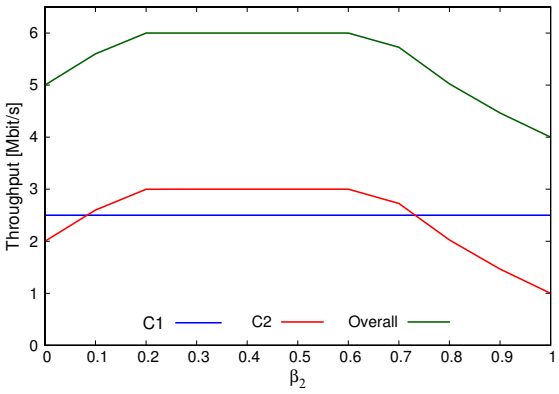

Figure 16. Throughput in MH-4 scenario.

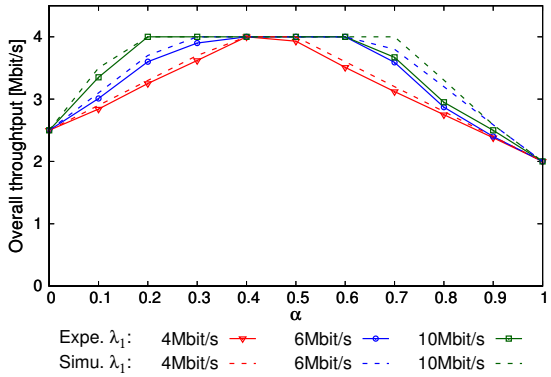

Figure 17. Validation of fluid model in the single-hop scenario. Solid lines refer to experimental results and dotted lines refer to the fluid model results.

6.2. Experimental validation

We run several tests to validate our fluid formulation model with real single-hop testbed with 2 AGs. Each AG is connected to an independent ADSL modem. We used two Linux laptops to act as AG and two Raspberry Pi with Edimax USB wireless card (with RTL8188 chipset) emulating the users and to setup the topology in Fig. 6. In order to emulate different ADSL scenarios, we rate-limited the traffic from the AG to each modem to 2 Mbit/s, through `qdisc` tool provided by Linux Traffic Control [13]. We used `iwconfig` command to fix the data rates of all the wireless interfaces. We generated UDP traffic at constant bit rates for each user. We run all tests at night in order to minimize the interference from campus Wi-Fi and other networks.

To validate the model, we adopted the same parameters for the single-hop scenario of Table I. Fig. 17 shows the experimental results obtained in the real testbed and the numerical results obtained by solving the fluid model, for different values of λ_1. By comparing the two curves, the throughput obtained by solving the fluid model appears to be very accurate. This behavior is interesting, given the level of approximation of the model, which is oblivious of the packet nature of the traffic and does not take into account some specific temporal overheads of the Wi-Fi protocol and the collision on the wireless channel. The relative error of the model is always below 10%, with the maximum error achieved when α is large. This is expected, since for small α actually just two interfaces contend for the channel (i.e. just the two users) with negligible collision probability. Instead, for larger values of α, we have 3 interfaces contending and then the collision probability is not anymore negligible and thus the error of the model (oblivious of the collisions) is larger.

Note also that the throughput obtained in the real testbed shows the concave behavior expected according to the fluid model, and thus confirms the validity of adopting a gradient-based approach to find the optimal parameters (α, β_1 and β_2) to distribute the traffic among the cooperating AGs.

7. EXPERIMENTAL EVALUATION

We adopted the same testbed as the one adopted in Sec. 6.2, but now with 3 AGs. The only difference is that we used a desktop PC with an Edimax USB wireless card (with RTL8188 chipset) to emulate each user. The evaluation mainly focus on: (i) evaluating the maximum performance gain achieved by BOB with real applications, (ii) comparing the two algorithms for flow-level balancing described in Sec. 3.2, (iii) and assessing the performance of HTB scheduler described in Sec. 3.3.

7.1. Performance for cloud storage applications

We select Google Drive [18], OneDrive [19] and Dropbox [20], which are some of the most popular Cloud Storage services, as test applications. To evaluate the performance, we run several tests varying the number of activated AGs and the application. We set the rate of all the ADSL links to 2 Mbit/s. In each experiment, we upload 10 files through a cloud storage application, with an average file size of 50 MB. We capture the traffic at the ADSL interface of the local AG and of neighboring AGs, considering just the packets directed to the specific IP addresses of the servers adopted by the applications. The list of IP addresses of the servers of a specific applications were obtained with `netstat` tool. Since all the considered applications adopt TCP as transport protocol, we were able to evaluate the upload throughput by considering the acked sequence numbers at TCP level.

Table III shows the *throughput gain*, defined as the ratio between the actual bandwidth obtained when BOB is enabled and the one when BOB is not enabled. The results for WRR and PFB flow-balancing are exactly the same. For GoogleDrive and OneDrive, BOB increases the throughput by a factor equal to the number of cooperative AGs. This large gain is due to the fact that GoogleDrive

Table III. Performance achieved by BOB during file upload of cloud storage services for WRR and PFB flow balancing

Scenario	Throughput gain		
	GoogleDrive	OneDrive	Dropbox
2 AGs	1.98	1.99	1.00
3 AGs	2.96	2.97	1.00

Table IV. Total upload time for 100 files under different load balancing schemes

Scenario	Upload time [s]		
	Minimum	PFB	WRR
Balanced	235.1	264.4	283.2
Unbalanced	235.1	272.3	357.7

and OneDrive open multiple TCP connections [21] while uploading files and thus the flow-level balancer is able to exploit fully the available aggregate bandwidth. On the contrary, Dropbox opens only one TCP connection to upload files [22]. Since BOB balances the traffic on per-flow basis, in this specific case it cannot exploit the available bandwidth at the neighboring AG. This highlights the main weakness of our flow-level balancer, which is effective only for many concurrent data flows.

7.2. Performance of flow-level balancing

We utilize two AGs to compare the performance of WRR and PFB algorithms described in Sec. 3.2. In this test, the user sends 100 files to a server and we repeat the experiment 10 times. We set WRR weights to balance equally the traffic across the two possible paths. The size of each file is uniformly distributed between 250 kB and 2.5 MB, and the user opens one TCP connection per file. The user starts to send each file according to a Poisson process at rate 0.4 file/s. We consider two ADSL scenarios. In the first one, denoted as *balanced*, the two ADSL links have the same bandwidth, equal to 2 Mbit/s, while in the second one, denoted as *unbalanced*, the rate of one link is set to 1.5 Mbit/s and the other one is set to 2.5 Mbit/s.

Table IV shows the total time to upload the 100 files with WRR and PFB for the two scenarios. The minimum value is calculated theoretically by dividing the aggregate file size by the sum of the two ADSL capacities and provides a lower bound to the upload time. In the balanced scenario, PFB and WRR perform almost the same. WRR assigns to each link the same number of files and the performance degradation with respect to PFB is due to the randomness of the file sizes, that do not allow a perfect load balancing. On the contrary, in the unbalanced scenario, PFB greatly outperforms WRR due to the ability to adapt to the available bandwidth, by considering the number of pending flows during the flow allocation. Since the available bandwidth of a real BOB system is expected to frequently change, PFB is expected to be the best choice to adopt in a real system.

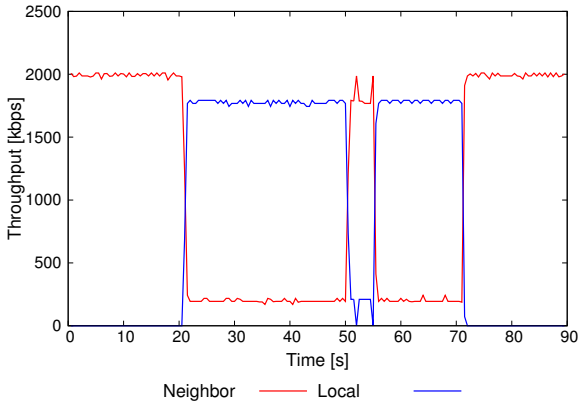

Figure 18. The throughout for the local and neighboring user measured at the Local AG.

7.3. Performance of the traffic scheduler

We test the behavior of HTB scheduler with 2 AGs. We let one user to associate to each AG. We set both ADSL capacities equal to 2 Mbit/s. We set β in HTB equal to 0.1, thus assuring 1.8 Mbit/s to the local user and 0.2 Mbit/s to the neighboring user. We let the neighbor user to continuously upload multiple files using OneDrive. As described in Sec. 7.1, OneDrive opens multiple TCP connections and BOB is able to fully exploit the ADSL link of the local AG.

Fig. 18 shows the throughput of the two users measured at the local AG. Initially, the AG correctly provides all its ADSL capacity to the neighbor user, since the local user is inactive. At time 20s, the local user starts a short upload using Dropbox and he is able to instantaneously get the allocated bandwidth of 1.8 Mbit/s, while the throughput of the neighboring user simultaneously drops to 200 kbit/s. After the local user finishes the upload, the neighbor fully regains the available bandwidth of the local AG. The result shows that the bandwidth is allocated as expected, i.e., according to the HTB settings. Besides this, it shows that HTB reacts very fast when the local user starts to generate traffic. This clearly shows that the traffic scheduler is able to guarantee a transparent behavior for the local user, which is affected by the neighboring users by a small throughput degradation, that can be easily controlled by the β parameter.

8. CONCLUSIONS

In this paper, we proposed BOB, i.e. Beyond One's Bandwidth, a distributed system composed of cooperative Access Gateways (AGs), which can improve the Internet upload speed for ADSL residential users. Differently from previous solutions, our system is totally transparent to the users. We presented the basic architecture and the communication topology of the system. We designed a

load balancer that distributes the traffic at flow level according to two schemes: Weighted Round Robin (WRR) and Pending Flow Balancing (PFB). We also designed a work-conserving traffic scheduler that exploits a Hierarchical Token Bucket (HTB) scheduler to achieve a transparent behavior for the local users of the AG.

We developed a fluid model to investigate the effect of the traffic balancing scheme on the throughput experienced by each user. The model was shown to be accurate in a real scenario, even if based on many simplifying assumptions.

To validate our design, we implemented a prototype of BOB on Linux machines and tested the performance of our approach with real applications. Notably, we showed that with some standard cloud storage services, the available bandwidth of the neighboring AGs is fully exploited with a beneficial effect on the file upload time. We also proved the performance gain achievable by our proposed PFB scheduler when the available bandwidth is not evenly distributed across the cooperative AGs. Finally, we showed the fast reactivity of the proposed scheme in preserving the local bandwidth to the local user of an AG.

9. ACKNOWLEDGMENTS

This work has been partially supported by project EIT-ICT Lab "SDN at the Edges", funded by EU H2020 program in 2015.

REFERENCES

1. L. DiCioccio, R. Teixeira, and C. Rosenberg, "Measuring and characterizing home networks," *SIGMETRICS Performance Evaluation Reviews*, vol. 40, no. 1, pp. 383–384, Jun. 2012.
2. Y. Duan, P. Giaccone, P. Castrogiovanni, D. Mana, C. Borean, and C. Rossi, "Transparent bandwidth aggregation for residential access networks," in *IEEE GLOBECOM*, 2015.
3. S. Kandula, K. C.-J. Lin, T. Badirkhanli, and D. Katabi, "FatVAP: Aggregating AP backhaul capacity to maximize throughput." in *NSDI*, 2008.
4. X. Xing, S. Mishra, and X. Liu, "ARBOR: hang together rather than hang separately in 802.11 Wi-Fi networks," in *IEEE INFOCOM*, 2010.
5. D. Giustiniano, E. Goma, A. Lopez Toledo, I. Dangerfield, J. Morillo, and P. Rodriguez, "Fair WLAN backhaul aggregation," in *ACM MobiCom*, 2010.
6. E. Chai and K. G. Shin, "Sidekick: AP aggregation over partially overlapping channels," in *IEEE ICNP*, 2011.
7. S. Jakubczak, D. G. Andersen, M. Kaminsky, K. Papagiannaki, and S. Seshan, "Link-alike: using wireless to share network resources in a neighborhood," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 12, no. 4, pp. 1–14, 2009.
8. E. Goma and D. Giustiniano, "SmartAP: Practical WLAN backhaul aggregation," in *IFIP Wireless Days*, 2013.
9. C. Rossi, N. Vallina-Rodriguez, V. Erramilli, Y. Grunenberger, L. Gyarmati, N. Laoutaris, R. Stanojevic, K. Papagiannaki, and P. Rodriguez, "3GOL: Power-boosting ADSL using 3G OnLoading," in *ACM CoNEXT*, 2013.
10. H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Multi-path TCP: A joint congestion control and routing scheme to exploit path diversity in the Internet," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1260–1271, 2006.
11. "Hierarchical Token Bucket theory," http://luxik.cdi.cz/~devik/qos/htb/manual/theory.htm, 2016.
12. R. Garroppo, S. Giordano, S. Lucetti, and E. Valori, "The wireless hierarchical token bucket: a channel aware scheduler for 802.11 networks," in *IEEE WoWMoM*, June 2005.
13. "Queueing Disciplines for Bandwidth Management," http://www.lartc.org/howto/lartc.qdisc.html, 2016.
14. M. Lacage, M. H. Manshaei, and T. Turletti, "IEEE 802.11 rate adaptation: a practical approach," in *ACM MSWiM*, 2004.
15. M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *IEEE INFOCOM*, 2003.
16. D. Vassis, G. Kormentzas, A. Rouskas, and I. Maglogiannis, "The IEEE 802.11g standard for high data rate WLANs," *IEEE Network*, vol. 19, no. 3, pp. 21–26, 2005.
17. M. Loiacono, J. Rosca, and W. Trappe, "The snowball effect: Detailing performance anomalies of 802.11 rate adaptation," in *IEEE GLOBECOM*, 2007.
18. "Google drive," https://www.google.com/drive/, 2016.
19. "Onedrive," https://onedrive.live.com/, 2016.
20. "Dropbox," https://www.dropbox.com/, 2016.
21. I. Drago, E. Bocchi, M. Mellia, H. Slatman, and A. Pras, "Benchmarking personal cloud storage," in *ACM IMC*, 2013.
22. I. Drago, M. Mellia, M. Munafo, A. Sperotto, R. Sadre, and A. Pras, "Inside Dropbox: understanding personal cloud storage services," in *ACM IMC*, 2012.