# The Power of Swarming in Personal Clouds Under Bandwidth Budget

Rahma Chaabouni[1], Marc Sánchez-Artigas[1], Pedro García-López[1] and Lluís Pàmies-Juàrez[2]

[1] *Universitat Rovira i Virgili, Tarragona (Spain)*
*{rahma.chaabouni|marc.sanchez|pedro.garcia}@urv.cat*
[2] *HGST Resarch*
*lluis.pamies-juarez@hgst.com*

**Abstract**

Users are unceasingly relying on personal clouds (like Dropbox, box ...) to store, edit and retrieve their files stored in remote servers. These systems generally use HTTP to distribute the files to end-users. This means that they require a huge amount of bandwidth to meet the requirements of their clients. Personal clouds with limited bandwidth budget can benefit from the upload speed of the clients sharing the same file to improve the quality of service. This can be done by introducing a peer-to-peer protocol, BitTorrent for instance, when the load on a certain file becomes high. The main challenge is to decide when to switch to BitTorrent and how to allocate the cloud's available bandwidth to the different clients. In this paper, we propose an algorithm for the allocation of the cloud's bandwidth. Based on the current load and the predefined quality of service constraints, the algorithms identifies the most suitable protocol for each swarm and provides the corresponding bandwidth allocation. We validate the algorithm using a real trace of the Ubuntu One system and the results show important gains in the download times experienced by the clients.

*Keywords:* BitTorrent, peer-assisted content distribution, personal clouds, bandwidth allocation

## 1. Introduction

Nowadays, users are unceasingly relying on cloud storage services (like Dropbox, Google Drive or Box ...) to store, edit and retrieve their data stored in remote servers and which can be accessed all over the Internet. Such systems are hosted by cloud-based datacenters spread all over the world and are generally equipped with a set of features that allow sharing and collaboration between the users. That is why these popular applications account for a major share of Internet traffic today [1].

Small and medium-sized personal clouds with limited budget constraints generally have fixed amount of bandwidth. This bandwidth is shared by all the concurrent active end-users, which might jeopardize the overall quality of service especially when the demand becomes high. As a matter of fact, these systems are based on a client-server architecture and the default content distribution protocol is usually HTTP. This means that all download requests are handled by a central entity which sends the requested content in a single stream. Unfortunately, such transfer is limited by the narrowest network condition along the way, or by the server being overloaded by requests from many clients.

To cope with these limitations, the cloud can benefit from the clients' upload capacities to overcome its bandwidth limits. This can be done by using the BitTorrent (BT) protocol [2] to distribute the files that are shared between a set of devices. In such scenarios, it is possible to benefit from the common interest of users in the same file and use their own upload bandwidth to offload the cloud from doing all the serving. However, the use of BitTorrent may incur a longer download time compared to HTTP especially for small files [3]. The main challenge is to decide for each swarm which protocol is more suitable (HTTP or BT) for transferring the requested files and how much cloud bandwidth should be allocated to each swarm.

In this paper, we study the relationship between the cloud bandwidth allocated to a swarm of clients and the resulting download time for the end-users. We also propose a bandwidth allocation and protocol decision algorithm that evaluates for each swarm the most suitable protocol (HTTP or BT) and returns the amount of bandwidth to be allocated, based on the current load on the cloud. Our key contributions are as follows:

- We analyze the relationship between the amount of cloud bandwidth allocated to a given swarm and the resulting download time. Based on a fixed quality of service constraint, we calculate the amount of seed bandwidth needed to ensure a given ratio between the download times in HTTP and BT.
- We propose a dynamic algorithm (Algorithm 1) which uses simple parameters that can be collected by the system and evaluates the efficacy of using HTTP and BitTorrent as a distribution protocol for each requested file. Based on the load of the seed and the predefined switching constraints, the algorithms decides the most suitable protocol for each case and provides the corresponding bandwidth allocations at the swarm level. This algorithm can be applied in cloud-based content distribution systems to achieve important improvements in the overall quality of service.
- We develop two simulators to evaluate the efficiency of our proposal. The first one simulates the default behavior of the seed where each download operation is treated individually and the content is delivered using HTTP. The second simulator simulates the bandwidth distribution and switching algorithm where BT can be used along with HTTP to distribute content. We validate both approaches using a real trace of the Ubuntu One [1] system: We vary the switching constrains and the cloud upload speed limits and measure the degree of improvement in download time of the involved clients using our algorithm (BT and HTTP together) compared to the use of HTTP alone. The results show important improvements in the download time experienced by the peers.

The remainder of this paper is organized as follows: We discuss related work in Section 2 and present some background on BitTorrent and personal clouds in Section 3. Section 4 presents the architecture of the system and highlights the main differences compared to the classic personal clouds. In Section 5, we state the bandwidth allocation problem and propose an algorithm to solve it in section 6. Section 7 evaluates the efficacy of the algorithm based on a real trace of the Ubuntu One system. Finally, Section 8 concludes the paper and presents our future plans.

## 2. Related work

Integrating BitTorrent with the cloud is not a new idea in the literature. Several previous studies have tried to combine BitTorrent content distribution technologies with cloud environments. Many previous works have focused on reducing download times for large contents using BitTorrent in cloud settings. BitTorrent has proven its efficiency not only for bulk synchronous content distribution [4] but also for reducing transfer times for cloud virtual images [5, 6, 7]. However, to the best of our knowledge, we were the first to propose to use both HTTP and BitTorrent together in cloud systems [8, 3].

In [8], we introduced the idea of transparent switching from HTTP to BitTorrent upon detection of a certain critical mass demand on a specific content. The threshold was placed on the number of users requesting the same files. The system tested with each new request whether the current number of requesters of the corresponding file passed the predefined threshold or not. When the threshold was reached, the system decided to adopt BitTorrent instead of HTTP in order to avoid bottlenecks on the one hand, and to save cloud bandwidth on the other hand.

In [3], we investigated further the threshold at which the system should switch from HTTP to BitTorrent. Instead of placing a static condition on the number of peers requesting the same file, we elaborated a complete analysis and experimental evaluation of a dynamic threshold that takes into consideration not only the number of peers, but also their corresponding bandwidths and the size of the shared file. With recourse to previous studies related to the study of HTTP and BT protocols [9, 10, 11, 12], we proposed accurate estimations of the download times in both protocols and introduced some evaluation metrics to evaluate the efficiency of each of them. These metrics provide accurate estimations of the gain in download time and the amount of data contributed by the peers.

This paper differs from [3] in that it considers personal clouds with a fixed bandwidth budget constraint. Various related works focused on the allocation of the seed's bandwidth within the BT swarms when the bandwidth budget is limited [13, 14, 15]. However, we believe we are the first to propose an algorithm that manages to distribute the limited cloud bandwidth between the different peers using two different download protocols: HTTP and BitTorrent.

[1] https://wiki.ubuntu.com/UbuntuOne

## 3. Background

### *3.1. The BitTorrent protocol (BT)*

BitTorrent [2] is a P2P application whose goal is to facilitate fast downloads of files by taking advantage of the upload bandwidth of the peers. A user who wants to share a certain content via BT has to generate first a *.torrent* file that contains the related meta-data information and a link to a *tracker*. A tracker is a server that assists in the communication between peers interested in the same content. After the generation of the meta-data file, the user has to make the content to be shared available through a BitTorrent node acting as a *seed*. A seed is a node that has a complete copy of a particular content, whereas a *leecher* is one that has only a partial copy. In this paper, seeds and leechers are simply referred to as *peers* and the set of all the peers sharing the same content is called a *swarm*. Peers interested in a certain content are in charge of getting the corresponding *.torrent* file and then contacting the tracker to acquire a set of peers sharing the same content. Once these peers are located, it is possible for them to communicate to one another in order to distribute the file among them. The official protocol specification can be found at: *http://www.bittorrent.org/beps/bep_0003.html*

### *3.2. Personal Cloud systems*

*Definition.* A Personal Cloud (PC) is a term generally used to refer to a file hosting service that allows its users to store, synchronize and share content over the Internet. The authors in [16] propose the following definition: *"The Personal Cloud is a unified digital locker for our personal data offering three key services: Storage, Synchronization and Sharing. On the one hand, it must provide redundant and trustworthy cloud data storage for our information flows irrespective of their type. On the other hand, it must provide syncing and file exploring capabilities in different heterogeneous platforms and devices. And finally, it must offer fine-grained information sharing to third-parties (users and applications)."* Personal Clouds have attracted the researchers' attention lately and there have been important work related to the benchmark and design of these services [17, 18, 19, 20].

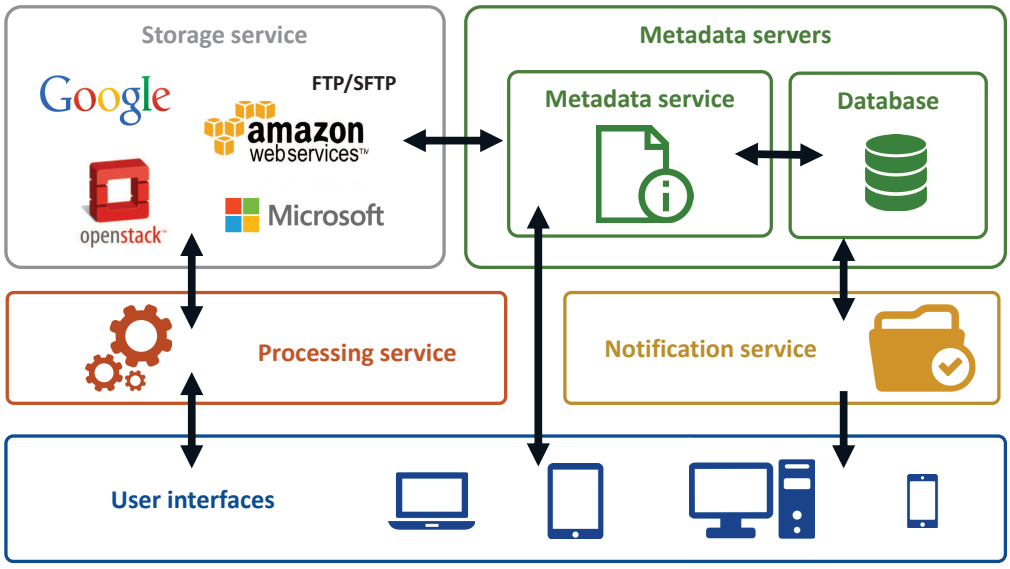



Figure 1: General architecture of personal clouds

*Architecture.* Figure 1 presents the general architecture of a PC. This figure is inspired from the official architecture of Dropbox [21]. It presents the core elements of a PC, without taking into consideration the authentication and encryption layers that are deployed to reinforce security. These elements are:

- **Meta-data service:** The meta-data servers contain all the meta-data information related to the clients and the files. They can be equipped with a local database where all the meta-data is stored.
- **Storage service:** The storage service of storage back-end refers to the physical locations where the users' file content are stored. It can be local, in the form of local storage servers accessed via FTP/SFTP, or external, provided by a third-party (Amazon, Google...).
- **Notification service:** The notification service is dedicated to monitoring whether or not any changes have been made to the users' accounts. Whenever a change to any file takes place, the client is notified in order to synchronize these changes.
- **PC clients or user interfaces:** The services offered by personal clouds can be utilized and accessed by physical clients through a number of interfaces, including web interfaces (accessed through web browsers), desktop applications or mobile apps.
- **Processing service:** The processing service is responsible for processing the files and ensuring their delivery to the end-users. To download a file, the client sends an HTTP GET request to the processing service. The latter verifies the existence of the file in the storage nodes and the file is transferred using the HTTP protocol.

## 4. System architecture

In this paper, we consider a classic personal cloud (store, sync and share functionalities) enriched with extra components that allow inter-client content transfers via BT. Several components are added to accommodate the BT behavior as shown in Figure 2, including:

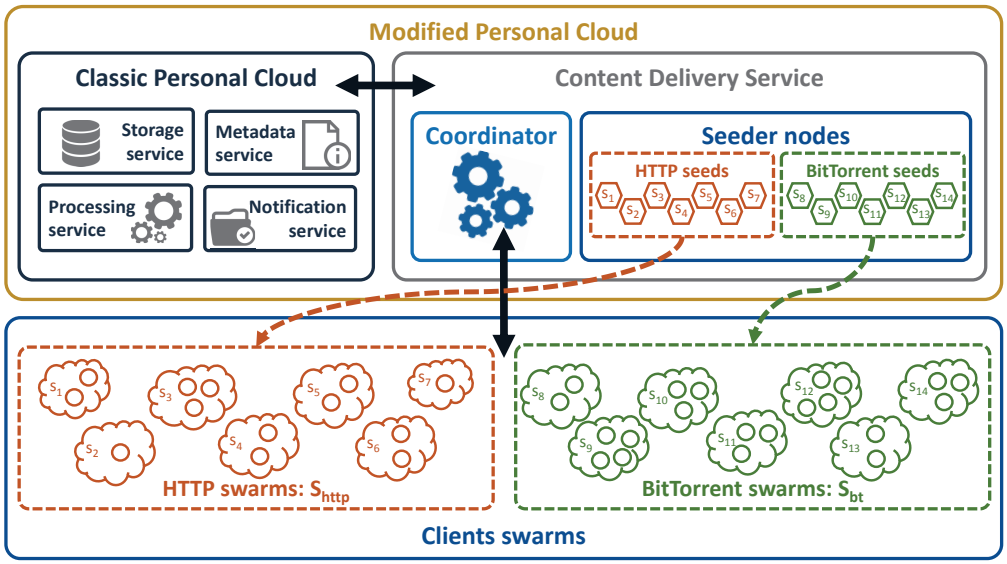


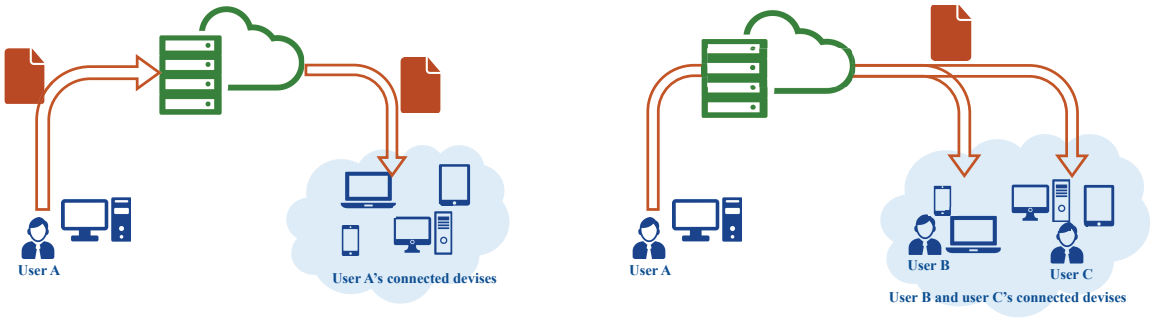

Figure 2: Global view of the system architecture

- **Content Delivery Service :** The content delivery service is also referred to as cloud. Its main role is to process the requests coming from the end-users and ensure the delivery of the files to the corresponding requesters. Several components are added, compared to the default architecture (Figure 1), including:
  - **Coordinator:** The coordinator is the core component of the cloud. It is responsible for managing all the clients' requests and ensuring they are processed correctly. The coordinator is also responsible for the proper management of the cloud's resources.
  - **Seeder nodes:** The seeder nodes are the entities responsible for delivering the requested content from the storage back-end servers to the end-users. To each file being distributed corresponds one seeder node.

In our paper, we refer to these seeder nodes as *cloud seeds* or *seeds*. We distinguish two types of seeds: *HTTP seeds* and *BitTorrent seeds* depending on the algorithm adopted to distribute the requested content to end-users.

- **Clients swarms:** All the end-user peers are organized into swarms. We define a swarm by the set of peers that are requesting the same file. If a file is being downloaded by a single peer, we consider it as a single-peer swarm. This means that, at a given time, there are as many swarms as the number of files being downloaded (to each file corresponds only one swarm and one seeder node). In our model, we distinguish between two types of swarms:
  - **HTTP Swarms:** The HTTP swarms are the swarms whose peers are downloading the corresponding file from HTTP seeds via HTTP. Clearly, these peers are not collaborating with each other, but grouping them in swarms is a simple means of control which will help, later on, in making the switching decision.
  - **BitTorrent Swarms:** Also referred to as *BT swarms*. Similar to HTTP swarms, BT swarms are the swarms whose peers are downloading the corresponding file from BT seeds via the BT protocol. Typically, these swarms are composed of two peers or more. Since the peers are supposed to collaborate between each other with the help of the cloud seed, it makes no sense to have a single-peer BT swarm.

To download a file, the client sends an HTTP GET request to the coordinator. The latter verifies the existence of the file in the storage nodes and decides the download protocol to be used: HTTP or BitTorrent. The decision is made based on the load on the seed and the swarms' characteristics. In the case of a HTTP download, a HTTP seeder node is associated with the requested file which will be transferred using the HTTP protocol. Otherwise, in the case of a BT transfer, the coordinator creates a torrent meta-data file and runs a corresponding BT seed. After that, the recently created *.torrent* file will be transmitted to the corresponding clients who, unaware of all these interactions, will then start downloading the file using the BitTorrent protocol (from the cloud seed and/or from the other clients). Evidently, the "old" clients who arrived before the switch to BitTorrent will also benefit from the switch if they did not finish the download. In fact, when an "old" client requests a new part of the file to be downloaded, he will realize that the transfer protocol has changed and will automatically adapt to the new one. Thus, each "old" client will join the swarm with the pieces he already has, which means that he will be probably contributing to the swarm as soon as he switches to BitTorrent in a very transparent way.




(a) Synchronization (b) Sharing

Figure 3: Synchronization and sharing in personal cloud systems

This approach can applied widely in any cloud-based system. Personal clouds are the most appropriate for this proposal since the developers can tune the client's implementation to extend them with the BT functionality. The two following common file distribution scenarios could benefit from our hybrid download strategy:

1. *Synchronization:* User A is adding a new file $f$ to his personal account. During the synchronization process, the same file will be download by all the other synchronized devices of the user (figure 3a)
2. *Sharing:* User A is sharing a file $f$ with other users. In this case, the file will be downloaded by all the synchronized devices of the users (Figure 3b)

Both cases can be modeled by the problem of distributing a file $f$ from the cloud servers to a swarm $s$ of different devices (peers) as detailed in Figure 4. We will focus in the following section on this scenario and state the problem of bandwidth distribution.

For the rest of the paper, we consider the notation presented in Table 1.

Table 1: Table of notations

| | |
|---|---|
| $W$ | the cloud's upload budget limit |
| $S$ | the set of all active swarms |
| $S_{http}$ | a subset of $S$ that corresponds to the set of the swarms downloading the files via HTTP |
| $S_{bt}$ | a subset of $S$ that corresponds to the set of the swarms switched to BitTorrent |
| $s$ | a swarm $s = (P_s, f_s, w_s, isBT_s)$ is identified by the set of the peers forming it $P_s$, the file being shared $f_s$, the corresponding amount of allocated cloud bandwidth $w_s$ and a boolean variable $isBT_s$ that indicates the download protocol. |
| $P_s$ | set of all the peers in $s$. $P_s = \left\{(u_p, d_p),\ \ \forall p \in P_s\right\}$ where $u_p$ and $d_p$ are respectively the upload and download speeds of a given peer $p \in s$ |
| $f_s$ | file requested by the peers in $P_s$ |
| $w_s$ | amount of cloud bandwidth allocated to the swarm $s$ |
| $isBT_s$ | boolean variable that indicates the download protocol adopted by the peers in $s$. $isBT_s = True$, if peers in $P_s$ are downloading $f_s$ via BitTorrent and $isBT_s = False$, otherwise |
| $F_s$ | size of the requested file $f_s$ |
| $L_s$ | number of peers in $P_s s$ ($L_s = \|P_s\|$) |
| $D_s$ | aggregated download speed of all the peers in $P_s$ ($D_s = \sum_{p \in P_s} d_p$) |
| $d_{min,s}$ | the download speed of the slowest peer in $P_s$ ($d_{min,s} = \min_{\forall p \in P_s} d_p$) |
| $u_s$ | the average upload speed of all the peers in $P_s$ ($u_s = \sum_{p \in P_s} \frac{u_p}{L_s}$) |
| $\eta_s$ | the effectiveness of file sharing, introduced in [11] and reused in [3]. $\eta_s$ takes real values in $[0, 1]$ where 1 means maximum effectiveness while a value of 0 signals the absence of collaboration between peers |
| $\alpha_{bt}$ | the overhead related to the start-up phase in BitTorrent transfers |
| $\tau$ | the QoS constraint that defines the switching point from HTTP to BT |

## 5. Bandwidth Allocation Problem

Our main goal is to minimize the upload bandwidth $w_s$ allocated to a swarm $s \in S$ while taking into consideration the quality of service offered to the different clients. These allocations should be non-negative ($w_s \geq 0, \forall s \in S$) and their sum should not exceed the cloud's upload budget limit ($\sum_{s \in S} w_s \leq W$). The bandwidth allocation strategy follows these two rules:

**Rule 1.** *HTTP is the default download protocol and each swarm is allocated a share of the cloud's bandwidth equal to its demand.*

**Rule 2.** *The cloud can decide to switch the download protocol for a given swarm s from HTTP to BT if it (the cloud) will save in bandwidth and if this change of protocol will not jeopardize the quality of service constraint related to the download time of the peers in s.*

Rule 1 defines HTTP as the default download protocol and sets the share of the cloud's bandwidth allocated to $s$ to be equal to the aggregated download speeds of the peers in $s$ ($w_s = D_s$). When it is possible to gain in bandwidth, the

cloud can decide to switch the download protocol from HTTP to BT, as stated in Rule 2. This change of protocols is fixed by a quality of service constraint $\tau$. This constraint defines the degree of improvement (or degrade) in download time that is accepted when considering the switch to BT. For instance, a $\tau = 0.2$ requires an improvement of a least 20% in download time that a swarm would gain if it adopts BT rather than HTTP. A negative value of $\tau$, such as $-0.5$ means that losses in download time up to 50% are accepted. The bandwidth allocated to BT swarms should be the minimal that satisfies the switching constraint. As the swarms evolve over time with new peers joining and leaving, we will need to adapt the assignments and allocations accordingly since the cloud's upload speed is limited by $W$.

To better understand the problem, it is important to introduce some parameters that will interfere in making the decision about the bandwidth allocation. These parameters serve to evaluate the performance of using HTTP or BT to distribute a given file shared between a set of peers.

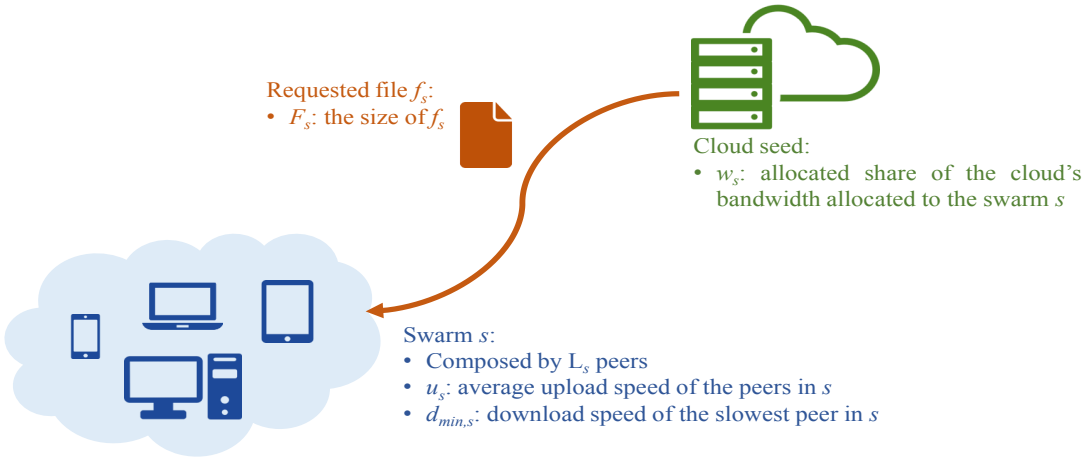



Figure 4: File distribution scenario

We consider the case of a swarm $s$ composed of $L_s$ distinct peers requesting the same file $f_s$ from the same source, called the cloud seed (or simply the seed). We denote by $w_s$ the allocated share of the cloud's upload bandwidth reserved to $s$, $F_s$ the size of the shared file $f_s$, $u_s$ the average upload speed of the peers in the $s$ and by $d_{min,s}$ the download speed of the slowest peer among them (see Figure 4 for more details).

In the following subsections, we will present some formulas related to the estimation of the download times in HTTP and BT (respectively $T_{http}$ and $T_{bt}$. We will also define the gain percentage *Gain* and estimate the minimum cloud upload bandwidth needed to satisfy the QoS constraint for $s$ by solving the equation $Gain(w_s^{bt}, s) \geq \tau$.

### *5.1. Download Time in HTTP $T_{http}$*

In the case $f_s$ is distributed via HTTP, the distribution time $T_{http}$ is limited by the download speed of the slowest peer $d_{min,s}$ or the seed bandwidth $w_s$ divided equally between the $L_s$ clients. It can be defined as follows:

$$T_{http}(w_s, s) = \frac{F_s}{\min\left\{d_{min,s}, \frac{w_s}{L_s}\right\}}. \tag{1}$$

### *5.2. Download Time in BitTorrent $T_{bt}$*

When $f_s$ is distributed via BT, then the distribution time $T_{bt}$ depends on the download speed of the slowest peer $d_{min,s}$, the aggregated upload bandwidth of all the nodes divided equally between all the $L_s$ leechers, and the cloud's allocated share $w_s$. $T_{bt}$ has been studied before in [10] and an approximation of the distribution was given. In [3], we extend that approximation and propose an accurate estimation of $T_{bt}$ that takes into consideration the overheads related to the nature of the protocol, as follows:

$$T_{bt}(w_s, s) = \frac{F_s}{\min\left\{d_{min,s}, \frac{w_s+\eta_s.u_s.L_s}{L_s}, w_s\right\}} + \alpha_{bt}, \tag{2}$$

where $\alpha_{bt}$ is the overhead related to the start-up phase, and $\eta_s$ measures the effectiveness of file sharing for $s$. For more details about $\alpha_{bt}$ and $\eta_s$, please refer to [3]. Equation 2 was validated in [3] using different bandwidth settings and different file sizes, and was proven to be accurate even with small files.

### *5.3. The Gain Ratio*

Sometimes the use of BitTorrent may incur a longer download time compared to HTTP especially for small files. The main challenge is to decide when it is worth switching to BitTorrent. The key element in making the decision is the gain in download time which represents the difference in download time between HTTP and BitTorrent. To this extend, we introduced in [3] the gain ratio which measures the improvement in terms of download time between client-server and peer-assisted systems, as follows:

$$Gain\,(w_s, s) = \frac{T_{http}(w_s, s) - T_{bt}(w_s, s)}{T_{http}(w_s, s)}.$$

The gain can take different values which can be either negative, positive or equal to zero. A positive (respectively negative) gain ratio equal to $x$ (respectively $-x$) means that downloading the file via BT entails a gain (respectively a loss) of *100x%* in download time compared to HTTP. A gain ratio equal to zero indicates that both protocols (BT and HTTP) have the same estimated download time.

We derived in [3] the analytic equation of the gain based on the values of the divisors of $T_{http}$ and $T_{bt}$, which are respectively $\min\left\{d_{min,s}, \frac{w_s}{L_s}\right\}$ and $\min\left\{d_{min,s}, \frac{w_s+\eta_s L_s u_s}{L_s}, w_s\right\}$, as follows:

$$Gain\,(w_s, s) = \begin{cases} -\dfrac{\alpha_{bt}\, d_{min,s}}{F_s}, & \text{if } d_{min,s} \le \dfrac{w_s}{L_s} \text{ and } d_{min,s} \le \min\left\{\dfrac{w_s + \eta_s\, u_s\, L_s}{L_s}, w_s\right\} \\ 1 - \dfrac{w_s}{L_s\, d_{min,s}} - \dfrac{\alpha_{bt}\, w_s}{F_s\, L_s}, & \text{if } \dfrac{w_s}{L_s} \le d_{min,s} \text{ and } d_{min,s} \le \min\left\{\dfrac{w_s + \eta_s\, u_s\, L_s}{L_s}, w_s\right\} \\ 1 - \dfrac{w_s}{w_s + \eta_s\, u_s\, L_s} - \dfrac{\alpha_{bt}\, w_s}{F_s\, L_s}, & \text{if } \dfrac{w_s + \eta_s\, u_s\, L_s}{L_s} \le \min\{d_{min,s}, w_s\} \\ 1 - \dfrac{1}{L_s} - \dfrac{\alpha_{bt}\, w_s}{F_s\, L_s}, & \text{if } w_s \le \min\left\{d_{min,s}, \dfrac{w_s + \eta_s\, u_s\, L_s}{L_s}\right\}. \end{cases} \tag{3}$$

### *5.4. Solving the equation $Gain(w_s^{bt\,??}, s) \ge \tau$*

In order to calculate the minimum amount of cloud bandwidth needed to ensure that the switching condition $Gain(w_s^{bt}, s) \ge \tau$ is satisfied, it essential to reverse the gain formulation (equation 3). To this extend, we study the behavior of the gain formulas when $w_s^{bt}$ varies. Based on this constraint, we identify two exhaustive cases in which the gain equations are monotonically decreasing. For each case, we deduce the reversed equations of the gain, interval per interval, as follows[2]:

- **Case A:** $(L_s - 1)\, d_{min,s} \ge L_s\, \eta_s\, u_s$: the average download speed of the peers in the swarm $s$ is higher than the upload bandwidth the whole swarm can provide:

$$w_s^{bt} = \begin{cases} L_s\, d_{min,s}, & \forall\tau \in \left]-\infty, -\dfrac{\alpha_{bt} d_{min,s}}{F_s}\right] \\ \dfrac{(1-\tau) F_s\, L_s\, d_{min,s}}{F_s + d_{min,s}\, \alpha_{bt}}, & \forall\tau \in \left[-\dfrac{\alpha_{bt}\, d_{min,s}}{F_s}, \dfrac{\eta_s\, u_s}{d_{min,s}} - \dfrac{\alpha_{bt}\,(d_{min,s} - \eta_s\, u_s)}{F_s}\right] \\ \dfrac{\sqrt{a^2b^2 - 2abc + 4ab + c^2} - ab - c}{2\,b}, & \forall\tau \in \left[\dfrac{\eta_s\, u_s}{d_{min,s}} - \dfrac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s}, 1 - \dfrac{1}{L_s} - \dfrac{\alpha_{bt}\, \eta_s\, u_s}{(L_s - 1)\, F_s}\right] \\ \dfrac{F_s\,[L_s\,(1-\tau) - 1]}{\alpha_{bt}}, & \forall\tau \in \left[1 - \dfrac{1}{L_s} - \dfrac{\alpha_{bt}\, \eta_s\, u_s}{(L_s - 1)\, F_s}, 1 - \dfrac{1}{L_s}\right[ \\ \nexists, & \forall\tau \in \left[1 - \dfrac{1}{L_s}, +\infty\right[ \end{cases}$$

[2] For the complete proof of the solution, please refer to Appendix A

Where:

$$a = \eta_s L_s u_s, \quad b = \frac{\alpha_{bt}}{F_s L_s} \text{ and } c = \tau \tag{4}$$

- **Case B:** $(L_s - 1)\, d_{min,s} \leq L_s \eta_s u_s$: the average download speed of the peers in the swarm $s$ is lower than the upload bandwidth the whole swarm can provide:

$$w_s^{bt} = \begin{cases} L_s\, d_{min,s}, & \forall \tau \in \left]-\infty, -\dfrac{\alpha_{bt}\, d_{min,s}}{F_s}\right] \\ \dfrac{(1-\tau) F_s\, L_s . d_{min,s}}{F_s + d_{min,s}\, \alpha_{bt}}, & \forall \tau \in \left[-\dfrac{\alpha_{bt}\, d_{min,s}}{F_s}, 1 - \dfrac{1}{L_s} - \dfrac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}\right] \\ \dfrac{F_s\ [L_s(1-\tau) - 1]}{\alpha_{bt}}, & \forall \tau \in \left[1 - \dfrac{1}{L_s} - \dfrac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}, 1 - \dfrac{1}{L_s}\right[ \\ \nexists, & \forall \tau \in \left[1 - \dfrac{1}{L_s}, +\infty\right[ \end{cases}$$

## 6. Bandwidth Allocation and Protocol Management Algorithm

In this section we present our bandwidth distribution and protocol management algorithm. This algorithm aims to minimize the cloud's allocated bandwidth among the seeder nodes, while respecting the QoS constraint. We remind that a seeder node is an entity responsible for distributing a given file to the corresponding set of clients. The share of the cloud's upload bandwidth allocated to each seeder node should verify the constraints of the problem previously explained in section 5.

In addition to the bandwidth allocations, the algorithm is also responsible for evaluating for each swarm the most suitable content distribution protocol: HTTP or BitTorrent. A swarm would switch to BitTorrent if it satisfies the following conditions:

1. The number of clients in the swarms is higher or equal to 2. In fact, it makes no sense to use BitTorrent with only one client interested in the file.
2. The switch to BitTorrent should satisfy the quality of service constraint $\tau$. This means that BitTorrent can be used only when the gain percentage (equation 3) is higher or equal than $\tau$.
3. The amount of cloud bandwidth allocated in BitTorrent should be smaller than the one using HTTP. This means that the switch will only take place if the cloud would gain in terms of bandwidth.

We proposed in [3] a simple decision approach that consists in calculating for each swarm $s \in S$, with more than one peer, how much the swarm would gain in terms of download time if BT is used instead of HTTP. If that gain satisfies the quality of service constraint $\tau$, then BitTorrent is chosen. Otherwise, HTTP is kept as the download protocol. Even though this approach is simple and direct, it has to be further improved in order to satisfy the third switching condition.

In this paper, we go a step further and calculate the minimum amount of bandwidth $w_s^{bt}$ needed to satisfy the constraint $Gain(w_s^{bt}, s) \geq \tau$ (Section 5.4). Thus, instead of comparing $Gain(w_s^{bt}, s)$ and $\tau$, we compare $D_s$ and $w_s^{bt}$ and the protocol that requires less bandwidth is chosen.

*Algorithm Description*

The main goal of our bandwidth distribution and switching algorithm is to optimally manage the cloud's limited bandwidth among the seeder nodes. It is also responsible for evaluating for each requested file the most suitable content distribution model: client-server or peer-assisted, based on the current demand load. Each active seeder node in the system is associated with a swarm of clients that are interested in the same file. It is important to remind here that the default bandwidth distribution protocol is HTTP, but BitTorrent can be also used when the switching conditions previously stated are satisfied. The swarms whose peers are using HTTP as a transfer protocol are referred to as *HTTP swarms* ($S_{http}$ is the set of HTTP swarms) and the ones with peers downloading via BT are labeled as *BT swarms* ($S_{bt}$ is the set of BT swarms).

The algorithm is executed whenever a change affects a swarm $s^* \in S$. This change can be related to a modification in one or more of the parameters of a certain swarm. It can be due to one or more of the following cases:

- A new peer $p^*$ wants to download a file $f_{s^*}$. If the file is already requested by other peers, then $p^*$ will be added to the existing swarm $s^*$. Otherwise, a new swarm $s^*$ will be created containing a single peer $p^*$.
- A peer $p^*$ leaves a swarm $s^*$. If $p^*$ was not the only peer in the swarm, then the modified swarm will contain a list of the other remaining peers. If $p^*$ was the last peer in $s^*$, then $s^*$ will be removed from $S$.
- The upload or download speed of one or more of the peers in $s^*$ changes.

**Algorithm 1** Bandwidth distribution and switching algorithm

**Input** $S$ ▷ the set of all the current swarms
**Input** $s^*$ ▷ swarm affected by a change
**Input** $W$ ▷ the cloud's upload bandwidth budget limit
**Input** $\tau$ ▷ the switching constraint

1: **if** $L_{s^*} = 1$ **then** ▷ $s^*$ is a single-peer swarm
2: $w_{s^*} = D_{s^*}$
3: **else if** $L_{s^*} > 1$ **then** ▷ $s^*$ has more than one peer
4: **if** $s^* \in S_{http}$ **then** ▷ $s^*$ is a HTTP swarm
5: calculate $w^{bt}_{s^*}$ using equation (4)
6: **if** $w^{bt}_{s^*} \leq D_{s^*}$ **then** ▷ switching to BT
7: switch the transfer protocol from HTTP to BT
8: $isBT_{s^*} = True$ ▷ mark $s^*$ as a BT swarm
9: $w_{s^*} = w^{bt}_{s^*}$
10: **else** ▷ not switching to BT
11: $w_{s^*} = D_{s^*}$
12: **end if**
13: **else** ▷ $s^* \in S_{BT}$, $s^*$ is a BT swarm
14: $w_{s^*} = w^{bt}_{s^*}$ calculated using equation (4)
15: **end if**
16: **else** ▷ $L_{s^*} = 0$, $s^*$ no longer exists
17: remove $s^*$ from $S$
18: **if** $\sum_{s \in S} w_s + w_{s^*} = W$ **then** ▷ the cloud was overloaded
19: **for each** $s$ **in** $S_{bt}$ **do**
20: $w_s = w_s + \frac{w_s}{\sum_{s \in S_{bt}} w_s} w_{s^*}$ ▷ redistribute $w_{s^*}$ to the BT swarms
21: **end for**
22: **end if**
23: **end if**

24: **if** $\sum_{s \in S} w_s > W$ **then**
25: **for each** $s$ **in** $S$ **do**
26: $w_s = \frac{w_s}{\sum_{s \in S} w_s} W$ ▷ scale down all the bandwidth shares
27: **end for**
28: **end if**

The algorithm requires the following input parameters: the set of all current swarms $S$, the swarm affected by the change $s^*$, the cloud's upload bandwidth budget limit $W$ and the switching constraint $\tau$.

Using these input parameters, the algorithm identifies for each swarm the most suitable download protocol (HTTP or BT) and calculates the amount of bandwidth to be allocated to the corresponding seed, as follows:

- If $s^*$ is a single-peer swarm ($L_{s^*} = 1$), then the cloud allocates to $s^*$ a share of bandwidth equal to its download capacity: $w_{s^*} = D_{s^*}$ (*lines 1 and 2*). In this case, the file will be distributed directly from the cloud seed to the single-peer using HTTP.
- If the number of peers in $s^*$ is strictly higher than 1 (*lines 3 to 12*), then there are two possible cases:

- If the peers in $s^*$ are using HTTP to download $f_{s^*}$ ($isBT_{s^*} = False$), the algorithm verifies if it is worth it to switch to BT. To do so, $w^{bt}_{s^*}$ is calculated according to equation (4). We remind that $w^{bt}_{s^*}$ measures the amount of seed bandwidth required to verify the quality of service constraint $\tau$ when using BT for $s^*$. The algorithm compares later this bandwidth ($w_{s^*}$) with the bandwidth allocated by default to the swarm (which is equal to $D_{s^*}$).
  - If the bandwidth required using BT is smaller than the one allocated by default ($w^{bt}_{s^*} \leq D_{s^*}$), then the download protocol more suitable for $s^*$ is BT (*lines 4 to 9*). In this case, a .torrent file associated to $f_{s^*}$ is created and a BT seed is launched in the cloud. All the peers in $s^*$ have to download the .torrent file recently created and then can start downloading $f_{s^*}$ via BT.
  - If the use of BT requires more bandwidth than HTTP, then it is not worth switching to BT. In this case, the cloud allocates a share of bandwidth equal to $D_{s^*}$ (*line 11*).
- If $s^*$ has already switched to BT, then the algorithm recalculates $w^{bt}_{s^*}$: the bandwidth needed to maintain the quality of service constraint $\tau$, which represents also the amount of bandwidth allocated to $s^*$.

- If $s^*$ is an empty swarm ($L_{s^*} = 0$), then the swarm is removed from the swarms' list. If the cloud was overloaded before the removal of $s^*$, then the amount of bandwidth that was previously allocated to $s^*$ is redistributed among the BitTorrent swarms (*lines 18 to 22*). This will prevent the cloud's bandwidth from being underutilized and will boost the distribution of the files among the BT swarms.

When the number of simultaneous requests becomes high, the seed might be unable to serve all the swarms at their full speed. In such a case, the cloud has to scale down all the bandwidth allocations proportionally to the demand (*lines 24 to 28*).

*Complexity of the algorithm*

Our bandwidth distribution and switching algorithm has a complexity of $O(n)$ which is linear with the number of current swarms. The coordinator only keeps in memory the state of the swarms during the iterations. This corresponds to $n \times k$ units of storage where $n$ is the maximum number of simultaneous swarms and $k$ is the size, in unit of storage, required to store the essential information about a current swarm. $k$ is rather small (compared to the size of the files) and it depends on the number of peers in the swarm and the storage space required to store the information of each one of these peers.

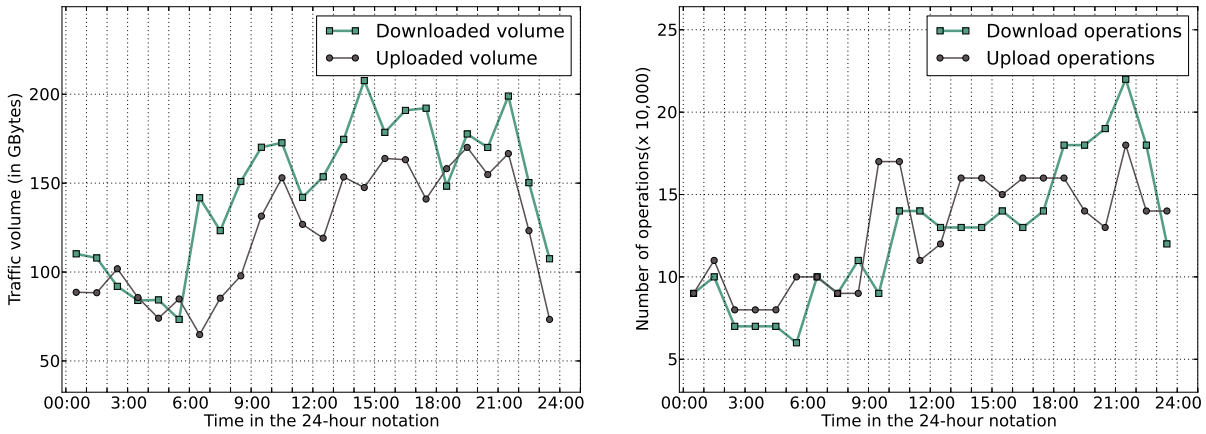
## 7. Validation of the algorithm: Application to the Personal Cloud scenario

In order to evaluate the performance of the proposed algorithm, we implement two simulators. The first simulates the default behavior of the cloud where all the download requests are treated individually and the files are distributed via HTTP. The second simulates the bandwidth distribution and protocol management algorithm. We compare later the results of both approaches using a trace of a real personal cloud system.

### *7.1. The Ubuntu One trace*

In our validation, we use a real trace of the Ubuntu One (UB1) system. The trace was provided by *Canonical Ltd.*[3] in the context of the CloudSpaces project[4]. The logs were collected for about a month from their servers located in London, based on the behavior of real users. Each line of the trace represents an upload or download operation performed by one user on a given file. For the sake of privacy, files and user real identifiers are presented in the form of unique hash codes. For each operation, several information were collected, including: the timestamp, the type of operation ("up" or "down"), the hash and size of the file in question, the user's hash identifier and the corresponding upload and download bandwidths. We filtered the original trace and focused on the upload and download operations that were performed during a random day of the trace (January $21^{st}$, 2014). For that day, $6,331,131$ operations performed by $33,257$ distinct client on $4,095,057$ unique files were logged.

Figure 5 shows the total uploaded and downloaded volume along with the total number of upload and download operations measured per hour during the whole day. The hours are logged according to the Greenwich Mean Time




(a) Volume

(b) Operations

Figure 5: Total uploaded and downloaded volume and the number of upload and download operations per hour during the day January 21$^{st}$, 2014 for the UB1 system.

(GMT). We notice that the peak with the highest number of upload and download operations corresponds to the hour between 21:00 and 22:00 GMT.

To facilitate the validation process, we focus only on that peak hour, but we believe the results would be similar for the whole trace. Since we aim to manage efficiently the upload speed of the cloud, we filter the one-hour sample to keep only 225, 514 download operations which correspond to 173, 756 distinct files. The average file size in the sample is about 1 MB, 988.26 KB to be precise. We notice in that sample that about 68.62% of the operations correspond to single downloads. Single downloads are operations related to files downloaded only once. These files account for about 89% of the total files downloaded between 21:00 and 22:00. This means that only 31.38% of the operations correspond to multiple downloads of the same file and that only 11% of the files were downloaded more than once. This signifies that only 31.38% of the operations are potential candidates for switching to BitTorrent.

Focusing more on that peak hour, we calculate how much bandwidth should be provided by the cloud seed in order to satisfy the need of all the requesting peers if the download protocol was HTTP. To do so, we went through the trace tracking the active swarms at each timestamp and summing up all the download capacities of the active peers. With each new download request, we updated the amount of data left to be downloaded by each peer. Once a peer has finished downloading the file, it was removed from the active peers list. The download times were calculated according to equation (1). We plot the resulting amount of needed bandwidth in Figure 6.

This figure will be useful later to set a potential limit on the cloud seed's bandwidth when evaluating the algorithm's efficiency. It shows that the total need in cloud bandwidth does not exceed 650 Mbps. So, when evaluating the algorithm, it would be better to vary the cloud limit $W \in ]0, 650[$ in order to measure the effect of the seed's capacity on the algorithm's performance.

### *7.2. Experimental settings*

To evaluate the efficiency of our proposal, we developed a Python [5] script that simulates the bandwidth distribution and switching algorithm (Algorithm 1) and logs the results in two different log files. The first log file is related to the seed: it keeps a log of the current state of the seed. At each timestamp, several parameters are logged including: the amount of needed bandwidth, the amount of bandwidth served by the seed and the current number of swarms and clients. The second log keeps track of the start time and end time of each download. The download times are updated

[3] Canonical Ltd: http://www.canonical.com
[4] FP7 CloudSpaces Project: http://www.cloudspaces.eu
[5] Python Software Foundation: http://www.python.org

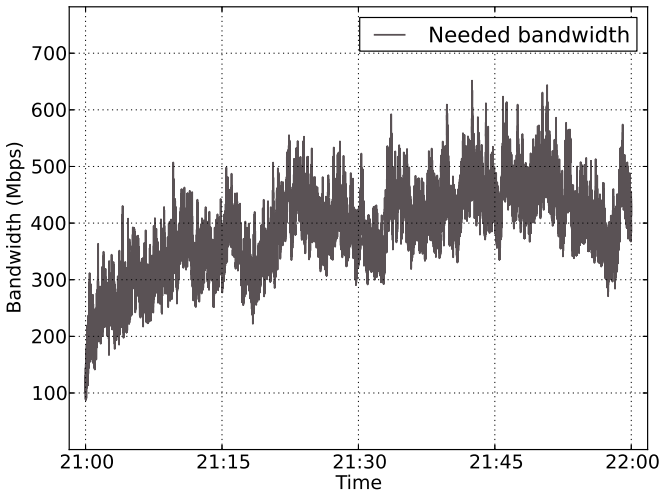



Figure 6: Needed upload bandwidth over time during the peak hour

at each timestamp according to equations (1) and (2), for HTTP and BT swarms, respectively.
In order to evaluate the results, we also developed another script that simulates the default behavior of the seed in which each download operation is treated individually and the download times are calculated according to equation (1). This simulator also keeps similar logs as the algorithm simulator in order to facilitate the comparison of the approaches.

### *7.3. Results*

We exploit the previously described trace sample of UB1 and re-simulate the arrival pattern of the peers to validate our approach. We run both simulators with a wide combination of $\tau$ and $W$ values and collect the logs of each experiment. Then, we evaluate our algorithm comparing the results with the ones obtained using the default strategy with the same bandwidth limits.

First of all, we run the simulator fixing the upload bandwidth budget at 300 Mbps and varying the switching constraint $\tau$. The goal is to get a first idea of the performance of the algorithm. We measure for each simulation, the download time taken by each operation and compare them to the times measured using the HTTP-only simulator with the same budget limit. It is important here to note that the download times are measured in seconds with a precision of one millisecond. We classify the operations into three different categories: operations that have gained in download time with the algorithm, operations that experienced losses and operations whose download time is left unchanged for both approaches.

Table 2: Percentages of operations with gains and losses in download time resulted by the algorithm compared to pure HTTP use. The cloud upload bandwidth budget limit is $W$=300 Mbps.

| | $\tau_1 = -0.2$ | $\tau_2 = 0$ | $\tau_3 = 0.2$ | $\tau_4 = 0.4$ |
|---|---|---|---|---|
| % of operations with gain | 82.89 % | 82.9 % | 83.43 % | 83.53 % |
| % of operations with loss | 2.23 % | 2.31 % | 2.48 % | 2.85 % |
| % of operations with no difference | 14.88 % | 14.79 % | 14.09 % | 13.62 % |
| Total % | 100 % | 100 % | 100 % | 100 % |

Table 2 presents the percentages of the operations in each category. We notice that for the three different values of $\tau$, more than 80% of the operations benefited from a gain in download time, about 15% kept the same time and only about 2.5% of them lost in download time. Even though these percentages are quite good, we need to make sure that the cumulative gains are higher than the losses. To do so, we sum all the download times of all operations for

both approaches and calculate the total net gain percentage (*net_gain_%*). *net_gain_%* represents the percentage ratio between the total time gained (or lost) by using the algorithm (*net_gain_hours* = *sum_http_hours* − *sum_algo_hours*) and the total download times using HTTP only (*sum_http_hours*).

$$net\_gain\_\% = \frac{net\_gain\_hours}{sum\_http\_hours} \times 100 = \frac{sum\_http\_hours - sum\_algo\_hours}{sum\_http\_hours} \times 100$$

Table 3: Total sum of all the download times for all the operations and the net gain percentage for the algorithm applied on the one-hour sample of the UB1 trace. The cloud upload bandwidth budget limit is $W$=300 Mbps.

| | $\tau_1 = -0.2$ | $\tau_2 = 0$ | $\tau_3 = 0.2$ | $\tau_4 = 0.4$ |
|---|---|---|---|---|
| *sum_http_hours* (in hours) | 2450.2 | 2450.2 | 2450.2 | 2450.2 |
| *sum_algo_hours* (in hours) | 2000.98 | 1997.05 | 1952.73 | 1906.56 |
| *net_gain_hours* (in hours) | 449.22 | 453.15 | 497.47 | 543.64 |
| *net_gain_%* | **18.33**% | **18.49**% | **20.3**% | **22.19**% |

Table 3 presents the total sum of all the download times of all the download operations and the net gain percentage based on the UB1 one-hour sample. The first row represents the sum of download times using HTTP. It is important to mention here that, for HTTP, this sum depends only on the cloud upload bandwidth budget $W$. Hence, for the fixed bandwidth $W$ = 300Mbps, it is always equal to 2450.2 hours, regardless of the $\tau$ constraint. However, the sum of the download times using the algorithm with a given cloud bandwidth limit depends highly on the switching constraint $\tau$. In Table 3, we compare the results with three different $\tau$ values:

Similarly, with the third and fourth constraints $\tau_3 = +0.2$ and $\tau_4 = +0.4$ (switch only if the corresponding peers will gain 20%, respectively 40%, or more gain in download time), the net gain percentage gets higher and reaches more than 20% of the total download time of all peers.

The first constraint is $\tau_1 = -0.2$: this constraint can be translated as follows: at a certain timestamp, a swarm can switch from HTTP to BitTorrent only if it will only lose less than 20% in download time. Under this constraint, we notice that the algorithm performs better than HTTP with a net gain in the client's download time equal to 18.33%. Next, we make the constraint a little bit stricter and we accept only switches to BT when the peers in question will only gain in download ($\tau_2 = 0$, no loss is permitted). We notice that the net gain percentage improves slightly. This is because the constraint will prevent swarms with negative gains from switching which will result in an increase of the total amount of net gain hours.

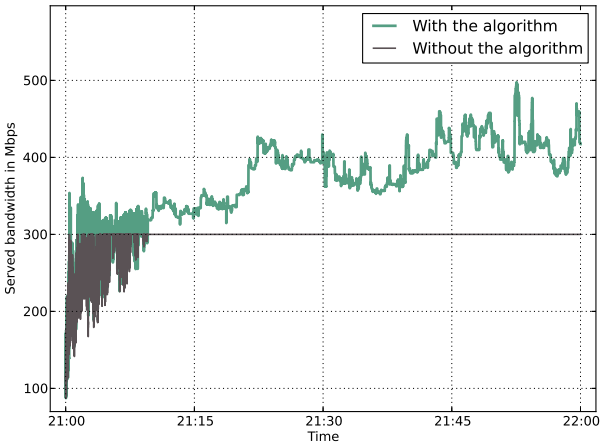
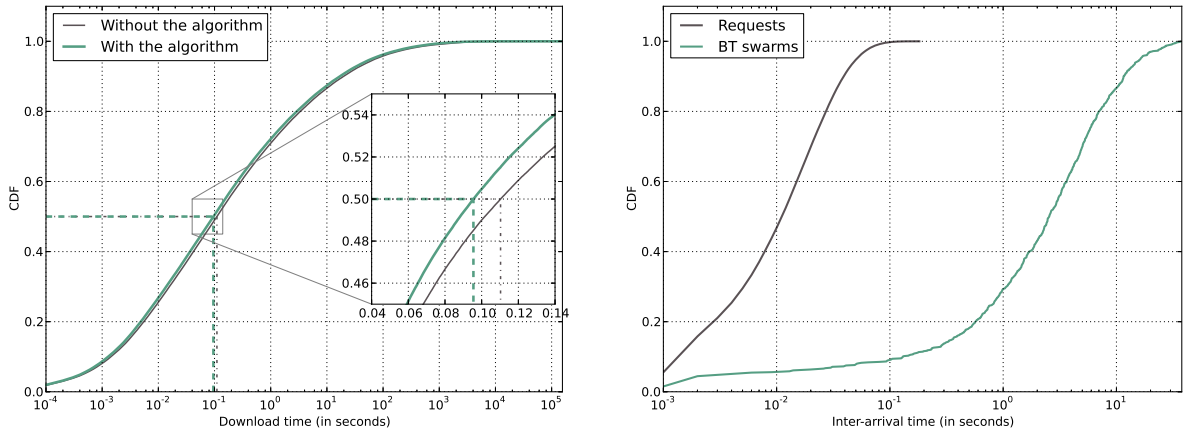



Figure 7: Amount of bandwidth served to the clients with and without the algorithm. The extra served bandwidth with the algorithm comes from the peers involved in BitTorrent swarms. Settings: $W$ = 300 Mbps and $\tau$ = 0.4.

To measure the efficiency of the algorithm for a specific configurations, we fix the switching constraint $\tau = 0.4$ and we suppose that the cloud's upload budget limit $W = 300$ Mbps. Figure 7 shows the efficiency of taking advantage of the upload speed of the peers. It compares the amount of bandwidth served by the seed to clients without using the algorithm (all files are distributed via HTTP) versus the total amount that becomes available when using the algorithm. This latter includes, in addition to the cloud's upload limit, the upload speed of the clients who switched to BitTorrent. We notice that the peers' contribution can reach up to 60% of the total cloud's budget. The total operations switched to BT represents only 2.45% of the operations with multiple downloads, which corresponds to less than 1% of the total number of operations, and the average size of the swarms switched to BitTorrent is 2.206 peers per swarm. Despite this limited number of switched operations, we notice in Figure 8b that the download times are reduced using the algorithm. As a matter of fact, the average download time without using the algorithm is 39.11 seconds. This time is reduced by 22.19% using the algorithm to only 30.43 seconds. Figure 8b presents the CDF of the inter-arrival times of requests and BT swarms. The average inter-arrival rate of the download requests (time between each arrival of a download request into the system and the next) is 0.0159 seconds. The average inter-arrival rate of the BT swarms (time between each creation of a BT swarm and the next) is 4.5702 seconds.



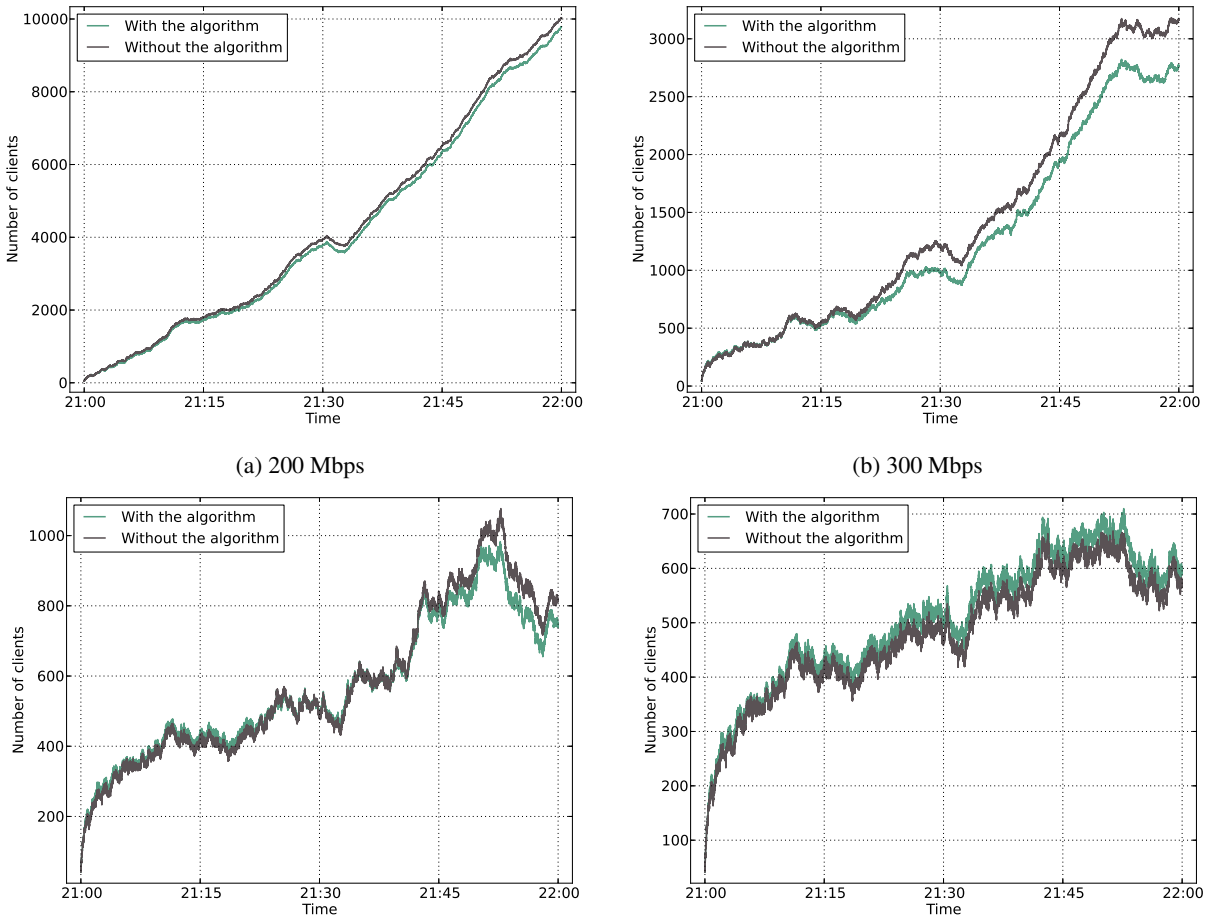

(a) Download time

(b) Interarrival time

Figure 8: CDF download times and inter-arrival time of requests and BT swarms with and without the algorithm. Settings: $W = 300$ Mbps and $\tau = 0.4$.

After evaluating the general performance of the algorithm, we study the effect of the bandwidth limit $W$. Figure 9 compares the number of simultaneous clients in HTTP-only mode and using the algorithm for four different values of $W$ (200, 300, 400 and 500 Mbps, respectively). It is important to mention here that in our simulator, peers do not stay as seeders in the system. They leave as soon as they finish downloading the requested files. When comparing between the number of simultaneous peers using each of the approaches, we note that a lower number of simultaneous peers means that the clients are downloading faster which proves that the corresponding approach is more efficient. We notice that with very limited bandwidth budget (200 and 300 Mbps), the algorithm performs better than pure HTTP. This due to the fact that when the seed has a very limited bandwidth budget, the share each client gets will be small. Therefore, the HTTP download time will be "high" and the overhead of switching to BitTorrent will be negligible. However, the higher the seed bandwidth gets, the bigger the overhead becomes compared to the download time in HTTP. This explains the degrade in the algorithm's performance when the seed's bandwidth budget becomes quite high (400 Mbps).

Figure 10 presents, for different values of $W$ ranging from 180 Mbps to 500 Mbps, the net gain percentage the average size of BT swarms. The first figure (Figure 10a) plots the evolution of the net gain percentage with the cloud's bandwidth. Similar to the aforementioned conclusions, when the bandwidth is small (lower than 320 Mbps), the net gain percentage in download time of the clients is important (between 17.5% and 21%). However it gets lower with the increasing budget of the cloud, until reaching negative values when the seed's bandwidth is higher than 420 Mbps. This confirms our previous conclusions that the algorithm is more efficient when the cloud seed has very limited




(a) 200 Mbps

(b) 300 Mbps

(c) 400 Mbps

(d) 500 Mbps

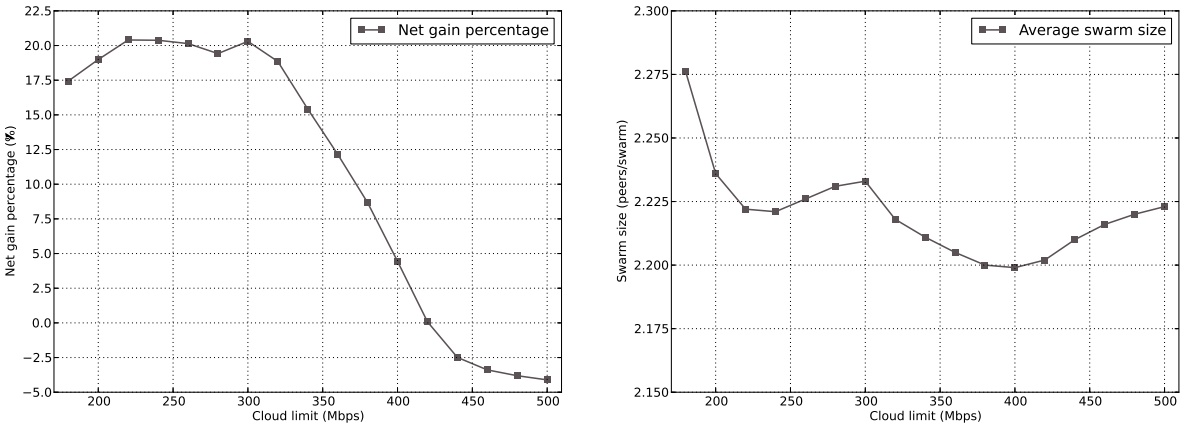
Figure 9: CDF download times and inter-arrival time of requests and BT swarms with and without the algorithm. Settings: $W = 300$ and $\tau = 0.4$.

bandwidth resources. The second figure (Figure 10b) presents the average number of peers in BT swarms. We notice that most of the BT swarms are very small with an average size of about 2.22 peers per swarm. This can be due to the limited sharing in UB1 system and to the fact that most of UB1 users are using the service for data backup only.

### *7.4. Algorithm's performance*

To measure the efficiency of the algorithm, we measure the time needed to simulate the trace. The total simulation of the trace (more than 225,000 operations) took around 82 minutes until all the downloads have finished. We also measure the time needed to calculate the bandwidth distribution for each timestamp (with the arrival of each new download operation). This time corresponds to one execution of the algorithm and we refer to it as the execution time. Figure 11a presents the CDF of this execution time. It varies between 0.005 and 71.566 milliseconds. The mean and median execution times are 15.178 and 8.006 milliseconds, respectively.

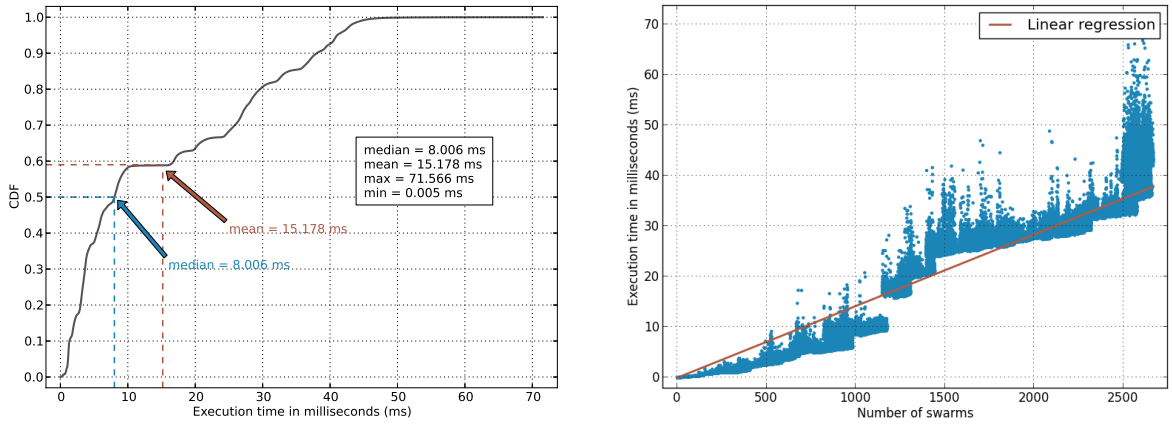
To measure the effect of the number of swarms on the execution time at each round of the algorithm, we depict the scatter plot of the execution time as a function of the number of swarms in Figure 11b. We also plot the regression line using the ordinary least squares (OLS) method. The regression coefficient is equal to 0.01418. This means that there is a potential linear relationship between the number of swarms and the execution time.




(a) Net gain percentage

(b) Average BitTorrent swarms' size

Figure 10: Net gain percentage and average size of the BT swarms for different cloud bandwidth limits ranging between 180 and 500 Mbps. The switching constraint considered here is $\tau = 0.2$.




(a) CDF execution time

(b) Execution time per number of swarms

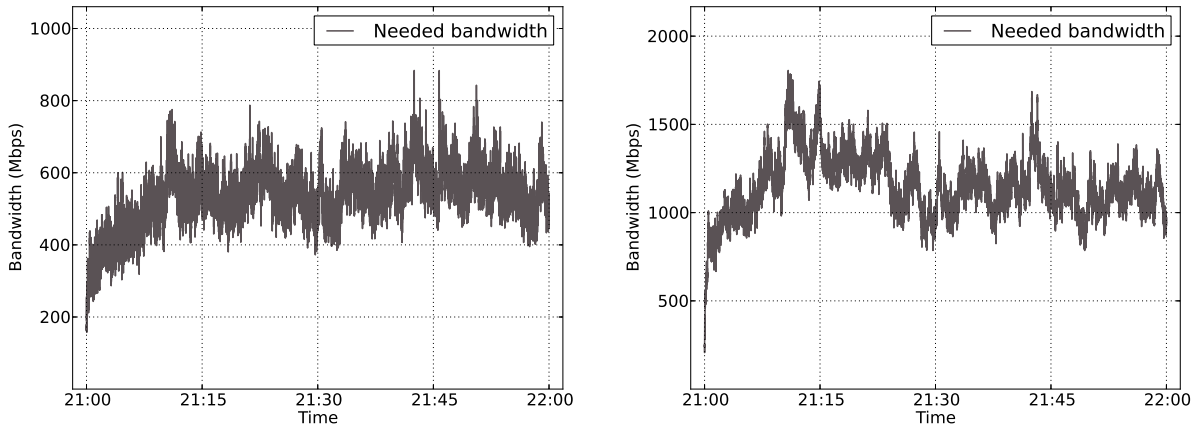
Figure 11: Algorithm's perfomance during the trace simulation

### *7.5. Modified trace: bigger shared files: Results*

Even though the UB1 trace has limited sharing and very small files (most of the files are smaller than 1 MB), we could achieve relatively important improvements in the system's performance. To further validate our proposal, we modified the trace in order to have bigger shared files. Our idea was to keep the same arrival pattern of the peers and just increase the size of the files that were downloaded more than once. We obtained two different modified traces:

- *trace*_1: This trace preserves the same arrival pattern as in the original trace, but we increase the size of the files smaller than 1 MB by 1 MB. For instance, if a file $f_s$ is downloaded more than once in the original trace sample and has a file size of 100 KB, then, in *trace*_1, the same file would be 1 MB (1024 KB) bigger, that is 1124 KB. We chose this value (1 MB) because it represents the mean file size of all the files in the original trace.

- *trace*_2: This trace is obtained the same way as *trace*_1. We chose a bigger limit on size equal to 5 MB, which is the average size of a picture. This means that *trace*_2 also preserves the same arrival pattern as in the original trace, but here we increase the size of the files smaller than 5 MB by 5 MB. For instance, if a file $f_s$ is




(a) *trace*_1 (b) *trace*_2

Figure 12: New bandwidth requirements of the modified traces

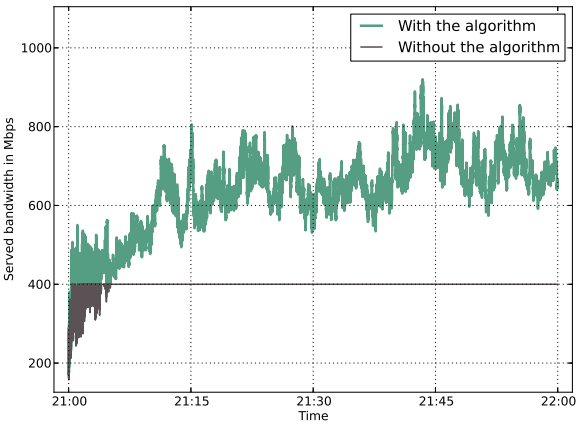
downloaded more than once in the original trace sample and has a file size of 1 MB, in *trace*_2, the same file would be 5 MB bigger, that is 6 MB.

Clearly, when we increase the size of some files, the amount of bandwidth needed to distribute the requested file to the peers will increase too. Figure 12 presents the new required bandwidth for both traces and shows that *trace*_1 and *trace*_2 require clearly more bandwidth compared to the original requirements (Figure 6). We apply later our algorithm on both traces and compare the results. We use the same switching constraint $\tau = 0.4$ and we fix the cloud's upload budget limit $W$ to 400 Mbps (1000 Mbps, respectively) for *trace*_1 (*trace*_2, respectively).

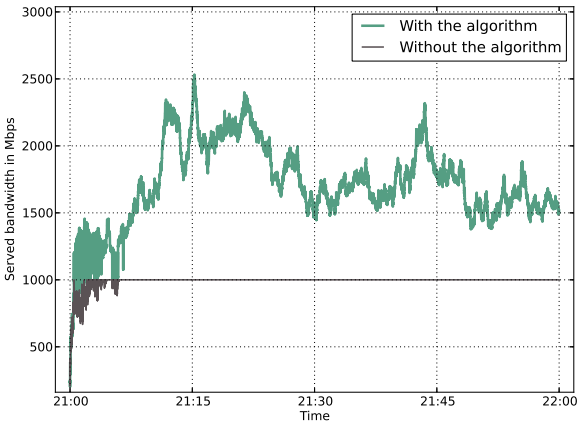
Table 4: Comparison of the results with the three different traces. The experiments settings are the followings: $\tau = 0.4$ for all the traces, $W$ is 300 Mbps for the original trace, 400 Mbps for *trace*_1 and 1000 Mbps for *trace*_2

| | original trace | *trace*_1 | *trace*_2 |
|---|---|---|---|
| Number of operations affected by the change | 0 | 69,286 | 65,992 |
| (% of the total number of operations) | (0%) | (30.72%) | (29.26%) |
| Number of operations switched to BitTorrent | 1734 | 36,727 | 43,997 |
| (% of the total number of operations) | (0.76%) | (16.28%) | (19.5%) |
| Number of BitTorrent swarms created | 786 | 9,324 | 10,763 |
| Average BitTorrent swarm size (in peer/swarm) | 2.206 | 3.93 | 4.08 |
| Average inter-arrival time of BitTorrent swarms (in seconds) | 4.570 | 0.386 | 0.334 |
| Average download time without the algorithm (in seconds) | 39.11 | 106.09 | 186.82 |
| Average download time with the algorithm (in seconds) | 30.43 | 52.77 | 61.08 |
| *net_gain_%* | **22.19%** | **50.26%** | **67.30%** |

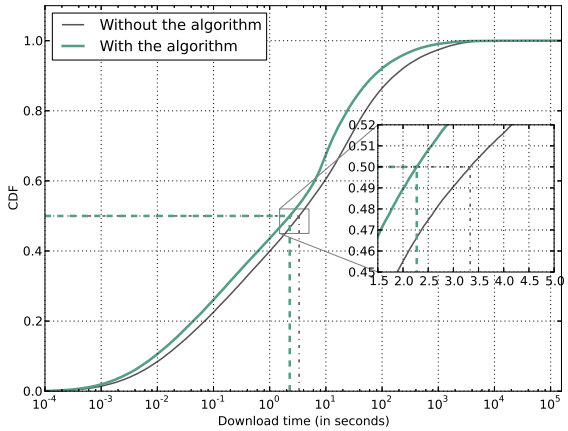
Figure 13 and Table 4 summarize the results with and without the algorithm. As we can see in Figures 13a and 13b, the amount of bandwidth contributed by the BT clients can reach up to 100% of the cloud's initial limit. In addition, we notice important improvements in the net gain percentage that increases from about 22% for the original trace to reach over 50% when the files are bigger than 1 MB and more than 65% when the files are bigger than 5 MB. In fact, increasing the file sizes results in increased probability of switching to BitTorrent. Actually, the percentage of operations switched to BT grows from 0.76% in the original trace and reaches 19.5% in *trace*_2. This leads to a noticeable decrease in the inter-arrival time of BT swarms as seen in Figures 13e and 13f. The frequency of creation of new BT swarms increases from 0.21 swarms per second for the original trace to 2.59 swarms per second for *trace*_1



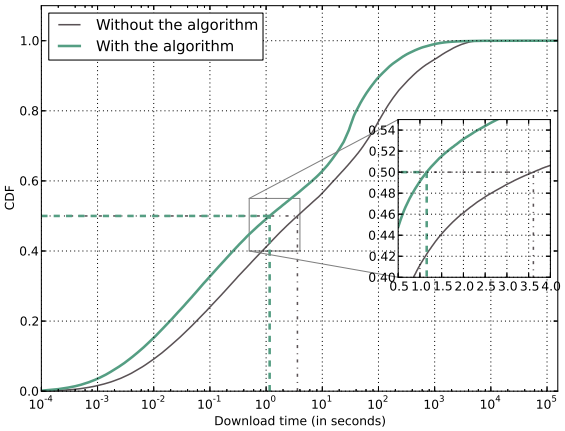

(a) *trace*_1: Served bandwidth



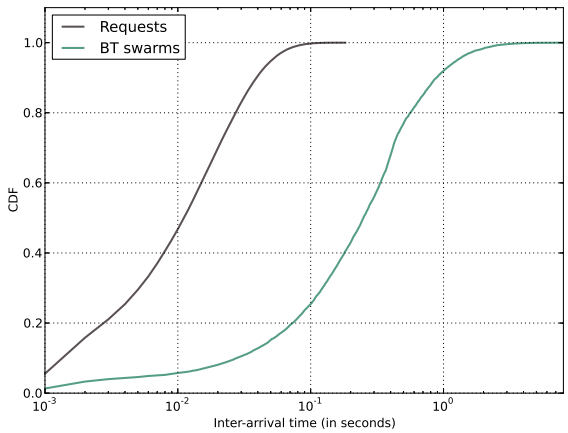

(b) *trace*_2: Served bandwidth



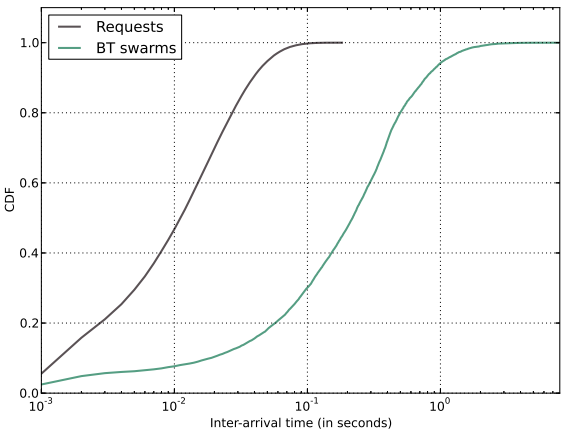

(c) *trace*_1: CDF download times




(d) *trace*_2: CDF download times




(e) *trace*_1: CDF inter-arrival times




(f) *trace*_2: CDF inter-arrival times

Figure 13: Results using the modified traces

and reaches 2.98 swarms per second for *trace*_2. Similarly, the average size of the swarms increases from 2.206 peers per swarm to reach about 4.08 peers per swarm when the sizes of the shared files become bigger.

## 8. Conclusions

In this paper, we propose a bandwidth allocation and protocol management algorithm that can be implemented in personal cloud systems with limited bandwidth budget. Based on the demand on the cloud and the load on each file, the cloud server is able to decide whether to use a client-server approach (HTTP) or a peer-assisted one (BitTorrent) to distribute that file. Our proposed algorithm for the management of cloud bandwidth achieves important improvements in terms of download time for the clients, even though in our simulator's implementation we were "stricter" on BitTorrent than HTTP. In fact, we used an estimation of the download time in HTTP that does not take into account the protocol's overheads. However, on the other hand, we added to BitTorrent the potential latency of the peers discovery phase and the delay that can be caused by pieces unavailability. Moreover, we considered the "worst case scenario" where the peers leave the system as soon as they finish download, while in reality, the synchronization process works always in the background without the user being aware of it. This means that it is more probable that the peers will stay longer, even after finishing the download and contribute more to the system. Despite that, the results prove that the use of BitTorrent in personal cloud systems can help the clients gain in download time, especially when the bandwidth resources of the seed are limited. In such conditions, the net gain percentage in the download time of all the peers exceeds 20% of their download time in most cases, based on a real trace of the UB1 system.

The original UB1 trace has limited sharing and very small files. For this reason, we modified it in order to have bigger shared files. The application of the algorithm on the modified traces results in important improvements in the download time that exceed 65% of the original download time of all the peers.

Nevertheless, several extensions can be added to the algorithm. For instance, it is possible to consider two different values of the switching constraint based on the load of the seed: $\tau_{overloaded}$ and $\tau_{not\_overloaded}$. This way, strict constraints can be put when the seed is not overloaded and loosened it up when the load on the seed increases.

Our future plans include the study of the efficiency of file bundling in personal cloud systems and measure the gain in terms of cloud bandwidth and peers download time.

## Acknowledgments

This work has been partially funded by the Spanish Ministry of Economy and Competitiveness in the context of the project *Cloud Services and Community Networks* (TIN2013-47245-C2-2-R) and by EU in the context of the projects *CloudSpaces: Open Service Platform for the Next Generation of Personal clouds* (FP7-317555) and *IOStack: Software-defined Storage for Big Data* (H2020-ICT-2014-7-1).

## References

[1] I. Drago, Understanding and Monitoring Cloud Services., Ph.D. thesis, University of Twente (2013).

[2] B. Cohen, Incentives Build Robustness in BitTorrent (2003).

[3] R. Chaabouni, M. Sanchez-Artigas, P. Garcia-Lopez, Reducing Costs in the Personal Cloud: Is Bittorrent a Better Bet?, in: Peer-to-Peer Computing (P2P), 14-th IEEE International Conference on, 2014, pp. 1–10. doi:10.1109/P2P.2014.6934302.

[4] R. Sweha, V. Ishakian, A. Bestavros, Angels in the Cloud: A Peer-Assisted Bulk-Synchronous Content Distribution Service, in: Cloud Computing (CLOUD), 2011 IEEE International Conference on, 2011, pp. 97–104. doi:10.1109/CLOUD.2011.84.

[5] R. Wartel, T. Cass, B. Moreira, E. Roche, M. Guijarro, S. Goasguen, U. Schwickerath, Image Distribution Mechanisms in Large Scale Cloud Providers, in: Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, 2010, pp. 112–117. doi:10.1109/CloudCom.2010.73.

[6] M. Schmidt, N. Fallenbeck, M. Smith, B. Freisleben, Efficient Distribution of Virtual Machines for Cloud Computing, in: Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on, 2010, pp. 567–574. doi:10.1109/PDP.2010.39.

[7] J. Reich, O. Laadan, E. Brosh, A. Sherman, V. Misra, J. Nieh, D. Rubenstein, VMtorrent: virtual appliances on-demand, in: SIGCOMM, 2010, pp. 453–454.

[8] R. Chaabouni, P. Garcia Lopez, M. Sanchez Artigas, S. Ferrer Celma, C. Cebrian, Boosting Content Delivery with BitTorrent in Online Cloud Storage Services, in: Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on, 2013, pp. 1–2. doi:10.1109/P2P.2013.6688731.

[9] B. Wei, G. Fedak, F. Cappello, Scheduling Independent Tasks Sharing Large Data Distributed with BitTorrent, in: Grid Computing, 2005. The 6th IEEE/ACM International Workshop on, 2005, pp. 8 pp.–. doi:10.1109/GRID.2005.1542745.

[10] R. Kumar, K. Ross, Peer-Assisted File Distribution: The Minimum Distribution Time, in: Hot Topics in Web Systems and Technologies, 2006. HOTWEB '06. 1st IEEE Workshop on, 2006, pp. 1–11. doi:10.1109/HOTWEB.2006.355259.

[11] D. Qiu, R. Srikant, Modeling and Performance Analysis of BitTorrent-like Peer-to-peer Networks, in: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM '04, ACM, New York, NY, USA, 2004, pp. 367–378. doi:10.1145/1015467.1015508.

[12] C. Carbunaru, Y. M. Teo, B. Leong, T. Ho, Modeling Flash Crowd Performance in Peer-to-Peer File Distribution, Parallel and Distributed Systems, IEEE Transactions on 25 (10) (2014) 2617–2626. doi:10.1109/TPDS.2013.220.

[13] X. Leon, R. Chaabouni, M. Sanchez Artigas, P. Garcia Lopez, Smart Cloud Seeding for BitTorrent in Datacenters, Internet Computing, IEEE 18 (4) (2014) 47–54. doi:10.1109/MIC.2014.43.

[14] Peterson, Ryan and Sirer, Emin Gün, AntFarm: Efficient Content Distribution with Managed Swarms., in: NSDI, Vol. 9, 2009, pp. 107–122.

[15] A. Sharma, A. Venkataramani, A. Rocha, Pros amp; cons of model-based bandwidth control for client-assisted content delivery, in: Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference on, 2014, pp. 1–8.

[16] Pedro García-López, Marc Sánchez-Artigas, Cristian Cotes, Guillermo Guerrero, Adrian Moreno and Sergi Toda, StackSync: Architecturing the Personal Cloud to Be in Sync, http://stacksync.org/wp-content/uploads/2013/11/stacksync_full_paper.pdf.

[17] I. Drago, M. Mellia, M. M. Munafo, A. Sperotto, R. Sadre, A. Pras, Inside Dropbox: Understanding Personal Cloud Storage Services, in: Proceedings of the 2012 ACM Conference on Internet Measurement Conference, IMC '12, ACM, New York, NY, USA, 2012, pp. 481–494. doi:10.1145/2398776.2398827.

[18] I. Drago, E. Bocchi, M. Mellia, H. Slatman, A. Pras, Benchmarking Personal Cloud Storage, in: Proceedings of the 2013 conference on Internet measurement conference, ACM, 2013, pp. 205–212.

[19] R. Gracia-Tinedo, M. Sanchez Artigas, A. Moreno-Martinez, C. Cotes, P. Garcia Lopez, Actively Measuring Personal Cloud Storage, in: Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on, 2013, pp. 301–308. doi:10.1109/CLOUD.2013.25.

[20] P. G. Lopez, M. Sanchez-Artigas, S. Toda, C. Cotes, J. Lenton, StackSync: Bringing Elasticity to Dropbox-like File Synchronization, in: Proceedings of the 15th International Middleware Conference, Middleware '14, ACM, New York, NY, USA, 2014, pp. 49–60. doi:10.1145/2663165.2663332.

[21] Dropbox, Inc., Dropbox for Business Security: A Dropbox Whitepaper, https://www.dropbox.com/static/business/resources/dfb_security_whitepaper.pdf.

## Appendix A. Inverting the gain formulas: Solving the equation: $Gain(\overset{??}{w_s}, s) = \tau$

The goal of this section is to reverse the equations of the gain and get, for a given swarm $s$ and a given file of size $F_s$, the amount of bandwidth needed to be provided by the seed $w_s$ that satisfies the condition $Gain(w_s, s) \geq \tau$. We remind our reader that the gain percentage is defined as follows:

$$Gain\,(w_s, s) = \begin{cases} -\dfrac{\alpha_{bt}\, d_{min,s}}{F_s}, & \text{if } d_{min,s} \leq \dfrac{w_s}{L_s} \text{ and } d_{min,s} \leq \min\left\{\dfrac{w_s + \eta_s\, u_s\, L_s}{L_s}, w_s\right\} \\ 1 - \dfrac{w_s}{L_s\, d_{min,s}} - \dfrac{\alpha_{bt}\, w_s}{F_s\, L_s}, & \text{if } \dfrac{w_s}{L_s} \leq d_{min,s} \text{ and } d_{min,s} \leq \min\left\{\dfrac{w_s + \eta_s\, u_s\, L_s}{L_s}, w_s\right\} \\ 1 - \dfrac{w_s}{w_s + \eta_s\, u_s\, L_s} - \dfrac{\alpha_{bt}\, w_s}{F_s\, L_s}, & \text{if } \dfrac{w_s + \eta_s\, u_s\, L_s}{L_s} \leq \min\{d_{min,s}, w_s\} \\ 1 - \dfrac{1}{L_s} - \dfrac{\alpha_{bt}\, w_s}{F_s\, L_s}, & \text{if } w_s \leq \min\left\{d_{min,s}, \dfrac{w_s + \eta_s\, u_s\, L_s}{L_s}\right\}. \end{cases}$$

While inverting this equation and in order to be able to define correctly the interval delimiters, we need to distinguish two different cases based on the maximum of $(L_s - 1)\, d_{min,s}$ and $L_s\, \eta_s\, u_s$:

- **Case A:** $(L_s - 1)\, d_{min,s} \geq L_s\, \eta_s\, u_s$
- **Case B:** $(L_s - 1)\, d_{min,s} \leq L_s\, \eta_s\, u_s$

### *Appendix A.1. Case A:* $(L_s - 1)\, d_{min,s} \geq L_s\, \eta_s\, u_s$

The general shape of that function is given in figure A.14. The next step is to find expressions of the intervals delimiters ($lim_1$, $lim_2$, $lim_3$ and $lim_4$) and the corresponding gain values ($Gain(lim_1, s)$, $Gain(lim_2, s)$, $Gain(lim_3, s)$ and $Gain(lim_4, s)$).

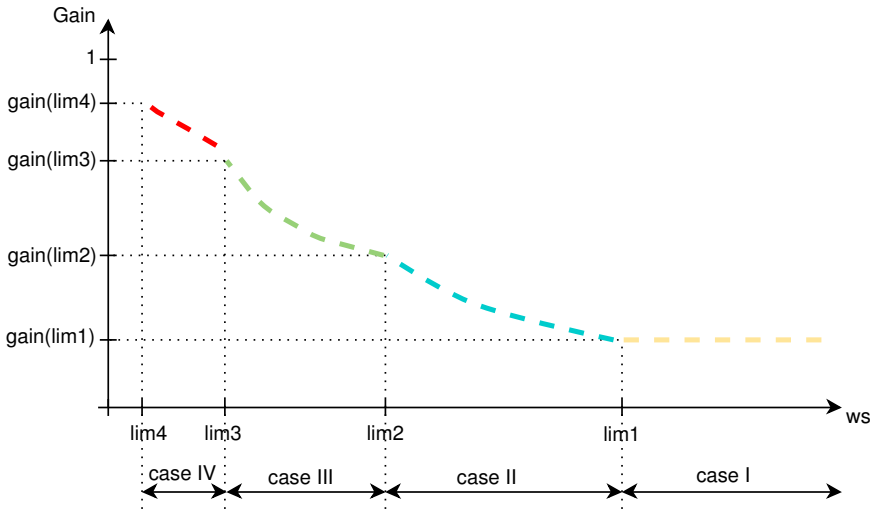



Figure A.14: General shape of the gain ratio as a function of the upload speed of the seed in case A when $(L_s - 1)\, d_{min,s} \geq L_s\, \eta_s\, u_s$

1. $lim_1$ and $Gain(lim_1, s)$: The conditions of case I are the followings:

$$\begin{cases} d_{min,s} \leq \frac{w_s}{L_s} \\ d_{min,s} \leq \frac{w_s + \eta_s\, L_s\, u_s}{L_s} \\ d_{min,s} \leq w_s \end{cases} \xrightarrow[\eta_s\, u_s \geq 0]{L_s \geq 1} \begin{cases} d_{min,s} \leq \frac{w_s}{L_s} \leq w_s \\ d_{min,s} \leq \frac{w_s}{L_s} \leq \frac{w_s + \eta_s\, L_s\, u_s}{L_s} \end{cases} \implies \boxed{w_s \geq L_s\, d_{min,s} \implies lim_1 = L_s d_{min,s}}$$

The corresponding $Gain(lim_1, s)$ for case I is as follows:

$$Gain(lim_1, s) \overset{caseI}{=} Gain(L_s\, d_{min,s}, s) \overset{caseI}{=} -\frac{\alpha_{bt}.d_{min,s}}{F_s}$$

**- Resolution of the equation** $Gain(\overset{??}{w_s}, s) = \tau,\ \forall \tau \in ]-\infty, -\frac{\alpha_{bt}\, d_{min,s}}{F_s}]$

We have $\tau \in ]-\infty, -\frac{\alpha_{bt}.d_{min,s}}{F_s}]$ and $Gain(\overset{??}{w_s}, s) = \tau$.

This leads to $Gain(\overset{??}{w_s}, s) \in ]-\infty, -\frac{\alpha_{bt}\, d_{min,s}}{F_s}]$, this means that $w_s \geq L_s\, d_{min,s}$.

Thus, $\forall \tau \in ]-\infty, -\frac{\alpha_{bt}\, d_{min,s}}{F_s}]$, the optimal bandwidth that should be allocated to the swarm without violating the constraint ($Gain(w_s, s) \geq \tau$) is $w_s = L_s.d_{min,s}$

$$\boxed{\forall \tau \in \left]-\infty, -\frac{\alpha_{bt}\, d_{min,s}}{F_s}\right], \quad \left(Gain(\overset{??}{w_s}, s) = \tau\right) \Rightarrow (w_s = L_s\, d_{min,s})}$$

2. $lim_2$ and $Gain(lim_2, s)$: The conditions of case II are the followings:

$$\begin{cases} d_{min,s} \geq \frac{w_s}{L_s} & (lim_1) \\ d_{min,s} \leq \frac{w_s + \eta_s\, L_s\, u_s}{L_s} \\ d_{min,s} \leq w_s \end{cases} \Rightarrow \begin{cases} w_s \geq d_{min,s} \\ w_s \geq L_s\,(d_{min,s} - \eta_s\, u_s) \end{cases} \Rightarrow w_s \geq \max\left(d_{min,s}, L_s\,(d_{min,s} - \eta_s\, u_s)\right)$$

$$\Rightarrow lim_2 = \max\left(d_{min,s}, L_s\,(d_{min,s} - \eta_s\, u_s)\right) \overset{\text{Case A}}{\Longrightarrow} \boxed{lim_2 = L_s\,(d_{min,s} - \eta_s\, u_s)}$$

Let's verify whether $lim_1 \overset{?}{\geq} lim_2$:

$$\begin{aligned} lim_1 - lim_2 &= L_s\, d_{min,s} - L_s\,(d_{min,s} - \eta_s\, u_s) = L_s\, d_{min,s} - L_s\, d_{min,s} + L_s\, \eta_s\, u_s \\ &= L_s\, \eta_s\, u_s \geq 0 \quad \text{(because } L_s \geq 1,\ \eta_s \geq 0 \text{ and } u_s \geq 0) \quad \text{(verified ✓)} \end{aligned}$$

The corresponding $Gain(lim_2, s)$ for case II is as follows:

$$Gain(lim_2, s) \overset{caseII}{=} 1 - \frac{L_s\, d_{min,s} - L_s\, \eta_s\, u_s}{L_s\, d_{min,s}} - \frac{\alpha_{bt}(L_s\, d_{min,s} - L_s\, \eta_s\, u_s)}{F_s\, L_s} = \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s}$$

This formula should be verified using the gain formula for case III since the two cases share the same border $lim_2$:

$$\begin{aligned} Gain(lim_2, s) &\overset{caseIII}{=} 1 - \frac{L_s\, d_{min,s} - L_s\, \eta_s\, u_s}{L_s\, d_{min,s} - L_s\, \eta_s\, u_s + \eta_s\, L_s\, u_s} - \frac{\alpha_{bt}\,(L_s\, d_{min,} - L_s\, \eta_s\, u_s)}{F_s\, L_s} \\ &= \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}\,(d_{min,s} - \eta_s\, u_s)}{F_s} \quad \text{(verified ✓)} \end{aligned}$$

We need now to verify whether $Gain(lim_2, s) \overset{??}{\geq} Gain(lim_1, s)$

$$Gain(lim_2, s) - Gain(lim_1, s) = \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}\,(d_{min,s} - \eta_s\, u_s)}{F_s} + \frac{\alpha_{bt}\, d_{min,s}}{F_s}\,\frac{\eta_s\, u_s}{d_{min,s}} + \frac{\alpha_{bt}\, \eta_s\, u_s}{F_s} \geq 0$$

$$\text{Thus,} \quad \frac{\alpha_{bt}.d_{min,s}}{F_s} \leq \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s} \quad \text{(verified ✓)}$$

**- Resolution of the equation** $Gain(\overset{??}{w_s}, s) = \tau,\ \forall \tau \in \left]-\frac{\alpha_{bt}.d_{min,s}}{F_s}, \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s}\right]$

We have: $Gain(\overset{??}{w_s}, s) = \tau$ and $\tau \in \left]-\frac{\alpha_{bt}\, d_{min,s}}{F_s}, \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s}\right]$

This means that $Gain(\overset{??}{w_s}, s) \in \left]-\frac{\alpha_{bt}\, d_{min,s}}{F_s}, \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s}\right]$ which corresponds to the formula of the gain

related to case II.
Let's try now to invert that formula in order to get an estimation of $w_s$

$$Gain(\overset{??}{w_s}, s) = \tau \iff^{\text{case II}} 1 - \frac{w_s}{L_s.d_{min,s}} - \frac{\alpha_{bt}.w_s}{F_s\,L_s} = \tau$$
$$\iff 1 - \tau = w_s\left(\frac{1}{L_s\,d_{min,s}} + \frac{\alpha_{bt}}{F_s\,L_s}\right) \iff w_s = \frac{(1-\tau)F_s\,L_s\,d_{min,s}}{F_s + d_{min,s}\,\alpha_{bt}}$$

We can then conclude that:

$$\boxed{\forall\tau \in \left[-\frac{\alpha_{bt}\,d_{min,s}}{F_s}, \frac{\eta_s\,u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\,u_s)}{F_s}\right], \left(Gain(\overset{??}{w_s}, s) = \tau\right) \Rightarrow w_s = \frac{(1-\tau)\,F_s\,L_s\,d_{min,s}}{F + d_{min,s}\,\alpha_{bt}}}$$

3. $lim_3$ and $Gain(lim_3, s)$: The conditions of case III are the followings:

$$\begin{cases} \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le d_{min,s} \\ \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le w_s \end{cases} \implies \begin{cases} w_s \le L_s(d_{min,s} - \eta_s\,u_s) \quad (lim_2) \\ w_s \ge \frac{\eta_s\,L_s\,u_q}{L_s-1} \end{cases} \implies \boxed{lim_3 = \frac{\eta_s\,L_s\,u_s}{L_s - 1}}$$

Let's verify whether $lim_2 \overset{?}{\ge} lim_3$:

$$\begin{aligned} lim_2 - lim_3 &= L_s\,(d_{min,s} - \eta_s\,u_s) - \frac{\eta_s\,L_s\,u_s}{L_s - 1} = \frac{L_s}{L_s - 1}\,(L_s\,(d_{min,s} - \eta_s\,u_s) - d_{min,s}) \\ &\ge 0 \quad \text{(because } L_s > 1 \text{ and } d_{min,s} \le L_s\,(d_{min,s} - \eta_s\,u_s)) \quad \text{(verified ✓)} \end{aligned}$$

The corresponding $Gain(lim_3, s)$ for case III is:

$$\begin{aligned} Gain(lim_3, s) &\overset{caseIII}{=} Gain\left(\frac{\eta_s\,L_s\,u_s}{L_s - 1}, s\right) \overset{caseIII}{=} 1 - \frac{\frac{\eta_s\,L_s\,u_s}{L_s-1}}{\frac{\eta_s\,L_s\,u_s}{L_s-1} + \eta_s\,L_s\,u_s} - \frac{\alpha_{bt}\,\frac{\eta_s L_s\,u_s}{L_s-1}}{F_s\,L_s} \\ &= 1 - \frac{\eta_s\,L_s\,u_s}{\eta_s\,L_s\,u_s + \eta_s\,L_s\,u_s\,(L_s - 1)} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s} = 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s} \end{aligned}$$

This expression needs to be verified also using the gain formula for case IV since the two cases share the same border $lim_3$:

$$Gain(lim_3, s) \overset{caseIV}{=} Gain\left(\frac{\eta_s\,L_s\,u_s}{L_s - 1}, s\right) \overset{caseIV}{=} 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\left(\frac{\eta_s\,L_s\,u_s}{L_s-1}\right)}{F_s\,L_s} = 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s}$$

We need now to verify whether $Gain(lim_3, s) \overset{??}{\ge} Gain(lim_2, s)$

$$\begin{aligned} Gain(lim_3, s) - Gain(lim_2, s) &= 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s} - \left(\frac{\eta_s\,u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\,u_s)}{F_s}\right) \\ &= 1 - \frac{1}{L_s} - \frac{\eta_s\,u_s}{d_{min,s}} + \frac{\alpha_{bt}\,d_{min,s}}{F_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{F_s}\left(1 + \frac{1}{L_s - 1}\right) \\ &= \left[1 - \frac{1}{L_s} - \frac{\eta_s\,u_s}{d_{min,s}}\right] + \left[\frac{\alpha_{bt}\,d_{min,s}}{F_s} - \frac{\alpha_{bt}\,\eta_s\,u_s\,L_s}{F_s\,(L_s - 1)}\right] \\ &= (L_s(d_{min,s} - \eta_s\,u_s) - d_{min,s})\left(\frac{1}{L_s\,d_{min,s}} + \frac{\alpha_{bt}}{F_s\,(L_s - 1)}\right) \\ &\ge 0 \quad \text{(since all the terms are} \ge 0,\ L_s > 1 \text{ and } d_{min,s} \le L_s(d_{min,s} - \eta_s\,u_s)\text{ )} \end{aligned}$$

$$\text{Thus,} \quad \frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s} \leq 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\, u_s}{(L_s - 1)\, F_s} \qquad \text{(verified ✓)}$$

**- Resolution of the equation** $Gain(\overset{??}{w_s}, s) = \tau,\ \forall \tau \in \left[\frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s},\ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\, u_s}{(L_s - 1)\, F_s}\right]$

We have: $Gain(\overset{??}{w_s}, s) = \tau$ and $\tau \in \left[\frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s},\ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\, u_s}{(L_s - 1)\, F_s}\right]$

This means that $Gain(\overset{??}{w_s}, s) \in \left[\frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s},\ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\, u_s}{(L_s - 1)\, F_s}\right]$ which corresponds to the formula of the gain related to case III.

Let's try now to invert that formula in order to get an estimation of $w_s$

$$Gain(\overset{??}{w_s}, s) = \tau \quad \overset{\text{case III}}{\Longleftrightarrow} \quad 1 - \frac{w_s}{w_s + \eta_s\, L_s\, u_s} - \frac{\alpha_{bt}\, w_s}{F_s\, L_s} = \tau$$

Since the resolution of this equation is too complex, we tried to simplify it by introducing the following symbols: $a = \eta_s\, L_s\, u_s$, $b = \frac{\alpha_{bt}}{F_s\, L_s}$ and $c = \tau$. The simplified equation becomes:

$$1 - \frac{w_s}{w_s + a} - b\, w_s = c$$

To solve this second degree equation, we used an online solver [6]. We obtained the following solutions:

$$\begin{aligned} w_{s_1} &= \frac{\sqrt{a^2\, b^2 - 2\, a\, b\, c + 4\, a\, b + c^2} - a\, b - c}{2\, b} \\ w_{s_2} &= \frac{-\left(\sqrt{a^2\, b^2 - 2\, a\, b\, c + 4\, a\, b + c^2} + a\, b + c\right)}{2\, b} \end{aligned}$$

Clearly, $w_{s_2} < 0$ so it cannot be considered as a solution. Then, we can conclude that:

$$\begin{aligned} \forall \tau \quad \in \quad & \left[\frac{\eta_s\, u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\, u_s)}{F_s},\ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\, u_s}{(L_s - 1)\, F_s}\right], \\ & \left(Gain(\overset{??}{w_s}, s) = \tau\right) \Rightarrow w_s = \frac{\sqrt{a^2\, b^2 - 2\, a\, b\, c + 4\, a\, b + c^2} - a\, b - c}{2\, b} \end{aligned}$$

where $a = \eta_s\, L_s\, u_s$, $b = \dfrac{\alpha_{bt}}{F_s\, L_s}$, and $c = \tau$

4. $lim_4$ and $Gain(lim_4, s)$: The conditions of case IV are the followings:

$$\begin{cases} w_s \leq \frac{w_s + \eta_s\, L_s\, u_s}{L_s} \\ w_s \leq d_{min,s} \end{cases} \Longrightarrow \begin{cases} w_s \leq \frac{\eta_s\, L_s\, u}{L_s - 1} \quad (lim_2) \\ w_s \leq d_{min,s} \end{cases}$$

We need compare $d_{min,s}$ and $\frac{\eta_s\, L_s\, u_s}{L_s - 1}$ in order to verify $lim_3$.

$$\begin{aligned} d_{min,s} - \frac{\eta_s\, L_s\, u_s}{L_s - 1} &= \frac{(L_s - 1)\, d_{min,s} - \eta_s\, L_s\, u_s}{L_s - 1} = \frac{L_s\,(d_{min,s} - \eta_s\, u_s) - d_{min,s}}{L_s - 1} \\ &\geq 0 \quad \text{(because } L_s > 1 \text{ and } d_{min,s} \leq L_s(d_{min,s} - \eta_s\, u_s)\text{)} \qquad \text{(verified ✓)} \end{aligned}$$

Thus $lim_3$'s definition is correct and there no analytic definition for $lim_4$. We can suppose that it can be equal to 0 since $w_s$ can only be positive (or equal to 0). So we can suppose that $\boxed{lim_4 = 0}$ even thought attaining that limit means that the download might be interrupted.

[6] The online solver is available at: http://www.wolframalpha.com

We need now to calculate $\lim_{w_s \to 0} Gain(w_s, s)$ that will be considered the upper bound of the gain values

$$\lim_{w_s \to 0} (Gain(w_s, s)) \overset{caseIV}{=} \lim_{w_s \to 0} \left(1 - \frac{1}{L_s} - \frac{\alpha_{bt}.w_s}{F.L_s}\right) = 1 - \frac{1}{L_s}$$

**- Resolution of the equation** $Gain(\overset{??}{w_s}, S_f, F) = \tau,\ \forall \tau \in \left[1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s}, 1 - \frac{1}{L_s}\right[$

We have: $Gain(\overset{??}{w_s}, s) = \tau$ and $\tau \in \left[1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s}, 1 - \frac{1}{L_s}\right[$

This means that $Gain(\overset{??}{w_s}, s) \in \left[1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s}, 1 - \frac{1}{L_s}\right[$ which corresponds to the formula of the gain related to case IV. Let's try now to invert that formula in order to get an estimation of $w_s$:

$$\begin{aligned} Gain(\overset{??}{w_s}, s) = \tau \quad &\overset{\text{case IV}}{\Longleftrightarrow} \quad 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,w_s}{F_s\,L_s} = \tau \Longleftrightarrow \frac{\alpha_{bt}\,w_s}{F_s\,L_s} = 1 - \frac{1}{L_s} - \tau \\ &\Longleftrightarrow \quad w_s = \frac{F_s\,L_s}{\alpha_{bt}} \left(\frac{L_s\,(1-\tau) - 1}{L_s}\right) \Longleftrightarrow w_s = \frac{F_s\,\left[L_s\,(1-\tau) - 1\right]}{\alpha_{bt}} \end{aligned}$$

We can then conclude that:

$$\boxed{\forall \tau \in \left[1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta\,u}{(L_s - 1)\,F_s}, 1 - \frac{1}{L_s}\right[, \quad \left(Gain(\overset{??}{w_s}, s) = \tau\right) \Rightarrow w_s = \frac{F_s\,\left[L_s\,(1-\tau) - 1\right]}{\alpha_{bt}}}$$

**General conclusion for Case A:** $w_s = f(\tau)$

The equation: $Gain(\overset{??}{w_s}, s) = \tau$ has the following solution.

$$w_s^{bt} = \begin{cases} L_s\,d_{min,s}, & \forall \tau \in \left]-\infty, -\frac{\alpha_{bt} d_{min,s}}{F_s}\right] \\ \frac{(1-\tau) F_s\,L_s\,d_{min,s}}{F_s + d_{min,s}\,\alpha_{bt}}, & \forall \tau \in \left[-\frac{\alpha_{bt}\,d_{min,s}}{F_s}, \frac{\eta_s\,u_s}{d_{min,s}} - \frac{\alpha_{bt}\,(d_{min,s} - \eta_s\,u_s)}{F_s}\right] \\ \frac{\sqrt{a^2b^2 - 2abc + 4ab + c^2} - ab - c}{2\,b}, & \forall \tau \in \left[\frac{\eta_s\,u_s}{d_{min,s}} - \frac{\alpha_{bt}(d_{min,s} - \eta_s\,u_s)}{F_s}, 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s}\right] \\ \frac{F_s\,[L_s\,(1-\tau) - 1]}{\alpha_{bt}}, & \forall \tau \in \left[1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{(L_s - 1)\,F_s}, 1 - \frac{1}{L_s}\right[ \\ \nexists, & \forall \tau \in \left[1 - \frac{1}{L_s}, +\infty\right[ \end{cases}$$

Where:

$$a = \eta_s\,L_s\,u_s, \quad b = \frac{\alpha_{bt}}{F_s\,L_s} \text{ and } c = \tau$$

*Appendix A.2. Case B:* $(L_s - 1)\,d_{min,s} \leq L_s\,\eta_s\,u_s$

In this case, the intervals and the delimiters are not as evident as in Case A. So let's start first by studying the limits in the gain cases and try to locate the cases based on their order.

*Appendix A.2.1. Fixing the interval delimiters*

For each interval we part from its constraints and determine the range of values pf $w_s$ in each of these intervals. The goal is to verify if theses intervals are disjoint and do not superimpose.

1. $lim_1$: **(case I)**

$$\begin{cases} d_{min,s} \leq \frac{w_s}{L_s} \\ d_{min,s} \leq \frac{w_s + \eta_s\,L_s\,u_s}{L_s} \\ d_{min,s} \leq w_s \end{cases} \Rightarrow d_{min,s} \leq \frac{w_s}{L_s} \Rightarrow \boxed{w_s \geq lim_1 = L_s\,d_{min,s}}$$

2. $lim_2$: **(case II)**

$$\begin{cases} d_{min,s} \geq \frac{w_s}{L_s} \\ d_{min,s} \leq \frac{w_s + \eta_s L_s u_s}{L_s} \\ d_{min,s} \leq w_s \end{cases} \implies \begin{cases} w_s \leq L_s\, d_{min,s} = lim_1 \\ w_s \geq d_{min,s} \\ w_s \geq L_s(d_{min,s} - \eta_s\, u_s) \end{cases} \xrightarrow{\text{Case B}} \begin{cases} \boxed{w_s \leq lim_1} \\ \boxed{w_s \geq d_{min,s} = lim_2} \end{cases}$$

Comparing $lim_1$ and $lim_2$: We know that $L_s > 1$ and $d_{min,s} > 0$ Thus $L_s\, d_{min,s} > d_{min,s} \Rightarrow \boxed{lim_1 > lim_2}$

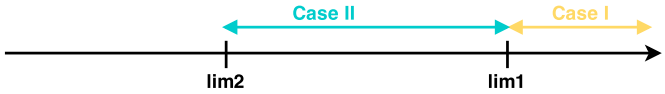



Figure A.15: Delimiter $lim_2$ and interval for Case II

3. $lim_3$ and $\lim_{2p}$**(case III)**

$$\begin{cases} \frac{w_s + \eta_s L_s u_s}{L_s} \leq d_{min,s} \\ \frac{w_s + \eta_s L_s u_s}{L_s} \leq w_s \end{cases} \implies \begin{cases} \boxed{w_s \leq L_s\,(d_{min,s} - \eta_s\, u_s) = lim_{2p}} \\ \boxed{w_s \geq \frac{\eta\, L_s\, u_s}{L_s - 1} = lim_3} \end{cases}$$

*Comparing $lim_{2p}$ and $lim_2$:.* Based on the conditions of Case B, we already know that $lim_{2p} \leq lim_2$

*Comparing $lim_3$ and $lim_2$:.*

$$\begin{aligned} lim_3 - lim_2 &= \frac{\eta_s L_s u_s}{L_s - 1} - d_{min,s} = \frac{\eta_s L_s u_s - (L_s - 1)\, d_{min,s}}{L_s - 1} = \frac{1}{L_s - 1}\,(d_{min,s} - L_s\,(d_{min,s} - \eta_s\, u_s)) \\ &\geq 0 \quad \text{(because } L_s > 1 \text{ and } d_{min,s} \geq L_s(d_{min,s} - \eta_s\, u_s)) \Rightarrow \boxed{lim_3 \geq lim_2} \end{aligned}$$

*Comparing $lim_3$ and $lim_1$:.*

$$\begin{aligned} lim_3 - lim_1 &= \frac{\eta_s L_s u_s}{L_s - 1} - L_s\, d_{min,s} = \frac{\eta_s L_s u_s - L_s(L_s - 1).d_{min,s}}{L_s - 1} \frac{\left(L_s\,(d_{min,s} - \eta_s\, u_s) - L_s^2\, d_{min,s}\right)}{L_s - 1} \\ \text{We have} \quad & L_s \geq 1 \Rightarrow L_s^2 \geq 1 \;\&\; d_{min,s} \geq 0 \Rightarrow L_s^2.d_{min,s} \geq d_{min,s} \\ \text{and since} \quad & \begin{cases} d_{min,s} \geq L_s(d_{min,s} - \eta_s\, u_s) \\ L_s^2\, d_{min,s} \geq d_{min,s} \end{cases} \Rightarrow L_s^2\, d_{min,s} \geq L_s(d_{min,s} - \eta_s\, u_s) \\ \text{Thus} \quad lim_3 - lim_1 &\leq 0 \Rightarrow \boxed{lim_1 \geq lim_3} \end{aligned}$$

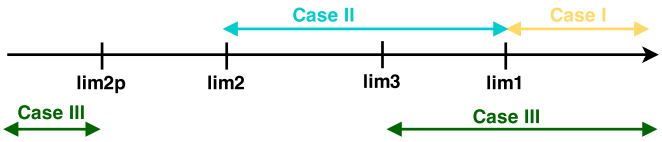



Figure A.16: Delimiter $lim_{2p}$ and $lim_3$ and interval for Case III

4. $lim_4$: **(case IV)**

$$\begin{cases} w_s \leq \frac{w_s + \eta_s L_s u_s}{L_s} \\ w_s \leq d_{min,s} \end{cases} \Rightarrow \begin{cases} w_s \leq \frac{\eta_s L_s u_s}{L_s - 1} \\ w_s \leq d_{min,s} \end{cases} \Rightarrow \begin{cases} w_s \leq lim_3 \\ w_s \leq lim_2 \end{cases} \xrightarrow{lim_3 \geq lim_2} \boxed{w_s \leq lim_2}$$

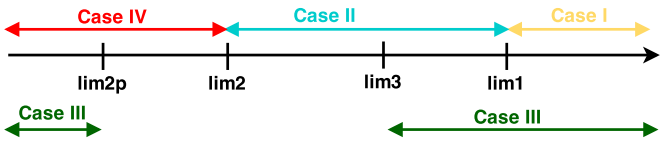



Figure A.17: Interval for Case IV

*Appendix A.2.2. Interpretation of the superimposed cases*

Based on figure A.17, we can distinguish 3 intervals where there are superimposed cases which are:

- **Interval 1:** $\left]-\infty, lim_{2p}\right] = ]-\infty, L_s\,(d_{min,s} - \eta_s\,u_s)]$ : superimposition of case IV and case III
- **Interval 2:** $[lim_3, lim_1] = \left[\frac{\eta_s\,L_s\,u_s}{L_s-1}, L_s\,d_{min,s}\right]$ : superimposition of case II and case III
- **Interval 3:** $[lim_1, +\infty[ = [L_s\,d_{min,s}, +\infty[$ : superimposition of case I and case III

Let's check interval by interval the implications of such a superimposition. For each interval we will define the new constraints resulting from the intersection of the corresponding cases and define accordingly the gain expression. The goal is to demonstrate that the solution will be the same for both cases.

i. **Interval 1:**

$$\begin{cases} \text{Case IV}: \begin{cases} w_s \le d_{min,s} \\ w_s \le \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \end{cases} \\ \text{Case III}: \begin{cases} \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le d_{min,s} \\ \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le w_s \end{cases} \end{cases} \quad \Rightarrow \quad w_s = \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le d_{min,s} \Rightarrow \boxed{w_s = \frac{L_s\,\eta_s\,u_s}{L_s-1}}$$

*Verification of the gain expression in both cases:.* Let's now verify that $Gain(w_s, s)$ has the same expression in both cases IV and III when $w_s = \frac{\eta_s\,L_s\,u_s}{L_s-1}$.

$$\begin{aligned} Gain\left(\frac{L_s\,\eta_s\,u_s}{L_s-1}, s\right) &\overset{\text{case IV}}{=} 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\frac{\eta_s\,L_s\,u_s}{L_s-1}}{F_s\,L_s} = 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,\eta_s\,u_s}{F_s\,(L_s-1)} \\ Gain\left(\frac{\eta_s\,L_s\,u_s}{L_s-1}, s\right) &\overset{\text{case III}}{=} 1 - \frac{\frac{\eta_s\,L_s\,u_s}{L_s-1}}{\frac{\eta_s\,L_s\,u_s}{L_s-1} + \eta_s\,L_s\,u_s} - \frac{\alpha_{bt}.\frac{\eta_s\,L_s\,u_s}{L_s-1}}{F_s\,L_s} \\ &= 1 - \frac{1}{1+(L_s-1)} - \frac{\alpha_{bt}.\eta_s\,u_s}{F_s\,(L_s-1)} = 1 - \frac{1}{L_s} - \frac{\alpha_{bt}.\eta_s\,u_s}{F_s\,(L_s-1)} \end{aligned}$$

For this interval, both cases have the same expression. Thus, we can just consider just one of the cases instead of working with both. The best choice seems to be **case IV** since it has a simpler formulas.

ii. **Interval 2:**

$$\begin{cases} \text{Case II}: \begin{cases} d_{min,s} \ge \frac{w_s}{L_s} \\ d_{min,s} \le \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \\ d_{min,s} \le w_s \end{cases} \\ \text{Case III}: \begin{cases} \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le d_{min,s} \\ \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le w_s \end{cases} \end{cases} \quad \Rightarrow \quad \frac{w_s}{L_s} \le d_{min,s} = \frac{w_s+\eta_s\,L_s\,u_s}{L_s} \le w_s$$

$$\Rightarrow \quad \boxed{w_s = L_s\,(d_{min,s} - \eta_s\,u_s)}$$

*Verification of the gain expression in both cases:.* Let's now verify that $Gain(w_s, s)$ has the same expression in both cases II and III when $w_s = L_s\,(d_{min,s} - \eta_s\,u_s)$.

$$\begin{aligned} Gain(L_s\,d_{min,s}, s) &\overset{\text{case II}}{=} 1 - \frac{L_s\,(d_{min,s} - \eta_s\,u_s)}{L_s d_{min,s}} - \frac{\alpha_{bt}\,L_s\,(d_{min,s} - \eta_s\,u_s)}{F_s\,L_s} \\ Gain(L_s\,d_{min,s}, s) &\overset{\text{case III}}{=} 1 - \frac{L_s\,(d_{min,s} - \eta_s\,u_s)}{L_s\,(d_{min,s} - \eta_s\,u_s) + \eta_s\,L_s\,u_s} - \frac{\alpha_{bt}\,L_s\,(d_{min,s} - \eta_s\,u_s)}{F_s\,L_s} \\ &= 1 - \frac{L_s\,(d_{min,s} - \eta_s\,u_s)}{L_s\,d_{min,s}} - \frac{\alpha_{bt}\,L_s\,(d_{min,s} - \eta_s\,u_s)}{F_s\,L_s} \end{aligned}$$

For this interval, both cases have the same expression. Thus, we can just consider just one of the cases instead of working with both. The best choice seems to be **case II** since it has a simpler formulas.

iii. **Interval 3:**

*Constraints definition:.*

$$\begin{cases} \text{Case I}: \begin{cases} d_{min,s} \le \frac{w_s}{L_s} \\ d_{min,s} \le \frac{w_s + \eta_s\,L_s\,u_s}{L_s} \\ d_{min,s} \le w_s \end{cases} \\ \text{Case III}: \begin{cases} \frac{w_s + \eta_s\,L_s\,u_s}{L_s} \le d_{min,s} \\ \frac{w_s + \eta_s\,L_s\,u_s}{L_s} \le w_s \end{cases} \end{cases} \quad \overset{L_s > 1}{\Longrightarrow} \quad d_{min,s} = \frac{w_s + \eta_s\,L_s\,u_s}{L_s} \le \frac{w_s}{L_s}$$

$$\overset{\substack{L_s \ge 1 \\ \eta_s\,u_s \ge 0}}{\Longrightarrow} \quad \begin{cases} d_{min,s} = \frac{w_s + \eta_s\,L_s\,u_s}{L_s} \\ \eta_s\,L_s\,u_s = 0 \end{cases} \quad \Rightarrow \boxed{w_s = L_s\,d_{min,s}}$$

*Verification of the gain expression in both cases:.* Let's now erify that $Gain(w_s, s)$ has the same expression in both cases II and III when $w_s = L_s\,d_{min,s}$.

$$\begin{aligned} Gain(L_s.d_{min,s}, s) &\overset{\text{case III}}{=} 1 - \frac{L_s\,d_{min,s}}{L_s\,d_{min,s} + \eta_s\,L_s\,u_s} - \frac{\alpha_{bt}\,L_s\,d_{min,s}}{F_s\,L_s} \overset{\eta_s\,u_s = 0}{=} 1 - 1 - \frac{\alpha_{bt}\,d_{min,s}}{F_s} \\ &= -\frac{\alpha_{bt}\,d_{min,s}}{F_s} \overset{\text{case I}}{=} Gain(L_s\,d_{min,s}, s) \end{aligned}$$

For this interval, both cases have the same expression. Thus, we can just consider just one of the cases instead of working with both. The best choice seems to be **case I** since it has a simpler formulas.

After the verification of the superimposed intervals, we can just consider the following intervals (figure A.18)

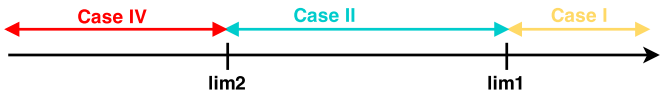



Figure A.18: Final interval to be considered for Case B

*Appendix A.2.3. Inverting the gain*

The goal of this section is to derive a potential equation of the gain ratio as a function of the seed's upload speed. The general shape of that function is given in figure A.14. We have already defined the intervals delimiters ($lim_1$ and $lim_2$) and $lim_3$ is the lower bound of $w_s$ (that is 0). We just need to verify the corresponding gain values ($Gain(lim_1, s)$ , $Gain(lim_2, s)$ and $Gain(lim_3, s)$) that should be equal to the ones already defined in Case A.

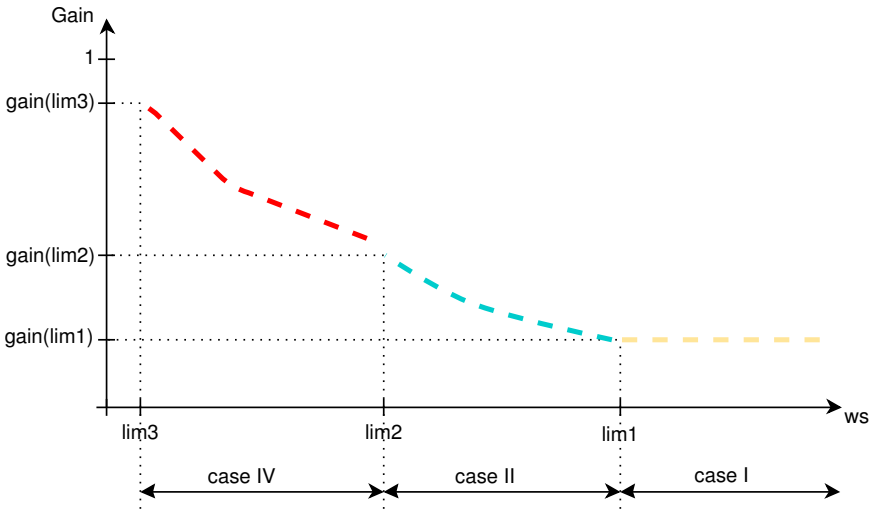



Figure A.19: General shape of the gain ratio as a function of the upload speed of the seed when $\max(d_{min,s}, L_s\,(d_{min,s} - \eta_s\,u_s)) = d_{min,s}$

1. $lim_1$ and $Gain(lim_1, s)$: We have already found that $\boxed{lim_1 = L_s\,d_{min,s}}$. The corresponding gain has been already calculated and verified for both case I and case II. It is equal to: $\boxed{Gain(lim_1, s) = -\frac{\alpha_{bt}\,d_{min,s}}{F_s}}$.

   **- Resolution of the equation** $Gain(\overset{??}{w_s}, s) = \tau,\ \forall \tau \in \left]-\infty, -\frac{\alpha_{bt}\,d_{min,s}}{F_s}\right]$

   Since $\tau \in \left]-\infty, -\frac{\alpha_{bt}\,d_{min,s}}{F_s}\right]$ and $Gain(\overset{??}{w_s}, s) = \tau$.

   This leads to $Gain(\overset{??}{w_s}, s) \in \left]-\infty, -\frac{\alpha_{bt}\,d_{min,s}}{F_s}\right]$, this means that $w_s \geq L_s\,d_{min,s}$.

   Thus, $\forall \tau \in \left]-\infty, -\frac{\alpha_{bt}\,d_{min,s}}{F_s}\right]$, the optimal bandwidth that should be allocated to the swarm without violating the constraint $(Gain(w_s, s) \geq \tau)$ is $w_s = L_s\,d_{min,s}$

   $$\boxed{\forall \tau \in \left]-\infty, -\frac{\alpha_{bt}\,d_{min,s}}{F}\right],\quad \left(Gain(\overset{??}{w_s}, s) = \tau\right) \Rightarrow (w_s = L_s\,d_{min,s})}$$

2. $lim_2$ and $Gain(lim_2, s)$: We have already found that $\boxed{lim_2 = d_{min,s}}$. Now, we need to calculate $Gain(d_{min,s}, s)$ for both cases II and IV and we should obtain the same value to which we will refer to as: $Gain(lim_2, s)$

   $$Gain(d_{min,s}, s) \overset{\text{case II}}{=} 1 - \frac{d_{min,s}}{L_s\,d_{min,s}} - \frac{\alpha_{bt}\,d_{min,s}}{F_s\,L_s} = 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,d_{min,s}}{F_s\,L_s}$$
   $$\overset{\text{case IV}}{=} Gain(d_{min,s}, s)$$

   We obtain finally that: $\boxed{Gain(lim_2, s) = 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,d_{min,s}}{F_s\,L_s}}$.

   **- Resolution of the equation** $Gain(\overset{??}{w_s}, s) = \tau,\ \forall \tau \in \left[-\frac{\alpha_{bt}\,d_{min,s}}{F_s}, 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,d_{min,s}}{F_s\,L_s}\right]$

   Since $\tau \in \left[-\frac{\alpha_{bt}\,d_{min,s}}{F_s}, 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,d_{min,s}}{F_s\,L_s}\right]$ and $Gain(\overset{??}{w_s}, s) = \tau$.

   This leads to $Gain(\overset{??}{w_s}, s) \in \left[-\frac{\alpha_{bt}\,d_{min,s}}{F_s}, 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\,d_{min,s}}{F_s\,L_s}\right]$, which corresponds to the formula of the gain related to case II.

Let's try now to invert that formula in order to get an estimation of $w_s$

$$Gain(\overset{??}{w_s}, s) = \tau \quad \overset{\text{case II}}{\Longleftrightarrow} \quad 1 - \frac{w_s}{L_s\, d_{min,s}} - \frac{\alpha_{bt}\, w_s}{F_s\, L_s} = \tau \Longleftrightarrow 1 - \tau = w_s \left( \frac{1}{L_s\, d_{min,s}} + \frac{\alpha_{bt}}{F_s\, L_s} \right)$$
$$\Longleftrightarrow \quad 1 - \tau = w_s \left( \frac{F_s + d_{min,s}\, \alpha_{bt}}{F_s\, L_s\, d_{min,s}} \right) \Longleftrightarrow w_s = \frac{(1-\tau) F_s\, L_s\, d_{min,s}}{F_s + d_{min,s}\, \alpha_{bt}}$$

We can then conclude that:

$$\boxed{\forall \tau \in \left[ -\frac{\alpha_{bt}\, d_{min,s}}{F_s}, 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s} \right], \left( Gain(\overset{??}{w_s}, s) = \tau \right) \Rightarrow w_s = \frac{(1-\tau)\, F_s\, L_s\, d_{min,s}}{F_s + d_{min,s}\, \alpha_{bt}}}$$

3. $lim_3$ and $Gain(lim_3, s)$: the definition of these parameters is very similar to the one done for $lim_4$ and $Gain(lim_4, s)$ in Case A.
Since $w_s$ can only be positive (or equal to 0), we can suppose that $\boxed{lim_3 = 0}$ even thought attaining that limit means that the download might be interrupted.
We need now to calculate $\lim\limits_{w_s \to lim_3} Gain(w_s, s)$ that will be considered as the upper bound of the gain values:

$$\lim_{w_s \to lim_3} (Gain(w_s, s)) \overset{caseIV}{=} \lim_{w_s \to 0} \left( 1 - \frac{1}{L_s} - \frac{\alpha_{bt}.w_s}{F_s\, L_s} \right) = 1 - \frac{1}{L_s} = Gain(lim_3, s)$$

**- Resolution of the equation** $Gain(\overset{??}{w_s}, s) = \tau,\ \forall \tau \in \left[ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}, 1 - \frac{1}{L_s} \right[$
We have: $Gain(\overset{??}{w_s}, s) = \tau$ and $\tau \in \left[ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}, 1 - \frac{1}{L_s} \right[$
This means that $Gain(\overset{??}{w_s}, s) \in \left[ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}, 1 - \frac{1}{L_s} \right[$ which corresponds to the formula of the gain related to case IV.
Let's try now to invert that formula in order to get an estimation of $w_s$

$$Gain(\overset{??}{w_s}, s) = \tau \quad \overset{\text{case IV}}{\Longleftrightarrow} 1 - \frac{1}{L} - \frac{\alpha_{bt}\, w_s}{F_s\, L_s} = \tau \Longleftrightarrow \frac{\alpha_{bt}\, w_s}{F_s\, L_s} = 1 - \frac{1}{L_s} - \tau$$
$$\Longleftrightarrow w_s = \frac{F_s\, L_s}{\alpha_{bt}} \left( \frac{L_s\,(1-\tau) - 1}{L_s} \right) \Longleftrightarrow w_s = \frac{F_s\, [L_s(1-\tau) - 1]}{\alpha_{bt}}$$

We can then conclude that:

$$\boxed{\forall \tau \in \left[ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}, 1 - \frac{1}{L_s} \right[, \left( Gain(\overset{??}{w_s}, s) = \tau \right) \Rightarrow w_s = \frac{F_s\ (L_s\,(1-\tau) - 1)}{\alpha_{bt}}} \tag{A.1}$$

**General conclusion for Case B:** $w_s = f(\tau)$
The equation: $Gain(\overset{??}{w_s}, s) = \tau$ has the following solution.

$$w_s^{bt} = \begin{cases} L_s\, d_{min,s}, & \forall \tau \in \left] -\infty, -\frac{\alpha_{bt}\, d_{min,s}}{F_s} \right] \\ \frac{(1-\tau) F_s\, L_s . d_{min,s}}{F_s + d_{min,s}\, \alpha_{bt}}, & \forall \tau \in \left[ -\frac{\alpha_{bt}\, d_{min,s}}{F_s}, 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s} \right] \\ \frac{F_s\, [L_s(1-\tau) - 1]}{\alpha_{bt}}, & \forall \tau \in \left[ 1 - \frac{1}{L_s} - \frac{\alpha_{bt}\, d_{min,s}}{F_s\, L_s}, 1 - \frac{1}{L_s} \right[ \\ \nexists, & \forall \tau \in \left[ 1 - \frac{1}{L_s}, +\infty \right[ \end{cases}$$