Energy-Efficient P2P File Sharing for Residential BitTorrent Users

Ilaria Giannetti
Dept. of Information Engineering
University of Pisa, Italy
ilaria.giannetti@iet.unipi.it

Giuseppe Anastasi
Dept. of Information Engineering
University of Pisa, Italy
giuseppe.anastasi@iet.unipi.it

Marco Conti
IIT-CNR
National Research Council, Italy
marco.conti@iit.cnr.it

***Abstract* —BitTorrent is currently a very popular protocol for P2P file sharing, and most of BitTorrent users access the Internet from residential networks. In this paper we compare the legacy BitTorrent protocol with *EE-BitTorrent*, a proxy-based version recently proposed for energy efficiency, in a residential scenario. We show that the performance achieved by users is strongly influenced by the uplink throughput allowed by the access network. When the available uplink rate is low, the legacy BitTorrent protocol performs poorly and EE-BitTorrent outperforms it, in terms of average download time and energy consumption at the user's PC. The opposite occurs when the uplink rate is good. Motivated by these results, we designed and implemented *AdaBT*, an adaptive algorithm that dynamically selects the most efficient BitTorrent option (i.e., legacy or proxy-based), depending on the operating conditions experienced by the user. Our experimental results show that AdaBT is able to reduce significantly the download time provided by either the legacy BitTorrent or EE-BitTorrent.**

Keywords: P2P File Sharing; BitTorrent; Energy Efficiency; Green Internet.

I. INTRODUCTION

In developed countries, the overall energy consumption due to the Internet accounts for approximately 2-3% of the overall energy consumption. Although this percentage is not so high, its absolute value is nevertheless remarkable (about 74 TWh only in USA [1]). More important, it is estimated that about one third of the overall energy consumed by the Internet is wasted due to a non appropriate usage of network devices and user equipments. Thus, a significant energy saving could be achieved through appropriate power management strategies. This has stimulated the interest and efforts of the research community. However, most of the on-going research projects and activities are aimed at reducing the energy consumption in the Internet core (i.e., at routers) and at data centers [2]. Less attention has been devoted to reduce the power consumption at user premises, even if the total amount of energy consumed by *personal computers* (PCs) accounts for the major fraction of the overall energy consumption. This is mainly due to the large number of PCs used in homes and offices. In addition, PCs are often left on by users, especially at home, to perform networking activities like, for example, Peer-to-Peer (P2P) file-sharing.

P2P file sharing applications accounts for the largest fraction of the Internet traffic today, ranging from 40% to 73% [3]. Among current P2P platforms, BitTorrent [4] is probably the most popular protocol, accounting for 50-75% of the overall P2P traffic. These figures show that making P2P (and, particularly, BitTorrent) applications more energy efficient can result in significant energy savings. To this end, Blackburn and Christensen have proposed *Green BitTorrent* [5], a modified version of BitTorrent where peers that have completed their download process put their PC in sleep mode for saving energy. When the number of connected peers is below a pre-defined threshold, a peer can explicitly wake up another sleeping peer. Green BitTorrent assumes that PCs are equipped with *Wake-on-LAN* (WoL) network interfaces which, however, are not available for many access network technologies. A proxy-based version of BitTorrent has been recently proposed in [6]. In this solution, called *EE-BitTorrent*, the software running on the user's PC delegates the download process to a proxy and switches off the PC. Later, the user reconnects and fetches the file from the proxy. A similar solution, aimed at reducing the power consumption associated with the P2P sharing of unpopular files, has been proposed in [7]. Finally, proxy-based solutions have been also considered for energy-efficient file sharing from a mobile device [8, 9].

According to the experimental results in [6], EE-BitTorrent can save a large fraction of the energy consumed by legacy BitTorrent, without increasing the average download time. The analysis in [6] focused on a departmental scenario where the BitTorrent proxy is located on the same high-speed Local Area Network (LAN) of PCs (i.e., a 100-Mbps Ethernet). However, most of BitTorrent users access the BitTorrent overlay from residential access networks (e.g., ADSL), whose available bandwidth is much lower than that of a departmental LAN. Thus, it is important to investigate whether EE-BitTorrent is more convenient than the legacy protocol, in terms of energy consumption and average download time, also in a residential environment. In this paper we first extend the analysis in [6] by focusing on residential BitTorrent users. As expected, the time required for downloading a file (and the corresponding energy consumption) with the legacy BitTorrent protocol is strongly influenced by the available uplink rate (due to the Tit-for-Tat strategy). When the available uplink rate is low, the legacy BitTorrent protocol performs poorly and EE-BitTorrent outperforms it in terms of average download time and energy consumption at the user's PC. The opposite occurs when the available uplink rate is good. Our experimental analysis has shown that, in fact, the available uplink rate (and, hence, the download time and energy consumption), may be extremely different in different locations and, for the same location, it may also vary from time to time. Motivated by these experimental results, in the second part of the paper, we define *Adaptive BitTorrent (AdaBT),* an adaptive algorithm that is able to dynamically select the most convenient option (between legacy BitTorrent and EE-BitTorrent) depending

on the actual operating conditions available at download time. We performed an experimental evaluation of AdaBT, which shows that the proposed algorithm can significantly reduce the download time provided by both the legacy and proxy-based BitTorrent version.

The rest of the paper is organized as follows. Section II describes the legacy BitTorrent protocol and EE- BitTorrent protocol. Section III presents the experimental results obtained with EE-BitTorrent in residential scenarios. Session IV describes the AdaBT algorithm, while Section V presents its experimental evaluation. Section VI concludes the paper.

II. BITTORRENT PROTOCOL

A. Legacy BitTorrent

In this section we provide a brief description of the BitTorrent protocol (the reader can refer to [4] and [10] for details). BitTorrent implements an unstructured overlay network customized for file sharing. According to the protocol terminology, nodes of the overlay are called *peers*, and the set of peers involved in the distribution of a file is referred to as *torrent* or *swarm*. Each peer downloads the desired file, in *chunks*, from a multitude of other peers instead of fetching it from a single server (as in the conventional client-server model). While downloading missing chunks, peers upload to other peers in the same torrent the chunks they have already obtained. For each torrent there is a *tracker*, i.e., a node that constantly tracks which peers are involved in the torrent. A peer that wants to join a torrent must register with the tracker and, then, it must periodically inform the tracker that it is still in the torrent.

Figure 1 shows the different phases in the download process. As a preliminary step, a peer interested in downloading a file has to get the corresponding *torrent* file. Torrents are very small files, typically hosted by conventional Web servers (torrent servers), and contain the name of the torrent tracker. Then, the peer contacts the tracker and receives a random list of peers belonging to the torrent. It can thus open a TCP connection with a number of them and start exchanging chunks of the file. At any given time, the peer will be in contact with a set of other peers, referred to as *neighbors*. The set of neighbors changes dynamically as peers join and leave the torrent. In addition, each peer preferentially selects, for downloading chunks, those peers from which it can achieve the highest download rate (see below). Finally, every 30 seconds the peer randomly selects a new peer from the list, as a way to discover new neighbors and allow new peers in the torrent to start-up. At any given time, the peer has a subset of chunks composing the file. To figure out where missing chunks can be downloaded from, it periodically asks each of its neighbors for the list of chunks they have. Then, it uses the *Rarest First* strategy to decide the missing chunks to be requested first, i.e., it gives priority to chunks that are less spread in the torrent. Finally, to decide which requests from other peers has to be served, the peer uses the *Tit-for-Tat* (*TAT*) strategy, i.e., it gives priority to peers from which it is downloading data at the highest rate. Specifically, it measures the data rate it is achieving from each of its current neighbors and, then, selects the four neighbors with the highest data rate.

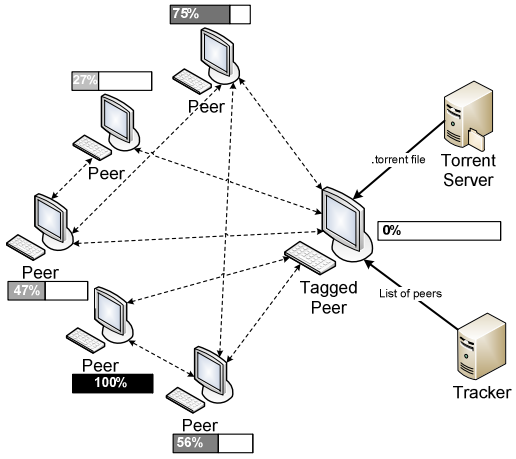

Figure 1. File Distribution Process.

B. EE-BitTorrent

EE-BitTorrent is a proxy-based version of BitTorrent where a number of peers is served by a *BitTorrent proxy*. These peers are not directly involved in the torrent, as they delegate the task to the associated proxy, which downloads the file on behalf of them. The proxy participates to the BitTorrent overlay network, just like any other regular peer, and takes care of the overall process. Therefore, the user's PC can be switched off during the download phase. The file will be transferred from the proxy to the user's PC later, when the user reconnects.

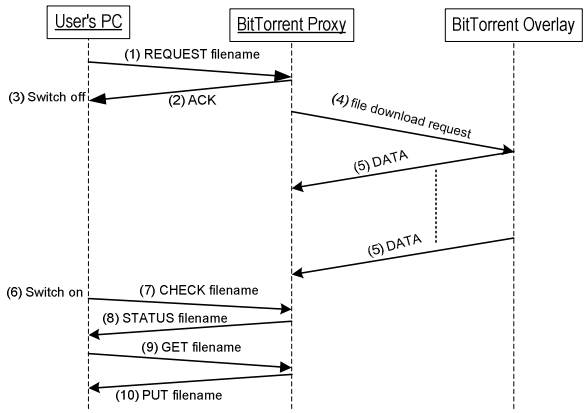

Figure 2. Action performed in the EE-BitTorrent Protocol.

Figure 2 shows the actions performed by the various actors of EE-BitTorrent. Upon receiving a request from the user, the software running on the user's PC contacts the proxy and requests the desired file (step 1). The proxy acknowledges the request (step 2). If the requested file is already available in the local cache of the proxy, it is immediately transferred to the user. Otherwise, the proxy starts downloading the requested file from the BitTorrent overlay network, acting as a regular peer and following the

legacy BitTorrent protocol (steps 4-5). The user's PC can be switched off just after receiving the acknowledgement from the proxy (step 3). Later, when the user reconnects (step 6), he/she can check the status of the download process at the proxy (steps 7-8). If the file is completely available, it can be transferred from the proxy to the user's PC (steps 9-10).

III. EXPERMENTAL ANALYSIS WITH RESIDENTIAL USERS

In this section we investigate the performance of EE-BitTorrent in a residential scenario. To this end, we set up an experimental testbed and performed experiments with both the legacy and proxy-based BitTorrent protocol. However, before describing our experimental setup, it is worthwhile introducing the performance metrics considered in our analysis. Let us consider a set of *N* BitTorrent users. Let $D_L(i)$ and $D_P(i)$, $i = 1,2,...,N$, denote the total activity time required by the generic user's PC for downloading a file with the legacy and proxy-based approach, respectively. In both cases, this time is proportional to the energy consumed by the user for achieving a copy of the desired file. Hence, the energy saving introduced by EE-BitTorrent, with respect to the legacy approach, for a generic user *i*, can be expressed as

$$S_{usr}^{i}(N) = 1 - \frac{D_P(i)}{D_L(i)} = 1 - \frac{D_r(i) + D_t(i)}{D_L(i)} \approx 1 - \frac{D_t(i)}{D_L(i)} \quad (1)$$

where $D_r(i)$ is the time for sending the request to the proxy, and $D_t(i)$ is the time for transferring the file from the proxy to the user's PC. The last passage in Equation (1) is because $D_r(i) << D_t(i)$, for any *i*.

In order to make the comparison fair, we need to take into account also the energy consumed by the proxy, when considering EE-BitTorrent. Thus, assuming that the proxy and PCs have similar power consumptions, the energy saving introduced by EE-BitTorrent, with respect to the legacy approach, can be expressed as

$$S_{sys}(N) = 1 - \frac{2\sum_{i=1}^{N} D_r(i) + \sum_{i=1}^{N} D_o(i) + 2\sum_{i=1}^{N} D_t(i)}{\sum_{i=1}^{N} D_L(i)} \quad (2)$$

where $D_o(i)$ indicates the time spent by the proxy to download the file requested by user *i* from the BitTorrent overlay network. As above, since $D_r(i) << D_t(i)$ for any *i*, Equation (2) can be re-written as

$$S_{sys}(N) = 1 - \frac{\sum_{i=1}^{N} D_o(i) + 2\sum_{i=1}^{N} D_t(i)}{\sum_{i=1}^{N} D_L(i)} \quad (3)$$

A. Testbed Description

In order to measure the delay components introduced in the previous section (i.e., $D_L(i)$, $D_t(i)$ and $D_o(i)$ for $i = 1,2,...,N$), and evaluate Equations (1) and (3), we used an experimental testbed consisting of N PCs connected to the Internet through ADSL links. When using EE-BitTorrent, the proxy was located at the University of Pisa and was connected to the Internet through a 100 Mbps Ethernet LAN. The residential PCs were located in different locations, many kilometers apart from the proxy (in some cases in a different city). We considered ADSL links with 8.0 Mbps downlink and 512 Kbps uplink (nominal) bit rate. In our experiments, we assumed that the user is interested in downloading one of the following files, which represent three typical file categories, frequently downloaded by BitTorrent users

- CD audio file (MP3 format, ~100 MB)
- Episode of a TV series (avi format, ~350 MB)
- Ubuntu 10.10 distribution (iso format, ~4 GB)

In all the experiments we considered the same number of seeds in the torrent (i.e., about 500). The experiments with legacy and proxy-based BitTorrent were interleaved, so as to experience similar operating conditions (i.e., network traffic, number of peers in the torrent, etc.). Finally, to increase the accuracy of our results, we replicated the same experiment several times, in different days and hours. For each measure we will show the average value (calculated over all the replications) and standard deviation.

B. Experimental Results

We started our analysis considering a single BitTorrent user located at a location A. In Figure 3, as well as in all subsequent figures, we show the average time the user's PC must remain active to get a copy of the file with the legacy (i.e., D_L) and the proxy-based approach (i.e., $D_r + D_t \approx D_t$). For EE-BitTorrent, the average time the proxy remains active is also shown (i.e., $D_r + D_o + D_t \approx D_o + D_t$). The difference between the values related to the proxy and PC, in the proxy-based version, represents the time taken by the proxy to download the file from the BitTorrent overlay network (i.e., D_o). The results in Figure 3 show that the (average) time the user's PC must remain active to download the file is always shorter when using the proxy-based approach. Specifically, for the 4 GB file the average active download time with EE-BitTorrent is about 25% lower (4.8 vs. 6.4 hours). Similar results are also obtained with the other two files (see Figure 3).

Then, we replicated the same set of experiments with a single BitTorrent user located at a different location B. The obtained results are summarized in Figure 4. The situation is now totally different, as the legacy approach always outperforms EE-BitTorrent in terms of average download time. The completely different behavior in the two locations is due to the available uplink throughput. Although the two locations have the same nominal (uplink and downlink) bandwidth, the actual uplink throughput available at location A is much smaller than that available at location B. Due to the *Tit-for-Tat* policy used by the (legacy) BitTorrent protocol, peers with a low uplink rate are penalized and experience a very low download rate which increases the file download time (and the corresponding energy consumption). When using EE-BitTorrent, the time for transferring the file from the proxy to the user PC depends only on the available downlink rate. In addition, since the proxy accesses the

Internet through a 100 Mbps link, and can thus provide a high uplink rate, it achieves a very high download rate from the BitTorrent overlay network. Hence, $D_o(i)$ is typically very small, compared with $D_L(i)$ and $D_t(i)$. Specifically, when using the legacy protocol at location A, in the best case, we measured an average download rate of 0.89 Mbps, while with EE-BitTorrent the average transfer rate from the proxy, again in the best case, was 2.86 Mbps. Conversely, for location B in the best case we measured an average download rate of 6.66 Mbps with legacy BitTorrent, and a transfer rate of 6.18 Mbps with EE-BitTorrent.

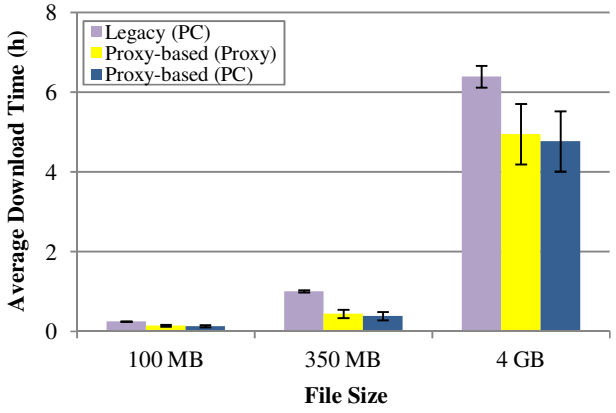

Figure 3. Average download time for the user at location A.

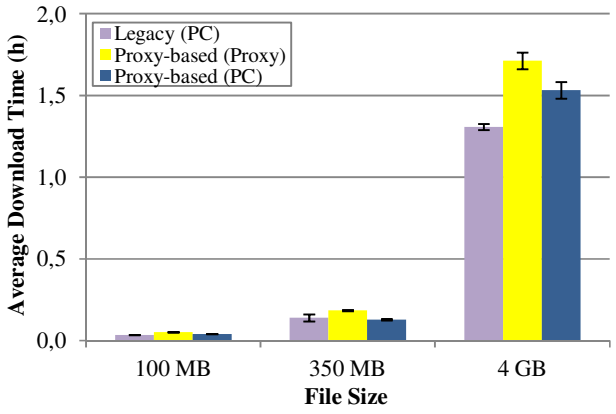

Figure 4. Average download time for the user at location B.

In terms of energy consumed by the single user, by introducing the previous results in Equation (1) it follows that using EE-BitTorrent is very beneficial for location A (25.4% reduction with respect to legacy BitTorrent for the 4 GB file), while it is not for location B (16.8% increase with respect to legacy BitTorrent for the 4 GB file). In terms of energy consumed by the overall system (i.e., user's PC + proxy), EE-BitTorrent is never convenient. This was expected as there is a single PC using the proxy.

We now consider the (more realistic) case when there are several BitTorrent users downloading their files in parallel. Specifically, we considered 4 different users accessing the Internet from different locations, namely A, B (already introduced), C and D. When there are several users transferring bytes from the proxy in parallel, the total transfer time experienced by the proxy can be approximated to the maximum time taken by the slower PC, i.e., $\sum_{i=1}^{N} D_t(i) = \max_i \{D_t(i)\}$. Table 1 shows the average time taken by user at locations A and B (the users with the worst and best transfer rates, respectively) to download the 4 GB file from the proxy, for an increasing number of involved PCs. For the same location, the transfer time does not vary significantly along with the number of involved PCs, at least up to some extent. This is because the downlink bandwidth of ADSL users is much lower than the uplink bandwidth of the proxy. Hence, the proxy can serve a number of users in parallel. Equation (3) can thus be re-written as

$$S_{sys}(N) \approx 1 - \frac{\sum_{i=1}^{N} D_o(i) + \sum_{i=1}^{N} D_t(i) + \max_i \{D_t(i)\}}{\sum_{i=1}^{N} D_L(i)} \qquad (4)$$

In (4) the total time taken by the proxy to download all the requested files from the overlay network may be lower than $\sum_{i=1}^{N} D_o(i)$, as (**i**) some files could be already available on the proxy, and (**ii**) files are downloaded in parallel. The values of $S_{usr}^{i}(N)$ and $S_{sys}(N)$, calculated according to (1) and (4), respectively, for different N values, are shown in Table 2. For the sake of space, we show the results for only two locations (A and B) and for the 4 GB file. For the other file sizes the results are similar. The energy saving introduced by EE-BitTorrent for the specific user (i.e., S_{usr} in Table 2) does not depend significantly on the number of other involved PCs; it only depends on the specific location. The variations of S_{usr}^{A} and S_{usr}^{B} in Table 2, for different values of N, are only due to variations in the rate achieved by the user at time when the different experiments were performed (see also Table 1). Specifically, energy savings introduced by EE-BitTorrent are always positive for user at location A, while they are always negative for user at location B. Conversely, the system-level energy saving (i.e., S_{sys}) is the same for all users (for N>1), as it is calculated over all the users. As expected, it is largely negative for N=1, because there is only a single PC using the proxy. However, it increases with the number of involved PCs (i.e., users downloading in parallel) because the additional energy consumed by the proxy, in comparison with the legacy approach, is shared among a large number of users, as expected in practical scenario.

TABLE 1. AVERAGE DOWNLOAD TIME FOR USERS A AND B WITH DIFFERENT NUMBER OF PARALLEL DOWNLOADS (4 GB FILE).

Location	1 PC	2 PCs	3 PCs	4 PCs
A	4.77h (4.01-5.53)	3.72h (2.96-4.49)	4.27h (3.72-4.82)	4.55h (4.40-4.71)
B	1.53h (1.48-1.58)	1.50h (1.47-1.54)	1.56h (1.52-1.61)	1.57h (1.52-1.62)

TABLE 2. USER AND SYSTEM ENERGY SAVINGS (4 GB FILE).

Location		1 PC	2 PCs	3 PCs	4 PCs
A	S^{A}_{usr}	25.4%	41.8%	33.2%	28.8%
	S_{Sys}	-52.1%	-1.2%	0.8%	4.8%
B	S^{B}_{usr}	-16.8%	-14.5%	-19.1%	-19.8%
	S_{Sys}	-147.3%	-1.2%	0.8%	4.8%

C. Learned Lesson

From the results presented in the previous sections, we can draw the following conclusions. When using a residential (ADSL) network, the available uplink throughput and, hence, the performance achieved with the legacy BitTorrent protocol, is highly dependent on the specific location. In addition, for the same location, it may also vary over time. As a consequence, when the available uplink throughput is not so high, EE-BitTorrent largely outperforms the legacy approach. Otherwise, the legacy protocol is more convenient than EE-BitTorrent. Thus, the situation is different from that in a departmental scenario where, according to the experimental results in [6], EE-BitTorrent always outperforms the legacy approach (when the number of PCs served by proxy is larger than 1), thanks to the high bandwidth available in such a scenario. In a residential scenario the most convenient approach cannot be decided *a priori*, as it ultimately depends on the operating conditions of the access network. Motivated by these conclusions, in the next section we will define *Adaptive BitTorrent (AdaBT)*, an adaptive algorithm that selects the most convenient option (among legacy BitTorrent and EE-BitTorrent) depending on the operating conditions available at download time.

IV. ADABT ALGORITHM

AdaBT takes a very simple approach. The basic idea is that, upon receiving a request from the user, the AdaBT module running on the user's PC measures the download rate that can be achieved from the BitTorrent overlay network and the BitTorrent proxy. Then, on the basis of these measurements, it selects the most convenient option. When the selected option is EE-BitTorrent, if the requested file is already available on the proxy (which is a very frequent case), the data transfer to the user's PC can start immediately. Otherwise, the BitTorrent user is invited to switch off the PC (for energy saving) and reconnect later for downloading the file from the proxy.

More precisely, Algorithm 1 shows the detailed actions performed by AdaBT. Upon receiving a request from the user, AdaBT starts estimating the download rate currently available from the BitTorrent overlay network (lines 2a-7a) and from the configured BitTorrent proxy (lines 2b-7b). In the latter case, if the requested file is not yet available on the proxy, dummy packets are sent. The two measurements are carried out in parallel. For legacy BitTorrent, the algorithm waits for the first chunk of data being received before starting to estimate the available download rate. In both cases, the available rate is estimated by downloading a pre-defined amount of bytes Q, and by measuring the corresponding time. When of taking the decision about the most convenient option (lines 10-11), AdaBT takes into account the overhead introduced by the preliminary bandwidth estimation phase. EE-BitTorrent is selected only when the estimated rate from the proxy is significantly larger than the estimated rate from the overlay network. This is the reason of using the parameter α in line 10. In our experiments (discussed in Section V), we used $\alpha = 0.9$.

Algorithm 1: AdaBT Algorithm

```
1   AdaBT_start_time = time();

2a  Start Legacy BitTorrent; wait for the first chunk of data;
3a  Legacy_start_time=time();
4a  Wait for Q data;
5a  Legacy_end_time = time();
6a  TL=Legacy_end_time-Legacy_start_time;
7a  RL=Q/TL; // estimated rate with legacy protocol

2b  request S data from Proxy;
3b  Proxy_start = time();
4b  Wait for Q data;
5b  Proxy_end_time = time();
6b  TP=Proxy_end_time-Proxy_start_time;
7b  RP=Q/TP; // estimated rate from proxy

8   AdaBT_end_time=time();
9   TOH = AdaBT_end_time-AdaBT_start_time;
10  if (L/RP+TOH) < α·((L-Q)/RL)) then use EE-BitTorrent
11  else use legacy BitTorrent;
```

An important issue in the design of the AdaBT algorithm is related with the setting of Q, i.e., the number of bytes on which the estimate of the available throughput should be calculated. Of course, a low value for Q could compromise the accuracy of the estimate. On the other side, this parameter has a direct impact on the overhead introduced by AdaBT. Obviously, the overhead should be very small, compared with the total time taken to download the desired file. However, this time cannot be known in advance as it strongly depends on the operating conditions, as highlighted in Section III. Of course, several criteria can be used to select the appropriate Q value. In our implementation we set Q as $Q = \min\left\{\frac{L}{10}, Q_{\max}\right\}$, where $Q_{\max}$ is a pre-defined threshold aimed at limiting the duration of the preliminary bandwidth estimation phase when the file is very large[1].

V. EXPERIMENTAL ANALYSIS OF ADABT

In this section we evaluate the performance of AdaBT and compare it with both legacy BitTorrent and EE-BitTorrent. To this end, we used the same testbed introduced in Section III. We limited our analysis to the case of a single BitTorrent user accessing the Internet (through an ADSL link), either from location A or from location B. We considered the same file sizes used in Section III (i.e., 100

[1] Since AdaBT introduces a fixed overhead, it is not appropriate when the file size is comparable with $Q_{\max}$. In such a case, it is more convenient to use straightforwardly the legacy BitTorrent protocol.

MB, 350 MB, and 4 GB), and the same number of seeds (about 500). In addition, for AdaBT, we set α =0.9 and Q_{max}= 10 MB.

Figure 5 shows the average active download time provided by AdaBT to the user at location A. This time also includes the overhead introduced by AdaBT to estimate the available throughput and select the most efficient option. For each considered file size, Figure 5 also shows the option selected by AdaBT to download the file, i.e., legacy (L) or proxy-based (P) approach. Finally, it also shows the average download time provided by the alternative option (i.e., the option not selected by AdaBT). For location A, AdaBT always chooses the proxy-based option (in accordance with the results shown in Section III for the same location). Despite the introduced overhead, AdaBT is able to provide a significant reduction in the download time, with respect to the legacy approach.

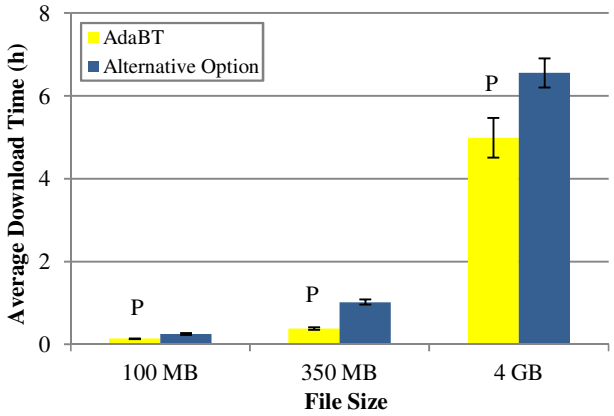

Figure 5. Average download time provided by AdaBT at location A.

We repeated the same set of experiments for the user at location B. In this location the rates available from the proxy and the BitTorrent overlay are similar. Figure 6 shows the download time experienced by AdaBT in different replications of the experiment with the 4 GB file (the results for the other files are similar). The selected BitTorrent option varies with the replication of the experiments; however, AdaBT always selects the most efficient option (given the current operating conditions), as highlighted by the comparison with the alternative option in Figure 6.

VI. CONCLUSIONS

In this paper we have addressed the problem of energy-efficient P2P file sharing for residential BitTorrent users. In the first part, we have compared, through measurements on a real testbed, the legacy BitTorrent protocol with EE-BitTorrent, a proxy-based approach that has proved to be very efficient in a departmental scenario. We found that, for residential (ADSL) users, the achieved performance is strongly influenced by the specific location and, for the same location, it may also vary over time. When the available uplink rate is low, EE-BitTorrent outperforms the legacy BitTorrent, in terms of average download time and energy consumption at the user's PC. The opposite occurs when the available uplink rate is good. Motivated by these results, in the second part of the paper, we have defined AdaBT, an adaptive algorithm that is able to select the most efficient BitTorrent option (i.e., legacy or proxy-based BitTorrent), depending on the current operating conditions experienced by user. Our experimental results have shown that AdaBT always selects the best BitTorrent option, thus enabling a significant reduction in the file download time.

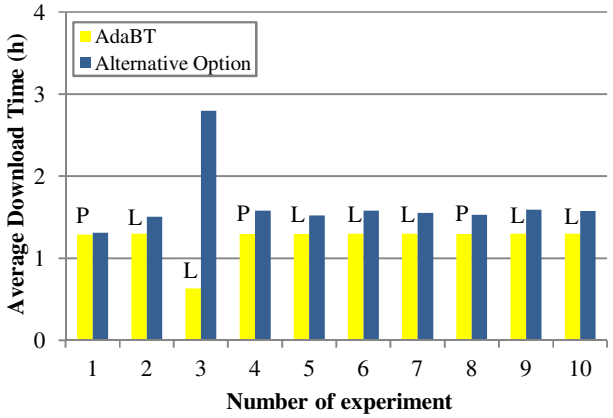

Figure 6. Average download time provided by AdaBT at location B (file size = 4 GB).

REFERENCES

[1] K. Christensen, A. D. George, "Increasing the Energy Efficiency of the Internet with a Focus on Edge Devices", *The Energy Efficient Internet Project*, University of South Florida and University of Florida, Florida, 2005 – 2008.

[2] R. Bolla, R. Bruschi, F. Davoli, F. Cucchietti, "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-aware Fixed Network Infrastructures", *IEEE Communications Surveys & Tutorials*, Vol. 13, N.2, 2011.

[3] H. Schulze, K. Mochalski, "The Impact of Peer-To-Peer file sharing, voice over IP, Skype, Joost, Instant Messaging, One-Click Hosting and Media Streaming such as YouTube on the Internet", *IPOQUE – Internet Study 2007*, Leipzig, Germany, September 2007.

[4] A. R. Bharambe, C. Herley, V. N. Padmanabhan, "Analyzing and Improving BitTorrent Performance", *Technical Report MSR-TR-2005-03*, February 2005.

[5] J. Blackburn, K. Christensen, "A Simulation Study of a New Green BitTorrent", Proceedings *International Workshop on Green Communications (GreenComm 2009)*, Dresden, Germany, June 2009.

[6] G. Anastasi, I. Giannetti, A. Passarella, "A BitTorrent Proxy for Green Internet File Sharing: Design and Experimental Evaluation", *Computer Communications*, Vol. 33, N. 7, pp. 794-802, May 2010.

[7] H. Hlavacs, R. Weidlich, T. Treutner, "Energy Efficient Peer-to-Peer File Sharing, *Journal of Supercomputing*, available online at http://dx.doi.org/10.1007/s11227-011-0602-8.

[8] I. Kelenyi, A. Ludanyi, J. Nurminen, I. Pusstinen, "Energy-efficient Mobile BitTorrent with Broadband Router Hosted Proxies", Proceedings *IFIP Wireless and Mobile Networking Conference (WMNC 2010)*, Budapest, Hungary, October 13-15, 2010.

[9] I. Kelenyi, A. Ludanyi, J. Nurminen, "BitTorrent on Mobile Phones – Energy Efficiency of a Distributed Proxy Solution", Proceedings *International Green Computing Conference (IGCC 2010)*, Chicago, USA, August 15-18, 2010.

[10] J. Kurose, K. Ross, "Peer-to-Peer Applications", in *Computer Networking. A Top-Down Approach*, IV Edition, Addison Wesley, 2007.