Dear Dr. Fonseca,

we thank you and the referees for reviewing our paper. Below we provide detailed answers to the reviewers' comments. Our responses are marked by (*). We highlight the changes in the manuscript using blue italic font.

Best regards,
Amit Mondal
Ionut Trestian
Zhen Qin
Aleksandar Kuzmanovic

Reviewer #1: I'm satisfied with the authors corrections, but I still have three questions

1) as another reviewer pointed out, it is unclear what was implemented by the authors in MirrorPlane. Is it only a monitoring tool? Or is it also a scheduler prototype? In case it is a scheduler prototype, how was it tested? What was the experimental setup used to generate Figure 12? Why do the authors claim that "the replication scheduling module is currently implemented in an event-driven simulator where we replay our collected traces and hence validate our solution"? Isn't the replication scheduling also implemented in MirrorPlane?

(*)
Indeed, the monitoring part of MirrorPlane is fully implemented by changing the Bitflu BitTorrent client just to monitor swarms.

As for the scheduler, it is a prototype that is built inside an event-driven simulator, the same event-driven simulator that we introduced in section 4.1.2.

Figure 12 was obtained by replaying the traces we collected in section 4.1.1 in the simulator described in section 4.1.2. It provides an aggregate picture across torrents for the upload/download imbalance without and with MirrorPlane.

We further clarify this on page 17, paragraph 2 of section 5.

"Note that the monitoring component is fully implemented by modifying the Bitflu BitTorrent client and incorporating characteristics that we detail below. Because deploying and testing MirrorPlane inside an ISP is difficult, the scheduling component is currently implemented and evaluated as a module in the event-driven simulator we described in Section 4.1.2."

And with regards to Figure 12 we add on page 18, paragraph 6 of section 5.

"The same setup was employed here as in Section 4. We present these results in order to get a big picture view of the functioning of our system. Top swarms now have the necessary resources at almost all intervals of time. Torrents beyond rank 30 have very limited demand as leecher arrivals are rather sparse and unpredictable and as a result, our algorithm does not manage to efficiently place resources to such torrents."

---

2) the authors claim that they "abandon the common approach of deploying novel incentives for cooperation". Nonetheless, the download of (unrequested but recommended) content could be a strong enough incentive for cooperation. I suggest the authors discuss this possibility. Even if users are not interested in the downloaded content, they might be willing to download unrequested content to share, for bartering purposes. See

Daniel Sadoc Menasché, Laurent Massoulié, Donald F. Towsley: Reciprocity and Barter in Peer-to-Peer Systems. INFOCOM 2010: 1505-1513

Thinking in terms of incentives, if a user downloads unrequested content (and shares it while downloading) 1) he gains, by getting content that might be helpful to him; 2) the system gains, as described in the paper. Actually, the download of additional content is an incentive for users, that otherwise would leave the system, to stay online longer (even though seeding is prevalent, there is no incentive to altruistic seeding, but there ARE incentives to download and share unrequested content).

(*)
We agree with the reviewer that downloading unrequested content could indeed be incorporated in a novel incentive mechanism for BitTorrent. We acknowledge this by incorporating the following paragraph in the paper on page 19, paragraph 4 of section 6 and cite the recommended reference.

"Furthermore, the authors of [38] observe that the download of unrequested but recommended content could be a strong incentive for cooperation among users. Even without a direct interest in the content that is placed on their machines, users might still be willing to download the content for bartering purposes. The authors study direct reciprocity systems similar to the ones employed by BitTorrent and find out that in some cases it is crucial that users download content for the purpose of bartering as it can lead to higher performance with marginal overhead."
---

3) how do you infer the supplied (upload) bandwidth and the demanded (download) bandwidth in a swarm, e.g., to plot Figure 5(b)? In a real swarm, aggregate download rate should never be larger than aggregate upload rate.

(*) Figure 5 represents the upload bandwidth and the download bandwidth for a swarm inside a single ISP. This is already specified in the caption. Basically this covers only the peers that a swarm has inside an ISP. While for overall swarm aggregates the download rate should never exceed upload rate as the reviewer observes, this it is not the case if analyzing a swarm inside a single ISP (there exist peers outside the ISP also). For example, the download rate might be higher than the upload rate for a swarm inside an ISP with asymmetric connections that has some peers outside the ISP with better connections (in a campus environment for example).

Reviewer #2: With respect to point (1) of my first review, Piatek et al. (2009) say that "local peers are not always faster, particularly for users in regions where asymmetric bandwidth capacities are typical". This is why in their experiments, Piatek et al. swap distant for local peers only if the switch does not degrade performance (because otherwise a peer would not follow the protocol). On the contrary, the authors explicitly say on page 13 that their algorithm chooses a local peer, whenever there is a local peer with the file. And this peer selection occurs irrespectively of performance. So in point (1) I was asking whether this peer selection policy degrades the performance of some peers (even though it improves aggregate performance). I guess that the authors didn't understand my question, since in their answer they talked about what a seeder does (who they say gives priority to the original torrent) and not about the leecher.

(*) The reviewer talks about swapping peers which we never suggest doing in our algorithm. Indeed, according to the BitTorrent protocol, a BitTorrent client is free to look for other peers as long as the ones he currently has do not saturate his download bandwidth. We do not change the BitTorrent protocol in that respect. What we do is suggest local peers that have the file when the client requests them. If local peers will not saturate his download bandwidth, the client is free to look for distant peers that indeed might offer better performance. That is why there is still some amount of inter-AS traffic left.

If the client is connected to a distant peer that offers him a good performance download, we never suggest a swap. Only when the client requests peers we suggest local peers that have the file. So no, performance is not degraded since if we could not identify local peers that would saturate his download bandwidth the client is free to connect to distant peers and stick with them if they offer him good performance.

With respect to point (2) of my first review, the paragraph that the authors added on page 19 doesn't clearly describe the issue. They should more clearly explain what a change in the context means.

(*)
We agree with the reviewer and in order to explain what we mean by a change in context in this case we have added the following sentence on page 20, paragraph 8 of section 6.

"In this case, by change of context we mean users downloading content that they are not interested in on their computers as opposed to the previous case when they are only downloading content that they are directly interested in."

With respect to point (3) of my first review, it is not clear to me from the text that when replicating content peers download content only from local peers. I suggest that the authors clarify this.

(*) We agree with the reviewer that this is not clearly explained in the paper and we add the following on page 11, paragraph 4 of section 3.2.

"Finally, when actively replicating content, peers download content only from local peers, thus does they do not generate any more inter-AS traffic."

With respect to the last point of my first review (about Algorithm 3.1), it is still not clear from the text what Algorithm 3.1 does. The authors should provide more details in Algorithm 3.1 and/or the discussion below it. Right now, by looking at Algorithm 3.1, it seems that there are only two cases:
(a) if aggrUpload > aggrDownload + 2 * stddev, then do nothing
(b) if aggrUpload < aggrDownload + 2 * stddev, then add to the priority queue with the goal of replicating

---
(*) We agree with the reviewer about the lack of clarity of Algorithm 3.1 and the surrounding text. Therefore we add the following text on page 11, paragraph 4 of section 3.2.

"Note that in the case when i.e., aggrDownload + 2stdev < aggrUpload < aggrDownload + 3stddev this is the case when the swarm has enough upload capacity so we do not do anything, we do not take resources from this swarm otherwise it will move to the previous case and we will have to actively replicate it. Also, when i.e., aggrUpload < aggrDownload we do not actively replicate as upload is already scarce and therefore we do not want to increase load. The last two cases are not included in the algorithm as they result in no action."
