This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE P2P 2011 proceedings.

Identifying, Analyzing, and Modeling Flashcrowds in BitTorrent

Boxun Zhang, Alexandru Iosup, Johan Pouwelse, Dick Epema
Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology
{B.Zhang,A.Iosup,J.A.Pouwelse,D.H.J.Epema}@tudelft.nl

***Abstract*—Flashcrowds—sudden surges of user arrivals—do occur in BitTorrent, and they can lead to severe service deprivation. However, very little is known about their occurrence patterns and their characteristics in real-world deployments, and many basic questions about BitTorrent flashcrowds, such as *How often do they occur?* and *How long do they last?*, remain unanswered. In this paper, we address these questions by studying three datasets that cover millions of swarms from two of the largest BitTorrent trackers. We first propose a model for BitTorrent flashcrowds and a procedure for identifying, analyzing, and modeling BitTorrent flashcrowds. Then we evaluate quantitatively the impact of flashcrowds on BitTorrent users, and we develop an algorithm that identifies BitTorrent flashcrowds. Finally, we study statistically the properties of BitTorrent flashcrowds identified from our datasets, such as their *arrival time*, *duration*, and *magnitude*, and we investigate the relationship between flashcrowds and swarm growth, and the arrival rate of flashcrowds in BitTorrent trackers. In particular, we find that BitTorrent flashcrowds only occur in very small fractions (0.3-2%) of the swarms but that they can affect over ten million users.**

I. Introduction

The term "Flash Crowd" was introduced by Larry Niven [16] in 1973 as the title of his science-fiction novel about the social consequences of an instantaneous teleportation device. One of these consequences was that newsworthy events would cause tens of thousands of people to teleport to the scenes of those events, causing public disorder. Decades later, the term flashcrowd was used to describe the phenomenon of the services of a website being severely degraded, or even interrupted, due to an unexpected surge of visitors after the website is mentioned by popular news websites—thus, this phenomenon has also been called the "Slashdot effect" [25]. Flashcrowds can also hit peer-to-peer systems, and in this paper we will investigate flashcrowds as they appear in BitTorrent, which has been one of the most popular P2P file-sharing applications for a decade [2].

Flashcrowds can lead to decreased responsiveness and increased backlogs for many types of traditional systems, which in turn leads to user dissatisfaction. The cost of flashcrowds is difficult to estimate for the whole IT industry, but sample information is available: Amazon reports [14] that even small (100 ms) delays for web page generation will cause a significant (1%) drop in sales, and Google reports [14] that an additional half a second in the average search response time causes a traffic drop of up to 20%. A variety of techniques, such as DNS load balancing [8] and geo-replication services [21], have been proposed to alleviate the problem of flashcrowds in web servers. However, there is an important difference between flashcrowds in web servers and BitTorrent in that the service capacity of a BitTorrent swarm (the group of peers in the process of downloading the same file) grows as the number of peers increases, while web servers have fixed capacities.

Flashcrowds seem to have become more prevalent in BitTorrent in recent years because of the wide adoption of such automated download techniques as RSS feeds. However, only few algorithms have been proposed [6] to address BitTorrent flashcrowds, and in fact, not much is known about their patterns of occurrence and their characteristics in real-world deployments. So, many basic questions about BitTorrent flashcrowds remain unanswered, such as *How often do they occur?*, *How long do they last?*, and *Are BitTorrent peers joining flashcrowds worse off than peers joining regular swarms?* Because tens of millions of peers are active daily in BitTorrent communities, and because several measurement [11] and analytical [20] studies have shown that BitTorrent can achieve efficient large-scale content distribution, flashcrowds have largely been ignored as a potential problem for BitTorrent users. However, our findings show that flashcrowds can affect the download performance of up to 21-45% of BitTorrent users. In addition, although several studies [9], [18] have observed limitations of BitTorrent in handling flashcrowds, they did not have access to the large datasets needed to analyze and model BitTorrent flashcrowds.

In this paper we conduct the first comprehensive study of BitTorrent flashcrowds. We propose a model of flashcrowds that characterizes their properties, such as their *duration* and their *magnitude*, and we develop a flashcrowd identification algorithm that is able to identify BitTorrent flashcrowds from the evolution of swarm sizes. We study three datasets [26] collected in 2009 and 2010 from OpenBitTorrent and PublicBitTorrent, which are two of the largest public BitTorrent trackers nowadays, and which provide us with two million swarms in total. Finally, we perform an analysis and statistical modeling of the BitTorrent flashcrowds identified from our datasets, revealing for their properties the best-fitting probability distributions, and the parameters of the best fits.

Our findings can be readily used to generate synthetic yet realistic workloads for simulation studies and for real-system tuning. Furthermore, the results of this paper can be used to improve the BitTorrent protocol and to build effective flashcrowd mitigation mechanisms. In particular, we are currently trying to apply our models and our flashcrowd-identification algorithm to improve our BitTorrent-based system Tribler [19]. The main contributions of this paper are:

1) We propose a model for BitTorrent flashcrowds (Section III).

Dataset	Tracker	Period	Sanitized swarms
Training dataset	FileList.org etc.	2005 - 2006	40
OBT'09	OpenBitTorrent	Dec 15 - 31, 2009	1,524,743
OBT'10	OpenBitTorrent	Mar 15 - 31, 2010	193,598
PBT'09	PublicBitTorrent	Dec 15 - 31, 2009	311,333

TABLE I: Dataset overview.

2) We show that flashcrowds have a negative impact on millions of BitTorrent peers (Section IV).
3) We develop a flashcrowd identification algorithm (Section V).
4) We study statistically the flashcrowds in two of the largest public BitTorrent trackers (Section VI).

II. BitTorrent and Datasets

In this section, we first introduce briefly BitTorrent. Then, we introduce the datasets used in this work, and the methods that we use to sanitize the datasets.

A. BitTorrent

In BitTorrent, files are divided into small pieces, and BitTorrent peers download these file pieces from each other instead of complete files. Peers downloading the same file connect to each other and form a *swarm*. Swarms are managed by *trackers*, the centralized components in BitTorrent. Trackers do not host any file, but provide peer discovery services in swarms. When a peer joins a swarm, it first asks from a tracker for a list of random peers already in that swarm, and then exchanges file pieces with those random peers. A tracker can serve large numbers of swarms at the same time, but it can also become overloaded under intensive peer requests.

There are two types of peers in BitTorrent: *seeders*, who own complete files and give away file pieces for free, and *leechers*, who do not have complete files. To ensure the pieces of a file are equally distributed in a swarm, BitTorrent employs the *rarest first* piece selection policy, which makes peers always download the pieces that have the least replicas among neighboring peers. To deter *free riders* and maintain reciprocity, BitTorrent employs the *tit-for-tat* peer selection policy, which makes peers always upload to those who recently provided the highest download speed. BitTorrent also uses the *optimistic unchoking* mechanism, by which peers periodically upload to randomly selected peers. This mechanism provides newly joined peers opportunities to obtain their first pieces, and it also allows peers to find better piece-exchange partners.

B. Datasets

In this work we use four BitTorrent datasets: one small dataset manually selected from the P2P Trace Archive [29], and three much larger datasets collected as parts of BTWorld, a BitTorrent measurement that monitors hundreds of trackers and millions of swarms [26]. The small dataset is used to develop and test our flashcrowd identification algorithm (Section V), and the large datasets are used to study BitTorrent flashcrowds. Table I gives an overview of these datasets, which we now describe in turn.

The manually selected dataset (the "Training dataset" in Table I) comprises traces of 40 swarms that are manually selected from the P2P Trace Archive. This training dataset contains detailed peer-level information such as peer arrivals and departures, and swarm-level information. This allows us to experiment with various flashcrowd identification algorithms that are based on different pieces of information.

The three large datasets are collected within BTWorld by scraping two of the world's largest public trackers for half month, with a sampling interval of one hour. Each sample contains information of the numbers of seeders and leechers in each of the swarms served by those trackers. Two of these datasets were collected from OpenBitTorrent (OBT) three months apart, and the other one was collected from PublicBitTorrent (PBT) during a period that matches the collection period of one of the OBT datasets. During the matched period, the overlap of swarms between the OBT and PBT datasets was below 50%.

One reason for using these large datasets is to reduce the potential analysis biases of a single tracker or a single measurement period. Another reason is the difficulty of validating BitTorrent models using synthetic data or (only) small datasets—the complexity and heterogeneity of BitTorrent increase [30] the risk of biased models and over-fitting. Since OpenBitTorrent and PublicBitTorrent are nowadays two of the most populated public trackers, we assume that the results derived from the datasets of these trackers are representative for public BitTorrent trackers.

C. Data Sanitation

The OBT and PBT datasets have been collected by automated tools, and they can be polluted by measurement artifacts, tracker failures [18], and compromised trackers [26]. We employ four criteria to sanitize the raw data:

1) *The sampling interval of a swarm is one hour. Defective* swarms have longer sampling intervals due to tracker or measurement failures.
2) *The swarm size is valid.* Our study [26] of the global BitTorrent network reports that the largest working swarm does not exceed 373,000 peers; spam trackers may report much larger swarm sizes. *Spam* swarms are larger or have only zero peer during our measurement periods.
3) *The initial swarm size is valid.* This criterion ensures that swarms are measured at or very soon after their creation, so that we do not miss the flashcrowds that occur right after swarm creation. *Halfway* swarms are swarms that do not meet this or the next criterion. We consider conservatively a valid initial swarm size to be at most 20, which is 20% of the size of the smallest swarm among the largest 2,000 PirateBay swarms we have examined [10]. We have tried various values for this threshold, and found that 20 is around the "knee" of the threshold-selection size curve (see our technical report [28]).
4) *The swarm size at the end of measurement is smaller than the half of the swarm peak size.* This criterion ensures that we capture the complete growth of swarms.

From our OBT and PBT datasets, about two million swarms are not defective, spam, or halfway; these swarms have been sanitized. The number of swarms that do not pass our sanitizing criteria is detailed later in Table II. We find a huge number of defective swarms, and a significant overlap between small and short swarms. From hereon, we use the term "datasets" to denote the set of the sanitized swarms.

III. A METHOD FOR STUDYING BITTORRENT FLASHCROWDS

In this section, we first propose a model for BitTorrent flashcrowds, and then we introduce a three-step procedure for studying these flashcrowds.

A. Model of BitTorrent Flashcrowds

We loosely define a BitTorrent flashcrowd as a (significant) increase in the number of peers in a swarm (the swarm size). The flashcrowd starts at the beginning of the increase and finishes at the end of the increase. Similar definitions are also used [3], [7] in modeling web server workloads. As illustrated in Figure 1, our flashcrowd model consists of four components: the *arrival time*, which is the time between the creation of a swarm and the start of a flashcrowd; the *duration*, which is the time between the start and the end of the increase; the *plateau period*, which is the time period with limited churn immediately after a flashcrowd ends; and the *magnitude*, which indicates the significance of a flashcrowd in terms of the increase. It is the purpose of a flashcrowd identification algorithm to find flashcrowds with their arrival times and durations; below we present how then their magnitudes are computed and how their plateau periods are determined.

For the definition of the magnitude of a flashcrowd we will use the following quantities:

- n_b: the swarm size at the beginning of a flashcrowd (initial swarm size).
- Δn: the increase in the swarm size during a flashcrowd.
- Δt: the duration of a flashcrowd in minutes.
- c: the seeder capacity, which is the number of peers a well-provisioned seeder (such as the injector of the file) can serve in a swarm. This is a constant value, and it is used as a lower-bound on the swarm size when calculating the magnitude of a flashcrowd.

The magnitude of a flashcrowd is now calculated as:

$$M = \frac{\Delta n}{\Delta t} \times \frac{\Delta n}{\max(n_b, c)}, \tag{1}$$

where the first factor characterizes the growth rate of the swarm size in a flashcrowd, and the second factor characterizes the increase in the swarm size during the flashcrowd relative to the swarm size at its start. The system capacity at the start of the flashcrowd is characterized by the swarm size at that time, or by the capacity c of a (well-provisioned) seeder if the swarm size at that time is smaller than c. Using the seeder capacity c prevents the magnitude as defined in Eq. (1) from becoming over-reactive to small increases in the swarm size when the swarm is small. The seeder capacity c in Eq. (1) plays a similar role as the minimum job duration in the definition

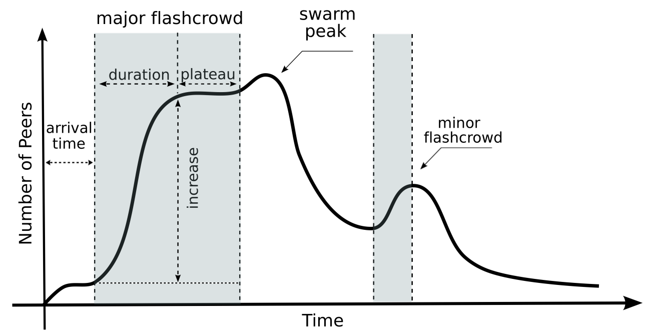

Fig. 1: Model of BitTorrent flashcrowds.

of bounded slowdown of jobs as the ratio of the job residence time and the job duration in computing systems [31]. We discuss the selection of the value of c in Section V. Using only the first factor (the swarm growth rate) in Eq. (1) can cause the algorithm to mistakenly identify flashcrowds with very high growth rates but very small swarm size increases. Finally, as the reverse of flashcrowds, we could define *drops*, that is, (relatively significant) decreases in the swarm size, in a similar way, with resulting magnitudes computed by Eq. (1) that are negative.

We define the plateau period of a flashcrowd in terms of the fluctuation of the swarm size relative to the swarm size at the end of a flashcrowd. More precisely, the plateau period is the maximal period after a flashcrowd ends during which the swarm size does not deviate by more than a certain threshold from the swarm size at the end of the flashcrowd. We discuss the selection of this threshold in Section V.

A BitTorrent swarm can have multiple flashcrowds. We distinguish two types of flashcrowds in a swarm: the *major flashcrowd* and the *minor flashcrowds*. The major flashcrowd is the flashcrowd with the largest magnitude, and the minor flashcrowds are all the other flashcrowds with lower magnitudes. Swarms do not necessarily always reach their peak swarm sizes at the end of their major flashcrowds. We investigate the relationship between the major flashcrowds and swarm growth in Section VI.

B. Metrics for Flashcrowds

We will now define the metrics we use for quantifying properties of flashcrowds (in addition to those presented in Section III-A), which will be used in Section VI:

- The *seed lag* of a flashcrowd is defined as the time elapsed from the start of the flashcrowd until the moment when the first peer completes its download and becomes a seeder. Intuitively, the seed lag is roughly equal to the download time of the first (group of) peer(s) that join the flashcrowd.
- The *peak lag* of a swarm is the time elapsed from the end of its major flashcrowd until the moment when the swarm reaches its peak size.
- The *Average Download Slowdown* (ADS) of a swarm during a certain period is the ratio of the current average download bandwidth of BitTorrent peers in general and the maximum per-peer download bandwidth observed for the swarm during that period.

- The *Pre-Flashcrowd Contribution* (PFC) of a flashcrowd is defined as the ratio of the swarm size at its start and the peak swarm size.
- The *flashcrowd contribution* of a flashcrowd is defined as the ratio of the increase in the swarm size during the flashcrowd and the peak swarm size.

C. Three-Step Procedure

In this paper, we use the following three-step procedure to study BitTorrent flashcrowds:

1) Identification: Visually identifying a flashcrowd from a graph of the evolution of the size of a single swarm is simple, but it is impractical to apply this visual approach to large datasets. Thus, we develop an algorithm that identifies BitTorrent flashcrowds, and also quantifies their properties (Section V).

2) Analysis: We characterize each of the properties of the identified flashcrowds with basic descriptive statistics, and we compare each of the properties of these flashcrowds across our three datasets (Section VI).

3) Statistical Modeling: To find the best probability models for flashcrowd properties, we conduct parameter fitting with a set of well-known, widely used probability distributions that can easily be implemented in simulation models, namely the Exponential, the Weibull, the Pareto, the Log-normal, and the Gamma distributions (Section VI). The parameter fitting is performed using *maximum likelihood estimation* (MLE), which determines for a distribution the parameters that lead to the best fit with the empirical data.

The fitting results are tested against two *goodness of fit* (GOF) tests, the *Kolmogorov-Smirnov* (KS) test and the *Anderson-Darling* (AD) test. Using both of these tests provides less biased results, since the KS test is more sensitive to the center of distributions than the AD test, and the AD test is more sensitive to the tail of distributions than the KS test. We use 0.05 as the significance level for the *p-value*, below which the null hypothesis that the fitted distribution represents the empirical data is rejected. The p-value used in this study is the average of 1000 p-values, each of which is calculated by randomly selecting 30 samples from the empirical data and applying the GOF tests. This is a standard method for computing p-values for large datasets in distributed systems studies [13], [17].

We consider a probability distribution as a good fit for the studied properties only if that distribution passes both the KS and AD tests for all of our three datasets. The best fit of a property is the distribution that has the smallest *D-Statistic*, which is the greatest discrepancy between the empirical and the fitting distributions.

D. Synthetically Generating Flashcrowds

The results of this paper can be used to generate synthetic (major) flashcrowds, for instance, for simulating flashcrowd mitigation algorithms, or more generally, for simulating variations of the BitTorrent protocol. To do so, random samples should be taken from the distributions of the swarm size at the start of their major flashcrowd, of the duration of the major flashcrowd, and of the magnitude of the major flashcrowd. As we will present in Section VI-A5, these three random variables have a very low correlation, and so independent samples can be taken from their distributions. From the duration and the magnitude of the flashcrowd, and the initial swarm size, the increase in the swarm size during the flashcrowd can easily be obtained.

Dataset	Defective	Spam	Halfway	Small	Short	Small & Short
OBT'09	1,788,920	24,312	649,745	1,517,842	43,684	43,661
OBT'10	4,220,819	28,674	432,939	186,760	31,451	31,232
PBT'09	272,792	4,013	117,631	308,344	6,336	6,333

TABLE II: Categories of swarms in our datasets.

Dataset	#Swarms		Total #peers	Increase #peers
	All	Magnitude>1	All	Magnitude>1
OBT'09	1,524,743	4,451 (0.3%)	21,124,765	4,591,676 (22%)
OBT'10	193,598	4,238 (2.0%)	10,465,509	4,731,159 (45%)
PBT'09	311,333	1,821 (0.5%)	5,971,234	1,249,214 (21%)

TABLE III: Numbers of users that may be affected by major flashcrowds of magnitude higher than 1.

IV. Impact of BitTorrent Flashcrowds

For traditional systems where the system service capacity is fixed such as web servers, the effect of flashcrowds on users is clearly negative [3], [7]. However, the negative effect of flashcrowds may not become immediately apparent for systems such as BitTorrent since users provide additional service capacity while being present in the system. Intuitively, a long flashcrowd can lead to a beneficial accumulation of service capacity. In this section we assess the impact of flashcrowds on BitTorrent users by answering two main questions. We find that flashcrowds have a significantly negative impact on many contemporary BitTorrent users.

A. How Many Users Are Affected?

We restrict our study to the swarms that have attained at least some level of popularity. We select such swarms from our sanitized datasets using the following criteria, with the results summarized in Table II:

1) *The swarm size is significant*. We conservatively consider a significant swarm size to be at least 150, which is 50% larger than the size of the smallest swarm among the largest 2,000 PirateBay swarms we have examined [10]. *Small* swarms are swarms that do not meet this criterion.
2) *The swarm lifespan is long*. This criterion eliminates swarms whose activity is short-lived. *Short* swarms are swarms that do not meet this criterion. We conservatively consider a long swarm lifespan to be at least 24 hours, which is much lower than the 30-300 hours reported for BitTorrent swarm lifespans [9].

Applying these two criteria to the sanitized datasets leads to a selection of 6,830 swarms from the OBT'09 dataset, 5,901 swarms from the OBT'10 dataset, and 2,847 swarms from the PBT'09 dataset.

We then identify the major flashcrowds in the selected swarms and calculate their magnitude using our flashcrowd

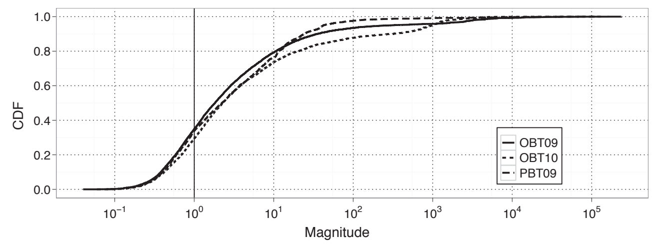

Fig. 2: Magnitude of major flashcrowds in the selected swarms.

identification algorithm (Section V). Figure 2 shows the distribution of the magnitude of major flashcrowds in our selected swarms. We apply a threshold of the magnitude, which we set to 1, to the selected swarms to remove those major flashcrowds that we consider to be insignificant. As an example, a flashcrowd doubles the swarm size with a peer arrival rate of 1 peer/minute has a magnitude of 1. From hereon, we use the term "major flashcrowds" to denote the major flashcrowds in our selected swarms of magnitude higher than 1.

The fractions of the selected swarms whose major flashcrowds have magnitude higher than 1 are very low (0.3-2.0%) for our datasets. We use the ratio of the sum of the increase in swarm size during the major flashcrowds to the sum of the peak sizes of all swarms in our datasets to quantify how many BitTorrent users may be affected by these flashcrowds in Table III. **Finding: The major flashcrowds may affect significant fractions of peers (21-45% for our datasets), although they only occur in very small fractions of the swarms (0.3-2%)**.

B. How Are Users Affected?

To answer the question of how BitTorrent users are affected, we first study the results of previous measurement and simulation-based work and then we conduct our own evaluation of the impact of flashcrowds on BitTorrent peers.

While the high scalability and efficiency of BitTorrent have been shown in previous studies [11], [20], several measurement studies have uncovered limitations of BitTorrent in handling flashcrowds in practice. Pouwelse et al. [18] observed that very few peers finish downloading a popular movie (1.87GB) during a five-day flashcrowd, while it should have taken less than one day to download the same movie at the average download speed of BitTorrent users at that time. Guo et al. [9] have also observed that the increase of the number of seeders lags behind the increase of the number of leechers during flashcrowds, and that peers joining during flashcrowds suffer lower download rates and longer downloading times than peers joining later.

Several simulation-based studies provide additional insight into BitTorrent's limitations in handling flashcrowds. Bharambe et al. [6] find that "pre-seeded" peers (that is, peers that have received most, but not all, file blocks) could take a long time to finish downloading during flashcrowds, because specific file blocks are difficult to find. Urvoy-Keller et al. [24] show that, during flashcrowds, the distribution of file blocks is heavily skewed, which causes a low utilization of the upload capacity of peers. Kaune et al. [12] demonstrate that peers joining a flashcrowd can be treated "unfairly" as they upload more bytes than peers joining later.

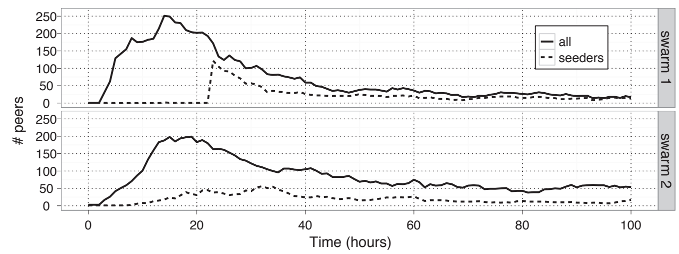

Fig. 3: Swarms with short (top) and long (bottom) increase period of the number of seeders.

For our own evaluation, we quantify the impact of flashcrowds on the download performance of individual peers using the over 10,000 swarms with major flashcrowds of magnitude above 1 selected in Section IV-A. We have computed the seed lags for those swarms. A seed lag equal to zero indicates that the first seeder appears within the first hour after the flashcrowd starts. **Finding: Around 65% of the swarms in which major flashcrowds occur exhibit a non-zero seed lag. For them, the average seed lag is about 19 hours, and it takes on average about 30 hours after the seed lag period for the number of seeders to peak.** As an example, Figure 3 depicts the behavior of two similarly sized swarms: a swarm for which the number of seeders peaks soon after the seed lag period (top) and a swarm in which the number of seeders rises much more slowly (bottom). The peers in the top graph joining early have difficulties in finding the necessary file blocks and cannot finish their download until the end of the flashcrowd. Intuitively, the situation depicted in the bottom plot in Figure 3 is desirable, because the swarm produces seeders quicker so that the flashcrowd becomes less pronounced.

We have examined the correlations between the seed lag and torrent size visually through scatter plots and by computing the Pearson correlation coefficient, and we didn't observe significant correlations between these two quantities. A possible explanation is that the seed lag is decided not only by the torrent size, but also by other factors like uplink utilization and the capacity of the initial seeder.

To analyze the average download slowdown (ADS), we first obtain from a torrent search engine (www.torrentz.eu) the file size of all the swarms that exhibit a non-zero seed lag, and we find the average of these file sizes to be 1.2 GB. Using the average file size and seed lag, we compute the average download rate to be around 147 kbps. According to recent measurement studies [15], [29], the average download rate of BitTorrent users is around 1,000 kbps. **Finding: The ADS of swarms in which major flashcrowds occur is almost 7—a nearly seven-fold decrease in the performance of peers during flashcrowds versus their non-flashcrowd counterparts.**

V. Identifying BitTorrent Flashcrowds

In this section, we first present our magnitude-based flashcrowd identification algorithm. Then, we compare this algorithm with two alternative identification algorithms.

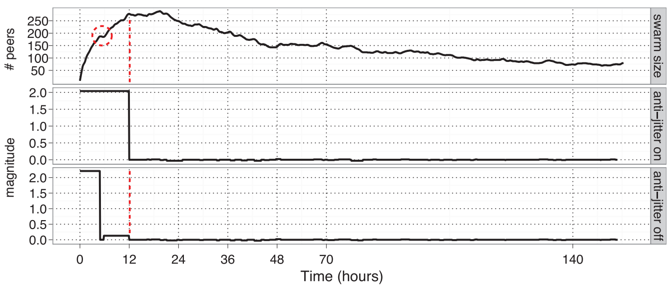

Fig. 4: Magnitude of the flashcrowd of a selected swarm (top) computed by the magnitude-based algorithm with anti-jitter on (middle) and off (bottom).

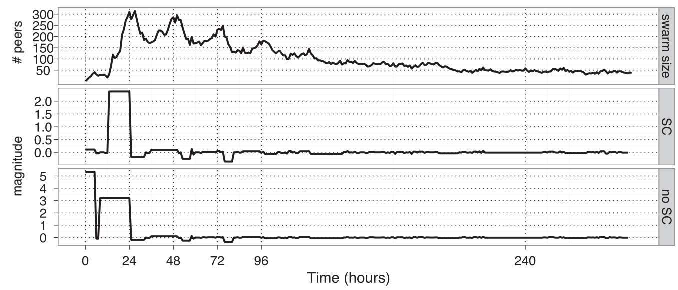

Fig. 5: Magnitude of the flashcrowd of a selected swarm (top) computed by the magnitude-based algorithm with (middle) and without (bottom) the service capacity in Eq. (1).

A. Magnitude-Based Flashcrowd Identification Algorithm

The magnitude-based algorithm for identifying flashcrowds can identify BitTorrent flashcrowds using the information of the evolution of the swarm size. The output of this algorithm consists of the arrival times, the durations, and the magnitudes of the identified flashcrowds. It runs as follows:

1) Compute the swarm size per time step, e.g., per hour.
2) Label every pair of consecutive time steps as increasing, decreasing, or no-change, respectively, depending on the swarm sizes in these time steps.
3) Combine all consecutive time steps that contain only increase, only no-change, or only decrease into segments; these segments are tentatively defined as flashcrowds, stagnancies, and drops, respectively.
4) Identify *jitter* segments, which are small and non-flashcrowd segments, and merge them with neighboring segments.
5) Calculate the magnitudes of flashcrowd and drop segments using Eq. (1).

Step 4 in this algorithm is crucial for correctly identifying flashcrowds, since without it, a jitter segment might cause the algorithm to mistakenly split a large flashcrowd into multiple smaller ones, as shown in Figure 4. We use thresholds for the magnitude and the duration in order to determine if a segment is a jitter segment that should be merged with its neighboring segments. A segment is a jitter segment if its magnitude and duration are below these thresholds.

To find the best combinations of the thresholds for determining jitter segments, we first manually select 300 swarms from our datasets as the test dataset. Then, we experimented with $0.02, 0.05$, and 0.10 for the magnitude threshold, and $1, 2$, and 3 hours for the duration threshold on the test dataset. These numbers are based upon our observations and experiments on many traces, and we observe slight and insignificant differences among different combinations of these values. In the end, we select 0.05 and 2 hours as the thresholds for magnitude and duration respectively, as a middle point between being too sensitive and too insensitive to jitter segments. Noticeably, the duration threshold should be adjusted accordingly when applied to datasets with different sampling intervals.

As mentioned in Section III-A, we use the system capacity c as the lower bound on the swarm size when calculating flashcrowd magnitude, and a threshold of the fluctuation of swarm size to determine the plateau periods of flashcrowds. Here we use the same test dataset to experiment with various values for the system capacity c and the plateau period threshold. We manually examine the results of running this algorithm with these values on the test dataset. We choose the values that lead to the best identification results based on visual inspection. In the end, we select 50 as the value for the system capacity c, and we also believe that it is a reasonable estimation for the capacity of a well provisioned seeder nowadays. Figure 5 gives an example of running this algorithm without having the service capacity c in Eq. (1): the small increase at the beginning of a swarm is identified mistakenly as the flashcrowd of the highest magnitude. We set the threshold for determining the plateau periods of flashcrowds to 5%, and this value also conforms with our observation of the plateau period of the flashcrowd reported in [18].

In the rest of this paper, we focus only on the major flashcrowds of magnitude above 1 in our selected swarms. We do not present in this paper the results for minor flashcrowds, which have much lower magnitude than major flashcrowds. We refer readers to our technical report [28] for the analysis of minor flashcrowds.

B. Alternative Flashcrowd Identification Algorithms

Before having the magnitude-based algorithm, we experimented with two alternative flashcrowd identification algorithms. In this section, we explain briefly why these alternatives do not meet our needs in this work and compare them with our magnitude-based algorithm.

The swarm-size-based algorithm identifies the time periods, during which the ratio of the average swarm size to the swarm size at that moment is above certain threshold T, as flashcrowds, and it indicates a/no flashcrowd by 1/0. This algorithm has three major problems: first, the calculation of the average swarm size depends on the measurement periods, which leads to different results of running this algorithm on the same swarm but with different measurement periods. Second, this algorithm may miss the initial part of a flashcrowd which only has a small swarm size, and may also identify non-flashcrowd (such as decreasing) periods with large swarm size as flashcrowds. Third, it is difficult to choose a meaningful value for the threshold T.

Similarly to the swarm-size-based algorithm, the arrival-rate-based algorithm identifies flashcrowd periods based on peer arrival rate instead of swarm size, and indicates a/no

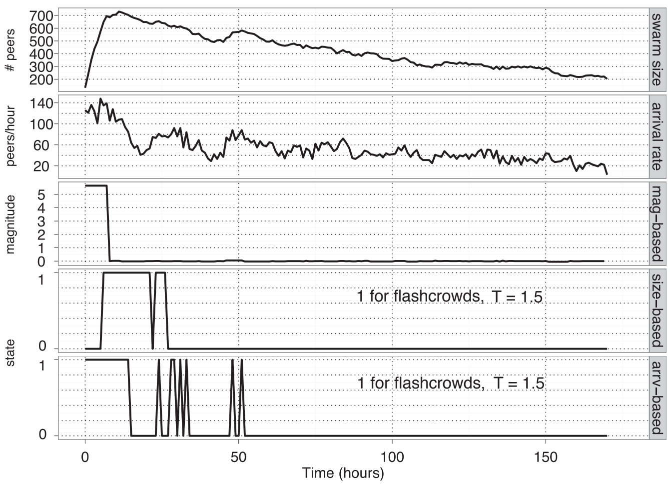

Fig. 6: Comparison of identification results of the magnitude-based (mag-based), the swarm-size-based (size-based), and the arrival-rate-based (arrv-based) algorithms.

flashcrowd by 1/0. Since the peer arrival rates in BitTorrent swarms are usually high at the beginning of swarms, this algorithm can capture the initial part of a flashcrowd. However, this algorithm may still identify non-flashcrowd periods as flashcrowds due to the improperly selected threshold T.

Figure 6 gives an example of running the magnitude-based algorithm and two alternatives ($T = 1.5$) on the trace of a single swarm: the magnitude-based algorithm correctly identifies the flashcrowd and do not treat any non-flashcrowd period as flashcrowd; the swarm-size-based algorithm fails to capture the beginning of the flashcrowd and mistakenly treats some parts in the decrease phase as flashcrowd; and the arrival-rate-based algorithm captures the initial parts of the flashcrowd but becomes over-reactive in the post-flashcrowd phase.

VI. Analyzing and Modeling BitTorrent Flashcrowds

In this section, we present our flashcrowd analysis and modeling results. We first study the basic properties of the major flashcrowds. Then, we investigate their relationship with swarm growth, and their arrival rates at tracker level.

A. The Major Flashcrowds

In this section, we study, in turn, the arrival time, the duration, the plateau period, and the magnitude of the major flashcrowds. We summarize the statistics and the best fitting distributions for these properties in Tables IV and V, respectively. We refer readers to our technical report [28] for the p-values and parameters found for all the distributions.

1) Arrival time: **Finding: Most major flashcrowds start soon after swarm creation**. As shown in Figure 7 (left), the arrival time is 0 for around 70% of the major flashcrowds, and is longer than 24 hours for less than 20% of the major flashcrowds. We notice that the major flashcrowds in the OBT'10 dataset have, on average (Table IV), much shorter arrival time than the major flashcrowds in the other datasets.

The arrival time of flashcrowds is largely determined by how soon BitTorrent users are notified about new torrents, in other words, *how torrents are published*. Nowadays, most (popular) torrents are published via websites, such as Ubuntu.com; via BitTorrent communities, such as ThePirateBay; and via RSS feeds. We believe that the presence of the large numbers of "rapidly arriving" major flashcrowds is due to the wide adoption of RSS feeds by BitTorrent clients and websites. We expect that the trend of using RSS feeds in BitTorrent will continue. Thus, the average arrival time of major flashcrowds may become even shorter in future.

For swarms whose major flashcrowds have non-zero arrival time (late flashcrowds), we show their pre-flashcrowd contribution (PFC). In Figure 7 (right), we find that he PFC is below 0.1 for nearly 90% of swarms with late flashcrowds, and is higher than 0.3 for only very few (1%). This means that even for the swarms whose major flashcrowds do not start within one hour after swarm creation, most pre-flashcrowd phases do not contribute significantly to swarm size.

Property	Dataset	Min	1st Q	Median	Mean	3rd Q	Max
Arrival time (hours)	OBT'09	0	0	0	18.66	3	379
	OBT'10	0	0	0	5.04	2	303
	PBT'09	0	0	0	10.63	1	330
Duration (hours)	OBT'09	1	6	11	13.18	18	138
	OBT'10	1	6	10	12.16	17	78
	PBT'09	1	5	10	11.97	17	101
Plateau (hours)	OBT'09	0	0	0	0.55	0	42
	OBT'10	0	0	0	0.49	0	22
	PBT'09	0	0	0	1.67	1	50
Magnitude	OBT'09	1.00	1.90	4.40	559.26	14.91	237,167
	OBT'10	1.00	1.99	5.16	248.50	27.93	139,359
	PBT'09	1.00	2.27	5.67	186.01	14.05	109,230
Initial swarm size (peers)	OBT'09	1	1	4	13.05	12	399
	OBT'10	1	1	3	12.59	12	426
	PBT'09	1	1	1	9.55	8	434

TABLE IV: Statistics of the properties of major flashcrowds.

Property	Distribution	Dataset	D-Statistic	Parameters
Arrival time (hours)	LogNormal	OBT'09	0.06	μ = 3.00, σ = 1.70
		OBT'10	0.05	μ = 2.08, σ = 1.23
		PBT'09	0.15	μ = 2.16, σ = 1.71
Duration (hours)	LogNormal	OBT'09	0.04	μ = 2.44, σ = 0.68
		OBT'10	0.04	μ = 2.38, σ = 0.67
		PBT'09	0.04	μ = 2.30, σ = 0.77
Plateau (hours)	Gamma	OBT'09	0.22	κ = 1.61, λ = 1.79
		OBT'10	0.20	κ = 1.83, λ = 1.54
		PBT'09	0.15	κ = 1.20, λ = 4.59
Magnitude	LogNormal	OBT'09	0.16	μ = 2.09, σ = 2.10
		OBT'10	0.14	μ = 2.40, σ = 2.23
		PBT'09	0.09	μ = 1.91, σ = 1.47
Initial swarm size (peers)	Gamma	OBT'09	0.18	κ = 0.57, λ = 22.94
		OBT'10	0.19	κ = 0.55, λ = 23.04
		PBT'09	0.26	κ = 0.52, λ = 18.32

TABLE V: Parameters and the D-Statistic of the best fitting probability distributions for the properties of major flashcrowds.

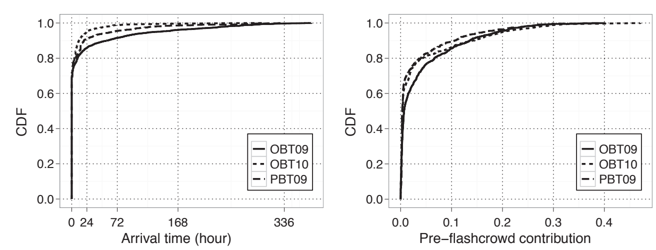

Fig. 7: Left: CDF of the arrival time of major flashcrowds. Right: CDF of the PFC of swarms whose major flashcrowds have non-zero arrival time.

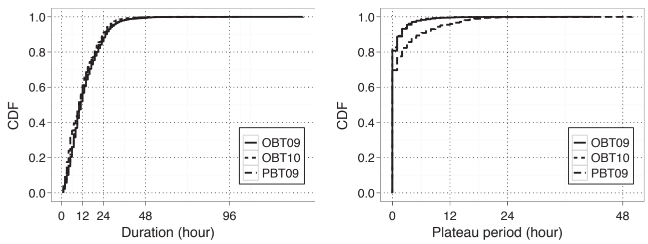

Fig. 8: Left: CDF of the duration of major flashcrowds. Right: CDF of the length of plateau periods of major flashcrowds.

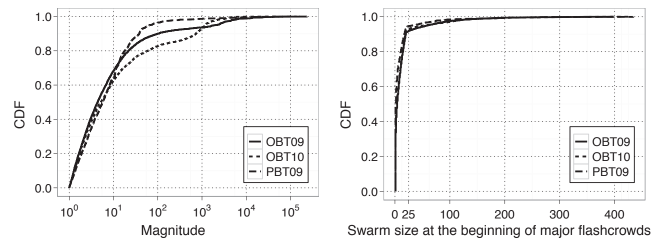

Fig. 9: Left: CDF of the magnitude of major flashcrowds. Right: CDF of the swarm size at the beginning of major flashcrowds.

Modeling results: The Weibull, the LogNormal, and the Gamma distributions provide good fits for the arrival time of major flashcrowds. The LogNormal distribution provides the best fit, for its parameters summarized in Table V.

2) Duration: **Finding: Most major flashcrowds end in one day**. As shown in Figure 8 (left), the distribution of the major flashcrowd duration for all of our datasets are very similar. Nearly 50% of the major flashcrowds end within 12 hours. In contrast, only about 10% of the major flashcrowds last longer than 24 hours. The average duration of major flashcrowds (Table IV) is around 12 hours.

As we have argued earlier, using RSS feeds in BitTorrent can shorten the arrival time of major flashcrowds. However, we believe that this technological change in BitTorrent has much less influence on the duration of major flashcrowds. Although RSS can facilitate users to start downloading new torrents much sooner than the manual approach when BitTorrent clients are running, the time when BitTorrent clients are started is still determined mostly by the daily pattern of user activity [18], [22], [26]. Using RSS in BitTorrent would significantly shorten the flashcrowd duration only when the majority of BitTorrent users will keep their clients or BitTorrent-enabled devices [1] running all the time, which is unlikely to happen soon.

Modeling results: All distributions except the Pareto distribution provide good fits for the duration of major flashcrowds. The LogNormal distribution provides the best fit, for its parameters summarized in Table V.

3) Plateau periods: **Finding: Most major flashcrowds do not have plateau periods.** As shown in Figure 8 (right), 70-80% of the major flashcrowds do not have plateau periods, which means that most swarms start shrinking immediately after their major flashcrowds end. The average length of the plateau periods of major flashcrowds (Table IV) is very short (0.49-1.67 hours). This finding also means that the flashcrowd reported in [18], which has a plateau period of almost 3 days, is not common in BitTorrent.

Similarly to the seed lag (section IV-B), our correlation analysis didn't reveal significant correlations between the plateau period and torrent size. This can be explained by that the plateau period is not determined only by torrent size, but also other factors like seeder capacity.

Modeling results: The Gamma distribution provides the best fit for the length of plateau periods of major flashcrowds, for its parameters summarized in Table V. The D-Statistic of this fitting distributions is large (around 0.20), because the distribution of the length of plateau periods is heavily skewed by the large amount of values equal to zero.

4) Magnitude: As shown in Figure 9 (left), the magnitude is between 1 and 10 for nearly 70% of the major flashcrowds, and is higher than 1,000 for about 5% of the major flashcrowds. We notice that within the upper 70-90% of the distribution, the major flashcrowds in the OBT'10 dataset have considerably higher magnitude than the major flashcrowds in the other two datasets. We find that the major flashcrowds of magnitude in this range have very similar distributions of both the swarm size at the beginning of major flashcrowds and duration, but the increase in the swarm size in such major flashcrowds in the OBT'10 dataset are much larger than those in the other two datasets, which leads to their higher magnitude.

In addition to the duration and the magnitude of major flashcrowds, the swarm size at the beginning of major flashcrowds is also needed to generate synthetic flashcrowds. As shown in Figure 9 (right), the swarm size at the beginning of over 90% of the major flashcrowds is below 25. The average swarm size at the beginning of major flashcrowds (Table IV) is between 9-13, which indicates again that the pre-flashcrowd phases have little contribution to swarm size.

Modeling results: The LogNormal distribution provides the best fit for the magnitude of major flashcrowds, and the Gamma distribution provides the best fit for the swarm size at the beginning of major flashcrowds, for their parameters summarized in Table V. Similarly to the plateau periods, the distribution of the swarm size at the beginning of major flashcrowds is heavily skewed by small values, causing the large D-Statistic of its fitting distribution.

5) Correlations among flashcrowd properties: We have computed the pair-wise correlations between the magnitude, the duration, and the initial swarm size of major flashcrowds, which can be used for generating synthetic flashcrowds. However, we find that there is little correlation between these properties, indicated by the near-zero values for their pair-wise Pearson correlation coefficient.

B. Major Flashcrowds and Swarm Growth

In this section, we answer the question *What is the relationship between major flashcrowds and swarm growth?*

Finding: Many swarms continue to grow after their major flashcrowds end. We first examine the peak lag of the swarms in which the major flashcrowds occur. Previous studies [5], [9], [11], [18] indicate that most swarms stop growing very soon after their (major) flashcrowds end, thus having very short peak lag. However, as shown in Figure 10 (left), less than 60% of the swarms in which the major

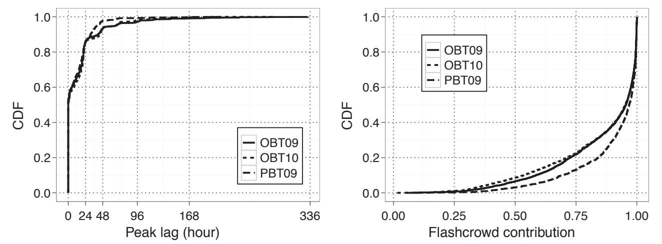

Fig. 10: Left: CDF of the peak lag of swarms in which major flashcrowds occur. Right: CDF of the flashcrowd contribution of major flashcrowds.

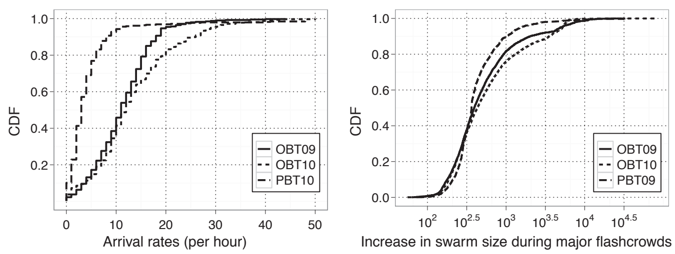

Fig. 11: Left: CDF of the hourly arrival rates of major flashcrowds. Right: CDF of the increase in swarm size of major flashcrowds.

flashcrowds occur have zero-length peak lag, and the peak lag is longer than 12 hours for nearly 40% of such swarms. The average length of the non-zero peak lag is between 24-30 hours for our datasets. We refer readers to our technical report [28] for detailed statistics of the swarm peak lag.

We now examine the flashcrowd contribution of the major flashcrowds. As shown in Figure 10 (right), the flashcrowd contribution is very high (> 0.75) for around 80-90% of the major flashcrowds, and is below 0.5 for only less than 10% of the major flashcrowds. **Finding: most major flashcrowds contribute the majority of peers in the swarms**. We find that for major flashcrowds of low flashcrowd contribution, they are usually followed by multiple minor flashcrowds, which together consist of many peers.

The reason for the existence of long peak lag and low flashcrowd contribution is that a torrent can be published (or mentioned) by not only one, but several websites at different time. A torrent usually receives most attention (major flashcrowd) when it is published at the very first place, and some extra attention is drawn (minor flashcrowds) when that torrent is mentioned later by some other websites.

C. Arrival Rates of Major Flashcrowds

We have studied so far the flashcrowds appearing in individual swarms. However, a BitTorrent tracker may manage millions of individual swarms. In this section, we examine the arrival rates of major flashcrowds at tracker level, and their impact on trackers. To this end, we first calculate the arrival time of major flashcrowds at tracker level, which is equal to the swarm creation time plus the arrival time of the major flashcrowd. Then, we compute their hourly arrival rates.

The distributions of the arrival rates of major flashcrowds in the OBT'09 and OBT'10 datasets, as shown in Figure 11, are very similar. The average arrival rate of major flashcrowds in

Property	Dataset	Min	1st Q	Median	Mean	3rd Q	Max
Arrival rate (flashcrowds /hour)	OBT'09	0	7	11	11.27	15	44
	OBT'10	0	8	12	13.41	18	50
	PBT'09	0	2	3	4.73	5	61
Increase in swarm size	OBT'09	63	255	393	1,034	773	29,390
	OBT'10	56	255	430	1,118	955	81,790
	PBT'09	73	277	363	686	544	34,040

TABLE VI: Statistics of the arrival rates and the increase in swarm size of major flashcrowds.

Property	Datasets	Distribution	D-Statistic	Parameters
Arrival rate (flashcrowds /hour)	OBT'09	Weibull	0.06	$\kappa = 12.99$, $\lambda = 2.00$
	OBT'10	Weibull	0.06	$\kappa = 15.64$, $\lambda = 1.74$
	PBT'09	LogNormal	0.09	$\mu = 1.24$, $\sigma = 0.82$

TABLE VII: Parameters and the D-Statistic of the best fitting probability distributions for the arrival rates of major flashcrowds.

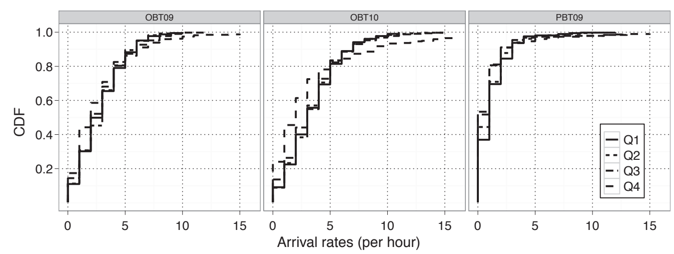

Fig. 12: CDF of the hourly arrival rates of major flashcrowds of different magnitude.

these two datasets is around 12 per hour (Table VI). The major flashcrowds in the PBT'09 dataset have much lower average arrival rate (only 4.73 per hour). This is due to the much smaller size and lower popularity of PublicBitTorrent, compared to OpenBitTorrent. **Modeling results**: The Weibull and LogNormal distributions provide the best fits for the arrival rates of major flashcrowds in the OBT datasets and PBT'09 datasets, for their parameters summarized in Table VII.

To understand if the arrival rate is correlated with the magnitude of major flashcrowds, we examine the arrival rates of major flashcrowds of different magnitude. We first divide the major flashcrowds into four groups (Q1-4) by the lower quartile, median, and upper quartile values of the magnitude distributions. Then, we compute the arrival rates of the major flashcrowds in each group. As shown in Figure 12, the distributions of the arrival rates of major flashcrowds in different groups are similar, except that the major flashcrowds in group Q4 have slightly lower arrival rates than those in other groups.

We define the impact of flashcrowds on trackers as the number of requests sent to trackers during flashcrowd periods. Since our datasets do not include such information, we use the increase in swarm size during flashcrowds to estimate this impact; specifically, we use the the increase in swarm size during major flashcrowds (Table VI) and the arrival rate of the major flashcrowds. **Finding: the major flashcrowds in OpenBitTorrent brought on average 11,000-15,000 peers per hour, and the major flashcrowds in PublicBitTorrent brought on average around 3,250 peers per hour**.

VII. Related work

Related to our study, much research has focused so far on the theoretical modeling of BitTorrent flashcrowds [9], [23],

[27], on the empirical study of BitTorrent swarms [5], [11], [18], and on the performance of non-BitTorrent systems under flashcrowds [3], [7]. Our work complements this large body of related work in three main ways. First, ours is the first large-scale empirical study of BitTorrent flashcrowds. Second, we have investigated algorithms for flashcrowd detection and the impact of flashcrowds on the users of BitTorrent. Third, we have developed a new statistical model for BitTorrent flashcrowds.

Several theoretical models of BitTorrent flashcrowds have been proposed: an age-dependent branching process used to describe the exponential growth of service capacity during flashcrowds [27], a influential model of BitTorrent flashcrowds based on an exponentially decreasing peer-arrival process [9], and an urn-and-ball-based model for system dynamics under flashcrowds [23]. The presence of flashcrowds in BitTorrent swarms has been reported by several measurement studies, from reports based on the study of individual swarms [11], [18] to the analysis of the swarms of two BitTorrent trackers [5].

The effect of flashcrowds on the users of non-BitTorrent systems, such as web servers, has been found to be negative [3], [7]. Baryshnikov [4] studied a simple predictive model for the occurrence of flashcrowds in the workloads of web servers.

VIII. Conclusion

The importance of BitTorrent flashcrowds and their impact on BitTorrent users have received little attention in the past decade, and very little is known about their characteristics. In this paper, we conducted the first comprehensive study of BitTorrent flashcrowds. We studied two million swarms in two of the world's largest BitTorrent trackers. We developed a flashcrowd identification algorithm to identify flashcrowds from our datasets. We found that BitTorrent flashcrowds have a significant negative impact on BitTorrent users. We analyzed and modeled statistically the properties of major flashcrowds in our selected swarms, including arrival time, magnitude, and duration. We also investigated the arrival rates of flashcrowds in the swarms managed by the same tracker. The highlights of our findings are:

1) *Flashcrowds are important*: flashcrowds appear in small fractions (0.3-2%) of swarms but can affect a significant fraction of peers (21-45%).
2) *Flashcrowds arrive rapidly*: Most (70%) major flashcrowds start right after swarm creation.
3) *Flashcrowds are short*: the average duration of the major flashcrowd is around 12 hours.

Our findings can be used to generate synthetic yet realistic flashcrowds for simulation studies. They may be used in the future for real-system tuning and for improving the operation of BitTorrent through the design of flashcrowd mitigation mechanisms. In particular, we are currently trying to apply these findings to improve our operational BitTorrent system Tribler [19].

Acknowledgements

The research leading to this contribution has received funding from the EU FP7 P2P-Next project under grant agreement no. 216217.

References

[1] D-Link DNS-323 Network Storage.
[2] Ipoque Internet Studies. www.ipoque.com/resources/internet-studies/.
[3] I. Ari, B. Hong, E. L. Miller, S. A. Brandt, and D. D. E. Long. Managing flash crowds on the internet. In *IEEE/ACM MASCOTS*, 2003.
[4] Y. Baryshnikov, E. G. Coffman, G. Pierre, D. Rubenstein, M. Squillante, and T. Yimwadsana. Predictability of web-server traffic congestion. In *IEEE WCW*, 2005.
[5] A. Bellissimo, B. N. Levine, and P. Shenoy. Exploring the use of bittorrent as the basis for a large trace repository. Technical Report 04-41, University of Massachusetts Amherst, 2004.
[6] A. R. Bharambe, C. Herley, and V. N. Padmanabhan. Analyzing and improving a bittorrent networks performance mechanisms. In *IEEE INFOCOM*, 2006.
[7] P. Bodik, A. Fox, M. J. Franklin, M. I. Jordan, and D. A. Patterson. Characterizing, modeling, and generating workload spikes for stateful services. In *ACM SOCC*, 2010.
[8] J. Elson and J. Howell. Handling flash crowds from your garage. In *USENIX ATC*, 2008.
[9] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of BitTorrent-like systems. In *ACM IMC*, 2005.
[10] A. Iosup, P. Garbacki, J. Pouwelse, and D. Epema. Correlating topology and path characteristics of overlay networks and the internet. In *IEEE CCGRID*, 2006.
[11] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, Al, and L. Garcés-Erice. Dissecting BitTorrent: Five months in a torrent's lifetime. In *PAM*. 2004.
[12] S. Kaune, J. Stolzenburg, A. Kovacevic, and R. Steinmetz. Understanding bittorrent?s suitability in various applications and environments. In *ICCGI*, 2008.
[13] D. Kondo, B. Javadi, A. Iosup, and D. Epema. The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems. In *IEEE/ACM CCGrid*, 2010.
[14] G. Linden. Make data useful, 2006.
[15] M. Meulpolder, L. D'Acunto, M. Capotă, M. Wojciechowski, J. A. Pouwelse, D. H. J. Epema, and H. J. Sips. Public and private bittorrent communities: a measurement study. In *IPTPS*, 2010.
[16] L. Niven. *The Flight of the Horse*. Ballantine Books, 1973.
[17] D. Nurmi, J. Brevik, and R. Wolski. Modeling machine availability in enterprise and wide-area distributed computing environments. In *Euro-Par*, 2005.
[18] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips. The bittorrent p2p file-sharing system: Measurements and analysis. In *IPTPS*, 2005.
[19] J. A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, M. R. van Steen, and H. J. Sips. Tribler: a social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 2008.
[20] D. Qiu and R. Srikant. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *ACM SIGCOMM*, 2004.
[21] M. Rabinovich, R. Irina, R. Rajaraman, and A. Aggarwal. A dynamic object replication and migration protocol for an internet hosting service. In *ICDCS*, 1999.
[22] S. Saroiu, K. P. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *MMCN*, 2002.
[23] F. Simatos, P. Robert, and F. Guillemin. A queueing system for modeling a file sharing principle. In *ACM SIGMETRICS*, 2008.
[24] G. Urvoy-keller and P. Michiardi. Impact of inner parameters and overlay structure on the performance of bittorrent. In *Global Internet Symposium*, 2006.
[25] Wikipedia. Slashdot effect. http://en.wikipedia.org/wiki/Slashdot_effect.
[26] M. Wojciechowski, M. Capotâ, J. Pouwelse, and A. Iosup. Btworld: towards observing the global bittorrent file-sharing network. In *ACM HPDC*, 2010.
[27] X. Yang and G. de Veciana. Service capacity of peer to peer networks. In *IEEE INFOCOM*, 2004.
[28] B. Zhang, A. Iosup, J. Pouwelse, and D. Epema. Identifying, analyzing, and modeling flashcrowds in bittorrent. Technical Report PDS-2010-009, Delft University of Technology, 2010.
[29] B. Zhang, A. Iosup, J. Pouwelse, and D. Epema. The peer-to-peer trace archive: design and comparative trace analysis. In *ACM CoNEXT Student Workshop*, 2010.
[30] B. Zhang, A. Iosup, J. Pouwelse, D. Epema, and H. Sips. Sampling bias in bittorrent measurements. In *Euro-Par*, 2010.
[31] D. Zotkin and P. Keleher. Sloppiness as a virtue: Job-length estimation and performance in backfilling schedulers. In *HPDC*, 1999.