The BitTorrent Peer Collector Problem

Andri Lareida
CSG@IfI
University of Zurich
lareida@ifi.uzh.ch

Tobias Hoßfeld
University of Duisburg-Essen
tobias.hossfeld@uni-due.de

Burkhard Stiller
CSG@IfI
University of Zurich
stiller@ifi.uzh.ch

***Abstract*—Peer-to-Peer (P2P) systems measurements are still a relevant research topic, since insights in large swarm sizes and churn are not yet available for the BitTorrent network. To improve existing measurement methodology, this work here tackles the aspect of swarm size estimation and complete collection in the BitTorrent network. For this purpose the Coupon Collector Problem is modified and formulated as the BitTorrent Peer Collector (BTPC) Problem. Thus, (a) simulations are used to test simple and maximum likelihood estimation for hidden swarm sizes, (b) an analytical solution to the BTPC problem is presented, and (c) measurements are used to evaluate estimators of the BTPC model. Obtained results show that this estimation works well for classical trackers and that churn constantly influences measurements. Those results show that more peers use the Mainline DHT than a single tracker, however, client implementations challenge those models working well for trackers.**

I. Introduction

Various Peer-to-Peer (P2P) system measurements were conducted in the past two decades, typically with one specific research goal in mind. Therefore measurement methodologies and collected data sets as such received little attention in the respective publications. Often, assumptions are made which are, although reasonably chosen, not confirmed or falsified in the course of a measurement study. One example is [5] where the authors claim that BitTorrent (BT) swarms, *i.e.,* all peers sharing the same content, are stable over the course of a few hours and that, therefore, it is valid to consider results gathered over the course of 90 minutes as concurrent. However, other measurements [13] indicate that BT swarms constantly exhibit strong user fluctuations, contradicting the stable swarm assumption at least during some hours of the day. Such discrepancies between measurements indicate that P2P measurements in general and specifically in the BitTorrent network are not well enough understood and that research on BT measurement methodologies is still essential, as P2P technology is still being developed [9], [18] or being used for market research [14]. Furthermore, with the narrow scopes of those measurements conducted in the past, the resulting data sets are hardly re-usable. With the vision of creating a general large scale measurement system for BT it becomes even more important that those measurements are accurate and well understood.

This work here investigates the question of how many peers are sharing a file at the same time, *i.e.,* how big is a swarm? While this question is trivial, the answer is not. Due to BT's distributed architecture with many trackers, it is very unlikely that one tracker knows all peers being active at any point in time. Furthermore, two Distributed Hash Tables (DHT), of which the Mainline DHT is most used, with an abundance of different client implementations are actively used. Owing to the random tracker and DHT responses, collecting all peers from trackers or DHTs take time, during which the state of a swarm is changing due to churn. Thus, the longer a measurement is taking, the more inaccuracies are introduced by churn. Since the collection of all peers from one or multiple trackers is a modified version of the Coupon Collector Problem [22], the newly termed BitTorrent Peer Collector (BTPC) is introduced here. This new approach contributes a novel method to estimate swarm sizes based on tracker responses and on DHT time series, a data set which contains these responses for one file over 24 hours, and an analysis of the impact of churn on those measurements and estimations.

The remainder of this paper is organized as follows. While Sec. II discusses related work, Sec. III presents an in-depth problem analysis and its formalization. The measurement results and their interpretation under this formalization are contained in Sec. IV, before Sec. V draws conclusions and indicates next steps ahead.

II. Related Work

The generalized Coupon Collector Problem (CCP) is well investigated [22]. It can be formulated as: "Given that there are N different coupons available in boxes of a certain product, what is the probability that after buying m such boxes, one will have collected exactly i different coupons?" [11]. Also, the variant with k coupons being drawn at the same time, *i.e.,* one box containing a set of distinct coupons, is solved [19]. However, in case of BT the problem is still different as the number of peers returned by the DHT, *i.e.,* k, varies. Efforts to find analytical solutions for BT have been made [3], but the results are not generally formulated and not evaluated.

BT measurements have been conducted for many years [7], thus, general techniques are well understood [10] and measurement types are classified. [10] identified two main classes of measurements: microscopic and macroscopic. The microscopic class focuses on a small number of swarms but in great detail, like [6], which explored the connections between peers in a swarm. The macroscopic class includes a large number of swarms but with less details, such as [5][16]. [5] collected IP (Internet Protocol) addresses of peers from trackers to investigate locality. However, the BT ecosystem keeps changing, requiring adapted measurement systems and models, *e.g.,* including DHT in peer collection.

The term "churn" describes the changes observed in a P2P system due to peers joining and leaving the system and is an integral part of P2P systems. To join a BT swarm a peer needs to announce itself to a tracker, from that moment the peer is visible to others. Leaving a system can be done in a friendly way by un-registering with the tracker or by just disappearing.

TABLE I. RELATED WORK OVERVIEW.

Reference	Coupon Collector	BT Meas.	Churn
[22] [19]	✓	×	×
[6] [5] [8] [13]	×	✓	×
[20]	×	✓	✓
[2]	×	×	✓

Thus, churn has 2 components the join and leave rate. Churn models derived from P2P measurements typically provide a session or inter-arrival time distribution [20]. [20] points out that time of day does have a critical effect on churn, which is supported by results in [13]. [20] focused only on Linux distributions torrents which is a special user group that may be different from general file-sharing users. For peers it is possible to estimate churn in their neighborhood of the network [2]. Further methods to estimate churn based on time series are available [2], [21]. However, the time series constituting the basis for these measurements are not currently available.

Finally, the overview over the related work is given in Table I, showing that the combination of churn, Coupon Collector, and BT measurements is missing so far. However, to quantify the accuracy of BT measurements such a view of the problem is critical. Therefore, this work investigates methods to obtain swarm size estimates, forming the basis for collecting complete swarms in a short time to obtain the required time series to estimate churn.

III. PROBLEM ANALYSIS

What is the size of a swarm? The answer to this question is the key to collect all its peers, and is similar to the reverse CCP [11]. Trackers include the number of seeders and leechers of a swarm within their responses to announce queries. However, a tracker only knows the peers that announced with it and since there are typically multiple trackers used for one torrent it cannot be assumed that one tracker knows all peers in a swarm. Furthermore, some peers might not even use a tracker and rely solely on the two DHTs for peer discovery. The DHTs do not provide swarm size or seeder-leecher ratio. Therefore, to answer the question the only option is to estimate how many peers are in a swarm. To solve the BTPC problem it needs to be investigated how many peers need to be collected from trackers and DHT to collect the whole swarm. The difference to the classical Coupon Collector is the response size which is 1 in the standard Coupon Collector as opposed to the case of BT, where the response size is typically larger than 50 and can be heterogeneous. Table II summarizes the notation used in the remainder of this work.

A. The BitTorrent Peer Collector

In the general CCP [22] the goal is to find all coupons from a set of coupons by drawing one coupon at a time randomly. The distribution of coupons is not uniform and therefore the probability of drawing a coupon depends on the type of coupon.

To collect all peer addresses – coupons – of a swarm the collector has to query a tracker to receive a set of peer addresses – draw – until all addresses are collected. How many times does the collector need to query the tracker to collect a complete swarm? This question is termed the BT Coupon Collector Problem (BTPC) and it defers from the general instance [22] in the number, k, of distinct coupons, *i.e.,* peer addresses, in one draw. A tracker has a set of IP addresses of size N from which it randomly chooses 50 addresses to return [4]. Since every address is in N only once and the selection of k addresses is random, the distribution is uniform as in the basic CCP. A tracker response contains up to $k = 50$ unique peer addresses for each request, in case of DHT queries k can be larger than 1,000. Thus, a response is equivalent to the random combination of N choose k (binomial coefficient). Meaning that tracker responses can be accurately modeled by randomly selecting k unique addresses from N. For the remainder of this paper the number of unique peers collected after i draws shall be denoted M_i, the duplicates contained within response i by d_i, and the total number of peers collected Y_i.

TABLE II. NOTATION

N	Real swarm size, *i.e.,* ground truth
N^*	Estimated swarm size
k	Number of peers in a response
k_{rel}	Response size relative to swarm size
Δk	Time required for one request
M	Number of unique peers collected
M^*	Predicted unique peers collected
Y	Number of total peers collected
i	Number of queries
d_i	Duplicates in response i
λ	The join rate of peers per second

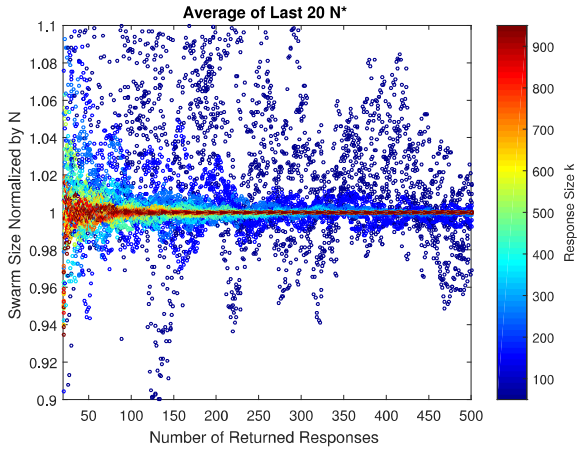
B. Simple Estimation

The simplest approach for estimating a swarm's size is to look at duplicates contained in responses. Since the peers in the response are uniformly randomly distributed the ratio of duplicates to response size is, on average, the same as the ratio of discovered peers to swarm size, *i.e.,* $\frac{M_{i-1}}{N_i^*} = \frac{d_i}{k}$. Therefore, an estimation N^*_{simple} can be made with each response after the second response is received (because $M_0 = 0$ and $d_1 = 0$) by solving for N^*_{simple} as in Equ. 1.

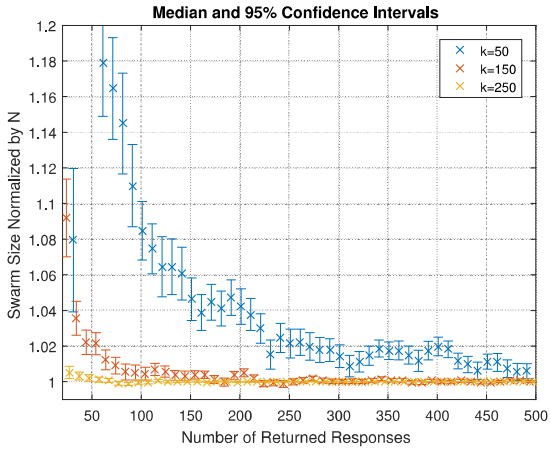
$$N_i^* = M_{i-1}\frac{k}{d_i} \tag{1}$$

Fig. 1 shows simulation results for swarm size of $N = 20,000$ chosen to be comparable to the swarm measured and analyzed in Section IV. The response size was varied between $k = 50$ and $k = 1'000$ in steps of 100 and each k was run a 100 times. The estimate, N^*_{simple}, has been normalized by the swarm size N to center the plots around 1. To receive better and more consistent results the moving average of the last 20 estimates was taken, for this reason the plots start at 20 responses. Fig. 1a illustrates the estimates of 4 runs for the different ks showing that the accuracy increases with increasing response size and increasing number of responses. Even with the smallest k accurate estimates can be made for this swarm after 200 requests. With increasing response size, k, the accuracy of the estimate increases.

Fig. 1b shows the median and the 95% confidence intervals for 100 runs of the simulation. The three smallest ks

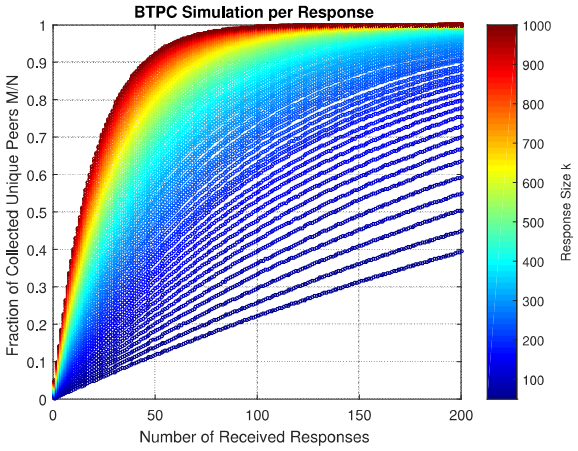


(a) Illustration of the estimates with 4 samples.

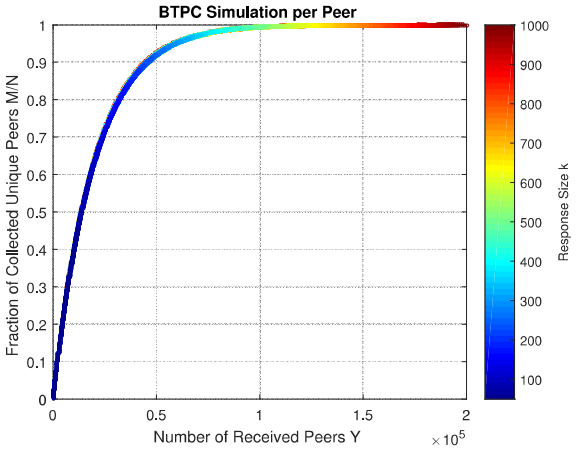


(b) Median and 95% confidence intervals from 100 runs.

Fig. 1. Swarm size estimates for a swarm size of 20,000 with response sizes between 50 and 1000.



(a) After X responses have been received.



(b) After X peers have been responded.

Fig. 2. Portion of the swarm of size $N = 20'000$ discovered per requests and per peers.

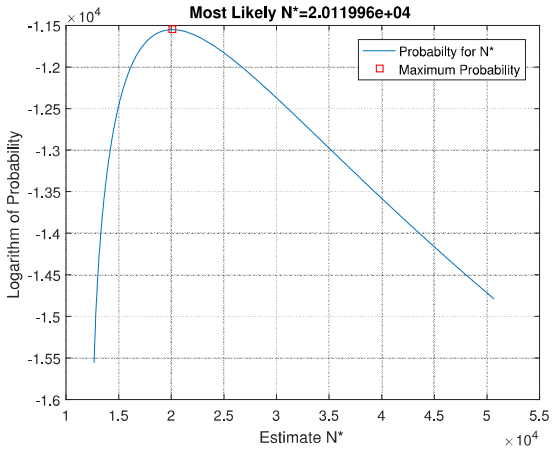
are shown since those are the least accurate. The statistical analysis confirms that the simple estimate converges to N with increasing number of responses received and it converges faster with larger response size. All N^*_{simple} approach the real swarm size from above, thus, the simple estimate has a tendency to overestimate the swarm size.

Furthermore, the simulation allows to investigate what portion of a swarm has been collected after i responses have been received. Fig. 2a shows the portion of the swarm discovered after X responses of size k have been received where k is varied from 50 to 1,000. As expected, with larger response sizes more unique peers are discovered than with the same amount of smaller responses. More importantly, the shape of the curves indicate that the fraction of unique collected peers asymptotically approaches 100%. That means the more peers from a swarm are collected the more difficult it becomes to collect new peers, as in the CCP. Therefore, it will be very difficult to collect all peers of a large swarm. However, discovering a large part of a swarm, *e.g.,* 95%, seems to be feasible.

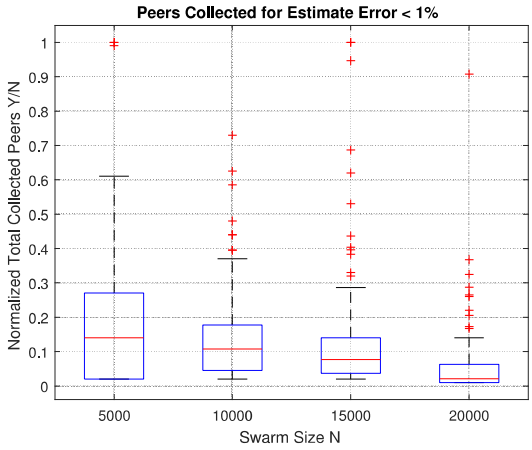
Fig. 2b presents the same simulation data as Fig. 2a the only difference being the x-axis, which has been changed to show returned peers. While Fig. 2a shows the number of responses received, Fig. 2b shows the cumulative sum of peers returned by all the responses, or $k \cdot \#responses$. The fact that all the points lie on the same trajectory indicates that the size of a response k does not influence the number of unique peers found for the number of peers received, at least for large swarms. This observation indicates that if the k is significantly smaller than the swarm size N, k does not have a measurable influence on the discovery rate of unique peers. To get accurate results instantaneously several requests can be sent in parallel using multiple machines if necessary.

C. Maximum Likelihood Estimation

A more general and accurate solution to swarm size estimation can be achieved with a Maximum Likelihood Estimator (MLE), which is one solution to the reverse CCP [11]. An MLE calculates the probability at each step i for a range of possible swarm sizes N, the swarm size with the highest



(a) Illustration of an MLE.



(b) MLE comparison with boxes showing median, 25th and 75th percentile, and outliers.

Fig. 3. Illustration of an MLE calculation and comparison of accuracy with different swarm sizes.

probability will become the estimate $N*_{MLE}$. This way, response size k can be ignored and each returned peer address is treated as a single observation equivalent to $k = 1$. Therefore, a sequence of peers x is observed, *e.g.,* $x = [1, 2, 3, 4]$. The probability, q_i, to observe a new i-th peer is the number of undiscovered peers divided by the swarm size N:$q_i = \frac{N - M_{i-1}}{N}$. Vice-versa the probability, $1 - q_i$, to observe a duplicate i-th peer is the number of discovered peers divided by the swarm size N: $1 - q_i = \frac{M_{i-1}}{N}$ Using both formulas the probability $P(N|x)$ to observe x for a given N can be expressed like:

$$P(N|x) = \prod_{i=1}^{Y} p_i \tag{2}$$

$$with \quad p_i = \begin{cases} q_i, & \text{if i-th peer is new} \\ 1 - q_i, & \text{if i-th peer is duplicate} \end{cases}$$

It is $Y = |x|$ the total number of returned peers.

It is $M_0 = 0$ and $M_1 = 1$

Equ. 2 can be used as an MLE by finding the maximum probability $P(N|x)$ for a pattern x indicating the most likely estimate N^*_{MLE}:

$$N^* = \max_{N=|M|:\infty} P(N|x) \tag{3}$$

After Y_i peers have been collected, possibilities for several N^* larger than M_i is calculated. The N^* with the largest probability is selected as the most likely N and denoted N^*_{MLE}. Fig. 3a illustrates such an estimation for different N^*.

Fig. 3b shows a comparison of MLE swarm size estimation results which are in the range of $\pm 1\%$ of the real N. For numerical reasons, the log-likelihood is used to give a more robust numerical evaluation, *e.g.,* for large Ns. The box plot shows median, 25th and 75th percentile, and the outliers of the number of peers collected, Y, divided by the swarm size, N, of the first values that estimated $N^* = \pm 1\%$ of N. First, the plot shows that for larger swarms a smaller fraction of collected peers Y is required to get an accurate measurement. This implies that the MLE is more dependent on collected peers Y than on swarm size N. As a general rule, 4,000 peers or more need to be collected to get accurate estimates. In a practical implementation the accuracy also depends on the range and resolution of the Ns selected to calculate the probabilities.

D. Analytical Collector

An analytical solution to the BTPC problem is preferable as it can be used without much overhead to decide when to stop querying a tracker or to determine how many queries have to be executed to collect a certain fraction of a swarm. Therefore, k is expressed relative to N as $k_{rel} = k/N$ which is the fraction of the swarm returned within each response. As N is always bigger than or equal to k and k is not zero the range of k_{rel} is $(0, 1]$. Thus, with each response i the number of collected peers M_{i-1} grows by the pool of collected peers $N - M_{i-1}$ times the relative response size, *i.e.,* Equ. 4

$$M^*_i = M_{i-1} + (N - M_{i-1}) \cdot k_{rel} \tag{4}$$

This formula is simpler than using simulation data but still not elegant since it is iterative and, thus, hard to compute for large is. To simplify things one can look at the number of a swarm's not collected peers which will decrease with the rate $r = 1 - k_{rel}$. With each response received the number of unknown peers decreases as in Equ. 5.

$$N - M_i = N \cdot r^i \tag{5}$$

To obtain the number of collected peers, the expression can be subtracted from 1, and r can be substituted with $1 - k_{rel}$ which gives the formula for the fraction of peers collected after the i-th response of size k has been received, *i.e.,* Equ. 6. $1 - (1 - k_{rel})^i$

$$M^*_{rel} = 1 - (1 - k_{rel})^x \tag{6}$$

This formula does produce a result relative to N, if the absolute number is desired the result has to be multiplied by N. The calculated M^* can be compared to the simulated M

to determine the goodness of fit of the model by calculating the coefficient of determination (R^2). Applying R-squared to a simulation with 50 ks ranging from 1 to 1'000 the mean of R^2 is $\bar{R}^2 \approx 0.999987$ for all the ks simulated. That is an almost complete fit and means that 99.9987% of the variance in the model can be explained by the model. The advantage of this model is that it can be used to coordinate multiple distributed collectors. The number of collectors is x and k_{rel} is the fraction of the swarm that a collector has collected. As long as each collector behaves the same, each can decide when the collection is complete.

IV. Measurement and Results

To evaluate those concepts established in Section III and to investigate the impact of churn on actual measurements a measurement series was conducted. The measurement, consisting of one torrent with the Mainline DHT and all trackers found in the corresponding meta data file, was executed for 25 hours from May 9, 2016 08:05 GMT. The measurement was conducted on a single machine to introduce as little noise into the data as possible. The data set was acquired by sending one announce request per second, $\Delta k = 1s$, to each of the 4 responding trackers in the torrent "Deadpool 2016 1080p BluRay x264 DTS-JYK" and the DHT. Although, it has been claimed that trackers block or ban clients that send requests too frequently, no such effect could be observed in this measurement. The torrent was chosen, because it ranked highest among movie based torrents at the time of measurement. The data contains a time stamp, IP addresses, seeder, leecher, and total peer count data for tracker responses. The data set is available for download at [12]. For ethical reasons IP addresses contained in the data set were anonymized to prevent the identification of individual users while still maintaining the uniqueness of IP addresses for a detailed analysis.

A. Quantifying Churn

The simulations in Sec. III did not account for churn, thus, the difference between the measurement and the simulations can be either accounted to churn or wrong assumptions. A BT measurement over some period of time, such as those 24 hours, will inevitably be biased by churn. Thus, it is of key importance to quantify that bias for the conducted measurement.

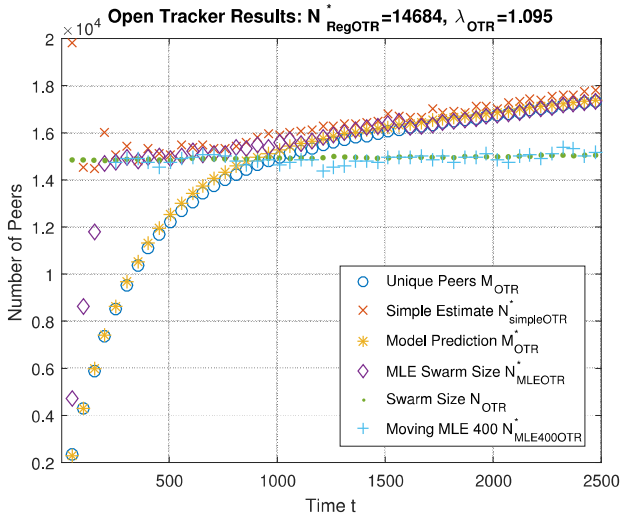
To evaluate the tracker case, responses from the tracker with the largest swarm size during the measurement period are used. In this case this was the Open Trackr (OTR) [15] which initially reported a swarm size of $14,877$. Fig. 4a depicts the first 2,500 OTR replies, which equals the first 2,500 seconds of the measurement. The circles show M_{OTR}, the unique peers collected from OTR. At a first glance the pattern seems to be as expected from the simulations in Sec. III. However, the circles surpass the swarm size announced by the tracker N_{OTR}, what should not be possible since collecting more unique peers than the swarm size is not possible. Since peers are constantly joining and leaving the swarm M_{OTR} will contain peers that have already left, which is not possible to filter out with the current measurement as very accurate snapshots of the swarm would be required. The crosses represent the simple estimates, $N^*_{simpleOTR}$, and the diamonds the MLE, $N^*_{simpleMLE}$. After being close to the reported swarm size between 250 s and 500 s both follow a curve close to M_{OTR}.

The increase of M_{OTR} and both estimates is a result of peers joining the swarm over time, which will be collected and added to M unique peers. Leaving peers that were already collected do not influence this result anymore since they are not removed from M. Only those leaving before being collected might slightly reduce estimates in the beginning since M_{OTR} is slightly smaller than expected, but this effect is countered by the joining peers. Therefore, for measurements the main concern are the peers joining per second which is called the join rate and denoted λ. Since the increase of M_{OTR} is almost linear for $3,000s < t < 5,000s$, λ_{OTR} is approximately constant and linear regression can be applied to estimate λ_{OTR}, *i.e.,* the slope of the curve. This is only valid for a short period of time and does not mean the join rate is constant at all times. The y-intercept of the regression gives another estimate of the swarm size N^*_{RegOTR} at $t = 0$. In this case the slope was $\lambda_{OTR} = 1.095$, meaning that 1.095 peers join the swarm per second. The intercept was at $N^*_m RegOTR = 14,684$, meaning that there were about 14,700 peers in the swarm at the start of measurement, which is very close to the swarm size reported by the tracker $N_{OTR} = 14,877$ being 1.3% off. Finally, the join rate λ can be included in the model from Equ. 6 to analytically calculate M^*_t with Equ. 7, the unique peers at time t. The result of applying the revised model is shown in Fig. 4a as asterisks. It does not reach an exact match to M_{OTR}, but a very close one. Note, that this model is time dependent as λ is time dependent.

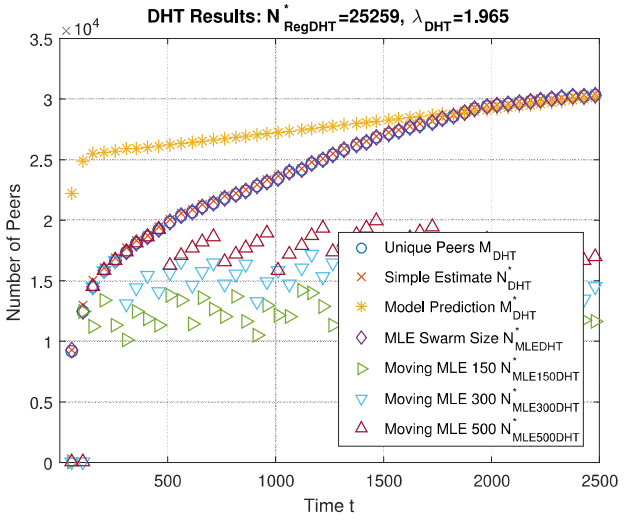
$$M^*_t = (\lambda t + N) \cdot (1 - (1 - \frac{k}{N})^{\frac{t}{\Delta k}}) \tag{7}$$

As these MLE results are accurate for a period of time Δt in which churn does not have a noticeable influence on N^*_{MLEOTR}. Section III revealed that with 30% of N collected $Y \geq N \cdot 30\%$, in 75% of the cases MLE estimates are accurate to 1%. Therefore, accurate estimates can be expected for $Y \geq 5,000 peers$ which translates to 100 received responses or $\Delta t_{min} = 100s$ of measurements. Δt can be reduced by sending queries faster or by using a distributed measurement sending queries simultaneously. Fig. 4a shows $N^*_{MLE400OTR}$ for $200s < \Delta t < 400s$ is plotted, twice Δt_{min} to receive smoother results and four times Δt_{min} as the upper bound to reduce calculation overhead. These $N^*_{MLE400OTR}$ results follow N_{OTR} with very small deviation. The relative error E_{M_i} introduced by λ_{OTR} for $t_{max} = 400$ can be calculated by subtracting the predicted value without churn $M*_{OTR400}$, Equ. 6, from the real M_{OTR_400} and dividing by M_{OTR400}, which amounts to $E_{M_{OTR_{400}}} = 1\%$. Based on Eq. 7, the relative error between our measurements and the model can be derived, allowing to derive the required time span Δt such that the relative error is $< \epsilon$ with probability p for given λ. The main problem is that the churn rate λ needs to be accurately determined. This can either be done with linear regression as presented or by applying more sophisticated methods for estimating churn in overlay networks.

Fig. 4b shows the respective DHT results, lacking the swarm size due to the swarm size information not being available in the DHT. As expected, due to the larger response size M_{DHT} increases faster than M_{OTR} in the beginning, but also the $\lambda_{DHT} = 1.965$ and $N*_{RegDHT} = 25,259$ estimated



(a) Results for OTR.



(b) Results for Mainline DHT.

Fig. 4. The first 2,500 s of measurements for Mainline DHT and Open Tracker (OTR).

by linear regression are higher. This reads as, that more peers use the DHT than the OTR and that over-proportionally more peers join the DHT than the OTR. This is due to the fact that there is only one official DHT, while there are multiple trackers, *i.e.,* 4 in this case. Also, censorship and Internet blockades can have an influence. An effort of fitting the model from Equ. 7 to the DHT results in a bad fit to the actual M_{DHT} as the discovery of unique peers is not as fast as expected. This is due to non-random DHT responses. Accordingly, moving MLEs do not work for the DHT case. Fig. 4b shows MLEs calculated in the same fashion as for the OTR data. The larger Δt the larger the estimate becomes. A reason for this observation is that DHT tracker implementations [1] specify a time to live of 5 minutes on address entries in the DHT (src/kademlia/dht_tracker.cpp line 65), but the client refreshes its address only every 15 minutes (src/session.cpp line 1341). Thus, at any point only the peers that refreshed in the last 5 minutes are contained in the DHT, which amounts to 1 third on average if peers refresh only every 15 minutes. One possible practical solution is to use multiple measurement nodes in parallel to reduce the time needed to collect peers and measure for at least 15 minutes, thus, the effects of churn on peer collection are reduced to this 15 minutes window.

The presented results of this work show that the BTPC Problem introduces an error into measurements which can be quantified by comparing the theoretical model in Equ. 6 to the measured M. With evaluated models it becomes possible to estimate swarm size and calculate the number of requests required to collect the swarm. Those requests can be executed in parallel and churn can be neglected.

V. Conclusions and Future Work

This work showed that churn is constantly influencing measurements in P2P systems. In case of BitTorrent this is not an issue with small swarms, but with larger ones, where collecting all peers becomes a time consuming challenge due to the BitTorrent Peer Collector (BTPC) Problem. With the models presented here it is possible to quantify those effects of churn or to mitigate them by issuing a sufficient number of parallel requests. The simulations presented in Section III show that a factor of 5 of returned peers to swarm size is sufficient to collect a swarm of size 20,000. This means that with 10 parallel requests and 1s per request, as in the measurement, and 50 returned peers per request, 200 seconds is enough to collect such a swarm, limiting the effects of churn on the measurement. The measurements presented in Section IV show that a 400s window is small enough for MLE to provide accurate results.

Therefore, these results presented confirm that the proposed estimators and model can deliver accurate results for classical trackers under practical and realistic circumstances. More work remains with regard to the Mainline DHT results which showed unexpected behavior. It needs to be investigated whether the Mainline DHT returns peers randomly in this 5 minutes window and if there are more differences among popular implementations. The presented model within this paper can be adapted to take such a bias into account.

Furthermore, the effects and characteristics of churn this work presents are merely a first step. The collected data set represents one example of a swarm where there are many more. To further validate the models presented a large scale data set needs to be applied to it. Furthermore, analyzing churn itself and the impact external factors, such as location and time of day, have on churn rates and user behavior, *e.g.,* [4], [17] is essential.

Finally, the BTPC work shows that those gaps observed and partially closed in BitTorrent research still require efforts to fully achieve accurate measurements for more cases, as these are essential to improve BT operations, traffic optimizations, and its energy efficiency.

Acknowledgments

This work was supported by the FLAMINGO project funded by the EU FP7 Program under Contract No. FP7-2012-ICT-318488.

REFERENCES

[1] arvidn. libtorrent. https://sourceforge.net/p/libtorrent/code/HEAD/tree/tags/libtorrent_0_16_6/, June 2016. [Online, accessed 2016-11-29].

[2] A. Binzenhöfer and K. Leibnitz. Estimating Churn in Structured P2P Networks. *Managing Traffic Performance in Converged Networks*, pp 630–641. Springer, 2007.

[3] X. Chen, K. Lin, B. Wang, and Z. Yang. Active Measurements on BitTorrent and eMule Ecosystem over the Internet. *2nd International Conference on Consumer Electronics, Communications and Networks (CECNet 2012)*, Yichang, China, April 2012.

[4] B. Cohen. Incentives Build Robustness in BitTorrent. *1st Workshop on Economics of Peer-to-Peer Systems (P2PECON 2003)*, Berkeley, CA, USA, June 2003.

[5] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez. Deep Diving into BitTorrent Locality. *IEEE INFOCOM 2011*, Shanghai, China, April 2011.

[6] C. Decker, R. Eidenbenz, and R. Wattenhofer. Exploring and Improving BitTorrent Topologies. *IEEE Thirteenth International Conference on Peer-to-Peer Computing (P2P 2013)*, Trento, Italy, September 2013.

[7] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, Analysis, and Modeling of BitTorrent-like Systems. *5th ACM SIGCOMM Conference on Internet Measurement (IMC 2005)*, Berkeley, CA, USA, October 2005.

[8] T. Hoßfeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel. Characterization of BitTorrent Swarms and their Distribution in the Internet. *Computer Networks*, 55(5), April 2011.

[9] Katherine Noyes. Is Dropbox Planning a P2P Option? New Patent Suggests it's Looking Beyond the Cloud. http://www.pcworld.com/article/3018983/data-center-cloud/is-dropbox-planning-a-p2p-option-new-patent-suggests-its-looking-beyond-the-cloud.html, Jan 2016. [Online, accessed 2016-11-25].

[10] M. Kryczka, R. Cuevas, C. Guerrero, A. Azcorra, and A. Cuevas. Measuring the BitTorrent Ecosystem: Techniques, Tips, and Tricks. *IEEE Communications Magazine*, 49(9), September 2011.

[11] E. Langford and R. Langford. Solution of the Inverse Coupon Collector's Problem. *The Mathematical Scientist*, 27(2), December 2002. [Online, accessed 2016-11-25].

[12] A. Lareida, T. Hoßfeld, and B. Stiller. The BitTorrent Peer Collector Data Set. http://www.csg.uzh.ch/publications/data/peercollector.html. [Online, accessed 2016-05-10].

[13] A. Lareida, S. Schrepfer, T. Bocek, and B. Stiller. Overlay Network Measurements with Distribution Evolution and Geographical Visualization. *2016 IEEE Network Operations and Management Symposium (NOMS 2016)*, Istanbul, Turkey, April 2016.

[14] Nick Rego. Exclusive: NetflixâĂŹs CEO Talks Competition, VPNs, and the Future of Streaming. http://www.tbreak.com/exclusive-netflixs-ceo-talks-competition-vpns-and-the-future-of-streaming, April 2016. [Online, accessed 2016-11-25].

[15] OpenTrackr. Free to use Torrent tracker! http://opentrackr.org/. [Online, accessed 2016-05-12].

[16] J. S. Otto, M. A. Sánchez, D. R. Choffnes, F. E. Bustamante, and G. Siganos. On Blind Mice and the Elephant: Understanding the Network Impact of a Large Distributed System. *ACM SIGCOMM 2011*, Toronto, Ontario, Canada, 2011.

[17] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do incentives build robustness in bittorrent. *4th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2007)*, volume 7, 2007.

[18] Ryan Lawler. Akamai to Launch P2P-Based Streaming Video Client. https://gigaom.com/2011/04/15/akamai-p2p-streaming-video/, April 2011. [Online, accessed 2016-11-25].

[19] W. Stadje. The Collector's Problem with Group Drawings. *Advances in Applied Probability*, 1990.

[20] D. Stutzbach and R. Rejaie. Understanding Churn in Peer-to-peer Networks. *6th ACM SIGCOMM Conference on Internet Measurement (IMC 2006)*, Rio de Janeriro, Brazil, October 2006.

[21] X. Wang, Z. Yao, and D. Loguinov. Residual-Based Estimation of Peer and Link Lifetimes in P2P Networks. *IEEE/ACM Transactions on Networking (TON)*, 17(3):726–739, 2009.

[22] W. Xu and A. K. Tang. A Generalized Coupon Collector Problem. *Journal of Applied Probability*, 48(4), December 2011.