Modeling Chunk Availability in P2P Swarming Systems

Daniel Sadoc Menasché
University of Massachusetts

Antonio A.A. Rocha
Federal Univ. of Rio de Janeiro

Edmundo de Souza e Silva
Federal Univ. of Rio de Janeiro

Rosa M. Meri Leão
Federal Univ. of Rio de Janeiro

Don Towsley
University of Massachusetts

Arun Venkataramani
University of Massachusetts

1. INTRODUCTION

Peer-to-peer swarming systems à la BitTorrent are usually deployed for the dissemination of popular content. Popular content naturally gets highly replicated in the network and capacity scales with demand ensuring high performance for peers requesting popular content.

Nevertheless, the behavior of swarming systems in the face of unpopular content and small populations of users also deserves attention. First, it is important to understand what is the popularity threshold above which the use of swarming systems is most beneficial for a publisher. The second reason is economic. With the monetization of BitTorrent clients such as Vuze (previously known as Azureus), and surveys showing a huge demand for legal P2P content [2], publishers need to identify how to best allocate their resources across multiple swarms. For that purpose, it is imperative to identify whether a swarm is *self sustaining* or not. This is particularly evident in a market where enterprises that can make "everything available, with small costs", thrive [1]. Third, models focusing on small user populations may provide insight on when and if coding, bundling [6] or other techniques can help to make unpopular swarms last longer without the support of a publisher.

For large populations, Massoulie and Vojnovic [5] used a coupon collector model to show that rarest-first guarantees almost uniform distribution of chunks across the population leading to a robust system. Fan et al. [4] considered the large population regime, and used fluid approximations and stochastic differential equations to model the dynamics of a population of users. Qiu and Srikant [8], also considered large populations and concluded that the efficiency of the system is always high.

For small populations we have Markov Chain (MC) models [10] that provide insights on the performance of the system but that do not consider the problem of chunk availability.

In comparison, the goal of this paper is to analyze, especially for a small population of users, how chunk availability varies as a function of different system parameters such as arrival rate of peers and download capacity.

2. AVAILABILITY MODEL

In this section, we present our availability model. We consider an infinite population of users that arrive according to a Poisson process with rate λ and whose download rate is μ chunks/second. The file is divided into b chunks. The distribution of chunks across users is uniform, which can be realized in practice using rarest-first. Consider a user that has h chunks of a file. The probability that this user holds any given combination of h chunks equals $1/C_h^b$ where $C_h^b = b!/[h!(b-h)!]$.

Our model is hierarchical: in the upper layer we characterize the population of users, and in the lower layer the distribution of chunks.

2.1 Upper Layer

We consider a Jackson network of $b+1$ $M/G/\infty$ queues in series. The number of customers in queue i, n_i, represents the number of users that have i chunks of the file. The state of the system is characterized by a $(b+1)$-tuple, $\sigma = (n_0, n_1, n_2, \ldots, n_b)$.

Peers arrive according to a Poisson process with rate λ to queue 0 and transit from queue i to queue $i+1$ $(0 \leq i \leq b-1)$ with rate μ. The b^{th} queue captures the mean time that peers remain in the system after completing their downloads, $1/\gamma$. Making $\gamma \to \infty$ models the case when all peers leave the system immediately after completing the download.

The steady state distribution of the queueing system has product form and is given by $\pi(n_0, \ldots, n_b) = \prod_{i=0}^{b} \pi_i(n_i) = \prod_{i=0}^{b} [\frac{\rho_i^{n_i}}{n_i!} e^{-\rho_i}]$ where $\rho_i = \lambda/\mu_i$ and $\mu_i = \mu$ for $0 \leq i < b$ and $\mu_i = \gamma$ for $i = b$.

2.2 Lower Layer

Given the current state $\sigma = (n_0, n_1, n_2, \ldots, n_b)$ of the upper layer's queueing system, we compute the probability that at least one chunk is not available in the system.

2.2.1 Distribution

We first consider the case where all the users have the same number of chunks, h, i.e., $n_h > 0$ and $n_i = 0$ for all $i \neq h$. There is a total of hn_h chunks in the system.

A user u is characterized by its signature s_u, which is a binary number of b bits where the i^{th} bit, b_i, is set to 1 if the user has chunk i and 0 otherwise. There are C_h^b possible signatures.

We now describe the sample space of the model's lower layer, which we denote by Ω. Each element $\underline{s} \in \Omega$ is characterized by the concatenation of n_h user signatures. Once an ordering of the n_h users is established, an element of the sample space is fully specified by the string of zeros and ones $\underline{s} = s_1 \cdot s_2 \cdot \ldots \cdot s_{n_h}$ where $\cdot$ denotes concatenation,

$$\Omega = \{(s_1 \cdot s_2 \cdot \ldots \cdot s_{n_h})\}, \qquad |\Omega| = (C_h^b)^{n_h} \tag{1}$$

Since the distribution of chunks across users is assumed to be uniform, each element in the sample space is equiprobable. We are interested in the event U corresponding to at least one chunk being unavailable. Let U_i be the event corresponding to chunk i being unavailable,

$$P\{U\} = P\{U_1 \cup U_2 \cup \ldots \cup U_b\} = N_U/|\Omega| \tag{2}$$

where N_U denotes the number of elements in Ω where at least one chunk is unavailable.

Using the inclusion exclusion principle, one can show that

$$N_U = \sum_{i=1}^{b-1} (-1)^{i+1} C_i^b (C_h^{b-i})^{n_h} \tag{3}$$

with the convention that $C_k^m = 0$ if $m < k$.

We denote by $E(z)$ the generator function of the number of states in which *exactly* i chunks are missing. [9, Section 4.2] yields $E(z) = \sum_{i=0}^{b-1} C_i^b (C_h^{b-i})^{n_h} (z-1)^i$. Note that $E(0)$ is the number of states where no chunk is missing and $N_U = |\Omega| - E(0)$ (equation (3)).

Recursion: Inspired by [3, Figure 1], we derive a recursion to compute N_U. Let $e(n,u)$ be the number of states in which u chunks are left unavailable after n users get h chunks each. Then,

$$e(n,u) = \sum_{i=u}^{b-h} e(n-1,i) C_{i-u}^{i} C_{h-(i-u)}^{b-i} \tag{4}$$

where C_{i-u}^i counts the number of ways in which $i-u$ chunks of the n^{th} user can be distributed without overlap with the chunks of the first $n-1$ users and $C_{h-(i-u)}^{b-i}$ counts the number of ways in which $h-(i-u)$ chunks can be distributed overlapping with previously distributed chunks.

The initialization step consists of setting $e(1, b-h) = C_h^b$ and $e(1,j) = 0$ for $j \neq b-h$. After evaluating $e(n,u)$, $1 \leq n \leq n_h$, $1 \leq u \leq b-h$, the number of configurations in which at least one chunk is missing is given by $N_U = \sum_{i=1}^{b-h} e(n_h, i)$. Assuming that the binomial coefficients are precomputed, the complexity of the dynamic program is $O(b^2 n_h)$. Note also that the dynamic program only involves additions of positive numbers, which avoids numerical problems.

Extension to heterogeneous population case: We now consider the case where different users may have different numbers of chunks. Once an ordering of the $n = \sum_{i=1}^b n_i$ users is established, an element of the sample space is fully specified by the string of zeros and ones $\underline{s} = s_1 \cdot s_2 \cdot \ldots \cdot s_n$ where $\cdot$ denotes concatenation, $\Omega = \{(s_1 \cdot s_2 \cdot \ldots \cdot s_n)\}$, $|\Omega| = \prod_{i=1}^{b-1} (C_i^b)^{n_i}$. Hence,

$$N_U = \sum_{i=1}^{b-1} (-1)^{i+1} C_i^b \prod_{j=1}^{b-1} (C_j^{b-i})^{n_j}. \tag{5}$$

Substituting (5) into (2) we obtain the probability of at least one chunk being unavailable.

The recursion presented above can be adapted to an heterogeneous population as follows.

1) For the n_1 users that have one chunk we initialize the dynamic programming table $\mathcal{T}_1$ as described above and compute $e(n,i)$ using equation (4) $(h=1)$.

2) Varying h from 2 to b,

2.1) For the population of n_h users that have h chunks, build a new dynamic programming table, $\mathcal{T}_h$, initializing its 0^{th} row using the last line of $\mathcal{T}_{h-1}$. The first row of $\mathcal{T}_h$ is then obtained from the 0^{th} row after adding C_h^b to $e(0, b-h)$. The other rows are obtained using (4).

3) The number of configurations in which at least one chunk is unavailable is $\sum_{i=1}^b e(n_b, i)$ (table $\mathcal{T}_b$).

Note that if at least one user has all the chunks $(n_b > 0)$ the output of the procedure above is 0. Otherwise, it is the sum of all but the first element of the 0^{th} row of table $\mathcal{T}_b$.

2.2.2 Expectations

In this section we take the expected number of distinct chunks as being the measure of content availability. First, we consider the homogeneous case where there are n_h users in the system all of them possessing h chunks of the file. We are interested in the expected number of distinct chunks available in the system, $E[N]$, $E[N] = E[\mathbf{1}_1] + E[\mathbf{1}_2] + \ldots + E[\mathbf{1}_b] = bE[\mathbf{1}_i]$, where $\mathbf{1}_i$ is an indicator random variable equal to 1 if chunk i is available and 0 otherwise. Let p be the probability that chunk i is available, $p = E[\mathbf{1}_i] = 1 - \left(\frac{b-h}{b}\right)^{n_h}$.

The minimum number of users $n_h^\star$ such that $E[N] \geq b - \epsilon$ is

$$b\Big[1 - \Big(\frac{b-h}{b}\Big)^{n_h^\star}\Big] = b - \epsilon; \quad n_h^\star = \frac{\log(b/\epsilon)}{\log(b/(b-h))} \tag{6}$$

where $n_h^\star = \Theta(\log 1/\epsilon)$. We refer to $n_h^*(\epsilon)$ as the *threshold for self sustainability* since, for small enough ϵ, if the average number of peers in the system is greater than $n_h^*(\epsilon)$ content is fully available even in the absence of a seed.

From [7, Theorem 1] we can show that

$$P\{|pb - N| > \xi b\} \leq 2e^{\frac{-\xi^2 b^2}{2n_h h}} \tag{7}$$

Eq. (7) is useful if the number of chunks b is large. For example, if we let $b = 15000, h = 1000, \epsilon/b = 0.01$, $n_h^\star = 66.75$, and $E[N] = 14850$. From (7), setting $\xi = 0.05$ and for $n_h^\star$ peers, the probability that the peers have less than 95% of all distinct chunks of the file is at most 0.03. For small values of b the recursion previous developed should be used. Nevertheless, $E[N]$ provides a good estimate for $n_h^\star$.

Extension to heterogeneous population case: For a heterogeneous population of users, n_i users owning i chunks of the file, $\sigma = (n_0, n_1, n_2, \ldots, n_b)$,

$$E[N_\sigma] = b\Big[1 - \prod_{i=0}^{b} \Big(\frac{b-i}{b}\Big)^{n_i}\Big] \tag{8}$$

where N_σ denotes the number of distinct chunks available in the system when the upper layer's state is σ. As discussed above, $\left(\frac{b-i}{b}\right)^{n_i}$ is the probability that none of the n_i users that have exactly i chunks owns a given tagged chunk.

Applicability of the model: Once the threshold for content availability, ϵ, is fixed we assume that content is available in states σ where $E[N_\sigma] > b - \epsilon$ and not available otherwise, and the fraction of time that all chunks are available, A, is

$$A = \sum_{\sigma=(n_0,n_1,\ldots,n_b)} \mathbf{1}_{E[N_\sigma]>b-\epsilon} \prod_{i=0}^{b} [\frac{\rho_i^{n_i}}{n_i!} e^{-\rho_i}] \tag{9}$$

3. NUMERICAL EVALUATION

Our goal in this section is to illustrate how availability depends on different system parameters. To this end, we consider a file of 4Mb chopped into $b = 16$ chunks of size 256Kb each. Figure 1(a) depicts n^*, the minimum number

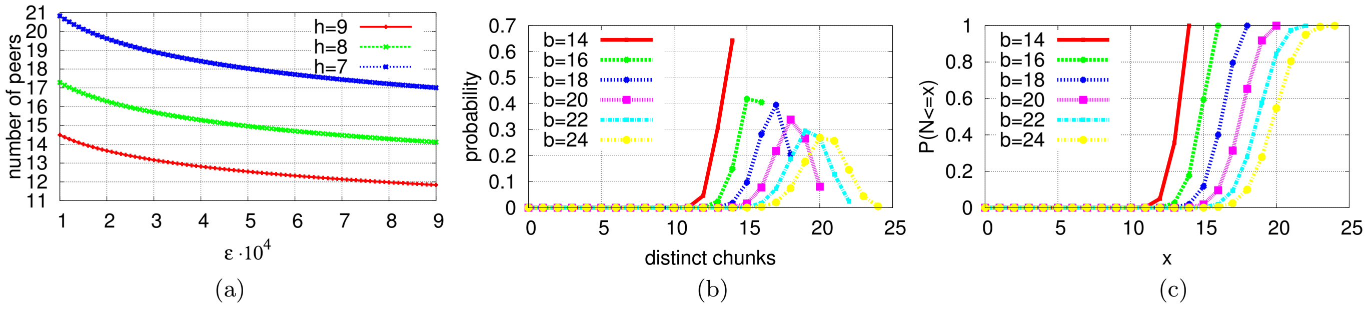

Figure 1: (a) Minimum number of peers necessary to attain $E[N] = 16 - \epsilon$ (eq. (6)); (i) probability mass function of the number of available chunks (eq. (4)); (ii) CDF of the number of available chunks.

of users necessary to attain $E[N] = b - \epsilon = 16 - \epsilon$ distinct chunks in the system, as a function of ϵ (eq. (6)). The red, green and blue curves correspond to a homogeneous population of users, each one owning 9, 8 and 7 chunks of the file, respectively. For instance, if each user owns 9 chunks (red curve) and the population size is 12, the expected number of distinct chunks in the system is $16 - \epsilon = 16 - 9 \times 10^{-4}$. We will return to this scenario in the end of this section.

Note that for a fixed value of ϵ, n^* decreases in h. For a fixed value of h, moving from right to left in Figure 1(a), a decrease of one order of magnitude in ϵ corresponds to an increase by only 10 peers in n^*. This reflects the fact that a small increase in the population size may lead to a significant increase in the availability level.

We now illustrate the distribution of the number of distinct chunks in the system. To this end, we consider a population of 8 users each one owning 5 chunks, varying the file size b from 14 to 24 in increments of 2. The expected number of chunks in the system, $E[N]$, is 13.59, 15.20, 16.67, 17.99, 19.20 and 20.29, respectively. The distribution of the number of distinct chunks in the system is shown in Figure 1(b) and its CDF in Figure 1(c).

Note that the probability mass function is relatively concentrated around its mean. If $b = 20$, for instance, the mode is 18 (Figure 1(b)), the probability of finding less than 17 chunks in the system is around 0.3 and the probability of finding less than 15 chunks is smaller than 0.02 (Figure 1(c)).

To ground the numerical evaluation presented above we consider preliminary results from a PlanetLab experiment. In our controlled experiments, we deployed 100 BitTorrent clients in a private swarm, each downloading a file of size s = 4MB (16 chunks of size 256KB each), each node offering capacity μ = 100KBps. A seed joins the system at time $t = 0$, leaves at t = 80s and never returns. Peers leave the system as soon as they conclude their downloads. We varied the arrival rate, λ, of peers from 1/60 to 8/60 peers/s and conducted 50 runs for each arrival rate. When $\lambda \leq 4/60$, in all the conducted experiments some chunks were missing after 5000 seconds. The scenario drastically changed for $\lambda^\star \geq 7/60$, when almost all the swarms were active even after 10000 seconds, time at which we stopped the runs. The measured expected number of peers for $\lambda^\star$ is around 12. Our model suggests that 12 is the self sustainability threshold, for instance, in case each user owns 9 chunks and $\epsilon = 9 \times 10^{-4}$ (See Figure 1(a)). It is interesting to note that, for $\lambda^\star$ and using the upper layer model, the expected number of peers in the system is 12. In addition, there is a high probability (0.96) that at least 2 peers have downloaded 9 or more chunks and, from the recursion, the probability of a missing chunk is negligible (1.2×10^{-7}).

4. CONCLUSION AND FUTURE WORK

In this paper we used a two layer model to study how chunk availability varies as a function of the number of users in the system. With our model we have shown that relying on relatively small populations of users may be enough to have all chunks available almost all the time, a result corroborated by our preliminary PlanetLab experiments. Refining the model and performing thorough validations is a future work. We believe that our work sheds light on the important but not very explored problem of chunk availability in unpopular swarms.

Acknowledgement: This work was supported in part by the NSF under awards CNS-0519922 and CNS-0721779, fellowships from CAPES and grants from CNPq and FAPERJ (Brazil).

5. REFERENCES

[1] ANDERSON, C. *The Long Tail: Why the Future of Business is Selling Less of More*. Hyperion, 2006.

[2] BRITISH MUSIC RIGHTS. Huge demand for legal P2P, 2008. `http://torrentfreak.com/`.

[3] BURGER, A. P., AND VAN VUUREN, J. H. Balanced minimum covers of a finite set. *Discrete Mathematics 307* (2007), 2853–2860.

[4] FAN, B., CHIU, D.-M., AND LUI, J. Stochastic differential equation approach to model peer to peer systems. In *ICC* (2006).

[5] MASSOULIE, L., AND VOJNOVIC, M. Coupon replication systems. In *SIGMETRICS'06* (2006).

[6] MENASCHE, D., ROCHA, A., , LI, B., TOWSLEY, D., AND VENKATARAMANI, A. Bundling builds availability in Bittorrent. *UMass TR UM-CS-2009-010* (2009).

[7] MITZENMCHER, M. Compressed bloom filters. In *PODC* (2001).

[8] QIU, D., AND SRIKANT, R. Modeling and performance analysis of BT-like P2P nets. In *SIGCOMM* (2004).

[9] WILF, H. *Generatingfunctionology*. Acad. Press, 1994.

[10] YANG, X., AND DE VECIANA, G. Service capacity of peer to peer networks. In *INFOCOM* (2004).

APPENDIX

Proof of equation (7): The proof relies on Azuma's inequality and Doob's martingale. Pick an ordering for the hn_h chunks owned by the users, which leads to a sequence $\{S_1, S_2, \ldots, S_{nn_h}\}$ of chunks, where each S_i is a discrete random variable with support $0 \ldots b-1$. In what follows, we show how to construct a martingale based on this sequence. Let Y_j be the expected number of chunks unavailable given that j chunks were distributed ($0 \leq j \leq nn_h$). $Y_j = E[b - N | S_1, \ldots, S_j]$. In particular, $Y_0 = E[b-N]$ and $Y_{n_h} = b-N$. The sequence $\{Y_0, \ldots, Y_j\}$ is a special case of the Doob's construction, which always leads to a martingale. Finally, note that the addition of a chunk in the system may increase the number of available chunks by at most 1, hence $|Y_{j+1} - Y_j| \leq 1$. The rest of the proof is a direct application of Azuma's inequality to the sequence $\{Y_0, \ldots, Y_{hn_h}\}$. $\square$