A new caching policy for cloud assisted Peer-to-Peer video-on-demand services

Franco Robledo Amoza* frobledo@fing.edu.uy
Pablo Rodríguez-Bocca* prbocca@fing.edu.uy
Pablo Romero* promero@fing.edu.uy
Claudia Rostagnol* crostag@fing.edu.uy

* Departamento de Investigación Operativa, Facultad de Ingeniería, Universidad de la República.
Julio Herrera y Reissig 565, 11300, Montevideo, Uruguay.

***Abstract*—We propose a mathematical model to minimize the expected download time of cloud assisted Peer-to-Peer video on demand services. First, we define a simple fluid model that quantifies the evolution of peers, which are grouped into different classes regarding the number of concurrent video downloads. Then, analytical expressions for the expected download time are obtained under steady state, via Little's law. The goal is to minimize the expected download time with limited storage capacity in cache nodes of the network, called super-peers. The nature of this combinatorial problem is similar to the Multi-Knapsack Problem (MKP): the number of copies must be chosen for each video stream, with storage capacity constraints. We resolve the problem with a greedy randomized technique. The performance of this co-operative system is compared with a traditional content delivery network. Finally, the new caching-policy is tested in a real scenario. The results confirm that the swarm assisted peer-to-peer service is both more economical and suitable to address massive scenarios, whereas the performance of both systems is similar in small scale instances. *Index Terms*— P2P, VoD, Performance, GRASP**

I. Introduction

Nowadays, most of the systems used to distribute video streaming over Internet are running within well-known Content Delivery Networks (CDNs), which are deployed over clustered servers. These systems offer a predictable service with static resources under controlled scenarios. However, when the demand increases, such systems can become easily overloaded. An alternative approach is the deployment of peer-to-peer (P2P) networks, in which users are both clients and servers, defining a cooperative highly scalable community. Basically, there are three P2P video streaming modes, which are similar from the users viewpoint, but extremely different when regarding design aspects. The first one is *file sharing*, in which the video stream must be completely downloaded before its playback. The second one is *video on-demand*, where the video is distributed as progressive download to end-users, using a "best effort" protocol that does not include any consumer-producer synchronization. The third streaming mode is called *live-video streaming*, where there is an explicit synchronization between consumers and producers.

An inspirational system for replication and fast dissemination is BitTorrent, created by Bram Cohen [1], [2]. BitTorrent was originally designed for file sharing applications. However, most currently deployed P2P applications over the Internet are BitTorrent-based. One of such applications is the GoalBit Video Platform [3], [4], currently used to share live and on-demand video streaming over a P2P network. In BitTorrent, peers are either *seeders* or *leechers*. A downloader peer is a leecher. Once a peer completely downloads the video stream, it becomes a seeder, and eventually provides service to other peers. There is also an entity or node named *tracker*, which knows all the peers that are seeding or downloading a certain video stream. GoalBit introduces a third node-type called *super-peers*. These nodes have higher bandwidth resources than normal peers, and longer life-time in the system as well. Super-peers are intended to store and forward the video stream to common peers (who only have short sessions within the system). In the current GoalBit protocol specification, super-peers are nodes managed by the operator of the platform, hosted in the cloud, and they implement a specific caching policy. We are studying the promotion from peers to super-peers in future specifications. The reader is directed to more detailed specification of GoalBit [3].

The optimization process of real P2P systems via *trial-and-error* is not practical. Poor streaming rates have disappointing effects in end-users. As a consequence, the scientific community is engaged with an in-depth understanding of BitTorrent via mathematical analysis. Several studies of BitTorrent-like systems have been carried out by different research-groups, resulting in some models that represent its dynamic, helping to predict the behavior of the system.

Qiu and Srikant found the expected download time of a user in a BitTorrent network, assuming that the behavior of peers can be modeled by a Markov chain [5]. This work is generalized in a first stage by extending the model to analyze concurrent downloading [6]. A second generalization includes the presence of super-peers in the network, and it studies a video on-demand application under steady state [7]. Here, we propose a further generalization of the mathematical approach, including the churn phenomenon, since peers (as well as seeders) may abort the system when they wish. We also prove that the expected download time in a CDN session is never better than that of its equivalent P2P system, and discuss scalability and performance of both systems. Our goal is to distribute on-demand video streaming over the GoalBit platform, planning the efficient allocation of storage capacity. We want to minimize the expected download time perceived by end-users. The intuition suggests that the most popular

video files should be stored in more super-peers. In this paper we will explore this trade-off between storage capacity and fast on-demand propagation.

In Section II we present our combinatorial optimization problem based on a fluid model, that extends the previous work [6]. This problem is similar to the Multi-Knapsack Problem (MKP), which is known to be in the class of NP-Hard problems. Section III presents a Greedy Randomized metaheuristic [8]. We discuss design aspects of CDN and swarm-assisted P2P systems in Section IV. There, we show the performance of our caching methodology in a real scenario, and present concluding remarks of our work. Algebraic details are provided in an Appendix.

II. Fluid Model for P2P and CDN Architectures

In order to understand the behavior of peers in a P2P system like GoalBit, we should analyze peer evolution, scalability and sharing efficiency. We extend the stochastic fluid model [6], providing insightful results for performance issues and the expected download time for concurrent downloads. Since each peer can download multiple streams at time t, peers are grouped into classes: $\{C^1, C^2, \ldots, C^K\}$, such that a peer is in class C^i if it is downloading i different streams at the same time. The data and variables of the model are described as follows:

- K: available videos.
- s_j: size (in `kbits`) of video j.
- $x_j^i(t)$: *downloaders* in class C^i downloading video j at time t.
- $y_j^i(t)$: *seeders* in class C^i seeding video j at time t.
- $sp_j^i(t)$: *super-peers* in class C^i seeding video j at time t.
- λ_j^i: arrival rate for peers in class C^i requesting video j.
- θ_j^i: departure rate of *leechers* in class C^i downloading video j.
- γ_j^i: departure rate of *seeders* in class C^i seeding video j.
- c: download bandwidth for each peer (in `kbps`).
- μ: upload bandwidth for each peer (in `kbps`).
- ρ: upload bandwidth for each *super-peer* (in `kbps`).
- η: video sharing efficiency between peers ($\eta \in [0,1]$).

In order to simplify the model representation, we shall assume the following hypothesis:

1) *Fairness:* Resources are used equally distributed among concurrent videos. A peer from class C^i will allocate the i-th part of its peer's bandwidth for each concurrent download. Hence, the file portion downloaded per second for video j is $c_j^i = \frac{c}{is_j}$. Analogously, $\mu_j^i = \frac{\mu}{is_j}$ and $\rho_j^i = \frac{\rho}{is_j}$ $\forall i,j \in \{1 \ldots K\}$.
2) *Tit-for-tat:* Peers in class C^i that at time t are downloading video j, receive from all other ***downloaders*** a streaming rate proportional to the upload bandwidth μ_j^i and their population $x_j^i(t)$: $\left(\frac{\mu_j^i x_j^i(t)}{\sum_k \mu_j^k x_j^k(t)}\right) \sum_k \eta \mu_j^k x_j^k(t) = \eta \mu_j^i x_j^i(t)$.
3) *Seeders Availability:* Peers in class C^i that at time t are downloading video j, receive from all ***seeders*** a streaming rate proportional to the download bandwidth c_j^i and their population $x_j^i(t)$: $\left(\frac{c_j^i x_j^i(t)}{\sum_k c_j^k x_j^k(t)}\right) \sum_k \mu_j^k y_j^k(t)$.
4) *Super-peers Availability:* Peers in class C^i that at time t are downloading video j, receive from all ***super-peers*** a streaming rate proportional to the download bandwidth c_j^i and their population $x_j^i(t)$:
$$\left(\frac{c_j^i x_j^i(t)}{\sum_k c_j^k x_j^k(t)}\right) \sum_k \rho_j^k sp_j^k(t) = \alpha_j^i \sum_k \rho_j^k sp_j^k(t).$$
5) *Peers abortion:* the departures are linearly decreasing with respect to the numbers of concurrent video downloadings: $\gamma_j^i = \gamma/i$, and $\theta_j^i = \theta/i$.

Assuming that we can model the problem as a Markov chain, we want to determine the peers evolution $x_j^i(t)$ and $y_j^i(t)$. Modeling the behavior as a simple fluid model we get the following differential equations: $\forall i,j \in \{1 \ldots K\}$:

$$\frac{dx_j^i}{dt} = \lambda_j^i - \frac{\theta}{i} x_j^i - \min\left\{ \frac{c}{is_j} x_j^i, \eta \frac{\mu}{is_j} x_j^i + \alpha_j^i \sum_k \left(\frac{\mu}{s_j} \frac{y_j^k}{k} + \frac{\rho}{s_j} \frac{sp_j^k}{k} \right) \right\} \tag{1}$$

$$\frac{dy_j^i}{dt} = \min\left\{ \frac{c}{is_j} x_j^i, \eta \frac{\mu}{is_j} x_j^i + \alpha_j^i \sum_k \left(\frac{\mu}{s_j} \frac{y_j^k}{k} + \frac{\rho}{s_j} \frac{sp_j^k}{k} \right) \right\} - \gamma_j^i y_j^i, \tag{2}$$

The minimum function of Equations (1) and (2) ensures that the bottleneck of the system is either in the upload or in the download capacity. Via elementary algebra, the number of downloaders and seeders in steady state are (see the Appendix for details):

$$\overline{x_j^i} = \max\left\{ \frac{\lambda_j^i i s_j}{c}, \frac{i\lambda_j^i (\lambda_j - \overline{\rho_j} - \overline{\phi_j})}{\lambda_j \left(\theta + \frac{\eta\mu}{s_j} - \frac{\mu\theta}{\gamma s_j}\right)} \right\} \tag{3}$$

$$\overline{y_j^i} = \frac{\lambda_j^i - \theta_j^i \overline{x_j^i}}{\gamma_j^i}, \tag{4}$$

where $\overline{\rho_j} = \sum_k \rho_j^k sp_j^k$ and $\overline{\phi_j} = \frac{\mu}{\gamma}\lambda_j$.

A traditional CDN can be viewed as a particular case of this analytical approach, where users do not cooperate ($\eta = 0$, $\mu = 0$), there are no seeders ($y_j^i(t) = 0$), and the previously named super-peers are now static servers. Replacing in Equation (1), the CDN Fluid Model is defined by the following system of ordinary differential equations:

$$\frac{dx_j^i}{dt} = \lambda_j^i - \theta_j^i x_j^i - \min\{\frac{c}{is_j} x_j^i, + \alpha_j^i \sum_k \frac{\rho}{s_j} \frac{sp_j^k}{k}\} \tag{5}$$

The expressions for the steady state in the CDN system can be immediately obtained from (3) taking $\mu = 0$ and $\eta = 0$:

$$\overline{x_j^i}_{CDN} = \max\left\{ \frac{\lambda_j^i i s_j}{c}, \frac{i\lambda_j^i (\lambda_j - \overline{\rho_j})}{\theta \lambda_j} \right\} \tag{6}$$

Let us denote T^{P2P} and T^{CDN} the expected download times under steady state for peers in a P2P network and users in the CDN system, respectively. The following proposition is intuitive.

Proposition II.1. $T^{P2P} < T^{CDN}$

Proof: The expected download time for any particular *downloader* under steady state can be computed by applying *Little's law*:

$$T_j^i = \frac{\overline{x_j^i}}{\overline{\lambda_j^i}} \tag{7}$$

The expected download time of the P2P Poisson processes can be found with conditional expectation:

$$T^{P2P} = \sum_i \sum_j \frac{\lambda_j^i}{\lambda} T_j^i = \frac{1}{\lambda} \sum_i \sum_j \overline{x_j^i}, \tag{8}$$

where $\lambda = \sum_j \lambda_j$ is the total rate, and $\overline{x_j^i}$ the number of peers in class C^i downloading video j under steady state. An identical argument can be applied for T^{CDN}. Hence, it suffices to prove that $\overline{x^i_j}_{P2P} < \overline{x^i_j}_{CDN}$. If the system's bottleneck is the download capacity then the equality is obvious (just compare Expressions (6) and (3)). Otherwise, the second argument of the maximum function from Expression (3) must dominate, and its numerator $i\lambda_j^i(\lambda_j - \overline{\rho_j} - \overline{\phi_j})$ is positive. The following chain of inequalities holds:

$$\begin{aligned} \overline{x^i_j}_{P2P} &= \frac{i\lambda_j^i(\lambda_j - \overline{\rho_j}) - i\lambda_j^i\overline{\phi_j}}{\lambda_j\left(\theta + \frac{\mu}{s_j}(\eta - \frac{\theta}{\gamma})\right)} \\ &< \frac{i\lambda_j^i(\lambda_j - \overline{\rho_j})}{\theta\lambda_j} \\ &= \overline{x^i_j}_{CDN}. \end{aligned}$$

This inequality holds if and only if

$$\left(-i\lambda_j^i\overline{\phi_j}\right)(\theta\lambda_j) < i\lambda_j^i(\lambda_j - \overline{\rho_j})\left(\lambda_j\frac{\mu}{s_j}\left(\eta - \frac{\theta}{\gamma}\right)\right).$$

Using that $\overline{\phi_j} = \frac{\mu\lambda_j}{\gamma s_j}$ and canceling common factors, the inequality holds if and only if

$$\lambda_j\frac{\theta}{\gamma} + (\lambda_j - \overline{\rho_j})\left(\eta - \frac{\theta}{\gamma}\right) > 0,$$

which is equivalent to $\eta\lambda_j > \overline{\rho_j}(\eta - \frac{\theta}{\gamma})$. But the latter inequality is obviously true, since $\lambda_j \geq \overline{\rho}_j + \overline{\phi_j} > \overline{\rho_j}$. ∎

We want to conduct an optimal file-distribution in super-peer nodes, in order to minimize the expected download time T^{P2P} of the P2P system. On the other hand, and for simplicity, we reduce the combinatorial problem to the case in which all peers download a single video stream ($i = 1$). We will study the special case in which peers download a single video file (i.e. a sequential system). Our problem can now be written as a *Combinatorial Optimization Problem* (COP), where the decision variable is the binary matrix E_j^p, that indicates either if super-peer p seeds video j or not, and the objective function is exactly T^{P2P} from Equation (8), times the constant λ:

$$\min_{E_j^p} \sum_{j=1}^{K} \max\left\{\frac{\lambda_j s_j}{c}, \frac{\lambda_j - \overline{\rho_j} - \overline{\phi_j}}{\theta + \frac{\eta\mu}{s_j} - \frac{\mu\theta}{\gamma s_j}}\right\}$$

$$\begin{aligned} s.t. \\ \sum_j E_j^p s_j &\leq S^p \,\forall p \\ \sum_p E_j^p &\geq 2 \,\forall j \\ sp_j &= \sum_p E_j^p \,\forall j \\ E_j^p &\in \{0,1\} \,\forall j,p \end{aligned}$$

The first constraint states that the storage capacity of super-peers cannot be exceeded. The second constraint is 2-redundancy: each video must be stored at least twice in the whole system. Finally, the number of video replicas sp_j for video j in the system can be computed from the Boolean matrix E_j^p by a direct sum.

III. Greedy Randomized Optimization

The COP we address here is similar to the Multi-Knapsack Problem (MKP), where each super-peer holds a knapsack (storage) with limited capacity and items (objects) should to be placed inside the knapsacks (super-peers' storage). However, in our COP the objective function has a non-linear relation with the selected items.

Its nature is similar to that of the MKP, which is known to be Strongly NP-Hard (it reduces to the Bin Packing Problem [9]). In our constrained version with non-linear objective in the profits, we will design a Greedy Randomized Adaptive Search Procedure (GRASP) [8]. This is a well-known metaheuristic, very successful to solve COP related with robustness and scheduling of P2P networks [10], [11], as well as problems similar to the multi-knapsack with multiple constraints [12]. It is an iterative process which operates in two phases. In the *Construction Phase* an initial feasible solution is built, whose neighborhood is then explored in the *Local Search Phase*.

We present a GRASP customization to solve our problem. As a result we will get a possible distribution of videos in the available super-peers, which gives the best (minimum) download time of all tested solutions.

Algorithm 1 $E = RandomGreedy(\lambda, \theta, \gamma.\eta, s, S, \mu, c, \rho)$

```
1: V = SortVideos(s)
2: for j = 1 TO K do
3:    (p1, p2, p3) ← FeasibleSelect(S1, ..., SP)
4:    E(p1, V(j)) ← 1
5:    E(p2, V(j)) ← 1
6:    if p3 > 0 then
7:       E(p3, V(j)) ← 1
8:    end if
9:    Update(S1, ..., SP)
10: end for
11: return E
```

Algorithm *RandomGreedy* represents the *Construction Phase*. Its pseudo-code is specified in (1). Initially, in first line, all videos are sorted in decreasing order with its size (i.e. $s_1 > s_2 > \ldots s_K$). Straight afterwards, for each video, we select 2 or 3 super-peers to add video replicas (line 3). The algorithm must assure feasibility, so function *FeasibleSelect* must return at least 2 super-peers (p_1 and p_2, randomly chosen) where to place the video. Additionally, only for the 20% of most popular videos, we add a third replica in a different super-peer (p_3). Note that p_3 can take negative values if it is not selected for the current video (checked in line 6). Once the super-peers are selected, the distribution matrix E is updated correspondingly (lines 4 to 8). Finally, we update the available resources (line 9) and the resulting matrix E is returned (line 11).

A *Local Search* phase is introduced in order to improve the solution returned by *RandomGreedy*. A neighbor solution is obtained either by adding, deleting or swapping video files between super-peers. Each step takes effect only if the movement produces both a better and feasible solution. The pseudo-code for this local search phase is shown in Algorithm (2).

Algorithm 2 $E_{out} = LocalSearch(E)$

1: $(E, improve) \leftarrow \mathsf{Add}(Rand(SP, Video))$
2: IF *improve* GO TO Line 1
3: $(E, improve) \leftarrow \mathsf{Delete}(Rand(SP, Video))$
4: IF *improve* GO TO Line 1
5: $(E, improve) \leftarrow \mathsf{Swap}(Rand(SP1, V1), Rand(SP2, V2))$
6: IF *improve* GO TO Line 1
7: **return** $E_{out} = E$

IV. Numerical Results and Discussion

We executed our GRASP implementation in MATLAB [13], using available data from our National Telephony Operator ANTEL (see [14] for details). In this way, we test our algorithm in a real-life scenario. From the log information of this service, we obtained the size, popularity, videos' popularity, videos' size, number of servers and servers' streaming rate. Our scenario has the following data:

- 700 videos (K),
- videos' average size is 23 MB (s),
- 4 super-peers (P),
- super-peers' storage capacity is 1500 GB (S),
- super-peers' upload rate is 10 Mbps.(ρ),
- peers' download rate is 1 Mbps (c),
- peers' upload rate is 0.25 Mbps (μ),
- P2P effectiveness of 50% ($\eta = 0.5$),
- seeders' departure with rate 100 ($\gamma = 100$) and
- peers' departure with rate 0.1 ($\theta = 0.1$).

We contrast the performance of both CDN a P2P system in two aspects: scalability and economical savings. On one hand, in order to determine the scalability of both services, we stressed the system keeping the popularity proportional with the real data (i.e. multiplying the real λ by an increasing factor). The results are shown in Figure (1). The average download time of an end-user in the CDN system is never lower than that of a peer in a P2P system, as Proposition II.1 predicts. The performance of both systems is similar for small scale scenarios. However, the CDN system cannot support highly populated scenarios.

The expected download time in the P2P system almost unchanged despite of an increase in the demand, showing its scalability. In order to reach the same level of service (the same average download time) in a client-server system we should increase the number of servers (or super-peers) proportionally with the end-user requests, while in the P2P system we have a natural scalability with the growing resources offered by users (downloaders and seeders).

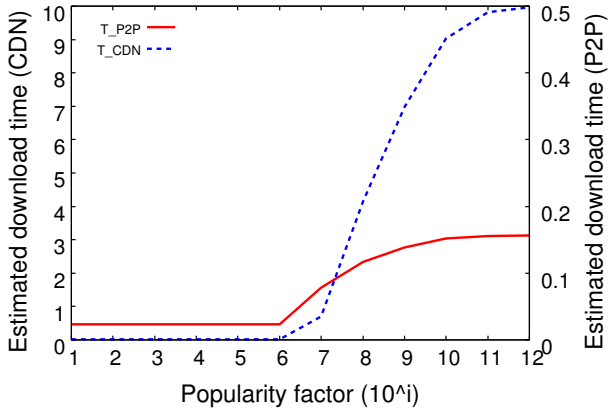

Fig. 1. Performance of CDN and P2P systems versus popularity.

On the other hand, we chose a fixed value for the system popularity and then changed the number of super-peers/servers (increasing the available resources at the platform side). This experiment relates the performance of both systems with respect to the number of node-source servers. As shown in Figure (2) the CDN system is much more sensible to the changes in the available resources.

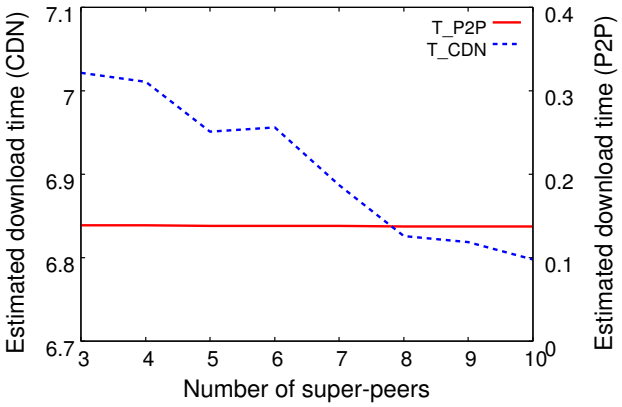

Fig. 2. P2P performance versus number of super-peers.

Both results reveal that P2P system can perform better under limited platform resources than a CDN system. We remark that there are a number of successful commercial CDN providers

that use their own infrastructure. Our results, however, agree with the intuition that such networks may improve even further by also leveraging P2P links for video delivery. We expect that the efficiency will be more evident in larger deployments. Currently, we are working on embedding this model within the GoalBit platform to evaluate it with real-world inputs. We also hope to gather data from popular commercial services to further validate our model, and to evaluate other related methods for comparison.

V. Appendix

A. Steady State of the Fluid Model

If the upload is the bottleneck, the steady state can be trivially obtained by solving a linear system. Otherwise, the system of equations in the steady state is specified for all $i, j \in \{1, \ldots, K\}$ as follows:

$$0 = \lambda_j^i - \theta_j^i \overline{x_j^i} - \eta \mu_j^i \overline{x_j^i} - \frac{c_j^i \overline{x_j^i}}{\sum_k c_j^k \overline{x_j^k}} \sum_k (\mu_j^k \overline{y_j^k} + \rho_j^k s p_j^k) \quad (9)$$

$$0 = \eta \mu_j^i \overline{x_j^i} + \frac{c_j^i \overline{x_j^i}}{\sum_k c_j^k \overline{x_j^k}} \sum_k (\mu_j^k \overline{y_j^k} + \rho_j^k s p_j^k) - \gamma_j^i \overline{y_j^i} \quad (10)$$

Several terms cancel out when summing Equations (9) and (10), to obtain (2). Recall that $\mu_j^i = \frac{\mu}{i s_j}$, $\theta_j^i = \theta / i$ and $\gamma_j^i = \gamma / i$. Denote for short $\overline{\rho_j} = \sum_k \rho_j^k s p_j^k$ and $\overline{\phi_j} = \frac{\mu}{\gamma} \lambda_j$. Summing (9) for all $i \in \{1, \ldots, K\}$:

$$\lambda_j - \overline{\phi_j} - \overline{\rho_j} = \left(\theta + \frac{\eta \mu}{s_j} - \frac{\mu \theta}{\gamma s_j} \right) \sum_i \frac{\overline{x_j^i}}{i}. \quad (11)$$

Additionally, Equation (9) can be re-written:

$$\lambda_j^i = \left(\theta + \frac{\eta \mu}{s_j} - \frac{\mu \theta}{\gamma s_j} + \frac{\overline{\rho_j} + \overline{\phi_j}}{\sum_i \frac{\overline{x_j^i}}{i}} \right) \frac{\overline{x_j^i}}{i} \quad (12)$$

Combining Expressions (11) and (12), we obtain (3).

VI. Acknowledgements

This paper was supported by PEDECIBA Informática and Universidad de la República, Montevideo, Uruguay. The authors wish to thank the five anonymous reviewers and Dr. Ymir Vigfusson for their constructive comments.

References

[1] B. Cohen, "Incentives build robustness in bittorrent," *www.bramcohen.com*, vol. 1, pp. 1–5, May 2003.

[2] "Bittorrent protocol specification v1.0," http://wiki.theory.org/BitTorrentSpecification, 2010.

[3] M. E. Bertinat, D. D. Vera, D. Padula, F. Robledo, P. Rodríguez-Bocca, P. Romero, and G. Rubino, "Goalbit: The first free and open source peer-to-peer streaming network," in *Proceedings of the 5th international IFIP/ACM Latin American conference on Networking (LANC'09)*. New York, USA: ACM, September 2009, pp. 83–93.

[4] GoalBit - The First Free and Open Source Peer-to-Peer Streaming Network, http://goalbit.sf.net/, 2008.

[5] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," in *Proceedings of SIGCOMM'04*, ACM. New York, NY, USA: ACM, September 2004, pp. 367–378.

[6] Y. Tian, D. Wu, and K.-W. Ng, "Analyzing multiple file downloading in bittorrent," in *Proceedings of ICPP'06*. IEEE, August 2006, pp. 297–306.

[7] P. Rodríguez-Bocca and C. Rostagnol, "Optimal download time in a cloud-assisted peer-to-peer video on demand service," in *Proceedings of the International Network Optimization Conference (INOC'11)*. London, UK: Springer, Lecture Notes in Computer Science, 13-16 June 2011.

[8] M. G. C. Resende and C. C. Ribeiro, *Greedy Randomized Adaptive Search Procedures*. Kluwer Academic Publishers: In F. Glover and G. Kochenberger, editors, Handbook of Methaheuristics, 2003.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1979.

[10] M. Martínez, A. Morón, F. Robledo, P. Rodríguez-Bocca, H. Cancela, and G. Rubino, "A GRASP algorithm using RNN for solving dynamics in a P2P live video streaming network," in *8th International Conference on Hybrid Intelligent Systems (HIS'08)*, Barcelona, Spain, September 10-12 2008. [Online]. Available: http://his2008.lsi.upc.edu/

[11] M. E. Bertinat, D. Padula, F. Robledo, P. Rodríguez-Bocca, and P. Romero, "A simple proactive provider participation technique in a mesh-based peer-to-peer streaming service," in *Proceedings of the 6th International Conference on Hybrid Artificial Intelligence Systems (HAIS'11)*. London, UK: Springer, Lecture Notes in Computer Science, 23-25 May 2011.

[12] T. Lust and J. Teghem, "The multiobjective multidimensional knapsack problem: a survey and a new approach," *Journals on Computers and Operations Research*, vol. abs/1007.4063, 2010.

[13] MATLAB - The Language of Technical Computing. Home Page., http://www.mathworks.com/, 2012.

[14] P. Rodríguez-Bocca, "Quality-centric design of Peer-to-Peer systems for live-video broadcasting," Ph.D. dissertation, INRIA/IRISA, Université de Rennes I, Rennes, France, April 2008. [Online]. Available: http://goalbit.sourceforge.net/publications/these.pdf

Summary Review Documentation for

"A new caching policy for cloud assisted Peer-to-Peer video-on-demand services"

Authors: Franco Robledo, Pablo Rodíguez-Bocca, Pablo Gabriel Romero, Claudia Rostagnol

REVIEWER #1

This paper presents a mathematical model based on fluid dynamics in order to improve download times in peer-based CDNs. An actual approach is developed and simulation based on a real data trace shows good scalability characteristics, although these could be presented better. The paper is very dense and therefore hard to read.

The scalability analysis seems a little iffy in places. Instead of saying "the performance is stable" (section 4, par 3) please tell us whether it scales linearly, exponentially, etc. The writing suffers from many minor language issues (e.g., in relation to prepositions) that can easily be fixed. Could the trace be made available to the research community? As mentioned, the paper is dense and I think it might have worked better in a longer format.

Strengths: Strong mathematical modeling and an evaluation on a real data trace.

Weaknesses: No comparison with other approaches. The discussion of scalability could be improved. It seems that the authors are more comfortable with mathematical modeling than scalability analysis. Not an easy read.

REVIEWER #2

The paper is a little hard to follow in the theoretical part. In particular, variable names should be chosen so they are acronyms of what they represent. The paper could be better written so a curious reader does not have to go through the gory details of the theory to understand what it is proposed and the actual solution. The evaluation is somewhat weak. I would have expected a simulation with different number of peers and super-peers to see how your solution behaves with different scales of the system.

Strengths: The paper provided a super-peer based system solution to the on-demand video distribution and it is compared with a traditional CDN solution and a pure client server solution.

Weaknesses: The theoretical analysis is hard to follow. Sometimes artificially, simply because the names of the variables have not been chosen as acronyms of what they represent (e.g. super-peers is "z"and could be "sp"). The contribution is a somewhat slim even for a short paper. Last, the evaluation is a little weak. A simulation with an increasing number of super-peers and peers would have been desirable taking value from the real evaluation.

REVIEWER #3

I think that the authors should make the difference between their work and prior art more obvious for the readers. How similar is the cashing policy compared to what exists today? What makes the authors think that peer behavior can be accurately modeled by a Markov chain? It would be very interesting to test this assumption in a practical setting, and I encourage the authors to consider it. The presentation of their results could be significantly improved.

Strengths: The authors tackle an interesting and important problem. The mathematical results appear to be sound. The preliminary results they obtained seem to indicate that the caching policy, although far from optimal, scales well as the number of requests per second increases.

Weaknesses: The only improvement over prior work is the consideration of the churn (attrition) rate of peers and seeders.No comparison with existing work is provided. For instance, the average download time is three times higher than what it would be for an ideal system. It this good or bad? The paper is poorly written and quite difficult to read.

REVIEWER #4

The paper addresses an interesting problem, proposes an interesting and appropriate extension to the model of Tian et al. [6] but feels incomplete. Rewriting the paper with a stronger focus on the task at hand, more extensive (ideally experimental) evaluation and much deeper treatment of related work (which is currently only briefly touched on) would give an extra edge.

Strengths: Our understanding of the large-scale behavioral aspects of BitTorrent and VoD are quite limited. The paper touches on an important scientific problem. The paper extends an interesting model to address the "how much to cache and where?" question for a realistic scenario: P2P-VoD system with additional super-peer nodes that help disseminate content within the system. The model is general enough to capture P2P, CDN and P2P/CDN hybrids. The authors provide an interesting optimization algorithm to reduce the average download time while making good use of the limited storage capacity of the super-peers. The algorithm is evaluated numerically, and compared to an intuitive baseline.

Weaknesses: The organization of the paper could be improved, and the grammar would benefit from a pass by a native English speaker (e.g. "The scheduling must attend videos' popularity, and looks forward to minimize the average download time for end-users"). However, the level of mathematical detail is quite readable and well presented. The model is not explicitly validated using real-world data. The evaluation section is preliminary. A more compelling empirical analysis would strengthen the paper, such as by (i) experimenting the algorithm on the live GoalBit system or on PlanetLab, (ii) measuring the algorithm execution time, overhead and determining its sensitivity to the parameters, and (iv) comparing the approach to other (simpler) algorithms or systems (such as Antfarm: Efficient Content Distribution with Managed Swarms (NSDI 2009)).

REVIEWER #5

I do not understand the motivation of this work. The paper starts by rejecting traditional CDNs for highly popular content distribution without *any* evidence. In fact, current commercial CDNs prosper in video distribution these days: about 30bandwidth in the residential area in the U.S. are related to on-line video streaming by Netflix, which is distributed by a number of commercial CDNs without many problem. Most popular video sites like YouTube, Hulu, DailyMotion, etc. are distributed by either an in-house CDN (Google) or other commercial CDNs. It is well known that P2P video distribution/streaming is very inefficient in terms of network resource usage (suboptimal peering, lack of upload bandwidth and efficient distribution tree) and it is much harder to enforce the quality of service for high-definition videos. It looks as if the authors cooked up the motivation to incorporate the mathematical expression development.

Why is efficient usage of storage capacity the most important aspect of the video distribution? The authors assume that in order to serve many clients, one needs to "generate more copies for the video in the cloud", but that is not true in general. Multiple CDN servers can share a networked-attached storage or they can distribute the chunks of the video file to multiple servers to reduce the storage space while providing the video by merging the chunks on the fly. The caching model that this assumes looks outdated where each server replicates the entire video file, which would increase the storage cost a lot.

Strengths: It attempts to formalize P2P video distribution into mathematical expressions.

Weaknesses: Unclear motivation.The text is hard to follow. Little explanation of the implication of the mathematical equations.

RESPONSE FROM THE AUTHORS

We carefully read and addressed all your comments. The manuscript suffered several improvements, specially in the combinatorial specification of the problem and its heuristic resolution. A real-life scenario is fully detailed and discussed in the last section, extending empirical results. Furthermore, the domination of the peer-to-peer system (Proposition II.1) holds in a more general context, and the proof was slightly modified to enrich the previous version. The Appendix had some algebraic mistakes, and we hope it is correct now.

The local Internet Service Provider is explicitely mentioned, and the log traces will be soon available to the scientific community. We know content delivery networks are largely deployed nowadays. However, we suspect our contribution re-reinforces the relevance of the peer-to-peer philosophy to assist pure-CDN deployments. A bit of effort has been deserved to improve the treatment of scalability and performance analysis.

Several grammatical mistakes were corrected, and the paper was enriched with new references.