# Using Centrality Metrics to Predict Peer Cooperation in Live Streaming Applications*

Glauber D. Gonçalves[1], Anna Guimarães[1],
Alex Borges Vieira[2], Ítalo Cunha[1], and Jussara M. Almeida[1]

[1]Computer Science Department, Universidade Federal de Minas Gerais, Brazil
[2]Computer Science Department, Universidade Federal de Juiz de Fora, Brazil
{ggoncalves,anna,jussara,cunha}@dcc.ufmg.br
alex.borges@ufjf.edu.br

**Abstract.** The lack of cooperation in Peer-to-Peer (P2P) applications poses serious challenges to the quality of service provided to their clients, specifically in P2P live streaming applications given their strict real-time constraints. We here investigate the potential of exploiting topological properties of the P2P overlay network to predict the level of cooperation of a peer, measured by the ratio of the upload to the download traffic during a pre-defined time window. Using data collected from SopCast, we first show that centrality metrics provide good evidence of a peer's cooperation level in the system. We then develop a regression-based model that is able to estimate, with reasonable accuracy, the level of cooperation of a peer in the near future given its centrality measures in the recent past. Our proposed strategy complements existing incentive mechanisms for cooperation in P2P live streaming, and can be applied to detect non-cooperative peers.

**Keywords:** Peer-to-peer live streaming, centrality metrics, cooperation level

## 1 Introduction

The peer-to-peer (P2P) architecture has emerged as a cost-effective platform for live video streaming on the Internet. Indeed, various P2P live streaming applications, such as SopCast, PPLive, and UUSee[1], have already reached the mark of millions of registered users [9]. Such systems are composed of clients (peers) that collaborate to disseminate the live media content by establishing partnerships and organizing themselves into an overlay network on top of the real computer network. A peer may request (and receive) chunks of the live media from its partners, thus releasing the central server from the responsibility, and ultimately from the associated costs, of serving all participant clients.

Given this architecture, the proper operation of P2P applications and ultimately the quality of service provided to the clients depend heavily on peer cooperation. However, a number of previous studies have already detected the presence of non-cooperative peers, also known as *free riders*, which download content from their partners but do not

* This work is supported by the INWeb (MCT/CNPq grant 57.3871/2008-6), and by the authors grants from CNPq and FAPEMIG

[1] http://www.sopcast.com, http://www.pptv.com, and http://www.uusee.com.

upload the received content to other peers [1]. The lack of cooperation may be caused by both opportunistic users, who consciously choose not to upload content to others, and by P2P protocols that induce unbalanced data exchanges among partners. The presence of non-cooperative peers may greatly degrade the overall system performance [20], particularly in P2P *live* streaming applications due to their strict time constraints [11]. For example, peers may experience video interruptions due to the lack of partners from which to download the live content, whereas altruistic peers, i.e., peers that upload more content than they received, thus serving multiple partners, may be overloaded with data requests and become unsatisfied with the system. Thus, maintaining estimates of the current level of cooperation among peers helps P2P application managers to identify potentially non-cooperative peers as well as altruistic overloaded peers, guiding them into future actions to guarantee the expected quality of service.

In this paper, we investigate whether the topological properties of a peer in the overlay network are related to its level of cooperation in P2P live streaming applications. In particular, we assess the potential benefit of exploiting centrality metrics to dynamically predict the level of cooperation of each peer. A peer's cooperation level is defined by the ratio of the total upload to the total download traffic the peer exchanged with its partners. Since a peer's cooperation level may change over time throughout the live transmission, it should be estimated periodically, during specific time windows.

Our study relies on data collected from one of the currently most popular P2P live applications, i.e., SopCast, using a large number of PlanetLab machines. It encompasses two main steps. We first show that centrality metrics [7] are reasonably strongly correlated with the peer's cooperation level and thus might be used to distinguish non-cooperative peers, i.e., peers that upload less than download, from the others. Moreover, we also show that a peer's centrality remains reasonably stable over consecutive 60-second time windows. Motivated by these findings, we then develop a regression-based model to predict the level of cooperation of a peer in the following time windows given its centrality measures collected in the last window. Using our collected data, we show that our approach can produce reasonably accurate predictions.

There are various techniques for detecting non-cooperative peers [2, 8, 14] and incentivizing peer cooperation [4, 16, 17] in live streaming applications. Our motivation for investigating the benefit of exploiting centrality metrics for that purposes lies in the observation that many P2P live applications keep a centralized server (*tracker*) that periodically receives control messages from each peer. In such messages, peers might include a list of their current partners [16], based on which the tracker could reconstruct the overlay topology and compute each peer's centrality. Moreover, some techniques collect the ratio of upload to download periodically from peers and use authentication mechanisms to be robust to malicious peers, which increases communication and computation overhead in the system. We believe that our strategy might be jointly applied with such techniques by complementing them and reducing their overhead [4, 16].

The rest of the paper is as follows. Section 2 discusses related work. Our SopCast data collection methodology is described in Section 3. The level of cooperation among SopCast peers is analyzed Section 4, whereas its correlation with peer centrality is discussed in Section 5. Section 6 presents our regression-based model, whereas related practical issues are discussed in Section 7. Section 8 concludes the paper.

## 2 Related Work

Low cooperation in P2P systems, or free riding, was first analyzed in the Gnutella file sharing system [1]. Since then, various studies have addressed the problem [2,8,15,20]. Most notably, incentive mechanisms that rely on bilateral peer contribution, such as Bittorrent's tit-for-tat [6] and further improvements [13,20], have been widely applied in file sharing systems. However, tit-for-tat may not be effective in live streaming because video segments become obsolete over time, which in turn prevents some peers from sharing every segment received [17].

FlightPath [14] is a P2P live streaming application with an incentive mechanism that uses tit-for-tat but also adds a relaxation parameter for the benefit of peers that are unable to contribute in a balanced way. Silverston *et al.* [17] proposed a different incentive mechanism where peers unable to serve data should answer with pointers to other peers that are able to serve. Chatzidrossos *et al.* [4] argued that many non-cooperative peers are unable to upload as much data as they download due asymmetric DSL or HSDPA connections. To maximize the social welfare, they proposed to reserve a fraction of server upload bandwidth to altruistic peers that upload more data than they download. Our model could be used by such incentive mechanisms to identify non-cooperative and altruistic peers, enabling the application to redirect traffic in its network. More than that, in the case of the incentive mechanism proposed by Silverston *et al.*, our model can indicate the amount of data a peer can serve in the near future based on its predicted cooperation level.

LiFTinG [8] is a technique for tracking free riders that consists of a series of checking and cross-checking operations performed by peers to verify whether data requests are served by their partners and whether the partners forwarded received data to their own partners. Failed verifications decrease the score of a peer, and peers with low score may be considered suspect and be banned from the system. Azzedin [2] proposed that peers keep a black list of untrustworthy contributors and send information about corrupted content received from their partners to monitoring peers. This information is then used to detect free riders and content polluters (i.e., peers that inject corrupted data in the system). Our model may be used by trust systems as an additional information source: peers reporting cooperation levels significantly higher than what the model predicts based on their out-degrees can be put in a list of suspect peers to check.

More related to our work is the centralized version of the Contracts incentive mechanism [16]. Contracts uses estimates of each peer's cooperation level to restructure the overlay such that altruistic peers are placed closer to the video source, thus having a better quality of service. At each peer, Contracts collects cryptographic receipts for uploaded data and sends them periodically to the tracker. The tracker processes receipts to compute each peer's cooperation level, which incurs in high processing cost. Our method could help reducing communication and computational overhead incurred by Contracts, as discussed in Section 7.

## 3 SopCast Data Collection Methodology

Our study relies on a set of traffic logs collected from SopCast, a currently very popular P2P live streaming application. SopCast maintains a number of *channels*, each

one transmitting live content over its own P2P overlay network, independent from the others. The SopCast P2P architecture is based on a data-driven mesh network [21], in which clients (peers) establish partnerships among themselves building a mesh like overlay network. A server, which generates the content, splits the media into *chunks* and transmits them over the network. To receive the live content, a peer explicitly requests each needed chunk from one of its partners. SopCast keeps a list of public channels but also allows the creation of private live channels, accessible to a restricted set of clients.

To collect data from SopCast, we used a set of PlanetLab machines [5] acting as both regular SopCast clients and data crawlers. Our crawling strategy follows the one used in [18], where each crawler collects and stores data regarding all packets it exchanges with its partners, and these traces are later merged to reconstruct the overlay network. We chose to collect data from a *private* channel of our own, restricting the clients to our PlanetLab crawlers, so as to be able to collect a complete view of the overlay network and compute exact measures of the centrality and cooperation level of each peer.

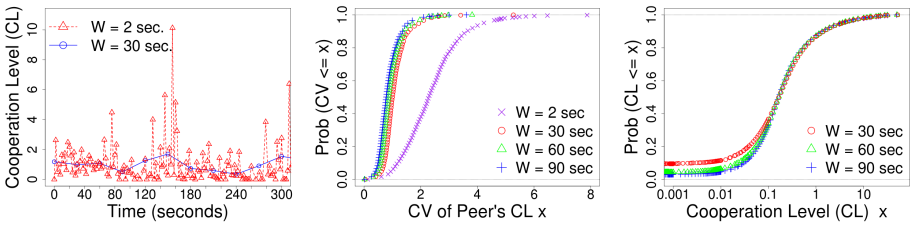
To build our private SopCast channel, we set up a server to encode and transmit a 1-hour 280 kbps video. We performed 6 experiments in November 2011. For each experiment, all crawlers joined the SopCast channel at the same time and remained connected through an initial 5-minute interval. Afterwards, our crawlers start mimicking the behavior of real peers by dynamically leaving and rejoining the channel (i.e., churn). The churn model adopted was based on a previous characterization of the dynamic behavior of real SopCast clients on a popular public channel [19]: each peer remains connected in the channel for a certain period $ON$ of time; afterwards, it leaves the system remaining inactive for a period $OFF$ of time after which it rejoins the channel. According to our analyses of real data collected during various transmissions, $ON$ times are well modeled by a Weibull distribution with parameters $\alpha = 2.032$ and $\beta = 0.233$, whereas $OFF$ times follow an Exponential distribution with parameter $\lambda = 0.054$[2]. While connected to the channel, each crawler uses Wireshark[3] to collect all SopCast related traffic. During our experiments, we used the largest number of crawlers possible, which varied between 350 and 450 stable PlanetLab nodes. We also checked clock drift among crawlers to guarantee that time differences could be neglected (less than 1 second).

Each crawler collected the timestamp (at the 1-second granularity) and the size of each packet sent (received) to (from) other SopCast clients. As our focus is on *data* packet exchanges, we only stored information for packets containing at least 1300 bytes, a threshold selected based on previous studies of SopCast and PPLive [9, 18].

For each experiment, once the monitoring period finished, we merged the log files created by all crawlers to reconstruct the overlay network during the experiment. We discarded an initial 5-minute warm-up phase, analyzing only data collected afterwards (during peer churn). We used the time information and the source and destination IP addresses of each packet to reconstruct the SopCast overlay network as a sequence of snapshots. More specifically, we took consecutive snapshots of the network, each one built from data collected during a pre-defined time window with duration $W$. In the following, we analyze peer cooperation level for all time windows of all 6 experiments.

[2] The Probability Density Functions are: $p_X(x) = \alpha\beta x^{\beta-0} e^{-\alpha x^\beta} I_{(0,\infty)}(x)$ for Weibull, and $p_X(x) = \lambda e^{-\lambda x}$ for Exponential.

[3] http://www.wireshark.org




(a) CL for a Selected Peer (b) Coef. of Variation of CLs (c) Distribution of CL values

**Fig. 1.** Peer Cooperation Level (CL) for Various Window Durations ($W$).

## 4 Peer Cooperation in SopCast

To analyze peer cooperation in SopCast during our experiments, we start by defining the *cooperation level* of peer $i$ during time window $t$, denoted $CL(t, i)$, as the ratio between the total number of bytes uploaded and downloaded by $i$ during $t$.

We then analyze $CL$ values as a function of the time window duration $W$. In practice, we want a duration that is neither too short nor too long, so as to smooth out great variations at very short time scales while still capturing dynamic peer properties over the course of a transmission. We tested with $W$ equal to 2, 30, 60 and 90 seconds. As an example, Figure 1(a) shows the CL values of a selected peer measured over consecutive windows, for $W$ equal to 2 and 30 seconds. The peer's CL fluctuates greatly for $W$=2, whereas for $W$=30 the curve shows a much smoother behavior. We quantify this variability with the coefficient of variation (CV) (i.e., the ratio of the standard deviation to the mean) of all measured CL values. Figure 1(b) shows the distributions of CVs, computed for each peer across all time windows in our experiments, for each duration $W$. Clearly, the variability is greater for $W$=2, whereas for $W \geq$30, the distributions are very close to each other. Thus, we focus on $W$ equal to 30, 60, and 90 seconds.

We now analyze the distribution of CL values across all peers during all time windows of all experiments. Figure 1(c) shows CL values with logarithmic scale on the x-axis. Note that the distributions are very similar for W equal to 30, 60, and 90 seconds. Indeed, 2% of the measured CL values are greater than 10 for all three values of $W$. In general, we observe two extreme peer behaviors. On one side, there are few altruistic peers that upload much more than they download. On the other side, there are many non-cooperative peers that upload much less than they download. In particular, Figure 1(c) shows that 34% of all measured CL values are below 0.1, corresponding to very uncooperative peers which, despite downloading the complete video, forwarded no more than one-tenth of it to their partners. These results illustrate how unbalanced load distribution is in SopCast.

Considering as non-cooperative during time window $t$ a peer $i$ with $CL(t, i) < 1$ (as in [15]), Figure 1(c) shows that, during a time window, the SopCast network contains around 87% of non-cooperative peers, on average. Even if we assume a stricter definition, with a non-cooperative peer $i$ being such that $CL(t, i) < 0.5$ (or $CL(t, i) < 0.2$), the fraction of non-cooperative peers during a window $t$ is 79% (56%), still quite large. These observations indicate that it is very likely that SopCast does not implement any

load balancing mechanism among peers, as already observed for PPLive [16]. The lack of such mechanism is a concern because cooperative peers (particularly very altruistic peers) may find themselves discouraged to participate in the network due to the high bandwidth consumption. Moreover, the large presence of non-cooperative peers may lead to extra delays, stream interruptions, and ultimately system collapse [16].

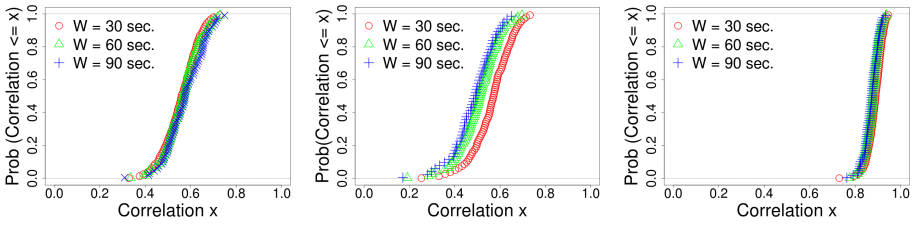
## 5 Peer Centrality and Cooperation Level

Our goal in this paper is to assess the potential benefit of using a peer's centrality measures to predict its cooperation level in the near future. More specifically, we want to assess the accuracy of using centrality measures of peer $i$ during window $t$ to predict its CL value in window $t+1$. This idea rely on two hypotheses: (1) a peer's centrality is correlated with its CL, and (2) a peer's centrality remains roughly stable over consecutive windows. We here investigate whether these hypotheses hold in our dataset.

We consider 3 commonly used centrality metrics, namely, out-degree, closeness, and betweenness, taking normalized measures so as to make them comparable across different time windows during which the number of clients in the network may change. The *out-degree* of a peer $i$ during window $t$, denoted by $d(t,i)$, is the number of partners to which $i$ uploaded media chunks during $t$. We define normalized out-degrees as $\overline{d}(t,i) = d(t,i)/(|\mathcal{P}(t)|-1)$, where $\mathcal{P}(t)$ is the set of network peers during $t$. The *closeness* of peer $i$ during $t$, $c(t,i)$, is the inverse of the average shortest path length between $i$ and all the other peers in the network during window $t$ [7]. Let $\delta(t,i,j)$ be the *length* of the shortest path between nodes $i$ and $j$ during $t$, then $c(t,i) = (|\mathcal{P}(t)|-1)/\sum_{j\in\mathcal{P}(t)}\delta(t,i,j)$. Note that $c(t,i) \in (0,1]$ regardless of network size, being already normalized. The *betweenness* of peer $i$ during $t$, $b(t,i)$, is the fraction of all shortest paths connecting pairs of nodes in the network that pass through $i$. Let $\sigma(t,j,k)$ be the *number* of shortest paths between nodes $j$ and $k$ during $t$ and $\sigma(t,j,k,i)$ be the number of those paths that pass through $i$, then $b(t,i) = \sum_{i,j,k\in\mathcal{P}(t),i\neq j\neq k}\sigma(t,j,k,i)/\sigma(t,j,k)$. Betweenness values increase with the number of pairs in the network, so we normalize them by the number of pairs of peers excluding $i$, which gives $\overline{b}(t,i) = 2b(t,i)/((|\mathcal{P}(t)|-1)(|\mathcal{P}(t)|-2))$ [7].

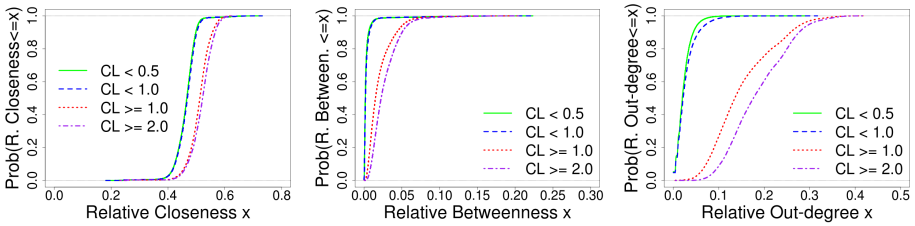
We disregard edge direction to compute closeness and betweenness, as Freeman [7]. Edge directions could partition the graph and lead to inconsistencies when computing these 2 metrics. In particular a peer that uploads more than it downloads could become unreachable from the majority of peers in the network, be traversed by few shortest paths, and have low betweenness. We only consider the direction of the edge to compute out-degree because in this case, direction captures information about the data flow.

We analyze the relationship between each centrality metric and cooperation level using the Spearman correlation coefficient, which is a non-parametric measure of statistical dependence [12]. We compute the correlation between peer centrality and CL for each time window. Figure 2 shows the distributions of the Spearman coefficients for all time windows in all experiments for different values of $W$. For all three metrics, window duration has small impact on the correlation coefficients, which are strictly positive and usually larger than 0.5. Thus, all three metrics may be useful to predict CL values. However, out-degree clearly has a significantly stronger correlation.




(a) $CL$ and Closeness (b) $CL$ and Betweenness (c) $CL$ and Out-Degree

**Fig. 2.** Distributions of Spearman Correlation Between Centrality and Cooperation Level (CL)




(a) Closeness (b) Betweenness (c) Out-degree

**Fig. 3.** Distributions of Centrality Metrics as a Function of Cooperation Level (CL)

We also analyze the capacity of each centrality metric to distinguish non-cooperative peers (i.e., with $CL<1$) from cooperative peers (i.e., with $CL \geq 1$). Figure 3 shows the distributions of closeness, betweenness, and out-degree for these two sets of peers over all time windows for $W$=60. Results for other values of $W$ are quantitatively similar, being thus omitted. All three metrics have very distinct distributions for cooperative and non-cooperative peers: non-cooperative peers tend to have much smaller closeness, betweenness, and out-degree. Once again out-degree stands out as the most discriminative metric. Figure 3 also shows the distributions for stricter definitions of non-cooperative peers with $CL<0.5$ and for cooperative peers with $CL \geq 2$. The same trend holds, providing further evidence that centrality metrics, in particular out-degree, may be useful to predict, with reasonable accuracy, peer cooperation levels.

We now turn to our second hypothesis, and analyze the difference between each peer's centrality in two windows $t$ and $t + k$, for positive $k$. We focus only on out-degree as, according to Figures 2-3, it is the most promising metric. Figure 4 shows the distributions of the differences of normalized out-degrees over all peers in all windows for various values of $k$ and $W = 60$s. Note that out-degrees remain reasonably stable for small values of $k$ (e.g., $k \leq 8$). For instance, for $80\%$ of the peers analyzed over pairs of windows that are 8 minutes apart from each other ($k = 8$), the difference in the normalized out-degree is less than $4\%$ of the total number of peers in the network. Thus, peer centrality remains reasonably stable across a few successive time windows.

## 6 Predicting a Peer's Cooperation Level

Given our findings in the previous section, we now study whether the centrality of a peer measured in window $t$ can help us predict its cooperation level in window $t + 1$.

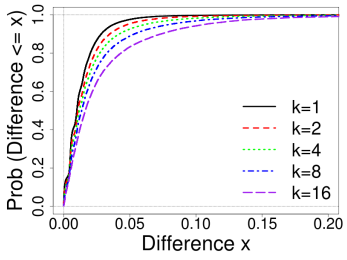



**Fig. 4.** Differences in Normalized Peer Out-Degree for Windows $t$ and $t+k$ ($W$=60).

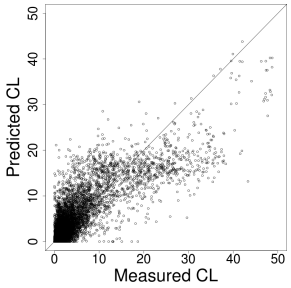



**Fig. 5.** Predicted vs. Measured CL ($W$=60)

By visual inspection, we identified that a peer's CL is a non-linear, monotonically increasing function of each considered centrality metric. This led us to evaluate non-linear regression models for computing CL. Let $m(t,i)$ denote a (normalized) centrality metric (out-degree, betweenness, or closeness). We evaluated polynomial and exponential models of the forms:

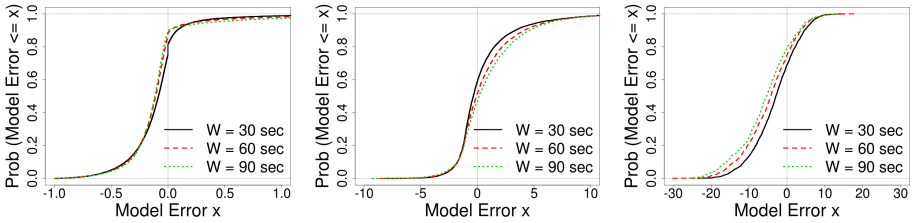
$$\widehat{CL}(t+1,i) = \sum_{j=0}^{4} a_j m^j(t,i) \quad \text{and} \quad \widehat{CL}(t+1,i) = a_0 + a_1 m(t,i) + b e^{m(t,i)}.$$

In both cases, we considered functions of a single centrality metric as well as functions combining multiple metrics. We computed model parameters $a_j$ and $b$ using curvilinear regression [10], where one applies a linear transformation to the predictor and response variables, i.e., $m(t,i)$ and $CL(t,i)$, and then computes the parameters minimizing the sum of squared residual errors. The estimates are produced in successive time windows as follows: model parameters $a_j$ and $b$ are computed based on centrality and CL values measured in window $t-1$, the model is applied using peer's centrality measured during window $t$, and its accuracy is evaluated by comparing, for each peer, the predicted CL against the CL measured in the next time window $t+1$. We quantify prediction accuracy using the prediction error $\widehat{CL}(t+1,i) - CL(t+1,i)$. Negative $\widehat{CL}(t+1,i)$ are truncated to zero, as negative cooperation does not make sense.

In the interest of space, we present a summary of our findings in the evaluation of the various models tested. As expected from the results in Figures 2 and 3, the models using out-degree have higher prediction accuracy among the three centrality metrics. Moreover, the exponential models usually have lower prediction accuracy as they systematically underestimate the cooperation level of altruistic peers, whereas cubic and quadric models have accuracy equivalent to quadratic models. Thus, we opted for the latter (simpler) ones. Finally, the intercept $a_0$ is usually statistically equivalent to zero, which is intuitive, as a peer's contribution is zero when it is not in the network (i.e., $m(t,i)$=0). Thus, we focus our following discussion in our best-performing model[4]:

$$\widehat{CL}(t+1,i) = a_1 \overline{d}(t,i) + a_2 [\overline{d}(t,i)]^2. \tag{1}$$

[4] This model has the advantages of taking only a peer's out-degree as input and requiring calibrating only two parameters. This reduces the computational cost, compared to, e.g., a model that requires computing betweenness, which has complexity $O(VE)$ for a network with $V$ nodes and $E$ edges [3], and might have limited practical deployability.




(a) $CL < 1$ (b) $1 \leq CL < 10$ (c) $CL \geq 10$

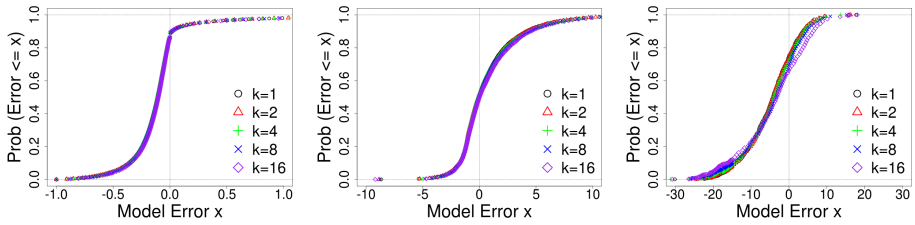
**Fig. 6.** Distribution of Prediction Errors for Various Ranges of Measured Cooperation Levels

We start analyzing the accuracy of our model by plotting, for each peer $i$ and window $t$, the measured and predicted cooperation levels. Figure 5 shows results for all peers and windows with $W$=60. The trend towards linearity is clear: a Pearson correlation coefficient [10] $\rho$=0.87 indicates strong positive correlation. Similarly, we obtained $\rho$ equal to 0.90 and 0.84 for $W$ equal to 30 and 90 seconds. Despite producing errors in a few cases, most predictions have reasonable accuracy. Indeed, Figure 6 shows the distributions of errors computed over all peers and windows, for $W$ equal to 30, 60 and 90. To improve readability, we group prediction errors based on the measured cooperation level: predictions for non-cooperative peers with $CL < 1$, predictions for cooperative peers with $1 \leq CL < 10$, and predictions for very altruistic peers with $CL \geq 10$.

In general, the first two sets of curves show that prediction errors are very concentrated around zero and that the choice of $W$ has little impact on them, although $W$ might impact whether the model tends to overestimate or underestimate the cooperation level. For instance, considering the predictions for non-cooperative peers ($CL$<1), around 68%, 67% and 66% of the predictions are reasonably accurate, falling within $\pm 0.2$ of the measured $CL$ values for $W$ equal to 30, 60 and 90 seconds, respectively. However, the fraction of predictions that underestimate the measured $CL$ tends to be larger for larger $W$. For example, Figure 6(a) shows that the fraction of negative errors (i.e., underestimates) is 84% for $W$=90, but only 64% for $W$=30. Similarly, most predictions for peers with $1 \leq CL < 10$ are within 2 units of the measured values. Indeed, 76%, 71% and 66% of the model errors fall in this range for $W$ equal to 30, 60 and 90 seconds. However, unlike observed for non-cooperative peers, the model tends to underestimate the real $CL$ values more for larger $W$.

The predictions for the very altruistic peers are somewhat less accurate: around 55%, 48% and 43% of the errors are within 5 units of the measured CLs for $W$ equal to 30, 60 and 90 seconds. The smaller number of very altruistic peers in the network (2%) makes it hard to build a model that is very accurate for such peers compared to the other two groups. We note, however, that our model is able to correctly identify most very altruistic peers: around 85% of the predictions for these peers produced CL estimates that fall in the expected range (i.e., $\widehat{CL} \geq 10$), and most remaining $\widehat{CL}$ estimates fall between 1 and 10. It is also able to identify most cases of non-cooperation: only 3% of predictions for peers with $CL < 1$ had $\widehat{CL} > 1$. In future work, we intend to exploit other topological properties to improve the prediction for very altruistic peers.

So far, we have discussed the accuracy of the regression model when parameters $a_j$ are computed at each window $t$ given the centrality measures collected during that




(a) $CL < 1$ (b) $1 \leq CL < 10$ (c) $CL \geq 10$

**Fig. 7.** Distribution of Prediction Errors as Function of Model Calibration Period $k$ ($W = 60$s)

window. We now consider the case where the model parameters are computed using data collected at window $t$ and applied at the following $k$ windows, i.e., $t+1$, $t+2$, $\cdots$, $t+k$. Note that we still collect centrality measures at each window and use them to predict cooperation levels during the next window. Our goal is to analyze the sensitivity of the model to the frequency at which it must be calibrated (i.e., parameters must be recomputed), where $k$ defines the time interval between two consecutive calibrations.

Figure 7 shows the distributions of prediction errors for various values of $k$ and $W$=60. Once again, we show results separately for each range of measured CLs. The impact of $k$ on the errors can be barely noticed, particularly for $CL$<10 (Figures 7(a-b)). This implies that model parameters can be computed once and applied repeatedly for the following 16 minutes, at least. The model is a little more sensitive to long calibration periods when estimating the CL of very altruistic peers (Figure 7(c)). For this range, the errors tend to increase slightly for $k$>16. This is because some very altruistic peers experience frequent changes in their centrality measures. We conjecture that such changes might be due to the policy, adopted by many current applications, of peers randomly selecting new neighbors every so often [8, 21]. Chances are that a peer that loses some of its partners ends up creating new partnerships with such very altruistic peers, as they are more willing to accept them. However, even for very altruistic peers, the impact of increasing $k$ to 16 on model accuracy is small and might be acceptable.

## 7 Practical Considerations

Our results in Section 6 indicate that centrality metrics, notably out-degree, may be used to predict a peer's cooperation level with reasonable accuracy. We now address three factors that impact its practical deployability: how to collect inputs for model calibration and application, processing overhead at the tracker, and robustness to malicious nodes.

To build (calibrate) and apply our model, a central participant must collect each peer's out-degree and cooperation level. This could be performed by the tracker which, in many current P2P applications, already periodically receives control messages from all peers. Indeed, we observed, during our SopCast experiments, control message exchanges between peers and the tracker every 2 minutes. A peer could piggyback the list of its current partners in these control messages, which would suffice for the tracker to compute the whole overlay topology. Such piggybacking would incur in a small bandwidth overhead as peers usually have only a few tens of partners [9] (15.65 on average in our SopCast experiments), each of which could be represented by only 4 bytes. For

the sake of illustration, if there are 10,000 pairs in the network informing their partners every 2 minutes, the aggregate overhead at the tracker is only 5Kbps.

This overhead is small given that the knowledge of the overlay topology is of interest to system administrators, who are often concerned about the traffic between peers, and thus is expected from a well monitored system. Indeed, in some applications [21], peers already periodically exchange information that provide a partial view of the network. In such systems, the partnership information could be collected by a few monitoring peers (as in [2]) which jointly could reconstruct the complete network.

The second required input, each peer's cooperation level, could be sent to the tracker by the peer itself, at the cost of extra 4 bytes, or, alternatively, peers could send their upload to each partner, which induces an overhead equivalent to that of the list of partners. However, note that, this could be performed at longer intervals, since, according to Figure 7, model calibration at each 16 minutes still yields reasonably accurate results.

Calibrating the prediction model at the tracker requires few computational resources, as the out-degree requires no extra computation once the topology is available, and the regression algorithms to compute parameters $a_1$ and $a_2$ are efficient [10]. For example, we can calibrate our model in GNU R for one million of peers in about 3 seconds.

Regarding malicious peers, we note that one peer alone would not be able to deceive the system, as a partnership must be reported by both communicating peers. Moreover, as discussed in [16], a collusion attack by peers trying to appear as more cooperative than they really are could be detected as these peers would appear as a cluster in the overlay. A collusion of peers to promote a single peer is harder to detect. One approach is to analyze the history of partnerships of a peer to identify bias, as proposed by Guerraoui [8]. Another approach against malicious peers is to use cryptographically-signed messages, like in Contracts [16], to allow the server to verify partnerships and cooperation levels reported by peers. In this case, our model could be used jointly with Contracts, reducing the volume of cryptographic receipts and thus computational and communication costs. Instead of collecting cryptographic receipts for each sequence of packets exchanged, as originally proposed, the tracker could infer the authenticity of the out-degree collected from each peer by first checking the reports of communicating peers, and requesting cryptographic receipts only to audit lying or omissive peers.

Finally, one question that may arise is why not simply send the measured $CL$ values directly to the tracker instead of relying on centrality measures to predict them. The main reasons for not doing that are: (1) partnership information is already available or could be easily obtained by the tracker in most current applications, and (2) frequently sending measured $CL$ values, either aggregate values (as used here) or per-partner values, would possibly incur in extra communication overhead, if authentication mechanisms are applied (as in [16]), or be less resilient to lies and collusion attacks, otherwise.

## 8 Conclusions and Future Work

We investigated the correlation between a peer's cooperation level $CL$ and its centrality in the overlay network of a P2P live streaming application. Using data collected from SopCast, we first showed that three centrality metrics, namely out-degree, closeness, and betweenness, are significantly correlated with $CL$, although the former presents

stronger correlations and is able to better discriminate non-cooperative and cooperative peers. We then proposed a non-linear regression model that uses previous measures of out-degree to predict the peer's $CL$ value in the near future. Our model produces reasonably accurate estimates in most cases, being able to correctly identify 97% of the non-cooperative peers and 85% of the very altruistic peers. Thus, it can be applied jointly with existing techniques, providing evidence to help identify such peers and to drive incentive mechanisms that rely on estimates of peer cooperation [4, 16].

Future directions include analyzing our model under approximate inputs, larger and dynamic client populations, and collusion attacks.

## References

1. Adar, E., Huberman, B.: Free Riding on Gnutella. First Monday (2000)
2. Azzedin, F.: Trust-Based Taxonomy for Free Riders in Distributed Multimedia Systems. In: Proc. HPCS (2010)
3. Brandes, U.: A Faster Algorithm for Betweenness Centrality. Journal of Mathematical Sociology 25 (2001)
4. Chatzidrossos, I., Dán, G., Fodor, V.: Server guaranteed cap: An incentive mechanism for maximizing streaming quality in heterogeneous overlays. NETWORKING (2010)
5. Chun, B. *et al.*: PlanetLab: An Overlay Testbed for Broad-Coverage Services. ACM SIGCOMM Computer Communication Review (2003)
6. Cohen, B.: Incentives Build Robustness in BitTorrent. In: Workshop on Economics of Peer-to-Peer Systems (2003)
7. Freeman, L.: Centrality in social networks conceptual clarification. Social Networks (1979)
8. Guerraoui, R., Huguenin, K., Kermarrec, A.M., Monod, M.: LiFTinG: Lightweight Freerider-Tracking Protocol in Gossip. In: Proc. MIDDLEWARE (2010)
9. Hei, X., Liang, C., Liang, J., Liu, Y., Ross, K.: A measurement study of a large-scale p2p iptv system. Multimedia, IEEE Transactions on 9(8), 1672 –1687 (dec 2007)
10. Jain, R.: The Art of Computer Systems Performance Analysis:Techniques for Experimental Design, Measurement, Simulation, and Modeling. Wiley-Interscience (1991)
11. Karakaya, M., Korpeoglu, I., Ulusoy, O.: Free Riding in Peer-to-Peer Networks. IEEE Internet Computing 13(2) (2009)
12. Kendall, M., Gibbons, J.: Rank Correlation Methods. A Charles Griffin Title (1990)
13. Levin, D., LaCurts, K., Spring, N., Bhattacharjee, B.: BitTorrent is an Auction: Analyzing and Improving Bittorrent's Incentives. In: Proc. ACM SIGCOMM (2008)
14. Li, H., Clement, A., Marchetti, M., Kapritsos, M., Robison, L., Alvisi, L., Dahlin, M.: FlightPath: Obedience vs. Choice in Cooperative Services. In: Proc. OSDI (2008)
15. Locher, T. *et al.*: Free Riding in BitTorrent is Cheap. In: Proc. HotNets (2006)
16. Piatek, M., Krishnamurthy, A., Venkataramani, A., Yang, R., Zhang, D., Jaffe, A.: Contracts: Practical Contribution Incentives for P2P Live Streaming. In: Proc. USENIX NSDI (2010)
17. Silverston, T., Fourmaux, O., Crowcroft, J.: Towards an Incentive Mechanism for Peer-to-Peer Multimedia Live Streaming Systems. In: Proc. Peer-to-Peer Computing (2008)
18. Tang, S., Lu, Y., Hernández, J.M., Kuipers, F.A., Mieghem, P.V.: Topology Dynamics in a P2PTV Network. In: Proc. Networking (2009)
19. Vieira, A., Gomes, P., Nacif, J., Mantini, R., Almeida, J.M., Campos, S.: Characterizing SopCast Client Behavior. Tech. Report RT.DCC.002/2012, www.dcc.ufmg.br/∼borges (2011)
20. Xia, R., Muppala, J.: Discovering Free-Riders Before Trading: A Simple Approach. In: Proc. IEEE ICPADS (2010)
21. Zhang, X., Liu, J., Li, B., Yum, T.: CoolStreaming/DONet: A Data-Driven Overlay Network for Efficient Live Media Streaming. In: Proc. IEEE INFOCOM (2005)