Resource-constrained P2P Streaming Overlay Construction for Efficient Joining Under Flash Crowds

Eliseu César Miguel*, Ítalo Cunha†, Cristiano M. Silva‡, Fernando Carvalho†, Sérgio V. A. Campos†
*Instituto de Ciências Exatas, Universidade Federal de Alfenas (UNIFAL-MG), Brazil
†Departamento de Ciência da Computação, Universidade Federal de Minas Gerais (UFMG), Brazil
‡Departamento de Tecnologia, Universidade Federal de São João Del-rei (UFSJ), Brazil
eliseu.miguel@unifal-mg.edu.br, cunha@dcc.ufmg.br, cristiano@ufsj.edu.br, fccoelho@dcc.ufmg.br, scampos@dcc.ufmg.br

Abstract—**Video streaming now amounts to the majority of traffic in the Internet. Media streaming relies on large-scale content distribution networks (CDNs), that incur significant costs to build or use. P2P distribution of video content reduce reliance on CDNs and costs. Unfortunately, P2P distribution is fraught with QoE problems, specially during flash crowds or in scenarios where users have limited bandwidth to contribute to the overlay. In this paper, we propose a new P2P overlay construction mechanism to speed up peer joining during flash crowd events while preserving QoE for peers already in the overlay. We also show that our techniques work on resource-constrained overlays where a fraction of peers lack resources to contribute to the overlay, e.g., users on mobile devices and metered connections.**

I. Introduction

P2P networks allow live streaming to large audiences without relying entirely on (geographically-distributed) server upload bandwidth. In P2P networks, peers redistribute received content to other peers to improve system scalability and reduce infrastructure costs. Real systems support thousands of simultaneous users in multiple media distribution channels [1, 2]. Peers create partnerships in a decentralized way, forming *ad-hoc* overlay mesh topologies over the physical network for exchanging media content.

In the context of P2P media distribution, many interesting challenges arise in overlay topology maintenance and media redistribution strategies. For example, peer *churn*, caused by peers joining and leaving the overlay, break partnerships and disrupt media distribution [3]; *uncooperative* peers, which do not or cannot contribute with media redistribution (also known as *free riders*), increase resource competition in the overlay [4]. Such approaches often involve sophisticated overlay maintenance [5,6] or request scheduling [7,8].

Different from previous approaches, we investigate the ability of simple mechanisms that impose constraints on peer partnerships to build more robust and efficient overlays.

We propose a new P2P overlay construction mechanism to speed up peer joining on resource-constrained overlays during flash crowd events while preserving quality of experience (QoE) for peers already in the overlay. Our technique, *peer partnership constraints (PPC)*, groups peers into classes by considering each peer's contribution to media redistribution (upload bandwidth) and constrains which pairs of classes can establish partnerships.

PPC improves overlay distribution efficiency by bringing peers in classes with higher media redistribution scores closer to the server, as proposed in previous work [5,6,9]. However, PPC achieves this with simpler mechanisms that do not increase communication overhead or implementation complexity. To alleviate the impact of uncooperative peers, PPC puts uncooperative peers in a special class that is pushed to the edge of the distribution overlay.

PPC speeds up peer joining rate during flash crowds by reducing competition for partnerships in the overlay. As each peer class can only establish partnerships with a few select peer classes, peers can promptly find peers to establish partnerships with. Moreover, as uncooperative peers join the overlay at its edge, they do not disrupt media distribution for peers already in the overlay.

We run real experiments on PlanetLab varying overlay configurations and construction strategies. Our experiments show that PPC improves on traditional overlay construction strategies that do not restrict partnerships in the overlay: PPC is able to join a large number of peers during flash crowds while maintaining QoE even for resource-constrained overlays containing a significant share of free riders.

The remainder of this work is organized as follows. We describe TVPP, the real P2P streaming system we use in our work, in Section II. Section III introduces PPC. We explain our experimental method in Section IV and present our results in Section V. Section VI discusses related work. We offer conclusions in Section VII.

II. P2P Live Streaming System

A P2P live streaming system can be defined as a set of *peers* that collaborate with each other to disseminate live media transmissions. The *server* S is a special peer that encodes the video, splits the video into *chunks* (a chunk may contain multiple frames), and starts distribution of each chunk. In this section we present TVPP [10], a research-oriented P2P live streaming system used along our study. Each peer p has a set of *partners* $\mathcal{N}(p)$ for the exchange of video chunks. To get more control over partnerships, $\mathcal{N}(p)$ is split between two subsets of partner peers: $\mathcal{N}_i(p)$ containing *in-partners*, partners that provide chunks to p, and $\mathcal{N}_o(p)$ containing *out-partners*, partners that receive video chunks from p.

The maximum number of in-partners is denoted by $N_i(p)$. Similarly, the maximum number of out-partners is denoted by $N_o(p)$. For S, $\mathcal{N}_i(p) = \emptyset$. In order to join a live streaming channel, a peer p registers itself at a centralized *bootstrap* server B, which returns to p a subset of all peers currently active in the system as potential partners. Peer p selects peers from this subset and tries to establish partnerships with them. If $p \in \mathcal{N}_o(p')$, then $p' \in \mathcal{N}_i(p)$. We define an overlay's *partnership ratio*, denoted ρ, as the ratio of the total number of in-partners by the total number of out-parters, i.e., $\rho = T_{in}/T_{out}$ where $T_{in} = \sum_p N_i(p)$ and $T_{out} = \sum_p N_o(p)$ (for all peers). We say the network has in-partner surplus when $\rho > 1$, and out-partner surplus when $\rho < 1$.

Successfully established partnerships determine $\mathcal{N}(p)$. When p detects that one of its partners $p' \in \mathcal{N}(p)$ has been silent for longer than a predefined time period, p removes p' from $\mathcal{N}(p)$. However, peer p periodically contacts the bootstrap server to obtain a new list of potential partners to replace lost partnerships.

Peers randomly choose in-partners from peer lists received from the bootstrap server to connect to. To guarantee a new peer n is able to join the overlay, an existing peer p is able to accept an incoming partnership request even when $\mathcal{N}_o(p)$ is full. In this case, p disconnects a random out-partner $q \in \mathcal{N}_o(p)$ with less out-partnerships than the new peer n, i.e., $N_o(q) < N_o(n)$. Afterwards, peer p remains unable to disconnect more out-partners to accept incoming partnership requests during the next τ = 60s to prevent overlay instability.

Each peer has a local buffer to store video chunks. Periodically, peers exchange *buffer maps* reporting available chunks with their partners in $\mathcal{N}_o$. Complementarity, peers keep track of chunks not yet received in order to identify the partners in $\mathcal{N}_i$ able to serve those chunks.

In the scope of this work, we schedule chunk requests using the earliest deadline first policy together with the *Simple Unanswered Request Eliminator* (SURE) [11]. SURE makes peers prefer to request chunks to in-partners with less unanswered (timed-out) requests, balancing load, and reducing the number of unanswered requests. A peer considers that a request has timed out if it is not answered within 500 milliseconds. Finally, cooperative peers immediately serve received requests in order of arrival.

Peers send monitoring reports to the bootstrap server every 10 seconds. The reports include, among other information: (i) number of chunks generated (only reported by the video server); (ii) number of received chunks; (iii) number of sent chunks; and (iv) number of chunks that have missed their playback deadline.

III. Peer Partnership Constraint

The main idea in *Peer Partnership Constraint* (PPC) is the definition of peer classes. A peer class groups peers with similar potential for contributing in terms of chunks to the network. For each class, we impose peer partnership constraints in order to construct an efficient overlay topology.

We consider a scenario where users may be unable to contribute to stream distribution, e.g., mobile users on a metered connection or users with limited upload bandwidth. Such users effectively behave as free riders. In our previous work studying free riding behavior [11], we proposed the concept of *conscious free-riders*, which let other peers know that they cannot contribute to chunk distribution. Compared to oblivious free riders that do not provide such information, conscious free riding improves distribution efficiency by avoiding wasteful chunk requests to free riders. In this work, we assume free riders are conscious.

One issue related to current P2P systems that has not been addressed yet is that peers share their out-partner slots with both cooperative peers and free riders. When a peer p shares out-partner slots in $\mathcal{N}_o(p)$ between cooperative and free rider peers, the P2P overlay requires a mechanism to identify and push free riders to the overlay's edge. Moreover, both free-riding and cooperative peers compete for the same overlay resources. This is a particularly serious problem during flash crowds, when a large number of free-riding and cooperative peers try to join the system simultaneously.

To take advantage of conscious free-riding, our idea is to classify peers and provide different out-partner sets to peers of different classes. We split the out-partner set $\mathcal{N}_o$ into $\mathcal{N}_{o,c}$ and $\mathcal{N}_{o,f}$, for out-partnerships to cooperative and free-riding peers, respectively. Thus, a peer p accepts new cooperative peers in $\mathcal{N}_{o,c}(p)$ and new free-riding peers in $\mathcal{N}_{o,f}(p)$.

Given the $\mathcal{N}_o$ split into $\mathcal{N}_{o,c}$ and $\mathcal{N}_{o,f}$, the bootstrap server can allow free riders to join the overlay immediately even during during flash crowds without incurring the risk of compromising the overlay stability and efficiency. This is possible because the overlay is able to throttle the free rider join rate itself as a function of the $\mathcal{N}_{o,f}$ partnership slots surplus offered by (cooperative) peers already in the overlay. When cooperative peers try to join the overlay, $\mathcal{N}_{o,c}$ slots will still be available. As more cooperative peers join the overlay, they contribute more $\mathcal{N}_{o,c}$ and $\mathcal{N}_{o,f}$ slots to the overlay.

We need to split $\mathcal{N}_o$ into $\mathcal{N}_{o,c}$ and $\mathcal{N}_{o,f}$ in a way that optimizes overlay efficiency. Based on previous work showing that overlays with high-capacity peers closer to the server leads to higher distribution efficiency [5,6,9], we set $N_{o,c}$ proportional to a peer's ability to contribute to chunk distribution, and $N_{o,f}$ inversely proportional to a peer's ability to contribute. For free-riding peers, $N_{o,c} = N_{o,f} = 0$ by definition. We envision the four classes shown in Table I.

Table I
Peer Classes and Output Partnership Configurations

Peer Classes	$N_{o,c}$	$N_{o,f}$	Description
Hot Class	> 0	$= 0$	High contribution peers
Standard Class	> 0	> 0	Average contribution peers
Cold Class	$= 0$	> 0	Low contribution peers
FR Class	$= 0$	$= 0$	Free-riding peers

The **Hot Class** contains peers having the highest potential for contributing to the network since they have the highest bandwidth. Peers from the hot class only connect to peers from the hot and standard classes.

The **Standard Class** contains peers with average potential for contributing to the network. Peers from the standard class may establish partnerships with peers from any class.

The **Cold Class** contains peers with low potential for contributing to the overlay. Peers from the cold class connect to peers in the cold and standard classes as well as free riders. Finally, **FR Class** holds all uncooperative peers and free riders.

Surely, such set of classes may be progressively expanded (by adding classes) in order to handle specific scenarios demanded by applications. In a more generic sense, the system allows a peer p to accept a new out-partner request even when $\mathcal{N}_o(p)$ is full, replacing existing out-partner in $\mathcal{N}_o$ by the new out-partner if it has larger N_o.

In PPC, peers compete for collaborative out-partnership slots ($\mathcal{N}_{o,c}$) only with other peers that also have collaborative out-partnership slots. These peers apply the same disconnection strategy in Section II, preferring peers with higher $N_{o,c}$. Similarly, peers without collaborative out-partnership slots (i.e., with $N_{o,c} = 0$) only compete for free-riding out-partnership slots with other peers without collaborative out-partnership slots (e.g., cold and free-riding peers). Again, peers apply the same disconnection strategy, preferring peers with higher $N_{o,f}$.

IV. Methodology

We run experiments on PlanetLab [12] using TVPP. We use as many PlanetLab nodes as possible in our experiments (around 110, with slight variations between experiments). Peers in the overlay are subject to CPU and bandwidth restrictions in the underlying PlanetLab node.

In order to limit the upload bandwidth to more realistic scenarios, we also apply a bandwidth limit, defined for each experiment. The bootstrap server runs on our university's network and is not subject to bandwidth limitations (and has spare CPU capacity). The video server streams a 420kbps video. Each chunk is a MTU-sized packet, which gives around 40 chunks per second.

The experiment is composed of five runs, and we show averaged results over all runs. Each run lasts for 900s. During the first 400s, we run a single client on each PlanetLab node (i.e., around 110 peers). On second 400, we launch ten additional peers on each PlanetLab node to emulate a flash crowd event. We propose a static scenario in order to evaluate the performance of PPC. By static we mean that: (i) the class of a given peer is known in advance,[1] and (ii) peers never change classes during the experiment. As we fix the maximum number of in-parters and out-partners (N_i and N_o) that peers in each class have, we run experiments varying N_i and N_o to cover a range of overlay characteristics.

[1]In real deployments, we may rely on the upload bandwidth declared by the user for defining the class of each peer, or estimate its contributions at run time [9,13].

Table II
Default Experiment Configuration ($N_i = 20, \rho \approx 1$)

Peer Class	Upload (Mbps)	Peers (%)	Classic N_o	PPC-u $N_{o,c}$	PPC-u $N_{o,f}$	PPC $N_{o,c}$	PPC $N_{o,f}$
FR	0.0	50	0	0	0	0	0
Cold	1.5	24	38	18	20	0	38
Standard	2.5	17	40	20	20	18	22
Hot	4.0	09	46	26	20	46	0

In particular, we focus on varying the value of $\rho = T_{in}/T_{out} = \sum_p N_i(p)/\sum_p N_o(p)$. Small values of ρ (more out-partners than in-partners) capture aggressive overlays where partnerships are aplenty but peers may lack bandwidth to answer all chunk requests; large values of ρ (less out-partners than in-partners) capture conservative overlays where partnerships are more restricted but peers are likely to have bandwidth to answer chunk requests; and values of $\rho \approx 1$ capture balanced overlays.

V. Evaluation

Table II shows the default configuration for our experiments. We consider four classes of peers with different upload bandwidths. We set 50% of the peers as free riders to stress the overlay, and distribute the remaining peers in the other three classes. Results with the same upload bandwidth and lower fractions of free riders yield qualitatively *better* results, as the overlay has more bandwidth per peer. (not shown). We compare PPC with two overlay construction strategies:

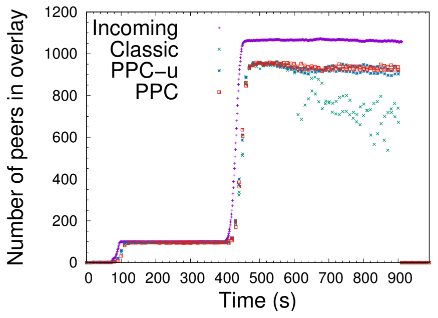
- **Classic:** Usual traditional strategy equivalent where peers have a single set of out-partners ($\mathcal{N}_o$).
- **PPC-uniform:** Naïve PPC strategy where all classes can establish partnerships with all classes (including free riders). PPC-uniform (PPC-u) can be seen as the classic strategy improved by splitting N_o into N_o, c and N_o, f. PPC-u differs improves on the classic strategy by avoiding the competition between cooperative and free-riding peers for partnerships.
- **PPC:** Compared to PPC-u, PPC enforces a more robust overlay topology organization that prevents free riders close to the media server.

Based on previous results on overlay configuration [14,15], we set $N_i = 20$ for all peers. We then compute N_o to achieve the desired ρ. In Table II, we show N_o, $N_{o,c}$, and $N_{o,f}$ values to have $\rho \approx 1$.[2]

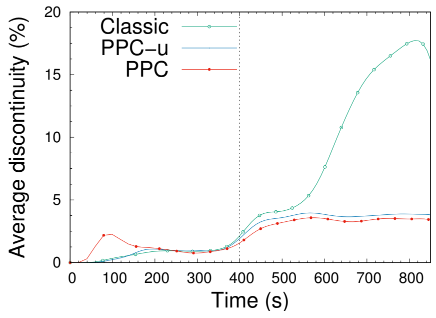
A. Balanced Overlays ($\rho \approx 1.0$)

In this experiment we consider a balanced overlay where the total number of in-partnerships is the same as the total number of out-partnerships, as shown in Table II. Figure 1 shows results averaged over the five runs. Figure 1(a) shows the absolute number of peers in the overlay and actively watching the stream. Peers that fail to receive enough video chunks

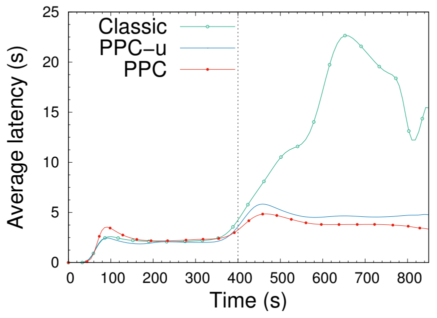
[2]Given that peers are free to enter and leave the P2P network, we consider the ratios to be approximated.



(a) Overlay Size

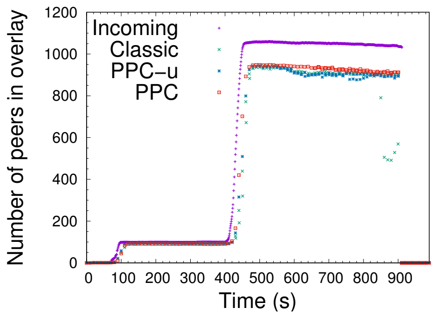


(b) Playback Quality

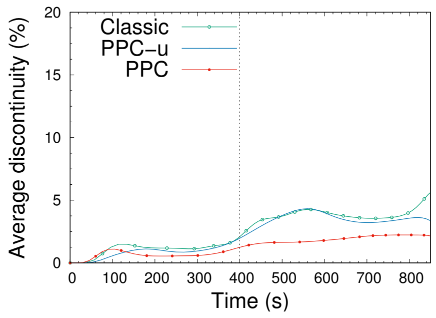


(c) Distribution Latency

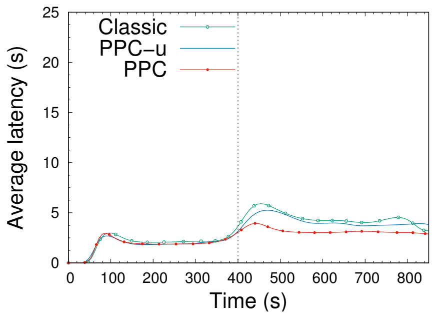
Figure 1. Overlay Size, Continuity, and Latency for Balanced Overlays ($\rho = T_i/T_o \approx 1.0$)



(a) Overlay Size



(b) Playback Quality



(c) Distribution Latency

Figure 2. Overlay Size, Continuity, and Latency for Conservative Overlays ($\rho = T_i/T_o \approx 1.5$)

to reproduce the media are not counted in the overlay. The Classic construction strategy joins approximately 800 peers in the overlay with severe fluctuation on the number of joined peers. Both PCC-u and PCC support a steady 950 peers in the overlay, with negligible fluctuation.

Figure 1(b) shows the average discontinuity over all peers. We define average discontinuity for each peer as the fraction of video chunks that miss their playback deadline. Only peers that are in the overlay, i.e., included in Figure 1(a), are considered. We observe that PPC-u and PPC provide a discontinuity below 4% throughout the experiment, [3], while Classic presents degraded performance after the flash crowd.

Figure 1(c) shows the average chunk distribution latency for all peers in the overlay. The latency of a chunk for a peer p is defined as the period of time between the generation of the chunk at the video server S and the chunk playback at peer p. Again, only peers that are in the overlay, i.e., included in Figure 1(a), are considered. Again, we observe PPC-u and PPC provide stable and reasonably low distribution latency, while Classic fails to distribute chunks in a timely fashion.

These results show that, while PPC-u and PPC achieve efficient and stable media distribution, the Classic overlay construction strategy leads to a disrupted overlay that fails at effective media distribution (fewer peers can reproduce the media, and the ones that can do so with degraded QoE). The main reason for this is the competition between cooperative and uncooperative peers for out-partnerships.

B. *Conservative Overlays ($\rho \approx 1.5$)*

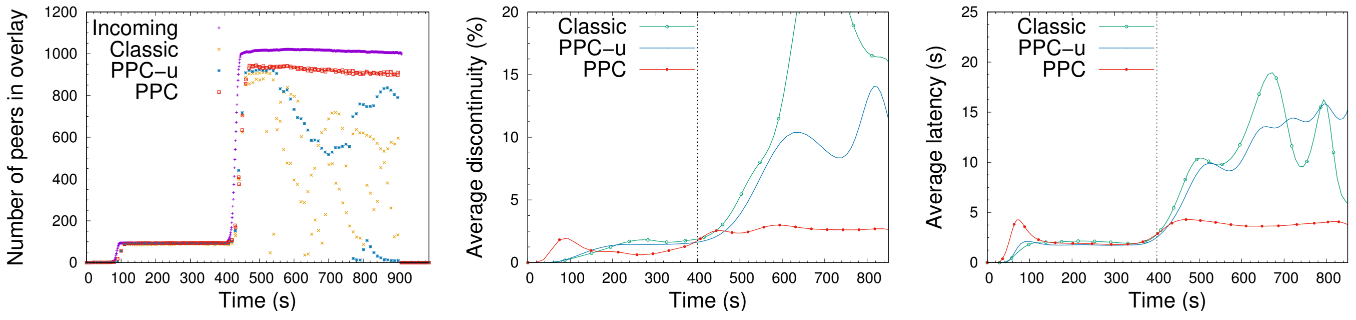
In this experiment we consider a conservative overlay where we increase the total number of in-partnerships by 50%, setting $N_i = 30$. We keep all other parameters unchanged (including the number of out-partners), as shown in Table II.

Figure 2(a) shows that all strategies reach approximately 950 joined peers. We also notice that the Classic strategy presents fluctuations after 800s. Since partners prioritize peers with large N_o, the peers with largest potential for contributing to the network tend to establish in-partnerships, which incurs in more available resources in the overlay and higher stability for all strategies (including the Classic strategy).

Figure 2(b) shows the average discontinuity. We observe all trategies provide discontinuity below the acceptable threshold of 4%, but PPC presents the lowest discontinuity among all overlay construction strategies. Figure 2(c) shows the average latency, with similar results: all strategies provide acceptable latency with a slight edge for PPC.

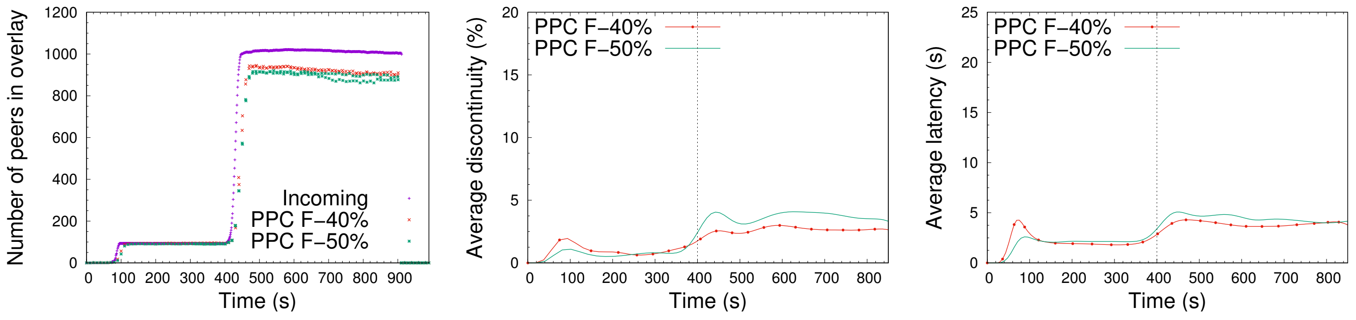
Overall, a conservative overlay ($\rho \approx 1.5$) where peers with higher contributions are accepted into the overlay and pulled close to the server yields a more robust and effective overlay. However, even in this scenario PPC presents a better performance than the Classic strategy.

[3]According to Traverso et al. [16], a discontinuity of 4% is acceptable.



(a) Overlay Size (b) Playback Quality (c) Distribution Latency

Figure 3. Overlay Size, Continuity, and Latency for Agressive Overlays ($\rho = T_i/T_o \approx 0.5$)



(a) Overlay Size (b) Playback Quality (c) Distribution Latency

Figure 4. Overlay Size, Continuity, and Latency for Agressive Overlays ($\rho = T_i/T_o \approx 0.5$)

C. *Aggressive Overlay ($\rho \approx 0.5$)*

In this experiment we consider an aggressive overlay where we double the total number of out-partnerships in the overlay while keeping $N_i = 20$. As overlay robustness and efficiency degrades in this configuration, we first show results for systems with 40% of free riders. Table III summarizes the configuration.

Table III
AGGRESSIVE OVERLAY EXPERIMENTS ($N_i = 20, \rho = 0.5$)

Peer	Upload	Peers	CONSTRUCTION STRATEGY				
			Classic	PPC-u		PPC	
Class	(Mbps)	(%)	N_o	$N_{o,c}$	$N_{o,f}$	$N_{o,c}$	$N_{o,f}$
FR	0.0	**40**	0	0	0	0	0
Cold	1.5	27	76	36	40	0	76
Standard	2.5	22	80	40	40	36	44
Hot	4.0	11	92	52	40	92	0

Figure 3(a) shows that both Classic and PPC-u build overlays that support less peers and are significantly more unstable. PPC builds a stable overlay that supports the same amount of peers as in the other configurations. As expected, Figs. 3(b) and 3(c) show that only PPC is able to keep low discontinuity and latency, while PPC-u and Classic result in very low QoE. Such result shows that PPC is more robust than Classic and PPC-u for a wider range of overlay characteristics. In order to complement the evaluation of PPC, Figure 4 shows results for PCC and agressive overlays with 40% and 50% of free riders (F-40% and F-50%, respectively). We observe that, even with 50% of free riders, PPC still manages to support around 940 peers and maintain reasonable QoE, strengthening our point that PPC builds more robust and effective overlays compared to PPC-u and Classic.

VI. RELATED WORK

Peers can have their neighbors classified as in-partners and out-partners. The maximum number of in-partners is referred to as *in-degree*, and the maximum number of out-partners is referred to as *out-degree*. Without limiting the out-degree, the amount of out-partners may extrapolate a peer's bandwidth. To couple with this, previous work proposed priority schemes for selecting the out-partners that should receive the next chunk [17,18]. According to [6], large neighborhoods incur many issues, such as an increased overhead for control messages that wastes bandwidth, and increases the complexity of request scheduling.

Traverso et al. [14] provide a comprehensive comparison benchmarking different overlay construction and maintenance strategies in P2P-TV systems. Similar to our work, their results show that topological properties of the overlay have a deep impact on the QoE and network load.

Ullah et al. [19] propose an autonomous management framework that (i) enables peers to learn user behavior and organize themselves to improve streaming quality, and (ii) controls the topology of push-based systems through a stabilization process that continuously pushes unstable peers towards the leaves of the tree.

The organization of peers also plays a fundamental role during flash crowds. Most of the strategies for handling flash crowds are based on throttling new peers joining rate to allow the overlay to adapt and scale. Rückert et al. [20] propose an hybrid streaming approach to prepare the system overlay (going beyond just throttling the joining rate). Fangming et al. [15] propose the use of Content Delivery Networks to absorb the load of flash crowds while the overlay adapts. Complementary to this approach, we propose a mechanism that classifies peers and organizes the overlay to increase its robustness and ability to receive new peers. Differently from these works, we allow continuous joining of peers in the network and we turn our attention for managing the share of cooperative peers and free-riders connecting each peer.

Additionally, many overlay construction strategies propose pushing free riders to the edge of the overlay to improve distribution efficiency. Piatek et al. [21] identify uncooperative peers by auditing cryptographic receipts of chunk transfers, while Guerraoui et al. [13] identify uncooperative and malicious peers based on their partnerships.

Finally, Payberah et al. [5] address the problem of free-riding through parent peers auditing the behavior of their children peers by querying their grandchildren. They conclude that the proposed strategy has better performance in different scenarios compared to a multiple-tree implementation of the system.

VII. Conclusion

In this work we investigated the usefulness of splitting cooperative peers and free riders into distinct classes of peers, and differentiating partnerships to each class. We have shown that this approach improves overlay robustness and efficiency by avoiding competition for resources between cooperative peers and free riders.

We presented the *Peer Partnership Constraints (PPC)* overlay construction strategy, which groups peers with similar contributions into classes and constrains which classes can establish partnerships with each other class.

We evaluated PPC in real experiments in PlanetLab, with a significant fraction of free riders, limited upload bandwidth, and different overlay characteristics. Our results show that PPC builds robust and effective overlays that support more peers and achieve higher quality of experience compared to traditional overlay construction strategies.

As future work, we plan to study an algorithm to quantify the contributions of a peer to the overlay and set its class dynamically.

Acknowledgment

Work partially funded by FAPEMIG, CAPES, and CNPq.

References

[1] "PPLive." http://www.pplive.com, 2008.

[2] "UUSee Inc.." http://www.uusee.com/, 2008.

[3] Q. Zheng, Y. Long, T. Qin, and L. Yang, "Lifetime Characteristics Measurement of a P2P Streaming System: Focusing on Snapshots of the Overlay," in *Intelligent Control and Automation (WCICA), 2011 9th World Congress on*, pp. 805–810, June 2011.

[4] M. Meulpolder, L. Meester, and D. Epema, "The Problem of Upload Competition in Peer-to-Peer Systems With Incentive Mechanisms," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 7, pp. 899–917, 2012.

[5] A. H. Payberah, J. Dowling, and S. Haridi, "GLive: The Gradient Overlay as a Market Maker for Mesh-Based P2P Live Streaming," in *Parallel and Distributed Computing (ISPDC), 2011 10th International Symposium on*, pp. 153–162, July 2011.

[6] R. J. Lobb, A. P. Couto da Silva, E. Leonardi, M. Mellia, and M. Meo, "Adaptive Overlay Topology for Mesh-based P2P-TV Systems," in *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, NOSSDAV '09, (New York, NY, USA), pp. 31–36, ACM, 2009.

[7] H. Wu, J. Liu, H. Jiang, Y. Sun, J. Li, and Z. Li, "Bandwidth-aware peer selection for P2P live streaming systems under flash crowds," in *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)*, pp. 360–367, Dec 2012.

[8] A. P. C. d. Silva, E. Leonardi, M. Mellia, and M. Meo, "A Bandwidth-Aware Scheduling Strategy for P2P-TV Systems," in *2008 Eighth International Conference on Peer-to-Peer Computing*, pp. 279–288, Sept 2008.

[9] M. Piatek, A. Krishnamurthy, A. Venkataramani, R. Yang, D. Zhang, and A. Jaffe, "Contracts: Practical Contribution Incentives for P2P Live Streaming," in *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI'10, (Berkeley, CA, USA), pp. 6–6, USENIX Association, 2010.

[10] J. Oliveira, R. Viana, A. B. Vieira, M. Rocha, and S. Campos, "TVPP: A Research Oriented P2P Live Streaming System," in *SBRC 2013 - Salão de Ferramentas*, 2013.

[11] J. F. e Oliveira, Í. Cunha, E. C. Miguel, M. V. Rocha, A. B. Vieira, and S. V. Campos, "Can Peer-to-Peer Live Streaming Systems Coexist With Free Riders?," in *IEEE P2P 2013 Proceedings*, pp. 1–5, IEEE, 2013.

[12] "PlanetLab: An Open Platform for Developing, Deploying, and Accessing Planetary-Scale Services." http://www.planet-lab.org/, 2009.

[13] R. Guerraoui, K. Huguenin, A.-M. Kermarrec, M. Monod, and S. Prusty, "LiFTinG: Lightweight Freerider-Tracking in Gossip," in *ACM/IFIP/USENIX International Conference on Middleware*, 2010.

[14] S. Traverso, L. Abeni, R. Birke, C. Kiraly, E. Leonardi, R. Lo Cigno, and M. Mellia, "Neighborhood Filtering Strategies for Overlay Construction in P2P-TV Systems: Design and Experimental Comparison," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 3, pp. 741–754, 2015.

[15] F. Liu, B. Li, L. Zhong, B. Li, H. Jin, and X. Liao, "Flash Crowd in P2P Live Streaming Systems: Fundamental Characteristics and Design Implications," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, pp. 1227–1239, July 2012.

[16] S. Traverso, L. Abeni, R. Birke, C. Kiraly, E. Leonardi, R. L. Cigno, and M. Mellia, "Experimental comparison of neighborhood filtering strategies in unstructured P2P-TV systems," in *2012 IEEE 12th International Conference on Peer-to-Peer Computing (P2P)*, pp. 13–24, Sept 2012.

[17] Y. Sakata, K. Takayama, R. Endo, and H. Shigeno, "A Chunk Scheduling Based on Chunk Diffusion Ratio on P2P Live Streaming," in *2012 15th International Conference on Network-Based Information Systems*, pp. 74–81, Sept 2012.

[18] Y. Guo, C. Liang, and Y. Liu, *AQCS: Adaptive Queue-Based Chunk Scheduling for P2P Live Streaming*, pp. 433–444. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[19] I. Ullah, G. Doyen, and D. Gaïti, "Towards User-Aware Peer-to-Peer Live Video Streaming Systems," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, pp. 920–926, IEEE, 2013.

[20] J. Rückert, B. Richerzhagen, E. Lidanski, R. Steinmetz, and D. Hausheer, "TOPT: Supporting Flash Frowd Events in Hybrid Overlay-Based Live Streaming," in *2015 IFIP Networking Conference (IFIP Networking)*, pp. 1–9, May 2015.

[21] M. Piatek, A. Krishnamurthy, A. Venkataramani, R. Yang, D. Zhang, and A. Jaffe, "Contracts: Practical Contribution Incentives For P2P Live Streaming," in *USENIX NSDI*, 2010.