Small Coalitions: Lightweight Collaboration for Efficient P2P Downloads

Ghassan Karame
ETH Zürich
8092 Zürich, Switzerland
karameg@inf.ethz.ch

Mario Čagalj
University of Split
21000 Split, Croatia
mario.cagalj@fesb.hr

Srdjan Čapkun
ETH Zürich
8092 Zürich, Switzerland
capkuns@inf.ethz.ch

***Abstract*—Peer-to-peer (P2P) architectures are gaining increasing popularity in disseminating content to a large number of nodes. In this paper, we show that small coalitions between peers can further enhance the performance of current P2P architectures. Small coalitions bridge the gap between inefficient non-cooperative and fully cooperative architectures by establishing a robust tradeoff between the complexity and performance of the resource distribution process. Owing to their small size, small coalitions are inherently resilient to the churn in existing P2P systems and embed natural incentives for peers to self-organize in order to improve their download times. We evaluate several coalition strategies analytically and empirically via simulations and we show that our solutions considerably improve the download performance in current P2P systems.**

***Keywords*-P2P; Content Distribution; Coalitions.**

I. INTRODUCTION

Peer-to-Peer (P2P) architectures (e.g., BitTorrent [1]) constitute robust solutions for “flash-crowd" scenarios, in which a *large* group of receivers wish to retrieve popular content offered by *few* peers. Previous work includes two major paradigms for P2P content distribution: mesh-based [2] and tree-based topologies [3], [4]. Although both architectures are considered to be efficient in disseminating content to a large number of peers, current researches suggest that there might be considerable room for improvement in these architectures. For instance, since nodes often make local decisions in mesh-based architectures, then the same content might be downloaded concurrently from the same offerer or might travel over competing paths which results in the under-utilization of the nodes’ resources [5], [6]. On the other hand, tree-based topologies typically face significant reliability and performance limitations when subject to selfish peer behavior [7] and high network “churn" [8].

Previous work suggested the formation of cooperative clusters among nodes to improve various aspects of network performance. In this work, we re-use this concept in the context of P2P content distribution and we propose *small coalitions* between peers as an efficient and robust solution to enhance the performance of current P2P architectures. Small coalitions are inspired by an analysis of two extreme strategies: non-cooperative and fully cooperative architectures. Non-cooperation between peers results in a suboptimal equilibrium where peers compete over shared resources leading to increased distribution times. On the other hand, full cooperation among all peers in the network suffers from performance limitations when subject to free-riders and high churn. Small coalitions bridge the gap between these two extremes by providing improved download performance, enhanced robustness and reliability while requiring minimal overhead. Small coalitions are formed between peers sharing common interests (e.g., peers downloading the same content) to efficiently leverage on their bandwidths as a mean of improving their download performance. In each coalition, near-optimal solutions are used for efficient resource distribution. Small coalitions are inherently resilient to churn and free-riders, owing to their *small* size, and can be easily embedded in current architectures.

In this paper, we propose cooperation strategies based on small peer-coalitions and we analyze their benefits in the distribution of *multi-chunk* and *single-chunk* resources in flash-crowd scenarios. More specifically, we consider coalition strategies where coalition members abstain from competing on the same file-fragment and coordinate their respective downloads and we show that these strategies improve the performance of existing architectures (e.g., BitTorrent). We further discuss mechanisms that enable efficient coalition formation and ensure fair behavior of coalition members. We validate our findings analytically and empirically via simulations.

The remainder of this paper is organized as follows. In Section II, we outline the main intuition behind small coalitions. In Section III, we present two small coalitions strategies for multi-chunk resource distribution. In Section IV, we highlight the small coalition strategy in the dissemination of single-chunk resources. In Section V, we empirically evaluate the performance of our proposals via simulations. Section VI overviews related work and we conclude the paper in Section VII.

II. THE INTUITION BEHIND SMALL COALITIONS

The main intuition behind our scheme is that the aggregation of various *local* optimizations constitutes an *efficient* alternative to the near-optimal, but costly global solution.

Small coalitions imply the formation of clusters of size k within which resource distribution is near-optimal. The key benefits of using small collaborative clusters in P2P content distribution are threefold: *1)* by alleviating competition between peers over shared resources, small coalitions result in improved download performance for *both* collaborating and non-collaborating members, *2)* the moderate coalition size ensures that near-optimal resource distribution can be used within each coalition with marginal overhead and *3)* due to their small size, small coalitions are inherently resilient to churn and free-riders. In fact, small coalitions rely on the efficiency of near-optimal content distribution when applied to a small cluster scenario. The high maintenance cost ($O(N)$) of fully cooperative approaches makes near-optimal solutions inadequate for large cooperative structures. However, these solutions are suitable for small coalitions due to their small size; by allowing nodes to self-organize in small clusters, small coalitions of size k induce a fixed and modest overhead of $O(k)$ in structure maintenance subject to peers leaving/joining; the departure of a single node can only affect its $k-1$ coalition members.

Based on this fundamental concept, we discuss in this paper cooperation strategies between subsets of peers where: 1) peers coordinate their file-fragment downloads and 2) peers abstain from competing over the same resource and we show that the integration of these strategies in current P2P architectures considerably enhances the overall download performance in the network.

Resource Sharing Model in Small Coalitions: Small coalitions target “flash-crowds” where popular content, hosted by only few peers, is subject to heavy download requests. We consider a realistic scenario where the peers' bandwidths are *limited* resources [7]. We assume that files are split into $F \geq 1$ fragments[1] (*chunks*). Peers that offer the resource are called *seeds*, while *leechers* refer to peers seeking to download the resource. To simplify our analysis, we abstract away the effects of network delays and we assume that the download rates do not restrain the upload throughput[2].

Our intra-coalition resource distribution scheme unfolds as follows. Coalition members adopt the Fastest-Node First (FNF) heuristic [9] to optimize bandwidth sharing. Khuller *et al.* [10] proved that any optimal resource distribution solution in heterogeneous settings is NP-hard and that the FNF heuristic is a near-optimal solution that minimizes the average completion time, and produces a 1.5 approximation for minimizing the maximum completion time. In our scheme, the coalition member with the highest upload bandwidth is selected to download the resource from the next fastest available offerer. To agree on the distribution solution, coalition members broadcast their upload rate and their ready time in the coalition. This overcomes the various shortcomings caused by heavy churn since the solution is constructed “on the fly".

III. SMALL COALITIONS IN MULTI-CHUNK DISTRIBUTION

In this section, we show how small coalitions enhance the performance of current distribution architectures, namely of BitTorrent [1]. We analyze two coalition strategies where: *1)* peers coordinate their chunk downloads and *2)* efficient bandwidth sharing is enforced between peers to alleviate competition over bandwidth.

Strategy 1 – Coordinating Chunk Selection: Current architectures make use of strict incentive mechanisms to encourage bandwidth sharing among peers. These incentives often limit the download rate of peers to their upload speed [11], [12]. Since most peers are connected using asymmetric links [7], their download bandwidths thus remain under-utilized in steady state conditions.

Although existing incentives and mechanisms are considered to be efficient given the absence of a global structure that orchestrates bandwidth sharing among all peers [2], we show that small coalitions can further enhance the service capacity of current architectures by allowing peers to leverage on their unused bandwidth capacity.

Consider a coalition strategy where the peers downloading the same file self-organize in coalitions of size k and coordinate their chunk selection strategy by separately downloading *different* chunks of the same file; the set $\mathbb{S}$ of F chunks constituting the resource is split into k mutually exclusive subsets $\mathbb{S}_0$,..,$\mathbb{S}_k$ of size $\frac{F}{k}$. That is, $\forall i, j \leq k$, then $\mathbb{S}_i \bigcap \mathbb{S}_j = \phi$. Coalition member p_i only downloads those chunks in its subset $\mathbb{S}_i$. To obtain the full file, coalition members exchange their chunks using near-optimal policies as described in Section II. This approach ensures a faster spread of all file-chunks in each coalition, and therefore in the network, thus alleviating the problem caused by selfish peers disconnecting early, while enhancing the download speed of peers.

In BitTorrent [1], the download speed that a peer p_i achieves is the sum the upload bandwidth it acquires from the seeds and its own upload bandwidth since the tit-for-tat policy ensures that its download rate is roughly equal to its upload bandwidth [11]: $R_{(1)_i} = \frac{S \cdot U_s}{C} + U_i$, where $R_{(1)_i}$ is the time average of download rate for p_i in a network with no coalitions, S is the number of seeds, U_s is the average upload rate of the seeds, C is the maximum number of connections to the seed and U_i is the upload rate of p_i.

In a network featuring coalitions of size k, p_i downloads its chunks in $\mathbb{S}_i$ using the underlying bandwidth sharing model (e.g., tit-for-tat [12], [13]) at a maximum rate $R_{(1)_i}$. However, p_i downloads the remaining chunks from its coalition members at a rate bounded by the minimum of its own download rate and the sum of the upload bandwidth of its coalition members since the FNF-based distribution of fragments within coalitions ensures that coalition members contribute all their upload bandwidth when exchanging

---

[1] The size of each fragment is typically 256 KB.

[2] Since most P2P users are connected through asymmetric links [7], the download rates are at least 3-8 times higher than the upload rates.

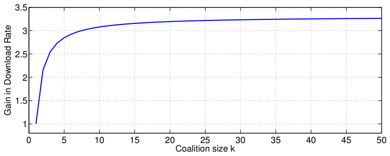


Figure 1. Gain in Download Rate w.r.t the coalition size k.

chunks. The download rate of p_i in a network featuring coalitions of size k is therefore:

$$R_{(k)_i} = \frac{1}{k} \cdot R_{(1)_i} + \frac{k-1}{k} \left(\min \left(R_i, \sum_{m=0}^{k-1} U_m \right) \right) \tag{1}$$

Here, $R_{(k)_i}$ denotes the time average of download rate[3] for p_i in a network featuring coalitions of size k, R_i is the maximum download speed of p_i and U_m is the upload rate of p_m. In Equation 1, we assume that each peer finds enough bartering partners to saturate its upload rate. In Section V, we show by simulations that this assumption roughly estimates content exchange in current P2P systems. The gain in download rate provided by coalitions of size k is then computed to be $G_{(k)_i} = \frac{R_{(k)_i}}{R_{(1)_i}}$. Figure 1 depicts $G_{(k)_i}$ with respect to the coalition size k, assuming the bandwidth distribution in current P2P systems [7]. Indeed, small coalitions considerably improve the download rate of coalition members (by a factor of up to 3 given the bandwidth distribution of peers in [7]). When coalitions contain enough members, the gain in download rates stabilizes since the upload rates of the coalition members will saturate the peers' download speed. The beauty behind our approach is that modest-sized coalitions are likely to perform favorably when compared to larger coalitions.

Strategy 2 – Alleviating Competition: Current architectures make use of "best-effort" local chunk selection strategies [2] – the random-first and the local rarest-first (LRF) policies – and do not handle peer-competition over bandwidth. Namely, they do not prevent concurrent downloads of the same chunk from the same offerer; all such competing streams will be allocated with a small portion of the offerer's bandwidth [5].

Small coalitions can efficiently address this problem: consider the strategy where coalitions of size k are *only* formed between peers downloading the same fragment from the same offerer. Only one peer in each coalition, the *coalition leader*, downloads the fragment and distributes it to the rest of the coalition using near-optimal policies (Section II). In the mean time, the remaining $k-1$ coalition members abstain from downloading from the offerer but can download different fragments from other offerers.

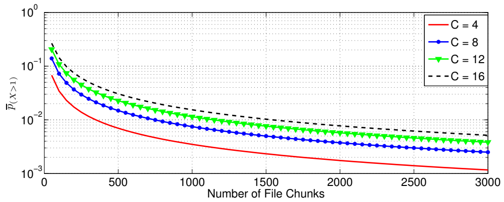


Figure 2. Probability of at least 2 concurrent downloads of a chunk from the same peer.

We now derive the probability of concurrent downloads of the same chunk in the network. Since the LRF policy ensures that all chunks are equally spread in the network, peer p_i can have up to i fragments, where $0 \leq i \leq F$. To simplify our analysis, we assume that i is a random variable and that p_i can have up to i fragments with probability $\frac{1}{F}$. We show later that this is a reasonable assumption. Let X denote the number of concurrent downloads of the same fragment. Assuming C connections per node, the probability of at least 2 concurrent downloads of the same fragment from a peer hosting i fragments ($C \geq i$) is:

$$p_{(X>1)_i} = 1 - p_{(X \leq 1)_i} = 1 - \frac{C \cdot (i-1)^{C-1}}{i^C} \tag{2}$$

In Equation 2, we assume that the file-fragments are downloaded at random (using the random-first policy). This is not always the case; the rarest-first strategy incurs a *bias* in the selection of fragments to ensure their fair replication which increases the probability of concurrent downloads of these fragments. Therefore, this analysis provides a lower bound on the number of concurrent downloads. On average, the probability of at least 2 concurrent downloads of the same fragment from a peer having i fragments ($0 \leq i \leq F$) is computed as follows:

$$\overline{p}_{(X>1)} = \sum_{i=1}^{F} \frac{p_{(X>1)_i}}{F} = \sum_{i=1}^{F} \left(\frac{1}{F} - \frac{C \cdot (i-1)^{C-1}}{F \cdot i^C} \right) \tag{3}$$

Figure 2 depicts $\overline{p}_{(X>1)}$ with respect to F and C. As expected, concurrent downloads are more likely to occur at early distribution stages when leechers have few fragments to exchange and/or when the file is comprised of a moderate number of fragments.

By alleviating competition over the same chunk, small coalitions can ensure faster spread of *each* chunk in the network. This solution further improves the distribution times of all other file-fragments; by alleviating competition among peers that are downloading the same fragment from the same offerer, *all* streams – including those pertaining to different resources – will be allocated with a higher

---

[3] For simplicity, we assume that F is a multiple of k and that the time required to download $\frac{F}{k}$ chunks is proportional to that needed to download F chunks.

portion of the offerer's bandwidth. For instance, consider coalitions of size 2: in the absence of coalitions, when C peers simultaneously download the same fragment from the same host, the average download time of the fragment D_1 is given by: $D_1 = \frac{C \times L}{U_s}$, where L is the fragment size and U_s is the upload speed of the host. When coalitions of size 2 are formed in the network, the average download time of C peers simultaneously downloading the same fragment is: $D_2 = \frac{C \times L}{2U_s} + \frac{L}{U_m}$, where U_m is the upload speed of the coalition leader. This strategy will always result in better download times provided that $U_m > \frac{2U_s}{C}$. In Section V, we analyze the gain provided by this strategy in realistic P2P settings for larger coalition sizes.

A. Coalition Formation

To enable coalition formation, one alternative is to adopt a *tracker-based* approach. Similar to BitTorrent [1], a central component, the *tracker*, keeps track of the chunks being downloaded by the peers in the system. When p_i downloads a file-chunk, it first contacts the tracker to obtain a list of peers downloading the chunk in question. p_i then contacts those peers for potential coalition formation. In the case where no other leecher is downloading the same content, p_i downloads directly from the offerer and registers its entry to the tracker. This solution might, however, add complexity to trackers. Alternatively, p_i can acquire the list of coalition members from the original offerer, p_j, itself since the latter already keeps information about the peers that it is uploading content to. To decide which coalition to join, an ideal choice for peer p_i requires an optimal assignment algorithm that maximizes its profit in the network; however, this would require the knowledge of the upload bandwidths of all network peers. In our scheme, p_i contacts the fastest coalition leader; if its bandwidth is small, p_i directly downloads from the original offerer. This would ensure a fair allocation of bandwidths among coalitions due to the randomness introduced by network churn.

Fair Intra-Coalition Collaboration: Since intra-coalition distribution depends to a large extent on the credibility of coalition members, selfish peers can abuse our scheme and deny service to other coalition members. Thus, additional countermeasures need to be used to ensure fair intra-coalition collaboration among peers.

In [14], Sirivianos *et al.* proposed an incentive mechanism, Dandelion, based on a cryptographic fair exchange mechanism that uses symmetric cryptography. In Dandelion, the offerer can act as a trusted third party mediating content exchange in each coalition. Consider the case where peers p_i and p_j form a coalition: when p_i uploads content to p_j, it sends it encrypted. The latter requests the decryption key from the offerer to acquire the content. This serves as a proof that p_i has indeed uploaded content to p_j. Another way of ensuring fair intra-coalition collaboration is to leverage on trust within coalition members. Trust can be managed through the use of reputation management systems (e.g., [15]). In this way, selfish peers will be associated with low reputation values and will therefore not be chosen in subsequent interactions.

IV. Small Coalitions in Single-Chunk Distribution

In this section, we analyze the benefits of small coalitions in the special case where the resource is only comprised of a *single* chunk (e.g., distribution of critical updates to a large number of clients [16]).

Consider the formation of small coalitions among peers that concurrently download the same resource from the same offerer. Within each coalition, only one peer, the *coalition leader*, downloads the resource from the original offerer. It then distributes the resource to other coalition members using the FNF scheme (Section II). Meanwhile, the other coalition members can increase their throughput by downloading other files offered by different peers.

Example – Coalitions of Size Two: Suppose that the resource is initially hosted by only one seed and that the network contains N leechers (N is a multiple of two). Let $D_{(k)}$ denote the average time over all peers to download the file in a network featuring coalitions of size k. $D_{(1)}$ denotes the average download time in a network featuring no coalitions. Assuming that the seed accepts up to C concurrent connections (C divides N), $D_{(1)}$ is given by:

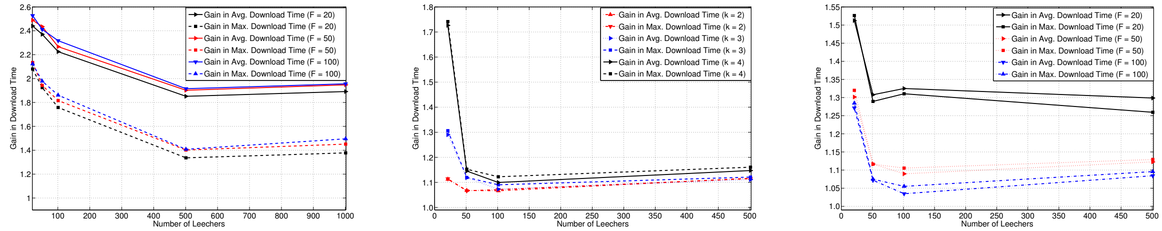
$$D_{(1)} = \frac{C}{N} \cdot \frac{C \cdot L}{U_s} \cdot \sum_{i=1}^{\frac{N}{C}} i = \frac{C \cdot L \cdot (1 + \frac{N}{C})}{2U_s}, \qquad (4)$$

where L is the content size and U_s is the seed's upload bandwidth. We, now, consider coalitions of size two; peers self-organize into $\frac{N}{2}$ clusters of size 2. *Only* the coalition leader downloads the file from the seed, after which it dedicates its upload bandwidth U_m to transfer content to its coalition member. $D_{(2)}$ is given by:

$$D_{(2)} = \frac{2C^2 \cdot L}{N \cdot U_s} \cdot \sum_{i=1}^{\frac{N}{2C}} i + \frac{N \cdot L}{2U_m} = \frac{C \cdot L + \frac{L \cdot N}{2}}{2U_s} + \frac{L}{2U_m}$$

We define $G_{D_{(2)}} = \frac{D_{(1)}}{D_{(2)}}$ to be the gain provided by small coalitions. For coalitions of size $k = 2$, the gain in average distribution times for all peers approaches 2 as N increases. Note that coalitions of bigger size are needed to provide considerable gains if the seed is a high-bandwidth server.

Natural Incentives: We now analyze the incentives for peers to join small coalitions. To simplify our analysis, we assume that peers have homogeneous bandwidths. In Section V, we validate that our analysis equally applies to heterogeneous peers.

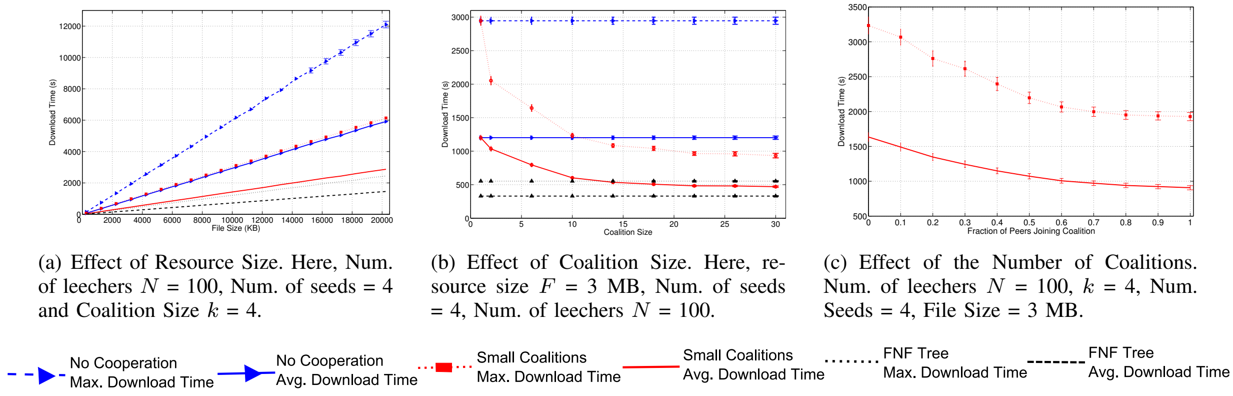


(a) Coalition members coordinate their chunk selection strategy. Here, $k = 3$.

(b) Coalitions are formed between peers downloading the same chunk. $F = 50$.

(c) Coalitions are formed between peers downloading the same chunk. $k = 3$.

Figure 3. Simulation Results of Small Coalitions in Multi-Chunk Resource Distribution.



(a) Effect of Resource Size. Here, Num. of leechers N = 100, Num. of seeds = 4 and Coalition Size k = 4.

(b) Effect of Coalition Size. Here, resource size F = 3 MB, Num. of seeds = 4, Num. of leechers N = 100.

(c) Effect of the Number of Coalitions. Num. of leechers N = 100, k = 4, Num. Seeds = 4, File Size = 3 MB.

Figure 4. Simulation Results of Small Coalitions in Single-Chunk Resource Distribution (we also show the 95 % confidence intervals).

In the absence of coalitions, the maximum download time of the leechers is $M_1^N = \frac{N \cdot L}{U_s}$, where C is the maximum number of concurrent connections to the seed and M_k^m is the maximum download time when the network contains m coalitions of size k. In case k peers form a coalition, the number of peers that acquire the resource from the seed decreases by $(k-1)$. Given this, even those peers that do not join coalitions improve their download times. All coalition members acquire the resource in $\frac{L}{U_s} log_2(k)$ time units[4] after the coalition leader finishes downloading. The resulting maximum download time M_k^m is then given by:

$$M_k^m = \frac{L \cdot (N - m \cdot k + m)}{U_s} + \frac{L}{U_s} log_2(k) \qquad (5)$$

From equation (5), it follows that $M_1^N > M_k^1 > M_k^m > M_k^{m+1}$. *We can conclude that our solution embeds natural incentives for peers to join coalitions and performs well in spite of selfish peers that do not join coalitions; even if only two peers form a coalition, the download times of all peers – including those that do not collaborate – still improve.*

V. PERFORMANCE EVALUATION

We implemented C-based simulators that compare the performance of resource distribution *with* and *without* small coalitions. We assume a realistic distribution of bandwidths in the network derived from the findings in [7]. We further assume that peers accept up to $C = 8$ concurrent connections per file. To simulate flash-crowd scenarios, leechers join our system randomly in time in an interval of 120 seconds. We adopted the churn model derived from [8]. When peers receive coalition formation requests, they check if their maximum coalition size is reached, otherwise they accept the request. This results in the formation of coalitions of equal size in the network. Every data point in our plots is averaged over 1000 runs.

Coalitions in Multi-Chunk Resource Distribution: By coordinating their chunk downloads, small coalitions provide *significant* gains in download times when compared to existing approaches (Figure 3(a)). The gain in average download times approaches 2 for coalitions of size $k = 3$. This gain stabilizes towards the theoretical limit set by Equation 1 when peers find enough bartering partners to saturate their upload rate.

Furthermore, by alleviating competition over the same fragment, even coalitions of size 2 achieve an average gain of 10% over scenarios where nodes don't form coalitions (Figure 3(b)). When the network is small, leechers are likely to share the same view about the network; the LRF policy results in a bias in the selection of fragments which leads to a large number of concurrent downloads for the same

---

[4] In homogeneous settings, and given $k = 2^a$ where $a \in \mathbb{N}^*$, an optimal intra-coalition resource distribution solution exists where each peer serves the resource upon its download completion [17].

fragment (Figure 3(b)). As k increases, the gain provided by small coalitions increases since the distribution solution becomes closer to the optimal one. We point that this gain is considerable (30%) for a small number of chunks per file F (Figure 3(c)); the smaller is F, the bigger is the probability of concurrent downloads of the same chunk from the same peer.

Coalitions in Single-Chunk Resource Distribution: We compared the small coalitions approach with both the non-cooperative (coalition size $k = 1$) and the fully cooperative approach. In the latter scenario, we assume that the seed computes a near-optimal *global* distribution solution. Our findings in Figures 4(a) and 4(b) show that irrespective of the resource size, small coalitions scale well as the network grows and provide a robust tradeoff between naive non-cooperative schemes and costly fully-cooperative resource distribution architectures. Even very small coalitions result in significant improvements in download times. Figure 4(c) confirms the analysis conducted in Section IV; even if a fraction (30%) of peers form coalitions, the download times significantly reduce for all leechers in the system.

VI. RELATED WORK

Biersack *et al.* [3] investigated a wide range of tree-based architectures for P2P systems. They showed that $PTree_k$ architectures (where peers are organized in a forest of k spanning trees) are very efficient in distributing k-fragments to a large number of nodes. Schiely *et al.* [4] proposed a tree distribution scheme based on the *heapsort* algorithm where "fast" peers are located closer to the root for faster resource dissemination. However, these architectures do not cope well with heavy churn and free-riders [3].

Several mesh-based architectures were proposed in the literature (e.g., BitTorrent [1], [18]). A major drawback of these architectures is that since nodes make local decisions, then it is possible for the same content to be downloaded from the same offerer [5] or to travel over competing paths resulting in the under-utilization of the resources [6]. In [11], Garbacki *et al.* propose a protocol that allows peers to form groups within which *dedicated* helpers download content on behalf of a peer. In our scheme, groups are formed between peers downloading the same content thus sharing natural incentives to enhance their download performance.

VII. CONCLUSION

In this paper, we analyzed several *small coalitions* strategies and we showed that they significantly enhance the performance of current multi-chunk and single-chunk content distribution architectures, while introducing marginal overhead. Small coalitions establish a robust and efficient tradeoff between non-cooperative and fully cooperative architectures and can be easily integrated in existing distribution architectures. In addition to their efficiency, small coalitions are resilient to churn and free-riders and therefore embed natural incentives for peers to help each other in the download of network resources.

REFERENCES

[1] BitTorrent, "http://www.bittorrent.org/protocol.html."

[2] A. L. Sophia, "Rarest first and choke algorithms are enough," 2006.

[3] E. Biersack, D. Carra, R. Cigno, P. Rodriguez, and P. Felber, "Overlay architectures for file distribution: Fundamental performance analysis for homogeneous and heterogeneous cases," in *Computer Networks*, 2007.

[4] M. S. Lars, L. Renfer, and P. Felber, "Self-organization in cooperative content distribution networks," in *Proceedings of DOA*, 2005.

[5] Y. N. Lien, "Performance issues of p2p file sharing over asymmetric and wireless networks," in *Proceedings of ICDCSW*, 2005.

[6] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proceedings of INFOCOM*, 2005.

[7] S. Saroiu, P. Gummadi, and S. Gribble, "A measurement study of p2p file sharing systems," in *Proceedings of MMCN*, 2002.

[8] D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks," in *Proceedings of IMC*, 2006.

[9] M. Banikazemi, K. Panda, and V. Moorthy, "Efficient collective communication on heterogeneous networks of workstations," in *Proceedings of ICPP*, 1998.

[10] S. Khuller and Y. A. Kim, "On broadcasting in heterogeneous networks," in *Proceedings of the ACM-SIAM symposium on Discrete algorithms*, 2004.

[11] P. Garbacki, A. Iosup, D. Epema, and M. Steen, "2fast: Collaborative downloads in p2p networks," in *Proceedings of P2P*, 2006.

[12] B. Cohen, "Incentives build robustness in bittorrent," 2003.

[13] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "Do incentives build robustness in bittorrent?" in *Proceedings of the USENIX Symposium on Networked Systems Design & Implementation*, 2007.

[14] M. Sirivianos, J. Park, X. Yang, and S. Jarecki, "Dandelion: Cooperative content distribution with robust incentives," in *Proceedings of the USENIX Technical Conference*, 2007.

[15] G. Karame, I. Christou, and T. Dimitriou, "A secure hybrid reputation management system for super-peer networks," in *Proceedings of the IEEE CCNC*, 2008.

[16] Symantec, "www.symantec.com/avcenter/defs.download.html."

[17] X. Yang and G. de Veciana, "Service capacity of peer to peer networks," in *Proceedings of INFOCOM*, 2004.

[18] A. L. Chow, L. Golubchik, and V. Misra, "Improving bittorrent: A simple approach," in *Proceedings of IPTPS*, 2008.