On Using a CDN's Infrastructure to Improve File Transfer Among Peers*

Minh Tran Wallapak Tavanapong

Department of Computer Science, Iowa State University, Ames, IA, 50011, USA
{ttminh,tavanapo}@cs.iastate.edu

Abstract. Content Distribution Network (CDN) technology has been proposed to deliver content from content nodes placed at strategic locations on the Internet. However, only companies or organizations, who can pay for the services of CDNs, have the privilege of using CDNs to distribute their content. Individual users (peers) have to resort to more economical peer-to-peer (P2P) technologies to distribute their content. Although P2P technologies have demonstrated tremendous successes, they have inherent problems such as the instability and the limited bandwidth of peers. In this paper, we propose a new approach to build bandwidth bounded data distribution trees inside a CDN so that external peers can leverage the power of a CDN's infrastructure for distributing their content. Our performance evaluation shows that, with some limited help from a CDN, the content distribution time among peers can be speeded up from 1.5 to 3 times.

1 Introduction

Content distribution networks (CDNs) have contributed significantly to transform the Internet into a successful content dissemination system. They have been proposed and deployed to primarily distribute content from companies and organizations (who are content producers/publishers such as CNN, Yahoo, etc.) to individual Internet users. A CDN operator deploys CDN nodes at *strategic and fixed* locations on the Internet to replicate data of content producers/publishers [1, 2] (publishing content on a CDN is only available to some companies/organizations). Thus, current CDNs do not address the need of individual users (also known as peers[1]) to publish/distribute their own content. On the other hand, peer-to-peer (P2P) networks have emerged as an alternative approach for sharing content among thousands of peers on the Internet. This approach employs peers' resources (network bandwidth, storage space, and available time, etc.) to disseminate shared files. A P2P network is *flexible* because anyone could participate in the network. However, the resources of a P2P network are only as good as the aggregated resources of the contributing peers. Building on the CDN and P2P content distribution models, we investigate an integrated content distribution framework named *Synergetic Content Distribution* that takes advantage of both models. We shift from the conventional wisdom by not considering CDN and P2P as two separate content delivery models. We instead see a potential for merging them so that peers help a CDN in

* This work is partially supported by National Science Foundation under Grant No. 0092914. Any opinions, findings, and conclusions or recommendation expressed in this paper are those of author(s) and do not necessarily reflect the views of the National Science Foundation.

[1] We hereafter use the term peers instead of individual users.

delivering the CDN's content (that of big content producers/publishers) and the CDN in return helps peers to distribute peers' own content.

Three main challenges for the realization of our Synergetic Content Distribution framework are (i) designing a mechanism to help a CDN to recruit peers to become part of the CDN. The recruited peers collaborate with the CDN to deliver the CDN's content; (ii) designing a mechanism to help a CDN to open up its network efficiently and securely[2] so that peers can take advantage of the CDN's infrastructure in distributing peers' content[3]; and (iii) designing an incentive mechanism to entice peers to become part of a CDN and also to entice a CDN to open its distribution network to benefit peers. Because each of these challenges deserves its own study and requires a different technical solution, our methodology is to address these challenges separately while still considering each of them as an integral part of our Synergetic Content Distribution framework. The eventual deployment of our framework requires the presence of satisfactory solutions to all three challenges.

In this paper, we focus on addressing the second challenge discussed in the preceding paragraph. The idea of opening up a CDN so that peers can take advantage of the CDN's infrastructure will bring the power of having efficient and high quality content distribution to everyone (both companies/organizations and individual users). To the best of our knowledge this idea has not been documented in the literature. Our contributions in this paper are (i) a formulation of the problem of peers using a CDN's infrastructure to distribute their content. Our proposed solution to this problem is to build bandwidth bounded trees inside the CDN to allow peers to send/receive content. The bandwidth bounded trees provide peers with reliable and higher bandwidth than normal end-to-end direct connections among peers. They also prevent the CDN from using too much of its bandwidth for peers' traffic. (ii) a CDN-assisted peers' content delivery protocol; and (iii) an evaluation of our proposed approach showing that with some limited help from a CDN the content distribution time among peers can be speeded up from 1.5 to 3 times.

The rest of this paper is organized as follows. In Section 2, we present an overview of our Synergetic Content Distribution framework to give an idea of our overall research effort in content distribution. We then focus on one specific research problem and provide a solution in Section 3. We present the performance study and simulation results Section 4. In Section 5, we discuss related work. Finally, we conclude the paper in Section 6.

2 Overview of our Synergetic Content Distribution Framework

In this section, we provide an overall picture of our current research directions in content distribution. We show how this paper relates to our other research directions. Our overall research goal in content distribution is to design and evaluate a Synergetic Content Distribution framework that takes advantage of both the CDN and the P2P distribution models. Our motivation for this new framework comes from our observation that peers can help a CDN to deliver the CDN's content while a CDN can help peers in return

[2] Not allowing peers to abuse or to pose a security concern for a CDN.

[3] We make a distinction between CDN's content belonging to the content producers/publishers and peers' content belonging to individual users

to distribute peers' content. Distributing CDN's content and distributing peers' content differs in that CDN's content is always replicated on CDN nodes while peers' content are never replicated on CDN nodes. To achieve this research goal, we pursue three different, but closely related, research directions to be described briefly in the following.

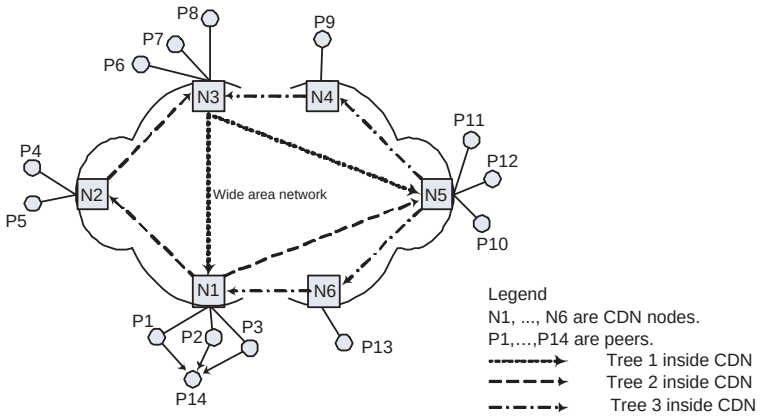


Fig. 1. Synergetic Content Distribution

The first research direction addresses the challenge of designing a mechanism to help a CDN to recruit peers to become a part of the CDN. Then, the CDN and the recruited peers collaborate to deliver the CDN's content. For example, in Figure 1 the CDN node N_1 recruits peers P_1, P_2, and P_3 so that they can collaborate to deliver the CDN's content to peer P_{14}. The benefit of this approach is to allow a CDN to be more dynamic by exploiting readily available resources of peers to deliver the CDN's content. Our approach improves the service latency of streaming content by 30% compared with an approach where a CDN does not exploit peers' resources to distribute CDN's content. We present this research direction in more details in [3].

The second research direction, **which is the main focus of this paper**, aims at designing a mechanism to allow a CDN to use its infrastructure to help peers in their content distribution[4]. An important issue is to limit the resources a CDN makes available to peers. Therefore, we propose that a CDN only contributes its resources to peers proportional to what it received from peers earlier when peers used their resources to deliver the CDN's content. For example, in Figure 1, the set of twelve peers $\{P_1, \ldots, P_{12}\}$ had contributed their resources to help the CDN earlier. Now, when they want to distribute content among themselves, the CDN creates three different bandwidth bounded trees rooted at CDN nodes N_1, N_3, and N_5, respectively. The peers transfer their content through these trees inside the CDN. We provide more details on this direction in Section 3.

The third research direction, which is under investigation at the time of this writing, focuses on an incentive mechanism to entice peers to contribute their resources to a CDN and also to entice a CDN to provide peers accesses to its infrastructure. Our incentive mechanism builds on the fairness and the reciprocation principles. The mechanism

[4] By peer content distribution we mean a peer (or some peers) wants to send an entire file (e.g., MP3 or movie file) to a group of other peers.

strictly follows a policy to require peers to contribute first to build up their credit before being able to use the CDN to distribute their content. Another important element for our incentive mechanism is the ability to prevent malicious behavior. That is, we do not want peers to collaborate to cheat a CDN nor do we want a CDN to refuse to provide accesses to its infrastructure to good peers who already contributed their resources to the CDN. We have briefly presented our overall research effort in content distribution. In the next section, we discuss the main research problem of this paper and we propose a solution.

3 CDN-assisted Peers' Content Delivery

3.1 System Model and Assumptions

We assume that the network topology of a CDN is known and stable. The CDN has network measurement features to maintain an accurate view of the bandwidth, delay, and other metrics of the links inside the CDN. Each CDN node is responsible for handling a network area consisting of a number of peers. For example, in Figure 1 CDN node N_1 is responsible for handling peers $\{P_1, P_2, P_3\}$. The bandwidth between any two CDN nodes is higher than the end-to-end bandwidth between two peers in the respective network areas that the two CDN nodes are responsible for. The bandwidth is not necessarily symmetric.

Peers run a P2P protocol that has the following key features. A centralized node provides a new peer with information about a subset of nearby peers and a nearby CDN node responsible for the network area. This can be achieved through a network positioning system such as [4]. A group of peers distribute content among themselves in sessions. At the beginning of a session only one peer (or only a few peers) is a seed (i.e., having the whole content other peers want). At the end of the session, all peers in the group have the whole content. Content is divided into data blocks of equal size. A peer uses parallel downloading to get different data blocks from different peers.

3.2 Delivering Peers' Content through a CDN

There are two scenarios for content delivery among peers. First, if the content delivery involves only peers in the same network area of one CDN node, the CDN node may not help much the peers in improving (i.e., provide faster delivery or more reliable network connections) the delivery of content. Therefore, a natural solution in this scenario is to let the peers to deliver content directly among themselves without the involvement of the CDN. The peers use their default P2P protocol to transfer content. For example, in Figure 1 if only peers $\{P_1, P_2, P_3\}$ want to distribute content among themselves, N_1 may not help much. Second, if the content delivery involves peers in many different network areas handled by many CDN nodes, the CDN nodes can help to improve the transfer rate of content. Without such a help from the CDN, the peers would have to establish several end-to-end connections among themselves to transfer data. The quality (bandwidth and reliability) of these end-to-end connections is not as good as that of the connections among CDN nodes.

A good solution for our problem should (i) enable a fast exchange of data blocks through the CDN among multiple peers and (ii) limit the resources of CDN used in helping distributing peers' content. There are several methods for building a communication medium to achieve many-to-many communications such as using a mesh, a

graph, or a tree. In our case, a mesh or a graph would not be the best choice because we do not need redundant links among CDN nodes to distribute peers' content. We choose tree for its simplicity. With regards to limiting the resources a CDN uses to help peers, one should limit the transfer rate (i.e., transfer bandwidth) because a CDN should not be overloaded with peers' content. Note that the primary goal of a CDN is to distribute CDN's content. The main idea of our solution is to build bandwidth bounded trees in the CDN as a common medium for peers to distribute their content. Each tree is rooted at one of the CDN nodes, where there are peers wishing to send data, and spans to the remaining CDN nodes, where there are peers wishing to receive data. A CDN commits to provide a better transfer rate than the maximum achievable transfer rate of most peers. This is done by first taking into account the outgoing bandwidth distribution of peers and the mean and mode of that bandwidth distribution. Then a CDN provides peers with a transfer rate in a range that is in between the mean and a factor improvement of the mode. We next present our formal problem formulation.

Problem Formulation *Given a graph $G = (V, E)$ representing a content distribution network, where V is the set of all CDN nodes and E is the set of logical links connecting the CDN nodes. Let P be the set of peers who want to distribute content. Let V' be the set of the CDN nodes responsible for the network areas of peers in P. Set P can be categorized into $|V'|$ subsets of peers, each subset S_i $(i = 1 \dots |V'|)$ is handled by a CDN node $v_i \in V'$. Let B_i be the median of the set of outgoing access bandwidth in a subset of peers S_i. Let M_i be the mode of the set of outgoing access bandwidth in a subset of peers S_i. Let C_i be the growth factor that a subset of peers S_i provides to the CDN.* **The goal is to** *construct $|V'|$ trees such that each tree satisfies the following conditions: (1) each tree covers all vertexes in V'; (2) each tree is rooted at a vertex $v_i \in V'$. Vertex v_i is a CDN node handling a subset of peers S_i; and (3) each tree rooted at v_i has a bottleneck bandwidth of at least B_i, and a maximum bandwidth of $C_i \times M_i$.*

We propose the following BUILDTREES algorithm to solve our formal problem. The algorithm consists of $|V'|$ steps. Each step produces a tree that is rooted at a node $v_i \in V'$, covers all the other nodes in V', has a minimum bandwidth of B_i, and that has a maximum bandwidth of $C_i \times M_i$. At each step we choose to build a tree in a way that leaves as much bandwidth as possible for the remaining steps. This is an insight we learned from a recent work in fast replication of content in CDN [5]. However, their algorithm cannot be directly applied to solve our problem because they consider the problem of replicating data from *a single source* to multiple CDN nodes. We consider the problem of using a CDN to help many peers (i.e., *many sources*) to distribute content. Their algorithm builds multiple trees from a single source to *all* nodes in a CDN and it builds trees with the highest possible throughput. Whereas, our algorithm builds multiple trees for *multiple sources* (one tree for one source) and each tree reaches only *some* nodes in a CDN. Our algorithm does not build trees with the highest possible throughput, it only finds trees with a throughput being in a predetermined range (i.e., between the mean and the mode of outgoing bandwidth of peers).

We use the BUILDTREES algorithm to create one tree for each vertex v_i in V'. We start by sorting the vertexes (line 4) and proceed by this order of vertexes in the for loop (line 5). This method gives the set of peers that contributed the most to the CDN the highest chance of having the best links in CDN first. For constructing the tree

Algorithm 1 Building bandwidth bounded trees in a CDN to distribute peers' content

1: **procedure** BUILDTREES($G(V, E), V', set\ of\ B_i,\ set\ of\ M_i,\ and\ set\ of\ C_i$)
2: $TreesList \leftarrow \emptyset$
3: Sort V' to rank vertex(es) responsible for seed(s) first, then the remaining vertexes in decreasing order of growth factor provided to the CDN.
4: **for** each vertex $v_i \in V'$ in the sorted order **do**
5: Temporarily remove all edges $(u, v) \in E$ whose current bandwidth $bw_{uv} < B_i$
6: $Tree_i \leftarrow \emptyset$
7: $NodesInTree_i \leftarrow v_i$
8: **while** $NodesInTree_i \neq V'$ **do**
9: **for** each vertex $u \in NodesInTree_i$ **do**
10: $Edge_u \leftarrow$ Find edge (u, j) with $bw_{uj} \geq (C_i \times M_i)$ and with $min\{bw_{uj} - (C_i \times M_i)\}$
11: $RemainedBW_u \leftarrow BW_u$ - {Bandwidth of $Edge_u$}
12: **end for**
13: $x \leftarrow$ Node j with $max\{RemainedBW_j\}$
14: $NodesInTree_i \leftarrow NodesInTree_i \cup$ {destination of $Edge_x$}
15: $Tree_i \leftarrow Tree_i \cup Edge_x$
16: **end while**
17: **for** each edge $(u, v) \in E\ and\ (u, v) \in Tree_i$ **do**
18: $bw_{uv} \leftarrow bw_{uv}$ - {bottleneck in $Tree_i$}
19: **end for**
20: $TreesList \leftarrow TreesList \cup Tree_i$
21: Restore temporarily remove edges for next vertex $v_{i'}$
22: **end for**
23: Return $TreesList$
24: **end procedure**

rooted at each vertex, all edges that do not satisfy the minimum bandwidth requirement of B_i are temporarily removed (line 5). Note that this removal is only temporary for building the current tree. After a tree is constructed, the removed edges are restored so that they can be reconsidered in the next tree construction (line 21). We then add the root to the tree (line 7) and continuously add the remaining vertexes of V' until all of them are included (lines 8-16). The for loop (lines 9-12) is used to find an edge $Edge_u$, which is incident on each existing vertex u in the tree, that has the smallest bandwidth but still higher than $C_i \times M_i$. This guarantees the tree would be able to provide a bandwidth growth rate of C_i for the subset of peers S_i who already contributed to the CDN. The remaining available bandwidth $RemainedBW_u$ of node u is then calculated (line 11). Note that BW_u is the actual bandwidth that node u has at the time. Whereas, the $RemainedBW_u$ would be the new bandwidth that node u would have if $Edge_u$ was to be added to the tree. At the end of the for loop we have a set $\{RemainedBW_j\}$ of remaining available bandwidth of the nodes currently in the tree. We pick the node j that has the most remaining available bandwidth and assign it to variable x (line 12). We then add the corresponding neighbor of that node (chosen earlier in line 10) to the list of nodes in the tree (line 14). We next add the new edge to the tree. Finally, we reduce the bandwidth of edges in E that are also in the new tree by the bottleneck bandwidth of the new tree (line 18) and add the new tree to the list of trees (line 20).

We construct one tree per a CDN node in V', instead of following existing multiple trees approach in the literature [6, 7, 5], because the nodes in our tree are only CDN nodes (not peers). Once the tree is constructed it is stable, therefore, we do not need redundant trees. Moreover, we do not aim to use a CDN to achieve the fastest possible transfer of peers' content, therefore, we do not need multiple trees to get the highest throughput. We only want trees inside a CDN to provide higher bandwidth than the bandwidth of the end-to-end direct connections among peers. This is why we limit the

bandwidth of a tree rooted at a CDN node to be within a specific range between the median (B_i) of the outgoing access bandwidth and at most C_i times of the mode (M_i) of the outgoing access bandwidth of the peers in the network area the CDN node (the root) is responsible for. Another reason that we construct one tree per CDN node in V' is because our incentive mechanism only allows a set of external peers to send data through a CDN for as long as they have enough credit. When the set of peers run out of credit, the corresponding tree rooted at the CDN responsible the set of peers will be deactivated. In other words, the tree is removed so that set of peers cannot use the CDN to send data anymore. Note that the peers can still receive data from other trees (which belong to other sets of peers who still have enough credit). This approach provides us more simplicity and more flexibility in enforcing our incentive mechanism compared to other more complicated tree building approaches.

3.3 CDN-assisted Peers' Content Delivery Protocol

We assume that a group of peers, who want to use a CDN to distribute their content, already contributed their resources to the CDN. Now, the CDN is going to help the peers in return by increasing the transfer bandwidth of the content transfer among peers. The peers follow their P2P protocol to find out which content they want to distribute and which peers they want to distribute the content to. The peers also know which CDN node is responsible for the their network area (all these CDN nodes constitute the set V'). This information is handed over to one of the CDN nodes in V' who will run the BUILDTREES algorithm to construct the trees. Once the trees construction is completed, each CDN node in V' receives its tree information. The content distribution starts. Instead of sending data blocks to and receiving data blocks from other peers directly, as in the default P2P delivery protocol, each peer now sends to and receives from the CDN node responsible for the peer's network area.

When a CDN node receives a data block from a peer, it forwards the data block along the branches of the tree rooted at itself to other CDN nodes in the set V'. These CDN nodes are responsible for the network areas of the peers who interested in receiving the data block in question. Note that a peer may not be interested in some data blocks because it got them directly (without the help of the CDN) from other peers in its local network area. When a CDN node receives a data block from another CDN node, it forwards the data blocks to the peers that are participating in the content distribution in its network area. In addition, if the CDN node is not a leaf of a tree, it also replicates the data blocks and forward them to its children in the tree. Note that during content distribution, each data block traverses only one tree. A CDN node only replicate and forward data blocks of peers to other nodes down a tree, it does not cache nor store the data blocks of peers for later usage[5]. When a set of peers in a network area completely finishes a content distribution session (received all data blocks), the tree rooted at the CDN node handling that area is still kept to distribute data to the remaining unfinished peers. A tree rooted at a CDN node responsible for one area is destroyed only when either the peers in the network area do not have credits to send data anymore or there no data sending out on the tree for an extended amount of time. The latter case is possi-

[5] This is a major constrast to using the CDN to distribute CDN's content.

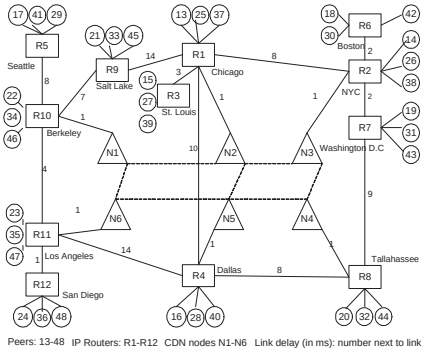


Fig. 2. Simulated network topology

ble when peers just leave after getting the content. Nevertheless, all trees are destroyed when all the peers finish.

4 Performance Study

In this section, we present the evaluation of our technique. We use a packet level simulator (ns-2 version 2.27) to compare our CDN-assisted peers' content delivery protocol to a default P2P delivery protocol. Some main features of the P2P protocol are (i) using parallel downloading of data blocks; (ii) allowing upload to at most five other peers at a time; and (iii) changing the corresponding peer often. The performance metric is the total time it takes so that all peers in the content distribution group finish receiving the content. We next discuss more details about our simulation setup.

4.1 Simulation Setup

Fig. 2 shows the network topology used in all the simulations. Due to the lack of actual topology information of a CDN, we adapt this topology from real cities in the United States with an assumption that a CDN operator would also want to deploy services in these geographical locations. The topology has 12 IP routers (R1-R12). The link propagation delays among these IP routers correspond relatively to their geographical distances. The bandwidth between a pair of IP router is symmetric (assuming a leased line connection) and is set at 1.5 Mbps. Each IP router connects to some peers via asymmetric connections. The maximum download bandwidth of a peer from an IP router is 700 Kbps while the maximum upload bandwidth of a peer to an IP router is only 200 Kbps (note that when there are a maximum of five concurrent receiving peers, each peer only gets 40 Kbps). This is a typical connection for DSL/Cable Internet users. Peers are assumed to have a 2-millisecond delay from the nearest IP router in our topology. We use 12, 24, and 36 peers in our simulations.

On top of this IP level topology, we build an overlay network of six CDN nodes (N1-N6) connecting to six IP routers as shown. Node N_1 is responsible for peers connecting to IP routers R_5,R_9, and R_{10}. Node N_2 is responsible for peers connecting to IP routers R_1 and R_3. Node N_3 is responsible for peers connecting to IP routers R_2,R_6, and R_7. Node N_4 is responsible for peers connecting to IP router R_8. Node N_5 is responsible for peers connecting to IP router R_4. Node N_6 is responsible for peers connecting to

IP routers R_{11} and R_{12}. Each CDN node has a 10 Mbps symmetric bandwidth to its corresponding IP router. The maximum bandwidth between a pair of CDN nodes is 10 Mbps. However, at anytime *the CDN only gives at most 1 Mbps for distributing peers' content*. The propagation delay among CDN nodes are 2 milliseconds. In our CDN-assisted technique, the peers use the CDN overlay to transfer only data blocks. Other types of packet (protocol signals, requests, etc.) have to go through the IP level network. The default P2P protocol only uses the IP level network for both data and other types of packets.

In our simulations, a group of peers distribute files of medium size (50 MBytes) and large size (500 MBytes). At the beginning of the distribution, there are only two seeds (two peers that have the complete content). This is why in our figures the lines start out flat. At the end, all peers have the complete content. We also consider two different scenarios for peers' arrival. The first one is the flash crowd situation in which all peers suddenly want to download the content at once (all peers arrive at the same time). The second one is a normal situation in which peers' arrival time follows a Poisson distribution.

4.2 Simulation Results

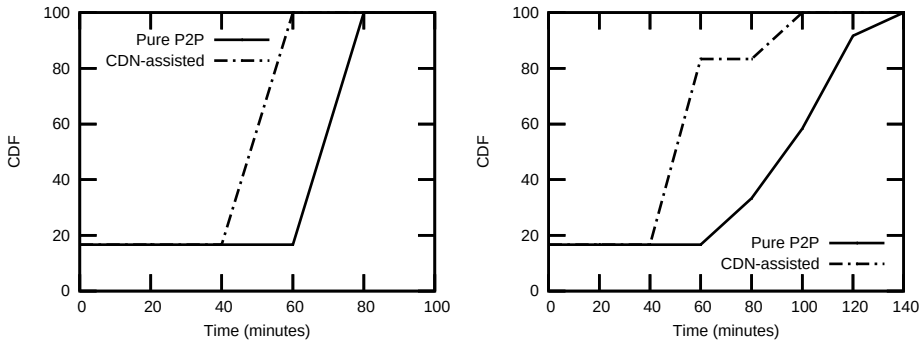
We run ten simulations, each with a different random seed to obtain an average value of the completion time of each peer. The standard errors of the completion times of peers are small: with an average of 2 minutes for flash crowd situation and 5 minutes for normal situation.

Fig. 3 and Fig. 4 show the CDF of distribution completion time when we use 12 peers (peers numbered 13-24 in the topology) and 24 peers (peers numbered 13-48 in the topology), respectively, to distribute small objects. Our technique (CDN-assisted) provides a speed up of 1.5 to 2 times in terms of distribution completion time in both flash crowd and normal scenarios. This result is expected because the CDN in our technique allows the peers to use its infrastructure to transfer data blocks. Note that the CDN only devotes at most 1 Mbps of its bandwidth to peers's traffic. We observe that during a flash crowd situation, the performance gap between our technique the default P2P protocol is reduced. This is because there are many peers available within a short period of time, increasing the service capacity of the P2P network.

Fig. 5 and Fig. 6 show the CDF of distribution completion time when we use 12 peers (peers 13-24) and 24 peers (peers 13-48), respectively, to distribute large objects of 500 MBytes each. Similarly, our technique offers a speed up of 1.5 to 3 times in terms of completion time in both flash crowd and normal scenarios. The result of simulations with 36 peers shows a similar conclusion; hence, it is omitted to save space.

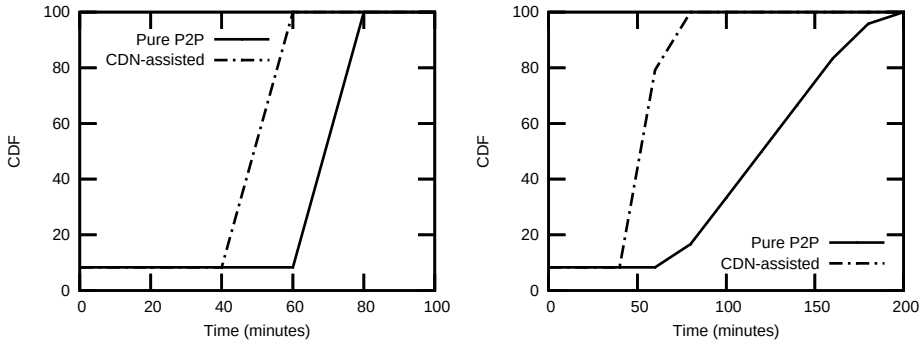
5 Related Work

There have been previous research on using pure CDNs for distributing web content [8, 1, 9, 10] and video content [11, 12], but they did not consider the problem of using a CDN to help peers in exchanging their content, which is the main focus of this paper. There have also been some recent proposals on building pure P2P networks for video streaming [13–15]. These systems differ from our approach mainly because they rely on purely peer nodes to build a distribution tree. Although this is a perfectly cost-effective approach, it differs from our approach in that our approach has the availability



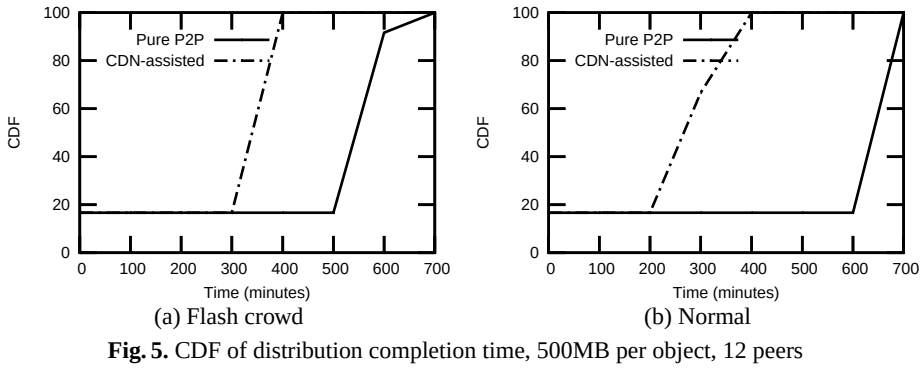
(a) Flash crowd (b) Normal

Fig. 3. CDF of distribution completion time, 50MB per object, 12 peers



(a) Flash crowd (b) Normal

Fig. 4. CDF of distribution completion time, 50MB per object, 24 peers

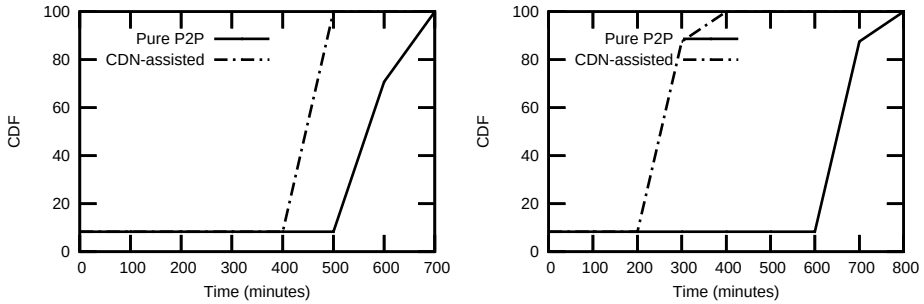


(a) Flash crowd (b) Normal

Fig. 5. CDF of distribution completion time, 500MB per object, 12 peers

and reliability provided by the CDN nodes. Our decision to use a CDN to help peers in content distribution is also strengthen by recent evidence that a purely P2P based streaming system needs some reliable nodes (some PlanetLab nodes in this case) to provide acceptable quality [16, 17].

There are some existing hybrid approaches of using both CDN and P2P networks for distributing content, such as PROP [18] and that of Xu *et al.* [19]. PROP [18] uses



(a) Flash crowd (b) Normal

Fig. 6. CDF of distribution completion time, 500MB per object, 24 peers

arbitrary local peer nodes to assist a local proxy server in an enterprise video streaming environment. PROP uses peers belonging to structured P2P networks [20] and it relies on the distributed hash table of structured P2P networks to perform content location. Xu *et al.* [19] proposed a hybrid CDN and P2P video delivery system in which a CDN hands over the serving of requests to a set of serving peer nodes at a calculated hand-off time. After this hand-off time, the system practically becomes a pure P2P content delivery system because the CDN does not serve requests anymore. However, these approaches differ from our approach in this paper in that they only look at peers as potential helpers for distributing a CDN's content, but they do not consider using a CDN to help peers distributing their content. A similar idea, to that of PROP [18] of using local proxies (web caches) in conjunction with peers, is implemented as a modification [21] to the original eMule file sharing network to improve download for peers. However, this approach considers local proxies as stand alone nodes and requires local proxies to cache peers' content. Our approach leverages the collaboration of CDN nodes and does not require CDN nodes to cache peers' content.

Comparing with recent research efforts in large file transfer like SplitStream [6], Bullet [7], Slurpie [22], ROMA [23], SPIDER [5], and using network coding with peers [24], our problem/solution is different because we build trees inside a CDN to help external peers to distribute their content. Our method of opening up a CDN to allow the transfer of peers content, as discussed in this paper, is a part of our larger Synergetic Content Distribution framework in which we maintain a two-way collaborative relationship between a CDN and the peers so that everybody (both companies/organization and individual users) can distribute content easily and efficiently.

6 Conclusion

In this paper, we have proposed a new approach of using a CDN to help peers in transferring their content. Our main new idea is to open up a CDN's infrastructure through which many individual users can deliver their content. This approach is in a direct contrast with the common wisdom of keeping CDNs closed to a small group of big content producers/publishers (e.g., CNN, Yahoo). We have proposed and evaluated an algorithm to build trees within a CDN to be used as communication medium for external peers to distribute content. Our initial results have shown the potential of this approach. Our future work includes the implementation and evaluation of our approach on PlanetLab.

References

1. Qiu, L., Padmanabhan, V., Voelker, G.: On the placement of web server replicas. In: Proc. of the 20th IEEE INFOCOM, Anchorage, AK (2001) 1587–1596
2. Jamin, S., Jin, C., Kurc, A., Raz, D., Shavitt, Y.: Constrained mirror placement on the internet. In: Proc. of the 20th IEEE INFOCOM, Anchorage, AK (2001) 31–40
3. Tran, M., Tavanapong, W.: Peers-assisted dynamic content distribution networks. In: Proc. of the 30th IEEE Local Computer Networks Conference (LCN), Sydney, Australia (2005)
4. Ng, T., Zhang, H.: A network positioning system for the internet. In: Proc. of the USENIX Annual Technical Conference, Boston, MA (2004) 141–154
5. Ganguly, S., Saxena, A., Bhatnagar, S., Banerjee, S., Izmailov, R.: Fast replication in content distribution overlays. In: Proc. of the 24rd IEEE INFOCOM, Miami, FL (2005) –
6. Castro, M., Druschel, P., Kermarrec, A.M., Nandi, A., Rowston, A., Singh, A.: Splitstream: high-bandwidth multicast in cooperative environments. In: Proc. of the 19th ACM Symposium on Operating Systems Principles (SOSP), Bolton Landing, NY (2003) 298–313
7. Kostic, D., Rodriguez, A., Albrecht, J., Vahdat, A.: Bullet: high bandwidth data dissemination using an overlay mesh. In: Proc. of the 19th ACM Symposium on Operating Systems Principles (SOSP), Bolton Landing, NY (2003) 282–297
8. Michel, S., Nguyen, K., Rosenstein, A., Zhang, L., Floyd, S., Jacobson, V.: Adaptive web caching: towards a new global caching architecture. Computer Networks and ISDN Systems **22** (1998) 2169–2177
9. Kangasharju, J., Ross, K., Roberts, J.: Performance evaluation of redirection schemes in content distribution networks. Elsevier Computer Communications Journal **24** (2001) 207–214
10. Wang, L., Pai, V., Peterson, L.: The effectiveness of request redirection on cdn robustness. In: Proc. of the 5th Symposium on Operating Systems Design and Implementation (OSDI), Boston, MA (2002) 345–360
11. Apostolopoulos, J., Wong, T., Wee, S., Tan, D.: On multiple description streaming with content delivery networks. In: Proc. of the IEEE INFOCOM 2002, New York, NY (2002) 1736–1745
12. Chawathe, Y.: Scattercast: an adaptable broadcast distribution framework. ACM/Springer Multimedia Systems Journal, Special Issue on Multimedia Distribution **9** (2003) 104–118
13. Padmanabhan, V., Wang, H., Chou, P.: Resilient peer-to-peer streaming. In: Proc. of 11th IEEE International Conference on Network Protocols (ICNP 2003), Atlanta, GA (2003) 16–27
14. Hefeeda, M., Habib, A., Botev, B., Xu, D., Bhargava, B.: Promise: Peer-to-peer media streaming using collectcast. In: Proc. of ACM Multimedia 2003, Berkeley, CA (2003) 45–54
15. Guo, Y., Suh, K., Kurose, J., Towsley, D.: A peer-to-peer on-demand streaming service and its performance evaluation. In: Proc. of IEEE International Conference on Multimedia and Expo (ICME 2003), Baltimore, MD (2003) 649 –652
16. Chu, Y., Ganjam, A., Ng, T., Rao, S., Sripanidkulchai, K., Zhan, J., Zhang, H.: Early experience with an internet broadcast system based on overlay multicast. In: Proc. of the USENIX Annual Technical Conference, Boston, MA (2004) 155–170
17. Sripanidkulchai, K., Ganjam, A., Maggs, B., Zhang, H.: The feasibility of supporting large-scale live streaming applications with dynamic application end–points. In: Proc. of the ACM SIGCOMM, Portland, OR (2004) 107–120
18. Guo, L., Chen, S., Ren, S., Chen, X., Jiang, S.: PROP: A scalable and reliable p2p assisted proxy streaming system. In: Proc. of the 24th International Conference on Distributed Computing Systems (ICDCS), Tokyo, Japan (2004) 778–786
19. Xu, D., Chai, H., Rosenberg, C., Kulkarni, S.: Analysis of a hybrid architecture for cost-effective streaming media distribution. In: Proc. of the SPIE/ACM Multimedia Computing and Networking conference (MMCN), Santa Clara, CA (2003) 87–101
20. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A scalable content-addressable network. In: Proc. of the ACM SIGCOMM, San Diego, CA (2001) 161–172
21. eMule Webcache Project: WebCache modifications for eMule. In: URL http://www.emule-mods.de/?mods=webcache. (Last accessed July 2005)
22. Sherwood, R., R.Braud, Bhattacharjee, B.: Slurpie: A cooperative bulk data transfer protocol. In: Proc. of the 23rd IEEE INFOCOM, Hongkong, China (2004) 941–951
23. Kwon, G., Byers, J.: Roma: reliable overlay multicast with loosely coupled tcp connections. In: Proc. of the 23rd IEEE INFOCOM, Hongkong, China (2004) 385–395
24. Gkantsidis, C., Rodriguez, P.: Network coding for large scale content distribution. In: Proc. of the 24rd IEEE INFOCOM, Miami, FL (2005) –