

High Performance Overlay File Distribution by Integrating Resource Discovery and Service Scheduling*

TEIN-YAW CHUNG⁺, YANG-HUI CHANG, KUN-HUNG CHEN AND YUNG-MU CHEN

Department of Computer Science and Engineering

Yuan Ze University

Chungli, 320 Taiwan

⁺*E-mail: csdchung@netlab.cse.yzu.edu.tw*

Many existing studies in overlay networking have focused on resource discovery or server selection. This work studies the integrated performance of peer-to-peer (P2P) file sharing systems using location-aware resource discovery and service scheduling schemes. A novel file capacity amplification (FCA) model is first presented to capture the problem of the file distribution problem. Then two novel service scheduling schemes and protocols, Capacity Amplification (CA) and its variant CA with Penetration (CAP), are presented to enhance the performance of file distribution in overlay networks. The CA scheme represents a greedy approach in selecting clients for services disregarding the effect of delay latency on transport capacity. The CAP scheme, on the other hand, adopts the semantics of small world networks to reduce effectively delay latency among peer servers and clients. Consequently, the effective transport capacity can be increased efficiently leading to fast file distribution. The analytical results indicate that traditional scheduling schemes such as FCFS perform poorly compared with CA and CAP schemes. Furthermore, a high-performance P2P file distribution system needs both an efficient resource recovery scheme and a good service scheduling scheme.

Keywords: peer-to-peer, P2P, file distribution, capacity amplification, penetration, small world network

1. INTRODUCTION

Peer-to-peer (P2P) networking [1, 2] is considered an effective approach to avoiding performance bottlenecks in client-server services. In traditional client-server architecture, files are mainly stored at the centralized server, and are downloaded when clients send requests to a server. In the P2P model, however, file distribution is much more rapid and efficient because every peer that has downloaded a file can also act as a server. Thus, P2P networking is a good candidate for file distribution on Internet.

P2P applications, such as file sharing, video streaming and hot news broadcasting [3-5] can benefit from efficient file distribution. In the SIU-based video streaming application like [6], a video file is fragmented into small chunks of files call SIUs. Efficient file distribution can help placing the SIUs to some potential nodes, which can relay the SIUs further to other nodes efficiently. For the hot news and stock market updates distribution, file distribution with the P2P pull model [7] allows content providers to publish content to subscribers rapidly. Thus, building an efficient file distribution system based on the P2P pull model is a very interesting issue.

Received July 4, 2007; revised December 3, 2007 & April 16, 2008; accepted June 5, 2008.

Communicated by Chung-Ta King.

* This work was supported in part by the National Science Council of Taiwan, R.O.C. under grant No. NSC 96-2221-E-155-033.

File distribution in P2P networks is a distributed application consisting of two elements, file search and service scheduling. In the past, these two elements were considered two unrelated entities in literature. Many studies [1, 4, 6, 8, 14] on P2P have mainly focused on scalable and efficient P2P infrastructures in term of object location and routing, they are not concerned about how a peer server responds to clients' requests. Some other studies addressed service scheduling by which a peer server decides which requesting peer client to serve concerning free riding, reputation [9-11] and the peer server's response time [12, 13]. These approaches, however, do not consider how a client discovers the resource in which it is interested. Thus, there exists a need to study the file distribution performance in a P2P network by tightly coupling both resource discovery and service scheduling scheme.

In this study, we first present a file capacity amplification (FCA) model to capture the scheduling problem of file distribution. By ignoring the effect of delay latency on transport capacity, FCA can be solved optimally by using a centralized scheme. However, when the delay latency is considered, FCA can be reduced to a game problem and cannot be solved optimally in real time. Then this work presents a decentralized scheme to solve FCA, which includes two components: server selection and client service scheduling. To solve FCA efficiently, the clients and servers must work jointly to increase the accumulative file distribution capacity fast. For a client, it must apply a service discovery scheme to obtain a list of candidate servers, from which they select a good server. For a server, it must schedules its service for client requests received to increase quickly file distribution capacity.

This study evaluates the performance of P2P file sharing systems which integrate the location-aware resource discovery and service scheduling as a single entity. Instead of studying a specific file search algorithm, this study identified two generic file searching approaches, unstructured and structured search. In the service scheduling, two novel service scheduling schemes and protocols are proposed in this study: Capacity Amplification (CA) and Capacity Amplification with Penetration (CAP). The CA scheme represents a greedy approach in selecting clients for services disregarding the effect of delay latency on transport capacity. Inspired by the nice feature of small world networks (SWNs) [15, 16] which possess small average distance among nodes, the CAP scheme gives priority to peer requests with large delay latency and penetration in the service scheduling. The delay latency of a request represents the distance that a link can reach while the penetration, as defined here, represents the cluster size. By considering the delay latency and penetration, CAP is able to capture similar effect of a small-world model which is featured with long haul links and large cluster coefficient.

To integrate the impact of various parameters on the performance of file distribution, CAP uses a novel function called potential to evaluate the contribution of peers in increasing overall effective file service capacity. In contrast to previous works that only consider bandwidth [17], delay latency [18] or living time [19], the potential function is defined as a normalized weighted summation of parameters including penetration, delay latency, peer living time and link bandwidth. Thus, through tuning of weights of parameters in the potential function, we are able to examine the effect of various parameters on the performance of P2P file sharing systems, and provide insight into the elements of an efficient P2P file sharing system.

A computer simulation was implemented to study the performance of P2P file shar-

ing systems based on the spreading speed of a hot file over a P2P network. In the simulation, existing service scheduling schemes, First-Come-First-Served (FCFS) and High-Credit-First (HCF) [9-11], are compared with our presented CA and CAP schemes under various service discovery approaches. Simulation results confirm that to achieve high-performance file distribution, it is important to integrate service discovery and service scheduling together. Moreover, the results demonstrate that CAP performs better than the other three schemes in most cases. An optimal scheme based on backward induction is also used as a benchmark of the performance of CA and CAP. The result shows that CAP performs close to that of the backward induction scheme when client arrival rate is small and performs reasonably well at high client arrival rate.

The rest of this paper is organized as follows. Section 2 defines the file distribution model in P2P file sharing systems. Section 3 defines schemes and parameters of file search and service scheduling in P2P file sharing systems. Section 4 analyzes the performance of P2P file sharing systems in various combinations of file searching and service scheduling schemes. Finally section 5 draws conclusions.

2. PROBLEM STATEMENT

First, the network model is described, the definition and objectives of peer-to-peer file distribution model are given.

2.1 Network Model

Let N be a set of vertices denoting the peer clients interested in file F . Let a file distribution network T at t_i be T_i with vertex set $N_{t_i} = \{v \mid v \in N\}$ where v has F , and link set $E_{t_i} = \{l_{uv} \mid u, v \in N\}$ where u downloads F from v .

Host is highly asymmetric, most up-link bandwidth (upstream) is smaller than the down-link bandwidth (downstream), such as ADSL. Also, the bottleneck is at or very close to the last-hop to the peers. In this study, we assume that the up-link bandwidth of a server peer (supplying peer) dominates a requesting peer's downloading bandwidth. For example, in a communication pair AB , peer A has 100Mbps downloading bandwidth and peer B has 10Mbps uplink bandwidth. The peer A only can download a file from the peer B in a maximum speed 10Mbps.

Moreover, when two peers are far away, the effective downloading bandwidth between them tends to become smaller due to congestion or delay in the transport network [19]. Thus, the downloading capacity C_{ij} from node i to node j is defined as $C_{ij} = \alpha_{ij}B_i$, where B_i is the up-link bandwidth of node i and α_{ij} is a normalized value of C_{ij} by link capacity B_i . According to the analytical analysis of TCP [20], C_{ij} is inversely proportional to the round trip delay RTT_{ij} between node i and node j as follows:

$$C_{ij} = \frac{\beta \times MSS}{RTT_{ij} \sqrt{\rho_{ij}}}. \quad (1)$$

Where β is a constant, MSS is the maximum segment size and ρ_{ij} is the packet loss rate due to congestion. Thus, α_{ij} can be expressed as

$$\alpha_{ij} = \frac{\beta \times MSS}{B_i \times RTT_{ij} \sqrt{\rho_{ij}}} = \frac{\beta}{\sqrt{\rho_{ij}}} \times \frac{MSS}{FSS_i}. \quad (2)$$

Where FSS_i is the maximum data size that can be transmitted by the up-link bandwidth of node i within RTT_{ij} . Thus, α_{ij} is inversely proportional to RTT_{ij} .

2.2 P2P File Capacity Amplification Problem

Let the server set at time t be N_t . If node i requests node j for file downloading at t_i , the service completion time of node i , t_i^c , is given by

$$t_i^c = t_i + TS_i \quad (3)$$

where TS_i is the downloading service time spent for node i to finish file downloading. After t_i^c , node i is put into $N_{t_i^c}$ and can offer downloading service to other client peers. The downloading service time TS_i includes the service waiting time and file downloading time. If node i downloads file F of size L_F from node j , the file downloading time includes the delay latency D_{ji} , which is equal to the round-trip delay [12, 13] between j and i , and the transmission time $L_F/\alpha_{ji}B_j$. Based on the above assumption, the file capacity amplification problem $T(N, F_s, S)$ is defined as follows.

Definition 1 Let F_s be the source for file F , and $S = \{t_1, t_2, \dots, t_n\}$ be the request timing sequence generated by the peer clients $v_1, v_2, \dots, v_n \in N$ for F . The file capacity amplification problem $T(N, F_s, S)$ is to select client peers awaiting for downloading service in an order to maximize the cumulative service capacity $C_{F_s}(t)$ for file F at any time t , where

$$C_{F_s}(t) = \int_0^t B_{F_s}(t) dt \quad (4)$$

and

$$B_{F_s}(t) = \sum_{j \in N_t} B_j. \quad (5)$$

By ignoring the possibility of peer node leaving and free-riding, $C_{F_s}(t)$ becomes a monotonic increasing function. In a centralized scheduling scheme, $T(N, F_s, S)$ can be solved optimally if the delay latency between peer nodes is ignored. If $T(N, F_s, S)$ is modeled as a single server queuing system with server capacity $B_{F_s}(t)$, which implies that a peer request can be served jointly by all peers with file F in an infinite granularity, *i.e.*, bit, $B_{F_s}(t)$ can be maximized $\forall t$ by serving awaiting peer clients in a descending order of their capacities. According to M/M/1, the average service time of peer requests gets smaller when the server capacity gets large. Thus, when $B_{F_s}(t)$ is maximized, the average service time of peer requests is minimized.

$T(N, F_s, S)$ can also be modeled as a multi-server queuing system, in which a peer request can only be served by a peer server, and solved optimally. When a peer request arrives, it is inserted to current peer request list in a descending order of capacities. Then, assign the peer and those peer clients behind it to one of candidate servers so that each

peer request has a minimum service completion time. Note that the candidate servers must include those peer servers available now and those peer clients currently waiting for service before it. With such scheduling, peers with large capacities can be made available in the earliest time for serving other clients and hence $C_{F_s}(t)$ is maximized.

When taking delay latency effect into consideration, the above solution for the single-server and multi-server queuing system model of $T(N, F_s, S)$ can be restated as to serve a waiting node j with maximal B in the earliest time, *i.e.*, maximizing $B_{F_s}(t)$ whenever possible. Such a greedy approach, however, becomes sub-optimal because the effective system capacity enjoyed by a peer client now depends on the distance between the peer client and those peer servers with file F . In other words, without knowing ahead of time which peers will issue request later in time, maximizing $B_{F_s}(t)$ may not always lead to optimal effective service capacity for peer requests to come.

To examine the optimality of $T(N, F_s, S)$, we can reduce $T(N, F_s, S)$ to a game problem with the following game rules:

1. This game is played on an $N \times N$ game board with two kinds of role, server and client.
2. A server can convert one client at a time to a server in a speed inverse proportional to the distance between the fields of the server and client.
3. Two players, one plays the client generator and another plays the server scheduler. The client generator assigns clients on the game board. The server scheduler assigns servers to convert clients into a server.
4. When a game begins, there is only one server.
5. The client generator puts a client at a randomly selected empty field in a constant frequency. The server scheduler assigns a server to an un-served client, if available, whenever a server becomes idle.

The server scheduler is given a period of time to convert as many clients as possible to a server. When the time is expired or all fields of the game board are full, the game is over and the more servers are on the game board the more credit the server scheduler earns. Therefore, the server scheduler must find out a best serving sequence and a server assignment strategy to minimizing client conversion time, or to maximizing the number of servers on the game board.

Obviously, the sequential interactions between the server scheduler and client generator in the $T(N, F_s, S)$ game can be expressed as a game tree and optimal solution can be achieved through backward induction. However, in the real world, $T(N, F_s, S)$ is much more complex than the game described above. For instance, servers may leave arbitrarily, clients may arrive randomly during the time, and the number of clients can be an arbitrarily large number. Therefore, it is impractical to apply backward induction to solve the $T(N, F_s, S)$ problem and a better approach is needed.

The server scheduler in the $T(N, F_s, S)$ game is a centralized scheme that schedules all peer services alone, which is not scalable and creates a single point of failure. Therefore, a decentralized scheme is commonly used in many peer-to-peer systems. In the following section, we focus on solving $T(N, F_s, S)$ in a decentralized manner based on the multi-server queuing model. Although different queuing model has different file downloading implication, this study intends to study effect of various characteristics on $T(N, F_s, S)$ solutions and thus, the result can be equally applied at that of a single server queuing system.

3. STRATEGIES FOR P2P CAPACITY AMPLIFICATION PROBLEM

When solving $T(N, F_s, S)$ in a decentralized manner, the service scheduling must include two components: server selection and service scheduling. Due to the lack of global peer clients and servers information in a decentralized scheme, peer clients must acquire servers with file F by resource searching, and then request a selected server for file downloading. On the other hand, peer servers must schedule their services for peer requests in a proper order to increase service capacity rapidly. This section addresses the issue how a peer client selects its peer server and how a peer server schedules its service for requesting peers. Section 3.1 describes two server selection schemes that can be adopted by a peer client. Section 3.2 describes two existing service scheduling schemes, FCFS and HCF, and proposes two novel approaches, Capacity Amplification (CA) and Capacity Amplification with Penetration (CAP), based on the proposed capacity amplification concept.

3.1 Server Selection Strategies

Before a client sends a request to a peer server for file downloading, it must first search for peer server candidates. Thus, the resource searching scheme used in an overlay network is a key element in File Capacity Amplification (FCA). The result of resource searching has significant impact on FCA as the server choices of peer clients are limited by the obtained server list. The less server information is available, the fewer choice a peer client has and hence the worse is the performance of FCA. The section discusses two generic resource searching models: unstructured and structured. These models are described as follows:

3.1.1 Unstructured peers

‘Unstructured’ peers are unorganized and loosely-coupled, as in Gnutella. Peers link to other peers arbitrarily depending on factors, such as number of hop counts, latency and bandwidth. In the unstructured model, when a peer initially logs onto the system, it floods a PING message to find other peers in the system. Peers who receive the PING message respond with a PONG message with their sharing information. Each peer maintains a list of peers and its corresponding round trip time (RTT) to them. This study assumes that the PING message would not flood peers more than 7 hops away. Thus, a published resource may not always be found if some peers are more than 7 hops away.

Peers search a file by sending a QUERY message to all peers in its list and receive a QUERY-HIT message if the query is successful. Once the requesting peer obtains a list of peer servers which possess the file, it chooses a peer server with low round-trip time delay and large bandwidth for file download.

3.1.2 Structured peers

In contrast to the unstructured model, peers in a structured model are organized for resource searching efficiency [21]. In [21] a cluster-based topology (CBT) algorithm was proposed to improve file searching efficiency using a well-organized network architec-

ture. In CBT, peers are grouped into different clusters according to their relative physical proximity in terms of hop counts. A new coming peer obtains a list of clusterheads from a bootstrap server. Once the information is retrieved, the peer chooses a cluster to join, or creates its own cluster, and then publishes resources to its associated cluster.

In the searching phase, the peer searches for its file of interest in the local cluster first. If the file is not found, the query message is redirected to other clusters for further searching. The peer servers in the local cluster are always chosen first. If all peer server candidates are in remote clusters, the peer server with a lower round-trip time and larger bandwidth is chosen.

3.2 Service Scheduling Strategies and Protocols

When a peer server receives many service requests, like a centralized scheme, the server must schedule its service for the requests. From a global FCA point of view, a server should attempt to amplify the total service capacity of file F as presented in section 2.2. For completeness, this section presents two existing service scheduling schemes FCFS and HCF and then describes our proposed Capacity Amplification (CA) and CAP (Capacity Amplification with Penetration) schemes below.

In FCFS, a service request is served according to its arrival sequence. This scheme is used in many P2P file sharing systems. In HCF, the service request with the largest credit is served first. The ranking of peers in the service queue of a supplying peer is determined by their reputation or credits. For convenience, this study chose the credit-only reputation computation (CORC) scheme [9] as the candidate strategy. The credit is calculated as follow:

$$CORC = QRC + UC + SC \quad (6)$$

where QRC is query-response credit, UC is upload credit and SC is share credit. Every peer that processes a query-response receives credit QRC per mega byte which is specified by 0.00006 [9] based on the rough estimation of average query-response message size in the Gnutella network. UC defines the credit a peer can get for serving content. UC is calculated as $(L_F/f) \times (B/b)$, where L_F is the size of the uploaded file; B is the bandwidth of the serving peer, and both f and b , respectively the file size factor and bandwidth factor, are tunable system parameters that are used to ensure that the reputation scores stay within a certain range of nonnegative values. Share credit is given to peers sharing files and staying online for a long time. The SC is calculated by $(S_j/f) \times L_T$, where S_j is the total size of shared files, and L_T is the duration of time that the peer stays online.

The Capacity Amplification (CA) scheme implements the greedy algorithm. A peer server schedules its service based solely on the uplink bandwidth of clients. It serves a request made by a peer with the largest uplink bandwidth first. Thus, with CA, all peer servers jointly attempt to increase file service capacity without considering the impact of delay latency on transport throughput.

The Capacity Amplification with Penetration (CAP) scheme is inspired partially by the nice feature of small world networks (SWNs). In SWN, a node not only connects to all nodes around its neighborhood but also connects to nodes far away with long haul links with probability p . With such a connection pattern, SWN can have small average

distance between nodes. Likewise, if a file can be downloaded to a remote client early, the file can also be accessed by many peer clients in a short distance soon. Thus, CAP mimics the effect of long haul links of SWN by selecting a peer with longer delay latency for service with higher priority.

To reduce further the effect of delay latency or to increase effective file service capacity, CAP considers a metric called penetration to enhance the effect of a long haul link. The penetration of a peer node is defined as the number of peers in its neighborhood. In this paper, the neighborhood is defined as the flooding area in an unstructured P2P environment and a P2P cluster in a structured P2P environment. Thus, in an unstructured P2P environment, the neighbors are the peers that are within N hops distance of the flooding peer while in a structured P2P environment the neighbors are the peers that are in the same cluster. By giving priority to the peer clients with potential to provide many peer clients with a large service capacity, the penetration metric may substantially increase effective throughput of a certain file.

Based on the above analysis, we define a novel metric called *potential* of a peer client as the service scheduling parameter for a peer server. The cost function *potential* P is defined as follows:

$$P = \alpha \times \left(\frac{D}{D_{\max}} \right) + \beta \times \left(\frac{L_T}{L_{T_{\max}}} \right) + \gamma \times \left(\frac{B^U}{B_{\max}^U} \right) + (1 - \alpha - \beta - \gamma) \times \left(\frac{Pi}{Pi_{\max}} \right) \quad (7)$$

Table 1. Notations definitions.

D	The round-trip time delay [20] between the requesting and supplying peers.
D_{\max}	The maximal round-trip time delay between a peer server and its peer clients.
L_T	The living time [10] of each peer in the system.
$L_{T_{\max}}$	The longest living time of all peers.
B^U	The upload capacity of the selecting peer.
B^D	The download capacity of the selecting peer.
B_{\max}^U	The maximum possible upload bandwidth of all peers.
Pi	The number of the requesting peer's neighbors within a cluster.
Pi_{\max}	The maximum number of neighbors in a cluster.
$\frac{Pi}{Pi_{\max}}$	The penetration index.

where $0 < \alpha, \beta, \gamma < 1$ and $\alpha + \beta + \gamma \leq 1$. The variables and some properties involved in the potential function are defined as shown in Table 1. Note that the potential P also includes the peer's living time. Thus, a stable peer client will be given a higher priority when a peer server selecting peer clients for services. This can keep N_i close to a monotonously increasing set, and hence improves the efficiency of file capacity amplification. Moreover, the CA is a special case of the CAP when $\alpha = 0$, $\beta = 0$ and $\gamma = 1$ in Eq. (7).

To implement these strategies, a new protocol must be associated with each of them, except for FCFS in which the arrival sequence of requests determines its scheduling. In HCF, a peer must include its credit in the service request, based on which peer servers

can schedule their service priority. In CA, a peer must indicate its up-link bandwidth in its request to help peer servers scheduling their services. In CAP, each peer must configure a set of common global parameters D_{\max} , $L_{T_{\max}}$, B_{\max}^U and $P_{i_{\max}}$. When a peer client makes a request, it must include a time stamp in the request and report its uplink bandwidth, living time and number of its neighbors. A peer server can compute delay latency based on the time stamp in the request and then use the information given by the peer client to calculate P and determines the service priority of its clients. At first sight CAP seems difficult to implement. However, the only thing special for CAP is that it requires the P2P network supports clustering and can report to a peer the number of active peers in its cluster.

4. SIMULATION RESULTS AND PERFORMANCE

To evaluate the performance of each strategy, a simulation environment for the FCA model was implemented based on parameters measured in existing operational P2P networks. The following sections present our simulation environment and results.

4.1 Simulation Environment

In simulation, the topology generator GT-ITM [22] is used to generate a topology which includes the location of nodes and the connectivity information, but not latency and bandwidth. Both the unstructured and structured models generate 10,000 peer nodes within a period of 72 hours. Each active peer at the request cycle has an opportunity to issue a request.

To reduce simulation complexity, we assume that requests are generated following Poisson distribution with average 10 in every request cycle [23]. Peers who send out request and cannot find the file do not resend the request until 30 minutes later [24]. If the request is blocked due to buffer overflow, the peer resends the request to other peer servers in its search list one by one until either the request is accepted or is rejected by more than ten peer servers. When a peer's request fails, the peer resends the request in 10 minutes later [24]. Although downloading different fractions of a file from many sources makes the distribution of the file faster, this simulation, for convenience in performance evaluation, assumes that the entire file is downloaded from a single source, meaning that a peer should choose one supplying peer if more than one source is available. Table 2 shows the detailed parameters configured in the simulation.

4.2 Simulation Results

First, the effect of hop counts was observed in both unstructured and structured environments. The hop count in the unstructured environment was defined as the maximum flooding radius of a file search, while in the structured P2P network, the hop count was defined as the radius of a cluster. In the simulation, the performance of file distribution was measured by the time taken to download an object to interested peers. The following sections study how the parameters of system and service scheduling strategy influence the file distribution performance.

Table 2. System parameters in the simulation.

Parameter	Description
Delay per physical link $D_{Link,i}$	$D_{Link,i} = W_i \times 1$ msec, W_i is uniformly distributed over $[0, 60]$; $D_{max} = 500$ ms.
File size L_F	650 MBytes
Bandwidth B [31]	70% with $B^U = 1$ Mbps, $B^D = 64$ Kbps; 23% with $B^U = 2$ Mbps, $B^D = 256$ Kbps; 7% with $B^U = 3$ Mbps, $B^D = 512$ Kbps.
Living time L_T [31]	Each peer has a certain probability of being online and is assigned an uptime and session duration based on the distribution measured in [31]: 60% of the peers maintain the session duration at most 60 minutes; 84% of the peers stay no longer for 10 minutes if they do not find desired files.
Service capacity $C_{ij} = \alpha B_i^U$	To simplify the simulation, in the experiments, α is simplified as follows: $\alpha = 1$ if $D_{ij} \leq 100$ msec; $\alpha = 0.7$ if 100 msec $< D_{ij} \leq 250$ msec; $\alpha = 0.5$ if 250 msec $< D_{ij}$.
Number of responses R [30]	A positive response to a query message means that the file transfer between two peers can be established. $R = 1$ for 87% peers; R is uniformly distributed over $[2, 15]$ for 7% peers; R is uniformly distributed over $[16, 500]$ for 6% peers.

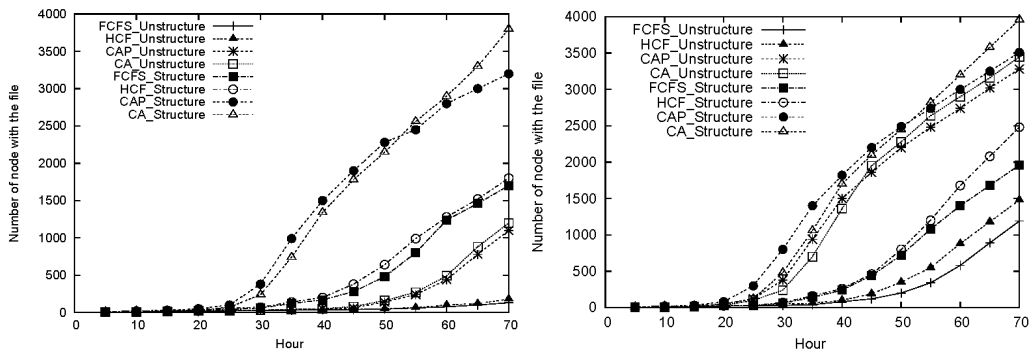
4.2.1 System performance

The results illustrated in Fig. 1 show that the time required to distribute a hot file is shorter in structured peers than that in unstructured peers in each respective scheduling strategy. The parameters for CAP are $\alpha = 0.2$, $\beta = 0.2$, and $\gamma = 0.5$ in Figs. 1 (a) and (b). In particular, when the radius is small, the difference in the distribution speed is large, mainly because a file search in a structured situation is always successful if the file does exist. By contrast, a file search in an unstructured environment fails if it cannot reach the nodes that own the file within the flooding radius. Consequently, a small flooding radius results in a high failure probability of a file search initially in the unstructured environment. Fig. 1 (b) also shows that the performance of CA and CAP in an unstructured network is close to that in a structured network at a radius of 7 hops, which has covered most peers in the simulated topology and thus the file search becomes successful most of time.

4.2.2 Service scheduling strategy parameters

After comparing different system parameters, the following simulation results focus on the relationship between α , β , and γ in the 5-hop structured peers. In all simulations, the upload bandwidth of the initial source was set to 256 kbps, and the queue size of each peer was 100.

Fig. 2, considering α , β , and γ shows that choosing peer clients with large bandwidth spreads a file to many nodes quickly. However, Fig. 3, which compares the aver-



(a) Cluster radius 3 hops. (b) Cluster radius 7 hops.
 Fig. 1. Performance of FCFS, HCF, CA, CAP in various cluster radius.

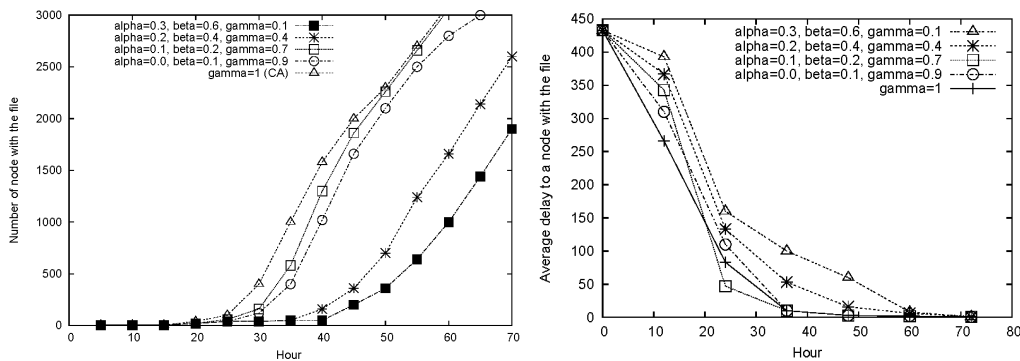


Fig. 2. Relationship between α , β , and γ without potential in CAP. Fig. 3. Relationship between α , β , and γ without potential in average delay in CAP.

age delay between nodes for file downloading, shows that the maximum γ value does not perform best in reducing the average delay, which represents the average distance of file downloading from a peer server to a peer client. After 12 hours, the combination of $\alpha = 0.1$, $\beta = 0.2$ and $\gamma = 0.7$ was found to reduce the average delay the most, generating less traffic load than $\gamma = 1$. Additionally, the file distribution with $\alpha = 0.1$, $\beta = 0.2$, $\gamma = 0.7$ was slightly better than that in $\alpha = 0$, $\beta = 0.1$, $\gamma = 0.9$. Thus the living time factor does help facilitate file distribution by serving stable peers first.

Fig. 4 shows the influence of the bandwidth and penetration factors. Bandwidth factor helps increase the distribution speed. Like the delay-time and living time factors, the penetration index can lower the average delay. When the coefficient of penetration index is 0.3, CAP can distribute a file quickly to many nodes with a smaller average delay and hence cause smaller traffic load to the network than CA. In Fig. 5, all factors are combined to observe their behaviors. Clearly, the ratio of $\alpha = 0.2$, $\beta = 0.2$, $\gamma = 0.5$ performs the best. CAP can not only cause smaller traffic load to network but also distribute a file at a greater speed than CA.

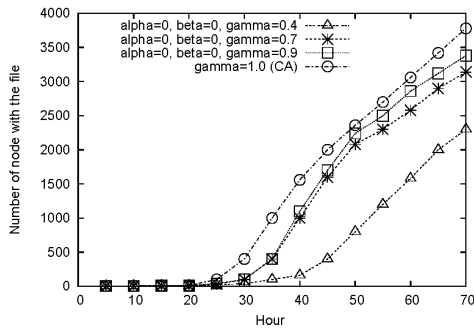


Fig. 4. Relationship between γ and $(1-\alpha-\beta, -\gamma)$ in CAP.

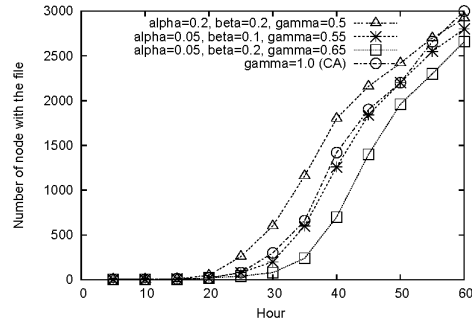


Fig. 5. Comparing different combinations of α , β , and γ and $(1-\alpha-\beta-\gamma)$ in CAP.

4.2.3 Benchmark among CA, CAP and backward induction

After finding out the best parameter setting of CAP, we use backward induction of a game tree for the $T(N, F_s, S)$ problem as a benchmark for the performance of CA and CAP. We first log the client arrival time and sequence and then generate a game tree. After that, each move is determined through backward induction. Therefore, the backward induction scheme represents an optimal result of the $T(N, F_s, S)$ problem.

In the simulation, the file size is set as 10MB and the arrival rates are $\lambda = 4$ peers/sec and 16 peers/sec following Poisson distribution. We log the peer arrival events in 500 seconds. The cluster radius in both experiments is set as 5-hop. The experiment result in Figs. 6 and 7 show CA always perform the worst among these three schemes. Fig. 6 also illustrates that when the client arrival rate is small, CAP performs close to optimal. When arrival rate increases to 16 peers/sec, as illustrated in Fig. 7, CAP performs worse than that of the backward induction scheme most of the time by around 10 to 30% before $t = 440$ seconds. This is because when arrival rate is large, many clients with large potential in a neighborhood may request servers far away simultaneously. Therefore, the advantage of serving clients with large potential early is subdued. With the backward induction technique, however, some of these clients are postponed to be served until one of the clients is converted to a server, which then provides high downloading speed for these clients. Therefore, the backward induction scheme performs superior to that of CAP.

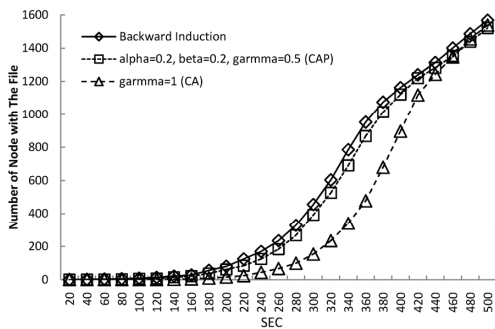


Fig. 6. Performance of CA, CAP and backward induction at $\lambda = 4$ peers/sec.

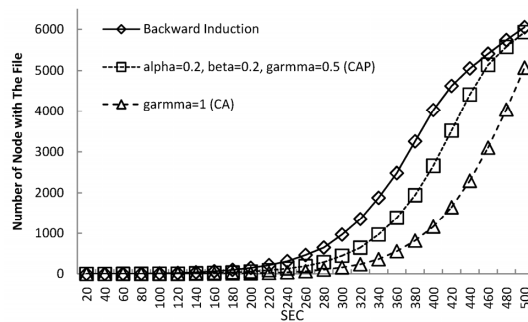


Fig. 7. Performance of CA, CAP and backward induction at $\lambda = 16$ peers/sec.

5. CONCLUSIONS

This study modeled the P2P file distribution as a file capacity amplification (FCA) problem. Based on the model for the FCA problem, two service scheduling strategies, the greedy algorithm capacity amplification (CA) scheme and a heuristic strategy called CA with penetration (CAP), were proposed to increase file distribution speed and reduce average delay. The study introduced resource discovery schemes for unstructured and structured peers, along with various service scheduling schemes. Results show that structured peers perform much better than the unstructured peers. Additionally, structured peers with small radii perform better in the long term than those with large radii. In the service scheduling schemes, various factors including bandwidth, delay, living time and penetration index are examined. The simulation results show that bandwidth has the greatest effect on the performance of file distribution, while other factors also help to decrease the average delay in file downloading. With appropriate weighting on various parameters, CAP outperforms CA in decreasing the average delay time, and hence can effectively decrease traffic load during the file distribution process. Our result also shows that CAP performs very close to optimal when peer client arrival rate is small and performs reasonably well in comparison to optimal result when arrival rate increases.

REFERENCES

1. H. Sunaga, *et al.*, "Technical trends in P2P-based communications," *IEICE Transactions on Communications*, Vol. E87-B, 2004, pp. 2831-2846.
2. A. Singh and M. Haahr, "A peer-to-peer reference architecture," in *Proceedings of the 1st International Conference on Communication System Software and Middleware*, 2006, pp. 1-10.
3. D. Hughes, *et al.*, "Peer-to-peer: Is deviant behavior the norm on P2P file-sharing networks?" *IEEE Distributed Systems Online 1541-4922*, Vol. 7, <http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8968>, 2006.
4. Z. Xiang, *et al.*, "Peer-to-peer based multimedia distribution service," *IEEE Transactions on Multimedia*, Vol. 6, 2004, pp. 343-355.
5. H. Tao, *et al.*, "SkipMard: A multi-attribute peer-to-peer resource discovery approach," in *Proceedings of the 2nd International Multi-Symposiums on Computer and Computational Sciences*, 2007, pp. 266-273.
6. T. Y. Chung, *et al.*, "SOP: SIU-based overlay proxy service for peer-to-peer streaming," in *Proceedings of International Computer Symposium*, <http://www.cse.yzu.edu.tw/ICS2006/docs/ICS2006.iso>, 2006.
7. L. Zhao, *et al.*, "Gridmedia: A practical peer-to-peer based live video streaming system," in *Proceedings of IEEE 7th Workshop on Multimedia Signal Processing*, 2005, pp. 1-4.
8. M. Srivatsa, *et al.*, "Large scaling unstructured peer-to-peer networks with heterogeneity-aware topology and routing," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 17, 2006, pp. 1277-1293.
9. M. Gupta, *et al.*, "A reputation system for peer-to-peer networks," in *Proceedings of ACM 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2003, pp. 144-152.

10. G. Zhitao, *et al.*, "An efficient hybrid P2P incentive scheme," in *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, Vol. 3, 2007, pp. 377-382.
11. R. Zhou and K. Hwang, "Gossip-based reputation aggregation for unstructured peer-to-peer networks," in *Proceedings of IEEE International Conference Parallel and Distributed Processing Symposium*, 2007, pp. 1-10.
12. H. Nakano, K. Harumoto, and S. Nishio, "Topology re-formation algorithms for ubiquitous P2P networks based on response statistics," in *Proceedings of International Symposium on Applications and the Internet Workshops*, 2007, pp. 55-58.
13. J. S. K. Chan, *et al.*, "Scheduling algorithms for peer-to-peer collaborative file distribution," in *Proceedings of International Conference on Networking, Applications and Worksharing*, 2005, pp. 10.
14. S. Weiqian and Q. Dongyu, "On the efficiency of peer-to-peer file sharing," in *Proceedings of IEEE International Conference on Multimedia and Expo*, 2007, pp. 32-35.
15. S. Manfredi, *et al.*, "Small world effects in networks: An engineering interpretation," in *Proceedings of IEEE International Symposium on Circuits and Systems*, Vol. 4, 2004, pp. IV-820-3.
16. R. A. Costa and J. Barros, "Network information flow in navigable small-world networks," in *Proceedings of the 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2006, pp. 1-6.
17. H. Yifeng, *et al.*, "Distributed rate allocation in P2P streaming," in *Proceedings of IEEE International Conference on Multimedia and Expo*, 2007, pp. 388-391.
18. R. Thierry, *et al.*, "A peer-to-peer scheme to discover and select QoS enhanced alternate paths," in *Proceedings of the 2nd International Conference on Communication System Software and Middleware*, 2007, pp. 1-9.
19. T. Yun, *et al.*, "Longer, better: On extending user online duration to improve quality of streaming service in P2P networks," in *Proceedings of IEEE International Conference on Multimedia and Expo*, 2007, pp. 2158-2161.
20. J. Martin, *et al.*, "Delay-based congestion avoidance for TCP," *IEEE/ACM Transactions on Networking*, Vol. 11, 2003, pp. 356-369.
21. T. Y. Chung, *et al.*, "Architecture and implementation of cluster-based peer-to-peer topology and its application in search," *Journal of Internet Technology*, Vol. 7, 2006, pp. 23-34.
22. GT-ITM: Georgia Tech Internetwork Topology Models, <http://www.cc.gatech.edu/projects/gtitm/>.
23. K. Kant, "An analytic model for peer to peer file sharing networks," in *Proceedings of IEEE International Conference on Communications*, Vol. 3, 2003, pp. 1801-1803.
24. D. Nogueira, *et al.*, "A methodology for workload characterization of file-sharing peer-to-peer networks," in *Proceedings of IEEE International Workshop on Workload Characterization*, 2002, pp. 118-126.



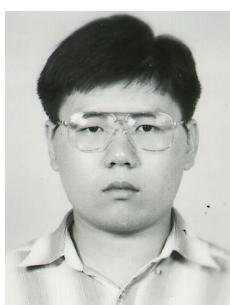
Tein-Yaw Chung (鍾添曜) received his M.S. and Ph.D. degree in the Department of ECE from North Carolina State University (NCSSU), Raleigh, NC, USA, at 1986 and 1990, respectively. After that, he joined IBM, RTP, NC and worked at network architecture development for two years. At 1993, he joined the Department of CSE, Yuan Ze University, Taiwan. Now, he services as an associate professor. His research interests included computer network, distributed system, multimedia communication, intelligent network systems and mobile networking. He is a member of IEEE Communication Society.



Yang-Hui Chang (張揚揮) received the M.S. degree in the Department of Computer Science from Yuan Ze University. He is a Ph.D. student in the Department of Computer Science from Yuan Ze University now. He was interesting in real-time multimedia transmission, active network, and the inter-media synchronization with Petri-net. His recent research has been in issue related to active network, programmable network, middleware network and service deployment technologies.



Kun-Hung Chen (陳坤鴻) received the B.S. and M.S. degrees in Computer Science and Engineering from the Yuan Ze University, Taiwan, in 2002 and 2004, respectively. His research interests include computer network, distributed system, and operating system.



Yung-Mu Chen (陳雍穆) received the B.S. and M.S. in Computer Science and Engineering from the Yuan Ze University, Taiwan. He is currently working toward the Ph.D. degree in Department of Computer Science and Engineering, Yuan Ze University, Taiwan. His primary research interests are in network, communications and distributed system, including Qos routing, intelligent network, overlay network, wireless network and grid. He is a student member of IEEE, ACM, and IEICE.