Small World Overlay P2P Networks

Ken Y. K. Hui and John C. S. Lui
Department of Computer Science & Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
{ykhui,cslui}@cse.cuhk.edu.hk

David K. Y. Yau
Department of Computer Science
Purdue University
West Lafayette, IN
yau@cs.purdue.edu

***Abstract*—We consider the problem of how to construct and maintain an overlay structured P2P network based on the *small world paradigm*. Two main attractive properties of a small world network are (1) low average hop distance between any two randomly chosen nodes, and (2) high clustering coefficient of nodes. Having a low average hop distance implies a low latency for object lookup, while having a high clustering coefficient implies the underlying network can effectively provide object lookup even under heavy demands (for example, in a flash crowd scenario). In this paper, we present a small world overlay protocol (SWOP) for constructing a small world overlay P2P network. We compare the performance of our system with that of other structured P2P networks such as Chord. We show that the SWOP protocol can achieve improved object lookup performance over the existing protocols. We also exploit the high clustering coefficient of a SWOP network to design an object replication algorithm that can effectively handle heavy object lookup traffic. As a result, a SWOP network can quickly and efficiently deliver *popular* and *dynamic* objects to a large number of requesting nodes. To the best of our knowledge, ours is the first piece of work that addresses how to handle dynamic flash crowds in a structured P2P network environment.**

Keywords: Small world phenomenon, structured P2P network, dynamic flash crowd.

I. Introduction

Peer-to-peer networks are distributed information sharing systems with no centralized control. Each node in a P2P network has similar functionalities and plays the roles of a server and a client at the same time. These systems provide immense flexibility for users in performing application-level routing, data posting, and information sharing in the Internet. The first generation P2P systems such as Napster require a centralized directory service. The second generation P2P systems (e.g., Gnutella and Freenet) are unstructured networks which use a fully distributed approach for object lookup. The major problem of the second generation systems is that the amount of query traffic necessary for object search may be enormous, leading to network congestion.

Recently, researchers have been working on distributed *structured* P2P network. Noted work includes Chord, Tapestry, and Pastry [13], [16], [19]. By applying the techniques of consistent distributed hashing and structured routing, these structured networks improve the efficiency of object lookup and reduce the amount of query traffic inside the network. For example, Chord has a worst case lookup complexity of $O(\log N)$ link traversals, where N is the number of nodes in a Chord network.

In this paper, we address two fundamental questions about the design of a distributed structured P2P network. They are:

- Can one further improve the performance of object lookup beyond the existing approaches?
- How can a P2P network handle heavy demands for *popular* and *dynamic* objects as in a flash crowd scenario? For example, immediately after the 9/11 incident, a large number of users tried to get the latest news about the incident from the CNN web server. The overwhelmed server was able to provide service for only a small fraction of the requesting users.

To address the first technical question, we propose to construct a P2P network having a *small world* structure [7], [17], in which the average shortest hop distance between two randomly chosen nodes is around six. To overcome the second problem, we take advantage of the high clustering coefficient property of a small world network to quickly *self organize* and *replicate* popular dynamic objects in the network.

We will present a small world overlay protocol (SWOP) to *construct* and *manage* a P2P network such that it exhibits small world properties. We show that the constructed small world P2P network has better object lookup performance than other structured P2P networks such as Chord. We will also illustrate that the small world network is robust under heavy traffic loading and can be used to satisfy requests for highly popular and dynamic data objects.

The balance of the paper is organized as follows. In Section II, we present the SWOP protocol for constructing and maintaining a small world P2P network. We also describe an object lookup protocol used by individual client nodes to locate any data object. In Section III, we present an algorithm for handling a high loading of query traffic as in flash crowd scenarios. Both static and dynamic flash crowds are considered. In Section IV, we discuss related work. Section V concludes.

II. Small world P2P Protocol

In this section, we first provide some necessary background about small world networks and state their important properties. We then present protocols for constructing and maintaining a small world P2P network, and for performing data lookup in the network. We derive analytically an upper bound for the average object lookup latency. Lastly, we present experimental results to illustrate the efficiency of data lookup in our system, when compared with the Chord protocol.

The notion of *small world phenomenon* originates from social science research [10]. It has developed to become a very active current research topic in physics, computer science, and mathematics [11]. It has been observed that the small world phenomenon is pervasive in a wide range of settings such as social communities, biological environments, and data/communication networks. For example, recent studies (e.g., [18]) have shown that peer-to-peer networks such as *Freenet* may exhibit small world properties. Informally, a small world network can be viewed as a connected graph in which two randomly chosen nodes are connected by just about six degrees of separation. In other words, the *average shortest distance* between two randomly chosen nodes is approximately six hops. This property implies that one can locate information stored at any random node of a small world network by only a small number of link traversals.

One way to construct a small world network is the following: (1) Each node in the network is connected to some neighboring nodes, and (2) each node keeps a small number of links to some randomly chosen "distant" nodes. Links to neighboring nodes are called *cluster links* while links to distant nodes are called *long links*. Figure 1 illustrates an example of a small world network with 11 nodes and six clusters. In the figure, nodes 9, 10, and 11 form one cluster. The nodes have neighboring links to each other, and node 9 has long links to nodes 6, 14, and 22.

Two important properties of a small world network are (1) a low average hop count between two randomly chosen nodes, and (2) a high clustering coefficient. To mathematically define the two properties, let $\mathcal{G} = (V, E)$ denote a connected graph representing a small world network. There are N vertices in $\mathcal{G}$ where $|V| = N$ and $D(i,j)$ represents the length (in hops) of the *shortest path* between two vertices i and $j \in V$. We have the following definitions:

Definition *1:* The average shortest hop count of a graph $\mathcal{G}$, denoted as $\mathcal{H}(\mathcal{G})$, is equal to

$$\mathcal{H}(\mathcal{G}) \quad = \quad \frac{1}{\binom{N}{2}} \sum_{i,j \in V} D(i,j). \tag{1}$$

$\mathcal{H}(\mathcal{G})$ is the ratio of the sum of all shortest paths between any two nodes in $\mathcal{G}$ and all possible pairwise connections of the connected graph.

To define the clustering coefficient, let κ_v be the number of attached links for a node $v \in V$. The *neighborhood* of a vertex v is a set of vertices $\Gamma_v = \{u : D(u,v) = 1\}$.

Definition *2:* For a given vertex $v \in V$, let C_v be the *local cluster coefficient* of v, which is equal to $C_v = |E(\Gamma_v)|/\binom{\kappa_v}{2}$, where $|E(\Gamma_v)|$ is the operator of counting the total number of links for all vertices in the set Γ_v. The cluster coefficient of a graph $\mathcal{G}$, denoted as $\mathcal{C}(\mathcal{G})$, is equal to

$$\mathcal{C}(\mathcal{G}) \quad = \quad \frac{1}{N} \sum_{v \in V} C_v. \tag{2}$$

In other words, $\mathcal{C}(\mathcal{G})$ measures the degree of compactness of the graph $\mathcal{G}$.

In the following, we first describe protocols for constructing and performing data lookup in a constructed small world P2P network. Then, we provide a mathematical analysis on the worst case average number of link traversals to locate an object in the network. Lastly, we show that when compared with other existing structured P2P networks, our small world P2P network achieves a *lower* number of link traversals for object lookup traffic.

A. Small World Overlay Protocol (SWOP)

The *small world overlay protocol* (SWOP) is designed to efficiently locate any object in the network. In particular, it can support efficient access to *popular* and *dynamic* objects under heavy traffic loading. SWOP is constructed as a layer on the top of a structured P2P network. This layer does not affect the functionalities provided by the P2P network layer, but it can improve the performance of object lookup in the network.

Let us give some brief background of a structured P2P protocol. Generally, the protocol consists of a consistent hashing function h (e.g., the SHA-1 function) to provide unique key assignments for each node or object in the network. With the key's value, each node can determine its logical position in the system. For example, for a Chord network, the logical position of a node is a point in a circular key space. For a CAN network, it is a point in a grid. Another property of a structured P2P protocol is its use of a routing table (e.g., the *finger table* in Chord), which speeds up the object lookup process. It has been shown that the worst case number of link traversals to locate an object in Chord is $O(\log(N))$, where N is the number of nodes in the network [16].

Each node can insert objects into the structured P2P system using the same consistent hashing function h. Each object has a unique key value; say, $h(o)$ is the key value of object o. Let $\mathcal{N}(o)$ be the set of nodes whose key values are greater than or equal to $h(o)$. The node in $\mathcal{N}(o)$ which has the minimum key value is responsible for maintaining the object o.

For a SWOP network, we use a circular ring as our logical representation, since it is a representative model in structured P2P networks and it helps to reveal the small world effects introduced by SWOP. Let us define the parameters for SWOP:

- Cluster size G: the maximum number of nodes within a cluster.
- Cluster distance D: the maximum hash space distance between two adjacent nodes within a cluster.
- The number of long links k in a cluster.

For our SWOP network, there are two types of node, namely *head nodes* and *inner nodes*, and two types of link, namely *long links* and *cluster links*. Long links connect two different nodes from different clusters while cluster links connect two different nodes in the same cluster. Each cluster has *one* head node, which has at most k long links and cluster links to all the nodes within its cluster. An inner node has a link to the head node within its cluster and cluster links to some of the nodes within its cluster. With the above configuration, an inner node, say i, can communicate with a target node, say j, within its cluster either by a cluster link (provided i and j are connected), or node i can send a message to its head node, and then the head node forwards the message to node j using a cluster link. For communicating with a target node in a different cluster, node i has to first send the message to its head node, and then the head node sends the message using the long link which is the closest to the target node j. The message may arrive at some nodes which are not within the same cluster as node j. The procedure is then repeated until the message is transferred to some node within the same cluster as node j. In figure 1, we show an example of a SWOP network with 11 nodes, six clusters, and parameters of $G = 3$, $D = 2$, and $k = 3$. In the figure, the flow of object lookup messages is illustrated when node 1 sends a lookup request to node 17.

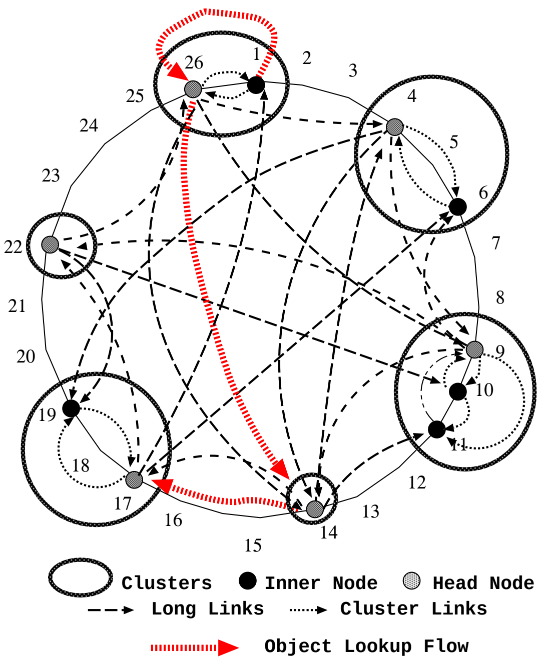

Fig. 1. A SWOP network with six clusters, $G = 3, D = 2, k = 3$ and the object lookup flow.

In the following, we describe protocols for forming and maintaining a SWOP network. The events of node join, node leave, and node failure are addressed.

Join Cluster Protocol (JCP): If a node i wishes to join a small world P2P network, it uses the consistent hashing function h to obtain its key $h(i)$. At the same time, node i creates a link to its predecessor node a and its successor node b in the underlying P2P network. The predecessor node a is an existing node in the network whose key value $h(a)$ is the largest key value such that $h(a) < h(i)$. The successor node b is an existing node in the network whose key value $h(b)$ is the smallest key value such that $h(b) > h(i)$. After finding its predecessor and successor nodes, node i executes a *join cluster protocol (JCP)*. The joining node i first determines the *distance* (which is defined based on the hashed key value) between the predecessor and successor nodes. Let d_1 and d_2 be the distances between the joining node i and its predecessor and successor nodes, respectively; i.e., $d_1 = h(i) - h(a)$ and $d_2 = h(b) - h(i)$. The joining node i asks its predecessor and successor nodes about their respective cluster sizes. If both nodes a and b have a cluster size greater than G (the maximum cluster size), the joining node i will form a *new* cluster with itself being the only node in the new cluster. Otherwise, i determines which cluster to join depending on the values of d_1 and d_2, as follows. If both d_1 and d_2 are greater than D, i will form a new cluster with itself being the only node in the new cluster. Otherwise, i joins the predecessor's cluster (respectively, the successor's cluster) if d_1 (respectively, d_2) is less than D and is smaller than d_2 (respectively, d_1).

Next, the joining node i determines whether it should be a head node or an inner node. If the node i forms a new cluster, or if it joins its successor node b which was a head node of that cluster, the joining node i becomes the new head node of the cluster. Otherwise, i becomes an inner node. When i is an inner node, it creates a cluster link to its head node. On the other hand, if the joining node i is the head node, it creates cluster links to all the inner nodes within the cluster and generates k long links to some other nodes in different clusters.

In order to achieve a low average shortest hop count $\mathcal{H}(\mathcal{G})$ and high cluster coefficient $\mathcal{C}(\mathcal{G})$, one needs to generate long links based on the following distance dependent probability density function $p(x)$. Let m be the number of clusters in a SWOP network. The head node generates a random variable $\tilde{X}$, which has the following probability mass function:

$$\text{Prob}[\tilde{X} = x] = p(x) = \frac{1}{x \ln(m)} \quad \text{where } x \in [1, m].$$

The above probability mass function is biased towards nodes that are farther away from the head node. Given the value of x, the head node creates a long link between itself and a random node that is in cluster x. It is important to point out that a long link serves as an *express link* between nodes in two different clusters. The *cluster distance* between two different clusters is defined as the number of long link traversals between the two clusters.

Leave Cluster Protocol (LCP): When the node i leaves the P2P system, it informs its neighboring nodes of its departure, by sending them a close_connection message and terminating its connection. The neighboring nodes are nodes connected to i by a cluster link, as well as nodes connected to i by a long link if i is a head node.

If a node receives a close_connection message, it will perform the following actions:

a) If the received message is from a neighbor connected by a long link, the receiving node will close the connection and generate a new long link to another node in a different cluster.
b) If the received message is from a neighbor connected by a cluster link, the receiving node will close the connection and ask its head node to reduce the cluster size by one.
c) If the received message is from a head node, the receiving node will close the connection. A node will become a new head node within a cluster if the received message is from its predecessor node which was the head node of the cluster. The new head node will also generate k long links using the probability mass function described above.

Stabilize Cluster Protocol (SCP): In the case of node failure, the SWOP network uses the SCP to recover from the failure and to maintain the proper link connectivities. Periodically,[1] each node sends probes to neighboring nodes to make sure that they are still operational. If a neighboring node is an inner node and it does not respond to the probe, the sending node will simply close the connection to the inner node and inform its head node to reduce the cluster size by one. If a neighboring node is a head node and it does not respond to the probe, the sending node will perform a search to find a new head node.

Object Lookup Protocol (OLP): The object lookup protocol is responsible for locating a data object within a small world P2P network. The object lookup process proceeds in two phases. In phase one, a node asks its cluster neighboring nodes if they contain the target object. If any of these cluster neighbors replies positively, then the lookup process is completed. Otherwise, the object lookup process continues and phase two begins. In phase two, the node first checks its own status within its cluster. If it is the head node, it forwards the lookup request to its long-link neighbor which has the closest distance to the target object. On the other hand, if it

[1]The period length is on the order of minutes.

is an inner node, it forwards the lookup request to its head node within the same cluster, and then the head node will recursively continue the object lookup process as described above. For example, when the long-link neighbor receives the object lookup request, it checks if it is the owner of the object. If not, it will act as if it is the node initiating the object lookup, and repeat the data lookup process. The lookup process continues in these two phases until the data object is found.

To illustrate, consider the example shown in Figure 1. Suppose that node 1 wants to look up an object whose key value is 16, and node 17 is responsible for managing this object. To begin the object lookup, node 1 starts phase one of the lookup process in which it asks its neighbor, node 26, if it contains item 16. Since node 26 is not the owner of the object, node 1 will receive a negative reply. Then node 1 starts phase two of the lookup process and forwards the lookup request to node 26, which is the head node within the cluster of node 1. Node 26 searches along its long link and forwards the object lookup message to its long link neighbor node 14, which is the closest node to the target object 16. Node 14 checks if it contains object 16. Since it does not contain the object, node 14 acts as an object lookup initiator node and starts another phase one lookup. Because node 14 is the only node within the cluster, it begins the phase two lookup and forwards the message to the nearest long link neighbor node 17. When the object lookup request reaches node 17, the object is found and the lookup process completes.

B. Mathematical Analysis

In this section, we mathematically analyze the *worst case* average number of link traversals for locating an object in a SWOP network.

Theorem *1:* Let Y be a non-negative random variable that represents the number of link traversals for object lookup in the SWOP network. We have:

$$E[Y] \quad \leq \quad (1 + \log_2(m/2)) 8 \ln(3m)/k, \qquad (3)$$

where m and k are the number of clusters and the number of long links, respectively, of the corresponding SWOP network.

Proof: Please refer to [3].

Remark: The importance of this theorem is that it can be used to estimate the proper value of k so that the SWOP P2P network has a better object lookup performance than other existing structured P2P networks.

C. Experimental results comparing SWOP with other structured P2P networks

In the following, we compare the object lookup performance between the SWOP network and Chord [16]. Our results show that the SWOP network can achieved better performance compared with its underlying Chord network.

Experiment A.1 (performance of object lookup): We consider a connected graph with $N = 2000$ nodes. We insert one object for each node, for a total of N distinct objects. Each node will perform object lookup 50 times, where the target object is randomly chosen from all the available objects. We measure the performance of object lookup as the number of message link traversals for both Chord and SWOP. The SWOP network is configured with parameters $G = 100$, $D = 120,000$, and $k = 24$. To obtain a fair comparison, we keep the size of the finger table in Chord to be also 24. Figure 2 shows the probability density functions of the number of link traversals for object lookup under Chord and SWOP, for $N = 2000$. Observe that the SWOP network has a *lower* average number of link traversals in object lookup than the Chord network.

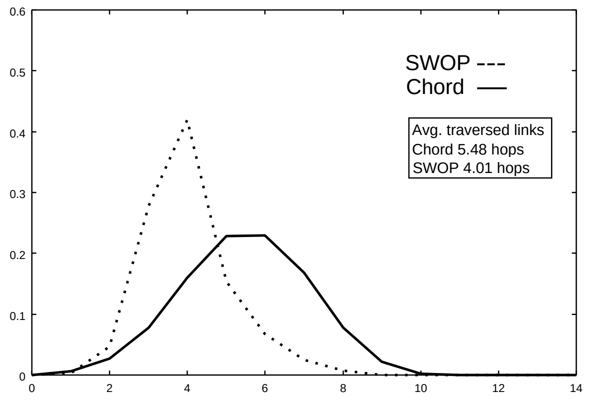

Fig. 2. Probability density function of link traversal for Chord and SWOP with $N = 2000$ nodes.

Experiment A.2 (Comparison of clustering coefficient): We next investigate the issue of the average clustering coefficients achieved by the two networks. (As discussed, a high clustering coefficient implies an improved ability to handle heavy traffic workloads.) In this experiment, we vary the number of nodes in the overlay network. For the Chord network, each node has a finger table of size equal to $\log(N)$. For a fair comparison, we also keep the number of long links for the SWOP network to be $k = \log(N)$. The clustering coefficient is computed based on Equation (2). The result is illustrated in Table I. Observe that the SWOP network has a *higher* clustering coefficient, implying that it can more effectively handle heavy traffic flash crowd scenarios. In the following section, we will explore this property in more detail.

N : # of nodes	Chord	SWOP
1000	0.288182	0.560587
2000	0.260332	0.649660
3000	0.250452	0.684012
4000	0.245469	0.704523
5000	0.240776	0.716463

TABLE I
AVERAGE CLUSTERING COEFFICIENT

III. Protocols for handling flash crowds

In this section, we address how to handle object access under heavy traffic loadings. Examples include flash crowd scenarios [14], [15] in which a large number of users try to access a popular object over a short period of time. For example, after the 9/11 incident, an enormous number of user requests overwhelmed several popular news sites, so that only a small number of users were able to obtain information while the majority of users could not.

One way to deal with flash crowds is to replicate the popular object in many nodes. This way, we spread out access to the object so as to avoid overwhelming a single source node. However, one has to address the following technical issues:

- The replication process cannot be arbitrarily self-initiated. Rather, it must be driven by a high traffic demand. Otherwise, someone may maliciously replicate many objects in the P2P network.
- In a traditional structured P2P network, object lookup is carried out by using the target object's key value and only *one* node is supposed to manage that object. How can one enhance the protocol so that more than one node can store the popular object?

We divide the study of the flash crowd problem into two cases: *static* and *dynamic*. The static flash crowd problem is concerned with heavy access to a popular object whose contents remain unchanged after the object was first inserted. Examples include a newly published book or a newly released video. On the other hand, the dynamic flash crowd problem is concerned with heavy access to a popular object whose contents will change over time, such as a frequently updated news article. We first describe algorithms for handling the static flash crowd problem. We then extend the algorithms to handle dynamic flash crowds by adding mechanisms to notify nodes of object changes, such that the objects can be updated efficiently.

A. Static flash crowds

In a flash crowd situation, lookup traffic can overwhelm a source node hosting a popular object. To avoid the problem, the source node needs to replicate this object in other nodes. The additional nodes can then serve some of the object lookup requests and reduce the traffic to the source node. Note that object replication has to be *demand driven*, or else a node may be able to maliciously replicate objects in all other nodes in the network, thereby wasting their resources. Implementation details of object replication can be found in [3]. In the implementation, each node periodically estimates the access rates of its own objects, thus allowing a node to maintain an *object demand list* used in the following algorithms.

Static-Chord algorithm: Assume that a node in a Chord P2P network can process up to λ_t requests per second with acceptable performance. Whenever a source node discovers that the request rate for an object is λ_1, where $\lambda_1 > \lambda_t$, it starts the replication process by pushing the popular object to *all* its neighboring nodes – i.e., all the nodes listed in its finger table. The neighbor nodes will cache the popular object for T_1 time units. Under the Chord object lookup protocol, any node that wishes to access the popular object may send a lookup request through these neighbor nodes in order to access the popular object. For a node that caches the popular object, if it receives lookup requests for the object at a rate λ_2, where $\lambda_2 > \lambda_t$, it will in turn push the object to *all* its neighboring nodes. The motivations for the above algorithm are: (1) the replication process of a popular object is purely demand driven, (2) for a popular object, it will be replicated in many nodes in the Chord network to support user access.

Static-SWOP algorithm: Note that a SWOP network exhibits a high clustering coefficient. We take advantage of this property to achieve a lower replication time and a lower number of link traversals in accessing the popular object. We also assume that a node in the SWOP network can process up to λ_t requests per second. Whenever a source node receives a request rate for an object equal to λ_1, where $\lambda_1 > \lambda_t$, the source node will start the replication process by pushing the popular object to its neighbors – i.e., all the nodes connected to the source node by *long links* in its cluster. All these neighbor nodes will cache the popular object for T_1 time units. Any node that wishes to access the popular object will use the *object lookup protocol* (OLP) described in Section II. If the popular object is cached by nodes in its cluster, requesting nodes can access the popular object more quickly. For a node that caches the popular object, if the lookup rate for the cached object is higher than λ_2, it will in turn push the cached object to all the nodes connected to it by long links of its cluster. Note that the main objective of replicating the popular object via *long links* is to propagate useful information to distant clusters, such that nodes in those clusters can also easily access the popular object.

B. Dynamic flash crowds

To handle the dynamic flash crowd problem, an additional communication message, called the *update message*, and an extra data structure for maintaining each object's version number are added to the static replication algorithm.

After the static algorithm has been applied in a SWOP network, each cluster in the network has exactly one node caching the popular object. Consider that at this point, each node caching the popular object will mark it as the "original copy", i.e., version 0. If an updated version, say version 1, is later inserted into the system by the source node, we only need a lightweight notification to inform all the cached nodes about the object update. This notification proces can similarly be carried out by exploiting the small world property of the network and the static replication scheme.

Hence, original copies of a popular object can be replicated using the static algorithm. Additional tasks are then carried out whenever the source node has to advertise a newer version. First, an update message is sent out by the source node and each caching node of the updated object. In general, there are two types of update message that can be sent by a node: (1) those sent to all the node's cluster neighbors, and (2) those sent to all the long link neighbors of each cluster head. The first type of message involves the cluster neighbors only; it reminds these neighbors to look up the latest version of the popular object being cached. The second type of message involves the long link neighbors; it informs these neighbors of the new update and transfers the updated popular object to them. These messages ensure that an updated version will be replicated from one cluster to another. If the sending node is not a head node, this type of message requires the cooperation between the sending node and a head node. As a receiver of the update message, if a node does not contain a cached copy of the popular object, it will use the OLP protocol described in Section II to retrieve the updated object.

C. Mathematical analysis of object replication time

In this section, we present a mathematical analysis of the average time needed to replicate a popular object in all the clusters in a SWOP network. We model the spreading process by a continuous time Markov chain (CTMC).

SWOP network: Let $\mathcal{M}$ be the CTMC representing the replication dynamics of a SWOP network. The SWOP network has N nodes. The CTMC $\mathcal{M}$ has a state space of $\mathcal{S}$ such that $\mathcal{S} = \{1, 2, \cdots, m\}$, where m is the number of clusters in the SWOP network. State i in $\mathcal{M}$ represents that the popular object has already been replicated to i nodes in the SWOP network. Since the source node contains and manages the popular object initially, the initial state of the CTMC $\mathcal{M}$ is state 1. Define x_1 to be the number of requests needed within a time period τ so that the source node of the popular object will start replicating the popular object to all its long link neighbors (e.g., $\lambda_1 = x_1/\tau$). After receiving the popular object, these neighbor nodes become the *replicated nodes*. Similarly, define x_2 as the number of object requests needed for a replicated node to start replicating the popular object to all its long link neighbors. Let λ be the request rate for the popular object from each node in the SWOP network. The rate of replicating a popular object from the source node is λ_s, which is equal to:

$$\lambda_s = (N-1)\lambda \left[1 - \sum_{j=0}^{x_1-1} \frac{e^{-(N-1)\lambda\tau}((N-1)\lambda\tau)^j}{j!} \right].$$

Similarly, let $\bar{C}$ represent the average number of nodes within a cluster. The rate of replicating a popular object from a replicated node is λ_r, which is equal to:

$$\lambda_r = (\bar{C}-1)\lambda \left[1 - \sum_{j=0}^{x_2-1} \frac{e^{-(\bar{C}-1)\lambda\tau}((\bar{C}-1)\lambda\tau)^j}{j!} \right].$$

Let $\boldsymbol{Q}$ be the infinitesimal rate matrix of $\mathcal{M}$, and we denote the elements in $\boldsymbol{Q}$ as $q_{i,j}$, which are the transition rates from state i to state j, for $i, j \in \{1, 2, \ldots, m\}$. Assume that a replicated node will cache the object for an average time of $1/\mu$. Let $\nu_1 = \lambda_s$ and $\nu_i = \lambda_r$ for $i \in \{2, \ldots, m\}$. The transition rate matrix of Q can be specified by the following transition events:

Object deletion event:

$$q_{i,i-1} = i\mu \qquad \text{for } 1 < i \leq m \tag{4}$$

Object replication event:

Case 1: For state $i \in \mathcal{S}$, if $(i + ik) \leq m$:

$$q_{i,j} = \begin{cases} (\frac{m-i}{m})^{j-i}(\frac{i}{m})^{ik-(j-i)}\nu_i & \text{if } k \leq j \leq (i+ik) \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Case 2: For state $i \in \mathcal{S}$, if $(i + ik) > m$:

$$q_{i,j} = \begin{cases} \sum_{x=m}^{(i+ik)}(\frac{m-i}{m})^{m-i}(\frac{i}{m})^{x-(m-i)}\nu_i & j = m \\ (\frac{m-i}{m})^{j-i}(\frac{i}{m})^{ik-(j-i)}\nu_i & \text{if } k \leq j \leq (i+ik) \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Once the rate matrix $\boldsymbol{Q}$ is specified, one can derive the average time to replicate a popular object to all the clusters in a SWOP network using the theory of fundamental matrix in Markov Chain [1], [12].

We first transform the rate matrix $\boldsymbol{Q}$ to a discrete time transition probability matrix $\boldsymbol{P}$ by using the uniformization technique [1], [12] such that $\boldsymbol{P} = \boldsymbol{I} + \boldsymbol{Q}/\Lambda$, where $\boldsymbol{I}$ is an identity matrix and Λ is a maximum absolute value for all the entries in $\boldsymbol{Q}$. Since we want to find the average time it takes to replicate a popular object, we can consider the state m in $\mathcal{M}$ as an absorbing state: i.e., this is the state wherein the popular object has been replicated to all the clusters in the SWOP network. Let $\boldsymbol{P}_c$ be the square matrix which is equal to $\boldsymbol{P}$ except that we remove the last row and column (i.e., the absorbing state m) from $\boldsymbol{P}$. The fundamental matrix $\boldsymbol{M}$ can be calculated using

$$\boldsymbol{M} = (\boldsymbol{I} - \boldsymbol{P}_c)^{-1} = \sum_{i=0}^{\infty} (\boldsymbol{P}_c)^i. \tag{7}$$

Let $E[T_c]$ be the average time to replicate the popular object to all the clusters in the SWOP network given that only one node (or cluster) has the popular object at time $t = 0$. We can compute $E[T_c]$ by:

$$E[T_c] = \left(\frac{1}{\Lambda}\right) \boldsymbol{e}_1 \boldsymbol{M} \boldsymbol{1}^T \tag{8}$$

where $\boldsymbol{e}_1$ is a row vector of zeroes except the first entry being one and $\boldsymbol{1}$ is a row vector of all ones.

D. Experimental Results

In this section, we present experimental results of comparing the performance of replicating a popular object for a Chord network and a SWOP network. The performance metric is the *effective object replcation time*, which measures the number of nodes that can successfully access the popular object by time t. To carry out the experiments, the topology generator developed in Section II and a discrete event driven simulator are used.

In our experimental study, $N = 2000$ nodes are generated in a 24-bit hash space P2P system. Each node has 11 long link neighbors for both SWOP and Chord. At all time, each node generates requests for one popular object, which is randomly picked at the beginning of an experiment. The requests generated are Poisson with an arrival rate of λ. Also, each node processes a request with a Poisson service rate of μ, normalized to 1, and has a finite queue for buffering incoming requests. If the queue is full, newly arriving requests will be dropped.

Experiment B.1: Comparison between Chord and SWOP: This is a basic comparison between Chord and SWOP for the static and dynamic flash crowd scenarios. We fix the per node average request arrival rates for both Chord and SWOP to be $\lambda = 0.003$ requests/second. Under the dynamic flash crowd scenario, the source node managing the popular object updates the version of the popular object at time $t = 25, 50$, and 75.

Static Result: Figure 3 shows the number of successful lookup requests processed as a function of time. The results show that the SWOP network has a much better performance in object replication than Chord. For example, beginning at time 20, most of the requests for the popular object are successfully processed under SWOP. In contrast, only around 6% of the requests are successful under Chord.

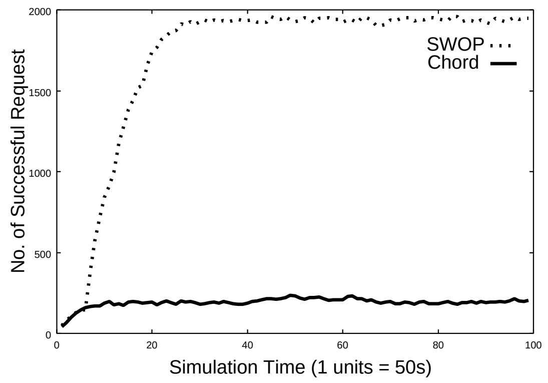

Fig. 3. Comparison of the number of successful requests for a popular object under the static flash crowd with $N = 2000$ nodes.

Dynamic Result: Figure 4 illustrates the performance of the Chord and SWOP networks under a dynamic flash crowd scenario. It uses the same plot as the previous static case results. Since the popular object is updated at time $t = 25, 50$, and 75, we consider a request successful if and only if a requesting node can access the most up-to-date version of the popular object. Figure 4 shows that the number of successful requests processed under SWOP is significantly higher than under Chord. Moreover, when the object changes its contents at time t = 25, 50, and 75, the SWOP network can quickly notify other nodes of the updates such that the requesting nodes can eventually access the most recent version of the popular object. In comparison, the Chord network is not as effective as SWOP in propagating the update information. Hence, SWOP performs significantly better in the dynamic flash crowd scenarios.

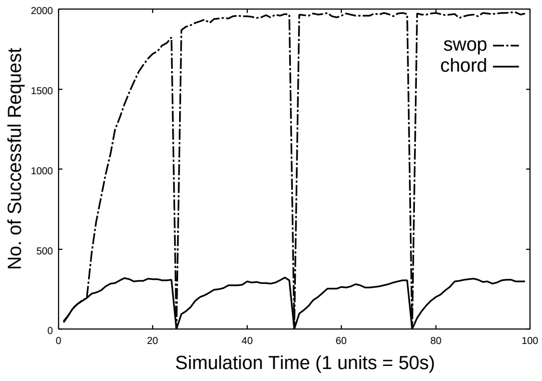

Fig. 4. Comparison of the number of successful lookup requests under a dynamic flash crowd with $N = 2000$ nodes. (object changes its version at t = 25, 50, and 75.)

Experiment B.2: Variations in object request rate: In this experiment, we further examine system performance in handling dynamic flash crowds when the object request rate is being varied. We keep the same configuration as in Experiment B.1. However, we vary the per node object request rate λ from 0.001 to 0.005. The results are shown in Figure 5. They show that SWOP performs the best under the arrival rate of 0.003. The reason is that when the request rate is greater than 0.003, the aggregate request rate is higher than the service rate for the individual nodes caching the popular object within a cluster. On the other hand, when the request rate is smaller than 0.003, the aggregate request rate is small enough for achieving a very high number of successful requests.

IV. **Related Work**

The small world phenomenon was first observed by Milgram [10], who discovered the interesting six

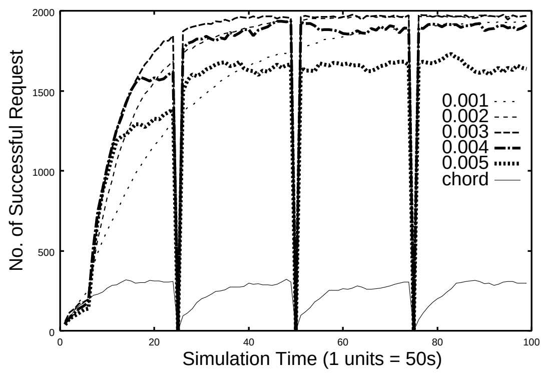

Fig. 5. Effect on the variation of per node request rate λ on the number of successful request for the SWOP & the Chord network (object changes its version at t = 25, 50, 75.)

degrees of separation in a social network. Kleinberg [6], [7] provides a theoretical framework for analyzing graphs with small world properties. In [2], the authors study the broadcast problem for communication in a small world network. In [18], the authors propose a scheme for storing data in an *unstructured* P2P network such as Freenet, such that the P2P network may exhibit some of the small world properties. In comparison, our proposal focuses on structured networks employing a consistent hashing function. Moreover, we have applied our techniques to resolve the dynamic flash crowd problem. In [4] and [5], the authors propose to form a small world P2P network for scientific communities. However, the details of creating and managing such a network are not clearly specified.

Recent research on structured P2P networks can be found in [9], [13], [16], [19]. The main feature of this body of work is to provide some form of data/topological structure for an overlay in order to support efficient object lookup without generating an excessive amount of query traffic. Compared with their approaches, the SWOP protocol further improves the performance of object lookup. In addition, we have proposed an efficient way to *replicate* popular and dynamic objects thereby helping to resolve the dynamic flash crowd problem. Ulysses [8] is a structured P2P network based on the concept of butterfly network and shortcut intuition. Their proposed protocol achieves a low number of link traversals for object lookup. However, the performance relies on a stable topology. If a query is routed through nodes performing a join or leave operation, the performance is not known. Moreover, they consider only moderate traffic loadings and do not address heavy traffic loading situations like flash crowds. In comparison, SWOP can take advantage of a high clustering coefficient in handling the dynamic flash crowd problem. The Cooperative File System has been proposed as a new peer-to-peer read only storage system, which helps handle the flash crowd problem. However, it requires a large storage overhead for storing the file information. Our SWOP network requires only a small amount of additional storage, namely $O(G+k)$ routing table entries, in handling the flash crowd problem. In [15], the authors propose a protocol for handling the static flash crowd problem in a P2P network. An elegant analysis of the static flash crowd problem is presented in [14]. In comparison, our work has focused on the dynamic flash crowd problem since popular data objects may have time varying contents.

V. **Conclusion**

The small world phenomenon is an active field of research in the social sciences, physics and mathematics. A small world graph has two important properties: a low average hop distance between two randomly chosen nodes and a high clustering coefficient. In this paper, we have proposed protocols to create and manage a small world structured P2P network. We show that the proposed small world network has both a small average hop distance between nodes and a high clustering coefficient of nodes. We have demonstrated how a low average hop distance between nodes can reduce the number of link traversals in object lookup. We have also proposed a protocol to replicate popular and dynamic objects in order to handle the dynamic flash crowd problem. We have conducted experiments to compare the performance of the proposed small world network with that of other structured P2P networks such as Chord. Our results show that a SWOP network can achieve a lower object lookup latency and can effectively satisfy a large number of users requesting a popular data object.

Acknowledgement

J. C. S. Lui was supported in part by an RGC Earmarked Grant. D. K. Y. Yau was supported in part by the National Science Foundation under grant numbers CCR-9875742 (CAREER) and CNS-0305496.

References

[1] U. Bhat. Elements of applied stochastic processes. *John Wiley & Son, New York*, 1984.

[2] F. Comellas and M. Mitjana. Broadcasting in small-world communication networks. *Proc. 9th Int. Coll. on Structural Information and Communication Complexity*, 13:73–85, 2002.

[3] Y. K. Hui, C. S. Lui, and K. Y. Yau. Small-world overlay p2p networks. *The Chinese University of Hong Kong , Technical report CS-TR-2004-04*, April, 2004.

[4] A. Iamnitchi, M. Ripeanu, and I. Foster. Locating data in (small-world?) peer-to-peer scientific collaborations. *First International Workshop on Peer-to-Peer Systems, Cambridge, MA.*, 2002.

[5] A. Iamnitchi, M. Ripeanu, and I. Foster. Small-world file-sharing communities. *IEEE infocom*, 2004.

[6] J. Kleinberg. Navigation in a small-world. *Nature*, 406, 2000.

[7] J. Kleinberg. The small-world phenomenon: an algorithmic perspective. cornell computer science technical report 99-1776. 2000.

[8] A. Kumar, S. Merugu, J. J. Xu, and X. Yu. Ulysses: A robust, low-diameter, low-latency peer-to-peer network. In *IEEE International Conference on Network Protocols*, 2003.

[9] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In *Proc. of the 21st annual ACM symposium on principles of distributed computing*, 2002.

[10] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.

[11] M. E. J. Newman, I. Jensen, and R. M. Ziff. Percolation and epidemics in a two-dimensional small world. *Phys. Rev. E65*, 2002.

[12] E. Parzen. Stochastic processes. *Holden-Day, San Francisco, California*, 1962.

[13] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. *18th IFIP/ACM Int. Conference on Distributed System Platforms*, pages 329–350, Nov,2001.

[14] D. Rubenstein and S. Sahu. An analysis of a simple P2P protocol for flash crowd document retrieval. *Columbia University, Technical report EE011109-1*, November, 2001.

[15] A. Stavrou, D. Rubenstein, and S. Sahu. A lightweight, robust P2P system to handle flash crowds. *IEEE ICNP*, November, 2002.

[16] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. pages 149–160.

[17] D. Watts. *Small-worlds: The dynamics of Networks between order and randomness*. Princeton University Press, 1999.

[18] H. Zhang, A. Goel, and R. Govindan. Using the small-world model to improve freenet performance. *Proc. IEEE Infocom*, 2002.

[19] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. pages Tech. Report, UCB/VSD–01–1141, U.C. Berkeley, 2001.