Entropy and Search Distance in Peer-to-Peer Networks

Saurabh Tewari, Leonard Kleinrock
Computer Science Department, University of California at Los Angeles, Los Angeles, CA 90095, U.S.A.
{stewari,lk}@cs.ucla.edu

Abstract— *We had previously shown that the average search distance in unstructured peer-to-peer networks having an Erdos-Renyi random graph search topology is minimized when the number of replicas of objects is proportional to the object request probabilities and this results in a minimum search distance is equal to the entropy in object request probabilities minus the logarithm of per-node cache size. In this paper we show that these results also hold for DHT-based peer-to-peer networks that employ prefix-based routing. The advantage of proportional replication in minimizing the search distance adds to the benefits of proportional replication for download time, network resources usage and fairness we have found in our earlier work.*

In order to show the above result, we generalize a recently proposed replication approach where an object is replicated at nodes with ids adjacent to the id of the node responsible for the object. We show that the average search distance for an object is logarithmically related to the number of replicas of the object in prefix-routing-based DHT systems when the replicas are placed in the manner described. This result can be easily extended to randomized DHTs that make exponential progress to the destination address in O(1) steps.

Finally, we generalize the relation between the optimal average search distance and entropy in object request probabilities to any peer-to-peer network with an overlay topology that exhibits exponential expansion regardless of whether the network is structured or unstructured by mapping the search problem in a peer-to-peer network to a coding problem.

1. Introduction

Peer-to-peer networks are loosely organized networks of autonomous entities (user nodes or "peers") which make their resources available to other peers. Since each new peer brings additional resources, these networks are fully scalable provided that the resources each peer offers can be found by the peers who need those resources. Thus, finding the desired resource is a critical issue in peer-to-peer networks. Keeping a centralized index of the resources each peer is offering is an approach that has scalability issues and a single point of failure. Alternatively, a direct approach for finding the desired resource is to have the peer wanting a resource to query other nodes to find a node that has that resource. Since a node cannot realistically keep the addresses of all other peers, an overlay network is constructed where each node keeps addresses of a few other peers (called its *neighbors*) through whom it reaches the rest of the peers. Peer-to-peer networks following this approach are referred to as *unstructured* peer-to-peer networks to distinguish them from structured networks [7, 9] which map each unique resource to a particular node in the network. Due to the resource-to-node mapping, searching for a resource in structured networks is equivalent to finding the node responsible for the resource. Unlike unstructured networks, well-defined paths can be followed in structured networks to locate any node. However, the efficiency of structured networks is at the cost of lack of flexibility which introduces other issues [16].

In both the structured and unstructured approaches, a key issue is the average number of hops (the *average search distance*) to find a peer who is offering a particular resource. In our earlier work [9, 12], we had investigated the minimum average search distance possible in unstructured peer-to-peer networks. We present these results in Section 2. In Section 3, we investigate the same issues for structured networks that use prefix-routing and find the same answers when the replicas are placed at nodes with ids adjacent to the id of the node responsible for the object, i.e. *(i)* the average search distance to an object is logarithmically related to the number of replicas of that object; *(ii)* the average search distance is minimized when the number of replicas of each object is proportional to the request probability of the object; and *(iii)* the minimum average search distance is equal to the entropy in object request probabilities minus the logarithm of the per-node cache size expressed in units of the number of objects that can be stored. This similarity in results suggests that the results

may be applicable to any peer-to-peer network. In Section 4, using information-theoretic arguments, we show that the average search distance in peer-to-peer networks with topologies that exhibit exponential expansion is lower bounded by the entropy in object request probabilities minus the logarithm of the per-node cache size. This result gives us an alternate method of showing the optimality of proportional replication for average search distance in structured and unstructured networks independent of the Lagrangian optimization approach used in [9, 12] and Section 3. Section 5 discusses some of the relevant work and Section 6 concludes this paper.

2. Optimal Search Distance in Unstructured Networks

We had conducted a detailed study of search performance in unstructured networks in [9]. In this section, we present the relevant results on average search distance from [9] for controlled flooding search.

The first result expresses the relation between the average search distance for an object and the number of replicas of the object and is summarized in the following theorem.

Theorem 1: In unstructured peer-to-peer networks with an Erdos-Renyi random graph search network topology, under the assumption that the replicas of an object are uniformly distributed over the nodes in the network, the average search distance to an object i, $\tau_i(n_i)$, is related to the number of replicas of object i, n_i, as

$$\tau_i(n_i) = \log_d(M/n_i) \qquad (1)$$

when $M \to \infty$ and $n_i/M << 1$, where M is the number of nodes in the network and d is the average per-node degree of the search network topology. ■

The average search distance averaged across all requests, τ, is

$$\tau = \sum_{i=1}^{N} \frac{\lambda_i}{\lambda} \tau_i(n_i) \qquad (2)$$

where λ_i is the request rate for object i for $i = 1,2, \ldots, N$, $\lambda = \sum_{i=1}^{N} \lambda_i$ and $\tau_i(n_i)$ is as given in (1).

The second result provides the replica distribution that minimizes the average search distance and is summarized in the following theorem from [9].

Theorem 2: Given an unstructured peer-to-peer network where the relation between the hop distance $\tau_i(n_i)$ to the nearest replica and the number of replicas n_i is of the form $\tau_i(n_i) = \alpha \log_d(M/n_i)$ and the storage capacity at each node is the same and is equal to K objects, then the average search distance in controlled flooding query searches is minimized when the number of replicas, n_i, of object i is piece-wise linear with respect to the object request rate λ_i, (i=1,2, …, N) i.e.

$$n_i = \begin{cases} \frac{\lambda_i}{\lambda} KM & if \ \frac{1}{KM} \le \frac{\lambda_i}{\lambda} \le \frac{1}{K} \ \forall i, \qquad (3) \\ Max(1, Min(\gamma \lambda_i, M)) & \\ \quad where \ \gamma is \ s.t. \sum_{i=1}^{N} n_i = KM & \text{in the general case.} \qquad (4) \end{cases}$$

■

Figure 1 shows this result for the two cases described.

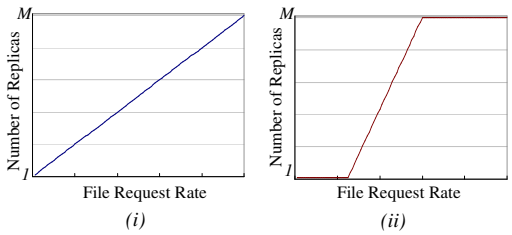

Figure 1. Optimal Replica Distribution: (i) under constraints on file request rates (ii) in the general case

The third result is directly obtained by substituting (3) in (2) and (1) as summarized in the following theorem.

Theorem 3: In unstructured peer-to-peer networks with Erdos-Renyi random graph search network topology, under the assumption that the replicas of an object are uniformly distributed over the nodes in the network and the storage capacity at each node is equal, the minimum average search distance in controlled flooding query searches is given by

$$\tau_{\text{opt}} = -\sum_{i=1}^{N} \frac{\lambda_i}{\lambda} log_{\text{d}} \frac{\lambda_i}{\lambda} - log_{\text{d}} K \qquad (5)$$

if $\frac{1}{KM} \le \frac{\lambda_i}{\lambda} \le \frac{1}{K}$ $\forall i$, where λ_i is the request rate for object i for $i = 1,2,\ldots, N$, $\lambda = \sum_{i=1}^{N} \lambda_i$, N is the number of objects in the network, M is the number of nodes in the network, K is the per-node cache size in number of objects and d is the average per-node degree of the search network topology. ■

We note that λ_i/λ is the probability a request will be for object i and, throughout the rest of the paper, we will use object request probability p_i instead of λ_i/λ.

In the next section we show that the same results hold for structured networks based on DHTs that use prefix-based routing.

3. Optimal Search Distance in Structured Networks

3.1 Distributed Hash Tables

A popular approach to solving the search problem is maintaining a Distributed Hash Table (DHT) over the participating peers – – the hash table is split into disjoint partitions and each peer is assigned the management of one partition based on its "ID", a hash function maps each resource to a hash-value and the peer responsible for the partition that includes this hash value is the contact point for this resource. Therefore, in DHT-based systems (the ID of) the contact point for any resource can be computed. However, one still needs to know the physical location (the IP address) of this contact point. Since in most cases it is impractical for each node to store the IP address of every peer in the network, links are maintained to a subset of peers selected based on their IDs (the IP address of these peers is stored so they are reachable). Taken together, these links form a routing overlay network and each peer is reachable via this overlay network. The *search distance* to a resource (in our definition) is the average number of hops needed on the overlay network to reach the peer responsible for that resource.

There are many competing DHT proposals that differ in the routing overlay network they use and the manner in which they handle peer dynamics among other things. In this paper, we restrict our attention to the systems that make exponential progress towards the target in each step (as opposed to *tori*-based systems like [6]). In the deterministic version of these systems [4, 7, 8, 14], "hard-wired" links at maintained at each resolution (e.g. if there are 16 partitions and a peer has the node ID, say, 0000, it will maintain a link to node ID 1xxx, another link to node ID 01xx and another to node ID 001x). In the randomized version [2, 3], each peer has a few short-distance links to their successors and predecessors to ensure (ring-like) connectivity and a few long-distance links for the "small-world" effect (i.e. a link with a node at distance x with probability $1/x$). In these types of networks, in a network of M nodes, the distance to any node is O(log$M$) as (with high probability) the distance can be cut in half by an appropriate long-distance link or by following a few short-distance links until a suitable long-distance link is found. In this paper we show our results for deterministic topologies but we believe that the procedure used can be easily modified to show the same results for the randomized topologies.

The standard DHT framework precludes any replication (a replica of a resource will map to the same node) and earlier systems [14] suggested caching query results at intermediate hops orthogonal to the DHT structure. Such replication has limited utility in reducing the search distance but an alternate method is proposed in [5]

which replicates an object (or the pointer to the object) at peers adjacent[1] (in the id space) to the peer responsible for that object. The proposal in [5] replicated an object 2^i times to decrease its search distance by i hops. In the next section, we generalize their approach to show that the average search distance to object in DHTs that use prefix-based routing is related to the number of replicas of the object in the same manner as described in (1).

3.2 Average Search Distance vs. Object Replication

For DHTs that use prefix-routing, [5] proposed that the search distance can be reduced by replicating an object at all nodes logically preceding the node responsible for the object on all query paths. An approximate description of the technique is as follows. Since each node maintains a link at each resolution, if an object is replicated at all nodes sharing the same prefix then, effectively, the object "address" has fewer bits and, hence, fewer hops are needed to reach the object. For example, if an object stored at '0121' is replicated at all nodes with an ID prefix of '01' (i.e. nodes with ID '01xx'), then no more than 2 steps would be needed to locate the object (it is sufficient to reach any node within the '01' prefix).

We now extend the replication strategy in [5] to obtain the following theorem.

Theorem 4: In a peer-to-peer network of M nodes, where $M=2^k$, that employs a DHT that uses prefix-routing for search, if an object i has n_i replicas placed at the nodes adjacent (on the unit perimeter circle) to the node responsible for the object, then for $M\rightarrow\infty$ and $n_i/M << 1$, the average number of hops to reach object i, $\tau_i(n_i)$, is given by

$$\tau_i(n_i) = \log_k(M/n_i) \qquad (6)$$

where k is the number of peers to which each peer maintains a link.

Proof:

Let us take an example of Chord [8] a variant of hypercubes which constitute a family of graphs defined over 2^k nodes, $k \geq 1$. A Chord node with id $\mathbf{x} \in [0, 1)$ maintains connections to nodes with id ($\mathbf{x} + 1/2$, $\mathbf{x} + 1/4$, . . . , $\mathbf{x} + 1/2^k$).

We wish to compute the average search distance to an object, say i, from the nodes in the network. The number of nodes that can reach the object in 1-hop = k (since the k nodes at $\mathbf{x} - 1/2$, $\mathbf{x} - 1/4$, . . . , $\mathbf{x} - 1/2^k$ maintain a direct link to this node). The number of nodes that can reach key i in 2-hops via these k nodes is k^2. Similarly, the number of nodes that reach the object in h-hops is k^h.

Let us now incorporate the effect of replication in this setting and assume that there are n_i replicas of the object that are replicated on the nodes adjacent (on the unit perimeter circle) to the node responsible for object i. In the Chord setting, each of these replicas will have $k-1$ different nodes in 1-hop distance. Neglecting the repeat nodes[2] and approximating $k-1$ by k, we can say that the M nodes are reached in H hops where H is s.t.

$$M = n_i[1 + k + k^2 + k^3 + \ldots + k^H] = n_i \frac{k^{H+1} - 1}{k - 1} \qquad (7)$$

The average number of hops to object i, $\tau_i(n_i)$, is

$$\tau_i(n_i) = n_i[(1)k + (2)k^2 + (3)k^3 + \ldots + (H)k^H]/M$$

[1] Consider the id space to be a unit parameter circle i.e. if there are 2^h nodes which implies that the id is an h-bit value H, then $H/2^h \in I$ where $I = [0, 1)$. The node adjacent to a node with id $x \in I$ is the node with id $x + 1/2^h$.

[2] If x_i is the id of the node managing object i, the k nodes $x_i - 1/2$, $x_i - 1/4$, . . . , $x_i - 1/2^k$ have direct link to x_i while nodes $(x_i - 1/2^k) - 1/2$, $(x_i - 1/2^k) - 1/4$, . . . , $(x_i - 1/2^k) - 1/2^k$ will have a direct link to $(x_i - 1/2^k)$, the node adjacent to x_i. Of the listed nodes with direct links, $x_i - 1/2^k$ and $(x_i - 1/2^k) - 1/2^k$ are repeated i.e. we have k-1 nodes (that count) at the 1-hop distance. For large M and small n_i, we can ignore this diminishing returns effect.

as kn_i nodes reach a node that has a replica of object i in 1-hop, $k^2 n_i$ nodes will reach a node that has a replica of object i in 2-hops and, in general, $k^h n_i$ nodes reach a node that has a replica of object i in h-hops. Using $\sum_{i=1}^{H} ik^{i-1} = \frac{\partial}{\partial k}\sum_{i=1}^{H} k^i$, we get

$$\tau_i(n_i) = \frac{n_i k}{M}\left[\frac{\partial}{\partial k}\frac{k^{H+1}-1}{k-1}\right] = \frac{n_i k}{M}\left[\frac{(H+1)k^H}{k-1} - \frac{k^{H+1}-1}{(k-1)^2}\right] = \left[\frac{n_i}{M}\frac{(H+1)k^{H+1}}{k-1} - \frac{n_i k}{M}\frac{M/n_i}{k-1}\right]$$

$$= \left[\frac{n_i(H+1)}{M}\frac{k^{H+1}-1+1}{k-1} - \frac{k}{k-1}\right] = \left[\frac{n_i(H+1)}{M}\frac{M}{n_i} + \frac{n_i}{M}\frac{H+1}{k-1} - \frac{k}{k-1}\right] = \left[H+1+\frac{n_i}{M}\frac{H+1}{k-1} - \frac{k}{k-1}\right]$$

For $M\to\infty$ ($\Rightarrow k \sim k-1$) and $n_i/M << 1$, we get

$$\tau_i(n_i) \sim H \tag{8}$$

Further, using $k \sim k-1$ for $M\to\infty$ in (7), we get $\frac{k^{H+1}-1}{k-1} \sim k^H$ i.e. $M = n_i k^H$. Therefore, for $M\to\infty$, we get

$$H \sim \log_k(M/n_i) \tag{9}$$

Substituting (9) in (8), we get $H = \log_k(M/n_i)$. ■

3.3 Optimal Replication

We note that the relation of the average search distance to the number of replicas in (6) is same as (1). The following theorem is analogous to Theorem 2 and Theorem 3.

Theorem 5: In a peer-to-peer network of M nodes, where $M=2^k$ and $M\to\infty$, that employs a DHT which uses prefix-routing for search, when each node has a storage capacity of K objects, in a system storing N unique equal-size[3] objects, the average search distance τ is minimized when the number of replicas, n_i, of object i is piece-wise linear with respect to the object request probabilities $\{p_i\}$, (i=1,2, …, N) i.e.

$$n_i = \begin{cases} p_i KM & \text{if } \frac{1}{KM} \le p_i \le \frac{1}{K} \ \forall i, \quad (10) \\ Max(1, Min(\gamma p_i, M)) & \\ \quad \text{where } \gamma \text{ is s.t. } \sum_{i=1}^{N} n_i = KM & \text{in the general case.} \quad (11) \end{cases}$$

Proof:

The average search distance τ is given by

$$\tau = \sum_{i=1}^{N} p_i \ \tau_i(n_i) \tag{12}$$

where $\tau_i(n_i)$ is as shown in (6).

Constrained optimization of τ w.r.t. n_i, the number of replicas of object i, (for all i) subject to the total storage space constraint

$$\sum_{i=1}^{N} n_i \le KM \tag{13}$$

[3] Since replication of the pointers to the objects is sufficient, we assume each object has the same size. Further, we consider the peers to be participating as "equals" and, therefore, assume that they contribute the same storage capacity.

and $2N$ other inequality constraints[4] –

$$n_i \le M \quad \text{for all } i = 1 \text{ to } N \tag{14}$$

$$n_i \ge 1 \quad \text{for all } i = 1 \text{ to } N \tag{15}$$

provides the (10) and (11) as shown below.

The classical approach to solving constrained optimization problems is the method of Lagrange multipliers. First we will show the result for part *(i)*. We will ignore the constraints specified by Eqs. (14) and (15) for now and instead show that our optimal solution satisfies these constraints under conditions on λ_i specified in part *(i)*.

The Lagrangian of our constrained optimization problem is:

$$H = \sum_{i=1}^{N} p_i \, \tau_i(n_i) + \gamma(\sum_{i=1}^{N} n_i - K M)$$

Minimizing H over all n_i for $i = 1$ to N :

$$\frac{\partial H}{\partial n_i} = p_i \frac{\partial \tau_i}{\partial n_i} + \gamma = 0 \qquad i = 1 \text{ to } N \tag{16}$$

Using $\tau_i(n_i) = log_k(M/n_i) = -\beta \, ln(n_i) + c$,

$$\frac{\partial \tau_i}{\partial n_i} = -\frac{\beta}{n_i}$$

Substituting this in (16), we obtain,

$$n_i = p_i \frac{\beta}{\gamma}$$

Applying the constraint $\sum_{i=1}^{N} n_i = KM$ to remove the unknown constant β/γ, we obtain the optimum number of replicas as:

$$n_i = p_i KM \qquad i = 1 \text{ to } N$$

The constraints in (14), (15) are clearly satisfied if $\frac{1}{KM} \le p_i \le \frac{1}{K} \; \forall i$. This proves part *(i)* of the theorem.

Without this condition on p_i, one can rewrite the problem as a maximization problem using a modified Lagrangian. Using $\tau_i(n_i) = -\beta \, ln(n_i) + c$, and including both the constraints in (14), (15) and rewriting the $n_i \ge 1$ constraint as $-n_i \le -1$, the modified Lagrangian is:

$$G = \beta [\sum_{i=1}^{N} p_i \, ln(n_i)] - \gamma_0 [\sum_{i=1}^{N} n_i - KM] - \sum_{i=1}^{N} \gamma_i \, (n_i - M) - \sum_{i=1}^{N} \alpha_i \, (-n_i + 1)$$

The Kuhn Tucker Conditions for the modified Lagrangian are:

$$\frac{p_i \beta}{n_i} - \gamma_i - \gamma_0 + \alpha_i = 0 \quad \text{for } i = 1 \text{ to } N \tag{17}$$

$$\sum_{i=1}^{N} n_i \le K M, \quad \gamma_0 \ge 0, \quad \text{and} \quad \gamma_0 [\sum_{i=1}^{N} n_i - K M] = 0 \tag{18a}$$

$$n_i \le M, \quad \gamma_i \ge 0, \quad \text{and} \quad \gamma_i (n_i - M) = 0 \quad \text{for } i = 1 \text{ to } N \tag{18b}$$

$$-n_i \le -1, \quad \alpha_i \ge 0, \quad \text{and} \quad \alpha_i (-n_i + 1) = 0 \quad \text{for } i = 1 \text{ to } N \tag{18c}$$

[4] Storing multiple replicas of an object at a node has no benefit. Therefore, the number of replicas of any object can be no more than the number of nodes. Since we assume that the average search distance is over the items present in the system, there is at least one replica of each object.

From (17): $n_i = \dfrac{p_i \beta}{\gamma_i + \gamma_0 - \alpha_i}$

(18b), (18c) imply that: either $\gamma_i = 0$ or $n_i = M$, and, either $\alpha_i = 0$ or $n_i = 1$, respectively. Therefore, the optimum solution is:

$$n_i = Max(1, Min(\gamma p_i, M))$$

where, from (18a), γ is such that $\sum_{i=1}^{N} n_i = KM$ when the storage size is not large enough to store all the files (i.e. $N \geq K$). This proves part *(ii)* of the theorem. ■

Substituting (10) in (6), (12), we get the following theorem.

Theorem 6: In a peer-to-peer network of M nodes, where $M=2^k$ and $M \to \infty$, that employs a DHT which uses prefix-routing for search, when each node has a storage capacity of K objects, in a system storing N unique equal-size[5] objects, the minimized average search distance is given by

$$\tau_{opt} = -\sum_{i=1}^{N} p_i \log_k p_i - \log_k K \qquad (19)$$

if $\dfrac{1}{KM} \leq p_i \leq \dfrac{1}{K}$ $\forall i$, where p_i is the request probability for object i for $i = 1,2, \ldots, N$. ■

4. Entropy

We notice from (5) that the first term in the minimum average hop distance expression is the entropy in the object request probabilities across the entire network. We attempt to explain why the optimum average search distance is directly related to the entropy in object request probabilities.

Viewing the network as an information source, the entropy term captures the uncertainty regarding which object will be requested next. In the coding problem [13] where the optimal (i.e., minimal) code word length is equal to the entropy of the source symbol, the entropy term captures the uncertainty in the symbol that is generated next. Thus, a mapping of our search problem to the coding problem may provide us the sought explanation.

In Section 3.2, we indicated that, in a DHT that uses prefix routing, one (approximate) way to view the effect of replication of an object is that replicating an object at a certain prefix level reduces the number of bits in the "effective address" of the object (e.g. the effective address of an object replicated at all nodes with prefix '01' is '01') and fewer bits in the address implies that the object can be reached in fewer hops (when links to each resolution are available in O(1) hops). In this simplified view, the search distance to an object is equal to the length of that object's effective address. This is analogous to the length of the code word for a symbol of an information source (since each node in our system stores only one object, the search addresses satisfy the prefix condition). This analogy to the coding problem explains the entropy term. The second term in (5) captures the effect of the storage size (larger storage space allows more replication per object which decreases the average number of hops required to reach the object). However, the interaction of storage size with the aforementioned coding analogy is not obvious. We formalize the coding analogy and the effect of per-node storage size next.

Consider a network of M nodes. Regardless of how the nodes are connected, we can construct a breadth-first search tree for each node. The breadth-first search tree from node A has: node A at the root of the tree; the nodes at the next level are the nodes that node A can reach in 1 hop; the nodes at a depth of 2 are the nodes that can be reached from the nodes at a depth of 1 but were not included in the tree yet, and so on. A search for an object from a node returns the path (from the root) to the nearest replica in the breadth-first search tree for that node. In a search overlay network with a topology exhibiting exponential expansion, each breadth-first search tree has

[5] Since replication of the pointers to the objects is sufficient, we assume each object has the same size. Further, we consider the peers to be participating as "equals" and, therefore, assume that they contribute the same storage capacity.

approximately d new neighbors at each internal node of the tree (i.e. a constant branching factor of d). Thus, the path from a node to an object on the node's BFS tree gives us a mapping ("code") from an object request by a node to a string on elements $\{0,\ldots,d-1\}$. We use this mapping to show our key result as summarized in the following theorem.

Theorem 7: In a network of M nodes which are interconnected such that the breadth-first search tree from each node has a branching factor[6] of d, where each node has a storage capacity to store K equal-sized objects, where each node requests objects from a set of N equal-sized objects with request probabilities $\{p_i\}$ $i = 1,2, \ldots, N$, then the minimum average search distance τ_{opt} is given by

$$\tau_{\text{opt}} = -\sum_{i=1}^{N} p_i \log_d p_i - \log_d K \qquad (20)$$

***Proof**:*

Consider a sequence of ML requests ordered by the node id of the requesting node[7]; we refer to this as the object request sequence. Each *typical* object request sequence of length ML, in the limit of large L, will have $Lp_1, Lp_2,\ldots, Lp_N$ requests for object 1, 2, …, N respectively by each of the M nodes[8]. The number of distinct object request sequences of length ML where each node is equally-likely to make a request for object 1, 2, …, N with probability $p_1, p_2,\ldots, p_N$ respectively, is equal to $\left(\frac{L!}{Lp_1!Lp_2!\ldots Lp_N!}\right)^M$.

Let us first show (20) for $K = 1$, i.e. assume that each node has the storage space to hold only 1 file.

If we knew ahead of time which node will make which request, we could place the appropriate files at each node (note that this results in replicating files in proportion to their probability of request) and the search distance would be 0. However, we do not know when and which objects will be requested by which nodes (this is the uncertainty captured by the entropy term in (20)) and, hence, the search distance is non-zero.

A search path for an object from a node is a sequence of branches taken at each intermediate node on the breadth-first search tree (e.g. if $d=2$, if the path to an object involves taking branch 0, then branch 1 and then branch 0, the search path is 010 and has length 3). Thus, a search path is a string in the alphabet $\{0, \ldots, d-1\}$ of length equal to the number of hops taken to reach the file.

Replacing the object id in the object request sequence by the search path to the object (from whichever node requested it), we obtain a new sequence whose elements are in the alphabet $\{0, \ldots, d-1\}$. The average search length of an object is equal to the length of this new sequence divided by the number of requests made. Let us call this new sequence the *code* for the requested sequence. For a given placement of objects, two distinct object request sequences must give two different codes (since each node can store only one object, if a node requests a different object, the search path for this new object has to be different and, hence, the code will be different).

---

[6] The number of nodes covered increases exponentially with each additional hop from all nodes in the network.

[7] That is, group together all the requests made by a given node while maintaining the original sequence of requests made by that node.

[8] We are applying the strong law of large numbers here in the same way as in Shannon's original work (Section 7 in [13]) i.e. the total probability of set of object request sequences which do not have $Lp_1, Lp_2,\ldots, Lp_N$ requests for object 1, 2, …, N from each of the M nodes, is negligible as $ML\rightarrow\infty$ (to be more precise, given any $\epsilon > 0$, we can find an L_0 such that for $L > L_0$, the total probability of this set is $< \epsilon$) .

To encode the $\left(\frac{L!}{Lp_1!Lp_2!...Lp_N!}\right)^M$ possible sequences of LM requests (each of which is equally likely), the minimum code length has to be $log_d\left(\frac{L!}{Lp_1!Lp_2!...Lp_N!}\right)^M$. Therefore[9], the optimum average search length for these ML requests is $\tau_{opt} = \frac{1}{ML}\log_d\left(\frac{L!}{Lp_1!Lp_2!...Lp_N!}\right)^M = \frac{1}{L}\log_d\frac{L!}{Lp_1!Lp_2!...Lp_N!}$. From Stirling's approximation, $n! \approx \sqrt{2\pi}\, n^{n+\frac{1}{2}}e^{-n}$, or, $\ln n! \approx \ln(\sqrt{2\pi}\, n^{n+\frac{1}{2}}e^{-n}) = \left(n+\frac{1}{2}\right)\ln n - n + \frac{1}{2}\ln 2\pi$. Therefore, for large n, we get $\ln n! \approx n\ln n - n$. Substituting the approximate values for $L!$ and $Lp_i!$ for $i=1, 2, \ldots, N$, we get

$$\begin{aligned}
\tau_{opt} &= \frac{1}{L}\log_d\frac{L!}{Lp_1!Lp_2!...Lp_N!} \\
&\approx \frac{1}{L}\left[L\log_d L - L - \sum_{i=1}^{N}\left(Lp_i\log_d Lp_i - Lp_i\right)\right] \\
&= \frac{1}{L}\left[L\log_d L - L\sum_{i=1}^{N}p_i\log_d L - L\sum_{i=1}^{N}p_i\log_d p_i - L + L\sum_{i=1}^{N}p_i\right] \\
&= \frac{1}{L}\left[L\log_d L - L\log_d L\sum_{i=1}^{N}p_i - L\sum_{i=1}^{N}p_i\log_d p_i - L + L\right] \\
&= \frac{1}{L}\left[L\log_d L - L\log_d L - L\sum_{i=1}^{N}p_i\log_d p_i\right] \\
&= -\sum_{i=1}^{N}p_i\log_d p_i
\end{aligned}$$

Let us now consider the case when each node can store K objects. With K distinct objects at all nodes, of the $\frac{L!}{Lp_1!Lp_2!...Lp_N!}$ possible object request sequences for this node, only $\frac{1}{K^L}\frac{L!}{Lp_1!Lp_2!...Lp_N!}$ will result in distinct search paths[10] (in the best case i.e. when K is small compared to N and no single p_i is very large[11]). Since, $\frac{1}{K^L}$ fewer codes need to be generated per node, the minimum code length necessary is reduced by $log_d\frac{1}{K^L}$ (over L requests) and, hence, the average search length is reduced by (at most) $-log_dK$. ■

Since the above theorem holds for all topologies with exponential expansion independent of how objects are stored in the network, it provides a proof of (19) (Section 3.3, Theorem 5) for DHTs that use prefix-routing independent of any requirement on object replication. Theorem 7 also explains the result in Theorem 3 (Section 2) for unstructured networks which we obtained in [9].

We can also rederive the proportional replication result expressed by (3) and (10) in Theorems 2 and 5 respectively (which we had proved in [9]) directly from (20) as follows. Since the average search distance τ is

[9] The average codeword length with our method of coding an object request sequence (concatenating the search path for each request) is lower-bounded by the average codeword length for an optimal code in the alphabet {0, …, d–1} for these sequences.

[10] K distinct request sequences (K different file requests by node 1 and a fixed set of requests at the other M–1 nodes) with appropriate file placement will have a search length of 0 when K files can be stored at node 1.

[11] For example, recall that (10) holds only if $\frac{1}{KM} \le p_i \le \frac{1}{K}\ \forall i$.

given by $\tau = \sum_{i=1}^{N} p_i \tau_i(n_i)$, if $\tau_i(n_i)$ is of the form $log_d(M/n_i)$ (as is the case for Theorems 2 and 5), rewriting (20) as $\tau_{opt} = -\sum_{i=1}^{N} p_i \log_d p_i K$, we can see that the assignment of n_i which minimizes τ is is such that $M/n_i = 1/p_i K$ or $n_i = p_i KM$, thus establishing (3) and (10).

While Theorem 7 gives a lower bound on the average search distance in peer-to-peer networks, Theorem 3 and 5 suggest that the lower bound can be achieved. However, we note that Theorem 3 is based on Theorem 1 and Theorem 1 was proven in [9] under the assumption that the flooding search tree grows exponentially with d "new" nodes per internal node on the tree. We know that after a certain number of hops, many nodes reached are repeats and, hence, the expression in (20) is a lower bound that is achieved only in the limiting case of $M \to \infty$ in unstructured networks. The issue of node repetition is present in the DHT case also (Section 3.2, Footnote 2) and, hence, the expression in (20) is a lower bound achieved only in the limiting case of $M \to \infty$ for DHTs as well.

Finally, we note that we have assumed the request probabilities to be uniform across the network in this paper (as assumed in [9]). Intuitively, we know that topological and geographic clustering in object popularities offers the opportunity to improve the search distance. Clustering in object popularity changes the entropy regarding which file is requested next as well as the number of possible request sequences discussed in the proof for Theorem 7. But, as [12] shows, the optimum average search distance is still related to the entropy in which an object is requested next in the same way. We can see this as follows.

Let p_{ij} be the probability that node j requests object i ($\sum_{i=1}^{N} p_{ij} = 1$). Assuming that each node still requests objects at the same rate, each *typical* request sequence of length ML, in the limit of large L, will have[12] L requests from each of the M nodes. The number of distinct object request sequences of length ML where 1, 2, …, N appear with probabilities $p_{1j}, p_{2j}, \ldots, p_{Nj}$ (j=1,2,…,M) is equal to $\prod_{j=1}^{M} \frac{L!}{Lp_{1j}!Lp_{2j}!\ldots Lp_{Nj}!}$. Thus, for K=1, the minimum number of distinct codes needed = $\log_d \prod_{j=1}^{M} \frac{L!}{Lp_{1j}!Lp_{2j}!\ldots Lp_{Nj}!} = \sum_{j=1}^{M} \log_d \frac{L!}{Lp_{1j}!Lp_{2j}!\ldots Lp_{Nj}!}$. Therefore, for K=1, the minimum average search length = $\frac{1}{ML}\sum_{j=1}^{M} \log_d \frac{L!}{Lp_{1j}!Lp_{2j}!\ldots Lp_{Nj}!} = -\frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{N} p_{ij} \log_d p_{ij}$ which is same as that shown in [12].

Finally, we can extend the argument to when nodes make requests at different rates. The number of distinct object request sequences of length ML where 1, 2, …, N appear with probabilities $p_{1j}, p_{2j}, \ldots, p_{Nj}$ (j=1,2,…,M) and node j makes requests with probability $q_j = \prod_{j=1}^{M} \frac{MLq_j!}{MLq_j p_{1j}!MLq_j p_{2j}!\ldots MLq_j p_{Nj}!}$. Thus, the minimum average search length $\tau_{opt} = \frac{1}{ML}\log_d \prod_{j=1}^{M} \frac{MLq_j!}{MLq_j p_{1j}!MLq_j p_{2j}!\ldots MLq_j p_{Nj}!} = \frac{1}{ML}\sum_{j=1}^{M} \log_d \frac{MLq_j!}{MLq_j p_{1j}!MLq_j p_{2j}!\ldots MLq_j p_{Nj}!}$ for K=1. Applying Stirling's approximation

$$\tau_{opt} = \frac{1}{ML}\sum_{j=1}^{M} \log_d \frac{MLq_j!}{MLq_j p_{1j}!MLq_j p_{2j}!\ldots MLq_j p_{Nj}!}$$

$$= \frac{1}{ML}\sum_{j=1}^{M}\left[\log_d MLq_j! - \sum_{i=1}^{N} \log_d MLq_j p_{ij}!\right]$$

[12] in the sense of Footnote 8.

$$\approx \frac{1}{ML}\sum_{j=1}^{M}\left[MLq_j \log_d MLq_j - MLq_j - \sum_{i=1}^{N} MLq_j p_{ij} \log_d MLq_j p_{ij} - \sum_{i=1}^{N} MLq_j p_{ij} \right]$$

$$= \frac{1}{ML}\sum_{j=1}^{M}\left[MLq_j \log_d ML + MLq_j \log_d q_j - \sum_{i=1}^{N} MLq_j p_{ij} \log_d ML - \sum_{i=1}^{N} MLq_j p_{ij} \log_d q_j p_{ij} \right]$$

$$= \sum_{j=1}^{M}\left[q_j \log_d q_j - \sum_{i=1}^{N} q_j p_{1j} \log_d q_j p_{ij} \right] = \sum_{j=1}^{M}\left[q_j \log_d q_j - \sum_{i=1}^{N} q_j p_{ij} \log_d q_j - \sum_{i=1}^{N} q_j p_{ij} \log_d p_{ij} \right]$$

$$= -\sum_{j=1}^{M} q_j \sum_{i=1}^{N} p_{ij} \log_d p_{ij}$$

As before, a larger cache size will contribute a $-\log_d K$ term. Thus, for the completely generalized case, we get the following theorem.

Theorem 8: In a network of M nodes which are interconnected such that the breadth-first search tree from each node has a branching factor[13] of d, where each node has a storage capacity to store K equal-sized objects and where each node makes a request with probability q_j, j = 1,2, …, M, for objects from a set of N equal-sized objects with request probabilities $\{p_{ij}\}$ i = 1,2, …, N, then the minimum average search distance τ_{opt} is given by

$$\tau_{opt} = -\sum_{j=1}^{M} q_j \sum_{i=1}^{N} p_{ij} \log_d p_{ij} - log_d K \qquad \blacksquare$$

5. Related Work

As already stated, our results for unstructured networks in Section 1 are from [9]. While [9] assumed the request rates to be the same across the network, [12] showed that even when the object request probabilities exhibit clustering, the minimum average search distance is equal to the entropy regarding which file is requested next minus the logarithm of the per-node cache size. Our results in Section 4 explain this relation between the average search distance and the entropy.

The replication method for DHTs that we used in Section 3 is a generalization of the proposal in [5] where the number of replicas was in terms of 2^i (to decrease the search distance by i) and, due to this restriction on the allowed number of replicas, the object request probability distribution was restricted to be power-law. In this paper, we allow the number of replicas of each object and the object request probability distribution to be arbitrary and show that replication of an object at nodes adjacent (in the id space) to the node responsible for the object reduces the average search distance by the logarithm of the number of replicas of the object. Another difference between our work and [5] is in the optimization objective. They wanted to use the least amount of storage to achieve a target average search distance while our goal was to allocate the available space efficiently (for which we showed that the number of replicas of each object should be proportional to the request probability of that object). Our optimization result strengthens the proposal in [1] where replicas are created on user requests (a model similar to that discussed in [11]) and the pointers to these replicas are populated adjacent to the node responsible for the object; we have shown in [11] that standard caching methods like LRU give near proportional replication which implies that the proposal in [1] has near-optimal search performance.

Our result on the optimality of proportional replication for search distance in structured networks adds to the download performance, network resource usage and stability benefits of proportional replication shown in [10], [11] and [9] respectively.

Our result on the relation between the minimum average search distance and the entropy regarding which object is requested next provides an alternate explanation in the field of search algorithms for the average number of steps to find a key (i.e. the average depth to which one has to go to find a key in the search tree) in

[13] The number of nodes covered increases exponentially with each additional hop from all nodes in the network.

the optimal binary balanced search tree [15]. While [15] used sub-tree balancing properties to show that the average number of steps to find a key in the optimal binary balanced search tree is related to the entropy in key frequencies, we can see this relation directly from our information-theoretic argument in Theorem 7.

6. Conclusions and Future Work

In this paper we showed that the average search distance (in number of overlay hops) in DHT-based peer-to-peer systems is logarithmically related to the number of replicas of the object for overlay topologies with exponential expansion if the replicas are placed in nodes adjacent to the node responsible for that object. We used this result to show that given a certain per-node storage constraint, the average search distance is minimized when each object is replicated in proportion to its request probabilities. This result complements the known benefits we showed earlier for proportional replication in download performance and network resource usage; the relative ease with which this replication can be achieved further demonstrates that peer-to-peer networks have a self-scaling property. We also show that the minimum average search distance is related to the entropy in object request probabilities. Finally, we generalized this result to any search tree with exponential expansion by showing the same result using information-theoretic arguments.

Bibliography:

[1] Michael J. Freedman, Eric Freudenthal, and David Mazières. "Democratizing Content Publication with Coral." *NSDI 2004,* San Francisco, CA, March 2004.

[2] J. Li, J. Stribling, R. Morris and M. F. Kaashoek. "Bandwidth-efficient management of DHT routing tables." *NSDI 2005,* Boston, MA, 2005.

[3] G. S. Manku, M. Bawa and P. Raghavan. "Symphony: Distributed Hashing in a Small World." *USENIX USITS 2003*, March 2003.

[4] P. Maymounkov and D. Maziéres. "Kademlia: A Peer-to-peer Information System Based on the XOR Metric." *IPTPS 2002*, Cambridge MA, Mar 2002.

[5] V. Ramasubramanian and E. G. Sirer, “Beehive: Exploiting power law query distributions for O(1) lookup performance in peer-to-peer overlays,” *NSDI 2004*, San Francisco, CA, March 2004

[6] S. Ratnasamy, P. Francis, M. Hadley, R. Karp, and S. Shenker. "A Scalable Content-Addressable Network." *ACM SIGCOMM 2001*, San Diego CA, Aug 2001.

[7] A. Rowstorn and P. Druschel. "Pastry: Scalable, Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems." *IFIP/ACM Middleware 2001*, Heidelberg, Germany, Nov 2001.

[8] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications." *ACM SIGCOMM 2001*, San Diego CA, Aug 2001.

[9] S. Tewari and L. Kleinrock. "Analysis of Search and Replication in Unstructured Peer-to-Peer Networks," UCLA Computer Science Dept Technical Report UCLA-CSD-TR050006, March 2005.

[10] S. Tewari and L. Kleinrock. "On Fairness, Optimal Download Performance and Proportional Replication in Peer-to-Peer Networks," *IFIP Networking 2005*, Waterloo, Canada, May 2005.

[11] S. Tewari and L. Kleinrock. "Proportional Replication in Peer-to-Peer Networks," (to appear) *IEEE INFOCOM 2006*, Barcelona, Spain, April 2006.

[12] S. Tewari and L. Kleinrock. "Optimal Search Performance in Unstructured Peer-to-Peer Networks With Clustered Demands," UCLA Computer Science Dept Technical Report UCLA-CSD-TR050040, Sept 2005.

[13] W. Weaver and C. E. Shannon. *The Mathematical Theory of Communication*, Urbana, Illinois: University of Illinois Press, 1949.

[14] B. Zhao, L. Huang, J. Stribling, S. Rhea, A. Joseph, and J. Kubiatowicz. "Tapestry: A Resilient Global-scale Overlay for Service Deployment." *IEEE Journal on Selected Areas in Communications, JSAC*, 2003.

[15] Knuth, D. E., The Art of Computer Programming, Vol. 3: Sorting and Searching, Reading, Massachusetts: Addison-Wesley, 1998.

[16] Liben-Nowell, D., Balakrishnan, H, Karger, D., "Analysis of the evolution of peer-to-peer systems," in Proc. of PODC, July 2002.