A Performance Model for Peer to Peer File Sharing Services

Krishna Kant and Ravi Iyer
Enterprise Architecture Lab, Intel Corporation, OR

November 16, 2001

Abstract

In this paper we introduce a random-graph based model for studying the evolution of ad hoc peer-to-peer (P2P) communities such as in Gnutella or Freenet. Our model incorporates some essential aspects of the P2P environment such as different classes of nodes, variable number of neighboring nodes and hop-count limited document search. We study basic properties such as reachability from a given node in the network using an analytical approach and complex properties such as queuing behavior using a simulation based approach. Although the current model targets a simple P2P file sharing environment, it lays the foundation for detailed studies of P2P application behavior. The paper also briefly discusses a number of performance issues that could be studied via the extended model.

Keywords: peer-to-peer computing, random graphs, performance modeling, probability generating function.

Approximate word count = 8400

1 Introduction

Past one year has seen an explosive growth in the use of file-sharing software in order to exchange digital audio, video and other types of files. The trend was started by Napster (`www.napster.com`), which allows sharing of MP3 music files among an arbitrary set of users. Napster uses a centralized server to maintain meta information, i.e., which machine (or "agent") has what files and thereby assists in locating the desired files. The actual transfer of files is, however, done directly between the agents. Napster quickly led to numerous variants of file-sharing software including wrapster (which allows sharing of files of any type by putting a MP3 type of wrapper around them) and Pointerra (www.pointerra.com) (which provides general file sharing by means of a modified browser interface). Napster also led to fully distributed versions of file-sharing where there is no central entity and both file-location and transfers are handled in a completely distributed manner. Gnutella (`gnutella.wego.com`) and Freenet [2] represent this lineage. Freenet supplants the basic distributed file-sharing capability with secure exchange and anonymity. It also includes a mechanism for automatically migrating files closer to request points. These services themselves have led to a number of variants with various ways of maintaining and locating the shared information. For example, see `gnucleus` and `freekey` under (`sourceforge.net/projects`). Reference [4] discusses many peer to peer applications and introduces a taxonomy for them.

These recent developments have sparked a new interest and flurry of activity in the so-called peer-to-peer (P2P) computing space, including its application to improve the current web infrastructure. The goal in this research is to analyze performance issues for the evolving peer-to-peer approach to file sharing. Our primary contribution in this paper is a random-graph based model that formulates a procedure for ad hoc network evolution. We apply this model for the construction of a Gnutella-like network of peers and study its reachability properties (number of nodes of a certain type reached in a fixed number of hops from a given node), the amount of traffic processed and queuing behavior for each node. We study the basic aspects such as reachability and amount of traffic using an analytical approach. For more detailed studies on the throughput, utilization and queueing behavior at each node in the network, we use a simulation based approach. Although our modeling primarily concentrates on rather simple file sharing networks, it can be extended to more complex scenarios as well.

The rest of this paper is organized as follows. In Section 2 we note some salient characteristics of file sharing networks and present a graph model that captures the build-out of peer-to-peer communities. In Section 3, we present an analytical model to characterize the basic topological properties of the graph model and the average traffic arriving at each node. In section 4, we present a preliminary simulation model to explore response time characteristics for the graph model. Section 5 presents potential extensions to the model and areas for further research on the topic.

2 Performance Modeling of Peer-to-Peer File Sharing

The most important aspects in peer-to-peer computing performance are (a) characterization of how a community of users is established, (b) how the requests for documents are handled, & (c) how the requested files are sent. Here, we discuss these aspects.

We are primarily interested in the performance of a file-sharing network assuming that it has grown to a certain size. Thus, although the mechanism of community establishment is not directly relevant, it provides a convenient way to characterize the topology of the eventual network. We assume that a node first connects to a few known nodes, each of which sends it a list of other nodes that they are aware of. Based on this list, the node can choose to connect to a few additional nodes. This is very similar to the approach taken by Gnutella. In particular, a Gnutella node maintains open TCP/IP connections with all these nodes. Freenet also uses a similar approach.

Because of lack of global knowledge in a peer-to-peer environment, requests for documents invariably result in a search through the network. However, there are many ways of directing the search from one node to the next. In the simplest case (assumed in this paper), the search progresses to all the unvisited nearest neighbors up to a fixed number of "hops". We also assume that each message is stamped with a globally unique ID to prevent duplicate responses for the same request. Gnutella uses this approach, where the searches are based on partial/full file-name matching in the shared folders at each node. A more sophisticated approach is for each node to glean information about the documents stored at other nodes, either by snooping the responses to the document requests or by a separate protocol (e.g., each node periodically exchanges information with its neighbors about what documents it has). Freenet uses the former method. It shares storage, rather than files; files and meta-information about the files are placed in the storage as a result of searches. In particular, each node in the reverse path caches the document (using a LRU scheme) in order to satisfy future requests for this object more quickly. This scheme also has the advantage of dynamically shifting documents closer to the place of demand. The search itself could be based on "keys" rather than document names, and the relationship between keys and names could be arbitrarily complex. When a node finds a matching document in its local storage, it could either return the matched document itself, or simply some meta information (or "handle") for the document. The path along which the response is returned is also important since it dictates where the returned information can be cached without using a separate update protocol. In this paper, we assume that the responses follow the same path as the requests, which is the scheme used in Freenet.

Along with the above simplifying assumptions, we also assume a rather idealized case of a corporate intranet type of environment where all nodes sit on a high-speed network, all peer nodes are always turned on and connected to the network, and there are no significant down-time issues to be considered for performance modeling purposes. Furthermore, in order to concentrate on nodal behavior, we simply absorb network delays into nodal delays (this

amounts to assuming that there is no significant queuing inside the network).

We now describe a simple graph model for peer to peer interactions. The model consists of two types of nodes, henceforth labeled as "distinguished" and "undistinguished". The distinguished nodes can be thought of either as the traditional servers or as a set of globally known nodes which act as the nuclei of the file sharing network. We further assume that only the undistinguished nodes generate any requests. We view the network as a random graph that is built incrementally using the following procedure.

Initially, the network consists of a connected graph of N_d distinguished nodes. We assume a regular connectivity, with each node connected to $M > 0$ other nodes. The build-up occurs with undistinguished nodes joining the network sequentially. Each joining node, say X, connects to a total of $\mathcal{K}$ nodes, where $\mathcal{K}$ is a random variable with the mass function $\theta_k, k = 1..K$. Each connection attempt from joining node X is directed to a undistinguished node with probability q_u, and to a distinguished node with probability $(1 - q_u)$. Care is taken so that X does not connect to any given node more than once. While the network contains less than K undistinguished nodes, any excess connections are directed to distinguished nodes. Given that the total number of nodes in the graph, henceforth denoted as N_t, is much larger than K, this initial perturbation has very little impact on the overall network.

One important property of such a graph construction is nonuniformity: distinguished nodes and the nodes that enter the graph early on have, on the average, higher connectivity than others. Moreover, the connectivity distribution has a heavy tail (unlike the exponentially decaying tail in a classical random graph). This property is consistent with experiences with real peer-to-peer network [7]. It is also consistent with a nature of web viewed as a graph [6]. It is possible to achieve a more general tail decay rate by making the connection probability dependent on the recency of the node, but for simplicity this extension is not considered here.

In order to study the performance issues stated above, we need to characterize the following performance metrics.

1. Overall and *undistinguished degree* of a node at each level. The latter is needed since we assume that only undistinguished nodes generate requests.
2. Number of distinct nodes (overall and undistinguished) reachable from any given node in a given number of hops.
3. Given some request generation process at each node, the total request traffic arriving at each node.
4. Given some assumptions about response generation (e.g., response for every request, response only if the requested content exists, etc.), total response traffic passing through each node.
5. Queuing delays experienced by requests and responses at each node.

6. Response time (i.e., time between request transmission and response reception) from every node for requests issued by each node.

We attempt to do this initially with an analytical model in the next section. For obvious reasons, the last 2 issues are analytically intractable in most cases, and are best handled via a simulation model. We are developing a simulation environment to evaluate performance aspects of a peer-to-peer environment from various angles: basic issues such as complex arrival patterns, nodal service times and finite queue lengths as well as complex issues such as designing request-response protocols node interactions, caching mechanisms and load distribution techniques. For lack of space, this paper will only cover the basic simulation model and analyze obtained results.

3 Analytic Model

Let us start with the characterization of nodal degree. It is convenient to introduce the notion of a "level" for each node. Since all distinguished nodes are treated identically, we consider all of them to lie at level 0. Each undistinguished node, however, introduces a new level. Thus, when the network has a total of N_t nodes, it has $N_t - N_d + 1$ levels, numbered $0 \,..\, N_t - N_d$, with all level 0 nodes being distinguished, and others undistinguished. Henceforth, we let $L = N_t - N_d$ as the maximum level number.

Let $P_n(\ell_2, \ell)$ denote the probability that a node that entered the network at level ℓ has a degree n when a level ℓ_2 node enters the network. Note that following the inclusion of level ℓ_2 node, the total number of nodes in the network is $\ell_2 + N_d$. Also, ℓ_2 denotes the number of undistinguished nodes *after* the addition of the new node. Appendix A shows the details of developing a recurrence equation for $P_n(\ell_2, \ell)$, obtaining its probability generating function $\Phi(z|L, \ell)$, and from there computing $D_{at}(\ell|L)$, defined as the mean degree *at* a node that enters the network at level ℓ assuming a total of L levels. The appendix also show how to adapt the analysis to obtain the undistinguished degree of a node that enters the network at a given level ℓ.

Next we consider the computation of the number of nodes reached starting from a given node. For this, we first characterize the reachability matrix $\mathcal{R}_h$ whose (i, j)th element gives the probability of reaching level i from level j in exactly h hops (i.e., via paths with exactly $h - 1$ unique intermediate nodes different from nodes i and j). This reachability matrix, along with the nodal degree characterization, yields the average number of nodes reached in h hops. We henceforth let $G(\ell_2, h)$ denote the average number of nodes reached *at* hth hop, starting from level ℓ_2. Similarly, let $G_u(\ell_2, h)$ denote the number of undistinguished nodes reached in h hops starting from level ℓ_2. Appendix B shows how to compute $\mathcal{R}_h$, $G(\ell_2, h)$, and $G_u(\ell_2, h)$.

Given the above analysis, we are now in a position to estimate the total request and response traffic arriving at a

given node X. This is given by a superposition of the traffic generated by all nodes from which X can be reached in H or fewer hops (including requests generated by the node itself). A precise characterization of the traffic process is intractable; here we only compute its average rate. Let $N_{req}(\ell, H)$ denote the total number of requests reaching *undistinguished nodes* in at most H hops from a level ℓ node. Then,

$$N_{req}(\ell, H) = 1 + \sum_{h=1}^{H} G_u(\ell, h) \tag{1}$$

Turning this around, $N_{req}(\ell, H)$ gives the total request traffic processed by a level ℓ node. Thus, if λ_0 denotes the request rate for each undistinguished node, the total request traffic arriving at a level ℓ node, denoted $\lambda_{req}(\ell)$, is simply $\lambda_0 N_{req}(\ell, H)$.

The response traffic can also be computed similarly. That is, $\lambda_{resp}(\ell)$, the total response traffic arriving at a level ℓ node, is $\lambda_0 N_{resp}(\ell, H)$ where $N_{resp}(\ell, H)$ denotes the total number of responses processed at level ℓ node for a single request. The estimation of $N_{resp}(\ell, H)$ depends on how the responses are generated and forwarded back to the requesting node. We consider the following possibilities in this regard:

1. Every node generates a response (found or not found), and sends it to the requester directly (without going through any intermediate nodes). Then, $N_{resp}(\ell, H) = N_{req}(\ell, H)$.

2. Every node generates a response which travels back along the request path. In this case, a request going to a node that is h hops away, will require response processing at each one of the h intermediate nodes (excluding the response generating node where request processing and response generation are considered a single atomic operation). It follows that:

$$N_{resp}(\ell, H) = \sum_{h=1}^{H} h\, G(\ell, h) \tag{2}$$

Let S_{req} and S_{resp} denote the average service times of requests and responses. Then, the level ℓ node utilization is given by:

$$U(\ell) = \lambda_{req}(\ell) S_{req} + \lambda_{resp}(\ell) S_{resp} \tag{3}$$

As stated earlier, the queuing delays depend on the details of the arrival and service processes and are best determined via a simulation (which can also also issues such as limited queue depths, abandonments and retries). The above equation for utilization is useful for ensuring that the nodal utilization in the simulation model does not exceed a predetermined limit.

Now, we present some experimental results using the analytic model. We choose the parameters in this study as follows:

Table 1: Reachability and traffic for a 100 node network

undist prob	no of hops	tot nodes reached	undist reached	responses per node	tot traf per node
	1	5.9	3.3	4.9	6.1
	2	55.2	44.5	103.6	146.5
0.05	3	99.1	85.8	235.2	320.5
	4	100	90	238.8	328.8
	5	100	90	238.8	328.8
	1	5.9	4.3	4.9	8.4
	2	34.3	23.8	61.7	82.3
0.50	3	91	73.9	231.7	304
	4	99.9	89.4	267.5	356.9
	5	100	89.6	267.7	357.3
	1	5.9	5.3	4.9	10.6
	2	28.6	22.6	50.3	73.6
0.95	3	76.7	63.8	194.6	258.4
	4	98.5	87.4	281.8	369.2
	5	99.7	89.3	287.8	377.2

Table 2: Reachability and traffic for a 500 node network

undist prob	no of hops	tot nodes reached	undist reached	responses per node	tot traf per node
	1	6	3.6	5	6.2
	2	243.7	232.7	480.5	711.5
0.05	3	499.7	488.6	1248.4	1737
	4	500	490	1249.6	1739.6
	1	6	4.7	5	8.5
	2	95.7	84.2	184.3	264.6
0.50	3	483.5	465.1	1347.8	1812.4
	4	500	490	1413.9	1903.9
	1	6	5.8	5	10.7
	2	35.1	29.1	63.2	91.7
0.95	3	163.5	137.1	448.3	582.4
	4	405.7	367.7	1417.2	1782.7

1. The network initially has 10 distinguished nodes, each of which is connected to 4 others.

2. Each newly arriving node connects from 1 to 4 nodes with equal probability. (Thus, the average connectivity is 2.5).

3. We consider two network sizes N_t of 100, and 500 nodes.

4. For each network, we consider 3 undistinguished node connection probabilities q_u of 5%, 50%, and 95%.

Tables 1 and 2 show overall averages on reachability, response traffic per node and total traffic per node for 100 and 500 node graphs respectively. For reachability, the tables list the average number of all nodes reached from an arbitrary node (column 3), and average number of undistinguished nodes reached from an undistinguished node (column 4). The response (and hence total) traffic computation assumes that the responses trace the request path backwards. These results point to a number of interesting conclusions about the network under consideration.

1. Even with a very limited immediate connectivity per node of 2.5, 4 hops (default for Gnutella and Freenet) can cover nearly all the nodes except when the undistinguished node connection probability is very high.

2. Distinguished nodes provide a "short cut" between nodes; therefore, with a low value of "undist prob" (or q_u), the overall reachability increases very fast.

3. As the size of the network increases, the percentage of nodes that can be covered with a given number of hops goes down (as expected), however, a small number of hops can still cover a lot of nodes.

4. Given the assumption of responses tracing back the request path, the response traffic per undistinguished node (column 5) increases very rapidly for the second and third hops (since these hops add an awful lot of nodes to the reachability tree). Beyond the third hop, the response traffic penalty is negligible.

5. An interesting traffic parameter is the total traffic divided by the network size. With 4 hops, this metric varies between 3.3 and 3.8. That is, if each request returns a response, the nodal capacity must increase 3-4 times as fast as the network size.

4 A Preliminary Simulation Model

In this section, we present our basic simulation model and study the impact of complex arrival patterns, service time at each node and finite queue length on performance metrics such as throughput, response time and node utilization.

4.1 Overview of the Simulation Model

The model constructs a network of a 100 peer nodes as follows. A group of 10 distinguished nodes are interconnected uniformly with each node's degree of connectivity equal to 2. Each undistinguished node is then added to the network as follows. When a node is added to the network, it connects to 3 distinguished nodes and 1 undistinguished node. These nodes are chosen from the existing nodes in the network based on the degree of connectivity, i.e. the nodes with the lowest degrees of connectivity are chosen first. This construction attempts to emulate an average case

network corresponding the random graph model discussed in Section 3 as opposed to an exhaustive approach which would require simulating an extremely large number of instances (not feasible).

We represent the request generation (or arrival) process at each peer by an on-off process with constant request generation rate during the on period. This allows us to emulate a self-similar process as shown in [9] with a given Hurst parameter. Although P2P traffic characteristics are unlikely to be well understood in the near future, it is reasonable to assume that it will show long-range dependence much like the request level web traffic as shown in [5]. Let X_0 and X_1 denote the off and on periods. Assume that X_0 and X_1 are iid with the following Pareto distribution:

$$P(X_i > x) = \left(\frac{x}{T_i}\right)^{-\alpha}, 1 < \alpha < 2, \;\; x \geq T_i, \;\; i = 0, 1 \tag{4}$$

where T_i is the minimum value of the off/on period and α is the decay rate of the Pareto distribution. The mean off/on period is given by $E(X_i) = \frac{\alpha T_i}{\alpha - 1}$. We shall choose a mean on/off period of 15 minutes (900 seconds). It is shown in [9] that the Hurst parameter of the combined arrival process is given by: $H = (3 - \alpha)/2$. For the analysis in this paper, we consider a high burstiness scenario [5] by choosing $H = 0.8$, which gives $\alpha = 1.4$, and $T_i = 6$ minutes. We also assume an average on-period request rate of $1/$ min.

In the simulation model, a search request for a document proceeds to all neighboring nodes up to a certain number of hops (h), and the response is returned to the requesting node along the same path as the request is received. The response from every node reached in the network only contains hit/miss and metadata information. For the request service process, we make the simplifying assumption that each request is served (e.g., search execution followed by a response return) within a constant delay at a node. Furthermore, we assume that both requests and responses share a single finite queue at each node. If a request or response arrives at a node with a full queue, it is simply dropped. In such cases, we assume that a response equivalent of an HTTP "server too busy" message is sent to the requesting node and does not consume any processing resources or bandwidth.

4.2 Simulation Results and Analysis

In the simulation experiments conducted, we vary three important parameters (number of hops, nodal service delay and queue length) to understand their effect on reachability, node utilization and overall response time. We present these simulation results in Table 3.

We start by analyzing the average number of requests issued per search transaction (column 4 & 5, Table 3). Since these requests or their associated responses can be dropped when the finite queues are full at the traversed nodes, we further divide it into the average number of successful (labeled as OK, col 4) and unsuccessful requests (labeled as dropped, col 5). The first three columns in the table represent the queue length (varied as 100 & 500), request service

Table 3: Simulation Results for a 100 node network

Queue	Req Serv	Num. of	Reqs/Search		Utilization		Response Time	
length	time	hops	okay	dropped	dnode	udnode	first_resp	last_resp
500	100	1	5.99	0.00	2.11%	0.62%	1.00	0.60
500	100	2	58.11	0.00	37.14%	8.87%	6.62	0.60
500	100	4	99.83	0.17	66.09%	18.67%	12.15	0.61
500	300	1	5.99	0.00	6.32%	1.08%	1.42	1.00
500	300	2	58.11	0.00	49.67%	16.50%	7.85	1.03
500	300	4	99.59	0.41	79.28%	32.55%	15.68	1.10
100	100	1	5.99	0.00	2.11%	0.62%	1.00	0.60
100	100	2	57.97	0.14	37.23%	8.87%	6.70	0.60
100	100	4	99.32	0.68	66.46%	18.73%	13.73	0.61
100	300	1	5.99	0.00	6.33%	1.08%	1.42	1.00
100	300	2	57.95	0.16	49.78%	16.51%	7.99	1.03
100	300	4	98.27	1.73	80.24%	37.45%	19.44	1.12

time (varied as 100ms & 300ms) and number of hops (h, varied from 1 to 4) allowed per search. The service time for processing responses along each node on the backward path is fixed at 100 ms. The link service time is also fixed at 100ms, a conservative estimate in most cases. Overall, we find that the number of unique OK responses received during a search is roughly 6 for $h = 1$, 58 for $h = 2$ and almost a 100 for $h = 4$. If the number of hops increase or if the queue length is reduced, the number of dropped requests/responses increases significantly. A detailed look at the utilization at each node further explains this behavior.

Next, we take a look at the utilization (column 6 & 7 in Table 3) at the distinguished and undistinguished nodes. Overall, we find that as the number of hops increases from 1 to 4, the utilization at the distinguished nodes (2% to 80%) increases very rapidly as compared to that at the undistinguished nodes (0.5% to 40%). From this observation, we can determine that the distinguished nodes need to be more powerful than the undistinguished nodes and actually come close to representing servers in the traditional client-server approach to computing. This happens because peers generally join the network by looking up a list of always-alive nodes (like the web-site `gnutellahosts.com` for Gnutella users). Later on, the the always-alive nodes can provide the incoming peer with a list of other nodes in the network and the user may choose to connect to them. We believe that the 3:1 ratio of distinguished nodes to undistinguished neighbors per node is rather optimistic i.e. clients are more likely to remain connected to more always-alive nodes rather than the other undistinguished nodes in the network. Thus, our estimation of utilization is probably optimistic and can only be considered as an indication of utilization in a very well-behaved network.

The average response time is presented in columns 8 & 9 of Table 3. Column 8 shows the time to the first response in seconds, while column 9 shows the time to the last response in seconds. While the three parameters have little impact on the time to the first response, the time to the last response is significantly affected. For a service time of

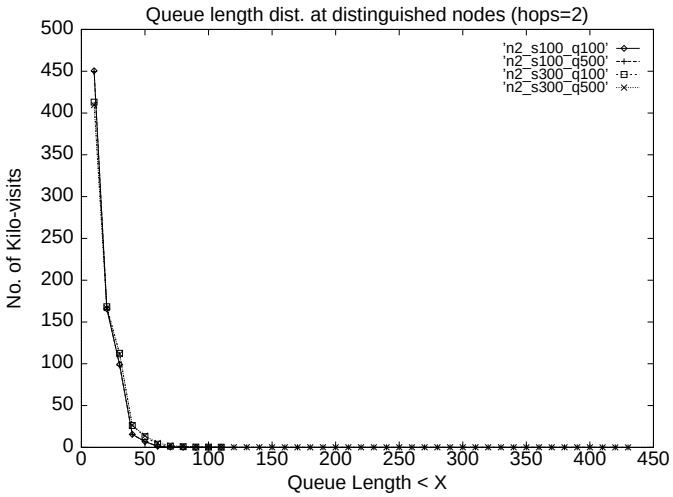

Figure 1: Qlength dist. at distinguished nodes with $h = 2$

100ms per node, the time to the first response is roughly 0.6 seconds. When the service time increases to 300 ms, the time to the first response increases to roughly 1 second for $h = 1$ and 1.1 seconds for $h = 4$. The time to the last response increases by roughly a factor of 12 as the number of hops increases from 1 to 4. Some studies in the web environment show that the patience limit for most users is about 8 seconds. If this is applied directly to the peer-to-peer file searching environment, allowing more than 2 hops for searches seems futile since the time to the last response will exceed 8 seconds, by which time the user would have either abandoned the request or retried it. Furthermore, as the service time increases from 100 to 300, there is a 30% or more increase in the response time. Much of this increase is due to queuing delays at the nodes in the network.

In particular, the queuing delays at the heavily utilized distinguished nodes are bound to be the performance limiter. Figure 1 shows the arriver's queue length distribution at a distinguished node when the number of hops fixed is at 2. The x-axis represents the length of the queue when a request or a response arrives at the node. The y-axis represents the number of visits to the node that find the queue length between X and $X - B$, where B is the bucket size. In other words, i.e. A point X, Y represents that Y visits arrive at the node and find a sum of Q requests and responses in the queue, where Q is within the range X, $X - B$. For these figures, we chose a bucket size of 10. Figure 1 shows that the queuing component is significant when the number hops is equal to 2. For example, with a service time of 100 ms, and a buffer size of 500 entries, we find that more than 40% of the requests have to wait for 10 or more requests/responses.

5 Model Extensions and Research Issues

In this paper, we introduced preliminary analytic and simulation models in order to study several interesting issues concerning ad hoc file sharing networks. We are in the process of extending this work in order to study more general issues for peer to peer computing. In this section, we elaborate on some of these potential extensions in order to highlight some of the problems that need to be resolved in this emerging area.

One important issue in the general peer to peer information location area is the need for careful re-design of request-response protocols that are appropriate for the environment under consideration. For example, in certain environments, the user may be interested in finding either a unique match or at most a few matches but the searches may still result in hits in many nodes (due to redundant object storage and/or the way matching algorithm operates). In such a scenario, it is best for each node to supply hit/miss and metadata information in the search response rather than respond with the object, especially if the object size is large. The actual object retrieval is then done later by an explicit request to the desired node(s). On the other hand, if, on the average, the number of responses returned is comparable to the number of objects of interest to the user, it may make sense for the response to contain the object itself. It is interesting to investigate the boundaries of these two approaches for various network sizes, redundancy level, popular key matching techniques, and the applications of interest (e.g., MP3 files, news items, technical literature search, etc.)

Apart from the content in response message(s), the path of the response is an equally important issue. For example, we could consider three different alternatives: (1) returning each individual response along the path that the request traveled as in Freenet, (2) aggregating response at each node along the return path, and (3) returning the individual response directly to the requester. In the first case, “aggregation of response” simply means putting multiple hit/miss responses into a single message which is sent along the backward path. This reduces the total number of small messages flowing in the network and thus makes the communication more efficient. However, such aggregation requires that the responses be delayed at each node along the backward path. If the response delays already more than a few seconds, such an approach may be counterproductive because of the possibility of user abandonments and retries. If the response contains the matched object, pooling multiple such responses into a single message is perhaps not very useful; therefore, by aggregation in this case we mean examining the associated metadata and squelching duplicate responses. We intend to evaluate advantages of aggregation in various environments using enhanced version of our simulator.

The choice of request-response protocol discussed above has a significant impact on response time (especially the time to the first response). Some studies in the web environment show that the patience limit for most users is about 8 seconds. If the user does not get response within this period, he/she may abandon the request and might even retry.

Abandonment in a distributed environment is very expensive because it is not possible to quickly identify all orphan transactions and squelch them. Intermittent connectivity of nodes, lack of global time, and low-speed last mile links in a peer-to-peer environment complicate the situation further. Retries put even more load and thus aggravate the situation.

Content management and redistribution is also an interesting topic when designing peer-to-peer protocols. One obvious mechanism is caching the content in responses as they flow through the intermediate nodes. In addition, several other mechanisms can be developed including distributing content to immediate neighbors and incorporating intelligent agents in the network that adapt to access patterns and identify appropriate locations for re-distributing content across the network. For example, a recent measurement study on Gnutella users [1] reaches the rather disturbing conclusion that most users don't contribute any files to the community and simply download files from a minority of contributing nodes. In order to prevent these few nodes from becoming hot-spots, intelligent re-distribution based on access frequency and location of the content can be envisioned. Reference [8] talks about mapping of application-level network to the physical network in Gnutella and suggests that agents be incorporated to monitor the network and provide hints on how to create the application topology.

Finally, an overall observation from our study is that a pure peer-to-peer computing paradigm may not be well-suited for many applications including file-sharing. Instead, it might be more attractive to consider a hybrid approach where the "distinguished nodes" are traditional servers that not only act as nuclei for forming peer-to-peer networks but actually host the most frequently accessed information. For example, in the web-server context, it is well known that 10-15% of the documents account for 80-85% of the hits, and moreover, these documents are typically small in size. Thus one strategy is to use traditional web-servers for serving frequently used documents and spread the others across peers. With such a hybrid architecture, server engineering [5] and mechanisms for server overload control [3] of these servers will become important research topics.

References

[1] E. Adar and B. A. Huberman, "Free riding in Gnutella", Xerox PARC report, Oct 2000, available at www.parc.xerox.com/istl/groups/iea/papers/gnutella.

[2] I. Clarke, "A Distributed Decentralized Information Storage and Retrieval system." M.S. Thesis, Division of Informatics, Univ of Edinburgh, UK, 1999.

[3] R. Iyer, V. Tewari, and K. Kant, "Overload control mechanisms for web servers", Performance and QoS of Next Generation Networks, Nagoya, Japan, Nov 2000, pp 225-244

[4] K. Kant, R. Iyer and V. Tewari, "On the Potential of Peer-to-Peer Computing: Classification and Evaluation", submitted for publication. available at http://kkant.ccwebhost.com/download.html

[5] K. Kant and Y. Won, "Server Capacity Planning for Web Traffic Workload", IEEE transactions on knowledge and data engineering, Oct 1999. pp731-747.

[6] R. Kumar, P. Raghvan, et. al., "The web as a graph", ACM POD conference, Dallas, TX, 2000.

[7] M.A. Jovanovic, F.S. Annexstein and K.A. Berman, "Scalability issues in large peer to peer networks – A case study of Gnutella", Tech. Report, Univ. of Cincinati, 2001.

[8] Matei Ripeanu, "Peer-to-Peer Architecture Case Study: Gnutella", Proc. of 2001 Intl Conf on Peer-to-peer Computing (P2P2001), Linkoping Sweden, 27-29 August 2001.

[9] W. Willinger, M.S. Taqqu et al., "Self-Similarity through High Variability: Statistical Analysis of Ethernet LAN traffic at the source level", Proc of SIGCOMM 95, pp100-113.

A Computation of Nodal Degree

In order to write a recurrence equation for $P_n(\ell_2, \ell)$, we introduce the connection probability $\alpha(\ell_2, \ell)$, defined as the probability that the addition of a new node to the network (at level ℓ_2) will result in one more connection to a level ℓ node.

Let us first suppose that $\ell > 0$, i.e., we are considering how the degree of a nondistinguished node changes with the new node addition. Suppose that the level ℓ_2 node attempts to connect to k nodes, of which i are undistinguished nodes. When $\ell_2 = 1$, $\alpha(\ell_2, \ell) = 0$ for $\ell > 0$ since there are no undistinguished nodes to connect. Otherwise, if $\ell_2 - 1 < i$, only $\ell_2 - 1$ connections are actually possible with undistinguished nodes. It follows that for $\ell_2 > 1$, $\ell > 0$,

$$\alpha(\ell_2, \ell) = \sum_{k=0}^{K} \theta_k \left[\sum_{i=0}^{\min(\ell_2 - 1, k)} \frac{i}{\ell_2 - 1} B_{k,i}(q_u) + \sum_{i=\ell_2}^{k} B_{k,i}(q_u) \right] \tag{5}$$

where $B_{k,i}(q_u)$ is the binomial mass function

$$B_{k,i}(q_u) = \begin{pmatrix} k \\ i \end{pmatrix} q_u^i (1 - q_u)^{k-i} \tag{6}$$

with $B_{0,0}(q_u)$ considered as 1. The equation follows from the fact that the probability of connection attempt to i undistinguished nodes is $B_{k,i}(q_u)$, and a given node will be selected with probability $i/(\ell_2 - 1)$ if $i \leq \ell_2 - 1$, and

with probability 1 otherwise. We note that the above expression for $\alpha(\ell_2, \ell)$ does not involve ℓ itself. This is a result of the fact that each newly arriving node connects to all existing undistinguished nodes equi-probably. It is certainly possible to extend the analysis where this is not the case.

For $\ell_2 > K$, the entire expression collapses to the following intuitively obvious expression:

$$\alpha(\ell_2, \ell) = q_u E[\mathcal{K}]/(\ell_2 - 1), \qquad \ell_2 > K \tag{7}$$

Note that only the mean value of $\mathcal{K}$ is relevant in this case; the rest of the distribution doesn't matter. For $\ell_2 \leq K$, no simplification appears possible; however, if necessary, simple approximations could be used without significant impact on results.

Next, let us evaluate $\alpha(\ell_2, 0)$. Note that if the new node connects to i undistinguished nodes, it will connect to $k-i$ distinguished nodes. Since there are a total of $N_d \geq K$ distinguished nodes, we have $\alpha(\ell_2, 0) = \sum_{k=0}^{K} \theta_k \zeta(k, \ell_2)$ where

$$\zeta(k, \ell_2) = \sum_{i=0}^{\min(\ell_2 - 1, k)} \frac{k-i}{N_d} B_{k,i}(q_u) + \sum_{i=\ell_2}^{k} \frac{k - \ell_2 + 1}{N_d} B_{k,i}(q_u)$$

If $\ell_2 > K$, the entire expression again collapses to the obvious equation:

$$\alpha(\ell_2, 0) = (1 - q_u) E[\mathcal{K}]/N_d, \qquad \ell_2 > K \tag{8}$$

Assuming that $P_n(.,.) = 0$ for $n < 0$, we can now write the following recurrence equation for $P_n(\ell_2, \ell)$ as:

$$\begin{aligned} P_n(\ell_2, \ell) &= \alpha(\ell_2, \ell) P_{n-1}(\ell_2 - 1, \ell) \\ &+ [1 - \alpha(\ell_2, \ell)] P_n(\ell_2 - 1, \ell) \end{aligned} \tag{9}$$

For $\ell > 0$, this equation applies for $\ell < \ell_2$, i.e., when level ℓ node in consideration is not the same as the node being added to the network. For $\ell = \ell_2$, the degree of the added node is simply $\mathcal{K}$. That is,

$$P_k(\ell, \ell) = \theta_k, \qquad 0 \leq k \leq K \tag{10}$$

For $\ell = 0$, the above recurrence equation fails to apply only initially, i.e., when the network consists of only distinguished nodes. Since we are assuming a regular M-ary connectivity initially, we have:

$$P_n(0, 0) = \begin{cases} 1 & n = M \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Let $\Phi(z|L, \ell)$ denote probability generating function of the degree distribution. We have:

$$\Phi(z|\ell_2, \ell) = \sum_{n=0}^{\ell_2 - \ell + K} z^n P_n(\ell_2, \ell)$$

$$
\begin{aligned}
&= \alpha(\ell_2,\ell) \sum_{m=0}^{\ell_2-\ell-1+K} z^{m+1} P_m(\ell_2-1,\ell) \\
&\quad + [1-\alpha(\ell_2,\ell)] \sum_{n=0}^{\ell_2-1-\ell+K} z^n P_n(\ell_2-1,\ell) \\
&= [1-\alpha(\ell_2,\ell)(1-z)]\Phi(z|\ell_2-1,\ell), \qquad \ell < L
\end{aligned}
$$

where we have used the fact that $P_{\ell_2-\ell+K}(\ell_2-1,\ell)=0$. Using this relationship recursively, we get

$$\Phi(z|L,\ell) = \Phi(z|\ell,\ell) \prod_{i=\ell+1}^{L} [1-\alpha(i,\ell)(1-z)] \tag{12}$$

It thus follows:

$$\Phi(z|L,\ell) = \Psi(z) \prod_{i=\ell+1}^{L} [1-\alpha(i,\ell)(1-z)] \tag{13}$$

where the function $\Psi(z)$ is defined as:

$$\Psi(z) = \begin{cases} \sum_{k=0}^{K} z^k \theta_i & \ell > 0 \\ \\ z^M & \ell = 0 \end{cases} \tag{14}$$

This completes the characterization of the degree distribution. Let $D_{at}(\ell|L)$ denote the mean degree *at* a node that enters the network at level ℓ, given a total of L levels. By definition,

$$D_{at}(\ell|L) = \Phi'(z|L,\ell)\Big|_{z=1} = \Psi'(1) + \sum_{j=\ell+1}^{L} \alpha(j,\ell) \tag{15}$$

Now we separately consider the $\ell > 0$ and $\ell = 0$ cases. In the former case, if $\ell \leq K$, the above summation can be divided into two parts, one for the range $\ell+1 \,..\, K$ and the other from $K+1 \,..\, L$. Then, using equation (7), for $0 < \ell < K$, we get:

$$D_{at}(\ell|L) = E[\mathcal{K}] + q_u E[\mathcal{K}] \sum_{i=K}^{L-1} 1/i + \sum_{j=\ell+1}^{K} \alpha(j,\ell) \tag{16}$$

For $\ell > K$, the summation term drops out completely, and we have:

$$D_{at}(\ell|L) = E[\mathcal{K}] + q_u E[\mathcal{K}] \sum_{i=\ell}^{L-1} 1/i, \qquad \ell > K \tag{17}$$

Now, for $\ell = 0$, we have two ranges, one $0 \,..\, K$, and the other $K+1 \,..\, L$. Using equation (8), we have:

$$D_{at}(0|L) = M + E[\mathcal{K}](1-q_u)\frac{L-K}{N_d} + \sum_{j=1}^{K} \alpha(j,0) \tag{18}$$

If needed, the overall average degree of a node could then be computed as:

$$D_{at}(L) = \frac{N_d D_{at}(L,0) + \sum_{\ell=1}^{L} D_{at}(\ell|L)}{N_t} \tag{19}$$

Let us now adapt the above analysis to obtain the undistinguished degree of a node, defined as the number of undistinguished nodes to which a given node is directly connected. Let us denote this as $P_n^{(u)}(\ell_2, \ell)$, defined just like the overall degree $P_n(\ell_2, \ell)$. Since every added node in the network construction process is a undistinguished node, equation (9) holds for $P^{(u)}$ as well, except that the initial conditions are different. In particular, let k_u denote the number of undistinguished nodes that a newly arriving level $\ell > 0$ node connects to. Obviously, $k_u \in 0 \,..\, \min(K, \ell - 1)$; therefore, $\theta_u(k_u, \ell)$, the probability of a newly arriving level ℓ node connecting to k_u undistinguished nodes is:

$$\theta_u(k_u, \ell) = \sum_{k=k_u}^{K} \theta_k B_{k,k_u}(q_u), \;\; 0 \le k_u \le \min(K, \ell - 1)$$

Note that even if $\theta_0 = 0$ (i.e., an incoming node always connects to at least one node), k_u could still be 0. Also, $\theta_u(k_u, \ell)$ depends on ℓ for $\ell \le K$ only. Now, equation (10) becomes $P^{(u)}(k_u|\ell, \ell) = \theta_u(k_u, \ell)$. Since, a distinguished node is initially not connected to any undistinguished node, eqn(11) reduces to:

$$P_n^{(u)}(0, 0) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

With this, the PGF of $P^{(u)}$ is given by an equation like (13), but with Ψ function suitably modified for the changed initial conditions, i.e.,

$$\Psi_u(z) = \begin{cases} \sum_{k=0}^{K} z^k \theta_u(i, \ell) & \ell > 0 \\ \\ 1 & \ell = 0 \end{cases} \tag{21}$$

Let $D_{at,u}$ be defined like D_{at} except that we are now interested in only the undistinguished degree of a node. Equations (16–18) then apply to $D_{at,u}$ as well except for the first term modified to correspond to $\Psi'_u(1)$ instead of $\Psi'(1)$. Note that for $\ell > K$,

$$\Psi'_u(1) = \sum_{k_u=0}^{K} k_u \theta_u(k_u, \ell) = \sum_{k=0}^{K} \theta_k \sum_{k_u=0}^{k} k_u B_{k,k_u}(q_u)$$

which evaluates to $q_u E[\mathcal{K}]$, as expected.

B Reachability Distribution

We start with the reachability of all nodes from a given node. For this, we arbitrarily number the distinguished nodes as $1 \,..\, N_d$, and subsequently a level ℓ node as $\ell + N_d$. Let $R_1(n_1, n_2)$ denote the probability of reaching node n_1 from node n_2 in one hop. Let R_1 denote the corresponding $N_t \times N_t$ matrix. To specify R_1, we first compute a

related matrix R_0 that enumerates the probability of finding an arc between any given pair of nodes. Since R_0 is a symmetric matrix and for any node n_1, $R_0[n_1, n_1] = 0$; therefore, it suffices to assume $n_2 > n_1$ in the following:

$$R_0[n_1, n_2] = \begin{cases} M/(N_d - 1) & n_1 \leq N_d, n_2 \leq N_d \\ \alpha(n_2 - N_d, 0) & n_1 \leq N_d, n_2 > N_d \\ \alpha(n_2 - N_d, n_1 - N_d) & n_1 > N_d, n_2 > n_1 \end{cases}$$

Given R_0, we construct R_1 by normalizing the columns to 1. This ensures that the total probability of reaching all other nodes from any node is 1. Note that R_1 is not a symmetric matrix, but it is possible to exploit its special structure.

Let $R_h[n_1, n_2]$ denote the probability of reaching node n_1 from node n_2 in h hops. In order to compute the matrix R_h, it is necessary to enumerate all unique paths of length h that do not use any intermediate node more than once. The latter requirement increases the computational cost, since a straightforward recursion on h cannot avoid node reuse. We also need to renormalize the matrix at each step to ensure that the columns sum to 1. We denote the unnormalized matrices as R'_h where we specifically set $R'_h[n, n] = 0$ for all n. Then, $R_h = R'_h \times \mathrm{diag}(\mathbf{e}.R'_h)$ where $\mathbf{e}$ denotes a row vector of all 1's and the function diag converts a row vector into a diagonal matrix by putting the vector elements along the diagonal. Now, for $n_1 \neq n_2$,

$$R'_2[n_1, n_2] = \sum_{n_3=1}^{N_t} R_1[n_1, n_3] R_1[n_3, n_2] \tag{22}$$

where it is not necessary to specifically avoid $n_3 = n_1$ or $n_3 = n_2$ since $R_1(n, n) =$ for all n. However, for R'_3, we have:

$$R'_3[n_1, n_2] = \sum_{\substack{n_4=1 \\ n_4 \neq n_2}}^{N_t} \sum_{\substack{n_3=1 \\ n_3 \neq n_1}}^{N_t} R_1[n_1, n_4] R_1[n_4, n_3] R_1[n_3, n_2] \tag{23}$$

Similar equations can be written for R'_i, $i > 3$. We now define a modified $(L + 1) \times (L + 1)$ matrix $\mathcal{R}_h$ where the rows and columns indicate level, rather than individual nodes:

$$\mathcal{R}_h[\ell_1, \ell_2] = \begin{cases} (N_d - 1)\, R_h[1, 2] & \ell_1 = 0, \ell_2 = 0 \\ R_h[\ell_1 + N_d, 1] & \ell_1 > 0, \ell_2 = 0 \\ N_d\, R_h[1, \ell_2 + N_d] & \ell_1 = 0, \ell_2 > 0 \\ R_h[\ell_1 + N_d, \ell_2 + N_d] & \ell_1 > 0, \ell_2 > 0 \end{cases} \tag{24}$$

This matrix suffices for further computation. The sole reason for defining R matrices in terms of node numbers earlier was to ensure that no node gets used more than once on a path.

Let $D_{fr}(\ell_2, h|L)$ denote the average degree of the nodes reached in h hops starting *from* a node at level ℓ_2 when the maximum level is L. It is convenient to denote row vectors of $D_{at}(\ell|L)$'s and $D_{fr}(\ell, h|L)$'s for $\ell = 0..L$ as $\mathbf{D}_{at}(L)$ and $\mathbf{D}_{fr}(h|L)$ respectively. Then, $\mathbf{D}_{fr}(1|L) = \mathbf{D}_{at}(L)$, and for $h > 1$

$$\mathbf{D}_{fr}(h|L) = \mathbf{D}_{at}(L)\mathcal{R}_{h-1} \tag{25}$$

In the analysis presented here, the last equation is the only place where the matrix $\mathcal{R}_h$ is used; therefore, explicit computation of $\mathcal{R}_h$ is unnecessary (and very expensive). Instead, it is desirable to compute $\mathbf{D}_{fr}(h|L)$ directly for each h.

Let $G(\ell_2, h)$ denote the average number of nodes reached *at* hth hop, and $G^{(t)}(\ell_2, h)$ the total number of nodes reached in h hops (ℓ_2 is the starting level in both cases). Again, it is convenient to work with the corresponding row vectors henceforth denoted as $\mathbf{G}(h)$ and $\mathbf{G}^{(t)}(h)$. Then, $\mathbf{G}(1) = \mathbf{D}_{at}(L)$ (i.e., the degree of the starting node) and $\mathbf{G}^{(t)}(0) = \mathbf{e}$ (the starting or "root" node). For $h > 0$, we have:

$$\mathbf{G}^{(t)}(h) = \mathbf{G}^{(t)}(h-1) + \mathbf{G}(h), \qquad 1 \le h \le H \tag{26}$$

where H denotes the highest hop count of interest. Now, to obtain an expression for $\mathbf{G}(h)$ for $h > 1$, we imagine that the nodes at $(h-1)^{st}$ hop are expanded sequentially in order to count the unique nodes reachable from each of them. Let $g_m(\ell_2, h)$ denote the number of unique nodes accounted for at the mth step of the expansion. Then, in the $G(\ell_2, h-1)$th step, we would have accounted for all $G(\ell_2, h)$ nodes. With $g_0(\ell_2, h) = 0$ for all ℓ_2 and h, we can write the following recurrence equation for $g_m(\ell_2, h)$, $h > 1$

$$\begin{aligned} g_m(\ell_2, h) = g_{m-1}(\ell_2, h) + D_{fr}(\ell_2, h|L) \\ \frac{N_t - G^{(t)}(\ell_2, h-1) - g_{m-1}(\ell_2, h)}{N_t} \end{aligned} \tag{27}$$

This equation is obtained as follows: At mth step, $N_t - G^{(t)}(\ell_2, h-1) - g_{m-1}(\ell_2, h)$ nodes out of a total of N_t nodes have already been accounted for; therefore, the number of new nodes added is simply the average node degree multiplied by the fraction of unreached nodes. This equation can be rewritten as

$$g_m(\ell_2, h) = a.g_{m-1}(\ell_2, h) + b \tag{28}$$

where the quantities a and b are independent of the index m and are given by $a = 1 - D_{fr}(\ell_2, h|L)/N_t$ and $b = [N_t - G^{(t)}(\ell_2, h)](1-a)$. A repeated expansion of equation (28) along with $g_0(\ell_2, h) = 0$, yields the following explicit recurrence for $G(\ell_2, h)$.

$$\begin{aligned} G(\ell_2, h) = g_{G(\ell_2,h-1)}(\ell_2, h) = b\frac{1 - a^{G(\ell_2,h-1)}}{1-a} \\ = [1 - a^{G(\ell_2,h-1)}][N_t - G^{(t)}(\ell_2, h-1)] \end{aligned} \tag{29}$$

Thus, $\mathbf{G}(h)$, and hence $\mathbf{G}^{(t)}(h)$ can be computed.

Next we adapt the above equations to obtain reachability to only undistinguished nodes. For this, we first need to compute the average undistinguished degree of nodes reached in h hops from a given level ℓ_2, given a total of L levels. We henceforth denote this as $D_{fr,u}(\ell_2, h|L)$ and also define the corresponding row vector $\mathbf{D}_{fr,u}(h|L)$. As with $\mathbf{D}_{fr}(h|L)$, we have $\mathbf{D}_{fr,u}(1|L) = \mathbf{D}_{at,u}(L)$, and for $h > 1$

$$\mathbf{D}_{fr,u}(h|L) = \mathbf{D}_{at,u}(L)\mathcal{R}_{h-1} \tag{30}$$

Next, we define the quantities $\mathbf{G}_u(h)$ and $\mathbf{G}_u^{(t)}(h)$ respectively, which have the same meaning as $\mathbf{G}(h)$ and $\mathbf{G}^{(t)}(h)$ except that reachability to only undistinguished nodes is being considered. These quantities can be determined by a slight tweak to the analysis presented above. Note that the paths to undistinguished nodes could well pass through distiniguished nodes; therefore, if we are interested in paths of length h, we still proceed as in the last subsection up to length $h-1$ and then estimate the number of reachable undistinguished nodes in the last step. Let $g_{m,u}(\ell_2, h)$ denote the number of undistinguished nodes reached in the mth step for length h paths. Note specifically that the upper bound on m is $G^{(t)}(\ell_2, h-1)$ [instead of $G_u^{(t)}(\ell_2, h-1)$]. With $g_{0,u}(\ell_2, h) = 0$, equation (27) can be adapted as follows:

$$\begin{aligned} g_{m,u}(\ell_2, h) = g_{m-1,u}(\ell_2, h) + D_{fr,u}(\ell_2, h|L) \\ \frac{L - G_u^{(t)}(\ell_2, h-1) - g_{m-1,u}(\ell_2, h)}{L} \end{aligned} \tag{31}$$

This equation is again based on the observation that at mth step the number of new undistinguished nodes added is the undistinguished degree multiplied by the fraction of unvisited undistinguished nodes. The quantities $\mathbf{G}_u^{(t)}(h)$ and $\mathbf{G}_u(h)$ are again related as:

$$\mathbf{G}_u^{(t)}(h) = \mathbf{G}_u^{(t)}(h-1) + \mathbf{G}_u(h), \qquad 1 \le h \le H \tag{32}$$

along with the initial conditions $\mathbf{G}_u(1) = \mathbf{D}_{at,u}(L)$ (i.e., the undistinguished degree of the starting node) and $\mathbf{G}_u^{(t)}(0) = [0, 1, \ldots, 1]$. As in the last section, equation (31) can be solved explicitly to give:

$$G_u(\ell_2, h) = [1 - a^{G(\ell_2, h-1)}][L - G_u^{(t)}(\ell_2, h-1)] \tag{33}$$

where $a = 1 - D_{fr,u}(\ell_2, h|L)/L$.