Telematics Institute

Technical Report

Addendum to „Performance of Lookup Operations in a Hypercube-based P2P Data Store: Theoretical Model and Performance Evaluation“

Dietrich Fahrenholtz

Juli 2006

Report No. **TR-2006-07-01**

Hamburg University of Technology
Schwarzenbergstrasse 95, 21073 Hamburg, Germany
phone: +49 40 42878-3860; Fax:+49 40 42878-2581
email: telematik@tuhh.de
web: http://www.ti5.tu-harburg.de

Addendum to "Performance of Lookup Operations in a Hypercube-based P2P Data Store: Theoretical Model and Performance Evaluation"

Dietrich Fahrenholtz
Hamburg University of Technology
Institute of Telematics
fahrenholtz@tu-harburg.de

Abstract

This is an addendum to our report "Performance of Lookup Operations in a Hypercube-based P2P Data Store: Theoretical Model and Performance Evaluation". Here we explore two questions: First, what is the probability of a lookup to get lost and, second, what is the probability that it will be duplicated at least once. To this end, we approach the questions on a probability theoretical basis and derive equations for both probabilities, means and variances. Graphs elucidate the behavior of the model.

1. Two ways of looking up data

It is not only interesting how much time/hops a lookup needs in our P2P data store to find a data item on average but also what is the probability of a lookup message to get lost during its execution and thus how much time a lookup originating peer has to wait the minimum before it can declare the lookup to have failed. In this respect, we have to distinguish two ways lookups can be executed by peers:

1. iteratively or
2. recursively

First of all, routing in both cases is done in a greedy fashion, i.e., in each step of the distributed algorithm a lookup message is forwarded to that node that is responsible for the smallest possible interval that contains the search key until the destination node is reached. In the iterative case, the lookup originating peer performs all routing steps itself. It contacts peers of the remote node belonging to the current routing step until one of them answers with a list of remote peers for the next routing step. Remote peers can be contacted simultaneously or one after the other. In the former case, a duplication of lookup messages occurs leading to waste of bandwidth, while in the latter case, the lookup message forwarding peer might have to try a number of remote peers until an answer arrives leading to a waste of time. Answering the question what is optimal, all depends on π_{inact}. If π_{inact} is small, then the one after the other principle might be better. If π_{inact} is large, then sending more than one lookup request is advisable to keep the waiting time bounded. Once the list of remote peers for the next routing step has been received, the iterative forwarding of the lookup message is repeated using peers of the new list. The iterative process terminates when the lookup reaches its destination and an answer to it was obtained. Most noteworthy advantages of performing lookups the iterative way are:

- A lookup message does not get lost and thus a lookup does not die. If it does after all, that means the lookup originating peer became inactive. Since no other peer is interested in the result of this lookup, it logically dies with its originating peer.
- The control over a lookup lies with the originating peer. It can decide how many peers should be contacted simultaneously, for instance.
- The originating peer always knows the status of the current step. Eventually, it definitely knows if the looked up data item is in the P2P data store or not.

The disadvantages of iterative lookups are:

- The whole work and communication costs of the lookup process lie with the lookup originating peer.

- List of remote peer references must be transmitted to lookup originating peers consuming additional bandwidth.

In the recursive case, lookup messages are forwarded from node to node until they reach their destination node. No peer forwards the same lookup message twice except for the case a remote peer turns out to be inactive. Forwarding of messages is as in the iterative case. However, a successful lookup message delivery does not require the prior sending of a peer list. Finally, the peer in the destination node contacts the originating peer and delivers the answer. Most noteworthy advantages of looking up data items recursively are:

- The lookup originating peer can do other things after it has forwarded a lookup message.
- Communication costs for lookup originating peers are much smaller than in the iterative case. In fact, they are the same on average for all peers on the lookup path.
- No lists containing remote peer references are transmitted between intermediate peers in order to keep a lookup going.
- Because of the last argument, looking up a data item recursively can be quicker than in the iterative case if peer inactivity is low. Most likely it will be slower if peer inactivity is high.
- Data items can be cached on intermediate peers. However, they out-date quickly in case updates of these data items occur. So some strategy keeping cache contents valid has to ensure consistency for at least a period of time.

The disadvantages of recursive lookups are:

- Responsibility for spurring lookups mainly lies with intermediate peers. So lookups either can get lost due to peers failing, they can be delayed or even may be dropped deliberately by malicious peers.
- The lookup originating peer does not know the overall status of the lookup.

Further disadvantages pertaining to both iterative and recursive way of looking up data items are

- If a lookup forwarding peer cannot contact any of the peers of the next routing step for whatever reasons, the lookup cannot proceed and thus fails. This could happen due to network infrastructure anomalies, for example.
- The lookup originating peer cannot compute the exact arrival time of the answer to the lookup or a peer references list. It has to rely on a timeout estimate. If this timeout passes, the lookup originating peer declares the lookup failed or the contacted remote peer dead, respectively, regardless whether or not the answer or the peer list arrives afterwards.
- There may be more than one remote peer that is answering. This could be due to a lookup request duplication by peers of intermediate routing steps.

2. Likelihood of loosing lookup messages

After listing advantages and disadvantages of both ways of looking up data items, we shall proceed with deriving an equation on the probability of a lookup operation to fail. Since one of the advantages of iteratively looking up data items is that originating peers follow lookup operations through as long as these peers are active, we do not need to consider them any longer. So we turn to the recursive case and determine the probability of a lookup message to get lost on intermediate peers. To this end, we, first, need the probability that a remote peer is active and neither the lookup message was lost in transit nor the associated acknowledgment was lost. Let

$$\pi_{\text{suc}} = (1 - \pi_{\text{inact}})(1 - \pi_{\text{fail\_pct}})(1 - \pi_{\text{fail\_ack}})$$

denote this probability, where π_{inact} is the probability that a peer is inactive, $\pi_{\text{fail\_pct}}$ is the probability that a lookup message is lost in transit, and $\pi_{\text{fail\_ack}}$ is the probability that the associated acknowledgment is lost. We assume that all three failure events are stochastically independent. Let us further define three random variables A, B, C where $\Pr[A = i]$ denotes the probability that forwarding the lookup message is unsuccessful $(i - 1)$ times and the next attempt is successful. A forwarding peer can try to contact remote peers indefinitely. So we get

$$\Pr[A = i] = \pi_{\text{suc}}(1 - \pi_{\text{suc}})^{i-1},\ i > 0. \tag{2.1}$$

However $\Pr[A = i]$ gives valid results only as long as the forwarding peer is still active at the i^{th} attempt. Let $\Pr[B > x \cdot t_{\text{out}}]$ denote the probability that a forwarding peer is still active when $x \cdot t_{\text{out}}$ time units have lapsed and t_{out} is the maximum time a peer waits for an acknowledgment to arrive. We further assume peers' lifetime is exponentially distributed and has a mean of λ^{-1} time units. This simplifies our calculation of the probability of being active. The simplification arises

from the fact that the exponential distribution is memoryless, i.e., $\Pr[B > x + t \mid B > t] = \Pr[B > x]$. So computing a valid peer lifetime probability is possible even if the peer is active for quite some time. However, recent investigations of the GNUTELLA file-sharing network showed peer lifetimes to be Pareto distributed [3]. This distribution is not memoryless and has infinite variance if its shape parameter $k \leq 1$. But it can be roughly approximated by it for small values of x. So we obtain

$$\Pr[B > xt_{\text{out}}] = 1 - (1 - e^{-\lambda x t_{\text{out}}}) = e^{-\lambda x t_{\text{out}}} \ .$$

We know for sure that a peer must have been active at the first attempt because otherwise a lookup could not have been routed onwards. Let $\Pr[C = i]$ denote the joint probability that both a lookup message can be forwarded at exactly the i^{th} attempt and the sending peer has a lifetime greater than $(i-1)t_{\text{out}}$ (is still active), i.e.,

$$\begin{aligned}\Pr[C = i] &= \Pr[(A = i) \cap (B > (i-1)t_{\text{out}})] \\ &= \Pr[A = i \mid B > (i-1)t_{\text{out}}] \cdot \Pr[B > (i-1)t_{\text{out}}] \\ &= \Pr[A = i] \cdot \Pr[B > (i-1)t_{\text{out}}] \ ,\end{aligned} \tag{2.2}$$

where the second equality follows from the definition of conditional probability and the last equality follows from independence of both events A, B.

Next we give an equation for the probability that at least one of n possible attempts succeeds. Let C_i be the random variable that denotes the i^{th} attempt. This leads to the probability

$$\Pr[C \leq n] = \Pr\left[\bigcup_{i=1}^{n} C_i\right] = \sum_{i=1}^{n} \Pr[C_i]$$

where n is the number of known remote peers the forwarding peer can try. Since $\forall i, j : i \neq j : C_i \cap C_j = \emptyset$, the last equality follows. This completes the derivation of the probability of a successful routing step. However, a complete routing path consists of at least one routing step and we want to know what is the likelihood of a lookup message to get lost on a path of length d. The probability of this event, D, to happen is

$$\Pr[D = d] = 1 - \Pr[C \leq n]^d \ .$$

Now let's have a look at some figures displaying the probability of a lookup to get lost. We would really have loved to underpin our theoretical findings with simulation results from our P2P data store simulator, however, we were not able to obtain simulation results due to technical restrictions pertaining to the simulator. Most important of these are

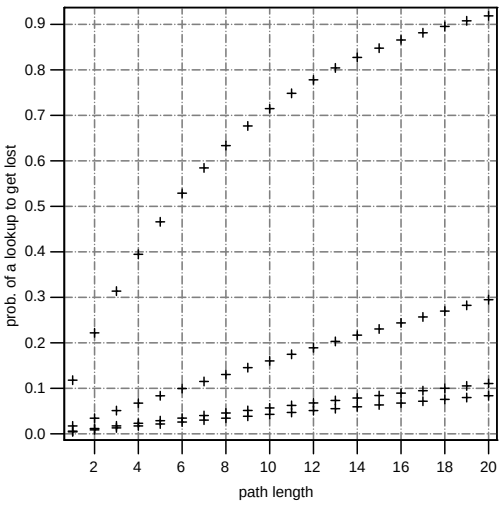

Figure 1. Dependency of number of remote peers on probability of lookup loss. $\lambda^{-1} = 2,\ \pi_{\text{inact}} = 0.1,\ \pi_{\text{fail\_pct}} = \pi_{\text{fail\_ack}} = 0.01,\ t_{\text{out}} = 1/15$

1. For the possibility of a lookup message to get lost, it is necessary that crash/leave events can occur simultaneously with lookup events on the same peer. Since our simulator is activity-based, i.e, each peer runs in its own thread, executes requests in a FIFO fashion without being interrupted by other events, both event types cannot occur simultaneously. The use of virtual parallelism, i.e., multi-tasking, where we cannot control the task scheduling and the amount of time each task is allowed to run aggravates the problem.

2. In case a lookup message is to be duplicated, either new threads must be created for each duplicate or the duplication event must be put into the simulation queue. However, our simulator currently only supports peer threads that get activity starts from the simulation queue. Furthermore, peer threads are not allowed to modify the simulation queue because otherwise statistical parameters which are computed when setting up all events of a simulation phase would be invalidated.

For all of the experiments we set the time unit to 30 seconds. The first figure shows the dependency on the number of remote peers, n. From top to bottom we show curves for $n \in \{1, 2, 3, 4, 100\}$. Curves for $n \in \{4, 100\}$ almost overlap so that they virtually appear the same. That means, for a peer to select from

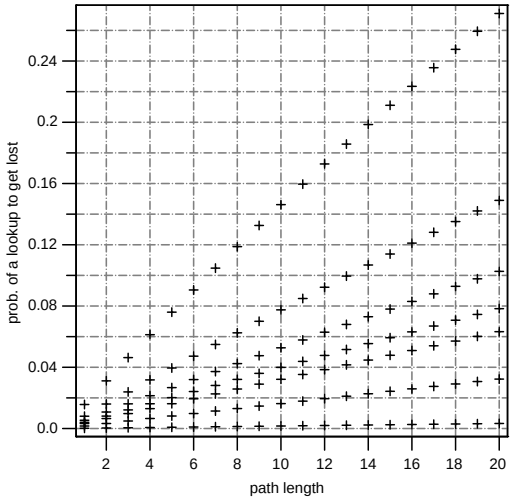

Figure 2. Dependency of λ on probability of lookup loss. $n = 10$, $\pi_{\text{inact}} = 0.1$, $\pi_{\text{fail\_pct}} = \pi_{\text{fail\_ack}} = 0.01$, $t_{\text{out}} = 1/15$

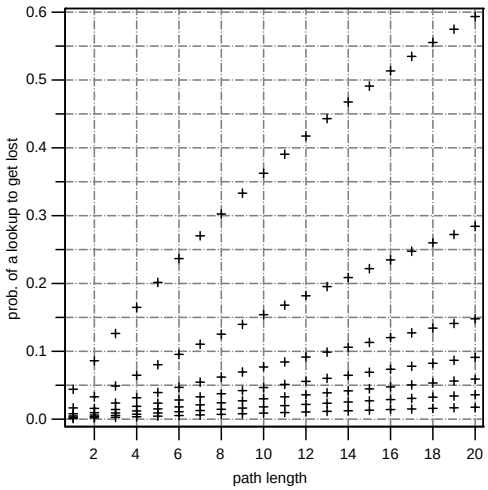

Figure 3. Dependency of π_{inact} on probability of lookup loss. $n = 10$, $\lambda^{-1} = 10$, $\pi_{\text{fail\_pct}} = \pi_{\text{fail\_ack}} = 0.01$, $t_{\text{out}} = 1/15$

between 4 or 10 peers does not make that difference. So we fix $n = 4$ for the next computations.

Figure 2 shows what happens if let $\lambda^{-1} \in \{1, 2, 3, 4, 5, 10, 100\}$. The topmost curve refers to $\lambda^{-1} = 1$ time units and the bottommost curve to $\lambda^{-1} = 100$ time units. As was to expect, the longer a peer is active on average, the less likely it is that lookups will die. Figure 3 displays probability curves when varying π_{inact}. The bottommost curve refers to $\pi_{\text{inact}} = 0.1$ whereas the topmost curve refers to $\pi_{\text{inact}} = 0.7$. Even if a forwarding peer cannot contact 40% of remote peers of the next routing step, the probability of a lookup to fail completely in a P2P data store of $2^{20} = 1.048.576$ nodes remains below 10%. However, this situation changes dramatically if $\pi_{\text{inact}} > 0.7$. The chances of lookups to fail completely easily exceed 40% even if the P2P data store contains much fewer ($2^{12} = 4096$) nodes.

Additional experiments could vary $\pi_{\text{fail\_pct}}$, $\pi_{\text{fail\_ack}}$, and t_{out}. We refrain from showing figures on these variations because they do not give any additional insights. Varying both network failure probabilities, for instance, would result in similar figures compared to the ones showed. Parameter t_{out} is the upper bound on the time a peer waits for an answer to its lookup request. Since all peers keep track of round trip times (RTTs) to remote peers and adapt their timeouts to these RTTs, real timeout values will be much smaller on average thus leading to better lookup failure probabilities.

Finally, a few numbers on how many lookups having path length d may fail on average are in order. Clearly, the expected value is $E[D] = \Pr[D = d]$ because we fix d and thus can take D as a Bernoulli distributed random variable. Letting $n = 10$, $\lambda^{-1} = 10$, $\pi_{\text{fail\_pct}} = \pi_{\text{fail\_ack}} = 0.01$, $t_{\text{out}} = 1/15$, $\pi_{\text{inact}} = 0.1$, and the path length produced by a lookup $d = 15$, we obtain a ratio of $1 : 75.6$ lookup failures on average. For $\pi_{\text{inact}} = 0.4$ we get $1 : 14.5$, for $\pi_{\text{inact}} = 0.6$ we get $1 : 4.5$, and for $\pi_{\text{inact}} = 0.8$ we get $1 : 1.1$, which means a lookup will be lost almost certainly. If we raise the number of potential remote peers that can be contacted from 10 to 100 and let $\pi_{\text{inact}} = 0.8$, we get a ratio of $1 : 3$ which is nearly three times better.

3. Likelihood of lookup duplication

Another interesting question is what is the probability of a lookup message being duplicated on its path from source to destination peer? A duplicate answer is not really a problem but it contributes to bandwidth consumption which should be kept as little as possible since it provides no benefit. If one or more duplicate answer(s) arrive(s) nevertheless, the source peer drops all answers that arrive after the first one. Here we consider duplicates that result from network packet loss only for the reason that the mathematical derivation does not get too complex. Of course, packets could

also be delayed by the network. An acknowledgment, for example, that arrives after it has timed out on the forwarding peer leads to a lookup message duplication with high probability provided the forwarding peer remains active for the next routing attempt(s) and there is at least one remote peer among the contacted peers that is still active.

First, we shall give the probability of the event that a lookup message duplication at any one routing attempt occurs.

$$\pi_{\text{dup}} = (1 - \pi_{\text{inact}})(1 - \pi_{\text{fail\_pct}})\pi_{\text{fail\_ack}} \ ,$$

i.e., the remote peer contacted is active, the lookup message was not lost on its way to the remote peer, but the acknowledgment sent back to the forwarding peer is lost due to some network error. Notice that this event is a subset of the unsuccessful routing attempt event whose probability is $1 - \pi_{\text{suc}}$. Let F be the event of having exactly $l \leq A - 1$ lookup message duplications. Recall that random variable A from equation 2.1 denotes the number of attempts that were necessary to forward the lookup message. There is no discrimination between remote peers. So a remote peer may be contacted again even if sending a lookup message to it was unsuccessful. This allows to keep our derivation simple. So a remote peer can be contacted multiple times and thus A has no upper bound. Random variable F follows a mixture distribution because two random variables combine to form a new one. This is also referred to as a hierarchical model. Therefore, if we let F = number of lookup message duplications and A = number of attempts needed, we have $F|A \sim \text{binomial}(A - 1, \pi_{\text{dup}})$ and $A \sim \text{geo}(\pi_{\text{suc}})$. By $F|A$ we mean that F is conditional on A given $A = i$ is $\text{geo}(\pi_{\text{suc}})$ and F's probability is given by

$$\begin{aligned}
&\Pr[F = l] = \sum_{i=0}^{\infty} \Pr[F = l, A = i + 1] = \\
&\sum_{i=0}^{\infty} \Pr[F = l \mid A = i + 1] \Pr[A = i + 1] = \\
&\sum_{i=l}^{\infty} \left[\binom{i}{l} {\pi_{\text{dup}}}^{l} (1 - \pi_{\text{dup}})^{i-l} \right] \left[\pi_{\text{suc}} (1 - \pi_{\text{suc}})^{i} \right] \ ,
\end{aligned}$$

where the second equality follows from the definition of conditional probability and the third equality follows because $\Pr[F = l] = 0$ if $i < l$. We refrain from giving lengthy algebraic manipulations and simplify the last expression with the aid of Maple [2] and get

$$\Pr[F = l] = \frac{{\pi_{\text{dup}}}^{l} \pi_{\text{suc}} (1 - \pi_{\text{suc}})^{l}}{((1 - \pi_{\text{suc}})\pi_{\text{dup}} + \pi_{\text{suc}})^{l+1}} \ .$$

Notice $\Pr[F = l]$ is only depended on the number of duplicates and duplication and success probabilities. We also give the expected value and the variance of F because we need them later on. Recalling equation 5, we have $\text{E}[A] = 1/\pi_{\text{suc}}$ as the mean number of routing attempts and $\text{VAR}[A] = (1 - \pi_{\text{suc}})/{\pi_{\text{suc}}}^2$ as the variance. For obtaining $\text{E}[F]$, we use the fact that if A and B are any two random variables, then $\text{E}[A] = \text{E}[\text{E}[A|B]]$, provided that the expected values exist (theorem 4.4.1 in [1]). Consequently,

$$\begin{aligned}
\text{E}[F] &= \text{E}[\text{E}[F|A]] \\
&= \text{E}[(A - 1)\pi_{\text{dup}}] \\
&= \frac{\pi_{\text{dup}}(1 - \pi_{\text{suc}})}{\pi_{\text{suc}}} \ ,
\end{aligned}$$

and (theorem 4.4.2 ibid.)

$$\begin{aligned}
\text{VAR}[F] &= \text{E}[\text{VAR}[F|A]] + \text{VAR}[\text{E}[F|A]] \\
&= \text{E}[(A - 1)\pi_{\text{dup}}(1 - \pi_{\text{dup}})] + 0 \\
&= (\text{E}[A] - 1)\pi_{\text{dup}}(1 - \pi_{\text{dup}}) \\
&= \frac{\pi_{\text{dup}}(1 - \pi_{\text{dup}})(1 - \pi_{\text{suc}})}{\pi_{\text{suc}}} \ .
\end{aligned}$$

It is interesting to know how real values of F deviate from its expected value. We first observe: the greater π_{suc} the smaller $\text{VAR}[A]$ and thus the sharper the concentration of values of A around the mean. Second, when plotting $\text{VAR}[F]$, one notices the strong influence of $\pi_{\text{fail\_ack}}$ keeping all other failure probabilities fixed. $\text{VAR}[F]$ grows exponentially if $\pi_{\text{fail\_ack}}$ grows linearly. That means, $\pi_{\text{fail\_ack}}$ should be kept as small as possible in order to obtain predictions that correspond well with real world observations.

However, some words of caution are in order. A calculation of probabilities of event F is only valid if the presupposition that the forwarding peer is continuously active throughout i routing attempts holds. If we so assume, we can exclude peer lifetimes from consideration. On the other hand, if a forwarding peer dies at the j^{th}, $j < i$ attempt but the lookup message makes it to an active remote peer at the j^{th} attempt, we treat this event no different than a successful routing step. We assume the probability of at least two duplication events, where one of them is taken as the successful routing event and the other(s) are lookup message duplication events, to be rather negligible.

What if we wanted to know the probability of having at least x duplicates per routing step? Use of basic probability properties leads to the following equation

$$\Pr[F \geq l] = 1 - \sum_{j=0}^{l-1} \Pr[F = j] \ . \tag{3.1}$$

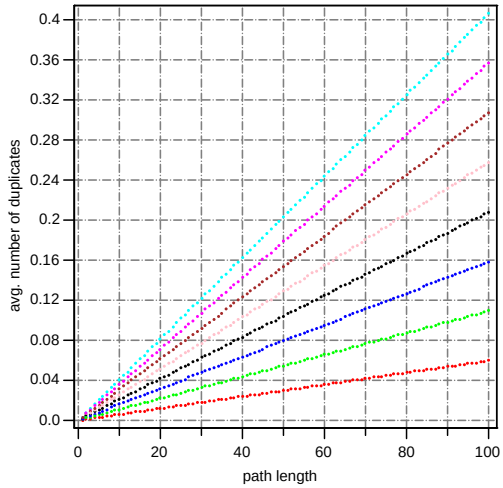

Figure 4. Average number of lookup message duplicates varying π_{inact}. $d \in [1..100]$, $\pi_{\text{inact}} \in [0.1..0.8]$, $\pi_{\text{fail\_pct}} = \pi_{\text{fail\_ack}} = 0.01$

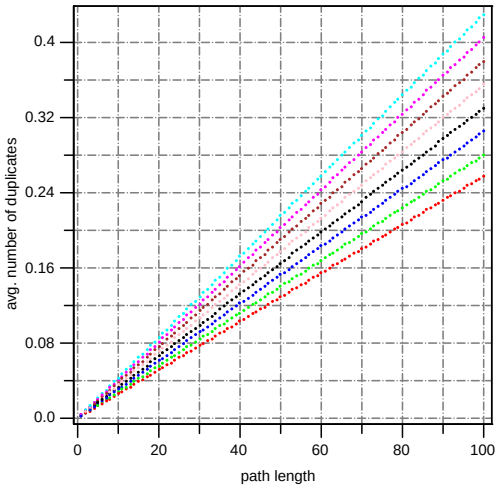

Figure 5. Average number of lookup message duplicates varying $\pi_{\text{fail\_pct}}$. $d \in [1..100]$, $\pi_{\text{inact}} = 0.5$, $\pi_{\text{fail\_pct}} \in [0.01..0.7]$, $\pi_{\text{fail\_ack}} = 0.01$

We showed in the original paper (Theorem 1) that a routing path consists of X routing steps each requiring A_r routing attempts to complete successfully. Again, X is a random variable and $1 \leq r \leq X \leq d$ and d is the number of dimensions of the hypercube. Let us define random variable G that is the sum of all duplicates generated on a path of length X.

$$G = \underbrace{F_1 + F_2 + \ldots + F_X}_{X \text{ terms}} + \underbrace{F_1F_2 + \ldots + F_1 \cdots F_X}_{X-1 \text{ terms}} = \sum_{s=1}^{X} F_s + \sum_{r=2}^{X} \prod_{j=1}^{r} F_j \ . \tag{3.2}$$

Random variables F_j, $1 \leq j \leq X$ follow the definition of F but pertain to each routing step j. Two facts are worth pointing out: First, the original lookup message generates $\sum_{s=1}^{X} F_s$ duplicates on a path of length X. Certainly, duplicates in turn can generate additional duplicates on the remaining routing path and these sum to $\sum_{r=2}^{X} \prod_{j=1}^{r} F_j$, where r is the number of routing steps performed successfully so far. Second, G also follows a mixture distribution across three stages of hierarchy (random variables A, F, and X are involved) and has new random variables like F_1F_2, for instance, that arise from a convolution of F_1 with F_2. Unfortunately, due to the complexity of G, we could not derive a closed-form solution.

So we resort to looking at the expected value of G. Recall that the average length of a routing path in a hypercube of dimension d is $\mathrm{E}[X] = d/2$ (cf. equation 2 in the original paper). By using theorem 4.4.1 from [1] again, we obtain

$$\begin{aligned}
\mathrm{E}[G] &= \mathrm{E}[\mathrm{E}[G|X]] = \mathrm{E}\left[\sum_{s=1}^{X} F_s + \sum_{r=2}^{X} \prod_{j=1}^{r} F_j\right] \\
&= \mathrm{E}\left[\sum_{s=1}^{X} F_s\right] + \mathrm{E}\left[\sum_{r=2}^{X} \prod_{j=1}^{r} F_j\right] \\
&= \frac{d}{2}\mathrm{E}[F] + \mathrm{E}\left[\frac{\mathrm{E}[F]^{X+1} - \mathrm{E}[F]^2}{\mathrm{E}[F] - 1}\right] \\
&= \frac{d\pi_{\text{dup}}(1-\pi_{\text{suc}})}{2\pi_{\text{suc}}} + \\
&\mathrm{E}\left[\frac{(\pi_{\text{dup}}(1-\pi_{\text{suc}})/\pi_{\text{suc}})^{X+1} - (\pi_{\text{dup}}(1-\pi_{\text{suc}})/\pi_{\text{suc}})^2}{\pi_{\text{dup}}(1-\pi_{\text{suc}})/\pi_{\text{suc}} - 1}\right] \\
&= \frac{d\pi_{\text{dup}}(1-\pi_{\text{suc}})}{2\pi_{\text{suc}}} + \\
&\frac{(\pi_{\text{dup}}(1-\pi_{\text{suc}})/\pi_{\text{suc}})^{d/2+1} - (\pi_{\text{dup}}(1-\pi_{\text{suc}})/\pi_{\text{suc}})^2}{\pi_{\text{dup}}(1-\pi_{\text{suc}})/\pi_{\text{suc}} - 1} \ .
\end{aligned}$$

The fourth equality follows because $\forall i, j : \mathrm{E}[F_i] = \mathrm{E}[F_j] = \mathrm{E}[F]$.

The following figures show how the expected value of G changes if we vary one parameter and keep the others fixed. First of all, figure 4 displays the influ-

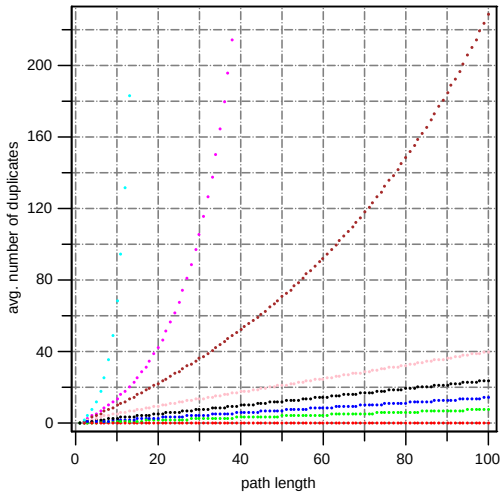

Figure 6. Average number of lookup message duplicates varying $\pi_{\text{fail\_ack}}$. $d \in [1..100]$, $\pi_{\text{inact}} = 0.5$, $\pi_{\text{fail\_ack}} \in \{0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.57, 0.6, 0.7\}$, $\pi_{\text{fail\_pct}} = 0.01$

ence of π_{inact} on the average number of duplicates. We set $\pi_{\text{inact}} \in [0.1..0.8]$, where the topmost line refers to 0.8 and the bottommost line refers to 0.1. Contrary to intuition, the greater the probability of a peer being inactive the greater the probability of a duplication event. This is because $\mathrm{E}[A]$ grows faster than π_{dup} shrinks when increasing π_{inact}. That means the number of "good" events, i.e., a lookup message reaches the receiver and the latter is active, grows thus leading to a greater number of duplications. Also notice, a hypercube with dimension $d > 90$ and $\pi_{\text{inact}} \geq 0.8$, $\pi_{\text{fail\_pct}} = \pi_{\text{fail\_ack}} = 0.01$ is needed so that three lookups generate at least one duplicate on their path through the hypercube on average.

What happens, if we vary $\pi_{\text{fail\_pct}}$ from 0.01 to 0.7? Figure 5 shows the result, where the bottommost line corresponds to the least value and the topmost line corresponds to the greatest value of $\pi_{\text{fail\_pct}}$. We note two observations. First, varying $\pi_{\text{fail\_pct}}$ from 0.01 to 0.7 does not influence the average number of duplicates as much as varying π_{inact} does. Second, a high values of $\pi_{\text{fail\_pct}}$ and a medium value of π_{inact} (topmost line) do not increase the average number of duplicates significantly. However, this changes dramatically - as we have expected - when we start to vary $\pi_{\text{fail\_ack}}$.

Figure 6 shows the dependency of $\pi_{\text{fail\_ack}}$ on the average number of lookup message duplicates. We can see the effects of varying $\pi_{\text{fail\_ack}}$ very clearly at even small values of d. The bottommost line corresponds to $\pi_{\text{fail\_ack}} = 0.01$ and the topmost line corresponds to $\pi_{\text{fail\_ack}} = 0.7$. All other lines correspond to values between these two. As one see very clearly, there is an exponential increase of the average number of duplicates if $\pi_{\text{fail\_ack}}$ increases linearly. If we set $\pi_{\text{fail\_ack}} = 0.57$ and $d = 20$, we can see that > 20 lookup message duplicates are generated on average that is at least one per routing step. If, for example, an adversary is able to suppress slightly more than half of the acknowledgments of lookup requests, it is very likely that duplicates of these requests flood the P2P network consuming huge amounts of bandwidth thus making data item manipulation very slow and or even impossible. This alludes to a varied Denial-of-Service scenario in which an attacker drops packets instead of swamping the P2P network with huge amounts of packets.

References

[1] G. Casella and R. L. Berger: *Statistical Inference*, 1st ed., Brooks/Cole Publishing Company, 1990

[2] A. Heck: *Introduction to Maple*, 3rd ed., Springer-Verlag, New York, 2003

[3] D. Stutzbach and R. Rejaie: *Towards a Better Understanding of Churn in Peer-to-Peer Networks.*, Technical Report CIS-TR-04-06, University of Oregon, 2004