# On the Distribution of Nodes in Distributed Hash Tables

Heiko Niedermayer, Simon Rieche, Klaus Wehrle, Georg Carle

Wilhelm-Schickard-Institute for Computer Science
University of Tübingen
*lastname*@informatik.uni-tuebingen.de

**Abstract:** We develop a model for the distribution of nodes in ring-based DHTs like Chord that position nodes randomly or based on hash-functions. As benefit of our model we get the distribution of interval sizes and an approximation of the distribution of data load among the participating peers. The results match previously published experimental studies on load distribution that motivated the research on load balancing. Most importantly, our model also gives a theoretical explanation for observed load behavior.

## 1 Introduction

A major purpose of Peer-to-Peer (P2P) systems is the management of large amounts of data distributed across many systems. Distributed Hash Tables (DHT) are designed for a highly scalable, self-organizing and efficient distribution and lookup of data. However, performance and fairness among the peers is not guaranteed and it depends heavily on the equal distribution of nodes and data in the identifier space. This is also true for the distribution of routing table entries as well as for the lengths of routing paths.

### 1.1 Related Work

The distribution of nodes in the identifier space is important for the distribution of load. Recently, many papers reported an unbalanced load and motivated research in balancing load as important for the efficiency of DHTs [RLS+03, BCM03, RPW04]. All authors showed by simulation or experiment that the distribution of data load in DHTs is not satisfying, without giving an explanation for this fact. Usually, these load balancing approaches react to an uneven distribution of the data. An exception is the approach in [KR04] which proposes to change the process of assigning IDs to nodes.

## 2 Modelling Node Distribution

In this section we model the distribution of nodes in a DHT and provide a theoretical explanation for the observed distribution of load.

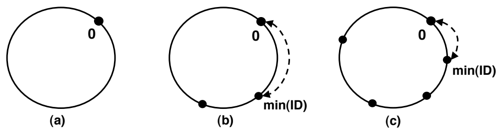



Figure 1: Interval size as minimum of IDs of all other nodes.

### 2.1 Distribution of Nodes

The distribution of nodes in a DHT heavily depends on the way node IDs are assigned. The proposal to use a hash of the IP address (160 bit SHA-1 in case of Chord) has two implications. First, the positioning can hardly be distinguished from a random ID assignment with all values equally likely. Second, the node ID assignment for one node is independent from the node ID assignments for other nodes. As a consequence, the first assumption of our model is that the ID of a node is a random variable following the uniform distribution. Second, the node IDs are considered to be independent. The DHT consists of $n$ nodes and without loss of generality, we assume that the node we look at has ID = 0 (Fig. 1(a)). As a consequence, the interval is determined as the minimum over the node IDs of the other nodes (Fig. 1(b-c)). To determine the distribution of the interval size we assume $n-1$ experiments (IID, uniform random variable) and take the minimum of these experiment.

### 2.2 Continuous Model

First, we use a Continuous Model, i.e. we consider the ID space to be real-valued in the interval $[0,1)$. The motivation for this continuous model is that it is easier than a discrete one and the ID space is large compared to the number of nodes in it.

The continuous uniform distribution is defined as:

$$U(x) = \begin{cases} 0 & x < 0 \\ x & 0 \leq x < 1 \\ 1 & x \geq 1 \end{cases}$$

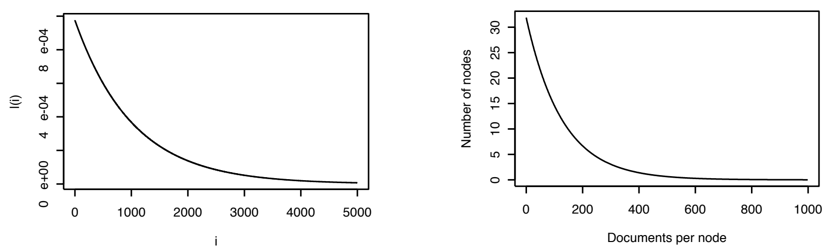
Let $L$ be the distribution of the interval size. It is given as minimum of $n-1$ experiments [**?**] :

$$L(x) = 1 - \prod_{i=1}^{n-1} (1 - U(x)) = 1 - (1 - U(x))^{n-1} = \begin{cases} 0 & x < 0 \\ 1 & x \geq 1 \\ 1 - (1-x)^{n-1} & else \end{cases}$$

And the probability density function:

$$l(x) = \frac{dL}{dx} = \begin{cases} (n-1)(1-x)^{n-2} & 0 \leq x < 1 \\ 0 & else \end{cases}$$

As expected, for $n = 1$ (only one node) the interval size is 1, i.e. the complete ID space. For $n = 2$ (2 nodes) the interval size is distributed with U(0,1). With more and more nodes in the ID space the interval size is approaching 0.




(a) Probability of Interval Size for Discrete Model with 4,096 nodes and ID space $[0, 2^{22} - 1]$.

(b) Load Distribution (mean load = 128; 4,096 nodes; parameters like in Figure 3) approximated with a scaled probability function from the Discrete Model.

Figure 2: Probability Distribution and Load

### 2.3 Discrete Model

In this section we transform our continuous model into a discrete one. The only difference is that we consider the ID space to be integers in the interval $[0, 2^m - 1]$.

The discrete uniform distribution is defined as:

$$U(i) = \begin{cases} 0 & i < 0 \\ \frac{i}{2^m} & 0 \leq i < n \\ 1 & i \geq 2^m \end{cases}$$

The probabilities can be derived similarly to the PDF of the continuous model:

$$l(i) = \left(1 - \frac{i}{2^m}\right)^{n-1} - \left(1 - \frac{i+1}{2^m}\right)^{n-1}$$

When we further solve this equation we can show that like $l(x)$ $l(i)$ is a polynomial of degree $n - 2$. Fig. 2(a) shows $l(i)$, i.e. the probabilities for intervals of size i.

### 2.4 Load Distribution

The load of a node is usually defined as the number of items stored on a node. All the load balancing papers in our references define load in this way [RLS$^+$03, BCM03, RPW04]. This is motivated by the fact that DHTs usually only store pointers to data items and not the items themselves. With respect to the positioning of data items, we can use the same argument as in the Section 2, i.e. that the output of a hash function like SHA-1 is similar to a uniform random number. Thus, when we assume that all intervals are of equal size, the load distribution is given by the binomial distribution. However, we have seen in the last section that this is not the case. But we can divide the intervals into equal-sized smaller

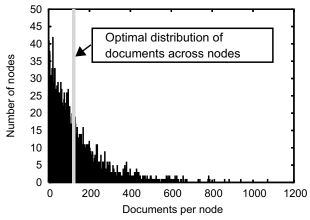



Figure 3: Measured Load Distribution according to [RPW04]

intervals and let each node own a number of intervals depending on its original interval size. The load distribution is then given as a sum of binomial random variables. The number of binomial random variables is given by the distribution in the previous section.

However, to shorten the analysis we approximated the load distribution with the interval size distribution of the Discrete Model in Section 2.3. Figure 2(b) is a frequency distribution based on this approximation. In the formula of the model, $n$ is then the number of documents and the size of the ID space in the model (originally $2^m$) is then $n * \frac{n}{N}$ with $N$ being the number of nodes. $\frac{n}{N}$ is the mean load per node. We used the same parameters as for the measured load given in Figure 3. Measured load distribution and the distribution of the model are almost identical, which supports our assumption.

## 3 Conclusions

In this paper we determined the distribution of interval sizes in ring-based DHT approaches by modeling the standard joining process. These models also provide a theoretical explanation for the unequal distribution of load that has been reported in recent publications and allow to predict the degree of imbalance in given scenarios.

## References

[BCM03] John Byers, Jeffrey Considine, and Michael Mitzenmacher. Simple Load Balancing for Distributed Hash Tables. In *Proceedings of 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, Berkeley, USA, February 2003.

[KR04] David Karger and Mathias Ruhl. Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems. In *Proceedings of 4th International Workshop on Peer-to-Peer Systems (IPTPS '04)*, San Diego, USA, February 2004.

[RLS+03] Ananth Rao, Karthik Lakshminarayanan, Sonesh Surana, Richard Karp, and Ion Stoica. Load Balancing in Structured P2P Systems. In *Proceedings of 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, Berkeley, USA, February 2003.

[RPW04] Simon Rieche, Leo Petrak, and Klaus Wehrle. A Thermal-Dissipation-based Approach for Balancing Data Load in Distributed Hash Tables. In *Proc. of IEEE Conference on Local Computer Networks. (LCN 2004)*, Tampa, FL, USA, November 2004.