Diversity Entails Improvement: A New Neighbour Selection Scheme for Kademlia-type Systems

Hani Salah[†] Stefanie Roos[‡] Thorsten Strufe[‡]
[†]TU Darmstadt, Germany [‡]TU Dresden, Germany
hsalah(at)cs.tu-darmstadt.de, firstname.lastname(at)tu-dresden.de

***Abstract*—Discovery of nodes and content in large-scale distributed systems is generally based on Kademlia, today. Understanding Kademlia-type systems to improve their performance is essential for maintaining a high service quality for an increased number of participants, particularly when those systems are adopted by latency-sensitive applications.**

This paper contributes to the understanding of Kademlia by studying the impact of *diversifying* neighbours' identifiers within each routing table bucket on the lookup performance. We propose a new, yet backward-compatible, neighbour selection scheme that attempts to maximize the aforementioned diversity. The scheme does not cause additional overhead except negligible computations for comparing the diversity of identifiers. We present a theoretical model for the actual impact of the new scheme on the lookup's hop count and validate it against simulations of three exemplary Kademlia-type systems. We also measure the performance gain enabled by a partial deployment for the scheme in the real KAD system. The results confirm the superiority of the systems that incorporate our scheme.

***Index Terms*—Kademlia; Lookup Performance; Performance Improvement; Markov Chain; Formal Routing Analysis**

I. Introduction

The family of Kademlia-type [1] systems represents the most widely used and deployed type of Distributed Hash Tables (DHT). These systems are adopted, for the discovery of nodes and content, by several peer-to-peer (P2P) file-sharing applications, accounting for multi-million users today [2], [3], like Bittorrent and eMule. Implementations of Kademlia have been also experimented as communication overlays in video streaming applications [4] and botnets [5], [6].

The routing (also called *lookup*), the key operation in these systems, has received a great deal of attention from the research community in the last years. Some studies (e.g. [7]–[13]) identified limitations in the *standard* system designs, thus raised doubts about their suitability, particularly, for latency-sensitive applications. The authors accordingly proposed modifications to the standard designs in order to improve the system performance. Considered performance metrics are the hop count, the lookup latency, and the overhead in terms of the number of sent messages.

We aim to further improve the performance based on concrete theoretical models. In contrast to the above studies, we do not focus on optimizing the parameters governing routing table structure and lookup mechanism. Rather, the goal is to develop a low-overhead scheme, which can be integrated easily into all existing designs.

Towards this end, we study a previously disregarded lookup performance factor – the diversity of neighbours' identifiers within each routing table bucket. Consequently, we propose, model, and evaluate a new neighbour selection scheme that attempts to maximize this diversity. The scheme is compatible not only with the standard Kademlia and its variations, but also with previously proposed improvements. It thus can be combined to any of them in order to improve their achieved performance further. It does not change the standard routing protocol nor the routing table structure, and it does not cause additional communication overhead. Only slight changes in the standard routing table's maintenance processes are required, causing only a negligible extra computational overhead.

Our main contributions can be summarized as follows:

- We propose a new neighbour selection scheme to reduce the average lookup's hop count in Kademlia-type systems, with almost no extra overhead.
- We develop a theoretical model (extending our prior work [13]) to asses the impact of the proposed scheme on the lookup performance.
- We evaluate the scheme using both the model and extensive simulations of three Kademlia-type systems.
- We measure the impact of our scheme on the lookup performance of modified KAD clients in the real KAD system.

The model predictions and simulation results, which agree very closely to each others, show that the new scheme improves the lookup performance in form of reduction in the average hop count, and thus in the number of sent messages. The improvement applies also for the measurement results of the KAD clients that incorporate our scheme.

The remainder of this paper is structured as follows: We give an overview of Kademlia and its variations in Sec. II, and then discuss the related work in Sec. III. Next, Sec. IV presents an overview of our proposed scheme, Sec. V describes our model, and Sec. VI discusses the results. Finally, Sec. VII concludes the paper.

II. Kademlia-type Systems

Kademlia [1] is a structured peer-to-peer (P2P) system. It uses a b-bit identifier space from which the identifiers of nodes and objects are assigned. Nodes store key-value (key-object) pairs, such that the nodes at the closest distance to an identifier are responsible for storing it. The distance between two identifiers is defined as the XOR of their values.

978-1-4799-6201-3/14/$31.00

Each node v stores the identifiers and addresses of other nodes, also called neighbours or contacts[1], in a b-level tree-structured routing table. Each level in the routing table consists of so called k-buckets, such that each bucket stores up to k known contacts that share a common prefix with v's identifier. Contacts that represent nodes which have left the system are called *stale*.

Kademlia implements a key-based routing protocol: To route a message from a node v to a target identifier x, v picks α known contacts that are closest to x and sends them lookup requests in parallel. Every queried contact that is online replies with the set of β contacts that are locally known as being closest to x, thus extending v's set of candidate contacts. This process iterates until no further contacts closer to x are discovered or a timeout is held. The original Kademlia paper suggests $k = 20$ and $\alpha = 3$.

In order to mitigate the effect of churn, each node performs *maintenance* processes for its routing table buckets. In practice[2], two periodic *maintenance processes* are performed: The first process aims to increase the amount of contacts that are stored in the routing table by searching for potential new contacts belonging to low populated buckets. The second process aims to keep the routing information up-to-date by checking if the stored contacts are still responsive and removing stale ones. Long-lived contacts are checked less frequently than newer ones. This preference for old contacts is based on the observation that the longer a contact has been online, the more likely it is to remain online in the future [1], [14].

The above design is the basis for a family of Kademlia-type systems. In this paper, we focus on three of them. The mainline implementation of BitTorrent (MDHT) integrates one of those systems for nodes discovery. In MDHT, the routing table (Fig. 1(a)) includes a single k-bucket per level. uTorrent, the most popular MDHT implementation uses $k = 8$, $\alpha = 4$, and $\beta = 1$ [12].

Considering the fractions of the identifier space that are covered at each routing table level i, Jimenez et al. [12] introduced a variation of MDHT implementing variable bucket sizes (iMDHT) to increase the distance reduction at each hop. The bucket size is chosen to be 128, 64, 32, and 16 for the buckets at levels $i \in (0..3)$ respectively, and 8 for the rest (Fig. 1(b)). Both MDHT and iMDHT use $b = 160$.

KAD, the DHT used by the popular file-sharing application eMule, uses $b = 128$, $k = 10$, $\alpha = 3$, and $\beta = 2$. It implements a different routing table structure: As shown in Fig. 1(c), starting from the fourth level, the routing table includes multiple buckets per level, grouping contacts according to the first $l \in \{3, 4\}$ bits after the first varying bit. Consequently, the difference between the common prefix length of the current hop and the next hop to x, called the *bit gain*, is at least l.

[1]From here onwards, these two terms are used interchangeably.

[2]This is how it is implemented, for example, in the eMule software: http://www.emule-project.net.

III. Related Work

Motivated by the high popularity of Kademlia and its variations, those systems have been the subject of many studies in the past few years. In this Section, we discuss only those studies which focused on the lookup process, or those which proposed improvements for the standard system design.

Crosby and Wallach [11] measured the lookup latency in MDHT and Azureus (the DHT that is used by the Vuze Bittorent client). They reported high latency values, and attributed this to the observed high ratio of stale contacts in the routing tables. Similarly, Stutzbach and Rejaie [8] analysed the lookup process and measured the lookup latency in KAD.

Several studies investigated the possibility to improve the lookup performance. The approach by Falkner et al. [10] adapted the lookup parameters at runtime according to the number of expected lookup response messages. Their design reduced the median lookup latency but at the same time increased the lookup overhead. Steiner et al. [9], in addition to analysing the lookup latency in KAD by evaluating the impact of both external factors (e.g. RTT of lookup messages) as well as internal lookup parameters, achieved an improved lookup latency by coupling the lookup with the content retrieval process. More recently, Jimenez et al. [12] suggested several modifications to MDHT, and achieved better lookup latency with low additional overhead.

A number of other studies succeed to reduce the lookup costs (measured by number of lookup messages or latency) via: caching [16], [17], geographical proximity [18], [19], or recursive lookup [20].

In this paper, we propose to improve the lookup performance, in form of reduction in the average hop count, by adapting the standard neighbour selection scheme. Although the approach differs from earlier improvements, it is orthogonal and hence compatible with them.

The model that we present in this paper extends on our prior model of Kademlia-type systems [13], which allows a very accurate prediction of the routing overhead. Though theoretical analysis of P2P routing performance is widely studied, traditionally only asymptotic bounds have been derived (e.g. [1], [21]–[23]). The few studies deriving exact formulas commonly only consider the average routing length, special cases such as bijective mapping from identifiers to nodes, and are of limited accuracy when compared to measurements or simulations (e.g. [8], [24], [25]). In particular, [24], [25] model P2P routing using a Markov chain approach similar to the one suggested in [13], but are restricted to systems without parallelism.

IV. Improving the Lookup Performance

In this Section, we introduce an approach for improving the lookup performance in Kademlia-type systems: We give an overview of the idea in Sec. IV-A and then validate the main assumption on which it is based (against results obtained from a real Kademlia-type system) in Sec. IV-B. In Sec. IV-C, we describe how the approach can be implemented in practice.

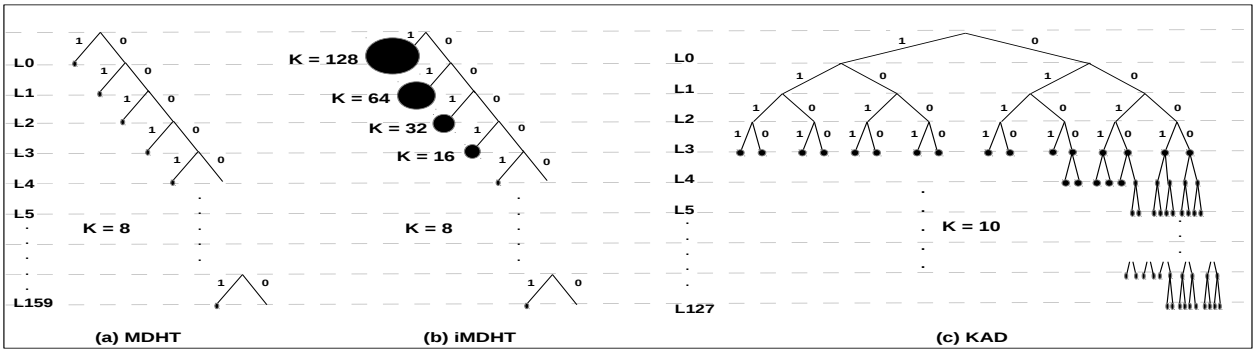


Fig. 1: The routing table structures of three Kademlia-type systems (adapted from: [15]): (a) MDHT, (b) iMDHT, and (c) KAD.

A. Increasing the Diversity

Our approach is based on a new neighbour selection scheme. As described in Sec. II, the k contacts (i.e. neighbours) that each routing table bucket can store share a common prefix. This means that they all belong to a specific range of identifiers, thus represent a certain region of the identifier space. This way, without further restrictions, multiple contacts in the bucket can have very close identifiers (e.g. represent contiguous positions in the respective identifier space region), whereas there are other portions of the region not closely covered by the stored contacts.

By storing a large number of contacts from only one portion of the region, the node's view for the respective identifier space region is narrowed. Consequently, we propose to improve the lookup performance by *widening* this view. That is, the node should try to maximize the *diversify* of the identifiers that are stored in each of its routing table buckets independently, by choosing contacts such that their identifier prefixes are maximally diverse. Then the expected common prefix length of the closest contact to an arbitrary target identifier is maximized, which should lead to a lower average number of hops.

We now show that the expected bit gain, i.e. the difference between the node and the closest contact in its routing table to a target identifier x, is increased by maximizing the aforementioned diversity. A general model of the actual impact on the average hop count is presented in Sec. V.

Theorem 1: Consider a k-bucket such that contacts in the bucket offer a bit gain of at least l. The expected bit gain bg^{div} offered by the closest contact to x in that k-bucket when maximizing the diversity is at least as big as bg^{norm}, the expected bit gain for the standard contact selection.

Proof: The cumulative distribution function (CDF) of the expected bit gain of one contact chosen uniformly at random from all identifiers in the k-bucket is given by

$$F^l(i) = \begin{cases} 0, & i < l \\ 1 - 1/2^{i-l}, & i \geq l \end{cases}$$

because there is a guaranteed improvement of l and the probability for every further bit to agree with the respective bit of x is $\frac{1}{2}$. Note that the CDFs for the maximum of independent random variables $X_1, \ldots, X_m$ with CDFs $F_1, \ldots, F_m$ is $F(x) = 1 - \prod_{i=1}^{m} F_i(x)$. Furthermore, the expected value of a random variable X with values in $\mathbb{N}_0$ and CDF F is $\mathbb{E}(X) = \sum_{i=0}^{\infty} P(X > i) = \sum_{i=0}^{\infty} 1 - F(i)$. The distribution of the maximum bit gain of k contacts when selecting contacts uniformly at random from all nodes suitable for a bucket is thus

$$bg^{norm} = l + \sum_{i=l+1}^{\infty} 1 - F^l(i)^k.$$

When maximizing the diversity of the $q = \lfloor \log k \rfloor$ additional bits, there is one contact for each 2^q bit sequences as well as $k - 2^q$ contacts chosen uniformly at random. Thus, the guaranteed bit gain is q. The closest contact in the routing table is either the one contact guaranteed to agree with x in those q bits or one of the contacts chosen uniformly at random. The CDF of the first is given by $F_A^l(x) = F^l(x - q)$ and the CDF of the maximum bit gain of the contacts chosen uniformly at random is $F_B^l(x) = \left(F^l(x)\right)^{k-2^q}$. Thus, the CDF of the total bit gain is $F_A^l(x) F_B^l(x)$, so that the expected bit gain is given by

$$bg^{div} = l + q + \sum_{i=l+q+1}^{\infty} 1 - F^l(i-q) \left(F^l(i)\right)^{k-2^q}.$$

bg^{div} presents an upper bound on bg^{norm} because

$$\begin{aligned} bg^{norm} &= l + \sum_{i=l+1}^{\infty} 1 - \left(F^l(i)\right)^{2^q} \left(F^l(i)\right)^{k-2^q} \\ &\leq l + q + \sum_{i=l+q+1}^{\infty} 1 - \left(F^l(i)\right)^{2^q} \left(F^l(i)\right)^{k-2^q} \\ &\leq l + q + \sum_{i=l+q+1}^{\infty} 1 - \left(1 - \frac{2^q}{2^{i-l}}\right) \left(F^l(i)\right)^{k-2^q} \\ &= l + q + \sum_{i=q+1}^{\infty} 1 - \left(1 - \frac{1}{2^{i-l-q}}\right) \left(F^l(i)\right)^{k-2^q} \\ &= bg^{div}. \end{aligned}$$

■

Note that Theorem 1 only shows that the expected bit gain of the closest contacts in one node's routing table is increased.

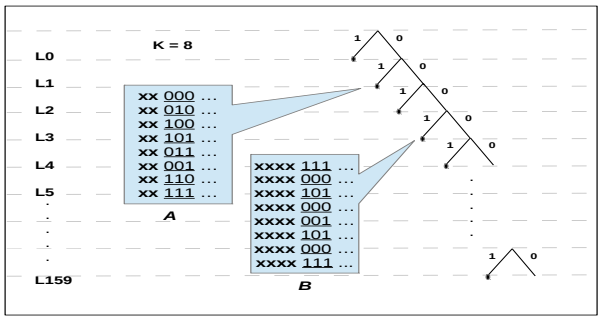


Fig. 2: An exemplary MDHT routing table: Buckets *A* and *B* are located at the second and fourth levels (i.e. levels 1 and 3), thus have common prefixes of lengths 2 and 4, respectively. Considering the first three bits after the common prefix, their respective diversity degrees are 8 (i.e. the maximal value in MDHT) and 4.

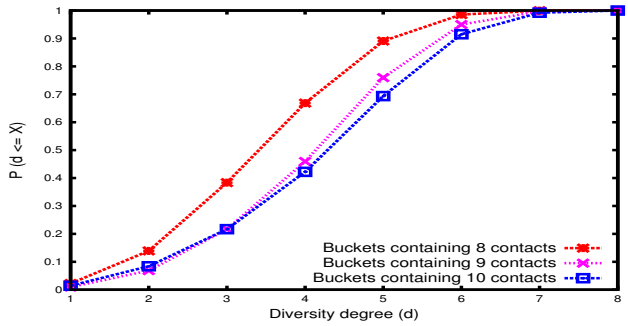


Fig. 3: CDF of the diversity degrees of measured standard KAD buckets.

Lookup parallelism ($\alpha > 1$) and further contacts in the lookup response ($\beta > 1$) are not considered so far. Nevertheless, the above result motivates an in-depth analysis of the contact selection scheme, which we present in the following sections.

From here onwards, we use the term *diversity degree* for a bucket to indicate how diverse are the prefixes of the identifiers stored in it. It is measured by the number of distinct $\lfloor log\ k \rfloor$ bits after the bucket's common prefix, resulting in a maximal degree of $2^{\lfloor log\ k \rfloor}$. That is 3 bits in MDHT and KAD, whereas iMDHT considers 7, 6, 5, and 4 bits for buckets at levels $i \in (0..3)$ respectively, and 3 bits for the rest. Fig. 2 shows an exemplary MDHT routing table: The diversity degrees of buckets *A* and *B* are 8 (i.e. the maximal value in MDHT) and 4, respectively.

Note that the idea above is similar to the KAD's routing table structure, which divides contacts having the same common prefix length with the routing table owner into buckets according to the first 4 bits after the common prefix. However, our approach is more flexible, since it does not restrict the number of prefixes per bucket, but rather selects more diverse prefixes if possible, allowing for a less diverse contact selection if maximal diversity is not achievable.

B. Diversity Degrees in Real Kademlia-type Systems

We here aim to validate our aforementioned assumption that the bucket in standard Kademlia-type systems is likely to store multiple contacts having very close identifiers. Towards this end, we downloaded routing tables of randomly selected online KAD nodes using an accurate KAD crawler [3], and we then analysed the diversity degrees of the contact identifiers contained in their buckets. We restricted our analysis only to the buckets located at the fourth routing table level, for two reasons: First, given the routing table structure of KAD (Fig. 1(c)), those buckets jointly are used, on average, in $\frac{11}{16}$ of the lookup requests, hence represent the most important part of the routing table. Second, there exist with very high probability nodes in the system that can fill those buckets with all possible prefixes, enabling those buckets to achieve higher completeness (as shown in [8]) and higher diversity degrees than buckets at lower levels.

The results that we discuss here represent 1,505,658 buckets. We classify them, by the number of contacts they contain, into three groups: (*i*) 170,262 buckets containing eight contacts each, (*ii*) 271,585 buckets containing nine contacts each, and (*iii*) 1,063,811 buckets containing ten contacts each (i.e. complete buckets). For each group, we computed the CDF of buckets with a diversity degree $\leq m$. Recall that in KAD $m \in (1..8)$.

Fig. 3 shows the results: 42% to 67% of the buckets have $m \leq 4$ (i.e. half the maximal degree), and at most 8% have the maximal degree. These results confirm our aforementioned assumption about diversity degrees in the real systems.

C. Implementation

The aforementioned idea to improve the lookup performance can be implemented in real Kademlia-type systems by only slightly modifying the standard routing table's maintenance processes that we discussed in Sec. II (i.e. modifying the neighbour selection), without changing their frequency. More precisely, when a node decides to find new contacts either to insert them to low-populated buckets, or to replace stale contacts, it selects contacts whose identifiers increase the diversity degrees of the corresponding buckets. This way, the approach does not require changing the original routing table structure nor the routing protocol, and it does not induce any additional overhead, except computing the diversity degrees over the identifiers.

V. Model

In this Section, we analytically derive the hop count distribution in Kademlia-type systems both for the standard contact selection scheme as well as for our modified scheme. More precisely, we determine the probability that the path to find the closest node to a target identifier x is of length h for both schemes.

The model that we present here is based on our previous model of Kademlia-type systems [13]. We chose to extend [13] because it is, to the best of our knowledge, the most comprehensive model of Kademlia routing and it is generic enough to incorporate our changes. In particular, our model extends [13] such that: (*i*) it allows queries for arbitrary identifiers rather than for only node identifiers, and (*ii*) it integrates our modified contact selection scheme. In Sec. V-A,

we review the main components of the original model [13]. In Sec. V-C and Sec. V-D, we describe in detail our modifications to the original model to achieve (*i*) and (*ii*), respectively.

A. Model Principles

In [13], we analytically answered the question: Given an arbitrary target identifier x and a Kademlia-type system, how likely is it to discover the responsible node within h hops for all h?

To this end, we modelled the routing process as a Markov chain, such that states of the chain represent the common prefix length of the α closest nodes with the target identifier x. Due to the prefix-based routing table structure, the common prefix length is sufficient to determine the transition probabilities, i.e. the probability to change from one set of common prefix lengths to either the terminal state of discovering the target or another set of common prefix lengths. The probability to reach the target in h hops is then given by the probability that the Markov chain is in the terminal state after h steps.

Recall that the assignment of a contact to a bucket is made on the basis of the common prefix length with the routing table owner. The corresponding distance function, referred to as the *bit distance*, is

$$dist(x,y) = b - commonprefixlength(x,y). \tag{1}$$

Following [13], we formally characterize a Kademlia-type system by the identifier space size b, and the routing table parameters k (the bucket size) and L (the number of resolved bits per level). The d-th entry k_d of the b-dimensional vector k gives the bucket size for contacts at bit distance d to the routing table owner. For example, in iMDHT we have

$$k_d = \begin{cases} 128, & d = b \\ 64, & d = b-1 \\ 32 & d = b-2 \\ 16, & d = b-3 \\ 8, & d < b-3. \end{cases}$$

The $b \times b$-matrix L determines the number of buckets per level as well as how the identifier space is split among these buckets. So the entry L_{ij} of the matrix $L \in \mathbb{R}^{(b+1)\times(b+1)}$ gives the fraction of the identifiers at bit distance i to the routing table owner for which j additional digits besides the common prefix are considered for deciding the bucket. For example, in KAD, $L_{b4} = 1$, $L_{i3} = 0.75$, and $L_{i4} = 0.25$ for $i < b$. In the context of our probabilistic model, we consider a random variable L_d, whose distribution is given by the d-th row of L. The routing algorithm is modelled for arbitrary α and β in [13]. However, we only consider here the standard values of KAD: $\alpha = 3$ and $\beta = 2$.

We now describe the general idea of the derivation presented in [13] using the above terminology. Kademlia routing is modelled as a Markov chain, so that states correspond to the bit distance of the closest α known nodes to the target identifier x. Routing termination is denoted by the state $\emptyset$. The state space is hence given by

$$S_\alpha = \{\emptyset\} \cup \{(d_1, \ldots, d_\alpha) : d_i \in \mathbb{Z}_{b+1}, d_i \le d_{i+1}\}.$$

The initial distribution I gives the probability that the α closest nodes in the requesting node's routing table have bit distances $d_1, \ldots, d_\alpha$ to x. For any non-terminal state, the transition matrix T gives the probability to get from a set of α contacted nodes with bit distances $d_1, \ldots, d_\alpha$ to either the terminal state or a set of nodes with bit distances $\tilde{d}_1, \ldots, \tilde{d}_\alpha$. The success rate after k hops is obtained by computing $T^{k-1}I$ and choosing the entry corresponding to the terminal state $\emptyset$.

The model is based upon various assumptions: Node identifiers are assumed to be chosen uniformly at random, and queries are blocking, i.e. at each hop exactly α nodes are queried before deciding on the next set of contacts to query. The basic model also assumes a steady-state, no churn system with maximally full buckets. However, it is shown in [13] how to extend the model to dynamic environments. For simplicity, we stick with the basic model in the following.

The essential part of the derivation is to determine the probability distribution of the bit distances of the closest $\gamma \in \{\alpha, \beta\}$ contacts in a node's routing table given the bit distance D from the routing table owner to x. The probability that the bit distances C of the closest contacts corresponds to $s \in S_\gamma$ is given by

$$\begin{aligned} &P(C = s | D = d) \\ &= \sum_{l=0}^{b} P(C = s | X_0 = d, L_d = l) P(L_d = l) \\ &= \sum_{l=0}^{b} P(C = s | X_0 = d, L_d = l) L_{dl}. \end{aligned} \tag{2}$$

The initial distribution, giving the probability of all possible distances of the requesting node, is obtained from Eq. 2 by summarizing over all possible distances, i.e. $P(X_0 = s) = \sum_{d=0}^{b} P(X_0 = s | D = d) P(D = d)$ with $P(D = d) = \frac{2^d - 2^{d-1}}{2^b}$. Similarly, the transition probabilities can be obtained from Eq. 2. There are $\alpha\beta$ returned contacts of which the closest α *distinct* contacts need to be selected. Let R denote the $\alpha\beta$ returned bit distances, Z the set of all possible returned bit distances, and $top_\alpha : Z \to S_\alpha$ the shortest α distances of distinct contacts. Then the transition probability is expressed by

$$\begin{aligned} &P\big(X_1 = s | (X_0 = (d_1, ..., d_\alpha)\big) \\ &= \sum_{z \in Z} P(top_\alpha(z) = s | X_0 = (d_1, ..., d_\alpha), R = z) \\ &\cdot P(R = z | X_0 = (d_1, ..., d_\alpha)) \prod_{j=1}^{\alpha} P\Big(C = z_j | D = d_j\Big). \end{aligned} \tag{3}$$

The only component of the above derivation influenced by our changes is the probability $P(C = s | D = d, L_d = l)$ in Eq. 2. Our first change to the original model [13] (allowing queries to any identifier) affects only the last step of the routing. There

is no guarantee that the bucket covering the region that x belongs to contains one node, whereas under the assumption of maximally full routing tables, this is given when x corresponds to a node identifier. We hence consider the case $C = \emptyset$ both for the standard scheme and for our modified scheme in Sec. V-C. The second change (selecting contacts such that the prefixes in the buckets are maximally diverse) modifies the probability $P(C = s|D = d)$ for all states s. We hence derive $P(C = s|D = d)$ for non-terminal states s in Sec. V-D, distinguishing the case of $\alpha = 3$ and $\beta = 2$.

B. Assumptions and Notations

In this Section, we state our assumptions and notations for the model. Our assumptions are mostly identical to those in [13], so we refer to that publication for an in-depth discussion of their impact.

1) The system is in steady state without churn, failures, and attacks. In particular, there are no stale contacts in the routing tables and nodes do not fail nor do they drop messages. Furthermore, buckets are maximally full, i.e. if a bucket contains $k_1 < k$ values, there are exactly those k_1 nodes in the region the bucket is responsible for.
2) Node identifiers are uniformly and independently distributed over the whole identifier space. Requested identifiers are also chosen uniformly at random from the whole ID space.
3) Routing table entries are chosen independently.
4) The lookup uses strict parallelism, i.e. a node awaits all answers to its queries before sending additional ones.

The following notations are used throughout the derivation:

- C^{norm} and the C^{own} denote the closest contact distributions in both the standard scheme and our modified scheme, respectively.
- α is the degree of parallelism, β the number of returned closest contacts when queried for an identifier, k_d is the bucket size at distance d.
- The number of additional bits considered for the replacement scheme is given by $q_d = \lfloor log\ k_d \rfloor$. Furthermore, we generally drop the index d for k_d and q_d if the conditioning on d is explicit.
- The probability that a binomially distributed random variable with parameters m and p takes value z is denoted by $\mathbb{B}(m, z, p) = \binom{m}{z} p^z (1-p)^{m-z}$.
- The probability that c elements chosen from a set B of size b are chosen from a subset $A \subset B$ of size a is abbreviated by

$$\Xi(a, b, c) = \frac{\binom{a}{c}}{\binom{b}{c}}.$$

- The probability that the γ closest contacts to x within a group of size a within distance d to x have bit distances $\delta_1, \ldots, \delta_\gamma$, is denoted $\Upsilon_\gamma((\delta_1, \ldots, \delta_\gamma), d, a)$ as derived in [13], Eq. 10. The contacts are assumed to be uniformly selected from all nodes within distance at most d.
- The probability that there are no nodes in a region with 2^d identifiers is abbreviated by $em(d) = \left(1 - 2^{d-b}\right)^{n-1}$.

C. Success Probability

We first consider the case $C = \emptyset$ both for the standard scheme and for our maximally-diverse contact selection scheme. The idea is to summarize over the number of possible nodes in the bucket closest to x. If there are $m > 0$ such nodes and at most $l \leq m$ edges to those nodes, the probability that one edge leads to the node responsible for x is l/m.

If $m = 0$, we assume that the responsible node is known to the routing table owner. This assumption does not considerably increase the success probability. The responsible node either belongs to a different bucket on the same level or it is at the same bit distance to x as the routing table owner, thus belonging to a bucket on a higher level. The responsible node is possibly not contained in the routing table if more than k_d nodes are in the region of the bucket that it belongs to. However, the said bucket covers at most the same number of identifiers as the empty bucket that x belongs to. For typical values of k being at least 8, the probability of such an event is barely above 0.0001, as we detail in an extended technical report of the paper [26].

Propositions 1 and 2 give the probability that the node responsible for x is contained in a node's routing table both for the standard and for our modified scheme, respectively. We do not include the proofs of Propositions 1 and 2 here due to space constraints. However, they can be found in [26].

Proposition 1: The probability that a node v at bit distance d to the target identifier x knows the responsible node given the number of further resolved bits L_d is

$$P(C^{norm} = \emptyset | D = d, L_d = l) \approx \tag{4}$$
$$\mathbb{B}(n-1, 0, 2^{d-l-b}) + \sum_{m=1}^{n-1} \mathbb{B}(n-1, m, 2^{d-l-b}) \min\{1, k/m\}$$

when selecting contacts in a bucket uniformly at random from all nodes in the region covered by the bucket.

Proposition 2: The probability that a node v at bit distance d to the target identifier x knows the responsible node given the number of further resolved bits L_d is

$$\begin{aligned} &P(C^{our} = \emptyset | D = d, L_d = l) \\ &\approx \mathbb{B}(n-1, 0, 2^{d-l-b}) + \sum_{m=1}^{n-1} \mathbb{B}(n-1, m, 2^{d-l-b}) \\ &\cdot \left(\mathbb{B}(m, 0, 2^{-q}) \min\{1, k/m\} + \sum_{j=1}^{m} \mathbb{B}(m, j, 2^{-q}) \rho_j \right) \end{aligned} \tag{5}$$

with

$$\rho_j = \frac{1}{j} + (1 - \frac{1}{j}) \sum_{i=1}^{2^q - 1} \frac{\binom{2^q-1}{i}\binom{m-j}{i-1}}{\binom{m-j+2^q-2}{2^q-2}} \quad \min\left\{1, \frac{k + 2^q - i - 1}{\max\{1, m-i-1\}}\right\}$$

when maximizing the diversity of the contact prefixes in the bucket.

D. Maximizing the Diversity

The standard contact selection has been treated in [13], thus we here only consider our modified scheme. We first consider the case that only $\beta = 2$ closest nodes are of interest, and then extend the result to $\alpha = 3$ which is needed to determine the initial distribution.

Consider the case that $k = 2^q$ and there are nodes for all the 2^q prefixes in the bucket. Then there is exactly one contact in the bucket that has bit distance of less than $d-l-q$ to x, and the node is selected uniformly at random from all these identifiers. The second closest contact is then the one that is at bit distance $d-l-q$, i.e. the one for which the last bit of the q-bit prefix is different. The third closest contact is chosen to be at bit distance $d-l-q+1$, not sharing the last two bits of the prefix with x. However, if $k > 2^q$ or there are prefixes with no matching nodes, more than one contact can have a bit distance less than $d-l-q$. Furthermore, the next closest contacts not within distance $d-l-q$ can be farther than for the standard contact selection.

Proposition 3 evaluates how likely these scenarios are, summarizing over all possible number of prefixes without matching nodes for the $\beta = 2$ closest contacts. The evaluation for the case $\alpha = 3$ is analogue to the proof of Proposition 3 and omitted from the paper due to space constraints. It is, however, included in the technical report [26].

Proposition 3: The probability that the two closest nodes to x in the routing table of a node v at bit distance $D = d$ to x are at distances (δ_1, δ_2) to x is

$$P\left(C^{own} = (\delta_1, \delta_2) | D = d, L_d = l, C^{own} \neq \emptyset\right) = 0 \quad (6)$$

if $\delta_1 \geq d-l-q$, and setting $\eta = d-l-\delta_2$

$$\begin{aligned} &P\left(C^{own} = (\delta_1, \delta_2) | D = d, L_d = l, C^{own} \neq \emptyset\right) \\ \approx &\sum_{r=0}^{2^q-1} \mathbb{B}(2^q-1, r, em(\max\{d-l-q-1, 0\})) \\ &\cdot \mathbb{B}(r+k-2^q, 0, 1/(2^q-r)) \\ &\cdot \Upsilon_1((\delta_1), \max\{d-l-q-1, 0\}, 1) \\ &\Xi(r, 2^q-1, 2^{q-\eta}-1) - \Xi(r, 2^q-1, 2^{q-\eta+1}-1) \end{aligned} \quad (7)$$

if $\delta_2 \geq d-l-q > \delta_1$, and

$$\begin{aligned} &P\left(C^{own} = (\delta_1, \delta_2) | D = d, L_d = l, C^{own} \neq \emptyset\right) \\ \approx &\sum_{r=0}^{2^q-1} \mathbb{B}(2^q-1, r, em(\max\{d-l-q-1, 0\})) \\ &\cdot \sum_{a=0}^{r+k-2^q} \mathbb{B}(r+k-2^q, a, 1/(2^q-r)) \\ &\cdot \Upsilon_2((\delta_1, \delta_2), \max\{d-l-q-1, 0\}, a+1) \end{aligned} \quad (8)$$

if $\delta_2 < d-l-q$.

Proof: Eq. 6 holds since at least one node within bit distance $d-l-q-1$ of x is chosen. For the remaining two cases, we summarise over the number r of prefixes without any matching node. These are approximately binomially distributed with parameters $2^q - 1$ for the number of other prefixes, and the probability that there is no node with the respective common prefix (strictly speaking the probability that common prefixes are not taken by a node are not independent of, hence the approximation). Given r, the number of additional contacts a within distance at least $d-l-q-1$ is binomially distributed with parameters $k-2^q+r$, the number of potential additional contacts, and $1/(2^q-r)$, the probability that exactly the prefix of x is also a prefix of the identifier of the contact. Eq. 7 considers the case $a = 1$: If only one link leads to a node with the same prefix as x, it is at distance δ_1 with probability $\Upsilon_1((\delta_1), \max\{d-l-q-1, 0\}, a+1)$. The second node is chosen as the closest of the $2^q - 1$ remaining prefixes. Note that there are $2^{q-i} - 1$ prefixes that agree with x's prefix in the first i bits. If the closest prefix of an existing node identifier is at distance δ_2, we have $i = \eta + 1$, and all closer $2^{q-\eta} - 1$ prefixes are chosen from the r prefixes without any node identifier, but the closest $2^{q-\eta+1} - 1$ are not chosen from those r. The probability that $m \in \{2^{q-\mu} - 1, 2^{q-\mu+1}\}$ prefixes are chosen from a set of r given that there are $2^q - 1$ prefixes to choose from is given by $\Xi(r, 2^q - 1, m)$, so that Eq. 7 follows. If a, the number of additional contacts within distance at least $d-l-q-1$, is at least 1, there are at least two nodes within distance $d-l-q-1$ of x, and the probability distribution of their distance is given by $\Upsilon_2((\delta_1, \delta_2), \max\{d-l-q-1, 0\}, a+1)$ as derived in [13]. This accounts for Eq. 8 and completes the proof. ■

VI. Evaluation

In this Section, we aim to assess the impact of our modified neighbour selection scheme on the lookup performance in Kademlia-type systems. In particular, we answer the following three questions: (*i*) How much improvement over the default lookup performance can be gained if all system nodes implement the new scheme?, and do simulations validate our model? (Sec. VI-A), (*ii*) What is the impact of churn on the gained improvement? (Sec. VI-B), and (*iii*) How much improvement can be gained by a partial deployment for the new scheme in a real Kademlia-type system? (Sec. VI-C).

The *lookup's hop count* here refers to the number of edges on the shortest path traversed during the lookup process. In Kademlia, each routing hop (i.e. step) represents a transition from a set of queried contacts to either another set of queried contacts or routing termination [13].

A. Lookup Performance of a Full Deployment Without Churn: Model vs. Simulations

We discuss here the performance results of a full deployment for our approach (i.e. all system nodes incorporate the proposed neighbour selection scheme) as predicted by our model and validate them against simulations. We focus on three exemplary Kademlia-type systems: MDHT, iMDHT, and KAD, as they are described in Sec. II.

*1) **Simulation environment and setup**:* Note that none of the well-known P2P simulators (e.g. PeerSim [27], PeerfactSim.KOM [28], or OverSim [29]) has exact implementations for the aforementioned three systems. However, the modular design of the widely-used simulator, OverSim, allows to easily add new P2P overlays. We hence chose to develop MDHT, iMDHT, and KAD and their respective modified versions as new P2P overlays in OverSim. We use the source code of eMule as a basis for our implementation. Please note that eMule implements a *loose parallel lookup* whereas our theoretical model assumes a *strict parallel lookup* (Sec. V-B: Assumption 4). However, since our current model assumes no churn, the two lookup techniques shall perform similarly [8].

We performed simulations with three system sizes: 10,000, 15,000, and 20,000 nodes. At the beginning of simulations, nodes are added to the system until the target size is reached. After the system has stabilized (i.e. the nodes have populated their routing table buckets with contacts), the statistics of interests, i.e. the hop counts, are obtained. All simulation results are averaged over ten runs.

*2) **Results**:* Fig. 4 shows the resulting CDFs of hop count distributions for the three standard systems as well as for the respective modified ones, each with 10,000 nodes, both from the model predictions and from simulations. Table I shows the hop count values predicted by the model, and for simulations, the respective sample average values, the 95% confidence intervals (using the Student's t-distribution), and the median values, in addition to the hop count gain achieved by the modified scheme. The hop count gain values of simulations represent: (*i*) the difference between $standard\_hop\_count - CI$ and $modified\_hop\_count + CI$, and (*ii*) the minimum and maximum hop count gain values. These results represent systems without churn (as assumed by the current model).

All in all, the results show the following: (*i*) The three modified systems achieve improved hop counts, i.e. they outperform the respective standards systems, which confirms the utility of the proposed scheme. (*ii*) The model predictions and simulation results are very close to each others, which indicates that both the model derivations and implementation of simulations are correct. These conclusions apply for the other experimented system sizes, that we exclude here due to space constraints. As for the impact of network size, the average hop count increased in the larger sizes, which is to be expected. However, the size had no large impact on the improvement gained by the new scheme.

It can be seen also that the highest performance gain is achieved by iMDHT, whereas the lowest is achieved by KAD. More precisely, in this example with 10,000 nodes, iMDHT improves the hop count by a bit higher than 7%, whereas KAD improves only about 1.5%, and MDHT is in between by about 4.5%. We attribute this disparity to the different routing table structures in the three systems: On the one hand, KAD implements some form of diversity by default, as described in Sec. IV-A, which limits the impact of the additional diversity enabled by the new scheme. On the other hand, MDHT and iMDHT do not have such feature in their default designs, and therefore they are expected to benefit from the new scheme more than KAD. In addition, the larger bucket sizes at the top four routing table levels (i.e. the mostly used ones) in iMDHT can contain more diverse contacts, and thus achieve a higher performance gain, than MDHT.

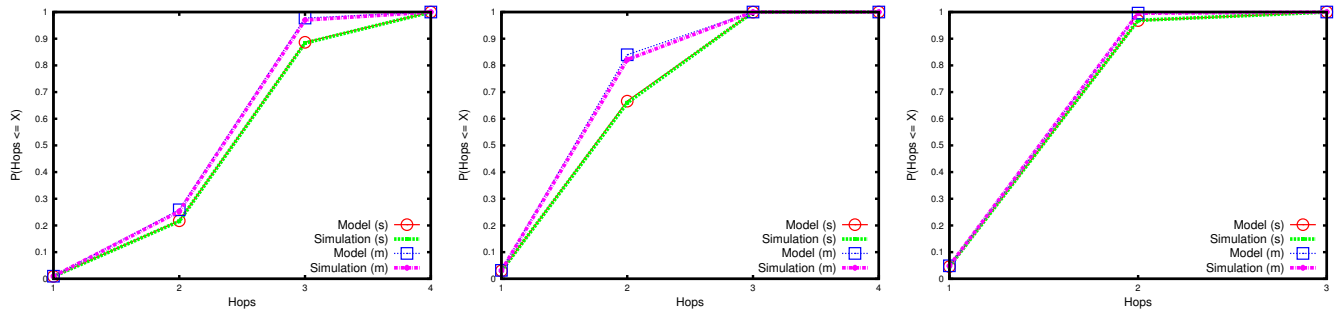
B. Impact of Churn

We aim here to evaluate the impact of churn on the lookup performance of systems incorporating the new neighbour selection scheme. As mentioned in Sec. V, the current version of our model supports only static scenarios (i.e. without churn). We hence perform the evaluations only by simulations. More precisely, the simulations apply the churn model proposed in [30], as implemented in OverSim. We simulated with two different average session lengths: 20,000 seconds and 10,000 seconds.

Table II summarizes the results. Comparing the sample average with the 95% confidence intervals (using the Student's t-distribution) and median hop count values of the two simulation settings both to each others and to the static scenario (Table I), the results show the following: (*i*) The results without churn are always better (i.e. achieve shorter hop counts), and those with the lower churn rate (i.e. with longer average session lengths of 20,000 seconds) outperform those with higher churn rate (10,000 seconds). These results are to be expected (see [31], [32] for explanation). (*ii*) More interestingly, it can be seen that for MDHT and KAD, the hop count gain increases with churn (only the gain from the new scheme, not the hop count itself). This can be explained as follows: Since the frequency of the routing table maintenance increases with higher churn rates, there is a chance in that case to discover more contacts, and hence to increase the buckets' diversity degrees. KAD achieves the highest improvement because its routing table size is not a power of 2. As for iMDHT, the observation above does not hold, i.e. the churn has almost no impact on the results. The reason for this can probably be found in the extremely large bucket size on the high levels (i.e. mostly used ones): It takes very long to fill these buckets, and hence it is hard to keep the contacts alive under churn, so that the high stale entry rate impairs further improvement.

C. Lookup Performance of a Partial Deployment: Measurements of Modified KAD Clients

We provide here an additional evaluation for the impact of the new scheme on the lookup performance by measuring it in a real Kademlia-type system. However, a large-scale deployment of the new scheme in a lifelike Kademlia-type system does not exist. Therefore, measuring the performance gain of a full deployment is infeasible. Instead, we measured the lookup performance on modified nodes (implementing the new scheme), during their participation in a standard Kademlia-type system. In this setting, the modified nodes are expected to benefit from the modified scheme, but the



(a) MDHT (b) iMDHT (c) KAD

Fig. 4: CDFs of hop count dist. (simulations vs. model expectations) for three exemplary Kademlia-type systems of size 10,000, without churn: (a) MDHT, (b) iMDHT, (c) KAD. (**s:** standard system; **m:** modified system)

TABLE I: Sample average hop count with 0.95 CI (t-value) and median values for simulations vs. model expectations for three exemplary Kademlia-type systems of size 10,000, without churn: Standard systems, modified systems, and the achieved hop count gain (+ : "*modified* + *CI*" to "*standard* - *CI*" and [min , max]).

	MDHT			**iMDHT**			**KAD**		
	Simulations		**Model**	**Simulations**		**Model**	**Simulations**		**Model**
	Sample Avg. ± CI	**Median**		**Sample Avg. ± CI**	**Median**		**Sample Avg. ± CI**	**Median**	
Standard	2.89185 ± 0.00019	2.89180	2.88697	2.31113 ± 0.00032	2.31121	2.30470	1.98609 ± 0.00028	1.98607	1.98609
Modified	2.76774 ± 0.00074	2.76755	2.75416	2.14716 ± 0.00106	2.14699	2.12828	1.95610 ± 0.00035	1.95621	1.95535
+ (%)	4.32376	-	4.60025	7.15366	-	7.65508	1.54158	-	1.54760
	[4.22405 , 4.33964]			[7.00106 , 7.19749]			[1.45703 , 1.58457]		

TABLE II: Median and sample average hop count with 0.95 CI (t-value) for simulations of size 10,000, with churn (two different average session lengths): Standard systems, modified systems, and the achieved hop count gain (+ : "*modified* + *CI*" to "*standard* - *CI*" and [min , max]).

	MDHT		**iMDHT**		**KAD**	
Avg. session time	**20,000 sec.**	**10,000 sec.**	**20,000 sec.**	**10,000 sec.**	**20,000 sec.**	**10,000 sec.**
Standard (Median)	3.21465	3.32382	2.57720	2.69044	2.21886	2.31689
Modified (Median)	3.01307	3.11266	2.38905	2.49321	2.09163	2.17313
Standard (Sample Avg. ± CI)	3.21380 ± 0.00109	3.32362 ± 0.00092	2.57716 ± 0.00029	2.69084 ± 0.00183	2.21842 ± 0.00103	2.31668 ± 0.00121
Modified (Sample Avg. ± CI)	3.01311 ± 0.00025	3.11264 ± 0.00203	2.38908 ± 0.00161	2.49309 ± 0.00128	2.09151 ± 0.00580	2.17308 ± 0.00182
+ (%)	6.20486	6.26100	7.22454	7.23849	5.41489	6.07047
	[6.16229 , 6.30019]	[6.19498 , 6.49519]	[7.14763 , 7.44761]	[7.22199 , 7.49043]	[4.77028 , 6.19054]	[6.00419 , 6.34748]

benefit is expected to be less pronounced than for a complete implementation.

*1) **Measurement environment and setup**:* We performed our measurements in KAD, using two clients: the first uses the standard KAD code as implemented in the eMule software, while the second implements the new neighbour selection scheme.

During each measurement run, each client issued 500 lookup requests, one request every three seconds. The target identifiers are selected such that they are uniformly distributed over the identifier space. In particular, we used 500 keywords from the list of Steiner et al. [9]. During measurements, the clients recorded the following information: (*i*) the number of hops traversed by each lookup request, and (*ii*) the diversity degrees of routing table buckets at level 4.

*2) **Results**:* We performed 40 measurement runs at several times of the day. Fig. 5 shows the CDFs of hop count distributions both for the standard client and for the modified client. The median values for the standard client and the modified client were: 3.39669 hops and 3.22624 hops, respectively. The corresponding sample average with the 95% confidence intervals (using the Student's t-distribution) were: 3.39862 hops ($\pm$0.0.01583) and 3.23172 hops ($\pm$0.01310). In this example, the achieved hop count reduction (computed as the difference between $standard\_hop\_count - CI$ and $modified\_hop\_count + CI$) is 4.07861%, with a minimum of 3.36571% and a maximum of 7.68003%.

Unfortunately, the measurement results cannot be compared to model predictions because the model does not support churn, nor to simulations because the simulator cannot scale to relatively very large sizes like the size of the real KAD system[3]. Since only our modified client implements the new neighbour selection scheme, while the rest are standard clients, one can expect the gain of the new scheme to be lower than what is achieved. However, when looking on the diversity degrees, the modified client could improve the diversity degrees of its buckets[4]: The modified client achieved a diversity degree of 7.37 (which is very close from the maximal value), on average, compared to 4.68 for the standard client (see also the results of measured diversity degrees of a large sample of standard clients in Sec. IV-B: Fig. 3). This increase in the diversity degrees can explain the achieved improvement, and

[3]Recently, [3] counted more than 300,000 concurrent online nodes in KAD.

[4]For the reasons that we mentioned in Sec. IV-B, we restricted our analysis here also only to the buckets located at the fourth routing table level.

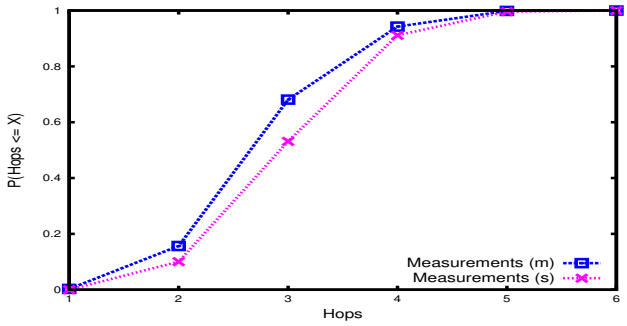


Fig. 5: CDFs of hop count distributions from measurements of standard and modified KAD clients. (**s:** standard client; **m:** modified client)

confirms the utility of increasing the diversity.

We attribute the obtained increase in the diversity degrees to the expected high frequency of executing the routing table maintenance processes in the real system, due to the high ratio of stale routing table entries. For instance, in [33] we reported 88% as the ratio of stale routing table entries, computed on almost complete and instantaneous graph snapshots of KAD. As explained in Sec. VI-B, the higher the churn rate in KAD (i.e. more frequent maintenance for routing tables) the higher the chance to discover new neighbours that increase the buckets' diversity degrees.

VII. Conclusion

We proposed, modelled, and evaluated a new neighbour selection scheme for Kademlia-type systems. It aims to improve the lookup performance, almost without extra cost, by only attempting to maximize the identifiers' diversity within each routing table bucket.

Our model predictions, in very close agreement with simulations, as well as measurements of modified KAD clients, confirm the positive impact of our scheme on the lookup performance, in form of reduction in the average hop count. The simulation results also show that the systems with small bucket sizes (namely: MDHT and KAD) can benefit more from our approach in the dynamic scenarios (i.e. with churn). We have attributed this to the resulting high frequency of routing table maintenance processes which are utilized by our scheme to improve the performance. Nevertheless, the dynamic scenario cannot be captured by the current version of the model, and therefore those results still require further study. Consequently, our plans for future work include modelling the impact of churn, network size, and the partial deployment scenario (where only part of the nodes implements our scheme). In addition, we have started studying the idea of extending our approach by integrating it to notable prior improvements like [18] and [34].

Acknowledgements

This work was supported by the German Academic Exchange Service (DAAD: A/09/97920) and the German Research Foundation (DFG: 231189459). The authors thank the anonymous reviewers for their valuable comments, and thank Sven Frese for his help with the measurements.

References

[1] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *IPTPS*, 2002.
[2] L. Wang and J. Kangasharju, "Measuring large-scale distributed systems: case of BitTorrent Mainline DHT," in *IEEE P2P*, 2013.
[3] H. Salah and T. Strufe, "Capturing connectivity graphs of a large-scale p2p overlay network," in *ICDCS Workshops*, 2013.
[4] R. Jimenez, "Kademlia on the open internet: How to achieve sub-second lookups in a multimillion-node DHT overlay," *Lic. Thesis, KTH*, 2011.
[5] G. Starnberger, C. Kruegel, and E. Kirda, "Overbot: a botnet protocol based on Kademlia," in *SecureComm*, 2008.
[6] T. Holz *et al.*, "Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm," *LEET*, 2008.
[7] J. Li *et al.*, "Comparing the performance of distributed hash tables under churn," in *P2P Systems*, 2005.
[8] D. Stutzbach and R. Rejaie, "Improving Lookup Performance Over a Widely-Deployed DHT," in *INFOCOM*, 2006.
[9] M. Steiner, D. Carra, and E. W. Biersack, "Faster content access in KAD," in *IEEE P2P*, 2008.
[10] J. Falkner *et al.*, "Profiling a million user DHT," in *IMC*, 2007.
[11] S. Crosby and D. Wallach, "An analysis of bittorrents two kademlia-based DHTs," TR07-04, Rice University, Tech. Rep., 2007.
[12] R. Jimenez, F. Osmani, and B. Knutsson, "Sub-second lookups on a large-scale Kademlia-based overlay," in *IEEE P2P*, 2011.
[13] S. Roos, H. Salah, and T. Strufe, "Comprehending Kademlia Routing-A Theoretical Framework for the Hop Count Distribution," *Technical Report, arXiv preprint arXiv:1307.7000*, 2013.
[14] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," *Technical Report UW-CSE-01-06-02, University of Washington*, 2001.
[15] P. Wang *et al.*, "Attacking the kad network," in *SecureComm*, 2008.
[16] G. Einziger, R. Friedman, and E. Kibbar, "Kaleidoscope: Adding Colors to Kademlia," in *IEEE P2P*, 2013.
[17] C. Wang *et al.*, "DiCAS: An efficient distributed caching mechanism for P2P systems," *TPDS*, 2006.
[18] S. Kaune *et al.*, "Embracing the peer next door: Proximity in kademlia," in *IEEE P2P*, 2008.
[19] M. Castroand *et al.*, "Exploiting network proximity in distributed hash tables," in *FuDiCo*, 2002.
[20] B. Heep, "R/Kademlia: Recursive and topology-aware overlay routing," in *IEEE ATNAC*, 2010.
[21] I. Stoica *et al.*, "Chord: A scalable peer-to-peer lookup service for internet applications," in *SIGCOMM*, 2001.
[22] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Middleware*, 2001.
[23] D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: A scalable and dynamic emulation of the butterfly," in *PODC*, 2002.
[24] A. Spognardi and R. D. Pietro, "A formal framework for the performance analysis of P2P networks protocols," in *IPDPS*, 2006.
[25] I. Rai *et al.*, "Performance Modelling of Peer-to-Peer Routing," in *IPDPS*, 2007.
[26] H. Salah, S. Roos, and T. Strufe, "A Lightweight Approach for Improving the Lookup Performance in Kademlia-type Systems," *Technical Report, TU Darmstadt: TUD-CS-2014-0888*, 2014.
[27] A. Montresor and M. Jelasity, "PeerSim: A Scalable P2P Simulator," in *IEEE P2P*, 2009.
[28] D. Stingl *et al.*, "Peerfactsim.kom: A simulation framework for peer-to-peer systems," in *HPCS*, 2011.
[29] I. Baumgart, B. Heep, and S. Krause, "OverSim: A flexible overlay network simulation framework," in *IEEE Global Internet Symposium*, 2007.
[30] Z. Yao *et al.*, "Modeling heterogeneous user churn and local resilience of unstructured p2p networks," in *IEEE ICNP*, 2006.
[31] D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks," in *IMC*, 2006.
[32] I. Baumgart and B. Heep, "Fast but economical: A simulative comparison of structured peer-to-peer systems," in *IEEE NGI*, 2012.
[33] H. Salah, S. Roos, and T. Strufe, "Characterizing Graph-Theoretic Properties of a Large-Scale Overlay: Measurements vs. Simulations," in *IEEE ISCC*, 2014.
[34] J. Li *et al.*, "Bandwidth-efficient management of DHT routing tables," in *Usenex NSDI*, 2005.